

---

**PyFFI**  
*Release 2.2.3*

**Amorilia**

**Oct 06, 2019**



# CONTENTS

<b>1</b>	<b>PyFFI</b>	<b>3</b>
1.1	Download . . . . .	3
1.2	Developing . . . . .	3
1.3	Testing . . . . .	4
1.4	Documentation . . . . .	4
1.5	Examples . . . . .	4
1.6	Questions? Suggestions? . . . . .	4
1.7	Documentation . . . . .	4
1.8	Indices and tables . . . . .	304
	<b>Python Module Index</b>	<b>305</b>
	<b>Index</b>	<b>307</b>



**Release** 2.2.3

**Date** Oct 06, 2019



The Python File Format Interface, briefly PyFFI, is an open source Python library for processing block structured binary files:

- **Simple:** Reading, writing, and manipulating complex binary files in a Python environment is easy! Currently, PyFFI supports the NetImmerse/Gamebryo NIF and KFM formats, CryTek's CGF format, the FaceGen EGM format, the DDS format, and the TGA format.
- **Batteries included:** Many tools for files used by 3D games, such as optimizers, stripifier, tangent space calculator, 2d/3d hull algorithms, inertia calculator, as well as a general purpose file editor QSkope (using PyQt4), are included.
- **Modular:** Its highly modular design makes it easy to add support for new formats, and also to extend existing functionality.

## 1.1 Download

Get PyFFI from [Github](#), or install it with:

```
easy_install -U PyFFI
```

or:

```
pip3 install PyFFI
```

## 1.2 Developing

To get the latest (but possibly unstable) code, clone PyFFI from its [Git repository](#):

```
git clone --recursive git://github.com/niftools/pyffi.git
virtualenv -p python3 venv
source venv/bin/activate
pip install -r requirements-dev.txt
```

Be sure to use the `--recursive` flag to ensure that you also get all of the submodules.

If you wish to code on PyFFI and send your contributions back upstream, get a [github account](#) and [fork PyFFI](#).

## 1.3 Testing

We love tests, they help guarantee that things keep working they way they should. You can run them yourself with the following:

```
source venv/bin/activate
nosetest -v test
```

or:

```
source venv/bin/activate
py.test -v tests
```

## 1.4 Documentation

All our documentation is written in ReST and can be generated into HTML, LaTeX, PDF and more thanks to Sphinx. You can generate it yourself:

```
source venv/bin/activate
cd docs
make html -a
```

## 1.5 Examples

- The Blender NIF Plugin
- QSkope PyFFI's general purpose file editor.
- The niftoaster (PyFFI's "swiss army knife") can for instance [optimize NIF files](#), and much more.

## 1.6 Questions? Suggestions?

- Open an issue at the [issue tracker](#).

## 1.7 Documentation

### 1.7.1 Introduction

#### Example and Problem Description

Consider an application which processes images stored in for instance the Targa format:

```
>>> # read the file
>>> stream = open("tests/tga/test.tga", "rb")
>>> data = bytearray(stream.read()) # read bytes
>>> stream.close()
>>> # do something with the data...
```

(continues on next page)

(continued from previous page)

```

>>> data[8] = 20 # change x origin
>>> data[10] = 20 # change y origin
>>> # etc... until we are finished processing the data
>>> # write the file
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> dummy = stream.write(data) # py3k returns number of bytes written
>>> stream.close()

```

This methodology will work for any file format, but it is usually not very convenient. For complex file formats, the *do something with the data* part of the program would be necessarily quite complicated for the programmer. For this reason, it is convenient to convert the data (a sequence of bytes) into an organized collection of Python objects (a class suits this purpose perfectly) that clearly reveal what is stored in the data. Such organized collection is called an *interface*:

```

>>> import struct
>>> from tempfile import TemporaryFile
>>> class TgaFile:
...     """A simple class for reading and writing Targa files."""
...     def read(self, filename):
...         """Read tga file from stream."""
...         stream = open(filename, "rb")
...         self.image_id_length, self.colormap_type, self.image_type, \
...         self.colormap_index, self.colormap_length, self.colormap_size, \
...         self.x_origin, self.y_origin, self.width, self.height, \
...         self.pixel_size, self.flags = struct.unpack("<BBBHHBHHHHBB",
...                                                     stream.read(18))
...         self.image_id = stream.read(self.image_id_length)
...         if self.colormap_type:
...             self.colormap = [
...                 stream.read(self.colormap_size >> 3)
...                 for i in range(self.colormap_length)]
...         else:
...             self.colormap = []
...         self.image = [[stream.read(self.pixel_size >> 3)
...                         for i in range(self.width)]
...                        for j in range(self.height)]
...         stream.close()
...     def write(self, filename=None):
...         """Read tga file from stream."""
...         if filename:
...             stream = open(filename, "wb")
...         else:
...             stream = TemporaryFile()
...         stream.write(struct.pack("<BBBHHBHHHHBB",
...                                 self.image_id_length, self.colormap_type, self.image_type,
...                                 self.colormap_index, self.colormap_length,
...                                 self.colormap_size,
...                                 self.x_origin, self.y_origin, self.width, self.height,
...                                 self.pixel_size, self.flags))
...         stream.write(self.image_id)
...         for entry in self.colormap:
...             stream.write(entry)
...         for line in self.image:
...             for pixel in line:
...                 stream.write(pixel)

```

(continues on next page)

```
...         stream.close()
>>> data = TgaFile()
>>> # read the file
>>> data.read("tests/tga/test.tga")
>>> # do something with the data...
>>> data.x_origin = 20
>>> data.y_origin = 20
>>> # etc... until we are finished processing the data
>>> # write the file
>>> data.write()
```

The reading and writing part of the code has become a lot more complicated, but the benefit is immediately clear: instead of working with a sequence of bytes, we can directly work with the members of our `TgaFile` class, and our code no longer depends on how exactly image data is organized in a Targa file. In other words, our code can now use the semantics of the `TgaFile` class, and is consequently much easier to understand and to maintain.

In practice, however, when taking the above approach as given, the additional code that enables this semantic translation is often difficult to maintain, for the following reasons:

- **Duplication:** Any change in the reader part must be reflected in the writer part, and vice versa. Moreover, the same data types tend to occur again and again, leading to nearly identical code for each read/write operation. A partial solution to this problem would be to create an additional class for each data type, each with its read and write method.
- **No validation:** What if `test/tga/test.tga` is not a Targa file at all, or is corrupted? What if `image_id` changes length but `image_id_length` is not updated accordingly? Can we catch such bugs and prevent data to become corrupted?
- **Boring:** Writing *interface* code gets boring very quickly.

## What is PyFFI?

PyFFI aims to solve all of the above problems:

- The *interface* classes are *generated at runtime*, from an easy to maintain description of the file format. The generated classes provides semantic access to *all* information in the files.
- Validation is automatically enforced by the generated classes, except in a few rare cases when automatic validation might cause substantial overhead. These cases are well documented and simply require an explicit call to the validation method.
- The generated classes can easily be extended with additional class methods, for instance to provide common calculations (for example: converting a single pixel into greyscale).
- Very high level functions can be implemented as *spells* (for example: convert a height map into a normal map).

## 1.7.2 Installation

### Requirements

To run PyFFI's graphical file editor QSkope, you need [PyQt4](#).

### Using the Windows installer

Simply download and run the Windows installer provided in our [Releases](#).

## Manual installation

If you install PyFFI manually, and you already have an older version of PyFFI installed, then you **must** uninstall (see *Uninstall*) the old version before installing the new one.

## Installing via setuptools

If you have `setuptools` installed, simply run:

```
easy_install -U PyFFI
```

at the command prompt.

## Installing from source package

First, get the [source package](#). Untar or unzip the source package via either:

```
tar xfvz PyFFI-x.x.x.tar.gz
```

or:

```
unzip PyFFI-x.x.x.zip
```

Change to the PyFFI directory and run the setup script:

```
cd PyFFI-x.x.x
python setup.py install
```

## Uninstall

You can uninstall PyFFI manually simply by deleting the `pyffi` folder from your Python's `site-packages` folder, which is typically at:

```
C:\Python25\Lib\site-packages\pyffi
```

or:

```
/usr/lib/python2.5/site-packages/pyffi
```

## 1.7.3 pyffi — Interfacing block structured files

### pyffi.formats — File format interfaces

When experimenting with any of the supported file formats, you can specify an alternate location where you store your modified format description by means of an environment variable. For instance, to tell the library to use your version of `cgf.xml`, set the `CGFXMLPATH` environment variable to the directory where `cgf.xml` can be found. The environment variables `NIFXMLPATH`, `KFMXMLPATH`, `DDXMLPATH`, and `TGXMLPATH` work similarly.

## Supported formats

### pyffi.formats.bsa — Bethesda Archive (.bsa)

**Warning:** This module is still a work in progress, and is not yet ready for production use.

A .bsa file is an archive format used by Bethesda (Morrowind, Oblivion, Fallout 3).

## Implementation

```
class pyffi.formats.bsa.BsaFormat
    Bases: pyffi.object_models.xml.FileFormat

    This class implements the BSA format.

class BZString (**kwargs)
    Bases: pyffi.object_models.common.SizedString

    get_size (data=None)
        Return number of bytes this type occupies in a file.
        Returns Number of bytes.

    read (stream, data=None)
        Read string from stream.
        Parameters stream (file) – The stream to read from.

    write (stream, data=None)
        Write string to stream.
        Parameters stream (file) – The stream to write to.

Data
    alias of Header

class FileVersion (**kwargs)
    Bases: pyffi.object_models.common.UInt

    Basic type which implements the header of a BSA file.

    get_size (data=None)
        Return number of bytes the header string occupies in a file.
        Returns Number of bytes.

    read (stream, data)
        Read header string from stream and check it.
        Parameters stream (file) – The stream to read from.

    write (stream, data)
        Write the header string to stream.
        Parameters stream (file) – The stream to write to.

class Hash (**kwargs)
    Bases: pyffi.object_models.common.UInt64

    get_detail_display ()
        Return an object that can be used to display the instance.
```

**class Header** (*template=None, argument=None, parent=None*)  
 Bases: `pyffi.formats.bsa._Header`, `pyffi.object_models.Data`

A class to contain the actual bsa data.

**inspect** (*stream*)  
 Quickly checks if stream contains BSA data, and reads the header.  
**Parameters** **stream** (*file*) – The stream to inspect.

**inspect\_quick** (*stream*)  
 Quickly checks if stream contains BSA data, and gets the version, by looking at the first 8 bytes.  
**Parameters** **stream** (*file*) – The stream to inspect.

**read** (*stream*)  
 Read a bsa file.  
**Parameters** **stream** (*file*) – The stream from which to read.

**write** (*stream*)  
 Write a bsa file.  
**Parameters** **stream** (*file*) – The stream to which to write.

**UInt32**  
 alias of `pyffi.object_models.common.UInt`

**class ZString** (*\*\*kwargs*)  
 Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

String of variable length (null terminated).

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> s = ZString()
>>> if f.write('abcdefghijklmnopqrst\x00'.encode("ascii")): pass # b'abc...'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f)
>>> str(s)
'abcdefghijklmnopqrst'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There!')
>>> s.write(f)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = ZString()
>>> m.read(f)
>>> str(m)
'Hi There!'
```

**get\_hash** (*data=None*)  
 Return a hash value for this string.  
**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)  
 Return number of bytes this type occupies in a file.  
**Returns** Number of bytes.

**get\_value** ()  
 Return the string.  
**Returns** The stored string.  
**Return type** `C{bytes}`

**read** (*stream*, *data=None*)

Read string from stream.

**Parameters** *stream* (*file*) – The stream to read from.

**set\_value** (*value*)

Set string to C{value}.

**Parameters** *value* (*str* (will be encoded as default) or C{bytes}) – The value to assign.

**write** (*stream*, *data=None*)

Write string to stream.

**Parameters** *stream* (*file*) – The stream to write to.

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** *version\_str* (*str*) – The version string.

**Returns** A version integer.

```
>>> BsaFormat.version_number('103')
103
>>> BsaFormat.version_number('XXX')
-1
```

## Regression tests

### Read a BSA file

```
>>> # check and read bsa file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'bsa')
>>> stream = open(os.path.join(format_root, 'test.bsa'), 'rb')
>>> data = BsaFormat.Data()
>>> data.inspect_quick(stream)
>>> data.version
103
>>> data.inspect(stream)
>>> data.folders_offset
36
>>> hex(data.archive_flags.get_attributes_values(data))
'0x703'
>>> data.num_folders
1
>>> data.num_files
7
>>> #data.read(stream)
>>> # TODO check something else...
```

## Parse all BSA files in a directory tree

```
>>> for stream, data in BsaFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/bsa/test.bsa
```

## Create an BSA file from scratch and write to file

```
>>> data = BsaFormat.Data()
>>> # TODO store something...
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> #data.write(stream)
```

## pyffi.formats.cgf — Crytek (.cgf and .cga)

### Implementation

**class** pyffi.formats.cgf.CgffFormat

Bases: pyffi.object\_models.xml.FileFormat

Stores all information about the cgf file format.

**class** AbstractMtlChunk (template=None, argument=None, parent=None)

Bases: pyffi.formats.cgf.Chunk

Common parent for MtlChunk and MtlNameChunk.

**class** AbstractObjectChunk (template=None, argument=None, parent=None)

Bases: pyffi.formats.cgf.Chunk

Common parent for HelperChunk and MeshChunk.

**class** BoneLink (template=None, argument=None, parent=None)

Bases: pyffi.object\_models.xml.struct\_.StructBase

A bone link.

**property** blending

Vertex weight.

**property** bone

The bone chunk.

**property** offset

The bone offset?

**exception CgfError**Bases: `Exception`

Exception for CGF specific errors.

**class Chunk** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.cgf._Chunk, object`**apply\_scale** (*scale*)

Apply scale factor on data.

**tree** (*block\_type=None, follow\_all=True*)A generator for parsing all blocks in the tree (starting from and including `C{self}`).**Parameters**

- **block\_type** – If not `None`, yield only blocks of the type `C{block_type}`.
- **follow\_all** – If `C{block_type}` is not `None`, then if this is `True` the function will parse the whole tree. Otherwise, the function will not follow branches that start by a non-`C{block_type}` block.

**class ChunkHeader** (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`

A CGF chunk header.

**property id**

The chunk identifier.

**property offset**

Position of the chunk in the CGF file.

**property type**

Type of chunk referred to.

**property version**

Version of the chunk referred to.

**class ChunkTable** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.cgf._ChunkTable, object`**get\_chunk\_types** ()

Iterate all chunk types (in the form of Python classes) referenced in this table.

**class ChunkType** (*\*\*kwargs*)Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 32-bit integer, describing the chunk type.

**class ChunkVersion** (*\*\*kwargs*)Bases: `pyffi.object_models.common.UInt`

The version of a particular chunk, or the version of the chunk table.

**class Data** (*filetype=4294901760, game='Far Cry'*)Bases: `pyffi.object_models.Data`

A class to contain the actual cgf data.

Note that `L{versions}` and `L{chunk_table}` are not automatically kept in sync with the `L{chunks}`, but they are resynchronized when calling `L{write}`.

**Variables**

- **game** – The cgf game.
- **header** – The cgf header.

- *chunks* – List of chunks (the actual data).
- *versions* – List of chunk versions.

**get\_detail\_child\_names** (*edge\_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

**Returns** Generator for detail tree child names.

**Return type** generator yielding `strs`

**get\_detail\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

**Parameters** *edge\_filter* (`EdgeFilter` or type `(None)`) – The edge type to include.

**Returns** Generator for detail tree child nodes.

**Return type** generator yielding `DetailNodes`

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)

Returns chunks without parent.

**inspect** (*stream*)

Quickly checks whether the stream appears to contain cgf data, and read the cgf header and chunk table. Resets stream to original position.

Call this function if you only need to inspect the header and chunk table.

**Parameters** *stream* (*file*) – The file to inspect.

**inspect\_version\_only** (*stream*)

This function checks the version only, and is faster than the usual inspect function (which reads the full chunk table). Sets the `L{header}` and `L{game}` instance variables if the stream contains a valid cgf file.

Call this function if you simply wish to check that a file is a cgf file without having to parse even the header.

**Raises** `ValueError` – If the stream does not contain a cgf file.

**Parameters** *stream* (*file*) – The stream from which to read.

**read** (*stream*)

Read a cgf file. Does not reset stream position.

**Parameters** *stream* (*file*) – The stream from which to read.

**replace\_global\_node** (*oldbranch, newbranch, edge\_filter=(True, True)*)

Replace a particular branch in the graph.

**update\_versions** ()

Update `L{versions}` for the given chunks and game.

**write** (*stream*)

Write a cgf file. The `L{header}` and `L{chunk_table}` are recalculated from `L{chunks}`. Returns number of padding bytes written (this is for debugging purposes only).

**Parameters** *stream* (*file*) – The stream to which to write.

**Returns** Number of padding bytes written.

**class DataStreamChunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.cgf._DataStreamChunk, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**class ExportFlagsChunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.cgf.Chunk`

Export information.

**class FRGB** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

R32G32B32 (float).

**class Face** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A mesh face.

**property material**

Material index.

**property sm\_group**

Smoothing group.

**property v\_0**

First vertex index.

**property v\_1**

Second vertex index.

**property v\_2**

Third vertex index.

**class FileOffset** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.Int`

Points to a position in a file.

**class FileSignature** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

The CryTek file signature with which every cgf file starts.

**get\_hash** (*data=None*)

Return a hash value for the signature.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes that the signature occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Get signature.

**Returns** The signature.

**read** (*stream, data*)

Read signature from stream.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Not implemented.

**write** (*stream, data*)

Write signature to stream.

**Parameters** **stream** (*file*) – The stream to read from.

---

```

class FileType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing the file type.

class GeomNameListChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Obsolete, not decoded.

class Header (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    The CGF header.

property offset
    Position of the chunk table in the CGF file.

property signature
    The CGF file signature.

property type
    The CGF file type (geometry or animation).

property version
    The version of the chunk table.

class IRGB (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    R8G8B8.

class IRGBA (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    R8G8B8A8.

class InitialPosMatrix (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A bone initial position matrix.

class MRMChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Obsolete, not decoded.

class Matrix33 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf._Matrix33, object

as_list ()
    Return matrix as 3x3 list.

as_tuple ()
    Return matrix as 3x3 tuple.

get_copy ()
    Return a copy of the matrix.

get_determinant ()
    Return determinant.

get_inverse ()
    Get inverse (assuming is_scale_rotation is true!).

```

**get\_scale()**  
Gets the scale (assuming `is_scale_rotation` is true!).

**get\_scale\_quat()**  
Decompose matrix into scale and quaternion.

**get\_scale\_rotation()**  
Decompose the matrix into scale and rotation, where scale is a float and rotation is a C{Matrix33}.  
Returns a pair (scale, rotation).

**get\_transpose()**  
Get transposed of the matrix.

**is\_identity()**  
Return `True` if the matrix is close to identity.

**is\_rotation()**  
Returns `True` if the matrix is a rotation matrix (a member of SO(3)).

**is\_scale\_rotation()**  
Returns true if the matrix decomposes nicely into scale \* rotation.

**set\_identity()**  
Set to identity matrix.

**set\_scale\_rotation(scale, rotation)**  
Compose the matrix as the product of scale \* rotation.

**class Matrix44** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.cgf._Matrix44, object`

**as\_list()**  
Return matrix as 4x4 list.

**as\_tuple()**  
Return matrix as 4x4 tuple.

**get\_copy()**  
Create a copy of the matrix.

**get\_inverse(fast=True)**  
Calculates inverse (fast assumes `is_scale_rotation_translation` is `True`).

**get\_matrix\_33()**  
Returns upper left 3x3 part.

**get\_translation()**  
Returns lower left 1x3 part.

**is\_identity()**  
Return `True` if the matrix is close to identity.

**set\_identity()**  
Set to identity matrix.

**set\_matrix\_33(m)**  
Sets upper left 3x3 part.

**set\_rows(row0, row1, row2, row3)**  
Set matrix from rows.

**set\_translation(translation)**  
Returns lower left 1x3 part.

**sup\_norm()**  
Calculate supremum norm of matrix (maximum absolute value of all entries).

**class MeshChunk** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.cgf._MeshChunk`, `object`

**apply\_scale** (*scale*)  
Apply scale factor on data.

**get\_colors** ()  
Generator for all vertex colors.

**get\_material\_indices** ()  
Generator for all materials (per triangle).

**get\_normals** ()  
Generator for all normals.

**get\_num\_triangles** ()  
Get number of triangles.

**get\_triangles** ()  
Generator for all triangles.

**get\_uv\_triangles** ()  
Generator for all uv triangles.

**get\_uvs** ()  
Generator for all uv coordinates.

**get\_vertices** ()  
Generator for all vertices.

**set\_geometry** (*verticeslist=None, normalslist=None, triangleslist=None, matlist=None, uvslist=None, colorslist=None*)  
Set geometry data.

```
>>> from pyffi.formats.cgf import CgfFormat
>>> chunk = CgfFormat.MeshChunk()
>>> vertices1 = [(0,0,0), (0,1,0), (1,0,0), (1,1,0)]
>>> vertices2 = [(0,0,1), (0,1,1), (1,0,1), (1,1,1)]
>>> normals1 = [(0,0,-1), (0,0,-1), (0,0,-1), (0,0,-1)]
>>> normals2 = [(0,0,1), (0,0,1), (0,0,1), (0,0,1)]
>>> triangles1 = [(0,1,2), (2,1,3)]
>>> triangles2 = [(0,1,2), (2,1,3)]
>>> uvs1 = [(0,0), (0,1), (1,0), (1,1)]
>>> uvs2 = [(0,0), (0,1), (1,0), (1,1)]
>>> colors1 = [(0,1,2,3), (4,5,6,7), (8,9,10,11), (12,13,14,15)]
>>> colors_2 = [(50,51,52,53), (54,55,56,57), (58,59,60,61), (62,63,64,65)]
>>> chunk.set_geometry(verticeslist = [vertices1, vertices2],
...                    normalslist = [normals1, normals2],
...                    triangleslist = [triangles1, triangles2],
...                    uvslist = [uvs1, uvs2],
...                    matlist = [2,5],
...                    colorslist = [colors1, colors_2])
>>> print(chunk)
<class 'pyffi.formats.cgf.MeshChunk'> instance at ...
* has_vertex_weights : False
* has_vertex_colors : True
* in_world_space : False
* reserved_1 : 0
```

(continues on next page)

(continued from previous page)

```

* reserved_2 : 0
* flags_1 : 0
* flags_2 : 0
* num_vertices : 8
* num_indices : 12
* num_uvs : 8
* num_faces : 4
* material : None
* num_mesh_subsets : 2
* mesh_subsets : <class 'pyffi.formats.cgf.MeshSubsetsChunk'> instance at ...
↳...
* vert_anim : None
* vertices :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 0.000 0.000 0.000 ]
  * n : [ 0.000 0.000 -1.000 ]
  1: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 0.000 1.000 0.000 ]
  * n : [ 0.000 0.000 -1.000 ]
  2: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 1.000 0.000 0.000 ]
  * n : [ 0.000 0.000 -1.000 ]
  3: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 1.000 1.000 0.000 ]
  * n : [ 0.000 0.000 -1.000 ]
  4: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 0.000 0.000 1.000 ]
  * n : [ 0.000 0.000 1.000 ]
  5: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 0.000 1.000 1.000 ]
  * n : [ 0.000 0.000 1.000 ]
  6: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 1.000 0.000 1.000 ]
  * n : [ 0.000 0.000 1.000 ]
  7: <class 'pyffi.formats.cgf.Vertex'> instance at ...
  * p : [ 1.000 1.000 1.000 ]
  * n : [ 0.000 0.000 1.000 ]
* faces :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: <class 'pyffi.formats.cgf.Face'> instance at ...
  * v_0 : 0
  * v_1 : 1
  * v_2 : 2
  * material : 2
  * sm_group : 1
  1: <class 'pyffi.formats.cgf.Face'> instance at ...
  * v_0 : 2
  * v_1 : 1
  * v_2 : 3
  * material : 2
  * sm_group : 1
  2: <class 'pyffi.formats.cgf.Face'> instance at ...
  * v_0 : 4
  * v_1 : 5
  * v_2 : 6
  * material : 5

```

(continues on next page)

(continued from previous page)

```

* sm_group : 1
3: <class 'pyffi.formats.cgf.Face'> instance at ...
* v_0 : 6
* v_1 : 5
* v_2 : 7
* material : 5
* sm_group : 1
* uvs :
<class 'pyffi.object_models.xml.array.Array'> instance at ...
0: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 0.0
* v : 0.0
1: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 0.0
* v : 1.0
2: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 1.0
* v : 0.0
3: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 1.0
* v : 1.0
4: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 0.0
* v : 0.0
5: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 0.0
* v : 1.0
6: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 1.0
* v : 0.0
7: <class 'pyffi.formats.cgf.UV'> instance at ...
* u : 1.0
* v : 1.0
* uv_faces :
<class 'pyffi.object_models.xml.array.Array'> instance at ...
0: <class 'pyffi.formats.cgf.UVFace'> instance at ...
* t_0 : 0
* t_1 : 1
* t_2 : 2
1: <class 'pyffi.formats.cgf.UVFace'> instance at ...
* t_0 : 2
* t_1 : 1
* t_2 : 3
2: <class 'pyffi.formats.cgf.UVFace'> instance at ...
* t_0 : 4
* t_1 : 5
* t_2 : 6
3: <class 'pyffi.formats.cgf.UVFace'> instance at ...
* t_0 : 6
* t_1 : 5
* t_2 : 7
* vertex_colors :
<class 'pyffi.object_models.xml.array.Array'> instance at ...
0: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 0
* g : 1
* b : 2

```

(continues on next page)

(continued from previous page)

```

1: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 4
* g : 5
* b : 6
2: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 8
* g : 9
* b : 10
3: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 12
* g : 13
* b : 14
4: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 50
* g : 51
* b : 52
5: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 54
* g : 55
* b : 56
6: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 58
* g : 59
* b : 60
7: <class 'pyffi.formats.cgf.IRGB'> instance at ...
* r : 62
* g : 63
* b : 64
* vertices_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at _
↳...
* normals_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at .
↳..
* uvs_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* colors_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at ..
↳.
* colors_2_data : None
* indices_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at .
↳..
* tangents_data : <class 'pyffi.formats.cgf.DataStreamChunk'> instance at _
↳...
* sh_coeffs_data : None
* shape_deformation_data : None
* bone_map_data : None
* face_map_data : None
* vert_mats_data : None
* reserved_data :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: None
  1: None
  2: None
  3: None
* physics_data :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: None
  1: None
  2: None
  3: None

```

(continues on next page)

(continued from previous page)

```

* min_bound : [ 0.000 0.000 0.000 ]
* max_bound : [ 1.000 1.000 1.000 ]
* reserved_3 :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: 0
  1: 0
  2: 0
  3: 0
  4: 0
  5: 0
  6: 0
  7: 0
  8: 0
  9: 0
  10: 0
  11: 0
  12: 0
  13: 0
  14: 0
  15: 0
  16: 0
  etc...

>>> print(chunk.mesh_subsets)
<class 'pyffi.formats.cgf.MeshSubsetsChunk'> instance at ...
* flags :
  <class 'pyffi.formats.cgf.MeshSubsetsFlags'> instance at ...
  * sh_has_decompr_mat : 0
  * bone_indices : 0
* num_mesh_subsets : 2
* reserved_1 : 0
* reserved_2 : 0
* reserved_3 : 0
* mesh_subsets :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: <class 'pyffi.formats.cgf.MeshSubset'> instance at ...
  * first_index : 0
  * num_indices : 6
  * first_vertex : 0
  * num_vertices : 4
  * mat_id : 2
  * radius : 0.7071067...
  * center : [ 0.500 0.500 0.000 ]
  1: <class 'pyffi.formats.cgf.MeshSubset'> instance at ...
  * first_index : 6
  * num_indices : 6
  * first_vertex : 4
  * num_vertices : 4
  * mat_id : 5
  * radius : 0.7071067...
  * center : [ 0.500 0.500 1.000 ]

>>> print(chunk.vertices_data)
<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : VERTICES
* num_elements : 8

```

(continues on next page)

(continued from previous page)

```
* bytes_per_element : 12
* reserved_1 : 0
* reserved_2 : 0
* vertices :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: [ 0.000 0.000 0.000 ]
  1: [ 0.000 1.000 0.000 ]
  2: [ 1.000 0.000 0.000 ]
  3: [ 1.000 1.000 0.000 ]
  4: [ 0.000 0.000 1.000 ]
  5: [ 0.000 1.000 1.000 ]
  6: [ 1.000 0.000 1.000 ]
  7: [ 1.000 1.000 1.000 ]

>>> print(chunk.normals_data)
<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : NORMALS
* num_elements : 8
* bytes_per_element : 12
* reserved_1 : 0
* reserved_2 : 0
* normals :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: [ 0.000 0.000 -1.000 ]
  1: [ 0.000 0.000 -1.000 ]
  2: [ 0.000 0.000 -1.000 ]
  3: [ 0.000 0.000 -1.000 ]
  4: [ 0.000 0.000 1.000 ]
  5: [ 0.000 0.000 1.000 ]
  6: [ 0.000 0.000 1.000 ]
  7: [ 0.000 0.000 1.000 ]

>>> print(chunk.indices_data)
<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : INDICES
* num_elements : 12
* bytes_per_element : 2
* reserved_1 : 0
* reserved_2 : 0
* indices :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: 0
  1: 1
  2: 2
  3: 2
  4: 1
  5: 3
  6: 4
  7: 5
  8: 6
  9: 6
  10: 5
  11: 7

>>> print(chunk.uvs_data)
```

(continues on next page)

(continued from previous page)

```

<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : UVS
* num_elements : 8
* bytes_per_element : 8
* reserved_1 : 0
* reserved_2 : 0
* uvs :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 0.0
  * v : 1.0
  1: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 0.0
  * v : 0.0
  2: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 1.0
  * v : 1.0
  3: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 1.0
  * v : 0.0
  4: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 0.0
  * v : 1.0
  5: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 0.0
  * v : 0.0
  6: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 1.0
  * v : 1.0
  7: <class 'pyffi.formats.cgf.UV'> instance at ...
  * u : 1.0
  * v : 0.0

>>> print(chunk.tangents_data)
<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : TANGENTS
* num_elements : 8
* bytes_per_element : 16
* reserved_1 : 0
* reserved_2 : 0
* tangents :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
  * x : 32767
  * y : 0
  * z : 0
  * w : 32767
  0, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
  * x : 0
  * y : -32767
  * z : 0
  * w : 32767
  1, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
  * x : 32767
  * y : 0

```

(continues on next page)

(continued from previous page)

```
* z : 0
* w : 32767
1, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
2, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
2, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
3, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
3, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
4, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
4, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
5, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
5, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
6, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
6, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767
```

(continues on next page)

(continued from previous page)

```

7, 0: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 32767
* y : 0
* z : 0
* w : 32767
7, 1: <class 'pyffi.formats.cgf.Tangent'> instance at ...
* x : 0
* y : -32767
* z : 0
* w : 32767

>>> print(chunk.colors_data)
<class 'pyffi.formats.cgf.DataStreamChunk'> instance at ...
* flags : 0
* data_stream_type : COLORS
* num_elements : 8
* bytes_per_element : 4
* reserved_1 : 0
* reserved_2 : 0
* rgba_colors :
  <class 'pyffi.object_models.xml.array.Array'> instance at ...
  0: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 0
  * g : 1
  * b : 2
  * a : 3
  1: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 4
  * g : 5
  * b : 6
  * a : 7
  2: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 8
  * g : 9
  * b : 10
  * a : 11
  3: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 12
  * g : 13
  * b : 14
  * a : 15
  4: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 50
  * g : 51
  * b : 52
  * a : 53
  5: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 54
  * g : 55
  * b : 56
  * a : 57
  6: <class 'pyffi.formats.cgf.IRGBA'> instance at ...
  * r : 58
  * g : 59
  * b : 60
  * a : 61
  7: <class 'pyffi.formats.cgf.IRGBA'> instance at ...

```

(continues on next page)

```
* r : 62
* g : 63
* b : 64
* a : 65
```

**Parameters**

- **verticeslist** – A list of lists of vertices (one list per material).
- **normalslist** – A list of lists of normals (one list per material).
- **triangleslist** – A list of lists of triangles (one list per material).
- **matlist** – A list of material indices. Optional.
- **uvslist** – A list of lists of uvs (one list per material). Optional.
- **colorslist** – A list of lists of RGBA colors (one list per material). Optional. Each color is a tuple (r, g, b, a) with each component an integer between 0 and 255.

**set\_vertices\_normals** (*vertices, normals*)

B{Deprecated. Use L{set\_geometry} instead.} Set vertices and normals. This used to be the first function to call when setting mesh geometry data.

Returns list of chunks that have been added.

**update\_tangent\_space** ()

Recalculate tangent space data.

**class MtlListChunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Obsolete, not decoded.

**class PatchMeshChunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Obsolete, not decoded.

**class Ptr** (*\*\*kwargs*)

Bases: `pyffi.formats.cgf.Ref`

Reference to a chunk, down the hierarchy.

**get\_refs** (*data=None*)

Ptr does not point down, so get\_refs returns empty list.

**Returns** C{[]}

**class Quat** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A quaternion (x,y,z,w).

**property w**

Fourth coordinate.

**property x**

First coordinate.

**property y**

Second coordinate.

**property z**

Third coordinate.

**class Ref** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

Reference to a chunk, up the hierarchy.

**fix\_links** (*data*)

Resolve chunk index into a chunk.

**Keyword Arguments** **block\_dct** – Dictionary mapping block index to block.

**get\_hash** (*data=None*)

Return a hash value for the chunk referred to.

**Returns** An immutable object that can be used as a hash.

**get\_links** (*data=None*)

Return the chunk reference.

**Returns** Empty list if no reference, or single item list containing the reference.

**get\_refs** (*data=None*)

Return the chunk reference.

**Returns** Empty list if no reference, or single item list containing the reference.

**get\_size** (*data=None*)

Return number of bytes this type occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Get chunk being referred to.

**Returns** The chunk being referred to.

**read** (*stream, data*)

Read chunk index.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Set chunk reference.

**Parameters** **value** (*L{CgfFormat.Chunk}*) – The value to assign.

**write** (*stream, data*)

Write chunk index.

**Parameters** **stream** (*file*) – The stream to write to.

**class ScenePropsChunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Not decoded. Nowhere used?

**class SizedString** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
```

(continues on next page)

```

>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'

```

**get\_hash** (*data=None*)

Return a hash value for this string.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes this type occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Return the string.

**Returns** The stored string.

**read** (*stream, data*)

Read string from stream.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Set string to C{value}.

**Parameters** **value** (*str*) – The value to assign.

**write** (*stream, data*)

Write string to stream.

**Parameters** **stream** (*file*) – The stream to write to.

## String

alias of `pyffi.object_models.common.ZString`

**class String128** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 128.

**class String16** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 16.

**class String256** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 256.

**class String32** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 32.

**class String64** (*\*\*kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 64.

**class Tangent** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Tangents. Divide each component by 32767 to get the actual value.

**property w**

Handness? Either 32767 (+1.0) or -32767 (-1.0).

**class UV** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Texture coordinate.

**class UVFace** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A texture face (vertex indices).

**property t\_0**

First vertex index.

**property t\_1**

Second vertex index.

**property t\_2**

Third vertex index.

**class UnknownAAFC0005Chunk** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.cgf.Chunk`

Unknown. An extra block written by the XSI exporter.

**class Vector3** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.cgf._Vector3, object`

**bool**

alias of `pyffi.object_models.common.Bool`

**byte**

alias of `pyffi.object_models.common.Byte`

**char**

alias of `pyffi.object_models.common.Char`

**float**

alias of `pyffi.object_models.common.Float`

**int**

alias of `pyffi.object_models.common.Int`

**short**

alias of `pyffi.object_models.common.Short`

**ubyte**

alias of `pyffi.object_models.common.UByte`

**uint**

alias of `pyffi.object_models.common.UInt`

**ushort**

alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** `version_str` (*str*) – The version string.

**Returns** A version integer.

```
>>> hex(CgfFormat.version_number('744'))
'0x744'
```

## Regression tests

### Read a CGF file

```
>>> # get file version and file type, and read cgf file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'cgf')
>>> stream = open(os.path.join(format_root, 'test.cgf'), 'rb')
>>> data = CgfFormat.Data()
>>> # read chunk table only
>>> data.inspect(stream)
>>> # check chunk types
>>> list(chunktype.__name__ for chunktype in data.chunk_table.get_chunk_types())
['SourceInfoChunk', 'TimingChunk']
>>> data.chunks # no chunks yet
[]
>>> # read full file
>>> data.read(stream)
>>> # get all chunks
>>> for chunk in data.chunks:
...     print(chunk)
<class '...SourceInfoChunk'> instance at ...
* source_file : <None>
* date : Fri Sep 28 22:40:44 2007
* author : blender@BLENDER

<class '...TimingChunk'> instance at ...
* secs_per_tick : 0.0002083333...
* ticks_per_frame : 160
* global_range :
  <class '...RangeEntity'> instance at ...
  * name : GlobalRange
  * start : 0
  * end : 100
* num_sub_ranges : 0
```

### Parse all CGF files in a directory tree

```
>>> for stream, data in CgfFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
```

(continues on next page)

(continued from previous page)

```

...     data.read(stream)
...     except Exception:
...         print("Warning: read failed due corrupt file, corrupt format description,
↳or bug.")
...     print(len(data.chunks))
...     # do something with the chunks
...     for chunk in data.chunks:
...         chunk.apply_scale(2.0)
reading tests/formats/cgf/invalid.cgf
Warning: read failed due corrupt file, corrupt format description, or bug.
0
reading tests/formats/cgf/monkey.cgf
14
reading tests/formats/cgf/test.cgf
2
reading tests/formats/cgf/vcols.cgf
6

```

### Create a CGF file from scratch

```

>>> from pyffi.formats.cgf import CgfFormat
>>> node1 = CgfFormat.NodeChunk()
>>> node1.name = "hello"
>>> node2 = CgfFormat.NodeChunk()
>>> node1.num_children = 1
>>> node1.children.update_size()
>>> node1.children[0] = node2
>>> node2.name = "world"
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data = CgfFormat.Data() # default is far cry
>>> data.chunks = [node1, node2]
>>> # note: write returns number of padding bytes
>>> data.write(stream)
0
>>> # py3k returns 0 on seek; this hack removes return code from doctest
>>> if stream.seek(0): pass
>>> data.inspect_version_only(stream)
>>> hex(data.header.version)
'0x744'
>>> data.read(stream)
>>> # get all chunks
>>> for chunk in data.chunks:
...     print(chunk)
<class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...
* name : hello
* object : None
* parent : None
* num_children : 1
* material : None
* is_group_head : False
* is_group_member : False
* reserved_1 :
    <class 'pyffi.object_models.xml.array.Array'> instance at 0x...

```

(continues on next page)

```
0: 0
1: 0
* transform :
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
* pos : [ 0.000 0.000 0.000 ]
* rot :
  <class 'pyffi.formats.cgf.Quat'> instance at 0x...
  * x : 0.0
  * y : 0.0
  * z : 0.0
  * w : 0.0
* scl : [ 0.000 0.000 0.000 ]
* pos_ctrl : None
* rot_ctrl : None
* scl_ctrl : None
* property_string : <None>
* children :
  <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
  0: <class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...

<class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...
* name : world
* object : None
* parent : None
* num_children : 0
* material : None
* is_group_head : False
* is_group_member : False
* reserved_1 :
  <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
  0: 0
  1: 0
* transform :
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
* pos : [ 0.000 0.000 0.000 ]
* rot :
  <class 'pyffi.formats.cgf.Quat'> instance at 0x...
  * x : 0.0
  * y : 0.0
  * z : 0.0
  * w : 0.0
* scl : [ 0.000 0.000 0.000 ]
* pos_ctrl : None
* rot_ctrl : None
* scl_ctrl : None
* property_string : <None>
* children : <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
```

**pyffi.formats.dae — COLLADA (.dae)**

**Warning:** This module is not yet fully implemented, and is certainly not yet useful in its current state.

**Implementation**

**class** `pyffi.formats.dae.DaeFormat`

Bases: `pyffi.object_models.xsd.FileFormat`

This class implements the DAE format.

**class** `Data` (*version=17039616*)

Bases: `pyffi.object_models.Data`

A class to contain the actual collada data.

**getVersion** ()

Get the collada version, as integer (for instance, 1.4.1 would be 0x01040100).

**Returns** The version, as integer.

**inspect** (*stream*)

Quickly checks whether the stream appears to contain collada data. Resets stream to original position. If the stream turns out to be collada, `L{getVersion}` is guaranteed to return the version.

Call this function if you simply wish to check that a file is a collada file without having to parse it completely.

**Parameters** `stream` (*file*) – The file to inspect.

**Returns** `True` if stream is collada, `False` otherwise.

**read** (*stream*)

Read collada data from stream.

**Parameters** `stream` (*file*) – The file to read from.

**write** (*stream*)

Write collada data to stream.

**Parameters** `stream` (*file*) – The file to write to.

**Regression tests****Create a DAE file**

```
>>> daedata = DaeFormat.Data()
>>> print(daedata.collada)
<...Collada object at ...>
```

**Read a DAE file**

```
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
```

(continues on next page)

(continued from previous page)

```

>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'dae')
>>> # check and read dae file
>>> stream = open(os.path.join(format_root, 'cube.dae'), 'rb')
>>> daedata = DaeFormat.Data()
>>> daedata.read(stream)
Traceback (most recent call last):
...
NotImplementedError
>>> # get DAE file root element
>>> #print(daedata.getRootElement())
>>> stream.close()

```

## Parse all DAE files in a directory tree

```

>>> for stream, data in DaeFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print("Warning: read failed due corrupt file, corrupt format description,
↳ or bug.")
reading tests/formats/dae/cube.dae
Warning: read failed due corrupt file, corrupt format description, or bug.

```

## Create a DAE file from scratch and write to file

```

>>> daedata = DaeFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> daedata.write(stream)
Traceback (most recent call last):
...
NotImplementedError

```

## pyffi.formats.dds — DirectDraw Surface (.dds)

### Implementation

```

class pyffi.formats.dds.DdsFormat
    Bases: pyffi.object_models.xml.FileFormat
    This class implements the DDS format.

class Data (version=150994944)
    Bases: pyffi.object_models.Data
    A class to contain the actual dds data.

```

**get\_detail\_child\_names** (*edge\_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

**Returns** Generator for detail tree child names.

**Return type** generator yielding `str`s

**get\_detail\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

**Parameters** **edge\_filter** (`EdgeFilter` or type (`None`)) – The edge type to include.

**Returns** Generator for detail tree child nodes.

**Return type** generator yielding `DetailNodes`

**inspect** (*stream*)

Quickly checks if stream contains DDS data, and reads the header.

**Parameters** **stream** (*file*) – The stream to inspect.

**inspect\_quick** (*stream*)

Quickly checks if stream contains DDS data, and gets the version, by looking at the first 8 bytes.

**Parameters** **stream** (*file*) – The stream to inspect.

**read** (*stream, verbose=0*)

Read a dds file.

**Parameters**

- **stream** (*file*) – The stream from which to read.
- **verbose** (*int*) – The level of verbosity.

**write** (*stream, verbose=0*)

Write a dds file.

**Parameters**

- **stream** (*file*) – The stream to which to write.
- **verbose** (*int*) – The level of verbosity.

**class FourCC** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 32-bit integer, describing the compression type.

**class HeaderString** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

Basic type which implements the header of a DDS file.

**get\_detail\_display** ()

Return an object that can be used to display the instance.

**get\_hash** (*data=None*)

Return a hash value for this value.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes the header string occupies in a file.

**Returns** Number of bytes.

**read** (*stream, data*)

Read header string from stream and check it.

**Parameters** **stream** (*file*) – The stream to read from.

**write** (*stream*, *data*)

Write the header string to stream.

**Parameters** **stream** (*file*) – The stream to write to.

**ImageData**

alias of `pyffi.object_models.common.UndecodedData`

**byte**

alias of `pyffi.object_models.common.Byte`

**char**

alias of `pyffi.object_models.common.Char`

**float**

alias of `pyffi.object_models.common.Float`

**int**

alias of `pyffi.object_models.common.Int`

**short**

alias of `pyffi.object_models.common.Short`

**ubyte**

alias of `pyffi.object_models.common.UByte`

**uint**

alias of `pyffi.object_models.common.UInt`

**ushort**

alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** **version\_str** (*str*) – The version string.

**Returns** A version integer.

```
>>> hex(DdsFormat.version_number('DX10'))
'0xa000000'
```

## Regression tests

### Read a DDS file

```
>>> # check and read dds file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'dds')
>>> file = os.path.join(format_root, 'test.dds')
>>> stream = open(file, 'rb')
>>> data = DdsFormat.Data()
>>> data.inspect(stream)
>>> data.header.pixel_format.size
32
```

(continues on next page)

(continued from previous page)

```
>>> data.header.height
20
>>> data.read(stream)
>>> len(data.pixeldata.get_value())
888
```

### Parse all DDS files in a directory tree

```
>>> for stream, data in DdsFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/dds/test.dds
```

### Create a DDS file from scratch and write to file

```
>>> data = DdsFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

### Get list of versions

```
>>> for vnum in sorted(DdsFormat.versions.values()):
...     print('0x%08X' % vnum)
0x09000000
0x0A000000
```

### pyffi.formats.egm — EGM (.egm)

An .egm file contains facial shape modifiers, that is, morphs that modify static properties of the face, such as nose size, chin shape, and so on.

#### Implementation

```
class pyffi.formats.egm.EgmFormat
    Bases: pyffi.object_models.xml.FileFormat
```

This class implements the EGM format.

```
class Data (version=2, num_vertices=0)
    Bases: pyffi.object_models.Data

    A class to contain the actual egm data.

    add_asym_morph ()
        Add an asymmetric morph, and return it.

    add_sym_morph ()
        Add a symmetric morph, and return it.

    apply_scale (scale)
        Apply scale factor to all morphs.

    get_detail_child_names (edge_filter=(True, True))
        Generator which yields all child names of this item in the detail view.

        Override this method if the node has children.
        Returns Generator for detail tree child names.
        Return type generator yielding strs

    get_detail_child_nodes (edge_filter=(True, True))
        Generator which yields all children of this item in the detail view (by default, all acyclic and active
        ones).

        Override this method if the node has children.
        Parameters edge_filter (EdgeFilter or type (None)) – The edge type to include.
        Returns Generator for detail tree child nodes.
        Return type generator yielding DetailNodes

    get_global_child_nodes (edge_filter=(True, True))
        Generator which yields all children of this item in the global view, of given edge type (default is edges
        of type 0).

        Override this method.
        Returns Generator for global node children.

    inspect (stream)
        Quickly checks if stream contains EGM data, and reads the header.
        Parameters stream (file) – The stream to inspect.

    inspect_quick (stream)
        Quickly checks if stream contains EGM data, and gets the version, by looking at the first 8 bytes.
        Parameters stream (file) – The stream to inspect.

    read (stream)
        Read a egm file.
        Parameters stream (file) – The stream from which to read.

    write (stream)
        Write a egm file.
        Parameters stream (file) – The stream to which to write.

class FileSignature (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Basic type which implements the header of a EGM file.

    get_detail_display ()
        Return an object that can be used to display the instance.

    get_hash (data=None)
        Return a hash value for this value.
```

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes the header string occupies in a file.

**Returns** Number of bytes.

**read** (*stream, data*)

Read header string from stream and check it.

**Parameters** **stream** (*file*) – The stream to read from.

**write** (*stream, data*)

Write the header string to stream.

**Parameters** **stream** (*file*) – The stream to write to.

**class FileVersion** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

**get\_detail\_display** ()

Return an object that can be used to display the instance.

**get\_hash** (*data=None*)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)

Returns size of the object in bytes.

**get\_value** ()

Return object value.

**read** (*stream, data*)

Read object from file.

**set\_value** (*value*)

Set object value.

**write** (*stream, data*)

Write object to file.

**class MorphRecord** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.egm._MorphRecord, object`

```
>>> # create morph with 3 vertices.
>>> morph = EgmFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> # scale should be 9/32768.0 = 0.0002746...
>>> morph.scale
0.0002746...
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[3000, 5000, 2000]
[1000, 3000, 2000]
[-8999, 3000, -999]
```

**apply\_scale** (*scale*)

Apply scale factor to data.

```
>>> # create morph with 3 vertices.
>>> morph = EgmFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
```

(continues on next page)

(continued from previous page)

```

>>> morph.apply_scale(2)
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[6000, 10000, 4000]
[2000, 6000, 4000]
[-17999, 6000, -1999]

```

**byte**  
alias of `pyffi.object_models.common.Byte`

**char**  
alias of `pyffi.object_models.common.Char`

**float**  
alias of `pyffi.object_models.common.Float`

**int**  
alias of `pyffi.object_models.common.Int`

**short**  
alias of `pyffi.object_models.common.Short`

**ubyte**  
alias of `pyffi.object_models.common.UByte`

**uint**  
alias of `pyffi.object_models.common.UInt`

**ushort**  
alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)  
Converts version string into an integer.

**Parameters** *version\_str* (*str*) – The version string.

**Returns** A version integer.

```

>>> EgmFormat.version_number('002')
2
>>> EgmFormat.version_number('XXX')
-1

```

## Regression tests

### Read a EGM file

```

>>> # check and read egm file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'egm')
>>> file = os.path.join(format_root, 'mmouthxivilai.egm')
>>> stream = open(file, 'rb')
>>> data = EgmFormat.Data()

```

(continues on next page)

(continued from previous page)

```

>>> data.inspect_quick(stream)
>>> data.version
2
>>> data.inspect(stream)
>>> data.header.num_vertices
89
>>> data.header.num_sym_morphs
50
>>> data.header.num_asym_morphs
30
>>> data.header.time_date_stamp
2001060901
>>> data.read(stream)
>>> data.sym_morphs[0].vertices[0].x
17249

```

### Parse all EGM files in a directory tree

```

>>> for stream, data in EgmFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/egm/mmouthisvilai.egm

```

### Create an EGM file from scratch and write to file

```

>>> data = EgmFormat.Data(num_vertices=10)
>>> data.header.num_vertices
10
>>> morph = data.add_sym_morph()
>>> len(morph.vertices)
10
>>> morph.scale = 0.4
>>> morph.vertices[0].z = 123
>>> morph.vertices[9].x = -30000
>>> morph = data.add_asym_morph()
>>> morph.scale = 2.3
>>> morph.vertices[3].z = -5
>>> morph.vertices[4].x = 99
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)

```

**pyffi.formats.egt — EGT (.egt)**

An .egt file contains texture tones for the different races.

**Implementation**

**class** pyffi.formats.egt.EgtFormat

Bases: pyffi.object\_models.xml.FileFormat

This class implements the EGT format.

**Data**

alias of *Header*

**class** FileSignature (\*\*kwargs)

Bases: pyffi.object\_models.xml.basic.BasicBase

Basic type which implements the header of a EGT file.

**get\_detail\_display**()

Return an object that can be used to display the instance.

**get\_hash**(data=None)

Return a hash value for this value.

**Returns** An immutable object that can be used as a hash.

**get\_size**(data=None)

Return number of bytes the header segtng occupies in a file.

**Returns** Number of bytes.

**read**(stream, data)

Read header string from stream and check it.

**Parameters** **stream**(file) – The stream to read from.

**write**(stream, data)

Write the header segtng to stream.

**Parameters** **stream**(file) – The stream to write to.

**class** FileVersion (template=None, argument=None, parent=None)

Bases: pyffi.object\_models.xml.basic.BasicBase

**get\_detail\_display**()

Return an object that can be used to display the instance.

**get\_hash**(data=None)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size**(data=None)

Returns size of the object in bytes.

**get\_value**()

Return object value.

**read**(stream, data)

Read object from file.

**set\_value**(value)

Set object value.

**write**(stream, data)

Write object to file.

**class Header** (*template=None, argument=None, parent=None*)  
 Bases: `pyffi.formats.egt._Header`, `pyffi.object_models.Data`

A class to contain the actual egt data.

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)  
 Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.  
**Returns** Generator for global node children.

**inspect** (*stream*)  
 Quickly checks if stream contains EGT data, and reads everything up to the arrays.  
**Parameters** **stream** (*file*) – The stream to inspect.

**inspect\_quick** (*stream*)  
 Quickly checks if stream contains EGT data, by looking at the first 8 bytes. Reads the signature and the version.  
**Parameters** **stream** (*file*) – The stream to inspect.

**read** (*stream*)  
 Read a egt file.  
**Parameters** **stream** (*file*) – The stream from which to read.

**write** (*stream*)  
 Write a egt file.  
**Parameters** **stream** (*file*) – The stream to which to write.

**byte**  
 alias of `pyffi.object_models.common.Byte`

**char**  
 alias of `pyffi.object_models.common.Char`

**float**  
 alias of `pyffi.object_models.common.Float`

**int**  
 alias of `pyffi.object_models.common.Int`

**short**  
 alias of `pyffi.object_models.common.Short`

**ubyte**  
 alias of `pyffi.object_models.common.UByte`

**uint**  
 alias of `pyffi.object_models.common.UInt`

**ushort**  
 alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)  
 Converts version segtng into an integer.  
**Parameters** **version\_str** (*str*) – The version segtng.  
**Returns** A version integer.

```
>>> EgtFormat.version_number('003')
3
```

(continues on next page)

(continued from previous page)

```
>>> EgtFormat.version_number('XXX')
-1
```

## Regression tests

### Read a EGT file

```
>>> # check and read egt file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'egt')
>>> file = os.path.join(format_root, 'test.egt')
>>> stream = open(file, 'rb')
>>> data = EgtFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> data.read(stream)
>>> # do more stuff?
```

### Parse all EGT files in a directory tree

```
>>> for stream, data in EgtFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/egt/test.egt
```

### Create an EGT file from scratch and write to file

```
>>> data = EgtFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

`pyffi.formats.esp` — Elder Scrolls plugin/master/save files (.esp, .esm, and .ess)

## Implementation

**class** `pyffi.formats.esp.EspFormat`

Bases: `pyffi.object_models.xml.FileFormat`

This class implements the ESP format.

**class** `Data`

Bases: `pyffi.object_models.Data`

A class to contain the actual esp data.

**get\_detail\_child\_names** (*edge\_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

**Returns** Generator for detail tree child names.

**Return type** generator yielding `str`s

**get\_detail\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

**Parameters** `edge_filter` (`EdgeFilter` or `type (None)`) – The edge type to include.

**Returns** Generator for detail tree child nodes.

**Return type** generator yielding `DetailNodes`

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

**Returns** Generator for global node children.

**inspect** (*stream*)

Quickly checks if stream contains ESP data, and reads the header.

**Parameters** `stream` (*file*) – The stream to inspect.

**inspect\_quick** (*stream*)

Quickly checks if stream contains ESP data, and gets the version, by looking at the first 8 bytes.

**Parameters** `stream` (*file*) – The stream to inspect.

**read** (*stream*)

Read a esp file.

**Parameters** `stream` (*file*) – The stream from which to read.

**write** (*stream*)

Write a esp file.

**Parameters** `stream` (*file*) – The stream to which to write.

**class** `GRUP`

Bases: `pyffi.formats.esp._GRUP, object`

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

**Returns** Generator for global node children.

```

read (stream, data)
    Read structure from stream.

write (stream, data)
    Write structure to stream.

class Record
    Bases: pyffi.formats.esp._Record, object

get_global_child_nodes (edge_filter=(True, True))
    Generator which yields all children of this item in the global view, of given edge type (default is edges
    of type 0).

    Override this method.
    Returns Generator for global node children.

get_sub_record (sub_record_type)
    Find first subrecord of given type.

read (stream, data)
    Read structure from stream.

write (stream, data)
    Write structure to stream.

class RecordType (**kwargs)
    Bases: pyffi.object_models.common.FixedString

class SubRecord (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A subrecord.

property data_size
    Length of the data.

property type
    The type of the record.

class ZString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase, pyffi.object_models.
    editable.EditableLineEdit

    String of variable length (null terminated).

```

```

>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> s = ZString()
>>> if f.write('abcdefghijklmnopqrst\x00').encode("ascii"): pass # b'abc...'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f)
>>> str(s)
'abcdefghijklmnopqrst'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There!')
>>> s.write(f)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = ZString()
>>> m.read(f)
>>> str(m)
'Hi There!'

```

**get\_hash** (*data=None*)

Return a hash value for this string.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes this type occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Return the string.

**Returns** The stored string.

**Return type** C{bytes}

**read** (*stream, data=None*)

Read string from stream.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Set string to C{value}.

**Parameters** **value** (*str* (will be encoded as default) or C{bytes}) – The value to assign.

**write** (*stream, data=None*)

Write string to stream.

**Parameters** **stream** (*file*) – The stream to write to.

**byte**

alias of `pyffi.object_models.common.Byte`

**char**

alias of `pyffi.object_models.common.Char`

**float**

alias of `pyffi.object_models.common.Float`

**int**

alias of `pyffi.object_models.common.Int`

**short**

alias of `pyffi.object_models.common.Short`

**ubyte**

alias of `pyffi.object_models.common.UByte`

**uint**

alias of `pyffi.object_models.common.UInt`

**uint64**

alias of `pyffi.object_models.common.UInt64`

**ushort**

alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** **version\_str** (*str*) – The version string.

**Returns** A version integer.

```
>>> hex(EspFormat.version_number('1.2'))
'0x102'
```

## Regression tests

### Read a ESP file

```
>>> # check and read esp file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'esp')
>>> file = os.path.join(format_root, 'test.esp')
>>> stream = open(file, 'rb')
>>> data = EspFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> #data.header....
>>> data.read(stream)
>>> # do some stuff...
```

### Parse all ESP files in a directory tree

```
>>> for stream, data in EspFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/esp/test.esp
```

### Create an ESP file from scratch and write to file

```
>>> data = EspFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

## pyffi.formats.kfm — NetImmerse/Gamebryo Keyframe Motion (.kfm)

### Implementation

```
class pyffi.formats.kfm.KfmFormat
    Bases: pyffi.object_models.xml.FileFormat

    This class implements the kfm file format.
```

```

class Data (version=33685515)
    Bases: pyffi.object_models.Data

    A class to contain the actual kfm data.

    get_global_child_nodes (edge_filter=(True, True))
        Generator which yields all children of this item in the global view, of given edge type (default is edges
        of type 0).

        Override this method.
        Returns Generator for global node children.

    get_global_display ()
        Display the KFM file name.

    inspect (stream)
        Quick heuristic check if stream contains KFM data, by looking at the first 64 bytes. Sets version and
        reads header string.
        Parameters stream (file) – The stream to inspect.

    read (stream)
        Read a kfm file.
        Parameters stream (file) – The stream from which to read.

    write (stream)
        Write a kfm file.
        Parameters stream (file) – The stream to which to write.

class FilePath (**kwargs)
    Bases: pyffi.object_models.common.SizedString

    get_hash (data=None)
        Return a hash value for this value. For file paths, the hash value is case insensitive.
        Returns An immutable object that can be used as a hash.

class HeaderString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    The kfm header string.

    get_detail_display ()
        Return an object that can be used to display the instance.

    get_hash (data=None)
        Return a hash value for this value.
        Returns An immutable object that can be used as a hash.

    get_size (data=None)
        Return number of bytes the header string occupies in a file.
        Returns Number of bytes.

    get_value ()
        Return object value.

    read (stream, data)
        Read header string from stream and check it.
        Parameters
            • stream (file) – The stream to read from.
            • data (pyffi.formats.kfm.KfmFormat.Data) – The KfmFormat.Data()

    set_value (value)
        Set object value.

```

**static version\_string**(*version*)

Transforms version number into a version string.

**Parameters** *version* (*int*) – The version number.

**Returns** A version string.

```
>>> KfmFormat.HeaderString.version_string(0x0202000b)
';Gamebryo KFM File Version 2.2.0.0b'
>>> KfmFormat.HeaderString.version_string(0x01024b00)
';Gamebryo KFM File Version 1.2.4b'
```

**write**(*stream*, *data*)

Write the header string to stream.

**Parameters**

- **stream** (*file*) – The stream to write to.
- **data** (*Data*) – The fileformat data to use

**class SizedString** (\*\**kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'
```

**get\_hash**(*data=None*)

Return a hash value for this string.

**Returns** An immutable object that can be used as a hash.

**get\_size**(*data=None*)

Return number of bytes this type occupies in a file.

**Returns** Number of bytes.

**get\_value**()

Return the string.

**Returns** The stored string.

**read**(*stream*, *data*)

Read string from stream.

**Parameters** *stream* (*file*) – The stream to read from.

**set\_value** (*value*)  
Set string to C{value}.  
**Parameters** **value** (*str*) – The value to assign.

**write** (*stream, data*)  
Write string to stream.  
**Parameters** **stream** (*file*) – The stream to write to.

**TextString**

alias of `pyffi.object_models.common.UndecodedData`

**byte**

alias of `pyffi.object_models.common.UByte`

**char**

alias of `pyffi.object_models.common.Char`

**float**

alias of `pyffi.object_models.common.Float`

**int**

alias of `pyffi.object_models.common.Int`

**short**

alias of `pyffi.object_models.common.Short`

**uint**

alias of `pyffi.object_models.common.UInt`

**ushort**

alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** **version\_str** (*str*) – The version string.

**Returns** A version integer.

```
>>> hex(KfmFormat.version_number('1.0'))
'0x1000000'
>>> hex(KfmFormat.version_number('1.2.4b'))
'0x1024b00'
>>> hex(KfmFormat.version_number('2.2.0.0b'))
'0x202000b'
```

## Regression tests

### Read a KFM file

```
>>> # read kfm file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> files_dir = os.path.join(repo_root, 'tests', 'spells', 'kfm', 'files')
>>> file = os.path.join(files_dir, 'test.kfm')
```

(continues on next page)

(continued from previous page)

```

>>> stream = open(file, 'rb')
>>> data = KfmFormat.Data()
>>> data.inspect(stream)
>>> data.read(stream)
>>> stream.close()
>>> print(data.nif_file_name.decode("ascii"))
Test.nif
>>> # get all animation file names
>>> for anim in data.animations:
...     print(anim.kf_file_name.decode("ascii"))
Test_MD_Idle.kf
Test_MD_Run.kf
Test_MD_Walk.kf
Test_MD_Die.kf

```

### Parse all KFM files in a directory tree

```

>>> for stream, data in KfmFormat.walkData(files_dir):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-5:]
...         rejoin = os.path.join(*split).replace("\\", "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/spells/kfm/files/invalid.kfm
reading tests/spells/kfm/files/test.kfm

```

### Create a KFM model from scratch and write to file

```

>>> data = KfmFormat.Data()
>>> data.nif_file_name = "Test.nif"
>>> data.num_animations = 4
>>> data.animations.update_size()
>>> data.animations[0].kf_file_name = "Test_MD_Idle.kf"
>>> data.animations[1].kf_file_name = "Test_MD_Run.kf"
>>> data.animations[2].kf_file_name = "Test_MD_Walk.kf"
>>> data.animations[3].kf_file_name = "Test_MD_Die.kf"
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
>>> stream.close()

```

### Get list of versions and games

```

>>> for vnum in sorted(KfmFormat.versions.values()):
...     print('0x%08X' % vnum)

```

(continues on next page)

(continued from previous page)

```

0x01000000
0x01024B00
0x0200000B
0x0201000B
0x0202000B
0x0202001B
>>> for game, versions in sorted(KfmFormat.games.items(),
...                               key=lambda x: x[0]):
...     print("%s " % game + " ".join('0x%08X' % vnum for vnum in versions))
Civilization IV 0x01000000 0x01024B00 0x0200000B
Dragonica 0x0202001B
Emerge 0x0201000B 0x0202000B
Loki 0x01024B00
Megami Tensei: Imagine 0x0201000B
Oblivion 0x01024B00
Prison Tycoon 0x01024B00
Pro Cycling Manager 0x01024B00
Red Ocean 0x01024B00
Sid Meier's Railroads 0x0200000B
The Guild 2 0x01024B00

```

## pyffi.formats.nif — NetImmerse/Gamebryo (.nif and .kf)

### Implementation

**class** `pyffi.formats.nif.NifFormat`

Bases: `pyffi.object_models.xml.FileFormat`

This class contains the generated classes from the xml.

**ARCHIVE\_CLASSES** = [`<class 'pyffi.formats.bsa.BsaFormat'>`]

**class** `ATextureRenderData` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._ATextureRenderData, object`

**save\_as\_dds** (*stream*)

Save image as DDS file.

**class** `AVObject` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Used in `NiDefaultAVObjectPalette`.

**property** `av_object`

Object reference.

**property** `name`

Object name.

**class** `AbstractAdditionalGeometryData` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**class** `AdditionalDataBlock` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property** `block_offsets`

Unknown

**property block\_size**  
Size of Block

**property data**  
Unknown

**property data\_sizes**  
Unknown

**property has\_data**  
Has data

**property num\_blocks**  
Unknown

**property num\_data**  
Unknown

**class AdditionalDataInfo** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property block\_index**  
Unsure. The block in which this channel is stored? Usually there is only one block, and so this is zero.

**property channel\_offset**  
Offset (in bytes) of this channel. Sum of all num channel bytes per element of all preceding block infos.

**property data\_type**  
Type of data in this channel

**property num\_channel\_bytes**  
Total number of bytes of this channel (num vertices times num bytes per element)

**property num\_channel\_bytes\_per\_element**  
Number of bytes per element of this channel

**property num\_total\_bytes\_per\_element**  
Number of bytes per element in all channels together. Sum of num channel bytes per element over all block infos.

**property unknown\_byte\_1**  
Unknown, usually equal to 2.

**class AlphaFormat** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 32-bit integer, describing how transparency is handled in a texture.

**ALPHA\_BINARY = 1**

**ALPHA\_DEFAULT = 3**

**ALPHA\_NONE = 0**

**ALPHA\_SMOOTH = 2**

**class AnimationType** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Animation type used on this position. This specifies the function of this position.

**Lean = 4**

```

    Sit = 1
    Sleep = 2

class ApplyMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing the apply mode of a texture.
    APPLY_DECAL = 1
    APPLY_HILIGHT = 3
    APPLY_HILIGHT2 = 4
    APPLY_MODULATE = 2
    APPLY_REPLACE = 0

class ArkTexture (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A texture reference used by NiArkTextureExtraData.
    property texture_name
    property texturing_property
    property unknown_bytes
    property unknown_int_3
    property unknown_int_4

class AvoidNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Morrowind specific?

class BSAanimNotes (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Bethesda-specific node.
    property unknown_short_1
        Unknown

class BSBehaviorGraphExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Links a nif with a Havok Behavior .hvx animation file
    property behaviour_graph_file
        Name of the hvx file.
    property controls_base_skeleton
        Unknown, has to do with blending appended bones onto an actor.

class BSblastNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node.
    property unknown_byte_1
        Unknown
    property unknown_short_2
        Unknown

```

```
class BSBoneLODExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown

    property bone_l_o_d_count
        Number of bone entries

    property bone_l_o_d_info
        Bone Entry

class BSBound (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._BSBound, object

    apply_scale (scale)
        Scale data.

class BSDamageStage (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-Specific node.

    property unknown_byte_1
        Unknown

    property unknown_short_2
        Unknown

class BSDebrisNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-Specific node.

    property unknown_byte_1
        Unknown

    property unknown_short_2
        Unknown

class BSDecalPlacementVectorExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Bethesda-specific node. (for dynamic decal projection?)

    property num_vector_blocks
        Number of groups

    property unknown_float_1
        Unknown

    property vector_blocks
        Number of Blocks

class BSDismemberBodyPartType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Biped bodypart data used for visibility control of triangles. Options are Fallout 3, except where marked
    for Skyrim (uses SBP prefix)Skyrim BP names are listed only for vanilla names, different creatures have
    different definitions for naming.

    BP_BRAIN = 13

    BP_HEAD = 1

    BP_HEAD2 = 2
```

```
BP_LEFTARM = 3
BP_LEFTARM2 = 4
BP_LEFTLEG = 7
BP_LEFTLEG2 = 8
BP_LEFTLEG3 = 9
BP_RIGHTARM = 5
BP_RIGHTARM2 = 6
BP_RIGHTLEG = 10
BP_RIGHTLEG2 = 11
BP_RIGHTLEG3 = 12
BP_SECTIONCAP_BRAIN = 113
BP_SECTIONCAP_HEAD = 101
BP_SECTIONCAP_HEAD2 = 102
BP_SECTIONCAP_LEFTARM = 103
BP_SECTIONCAP_LEFTARM2 = 104
BP_SECTIONCAP_LEFTLEG = 107
BP_SECTIONCAP_LEFTLEG2 = 108
BP_SECTIONCAP_LEFTLEG3 = 109
BP_SECTIONCAP_RIGHTARM = 105
BP_SECTIONCAP_RIGHTARM2 = 106
BP_SECTIONCAP_RIGHTLEG = 110
BP_SECTIONCAP_RIGHTLEG2 = 111
BP_SECTIONCAP_RIGHTLEG3 = 112
BP_TORSO = 0
BP_TORSOCAP_BRAIN = 213
BP_TORSOCAP_HEAD = 201
BP_TORSOCAP_HEAD2 = 202
BP_TORSOCAP_LEFTARM = 203
BP_TORSOCAP_LEFTARM2 = 204
BP_TORSOCAP_LEFTLEG = 207
BP_TORSOCAP_LEFTLEG2 = 208
BP_TORSOCAP_LEFTLEG3 = 209
BP_TORSOCAP_RIGHTARM = 205
BP_TORSOCAP_RIGHTARM2 = 206
BP_TORSOCAP_RIGHTLEG = 210
BP_TORSOCAP_RIGHTLEG2 = 211
```

BP\_TORSOCAP\_RIGHTLEG3 = 212  
BP\_TORSOSECTION\_BRAIN = 13000  
BP\_TORSOSECTION\_HEAD = 1000  
BP\_TORSOSECTION\_HEAD2 = 2000  
BP\_TORSOSECTION\_LEFTARM = 3000  
BP\_TORSOSECTION\_LEFTARM2 = 4000  
BP\_TORSOSECTION\_LEFTLEG = 7000  
BP\_TORSOSECTION\_LEFTLEG2 = 8000  
BP\_TORSOSECTION\_LEFTLEG3 = 9000  
BP\_TORSOSECTION\_RIGHTARM = 5000  
BP\_TORSOSECTION\_RIGHTARM2 = 6000  
BP\_TORSOSECTION\_RIGHTLEG = 10000  
BP\_TORSOSECTION\_RIGHTLEG2 = 11000  
BP\_TORSOSECTION\_RIGHTLEG3 = 12000  
SBP\_130\_HEAD = 130  
SBP\_131\_HAIR = 131  
SBP\_141\_LONGHAIR = 141  
SBP\_142\_CIRCLET = 142  
SBP\_143\_EARS = 143  
SBP\_150\_DECAPITATEDHEAD = 150  
SBP\_230\_HEAD = 230  
SBP\_30\_HEAD = 30  
SBP\_31\_HAIR = 31  
SBP\_32\_BODY = 32  
SBP\_33\_HANDS = 33  
SBP\_34\_FOREARMS = 34  
SBP\_35\_AMULET = 35  
SBP\_36\_RING = 36  
SBP\_37\_FEET = 37  
SBP\_38\_CALVES = 38  
SBP\_39\_SHIELD = 39  
SBP\_40\_TAIL = 40  
SBP\_41\_LONGHAIR = 41  
SBP\_42\_CIRCLET = 42  
SBP\_43\_EARS = 43  
SBP\_44\_DRAGON\_BLOODHEAD\_OR\_MOD\_MOUTH = 44

```

SBP_45_DRAGON_BLOODWINGL_OR_MOD_NECK = 45
SBP_46_DRAGON_BLOODWINGR_OR_MOD_CHEST_PRIMARY = 46
SBP_47_DRAGON_BLOODTAIL_OR_MOD_BACK = 47
SBP_48_MOD_MISC1 = 48
SBP_49_MOD_PELVIS_PRIMARY = 49
SBP_50_DECAPITATEDHEAD = 50
SBP_51_DECAPITATE = 51
SBP_52_MOD_PELVIS_SECONDARY = 52
SBP_53_MOD_LEG_RIGHT = 53
SBP_54_MOD_LEG_LEFT = 54
SBP_55_MOD_FACE_JEWELRY = 55
SBP_56_MOD_CHEST_SECONDARY = 56
SBP_57_MOD_SHOULDER = 57
SBP_58_MOD_ARM_LEFT = 58
SBP_59_MOD_ARM_RIGHT = 59
SBP_60_MOD_MISC2 = 60
SBP_61_FX01 = 61

class BSDismemberSkinInstance (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._BSDismemberSkinInstance, object

    get_dismember_partitions ()
        Return triangles and body part indices.

class BSDistantTreeShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty

    Bethesda-specific node.

class BSEffectShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Skyrim non-PP shader model, used primarily for transparency effects, often as decal.

    property emissive_color
        Emissive color

    property emissive_multiple
        Multiplier for Emissive Color (RGB part)

    property falloff_start_angle
        At this cosine of angle falloff will be equal to Falloff Start Opacity

    property falloff_start_opacity
        Alpha falloff multiplier at start angle

    property falloff_stop_angle
        At this cosine of angle falloff will be equal to Falloff Stop Opacity

    property falloff_stop_opacity
        Alpha falloff multiplier at end angle

```

**property greyscale\_texture**

Points to an external texture, used as palette for SLSF1\_Greyscale\_To\_PaletteColor/SLSF1\_Greyscale\_To\_PaletteAlpha

**property shader\_flags\_1**

**property shader\_flags\_2**

**property soft\_falloff\_depth**

**property source\_texture**

points to an external texture.

**property texture\_clamp\_mode**

How to handle texture borders.

**property uv\_offset**

Offset UVs

**property uv\_scale**

Offset UV Scale to repeat tiling textures

**class BSEffectShaderPropertyColorController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiFloatInterpController`

This controller is used to animate colors in BSEffectShaderProperty.

**property type\_of\_controlled\_color**

Which color in BSEffectShaderProperty to animate:

**class BSEffectShaderPropertyFloatController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiFloatInterpController`

This controller is used to animate float variables in BSEffectShaderProperty.

**property type\_of\_controlled\_variable**

Which float variable in BSEffectShaderProperty to animate:

**class BSFadeNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Bethesda-specific fade node.

**class BSFrustumFOVController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

Bethesda-specific node.

**property interpolator**

Frustum field of view animation interpolater and data.

**class BSFurnitureMarker** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Unknown. Marks furniture sitting positions?

**property num\_positions**

Number of positions.

**property positions**

Unknown. Probably has something to do with the furniture positions?

**class BSFurnitureMarkerNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.BSFurnitureMarker`

Furniture Marker for actors

**class BSInvMarker** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Orientation marker for Skyrim's inventory view. How to show the nif in the player's inventory. Typically attached to the root node of the nif tree. If not present, then Skyrim will still show the nif in inventory, using the default values. Name should be 'INV' (without the quotes). For rotations, a short of "4712" appears as "4.712" but "959" appears as "0.959" meshes/weapons/daedric/daedricbows/skinned.nif

**property rotation\_x**

**property rotation\_y**

**property rotation\_z**

**property zoom**

Zoom factor.

**class BSKeyframeController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiKeyframeController`

An extended keyframe controller.

**property data\_2**

A link to more keyframe data.

**class BSLODTriShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTriBasedGeom`

A variation on NiTriShape, for visibility control over vertex groups.

**property level\_0\_size**

Unknown

**property level\_1\_size**

Unknown

**property level\_2\_size**

Unknown

**class BSLagBoneController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

A controller that trails a bone behind an actor.

**property linear\_rotation**

How the bone lags rotation

**property linear\_velocity**

How long it takes to rotate about an actor back to rest position.

**property maximum\_distance**

How far bone will trail an actor.

**class BSLeafAnimNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Unknown, related to trees.

**class BSLightingShaderProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Skyrim PP shader for assigning material/shader/texture.

- property alpha**  
The material's opacity (1=non-transparent).
- property emissive\_color**  
Glow color and alpha
- property emissive\_multiple**  
Multiplied emissive colors
- property environment\_map\_scale**  
Scales the intensity of the environment/cube map. (0-1)
- property eye\_cubemap\_scale**  
Eye cubemap scale
- property glossiness**  
The material's specular power, or glossiness (0-999).
- property hair\_tint\_color**  
Tints the base texture. Overridden by game settings.
- property left\_eye\_reflection\_center**  
Offset to set center for left eye cubemap
- property lighting\_effect\_1**  
Controls strength for envmap/backlight/rim/softlight lighting effect?
- property lighting\_effect\_2**  
Controls strength for envmap/backlight/rim/softlight lighting effect?
- property max\_passes**  
Max Passes
- property parallax\_envmap\_strength**  
How strong the environment/cube map is. (0-??)
- property parallax\_inner\_layer\_texture\_scale**  
Scales the inner parallax layer texture.
- property parallax\_inner\_layer\_thickness**  
How far from the surface the inner layer appears to be.
- property parallax\_refraction\_scale**  
Depth of inner parallax layer effect.
- property refraction\_strength**  
The amount of distortion. **Not based on physically accurate refractive index** (0=none) (0-1)
- property right\_eye\_reflection\_center**  
Offset to set center for right eye cubemap
- property scale**  
Scale
- property shader\_flags\_1**  
Skyrim Shader Flags for setting render/shader options.
- property shader\_flags\_2**  
Skyrim Shader Flags for setting render/shader options.
- property skin\_tint\_color**  
Tints the base texture. Overridden by game settings.

**property sparkle\_parameters**  
Unknown/unused? CK lists “snow material” when used.

**property specular\_color**  
Adds a colored highlight.

**property specular\_strength**  
Brightness of specular highlight. (0=not visible) (0-999)

**property texture\_clamp\_mode**  
How to handle texture borders.

**property texture\_set**  
Texture Set, can have override in an esm/esp

**property uv\_offset**  
Offset UVs

**property uv\_scale**  
Offset UV Scale to repeat tiling textures, see above.

**class BSLightingShaderPropertyColorController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiFloatInterpController`  
This controller is used to animate colors in BSLightingShaderProperty.

**property type\_of\_controlled\_color**  
Which color in BSLightingShaderProperty to animate:

**class BSLightingShaderPropertyFloatController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiFloatInterpController`  
This controller is used to animate float variables in BSLightingShaderProperty.

**property type\_of\_controlled\_variable**  
Which float variable in BSLightingShaderProperty to animate:

**class BSLightingShaderPropertyShaderType** (*\*\*kwargs*)  
Bases: `pyffi.object_models.xml.enum.EnumBase`  
Values for configuring the shader type in a BSLightingShaderProperty

**Default = 0**

**Heightmap = 3**

**WorldMap1 = 9**

**WorldMap2 = 13**

**WorldMap3 = 15**

**WorldMap4 = 18**

**class BSMasterParticleSystem** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
Bethesda-Specific node.

**property max\_emitter\_objects**  
Unknown

**property num\_particle\_systems**  
Unknown

**property particle\_systems**  
Unknown

**class BSMaterialEmittanceMultController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiFloatInterpController`  
Bethesda-Specific node.

**class BSMultiBound** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Bethesda-specific node.

**property data**  
Unknown.

**class BSMultiBoundAABB** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.BSMultiBoundData`  
Bethesda-specific node.

**property extent**  
Extent of the AABB in all directions

**property position**  
Position of the AABB's center

**class BSMultiBoundData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Abstract base type for bounding data.

**class BSMultiBoundNode** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
Bethesda-specific node.

**property multi\_bound**  
Unknown.

**property unknown\_int**  
Unknown

**class BSMultiBoundOBB** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.BSMultiBoundData`  
Oriented bounding box.

**property center**  
Center of the box.

**property rotation**  
Rotation of the bounding box.

**property size**  
Size of the box along each axis.

**class BSMultiBoundSphere** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.BSMultiBoundData`  
Bethesda-specific node.

**property radius**  
Radius

```

property unknown_int_1
    Unknown Flag

property unknown_int_2
    Unknown Flag

property unknown_int_3
    Unknown Flag

class BSNiAlphaPropertyTestRefController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAlphaController
    Unkown

class BSOrderedNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node.

property alpha_sort_bound
    Unknown

property is_static_bound
    Unknown

class BSPSysArrayEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysVolumeEmitter
    Particle emitter that uses a node, its children and subchildren to emit from. Emission will be evenly spread along points from nodes leading to their direct parents/children only.

class BSPSysHavokUpdateModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

property modifier
    Unknown

property nodes
    Group of target NiNodes?

property num_nodes
    Unknown

class BSPSysInheritVelocityModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

property unknown_float_1
    Unknown

property unknown_float_2
    Unknown

property unknown_float_3
    Unknown

property unknown_int_1
    Unknown

class BSPSysLODModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

property unknown_float_1
    Unknown

```

```
property unknown_float_2
    Unknown

property unknown_float_3
    Unknown

property unknown_float_4
    Unknown

class BSPSysMultiTargetEmitterCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
    Particle system (multi?) emitter controller.

    property data
        This controller's data

    property unknown_int_1
        Unknown

    property unknown_short_1
        Unknown

    property visibility_interpolator
        Links to a bool interpolator. Controls emitter's visibility status?

class BSPSysRecycleBoundModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    property unknown_float_1
        Unknown

    property unknown_float_2
        Unknown

    property unknown_float_3
        Unknown

    property unknown_float_4
        Unknown

    property unknown_float_5
        Unknown

    property unknown_float_6
        Unknown

    property unknown_int_1
        Unknown

class BSPSysScaleModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    property floats
        Unknown

    property num_floats

class BSPSysSimpleColorModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Bethesda-Specific Particle node.

    property color_1_end_percent
        Unknown
```

**property color\_1\_start\_percent**  
Unknown

**property color\_2\_end\_percent**  
Unknown

**property color\_2\_start\_percent**  
Unknown

**property colors**  
Colors

**property fade\_in\_percent**  
Unknown

**property fade\_out\_percent**  
Unknown

**class BSPSysStripUpdateModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Bethesda-Specific (mesh?) Particle System Modifier.

**property update\_delta\_time**  
Unknown

**class BSPSysSubTexModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Similar to a Flip Controller, this handles particle texture animation on a single texture atlas

**property end\_frame**  
Ending frame/position on atlas

**property frame\_count**  
Unknown

**property frame\_count\_fudge**  
Unknown

**property loop\_start\_frame**  
Frame to start looping

**property loop\_start\_frame\_fudge**

**property start\_frame**  
Starting frame/position on atlas

**property start\_frame\_fudge**  
Random chance to start on a different frame?

**class BSPackedAdditionalDataBlock** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property atom\_sizes**  
The sum of all of these equal num total bytes per element, so this probably describes how each data element breaks down into smaller chunks (i.e. atoms).

**property block\_offsets**  
Block offsets in the data? Usually equal to zero.

**property data**  
Unknown

**property has\_data**

Has data

**property num\_atoms**

Number of atoms?

**property num\_blocks**

Number of blocks? Usually equal to one.

**property num\_total\_bytes**

Total number of bytes (over all channels and all elements, equals num total bytes per element times num vertices).

**property num\_total\_bytes\_per\_element**

Unsure, but this seems to correspond again to the number of total bytes per element.

**property unknown\_int\_1**

**class BSPackedAdditionalGeometryData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.AbstractAdditionalGeometryData`

**property block\_infos**

Number of additional data blocks

**property blocks**

Number of additional data blocks

**property num\_block\_infos**

Information about additional data blocks

**property num\_blocks**

Number of additional data blocks. Usually there is exactly one block.

**property num\_vertices**

**class BSParentVelocityModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that adds a blend of object space translation and rotation to particles born in world space.

**property damping**

Amount of blending?

**class BSPartFlag** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.bit_struct.BitStructBase`

**property pf\_editor\_visible**

**property pf\_start\_net\_boneset**

**property reserved\_bits\_1**

**class BSProceduralLightningController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiFloatInterpController`

Skyrim, Paired with dummy TriShapes, this controller generates lightning shapes for special effects. First interpolator controls Generation.

**property byte\_1**

Unknown

**property byte\_2**

Unknown

**property byte\_3**  
Unknown

**property distance\_weight**  
How far lightning will stretch to.

**property float\_2**  
Unknown

**property float\_5**  
Unknown

**property fork**  
Influences forking behavior

**property interpolator\_10**  
Unknown, unsure if this is actually another interpolator link.

**property interpolator\_2\_mutation**  
References interpolator for Mutation of strips

**property interpolator\_3**  
Unknown

**property interpolator\_4**  
Unknown

**property interpolator\_5**  
Unknown

**property interpolator\_6**  
Unknown

**property interpolator\_7**  
Unknown

**property interpolator\_8**  
Unknown

**property interpolator\_9\_arc\_offset**  
References interpolator for Amplitutde control. 0=straight, 50=wide

**property strip\_width**  
How wide the bolt will be

**property unknown\_short\_1**  
Unknown

**property unknown\_short\_2**  
Unknown

**property unknown\_short\_3**  
Unknown

**class BSRefractionFirePeriodController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiTimeController`  
Bethesda-specific node.

**property interpolator**  
Link to Interpolator.

```
class BSRefractionStrengthController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    Bethesda-specific node.

class BSRotAccumTransfInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTransformInterpolator

class BSSegment (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Bethesda-specific node.

    property flags
        Geometry present in the segment

    property internal_index
        Index multiplied by 1536 (0x0600)

    property unknown_byte_1
        Unknown

class BSSegmentFlags (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    property bsseg_water

    property reserved_bits_0

class BSSegmentedTriShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriShape

    Bethesda-specific node.

    property num_segments
        Number of segments in the square grid

    property segment
        Configuration of each segment

class BSShaderFlags (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    property sf_alpha_texture

    property sf_decals_single_pass

    property sf_dynamic_alpha

    property sf_dynamic_decals_single_pass

    property sf_empty

    property sf_environment_mapping

    property sf_external_emittance

    property sf_eye_environment_mapping

    property sf_face_gen

    property sf_fire_refraction

    property sf_hair

    property sf_localmap_hide_secret

    property sf_low_detail
```

```
property sf_multiple_textures
property sf_non_projective_shadows
property sf_parallax_occulsion
property sf_parallax_shader_index_15
property sf_refraction
property sf_remappable_textures
property sf_shadow_frustum
property sf_shadow_map
property sf_single_pass
property sf_skinned
property sf_specular
property sf_tree_billboard
property sf_unknown_1
property sf_unknown_2
property sf_unknown_3
property sf_unknown_4
property sf_vertex_alpha
property sf_window_environment_mapping
property sf_z_buffer_test
class BSShaderFlags2 (template=None, argument=None, parent=None)
  Bases: pyffi.object_models.xml.bit_struct.BitStructBase
property sf_2_1_st_light_is_point_light
property sf_2_2_nd_light
property sf_2_3_rd_light
property sf_2_alpha_decal
property sf_2_billboard_and_envmap_light_fade
property sf_2_envmap_light_fade
property sf_2_fit_slope
property sf_2_lod_building
property sf_2_lod_landscape
property sf_2_no_fade
property sf_2_no_lod_land_blend
property sf_2_no_transparecny_multisampling
property sf_2_premult_alpha
property sf_2_refraction_tint
property sf_2_show_in_local_map
```

```
property sf_2_skip_normal_maps
property sf_2_uniform_scale
property sf_2_unknown_1
property sf_2_unknown_10
property sf_2_unknown_2
property sf_2_unknown_3
property sf_2_unknown_4
property sf_2_unknown_5
property sf_2_unknown_6
property sf_2_unknown_7
property sf_2_unknown_8
property sf_2_unknown_9
property sf_2_vats_selection
property sf_2_vertex_colors
property sf_2_vertex_lighting
property sf_2_wireframe
property sf_2_z_buffer_write
```

```
class BSShaderLightingProperty (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.BSShaderProperty
```

```
    Bethesda-specific property.
```

```
property texture_clamp_mode
```

```
    How to handle texture borders.
```

```
class BSShaderNoLightingProperty (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.BSShaderLightingProperty
```

```
    Bethesda-specific property.
```

```
property falloff_start_angle
```

```
    At this cosine of angle falloff will be equal to Falloff Start Opacity
```

```
property falloff_start_opacity
```

```
    Alpha falloff multiplier at start angle
```

```
property falloff_stop_angle
```

```
    At this cosine of angle falloff will be equal to Falloff Stop Opacity
```

```
property falloff_stop_opacity
```

```
    Alpha falloff multiplier at end angle
```

```
property file_name
```

```
    The texture glow map.
```

```
class BSShaderPPLightingProperty (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.BSShaderLightingProperty
```

```
    Bethesda-specific Shade node.
```

```

property emissive_color
    Glow color and alpha

property refraction_fire_period
    Rate of texture movement for refraction shader.

property refraction_strength
    The amount of distortion. Not based on physically accurate refractive index (0=none) (0-1)

property texture_set
    Texture Set

property unknown_float_4
    Unknown

property unknown_float_5
    Unknown

class BSShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Bethesda-specific Property node

property environment_map_scale
    Scales the intensity of the environment/cube map.

property shader_flags
    Shader Property Flags

property shader_flags_2
    Shader Property Flags 2

property shader_type
    Unknown (Set to 0x21 for NoLighting, 0x11 for Water)

property smooth
    smooth yes
    Type Unknown.0
    Type smooth no1

class BSShaderTextureSet (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Bethesda-specific Texture Set.

property num_textures
    Number of Textures

property textures
    Normal/Gloss2:      Glow(SLSF2_Glow_Map)/Skin/Hair/Rim   light(SLSF2_Rim_Lighting)3:
    Height/Parallax4:  Environment5: Environment Mask6: Subsurface for Multilayer Parallax7:
    Back Lighting Map (SLSF2_Back_Lighting)
    Type Textures.0
    Type Diffuse1

class BSShaderType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The type of animation interpolation (blending) that will be used on the associated key frames.

SHADER_DEFAULT = 1

SHADER_LIGHTING30 = 29

```

**SHADER\_NOLIGHTING = 33**

**SHADER\_SKIN = 14**

**SHADER\_SKY = 10**

**SHADER\_TALL\_GRASS = 0**

**SHADER\_TILE = 32**

**SHADER\_WATER = 17**

**class BSSkyShaderProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Skyrim Sky shader block.

**property shader\_flags\_1**

**property shader\_flags\_2**

**property sky\_object\_type**

Sky Object Type

**property source\_texture**

points to an external texture.

**property uv\_offset**

Offset UVs. Seems to be unused, but it fits with the other Skyrim shader properties.

**property uv\_scale**

Offset UV Scale to repeat tiling textures, see above.

**class BSStripPSysData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysData`

Bethesda-Specific (mesh?) Particle System Data.

**property unknown\_byte\_6**

Unknown

**property unknown\_float\_8**

Unknown

**property unknown\_int\_7**

Unknown

**property unknown\_short\_5**

Unknown

**class BSStripParticleSystem** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleSystem`

Bethesda-Specific (mesh?) Particle System.

**class BSTreadTransfInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpolator`

Bethesda-specific node.

**property data**

Unknown float data.

**property num\_tread\_transforms**

Unknown.

**property tread\_transforms**  
Unknown.

**class BSTreadTransform** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`  
Bethesda-specific node.

**property name**  
Name of affected node?

**property transform\_1**  
Transform data.

**property transform\_2**  
Transform data.

**class BSTreadTransformData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`  
Bethesda-specific node.

**property rotation**  
Rotation.

**property scale**  
Scale (usually float\_min).

**property translation**  
Translation.

**class BSTreeNode** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
Node for handling Trees, Switches branch configurations for variation?

**property bones**  
Unknown

**property bones\_1**  
Unknown

**property num\_bones\_1**  
Unknown

**property num\_bones\_2**  
Unknown

**class BSValueNode** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
Bethesda-Specific node. Found on fxFire effects

**property unknown\_byte**  
Unknown

**property value**  
Value

**class BSWArray** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Bethesda-specific node.

```
property items
    Unknown

property num_items
    Unknown

class BSWaterShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Skyrim water shader property, different from “WaterShaderProperty” seen in Fallout.

property shader_flags_1

property shader_flags_2

property unknown_short_3
    Unknown, flag?

property uv_offset
    Offset UVs. Seems to be unused, but it fits with the other Skyrim shader properties.

property uv_scale
    Offset UV Scale to repeat tiling textures, see above.

property water_direction
    A bitflag, only the first/second bit controls water flow positive or negative along UVs.

property water_shader_flags
    Defines attributes for the water shader (will use SkyrimWaterShaderFlags)

class BSWindModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Particle Modifier that uses the wind value from the gamedata to alter the path of particles.

property strength
    The amount of force wind will have on particles.

class BSXFlags (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiIntegerExtraData

    Controls animation and collision. Integer holds flags:Bit 0 : enable havok, bAnimated(Skyrim)Bit 1 :
    enable collision, bHavok(Skyrim)Bit 2 : is skeleton nif?, bRagdoll(Skyrim)Bit 3 : enable animation,
    bComplex(Skyrim)Bit 4 : FlameNodes present, bAddon(Skyrim)Bit 5 : EditorMarkers presentBit 6 :
    bDynamic(Skyrim)Bit 7 : bArticulated(Skyrim)Bit 8 : bIKTarget(Skyrim)Bit 9 : Unknown(Skyrim)

class BallAndSocketDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

property unknown_4_bytes
    Unknown

property unknown_floats_1
    Unknown

property unknown_floats_2
    Unknown

property unknown_int_1
    Unknown

class BillboardMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
```

Determines the way the billboard will react to the camera. Billboard mode is stored in lowest 3 bits although Oblivion vanilla nifs uses values higher than 7.

**ALWAYS\_FACE\_CAMERA** = 0

**ALWAYS\_FACE\_CENTER** = 3

**BSROTATE\_ABOUT\_UP** = 5

**RIGID\_FACE\_CAMERA** = 2

**RIGID\_FACE\_CENTER** = 4

**ROTATE\_ABOUT\_UP** = 1

**ROTATE\_ABOUT\_UP2** = 9

#### **BlockTypeIndex**

alias of `pyffi.object_models.common.UShort`

**class BodyPartList** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Body part list for DismemberSkinInstance

**property body\_part**

Body Part Index

**property part\_flag**

Flags related to the Body Partition

**class BoneLOD** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Stores Bone Level of Detail info in a BSBoneLODExtraData

**property bone\_name**

The bones name

**property distance**

Distance to cull?

**class BoundVolumeType** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

**BASE\_BV** = 4294967295

**BOX\_BV** = 1

**CAPSULE\_BV** = 2

**HALFSPACE\_BV** = 5

**SPHERE\_BV** = 0

**UNION\_BV** = 4

**class BoundingBox** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Bounding box.

**property radius**

Radius, per direction.

**property rotation**

Rotation matrix.

**property translation**  
Translation vector.

**property unknown\_int**  
Usually 1.

**class BoundingBoxVolume** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property box**  
Box

**property capsule**  
Capsule

**property collision\_type**  
Type of collision data.

**property half\_space**  
Half Space

**property sphere**  
Sphere

**property union**  
Union

**class BoxBV** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Box Bounding Volume

**property axis**  
Axis

**property center**  
Center

**property extent**  
Extent

**class ByteArray** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

Array (list) of bytes. Implemented as basic type to speed up reading and also to prevent data to be dumped by `__str__`.

**get\_hash** (*data=None*)  
Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)  
Returns size of the object in bytes.

**get\_value** ()  
Return object value.

**read** (*stream, data*)  
Read object from file.

**set\_value** (*value*)  
Set object value.

**write** (*stream, data*)  
Write object to file.

---

```

class ByteColor3 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color without alpha (red, green, blue).

    property b
        Blue color component.

    property g
        Green color component.

    property r
        Red color component.

class ByteColor4 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color with alpha (red, green, blue, alpha).

    property a
        Alpha color component.

    property b
        Blue color component.

    property g
        Green color component.

    property r
        Red color component.

class ByteMatrix (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Matrix of bytes. Implemented as basic type to speed up reading and to prevent data being dumped by
    __str__.

    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.

    get_size (data=None)
        Returns size of the object in bytes.

    get_value ()
        Return object value.

    read (stream, data)
        Read object from file.

    set_value (value)
        Set object value.

    write (stream, data)
        Write object to file.

class CStreamableAssetData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property root

    property unknown_bytes

class CapsuleBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

```

Capsule Bounding Volume

**property center**  
Center

**property origin**  
Origin

**property unknown\_float\_1**  
Unknown.

**property unknown\_float\_2**  
Unknown.

**class ChannelConvention** (\*\*kwargs)  
Bases: pyffi.object\_models.xml.enum.EnumBase

**CC\_COMPRESSED** = 4

**CC\_EMPTY** = 5

**CC\_FIXED** = 0

**CC\_INDEX** = 3

**class ChannelData** (template=None, argument=None, parent=None)  
Bases: pyffi.object\_models.xml.struct\_.StructBase

Channel data

**property bits\_per\_channel**  
Bits per channel

**property convention**  
Data Storage Convention

**property type**  
Channel Type

**property unknown\_byte\_1**  
Unknown

**class ChannelType** (\*\*kwargs)  
Bases: pyffi.object\_models.xml.enum.EnumBase

**CHNL\_ALPHA** = 3

**CHNL\_BLUE** = 2

**CHNL\_COMPRESSED** = 4

**CHNL\_EMPTY** = 19

**CHNL\_GREEN** = 1

**CHNL\_INDEX** = 16

**CHNL\_RED** = 0

**class CloningBehavior** (\*\*kwargs)  
Bases: pyffi.object\_models.xml.enum.EnumBase

Sets how objects are to be cloned.

**CLONING\_BLANK\_COPY** = 2

**CLONING\_COPY** = 1

```

    CLONING_SHARE = 0

class CollisionMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    CM_NOTEST = 3

    CM_USE_ABV = 2

    CM_USE_NIBOUND = 4

    CM_USE_OBB = 0

    CM_USE_TRI = 1

class Color3 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color without alpha (red, green, blue).

    property b
        Blue color component.

    property g
        Green color component.

    property r
        Red color component.

class Color4 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color with alpha (red, green, blue, alpha).

    property a
        Alpha.

    property b
        Blue component.

    property g
        Green component.

    property r
        Red component.

class ComponentFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The data format of components.

    F_FLOAT16_1 = 66097

    F_FLOAT16_2 = 131634

    F_FLOAT16_3 = 197171

    F_FLOAT16_4 = 262708

    F_FLOAT32_1 = 66613

    F_FLOAT32_2 = 132150

    F_FLOAT32_3 = 197687

    F_FLOAT32_4 = 263224

    F_INT16_1 = 66065

```

```
F_INT16_2 = 131602
F_INT16_3 = 197139
F_INT16_4 = 262676
F_INT32_1 = 66593
F_INT32_2 = 132130
F_INT32_3 = 197667
F_INT32_4 = 263204
F_INT8_1 = 65793
F_INT8_2 = 131330
F_INT8_3 = 196867
F_INT8_4 = 262404
F_NORMINT16_1 = 66073
F_NORMINT16_2 = 131610
F_NORMINT16_3 = 197147
F_NORMINT16_4 = 262684
F_NORMINT32_1 = 66601
F_NORMINT32_2 = 132138
F_NORMINT32_3 = 197675
F_NORMINT32_4 = 263212
F_NORMINT8_1 = 65801
F_NORMINT8_2 = 131338
F_NORMINT8_3 = 196875
F_NORMINT8_4 = 262412
F_NORMINT_10_10_10_2 = 66621
F_NORMINT_10_10_10_L1 = 66618
F_NORMINT_11_11_10 = 66619
F_NORMUINT16_1 = 66077
F_NORMUINT16_2 = 131614
F_NORMUINT16_3 = 197151
F_NORMUINT16_4 = 262688
F_NORMUINT32_1 = 66605
F_NORMUINT32_2 = 132142
F_NORMUINT32_3 = 197679
F_NORMUINT32_4 = 263216
F_NORMUINT8_1 = 65805
F_NORMUINT8_2 = 131342
```

```

F_NORMUINT8_3 = 196879
F_NORMUINT8_4 = 262416
F_NORMUINT8_4_BGRA = 262460
F_UINT16_1 = 66069
F_UINT16_2 = 131606
F_UINT16_3 = 197143
F_UINT16_4 = 262680
F_UINT32_1 = 66597
F_UINT32_2 = 132134
F_UINT32_3 = 197671
F_UINT32_4 = 263208
F_UINT8_1 = 65797
F_UINT8_2 = 131334
F_UINT8_3 = 196871
F_UINT8_4 = 262408
F_UINT_10_10_10_2 = 66622
F_UINT_10_10_10_L1 = 66617
F_UNKNOWN = 0

```

```
class ConsistencyType (**kwargs)
```

```
Bases: pyffi.object_models.xml.enum.EnumBase
```

Used by NiGeometryData to control the volatility of the mesh. While they appear to be flags they behave as an enum.

```
CT_MUTABLE = 0
```

```
CT_STATIC = 16384
```

```
CT_VOLATILE = 32768
```

```
class ControllerLink (template=None, argument=None, parent=None)
```

```
Bases: pyffi.formats.nif._ControllerLink, object
```

```

>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.node_name_offset
-1
>>> link.set_node_name("Bip01")
>>> link.node_name_offset
0
>>> link.get_node_name()
b'Bip01'
>>> link.node_name
b'Bip01'
>>> link.set_node_name("Bip01 Tail")
>>> link.node_name_offset
6
>>> link.get_node_name()

```

(continues on next page)

(continued from previous page)

```
b'Bip01 Tail'
>>> link.node_name
b'Bip01 Tail'
```

**get\_controller\_type()**

**get\_node\_name()**

Return the node name.

```
>>> # a doctest
>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.string_palette = NifFormat.NiStringPalette()
>>> palette = link.string_palette.palette
>>> link.node_name_offset = palette.add_string("Bip01")
>>> link.get_node_name()
b'Bip01'
```

```
>>> # another doctest
>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.node_name = "Bip01"
>>> link.get_node_name()
b'Bip01'
```

**get\_property\_type()**

**get\_variable\_1()**

**get\_variable\_2()**

**set\_controller\_type(text)**

**set\_node\_name(text)**

**set\_property\_type(text)**

**set\_variable\_1(text)**

**set\_variable\_2(text)**

**class CoordGenType(\*\*kwargs)**

Bases: `pyffi.object_models.xml.enum.EnumBase`

Determines the way that UV texture coordinates are generated.

**CG\_DIFFUSE\_CUBE\_MAP = 4**

**CG\_SPECULAR\_CUBE\_MAP = 3**

**CG\_SPHERE\_MAP = 2**

**CG\_WORLD\_PARALLEL = 0**

**CG\_WORLD\_PERSPECTIVE = 1**

**class CycleType(\*\*kwargs)**

Bases: `pyffi.object_models.xml.enum.EnumBase`

The animation cycle behavior.

**CYCLE\_CLAMP = 2**

```
CYCLE_LOOP = 0
```

```
CYCLE_REVERSE = 1
```

```
class Data (version=67108866, user_version=0, user_version_2=0)
```

```
Bases: pyffi.object_models.Data
```

A class to contain the actual nif data.

Note that L{header} and L{blocks} are not automatically kept in sync with the rest of the nif data, but they are resynchronized when calling L{write}.

#### Variables

- **version** – The nif version.
- **user\_version** – The nif user version.
- **user\_version\_2** – The nif user version 2.
- **roots** – List of root blocks.
- **header** – The nif header.
- **blocks** – List of blocks.
- **modification** – Neo Steam (“neosteam”) or Ndoors (“ndoors”) or Joymaster Interactive Howling Sword (“jmihs1”) or Laxe Lore (“laxelore”) style nif?

```
class VersionUInt (**kwargs)
```

```
Bases: pyffi.object_models.common.UInt
```

```
get_detail_display()
```

Return an object that can be used to display the instance.

```
set_value (value)
```

Set value to C{value}. Calls C{int(value)} to convert to integer.

**Parameters** **value** (*int*) – The value to assign.

```
get_detail_child_names (edge_filter=(True, True))
```

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

**Returns** Generator for detail tree child names.

**Return type** generator yielding `strs`

```
get_detail_child_nodes (edge_filter=(True, True))
```

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

**Parameters** **edge\_filter** (`EdgeFilter` or `type (None)`) – The edge type to include.

**Returns** Generator for detail tree child nodes.

**Return type** generator yielding `DetailNodes`

```
get_global_child_nodes (edge_filter=(True, True))
```

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

**Returns** Generator for global node children.

```
inspect (stream)
```

Quickly checks whether the stream appears to contain nif data, and read the nif header. Resets stream to original position.

Call this function if you only need to inspect the header of the nif.

**Parameters** `stream(file)` – The file to inspect.

**inspect\_version\_only** (`stream`)

This function checks the version only, and is faster than the usual inspect function (which reads the full header). Sets the L{version} and L{user\_version} instance variables if the stream contains a valid NIF file.

Call this function if you simply wish to check that a file is a NIF file without having to parse even the header.

**Raises** `ValueError` – If the stream does not contain a NIF file.

**Parameters** `stream(file)` – The stream from which to read.

**read** (`stream`)

Read a NIF file. Does not reset stream position.

**Parameters** `stream(file)` – The stream from which to read.

**replace\_global\_node** (`oldbranch, newbranch, edge_filter=(True, True)`)

Replace a particular branch in the graph.

**property user\_version**

**property user\_version\_2**

**property version**

**write** (`stream`)

Write a NIF file. The L{header} and the L{blocks} are recalculated from the tree at L{roots} (e.g. list of block types, number of blocks, list of block types, list of strings, list of block sizes etc.).

**Parameters** `stream(file)` – The stream to which to write.

**class DataStreamAccess** (`template=None, argument=None, parent=None`)

Bases: `pyffi.object_models.xml.bit_struct.BitStructBase`

**property cpu\_read**

**property cpu\_write\_mutable**

**property cpu\_write\_static**

**property cpu\_write\_static\_initialized**

**property cpu\_write\_volatile**

**property gpu\_read**

**property gpu\_write**

**class DataStreamUsage** (`**kwargs`)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Determines how a data stream is used?

**USAGE\_SHADER\_CONSTANT = 2**

**USAGE\_USER = 3**

**USAGE\_VERTEX = 1**

**USAGE\_VERTEX\_INDEX = 0**

**class DeactivatorType** (`**kwargs`)

Bases: `pyffi.object_models.xml.enum.EnumBase`

**DEACTIVATOR\_INVALID = 0**

```

DEACTIVATOR_NEVER = 1
DEACTIVATOR_SPATIAL = 2

class DecalVectorArray (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Array of Vectors for Decal placement in BSDecalPlacementVectorExtraData.

    property normals
        Vector Normals

    property num_vectors
        Number of sets

    property points
        Vector XYZ coords

class DecayType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Determines decay function. Used by NiPSysBombModifier.

    DECAY_EXPONENTIAL = 2
    DECAY_LINEAR = 1
    DECAY_NONE = 0

class DistantLODShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific node.

EPSILON = 0.0001

class EffectShaderControlledColor (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing which color in BSEffectShaderProperty to animate.

class EffectShaderControlledVariable (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing which float variable in BSEffectShaderProperty to animate.

    EmissiveMultiple = 0

class EffectType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The type of information that's store in a texture used by a NiTextureEffect.

    EFFECT_ENVIRONMENT_MAP = 2
    EFFECT_FOG_MAP = 3
    EFFECT_PROJECTED_LIGHT = 0
    EFFECT_PROJECTED_SHADOW = 1

class ElementReference (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property normalize_flag
        Whether or not to normalize the data.

```

```
    property semantic
        The element semantic.

class EmitFrom (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Controls which parts of the mesh that the particles are emitted from.

    EMIT_FROM_EDGE_CENTER = 2
    EMIT_FROM_EDGE_SURFACE = 4
    EMIT_FROM_FACE_CENTER = 1
    EMIT_FROM_FACE_SURFACE = 3
    EMIT_FROM_VERTICES = 0

class EndianType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    ENDIAN_BIG = 0
    ENDIAN_LITTLE = 1

class ExportInfo (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Information about how the file was exported

    property creator
        Could be the name of the creator of the NIF file?

    property export_info_1
        Unknown. Can be something like 'TriStrip Process Script'.

    property export_info_2
        Unknown. Possibly the selected option of the export script. Can be something like 'Default Export
        Script'.

    property unknown
        Probably the number of strings that follow.

class ExtraMeshDataEpicMickey (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
    property unknown_int_4
    property unknown_int_5
    property unknown_int_6

class ExtraMeshDataEpicMickey2 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property end
    property start
    property unknown_shorts
```

```
class ExtraVectorsFlags (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
```

```
    None = 0
```

```
    Tangents_Bitangents = 16
```

```
class FaceDrawMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
```

This enum lists the different face culling options.

```
    DRAW_BOTH = 3
```

```
    DRAW_CCW = 1
```

```
    DRAW_CCW_OR_BOTH = 0
```

```
    DRAW_CW = 2
```

```
class Fallout3HavokMaterial (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
```

A material, used by havok shape objects in Fallout 3. Bit 5: flag for PLATFORM (for values 32-63 subtract 32 to know material number) Bit 6: flag for STAIRS (for values 64-95 subtract 64 to know material number) Bit 5+6: flag for STAIRS+PLATFORM (for values 96-127 subtract 96 to know material number)

```
    MAT_BABY_RATTLE = 30
```

```
    MAT_BARREL = 23
```

```
    MAT_BOTTLE = 24
```

```
    MAT_BOTTLECAP = 14
```

```
    MAT_BROKEN_CONCRETE = 19
```

```
    MAT_CHAIN = 13
```

```
    MAT_CLOTH = 1
```

```
    MAT_DIRT = 2
```

```
    MAT_ELEVATOR = 15
```

```
    MAT_GLASS = 3
```

```
    MAT_GRASS = 4
```

```
    MAT_HEAVY_METAL = 11
```

```
    MAT_HEAVY_STONE = 10
```

```
    MAT_HEAVY_WOOD = 12
```

```
    MAT_HOLLOW_METAL = 16
```

```
    MAT_LUNCHBOX = 29
```

```
    MAT_METAL = 5
```

```
    MAT_ORGANIC = 6
```

```
    MAT_PISTOL = 26
```

```
    MAT_RIFLE = 27
```

```
    MAT_RUBBER BALL = 31
```

```
MAT_SAND = 18
MAT_SHEET_METAL = 17
MAT_SHOPPING_CART = 28
MAT_SKIN = 7
MAT_SODA_CAN = 25
MAT_STONE = 0
MAT_VEHICLE_BODY = 20
MAT_VEHICLE_PART_HOLLOW = 22
MAT_VEHICLE_PART_SOLID = 21
MAT_WATER = 8
MAT_WOOD = 9
```

```
class Fallout3Layer(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Sets mesh color in Fallout 3 GECK. Anything higher than 72 is also null.
```

```
ACOUSTIC_SPACE = 21
ACTORZONE = 22
ADDONARM = 71
ADDONCHEST = 70
ADDONHEAD = 69
ADDONLEG = 72
ANIM_STATIC = 2
AVOIDBOX = 31
BIPED = 29
BODY = 46
CAMERAPICK = 35
CAMERASPHERE = 33
CHAIN = 68
CHARCONTROLLER = 30
CLOUD_TRAP = 16
CLUTTER = 4
COLLISIONBOX = 32
CUSTOMPICK1 = 39
CUSTOMPICK2 = 40
DEBRIS_LARGE = 20
DEBRIS_SMALL = 19
DOORDETECTION = 34
```

DROPPINGPICK = 42  
GASTRAP = 24  
GROUND = 17  
HEAD = 45  
INVISIBLE\_WALL = 27  
ITEMPICK = 36  
LINEOFSIGHT = 37  
L\_CALF = 53  
L\_FOOT = 54  
L\_FORE\_ARM = 50  
L\_HAND = 51  
L\_THIGH = 52  
L\_UPPER\_ARM = 49  
NONCOLLIDABLE = 15  
NULL = 43  
OTHER = 44  
PACK = 67  
PATHPICK = 38  
PONYTAIL = 65  
PORTAL = 18  
PROJECTILE = 6  
PROJECTILEZONE = 23  
PROPS = 10  
QUIVER = 63  
R\_CALF = 59  
R\_FOOT = 60  
R\_FORE\_ARM = 56  
R\_HAND = 57  
R\_THIGH = 58  
R\_UPPER\_ARM = 55  
SHELLCASING = 25  
SHIELD = 62  
SPELL = 7  
SPELLEXPLOSION = 41  
SPINE1 = 47  
SPINE2 = 48

```
    STATIC = 1
    TAIL = 61
    TERRAIN = 13
    TRANSPARENT = 3
    TRANSPARENT_SMALL = 26
    TRANSPARENT_SMALL_ANIM = 28
    TRAP = 14
    TREES = 9
    TRIGGER = 12
    UNIDENTIFIED = 0
    WATER = 11
    WEAPON = 64
    WING = 66

class FieldType (**kwargs)
    Bases: pyffi.object_models.xml.enum, EnumBase
    The force field's type.
    FIELD_POINT = 1
    FIELD_WIND = 0

class FilePath (**kwargs)
    Bases: pyffi.formats.nif.string
    A file path.
    get_hash (data=None)
        Returns a case insensitive hash value.

class FileVersion (**kwargs)
    Bases: pyffi.object_models.common.UInt
    get_detail_display ()
        Return an object that can be used to display the instance.
    read (stream, data)
        Read value from stream.
        Parameters stream (file) – The stream to read from.
    set_value ()
        Set value to C{value}. Calls C{int(value)} to convert to integer.
        Parameters value (int) – The value to assign.
    write (stream, data)
        Write value to stream.
        Parameters stream (file) – The stream to write to.

class Flags (**kwargs)
    Bases: pyffi.object_models.common.UShort

class Footer (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Footer, object
```

```

read (stream, data)
    Read structure from stream.

write (stream, data)
    Write structure to stream.

class ForceType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The type of force? May be more valid values.

    FORCE_PLANAR = 0

    FORCE_SPHERICAL = 1

    FORCE_UNKNOWN = 2

class FurnitureEntryPoints (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    property behind

    property front

    property left

    property right

    property up

class FurniturePosition (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Describes a furniture position?

    property animation_type
        Unknown

    property entry_properties
        Unknown/unused in nif?

    property heading
        Similar to Orientation, in float form.

    property offset
        Offset of furniture marker.

    property orientation
        Furniture marker orientation.

    property position_ref_1
        Refers to a furniturermarkerxx.nif file. Always seems to be the same as Position Ref 2.

    property position_ref_2
        Refers to a furniturermarkerxx.nif file. Always seems to be the same as Position Ref 1.

class FxButton (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.FxWidget

    Unknown.

class FxRadioButton (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.FxWidget

    Unknown.

```

**property buttons**

Unknown pointers to other buttons. Maybe other buttons in a group so they can be switch off if this one is switched on?

**property num\_buttons**

Number of unknown links.

**property unknown\_int\_1**

Unknown.

**property unknown\_int\_2**

Unknown.

**property unknown\_int\_3**

Unknown.

**class FxWidget** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Firaxis-specific UI widgets?

**property unknown\_292\_bytes**

Looks like 9 links and some string data.

**property unknown\_3**

Unknown.

**class HairShaderProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.BSShaderProperty`

Bethesda-specific node.

**class HalfSpaceBV** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property center**

Center

**property normal**

Normal

**property unknown\_float\_1**

Unknown.

**class HavokColFilter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

ColFilter property for Havok. It contains Layer, Flags and Part Number

**property flags\_and\_part\_number**

sets the LINK property and controls whether this body is physically linked to others.Bit 6: turns collision off (not used for Layer BIPED).Bit 5: sets the SCALED property.PART NUMBER is stored in bits 0-4. Used only when Layer is set to BIPED.Part Numbers for Oblivion, Fallout 3, Skyrim:0 - OTHER1 - HEAD2 - BODY3 - SPINE14 - SPINE25 - LUPPERARM6 - LFOREARM7 - LHAND8 - LTHIGH9 - LCALF10 - LFOOT11 - RUPPERARM12 - RFOREARM13 - RHAND14 - RTHIGH15 - RCALF16 - RFOOT17 - TAIL18 - SHIELD19 - QUIVER20 - WEAPON21 - PONY-TAIL22 - WING23 - PACK24 - CHAIN25 - ADDONHEAD26 - ADDONCHEST27 - ADDON-ARM28 - ADDONLEG29-31 - NULL

**Type** FLAGS are stored in highest 3 bits

**Type** Bit 7

**property layer**

Physical purpose of collision object? The setting affects object's havok behavior in game.

**property unknown\_short**  
Unknown.

**class HavokMaterial** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`

**property material**  
The material of the shape.

**class Header** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._Header, object`

**has\_block\_type** (*block\_type*)  
Check if header has a particular block type.  
**Raises** **ValueError** – If number of block types is zero (only nif versions 10.0.1.0 and up store block types in header).  
**Parameters** **block\_type** (`L{NiFormat.NiObject}`) – The block type.  
**Returns** True if the header's list of block types has the given block type, or a subclass of it. False otherwise.  
**Return type** bool

**class HeaderString** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.basic.BasicBase`

**get\_detail\_display** ()  
Return an object that can be used to display the instance.

**get\_hash** (*data=None*)  
Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)  
Returns size of the object in bytes.

**read** (*stream, data*)  
Read object from file.

**static version\_string** (*version, modification=None*)  
Transforms version number into a version string.

```
>>> NifFormat.HeaderString.version_string(0x03000300)
'NetImmerse File Format, Version 3.03'
>>> NifFormat.HeaderString.version_string(0x03010000)
'NetImmerse File Format, Version 3.1'
>>> NifFormat.HeaderString.version_string(0x0A000100)
'NetImmerse File Format, Version 10.0.1.0'
>>> NifFormat.HeaderString.version_string(0x0A010000)
'Gamebryo File Format, Version 10.1.0.0'
>>> NifFormat.HeaderString.version_string(0x0A010000,
...                                     modification="neosteam")
'NS'
>>> NifFormat.HeaderString.version_string(0x14020008,
...                                     modification="nddoors")
'NDSNIF....@....@...., Version 20.2.0.8'
>>> NifFormat.HeaderString.version_string(0x14030009,
...                                     modification="jmihs1")
'Joymaster HS1 Object Format - (JMI), Version 20.3.0.9'
```

**write** (*stream, data*)  
Write object to file.

```
class HingeDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    This constraint allows rotation about a specified axis.

    property axle_a
        Axis of rotation.

    property axle_b
        Axle A in second entity coordinate system.

    property perp_2_axle_in_a_1
        Vector in the rotation plane which defines the zero angle.

    property perp_2_axle_in_a_2
        Vector in the rotation plane, orthogonal on the previous one, which defines the positive direction of
        rotation. This is always the vector product of Axle A and Perp2 Axle In A1.

    property perp_2_axle_in_b_1
        Perp2 Axle In A1 in second entity coordinate system.

    property perp_2_axle_in_b_2
        Perp2 Axle In A2 in second entity coordinate system.

    property pivot_a
        Pivot point around which the object will rotate.

    property pivot_b
        Pivot A in second entity coordinate system.

class ImageType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Determines how the raw image data is stored in NiRawImageData.

    RGB = 1

    RGBA = 2

class InertiaMatrix (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._InertiaMatrix, object

    as_list ()
        Return matrix as 3x3 list.

    as_tuple ()
        Return matrix as 3x3 tuple.

    get_copy ()
        Return a copy of the matrix.

    is_identity ()
        Return True if the matrix is close to identity.

    set_identity ()
        Set to identity matrix.

class Key (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A generic key with support for interpolation. Type 1 is normal linear interpolation, type 2 has forward
    and backward tangents, and type 3 has tension, bias and continuity arguments. Note that color4 and byte
    always seem to be of type 1.
```

```

property backward
    The key backward tangent.

property forward
    Key forward tangent.

property tbc
    The key's TBC.

property time
    Time of the key.

property value
    The key value.

class KeyGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Array of vector keys (anything that can be interpolated, except rotations).

property interpolation
    The key type.

property keys
    The keys.

property num_keys
    Number of keys in the array.

class KeyType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The type of animation interpolation (blending) that will be used on the associated key frames.

    CONST_KEY = 5

    LINEAR_KEY = 1

    QUADRATIC_KEY = 2

    TBC_KEY = 3

    XYZ_ROTATION_KEY = 4

class LODRange (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    The distance range where a specific level of detail applies.

property far_extent
    End of Range.

property near_extent
    Beginning of range.

property unknown_ints
    Unknown (0,0,0).

class LightMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing how vertex colors influence lighting.

    LIGHT_MODE_EMISSIVE = 0

    LIGHT_MODE_EMI_AMB_DIF = 1

```

```
class Lighting30ShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderPPLightingProperty

    Bethesda-specific node.

class LightingShaderControlledColor (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which color in BSLightingShaderProperty to animate.

class LightingShaderControlledVariable (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which float variable in BSLightingShaderProperty to animate.

Alpha = 12

Glossiness = 9

class LimitedHingeDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._LimitedHingeDescriptor, object

    update_a_b (transform)
        Update B pivot and axes from A using the given transform.

class LineString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Basic type for strings ending in a newline character (0x0a).
```

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> l = NifFormat.LineString()
>>> f.write('abcdefg\x0a'.encode())
8
>>> f.seek(0)
0
>>> l.read(f)
>>> str(l)
'abcdefg'
>>> f.seek(0)
0
>>> l.set_value('Hi There')
>>> l.write(f)
>>> f.seek(0)
0
>>> m = NifFormat.LineString()
>>> m.read(f)
>>> str(m)
'Hi There'
```

```
get_hash (data=None)
    Returns a hash value (an immutable object) that can be used to identify the object uniquely.

get_size (data=None)
    Returns size of the object in bytes.

get_value ()
    Return object value.

read (stream, data=None)
    Read object from file.
```

```

set_value (value)
    Set object value.

write (stream, data=None)
    Write object to file.

class MTransform (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

property rotation
    Rotation.

property scale
    Scale.

property translation
    Translation.

class MatchGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Group of vertex indices of vertices that match.

property num_vertices
    Number of vertices in this group.

property vertex_indices
    The vertex indices.

class MaterialData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Data stored per-material by NiRenderObject

property material_extra_data
    Extra data associated with the material?

property material_name
    The name of the material.

class Matrix22 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A 2x2 matrix of float values. Stored in OpenGL column-major format.

property m_11
    Member 1,1 (top left)

property m_12
    Member 1,2 (top right)

property m_21
    Member 2,1 (bottom left)

property m_22
    Member 2,2 (bottom right)

class Matrix33 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Matrix33, object

as_list ()
    Return matrix as 3x3 list.

as_tuple ()
    Return matrix as 3x3 tuple.

```

**get\_copy()**  
Return a copy of the matrix.

**get\_determinant()**  
Return determinant.

**get\_inverse()**  
Get inverse (assuming `is_scale_rotation` is true!).

**get\_scale()**  
Gets the scale (assuming `is_scale_rotation` is true!).

**get\_scale\_quat()**  
Decompose matrix into scale and quaternion.

**get\_scale\_rotation()**  
Decompose the matrix into scale and rotation, where scale is a float and rotation is a `C{Matrix33}`.  
Returns a pair (scale, rotation).

**get\_transpose()**  
Get transposed of the matrix.

**is\_identity()**  
Return `True` if the matrix is close to identity.

**is\_rotation()**  
Returns `True` if the matrix is a rotation matrix (a member of `SO(3)`).

**is\_scale\_rotation()**  
Returns true if the matrix decomposes nicely into `scale * rotation`.

**set\_identity()**  
Set to identity matrix.

**set\_scale\_rotation(*scale, rotation*)**  
Compose the matrix as the product of `scale * rotation`.

**sup\_norm()**  
Calculate supremum norm of matrix (maximum absolute value of all entries).

**class Matrix44** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._Matrix44, object`

**as\_list()**  
Return matrix as 4x4 list.

**as\_tuple()**  
Return matrix as 4x4 tuple.

**get\_copy()**  
Create a copy of the matrix.

**get\_inverse(*fast=True*)**  
Calculates inverse (`fast` assumes `is_scale_rotation_translation` is `True`).

**get\_matrix\_33()**  
Returns upper left 3x3 part.

**get\_scale\_quat\_translation()**

**get\_scale\_rotation\_translation()**

**get\_translation()**  
Returns lower left 1x3 part.

```

is_identity()
    Return True if the matrix is close to identity.

is_scale_rotation_translation()

set_identity()
    Set to identity matrix.

set_matrix_33(m)
    Sets upper left 3x3 part.

set_rows(row0, row1, row2, row3)
    Set matrix from rows.

set_scale_rotation_translation(scale, rotation, translation)

set_translation(translation)
    Returns lower left 1x3 part.

sup_norm()
    Calculate supremum norm of matrix (maximum absolute value of all entries).

class MeshData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

property component_semantics
    Describes the semantic of each component.

property is_per_instance
    Sets whether this stream data is per-instance data for use inhardware instancing.

property num_components

property num_submeshes
    The number of submesh-to-region mappings that this data streamhas.

property stream
    Reference to a data stream object which holds the data used bythis reference.

property submesh_to_region_map
    A lookup table that maps submeshes to regions.

class MeshPrimitiveType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Describes the type of primitives stored in a mesh object.

MESH_PRIMITIVE_LINESTRIPS = 2

MESH_PRIMITIVE_POINTS = 4

MESH_PRIMITIVE_QUADS = 3

MESH_PRIMITIVE_TRIANGLES = 0

MESH_PRIMITIVE_TRISTRIPS = 1

class MipMap (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Description of a MipMap within a NiPixelData object.

property height
    Height of the mipmap image.

```

**property offset**

Offset into the pixel data array where this mipmap starts.

**property width**

Width of the mipmap image.

```
class MipMapFormat (**kwargs)
```

Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 32-bit integer, describing how mipmaps are handled in a texture.

```
MIP_FMT_DEFAULT = 2
```

```
MIP_FMT_NO = 0
```

```
MIP_FMT_YES = 1
```

```
class MoppDataBuildType (**kwargs)
```

Bases: `pyffi.object_models.xml.enum.EnumBase`

A byte describing if MOPP Data is organized into chunks (PS3) or not (PC)

```
BUILD_NOT_SET = 2
```

```
BUILT_WITHOUT_CHUNK_SUBDIVISION = 1
```

```
BUILT_WITH_CHUNK_SUBDIVISION = 0
```

```
class Morph (template=None, argument=None, parent=None)
```

Bases: `pyffi.object_models.xml.struct.StructBase`

Geometry morphing data component.

**property frame\_name**

Name of the frame.

**property interpolation**

Unlike most objects, the presense of this value is not conditional on there being keys.

**property keys**

The morph key frames.

**property num\_keys**

The number of morph keys that follow.

**property unknown\_int**

Unknown.

**property vectors**

Morph vectors.

```
class MorphWeight (template=None, argument=None, parent=None)
```

Bases: `pyffi.object_models.xml.struct.StructBase`

**property interpolator**

Interpolator

**property weight**

Weight

```
class MotionQuality (**kwargs)
```

Bases: `pyffi.object_models.xml.enum.EnumBase`

The motion type. Determines quality of motion?

```
MO_QUAL_BULLET = 6
```

```

MO_QUAL_CHARACTER = 8
MO_QUAL_CRITICAL = 5
MO_QUAL_DEBRIS = 3
MO_QUAL_FIXED = 1
MO_QUAL_INVALID = 0
MO_QUAL_KEYFRAMED = 2
MO_QUAL_KEYFRAMED_REPORT = 9
MO_QUAL_MOVING = 4
MO_QUAL_USER = 7

class MotionSystem(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The motion system. 4 (Box) is used for everything movable. 7 (Keyframed) is used on statics and animated
    stuff.

    MO_SYS_BOX = 4
    MO_SYS_BOX_STABILIZED = 5
    MO_SYS_CHARACTER = 9
    MO_SYS_DYNAMIC = 1
    MO_SYS_FIXED = 7
    MO_SYS_INVALID = 0
    MO_SYS_KEYFRAMED = 6
    MO_SYS_SPHERE = 2
    MO_SYS_SPHERE_INERTIA = 3
    MO_SYS_THIN_BOX = 8

class MotorDescriptor(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property unknown_byte_1
        Unknown
    property unknown_float_1
        Unknown
    property unknown_float_2
        Unknown
    property unknown_float_3
        Unknown
    property unknown_float_4
        Unknown
    property unknown_float_5
        Unknown
    property unknown_float_6
        Unknown

```

```
class MultiTextureElement (template=None, argument=None, parent=None)  
    Bases: pyffi.object_models.xml.struct_.StructBase  
  
    property clamp  
        May be texture clamp mode.  
  
    property filter  
        May be texture filter mode.  
  
    property has_image  
        Looks like a memory address, so probably a bool.  
  
    property image  
        Link to the texture image.  
  
    property ps_2_k  
        -75?  
  
    property ps_2_l  
        0?  
  
    property unknown_short_3  
        Unknown. Usually 0 but sometimes 257  
  
    property uv_set  
        This may be the UV set counting from 1 instead of zero.  
  
class Ni3dsAlphaAnimator (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiObject  
  
    Unknown.  
  
    property num_1  
        Unknown.  
  
    property num_2  
        Unknown.  
  
    property parent  
        The parent?  
  
    property unknown_1  
        Unknown.  
  
    property unknown_2  
        Unknown.  
  
class Ni3dsAnimationNode (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiObject  
  
    Unknown. Only found in 2.3 nifs.  
  
    property child  
        Child?  
  
    property count  
        A count.  
  
    property has_data  
        Unknown.  
  
    property name  
        Name of this object.
```

**property unknown\_array**  
Unknown.

**property unknown\_floats\_1**  
Unknown. Matrix?

**property unknown\_floats\_2**  
Unknown.

**property unknown\_short**  
Unknown.

**class Ni3dsColorAnimator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

Unknown!

**property unknown\_1**  
Unknown.

**class Ni3dsMorphShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

Unknown!

**property unknown\_1**  
Unknown.

**class Ni3dsParticleSystem** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

Unknown!

**property unknown\_1**  
Unknown.

**class Ni3dsPathController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

Unknown!

**property unknown\_1**  
Unknown.

**class NiAVObject** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._NiAVObject, object`

```
>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> prop1 = NifFormat.NiProperty()
>>> prop1.name = "hello"
>>> prop2 = NifFormat.NiProperty()
>>> prop2.name = "world"
>>> node.get_properties()
[]
>>> node.set_properties([prop1, prop2])
>>> [prop.name for prop in node.get_properties()]
[b'hello', b'world']
>>> [prop.name for prop in node.properties]
[b'hello', b'world']
>>> node.set_properties([])
>>> node.get_properties()
[]
```

(continues on next page)

(continued from previous page)

```

>>> # now set them the other way around
>>> node.set_properties([prop2, prop1])
>>> [prop.name for prop in node.get_properties()]
[b'world', b'hello']
>>> [prop.name for prop in node.properties]
[b'world', b'hello']
>>> node.remove_property(prop2)
>>> [prop.name for prop in node.properties]
[b'hello']
>>> node.add_property(prop2)
>>> [prop.name for prop in node.properties]
[b'hello', b'world']

```

**add\_property** (*prop*)

Add the given property to the property list.

**Parameters** **prop** (*L{NifFormat.NiProperty}*) – The property block to add.

**apply\_scale** (*scale*)

Apply scale factor on data.

**Parameters** **scale** – The scale factor.

**get\_properties** ()

Return a list of the properties of the block.

**Returns** The list of properties.

**Return type** list of *L{NifFormat.NiProperty}*

**get\_transform** (*relative\_to=None*)

Return scale, rotation, and translation into a single 4x4 matrix, relative to the *C{relative\_to}* block (which should be another *NiAVObject* connecting to this block). If *C{relative\_to}* is *None*, then returns the transform stored in *C{self}*, or equivalently, the target is assumed to be the parent.

**Parameters** **relative\_to** – The block relative to which the transform must be calculated.

If *None*, the local transform is returned.

**remove\_property** (*prop*)

Remove the given property to the property list.

**Parameters** **prop** (*L{NifFormat.NiProperty}*) – The property block to remove.

**set\_properties** (*proplist*)

Set the list of properties from the given list (destroys existing list).

**Parameters** **proplist** (*list of L{NifFormat.NiProperty}*) – The list of property blocks to set.

**set\_transform** (*m*)

Set rotation, translation, and scale, from a 4x4 matrix.

**Parameters** **m** – The matrix to which the transform should be set.

**class NiAVObjectPalette** (*template=None, argument=None, parent=None*)

Bases: *pyffi.formats.nif.NiObject*

Unknown.

**class NiAdditionalGeometryData** (*template=None, argument=None, parent=None*)

Bases: *pyffi.formats.nif.AbstractAdditionalGeometryData*

**property block\_infos**

Number of additional data blocks

**property blocks**

Number of additional data blocks

**property num\_block\_infos**  
Information about additional data blocks

**property num\_blocks**  
Number of additional data blocks

**property num\_vertices**  
Number of vertices

**class NiAlphaController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiFloatInterpController`  
Time controller for transparency.

**property data**  
Alpha controller data index.

**class NiAlphaProperty** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiProperty`  
Transparency. Flags 0x00ED.

**property flags**  
source blend modeBits 5-8 : destination blend modeBit 9 : alpha test enableBit 10-12 : alpha test modeBit 13 : no sorter flag ( disables triangle sorting )blend modes (glBlendFunc):0000  
GL\_ONE0001 GL\_ZERO0010 GL\_SRC\_COLOR0011 GL\_ONE\_MINUS\_SRC\_COLOR0100  
GL\_DST\_COLOR0101 GL\_ONE\_MINUS\_DST\_COLOR0110 GL\_SRC\_ALPHA0111  
GL\_ONE\_MINUS\_SRC\_ALPHA1000 GL\_DST\_ALPHA1001 GL\_ONE\_MINUS\_DST\_ALPHA1010  
GL\_SRC\_ALPHA\_SATURATEtest modes (glAlphaFunc):000 GL\_ALWAYS001 GL\_LESS010  
GL\_EQUAL011 GL\_LEQUAL100 GL\_GREATER101 GL\_NOTEQUAL110 GL\_GEQUAL111  
GL\_NEVER  
**Type** Bit 0  
**Type** alpha blending enableBits 1-4

**property threshold**  
`glAlphaFunc`  
**Type** Threshold for alpha testing (see

**property unknown\_int\_2**  
Unknown

**property unknown\_short\_1**  
Unknown

**class NiAmbientLight** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiLight`  
Ambient light source.

**class NiArkAnimationExtraData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Unknown node.

**property unknown\_bytes**

**property unknown\_ints**

**class NiArkImporterExtraData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Unknown node.

```
property importer_name
    Contains a string like "Gamebryo_1_1" or "4.1.0.12"

property unknown_bytes
property unknown_floats
property unknown_int_1
property unknown_int_2

class NiArkShaderExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown node.

    property unknown_int
    property unknown_string

class NiArkTextureExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown node.

    property num_textures
    property textures
    property unknown_byte
    property unknown_int_2
    property unknown_ints_1

class NiArkViewportInfoExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown node.

    property unknown_bytes

class NiAutoNormalParticles (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticles

    Unknown.

class NiAutoNormalParticlesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticlesData

    Particle system data object (with automatic normals?).

class NiBSAnimationNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-specific extension of Node with animation properties stored in the flags, often 42?

class NiBSBoneLODController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBoneLODController

    A simple LOD controller for bones.

class NiBSParticleArrayController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleSystemController

    A particle system controller, used by BS in conjunction with NiBSParticleNode.
```

```

class NiBSParticleNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Unknown.

class NiBSplineBasisData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Stores the number of control points of a B-spline.

    property num_control_points
        The number of control points of the B-spline (number of frames of animation plus degree of B-spline
        minus one).

class NiBSplineCompFloatInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBSplineFloatInterpolator

    Unknown.

    property base
        Base value when curve not defined.

    property bias
        Bias

    property multiplier
        Multiplier

    property offset
        Starting offset for the data. (USHRT_MAX for no data.)

class NiBSplineCompPoint3Interpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBSplinePoint3Interpolator

    Unknown.

class NiBSplineCompTransformEvaluator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiBSplineCompTransformInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiBSplineCompTransformInterpolator, object

    apply_scale (scale)
        Apply scale factor on data.

    get_rotations ()
        Return an iterator over all rotation keys.

    get_scales ()
        Return an iterator over all scale keys.

    get_translations ()
        Return an iterator over all translation keys.

class NiBSplineData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiBSplineData, object

```

```

>>> # a doctest
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiBSplineData()
>>> block.num_short_control_points = 50

```

(continues on next page)

(continued from previous page)

```

>>> block.short_control_points.update_size()
>>> for i in range(block.num_short_control_points):
...     block.short_control_points[i] = 20 - i
>>> list(block.get_short_data(12, 4, 3))
[(8, 7, 6), (5, 4, 3), (2, 1, 0), (-1, -2, -3)]
>>> offset = block.append_short_data([(1, 2), (4, 3), (13, 14), (8, 2), (33, 33)])
>>> offset
50
>>> list(block.get_short_data(offset, 5, 2))
[(1, 2), (4, 3), (13, 14), (8, 2), (33, 33)]
>>> list(block.get_comp_data(offset, 5, 2, 10.0, 32767.0))
[(11.0, 12.0), (14.0, 13.0), (23.0, 24.0), (18.0, 12.0), (43.0, 43.0)]
>>> block.append_float_data([(1.0, 2.0), (3.0, 4.0), (0.5, 0.25)])
0
>>> list(block.get_float_data(0, 3, 2))
[(1.0, 2.0), (3.0, 4.0), (0.5, 0.25)]
>>> block.append_comp_data([(1, 2), (4, 3)])
(60, 2.5, 1.5)
>>> list(block.get_short_data(60, 2, 2))
[(-32767, -10922), (32767, 10922)]
>>> list(block.get_comp_data(60, 2, 2, 2.5, 1.5))
[(1.0, 2.00...), (4.0, 2.99...)]

```

**append\_comp\_data** (*data*)

Append data as compressed list.

**Parameters data** – A list of elements, where each element is a tuple of integers. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)**Returns** The offset, bias, and multiplier.**append\_float\_data** (*data*)

Append data.

**Parameters data** – A list of elements, where each element is a tuple of floats. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)**Returns** The offset at which the data was appended.**append\_short\_data** (*data*)

Append data.

**Parameters data** – A list of elements, where each element is a tuple of integers. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)**Returns** The offset at which the data was appended.**get\_comp\_data** (*offset, num\_elements, element\_size, bias, multiplier*)Get an iterator to the data, converted to float with extra bias and multiplication factor. If  $C\{x\}$  is the short value, then the returned value is  $C\{\text{bias} + x * \text{multiplier} / 32767.0\}$ .**Parameters**

- **offset** – The offset in the data where to start.
- **num\_elements** – Number of elements to get.
- **element\_size** – Size of a single element.
- **bias** – Value bias.
- **multiplier** – Value multiplier.

**Returns** A list of  $C\{\text{num\_elements}\}$  tuples of size  $C\{\text{element\_size}\}$ .**get\_float\_data** (*offset, num\_elements, element\_size*)

Get an iterator to the data.

**Parameters**

- **offset** – The offset in the data where to start.

- **num\_elements** – Number of elements to get.
- **element\_size** – Size of a single element.

**Returns** A list of C{num\_elements} tuples of size C{element\_size}.

**get\_short\_data** (*offset, num\_elements, element\_size*)

Get an iterator to the data.

**Parameters**

- **offset** – The offset in the data where to start.
- **num\_elements** – Number of elements to get.
- **element\_size** – Size of a single element.

**Returns** A list of C{num\_elements} tuples of size C{element\_size}.

**class NiBSplineFloatInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiBSplineInterpolator`

Unknown.

**class NiBSplineInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiBSplineInterpolator, object`

**get\_times** ()

Return an iterator over all key times.

@todo: When code for calculating the bsplines is ready, this function will return exactly `self.basis_data.num_control_points - 1` time points, and not `self.basis_data.num_control_points` as it is now.

**class NiBSplinePoint3Interpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiBSplineInterpolator`

Unknown.

**property unknown\_floats**

Unknown.

**class NiBSplineTransformInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiBSplineTransformInterpolator, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**get\_rotations** ()

Return an iterator over all rotation keys.

**get\_scales** ()

Return an iterator over all scale keys.

**get\_translations** ()

Return an iterator over all translation keys.

**class NiBezierMesh** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObject`

Unknown

**property bezier\_triangle**

unknown

**property count\_1**

Data count.

**property count\_2**

data count 2.

**property data\_2**  
data count.

**property num\_bezier\_triangles**  
references.

**property points\_1**  
data.

**property points\_2**  
data.

**property unknown\_3**  
Unknown.

**property unknown\_4**  
Unknown.

**property unknown\_5**  
Unknown (illegal link?).

**property unknown\_6**  
unknown

**class NiBezierTriangle4** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Sub data of NiBezierMesh

**property matrix**  
unknown

**property unknown\_1**  
unknown

**property unknown\_2**  
unknown

**property unknown\_3**  
unknown

**property unknown\_4**  
unknown

**property unknown\_5**  
unknown

**property unknown\_6**  
unknown

**property vector\_1**  
unknown

**property vector\_2**  
unknown

**class NiBillboardNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

These nodes will always be rotated to face the camera creating a billboard effect for any attached objects. In pre-10.1.0.0 the Flags field is used for BillboardMode. Bit 0: hiddenBits 1-2: collision modeBit 3: unknown (set in most official meshes)Bits 5-6: billboard modeCollision modes:00 NONE01 USE\_TRIANGLES10 USE\_OBBS11 CONTINUEBillboard modes:00 ALWAYS\_FACE\_CAMERA01 ROTATE\_ABOUT\_UP10 RIGID\_FACE\_CAMERA11 ALWAYS\_FACE\_CENTER

**property billboard\_mode**  
The way the billboard will react to the camera.

**class NiBinaryExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Binary extra data object. Used to store tangents and bitangents in Oblivion.

**property binary\_data**  
The binary data.

**class NiBinaryVoxelData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Voxel data object.

**property num\_unknown\_bytes\_2**  
Unknown.

**property num\_unknown\_vectors**  
Unknown.

**property unknown\_5\_ints**  
Unknown.

**property unknown\_7\_floats**  
Unknown.

**property unknown\_bytes\_1**  
Unknown. Always a multiple of 7.

**property unknown\_bytes\_2**  
Unknown.

**property unknown\_short\_1**  
Unknown.

**property unknown\_short\_2**  
Unknown.

**property unknown\_short\_3**  
Unknown. Is this<sup>^3</sup> the Unknown Bytes 1 size?

**property unknown\_vectors**  
Vectors on the unit sphere.

**class NiBinaryVoxelExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Voxel extra data object.

**property data**  
Link to binary voxel data.

**property unknown\_int**  
Unknown. 0?

**class NiBlendBoolInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiBlendInterpolator`

An interpolator for a bool.

**property bool\_value**  
The interpolated bool?

**class NiBlendFloatInterpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiBlendInterpolator`  
An interpolator for a float.  
**property float\_value**  
The interpolated float?

**class NiBlendInterpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiInterpolator`  
An extended type of interpolater.  
**property unknown\_int**  
Unknown.  
**property unknown\_short**  
Unknown.

**class NiBlendPoint3Interpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiBlendInterpolator`  
Interpolates a point?  
**property point\_value**  
The interpolated point?

**class NiBlendTransformInterpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiBlendInterpolator`  
Unknown.

**class NiBone** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
A NiNode used as a skeleton bone?

**class NiBoneLODController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiTimeController`  
Level of detail controller for bones. Priority is arranged from low to high.  
**property node\_groups**  
A list of node groups (each group a sequence of bones).  
**property num\_node\_groups**  
Number of node groups.  
**property num\_node\_groups\_2**  
Number of node groups.  
**property num\_shape\_groups**  
Number of shape groups.  
**property num\_shape\_groups\_2**  
The size of the second list of shape groups.  
**property shape\_groups\_1**  
List of shape groups.  
**property shape\_groups\_2**  
Group of NiTriShape indices.  
**property unknown\_int\_1**  
Unknown.

**property unknown\_int\_2**  
Unknown.

**property unknown\_int\_3**  
Unknown.

**class NiBoolData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Timed boolean data.

**property data**  
The boolean keys.

**class NiBoolInterpController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiSingleInterpController`  
A controller that interpolates floating point numbers?

**class NiBoolInterpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiKeyBasedInterpolator`  
Unknown.

**property bool\_value**  
Value when posed? At time 0?

**property data**  
Refers to a NiBoolData object.

**class NiBoolTimelineInterpolator** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiBoolInterpolator`  
Unknown.

**class NiBooleanExtraData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Boolean extra data.

**property boolean\_data**  
The boolean extra data value.

**class NiCamera** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiAVObject`  
Camera object.

**property frustum\_bottom**  
Frustum bottom.

**property frustum\_far**  
Frustum far.

**property frustum\_left**  
Frustum left.

**property frustum\_near**  
Frustum near.

**property frustum\_right**  
Frustum right.

**property frustum\_top**  
Frustum top.

**property lod\_adjust**

Level of detail adjust.

**property unknown\_int**

Unknown. Changing value crashes viewer.

**property unknown\_int\_2**

Unknown. Changing value crashes viewer.

**property unknown\_int\_3**

Unknown.

**property unknown\_link**

Unknown.

**property unknown\_short**

Unknown.

**property use\_orthographic\_projection**

Determines whether perspective is used. Orthographic means no perspective.

**property viewport\_bottom**

Viewport bottom.

**property viewport\_left**

Viewport left.

**property viewport\_right**

Viewport right.

**property viewport\_top**

Viewport top.

**class NiClod** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTriBasedGeom`

A shape node that holds continuous level of detail information. Seems to be specific to Freedom Force.

**class NiClodData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTriBasedGeomData`

Holds mesh data for continuous level of detail shapes. Presumably a progressive mesh with triangles specified by edge splits. Seems to be specific to Freedom Force. The structure of this is uncertain and highly experimental at this point. No file with this data can currently be read properly.

**property unknown\_clod\_shorts\_1**

**property unknown\_clod\_shorts\_2**

**property unknown\_clod\_shorts\_3**

**property unknown\_count\_1**

**property unknown\_count\_2**

**property unknown\_count\_3**

**property unknown\_float**

**property unknown\_short**

**property unknown\_shorts**

**class NiClodSkinInstance** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSkinInstance`

A copy of NiSkinInstance for use with NiClod meshes.

**class NiCollisionData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiCollisionObject`

Collision box.

**property bounding\_volume**

Collision data.

**property collision\_mode**

Collision Mode

**property propagation\_mode**

Propagation Mode

**property use\_abv**

Use Alternate Bounding Volume.

**class NiCollisionObject** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

This is the most common collision object found in NIF files. It acts as a real object that is visible and possibly (if the body allows for it) interactive. The node itself is simple, it only has three properties. For this type of collision object, `bhkRigidBody` or `bhkRigidBodyT` is generally used.

**property target**

Index of the AV object referring to this collision object.

**class NiColorData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Color data for material color controller.

**property data**

The color keys.

**class NiColorExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Unknown.

**property data**

RGBA Color?

**class NiControllerManager** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

Unknown. Root of all controllers?

**property controller\_sequences**

Refers to a list of `NiControllerSequence` object.

**property cumulative**

Designates whether animation sequences are cumulative?

**property num\_controller\_sequences**

The number of controller sequence objects.

**property object\_palette**

Refers to a `NiDefaultAVObjectPalette`.

**class NiControllerSequence** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiControllerSequence, object`

**add\_controlled\_block()**  
Create new controlled block, and return it.

```
>>> seq = NifFormat.NiControllerSequence()
>>> seq.num_controlled_blocks
0
>>> ctrlblock = seq.add_controlled_block()
>>> seq.num_controlled_blocks
1
>>> isinstance(ctrlblock, NifFormat.ControllerLink)
True
```

**class NiDataStream** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**property access**

**property cloning\_behavior**

**property component\_formats**

The format of each component in this data stream.

**property data**

**property num\_bytes**

The size in bytes of this data stream.

**property num\_components**

Number of components of the data (matches corresponding field in MeshData).

**property num\_regions**

Number of regions (such as submeshes).

**property regions**

The regions in the mesh. Regions can be used to mark off submeshes which are independent draw calls.

**property streamable**

**property usage**

**class NiDefaultAVObjectPalette** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObjectPalette`

Unknown. Refers to a list of objects. Used by NiControllerManager.

**property num\_objs**

Number of objects.

**property objs**

The objects.

**property unknown\_int**

Unknown.

**class NiDirectionalLight** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiLight`

Directional light source.

**class NiDitherProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Unknown.

**property flags**

Enable dithering

**Type** 1's Bit**class NiDynamicEffect** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiAVObject`

A dynamic effect such as a light or environment map.

**property affected\_node\_list\_pointers**

This is probably the list of affected nodes. For some reason i do not know the max exporter seems to write pointers instead of links. But it doesn't matter because at least in version 4.0.0.2 the list is automagically updated by the engine during the load stage.

**property affected\_nodes**

The list of affected nodes?

**property num\_affected\_node\_list\_pointers**

The number of affected nodes referenced.

**property num\_affected\_nodes**

The number of affected nodes referenced.

**property switch\_state**

Turns effect on and off? Switches list to list of unaffected nodes?

**class NiEnvMappedTriShape** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObjectNET`

Unknown

**property child\_2**

unknown

**property child\_3**

unknown

**property children**

List of child node object indices.

**property num\_children**

The number of child objects.

**property unknown\_1**

unknown (=4 - 5)

**property unknown\_matrix**

unknown

**class NiEnvMappedTriShapeData** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiTriShapeData`

Holds mesh data using a list of singular triangles.

**class NiExtraData** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObject`

A generic extra data object.

**property name**

Name of this object.

**property next\_extra\_data**

Block number of the next extra data object.

```
class NiExtraDataController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    An controller for extra data.

class NiFlipController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Texture flipping controller.

property delta
    Time between two flips. $\text{delta} = (\text{start\_time} - \text{stop\_time}) / \text{sources.num\_indices}$ 

property images
    The image sources

property num_sources
    The number of source objects.

property sources
    The texture sources.

property texture_slot
    Target texture slot (0=base, 4=glow).

property unknown_int_2
    0?

class NiFloatData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Possibly the 1D position along a 3D path.

property data
    The keys.

class NiFloatExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Float extra data.

property float_data
    The float data.

class NiFloatExtraDataController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraDataController
    Unknown.

property controller_data
    Refers to a NiFloatExtraData name.

property num_extra_bytes
    Number of extra bytes.

property unknown_bytes
    Unknown.

property unknown_extra_bytes
    Unknown.

class NiFloatInterpController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    A controller that interpolates floating point numbers?
```

---

```

class NiFloatInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyBasedInterpolator

    Unknown.

    property data
        Float data?

    property float_value
        Value when posed? At time 0?

class NiFloatsExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown.

    property data
        Float data.

    property num_floats
        Number of floats in the next field.

class NiFogProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Describes... fog?

    property flags
        Sets Fog Function to FOG_RANGE_SQ4's bit: Sets Fog Function to FOG_VERTEX_ALPHA if 2's
        and 4's bit are not set, but fog is enabled, Fog function is FOG_Z_LINEAR.
        Type 1's bit
        Type Enables Fog2's bit

    property fog_color
        The color of the fog.

    property fog_depth
        The thickness of the fog? Default is 1.0

class NiFurSpringController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    property bones
        List of all armature bones.

    property bones_2
        List of all armature bones.

    property num_bones
        The number of node bones referenced as influences.

    property num_bones_2
        The number of node bones referenced as influences.

    property unknown_float

    property unknown_float_2

class NiGeomMorpherController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController

    Time controller for geometry morphing.

    property always_update
        Always Update

```

**property data**  
Geometry morphing data index.

**property extra\_flags**  
Unknown.

**property interpolator\_weights**  
Weighted Interpolators?

**property interpolators**  
List of interpolators.

**property num\_interpolators**  
The number of interpolator objects.

**property num\_unknown\_ints**  
A count.

**property unknown\_2**  
Unknown.

**property unknown\_ints**  
Unknown.

**class NiGeometry** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._NiGeometry, object`

```
>>> from pyffi.formats.nif import NifFormat
>>> id44 = NifFormat.Matrix44()
>>> id44.set_identity()
>>> skelroot = NifFormat.NiNode()
>>> skelroot.name = 'skelroot'
>>> skelroot.set_transform(id44)
>>> bone1 = NifFormat.NiNode()
>>> bone1.name = 'bone1'
>>> bone1.set_transform(id44)
>>> bone2 = NifFormat.NiNode()
>>> bone2.name = 'bone2'
>>> bone2.set_transform(id44)
>>> bone21 = NifFormat.NiNode()
>>> bone21.name = 'bone21'
>>> bone21.set_transform(id44)
>>> bone22 = NifFormat.NiNode()
>>> bone22.name = 'bone22'
>>> bone22.set_transform(id44)
>>> bone211 = NifFormat.NiNode()
>>> bone211.name = 'bone211'
>>> bone211.set_transform(id44)
>>> skelroot.add_child(bone1)
>>> bone1.add_child(bone2)
>>> bone2.add_child(bone21)
>>> bone2.add_child(bone22)
>>> bone21.add_child(bone211)
>>> geom = NifFormat.NiTriShape()
>>> geom.name = 'geom'
>>> geom.set_transform(id44)
>>> geomdata = NifFormat.NiTriShapeData()
>>> skininst = NifFormat.NiSkinInstance()
>>> skindata = NifFormat.NiSkinData()
>>> skelroot.add_child(geom)
```

(continues on next page)

(continued from previous page)

```

>>> geom.data = geomdata
>>> geom.skin_instance = skininst
>>> skininst.skeleton_root = skelroot
>>> skininst.data = skindata
>>> skininst.num_bones = 4
>>> skininst.bones.update_size()
>>> skininst.bones[0] = bone1
>>> skininst.bones[1] = bone2
>>> skininst.bones[2] = bone22
>>> skininst.bones[3] = bone211
>>> skindata.num_bones = 4
>>> skindata.bone_list.update_size()
>>> [child.name for child in skelroot.children]
[b'bone1', b'geom']
>>> skindata.set_transform(id44)
>>> for bonedata in skindata.bone_list:
...     bonedata.set_transform(id44)
>>> affectedbones = geom.flatten_skin()
>>> [bone.name for bone in affectedbones]
[b'bone1', b'bone2', b'bone22', b'bone211']
>>> [child.name for child in skelroot.children]
[b'geom', b'bone1', b'bone21', b'bone2', b'bone22', b'bone211']

```

**add\_bone** (*bone, vert\_weights*)

Add bone with given vertex weights. After adding all bones, the geometry skinning information should be set from the current position of the bones using the `L{update_bind_position}` function.

**Parameters**

- **bone** – The bone NiNode block.
- **vert\_weights** – A dictionary mapping each influenced vertex index to a vertex weight.

**flatten\_skin** ()

Reposition all bone blocks and geometry block in the tree to be direct children of the skeleton root.

Returns list of all used bones by the skin.

**get\_skin\_deformation** ()

Returns a list of vertices and normals in their final position after skinning, in geometry space.

**get\_skin\_partition** ()

Return the skin partition block.

**get\_vertex\_weights** ()

Get vertex weights in a convenient format: list bone and weight per vertex.

**is\_skin** ()

Returns True if geometry is skinned.

**send\_bones\_to\_bind\_position** ()

Send all bones to their bind position.

**@deprecated:** Use `L{NifFormat.NiNode.send_bones_to_bind_position}` instead of this function.

**set\_skin\_partition** (*skinpart*)

Set skin partition block.

**update\_bind\_position** ()

Make current position of the bones the bind position for this geometry.

Sets the NiSkinData overall transform to the inverse of the geometry transform relative to the skeleton root, and sets the NiSkinData of each bone to the geometry transform relative to the skeleton root times the inverse of the bone transform relative to the skeleton root.

**class NiGeometryData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiGeometryData`, `object`

```
>>> from pyffi.formats.nif import NifFormat
>>> geomdata = NifFormat.NiGeometryData()
>>> geomdata.num_vertices = 3
>>> geomdata.has_vertices = True
>>> geomdata.has_normals = True
>>> geomdata.has_vertex_colors = True
>>> geomdata.num_uv_sets = 2
>>> geomdata.vertices.update_size()
>>> geomdata.normals.update_size()
>>> geomdata.vertex_colors.update_size()
>>> geomdata.uv_sets.update_size()
>>> geomdata.vertices[0].x = 1
>>> geomdata.vertices[0].y = 2
>>> geomdata.vertices[0].z = 3
>>> geomdata.vertices[1].x = 4
>>> geomdata.vertices[1].y = 5
>>> geomdata.vertices[1].z = 6
>>> geomdata.vertices[2].x = 1.200001
>>> geomdata.vertices[2].y = 3.400001
>>> geomdata.vertices[2].z = 5.600001
>>> geomdata.normals[0].x = 0
>>> geomdata.normals[0].y = 0
>>> geomdata.normals[0].z = 1
>>> geomdata.normals[1].x = 0
>>> geomdata.normals[1].y = 1
>>> geomdata.normals[1].z = 0
>>> geomdata.normals[2].x = 1
>>> geomdata.normals[2].y = 0
>>> geomdata.normals[2].z = 0
>>> geomdata.vertex_colors[1].r = 0.310001
>>> geomdata.vertex_colors[1].g = 0.320001
>>> geomdata.vertex_colors[1].b = 0.330001
>>> geomdata.vertex_colors[1].a = 0.340001
>>> geomdata.uv_sets[0][0].u = 0.990001
>>> geomdata.uv_sets[0][0].v = 0.980001
>>> geomdata.uv_sets[0][2].u = 0.970001
>>> geomdata.uv_sets[0][2].v = 0.960001
>>> geomdata.uv_sets[1][0].v = 0.910001
>>> geomdata.uv_sets[1][0].v = 0.920001
>>> geomdata.uv_sets[1][2].v = 0.930001
>>> geomdata.uv_sets[1][2].v = 0.940001
>>> for h in geomdata.get_vertex_hash_generator():
...     print(h)
(1000, 2000, 3000, 0, 0, 1000, 99000, 98000, 0, 92000, 0, 0, 0, 0)
(4000, 5000, 6000, 0, 1000, 0, 0, 0, 0, 0, 310, 320, 330, 340)
(1200, 3400, 5600, 1000, 0, 0, 97000, 96000, 0, 94000, 0, 0, 0, 0)
```

**apply\_scale** (*scale*)

Apply scale factor on data.

**get\_vertex\_hash\_generator** (*vertexprecision=3, normalprecision=3, uvprecision=5, vcolprecision=3*)

Generator which produces a tuple of integers for each (vertex, normal, uv, vcol), to ease detection of duplicate vertices. The precision parameters denote number of significant digits behind the comma.

Default for uvprecision should really be high because for very large models the uv coordinates can be very close together.

For vertexprecision, 3 seems usually enough (maybe we'll have to increase this at some point).

#### Parameters

- **vertexprecision** (*float*) – Precision to be used for vertices.
- **normalprecision** (*float*) – Precision to be used for normals.
- **uvprecision** (*float*) – Precision to be used for uvs.
- **vcolprecision** (*float*) – Precision to be used for vertex colors.

**Returns** A generator yielding a hash value for each vertex.

**update\_center\_radius** ()

Recalculate center and radius of the data.

**class NiGravity** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

A particle modifier; applies a gravitational field on the particles.

**property direction**

The direction of the applied acceleration.

**property force**

The strength/force of this gravity.

**property position**

check for versions<= 3.1)

**Type** The position of the mass point relative to the particle system. (TODO)

**property type**

The force field's type.

**property unknown\_float\_1**

Unknown.

**class NiImage** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**property file\_name**

The filepath to the texture.

**property image\_data**

Link to the internally stored image data.

**property unknown\_float**

Unknown. Perhaps fImageScale?

**property unknown\_int**

Unknown. Often seems to be 7. Perhaps m\_uiMipLevels?

**property use\_external**

0 if the texture is internal to the NIF file.

**class NiInstancingMeshModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiMeshModifier`

**class NiIntegerExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Extra integer data.

**property integer\_data**

The value of the extra data.

**class NiIntegersExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Integers data.

**property data**

Integers.

**property num\_integers**

Number of integers.

**class NiInterpController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

A controller capable of interpolation?

**class NiInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Interpolator objects - function unknown.

**class NiKeyBasedInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpolator`

Interpolator objects that use keys?

**class NiKeyframeController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSingleInterpController`

A time controller object for animation key frames.

**property data**

Keyframe controller data index.

**class NiKeyframeData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiKeyframeData, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**class NiLODData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Abstract class used for different types of LOD selections.

**class NiLODNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSwitchNode`

Level of detail selector. Links to different levels of detail of the same model, used to switch a geometry at a specified distance.

**property lod\_center**

Point to calculate distance from for switching?

**property lod\_level\_data**

Refers to LOD level information, either distance or screen size based.

**property lod\_levels**

The ranges of distance that each level of detail applies in.

**property num\_lod\_levels**

Number of levels of detail.

---

```

class NiLight (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiDynamicEffect

    Light source.

    property ambient_color
        Ambient color.

    property diffuse_color
        Diffuse color.

    property dimmer
        Dimmer.

    property specular_color
        Specular color.

class NiLightColorController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPoint3InterpController

    Light color animation controller.

class NiLightDimmerController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    Unknown controller.

class NiLightIntensityController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    Unknown controller

class NiLines (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriBasedGeom

    Wireframe geometry.

class NiLinesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiGeometryData

    Wireframe geometry data.

    property lines
        Is vertex connected to other (next?) vertex?

class NiLookAtController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Unknown. Start time is 3.4e+38 and stop time is -3.4e+38.

    property look_at_node
        Link to the node to look at?

    property unknown_1
        Unknown.

class NiLookAtInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpolator

    Unknown.

    property look_at
        Refers to a Node to focus on.

    property rotation
        Rotation.

```

**property scale**  
Scale.

**property target**  
Target node name.

**property translation**  
Translate.

**property unknown\_link\_1**  
Refers to NiPoint3Interpolator.

**property unknown\_link\_2**  
Refers to a NiFloatInterpolator.

**property unknown\_link\_3**  
Refers to a NiFloatInterpolator.

**property unknown\_short**  
Unknown.

**class NiMaterialColorController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._NiMaterialColorController, object`

**get\_target\_color**()  
Get target color (works for all nif versions).

**set\_target\_color**(*target\_color*)  
Set target color (works for all nif versions).

**class NiMaterialProperty** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._NiMaterialProperty, object`

**is\_interchangeable**(*other*)  
Are the two material blocks interchangeable?

**class NiMesh** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiRenderObject`

**property bound**  
The combined bounding volume of all submeshes.

**property datas**

**property instancing\_enabled**  
Sets whether hardware instancing is being used.

**property modifiers**

**property num\_datas**

**property num\_modifiers**

**property num\_submeshes**  
The number of submeshes contained in this mesh.

**property primitive\_type**  
The primitive type of the mesh, such as triangles or lines.

**property unknown\_100**  
Unknown.

**property unknown\_101**  
Unknown.

**property unknown\_102**  
Size of additional data.

**property unknown\_103**

**property unknown\_200**

**property unknown\_201**

**property unknown\_250**

**property unknown\_251**

**property unknown\_300**

**property unknown\_301**

**property unknown\_302**

**property unknown\_303**

**property unknown\_350**

**property unknown\_351**

**property unknown\_400**

**property unknown\_51**

**property unknown\_52**

**property unknown\_53**

**property unknown\_54**

**property unknown\_55**

**property unknown\_56**

**class NiMeshHWInstance** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

**class NiMeshModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`

Base class for mesh modifiers.

**property complete\_points**  
The complete points used by this mesh modifier

**property num\_complete\_points**  
The number of complete points used by this mesh modifier.

**property num\_submit\_points**  
The number of submit points used by this mesh modifier.

**property submit\_points**  
The submit points used by this mesh modifier

**class NiMeshPSysData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysData`

Particle meshes data.

**property num\_unknown\_ints\_1**  
Unknown.

**property unknown\_byte\_3**

Unknown. 0?

**property unknown\_int\_2**

Unknown. Possible vertex count but probably not.

**property unknown\_ints\_1**

Unknown integers

**property unknown\_node**

Unknown NiNode.

**class NiMeshParticleSystem** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleSystem`

Particle system.

**class NiMorphController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpController`

Unknown! Used by Daoc->'healing.nif'.

**class NiMorphData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiMorphData, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**class NiMorphMeshModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiMeshModifier`

Performs linear-weighted blending between a set of target data streams.

**property elements**

Semantics and normalization of the morphing data stream elements.

**property flags**

FLAG\_RELATIVETARGETS = 0x01FLAG\_UPDATENORMALS = 0x02FLAG\_NEEDSUPDATE  
= 0x04FLAG\_ALWAYSUPDATE = 0x08FLAG\_NEEDSCOMPLETION = 0x10FLAG\_SKINNED  
= 0x20FLAG\_SWSKINNED = 0x40

**property num\_elements**

The number of morphing data stream elements.

**property num\_targets**

The number of morph targets.

**class NiMorphWeightsController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpController`

**property interpolators**

**property num\_interpolators**

**property num\_targets**

The number of morph targets.

**property target\_names**

Name of each morph target.

**property unknown\_2**

**class NiMorpherController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpController`

Unknown! Used by Daoc.

**property data**

This controller's data.

```
class NiMultiTargetTransformController (template=None, argument=None, parent=None)
```

Bases: `pyffi.formats.nif.NiInterpController`

Unknown.

**property extra\_targets**

NiNode Targets to be controlled.

**property num\_extra\_targets**

The number of target pointers that follow.

```
class NiMultiTextureProperty (template=None, argument=None, parent=None)
```

Bases: `pyffi.formats.nif.NiProperty`

(note: not quite complete yet... but already reads most of the DAoC ones)

**property flags**

Property flags.

**property texture\_elements**

Describes the various textures used by this mutli-texture property. Each slot probably has special meaning like thoes in `NiTexturingProperty`.

**property unknown\_int**

Unknown. Always 5 for DAoC files, and always 6 for Bridge Commander. Seems to have nothing to do with the number of Texture Element slots that follow.

```
class NiNode (template=None, argument=None, parent=None)
```

Bases: `pyffi.formats.nif._NiNode, object`

```
>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.num_children = 1
>>> x.children.update_size()
>>> y in x.children
False
>>> x.children[0] = y
>>> y in x.children
True
>>> x.add_child(z, front = True)
>>> x.add_child(y)
>>> x.num_children
2
>>> x.children[0] is z
True
>>> x.remove_child(y)
>>> y in x.children
False
>>> x.num_children
1
>>> e = NifFormat.NiSpotLight()
>>> x.add_effect(e)
>>> x.num_effects
```

(continues on next page)

(continued from previous page)

```

1
>>> e in x.effects
True

```

```

>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> child1 = NifFormat.NiNode()
>>> child1.name = "hello"
>>> child_2 = NifFormat.NiNode()
>>> child_2.name = "world"
>>> node.get_children()
[]
>>> node.set_children([child1, child_2])
>>> [child.name for child in node.get_children()]
[b'hello', b'world']
>>> [child.name for child in node.children]
[b'hello', b'world']
>>> node.set_children([])
>>> node.get_children()
[]
>>> # now set them the other way around
>>> node.set_children([child_2, child1])
>>> [child.name for child in node.get_children()]
[b'world', b'hello']
>>> [child.name for child in node.children]
[b'world', b'hello']
>>> node.remove_child(child_2)
>>> [child.name for child in node.children]
[b'hello']
>>> node.add_child(child_2)
>>> [child.name for child in node.children]
[b'hello', b'world']

```

```

>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> effect1 = NifFormat.NiSpotLight()
>>> effect1.name = "hello"
>>> effect2 = NifFormat.NiSpotLight()
>>> effect2.name = "world"
>>> node.get_effects()
[]
>>> node.set_effects([effect1, effect2])
>>> [effect.name for effect in node.get_effects()]
[b'hello', b'world']
>>> [effect.name for effect in node.effects]
[b'hello', b'world']
>>> node.set_effects([])
>>> node.get_effects()
[]
>>> # now set them the other way around
>>> node.set_effects([effect2, effect1])
>>> [effect.name for effect in node.get_effects()]
[b'world', b'hello']
>>> [effect.name for effect in node.effects]
[b'world', b'hello']
>>> node.remove_effect(effect2)

```

(continues on next page)

(continued from previous page)

```

>>> [effect.name for effect in node.effects]
[b'hello']
>>> node.add_effect(effect2)
>>> [effect.name for effect in node.effects]
[b'hello', b'world']

```

**add\_child** (*child*, *front=False*)

Add block to child list.

**Parameters** **child** (*L{NifFormat.NiAVObject}*) – The child to add.

**Keyword Arguments** **front** – Whether to add to the front or to the end of the list (default is at end).

**add\_effect** (*effect*)

Add an effect to the list of effects.

**Parameters** **effect** (*L{NifFormat.NiDynamicEffect}*) – The effect to add.

**get\_children** ()

Return a list of the children of the block.

**Returns** The list of children.

**Return type** list of *L{NifFormat.NiAVObject}*

**get\_effects** ()

Return a list of the effects of the block.

**Returns** The list of effects.

**Return type** list of *L{NifFormat.NiDynamicEffect}*

**get\_skinned\_geometries** ()

This function yields all skinned geometries which have self as skeleton root.

**merge\_external\_skeleton\_root** (*skelroot*)

Attach skinned geometry to self (which will be the new skeleton root of the nif at the given skeleton root). Use this function if you move a skinned geometry from one nif into a new NIF file. The bone links will be updated to point to the tree at self, instead of to the external tree.

**merge\_skeleton\_roots** ()

This function will look for other geometries whose skeleton root is a (possibly indirect) child of this node. It will then reparent those geometries to this node. For example, it will unify the skeleton roots in Morrowind's cliffcracer.nif file, or of the (official) body skins. This makes it much easier to import skeletons in for instance Blender: there will be only one skeleton root for each bone, over all geometries.

The merge fails for those geometries whose global skin data transform does not match the inverse geometry transform relative to the skeleton root (the maths does not work out in this case!)

Returns list of all new blocks that have been reparented (and added to the skeleton root children list), and a list of blocks for which the merge failed.

**remove\_child** (*child*)

Remove a block from the child list.

**Parameters** **child** (*L{NifFormat.NiAVObject}*) – The child to remove.

**remove\_effect** (*effect*)

Remove a block from the effect list.

**Parameters** **effect** (*L{NifFormat.NiDynamicEffect}*) – The effect to remove.

**send\_bones\_to\_bind\_position** ()

This function will send all bones of geometries of this skeleton root to their bind position. For best results, call *L{send\_geometries\_to\_bind\_position}* first.

**Returns** A number quantifying the remaining difference between bind positions.

**Return type** float

**send\_detached\_geometries\_to\_node\_position** ()

Some nifs (in particular in Morrowind) have geometries that are skinned but that do not share bones. In such cases, `send_geometries_to_bind_position` cannot reposition them. This function will send such geometries to the position of their root node.

Examples of such nifs are the official Morrowind skins (after merging skeleton roots).

Returns list of detached geometries that have been moved.

**send\_geometries\_to\_bind\_position** ()

Call this on the skeleton root of geometries. This function will transform the geometries, such that all skin data transforms coincide, or at least coincide partially.

**Returns** A number quantifying the remaining difference between bind positions.

**Return type** float

**set\_children** (*childlist*)

Set the list of children from the given list (destroys existing list).

**Parameters** **childlist** (list of L{NifFormat.NiAVObject}) – The list of child blocks to set.

**set\_effects** (*effectlist*)

Set the list of effects from the given list (destroys existing list).

**Parameters** **effectlist** (list of L{NifFormat.NiDynamicEffect}) – The list of effect blocks to set.

**class NiObject** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiObject`, `object`

**apply\_scale** (*scale*)

Scale data in this block. This implementation does nothing. Override this method if it contains geometry data that can be scaled.

**find** (*block\_name=None, block\_type=None*)

**find\_chain** (*block, block\_type=None*)

Finds a chain of blocks going from C{self} to C{block}. If found, self is the first element and block is the last element. If no branch found, returns an empty list. Does not check whether there is more than one branch; if so, the first one found is returned.

**Parameters**

- **block** – The block to find a chain to.
- **block\_type** – The type that blocks should have in this chain.

**is\_interchangeable** (*other*)

Are the two blocks interchangeable?

@todo: Rely on AnyType, SimpleType, ComplexType, etc. implementation.

**tree** (*block\_type=None, follow\_all=True, unique=False*)

A generator for parsing all blocks in the tree (starting from and including C{self}).

**Parameters**

- **block\_type** – If not None, yield only blocks of the type C{block\_type}.
- **follow\_all** – If C{block\_type} is not None, then if this is True the function will parse the whole tree. Otherwise, the function will not follow branches that start by a non-C{block\_type} block.
- **unique** – Whether the generator can return the same block twice or not.

**class NiObjectNET** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiObjectNET`, `object`

**add\_controller** (*ctrlblock*)

Add block to controller chain and set target of controller to self.

**add\_extra\_data** (*extrablock*)

Add block to extra data list and extra data chain. It is good practice to ensure that the extra data has empty `next_extra_data` field when adding it to avoid loops in the hierarchy.

**add\_integer\_extra\_data** (*name, value*)

Add a particular extra integer data block.

**get\_controllers** ()

Get a list of all controllers.

**get\_extra\_datas** ()

Get a list of all extra data blocks.

**remove\_extra\_data** (*extrablock*)

Remove block from extra data list and extra data chain.

```
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiNode()
>>> block.num_extra_data_list = 3
>>> block.extra_data_list.update_size()
>>> extrablock = NifFormat.NiStringExtraData()
>>> block.extra_data_list[1] = extrablock
>>> block.remove_extra_data(extrablock)
>>> [extra for extra in block.extra_data_list]
[None, None]
```

**set\_extra\_datas** (*extralist*)

Set all extra data blocks from given list (erases existing data).

```
>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> extra1 = NifFormat.NiExtraData()
>>> extra1.name = "hello"
>>> extra2 = NifFormat.NiExtraData()
>>> extra2.name = "world"
>>> node.get_extra_datas()
[]
>>> node.set_extra_datas([extra1, extra2])
>>> [extra.name for extra in node.get_extra_datas()]
[b'hello', b'world']
>>> [extra.name for extra in node.extra_data_list]
[b'hello', b'world']
>>> node.extra_data is extra1
True
>>> extra1.next_extra_data is extra2
True
>>> extra2.next_extra_data is None
True
>>> node.set_extra_datas([])
>>> node.get_extra_datas()
[]
>>> # now set them the other way around
>>> node.set_extra_datas([extra2, extra1])
>>> [extra.name for extra in node.get_extra_datas()]
[b'world', b'hello']
>>> [extra.name for extra in node.extra_data_list]
```

(continues on next page)

(continued from previous page)

```
[b'world', b'hello']
>>> node.extra_data is extra2
True
>>> extra2.next_extra_data is extral
True
>>> extral.next_extra_data is None
True
```

**Parameters** `extralist` (list of L{NifFormat.NiExtraData}) – List of extra data blocks to add.

**class** `NiPSBombForce` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**property** `name`

**property** `unknown_1`

**property** `unknown_10`

**property** `unknown_2`

**property** `unknown_3`

**property** `unknown_4`

**property** `unknown_5`

**property** `unknown_6`

**property** `unknown_7`

**property** `unknown_8`

**property** `unknown_9`

**class** `NiPSBoundUpdater` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**property** `unknown_1`

**property** `unknown_2`

**class** `NiPSBoxEmitter` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

**property** `name`

**property** `unknown_1`

**property** `unknown_10`

**property** `unknown_11`

**property** `unknown_12`

**property** `unknown_13`

**property** `unknown_14`

**property** `unknown_15`

**property** `unknown_16`

**property** `unknown_17`

**property** `unknown_18`

property unknown\_19  
property unknown\_2  
property unknown\_20  
property unknown\_21  
property unknown\_22  
property unknown\_23  
property unknown\_24  
property unknown\_25  
property unknown\_26  
property unknown\_27  
property unknown\_28  
property unknown\_29  
property unknown\_3  
property unknown\_30  
property unknown\_31  
property unknown\_32  
property unknown\_33  
property unknown\_34  
property unknown\_35  
property unknown\_36  
property unknown\_37  
property unknown\_38  
property unknown\_39  
property unknown\_4  
property unknown\_40  
property unknown\_41  
property unknown\_42  
property unknown\_43  
property unknown\_44  
property unknown\_45  
property unknown\_46  
property unknown\_47  
property unknown\_48  
property unknown\_5  
property unknown\_6  
property unknown\_7

```
    property unknown_8
    property unknown_9
class NiPSCylinderEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSphereEmitter
    property unknown_23
class NiPSDragForce (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_10
    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9
class NiPSEmitParticlesCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysEmitterCtrl
class NiPSEmitterDeclinationCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
class NiPSEmitterDeclinationVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterDeclinationCtrl
class NiPSEmitterLifeSpanCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
class NiPSEmitterPlanarAngleCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
class NiPSEmitterPlanarAngleVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterPlanarAngleCtrl
class NiPSEmitterRadiusCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property interpolator
    property unknown_2
class NiPSEmitterRotAngleCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
class NiPSEmitterRotAngleVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterRotAngleCtrl
class NiPSEmitterRotSpeedCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl
```

```
class NiPSEmitterRotSpeedVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterRotSpeedCtrl

class NiPSEmitterSpeedCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    property interpolator
    property unknown_3

class NiPSFacingQuadGenerator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9

class NiPSForceActiveCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    property interpolator
    property unknown_2

class NiPSGravityForce (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_13
    property unknown_14
    property unknown_15
    property unknown_16
    property unknown_17
    property unknown_18
    property unknown_19
    property unknown_2
```

```
property unknown_20
property unknown_21
property unknown_22
property unknown_23
property unknown_24
property unknown_25
property unknown_26
property unknown_27
property unknown_28
property unknown_29
property unknown_3
property unknown_30
property unknown_31
property unknown_32
property unknown_33
property unknown_34
property unknown_35
property unknown_36
    Gravity node?
property unknown_4
property unknown_5
property unknown_6
property unknown_7
property unknown_8
property unknown_9
```

```
class NiPSGravityStrengthCtrl (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiTimeController
```

```
property unknown_2
```

```
property unknown_3
```

```
class NiPSMeshEmitter (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiObject
```

```
property name
```

```
property unknown_1
```

```
property unknown_10
```

```
property unknown_11
```

```
property unknown_12
```

```
property unknown_13
```

property unknown\_14  
 property unknown\_15  
 property unknown\_16  
 property unknown\_17  
 property unknown\_18  
 property unknown\_19  
 property unknown\_2  
 property unknown\_20  
 property unknown\_21  
 property unknown\_22  
 property unknown\_23  
 property unknown\_24  
 property unknown\_25  
 property unknown\_26  
 property unknown\_27  
 property unknown\_28  
 property unknown\_3  
 property unknown\_4  
 property unknown\_5  
 property unknown\_6  
 property unknown\_7  
 property unknown\_8  
 property unknown\_9

**class NiPSMeshParticleSystem** (*template=None, argument=None, parent=None*)

Bases: pyffi.formats.nif.NiPSParticleSystem

property unknown\_23

property unknown\_24

Unknown - may or may not be emitted mesh?

property unknown\_25

property unknown\_26

**class NiPSParticleSystem** (*template=None, argument=None, parent=None*)

Bases: pyffi.formats.nif.NiAVObject

property emitter

Emitter?

property generator

Generator?

property simulator

Simulator?

property unknown\_10  
0?

property unknown\_11  
0?

property unknown\_12  
Counter?

property unknown\_15  
Simulator?

property unknown\_16  
Updater?

property unknown\_17  
1?

property unknown\_19  
0?

property unknown\_20  
Spawner?

property unknown\_21  
Unknown

property unknown\_22  
Unknown

property unknown\_27

property unknown\_28

property unknown\_29

property unknown\_3  
0?

property unknown\_30

property unknown\_31

property unknown\_32

property unknown\_33

property unknown\_34

property unknown\_35

property unknown\_36  
-1?

property unknown\_37

property unknown\_38

property unknown\_39

property unknown\_4  
-1?

property unknown\_5  
0?

```

    property unknown_6
        256?

    property unknown_7
        0?

    property unknown_8
        0?

    property unknown_9
        0?

class NiPSPlanarCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property name

    property unknown_byte_4

    property unknown_floats_5

    property unknown_int_1

    property unknown_int_2

    property unknown_link_6

    property unknown_short_3

class NiPSResetOnLoopCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

class NiPSSimulator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiMeshModifier

    The mesh modifier that performs all particle system simulation.

    property num_simulation_steps
        The number of simulation steps in this modifier.

    property simulation_steps
        Links to the simulation steps.

class NiPSSimulatorCollidersStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep

    Encapsulates a floodgate kernel that simulates particle colliders.

    property colliders
        The colliders affecting the particle system.

    property num_colliders
        The number of colliders affecting the particle system.

class NiPSSimulatorFinalStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep

    Encapsulates a floodgate kernel that updates particle positions and ages. As indicated by its name, this step
    should be attached last in the NiPSSimulator mesh modifier.

class NiPSSimulatorForcesStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep

    Encapsulates a floodgate kernel that simulates particle forces.

```

**property forces**  
The forces affecting the particle system.

**property num\_forces**  
The number of forces affecting the particle system.

**class NiPSSimulatorGeneralStep** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSSimulatorStep`

Encapsulates a floodgate kernel that updates particle size, colors, and rotations.

**property color\_keys**  
The particle color keys.

**property color\_loop\_behavior**  
The loop behavior for the color keys.

**property grow\_generation**  
Specifies the particle generation to which the grow effect should be applied. This is usually generation 0, so that newly created particles will grow.

**property grow\_time**  
The the amount of time over which a particle's size is ramped from 0.0 to 1.0 in seconds

**property num\_color\_keys**  
The number of color animation keys.

**property num\_rotation\_keys**  
The number of rotatoin animation keys.

**property num\_size\_keys**  
The number of size animation keys.

**property rotation\_keys**  
The particle rotation keys.

**property rotation\_loop\_behavior**  
The loop behavior for the rotation keys.

**property shrink\_generation**  
Specifies the particle generation to which the shrink effect should be applied. This is usually the highest supported generation for the particle system, so that particles will shrink immediately before getting killed.

**property shrink\_time**  
The the amount of time over which a particle's size is ramped from 1.0 to 0.0 in seconds

**property size\_keys**  
The particle size keys.

**property size\_loop\_behavior**  
The loop behavior for the size keys.

**property unknown\_1**

**property unknown\_2**

**property unknown\_3**

**class NiPSSimulatorMeshAlignStep** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSSimulatorStep`

Encapsulates a floodgate kernel that updates mesh particle alignment and transforms.

```

property num_rotation_keys
    The number of rotation keys.

property rotation_keys
    The particle rotation keys.

property rotation_loop_behavior
    The loop behavior for the rotation keys.

class NiPSSimulatorStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Abstract base class for a single step in the particle system simulation process. It has no seralized data.

class NiPSSpawner (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiPSSphereEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

property name

property unknown_10

property unknown_11

property unknown_12

property unknown_13

property unknown_14

property unknown_15

property unknown_16

property unknown_17

property unknown_18

property unknown_19

property unknown_2

property unknown_20

property unknown_21
    Target node?

property unknown_22

property unknown_3

property unknown_4

property unknown_5

property unknown_6

property unknown_7

property unknown_8

property unknown_9

class NiPSSphericalCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

property unknown_1

```

**property unknown\_2**

**property unknown\_3**

**property unknown\_4**

**property unknown\_5**

**property unknown\_6**

**property unknown\_7**

**class NiPSysAgeDeathModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Unknown particle modifier.

**property spawn\_modifier**

Link to NiPSysSpawnModifier object?

**property spawn\_on\_death**

Unknown.

**class NiPSysAirFieldAirFrictionCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for air field air friction.

**class NiPSysAirFieldInheritVelocityCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for air field inherit velocity.

**class NiPSysAirFieldModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in a field like wind.

**property direction**

Direction of the particle velocity

**property unknown\_boolean\_1**

Unknown

**property unknown\_boolean\_2**

Unknown

**property unknown\_boolean\_3**

Unknown

**property unknown\_float\_2**

Unknown

**property unknown\_float\_3**

Unknown

**property unknown\_float\_4**

Unknown

**class NiPSysAirFieldSpreadCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for air field spread.

**class NiPSysBombModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that uses a NiNode to use as a “Bomb Object” to alter the path of particles.

**property bomb\_axis**  
Orientation of bomb object.

**property bomb\_object**  
Link to a NiNode for bomb to function.

**property decay**  
Falloff rate of the bomb object.

**property decay\_type**  
Decay type

**property delta\_v**  
DeltaV / Strength?

**property symmetry\_type**  
Shape/symmetry of the bomb object.

**class NiPSysBoundUpdateModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Unknown particle system modifier.

**property update\_skip**  
Unknown.

**class NiPSysBoxEmitter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysVolumeEmitter`

Particle emitter that uses points within a defined Box shape to emit from..

**property depth**  
Defines the Depth of the box area.

**property height**  
Defines the Height of the box area.

**property width**  
Defines the Width of the box area.

**class NiPSysCollider** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Particle system collider.

**property bounce**  
Defines amount of bounce the collider object has.

**property collider\_object**  
Links to a NiNode that will define where in object space the collider is located/oriented.

**property die\_on\_collide**  
Kill particles on impact if set to yes.

**property next\_collider**  
The next collider.

**property parent**  
Link to parent.

**property spawn\_modifier**  
Link to NiPSysSpawnModifier object?

**property spawn\_on\_collide**  
Unknown.

**class NiPSysColliderManager** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that adds a defined shape to act as a collision object for particles to interact with.

**property collider**  
Link to a NiPSysPlanarCollider or NiPSysSphericalCollider.

**class NiPSysColorModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that adds keyframe data to modify color/alpha values of particles over time.

**property data**  
Refers to NiColorData object.

**class NiPSysCylinderEmitter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysVolumeEmitter`

Particle emitter that uses points within a defined Cylinder shape to emit from.

**property height**  
Height of the cylinders shape.

**property radius**  
Radius of the cylinder shape.

**class NiPSysData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiRotatingParticlesData`

Particle system data.

**property aspect\_ratio**  
Sets aspect ratio for Subtexture Offset UV quads

**property has\_subtexture\_offset\_u\_vs**  
Boolean for Num Subtexture Offset UVs

**property has\_unknown\_floats\_3**  
Unknown.

**property num\_subtexture\_offset\_u\_vs**  
How many quads to use in BSPSysSubTexModifier for texture atlasing

**property particle\_descriptions**  
Unknown.

**property subtexture\_offset\_u\_vs**  
Defines UV offsets

**property unknown\_byte\_4**  
Unknown

**property unknown\_floats\_3**  
Unknown.

**property unknown\_int\_4**  
Unknown

**property unknown\_int\_5**  
Unknown

**property unknown\_int\_6**  
Unknown

**property unknown\_short\_1**  
Unknown.

**property unknown\_short\_2**  
Unknown.

**property unknown\_short\_3**  
Unknown

**class NiPSysDragFieldModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in drag space warp.

**property direction**  
Direction of the particle velocity

**property use\_direction**  
Whether to use the direction field?

**class NiPSysDragModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Unknown.

**property drag\_axis**  
The drag axis.

**property parent**  
Parent reference.

**property percentage**  
Drag percentage.

**property range**  
The range.

**property range\_falloff**  
The range falloff.

**class NiPSysEmitter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

A particle emitter?

**property declination**  
Declination / First axis.

**property declination\_variation**  
Declination randomness / First axis.

**property initial\_color**  
Defines color of a birthed particle.

**property initial\_radius**  
Size of a birthed particle.

**property life\_span**  
Duration until a particle dies.

**property life\_span\_variation**

Adds randomness to Life Span.

**property planar\_angle**

Planar Angle / Second axis.

**property planar\_angle\_variation**

Planar Angle randomness / Second axis .

**property radius\_variation**

Particle Radius randomness.

**property speed**

Speed / Inertia of particle movement.

**property speed\_variation**

Adds an amount of randomness to Speed.

**class NiPSysEmitterCtrl** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiPSysModifierCtrl`

Particle system emitter controller.

**property data**

This controller's data

**property visibility\_interpolator**

Links to a bool interpolator. Controls emitter's visibility status?

**class NiPSysEmitterCtrlData** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObject`

Particle system emitter controller data.

**property float\_keys**

Unknown.

**property num\_visibility\_keys**

Number of keys.

**property visibility\_keys**

Unknown.

**class NiPSysEmitterDeclinationCtrl** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`

Unknown.

**class NiPSysEmitterDeclinationVarCtrl** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`

Unknown.

**class NiPSysEmitterInitialRadiusCtrl** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`

Unknown.

**class NiPSysEmitterLifeSpanCtrl** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`

Unknown.

---

```

class NiPSysEmitterPlanarAngleCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for emitter planar angle.

class NiPSysEmitterPlanarAngleVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for emitter planar angle variation.

class NiPSysEmitterSpeedCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.

class NiPSysFieldAttenuationCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field attenuation.

class NiPSysFieldMagnitudeCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field magnitude.

class NiPSysFieldMaxDistanceCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field maximum distance.

class NiPSysFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Base for all force field particle modifiers.

    property attenuation
        Controls how quick the field diminishes

    property field_object
        Force Field Object

    property magnitude
        Magnitude of the force

    property max_distance
        Maximum distance

    property use_max_distance
        Use maximum distance

class NiPSysGravityFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysFieldModifier
    Particle system modifier, used for controlling the particle velocity in gravity field.

    property direction
        Direction of the particle velocity

class NiPSysGravityModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Adds gravity to a particle system, when linked to a NiNode to use as a Gravity Object.

    property decay
        Falloff range.

```

**property force\_type**  
Planar or Spherical type

**property gravity\_axis**  
Orientation of gravity.

**property gravity\_object**  
Refers to a NiNode for gravity location.

**property strength**  
The strength of gravity.

**property turbulence**  
Adds a degree of randomness.

**property turbulence\_scale**  
Range for turbulence.

**property unknown\_byte**  
Unknown

**class NiPSysGravityStrengthCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`  
Unknown.

**class NiPSysGrowFadeModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifier`  
Particle modifier that controls the time it takes to grow a particle from Size=0 to the specified Size in the emitter, and then back to 0. This modifier has no control over alpha settings.

**property base\_scale**  
Unknown

**property fade\_generation**  
Unknown.

**property fade\_time**  
Time in seconds to fade out.

**property grow\_generation**  
Unknown.

**property grow\_time**  
Time in seconds to fade in.

**class NiPSysInitialRotAngleCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`  
Particle system controller for emitter initial rotation angle.

**class NiPSysInitialRotAngleVarCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`  
Particle system controller for emitter initial rotation angle variation.

**class NiPSysInitialRotSpeedCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`  
Particle system controller for emitter initial rotation speed.

**class NiPSysInitialRotSpeedVarCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrl`

Particle system controller for emitter initial rotation speed variation.

**class NiPSysMeshEmitter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysEmitter`

Particle emitter that uses points on a specified mesh to emit from.

**property emission\_axis**

The emission axis.

**property emission\_type**

The parts of the mesh that the particles emit from.

**property emitter\_meshes**

Links to meshes used for emitting.

**property initial\_velocity\_type**

The way the particles get their initial direction and speed.

**property num\_emitter\_meshes**

The number of references to emitter meshes that follow.

**class NiPSysMeshUpdateModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifier`

Unknown.

**property meshes**

Group of target NiNodes or NiTriShapes?

**property num\_meshes**

The number of object references that follow.

**class NiPSysModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Generic particle system modifier object.

**property active**

Whether the modifier is currently in effect? Usually true.

**property name**

The object name.

**property order**

Modifier ID in the particle modifier chain (always a multiple of 1000)?

**property target**

NiParticleSystem parent of this modifier.

**class NiPSysModifierActiveCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifierBoolCtrlr`

Unknown.

**property data**

This controller's data.

**class NiPSysModifierBoolCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysModifierCtrlr`

A particle system modifier controller that deals with boolean data?

**class NiPSysModifierCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSingleInterpController`

A particle system modifier controller.

**property modifier\_name**  
Refers to modifier object by its name?

**class NiPSysModifierFloatCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifierCtrl`

A particle system modifier controller that deals with floating point data?

**property data**  
This controller's data.

**class NiPSysPlanarCollider** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysCollider`

Particle Collider object which particles will interact with.

**property height**  
Defines the height of the plane.

**property width**  
Defines the width of the plane.

**property x\_axis**  
Defines Orientation.

**property y\_axis**  
Defines Orientation.

**class NiPSysPositionModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifier`

Unknown particle system modifier.

**class NiPSysRadialFieldModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in force field.

**property radial\_type**  
Unknown Enums?

**class NiPSysResetOnLoopCtrl** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiTimeController`

Unknown.

**class NiPSysRotationModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that adds rotations to particles.

**property initial\_axis**  
Unknown.

**property initial\_rotation\_angle**  
Sets the initial angle for particles to be birthed in.

**property initial\_rotation\_angle\_variation**  
Adds a random range to Initial angle.

**property initial\_rotation\_speed**  
The initial speed of rotation.

**property initial\_rotation\_speed\_variation**  
Adds a ranged randomness to rotation speed.

**property random\_initial\_axis**  
Unknown.

**property random\_rot\_speed\_sign**  
Unknown

**class NiPSysSpawnModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysModifier`  
Unknown particle modifier.

**property life\_span**  
Unknown.

**property life\_span\_variation**  
Unknown.

**property max\_num\_to\_spawn**  
Unknown.

**property min\_num\_to\_spawn**  
Unknown.

**property num\_spawn\_generations**  
Unknown.

**property percentage\_spawned**  
Unknown.

**property spawn\_dir\_chaos**  
Unknown.

**property spawn\_speed\_chaos**  
Unknown.

**property unknown\_int**  
Unknown

**class NiPSysSphereEmitter** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysVolumeEmitter`  
Particle emitter that uses points within a sphere shape to emit from.

**property radius**  
The radius of the sphere shape

**class NiPSysSphericalCollider** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysCollider`  
Particle Collider object which particles will interact with.

**property radius**  
Defines the radius of the sphere object.

**class NiPSysTrailEmitter** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiPSysEmitter`  
Guild 2-Specific node

**property unknown\_float\_1**  
Unknown

**property unknown\_float\_2**  
Unknown

**property unknown\_float\_3**  
Unknown

**property unknown\_float\_4**  
Unknown

**property unknown\_float\_5**  
Unknown

**property unknown\_float\_6**  
Unknown

**property unknown\_float\_7**  
Unknown

**property unknown\_int\_1**  
Unknown

**property unknown\_int\_2**  
Unknown

**property unknown\_int\_3**  
Unknown

**property unknown\_int\_4**  
Unknown

**class NiPSysTurbulenceFieldModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in drag space warp.

**property frequency**  
Frequency of the update.

**class NiPSysUpdateCtrlr** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

Particle system controller, used for ???.

**class NiPSysVolumeEmitter** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysEmitter`

An emitter that emits meshes?

**property emitter\_object**  
Node parent of this modifier?

**class NiPSysVortexFieldModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in force field.

**property direction**  
Direction of the particle velocity

**class NiPalette** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

A color palette.

**property num\_entries**  
The number of palette entries. Always = 256.

**property palette**  
The color palette.

**property unknown\_byte**  
Unknown, Usually = 0.

**class NiParticleBomb** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticleModifier`  
A particle modifier.

**property decay**  
Unknown.

**property decay\_type**  
Unknown.

**property delta\_v**  
Unknown.

**property direction**  
The direction of the applied acceleration?

**property duration**  
Unknown.

**property position**  
The position of the mass point relative to the particle system?

**property start**  
Unknown.

**property symmetry\_type**  
Unknown.

**class NiParticleColorModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticleModifier`  
Unknown.

**property color\_data**  
Color data index.

**class NiParticleGrowFade** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticleModifier`  
This particle system modifier controls the particle size. If it is present the particles start with size 0.0 . Then they grow to their original size and stay there until they fade to zero size again at the end of their lifetime cycle.

**property fade**  
The time from the end of the particle lifetime during which the particle fades.

**property grow**  
The time from the beginning of the particle lifetime during which the particle grows.

**class NiParticleMeshModifier** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticleModifier`  
Unknown.

**property num\_particle\_meshes**

The number of particle mesh references that follow.

**property particle\_meshes**

Links to nodes of particle meshes?

**class NiParticleMeshes** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticles`

Mesh particle node?

**class NiParticleMeshesData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiRotatingParticlesData`

Particle meshes data.

**property unknown\_link\_2**

Refers to the mesh that makes up a particle?

**class NiParticleModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

A particle system modifier.

**property controller**

Points to the particle system controller parent.

**property next\_modifier**

Next particle modifier.

**class NiParticleRotation** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

Unknown.

**property initial\_axis**

Unknown.

**property random\_initial\_axis**

Unknown.

**property rotation\_speed**

Unknown.

**class NiParticleSystem** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticles`

A particle system.

**property modifiers**

The list of particle modifiers.

**property num\_modifiers**

The number of modifier references.

**property unknown\_int\_1**

Unknown

**property unknown\_short\_2**

Unknown

**property unknown\_short\_3**

Unknown

**property world\_space**

If true, Particles are birthed into world space. If false, Particles are birthed into object space.

**class NiParticleSystemController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

A generic particle system time controller object.

**property color\_data****property emit\_flags**

Emit Rate toggle bit (0 = auto adjust, 1 = use Emit Rate value)

**Type** Bit 0

**property emit\_rate**

Particle emission rate (particles per second)

**property emit\_start\_time**

Particle emit start time

**property emit\_stop\_time**

Particle emit stop time

**property emitter**

find out what type of object this refers to).

**Type** This index targets the particle emitter object (TODO)

**property horizontal\_angle**

emitter's horizontal opening angle

**property horizontal\_direction**

horizontal emit direction

**property lifetime**

Particle lifetime

**property lifetime\_random**

Particle lifetime random modifier

**property num\_particles**

Size of the following array. (Maximum number of simultaneous active particles)

**property num\_valid**

Number of valid entries in the following array. (Number of active particles at the time the system was saved)

**property old\_emit\_rate**

Particle emission rate in old files

**property old\_speed**

Particle speed in old files

**property particle\_extra**

Link to some optional particle modifiers (NiGravity, NiParticleGrowFade, NiParticleBomb, ...)

**property particle\_lifetime**

The particle's age.

**property particle\_link****property particle\_timestamp**

Timestamp of the last update.

**property particle\_unknown\_short**  
Unknown short

**property particle\_unknown\_vector**  
Unknown

**property particle\_velocity**  
Particle velocity

**property particle\_vertex\_id**  
Particle/vertex index matches array index

**property particles**  
Individual particle modifiers?

**property size**  
Particle size

**property speed**  
Particle speed

**property speed\_random**  
Particle random speed modifier

**property start\_random**  
Particle random start translation vector

**property trailer**  
Trailing null byte

**property unknown\_byte**  
Unknown byte, (=0)

**property unknown\_color**  
Unknown.

**property unknown\_float\_1**

**property unknown\_float\_13**  
? float=1.0 ?

**property unknown\_floats\_2**

**property unknown\_int\_1**  
? int=1 ?

**property unknown\_int\_2**  
? int=0 ?

**property unknown\_link**  
unknown int (=0xffffffff)

**property unknown\_link\_2**  
Unknown int (=0xffffffff)

**property unknown\_normal**  
Unknown.

**property unknown\_short\_2**  
? short=0 ?

**property unknown\_short\_3**  
? short=0 ?

**property vertical\_angle**  
emitter's vertical opening angle [radians]

**property vertical\_direction**  
horizontal3.1416 : down  
**Type** vertical emit direction [radians]0.0  
**Type** up1.6

**class NiParticles** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiGeometry`

Generic particle system node.

**class NiParticlesData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiGeometryData`

Generic rotating particles data object.

**property has\_radii**  
Is the particle size array present?

**property has\_rotation\_angles**  
Are the angles of rotation present?

**property has\_rotation\_axes**  
Are axes of rotation present?

**property has\_rotations**  
Is the particle rotation array present?

**property has\_sizes**  
Is the particle size array present?

**property has\_uv\_quadrants**  
if value is no, a single image rendered

**property num\_active**  
The number of active particles at the time the system was saved. This is also the number of valid entries in the following arrays.

**property num\_particles**  
The maximum number of particles (matches the number of vertices).

**property num\_uv\_quadrants**  
2,4,8,16,32,64 are potential values. If "Has" was no then this should be 256, which represents a 16x16 framed image, which is invalid

**property particle\_radius**  
The particles'size.

**property radii**  
The individual partical sizes.

**property rotation\_angles**  
Angles of rotation

**property rotation\_axes**  
Unknown

**property rotations**  
The individual particle rotations.

**property sizes**  
The individual partical sizes.

**property unknown\_byte\_1**  
Unknown, probably a boolean.

**property unknown\_byte\_2**  
Unknown

**property unknown\_link**  
Unknown

**property uv\_quadrants**

**class NiPathController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

Time controller for a path.

**property float\_data**  
Path controller data index (float data). ?

**property pos\_data**  
Path controller data index (position data). ?

**property unknown\_float\_2**  
Unknown, often 0?

**property unknown\_float\_3**  
Unknown, often 0?

**property unknown\_int\_1**  
Unknown, always 1?

**property unknown\_short**  
Unknown, always 0?

**property unknown\_short\_2**  
Unknown.

**class NiPathInterpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiKeyBasedInterpolator`

Unknown interpolator.

**property float\_data**  
Links to NiFloatData.

**property pos\_data**  
Links to NiPosData.

**property unknown\_float\_1**  
Unknown.

**property unknown\_float\_2**  
Unknown.

**property unknown\_int**  
Unknown.

**property unknown\_short**  
Unknown.

**property unknown\_short\_2**  
Unknown. Zero.

---

```

class NiPersistentSrcTextureRenderData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.ATextureRenderData
    property num_faces
        Unknown
    property num_pixels
        Unknown
    property pixel_data
        Raw pixel data holding the mipmaps. Mipmap zero is the full-size texture and they get smaller by half
        as the number increases.
    property unknown_int_6
        Unknown, same as the number of pixels? / number of blocks?
    property unknown_int_7
        Unknown

class NiPhysXActorDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property shape_description
        PhysX Shape Description
    property unknown_byte_1
        Unknown
    property unknown_byte_2
        Unknown
    property unknown_int_1
        Unknown
    property unknown_int_2
        Unknown
    property unknown_int_4
        Unknown
    property unknown_int_5
        Unknown
    property unknown_int_6
        Unknown
    property unknown_quat_1
        Unknown
    property unknown_quat_2
        Unknown
    property unknown_quat_3
        Unknown
    property unknown_ref_0
        Unknown
    property unknown_ref_1
        Unknown

```

**property unknown\_ref\_2**  
Unknown

**property unknown\_refs\_3**  
Unknown

**class NiPhysXBodyDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown PhysX node.

**property unknown\_bytes**  
Unknown

**class NiPhysXD6JointDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown PhysX node.

**property unknown\_bytes**  
Unknown

**class NiPhysXKinematicSrc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown PhysX node.

**property unknown\_bytes**  
Unknown

**class NiPhysXMaterialDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown node.

**property unknown\_byte\_1**  
Unknown

**property unknown\_byte\_2**  
Unknown

**property unknown\_int**  
Unknown

**class NiPhysXMeshDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown PhysX node.

**property num\_vertices**  
Number of mesh vertices

**property unknown\_byte\_1**  
Unknown

**property unknown\_byte\_2**  
Unknown

**property unknown\_bytes\_0**  
NXS

**property unknown\_bytes\_1**  
MESH

**property unknown\_bytes\_2**  
Unknown

**property unknown\_bytes\_3**  
Unknown

**property unknown\_float\_1**  
Unknown

**property unknown\_float\_2**  
Unknown

**property unknown\_int\_1**  
Unknown

**property unknown\_int\_2**  
Unknown

**property unknown\_int\_4**  
Unknown

**property unknown\_ints\_1**  
Unknown

**property unknown\_short\_1**  
Unknown

**property unknown\_short\_2**  
Unknown

**property unknown\_shorts\_1**  
Unknown

**property vertices**  
Vertices

**class NiPhysXProp** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObjectNET`

Unknown PhysX node.

**property num\_dests**  
Number of NiPhysXTransformDest references

**property prop\_description**  
PhysX Property Description.

**property transform\_dests**  
Unknown

**property unknown\_byte**  
Unknown

**property unknown\_float\_1**  
Unknown

**property unknown\_int**  
Unknown

**property unknown\_int\_1**  
Unknown

**property unknown\_refs\_1**  
Unknown

```
class NiPhysXPropDesc (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiObject  
    Unknown PhysX node.  
property actor_descs  
    Unknown  
property joint_descs  
    PhysX Joint Descriptions  
property material_descs  
    PhysX Material Descriptions  
property num_dests  
    Number of NiPhysXActorDesc references  
property num_joints  
    Unknown  
property num_materials  
    Unknown  
property unknown_byte_6  
    Unknown  
property unknown_int_1  
    Unknown  
property unknown_int_2  
    Unknown  
property unknown_int_3  
    Unknown  
property unknown_int_5  
    Unknown  
property unknown_string_4  
    Unknown  
  
class NiPhysXShapeDesc (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiObject  
    Unknown PhysX node.  
property mesh_description  
    PhysX Mesh Description  
property unknown_float_1  
    Unknown  
property unknown_float_2  
    Unknown  
property unknown_float_3  
    Unknown  
property unknown_int_1  
    Unknown  
property unknown_int_2  
    Unknown
```

**property unknown\_int\_3**  
Unknown

**property unknown\_int\_4**  
Unknown

**property unknown\_int\_5**  
Unknown

**property unknown\_int\_7**  
Unknown

**property unknown\_int\_8**  
Unknown

**property unknown\_quat\_1**  
Unknown

**property unknown\_quat\_2**  
Unknown

**property unknown\_quat\_3**  
Unknown

**property unknown\_short\_1**  
Unknown

**property unknown\_short\_2**  
Unknown

**class NiPhysXTransformDest** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Unknown PhysX node.

**property node**  
Affected node?

**property unknown\_byte\_1**  
Unknown. =1?

**property unknown\_byte\_2**  
Unknown. =0

**class NiPixelData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.ATextureRenderData`

A texture.

**property num\_faces**  
Unknown

**property num\_pixels**  
Total number of pixels

**property pixel\_data**  
Raw pixel data holding the mipmaps. Mipmap zero is the full-size texture and they get smaller by half as the number increases.

**class NiPlanarCollider** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

Unknown.

**property unknown\_float\_1**  
Unknown.

**property unknown\_float\_10**  
Unknown.

**property unknown\_float\_11**  
Unknown.

**property unknown\_float\_12**  
Unknown.

**property unknown\_float\_13**  
Unknown.

**property unknown\_float\_14**  
Unknown.

**property unknown\_float\_15**  
Unknown.

**property unknown\_float\_16**  
Unknown.

**property unknown\_float\_2**  
Unknown.

**property unknown\_float\_3**  
Unknown.

**property unknown\_float\_4**  
Unknown.

**property unknown\_float\_5**  
Unknown.

**property unknown\_float\_6**  
Unknown.

**property unknown\_float\_7**  
Unknown.

**property unknown\_float\_8**  
Unknown.

**property unknown\_float\_9**  
Unknown.

**property unknown\_short**  
Usually 0?

**property unknown\_short\_2**  
Unknown.

**class NiPoint3InterpController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSingleInterpController`

A controller that interpolates point 3 data?

**property data**  
Material color controller data object index. Points to NiPosData.

**property target\_color**  
Selects which color to control.

**class NiPoint3Interpolator** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiKeyBasedInterpolator`

Unknown.

**property data**

Reference to NiPosData.

**property point\_3\_value**

Value when posed? Value at time 0?

**class NiPointLight** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiLight`

A point light.

**property constant\_attenuation**

Constant Attenuation

**property linear\_attenuation**

Linear Attenuation

**property quadratic\_attenuation**

Quadratic Attenuation (see `glLight`)

**class NiPortal** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObject`

A Portal

**property num\_vertices**

Number of vertices in this polygon

**property target**

Target portal or room

**property unknown\_flags**

Unknown flags.

**property unknown\_short\_2**

Unknown

**property vertices**

Vertices

**class NiPosData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Position data.

**property data**

The position keys.

**class NiProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObjectNET`

A generic property object.

**class NiRangeLODData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiLODData`

Describes levels of detail based on distance of object from camera.

**property lod\_center**

?

**property lod\_levels**

The ranges of distance that each level of detail applies in.

**property num\_lod\_levels**

Number of levels of detail.

**class NiRawImageData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Raw image data.

**property height**

Image height

**property image\_type**

The format of the raw image data.

**property rgb\_image\_data**

Image pixel data.

**property rgba\_image\_data**

Image pixel data.

**property width**

Image width

**class NiRenderObject** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObject`

An object that can be rendered.

**property active\_material**

The index of the currently active material.

**property material\_data**

Per-material data.

**property material\_needs\_update\_default**

The initial value for the flag that determines if the internal cached shader is valid.

**property num\_materials**

The number of materials affecting this renderable object.

**class NiRollController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSingleInterpController`

Unknown.

**property data**

The data for the controller.

**class NiRoom** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Grouping node for nodes in a Portal

**property in\_portals**

Number of portals into room

**property items**

All geometry associated with room.

**property num\_in\_portals**

Number of doors into room

**property num\_items**  
Number of unknowns

**property num\_portals\_2**  
Number of doors out of room

**property num\_walls**  
Number of walls in a room?

**property portals\_2**  
Number of portals out of room

**property wall\_plane**  
Face normal and unknown value.

**class NiRoomGroup** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiNode`  
Grouping node for nodes in a Portal

**property num\_rooms**  
Number of rooms in this group

**property rooms**  
Rooms associated with this group.

**property shell\_link**  
Outer Shell Geometry Node?

**class NiRotatingParticles** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticles`  
Unknown.

**class NiRotatingParticlesData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiParticlesData`  
Rotating particles data object.

**property has\_rotations\_2**  
Is the particle rotation array present?

**property rotations\_2**  
The individual particle rotations.

**class NiScreenElements** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiTriShape`  
Two dimensional screen elements.

**class NiScreenElementsData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiTriShapeData`  
Two dimensional screen elements.

**property max\_polygons**  
Maximum number of polygons?

**property num\_polygons**  
Number of Polygons actually in use

**property polygon\_indices**  
Polygon Indices

**property polygons**

Polygons

**property unknown\_u\_short\_1**

Unknown

**property unknown\_u\_short\_2**

Unknown

**property unknown\_u\_short\_3**

Maximum number of faces

**property used\_triangle\_points**

Number of in-use triangles

**property used\_vertices**

Number of in-use vertices

**class NiScreenLODData** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiLODData`

Describes levels of detail based on size of object on screen?

**property bound\_center**

The center of the bounding sphere?

**property bound\_radius**

The radius of the bounding sphere?

**property proportion\_count**

The number of screen size based LOD levels.

**property proportion\_levels**

The LOD levels based on proportion of screen size?

**property world\_center**

The center of the bounding sphere in world space?

**property world\_radius**

The radius of the bounding sphere in world space?

**class NiSequence** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObject`

Root node used in some Empire Earth II .kf files (version 4.2.2.0).

**property controlled\_blocks**

Refers to controlled objects.

**property name**

Name of this object. This is also the name of the action associated with this file. For instance, if the original NIF file is called "demon.nif" and this animation file contains an attack sequence, then the file would be called "demon\_attack1.kf" and this field would contain the string "attack1".

**property num\_controlled\_blocks**

Number of controlled objects.

**property text\_keys**

Link to NiTextKeyExtraData.

**property text\_keys\_name**

Name of following referenced NiTextKeyExtraData class.

```

    property unknown_int_1
        Unknown.

    property unknown_int_4
        Unknown

    property unknown_int_5
        Unknown

class NiSequenceData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiSequenceStreamHelper (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObjectNET

    Keyframe animation root node, in .kf files.

class NiShadeProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Determines whether flat shading or smooth shading is used on a shape.

    property flags
        Enable smooth phong shading on this shape.If 1's bit is not set, hard-edged flat shading will be used
        on this shape.
        Type 1's Bit

class NiShadowGenerator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property name
    property num_unknown_links_1
    property target
    property unknown_flags
    property unknown_links_1
    property unkown_byte_5
    property unkown_byte_9
    property unkown_float_4
    property unkown_int_2
    property unkown_int_6
    property unkown_int_7
    property unkown_int_8

class NiSingleInterpController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController

    A controller referring to a single interpolator.

    property interpolator
        Link to interpolator.

class NiSkinData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiSkinData, object

    apply_scale (scale)
        Apply scale factor on data.

```

```

>>> from pyffi.formats.nif import NifFormat
>>> id44 = NifFormat.Matrix44()
>>> id44.set_identity()
>>> skelroot = NifFormat.NiNode()
>>> skelroot.name = 'Scene Root'
>>> skelroot.set_transform(id44)
>>> bone1 = NifFormat.NiNode()
>>> bone1.name = 'bone1'
>>> bone1.set_transform(id44)
>>> bone1.translation.x = 10
>>> skelroot.add_child(bone1)
>>> geom = NifFormat.NiTriShape()
>>> geom.set_transform(id44)
>>> skelroot.add_child(geom)
>>> skininst = NifFormat.NiSkinInstance()
>>> geom.skin_instance = skininst
>>> skininst.skeleton_root = skelroot
>>> skindata = NifFormat.NiSkinData()
>>> skininst.data = skindata
>>> skindata.set_transform(id44)
>>> geom.add_bone(bone1, {})
>>> geom.update_bind_position()
>>> bone1.translation.x
10.0
>>> skindata.bone_list[0].skin_transform.translation.x
-10.0
>>> import pyffi.spells.nif.fix
>>> import pyffi.spells.nif
>>> data = NifFormat.Data()
>>> data.roots = [skelroot]
>>> toaster = pyffi.spells.nif.NifToaster()
>>> toaster.scale = 0.1
>>> pyffi.spells.nif.fix.SpellScale(data=data, toaster=toaster).recurse()
pyffi.toaster:INFO:--- fix_scale ---
pyffi.toaster:INFO: scaling by factor 0.100000
pyffi.toaster:INFO:   ~~~ NiNode [Scene Root] ~~~
pyffi.toaster:INFO:   ~~~ NiNode [bone1] ~~~
pyffi.toaster:INFO:   ~~~ NiTriShape [] ~~~
pyffi.toaster:INFO:   ~~~ NiSkinInstance [] ~~~
pyffi.toaster:INFO:   ~~~ NiSkinData [] ~~~
>>> bone1.translation.x
1.0
>>> skindata.bone_list[0].skin_transform.translation.x
-1.0

```

**get\_transform()**

Return scale, rotation, and translation into a single 4x4 matrix.

**set\_transform(mat)**

Set rotation, transform, and velocity.

**class NiSkinInstance** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Skinning instance.

**property bones**

List of all armature bones.

**property data**

Skinning data reference.

**property num\_bones**

The number of node bones referenced as influences.

**property skeleton\_root**

Armature root node.

**property skin\_partition**

Refers to a NiSkinPartition objects, which partitions the mesh such that every vertex is only influenced by a limited number of bones.

**class NiSkinPartition** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Skinning data, optimized for hardware skinning. The mesh is partitioned in submeshes such that each vertex of a submesh is influenced only by a limited and fixed number of bones.

**property num\_skin\_partition\_blocks**

Unknown.

**property skin\_partition\_blocks**

Skin partition objects.

**class NiSkinningLODController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

**class NiSkinningMeshModifier** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiMeshModifier`

**property bone\_bounds**

The bounds of the bones. Only stored if the RECOMPUTE\_BOUNDS bit is set.

**property bone\_transforms**

The transforms that go from bind-pose space to bone space.

**property bones**

Pointers to the bone nodes that affect this skin.

**property flags**

USE\_SOFTWARE\_SKINNING = 0x0001 RECOMPUTE\_BOUNDS = 0x0002

**property num\_bones**

The number of bones referenced by this mesh modifier.

**property skeleton\_root**

The root bone of the skeleton.

**property skeleton\_transform**

The transform that takes the root bone parent coordinate system into the skin coordinate system.

**class NiSortAdjustNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Unknown node. Found in Loki.

**property sorting\_mode**

Sorting

**property unknown\_int\_2**

Unknown.

**class NiSourceCubeMap** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiSourceTexture`

Unknown node. Found in Emerge Demo.

**class NiSourceTexture** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTexture`

Describes texture source and properties.

**property alpha\_format**

the `NiTriShape` linked to this object must have a `NiAlphaProperty` in its list of properties to enable material and/or texture transparency.

**Type Note**

**property direct\_render**

Load direct to renderer

**property file\_name**

The original source filename of the image embedded by the referred `NiPixelData` object.

**property is\_static**

Is Static?

**property persist\_render\_data**

Render data is persistent

**property pixel\_data**

Pixel data object index. `NiPixelData` or `NiPersistentSrcTextureRendererData`

**property pixel\_layout**

Specifies the way the image will be stored.

**property unknown\_byte**

Unknown. Seems to be set if Pixel Data is present?

**property unknown\_link**

Unknown.

**property use\_external**

Is the texture external?

**property use\_mipmaps**

Specifies whether mip maps are used.

**class NiSpecularProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Gives specularity to a shape. Flags 0x0001.

**property flags**

1's Bit = Enable specular lighting on this shape.

**class NiSphericalCollider** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

Unknown.

**property unknown\_float\_1**

Unknown.

**property unknown\_float\_2**

Unknown.

**property unknown\_float\_3**  
Unknown.

**property unknown\_float\_4**  
Unknown.

**property unknown\_float\_5**  
Unknown.

**property unknown\_short\_1**  
Unknown.

**property unknown\_short\_2**  
Unknown.

**class NiSpotLight** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPointLight`

A spot.

**property cutoff\_angle**  
The opening angle of the spot.

**property exponent**  
`glLight`  
**Type** Describes the distribution of light. (see

**property unknown\_float**  
Unknown

**class NiStencilProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Allows control of stencil testing.

**property draw\_mode**  
Used to enabled double sided faces. Default is 3 (DRAW\_BOTH).

**property fail\_action**

**property flags**  
Stencil EnableBits 1-3: Fail ActionBits 4-6: Z Fail ActionBits 7-9: Pass ActionBits 10-11: Draw ModeBits 12-14: Stencil Function  
**Type** Property flags  
**Type** Bit 0

**property pass\_action**

**property stencil\_enabled**  
Enables or disables the stencil test.

**property stencil\_function**  
`glStencilFunc`.  
**Type** Selects the compare mode function (see

**property stencil\_mask**  
A bit mask. The default is 0xffffffff.

**property stencil\_ref**  
Unknown. Default is 0.

**property z\_fail\_action**

**class NiStringExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Apparently commands for an optimizer instructing it to keep things it would normally discard. Also refers to NiNode objects (through their name) in animation .kf files.

**property bytes\_remaining**

The number of bytes left in the record. Equals the length of the following string + 4.

**property string\_data**

The string.

**class NiStringPalette** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

List of 0x00-separated strings, which are names of controlled objects and controller types. Used in .kf files in conjunction with NiControllerSequence.

**property palette**

A bunch of 0x00 separated strings.

**class NiStringsExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

List of strings; for example, a list of all bone names.

**property data**

The strings.

**property num\_strings**

Number of strings.

**class NiSwitchNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

A node used to switch between branches, such as for LOD levels?

**property unknown\_flags\_1**

Flags

**property unknown\_int\_1**

Index?

**class NiTextKeyExtraData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Extra data, used to name different animation sequences.

**property num\_text\_keys**

The number of text keys that follow.

**property text\_keys**

List of textual notes and at which time they take effect. Used for designating the start and stop of animations and the triggering of sounds.

**property unknown\_int\_1**

Unknown. Always equals zero in all official files.

**class NiTexture** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObjectNET`

A texture.

**class NiTextureEffect** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiDynamicEffect`

Enables environment mapping. Should be in both the children list and effects list of the NiTriShape object. For Morrowind: the bump map can be used to bump the environment map (note that the bump map is ignored if no NiTextureEffect object is present).

**property clipping\_plane**

Determines whether a clipping plane is used. 0 means that a plane is not used.

**property coordinate\_generation\_type**

The method that will be used to generate UV coordinates for the texture effect.

**property image**

Image index.

**property model\_projection\_matrix**

Model projection matrix. Always identity?

**property model\_projection\_transform**

Model projection transform. Always (0,0,0)?

**property ps\_2\_k**

-75?

**property ps\_2\_l**

0?

**property source\_texture**

Source texture index.

**property texture\_clamping**

Texture Clamp mode.

**property texture\_filtering**

Texture Filtering mode.

**property texture\_type**

The type of effect that the texture is used for.

**property unknown**

**property unknown\_float**

Unknown. 0?

**property unknown\_short**

0.

**Type** Unknown

**property unknown\_vector**

(1,0,0)?

**Type** Unknown

**class NiTextureModeProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Unknown

**property ps\_2\_k**

-75?

**property ps\_2\_l**

0?

**property unknown\_ints**

**property unknown\_short**  
Unknown. Either 210 or 194.

**class NiTextureProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

**property flags**  
Property flags.

**property image**  
Link to the texture image.

**property unknown\_ints\_1**  
Property flags.

**property unknown\_ints\_2**  
Unknown. 0?

**class NiTextureTransformController** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiFloatInterpController`

Texture transformation controller. The target texture slot should have “Has Texture Transform” enabled.

**property data**  
Link to NiFloatData.

**property operation**  
Determines how this controller animates the UV Coordinates.

**property texture\_slot**  
The target texture slot.

**property unknown\_2**  
Unknown.

**class NiTexturingProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Describes an object’s textures.

**property apply\_mode**  
Determines how the texture will be applied. Seems to have special functions in Oblivion.

**property base\_texture**  
The base texture.

**property bump\_map\_luma\_offset**  
Unknown.

**property bump\_map\_luma\_scale**  
Unknown.

**property bump\_map\_matrix**  
Unknown.

**property bump\_map\_texture**  
The bump map texture.

**property dark\_texture**  
The dark texture.

**property decal\_0\_texture**  
The decal texture.

---

**property decal\_1\_texture**  
Another decal texture.

**property decal\_2\_texture**  
Another decal texture.

**property decal\_3\_texture**  
Another decal texture. Who knows the limit.

**property detail\_texture**  
The detail texture.

**property flags**  
Property flags.

**property gloss\_texture**  
The gloss texture.

**property glow\_texture**  
The glowing texture.

**property has\_base\_texture**  
Do we have a base texture?

**property has\_bump\_map\_texture**  
Do we have a bump map texture?

**property has\_dark\_texture**  
Do we have a dark texture?

**property has\_decal\_0\_texture**  
Do we have a decal 0 texture?

**property has\_decal\_1\_texture**  
Do we have a decal 1 texture?

**property has\_decal\_2\_texture**  
Do we have a decal 2 texture?

**property has\_decal\_3\_texture**  
Do we have a decal 3 texture?

**property has\_detail\_texture**  
Do we have a detail texture?

**property has\_gloss\_texture**  
Do we have a gloss texture?

**property has\_glow\_texture**  
Do we have a glow texture?

**property has\_normal\_texture**  
Do we have a normal texture? (Normal guess based on file suffix in sample files)

**property has\_unknown\_2\_texture**  
Do we have a unknown texture 2?

**property normal\_texture**  
Normal texture.

**property num\_shader\_textures**  
Number of Shader textures that follow.

**property shader\_textures**

Shader textures.

**property texture\_count**

Number of textures. Always 7 in versions&lt;20.0.0.4. Can also be 8 in&gt;= 20.0.0.4.

**property unknown\_2\_float**

Unknown.

**property unknown\_2\_texture**

Unknown texture 2.

**class NiTimeController** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObject`

A generic time controller object.

**property flags**

Cycle type 00=Loop 01=Reverse 10=LoopBit 3 : ActiveBit 4 : Play backwards

**Type** Controller flags (usually 0x000C) Probably controls loops.Bit 0**Type** Anim type, 0=APP\_TIME 1=APP\_INITBit 1-2**property frequency**

Frequency (is usually 1.0).

**property next\_controller**

Index of the next controller.

**property phase**

Phase (usually 0.0).

**property start\_time**

Controller start time.

**property stop\_time**

Controller stop time.

**property target**

Controller target (object index of the first controllable ancestor of this object).

**property unknown\_integer**

Unknown integer.

**class NiTransformController** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiKeyframeController`

NiTransformController replaces the NiKeyframeController.

**class NiTransformData** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiKeyframeData`

Mesh animation keyframe data.

**class NiTransformEvaluator** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiObject`**class NiTransformInterpolator** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif._NiTransformInterpolator, object`**apply\_scale** (*scale*)

Apply scale factor &lt;scale&gt; on data.

**class NiTransparentProperty** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif.NiProperty`

Unknown

**property unknown**

Unknown.

**class NiTriBasedGeom** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiTriBasedGeom`, `object`

**get\_interchangeable\_tri\_shape** (*triangles=None*)

Returns a NiTriShape block that is geometrically interchangeable. If you do not want to set the triangles from the original shape, use the triangles argument.

**get\_interchangeable\_tri\_strips** (*strips=None*)

Returns a NiTriStrips block that is geometrically interchangeable. If you do not want to set the strips from the original shape, use the strips argument.

**get\_tangent\_space** ()

Return iterator over normal, tangent, bitangent vectors. If the block has no tangent space, then returns None.

**update\_skin\_center\_radius** ()

Update centers and radii of all skin data fields.

**update\_skin\_partition** (*maxbonesperpartition=4, maxbonespervortex=4, verbose=0, stripify=True, stitchstrips=False, padbones=False, triangles=None, trianglepartmap=None, maximize\_bone\_sharing=False*)

Recalculate skin partition data.

**Deprecated** Do not use the verbose argument.

**Parameters**

- **maxbonesperpartition** – Maximum number of bones in each partition. The `num_bones` field will not exceed this number.
- **maxbonespervortex** – Maximum number of bones per vertex. The `num_weights_per_vertex` field will be exactly equal to this number.
- **verbose** – Ignored, and deprecated. Set pyffi's log level instead.
- **stripify** – If true, stripify the partitions, otherwise use triangles.
- **stitchstrips** – If stripify is true, then set this to true to stitch the strips.
- **padbones** – Enforces the `numbones` field to be equal to `maxbonesperpartition`. Also ensures that the bone indices are unique and sorted, per vertex. Raises an exception if `maxbonespervortex` is not equal to `maxbonesperpartition` (in that case bone indices cannot be unique and sorted). This options is required for Freedom Force vs. the 3rd Reich skin partitions.
- **triangles** – The triangles of the partition (if not specified, then this defaults to `C{self.data.get_triangles()}`).
- **trianglepartmap** – Maps each triangle to a partition index. Faces with different indices will never appear in the same partition. If the skin instance is a `BSDismemberSkinInstance`, then these indices are used as body part types, and the partitions in the `BSDismemberSkinInstance` are updated accordingly. Note that the faces are counted relative to `L{triangles}`.
- **maximize\_bone\_sharing** – Maximize bone sharing between partitions. This option is useful for Fallout 3.

**update\_tangent\_space** (*as\_extra=None, vertexprecision=3, normalprecision=3*)

Recalculate tangent space data.

**Parameters** **as\_extra** – Whether to store the tangent space data as extra data (as in Oblivion) or not (as in Fallout 3). If not set, switches to Oblivion if an extra data block is found, otherwise does default. Set it to override this detection (for example when using this function to create tangent space data) and force behaviour.

**class NiTriBasedGeomData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiTriBasedGeomData, object`

**get\_triangle\_indices** (*triangles*)

Yield list of triangle indices (relative to `self.get_triangles()`) of given triangles. Degenerate triangles in the list are assigned index `None`.

```
>>> from pyffi.formats.nif import NifFormat
>>> geomdata = NifFormat.NiTriShapeData()
>>> geomdata.set_triangles([(0,1,2), (1,2,3), (2,3,4)])
>>> list(geomdata.get_triangle_indices([(1,2,3)]))
[1]
>>> list(geomdata.get_triangle_indices([(3,1,2)]))
[1]
>>> list(geomdata.get_triangle_indices([(2,3,1)]))
[1]
>>> list(geomdata.get_triangle_indices([(1,2,0), (4,2,3)]))
[0, 2]
>>> list(geomdata.get_triangle_indices([(0,0,0), (4,2,3)]))
[None, 2]
>>> list(geomdata.get_triangle_indices([(0,3,4), (4,2,3)]))
Traceback (most recent call last):
...
ValueError: ...
```

**Parameters triangles** (*iterator or list of tuples of three ints*) –

An iterable of triangles to check.

**is\_interchangeable** (*other*)

Heuristically checks if two `NiTriBasedGeomData` blocks describe the same geometry, that is, if they can be used interchangeably in a NIF file without affecting the rendering. The check is not fool proof but has shown to work in most practical cases.

**Parameters other** (`L{NifFormat.NiTriBasedGeomData}`) (if it has another type then the function will always return `False`) – Another geometry data block.

**Returns** `True` if the geometries are equivalent, `False` otherwise.

**class NiTriShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTriBasedGeom`

A shape node that refers to singular triangle data.

**class NiTriShapeData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiTriShapeData, object`

Example usage:

```
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiTriShapeData()
>>> block.set_triangles([(0,1,2), (2,1,3), (2,3,4)])
>>> block.get_strips()
[[0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 1, 2), (2, 1, 3), (2, 3, 4)]
>>> block.set_strips([[1,0,1,2,3,4]])
>>> block.get_strips() # stripifier keeps geometry but nothing else
[[0, 2, 1, 3], [2, 4, 3]]
>>> block.get_triangles()
[(0, 2, 1), (1, 2, 3), (2, 4, 3)]
```

**get\_strips** ()

```

get_triangles()
set_strips(strips)
set_triangles(triangles, stitchstrips=False)
class NiTriShapeSkinController(template=None, argument=None, parent=None)
  Bases: pyffi.formats.nif.NiTimeController
  Old version of skinning instance.
  property bone_data
    Contains skin weight data for each node that this skin is influenced by.
  property bones
    List of all armature bones.
  property num_bones
    The number of node bones referenced as influences.
  property vertex_counts
    The number of vertex weights stored for each bone.
class NiTriStrips(template=None, argument=None, parent=None)
  Bases: pyffi.formats.nif.NiTriBasedGeom
  A shape node that refers to data organized into strips of triangles
class NiTriStripsData(template=None, argument=None, parent=None)
  Bases: pyffi.formats.nif._NiTriStripsData, object
  Example usage:

```

```

>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiTriStripsData()
>>> block.set_triangles([(0,1,2), (2,1,3), (2,3,4)])
>>> block.get_strips()
[[0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 1, 2), (1, 3, 2), (2, 3, 4)]
>>> block.set_strips([[1,0,1,2,3,4]])
>>> block.get_strips()
[[1, 0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 2, 1), (1, 2, 3), (2, 4, 3)]

```

```

get_strips()
get_triangles()
set_strips(strips)
set_triangles(triangles, stitchstrips=False)
class NiUVController(template=None, argument=None, parent=None)
  Bases: pyffi.formats.nif.NiTimeController
  Time controller for texture coordinates.
  property data
    Texture coordinate controller data index.
  property unknown_short
    Always 0?

```

**class NiUVData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Texture coordinate data.

**property uv\_groups**  
Four UV data groups. Appear to be U translation, V translation, U scaling/tiling, V scaling/tiling.

**class NiVectorExtraData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Extra vector data.

**property unknown\_float**  
Not sure whether this comes before or after the vector data.

**property vector\_data**  
The vector data.

**class NiVertWeightsExtraData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiExtraData`  
Not used in skinning. Unsure of use - perhaps for morphing animation or gravity.

**property num\_bytes**  
Number of bytes in this data object.

**property num\_vertices**  
Number of vertices.

**property weight**  
The vertex weights.

**class NiVertexColorProperty** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiProperty`  
Property of vertex colors. This object is referred to by the root object of the NIF file whenever some `NiTriShapeData` object has vertex colors with non-default settings; if not present, vertex colors have `vertex_mode=2` and `lighting_mode=1`.

**property flags**  
Lighting Mode?Bits 4-5: Vertex Mode?  
**Type** Property flags. Appears to be unused until 20.1.0.3. Bits 0-2  
**Type** UnknownBit 3

**property lighting\_mode**  
The light mode. In Flags from 20.1.0.3 on.

**property vertex\_mode**  
`glColorMaterialIn` Flags from version 20.1.0.3 onwards.  
**Type** Determines how vertex and material colors are mixed.related gl function

**class NiVisController** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiBoolInterpController`  
Time controller for visibility.

**property data**  
Visibility controller data object index.

**class NiVisData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Visibility data for a controller.

**property keys**

The visibility keys.

**property num\_keys**

The number of visibility keys that follow.

**class NiWireframeProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Unknown.

**property flags**

Property flags.0 - Wireframe Mode Disabled1 - Wireframe Mode Enabled

**class NiZBufferProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

This Property controls the Z buffer (OpenGL: depth buffer).

**property flags**

Bit 0 enables the z testBit 1 controls whether the Z buffer is read only (0) or read/write (1)

**property function**

`glDepthFunc`). In Flags from 20.1.0.3 on.

**Type** Z-Test function (see

**exception NifError**

Bases: `Exception`

Standard nif exception class.

**class NodeGroup** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A group of NiNodes references.

**property nodes**

The list of NiNode references.

**property num\_nodes**

Number of node references that follow.

**class OblivionHavokMaterial** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

A material, used by havok shape objects in Oblivion.

**HAV\_MAT\_CHAIN = 13**

**HAV\_MAT\_CHAIN\_STAIRS = 28**

**HAV\_MAT\_CLOTH = 1**

**HAV\_MAT\_CLOTH\_STAIRS = 16**

**HAV\_MAT\_DIRT = 2**

**HAV\_MAT\_DIRT\_STAIRS = 17**

**HAV\_MAT\_ELEVATOR = 30**

**HAV\_MAT\_GLASS = 3**

**HAV\_MAT\_GLASS\_STAIRS = 18**

**HAV\_MAT\_GRASS = 4**

```
HAV_MAT_GRASS_STAIRS = 19
HAV_MAT_HEAVY_METAL = 11
HAV_MAT_HEAVY_METAL_STAIRS = 26
HAV_MAT_HEAVY_STONE = 10
HAV_MAT_HEAVY_STONE_STAIRS = 25
HAV_MAT_HEAVY_WOOD = 12
HAV_MAT_HEAVY_WOOD_STAIRS = 27
HAV_MAT_METAL = 5
HAV_MAT_METAL_STAIRS = 20
HAV_MAT_ORGANIC = 6
HAV_MAT_ORGANIC_STAIRS = 21
HAV_MAT_RUBBER = 31
HAV_MAT_SKIN = 7
HAV_MAT_SKIN_STAIRS = 22
HAV_MAT_SNOW = 14
HAV_MAT_SNOW_STAIRS = 29
HAV_MAT_STONE = 0
HAV_MAT_STONE_STAIRS = 15
HAV_MAT_WATER = 8
HAV_MAT_WATER_STAIRS = 23
HAV_MAT_WOOD = 9
HAV_MAT_WOOD_STAIRS = 24
```

```
class OblivionLayer(**kwargs)
```

```
    Bases: pyffi.object_models.xml.enum.EnumBase
```

```
    Sets mesh color in Oblivion Construction Set. Anything higher than 57 is also null.
```

```
    ANIM_STATIC = 2
```

```
    AVOID_BOX = 21
```

```
    BACK_WEAPON = 53
```

```
    BACK_WEAPON2 = 54
```

```
    BIPED = 8
```

```
    BODY = 34
```

```
    CAMERA_PICK = 24
```

```
    CHAR_CONTROLLER = 20
```

```
    CLOUD_TRAP = 16
```

```
    CLUTTER = 4
```

```
    CUSTOM_PICK_1 = 28
```

```
CUSTOM_PICK_2 = 29
DROPPING_PICK = 31
GROUND = 17
HEAD = 33
ITEM_PICK = 25
LINE_OF_SIGHT = 26
L_CALF = 41
L_FOOT = 42
L_FOREARM = 38
L_HAND = 39
L_THIGH = 40
L_UPPER_ARM = 37
NONCOLLIDABLE = 15
NULL = 57
OTHER = 32
PATH_PICK = 27
PONYTAIL = 55
PORTAL = 18
PROJECTILE = 6
PROPS = 10
QUIVER = 52
R_CALF = 47
R_FOOT = 48
R_FOREARM = 44
R_HAND = 45
R_THIGH = 46
R_UPPER_ARM = 43
SHIELD = 51
SIDE_WEAPON = 50
SPELL = 7
SPELL_EXPLOSION = 30
SPINE1 = 35
SPINE2 = 36
STAIRS = 19
STATIC = 1
TAIL = 49
```

```
TERRAIN = 13
TRANSPARENT = 3
TRAP = 14
TREES = 9
TRIGGER = 12
UNIDENTIFIED = 0
UNKNOWN1 = 22
UNKNOWN2 = 23
WATER = 11
WEAPON = 5
WING = 56

class OblivionSubShape (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Havok Information for packed TriStrip shapes.

    property havok_col_filter

    property material
        The material of the subshape.

    property num_vertices
        The number of vertices that form this sub shape.

class OldSkinData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Used to store skin weights in NiTriShapeSkinController.

    property unknown_vector
        Unknown. Perhaps some sort of offset?

    property vertex_index
        The index of the vertex that this weight applies to.

    property vertex_weight
        The amount that this bone affects the vertex.

class PSLoopBehavior (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    PS_LOOP_AGESCALE = 2

    PS_LOOP_CLAMP_BIRTH = 0

    PS_LOOP_CLAMP_DEATH = 1

    PS_LOOP_LOOP = 3

    PS_LOOP_REFLECT = 4

class Particle (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    particle array entry
```

```

property lifespan
    Maximum age of the particle.

property lifetime
    The particle's age.

property timestamp
    Timestamp of the last update.

property unknown_short
    Unknown short

property unknown_vector
    Unknown

property velocity
    Particle velocity

property vertex_id
    Particle/vertex index matches array index

class ParticleDesc (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Particle Description.

property translation
    Unknown.

property unknown_float_1
    Unknown.

property unknown_float_2
    Unknown.

property unknown_float_3
    Unknown.

property unknown_floats_1
    Unknown.

property unknown_int_1
    Unknown.

class PixelFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Specifies the pixel format used by the NiPixelData object to store a texture.

    PX_FMT_DXT1 = 4
    PX_FMT_DXT5 = 5
    PX_FMT_DXT5_ALT = 6
    PX_FMT_PAL8 = 2
    PX_FMT_RGB8 = 0
    PX_FMT_RGBA8 = 1

class PixelLayout (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing the color depth of a texture.

```

```
PIX_LAY_BUMPMAP = 4
PIX_LAY_COMPRESSED = 3
PIX_LAY_DEFAULT = 6
PIX_LAY_HIGH_COLOR_16 = 1
PIX_LAY_PALETTISED = 0
PIX_LAY_PALETTISED_4 = 5
PIX_LAY_TRUE_COLOR_32 = 2
```

```
class Polygon (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
```

Two dimensional screen elements.

```
property num_triangles
    Number of faces in this polygon
```

```
property num_vertices
    Number of vertices in this polygon
```

```
property triangle_offset
    Triangle offset in shape
```

```
property vertex_offset
    Vertex Offset
```

```
class PrismaticDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
```

```
property friction
    Friction.
```

```
property max_distance
    Describe the max distance the object is able to travel.
```

```
property min_distance
    Describe the min distance the object is able to travel.
```

```
property pivot_a
    Pivot.
```

```
property pivot_b
    Pivot in B coordinates.
```

```
property plane_a
    Plane normal. Describes the plane the object is able to move on.
```

```
property plane_b
    Plane normal. Describes the plane the object is able to move on in B coordinates.
```

```
property rotation_a
    Rotation axis.
```

```
property rotation_b
    Rotation axis.
```

```
property rotation_matrix_a
    4x4 rotation matrix, rotates the child entity.
```

```
property sliding_a
    Describes the axis the object is able to travel along. Unit vector.
```

**property sliding\_b**

Describes the axis the object is able to travel along in B coordinates. Unit vector.

**property unknown\_byte\_1**

Unknown. Do not set this to anything over 0 as it will crash the game.

**class PropagationMode** (\*\*kwargs)

Bases: `pyffi.object_models.xml.enum.EnumBase`

**PROPAGATE\_ALWAYS** = 2

**PROPAGATE\_NEVER** = 3

**PROPAGATE\_ON\_FAILURE** = 1

**PROPAGATE\_ON\_SUCCESS** = 0

**class Ptr** (\*\*kwargs)

Bases: `pyffi.formats.nif.Ref`

A weak reference to another block, used to point up the hierarchy tree. The reference is not returned by the `L{get_refs}` function to avoid infinite recursion.

**get\_hash** (data=None)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_refs** (data=None)

Return all references (excluding weak pointers) used by this object.

**get\_value** ()

Return object value.

**replace\_global\_node** (oldbranch, newbranch, edge\_filter=(True, True))

```
>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.add_child(y)
>>> x.children[0] is y
True
>>> x.children[0] is z
False
>>> x.replace_global_node(y, z)
>>> x.children[0] is y
False
>>> x.children[0] is z
True
>>> x.replace_global_node(z, None)
>>> x.children[0] is None
True
```

**set\_value** (value)

Set object value.

**class QTransform** (template=None, argument=None, parent=None)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property rotation**

Rotation.

**property scale**

Scale.

**property translation**

Translation.

**class QuatKey** (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`

A special version of the key type used for quaternions. Never has tangents.

**property tbc**

The TBC of the key.

**property time**

Time the key applies.

**property value**

Value of the key.

**class Quaternion** (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`

A quaternion.

**property w**

The w-coordinate.

**property x**

The x-coordinate.

**property y**

The y-coordinate.

**property z**

The z-coordinate.

**class QuaternionXYZW** (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`

A quaternion as it appears in the havok objects.

**property w**

The w-coordinate.

**property x**

The x-coordinate.

**property y**

The y-coordinate.

**property z**

The z-coordinate.

**RE\_FILENAME** = `re.compile('^.*\\.(nif|kf|kfa|nifcache|jmi|texcache|pcpatch|nft|item|nif`**class RagdollDescriptor** (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.nif._RagdollDescriptor, object`**update\_a\_b** (*transform*)

Update B pivot and axes from A using the given transform.

**class Ref** (*\*\*kwargs*)Bases: `pyffi.object_models.xml.basic.BasicBase`

Reference to another block.

**fix\_links** (*data*)  
Fix block links.

**get\_detail\_display** ()  
Return an object that can be used to display the instance.

**get\_hash** (*data=None*)  
Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_links** (*data=None*)  
Return all links referred to in this object.

**get\_refs** (*data=None*)  
Return all references (excluding weak pointers) used by this object.

**get\_size** (*data=None*)  
Returns size of the object in bytes.

**get\_value** ()  
Return object value.

**read** (*stream, data*)  
Read object from file.

**replace\_global\_node** (*oldbranch, newbranch, edge\_filter=(True, True)*)

```
>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.add_child(y)
>>> x.children[0] is y
True
>>> x.children[0] is z
False
>>> x.replace_global_node(y, z)
>>> x.children[0] is y
False
>>> x.children[0] is z
True
>>> x.replace_global_node(z, None)
>>> x.children[0] is None
True
```

**set\_value** (*value*)  
Set object value.

**write** (*stream, data*)  
Write block reference.

**class Region** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`

A range of indices, which make up a region (such as a submesh).

**property num\_indices**

**property start\_index**

**class RootCollisionNode** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiNode`

Morrowind-specific node for collision mesh.

**class SemanticData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property index**

An extra index of the data. For example, if there are 3 uv maps, then the corresponding TEXCOORD data components would have indices 0, 1, and 2, respectively.

**property name**

Type of data (POSITION, POSITION\_BP, INDEX, NORMAL, NORMAL\_BP, TEXCOORD, BLENDINDICES, BLENDWEIGHT, BONE\_PALETTE, COLOR, DISPLAYLIST, MORPH\_POSITION, BINORMAL\_BP, TANGENT\_BP).

**class ShaderTexDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

An extended texture description for shader textures.

**property is\_used**

Is it used?

**property map\_index**

Map Index

**property texture\_data**

The texture data.

**class ShortString** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

Another type for strings.

**get\_hash** (*data=None*)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)

Returns size of the object in bytes.

**get\_value** ()

Return object value.

**read** (*stream, data*)

Read object from file.

**set\_value** (*value*)

Set object value.

**write** (*stream, data*)

Write object to file.

**class SizedString** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
```

(continues on next page)

(continued from previous page)

```

>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'

```

**get\_hash** (*data=None*)

Return a hash value for this string.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes this type occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Return the string.

**Returns** The stored string.

**read** (*stream, data*)

Read string from stream.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Set string to C{value}.

**Parameters** **value** (*str*) – The value to assign.

**write** (*stream, data*)

Write string to stream.

**Parameters** **stream** (*file*) – The stream to write to.

**class SkinData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._SkinData`, `object`

**get\_transform** ()

Return scale, rotation, and translation into a single 4x4 matrix.

**set\_transform** (*mat*)

Set rotation, transform, and velocity.

**class SkinPartition** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._SkinPartition`, `object`

**get\_mapped\_triangles** ()

Get list of triangles of this partition (mapping into the geometry data vertex list).

**get\_triangles** ()

Get list of triangles of this partition.

```
class SkinShape (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Reference to shape and skin instance.

    property shape
        The shape.

    property skin_instance
        Skinning instance for the shape?

class SkinShapeGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Unknown.

    property link_pairs
        First link is a NiTriShape object.Second link is a NiSkinInstance object.

    property num_link_pairs
        Counts unknown.

class SkinTransform (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._SkinTransform, object

    get_transform()
        Return scale, rotation, and translation into a single 4x4 matrix.

    set_transform(mat)
        Set rotation, transform, and velocity.

class SkinWeight (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A weighted vertex.

    property index
        The vertex index, in the mesh.

    property weight
        The vertex weight - between 0.0 and 1.0

class SkyObjectType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Sets what sky function this object fulfills in BSSkyShaderProperty or SkyShaderProperty.

    BSSM_SKY = 2

    BSSM_SKY_CLOUDS = 3

    BSSM_SKY_MOON_STARS_MASK = 7

    BSSM_SKY_STARS = 5

    BSSM_SKY_SUNGLARE = 1

    BSSM_SKY_TEXTURE = 0

class SkyShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderLightingProperty

    Bethesda-specific node? Found in Fallout3

    property file_name
        The texture.
```

```
property sky_object_type
    Sky Object Type

class SkyrimHavokMaterial (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    A material, used by havok shape objects in Skyrim.

    MAT_BARREL = 732141076
    MAT_BOTTLE = 493553910
    MAT_BROKEN_STONE = 131151687
    MAT_CLOTH = 3839073443
    MAT_DIRT = 3106094762
    MAT_DRAGON = 2518321175
    MAT_GLASS = 3739830338
    MAT_GRASS = 1848600814
    MAT_GRAVEL = 428587608
    MAT_HEAVY_METAL = 2229413539
    MAT_HEAVY_STONE = 1570821952
    MAT_HEAVY_WOOD = 3070783559
    MAT_ICE = 873356572
    MAT_LIGHT_WOOD = 365420259
    MAT_MATERIAL_ARMOR_HEAVY = 3708432437
    MAT_MATERIAL_ARMOR_LIGHT = 3424720541
    MAT_MATERIAL_ARROW = 3725505938
    MAT_MATERIAL_AXE_1HAND = 1305674443
    MAT_MATERIAL_BASKET = 790784366
    MAT_MATERIAL_BLADE_1HAND = 1060167844
    MAT_MATERIAL_BLADE_1HAND_SMALL = 2617944780
    MAT_MATERIAL_BLADE_2HAND = 2022742644
    MAT_MATERIAL_BLUNT_2HAND = 3969592277
    MAT_MATERIAL_BONE = 3049421844
    MAT_MATERIAL_BOOK = 1264672850
    MAT_MATERIAL_BOTTLE_SMALL = 2025794648
    MAT_MATERIAL_BOULDER_LARGE = 1885326971
    MAT_MATERIAL_BOULDER_MEDIUM = 4283869410
    MAT_MATERIAL_BOULDER_SMALL = 1550912982
    MAT_MATERIAL_BOWS_STAVES = 1607128641
    MAT_MATERIAL_CARPET = 1286705471
    MAT_MATERIAL_CERAMIC_MEDIUM = 781661019
```

```
MAT_MATERIAL_CHAIN = 3074114406
MAT_MATERIAL_CHAIN_METAL = 438912228
MAT_MATERIAL_COIN = 3589100606
MAT_MATERIAL_SHIELD_HEAVY = 3702389584
MAT_MATERIAL_SHIELD_LIGHT = 3448167928
MAT_MATERIAL_SKIN_LARGE = 2965929619
MAT_MATERIAL_SKIN_SMALL = 2632367422
MAT_MATERIAL_STONE_AS_STAIRS = 1886078335
MAT_MATERIAL_WOOD_AS_STAIRS = 1803571212
MAT_MUD = 1486385281
MAT_ORGANIC = 2974920155
MAT_SAND = 2168343821
MAT_SKIN = 591247106
MAT_SNOW = 398949039
MAT_SOLID_METAL = 1288358971
MAT_STAIRS_BROKEN_STONE = 2892392795
MAT_STAIRS_SNOW = 1560365355
MAT_STAIRS_STONE = 899511101
MAT_STAIRS_WOOD = 1461712277
MAT_STONE = 3741512247
MAT_UNKNOWN_1028101969 = 1028101969
MAT_UNKNOWN_1440721808 = 1440721808
MAT_UNKNOWN_1574477864 = 1574477864
MAT_UNKNOWN_1591009235 = 1591009235
MAT_WATER = 1024582599
MAT_WOOD = 500811281
```

```
class SkyrimLayer (**kwargs)
```

```
    Bases: pyffi.object_models.xml.enum.EnumBase
```

Physical purpose of collision object? The setting affects object's havok behavior in game. Anything higher than 47 is also null.

```
ACOUSTIC_SPACE = 21
ACTORZONE = 22
ANIMSTATIC = 2
AVOIDBOX = 34
BIPED = 8
BIPED_NO_CC = 33
CAMERAPICK = 39
```

CAMERASHPERE = 36  
CHARCONTROLLER = 30  
CLOUD\_TRAP = 16  
CLUTTER = 4  
COLLISIONBOX = 35  
CONEPROJECTILE = 38  
CUSTOMPICK1 = 43  
CUSTOMPICK2 = 44  
DEADBIP = 32  
DEBRIS\_LARGE = 20  
DEBRIS\_SMALL = 19  
DOORDETECTION = 37  
DROPPINGPICK = 46  
GASTRAP = 24  
GROUND = 17  
INVISIBLE\_WALL = 27  
ITEMPICK = 40  
LINEOFSIGHT = 41  
NONCOLLIDABLE = 15  
NULL = 47  
PATHPICK = 42  
PORTAL = 18  
PROJECTILE = 6  
PROJECTILEZONE = 23  
PROPS = 10  
SHELLCASING = 25  
SPELL = 7  
SPELLEXPLOSION = 45  
STAIRHELPER = 31  
STATIC = 1  
TERRAIN = 13  
TRANSPARENT = 3  
TRANSPARENT\_SMALL = 26  
TRANSPARENT\_SMALL\_ANIM = 28  
TRAP = 14  
TREES = 9

```
TRIGGER = 12
UNIDENTIFIED = 0
WARD = 29
WATER = 11
WEAPON = 5

class SkyrimShaderPropertyFlags1 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    property slsf_1_cast_shadows
    property slsf_1_decals
    property slsf_1_dynamic_decals
    property slsf_1_environment_mapping
    property slsf_1_external_emittance
    property slsf_1_eye_environment_mapping
    property slsf_1_face_gen_rgb_tint
    property slsf_1_facegen_detail_map
    property slsf_1_fire_refraction
    property slsf_1_greyscale_to_palette_alpha
    property slsf_1_greyscale_to_palette_color
    property slsf_1_hair_soft_lighting
    property slsf_1_landscape
    property slsf_1_localmap_hide_secret
    property slsf_1_model_space_normals
    property slsf_1_multiple_textures
    property slsf_1_non_projective_shadows
    property slsf_1_own_emit
    property slsf_1_parallax
    property slsf_1_parallax_occlusion
    property slsf_1_projected_uv
    property slsf_1_recieve_shadows
    property slsf_1_refraction
    property slsf_1_remappable_textures
    property slsf_1_screendoor_alpha_fade
    property slsf_1_skinned
    property slsf_1_soft_effect
    property slsf_1_specular
    property slsf_1_temp_refraction
```

```
    property slsf_1_use_falloff
    property slsf_1_vertex_alpha
    property slsf_1_z_buffer_test
class SkyrimShaderPropertyFlags2 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    property slsf_2_anisotropic_lighting
    property slsf_2_assume_shadowmask
    property slsf_2_back_lighting
    property slsf_2_billboard
    property slsf_2_cloud_lod
    property slsf_2_double_sided
    property slsf_2_effect_lighting
    property slsf_2_env_map_light_fade
    property slsf_2_fit_slope
    property slsf_2_glow_map
    property slsf_2_hd_lod_objects
    property slsf_2_hide_on_local_map
    property slsf_2_lod_landscape
    property slsf_2_lod_objects
    property slsf_2_multi_index_snow
    property slsf_2_multi_layer_parallax
    property slsf_2_no_fade
    property slsf_2_no_lod_land_blend
    property slsf_2_no_transparency_multisampling
    property slsf_2_packed_tangent
    property slsf_2_premult_alpha
    property slsf_2_rim_lighting
    property slsf_2_soft_lighting
    property slsf_2_tree_anim
    property slsf_2_uniform_scale
    property slsf_2_unused_01
    property slsf_2_unused_02
    property slsf_2_vertex_colors
    property slsf_2_vertex_lighting
    property slsf_2_weapon_blood
    property slsf_2_wireframe
```

```
    property slsf_2_z_buffer_write
```

```
class SkyrimWaterShaderFlags (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    property swsf_1_bypass_refraction_map
    property swsf_1_enabled
    property swsf_1_highlight_layer_toggle
    property swsf_1_unknown_0
    property swsf_1_unknown_3
    property swsf_1_unknown_4
    property swsf_1_unknown_5
    property swsf_1_water_toggle
```

```
class SolverDeactivation (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    A list of possible solver deactivation settings. This value defines how the solver deactivates objects. The solver works on a per object basis. Note: Solver deactivation does not save CPU, but reduces creeping of movable objects in a pile quite dramatically.

    SOLVER_DEACTIVATION_HIGH = 4
    SOLVER_DEACTIVATION_INVALID = 0
    SOLVER_DEACTIVATION_LOW = 2
    SOLVER_DEACTIVATION_MAX = 5
    SOLVER_DEACTIVATION_MEDIUM = 3
    SOLVER_DEACTIVATION_OFF = 1
```

```
class SortingMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    SORTING_INHERIT = 0
    SORTING_OFF = 1
```

```
class SphereBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A sphere.

    property center
        The sphere's center.

    property radius
        The sphere's radius.
```

```
class StencilAction (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    This enum defines the various actions used in conjunction with the stencil buffer. For a detailed description of the individual options please refer to the OpenGL docs.

    ACTION_DECREMENT = 4
    ACTION_INCREMENT = 3
```

```
ACTION_INVERT = 5
```

```
ACTION_KEEP = 0
```

```
ACTION_REPLACE = 2
```

```
ACTION_ZERO = 1
```

```
class StencilCompareMode (**kwargs)
```

```
Bases: pyffi.object_models.xml.enum.EnumBase
```

This enum contains the options for doing stencil buffer tests.

```
TEST_ALWAYS = 7
```

```
TEST_EQUAL = 2
```

```
TEST_GREATER = 4
```

```
TEST_GREATER_EQUAL = 6
```

```
TEST_LESS = 1
```

```
TEST_LESS_EQUAL = 3
```

```
TEST_NEVER = 0
```

```
TEST_NOT_EQUAL = 5
```

```
class StiffSpringDescriptor (template=None, argument=None, parent=None)
```

```
Bases: pyffi.object_models.xml.struct_.StructBase
```

```
property length
```

```
Length.
```

```
property pivot_a
```

```
Pivot A.
```

```
property pivot_b
```

```
Pivot B.
```

```
StringIndex
```

```
alias of pyffi.object_models.common.UInt
```

```
class StringOffset (**kwargs)
```

```
Bases: pyffi.object_models.common.Int
```

This is just an integer with -1 as default value.

```
class StringPalette (template=None, argument=None, parent=None)
```

```
Bases: pyffi.formats.nif._StringPalette, object
```

```
add_string (text)
```

Adds string to palette (will recycle existing strings if possible) and return offset to the string in the palette.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> pal.add_string("")
```

(continues on next page)

(continued from previous page)

```
-1
>>> print(pal.get_string(4).decode("ascii"))
def
```

**clear()**

Clear all strings in the palette.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
>>> print(repr(pal.palette.decode("ascii")).lstrip("u"))
'abc\x00def\x00'
>>> pal.clear()
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
>>> print(repr(pal.palette.decode("ascii")).lstrip("u"))
''
```

**get\_all\_strings()**

Return a list of all strings.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> for x in pal.get_all_strings():
...     print(x.decode("ascii"))
abc
def
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
>>> print(repr(pal.palette.decode("ascii")).lstrip("u"))
'abc\x00def\x00'
```

**get\_string(offset)**

Return string at given offset.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> print(pal.get_string(0).decode("ascii"))
abc
>>> print(pal.get_string(4).decode("ascii"))
def
>>> pal.get_string(5)
pyffi.nif.stringpalette:WARNING:StringPalette: no string starts at offset_
↪5 (string is b'ef', preceding character is b'd')
b'ef'
>>> pal.get_string(100)
```

(continues on next page)

(continued from previous page)

```
Traceback (most recent call last):
...
ValueError: ...
```

```
class SubConstraint (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.object_models.xml.struct_.StructBase
```

```
    property ball_and_socket
```

```
    property entities
```

```
        Usually NONE. The entities affected by this constraint.
```

```
    property hinge
```

```
    property limited_hinge
```

```
    property num_entities
```

```
        Usually 2. Number of bodies affected by this constraint.
```

```
    property priority
```

```
        Usually 1. Higher values indicate higher priority of this constraint?
```

```
    property prismatic
```

```
    property ragdoll
```

```
    property stiff_spring
```

```
    property type
```

```
        Type of constraint.
```

```
class SymmetryType (**kwargs)
```

```
    Bases: pyffi.object_models.xml.enum.EnumBase
```

```
    Determines symmetry type used by NiPSysBombModifier.
```

```
    CYLINDRICAL_SYMMETRY = 1
```

```
    PLANAR_SYMMETRY = 2
```

```
    SPHERICAL_SYMMETRY = 0
```

```
class SyncPoint (**kwargs)
```

```
    Bases: pyffi.object_models.xml.enum.EnumBase
```

```
    Specifies the time when an application must synchronize for some reason.
```

```
    SYNC_ANY = 32768
```

```
    SYNC_PHYSICS_COMPLETED = 32864
```

```
    SYNC_PHYSICS_SIMULATE = 32848
```

```
    SYNC_POST_UPDATE = 32800
```

```
    SYNC_REFLECTIONS = 32880
```

```
    SYNC_RENDER = 32832
```

```
    SYNC_UPDATE = 32784
```

```
    SYNC_VISIBLE = 32816
```

```
class TBC (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.object_models.xml.struct_.StructBase
```

Tension, bias, continuity.

**property b**  
Bias.

**property c**  
Continuity.

**property t**  
Tension.

**class TallGrassShaderProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.BSShaderProperty`

Bethesda-specific node.

**property file\_name**  
Texture file name

**class TargetColor** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Used by `NiPoint3InterpControllers` to select which type of color in the controlled object that will be animated.

**TC\_AMBIENT = 0**

**TC\_DIFFUSE = 1**

**TC\_SELF\_ILLUM = 3**

**TC\_SPECULAR = 2**

**class TexClampMode** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Specifies the available texture clamp modes. That is, the behavior of pixels outside the range of the texture.

**CLAMP\_S\_CLAMP\_T = 0**

**CLAMP\_S\_WRAP\_T = 1**

**WRAP\_S\_CLAMP\_T = 2**

**WRAP\_S\_WRAP\_T = 3**

**class TexCoord** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._TexCoord, object`

**as\_list()**

**normalize()**

**class TexDesc** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Texture description.

**property center\_offset**  
The offset from the origin?

**property clamp\_mode**  
0=clamp S clamp T, 1=clamp S wrap T, 2=wrap S clamp T, 3=wrap S wrap T

**property filter\_mode**  
0=nearest, 1=bilinear, 2=trilinear, 3=..., 4=..., 5=...

**property flags**

Texture mode flags; clamp and filter mode stored in upper byte with 0xYZ00 = clamp mode Y, filter mode Z.

**property has\_texture\_transform**

Determines whether or not the texture's coordinates are transformed.

**property ps\_2\_k**

PS2 only; from the Freedom Force docs,"The K value is used as an offset into the mipmap levels and can range from -2047 to 2047. Positive values push the mipmap towards being blurry and negative values make the mipmap sharper."-75 for most v4.0.0.2 meshes.

**property ps\_2\_l**

PS2 only; from the Freedom Force docs,"L values can range from 0 to 3 and are used to specify how fast a texture gets blurry".

**property source**

NiSourceTexture object index.

**property tiling**

The number of times the texture is tiled in each direction?

**property transform\_type**

The texture transform type? Doesn't seem to do anything.

**property translation**

The amount to translate the texture coordinates in each direction?

**property unknown\_1**

Unknown, 0 or 0x0101?

**property unknown\_short**

Unknown, seems to always be 1

**property uv\_set**

The texture coordinate set in NiGeometryData that this texture slot will use.

**property w\_rotation**

2D Rotation of texture image around third W axis after U and V.

**class TexFilterMode** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Specifies the available texture filter modes. That is, the way pixels within a texture are blended together when textures are displayed on the screen at a size other than their original dimentions.

**FILTER\_BILERP** = 1

**FILTER\_BILERP\_MIPNEAREST** = 5

**FILTER\_NEAREST** = 0

**FILTER\_NEAREST\_MIPLERP** = 4

**FILTER\_NEAREST\_MIPNEAREST** = 3

**FILTER\_TRILERP** = 2

**class TexSource** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A texture source.

**property file\_name**

The original source filename of the image embedded by the referred NiPixelData object.

**property pixel\_data**  
Pixel data object index.

**property unknown\_byte**  
Unknown.

**property unknown\_link**  
Unknown.

**property use\_external**  
Is the texture external?

**class TexTransform** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Determines how a NiTextureTransformController animates the UV coordinates.

**TT\_ROTATE** = 2

**TT\_SCALE\_U** = 3

**TT\_SCALE\_V** = 4

**TT\_TRANSLATE\_U** = 0

**TT\_TRANSLATE\_V** = 1

**class TexType** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

The type of texture.

**BASE\_MAP** = 0

**BUMP\_MAP** = 5

**DARK\_MAP** = 1

**DECAL\_0\_MAP** = 8

**DECAL\_1\_MAP** = 9

**DECAL\_2\_MAP** = 10

**DECAL\_3\_MAP** = 11

**DETAIL\_MAP** = 2

**GLOSS\_MAP** = 3

**GLOW\_MAP** = 4

**NORMAL\_MAP** = 6

**UNKNOWN2\_MAP** = 7

**class TileShaderProperty** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.BSShaderLightingProperty`

Bethesda-specific node.

**property file\_name**  
Texture file name

**class Triangle** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

List of three vertex indices.

```

property v_1
    First vertex index.

property v_2
    Second vertex index.

property v_3
    Third vertex index.

class UnionBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

property bounding_volumes
    Bounding Volume.

property num_bv
    Number of Bounding Volumes.

class Vector3 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Vector3, object

as_list()

as_tuple()

assign(vec)
    Set this vector to values from another object that supports iteration or x,y,z properties

crossproduct(x)

get_copy()

norm(sqrt=<built-in function sqrt>)

normalize(ignore_error=False, sqrt=<built-in function sqrt>)

normalized(ignore_error=False)

class Vector4 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Vector4, object

```

```

>>> from pyffi.formats.nif import NifFormat
>>> vec = NifFormat.Vector4()
>>> vec.x = 1.0
>>> vec.y = 2.0
>>> vec.z = 3.0
>>> vec.w = 4.0
>>> print(vec)
[ 1.000 2.000 3.000 4.000 ]
>>> vec.as_list()
[1.0, 2.0, 3.0, 4.0]
>>> vec.as_tuple()
(1.0, 2.0, 3.0, 4.0)
>>> print(vec.get_vector_3())
[ 1.000 2.000 3.000 ]
>>> vec2 = NifFormat.Vector4()
>>> vec == vec2
False
>>> vec2.x = 1.0
>>> vec2.y = 2.0
>>> vec2.z = 3.0
>>> vec2.w = 4.0

```

(continues on next page)

```
>>> vec == vec2
True
```

```

as_list()
as_tuple()
get_copy()
get_vector_3()

class VelocityType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Controls the way the a particle mesh emitter determines the starting speed and direction of the particles
    that are emitted.

    VELOCITY_USE_DIRECTION = 2
    VELOCITY_USE_NORMALS = 0
    VELOCITY_USE_RANDOM = 1

class VertMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, which describes how to apply vertex colors.

    VERT_MODE_SRC_AMB_DIF = 2
    VERT_MODE_SRC_EMISSIVE = 1
    VERT_MODE_SRC_IGNORE = 0

class VolumetricFogShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty

    Bethesda-specific node.

class WaterShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty

    Bethesda-specific node? Found in Fallout3

class ZCompareMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    This enum contains the options for doing z buffer tests.

    ZCOMP_ALWAYS = 0
    ZCOMP_EQUAL = 2
    ZCOMP_GREATER = 4
    ZCOMP_GREATER_EQUAL = 6
    ZCOMP_LESS = 1
    ZCOMP_LESS_EQUAL = 3
    ZCOMP_NEVER = 7
    ZCOMP_NOT_EQUAL = 5

```

---

```

class bhkAabbPhantom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShapePhantom

    Bethesda-specific node.

    property unknown_ints_1

class bhkBallAndSocketConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkConstraint

    A Ball and Socket Constraint.

    property ball_and_socket
        Describes a ball and socket constraint

class bhkBallSocketConstraintChain (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    A Ball and Socket Constraint chain.

    property floats_1
        Unknown

    property links
        Unknown

    property links_2
        Unknown

    property num_floats
        Unknown

    property num_links
        Number of links in the chain

    property num_links_2
        Number of links in the chain

    property unknown_float_1
        Unknown

    property unknown_float_2
        Unknown

    property unknown_int_1
        Unknown

    property unknown_int_2
        Unknown

    property unknown_int_3
        Unknown

class bhkBlendCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkCollisionObject

    Unknown.

    property unknown_float_1
        Blending parameter?

    property unknown_float_2
        Another blending parameter?

```

```
class bhkBlendController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Unknown. Is apparently only used in skeleton.nif files.

    property unknown_int
        Seems to be always zero.

class bhkBoxShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkBoxShape, object

    apply_scale (scale)
        Apply scale factor C{scale} on data.

    get_mass_center_inertia (density=1, solid=True)
        Return mass, center, and inertia tensor.

class bhkBreakableConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkConstraint

    A breakable constraint.

    property remove_if_broken
        Unknown

    property sub_constraint
        Constraint within constraint.

    property threshold
        Amount of force to break the rigid bodies apart?

class bhkBvTreeShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape

    A tree-like Havok data structure stored in an assembly-like binary code?

class bhkCMSDBigTris (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Triangle indices used in pair with “Big Verts” in a bhkCompressedMeshShapeData.

    property triangle_1
    property triangle_2
    property triangle_3
    property unknown_int_1
        Always 0?

    property unknown_short_1

class bhkCMSDChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Defines subshape chunks in bhkCompressedMeshShapeData

    property indices
    property indices_2
        Compressed

    property material_index
        Index of material in bhkCompressedMeshShapeData::Chunk Materials

    property num_indices
```

**property num\_indices\_2**  
Number of

**property num\_strips**  
Number of compressed strips

**property num\_vertices**  
Number of compressed vertices

**property strips**  
Compressed strips

**property transform\_index**  
Index of transformation in `bhkCompressedMeshShapeData::Chunk Transforms`

**property translation**  
Local translation

**property unknown\_short\_1**  
Always 65535?

**property vertices**  
Compressed vertices

**class bhkCMSDMaterial** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`  
Per-chunk material, used in `bhkCompressedMeshShapeData`

**property byte\_set\_to\_0**  
Always set to 0. It is only remainder of “Layer” 32-bit integer above.

**property layer**  
Copy of Layer from `bhkRigidBody`. The value is stored as 32-bit integer.

**property material**  
Material.

**property short\_set\_to\_0**  
Always set to 0. It is only remainder of “Layer” 32-bit integer above.

**class bhkCMSDTransform** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.struct_.StructBase`  
A set of transformation data: translation and rotation

**property rotation**  
Rotation. Reference point for rotation is `bhkRigidBody` translation.

**property translation**  
A vector that moves the chunk by the specified amount. W is not used.

**class bhkCapsuleShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._bhkCapsuleShape, object`

**apply\_scale** (*scale*)  
Apply scale factor <scale> on data.

**get\_mass\_center\_inertia** (*density=1, solid=True*)  
Return mass, center, and inertia tensor.

**class bhkCollisionObject** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.bhkNiCollisionObject`  
Havok related collision object?

**class bhkCompressedMeshShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkShape`

Compressed collision mesh.

**property data**

The collision mesh data.

**property radius**

A shell with that radius is added around the shape.

**property scale**

Scale

**property target**

Points to root node?

**property unknown\_4\_bytes**

Unknown.

**property unknown\_float\_1**

Unknown.

**property unknown\_float\_3**

Unknown

**property unknown\_float\_4**

Unknown

**property unknown\_float\_5**

Unknown

**property unknown\_floats\_1**

Unknown

**property unknown\_int\_1**

Unknown.

**class bhkCompressedMeshShapeData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkRefObject`

A compressed mesh shape for collision in Skyrim.

**property big\_tris**

Unknown

**property big\_verts**

Compressed Vertices?

**property bits\_per\_index**

Number of bits in the shape-key reserved for a triangle index

**property bits\_per\_w\_index**

Number of bits in the shape-key reserved for a triangle index and its winding

**property bounds\_max**

The maximum boundary of the AABB (the coordinates of the corner with the highest numerical values)

**property bounds\_min**

The minimum boundary of the AABB (the coordinates of the corner with the lowest numerical values)

**property chunk\_materials**

Table (array) with sets of materials. Chunks refers to this table by index.

---

**property chunk\_transforms**  
Table (array) with sets of transformations. Chunks refers to this table by index.

**property chunks**

**property error**  
The radius of the storage mesh shape? Quantization error?

**property mask\_index**  
131071 = 0x1ffff  
**Type** Mask used to get the triangle index from a shape-key (common)

**property mask\_w\_index**  
262143 = 0x3ffff  
**Type** Mask used to get the triangle index and winding from a shape-key (common)

**property num\_big\_tris**  
Unknown

**property num\_big\_verts**  
Unknown

**property num\_chunks**  
Unknown

**property num\_materials**  
Number of chunk materials

**property num\_transforms**  
Number of chunk transformations

**property unknown\_byte\_1**  
Unknown

**property unknown\_byte\_2**  
Unknown

**property unknown\_int\_12**  
Unknown, end of block.

**property unknown\_int\_3**  
Unknown

**property unknown\_int\_4**  
Unknown

**property unknown\_int\_5**  
Unknown

**property unknown\_int\_6**  
Unknown

**class bhkConstraint** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._bhkConstraint, object`

**get\_transform\_a\_b** (*parent*)  
Returns the transform of the first entity relative to the second entity. Root is simply a nif block that is a common parent to both blocks.

**class bhkConvexListShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.bhkShape`

A havok shape. A list of convex shapes. Do not put a `bhkPackedNiTriStripsShape` in the Sub Shapes. Use a separate collision nodes without a list shape for those. Also, shapes collected in a `bhkListShape` may not have the correct walking noise, so only use it for non-walkable objects.

**property material**

The material of the shape.

**property num\_sub\_shapes**

The number of sub shapes referenced.

**property sub\_shapes**

List of shapes.

**property unknown\_byte\_1**

Unknown Flag

**property unknown\_float\_1**

Unknown Flag

**property unknown\_floats**

Unknown. Set to (0.0,0.0,-0.0,0.0,0.0,-0.0), where -0.0 is 0x80000000 in hex.

**class bhkConvexShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkSphereRepShape`

A havok shape.

**class bhkConvexTransformShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkTransformShape`

A convex transformed shape?

**class bhkConvexVerticesShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._bhkConvexVerticesShape, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**get\_mass\_center\_inertia** (*density=1, solid=True*)

Return mass, center, and inertia tensor.

**class bhkEntity** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkWorldObject`

A havok node, describes physical properties.

**class bhkHingeConstraint** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkConstraint`

A hinge constraint.

**property hinge**

Hinge constraining.

**class bhkLimitedHingeConstraint** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._bhkLimitedHingeConstraint, object`

**apply\_scale** (*scale*)

Scale data.

**update\_a\_b** (*parent*)

Update the B data from the A data. The parent argument is simply a common parent to the entities.

```

class bhkLiquidAction (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable
    Bethesda-specific node.

    property unknown_float_1
        Unknown Flag

    property unknown_float_2
        Unknown Flag

    property unknown_float_3
        Unknown Flag

    property unknown_float_4
        Unknown Flag

    property unknown_int_1
        Unknown Flag

    property unknown_int_2
        Unknown Flag

    property unknown_int_3
        Unknown Flag

class bhkListShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkListShape, object

    add_shape (shape, front=False)
        Add shape to list.

    get_mass_center_inertia (density=1, solid=True)
        Return center of gravity and area.

    remove_shape (shape)
        Remove a shape from the shape list.

class bhkMalleableConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMalleableConstraint, object

    apply_scale (scale)
        Scale data.

    update_a_b (parent)
        Update the B data from the A data.

class bhkMeshShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape

    property num_strips_data
        The number of strips data objects referenced.

    property num_unknown_floats

    property strips_data
        Refers to a bunch of NiTriStripsData objects that make up this shape.

    property unknown_1

    property unknown_2

    property unknown_floats

```

```
class bhkMoppBvTreeShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMoppBvTreeShape, object

get_mass_center_inertia (density=1, solid=True)
    Return mass, center of gravity, and inertia tensor.

mopp_from_tree (tree)

parse_mopp (start=0, depth=0, toffset=0, verbose=False)
    The mopp data is printed to the debug channel while parsed. Returns list of indices into mopp data of
    the bytes processed and a list of triangle indices encountered.

    The verbose argument is ignored (and is deprecated).

split_triangles (ts, bbox, dir=0)
    Direction 0=X, 1=Y, 2=Z

update_mopp ()
    Update the MOPP data, scale, and origin, and welding info.

    @deprecated: use update_mopp_welding instead

update_mopp_welding ()
    Update the MOPP data, scale, and origin, and welding info.

update_origin_scale ()
    Update scale and origin.

class bhkMultiSphereShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMultiSphereShape, object

get_mass_center_inertia (density=1, solid=True)
    Return center of gravity and area.

class bhkNiCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiCollisionObject

    Havok related collision object?

property body
    Links to the collision object data

property flags
    0=Active 2=Notify 3=Set Local 6=Reset.
    Type Set to 1 for most objects, and to 41 for animated objects (ANIM_STATIC) Bits

class bhkNiTriStripsShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkNiTriStripsShape, object

get_interchangeable_packed_shape ()
    Returns a bhkPackedNiTriStripsShape block that is geometrically interchangeable.

get_mass_center_inertia (density=1, solid=True)
    Return mass, center, and inertia tensor.

class bhkOrientHingedBodyAction (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    Bethesda-Specific node.

property unknown_ints_1

class bhkPCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkNiCollisionObject
```

Unknown.

**class bhkPackedNiTriStripsShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._bhkPackedNiTriStripsShape, object`

**add\_shape** (*triangles, normals, vertices, layer=0, material=0*)

Pack the given geometry.

**get\_mass\_center\_inertia** (*density=1, solid=True*)

Return mass, center, and inertia tensor.

**get\_sub\_shapes** ()

Return sub shapes (works for both Oblivion and Fallout 3).

**get\_triangle\_hash\_generator** ()

Generator which produces a tuple of integers, or None in degenerate case, for each triangle to ease detection of duplicate triangles.

```
>>> shape = NifFormat.bhkPackedNiTriStripsShape()
>>> data = NifFormat.hkPackedNiTriStripsData()
>>> shape.data = data
>>> data.num_triangles = 6
>>> data.triangles.update_size()
>>> data.triangles[0].triangle.v_1 = 0
>>> data.triangles[0].triangle.v_2 = 1
>>> data.triangles[0].triangle.v_3 = 2
>>> data.triangles[1].triangle.v_1 = 2
>>> data.triangles[1].triangle.v_2 = 1
>>> data.triangles[1].triangle.v_3 = 3
>>> data.triangles[2].triangle.v_1 = 3
>>> data.triangles[2].triangle.v_2 = 2
>>> data.triangles[2].triangle.v_3 = 1
>>> data.triangles[3].triangle.v_1 = 3
>>> data.triangles[3].triangle.v_2 = 1
>>> data.triangles[3].triangle.v_3 = 2
>>> data.triangles[4].triangle.v_1 = 0
>>> data.triangles[4].triangle.v_2 = 0
>>> data.triangles[4].triangle.v_3 = 3
>>> data.triangles[5].triangle.v_1 = 1
>>> data.triangles[5].triangle.v_2 = 3
>>> data.triangles[5].triangle.v_3 = 4
>>> list(shape.get_triangle_hash_generator())
[(0, 1, 2), (1, 3, 2), (1, 3, 2), (1, 2, 3), None, (1, 3, 4)]
```

**Returns** A generator yielding a hash value for each triangle.

**get\_vertex\_hash\_generator** (*vertexprecision=3, subshape\_index=None*)

Generator which produces a tuple of integers for each vertex to ease detection of duplicate/close enough to remove vertices. The precision parameter denote number of significant digits behind the comma.

For vertexprecision, 3 seems usually enough (maybe we'll have to increase this at some point).

```
>>> shape = NifFormat.bhkPackedNiTriStripsShape()
>>> data = NifFormat.hkPackedNiTriStripsData()
>>> shape.data = data
>>> shape.num_sub_shapes = 2
>>> shape.sub_shapes.update_size()
>>> data.num_vertices = 3
>>> shape.sub_shapes[0].num_vertices = 2
```

(continues on next page)

(continued from previous page)

```

>>> shape.sub_shapes[1].num_vertices = 1
>>> data.vertices.update_size()
>>> data.vertices[0].x = 0.0
>>> data.vertices[0].y = 0.1
>>> data.vertices[0].z = 0.2
>>> data.vertices[1].x = 1.0
>>> data.vertices[1].y = 1.1
>>> data.vertices[1].z = 1.2
>>> data.vertices[2].x = 2.0
>>> data.vertices[2].y = 2.1
>>> data.vertices[2].z = 2.2
>>> list(shape.get_vertex_hash_generator())
[(0, (0, 100, 200)), (0, (1000, 1100, 1200)), (1, (2000, 2100, 2200))]
>>> list(shape.get_vertex_hash_generator(subshape_index=0))
[(0, 100, 200), (1000, 1100, 1200)]
>>> list(shape.get_vertex_hash_generator(subshape_index=1))
[(2000, 2100, 2200)]

```

**Parameters** `vertexprecision` (*float*) – Precision to be used for vertices.

**Returns** A generator yielding a hash value for each vertex.

**class** `bhkPhantom` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkWorldObject`

Havok object that do not react with other objects when they collide (causing deflection, etc.) but still trigger collision notifications to the game. Possible uses are traps, portals, AI fields, etc.

**class** `bhkPrismaticConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkConstraint`

A prismatic constraint.

**property** `prismatic`

Describes a prismatic constraint

**class** `bhkRDTConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

**property** `entity_a`

Entity A in this constraint.

**property** `entity_b`

Entity B in this constraint.

**property** `malleable_constraint`

**property** `priority`

Usually 1. Higher values indicate higher priority of this constraint?

**property** `ragdoll`

**property** `type`

Type of constraint. 7 = RagDoll Constraint? 13 = Malleable Constraint?

**property** `unknown_int`

Unknown. Usually 2.

**class** `bhkRDTMalleableConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A malleable constraint.

**property damping**

**property entity\_a**  
Usually -1?

**property entity\_b**  
Usually -1?

**property hinge**

**property limited\_hinge**

**property priority**  
Usually 1. Higher values indicate higher priority of this constraint?

**property ragdoll**

**property type**  
Type of constraint.

**property unknown\_int**  
Unknown. Usually 2.

**class bhkRagdollConstraint** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._bhkRagdollConstraint, object`

**apply\_scale** (*scale*)  
Scale data.

**update\_a\_b** (*parent*)  
Update the B data from the A data.

**class bhkRagdollTemplate** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Found in Fallout 3, more ragdoll info? (`meshesragdollconstraint*.rdt`)

**property bones**  
Bones in index

**property name**

**property num\_bones**  
Number of target bones

**class bhkRagdollTemplateData** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.NiObject`  
Data for `bhkRagdollTemplate`

**property constraint**  
Unknown

**property flag\_or\_num\_constraints**  
a Constraint is present.  
**Type** Either a flag or a number of constraints.0  
**Type** no Constraint is present.1

**property friction**  
Probably a Friction for `bhkRigidBody` linked to this bone node.

**property mass**  
Probably a Mass for `bhkRigidBody` linked to this bone node.

**property name**

**property radius**

Probably a Radius for collision object shape of bhkRigidBody linked to this bone node.

**property restitution**

Probably a Restitution for bhkRigidBody linked to this bone node.

**property unknown\_int**

Unknown. Dependent on value of User Version 2? Value 7 found in Fallout3 meshesragdollconstraint-default.rdt. This file has User Version 2 = 34. Value 0 found in Fallout3 meshesragdollconstraints-tiff.rdt. This file has User Version 2 = 16.

**class bhkRefObject** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._bhkRefObject`, `object`

**get\_shape\_mass\_center\_inertia** (*density=1, solid=True*)

Return mass, center of gravity, and inertia tensor of this object's shape, if self.shape is not None.

If self.shape is None, then returns zeros for everything.

**class bhkRigidBody** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._bhkRigidBody`, `object`

**apply\_scale** (*scale*)

Apply scale factor <scale> on data.

**update\_mass\_center\_inertia** (*density=1, solid=True, mass=None*)

Look at all the objects under this rigid body and update the mass, center of gravity, and inertia tensor accordingly. If the C{mass} parameter is given then the C{density} argument is ignored.

**class bhkRigidBodyT** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkRigidBody`

Unknown.

**class bhkSPCollisionObject** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkPCollisionObject`

Unknown.

**class bhkSerializable** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkRefObject`

Havok objects that can be saved and loaded from disk?

**class bhkShape** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkSerializable`

A Havok Shape?

**class bhkShapeCollection** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkShape`

Havok collision object that uses multiple shapes?

**class bhkShapePhantom** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkPhantom`

A Havok phantom that uses a Havok shape object for its collision volume instead of just a bounding box.

**class bhkSimpleShapePhantom** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkShapePhantom`

Unknown shape.

**property unknown\_float**  
Unknown.

**property unknown\_floats\_2**  
Unknown. (1,0,0,0,0) x 3.

**property unknown\_floats**  
Unknown.

**class bhkSphereRepShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.bhkShape`

A havok shape, perhaps with a bounding sphere for quick rejection in addition to more detailed shape data?

**property material**  
The material of the shape.

**property radius**  
The radius of the sphere that encloses the shape.

**class bhkSphereShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._bhkSphereShape, object`

**apply\_scale** (*scale*)  
Apply scale factor <scale> on data.

**get\_mass\_center\_inertia** (*density=1, solid=True*)  
Return mass, center, and inertia tensor.

**class bhkStiffSpringConstraint** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.bhkConstraint`

A spring constraint.

**property stiff\_spring**  
Stiff Spring constraint.

**class bhkTransformShape** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif._bhkTransformShape, object`

**apply\_scale** (*scale*)  
Apply scale factor <scale> on data.

**get\_mass\_center\_inertia** (*density=1, solid=True*)  
Return mass, center, and inertia tensor.

**class bhkWorldObject** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.nif.bhkSerializable`

Havok objects that have a position in the world?

**property havok\_col\_filter**

**property shape**  
Link to the body for this collision object.

**class bool** (*\*\*kwargs*)  
Bases: `pyffi.object_models.xml.basic.BasicBase, pyffi.object_models.editable.EditableBoolComboBox`

Basic implementation of a 32-bit (8-bit for versions > 4.0.0.2) boolean type.

```
>>> i = NifFormat.bool()
>>> i.set_value('false')
>>> i.get_value()
False
>>> i.set_value('true')
>>> i.get_value()
True
```

**get\_hash** (*data=None*)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)

Returns size of the object in bytes.

**get\_value** ()

Return object value.

**read** (*stream, data*)

Read object from file.

**set\_value** (*value*)

Set object value.

**write** (*stream, data*)

Write object to file.

**byte**

alias of `pyffi.object_models.common.UByte`

**char**

alias of `pyffi.object_models.common.Char`

**float**

alias of `pyffi.object_models.common.Float`

**games** = {'?': [167772419], 'Atlantica': [335675400], 'Axis and Allies': [167837696]}

**class hkConstraintType** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

The type of constraint.

**BallAndSocket** = 0

**Hinge** = 1

**Prismatic** = 6

**Ragdoll** = 7

**StiffSpring** = 8

**class hkPackedNiTriStripsData** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._hkPackedNiTriStripsData, object`

**apply\_scale** (*scale*)

Apply scale factor on data.

**class hkResponseType** (*\*\*kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

**RESPONSE\_INVALID** = 0

**RESPONSE\_NONE** = 3

```

RESPONSE_REPORTING = 2

RESPONSE_SIMPLE_CONTACT = 1

class hkTriangle (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A triangle with extra data used for physics.

    property normal
        This is the triangle's normal.

    property triangle
        The triangle.

    property welding_info
        Additional havok information on how triangles are welded.

int
    alias of pyffi.object_models.common.Int

class physXMaterialRef (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property material_desc
        PhysX Material Description

    property number
        Unknown

    property unknown_byte_1
        Unknown

short
    alias of pyffi.object_models.common.Short

class string (**kwargs)
    Bases: pyffi.object_models.common.SizedString

    get_hash (data=None)
        Return a hash value for this string.
        Returns An immutable object that can be used as a hash.

    get_size (data=None)
        Return number of bytes this type occupies in a file.
        Returns Number of bytes.

    get_strings (data)
        Return all strings used by this object.

    read (stream, data)
        Read string from stream.
        Parameters stream (file) – The stream to read from.

    write (stream, data)
        Write string to stream.
        Parameters stream (file) – The stream to write to.

uint
    alias of pyffi.object_models.common.UInt

ulittle32
    alias of pyffi.object_models.common.ULittle32

```

**ushort**

alias of `pyffi.object_models.common.ushort`

**static version\_number** (*version\_str*)

Converts version string into an integer.

**Parameters** `version_str` (*str*) – The version string.

**Returns** A version integer.

```
>>> hex(NifFormat.version_number('3.14.15.29'))
'0x30e0f1d'
>>> hex(NifFormat.version_number('1.2'))
'0x1020000'
>>> hex(NifFormat.version_number('3.03'))
'0x3000300'
>>> hex(NifFormat.version_number('NS'))
'0xa010000'
```

```
versions = {'10.0.1.0': 167772416, '10.0.1.2': 167772418, '10.0.1.3': 167772419, '1
```

```
xml_alias = []
```

```
xml_bit_struct = [<bit_struct 'BSSegmentFlags'>, <bit_struct 'FurnitureEntryPoints'>, ]
```

```
xml_enum = [<enum 'AlphaFormat'>, <enum 'ApplyMode'>, <enum 'TexType'>, <enum 'KeyType
```

```
xml_file_name = 'nif.xml'
```

```
xml_file_path = [None, '/home/docs/checkouts/readthedocs.org/user_builds/pyffi-tagnum/
```

```
xml_struct = [<struct 'Color3'>, <struct 'ByteColor3'>, <struct 'Color4'>, <struct 'By
```

## Regression tests

These tests are used to check for functionality and bugs in the library. They also provide code examples which you may find useful.

## Read a NIF file

```
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'spells', 'nif', 'files')
>>> stream = open(os.path.join(format_root, 'test.nif'), 'rb')
>>> data = NifFormat.Data()
>>> # inspect is optional; it will not read the actual blocks
>>> data.inspect(stream)
>>> hex(data.version)
'0x14010003'
>>> data.user_version
0
>>> for blocktype in data.header.block_types:
...     print(blocktype.decode("ascii"))
NiNode
NiTriShape
```

(continues on next page)

(continued from previous page)

```

NiTriShapeData
>>> data.roots # blocks have not been read yet, so this is an empty list
[]
>>> data.read(stream)
>>> for root in data.roots:
...     for block in root.tree():
...         if isinstance(block, NifFormat.NiNode):
...             print(block.name.decode("ascii"))
test
>>> stream.close()

```

### Parse all NIF files in a directory tree

```

>>> for stream, data in NifFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-5:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/spells/nif/files/invalid.nif
Warning: read failed due corrupt file, corrupt format description, or bug.
reading tests/spells/nif/files/nds.nif
reading tests/spells/nif/files/neosteam.nif
reading tests/spells/nif/files/test.nif
reading tests/spells/nif/files/test_centerradius.nif
reading tests/spells/nif/files/test_check_tangentspace1.nif
reading tests/spells/nif/files/test_check_tangentspace2.nif
reading tests/spells/nif/files/test_check_tangentspace3.nif
reading tests/spells/nif/files/test_check_tangentspace4.nif
reading tests/spells/nif/files/test_convexverticesshape.nif
reading tests/spells/nif/files/test_dump_tex.nif
reading tests/spells/nif/files/test_fix_clampmaterialalpha.nif
reading tests/spells/nif/files/test_fix_cleanstringpalette.nif
reading tests/spells/nif/files/test_fix_detachhavoktristripsdata.nif
reading tests/spells/nif/files/test_fix_disableparallax.nif
reading tests/spells/nif/files/test_fix_ffvt3rskinpartition.nif
reading tests/spells/nif/files/test_fix_mergeskeletonroots.nif
reading tests/spells/nif/files/test_fix_tangentspace.nif
reading tests/spells/nif/files/test_fix_texturepath.nif
reading tests/spells/nif/files/test_grid_128x128.nif
reading tests/spells/nif/files/test_grid_64x64.nif
reading tests/spells/nif/files/test_mopp.nif
reading tests/spells/nif/files/test_opt_collision_complex_mopp.nif
reading tests/spells/nif/files/test_opt_collision_mopp.nif
reading tests/spells/nif/files/test_opt_collision_packed.nif
reading tests/spells/nif/files/test_opt_collision_to_boxshape.nif
reading tests/spells/nif/files/test_opt_collision_to_boxshape_notabox.nif
reading tests/spells/nif/files/test_opt_collision_unpacked.nif
reading tests/spells/nif/files/test_opt_delunusedbones.nif

```

reading tests/spells/nif/files/test\_opt\_dupverts.nif reading tests/spells/nif/files/test\_opt\_emptyproperties.nif reading tests/spells/nif/files/test\_opt\_grid\_layout.nif reading tests/spells/nif/files/test\_opt\_mergeduplicates.nif reading tests/spells/nif/files/test\_opt\_vertex\_cache.nif reading tests/spells/nif/files/test\_opt\_zeroscale.nif reading tests/spells/nif/files/test\_skincenterradius.nif reading tests/spells/nif/files/test\_vertexcolor.nif

### Create a NIF model from scratch and write to file

```

>>> root = NifFormat.NiNode()
>>> root.name = 'Scene Root'
>>> blk = NifFormat.NiNode()
>>> root.add_child(blk)
>>> blk.name = 'new block'
>>> blk.scale = 2.4
>>> blk.translation.x = 3.9
>>> blk.rotation.m_11 = 1.0
>>> blk.rotation.m_22 = 1.0
>>> blk.rotation.m_33 = 1.0
>>> ctrl = NifFormat.NiVisController()
>>> ctrl.flags = 0x000c
>>> ctrl.target = blk
>>> blk.add_controller(ctrl)
>>> blk.add_controller(NifFormat.NiAlphaController())
>>> strips = NifFormat.NiTriStrips()
>>> root.add_child(strips, front = True)
>>> strips.name = "hello world"
>>> strips.rotation.m_11 = 1.0
>>> strips.rotation.m_22 = 1.0
>>> strips.rotation.m_33 = 1.0
>>> data = NifFormat.NiTriStripsData()
>>> strips.data = data
>>> data.num_vertices = 5
>>> data.has_vertices = True
>>> data.vertices.update_size()
>>> for i, v in enumerate(data.vertices):
...     v.x = 1.0+i/10.0
...     v.y = 0.2+1.0/(i+1)
...     v.z = 0.03
>>> data.update_center_radius()
>>> data.num_strips = 2
>>> data.strip_lengths.update_size()
>>> data.strip_lengths[0] = 3
>>> data.strip_lengths[1] = 4
>>> data.has_points = True
>>> data.points.update_size()
>>> data.points[0][0] = 0
>>> data.points[0][1] = 1
>>> data.points[0][2] = 2
>>> data.points[1][0] = 1
>>> data.points[1][1] = 2
>>> data.points[1][2] = 3
>>> data.points[1][3] = 4
>>> data.num_uv_sets = 1
>>> data.has_uv = True
>>> data.uv_sets.update_size()
>>> for i, v in enumerate(data.uv_sets[0]):
...     v.u = 1.0-i/10.0

```

(continues on next page)

(continued from previous page)

```

...     v.v = 1.0/(i+1)
>>> data.has_normals = True
>>> data.normals.update_size()
>>> for i, v in enumerate(data.normals):
...     v.x = 0.0
...     v.y = 0.0
...     v.z = 1.0
>>> strips.update_tangent_space()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> nifdata = NifFormat.Data(version=0x14010003, user_version=10)
>>> nifdata.roots = [root]
>>> nifdata.write(stream)
>>> stream.close()

```

### Get list of versions and games

```

>>> for vnum in sorted(NifFormat.versions.values()):
...     print('0x%08X' % vnum)
0x02030000
0x03000000
0x03000300
0x03010000
0x0303000D
0x04000000
0x04000002
0x0401000C
0x04020002
0x04020100
0x04020200
0x0A000100
0x0A000102
0x0A000103
0x0A010000
0x0A010065
0x0A01006A
0x0A020000
0x0A020001
0x0A040001
0x14000004
0x14000005
0x14010003
0x14020007
0x14020008
0x14030001
0x14030002
0x14030003
0x14030006
0x14030009
0x14050000
0x14060000
0x14060500
0x1E000002
0x1E010003

```

(continues on next page)

(continued from previous page)

```

>>> for game, versions in sorted(NifFormat.games.items(), key=lambda x: x[0]):
...     print("%s " % game + " ".join('0x%08X' % vnum for vnum in versions))
? 0x0A000103
Atlantica 0x14020008
Axis and Allies 0x0A010000
Bully SE 0x14030009
Civilization IV 0x04020002 0x04020100 0x04020200 0x0A000100 0x0A010000 0x0A020000
↳0x14000004
Culpa Innata 0x04020200
Dark Age of Camelot 0x02030000 0x03000300 0x03010000 0x0401000C 0x04020100 0x04020200
↳0x0A010000
Divinity 2 0x14030009
Emerge 0x14020007 0x14020008 0x14030001 0x14030002 0x14030003 0x14030006 0x1E000002
Empire Earth II 0x04020200 0x0A010000
Empire Earth III 0x14020007 0x14020008
Entropia Universe 0x0A010000
Epic Mickey 0x14060500
Fallout 3 0x14020007
Freedom Force 0x04000000 0x04000002
Freedom Force vs. the 3rd Reich 0x0A010000
Howling Sword 0x14030009
Kohan 2 0x0A010000
KrazyRain 0x14050000 0x14060000
Lazeska 0x14030009
Loki 0x0A020000
Megami Tensei: Imagine 0x14010003
Morrowind 0x04000002
NeoSteam 0x0A010000
Oblivion 0x0303000D 0x0A000100 0x0A000102 0x0A010065 0x0A01006A 0x0A020000 0x14000004
↳0x14000005
Prison Tycoon 0x0A020000
Pro Cycling Manager 0x0A020000
Red Ocean 0x0A020000
Rocksmith 0x1E010003
Rocksmith 2014 0x1E010003
Sid Meier's Railroads 0x14000004
Skyrim 0x14020007
Star Trek: Bridge Commander 0x03000000 0x03010000
The Guild 2 0x0A010000
Warhammer 0x14030009
Wildlife Park 2 0x0A010000 0x0A020000
Worldshift 0x0A020001 0x0A040001
Zoo Tycoon 2 0x0A000100

```

## Reading an unsupported NIF file

```

>>> file = os.path.join(format_root, 'invalid.nif')
>>> stream = open(file, 'rb')
>>> data = NifFormat.Data()
>>> data.inspect(stream) # the file seems ok on inspection
>>> data.read(stream)
Traceback (most recent call last):
...
ValueError: ...
>>> stream.close()

```

## Template types

```
>>> block = NifFormat.NiTextKeyExtraData()
>>> block.num_text_keys = 1
>>> block.text_keys.update_size()
>>> block.text_keys[0].time = 1.0
>>> block.text_keys[0].value = 'hi'
```

## Links

```
>>> NifFormat.NiNode._has_links
True
>>> NifFormat.NiBone._has_links
True
>>> skelroot = NifFormat.NiNode()
>>> geom = NifFormat.NiTriShape()
>>> geom.skin_instance = NifFormat.NiSkinInstance()
>>> geom.skin_instance.skeleton_root = skelroot
>>> [block.__class__.__name__ for block in geom.get_refs()]
['NiSkinInstance']
>>> [block.__class__.__name__ for block in geom.get_links()]
['NiSkinInstance']
>>> [block.__class__.__name__ for block in geom.skin_instance.get_refs()]
[]
>>> [block.__class__.__name__ for block in geom.skin_instance.get_links()]
['NiNode']
```

## Strings

```
>>> extra = NifFormat.NiTextKeyExtraData()
>>> extra.num_text_keys = 2
>>> extra.text_keys.update_size()
>>> extra.text_keys[0].time = 0.0
>>> extra.text_keys[0].value = "start"
>>> extra.text_keys[1].time = 2.0
>>> extra.text_keys[1].value = "end"
>>> for extrastr in extra.get_strings(None):
...     print(extrastr.decode("ascii"))
start
end
```

## pyffi.formats.tga — Targa (.tga)

### Implementation

**class** pyffi.formats.tga.**TgaFormat**

Bases: pyffi.object\_models.xml.FileFormat

This class implements the TGA format.

**class** **ColorMapType** (\*\*kwargs)

Bases: pyffi.object\_models.xml.enum.EnumBase

An unsigned 8-bit integer, describing the color map type.

**class Data**

Bases: `pyffi.object_models.Data`

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

**Returns** Generator for global node children.

**inspect** (*stream*)

Quick heuristic check if stream contains Targa data, by looking at the first 18 bytes.

**Parameters** **stream** (*file*) – The stream to inspect.

**read** (*stream*)

Read a tga file.

**Parameters** **stream** (*file*) – The stream from which to read.

**write** (*stream*)

Write a tga file.

**Parameters** **stream** (*file*) – The stream to write to.

**class FooterString** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

The Targa footer signature.

**get\_hash** (*data=None*)

Return a hash value for the signature.

**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)

Return number of bytes that the signature occupies in a file.

**Returns** Number of bytes.

**get\_value** ()

Get signature.

**Returns** The signature.

**read** (*stream, data*)

Read signature from stream.

**Parameters** **stream** (*file*) – The stream to read from.

**set\_value** (*value*)

Set signature.

**Parameters** **value** (*str*) – The value to assign.

**write** (*stream, data*)

Write signature to stream.

**Parameters** **stream** (*file*) – The stream to read from.

**class Image**

Bases: `pyffi.utils.graph.GlobalNode`

**get\_detail\_child\_names** (*edge\_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

**Returns** Generator for detail tree child names.

**Return type** generator yielding `strs`

**get\_detail\_child\_nodes** (*edge\_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

**Parameters** **edge\_filter** (*EdgeFilter* or type (*None*)) – The edge type to include.

**Returns** Generator for detail tree child nodes.

**Return type** generator yielding *DetailNodes*

**class ImageType** (*\*\*kwargs*)

Bases: *pyffi.object\_models.xml.enum.EnumBase*

An unsigned 8-bit integer, describing the image type.

**ImageData**

alias of *pyffi.object\_models.common.UndecodedData*

**byte**

alias of *pyffi.object\_models.common.Byte*

**char**

alias of *pyffi.object\_models.common.Char*

**float**

alias of *pyffi.object\_models.common.Float*

**int**

alias of *pyffi.object\_models.common.Int*

**short**

alias of *pyffi.object\_models.common.Short*

**ubyte**

alias of *pyffi.object\_models.common.UByte*

**uint**

alias of *pyffi.object\_models.common.UInt*

**ushort**

alias of *pyffi.object\_models.common.UShort*

## Regression tests

### Read a TGA file

```
>>> # check and read tga file
>>> import os
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'tga')
>>> file = os.path.join(format_root, 'test.tga').replace("\\", "/")
>>> stream = open(file, 'rb')
>>> data = TgaFormat.Data()
>>> data.inspect(stream)
>>> data.read(stream)
>>> stream.close()
```

(continues on next page)

```
>>> data.header.width
60
>>> data.header.height
20
```

### Parse all TGA files in a directory tree

```
>>> for stream, data in TgaFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace("\\", "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/tga/test.tga
reading tests/formats/tga/test_footer.tga
```

### Create a TGA file from scratch and write to file

```
>>> data = TgaFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
>>> stream.close()
```

### pyffi.formats.tri — TRI (.tri)

A .tri file contains facial expression data, that is, morphs for dynamic expressions such as smile, frown, and so on.

#### Implementation

```
class pyffi.formats.tri.TriFormat
```

Bases: `pyffi.object_models.xml.FileFormat`

This class implements the TRI format.

#### Data

alias of `Header`

```
class FileSignature (**kwargs)
```

Bases: `pyffi.object_models.xml.basic.BasicBase`

Basic type which implements the header of a TRI file.

```
get_detail_display()
```

Return an object that can be used to display the instance.

**get\_hash** (*data=None*)  
Return a hash value for this value.  
**Returns** An immutable object that can be used as a hash.

**get\_size** (*data=None*)  
Return number of bytes the header string occupies in a file.  
**Returns** Number of bytes.

**read** (*stream, data*)  
Read header string from stream and check it.  
**Parameters** **stream** (*file*) – The stream to read from.

**write** (*stream, data*)  
Write the header string to stream.  
**Parameters** **stream** (*file*) – The stream to write to.

**class FileVersion** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.object_models.xml.basic.BasicBase`

**get\_detail\_display** ()  
Return an object that can be used to display the instance.

**get\_hash** (*data=None*)  
Returns a hash value (an immutable object) that can be used to identify the object uniquely.

**get\_size** (*data=None*)  
Returns size of the object in bytes.

**get\_value** ()  
Return object value.

**read** (*stream, data*)  
Read object from file.

**set\_value** (*value*)  
Set object value.

**write** (*stream, data*)  
Write object to file.

**class Header** (*template=None, argument=None, parent=None*)  
Bases: `pyffi.formats.tri._Header, pyffi.object_models.Data`

A class to contain the actual tri data.

**add\_modifier** (*name=None, relative\_vertices=None*)  
Add a modifier.

**add\_morph** (*name=None, relative\_vertices=None*)  
Add a morph.

**get\_global\_child\_nodes** (*edge\_filter=(True, True)*)  
Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).  
  
Override this method.  
**Returns** Generator for global node children.

**inspect** (*stream*)  
Quickly checks if stream contains TRI data, and reads everything up to the arrays.  
**Parameters** **stream** (*file*) – The stream to inspect.

**inspect\_quick** (*stream*)

Quickly checks if stream contains TRI data, by looking at the first 8 bytes. Reads the signature and the version.

**Parameters** *stream* (*file*) – The stream to inspect.

**read** (*stream*)

Read a tri file.

**Parameters** *stream* (*file*) – The stream from which to read.

**write** (*stream*)

Write a tri file.

**Parameters** *stream* (*file*) – The stream to which to write.

**class ModifierRecord** (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A modifier replaces the vertices from the base model (`Header.vertices`) with those in `Header.modifier_vertices`. Note that `Header.modifier_vertices` counts up from the first modifier onwards. For example, if you were to take the 4th vertex to be modified in the 2nd modifier and the size of the 1st modifier was 10 vertices, then you would need `Header.modifier_vertices[14]`. Therefore, `Header.num_modifier_vertices = sum(modifier.num_vertices_to_modify for modifier in Header.modifiers)`.

**property modifier\_vertices**

The actual modifier vertices (copied from `Header.modifier_vertices`).

**property name**

Name of the Modifier.

**property vertices\_to\_modify**

List of Vertices To Modify.

**class MorphRecord** (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.tri._MorphRecord, object`

```
>>> # create morph with 3 vertices.
>>> morph = TriFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> # scale should be 9/32768.0 = 0.0002746...
>>> morph.scale
0.0002746...
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[3000, 5000, 2000]
[1000, 3000, 2000]
[-8999, 3000, -999]
```

**apply\_scale** (*scale*)

Apply scale factor to data.

```
>>> # create morph with 3 vertices.
>>> morph = TriFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> morph.apply_scale(2)
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[6000, 10000, 4000]
```

(continues on next page)

(continued from previous page)

```
[2000, 6000, 4000]
[-17999, 6000, -1999]
```

**class QuadFace** (*template=None, argument=None, parent=None*)  
 Bases: `pyffi.object_models.xml.struct_.StructBase`

Not used by the Oblivion engine as it's tri based.

**class SizedStringZ** (*\*\*kwargs*)  
 Bases: `pyffi.object_models.common.SizedString`

**get\_size** (*data=None*)  
 Return number of bytes this type occupies in a file.  
**Returns** Number of bytes.

**read** (*stream, data*)  
 Read string from stream.  
**Parameters** **stream** (*file*) – The stream to read from.

**write** (*stream, data*)  
 Write string to stream.  
**Parameters** **stream** (*file*) – The stream to write to.

**byte**  
 alias of `pyffi.object_models.common.Byte`

**char**  
 alias of `pyffi.object_models.common.Char`

**float**  
 alias of `pyffi.object_models.common.Float`

**int**  
 alias of `pyffi.object_models.common.Int`

**short**  
 alias of `pyffi.object_models.common.Short`

**ubyte**  
 alias of `pyffi.object_models.common.UByte`

**uint**  
 alias of `pyffi.object_models.common.UInt`

**ushort**  
 alias of `pyffi.object_models.common.UShort`

**static version\_number** (*version\_str*)  
 Converts version string into an integer.  
**Parameters** **version\_str** (*str*) – The version string.

**Returns** A version integer.

```
>>> TriFormat.version_number('003')
3
>>> TriFormat.version_number('XXX')
-1
```

## Regression tests

### Read a TRI file

```
>>> # check and read tri file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'tri')
>>> file = os.path.join(format_root, 'mmouthxivilai.tri')
>>> stream = open(file, 'rb')
>>> data = TriFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> data.num_vertices
89
>>> data.num_tri_faces
215
>>> data.num_quad_faces
0
>>> data.num_uvs
89
>>> data.num_morphs
18
>>> data.read(stream)
>>> print([str(morph.name.decode("ascii")) for morph in data.morphs])
['Fear', 'Surprise', 'Aah', 'BigAah', 'BMP', 'ChJSh', 'DST', 'Eee', 'Eh', 'FV', 'I',
↵ 'K', 'N', 'Oh', 'OohQ', 'R', 'Th', 'W']
```

### Parse all TRI files in a directory tree

```
>>> for stream, data in TriFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/tri/mmouthxivilai.tri
```

### Create an TRI file from scratch and write to file

```
>>> data = TriFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

## Adding new formats

This section tries to explain how you can implement your own format in pyffi.

### Getting Started

Note that the files which make up the following example can all be found in the `examples/simple` directory of the source distribution of pyffi.

Suppose you have a simple file format, which consists of an integer, followed by a list of integers as many as described by the first integer. We start by creating an XML file, call it `simple.xml`, which describes this format in a way that pyffi can understand:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fileformat>
<fileformat version="1.0">
  <basic name="Int">A signed 32-bit integer.</basic>
  <struct name="Example">
    <add name="Num Integers" type="Int">
      Number of integers that follow.
    </add>
    <add name="Integers" type="Int" arr1="Num Integers">
      A list of integers.
    </add>
  </struct>
</fileformat>
```

What pyffi does is convert this simple XML description into Python classes which automatically can read and write the structure you've just described. Say this is the contents of `simple.py`:

```
import os
import pyffi.object_models.xml
import pyffi.object_models.common

class SimpleFormat(pyffi.object_models.xml.FileFormat):
    xml_file_name = 'simple.xml'
    xml_file_path = [ os.path.dirname(__file__) ]

    # basic types

    Int = pyffi.object_models.common.Int

    # extensions of generated types

    class Data(pyffi.object_models.xml.FileFormat.Data):
        def __init__(self):
            self.example = SimpleFormat.Example()

        def read(self, stream):
            self.example.read(stream, self)

        def write(self, stream):
            self.example.write(stream, self)

    class Example:
        def addInteger(self, x):
```

(continues on next page)

```
self.numIntegers += 1
self.integers.update_size()
self.integers[self.numIntegers-1] = x
```

What happens in this piece of code?

- The `pyffi.object_models.xml.FileFormat` base class triggers the transformation of xml into Python classes; how these classes can be used will be explained further.
- The `xml_file_name` class attribute provides the name of the xml file that describes the structures we wish to generate. The `xml_file_path` attribute gives a list of locations of where to look for this file; in our case we have simply chosen to put `simple.xml` in the same directory as `simple.py`.
- The `SimpleFormat.Example` class provides the generated class with a function `addInteger()` in addition to the attributes `numIntegers` and `integers` which have been created from the XML.
- Finally, the `pyffi.object_models.common` module implements the most common basic types, such as integers, characters, and floats. In the above example we have taken advantage of `pyffi.object_models.common.Int`, which defines a signed 32-bit integer, exactly the type we need.

## Reading and Writing Files

To read the contents of a file of the format described by `simple.xml`:

```
from simple import SimpleFormat
x = SimpleFormat.Data()
f = open('somefile.simple', 'rb')
x.read(f)
f.close()
print(x.example)
```

Or, to create a new file in this format:

```
from simple import SimpleFormat
x = SimpleFormat.Data()
x.example.num_integers = 5
x.example.integers.update_size()
x.example.integers[0] = 3
x.example.integers[1] = 1
x.example.integers[2] = 4
x.example.integers[3] = 1
x.example.integers[4] = 5
f = open('pi.simple', 'wb')
x.write(f)
f.close()
```

## Further References

With the above simple example in mind, you may wish to browse through the source code of `pyffi.formats.cgf` or `pyffi.formats.nif` to see how pyffi works for more complex file formats.

---

## pyffi.spells — High level file operations

---

**Note:** This module is based on wz’s NifTester module, although nothing of wz’s original code is left in this module.

---

A *toaster*, implemented by subclasses of the abstract *Toaster* class, walks over all files in a folder, and applies one or more transformations on each file. Such transformations are called *spells*, and are implemented by subclasses of the abstract *Spell* class.

A *spell* can also run independently of a *toaster* and be applied on a branch directly. The recommended way of doing this is via the *Spell.recurse()* method.

### Supported spells

For format specific spells, refer to the corresponding module.

#### pyffi.spells.cgf — Crytek Geometry/Animation (.cgf/.cga) spells

---

**Todo:** Write documentation.

---

#### pyffi.spells.dds — DirectDraw Surface spells

There are no spells yet.

#### pyffi.spells.kfm — NetImmerse/Gamebryo Keyframe Motion (.kfm) spells

#### pyffi.spells.nif — NetImmerse/Gamebryo File/Keyframe (.nif/.kf/.kfa) spells

Module which contains all spells that check something in a NIF file.

Spells for dumping particular blocks from nifs.

#### pyffi.spells.nif.fix — spells to fix errors

Module which contains all spells that fix something in a nif.

### Implementation

```
class pyffi.spells.nif.fix.SpellDelTangentSpace (toaster=None, data=None,  
                                              stream=None)
```

```
    Bases: pyffi.spells.nif.NifSpell
```

```
    Delete tangentspace if it is present.
```

```
    branchentry (branch)
```

```
        Cast the spell on the given branch. First called with branch equal to data’s children, then the grandchil-  
        dren, and so on. The default implementation simply returns True.
```

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (GlobalNode) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

```
class pyffi.spells.nif.fix.SpellAddTangentSpace (toaster=None, data=None,  
stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Add tangentspace if none is present.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (GlobalNode) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

---

```
class pyffi.spells.nif.fix.SpellFFVT3RSkinPartition (toaster=None, data=None,  

stream=None)
```

```
Bases: pyffi.spells.nif.NifSpell
```

Create or update skin partition, with settings that work for Freedom Force vs. The 3rd Reich.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** `branch` (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.fix.SpellFixTexturePath (toaster=None, data=None,  

stream=None)
```

```
Bases: pyffi.spells.nif.fix.SpellParseTexturePath
```

Fix the texture path. Transforms `0x0a` into `n` and `0x0d` into `r`. This fixes a bug in nifs saved with older versions of nifskope. Also transforms `/` into `.`. This fixes problems when packing files into a bsa archive. Also if the version is 20.0.0.4 or higher it will check for bad texture path form of e.g. `c:program filesbethsofttexturesfilepath.dds` and replace it with e.g. `texturesfilepath.dds`.

**substitute** (*old\_path*)

Helper function to allow subclasses of this spell to change part of the path with minimum of code. This implementation returns path unmodified.

```
class pyffi.spells.nif.fix.SpellDetachHavokTriStripsData (*args, **kwargs)
```

```
Bases: pyffi.spells.nif.NifSpell
```

For `NiTriStrips` if their `NiTriStripsData` also occurs in a `bhkNiTriStripsShape`, make deep copy of data in havok. This is mainly useful as a preparation for other spells that act on `NiTriStripsData`, to ensure that the havok data remains untouched.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.fix.SpellClampMaterialAlpha (toaster=None, data=None,
                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Clamp corrupted material alpha values.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect ()**

This is called after `pyffi.object_models.FileFormat.Data.inspect ()` has been called, and before `pyffi.object_models.FileFormat.Data.read ()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

```
class pyffi.spells.nif.fix.SpellSendGeometriesToBindPosition (toaster=None,
                                                             data=None,
                                                             stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform skinned geometries so similar bones have the same bone data, and hence, the same bind position, over all geometries.

**skelrootentry (branch)**

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellSendDetachedGeometriesToNodePosition (toaster=None,
                                                                        data=None,
                                                                        stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform geometries so each set of geometries that shares bones is aligned with the transform of the root bone of that set.

**skelrootentry (branch)**

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellSendBonesToBindPosition (toaster=None, data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform bones so bone data agrees with bone transforms, and hence, all bones are in bind position.

**skelrootentry (branch)**

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellMergeSkeletonRoots (toaster=None, data=None,
                                                      stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Merges skeleton roots in the NIF file so that no skeleton root has another skeleton root as child. Warns if merge is impossible (this happens if the global skin data of the geometry is not the unit transform).

**branchentry (branch)**

Cast the spell on the given branch. First called with branch equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect (branch)**

Like `_branchinspect ()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**dataentry ()**

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect ()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**datainspect ()**

This is called after `pyffi.object_models.FileFormat.Data.inspect ()` has been called, and before `pyffi.object_models.FileFormat.Data.read ()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

```
class pyffi.spells.nif.fix.SpellApplySkinDeformation (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Apply skin deformation to nif.

```
class pyffi.spells.nif.fix.SpellScale (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Scale a model.

**branchentry (branch)**

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect (branch)**

Like `_branchinspect ()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** `branch` (`GlobalNode`) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**dataentry ()**

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect ()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**classmethod** `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** bool

```
class pyffi.spells.nif.fix.SpellFixCenterRadius (toaster=None, data=None,  
                                              stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckCenterRadius`

Recalculate geometry centers and radii.

```
class pyffi.spells.nif.fix.SpellFixSkinCenterRadius (toaster=None, data=None,  
                                                    stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckSkinCenterRadius`

Recalculate skin centers and radii.

```
class pyffi.spells.nif.fix.SpellFixMopp (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckMopp`

Recalculate mopp data from collision geometry.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** bool

```
class pyffi.spells.nif.fix.SpellCleanStringPalette (toaster=None, data=None,  
                                                  stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Remove unused strings from string palette.

**branchentry** (*branch*)

Parses string palette of either a single controller sequence, or of all controller sequences in a controller manager.

```
>>> seq = NifFormat.NiControllerSequence()
>>> seq.string_palette = NifFormat.NiStringPalette()
>>> block = seq.add_controlled_block()
>>> block.string_palette = seq.string_palette
>>> block.set_variable_1("there")
>>> block.set_node_name("hello")
>>> block.string_palette.palette.add_string("test")
12
```

(continues on next page)

(continued from previous page)

```

>>> seq.string_palette.palette.get_all_strings()
[b'there', b'hello', b'test']
>>> SpellCleanStringPalette().branchentry(seq)
pyffi.toaster:INFO:parsing string palette
False
>>> seq.string_palette.palette.get_all_strings()
[b'hello', b'there']
>>> block.get_variable_1()
b'there'
>>> block.get_node_name()
b'hello'

```

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**substitute** (*old\_string*)

Helper function to substitute strings in the string palette, to allow subclasses of this spell can modify the strings. This implementation returns string unmodified.

```

class pyffi.spells.nif.fix.SpellDelUnusedRoots (toaster=None,          data=None,
                                                stream=None)

```

Bases: `pyffi.spells.nif.NifSpell`

Remove root branches that shouldn't be root branches and are unused in the file such as `NiProperty` branches that are not properly parented.

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

---

```
class pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots (toaster=None, data=None,  
                                                    stream=None)
```

```
Bases: pyffi.spells.nif.NifSpell
```

Fix empty skeleton roots in an as sane as possible way.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

## Regression tests

Spells for optimizing NIF files.

```
class pyffi.spells.nif.optimize.SpellCleanRefLists (toaster=None, data=None,  
                                                    stream=None)
```

```
Bases: pyffi.spells.nif.NifSpell
```

Remove empty and duplicate entries in reference lists.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**cleanreflist** (*reflist, category*)

Return a cleaned copy of the given list of references.

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**class** `pyffi.spells.nif.optimize.SpellMergeDuplicates` (\*args, \*\*kwargs)

Bases: `pyffi.spells.nif.NifSpell`

Remove duplicate branches.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

**class** `pyffi.spells.nif.optimize.SpellOptimizeGeometry` (\*args, \*\*kwargs)

Bases: `pyffi.spells.nif.NifSpell`

Optimize all geometries: - remove duplicate vertices - triangulate - recalculate skin partition - recalculate tangent space

**branchentry** (branch)

Optimize a NiTriStrips or NiTriShape block:

- remove duplicate vertices
- retriangulate for vertex cache
- recalculate skin partition
- recalculate tangent space

---

**Todo:** Limit the size of shapes (see operation optimization mod for Oblivion!)

---

**branchinspect** (branch)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

**class** `pyffi.spells.nif.optimize.SpellOptimize` (toaster=None, data=None, stream=None)

Bases: `pyffi.spells.SpellCleanFarNifSpellDelUnusedRootsSpellCleanRefListsSpellDetachHavoc`

Global fixer and optimizer spell.

**class** `pyffi.spells.nif.optimize.SpellDelUnusedBones` (toaster=None, data=None, stream=None)

Bases: `pyffi.spells.nif.NifSpell`

Remove nodes that are not used for anything.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** `branch` (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**pyffi.spells.nif.modify** — spells to make modifications

Module which contains all spells that modify a nif.

```
class pyffi.spells.nif.modify.SpellTexturePath (toaster=None, data=None,  
                                              stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellParseTexturePath`

Changes the texture path while keeping the texture names.

**substitute** (*old\_path*)

Helper function to allow subclasses of this spell to change part of the path with minimum of code. This implementation returns path unmodified.

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellSubstituteTexturePath (toaster=None,
                                                    data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellFixTexturePath`

Runs a regex replacement on texture paths.

**substitute** (*old\_path*)

Returns modified texture path, and reports if path was modified.

**classmethod** `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellLowResTexturePath (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellSubstituteTexturePath`

Changes the texture path by replacing 'textures\*' with 'textureslowres\*' - used mainly for making \_far.nifs

**substitute** (*old\_path*)

Returns modified texture path, and reports if path was modified.

**classmethod** `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellCollisionType (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Sets the object collision to be a different type

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** **toaster** (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellCollisionMaterial (toaster=None, data=None,  
stream=None)
```

```
Bases: pyffi.spells.nif.NifSpell
```

```
Sets the object's collision material to be a different type
```

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns False, then the spell is not used. The default implementation simply returns True.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters toaster** (Toaster) – The toaster this spell is called from.

**Returns** True if the spell applies, False otherwise.

**Return type** bool

**class** `pyffi.spells.nif.modify.SpellScaleAnimationTime` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.NifSpell`

Scales the animation time.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters branch** (GlobalNode) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** **toaster** (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

**class** `pyffi.spells.nif.modify.SpellReverseAnimation` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.NifSpell`

Reverses the animation by reversing datas in relation to the time.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**class** `pyffi.spells.nif.modify.SpellSubstituteStringPalette` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.fix.SpellCleanStringPalette`

Substitute strings in a string palette.

**substitute** (*old\_string*)

Returns modified string, and reports if string was modified.

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellChangeBonePriorities (toaster=None, data=None,  
stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Changes controlled block priorities based on controlled block name.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** `branch` (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellSetInterpolatorTransRotScale (toaster=None,  
data=None,  
stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Changes specified bone(s) translations/rotations in their NiTransformInterpolator.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** `branch` (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** `toaster` (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

```
class pyffi.spells.nif.modify.SpellDelInterpolatorTransformData (toaster=None,  
                                                                data=None,  
                                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Deletes the specified bone(s) NiTransformData(s).

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** `branch` (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** bool

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** **toaster** (`Toaster`) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** bool

```
class pyffi.spells.nif.modify.SpellDelBranches (toaster=None, data=None,  
stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Delete blocks that match the exclude list.

**branchentry** (*branch*)

Strip branch if it is flagged for deletion.

**is\_branch\_to\_be\_deleted** (*branch*)

Returns `True` for those branches that must be deleted. The default implementation returns `True` for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

```
class pyffi.spells.nif.modify._SpellDelBranchClasses (toaster=None, data=None,  
stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellDelBranches`

Delete blocks that match a given list. Only useful as base class for other spells.

**BRANCH\_CLASSES\_TO\_BE\_DELETED** = ()

List of branch classes that have to be deleted.

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** bool

**is\_branch\_to\_be\_deleted** (*branch*)

Returns True for those branches that must be deleted. The default implementation returns True for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

**class** `pyffi.spells.nif.modify.SpellDelSkinShapes` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.modify.SpellDelBranches`

Delete any geometries with a material name of 'skin'

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**is\_branch\_to\_be\_deleted** (*branch*)

Returns True for those branches that must be deleted. The default implementation returns True for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

**class** `pyffi.spells.nif.modify.SpellDisableParallax` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.NifSpell`

Disable parallax shader (for Oblivion, but may work on other nifs too).

**branchentry** (*branch*)

Cast the spell on the given branch. First called with branch equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (GlobalNode) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

---

```
class pyffi.spells.nif.modify.SpellAddStencilProperty (toaster=None, data=None,  
                                                    stream=None)
```

```
Bases: pyffi.spells.nif.NifSpell
```

Adds a NiStencilProperty to each geometry if it is not present.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (GlobalNode) – The branch to cast the spell on.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

```
class pyffi.spells.nif.modify.SpellDelVertexColor (toaster=None, data=None,  
                                                    stream=None)
```

```
Bases: pyffi.spells.nif.modify.SpellDelBranches
```

Delete vertex color properties and vertex color data.

**branchentry** (*branch*)

Strip branch if it is flagged for deletion.

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

**Parameters** **branch** (GlobalNode) – The branch to check.

**Returns** True if the branch must be processed, False otherwise.

**Return type** bool

**datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

**Returns** True if the file must be processed, False otherwise.

**Return type** bool

**is\_branch\_to\_be\_deleted** (*branch*)

Returns `True` for those branches that must be deleted. The default implementation returns `True` for branches that are not admissible as specified by `include/exclude` options of the toaster. Override in subclasses that must delete specific branches.

**class** `pyffi.spells.nif.modify.SpellMakeSkinlessNif` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.SpellDelSkinShapesSpellAddStencilProperty`

Spell to make fleshless CMR (Custom Model Races) clothing/armour type nifs.

**class** `pyffi.spells.nif.modify.SpellCleanFarNif` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.SpellDelVertexColorPropertySpellDelAlphaPropertySpellDelSpecularProperty`

Spell to clean `_far` type nifs (for even more optimizations, combine this with the `optimize` spell).

**datainspect** ()

Inspect every spell with `L{Spell.datainspect}` and keep those spells that must be cast.

**class** `pyffi.spells.nif.modify.SpellMakeFarNif` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.SpellDelVertexColorPropertySpellDelAlphaPropertySpellDelSpecularProperty`

Spell to make `_far` type nifs (for even more optimizations, combine this with the `optimize` spell).

## `pyffi.spells.tga` — Targa spells

There are no spells yet.

Some spells are applicable on every file format, and those are documented here.

**class** `pyffi.spells.SpellApplyPatch` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.Spell`

A spell for applying a patch on files.

**datainspect** ()

There is no need to read the whole file, so we apply the patch already at inspection stage, and stop the spell process by returning `False`.

**Returns** `False`

**Return type** `bool`

## Adding new spells

To create new spells, derive your custom spells from the `Spell` class, and include them in the `Toaster.SPELLS` attribute of your toaster.

**class** `pyffi.spells.Spell` (*toaster=None, data=None, stream=None*)

Bases: `object`

Spell base class. A spell takes a data file and then does something useful with it. The main entry point for spells is `recurse()`, so if you are writing new spells, start with reading the documentation with `recurse()`.

**READONLY = True**

A `bool` which determines whether the spell is read only or not. Default value is `True`. Override this class attribute, and set to `False`, when subclassing a spell that must write files back to the disk.

**SPELLNAME = None**

A `str` describing how to refer to the spell from the command line. Override this class attribute when subclassing.

**\_\_init\_\_** (*toaster=None, data=None, stream=None*)

Initialize the spell data.

**Parameters**

- **data** (*Data*) – The file *data*.
- **stream** (*file*) – The file *stream*.
- **toaster** (*Toaster*) – The *toaster* this spell is called from (optional).

**\_branchinspect** (*branch*)

Check if spell should be cast on this branch or not, based on exclude and include options passed on the command line. You should not need to override this function: if you need additional checks on whether a branch must be parsed or not, override the `branchinspect()` method.

**Parameters** **branch** (*GlobalNode*) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**\_datainspect** ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (*GlobalNode*) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchexit** (*branch*)

Cast a spell on the given branch, after all its children, grandchildren, have been processed, if `branchentry()` returned `True` on the given branch.

Typically, you will override this function to perform a particular operation on a block type, but you rely on the fact that the children must have been processed first.

**Parameters** **branch** (*GlobalNode*) – The branch to cast the spell on.

**branchinspect** (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (*GlobalNode*) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**data = None**

The *Data* instance this spell acts on.

**dataentry ()**

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via *data*, and unlike in the *datainspect ()* method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**dataexit ()**

Called after all blocks have been processed, if *dataentry ()* returned `True`.

Typically, you will override this function to perform a final spell operation, such as writing back the file in a special way, or making a summary log.

**datainspect ()**

This is called after *pyffi.object\_models.FileFormat.Data.inspect ()* has been called, and before *pyffi.object\_models.FileFormat.Data.read ()* is called. Override this function for customization.

**Returns** `True` if the file must be processed, `False` otherwise.

**Return type** `bool`

**recurse (branch=None)**

Helper function which calls *\_branchinspect ()* and *branchinspect ()* on the branch, if both successful then *branchentry ()* on the branch, and if this is successful it calls *recurse ()* on the branch's children, and once all children are done, it calls *branchexit ()*.

Note that *\_branchinspect ()* and *branchinspect ()* are not called upon first entry of this function, that is, when called with *data* as branch argument. Use *datainspect ()* to stop recursion into this branch.

Do not override this function.

**Parameters** **branch** (`GlobalNode`) – The branch to start the recursion from, or `None` to recurse the whole tree.

**stream = None**

The current file being processed.

**classmethod toastentry (toaster)**

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** **toaster** (*Toaster*) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

**toaster = None**

The *Toaster* instance this spell is called from.

**classmethod toastexit** (*toaster*)

Called when the toaster has finished processing all files.

**Parameters** *toaster* (*Toaster*) – The toaster this spell is called from.

## Grouping spells together

It is also possible to create composite spells, that is, spells that simply execute other spells. The following functions and classes can be used for this purpose.

`pyffi.spells.SpellGroupParallel` (*\*args*)

Class factory for grouping spells in parallel.

`pyffi.spells.SpellGroupSeries` (*\*args*)

Class factory for grouping spells in series.

**class** `pyffi.spells.SpellGroupBase` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.Spell`

Base class for grouping spells. This implements all the spell grouping functions that fall outside of the actual recursing (`__init__()`, `toastentry()`, `_datainspect()`, `datainspect()`, and `toastexit()`).

**ACTIVESPELLCLASSES** = []

List of active spells of this group (not instantiated). This list is automatically built when `toastentry()` is called.

**SPELLCLASSES** = []

List of *Spells* of this group (not instantiated).

**datainspect** ()

Inspect every spell with `L{Spell.datainspect}` and keep those spells that must be cast.

**spells** = []

List of active spell instances.

**classmethod toastentry** (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

**Parameters** *toaster* (*Toaster*) – The toaster this spell is called from.

**Returns** `True` if the spell applies, `False` otherwise.

**Return type** `bool`

**classmethod toastexit** (*toaster*)

Called when the toaster has finished processing all files.

**Parameters** *toaster* (*Toaster*) – The toaster this spell is called from.

**class** `pyffi.spells.SpellGroupParallelBase` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.SpellGroupBase`

Base class for running spells in parallel (that is, with only a single recursion in the tree).

**branchentry** (*branch*)

Run all spells.

**branchexit** (*branch*)

Cast a spell on the given branch, after all its children, grandchildren, have been processed, if *branchentry()* returned `True` on the given branch.

Typically, you will override this function to perform a particular operation on a block type, but you rely on the fact that the children must have been processed first.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**branchinspect** (*branch*)

Inspect spells with *Spell.branchinspect()* (not all checks are executed, only keeps going until a spell inspection returns `True`).

**property changed**

`bool(x) -> bool`

Returns `True` when the argument `x` is true, `False` otherwise. The builtins `True` and `False` are the only two instances of the class `bool`. The class `bool` is a subclass of the class `int`, and cannot be subclassed.

**dataentry** ()

Look into every spell with *Spell.dataentry()*.

**dataexit** ()

Look into every spell with *Spell.dataexit()*.

**class** `pyffi.spells.SpellGroupSeriesBase` (*toaster=None, data=None, stream=None*)

Bases: *pyffi.spells.SpellGroupBase*

Base class for running spells in series.

**branchentry** (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

**Parameters** **branch** (`GlobalNode`) – The branch to cast the spell on.

**Returns** `True` if the children must be processed, `False` otherwise.

**Return type** `bool`

**branchinspect** (*branch*)

Like *\_branchinspect()*, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

**Parameters** **branch** (`GlobalNode`) – The branch to check.

**Returns** `True` if the branch must be processed, `False` otherwise.

**Return type** `bool`

**property changed**

`bool(x) -> bool`

Returns `True` when the argument `x` is true, `False` otherwise. The builtins `True` and `False` are the only two instances of the class `bool`. The class `bool` is a subclass of the class `int`, and cannot be subclassed.

**dataentry** ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the *datainspect()* method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

**Returns** True if the children must be processed, False otherwise.

**Return type** bool

**dataexit** ()

Called after all blocks have been processed, if *dataentry()* returned True.

Typically, you will override this function to perform a final spell operation, such as writing back the file in a special way, or making a summary log.

**recurse** (*branch=None*)

Recurse spells in series.

## Creating toaster scripts

To create a new toaster script, derive your toaster from the *Toaster* class, and set the *Toaster.FILEFORMAT* attribute of your toaster to the file format class of the files it can toast.

**class** `pyffi.spells.Toaster` (*spellclass=None, options=None, spellnames=None, logger=None*)

Bases: `object`

Toaster base class. Toasters run spells on large quantities of files. They load each file and pass the data structure to any number of spells.

**ALIASDICTIONARY** = {}

Dictionary with aliases for spells.

**DEFAULT\_OPTIONS** = {'applypatch': False, 'archives': False, 'arg': '', 'createpatch': False}

List of spell classes of the particular *Toaster* instance.

**EXAMPLES** = ''

Some examples which describe typical use of the toaster.

**FILEFORMAT**

alias of `pyffi.object_models.FileFormat`

**SPELLS** = []

List of all available *Spell* classes.

**cli** ()

Command line interface: initializes spell classes and options from the command line, and run the *toast()* method.

**exclude\_types** = []

Tuple of types corresponding to the exclude key of *options*.

**get\_toast\_head\_root\_ext** (*filename*)

Get the name of where the input file *filename* would be written to by the toaster: head, root, and extension.

**Parameters** **filename** (*str*) – The name of the hypothetical file to be toasted.

**Returns** The head, root, and extension of the destination, or (None, None, None) if `--dry-run` is specified.

**Return type** tuple of three *strs*

**get\_toast\_stream** (*filename, test\_exists=False*)

Calls `get_toast_head_root_ext(filename)()` to determine the name of the toast file, and return a stream for writing accordingly.

Then return a stream where result can be written to, or in case `test_exists` is `True`, test if result would overwrite a file. More specifically, if `test_exists` is `True`, then no streams are created, and `True` is returned if the file already exists, and `False` is returned otherwise.

**include\_types** = []

Tuple of types corresponding to the include key of *options*.

**indent** = 0

An int which describes the current indentation level for messages.

**inspect\_filename** (*filename*)

Returns whether to toast a filename or not, based on `skip_regexs` and `only_regexs`.

**is\_admissible\_branch\_class** (*branchtype*)

Helper function which checks whether a given branch type should have spells cast on it or not, based in exclude and include options.

```
>>> from pyffi.formats.nif import NifFormat
>>> class MyToaster(Toaster):
...     FILEFORMAT = NifFormat
>>> toaster = MyToaster() # no include or exclude: all admissible
>>> toaster.is_admissible_branch_class(NifFormat.NiProperty)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiNode)
True
>>> toaster = MyToaster(options={"include": ["NiProperty", "NiNode"], "exclude":
↳ ["NiMaterialProperty", "NiLODNode"]})
>>> toaster.is_admissible_branch_class(NifFormat.NiProperty)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiNode)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiAVObject)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiLODNode)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiSwitchNode)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiMaterialProperty)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiAlphaProperty)
True
```

**logger** = <Logger pyffi.toaster (WARNING)>

A `logging.Logger` for toaster log messages.

**msg** (*message*)

Write log message with `logger.info()`, taking into account *indent*.

**Parameters** *message* (*str*) – The message to write.

**msgblockbegin** (*message*)

Acts like *msg()*, but also increases *indent* after writing the message.

**msgblockend** (*message=None*)

Acts like *msg()*, but also decreases *indent* before writing the message, but if the message argument is `None`, then no message is printed.

**only\_regexs** = []

Tuple of regular expressions corresponding to the only key of *options*.

```

options = {}
    The options of the toaster, as dict.

static parse_inifile (option, opt, value, parser, toaster=None)
    Initializes spell classes and options from an ini file.

skip_regexs = []
    Tuple of regular expressions corresponding to the skip key of options.

spellnames = []
    A list of the names of the spells.

toast (top)
    Walk over all files in a directory tree and cast spells on every file.

        Parameters top (str) – The directory or file to toast.

toast_archives (top)
    Toast all files in all archives.

top = ''
    Name of the top folder to toast.

write (stream, data)
    Writes the data to data and raises an exception if the write fails, but restores file if fails on overwrite.

writepatch (stream, data)
    Creates a binary patch for the updated file.

```

### pyffi.object\_models — File format description engines

**Warning:** The documentation of this module is very incomplete.

This module bundles all file format object models. An object model is a group of classes whose instances can hold the information contained in a file whose format is described in a particular way (xml, xsd, and possibly others).

```
class pyffi.object_models.MetaFileFormat
```

Bases: `type`

This metaclass is an abstract base class for transforming a file format description into classes which can be directly used to manipulate files in this format.

A file format is implemented as a particular class (a subclass of `FileFormat`) with class members corresponding to different (bit)struct types, enum types, basic types, and aliases.

```
static openfile (filename, filepaths=None, encoding=None)
```

Find *filename* in given *filepaths*, and open it. Raises `IOError` if file cannot be opened.

#### Parameters

- **filename** (*str*) – The file to open.
- **filepaths** (*list of str*s) – List of paths where to look for the file.

```
class pyffi.object_models.FileFormat
```

Bases: `object`

This class is the base class for all file formats. It implements a number of useful functions such as walking over directory trees (`walkData()`) and a default attribute naming function (`name_attribute()`). It also implements the base class for representing file data (`FileFormat.Data`).

**ARCHIVE\_CLASSES = []**

Override this with a list of archive formats that may contain files of the format.

**class Data**

Bases: `pyffi.utils.graph.GlobalNode`

Base class for representing data in a particular format. Override this class to implement reading and writing.

**inspect** (*stream*)

Quickly checks whether the stream appears to contain data of a particular format. Resets stream to original position. Call this function if you simply wish to check that a file is of a particular format without having to parse it completely.

Override this method.

**Parameters** *stream* (*file*) – The file to inspect.

**Returns** `True` if stream is of particular format, `False` otherwise.

**read** (*stream*)

Read data of particular format from stream. Override this method.

**Parameters** *stream* (*file*) – The file to read from.

**user\_version = None**

User version (additional version field) of the data.

**version = None**

Version of the data.

**write** (*stream*)

Write data of particular format to stream. Override this method.

**Parameters** *stream* (*file*) – The file to write to.

**RE\_FILENAME = None**

Override this with a regular expression (the result of a `re.compile` call) for the file extension of the format you are implementing.

**classmethod name\_attribute** (*name*)

Converts an attribute name, as in the description file, into a name usable by python.

**Parameters** *name* (*str*) – The attribute name.

**Returns** Reformatted attribute name, useable by python.

```
>>> FileFormat.name_attribute('tHis is A Silly nAME')
't_this_is_a_silly_na_me'
>>> FileFormat.name_attribute('Test:Something')
'test_something'
>>> FileFormat.name_attribute('unknown?')
'unknown'
```

**classmethod name\_class** (*name*)

Converts a class name, as in the xsd file, into a name usable by python.

**Parameters** *name* (*str*) – The class name.

**Returns** Reformatted class name, useable by python.

```
>>> FileFormat.name_class('this IS a sillyNAME')
'ThisIsASillyNAME'
```

**classmethod name\_parts** (*name*)

Intelligently split a name into parts:

- first, split at non-alphanumeric characters
- next, separate digits from characters
- finally, if some part has mixed case, it must be camel case so split it further at upper case characters

```

>>> FileFormat.name_parts("hello_world")
['hello', 'world']
>>> FileFormat.name_parts("HELLO_WORLD")
['HELLO', 'WORLD']
>>> FileFormat.name_parts("HelloWorld")
['Hello', 'World']
>>> FileFormat.name_parts("helloWorld")
['hello', 'World']
>>> FileFormat.name_parts("xs:NMTOKEN")
['xs', 'NMTOKEN']
>>> FileFormat.name_parts("xs:NCName")
['xs', 'N', 'C', 'Name']
>>> FileFormat.name_parts('this IS a sillyNAME')
['this', 'IS', 'a', 'silly', 'N', 'A', 'M', 'E']
>>> FileFormat.name_parts('tHis is A Silly naME')
['t', 'His', 'is', 'A', 'Silly', 'na', 'M', 'E']

```

**static version\_number** (*version\_str*)

Converts version string into an integer. This default implementation simply returns zero at all times, and works for formats that are not versioned.

Override for versioned formats.

**Parameters** *version\_str* (*str*) – The version string.

**Returns** A version integer. A negative number denotes an invalid version.

**classmethod walk** (*top*, *topdown=True*, *mode='rb'*)

A generator which yields all files in directory *top* whose filename matches the regular expression *RE\_FILENAME*. The argument *top* can also be a file instead of a directory. Errors coming from `os.walk` are ignored.

Note that the caller is not responsible for closing the stream.

This function is for instance used by `pyffi.spells` to implement modifying a file after reading and parsing.

**Parameters**

- **top** (*str*) – The top folder.
- **topdown** (*bool*) – Determines whether subdirectories should be iterated over first.
- **mode** (*str*) – The mode in which to open files.

**classmethod walkData** (*top*, *topdown=True*, *mode='rb'*)

A generator which yields the data of all files in directory *top* whose filename matches the regular expression *RE\_FILENAME*. The argument *top* can also be a file instead of a directory. Errors coming from `os.walk` are ignored.

Note that the caller is not responsible for closing the stream.

This function is for instance used by `pyffi.spells` to implement modifying a file after reading and parsing.

**Parameters**

- **top** (*str*) – The top folder.

- **topdown** (`bool`) – Determines whether subdirectories should be iterated over first.
- **mode** (`str`) – The mode in which to open files.

### 1.7.4 How to contribute

Do you want to fix a bug, improve documentation, or add a new feature?

#### Get git/msysgit

If you are on windows, you need `msysgit`. If you are already familiar with subversion, then, in a nutshell, `msysgit` is for git what TortoiseSVN is for subversion. The main difference is that `msysgit` is a command line based tool.

For more information about git and github, the [github help site](#) is a great start.

#### Track the source

If you simply want to keep track of the latest source code, start a shell (or, the Git Bash on windows), and type (this is like “svn checkout”):

```
git clone git://github.com/nifttools/pyffi.git
```

To synchronize your code, type (this is like “svn update”):

```
git pull
```

#### Development

##### Create a fork

- Get a [github account](#).
- [Log in on github](#) and [fork PyFFI](#) (yes! merging with git is easy so forking is encouraged!).

##### Use the source

PyFFI is entirely written in pure Python, hence the source code runs as such on any system that runs Python. Edit the code with your favorite editor, and install your version of PyFFI into your Python tree to enable your PyFFI to be used by other applications such as for instance QSkope, or the Blender NIF Scripts. From within your PyFFI git checkout:

```
C:\Python25\python.exe setup.py install
```

or on linux:

```
python setup.py install
```

To build the documentation:

```
cd docs
make html -a
```

PyFFI has an extensive test suite, which you can run via:

```
python rundoctest.py
```

The Blender NIF Scripts test suite provides additional testing for PyFFI. From within your niftools/blender checkout:

```
./install.sh  
blender -P runtest_XXX.py
```

To build the source packages and the Windows installer (on a linux system which has both wine and nsis installed):

```
makezip.sh
```

## Submit your updates

Simply do a [pull request](#) if you want your fork to be merged, and your contributions may be included in the next release!

## 1.7.5 Authors

### Main author

- Amorilia ([amorilia@users.sourceforge.net](mailto:amorilia@users.sourceforge.net))

### Previous Developers

- wz ([wz\\_@users.sourceforge.net](mailto:wz_@users.sourceforge.net))
  - niftester (the current spells module is a revamped version of that)
  - nifvis (which hopefully will be embedded into QSkope one day...)
- taarna23 ([taarna23@users.sourceforge.net](mailto:taarna23@users.sourceforge.net))
  - extraction of DXT compressed embedded textures for the nif format
- tazpn ([tazpn@users.sourceforge.net](mailto:tazpn@users.sourceforge.net))
  - mopp generator using the Havok SDK
  - suggestions for improving the spells module
- Scanti
  - EGM & TRI format info
- Carver13
  - EGM & TRI format xml

### Contributors

- PacificMorrowind ([pacmorrowind@sourceforge.net](mailto:pacmorrowind@sourceforge.net))
  - some nif/kf modifying spells
- Ulrim/Arthmoor
  - optimization kit

- seith ([seith@users.sourceforge.net](mailto:seith@users.sourceforge.net))
  - logo design for the Windows installer
- MorCroft
  - Patch for BSSTextureSet texture path substitution

### 1.7.6 License

Copyright © 2007-2012, Python File Format Interface. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Python File Format Interface project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

PyFFI uses Mopper. Copyright 2008 NIF File Format Library and Tools. All rights reserved. Run `pyffi/utils/mopper.exe --license` for details.

---

Mopper uses Havok. Copyright 1999-2008 Havok.com Inc. (and its Licensors). All rights reserved. See <http://www.havok.com> for details.

---

PyFFI uses xdelta3. Copyright 2001-2010 Joshua P. MacDonald xdelta3 is released under the GPLv2. See [external/xdelta3.0z/COPYING](http://external/xdelta3.0z/COPYING) for details.

### 1.7.7 ChangeLog

#### Release 2.2.3 (Mar 17, 2014)

- Update spell `texture_path_substitution` for BSTextureSet blocks (fix contributed by MorCroft)
- Updated to latest `nif.xml`, submodules moved to github.

**Release 2.2.2 (Nov 17, 2012)**

- Use VERSION file.
- Fix dump\_python for correct default values.
- Fix xml for Fallout 3.

**Release 2.2.1 (Nov 11, 2012)**

- Added Skyrim version detection.
- The check\_readwrite spell now handles file size differences due to empty strings.
- Bugfix in nif write method when entities are None (see Skyrim meshes/actors/character/character assets/hair/hairlonghumanm.nif).
- Various fixes for Fallout 3 and Fallout NV nifs.
- New dump\_python spell, to generate python code that recreates a nif file.
- Accept string palette strings that do not have a null character preceding it (reported by Koniption).
- New modify\_getbonepriorities and modify\_setbonepriorities spells, which work similar to the kfupdater.
- New fix\_fallout3stringoffsets spell to ‘fix’ empty offsets in Oblivion style kf files, if they are to be used in Fallout 3.
- Installer no longer targets Maya and Blender.

**Release 2.2.0 (Nov 26, 2011)**

- Added PSK and PSA file support (used by Unreal engine).
- A py3k fix (contributed by infectedsoundsystem).
- Updated installer for Blender 2.5x+.

**Release 2.1.11 (Nov 26, 2011)**

- Explicitly use wine for running mopper on non-win32 platforms (fixes issue on Arch Linux, reported by ifss000f, see issue #3423990).
- Removed skip list from extra fix\_texturepath stage in Oblivion optimization kit.
- Various optimizations (contributed by infectedsoundsystem). The optimizer spell now runs a fair bit faster.
- Garbage collection call after each spell has been removed as profiling showed that a lot of time was spent on it. You can still force the old (slow) behaviour by using the new `-gccollect` command line option or adding `“gccollect = True”` in your ini file.
- Encoding fix for xml and xsd parsing.
- Merge duplicates after optimizing geometry to work around de-duplication during geometry optimization phase (fixes issue #3425637, reported by chacky2).
- Removed xsd object model and dae format (wasn’t functional yet anyway); deferred to py3k.

### Release 2.1.10 (Oct 10, 2011)

- Fixed bspline data methods to handle invalid kfs with missing basis data (reported by K'Aviash).
- Fixed mass, center, inertia methods to deal with cases where shape is missing (reported by rlibiez, see niftools issue #3248754).
- Fixed center calculation of bhkListShape collisions, and fixed zero division error when creating very small collision shapes (reported by Koniption, see issues #3334577 and #3308638).
- Fixed shortcut to uninstaller in programs folder (reported by neomonkeus, see niftools issue #3397540).
- Fixed geometry optimizer to handle cases where number of morph vertices does not match number of shape vertices (reported by rlibiez, see issue #3395484).
- Merged ulrim's optimization kit, along with arthmoor's improved ini files.
- Integrated far nif optimization with the general purpose optimize spell (requested by Gratis\_monsta, see issue #3177316).
- New shell\_optimize.ini for configuring optimization as executed from Windows shell.
- Allow .ini files that do not have a [main] or [options] section.
- Fix Windows shell integration to point to the new shell\_optimize.ini file (reported by rlibiez, see issue #3415490).
- Fixed zombie process problem on Windows when a toaster was running with multiple jobs (reported by Alphanos, see issue #3390826).

### Release 2.1.9 (Mar 26, 2011)

- Improved documentation of .dir/.img unpack and pack scripts.
- Bugfix in .dir format, so it can now also handle single record images.
- Added new script to make and apply patches (functionality is identical to and OnmyojiOmn's and jonwd7's pyffi patcher scripts, but it is written in Python to work on all platforms).
- New fix\_emptyskeletonroots spell (automatically included in the optimize spell) to fix issues with nifs that do not have their NiSkinInstance skeleton root set (fixes #3174085, reported by xjdhdr).
- Fixed logging issue on Windows platform with multithreading enabled (fixes issue #3174339, reported by xjdhdr).
- Fixed QSkope shortcut issue when path contains spaces (reported by Brumbek).
- Added support for BSPackedAdditionalGeometryData (reported by Ghostwalker71, niftools issue #3177847).
- Skip terminal chars in mopper output (fixes issues with running mopper under wine).
- Bugfix in patch\_recursive\_make/apply scripts for "deep" folder layouts (fixes issue #3193914, reported by xjdhdr).
- Do not pack collisions in OL\_CLUTTER (fixes issue #3194017 reported by Gratis\_monsta).
- Fixed issue with skipping terminal chars in mopper output on Windows platform (fixes issue #3205569, reported and diagnosed by ulrim).
- Updates for Bully SE format (fixes issue reported by Tosyk).
- Bully SE nif header reading fix for BBonusB.nft.
- Added initial support for Epic Mickey (reported and test files provided by Tosyk).

- Bugfix for NiMesh read and write.
- Updated dump\_pixeldata spell to enable it to export Bully SE's nft files.
- Disabled copy in patch\_recursive\_apply script (see issue #3219744, suggested by ulrim).
- Pass additional arguments of patch\_recursive\_apply/make to the patch command (see issue #3219744, suggested by ulrim).
- Fix nif optimizer in case it contains tangent space data but no uv data (see issue #3218751, reported by krimhorn).
- Handle removal of redundant triangles when updating dismember skin partitions (see issue #3218751, reported by krimhorn).
- Fix vertex cache optimizer to handle more meshes with more than 255 triangles per vertex (see issue #3218751, reported by krimhorn).
- Skipping meshes that have NiAdditionalGeometryData (until we understand what this data does and how to optimize it).
- Sane default settings for bhkRigidBody unknowns to ensure that constraints behave properly (contributed by Koniption).
- Added ini file to unpack Bully SE .nft files.

### Release 2.1.8 (Feb 4, 2011)

- Quickhull bugfix for precision argument in degenerate cases (issue #3163949, fix contributed by liuhuanjim013).
- Fixed issue with detecting box shapes on degenerate collision meshes (fixes issue #3145104, reported by Gratis\_monsta).
- Ensure that enum has valid default value.
- Added CStreamableAssetData for Divinity 2 (reported by pertininen, niftools issue #3164929).
- NiPixelData.save\_as\_dds fourcc flag bugfix.
- Added Rockstar .dir format (used in Bully SE).
- Added Rockstar .dir/.img unpack and pack scripts (only tested for Bully SE).

### Release 2.1.7 (Jan 23, 2011)

- Added support for Fallout New Vegas (contributed by throttlekitty and saiden).
- Updated geometry optimizer to keep dismember body parts, for Fallout 3 and Fallout New Vegas (fixes issue #3025691 reported by Chaky).
- Added flag to enable debugging in vertex cache algorithm, to assess how suboptimal the solution is on any particular mesh (testing reveals that it finds the globally optimal solution in more than 99% of all iterations, for typical meshes).
- New check\_triangles\_atvr spell to find optimal parameters for vertex cache algorithm by simulated annealing.
- Fixed send\_geometries\_to\_bind\_position, send\_detached\_geometries\_to\_node\_position, and send\_bones\_to\_bind\_position in case skin instance has empty bone references (fixes issue #3114079, reported by drakonnen).
- Fix for verbose option in multithread mode (reported by Gratis\_monsta).

- Fix optimize spell when no vertices are left after removing duplicate vertices and degenerate triangles (reported by Gratis\_monsta).
- Fixed tangent space issue along uv seams (reported by Gratis\_monsta and Tommy\_H, see issue #3120585).
- Log an error instead of raising an exception on invalid enum values (fixes issue #3127161, reported by rlibiez).
- Disabled 2to3 in Windows installer; the Python 3 version of PyFFI will be released separately.

### Release 2.1.6 (Nov 13, 2010)

- The optimize spell now includes two new spells: `opt_collisiongeometry` for optimizing triangle based collisions, and `opt_collisionbox` for optimizing simple box collisions. This should result in faster loads and probably also a small but noticeable performance improvement.
- Fixed `opt_collisiongeometry` for multimaterial mopps (reported by wildcard\_25, see issue #3058096).
- New `SpellCleanFarNif` spell (suggested by wildcard\_25, see issue #3021629).
- Bad normals are now ignored when packing a `bhkNiTriStripsShape` (fixes issue #3060025, reported by rlibiez).
- The `opt_delunusedbones` spell no longer removes bones if they have a collision object (fixes issue #3064083, reported by wildcard\_25).
- If the `jobs` option is not specified in the toaster, then the number of processors is used—requires Python 2.6 or higher (suggested by chaky2, see issue #3052715, implements issue #3065503).
- New `opt_delzeroscale` spell to delete branches with zero scale (suggested by chaky2, see issue #3013004).
- The `opt_mergeduplicates` spell now ignores (non-special) material names, so identical materials with different names will get merged as well (suggested by chaky2, see issue #3013004).
- New spell to fix subshape counts (see issue #3060025, reported by rlibiez), it is included in the optimize spell.
- New `opt_collisionbox` spell which automatically converts triangle based box collisions to primitive box collisions, which are much faster in-game (contributed by PacificMorrowind).
- Optimizer spell now uses triangles to represent skin partitions (improves in-game fps).
- Better vertex map calculation when calculating skin partitions (improves in-game fps).
- Optimizer now always triangulates (improves in-game fps). Stripification will be readded later in a modularized version of the optimizer spell, for those that want minimal file size rather than maximal performance).
- Much faster implementation of vertex cache algorithm (now runs in linear time instead of quadratic time).
- Check triangle count when converting to box shape (fixes issue #3091150).
- Bugfix in vertex map reordering (fixes most nifs reported in issue #3071616).
- Bugfix in vertex cache algorithm (fixes a nif reported in issue #3071616).
- Cover degenerate case in ATVR calculation when there are no vertices (fixes a nif reported in issue #3071616).
- Cover degenerate case when calculating cache optimized vertex map (fixes a nif reported in issue #3071616).
- Remove branches if they have no triangles (again fixes a nif reported in issue #3071616).

### Release 2.1.5 (Jul 18, 2010)

- Improved interface for TRI files, and a bugfix in TRI file writing.
- Added EGT file support.
- The `fix_texturepath` spell now also converts double backslash in single backslash (suggested by Baphometal).

- Bugfix in `save_as_dds` function for newer `NiPixelData` blocks (reported by norocelmiau, issue #2996800).
- Added support for Laxe Lore nifs (reported by bobsobol, issue #2995866).
- New spells:
  - `opt_collisiongeometry`: to optimize collision geometry in nifs (contributed by PacificMorrowind).
  - `check_materialemissivevalue`: checks (and warns) about high values in material emissive settings (contributed by PacificMorrowind).
  - `modify_mirroranimation`: mirrors an animation (specifically left to right and vice versa) - use it to for example turn a right hand punch anim into a left hand punch anim (contributed by PacificMorrowind).
- Added big-endian support.
- Removed `**kwargs` argument passing for faster and more transparant implementation (reading and writing is now about 8% faster).
- Do not merge `BSShaderProperty` blocks (reported by Chaky, niftools issue #3009832).
- Installer now recognizes Maya 2011.
- Fixed `NiPSysData` read and write for Fallout 3 (reported by Chaky, niftools issue #3010861).

#### Release 2.1.4 (Mar 19, 2010)

- Extra names in `oblivion_optimize.ini` skip list for known mods (contributed by Tommy\_H).
- New spells
  - `modify_collisiontomopp`
  - `opt_reducegeometry`
  - `opt_packcollision`
- Windows right-click optimize method now uses `default.ini` and `oblivion_optimize.ini`.
- `fix_texturepath` now fixes paths that include the whole drive path to just the `textures/...` path.
- The optimize spell has been fixed to update Fallout 3 style tangent space (fixes issue #2941568, reported by xjdhdr).

#### Release 2.1.3 (Feb 20, 2010)

- Added toaster option to process files in archives (not yet functional).
- Added toaster option to resume, by skipping existing files in the destination folder.
- Toaster now removes incompletely written files on CTRL-C (to avoid corrupted files).
- Fixed `makefarnif` spell (now no longer deletes vertex colors).
- New spells
  - `fix_delunusedroots`
  - `modify_bonepriorities`
  - `modify_interpolatortransrotscale`
  - `modify_delinterpolatortransformdata`
  - `opt_delunusedbones`

- The niftoaster optimize spell now also includes fix\_delunusedroots.
- Removed unused pep8 attribute conversion code.
- Toasters can now be configured from an ini file.
- bhkMalleableHinge update\_a\_b bugfix (reported by Ghostwalker71).

### Release 2.1.2 (Jan 16, 2010)

- Fallout 3 skin partition flag bugfix (reported by Ghostwalker71).
- Fixed bug in optimize spell, when has\_vertex\_colors was False but vertex color array was present (reported by Baphometal, debugged by PacificMorrowind).
- Initial bsa file support (Morrowind, Oblivion, and Fallout 3).

### Release 2.1.1 (Jan 11, 2010)

- Accidentally released corrupted nif.xml (affected Fallout 3), so this is just a quick bugfix release including the correct nif.xml.

### Release 2.1.0 (Jan 10, 2010)

- Improved windows installer.
- Compatibility fix for Python 2.5 users (reported by mac1415).
- Renamed some internal modules for pep8 compliance.
- All classes and attributes are now in pep8 style. For compatibility, camelCase attributes are generated too (however this will be dropped for py3k).
- Renamed a few niftoaster spells.
  - fix\_strip -> modify\_delbranches
  - fix\_disableparallax -> modify\_disableparallax
- New niftoaster spells.
  - fix\_cleanstringpalette: removes unused strings from string palette.
  - modify\_substitutestringpalette: regular expression substitution of strings in a string palette.
  - modify\_scaleanimationtime: numeric scaling of animations.
  - modify\_reverseanimation: reverses an animation (ie useful for making only an open animation and then running this to get a close animation).
  - modify\_collisionmaterial: sets any collision materials in a nif to specified type.
  - modify\_delskinshapes: Delete any geometries with a material name of 'skin'
  - modify\_texturepathlowres: Changes the texture path by replacing 'textures/' with 'textures/lowres/'. used mainly for making \_far.nifs.
  - modify\_addstencilprop: Adds a NiStencilProperty to each geometry if it is not present.
  - modify\_substitutetexturepath: regular expression substitution of a texture path.
  - modify\_makeskinlessnif: Spell to make fleshless CMR (Custom Model Races) clothing/armour type nifs. (runs modify\_delskinshapes and modify\_addstencilprop)

- modify\_makefarnif: Spell to make \_far type nifs.
- Bugfix for niftoaster dump spell.
- New –suffix option for toaster (similar to the already existing –prefix option).
- New –skip and –only toaster options to toast files by regular expression.
- New –jobs toaster option which enables multithreaded toasting.
- New –source-dir and –dest-dir options to save toasted nifs in a given destination folder.
- Added workaround for memory leaks (at the moment requires –jobs >= 2 to be functional).
- The niftoaster opt\_geometry spell now always skips NIF files when a similarly named tri or egm file is found.
- Added support for Atlantica nifs.
- Added support for Joymaster Interactive Howling Sword nifs.

### Release 2.0.5 (Nov 23, 2009)

- Added regression test and fixed rare bug in stripification (reported by PacificMorrowind, see issue #2889048).
- Improved strip stitching algorithm: *much* more efficient, and now rarely needs more than 2 stitches per strip.
- Improved stripifier algorithm: runs about 30% faster, and usually yields slightly better strips.
- Added new modify\_texturepath and modify\_collisiontype niftoaster spells (contributed by PacificMorrowind).
- Various fixes and improvements for 20.5.0.0+ nifs.
- Check endian type when processing nifs.
- Source release now includes missing egm.xml and tri.xml files (reported by skomut, fixes issue #2902125).

### Release 2.0.4 (Nov 10, 2009)

- Write NaN on float overflow.
- Use pytristrip if it is installed.
- Implemented the FaceGen egm (done) and tri (in progress) file formats with help of Scanti and Carver13.
- The nif dump\_pixeldata spell now also dumps NiPersistentSrcTextureRenderData (reported by lusht).
- Set TSpace flags 16 to signal presence of tangent space data (fixes Fallout 3 issue, reported by Miaximus).

### Release 2.0.3 (Sep 28, 2009)

- Various bugfixes for the Aion cgf format.
- Updates for nif.xml to support more recent nif versions (20.5.0.0, 20.6.0.0, and 30.0.0.2).

### Release 2.0.2 (Aug 12, 2009)

- The source has been updated to be Python 3.x compatible via 2to3.
- New unified installer which works for all versions of Python and Maya at once (at the moment: 2.5, 2.6, 3.0, 3.1) and also for all versions of Maya that use Python 2.5 and 2.6 (2008, 2009, and 2010, including the 64 bit variants).

- Added support for Aion cgf files.
- Added support for NeoSteam header and footer.
- Log warning rather than raising exception on invalid links (fixes issue #2818403 reported by abubakr125).
- Optimizer can now recover from invalid indices in strips (this fixes some nifs mentioned in issue #2795837 by baphometal).
- Skin updater can now recover when some vertices have no weights (this fixes some nifs mentioned in issue #2795837 by baphometal).
- Skip zero weights and add up weights of duplicated bones when calculating vertex weights (this fixes some nifs mentioned in issue #2795837 by baphometal).
- The nif optimizer can now handle NiTriShapeData attached as a NiTriStrips data block (fixes some corrupt nifs provided by baphometal in issue #2795837).
- Optimizer can now recover from NaN values in geometry (sample nifs provided by baphometal).
- Do not attempt to optimize nifs with an insane amount of triangles, but put out a warning instead.
- Log error rather than raising exception when end of NIF file is not reached (fixes issue with sample nif provided by baphometal).

### Release 2.0.1 (Jul 22, 2009)

- Added Windows installer for Python 2.6.
- Updated mopper.exe compiled with msvc 2008 sp1 (fixes issue #2802413, reported by pacmorrowind).
- Added pdb session to track circular references and memory leaks (see issues #2787602 and #2795837 reported by alexkapi12 and xfrancis147).
- Added valgrind script to check memory usage, and to allow keeping track of it between releases (see issues #2787602 and #2795837 reported by alexkapi12 and xfrancis147).
- Removed parenting in xml model from everywhere except Array, and using weakrefs to avoid circular references, which helps with garbage collection. Performance should now be slightly improved.
- Updates to xml object model expression syntax.
  - Support for field qualifier ‘.’.
  - Support for addition ‘+’.
- Updates to Targa format.
  - Support for RLE compressed Targa files (test file contributed by Alphax, see issue #2790494).
  - Read Targa footer, if present (test file contributed by Alphax, see issue #2790494).
  - Improved interface: header, image, and footer are now global nodes.
- Updates to xsd object model.
  - Classes and attributes for Collada format are now generated (but not yet functional).

### Release 2.0.0 (May 4, 2009)

- Windows installer now detects Maya 2008 and Maya 2009, and their 64 bit variants, and can install itself into every Maya version that is found.
- Updates to the XML object model (affects CGF, DDS, KFM, NIF, and TGA).

- Class customizers are taken immediately from the format class, and not from separate modules — all code from customization modules has been moved into the main format classes. The result is that parsing is faster by about 50 percent.
- clsFilePath removed, as it is no longer used.
- Updates and fixes for the KFM format.
  - The Data element inherits from Header, and Header includes also all animations, so it is more straightforward to edit files.
  - The KFM files open again in QSkope.
- Updates for the CGF format.
  - CHUNK\_MAP no longer constructed in Data.\_\_init\_\_ but in a metaclass.
  - Deprecated functions in CgfFormat have been removed.
- Updates for the NIF format.
  - Synced nif.xml with nifskope's xml (includes fixes for Lazeska).
  - Removed deprecated scripts (niftexdump, nifdump, ffvt3rskinpartition, nifoptimize).
  - Fixed scaling bug on nifs whose tree has duplicate nodes. Scaling now no longer works recursively, unless you use the scaling spell which handles the duplication correctly.
- Updated module names to follow pep8 naming conventions: all modules have lower case names.

### Release 1.2.4 (Apr 21, 2009)

- Documentation is being converted to Sphinx. Currently some parts of the documentation are slightly broken with epydoc. Hopefully the migration will be complete in a month or so, resolving this issue.
- removed deprecated PyFFI.Spells code:
  - old style spells no longer supported
  - almost all old spells have been converted to the new spell system (the few remaining ones will be ported for the next release)
- nif:
  - nif optimizer can be run on folders from the windows context menu (right-click on any folder containing nifs and select “Optimize with PyFFI”)
  - synced nif.xml with upstream (adds support for Worldshift, bug fixes)
  - using weak references for Ptr type (this aids garbage collection)
  - added fix\_strip niftoaster spell which can remove branches selectively (feature request #2164309)
  - new getTangentSpace function for NiTriBasedGeom (works for both Oblivion and Fallout 3 style tangent spaces)
  - improved mergeSkeletonRoots function (will also merge roots of skins that have no bones in common, this helps a lot with Morrowind imports)
  - new sendDetachedGeometriesToNodePosition function and spell (helps a lot with Morrowind imports)
- tga:
  - added support for color map and image data in the xml
  - uses the new data model

- works again in QSkope
- xml object model:
  - added support for multiplication and division operators in expressions
- fixes for unicode support (prepares for py3k)

### Release 1.2.3 (Apr 2, 2009)

- removed reduce() calls (py3k compatibility)
- started converting print calls (py3k compatibility)
- removed relative imports (py3k compatibility)
- removed BSDiff module (not useful, very slow, use external bsdiff instead)
- nif:
  - fixed the update mopp spell for fallout 3 nifs
  - fixed addShape in bhkPackedNiTriStripsShape for fallout 3 nifs
  - niftoaster sends to stdout instead of stderr so output can be captured (reported by razorwing)

### Release 1.2.2 (Feb 15, 2009)

- cgf format:
  - fixed various regression bugs that prevented qskope to run on cgf files
  - updated to use the new data system

### Release 1.2.1 (Feb 2, 2009)

- nif format:
  - new addIntegerExtraData function for NiObjectNET

### Release 1.2.0 (Jan 25, 2009)

- installer directs to Python 2.5.4 if not installed
- using logging module for log messages
- nif format:
  - swapping tangents and binormals in xml; renaming binormals to bitangents (see <http://www.terathon.com/code/tangent.html>)
  - updates for Fallout 3 format
  - updated skin partition algorithm to work for Fallout 3
    - \* new triangles argument
    - \* new facemap argument to pre-define partitions (they will be split further if needed to meet constraints)
    - \* sort vertex weight list by weight in skin partitions (except if padbones is true; then sorted by bone index, to keep compatibility with ffvt3r)

- \* option to maximize bone sharing
- mopps take material indices into account and compute welding info (attempt to fix mopp multi-material issues, does not yet seem to work though)
- support for niftools bitflags by converting it to a bitstruct on the fly
- better algorithm for sending bones to bind position, including spells for automating this function over a large number of nifs
- disable fast inverse in bind pos functions to increase numerical precision
- new algorithm to sync geometry bind poses, along with spell (this fixes many issues with Morrowind imports and a few issues with Fallout 3 imports)
- more doctests for various functions
- a few more matrix functions (supNorm, subtraction)
- dds format:
  - updated to use the FileFormat.Data method (old inconvenient method removed)
- qskope:
  - refactored the tree model
  - all parenting functions are delegated to separate DetailTree and GlobalTree classes
  - the DetailNode and GlobalNode classes only implement the minimal functions to calculate the hierarchy, but no longer host the more advanced hierarchy functions and data (this will save memory and speed up regular use of pyffi outside qskope)
  - EdgeFilter for generic edge type filtering; this is now a parameter for every method that needs to list child nodes

### Release 1.1.0 (Nov 18, 2008)

- nif format:
  - a large number of functions have moved from the optimizer spell to to the main interface, so they can be easily used in other scripts without having to import this spell module (getInterchangeableTriShape, getInterchangeableTriStrips, isInterchangeable)
  - new convenience functions in NiObjectNET, NiAVObject, and NiNode (setExtraDatas, setProperties, setEffects, setChildren, etc.)
  - updates for Fallout 3
- niftoaster
  - new fix\_addtangentspace spell to add missing tangent space blocks
  - new fix\_deltangentspace spell to remove tangent space blocks
  - new fix\_texturepath spell to change / into and to fix corrupted newline characters (which sometimes resulted from older versions of nifskope) in NiSourceTexture file paths
  - new fix\_clampmaterialalpha spell
  - new fix\_detachhavoktristripsdata spell
  - the ffvt3r skin partition spell is now fix\_ffvt3rskinpartition
  - new opt\_cleanreflists spell

- new `opt_mergeduplicates` spell
- new `opt_geometry` spell
- the `optimize` spell is now simply implemented as a combination of other spells
- new internal implementation of `bsdiff` algorithm
- removed `cry dae` filter (an improved version of this filter is now bundled with `ColladaCGF`)
- reorganization of file format description code
  - all generic format description specific code has been moved to the `PyFFI.ObjectModels.FileFormat` module
  - all `xml/xsd` description specific code has been moved to the `PyFFI.ObjectModels.XML/XSD.FileFormat` modules
  - new `NifFormat.Data` class which now implements all the NIF file read and write functions
- completely revamped spell system, which makes it much easier to customize spells, and also enables more efficient implementations (thanks to `tazpn` for some useful suggestions, see issue #2122196)
  - toaster can call multiple spells at once
  - toaster takes spell classes instead of modules
  - for backwards compatibility, there is a class factory which turns any old spell module into a new spell class (the `Toaster` class will automatically convert any modules that it finds in its list of spells, so you do not need to be worried about call the factory explicitly)
  - early inspection of the header is possible, to avoid having to read all of the file if no blocks of interest are present
  - possibility to prevent the spell to cast itself on particular branches (mostly useful to speed up the spell casting process)
  - spells have callbacks for global initialization and finalization of data within the toaster
  - possibility to write output to a log file instead of to `sys.stdout`
  - better messaging system (auto indentation, list `nif` tree as spell runs)
  - support for spell hierarchies and spell grouping, in parallel or in series or any combination of these
- replaced ad hoc class customization with partial classes (still wip converting all the classes)
- `xml` object model expression parser
  - implemented `not` operator
  - expressions can combine multiple operators (only use this if the result is independent of the order in which these operators are applied)
  - new `<` and `>` operators
  - support for `vercond` attribute for `Fallout 3`
- started on a new object model based on an ANTLR parser of a grammar aimed at file format descriptions; this parser will eventually yield a more streamlined, more optimized, and more customizable version of the current `xml` object model (this is not yet bundled with the release, initial code is on `svn`)

### Release 1.0.5 (Sep 27, 2008)

- `niftoaster` optimize

- fix for materials named skin, envmap2, etc. (issue #2121098)
- fix for empty source textures in texdesc (issue #2118481)
- niftoaster
  - new spell to disable parallax (issue #2121283)
- toaster
  - new options `-diff` and `-patch` to create and apply patches; internal patcher uses `bsdifff` format, but you can also specify an arbitrary external diff/patch command via `-diff-cmd` and `-patch-cmd` options (the external command must take three arguments: `oldfile`, `newfile`, and `patchfile`); note that this is still in experimental stage, not ready for production use yet

### Release 1.0.4 (Sep 18, 2008)

- niftoaster optimize
  - morph data optimization (issue #2116594, fixes “bow” weapons)

### Release 1.0.3 (Sep 17, 2008)

- niftoaster optimize
  - detach `NiTriStripsData` from havok tree when block is shared with geometry data (fixes issue #2065018, `MiddleWolfRug01.NIF`)
  - fix in case merged properties had controllers (issue #2106668)
- fix writing of block order: `bhkConstraint` entities now always precede the constraint block (this also fixes the “falling sign” issue with the niftoaster optimize spell, issue #2068090)

### Release 1.0.2 (Sep 15, 2008)

- “negative mass” fix in inertia calculation

### Release 1.0.1 (Sep 12, 2008)

- small fix in uninstaller (didn’t remove `crydaefilter` script)
- `crydaefilter` converts `%20` back into spaces (as `rc` doesn’t recognize `%20`)
- bugfixes for niftoaster optimize spell (pyffi issue #2065018)

### Release 1.0.0 (Jul 24, 2008)

- new NSIS installer (this solves various issues with Vista, and also allows the documentation to be bundled)
- new filter to prepare collada (`.dae`) files for CryEngine2 resource compiler
  - wraps scenes into `CryExportNodes`
  - corrects `id/sid` naming
  - fixes `init_from` image paths
  - adds phong and lamber shader `sid`’s

- enforces material instance symbol to coincide with target
- sets material names in correct format for material library and physicalization
- started on support for collada format, by parsing the collada xsd schema description (this is still far from functional, but an initial parser is already included with the library, although it does not yet create any classes yet)
- fully optimal mopp generation for Oblivion (using the NifTools mopper.exe which is a command line utility that calls the mopp functions in the havok library, credit for writing the original wrapper goes to tazpn)
- minor updates to the nif.xml format description
- refactoring: library reorganized and some interfaces have been unified, also a lot of code duplication has been reduced; see README.TXT for more details on how to migrate from 0.x.x to 1.x.x
  - main format classes PyFFI.XXX have been moved to PyFFI.Formats.XXX
  - “XxxFormat.getVersion(cls, stream)” now always returns two integers, version and user\_version
  - “XxxFormat.read(self, stream, version, user\_version, ...)” for all formats
  - “XxxFormat.write(self, stream, version, user\_version, \*readresult, ...)” for all formats
  - in particular, CGF format game argument removed from read and write functions, but there are new CgfFormat.getGame and CgfFormat.getGameVersion functions to convert between (version, user\_version) and game
  - also for the CGF format, take care that getVersion no longer returns the file type. It is returned with the CgfFormat.read function, however there is a new CgfFormat.getFileType function, if you need to know the file type but you don’t want to parse the whole file
  - all XxxFormat classes derive from XmlFileFormat base class
  - common nameAttribute, walk, and walkFile functions
  - XxxTester modules have been moved to PyFFI.Spells.XXX, along with a much improved PyFFI.Spells module for toasters with loads of new options
  - some other internal code has been moved around
    - \* qskopelib -> PyFFI.QSkope
    - \* PyFFI.Bases -> PyFFI.ObjectModels.XML
  - a lot more internal code reorganization is in progress...
- much documentation has been added and improved

### Release 0.11.0 (Jun 16, 2008)

- nif:
  - fixed updateTangentSpace for nifs with zero normals
- cfg:
  - a lot of new physics stuff: MeshPhysicsDataChunk mostly decoded (finally!!)
  - fixes for reading and writing caf files (they are missing controller headers)
  - activated BoneMeshChunk and BoneInitialPosChunk for Crysis
- tga:
  - improved tga file detection heuristic

---

**Release 0.10.10 (Jun 8, 2008)**

- nif:
  - minor updates in xml
  - NiPixelFormat saveAsDDS function now also writes DXT compressed formats, that is, pixel formats 4, 5, and 6 (contributed by taarna23)
  - fixed nifoptimize for nifs with particle systems (niftools issue #1965936)
  - fixed nifoptimize for nifs with invalid normals (niftools issue #1987506)

**Release 0.10.9 (May 27, 2008)**

- nif:
  - bspline interpolator fix if no keys
  - fixed bspline scale bug

**Release 0.10.8 (Apr 13, 2008)**

- cgf:
  - more decoded of the mesh physics data chunk
- nif:
  - scaling for constraints
  - ported the A -> B spell from nifskope (see the new getTransformAB and updateAB methods)

**Release 0.10.7 (Apr 5, 2008)**

- cgf:
  - indices are unsigned shorts now (fixes geometry corruption on import of large models)
  - MeshChunk.setGeometry gives useful error message if number of vertices is too large
- nif:
  - nif.xml has minor updates in naming
  - added NiBSplineData access functions (experimental, interface could still change)
  - started on support for compressed B-spline data
  - fixed block order writing of bhkConstraints

**Release 0.10.6 (Mar 30, 2008)**

- tga: added missing xml file
- nif:
  - removed some question marks so the fields can be accessed easily in python interface
  - ControllerLink and StringPalette functions and doctests
  - quaternion functions in Matrix33 and Matrix44

- new bspline modules (still to implement later)
- fixed NiTransformInterpolator scaling bug
- cgf:
  - use tempfile for write test
- quick install batch file for windows

### Release 0.10.5 (Mar 27, 2008)

- qskope: make bitstructs editable
- cgf:
  - MeshChunk functions to get vertex colors (game independent).
  - Set vertex colors in setGeometry function.

### Release 0.10.4 (Mar 26, 2008)

- cgf:
  - fixed tangent space doctest
  - setGeometry argument sanity checking
  - setGeometry fix for empty material list
  - setGeometry tangent space update fix if there are no uvs

### Release 0.10.3 (Mar 24, 2008)

- added support for the TGA format
- tangentspace:
  - validate normals before calculating tangents
  - added new option to get orientation of tangent space relative to texture space (Crysis needs to know about this)
- installer detects Maya 2008 and copies relevant files to Maya Python directory for the Maya scripts to work
- cgf:
  - tangent space cgftoaster
  - new MeshChunk updateTangentSpace function

### Release 0.10.2 (Mar 22, 2008)

- cgf:
  - fixed “normals” problem by setting last component of tangents to -1.0
  - meshchunk function to get all material indices, per triangle (game independent)
  - scaling fixes for datastreamchunk, meshchunk, and meshsubsetschunk
  - fixed version of BreakablePhysicsChunk

- a few new findings in decoding the physics data (position and rotation)

### Release 0.10.1 (Mar 21, 2008)

- cgf:
  - some minor xml updates
  - setGeometry function for MeshChunk to set geometry for both Far Cry and Crysis in a unified way
  - uv.v opengl flip fix for Crysis MeshChunk data
- MathUtils: new function to calculate bounding box, center, and radius
- qskope: fixed bug which prevented setting material physics type to NONE

### Release 0.10.0 (Mar 8, 2008)

- cgf: ported A LOT of stuff from the Crysis Mod SDK 1.2; the most common CE2 chunks now read and write successfully

### Release 0.9.3 (Mar 7, 2008)

- cgf:
  - decoded a lot of geometry data
    - \* vertices
    - \* normals
    - \* vertex colors
    - \* uvs
    - \* mesh material info
  - started decoding many other chunk types
  - added chr chunk types so files containing them can be processed (the data is ignored)
  - started adding functions to MeshChunk to have unified access to geometry data for both Far Cry and Crysis cgf files
- windows installer registers chr extension with qskope

### Release 0.9.2 (Feb 26, 2008)

- full support for the xml enum tag type, with improved editor in qskope
- new common string types (shared between cgf and nif formats)
  - null terminated
  - fixed sized
  - variable sized starting with integer describing length
- qskope: no more duplicate ptr refs in global view
- qskope: refactored delegate editor system to be more transparent and much easier to extend

- cgf: crysis chunks have been partially decoded (still very much wip)
- cgf: added extra chunk size check on read to aid decoding
- dds: register dds extension with qskope on windows install
- nif: nifoptimize clamps material alpha to [0,1]

### Release 0.9.1 (Feb 22, 2008)

- full support for the xml bitstruct tag (for types that contain bit flags)
- added PyFFI.Formats.DDS library for dds file format
- nif: new function for NiPixelData to save image as dds file
- niftoaster: script for exporting images from NiPixelData blocks
- nifoptimize:
  - merge identical shape data blocks
  - remove empty NiNode children
  - update skin partition only if block already exists

### Release 0.9.0 (Feb 11, 2008)

- added PyFFI.Formats.KFM library for kfm file format
- cgf.xml and nif.xml updates
- new qBlockParent function to assign parents if the parent block does not contain a reference to the child, but the child contains a reference to the parent (as in MeshMorphTargetChunk and BoneInitialPosChunk)
- QSkope: root blocks sorted by reference number
- QSkope: added kfm format
- niftexdump: bug fixed when reading nifs that have textures without source

### Release 0.8.2 (Jan 28, 2008)

- fixed installer bug (nifoptimize would not launch from context menu)
- qskope:
  - handle back-references and shared blocks
  - blocks are now numbered
  - improved display references

### Release 0.8.1 (Jan 27, 2008)

- deep copy for structs and arrays
- nifoptimize:
  - detects cases where triangulated geometry performs better than stripified geometry (fixes a performance issue with non-smooth geometry reported by Lazarus)

- can now also optimize NiTriShapes
- throws away empty and/or duplicate children in NiNode lists

### Release 0.8.0 (Jan 27, 2008)

- qskope: new general purpose tool for visualizing files loaded with PyFFI
- cgf: corrected the bool implementation (using True/False rather than an int)
- nif: many xml updates, support for Culpa Innata, updates for emerge demo
- support for forward declaration of types (required for UnionBV)
- PyFFI.\_\_hexversion\_\_ for numeric representation of the version number

### Release 0.7.5 (Jan 14, 2008)

- added a DTD for the 'fileformat' document type, to validate the xml
- bits tag for bitstructs, instead of add tag, to allow validation
- cgf: write the chunk header table at start, for crysis
- nifoptimize:
  - new command line option '-x' to exclude blocks per type
  - fixes corrupted texture paths (that is, files that got corrupted with nifskope 1.0 due to the \r \n bug)
  - on windows, the script can now be called from the .nif context menu
  - accept both lower and upper case 'y' for confirmation
  - new command line option '-p' to pause after run
- niftoaster: fix reporting of file size difference in readwrite test
- bug fixed when writing nifs of version <= 3.1
- support for multiple 'Top Level Object' (roots) for nifs of version <= 3.1
- various xml fixes
  - new version 20.3.0.2 from emerge demo
  - NiMeshPSysData bugfix and simplification
  - replaced NiTimeController Target with unknown int to cope with invalid pointers in nif versions <= 3.1
- fixed bug nifmakehsl.py script
- fixed bug in nifdump.py script
- new post installation script for installing/uninstalling registry keys

### Release 0.7.4 (Dec 26, 2007)

- fix in nif xml for a long outstanding issue which caused some nifs with mopp shapes to fail
- fixed file size check bug in readwrite test for nif and cgf
- initial read and write support for crysis cgf files
- support for versions in structs

- updates for controller key types 6, 9, and 10, in cgf xml

### Release 0.7.3 (Dec 13, 2007)

- nif: fixed error message when encountering empty block type
- nif: dump script with block selection feature
- cgf: fix transform errors, ported matrix and vector operations from nif library

### Release 0.7.2 (Dec 3, 2007)

- NifTester: new raisereaderror argument which simplifies the older system and yields more instructive backtraces
- nif: better support for recent nif versions, if block sizes do not match with the number of bytes read then the bytes are skipped and a warning is printed, instead of raising an exception

### Release 0.7.1 (Nov 27, 2007)

- nif: fixed applyScale in bhkRigidBody

### Release 0.7 (Nov 19, 2007)

- fixed a problem locating the customized functions for Fedora 8 python which does not look in default locations besides sys.path
- new vector and matrix library under Utils (for internal use)
- new quick hull library for computing convex hulls
- new inertia library for computing mass, center of gravity, and inertia tensors of solid and hollow objects
- nif: fixed order of bhkCollisionObject when writing NIF files
- nif: new bhkRigidBody function for updating inertia, center of gravity, and mass, for all types of primitives

### Release 0.6 (Nov 3, 2007)

- nifoptimize removes duplicate property blocks
- reduced memory footprint in skin data center and radius calculation for the nif format
- new option to ignore strings when calculating hash
- code has been cleaned up using pylint
- added a lot more documentation
- refactored all common functions to take \*\*kwargs as argument
- read and write functions have the file stream as first non-keyword argument
- refactored and simplified attribute parsing, using a common \_filteredAttributeList method used by all methods that need to parse attributes; the version and user\_version checks are now also consistent over all functions (i.e. getRefs, getLinks, etc.)
- added more doctests

**Release 0.5.2 (Oct 25, 2007)**

- added hash functions (useful for identifying and comparing objects)

**Release 0.5.1 (Oct 19, 2007)**

- fixed a bug in the nif.xml file which prevented Oblivion skeleton.nif files to load

**Release 0.5 (Oct 19, 2007)**

- new functions to get block size
- various small bugs fixed
- nif: support for new versions (20.2.0.6, 20.2.0.7, 20.2.0.8, 20.3.0.3, 20.3.0.6, 20.3.0.9)
- nif: block sizes are now also written to the NIF files, improving support for writing 20.2.0.7+ nif versions
- nif: fixed flattenSkin bug (reported by Kikai)

**Release 0.4.9 (Oct 13, 2007)**

- nif: nifoptimize no longer raises an exception on test errors, unless you pass the -r option
- nif: nifoptimize will try to restore the original file if something goes wrong during write, so - in theory - it should no longer leave you with corrupt nifs; still it is recommended to keep your backups around just in case
- nif: niftesters recoded to accept arbitrary argument dictionaries; this could cause incompatibilities for people writing their own scripts, but the upgrade to the new system is fairly simple: check the niftemplate.py script
- nif: fixed bug in updateTangentSpace which caused an exception when uvs or normals were not present
- nif: doctest for unsupported blocks in nifs

**Release 0.4.8 (Oct 7, 2007)**

- cgf: MeshMorphTargetChunk is now supported too
- nif: new script (niftextdump.py) to dump texture and material info
- nif: added template script for quickly writing new nif scripts

**Release 0.4.7 (Oct 4, 2007)**

- nif: new optimizer script

**Release 0.4.6 (Sep 29, 2007)**

- nif and cgf documentation improved
- added a number of new doctests
- nif: new scripts
  - niftoaster.py for testing and modifying NIF files (contributed by wz)
  - nifvisualizer.py for visualizing nif blocks (contributed by wz)

- nifmakehsl.py for making hex workshop structure libraries for all nif versions
- nif: bundling NifVis and NifTester modules so you can make your own nif toasters and visualizers
- nif: fixed rare issue with skin partition calculation
- cgf: new script
  - cgftoaster.py for testing and modifying cgf files (similar to niftoaster.py)
- cgf: bundling CgfTester module so you can make your own cgf toasters
- cgf: various xml bugs fixed
- cgf: write support improved (but not entirely functional yet)
- cgf: material chunk custom function for extraction material shader and script
- Expression.py: support for empty string check in condition

### Release 0.4.5 (Sep 16, 2007)

- issue warning message instead of raising exception for improper rotation matrix in setScaleRotationTranslation
- fixed skin partition bug during merge
- skin partition bone index padding and sorting for Freedom Force vs. the 3rd Reich

### Release 0.4.4 (Sep 2, 2007)

- added mopp parser and simple mopp generator

### Release 0.4.3 (Aug 17, 2007)

- fixed bug that occurred if userver = 0 in the xml (fixes geometry morph data in NIF versions 20.0.0.4 and up)
- NIF:
  - tree() function has been extended
  - some minor cleanups and more documentation

### Release 0.4.2 (Aug 15, 2007)

- kwargs for getRefs
- NIF:
  - fixed bug in skin partition calculation
  - when writing NIF files the refs are written in sequence (instead of the links, so missing links will yield an exception, which is a good thing)
  - new functions to get list of extra data blocks and to add effect

**Release 0.4.1 (Aug 14, 2007)**

- NIF:
  - new function to add collision geometries to packed trisripsshape
  - fixed bug in bhkListShape.addShape

**Release 0.4 (Aug 12, 2007)**

- NIF:
  - new function updateBindPosition in NiGeometry to fix a geometry rest position from current bone positions
  - removed deprecated functions
  - (!) changed interface of addBone, no longer takes “transform” argument; use the new function updateBindPosition instead

**Release 0.3.4 (Aug 11, 2007)**

- improved documentation
- fixed the ‘in’ operator in Bases/Array.py
- NIF:
  - doctest for NiNode
  - flatten skin fix for skins that consist of multiple shapes
  - support for the most common oblivion havok blocks

**Release 0.3.3 (Aug 8, 2007)**

- NIF:
  - fixed a bug in the skin center and radius calculation
  - added copy function to Vector3
  - fixed NiGeometry doctest

**Release 0.3.2 (Aug 7, 2007)**

- simplified interface (still wip) by using keyword arguments for common functions such as read and write
- NIF:
  - fix for skin partition blocks in older nif versions such as Freedom Force vs. 3rd Reich
  - support for triangle skin partitions
  - added stitchstrips option for skin partitions
  - added a NiGeometry function to send bones to bind pose

### Release 0.3.1 (Aug 6, 2007)

- NIF:
  - new function for getting geometry skin deformation in rest pose
  - old rest pose functions are deprecated and will be removed from a future release

### Release 0.3 (Aug 2, 2007)

- NIF:
  - fixed an issue with writing skeleton.nif files
- CGF:
  - reading support for the most common blocks in static cgf files; experimental

### Release 0.2.1 (Jul 29, 2007)

- NIF:
  - fixed bug in getTransform
  - new option in findChain to fix block type

### Release 0.2 (Jul 29, 2007)

- fixed argument passing when writing arrays
- NIF: added getControllers function to NiObjectNET

### Release 0.1 (Jul 22, 2007)

- bug fixed when writing array of strings
- NIF
  - new function to add bones
  - XML update, supports newer versions from Emerge Demo

### Release 0.0 (Jul 7, 2007)

- first public release

## 1.7.8 Todo list

---

**Todo:** Write documentation.

---

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pyffi-tagnum/checkouts/update_theme/pyffi/spells/cgf/__init__.py:docstring of pyffi.spells.cgf, line 4.`)

---

**Todo:** Limit the size of shapes (see operation optimization mod for Oblivion!)

---

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pyffi-tagnum/checkouts/update_theme/pyffi/spells/nif/optimize.py:docstring of pyffi.spells.nif.optimize.SpellOptimizeGeometry.branchentry`, line 8.)

---

**Todo:**

- Write dedicated utilities to optimize particular games (start with Oblivion, maybe eventually also do Fallout 3, Morrowind, etc.).
- 

(The [original entry](#) is located in `../TODO.rst`, line 3.)

---

**Todo:** Aion caf format (MtlNameChunk header?).

---

(The [original entry](#) is located in `../TODO.rst`, line 9.)

---

**Todo:** refactoring plans

- what's up with `get_global_child_names`?
  - common base classes for `pyffi.object_models.xml.basic.BasicBase/StructBase` and `pyffi.object_models.xsd.SimpleType/ComplexType` (e.g. `pyffi.ObjectModel.SimpleType/ComplexType`)
  - derive `object_models.array_type` and `object_models.StructType` from common subclass `pyffi.object_models.ComplexType`, use these then as base classes for `object_models.xml.array` and `object_models.xml.struct_.StructBase`
  - use `pyffi.utils.graph` for all `object_models.XXX` implementations
  - upgrade QScope and XML model to use `GlobalNode` instead of the current ad hoc system with Refs
  - improve the abstract `object_models.Delegate` classes (i.e. naming, true abstract base classes, defining a common interface); also perhaps think about deriving these delegate classes from `TreeLeaf` (only leafs have editors!)?
  - ditch `version` and `user_version` from the `object_models` interface, and instead use `object_models.Data` as a global root element that contains all file information with a minimal format independent interface; implementation plan (this is already partially implemented, namely in the nif format):
    - use abstract `Data` and `Tree` base classes for the XSD parser, in subsequent 2.x.x releases
    - update the XML parser to follow the same scheme, when switching from 2.x.x to 3.0.0, and document the 2.x.x -> 3.0.0 migration strategy
    - deprecate the old methods (`XxxFormat.read`, `XxxFormat.write`, `XxxFormat.getVersion`, and so on) in 3.x.x
    - remove old method in 4.x.x
  - one of the aims is that qscope no longer relies on any `object_models.xml/object_models.xsd` specific implementations; if it only relies on the abstract base classes in `object_models.Graph` and `object_models.Data` then future expansions are a lot easier to cope with; in particular, qscope should never have to import from `object_models.XXX`, or `Formats.XXX`
- 

(The [original entry](#) is located in `../TODO.rst`, line 13.)

---

---

**Todo:** Doctests for all spells.

---

(The [original entry](#) is located in ../TODO.rst, line 62.)

---

**Todo:** Improve overall documentation, for instance:

- add docstrings for all spells
  - add docstrings for all spell methods
- 

(The [original entry](#) is located in ../TODO.rst, line 66.)

---

**Todo:**

- move all regression tests to the tests directory (but keep useful examples in the docstrings!)
  - add spell support for qscope directly using the pyffi.spells module
  - allow qscope to create new spells, from a user supplied spells module
    - custom spell module creation wizard (creates dir structure for user and stores it into the configuration)
    - custom spell creation wizard (adds new spell to user’s spell module)
    - automatic templates for typical spells
  - pep8 conventions
    - resolve all complaints from cheesecake’s pep8 checker
- 

(The [original entry](#) is located in ../TODO.rst, line 73.)

---

**Todo:**

- Write dedicated utilities to optimize particular games (start with Oblivion, maybe eventually also do Fallout 3, Morrowind, etc.).
- 

**Todo:** Aion caf format (MtlNameChunk header?).

---

**Todo:** refactoring plans

- what’s up with get\_global\_child\_names?
  - common base classes for pyffi.object\_models.xml.basic.BasicBase/StructBase and pyffi.object\_models.xsd.SimpleType/ComplexType (e.g. pyffi.ObjectModel.SimpleType/ComplexType)
  - derive object\_models.array\_type and object\_models.StructType from common subclass pyffi.object\_models.ComplexType, use these then as base classes for object\_models.xml.array and object\_models.xml.struct\_.StructBase
  - use pyffi.utils.graph for all object\_models.XXX implementations
  - upgrade QScope and XML model to use GlobalNode instead of the current ad hoc system with Refs
-

- improve the abstract `object_models.Delegate` classes (i.e. naming, true abstract base classes, defining a common interface); also perhaps think about deriving these delegate classes from `TreeLeaf` (only leafs have editors!)?
  - ditch `version` and `user_version` from the `object_models` interface, and instead use `object_models.Data` as a global root element that contains all file information with a minimal format independent interface; implementation plan (this is already partially implemented, namely in the `nif` format):
    - use abstract `Data` and `Tree` base classes for the XSD parser, in subsequent 2.x.x releases
    - update the XML parser to follow the same scheme, when switching from 2.x.x to 3.0.0, and document the 2.x.x -> 3.0.0 migration strategy
    - deprecate the old methods (`XxxFormat.read`, `XxxFormat.write`, `XxxFormat.getVersion`, and so on) in 3.x.x
    - remove old method in 4.x.x
  - one of the aims is that `qskope` no longer relies on any `object_models.xml/object_models.xsd` specific implementations; if it only relies on the abstract base classes in `object_models.Graph` and `object_models.Data` then future expansions are a lot easier to cope with; in particular, `qskope` should never have to import from `object_models.XXX`, or `Formats.XXX`
- 

**Todo:** Doctests for all spells.

---

**Todo:** Improve overall documentation, for instance:

- add docstrings for all spells
  - add docstrings for all spell methods
- 

**Todo:**

- move all regression tests to the tests directory (but keep useful examples in the docstrings!)
  - add spell support for `qskope` directly using the `pyffi.spells` module
  - allow `qskope` to create new spells, from a user supplied spells module
    - custom spell module creation wizard (creates dir structure for user and stores it into the configuration)
    - custom spell creation wizard (adds new spell to user's spell module)
    - automatic templates for typical spells
  - pep8 conventions
    - resolve all complaints from `cheesecake`'s pep8 checker
- 

## 1.7.9 Thanks

Special thanks go in particular to:

- Guido van Rossum, and with him many others, for Python, and in particular for having metaclasses in Python: metaclasses make PyFFI's implementation very easy.
- m4444x for `nifskope`, which has been an inspiration for PyFFI's xml based design, and of course also an inspiration for `QSkope`.

- wz for his support, and for testing of the library, when the first version was being written.
- seith for design of the windows installer artwork.
- Crytek for releasing the Far Cry SDK and Crysis SDK, which contains much information about the cgf file format. This has saved many months of hard work.
- Crytek and Bethesda for the great games they make.
- Havok, for releasing their SDK without which custom mopp generation would not have been possible.
- Karl Norby and Michael Summers for pyxsd, which forms the basis of the xsd object model, used for instance to support Collada.

### 1.7.10 Glossary

**PyFFI** Python File Format Interface. Also see *file*, *interface*, and *pyffi*.

**file** A byte stream.

**file format** See *interface*.

**file format interface** See *interface*.

**interface** An interface provides a semantic translation of a byte stream to an organized collection of named Python objects, which are typically bundled into a classes.

**spell** A transformation that can be applied to a file.

**toaster** Applies one or more spells to all files of all subdirectories of a given directory.

## 1.8 Indices and tables

- genindex
- modindex
- search

## PYTHON MODULE INDEX

### p

- pyffi, 7
- pyffi.formats, 7
  - pyffi.formats.bsa, 8
  - pyffi.formats.cgf, 11
  - pyffi.formats.dae, 32
  - pyffi.formats.dds, 34
  - pyffi.formats.egm, 37
  - pyffi.formats.egt, 41
  - pyffi.formats.esp, 44
  - pyffi.formats.kfm, 48
  - pyffi.formats.nif, 53
  - pyffi.formats.tga, 233
  - pyffi.formats.tri, 236
- pyffi.object\_models, 271
- pyffi.spells, 242
  - pyffi.spells.cgf, 243
  - pyffi.spells.dds, 243
  - pyffi.spells.kfm, 243
  - pyffi.spells.nif, 243
    - pyffi.spells.nif.check, 243
    - pyffi.spells.nif.dump, 243
    - pyffi.spells.nif.fix, 243
    - pyffi.spells.nif.modify, 254
    - pyffi.spells.nif.optimize, 251
  - pyffi.spells.tga, 264



## Symbols

- `_SpellDelBranchClasses` (class in `pyffi.spells.nif.modify`), 261
  - `__init__()` (`pyffi.spells.Spell` method), 265
  - `_branchinspect()` (`pyffi.spells.Spell` method), 265
  - `_datainspect()` (`pyffi.spells.Spell` method), 265
- ### A
- `a()` (`pyffi.formats.nif.NifFormat.ByteColor4` property), 79
  - `a()` (`pyffi.formats.nif.NifFormat.Color4` property), 81
  - `access()` (`pyffi.formats.nif.NifFormat.NiDataStream` property), 118
  - `ACOUSTIC_SPACE` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `ACOUSTIC_SPACE` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 200
  - `ACTION_DECREMENT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 204
  - `ACTION_INCREMENT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 204
  - `ACTION_INVERT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 204
  - `ACTION_KEEP` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 205
  - `ACTION_REPLACE` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 205
  - `ACTION_ZERO` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 205
  - `active()` (`pyffi.formats.nif.NifFormat.NiPSysModifier` property), 153
  - `active_material()` (`pyffi.formats.nif.NifFormat.NiRenderObject` property), 170
  - `ACTIVESPELLCLASSES` (`pyffi.spells.SpellGroupBase` attribute), 267
  - `actor_descs()` (`pyffi.formats.nif.NifFormat.NiPhysXPPropDesc` property), 166
  - `ACTORZONE` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `ACTORZONE` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 200
  - `add_asym_morph()` (`pyffi.formats.egm.EgmFormat.Data` method), 38
  - `add_bone()` (`pyffi.formats.nif.NifFormat.NiGeometry` method), 123
  - `add_child()` (`pyffi.formats.nif.NifFormat.NiNode` method), 133
  - `add_controlled_block()` (`pyffi.formats.nif.NifFormat.NiControllerSequence` method), 117
  - `add_controller()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 134
  - `add_effect()` (`pyffi.formats.nif.NifFormat.NiNode` method), 133
  - `add_extra_data()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 135
  - `add_integer_extra_data()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 135
  - `add_modifier()` (`pyffi.formats.tri.TriFormat.Header` method), 237
  - `add_morph()` (`pyffi.formats.tri.TriFormat.Header` method), 237
  - `add_property()` (`pyffi.formats.nif.NifFormat.NiAVObject` method), 106
  - `add_shape()` (`pyffi.formats.nif.NifFormat.bhkListShape` method), 219
  - `add_shape()` (`pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape` method), 221
  - `add_string()` (`pyffi.formats.nif.NifFormat.StringPalette` method), 205
  - `add_sym_morph()` (`pyffi.formats.egm.EgmFormat.Data` method), 38
  - `ADDONARM` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `ADDONCHEST` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `ADDONHEAD` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `ADDONLEG` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 90
  - `affected_node_list_pointers()` (`pyffi.formats.nif.NifFormat.NiDynamicEffect` method), 38

*property*), 119  
 affected\_nodes() (*pyffi.formats.nif.NifFormat.NiDynamicEffectproperty*), 180  
*property*), 119  
 ALIASDICT (*pyffi.spells.Toaster attribute*), 269  
 Alpha (*pyffi.formats.nif.NifFormat.LightingShaderControllerAttribute attribute*), 98  
 alpha() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 61  
 ALPHA\_BINARY (*pyffi.formats.nif.NifFormat.AlphaFormat attribute*), 54  
 ALPHA\_DEFAULT (*pyffi.formats.nif.NifFormat.AlphaFormat attribute*), 54  
 alpha\_format() (*pyffi.formats.nif.NifFormat.NiSourceTexture property*), 176  
 ALPHA\_NONE (*pyffi.formats.nif.NifFormat.AlphaFormat attribute*), 54  
 ALPHA\_SMOOTH (*pyffi.formats.nif.NifFormat.AlphaFormat attribute*), 54  
 alpha\_sort\_bound() (*pyffi.formats.nif.NifFormat.BSOrderedNode property*), 65  
 ALWAYS\_FACE\_CAMERA (*pyffi.formats.nif.NifFormat.BillboardMode attribute*), 77  
 ALWAYS\_FACE\_CENTER (*pyffi.formats.nif.NifFormat.BillboardMode attribute*), 77  
 always\_update() (*pyffi.formats.nif.NifFormat.NiGeomMorpherController property*), 121  
 ambient\_color() (*pyffi.formats.nif.NifFormat.NiLight property*), 127  
 ANIM\_STATIC (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 90  
 ANIM\_STATIC (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 188  
 animation\_type() (*pyffi.formats.nif.NifFormat.FurniturePositionormethod*), 93  
 ANIMSTATIC (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 200  
 append\_comp\_data() (*pyffi.formats.nif.NifFormat.NiBSplineData method*), 110  
 append\_float\_data() (*pyffi.formats.nif.NifFormat.NiBSplineData method*), 110  
 append\_short\_data() (*pyffi.formats.nif.NifFormat.NiBSplineData method*), 110  
 APPLY\_DECAL (*pyffi.formats.nif.NifFormat.ApplyMode attribute*), 55  
 APPLY\_HIGHLIGHT (*pyffi.formats.nif.NifFormat.ApplyMode attribute*), 55  
 APPLY\_HIGHLIGHT2 (*pyffi.formats.nif.NifFormat.ApplyMode attribute*), 55  
 apply\_mode() (*pyffi.formats.nif.NifFormat.NiTexturingProperty attribute*), 55  
 APPLY\_MODULATE (*pyffi.formats.nif.NifFormat.ApplyMode attribute*), 55  
 REPLACE (*pyffi.formats.nif.NifFormat.ApplyMode attribute*), 55  
 apply\_scale() (*pyffi.formats.cgf.CgfFormat.Chunk method*), 12  
 apply\_scale() (*pyffi.formats.cgf.CgfFormat.DataStreamChunk method*), 13  
 apply\_scale() (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17  
 apply\_scale() (*pyffi.formats.egm.EgmFormat.Data method*), 38  
 apply\_scale() (*pyffi.formats.egm.EgmFormat.MorphRecord method*), 39  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkBoxShape method*), 214  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkCapsuleShape method*), 215  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkConvexVerticesShape method*), 218  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkLimitedHingeConstraint method*), 218  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkMalleableConstraint method*), 219  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkRagdollConstraint method*), 223  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkRigidBody method*), 224  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkSphereShape method*), 225  
 apply\_scale() (*pyffi.formats.nif.NifFormat.bhkTransformShape method*), 225  
 apply\_scale() (*pyffi.formats.nif.NifFormat.BSBound method*), 56  
 apply\_scale() (*pyffi.formats.nif.NifFormat.hkPackedNiTriStripsData method*), 226  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiAVObject method*), 106  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiBSplineCompTransform method*), 109  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiBSplineTransformInterpo method*), 111  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiGeometryData method*), 124  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiKeyframeData method*), 126  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiMorphData method*), 130  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiObject method*), 134  
 apply\_scale() (*pyffi.formats.nif.NifFormat.NiSkinData method*), 173

- apply\_scale() (*pyffi.formats.nif.NifFormat.NiTransformInterpolator* property), 78  
     *method*), 182  
 apply\_scale() (*pyffi.formats.tri.TriFormat.MorphRecombine* property), 238  
 ARCHIVE\_CLASSES (*pyffi.formats.nif.NifFormat* attribute), 53  
 ARCHIVE\_CLASSES (*pyffi.object\_models.FileFormat* attribute), 271  
 as\_list() (*pyffi.formats.cgf.CgfFormat.Matrix33* property), 15  
     *method*), 16  
 as\_list() (*pyffi.formats.cgf.CgfFormat.Matrix44* property), 16  
     *method*), 16  
 as\_list() (*pyffi.formats.nif.NifFormat.InertiaMatrix* property), 96  
     *method*), 96  
 as\_list() (*pyffi.formats.nif.NifFormat.Matrix33* property), 99  
     *method*), 99  
 as\_list() (*pyffi.formats.nif.NifFormat.Matrix44* property), 100  
     *method*), 100  
 as\_list() (*pyffi.formats.nif.NifFormat.TexCoord* property), 208  
     *method*), 208  
 as\_list() (*pyffi.formats.nif.NifFormat.Vector3* property), 211  
     *method*), 211  
 as\_list() (*pyffi.formats.nif.NifFormat.Vector4* property), 212  
     *method*), 212  
 as\_tuple() (*pyffi.formats.cgf.CgfFormat.Matrix33* property), 15  
     *method*), 15  
 as\_tuple() (*pyffi.formats.cgf.CgfFormat.Matrix44* property), 16  
     *method*), 16  
 as\_tuple() (*pyffi.formats.nif.NifFormat.InertiaMatrix* property), 96  
     *method*), 96  
 as\_tuple() (*pyffi.formats.nif.NifFormat.Matrix33* property), 99  
     *method*), 99  
 as\_tuple() (*pyffi.formats.nif.NifFormat.Matrix44* property), 100  
     *method*), 100  
 as\_tuple() (*pyffi.formats.nif.NifFormat.Vector3* property), 211  
     *method*), 211  
 as\_tuple() (*pyffi.formats.nif.NifFormat.Vector4* property), 212  
     *method*), 212  
 aspect\_ratio() (*pyffi.formats.nif.NifFormat.NiPSysData* property), 148  
     *property*), 148  
 assign() (*pyffi.formats.nif.NifFormat.Vector3* property), 211  
     *method*), 211  
 atom\_sizes() (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* property), 67  
     *property*), 67  
 attenuation() (*pyffi.formats.nif.NifFormat.NiPSysFieldModifier* property), 151  
     *property*), 151  
 av\_object() (*pyffi.formats.nif.NifFormat.AVObject* property), 53  
     *property*), 53  
 AVOID\_BOX (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188  
 AVOIDBOX (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90  
 AVOIDBOX (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 200  
 axle\_a() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
     *property*), 96  
 axle\_b() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
     *property*), 96  
**B**  
 b() (*pyffi.formats.nif.NifFormat.ByteColor3* property), 79  
     *property*), 79  
 b() (*pyffi.formats.nif.NifFormat.ByteColor4* property), 79  
     *property*), 79  
 b() (*pyffi.formats.nif.NifFormat.Color3* property), 81  
     *property*), 81  
 b() (*pyffi.formats.nif.NifFormat.Color4* property), 81  
     *property*), 81  
 b() (*pyffi.formats.nif.NifFormat.TBC* property), 208  
     *property*), 208  
 BACK\_WEAPON (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188  
     *attribute*), 188  
 BACK\_WEAPON2 (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188  
     *attribute*), 188  
 backward() (*pyffi.formats.nif.NifFormat.Key* property), 96  
     *property*), 96  
 ball\_and\_socket() (*pyffi.formats.nif.NifFormat.bhkBallAndSocketConstraint* property), 213  
     *property*), 213  
 ball\_and\_socket() (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207  
     *property*), 207  
 BallAndSocket (*pyffi.formats.nif.NifFormat.hkConstraintType* attribute), 226  
     *attribute*), 226  
 base() (*pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator* property), 109  
     *property*), 109  
 BASE\_BV (*pyffi.formats.nif.NifFormat.BoundVolumeType* attribute), 77  
     *attribute*), 77  
 BASE\_MAP (*pyffi.formats.nif.NifFormat.TexType* attribute), 210  
     *attribute*), 210  
 base\_scale() (*pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier* property), 152  
     *property*), 152  
 base\_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 180  
     *property*), 180  
 behaviour\_graph\_file() (*pyffi.formats.nif.NifFormat.BSBehaviorGraphExtraData* property), 55  
     *property*), 55  
 behaviour\_graph\_file() (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints* property), 93  
     *property*), 93  
 bezier\_triangle() (*pyffi.formats.nif.NifFormat.NiBezierMesh* property), 111  
     *property*), 111  
 bias() (*pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator* property), 109  
     *property*), 109  
 big\_tris() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 216  
     *property*), 216  
 big\_verts() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 216  
     *property*), 216

billboard\_mode() (pyffi.formats.nif.NifFormat.NiBillboardNode property), 112

binary\_data() (pyffi.formats.nif.NifFormat.NiBinaryExtraData property), 113

BIPED (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 90

BIPED (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 188

BIPED (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 200

BIPED\_NO\_CC (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 200

bits\_per\_channel() (pyffi.formats.nif.NifFormat.ChannelData property), 80

bits\_per\_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 216

bits\_per\_w\_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 216

blending() (pyffi.formats.cgf.CgfFormat.BoneLink property), 11

block\_index() (pyffi.formats.nif.NifFormat.AdditionalDataInfo property), 54

block\_infos() (pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData property), 68

block\_infos() (pyffi.formats.nif.NifFormat.NiAdditionalGeometryData property), 106

block\_offsets() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 53

block\_offsets() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock property), 67

block\_size() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 53

blocks() (pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData property), 68

blocks() (pyffi.formats.nif.NifFormat.NiAdditionalGeometryData property), 106

BlockTypeIndex (pyffi.formats.nif.NifFormat attribute), 77

BODY (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 90

BODY (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 188

body() (pyffi.formats.nif.NifFormat.bhkNiCollisionObject property), 220

body\_part() (pyffi.formats.nif.NifFormat.BodyPartList property), 77

bomb\_axis() (pyffi.formats.nif.NifFormat.NiPSysBombModifier property), 147

bomb\_object() (pyffi.formats.nif.NifFormat.NiPSysBombModifier property), 147

bone() (pyffi.formats.cgf.CgfFormat.BoneLink property), 11

bounds() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 175

bounds() (pyffi.formats.nif.NifFormat.NiTriShapeSkinController property), 185

bone\_l\_o\_d\_count() (pyffi.formats.nif.NifFormat.BSBoneLODEExtraData property), 56

bone\_l\_o\_d\_info() (pyffi.formats.nif.NifFormat.BSBoneLODEExtraData property), 56

bone\_name() (pyffi.formats.nif.NifFormat.BoneLOD property), 77

bone\_transforms() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 175

compressedMeshShapeData (pyffi.formats.nif.NifFormat.bhkRagdollTemplate property), 223

bones() (pyffi.formats.nif.NifFormat.BSTreeNode property), 75

bones() (pyffi.formats.nif.NifFormat.NiFurSpringController property), 121

bones() (pyffi.formats.nif.NifFormat.NiSkinInstance property), 174

bones() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 175

bones() (pyffi.formats.nif.NifFormat.NiTriShapeSkinController property), 185

bones\_1() (pyffi.formats.nif.NifFormat.BSTreeNode property), 75

bones\_2() (pyffi.formats.nif.NifFormat.NiFurSpringController property), 121

bool (pyffi.formats.cgf.CgfFormat attribute), 29

bool\_value() (pyffi.formats.nif.NifFormat.NiBlendBoolInterpolator property), 113

boolean\_data() (pyffi.formats.nif.NifFormat.NiBoolInterpolator property), 115

boolean\_extra\_data() (pyffi.formats.nif.NifFormat.NiBooleanExtraData property), 115

bounce() (pyffi.formats.nif.NifFormat.NiPSysCollider property), 147

bound() (pyffi.formats.nif.NifFormat.NiMesh property), 128

bound\_center() (pyffi.formats.nif.NifFormat.NiScreenLODData property), 172

bound\_radius() (pyffi.formats.nif.NifFormat.NiScreenLODData property), 172

bounding\_volume() (pyffi.formats.nif.NifFormat.NiCollisionData property), 117

bounding\_volumes() (pyffi.formats.nif.NifFormat.UnionBV property), 211

bounds\_max() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 216

`bounds_min()` (`pyffi.formats.nif.NifFormat.bhkCompressedMeshShape` attribute), 216  
`box()` (`pyffi.formats.nif.NifFormat.BoundingBoxVolume` attribute), 78  
`BOX_BV` (`pyffi.formats.nif.NifFormat.BoundingBoxVolumeType` attribute), 77  
`BP_BRAIN` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 56  
`BP_HEAD` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 56  
`BP_HEAD2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 56  
`BP_LEFTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 56  
`BP_LEFTARM2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_LEFTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_LEFTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_LEFTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_RIGHTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_RIGHTARM2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_RIGHTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_RIGHTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_RIGHTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_BRAIN` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_HEAD` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_HEAD2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_LEFTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_LEFTARM2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_LEFTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_LEFTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_LEFTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_RIGHTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_RIGHTARM2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_RIGHTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_RIGHTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_SECTIONCAP_RIGHTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSO` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_BRAIN` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_HEAD` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_HEAD2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_LEFTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_LEFTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_LEFTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_LEFTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_RIGHTARM` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_RIGHTARM2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_RIGHTLEG` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_RIGHTLEG2` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57  
`BP_TORSOCAP_RIGHTLEG3` (`pyffi.formats.nif.NifFormat.BSDismemberBodyPartType` attribute), 57

BP\_TORSOCAP\_RIGHTLEG3 *branchentry()* (*pyffi.spells.nif.fix.SpellFFVT3RSkinPartition*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 245  
 attribute), 57 *branchentry()* (*pyffi.spells.nif.fix.SpellFFVT3RSkinPartition*  
*method*), 245  
 BP\_TORSOSECTION\_BRAIN *branchentry()* (*pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 251  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.fix.SpellFixMopp*  
*method*), 249  
 BP\_TORSOSECTION\_HEAD *branchentry()* (*pyffi.spells.nif.fix.SpellMergeSkeletonRoots*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 247  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.fix.SpellScale*  
*method*), 248  
 BP\_TORSOSECTION\_HEAD2 *branchentry()* (*pyffi.spells.nif.modify.SpellAddStencilProperty*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 248  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellAddStencilProperty*  
*method*), 263  
 BP\_TORSOSECTION\_LEFTARM *branchentry()* (*pyffi.spells.nif.modify.SpellChangeBonePriorities*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 259  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellCollisionMaterial*  
*method*), 256  
 BP\_TORSOSECTION\_LEFTARM2 *branchentry()* (*pyffi.spells.nif.modify.SpellCollisionType*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 256  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellCollisionType*  
*method*), 255  
 BP\_TORSOSECTION\_LEFTLEG *branchentry()* (*pyffi.spells.nif.modify.SpellDelBranches*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 261  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellDelInterpolatorTransformD*  
*method*), 260  
 BP\_TORSOSECTION\_LEFTLEG2 *branchentry()* (*pyffi.spells.nif.modify.SpellDelVertexColor*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 260  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellDelVertexColor*  
*method*), 263  
 BP\_TORSOSECTION\_LEFTLEG3 *branchentry()* (*pyffi.spells.nif.modify.SpellDisableParallax*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 262  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellReverseAnimation*  
*method*), 258  
 BP\_TORSOSECTION\_RIGHTARM *branchentry()* (*pyffi.spells.nif.modify.SpellScaleAnimationTime*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 258  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellScaleAnimationTime*  
*method*), 257  
 BP\_TORSOSECTION\_RIGHTARM2 *branchentry()* (*pyffi.spells.nif.modify.SpellSetInterpolatorTransRotSca*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 260  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.modify.SpellSetInterpolatorTransRotSca*  
*method*), 260  
 BP\_TORSOSECTION\_RIGHTLEG *branchentry()* (*pyffi.spells.nif.optimize.SpellCleanRefLists*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 251  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.optimize.SpellDelUnusedBones*  
*method*), 253  
 BP\_TORSOSECTION\_RIGHTLEG2 *branchentry()* (*pyffi.spells.nif.optimize.SpellMergeDuplicates*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 252  
 attribute), 58 *branchentry()* (*pyffi.spells.nif.optimize.SpellMergeDuplicates*  
*method*), 252  
 BP\_TORSOSECTION\_RIGHTLEG3 *branchentry()* (*pyffi.spells.nif.optimize.SpellOptimizeGeometry*  
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 253  
 attribute), 58 *branchentry()* (*pyffi.spells.Spell* *method*), 265  
 BRANCH\_CLASSES\_TO\_BE\_DELETED *branchentry()* (*pyffi.spells.SpellGroupParallelBase*  
 (*pyffi.spells.nif.modify.\_SpellDelBranchClasses* *method*), 267  
 attribute), 261 *branchentry()* (*pyffi.spells.SpellGroupSeriesBase*  
*method*), 268  
*branchentry()* (*pyffi.spells.nif.fix.SpellAddTangentSpace* *method*), 268  
*branchentry()* (*pyffi.spells.Spell* *method*), 265  
*branchentry()* (*pyffi.spells.nif.fix.SpellClampMaterialAlpha* *method*), 267  
*branchentry()* (*pyffi.spells.SpellGroupParallelBase*  
*method*), 267  
*branchentry()* (*pyffi.spells.nif.fix.SpellCleanStringPalette* *method*), 249  
*branchinspect()* (*pyffi.spells.nif.fix.SpellAddTangentSpace*  
*method*), 244  
*branchentry()* (*pyffi.spells.nif.fix.SpellDelTangentSpace* *method*), 243  
*branchinspect()* (*pyffi.spells.nif.fix.SpellClampMaterialAlpha*  
*method*), 246  
*branchentry()* (*pyffi.spells.nif.fix.SpellDetachHavokTriStripsData* *method*), 246  
*branchinspect()* (*pyffi.spells.nif.fix.SpellCleanStringPalette*  
*method*), 246

- method*), 250  
 branchinspect () (*pyffi.spells.nif.fix.SpellDelTangentSpace* *method*), 244  
 branchinspect () (*pyffi.spells.nif.fix.SpellDetachHavokTriStripsData* *attribute*), 77  
*method*), 246  
 branchinspect () (*pyffi.spells.nif.fix.SpellFFVT3RSkinPartition* *property*), 70  
*method*), 245  
 branchinspect () (*pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots* *attribute*), 198  
*method*), 251  
 branchinspect () (*pyffi.spells.nif.fix.SpellMergeSkeletonRoots* *attribute*), 198  
*method*), 247  
 branchinspect () (*pyffi.spells.nif.fix.SpellScale* *attribute*), 198  
*method*), 248  
 branchinspect () (*pyffi.spells.nif.modify.SpellAddStencilBump* *attribute*), 198  
*method*), 263  
 branchinspect () (*pyffi.spells.nif.modify.SpellChangeBonePosition* *attribute*), 198  
*method*), 259  
 branchinspect () (*pyffi.spells.nif.modify.SpellCollisionMaterial* *attribute*), 198  
*method*), 256  
 branchinspect () (*pyffi.spells.nif.modify.SpellCollisionType* *attribute*), 198  
*method*), 256  
 branchinspect () (*pyffi.spells.nif.modify.SpellDelInterpolatorTransform* *attribute*), 102  
*method*), 261  
 branchinspect () (*pyffi.spells.nif.modify.SpellDelSkinShapes* *attribute*), 102  
*method*), 262  
 branchinspect () (*pyffi.spells.nif.modify.SpellDelVertexColor* *attribute*), 102  
*method*), 263  
 branchinspect () (*pyffi.spells.nif.modify.SpellDisableParallax* *attribute*), 102  
*method*), 262  
 branchinspect () (*pyffi.spells.nif.modify.SpellReverseAnimation* *tribute*), 210  
*method*), 258  
 branchinspect () (*pyffi.spells.nif.modify.SpellScaleAnimationTime* *pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 180  
*method*), 257  
 branchinspect () (*pyffi.spells.nif.modify.SpellSetInterpolatorTransform* *Scale* *scale* () *pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 180  
*method*), 260  
 branchinspect () (*pyffi.spells.nif.optimize.SpellCleanRefLists* *property*), 180  
*method*), 252  
 branchinspect () (*pyffi.spells.nif.optimize.SpellDelUnusedBones* *pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 180  
*method*), 254  
 branchinspect () (*pyffi.spells.nif.optimize.SpellMergeDuplicatas* *map\_texture* () *pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 180  
*method*), 252  
 branchinspect () (*pyffi.spells.nif.optimize.SpellOptimizeGeometry* *property*), 180  
*method*), 253  
 branchinspect () (*pyffi.spells.Spell* *method*), 265  
 branchinspect () (*pyffi.spells.SpellGroupParallelBase* *byte* (*pyffi.formats.cgf.CgfFormat* *attribute*), 29  
*method*), 268  
*byte* (*pyffi.formats.dds.DdsFormat* *attribute*), 36  
 branchinspect () (*pyffi.spells.SpellGroupSeriesBase* *byte* (*pyffi.formats.egm.EgmFormat* *attribute*), 40  
*method*), 268  
*byte* (*pyffi.formats.egt.EgtFormat* *attribute*), 43  
*byte* (*pyffi.formats.esp.EspFormat* *attribute*), 47  
*byte* (*pyffi.formats.kfm.KfmFormat* *attribute*), 51  
*byte* (*pyffi.formats.nif.NifFormat* *attribute*), 226  
*byte* (*pyffi.formats.tga.TgaFormat* *attribute*), 235  
*byte* (*pyffi.formats.tri.TriFormat* *attribute*), 239  
*byte\_1* () (*pyffi.formats.nif.NifFormat.BSProceduralLightningController*



- CgfFormat.Ref (class in *pyffi.formats.cgf*), 26
- CgfFormat.ScenePropsChunk (class in *pyffi.formats.cgf*), 27
- CgfFormat.SizedString (class in *pyffi.formats.cgf*), 27
- CgfFormat.String128 (class in *pyffi.formats.cgf*), 28
- CgfFormat.String16 (class in *pyffi.formats.cgf*), 28
- CgfFormat.String256 (class in *pyffi.formats.cgf*), 28
- CgfFormat.String32 (class in *pyffi.formats.cgf*), 28
- CgfFormat.String64 (class in *pyffi.formats.cgf*), 28
- CgfFormat.Tangent (class in *pyffi.formats.cgf*), 28
- CgfFormat.UnknownAAFC0005Chunk (class in *pyffi.formats.cgf*), 29
- CgfFormat.UV (class in *pyffi.formats.cgf*), 29
- CgfFormat.UVFace (class in *pyffi.formats.cgf*), 29
- CgfFormat.Vector3 (class in *pyffi.formats.cgf*), 29
- CGFXMLPATH, 7
- CHAIN (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90
- changed() (*pyffi.spells.SpellGroupParallelBase* property), 268
- changed() (*pyffi.spells.SpellGroupSeriesBase* property), 268
- channel\_offset() (*pyffi.formats.nif.NifFormat.AdditionalDataInfor* attribute), 208
- char (*pyffi.formats.cgf.CgfFormat* attribute), 29
- char (*pyffi.formats.dds.DdsFormat* attribute), 36
- char (*pyffi.formats.egm.EgmFormat* attribute), 40
- char (*pyffi.formats.egt.EgtFormat* attribute), 43
- char (*pyffi.formats.esp.EspFormat* attribute), 47
- char (*pyffi.formats.kfm.KfmFormat* attribute), 51
- char (*pyffi.formats.nif.NifFormat* attribute), 226
- char (*pyffi.formats.tga.TgaFormat* attribute), 235
- char (*pyffi.formats.tri.TriFormat* attribute), 239
- CHAR\_CONTROLLER (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188
- CHARCONTROLLER (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90
- CHARCONTROLLER (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201
- child() (*pyffi.formats.nif.NifFormat.Ni3dsAnimationNode* property), 104
- child\_2() (*pyffi.formats.nif.NifFormat.NiEnvMappedTriShape* attribute), 119
- child\_3() (*pyffi.formats.nif.NifFormat.NiEnvMappedTriShape* attribute), 119
- children() (*pyffi.formats.nif.NifFormat.NiEnvMappedTriShape* attribute), 119
- CHNL\_ALPHA (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_BLUE (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_COMPRESSED (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_EMPTY (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_GREEN (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_INDEX (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- CHNL\_RED (*pyffi.formats.nif.NifFormat.ChannelType* attribute), 80
- chunk\_materials() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 216
- chunk\_transforms() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 216
- chunks() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217
- clamp() (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104
- clamp\_mode() (*pyffi.formats.nif.NifFormat.TexDesc* property), 208
- CLAMP\_S\_CLAMP\_T (*pyffi.formats.nif.NifFormat.TexClampMode* attribute), 208
- CLAMP\_S\_WRAP\_T (*pyffi.formats.nif.NifFormat.TexClampMode* attribute), 208
- cleanreflist() (*pyffi.spells.nif.optimize.SpellCleanRefLists* method), 252
- clear() (*pyffi.formats.nif.NifFormat.StringPalette* method), 206
- cli() (*pyffi.spells.Toaster* method), 269
- clipping\_plane() (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179
- cloning\_behavior() (*pyffi.formats.nif.NifFormat.NiDataStream* property), 118
- CLONING\_COPY (*pyffi.formats.nif.NifFormat.CloningBehavior* attribute), 80
- CLONING\_COPY (*pyffi.formats.nif.NifFormat.CloningBehavior* attribute), 80
- CLONING\_SHARE (*pyffi.formats.nif.NifFormat.CloningBehavior* attribute), 80
- CLOUD\_TRAP (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90
- CLOUD\_TRAP (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188
- CLOUD\_TRAP (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201
- CLUTTER (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90
- CLUTTER (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 188
- CLUTTER (*pyffi.formats.nif.NifFormat.SkyrimLayer* at-

tribute), 201  
 CM\_NOTEST (*pyffi.formats.nif.NifFormat.CollisionMode attribute*), 81  
 CM\_USE\_ABV (*pyffi.formats.nif.NifFormat.CollisionMode attribute*), 81  
 CM\_USE\_NIBOUND (*pyffi.formats.nif.NifFormat.CollisionMode attribute*), 81  
 CM\_USE\_OBB (*pyffi.formats.nif.NifFormat.CollisionMode attribute*), 81  
 CM\_USE\_TRI (*pyffi.formats.nif.NifFormat.CollisionMode attribute*), 81  
 collider () (*pyffi.formats.nif.NifFormat.NiPSysColliderManager property*), 148  
 collider\_object () (*pyffi.formats.nif.NifFormat.NiPSysCollider property*), 147  
 colliders () (*pyffi.formats.nif.NifFormat.NiPSSimulatorColliders property*), 143  
 collision\_mode () (*pyffi.formats.nif.NifFormat.NiCollisionData property*), 117  
 collision\_type () (*pyffi.formats.nif.NifFormat.BoundingBox property*), 78  
 COLLISIONBOX (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 90  
 COLLISIONBOX (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201  
 color\_1\_end\_percent () (*pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property*), 66  
 color\_1\_start\_percent () (*pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property*), 66  
 color\_2\_end\_percent () (*pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property*), 67  
 color\_2\_start\_percent () (*pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property*), 67  
 color\_data () (*pyffi.formats.nif.NifFormat.NiParticleColorModifier property*), 157  
 color\_data () (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159  
 color\_keys () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property*), 144  
 color\_loop\_behavior () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property*), 144  
 colors () (*pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property*), 67  
 complete\_points () (*pyffi.formats.nif.NifFormat.NiMeshModifier property*), 129  
 component\_formats () (*pyffi.formats.nif.NifFormat.NiDataStream property*), 118  
 component\_semantics () (*pyffi.formats.nif.NifFormat.MeshData property*), 101  
 CONEPROJECTILE (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201  
 CONST\_KEY (*pyffi.formats.nif.NifFormat.KeyType attribute*), 97  
 constant\_attenuation () (*pyffi.formats.nif.NifFormat.NiPointLight property*), 169  
 constant\_maint () (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property*), 223  
 controlled\_blocks () (*pyffi.formats.nif.NifFormat.NiSequence property*), 172  
 controllers () (*pyffi.formats.nif.NifFormat.NiParticleModifier property*), 158  
 controller\_data () (*pyffi.formats.nif.NifFormat.NiFloatExtraDataController property*), 120  
 controller\_sequences () (*pyffi.formats.nif.NifFormat.NiControllerManager property*), 117  
 controls\_base\_skeleton () (*pyffi.formats.nif.NifFormat.BSBehaviorGraphExtraData property*), 55  
 coordinate\_generation\_type () (*pyffi.formats.nif.NifFormat.NiTextureEffect property*), 179  
 count () (*pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property*), 104  
 count\_1 () (*pyffi.formats.nif.NifFormat.NiBezierMesh property*), 111  
 count\_2 () (*pyffi.formats.nif.NifFormat.NiBezierMesh property*), 111  
 create () (*pyffi.formats.nif.NifFormat.DataStreamAccess property*), 86  
 create\_mutable () (*pyffi.formats.nif.NifFormat.DataStreamAccess property*), 86  
 cpu\_write\_static () (*pyffi.formats.nif.NifFormat.DataStreamAccess property*), 86  
 cpu\_write\_static\_initialized () (*pyffi.formats.nif.NifFormat.DataStreamAccess property*), 86  
 cpu\_write\_volatile () (*pyffi.formats.nif.NifFormat.DataStreamAccess property*), 86  
 creator () (*pyffi.formats.nif.NifFormat.ExportInfo property*), 88

`crossproduct()` (*pyffi.formats.nif.NifFormat.Vector3* data () (*pyffi.formats.nif.NifFormat.BSMultiBound* method), 211 property), 64  
`CT_MUTABLE` (*pyffi.formats.nif.NifFormat.ConsistencyType* data () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* attribute), 83 property), 67  
`CT_STATIC` (*pyffi.formats.nif.NifFormat.ConsistencyType* data () (*pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrlr* attribute), 83 property), 66  
`CT_VOLATILE` (*pyffi.formats.nif.NifFormat.ConsistencyType* data () (*pyffi.formats.nif.NifFormat.BSTreadTransfInterpolator* attribute), 83 property), 74  
`cumulative()` (*pyffi.formats.nif.NifFormat.NiControllerManager* data () (*pyffi.formats.nif.NifFormat.NiAlphaController* property), 117 property), 107  
`CUSTOM_PICK_1` (*pyffi.formats.nif.NifFormat.OblivionLayer* data () (*pyffi.formats.nif.NifFormat.NiBinaryVoxelExtraData* attribute), 188 property), 113  
`CUSTOM_PICK_2` (*pyffi.formats.nif.NifFormat.OblivionLayer* data () (*pyffi.formats.nif.NifFormat.NiBoolData* attribute), 188 property), 115  
`CUSTOM_PICK1` (*pyffi.formats.nif.NifFormat.Fallout3Layer* data () (*pyffi.formats.nif.NifFormat.NiBoolInterpolator* attribute), 90 property), 115  
`CUSTOM_PICK1` (*pyffi.formats.nif.NifFormat.SkyrimLayer* data () (*pyffi.formats.nif.NifFormat.NiColorData* attribute), 201 property), 117  
`CUSTOM_PICK2` (*pyffi.formats.nif.NifFormat.Fallout3Layer* data () (*pyffi.formats.nif.NifFormat.NiColorExtraData* attribute), 90 property), 117  
`CUSTOM_PICK2` (*pyffi.formats.nif.NifFormat.SkyrimLayer* data () (*pyffi.formats.nif.NifFormat.NiDataStream* attribute), 201 property), 118  
`cutoff_angle()` (*pyffi.formats.nif.NifFormat.NiSpotLight* data () (*pyffi.formats.nif.NifFormat.NiFloatData* property), 177 property), 120  
`CYCLE_CLAMP` (*pyffi.formats.nif.NifFormat.CycleType* data () (*pyffi.formats.nif.NifFormat.NiFloatInterpolator* attribute), 84 property), 121  
`CYCLE_LOOP` (*pyffi.formats.nif.NifFormat.CycleType* data () (*pyffi.formats.nif.NifFormat.NiFloatsExtraData* attribute), 84 property), 121  
`CYCLE_REVERSE` (*pyffi.formats.nif.NifFormat.CycleType* data () (*pyffi.formats.nif.NifFormat.NiGeomMorpherController* attribute), 85 property), 121  
`CYLINDRICAL_SYMMETRY` data () (*pyffi.formats.nif.NifFormat.NiIntegersExtraData* (*pyffi.formats.nif.NifFormat.SymmetryType* attribute), 207 property), 126  
  
**D**  
`DaeFormat` (class in *pyffi.formats.dae*), 33  
`DaeFormat.Data` (class in *pyffi.formats.dae*), 33  
`damping()` (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint* data () (*pyffi.formats.nif.NifFormat.NiPoint3InterpController* property), 222 property), 168  
`damping()` (*pyffi.formats.nif.NifFormat.BSParentVelocityModifier* data () (*pyffi.formats.nif.NifFormat.NiPoint3Interpolator* property), 68 property), 169  
`DARK_MAP` (*pyffi.formats.nif.NifFormat.TextType* data () (*pyffi.formats.nif.NifFormat.NiPosData* attribute), 210 property), 169  
`dark_texture()` (*pyffi.formats.nif.NifFormat.NiTexturingProperty* data () (*pyffi.formats.nif.NifFormat.NiPSysColorModifier* property), 180 property), 148  
`Data` (*pyffi.formats.bsa.BsaFormat* attribute), 8 data () (*pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlr* property), 150  
`Data` (*pyffi.formats.egt.EgtFormat* attribute), 42 data () (*pyffi.formats.nif.NifFormat.NiPSysModifierActiveCtrlr* property), 153  
`Data` (*pyffi.formats.tri.TriFormat* attribute), 236 data () (*pyffi.formats.nif.NifFormat.NiPSysModifierFloatCtrlr* property), 154  
`data` (*pyffi.spells.Spell* attribute), 265 data () (*pyffi.formats.nif.NifFormat.NiRollController* property), 170  
`data()` (*pyffi.formats.nif.NifFormat.AdditionalDataBlock* data () (*pyffi.formats.nif.NifFormat.NiSkinInstance* property), 54 property), 174  
`data()` (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape* data () (*pyffi.formats.nif.NifFormat.NiSkinInstance* property), 216 property), 174

<code>data()</code> ( <code>pyffi.formats.nif.NifFormat.NiStringsExtraData</code> property), 178	<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellFFVT3RSkinPartition</code> method), 245
<code>data()</code> ( <code>pyffi.formats.nif.NifFormat.NiTextureTransformController</code> property), 180	<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots</code> method), 251
<code>data()</code> ( <code>pyffi.formats.nif.NifFormat.NiUVController</code> property), 185	<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellMergeSkeletonRoots</code> method), 248
<code>data()</code> ( <code>pyffi.formats.nif.NifFormat.NiVisController</code> property), 186	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify._SpellDelBranchClasses</code> method), 261
<code>data_2()</code> ( <code>pyffi.formats.nif.NifFormat.BSKeyframeController</code> property), 61	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellAddStencilProperty</code> method), 263
<code>data_2()</code> ( <code>pyffi.formats.nif.NifFormat.NiBezierMesh</code> property), 111	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellChangeBonePriorities</code> method), 259
<code>data_size()</code> ( <code>pyffi.formats.esp.EspFormat.SubRecord</code> property), 46	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellCleanFarNif</code> method), 264
<code>data_sizes()</code> ( <code>pyffi.formats.nif.NifFormat.AdditionalDataBlock</code> property), 54	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellCollisionMaterial</code> method), 257
<code>data_type()</code> ( <code>pyffi.formats.nif.NifFormat.AdditionalDataBlock</code> property), 54	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellCollisionType</code> method), 256
<code>dataentry()</code> ( <code>pyffi.spells.nif.fix.SpellDelUnusedRoots</code> method), 250	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellDelInterpolatorTransformData</code> method), 261
<code>dataentry()</code> ( <code>pyffi.spells.nif.fix.SpellDetachHavokTriStripsData</code> method), 246	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellDelVertexColor</code> method), 263
<code>dataentry()</code> ( <code>pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots</code> method), 251	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellDisableParallax</code> method), 262
<code>dataentry()</code> ( <code>pyffi.spells.nif.fix.SpellMergeSkeletonRoots</code> method), 248	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellReverseAnimation</code> method), 258
<code>dataentry()</code> ( <code>pyffi.spells.nif.fix.SpellScale</code> method), 248	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellScaleAnimationTime</code> method), 257
<code>dataentry()</code> ( <code>pyffi.spells.nif.optimize.SpellCleanRefLists</code> method), 252	<code>datainspect()</code> ( <code>pyffi.spells.nif.modify.SpellSetInterpolatorTransRotScale</code> method), 260
<code>dataentry()</code> ( <code>pyffi.spells.nif.optimize.SpellDelUnusedBones</code> method), 254	<code>datainspect()</code> ( <code>pyffi.spells.nif.optimize.SpellCleanRefLists</code> method), 252
<code>dataentry()</code> ( <code>pyffi.spells.Spell</code> method), 266	<code>datainspect()</code> ( <code>pyffi.spells.nif.optimize.SpellDelUnusedBones</code> method), 254
<code>dataentry()</code> ( <code>pyffi.spells.SpellGroupParallelBase</code> method), 268	<code>datainspect()</code> ( <code>pyffi.spells.nif.optimize.SpellMergeDuplicates</code> method), 253
<code>dataentry()</code> ( <code>pyffi.spells.SpellGroupSeriesBase</code> method), 268	<code>datainspect()</code> ( <code>pyffi.spells.nif.optimize.SpellOptimizeGeometry</code> method), 253
<code>dataexit()</code> ( <code>pyffi.spells.Spell</code> method), 266	<code>datainspect()</code> ( <code>pyffi.spells.Spell</code> method), 266
<code>dataexit()</code> ( <code>pyffi.spells.SpellGroupParallelBase</code> method), 268	<code>datainspect()</code> ( <code>pyffi.spells.SpellApplyPatch</code> method), 264
<code>dataexit()</code> ( <code>pyffi.spells.SpellGroupSeriesBase</code> method), 269	<code>datainspect()</code> ( <code>pyffi.spells.SpellGroupBase</code> method), 267
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellAddTangentSpace</code> method), 244	<code>datas()</code> ( <code>pyffi.formats.nif.NifFormat.NiMesh</code> property), 128
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellClampMaterialAlpha</code> method), 246	<code>DdsFormat</code> (class in <code>pyffi.formats.dds</code> ), 34
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellCleanStringPalette</code> method), 250	<code>DdsFormat.Data</code> (class in <code>pyffi.formats.dds</code> ), 34
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellDelTangentSpace</code> method), 244	<code>DdsFormat.FourCC</code> (class in <code>pyffi.formats.dds</code> ), 35
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellDelUnusedRoots</code> method), 250	<code>DdsFormat.HeaderString</code> (class in <code>pyffi.formats.dds</code> ), 35
<code>datainspect()</code> ( <code>pyffi.spells.nif.fix.SpellDetachHavokTriStripsData</code> method), 246	<code>DDSPATH</code> , 7
	<code>DEACTIVATOR_INVALID</code>
	<code>pyffi.formats.nif.NifFormat.DeactivatorType</code> (attribute), 86

DEACTIVATOR\_NEVER  
     (*pyffi.formats.nif.NifFormat.DeactivatorType attribute*), 86

DEACTIVATOR\_SPATIAL  
     (*pyffi.formats.nif.NifFormat.DeactivatorType attribute*), 87

DEADBIP (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201

DEBRIS\_LARGE (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 90

DEBRIS\_LARGE (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201

DEBRIS\_SMALL (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 90

DEBRIS\_SMALL (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201

DECAL\_0\_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 210

decal\_0\_texture ()  
     (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 180

DECAL\_1\_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 210

decal\_1\_texture ()  
     (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 180

DECAL\_2\_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 210

decal\_2\_texture ()  
     (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 181

DECAL\_3\_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 210

decal\_3\_texture ()  
     (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 181

decay () (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 157

decay () (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 147

decay () (*pyffi.formats.nif.NifFormat.NiPSysGravityModifier property*), 151

DECAY\_EXPONENTIAL  
     (*pyffi.formats.nif.NifFormat.DecayType attribute*), 87

DECAY\_LINEAR (*pyffi.formats.nif.NifFormat.DecayType attribute*), 87

DECAY\_NONE (*pyffi.formats.nif.NifFormat.DecayType attribute*), 87

decay\_type () (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 157

decay\_type () (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 147

declination () (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 149

declination\_variation ()  
     (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 149

Default (*pyffi.formats.nif.NifFormat.BSLightingShaderPropertyShaderType attribute*), 63

DEFAULT\_OPTIONS (*pyffi.spells.Toaster attribute*), 269

delta () (*pyffi.formats.nif.NifFormat.NiFlipController property*), 120

delta\_v () (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 157

delta\_v () (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 147

depth () (*pyffi.formats.nif.NifFormat.NiPSysBoxEmitter property*), 147

DETAIL\_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 210

detail\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 181

die\_on\_collide () (*pyffi.formats.nif.NifFormat.NiPSysCollider property*), 147

diffuse\_color () (*pyffi.formats.nif.NifFormat.NiLight property*), 127

dimmer () (*pyffi.formats.nif.NifFormat.NiLight property*), 127

direct\_render () (*pyffi.formats.nif.NifFormat.NiSourceTexture property*), 176

direction () (*pyffi.formats.nif.NifFormat.NiGravity property*), 125

direction () (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 157

direction () (*pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property*), 146

direction () (*pyffi.formats.nif.NifFormat.NiPSysDragFieldModifier property*), 149

direction () (*pyffi.formats.nif.NifFormat.NiPSysGravityFieldModifier property*), 151

direction () (*pyffi.formats.nif.NifFormat.NiPSysVortexFieldModifier property*), 156

distance () (*pyffi.formats.nif.NifFormat.BoneLOD property*), 77

distance\_weight ()  
     (*pyffi.formats.nif.NifFormat.BSProceduralLightningController property*), 69

DOORDETECTION (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 90

DOORDETECTION (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 201

drag\_axis () (*pyffi.formats.nif.NifFormat.NiPSysDragModifier property*), 149

DRAW\_BOTH (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 89

DRAW\_CCW (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 89

DRAW\_CCW\_OR\_BOTH (*pyffi.formats.nif.NifFormat.FaceDrawMode* attribute), 89

DRAW\_CW (*pyffi.formats.nif.NifFormat.FaceDrawMode* attribute), 89

draw\_mode () (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 177

DROPPING\_PICK (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

DROPPING\_PICK (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 90

DROPPING\_PICK (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201

duration () (*pyffi.formats.nif.NifFormat.NiParticleBomb* property), 157

**E**

EFFECT\_ENVIRONMENT\_MAP (*pyffi.formats.nif.NifFormat.EffectType* attribute), 87

EFFECT\_FOG\_MAP (*pyffi.formats.nif.NifFormat.EffectType* attribute), 87

EFFECT\_PROJECTED\_LIGHT (*pyffi.formats.nif.NifFormat.EffectType* attribute), 87

EFFECT\_PROJECTED\_SHADOW (*pyffi.formats.nif.NifFormat.EffectType* attribute), 87

EgmFormat (class in *pyffi.formats.egm*), 37

EgmFormat.Data (class in *pyffi.formats.egm*), 37

EgmFormat.FileSignature (class in *pyffi.formats.egm*), 38

EgmFormat.FileVersion (class in *pyffi.formats.egm*), 39

EgmFormat.MorphRecord (class in *pyffi.formats.egm*), 39

EgtFormat (class in *pyffi.formats.egt*), 42

EgtFormat.FileSignature (class in *pyffi.formats.egt*), 42

EgtFormat.FileVersion (class in *pyffi.formats.egt*), 42

EgtFormat.Header (class in *pyffi.formats.egt*), 42

elements () (*pyffi.formats.nif.NifFormat.NiMorphMeshModifier* property), 130

emission\_axis () (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter* property), 153

emission\_type () (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter* property), 153

emissive\_color () (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty* property), 59

emissive\_color () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62

emissive\_color () (*pyffi.formats.nif.NifFormat.BSShaderPPLightProperty* property), 72

emissive\_multiple () (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty* property), 59

emissive\_multiple () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62

emissiveMultiple (*pyffi.formats.nif.NifFormat.EffectShaderController* attribute), 87

emit\_flags () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

EMIT\_FROM\_EDGE\_CENTER (*pyffi.formats.nif.NifFormat.EmitFrom* attribute), 88

EMIT\_FROM\_EDGE\_SURFACE (*pyffi.formats.nif.NifFormat.EmitFrom* attribute), 88

EMIT\_FROM\_FACE\_CENTER (*pyffi.formats.nif.NifFormat.EmitFrom* attribute), 88

EMIT\_FROM\_FACE\_SURFACE (*pyffi.formats.nif.NifFormat.EmitFrom* attribute), 88

EMIT\_FROM\_VERTICES (*pyffi.formats.nif.NifFormat.EmitFrom* attribute), 88

emit\_rate () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

emit\_start\_time () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

emit\_stop\_time () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

emitter () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

emitter () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 141

emitter\_meshes () (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter* property), 153

emitter\_object () (*pyffi.formats.nif.NifFormat.NiPSysVolumeEmitter* property), 156

end () (*pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey2* property), 88

end\_frame () (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* property), 67

ENDIAN\_BIG (*pyffi.formats.nif.NifFormat.EndianType* attribute), 88

ENDIAN\_LITTLE (*pyffi.formats.nif.NifFormat.EndianType* attribute), 88

entities () (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207

entity\_a () (*pyffi.formats.nif.NifFormat.bhkRDTConstraint* property), 222

entity\_a () (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint* property), 223

entity\_b() (pyffi.formats.nif.NifFormat.bhkRDTConstraint.F\_FLOAT16\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 222  
 entity\_b() (pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint.F\_FLOAT16\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 223  
 entry\_properties() (pyffi.formats.nif.NifFormat.FurniturePosition.F\_FLOAT16\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 93  
 environment variable  
   CGFXMLPATH, 7  
   DDFXMLPATH, 7  
   KFMFXMLPATH, 7  
   NIFFXMLPATH, 7  
   TGAXMLPATH, 7  
 environment\_map\_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty.F\_FLOAT32\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 62  
 environment\_map\_scale() (pyffi.formats.nif.NifFormat.BSShaderProperty.F\_FLOAT32\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 73  
 EPSILON (pyffi.formats.nif.NifFormat attribute), 87  
 error() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData.F\_FLOAT32\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 217  
 EspFormat (class in pyffi.formats.esp), 45  
 EspFormat.Data (class in pyffi.formats.esp), 45  
 EspFormat.GRUP (class in pyffi.formats.esp), 45  
 EspFormat.Record (class in pyffi.formats.esp), 46  
 EspFormat.RecordType (class in pyffi.formats.esp), 46  
 EspFormat.SubRecord (class in pyffi.formats.esp), 46  
 EspFormat.ZString (class in pyffi.formats.esp), 46  
 EXAMPLES (pyffi.spells.Toaster attribute), 269  
 exclude\_types (pyffi.spells.Toaster attribute), 269  
 exponent() (pyffi.formats.nif.NifFormat.NiSpotLight.F\_FLOAT32\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 177  
 export\_info\_1() (pyffi.formats.nif.NifFormat.ExportInfo.F\_INT16\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 88  
 export\_info\_2() (pyffi.formats.nif.NifFormat.ExportInfo.F\_INT16\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 88  
 extent() (pyffi.formats.nif.NifFormat.BoxBV property), 78  
 extent() (pyffi.formats.nif.NifFormat.BSMultiBoundAABB.F\_INT16\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 64  
 extra\_flags() (pyffi.formats.nif.NifFormat.NiGeomMorpherContainer.F\_INT16\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 122  
 extra\_targets() (pyffi.formats.nif.NifFormat.NiMultiTargetTransformer.F\_INT32\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 131  
 eye\_cubemap\_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty.F\_INT32\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 62  
**F**  
 F\_FLOAT16\_1 (pyffi.formats.nif.NifFormat.ComponentFormat.F\_INT32\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT16\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT16\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT16\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT32\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT32\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT32\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_FLOAT32\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_INT16\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_INT16\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 81  
 F\_INT16\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT16\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT32\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT32\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT32\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT32\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT8\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT8\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT8\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_INT8\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT16\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT16\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT16\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT16\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT32\_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT32\_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT32\_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82  
 F\_NORMINT32\_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 82

F_NORMINT8_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT32_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT8_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT32_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT8_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT32_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT8_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT8_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT_10_10_10_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT8_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT_10_10_10_L1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT8_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMINT_11_11_10 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT8_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMUINT16_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT_10_10_10_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMUINT16_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UINT_10_10_10_L1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMUINT16_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	F_UNKNOWN ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83
F_NORMUINT16_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade () ( <i>pyffi.formats.nif.NifFormat.NiParticleGrowFade property</i> ), 157
F_NORMUINT32_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_generation () ( <i>pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property</i> ), 152
F_NORMUINT32_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_in_percent () ( <i>pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property</i> ), 67
F_NORMUINT32_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_out_percent () ( <i>pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property</i> ), 67
F_NORMUINT32_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_time () ( <i>pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property</i> ), 152
F_NORMUINT8_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_action () ( <i>pyffi.formats.nif.NifFormat.NiStencilProperty property</i> ), 177
F_NORMUINT8_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	fade_offset_start_angle () ( <i>pyffi.formats.nif.NifFormat.BSEffectShaderProperty property</i> ), 59
F_NORMUINT8_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 82	falloff_start_angle () ( <i>pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property</i> ), 72
F_NORMUINT8_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_start_opacity () ( <i>pyffi.formats.nif.NifFormat.BSEffectShaderProperty property</i> ), 59
F_NORMUINT8_4_BGRA ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_stop_angle () ( <i>pyffi.formats.nif.NifFormat.BSEffectShaderProperty property</i> ), 59
F_UINT16_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_stop_angle () ( <i>pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property</i> ), 72
F_UINT16_2 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_start_opacity () ( <i>pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property</i> ), 72
F_UINT16_3 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_stop_angle () ( <i>pyffi.formats.nif.NifFormat.BSEffectShaderProperty property</i> ), 59
F_UINT16_4 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	falloff_stop_angle () ( <i>pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property</i> ), 72
F_UINT32_1 ( <i>pyffi.formats.nif.NifFormat.ComponentFormat attribute</i> ), 83	

*property*), 72  
 falloff\_stop\_opacity() (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty* *property*), 59  
 falloff\_stop\_opacity() (*pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty* *property*), 72  
 far\_extent() (*pyffi.formats.nif.NifFormat.LODRangeProperty*), 97  
 field\_object() (*pyffi.formats.nif.NifFormat.NiPSysFieldModifier* *property*), 151  
 FIELD\_POINT (*pyffi.formats.nif.NifFormat.FieldType* *attribute*), 92  
 FIELD\_WIND (*pyffi.formats.nif.NifFormat.FieldType* *attribute*), 92  
 file, 304  
 file format, 304  
 file format interface, 304  
 file\_name() (*pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty* *property*), 72  
 file\_name() (*pyffi.formats.nif.NifFormat.NiImageProperty*), 125  
 file\_name() (*pyffi.formats.nif.NifFormat.NiSourceTextureProperty*), 176  
 file\_name() (*pyffi.formats.nif.NifFormat.SkyShaderProperty* *property*), 198  
 file\_name() (*pyffi.formats.nif.NifFormat.TallGrassShaderProperty* *property*), 208  
 file\_name() (*pyffi.formats.nif.NifFormat.TexSourceProperty*), 209  
 file\_name() (*pyffi.formats.nif.NifFormat.TileShaderProperty* *property*), 210  
 FileFormat (*class in pyffi.object\_models*), 271  
 FILEFORMAT (*pyffi.spells.Toaster* *attribute*), 269  
 FileFormat.Data (*class in pyffi.object\_models*), 272  
 filter() (*pyffi.formats.nif.NifFormat.MultiTextureElement* *property*), 104  
 FILTER\_BILERP (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 FILTER\_BILERP\_MIPNEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 filter\_mode() (*pyffi.formats.nif.NifFormat.TexDescProperty*), 208  
 FILTER\_NEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 FILTER\_NEAREST\_MIPLERP (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 FILTER\_NEAREST\_MIPNEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 FILTER\_TRILERP (*pyffi.formats.nif.NifFormat.TexFilterMode* *attribute*), 209  
 find() (*pyffi.formats.nif.NifFormat.NiObject* *method*), 134  
 find\_chain() (*pyffi.formats.nif.NifFormat.NiObject* *method*), 134  
 fix\_links() (*pyffi.formats.cgf.CgfFormat.Ref* *property*), 27  
 fix\_links() (*pyffi.formats.nif.NifFormat.Ref* *method*), 195  
 flag\_or\_num\_constraints() (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* *property*), 223  
 flags() (*pyffi.formats.nif.NifFormat.bhkNiCollisionObject* *property*), 220  
 flags() (*pyffi.formats.nif.NifFormat.BSSegment* *property*), 70  
 flags() (*pyffi.formats.nif.NifFormat.NiAlphaProperty* *property*), 107  
 flags() (*pyffi.formats.nif.NifFormat.NiDitherProperty* *property*), 118  
 flags() (*pyffi.formats.nif.NifFormat.NiFogProperty* *property*), 121  
 flags() (*pyffi.formats.nif.NifFormat.NiMorphMeshModifier* *property*), 130  
 flags() (*pyffi.formats.nif.NifFormat.NiMultiTextureProperty* *property*), 131  
 flags() (*pyffi.formats.nif.NifFormat.NiShadeProperty* *property*), 173  
 flags() (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier* *property*), 175  
 flags() (*pyffi.formats.nif.NifFormat.NiSpecularProperty* *property*), 176  
 flags() (*pyffi.formats.nif.NifFormat.NiStencilProperty* *property*), 177  
 flags() (*pyffi.formats.nif.NifFormat.NiTextureProperty* *property*), 180  
 flags() (*pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 181  
 flags() (*pyffi.formats.nif.NifFormat.NiTimeController* *property*), 182  
 flags() (*pyffi.formats.nif.NifFormat.NiVertexColorProperty* *property*), 186  
 flags() (*pyffi.formats.nif.NifFormat.NiWireframeProperty* *property*), 187  
 flags() (*pyffi.formats.nif.NifFormat.NiZBufferProperty* *property*), 187  
 flags() (*pyffi.formats.nif.NifFormat.TexDesc* *property*), 208  
 flags\_and\_part\_number() (*pyffi.formats.nif.NifFormat.HavokColFilter* *property*), 94  
 flatten\_skin() (*pyffi.formats.nif.NifFormat.NiGeometry* *method*), 123  
 float (*pyffi.formats.cgf.CgfFormat* *attribute*), 29  
 float (*pyffi.formats.dds.DdsFormat* *attribute*), 36

- float (*pyffi.formats.egm.EgmFormat* attribute), 40
- float (*pyffi.formats.egt.EgtFormat* attribute), 43
- float (*pyffi.formats.esp.EspFormat* attribute), 47
- float (*pyffi.formats.kfm.KfmFormat* attribute), 51
- float (*pyffi.formats.nif.NifFormat* attribute), 226
- float (*pyffi.formats.tga.TgaFormat* attribute), 235
- float (*pyffi.formats.tri.TriFormat* attribute), 239
- float\_2 () (*pyffi.formats.nif.NifFormat.BSPProceduralLightningController* property), 69
- float\_5 () (*pyffi.formats.nif.NifFormat.BSPProceduralLightningController* property), 69
- float\_data () (*pyffi.formats.nif.NifFormat.NiFloatExtraData* property), 120
- float\_data () (*pyffi.formats.nif.NifFormat.NiPathController* property), 162
- float\_data () (*pyffi.formats.nif.NifFormat.NiPathInterpolator* property), 162
- float\_keys () (*pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlData* property), 150
- float\_value () (*pyffi.formats.nif.NifFormat.NiBlendFloatInterpolator* property), 114
- float\_value () (*pyffi.formats.nif.NifFormat.NiFloatInterpolator* property), 121
- floats () (*pyffi.formats.nif.NifFormat.BSPSysScaleModifier* property), 66
- floats\_1 () (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain* property), 213
- fog\_color () (*pyffi.formats.nif.NifFormat.NiFogProperty* property), 121
- fog\_depth () (*pyffi.formats.nif.NifFormat.NiFogProperty* property), 121
- force () (*pyffi.formats.nif.NifFormat.NiGravity* property), 125
- FORCE\_PLANAR (*pyffi.formats.nif.NifFormat.ForceType* attribute), 93
- FORCE\_SPHERICAL (*pyffi.formats.nif.NifFormat.ForceType* attribute), 93
- force\_type () (*pyffi.formats.nif.NifFormat.NiPSysGravityModifier* property), 151
- FORCE\_UNKNOWN (*pyffi.formats.nif.NifFormat.ForceType* attribute), 93
- forces () (*pyffi.formats.nif.NifFormat.NiPSSimulatorForcesStep* property), 143
- fork () (*pyffi.formats.nif.NifFormat.BSPProceduralLightningController* property), 69
- forward () (*pyffi.formats.nif.NifFormat.Key* property), 97
- frame\_count () (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* property), 67
- frame\_count\_fudge () (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* property), 67
- frame\_name () (*pyffi.formats.nif.NifFormat.Morph* property), 102
- frequency () (*pyffi.formats.nif.NifFormat.NiPSysTurbulenceFieldModifier* property), 156
- frequency () (*pyffi.formats.nif.NifFormat.NiTimeController* property), 182
- friction () (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* property), 223
- friction () (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192
- front () (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints* property), 93
- frustum\_bottom () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- frustum\_far () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- frustum\_left () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- frustum\_near () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- frustum\_right () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- frustum\_top () (*pyffi.formats.nif.NifFormat.NiCamera* property), 115
- function () (*pyffi.formats.nif.NifFormat.NiZBufferProperty* property), 187
- g () (*pyffi.formats.nif.NifFormat.ByteColor3* property), 79
- g () (*pyffi.formats.nif.NifFormat.ByteColor4* property), 79
- g () (*pyffi.formats.nif.NifFormat.Color3* property), 81
- g () (*pyffi.formats.nif.NifFormat.Color4* property), 81
- games (*pyffi.formats.nif.NifFormat* attribute), 226
- GASTRAP (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- GASTRAP (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201
- generator () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 141
- get\_all\_strings () (*pyffi.formats.nif.NifFormat.StringPalette* method), 206
- get\_children () (*pyffi.formats.nif.NifFormat.NiNode* method), 133
- get\_chunk\_types () (*pyffi.formats.cgf.CgfFormat.ChunkTable* method), 12
- get\_colors () (*pyffi.formats.cgf.CgfFormat.MeshChunk* method), 17
- get\_comp\_data () (*pyffi.formats.nif.NifFormat.NiBSplineData* method), 110
- get\_controller\_type () (*pyffi.formats.nif.NifFormat.ControllerLink* method), 84

`get_controllers()` (*pyffi.formats.nif.NifFormat.NiObjectNET method*), 135  
`get_copy()` (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 15  
`get_copy()` (*pyffi.formats.cgf.CgfFormat.Matrix44 method*), 16  
`get_copy()` (*pyffi.formats.nif.NifFormat.InertiaMatrix method*), 96  
`get_copy()` (*pyffi.formats.nif.NifFormat.Matrix33 method*), 99  
`get_copy()` (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
`get_copy()` (*pyffi.formats.nif.NifFormat.Vector3 method*), 211  
`get_copy()` (*pyffi.formats.nif.NifFormat.Vector4 method*), 212  
`get_detail_child_names()` (*pyffi.formats.cgf.CgfFormat.Data method*), 13  
`get_detail_child_names()` (*pyffi.formats.dds.DdsFormat.Data method*), 34  
`get_detail_child_names()` (*pyffi.formats.egm.EgmFormat.Data method*), 38  
`get_detail_child_names()` (*pyffi.formats.esp.EspFormat.Data method*), 45  
`get_detail_child_names()` (*pyffi.formats.nif.NifFormat.Data method*), 85  
`get_detail_child_names()` (*pyffi.formats.tga.TgaFormat.Image method*), 234  
`get_detail_child_nodes()` (*pyffi.formats.cgf.CgfFormat.Data method*), 13  
`get_detail_child_nodes()` (*pyffi.formats.dds.DdsFormat.Data method*), 35  
`get_detail_child_nodes()` (*pyffi.formats.egm.EgmFormat.Data method*), 38  
`get_detail_child_nodes()` (*pyffi.formats.esp.EspFormat.Data method*), 45  
`get_detail_child_nodes()` (*pyffi.formats.nif.NifFormat.Data method*), 85  
`get_detail_child_nodes()` (*pyffi.formats.tga.TgaFormat.Image method*), 234  
`get_detail_display()` (*pyffi.formats.bsa.BsaFormat.Hash method*), 8  
`get_detail_display()` (*pyffi.formats.dds.DdsFormat.HeaderString method*), 35  
`get_detail_display()` (*pyffi.formats.egm.EgmFormat.FileSignature method*), 38  
`get_detail_display()` (*pyffi.formats.egm.EgmFormat.FileVersion method*), 39  
`get_detail_display()` (*pyffi.formats.egt.EgtFormat.FileSignature method*), 42  
`get_detail_display()` (*pyffi.formats.egt.EgtFormat.FileVersion method*), 42  
`get_detail_display()` (*pyffi.formats.kfm.KfmFormat.HeaderString method*), 49  
`get_detail_display()` (*pyffi.formats.nif.NifFormat.Data.VersionUInt method*), 85  
`get_detail_display()` (*pyffi.formats.nif.NifFormat.FileVersion method*), 92  
`get_detail_display()` (*pyffi.formats.nif.NifFormat.HeaderString method*), 95  
`get_detail_display()` (*pyffi.formats.nif.NifFormat.Ref method*), 195  
`get_detail_display()` (*pyffi.formats.tri.TriFormat.FileSignature method*), 236  
`get_detail_display()` (*pyffi.formats.tri.TriFormat.FileVersion method*), 237  
`get_determinant()` (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 15  
`get_determinant()` (*pyffi.formats.nif.NifFormat.Matrix33 method*), 100  
`get_dismember_partitions()` (*pyffi.formats.nif.NifFormat.BSDismemberSkinInstance method*), 59  
`get_effects()` (*pyffi.formats.nif.NifFormat.NiNode method*), 133  
`get_extra_datas()` (*pyffi.formats.nif.NifFormat.NiObjectNET method*), 135  
`get_float_data()` (*pyffi.formats.nif.NifFormat.NiBSplineData method*), 110  
`get_global_child_nodes()` (*pyffi.formats.cgf.CgfFormat.Data method*), 13  
`get_global_child_nodes()` (*pyffi.formats.egm.EgmFormat.Data method*), 38

`get_global_child_nodes()`  
 (*pyffi.formats.egt.EgtFormat.Header* method), 43  
`get_global_child_nodes()`  
 (*pyffi.formats.esp.EspFormat.Data* method), 45  
`get_global_child_nodes()`  
 (*pyffi.formats.esp.EspFormat.GRUP* method), 45  
`get_global_child_nodes()`  
 (*pyffi.formats.esp.EspFormat.Record* method), 46  
`get_global_child_nodes()`  
 (*pyffi.formats.kfm.KfmFormat.Data* method), 49  
`get_global_child_nodes()`  
 (*pyffi.formats.nif.NifFormat.Data* method), 85  
`get_global_child_nodes()`  
 (*pyffi.formats.tga.TgaFormat.Data* method), 234  
`get_global_child_nodes()`  
 (*pyffi.formats.tri.TriFormat.Header* method), 237  
`get_global_display()`  
 (*pyffi.formats.kfm.KfmFormat.Data* method), 49  
`get_hash()` (*pyffi.formats.bsa.BsaFormat.ZString* method), 9  
`get_hash()` (*pyffi.formats.cgf.CgfFormat.FileSignature* method), 14  
`get_hash()` (*pyffi.formats.cgf.CgfFormat.Ref* method), 27  
`get_hash()` (*pyffi.formats.cgf.CgfFormat.SizedString* method), 28  
`get_hash()` (*pyffi.formats.dds.DdsFormat.HeaderString* method), 35  
`get_hash()` (*pyffi.formats.egm.EgmFormat.FileSignature* method), 38  
`get_hash()` (*pyffi.formats.egm.EgmFormat.FileVersion* method), 39  
`get_hash()` (*pyffi.formats.egt.EgtFormat.FileSignature* method), 42  
`get_hash()` (*pyffi.formats.egt.EgtFormat.FileVersion* method), 42  
`get_hash()` (*pyffi.formats.esp.EspFormat.ZString* method), 46  
`get_hash()` (*pyffi.formats.kfm.KfmFormat.FilePath* method), 49  
`get_hash()` (*pyffi.formats.kfm.KfmFormat.HeaderString* method), 49  
`get_hash()` (*pyffi.formats.kfm.KfmFormat.SizedString* method), 50  
`get_hash()` (*pyffi.formats.nif.NifFormat.bool* method), 226  
`get_hash()` (*pyffi.formats.nif.NifFormat.ByteArray* method), 78  
`get_hash()` (*pyffi.formats.nif.NifFormat.ByteMatrix* method), 79  
`get_hash()` (*pyffi.formats.nif.NifFormat.FilePath* method), 92  
`get_hash()` (*pyffi.formats.nif.NifFormat.HeaderString* method), 95  
`get_hash()` (*pyffi.formats.nif.NifFormat.LineString* method), 98  
`get_hash()` (*pyffi.formats.nif.NifFormat.Ptr* method), 193  
`get_hash()` (*pyffi.formats.nif.NifFormat.Ref* method), 195  
`get_hash()` (*pyffi.formats.nif.NifFormat.ShortString* method), 196  
`get_hash()` (*pyffi.formats.nif.NifFormat.SizedString* method), 197  
`get_hash()` (*pyffi.formats.nif.NifFormat.string* method), 227  
`get_hash()` (*pyffi.formats.tga.TgaFormat.FooterString* method), 234  
`get_hash()` (*pyffi.formats.tri.TriFormat.FileSignature* method), 236  
`get_hash()` (*pyffi.formats.tri.TriFormat.FileVersion* method), 237  
`get_interchangeable_packed_shape()`  
 (*pyffi.formats.nif.NifFormat.bhkNiTriStripsShape* method), 220  
`get_interchangeable_tri_shape()`  
 (*pyffi.formats.nif.NifFormat.NiTriBasedGeom* method), 183  
`get_interchangeable_tri_strips()`  
 (*pyffi.formats.nif.NifFormat.NiTriBasedGeom* method), 183  
`get_inverse()` (*pyffi.formats.cgf.CgfFormat.Matrix33* method), 15  
`get_inverse()` (*pyffi.formats.cgf.CgfFormat.Matrix44* method), 16  
`get_inverse()` (*pyffi.formats.nif.NifFormat.Matrix33* method), 100  
`get_inverse()` (*pyffi.formats.nif.NifFormat.Matrix44* method), 100  
`get_links()` (*pyffi.formats.cgf.CgfFormat.Ref* method), 27  
`get_links()` (*pyffi.formats.nif.NifFormat.Ref* method), 195  
`get_mapped_triangles()`  
 (*pyffi.formats.nif.NifFormat.SkinPartition* method), 197  
`get_mass_center_inertia()`  
 (*pyffi.formats.nif.NifFormat.bhkBoxShape* method), 214  
`get_mass_center_inertia()`

(*pyffi.formats.nif.NifFormat.bhkCapsuleShape method*), 215  
 (*pyffi.formats.nif.NifFormat.bhkConvexVerticesShape method*), 218  
 (*pyffi.formats.nif.NifFormat.bhkListShape method*), 219  
 (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method*), 220  
 (*pyffi.formats.nif.NifFormat.bhkMultiSphereShape method*), 220  
 (*pyffi.formats.nif.NifFormat.bhkNiTriStripsShape method*), 220  
 (*pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape method*), 221  
 (*pyffi.formats.nif.NifFormat.bhkSphereShape method*), 225  
 (*pyffi.formats.nif.NifFormat.bhkTransformShape method*), 225  
 (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17  
 (*pyffi.formats.cgf.CgfFormat.Matrix44 method*), 16  
 (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
 (*pyffi.formats.nif.NifFormat.ControllerLink method*), 84  
 (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17  
 (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17  
 (*pyffi.formats.nif.NifFormat.NiAVObject method*), 106  
 (*pyffi.formats.nif.NifFormat.ControllerLink method*), 84  
 (*pyffi.formats.cgf.CgfFormat.Ptr method*), 26  
 (*pyffi.formats.cgf.CgfFormat.Ref method*), 27  
 (*pyffi.formats.nif.NifFormat.Ptr method*), 193  
 (*pyffi.formats.nif.NifFormat.Ref method*), 195  
 (*pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator method*), 109  
 (*pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator method*), 111  
 (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 15  
 (*pyffi.formats.nif.NifFormat.Matrix33 method*), 100  
 (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 16  
 (*pyffi.formats.nif.NifFormat.Matrix33 method*), 100  
 (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
 (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 16  
 (*pyffi.formats.nif.NifFormat.Matrix33 method*), 16  
 (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
 (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 16  
 (*pyffi.formats.nif.NifFormat.Matrix33 method*), 100  
 (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
 (*pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator method*), 109  
 (*pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator method*), 111  
 (*pyffi.formats.nif.NifFormat.bhkRefObject method*), 224  
 (*pyffi.formats.nif.NifFormat.NiBSplineData method*), 111  
 (*pyffi.formats.bsa.BsaFormat.BZString method*), 8  
 (*pyffi.formats.bsa.BsaFormat.FileVersion method*), 8  
 (*pyffi.formats.bsa.BsaFormat.ZString method*), 9  
 (*pyffi.formats.cgf.CgfFormat.FileSignature method*), 14  
 (*pyffi.formats.cgf.CgfFormat.Ref method*), 27  
 (*pyffi.formats.cgf.CgfFormat.SizedString method*), 28  
 (*pyffi.formats.dds.DdsFormat.HeaderString method*), 35  
 (*pyffi.formats.egm.EgmFormat.FileSignature method*), 39  
 (*pyffi.formats.egm.EgmFormat.FileVersion method*), 39  
 (*pyffi.formats.egt.EgtFormat.FileSignature method*), 42  
 (*pyffi.formats.egt.EgtFormat.FileVersion method*), 42

`get_size()` (*pyffi.formats.esp.EspFormat.ZString method*), 47  
`get_size()` (*pyffi.formats.kfm.KfmFormat.HeaderString method*), 49  
`get_size()` (*pyffi.formats.kfm.KfmFormat.SizedString method*), 50  
`get_size()` (*pyffi.formats.nif.NifFormat.bool method*), 226  
`get_size()` (*pyffi.formats.nif.NifFormat.ByteArray method*), 78  
`get_size()` (*pyffi.formats.nif.NifFormat.ByteMatrix method*), 79  
`get_size()` (*pyffi.formats.nif.NifFormat.HeaderString method*), 95  
`get_size()` (*pyffi.formats.nif.NifFormat.LineString method*), 98  
`get_size()` (*pyffi.formats.nif.NifFormat.Ref method*), 195  
`get_size()` (*pyffi.formats.nif.NifFormat.ShortString method*), 196  
`get_size()` (*pyffi.formats.nif.NifFormat.SizedString method*), 197  
`get_size()` (*pyffi.formats.nif.NifFormat.string method*), 227  
`get_size()` (*pyffi.formats.tga.TgaFormat.FooterString method*), 234  
`get_size()` (*pyffi.formats.tri.TriFormat.FileSignature method*), 237  
`get_size()` (*pyffi.formats.tri.TriFormat.FileVersion method*), 237  
`get_size()` (*pyffi.formats.tri.TriFormat.SizedStringZ method*), 239  
`get_skin_deformation()` (*pyffi.formats.nif.NifFormat.NiGeometry method*), 123  
`get_skin_partition()` (*pyffi.formats.nif.NifFormat.NiGeometry method*), 123  
`get_skinned_geometries()` (*pyffi.formats.nif.NifFormat.NiNode method*), 133  
`get_string()` (*pyffi.formats.nif.NifFormat.StringPalette method*), 206  
`get_strings()` (*pyffi.formats.nif.NifFormat.string method*), 227  
`get_strips()` (*pyffi.formats.nif.NifFormat.NiTriShapeData method*), 184  
`get_strips()` (*pyffi.formats.nif.NifFormat.NiTriStripsData method*), 185  
`get_sub_record()` (*pyffi.formats.esp.EspFormat.Record method*), 46  
`get_sub_shapes()` (*pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsData method*), 221  
`get_tangent_space()` (*pyffi.formats.nif.NifFormat.NiTriBasedGeom method*), 183  
`get_target_color()` (*pyffi.formats.nif.NifFormat.NiMaterialColorController method*), 128  
`get_times()` (*pyffi.formats.nif.NifFormat.NiBSplineInterpolator method*), 111  
`get_toast_head_root_ext()` (*pyffi.spells.Toaster method*), 269  
`get_toast_stream()` (*pyffi.spells.Toaster method*), 269  
`get_transform()` (*pyffi.formats.nif.NifFormat.NiAVObject method*), 106  
`get_transform()` (*pyffi.formats.nif.NifFormat.NiSkinData method*), 174  
`get_transform()` (*pyffi.formats.nif.NifFormat.SkinData method*), 197  
`get_transform()` (*pyffi.formats.nif.NifFormat.SkinTransform method*), 198  
`get_transform_a_b()` (*pyffi.formats.nif.NifFormat.bhkConstraint method*), 217  
`get_translation()` (*pyffi.formats.cgf.CgfFormat.Matrix44 method*), 16  
`get_translation()` (*pyffi.formats.nif.NifFormat.Matrix44 method*), 100  
`get_translations()` (*pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator method*), 109  
`get_translations()` (*pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator method*), 111  
`get_transpose()` (*pyffi.formats.cgf.CgfFormat.Matrix33 method*), 16  
`get_transpose()` (*pyffi.formats.nif.NifFormat.Matrix33 method*), 100  
`get_triangle_hash_generator()` (*pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape method*), 221  
`get_triangle_indices()` (*pyffi.formats.nif.NifFormat.NiTriBasedGeomData method*), 184  
`get_triangles()` (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17  
`get_triangles()` (*pyffi.formats.nif.NifFormat.NiTriShapeData method*), 184  
`get_triangles()` (*pyffi.formats.nif.NifFormat.NiTriStripsData method*), 185  
`get_triangles()` (*pyffi.formats.nif.NifFormat.SkinPartition method*), 197  
`get_uv_triangles()` (*pyffi.formats.cgf.CgfFormat.MeshChunk method*), 17

method), 17  
 get\_uvcs () (pyffi.formats.cgf.CgfFormat.MeshChunk  
 method), 17  
 get\_value () (pyffi.formats.bsa.BsaFormat.ZString  
 method), 9  
 get\_value () (pyffi.formats.cgf.CgfFormat.FileSignature  
 method), 14  
 get\_value () (pyffi.formats.cgf.CgfFormat.Ref  
 method), 27  
 get\_value () (pyffi.formats.cgf.CgfFormat.SizedString  
 method), 28  
 get\_value () (pyffi.formats.egm.EgmFormat.FileVersion  
 method), 39  
 get\_value () (pyffi.formats.egt.EgtFormat.FileVersion  
 method), 42  
 get\_value () (pyffi.formats.esp.EspFormat.ZString  
 method), 47  
 get\_value () (pyffi.formats.kfm.KfmFormat.HeaderString  
 method), 49  
 get\_value () (pyffi.formats.kfm.KfmFormat.SizedString  
 method), 50  
 get\_value () (pyffi.formats.nif.NifFormat.bool  
 method), 226  
 get\_value () (pyffi.formats.nif.NifFormat.ByteArray  
 method), 78  
 get\_value () (pyffi.formats.nif.NifFormat.ByteMatrix  
 method), 79  
 get\_value () (pyffi.formats.nif.NifFormat.LineString  
 method), 98  
 get\_value () (pyffi.formats.nif.NifFormat.Ptr  
 method), 193  
 get\_value () (pyffi.formats.nif.NifFormat.Ref  
 method), 195  
 get\_value () (pyffi.formats.nif.NifFormat.ShortString  
 method), 196  
 get\_value () (pyffi.formats.nif.NifFormat.SizedString  
 method), 197  
 get\_value () (pyffi.formats.tga.TgaFormat.FooterString  
 method), 234  
 get\_value () (pyffi.formats.tri.TriFormat.FileVersion  
 method), 237  
 get\_variable\_1 () (pyffi.formats.nif.NifFormat.ControllerLink  
 method), 84  
 get\_variable\_2 () (pyffi.formats.nif.NifFormat.ControllerLink  
 method), 84  
 get\_vector\_3 () (pyffi.formats.nif.NifFormat.Vector4  
 method), 212  
 get\_vertex\_hash\_generator ()  
 (pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape  
 method), 221  
 get\_vertex\_hash\_generator ()  
 (pyffi.formats.nif.NifFormat.NiGeometryData  
 method), 124  
 get\_vertex\_weights ()  
 (pyffi.formats.nif.NifFormat.NiGeometry  
 method), 123  
 get\_vertices () (pyffi.formats.cgf.CgfFormat.MeshChunk  
 method), 17  
 getVersion () (pyffi.formats.dae.DaeFormat.Data  
 method), 33  
 GLOSS\_MAP (pyffi.formats.nif.NifFormat.TextType  
 attribute), 210  
 gloss\_texture () (pyffi.formats.nif.NifFormat.NiTexturingProperty  
 property), 181  
 Glossiness (pyffi.formats.nif.NifFormat.LightingShaderControlledVariation  
 attribute), 98  
 glossiness () (pyffi.formats.nif.NifFormat.BSLightingShaderProperty  
 property), 62  
 GLOW\_MAP (pyffi.formats.nif.NifFormat.TextType at-  
 tribute), 210  
 glow\_texture () (pyffi.formats.nif.NifFormat.NiTexturingProperty  
 property), 181  
 gpu\_read () (pyffi.formats.nif.NifFormat.DataStreamAccess  
 property), 86  
 gpu\_write () (pyffi.formats.nif.NifFormat.DataStreamAccess  
 property), 86  
 gravity\_axis () (pyffi.formats.nif.NifFormat.NiPSysGravityModifier  
 property), 152  
 gravity\_object () (pyffi.formats.nif.NifFormat.NiPSysGravityModifier  
 property), 152  
 greyscale\_texture ()  
 (pyffi.formats.nif.NifFormat.BSEffectShaderProperty  
 property), 59  
 GROUND (pyffi.formats.nif.NifFormat.Fallout3Layer at-  
 tribute), 91  
 GROUND (pyffi.formats.nif.NifFormat.OblivionLayer at-  
 tribute), 189  
 GROUND (pyffi.formats.nif.NifFormat.SkyrimLayer  
 attribute), 201  
 grow () (pyffi.formats.nif.NifFormat.NiParticleGrowFade  
 property), 157  
 grow\_generation ()  
 (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep  
 property), 144  
 grow\_generation ()  
 (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier  
 property), 152  
 grow\_time () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep  
 property), 144  
 grow\_time () (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier  
 property), 152  
 hair\_tint\_color ()  
 (pyffi.formats.nif.NifFormat.BSLightingShaderProperty  
 property), 62  
 half\_space () (pyffi.formats.nif.NifFormat.BoundingBox  
 property), 78

HALFSPACE\_BV (*pyffi.formats.nif.NifFormat.BoundVolumeType* property), 161  
 attribute), 77

has\_base\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_block\_type () (*pyffi.formats.nif.NifFormat.Header* property), 161  
 method), 95

has\_bump\_map\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_dark\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_data () (*pyffi.formats.nif.NifFormat.AdditionalDataBlock* property), 54

has\_data () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* property), 67

has\_data () (*pyffi.formats.nif.NifFormat.Ni3dsAnimationNode* property), 104

has\_decals\_0\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_decals\_1\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_decals\_2\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_decals\_3\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_detail\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_gloss\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_glow\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_image () (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104

has\_normal\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_radii () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

has\_rotation\_angles () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

has\_rotation\_axes () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

has\_rotations () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

has\_rotations\_2 () (*pyffi.formats.nif.NifFormat.NiRotatingParticlesData* property), 171

has\_sizes () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

has\_subtexture\_offset\_u\_vs () (*pyffi.formats.nif.NifFormat.NiPSysData* property), 148

has\_texture\_transform () (*pyffi.formats.nif.NifFormat.TexDesc* property), 209

has\_unknown\_2\_texture () (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181

has\_unknown\_3\_texture () (*pyffi.formats.nif.NifFormat.NiPSysData* property), 148

has\_uv\_quadrants () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

HAV\_MAT\_CHAIN (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_CHAIN\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_CLOTH (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_CLOTH\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_DIRT (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_DIRT\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_ELEVATOR (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_GLASS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_GLASS\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_GRASS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_GRASS\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 187

HAV\_MAT\_HEAVY\_METAL (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188

HAV\_MAT\_HEAVY\_METAL\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188

HAV\_MAT\_HEAVY\_STONE (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_HEAVY\_STONE\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_HEAVY\_WOOD (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_HEAVY\_WOOD\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_METAL (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_METAL\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_ORGANIC (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_ORGANIC\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_RUBBER (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_SKIN (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_SKIN\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_SNOW (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_SNOW\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_STONE (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_STONE\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_WATER (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_WATER\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_WOOD (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 HAV\_MAT\_WOOD\_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial* attribute), 188  
 havok\_col\_filter (*pyffi.formats.nif.NifFormat.bhkWorldObject* property), 225  
 havok\_col\_filter (*pyffi.formats.nif.NifFormat.OblivionSubShape* property), 190  
 HEAD (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 heading () (*pyffi.formats.nif.NifFormat.FurniturePosition* property), 93  
 height () (*pyffi.formats.nif.NifFormat.MipMap* property), 101  
 height () (*pyffi.formats.nif.NifFormat.NiPSysBoxEmitter* property), 147  
 height () (*pyffi.formats.nif.NifFormat.NiPSysCylinderEmitter* property), 148  
 height () (*pyffi.formats.nif.NifFormat.NiPSysPlanarCollider* property), 154  
 height () (*pyffi.formats.nif.NifFormat.NiRawImageData* property), 170  
 HeightMaterial (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* attribute), 63  
 Hinge (*pyffi.formats.nif.NifFormat.hkConstraintType* attribute), 226  
 hinge () (*pyffi.formats.nif.NifFormat.bhkHingeConstraint* property), 218  
 hinge () (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint* property), 223  
 hinge () (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207  
 horizontal\_angle () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159  
 horizontal\_direction () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159  
 id () (*pyffi.formats.cgf.CgfFormat.ChunkHeader* property), 12  
 image () (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104  
 image () (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
 image () (*pyffi.formats.nif.NifFormat.NiTextureProperty* property), 180  
 image\_data () (*pyffi.formats.nif.NifFormat.NiImage* property), 125  
 image\_type () (*pyffi.formats.nif.NifFormat.NiRawImageData* property), 170  
 images () (*pyffi.formats.nif.NifFormat.NiFlipController* property), 120  
 importer\_name () (*pyffi.formats.nif.NifFormat.NiArkImporterExtraData* property), 107  
 in\_portals () (*pyffi.formats.nif.NifFormat.NiRoom* property), 170  
 include\_types (*pyffi.spells.Toaster* attribute), 270

*indent* (*pyffi.spells.Toaster* attribute), 270  
*index*() (*pyffi.formats.nif.NifFormat.SemanticData* property), 196  
*index*() (*pyffi.formats.nif.NifFormat.SkinWeight* property), 198  
*indices*() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* property), 214  
*indices\_2*() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* method), 35  
*initial\_axis*() (*pyffi.formats.nif.NifFormat.NiParticleRotation* method), 38  
*initial\_axis*() (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier* method), 43  
*initial\_color*() (*pyffi.formats.nif.NifFormat.NiPSysEmitter* method), 45  
*initial\_radius*() (*pyffi.formats.nif.NifFormat.NiPSysEmitter* method), 237  
*initial\_rotation\_angle*() (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier* property), 154  
*initial\_rotation\_angle\_variation*() (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier* property), 154  
*initial\_rotation\_speed*() (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier* property), 154  
*initial\_rotation\_speed\_variation*() (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier* property), 154  
*initial\_velocity\_type*() (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter* property), 153  
*inspect*() (*pyffi.formats.bsa.BsaFormat.Header* method), 9  
*inspect*() (*pyffi.formats.cgf.CgfFormat.Data* method), 13  
*inspect*() (*pyffi.formats.dae.DaeFormat.Data* method), 33  
*inspect*() (*pyffi.formats.dds.DdsFormat.Data* method), 35  
*inspect*() (*pyffi.formats.egm.EgmFormat.Data* method), 38  
*inspect*() (*pyffi.formats.egt.EgtFormat.Header* method), 43  
*inspect*() (*pyffi.formats.esp.EspFormat.Data* method), 45  
*inspect*() (*pyffi.formats.kfm.KfmFormat.Data* method), 49  
*inspect*() (*pyffi.formats.nif.NifFormat.Data* method), 85  
*inspect*() (*pyffi.formats.tga.TgaFormat.Data* method), 234  
*inspect*() (*pyffi.formats.tri.TriFormat.Header* method), 237  
*inspect*() (*pyffi.object\_models.FileFormat.Data* method), 272  
*inspect\_filename*() (*pyffi.spells.Toaster* method), 270  
*inspect\_quick*() (*pyffi.formats.bsa.BsaFormat.Header* method), 9  
*inspect\_quick*() (*pyffi.formats.dds.DdsFormat.Data* method), 35  
*inspect\_quick*() (*pyffi.formats.egm.EgmFormat.Data* method), 38  
*inspect\_quick*() (*pyffi.formats.egt.EgtFormat.Header* method), 43  
*inspect\_quick*() (*pyffi.formats.esp.EspFormat.Data* method), 45  
*inspect\_quick*() (*pyffi.formats.tri.TriFormat.Header* method), 237  
*inspect\_version\_only*() (*pyffi.formats.cgf.CgfFormat.Data* method), 13  
*inspect\_version\_only*() (*pyffi.formats.nif.NifFormat.Data* method), 86  
*instancing\_enabled*() (*pyffi.formats.nif.NifFormat.NiMesh* property), 128  
*int* (*pyffi.formats.cgf.CgfFormat* attribute), 29  
*int* (*pyffi.formats.dds.DdsFormat* attribute), 36  
*int* (*pyffi.formats.egm.EgmFormat* attribute), 40  
*int* (*pyffi.formats.egt.EgtFormat* attribute), 43  
*int* (*pyffi.formats.esp.EspFormat* attribute), 47  
*int* (*pyffi.formats.kfm.KfmFormat* attribute), 51  
*int* (*pyffi.formats.nif.NifFormat* attribute), 227  
*int* (*pyffi.formats.tga.TgaFormat* attribute), 235  
*int* (*pyffi.formats.tri.TriFormat* attribute), 239  
*integer\_data*() (*pyffi.formats.nif.NifFormat.NiIntegerExtraData* property), 125  
*interface*, 304  
*internal\_index*() (*pyffi.formats.nif.NifFormat.BSSegment* property), 70  
*interpolation*() (*pyffi.formats.nif.NifFormat.KeyGroup* property), 97  
*interpolation*() (*pyffi.formats.nif.NifFormat.Morph* property), 102  
*interpolator*() (*pyffi.formats.nif.NifFormat.BSFrustumFOVController* property), 60  
*interpolator*() (*pyffi.formats.nif.NifFormat.BSRefractionFirePeriodC* property), 69  
*interpolator*() (*pyffi.formats.nif.NifFormat.MorphWeight* property), 102  
*interpolator*() (*pyffi.formats.nif.NifFormat.NiPSEmitterRadiusCtrl* property), 138  
*interpolator*() (*pyffi.formats.nif.NifFormat.NiPSEmitterSpeedCtrl* property), 139  
*interpolator*() (*pyffi.formats.nif.NifFormat.NiPSForceActiveCtrl* property), 139



## K

keys () (*pyffi.formats.nif.NifFormat.KeyGroup* property), 97

keys () (*pyffi.formats.nif.NifFormat.Morph* property), 102

keys () (*pyffi.formats.nif.NifFormat.NiVisData* property), 186

KfmFormat (class in *pyffi.formats.kfm*), 48

KfmFormat.Data (class in *pyffi.formats.kfm*), 48

KfmFormat.FilePath (class in *pyffi.formats.kfm*), 49

KfmFormat.HeaderString (class in *pyffi.formats.kfm*), 49

KfmFormat.SizedString (class in *pyffi.formats.kfm*), 50

KFMXMLPATH, 7

## L

L\_CALF (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_CALF (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

L\_FOOT (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_FOOT (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

L\_FORE\_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_FOREARM (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

L\_HAND (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_HAND (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

L\_THIGH (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_THIGH (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

L\_UPPER\_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91

L\_UPPER\_ARM (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

layer () (*pyffi.formats.nif.NifFormat.bhkCMSDMaterial* property), 215

layer () (*pyffi.formats.nif.NifFormat.HavokColFilter* property), 94

Lean (*pyffi.formats.nif.NifFormat.AnimationType* attribute), 54

left () (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints* property), 93

left\_eye\_reflection\_center () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62

length () (*pyffi.formats.nif.NifFormat.StiffSpringDescriptor* property), 205

level\_0\_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape* property), 61

level\_1\_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape* property), 61

level\_2\_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape* property), 61

life\_span () (*pyffi.formats.nif.NifFormat.NiPSysEmitter* property), 149

life\_span () (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier* property), 155

life\_span\_variation () (*pyffi.formats.nif.NifFormat.NiPSysEmitter* property), 149

life\_span\_variation () (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier* property), 155

lifespan () (*pyffi.formats.nif.NifFormat.Particle* property), 190

lifetime () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

lifetime () (*pyffi.formats.nif.NifFormat.Particle* property), 191

lifetime\_random () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159

LIGHT\_MODE\_EMI\_AMB\_DIF (*pyffi.formats.nif.NifFormat.LightMode* attribute), 97

LIGHT\_MODE\_EMISSIVE (*pyffi.formats.nif.NifFormat.LightMode* attribute), 97

lighting\_effect\_1 () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62

lighting\_effect\_2 () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62

lighting\_mode () (*pyffi.formats.nif.NifFormat.NiVertexColorProperty* property), 186

limited\_hinge () (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint* property), 223

limited\_hinge () (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207

LINE\_OF\_SIGHT (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

linear\_attenuation () (*pyffi.formats.nif.NifFormat.NiPointLight* property), 169

LINEAR\_KEY (*pyffi.formats.nif.NifFormat.KeyType* attribute), 97

linear\_rotation () (*pyffi.formats.nif.NifFormat.BSLagBoneController* property), 61

linear\_velocity ()

(*pyffi.formats.nif.NifFormat.BSLagBoneControllemask\_index()* (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 61  
 (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
 LINEOFSIGHT (*pyffi.formats.nif.NifFormat.Fallout3Layer mask\_w\_index()* (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* attribute), 91  
 (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
 LINEOFSIGHT (*pyffi.formats.nif.NifFormat.SkyrimLayer mass()* (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* attribute), 201  
 (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* property), 223  
 lines() (*pyffi.formats.nif.NifFormat.NiLinesData* MAT\_BABY\_RATTLE (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 127  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 link\_pairs() (*pyffi.formats.nif.NifFormat.SkinShapeGroup* MAT\_BARREL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 198  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 links() (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraint* MAT\_BARREL (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 213  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 links\_2() (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraint* MAT\_CHAIN (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 213  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 lod\_adjust() (*pyffi.formats.nif.NifFormat.NiCamera* MAT\_BOTTLE (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 115  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 lod\_center() (*pyffi.formats.nif.NifFormat.NiLODNode* MAT\_BOTTLECAP (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 126  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 lod\_center() (*pyffi.formats.nif.NifFormat.NiRangeLODData* MAT\_BROKEN\_CONCRETE (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 169  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 lod\_level\_data() (*pyffi.formats.nif.NifFormat.NiLODNode* MAT\_BROKEN\_STONE (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 126  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 lod\_levels() (*pyffi.formats.nif.NifFormat.NiLODNode* MAT\_CHAIN (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 126  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 lod\_levels() (*pyffi.formats.nif.NifFormat.NiRangeLODData* MAT\_CLOTH (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 169  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 logger (*pyffi.spells.Toaster* attribute), 270  
 look\_at() (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* MAT\_CLOTH (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 127  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 look\_at\_node() (*pyffi.formats.nif.NifFormat.NiLookAtController* MAT\_DIRT (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* property), 127  
 (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 loop\_start\_frame() (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* MAT\_DIRT (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 67  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 loop\_start\_frame\_fudge() (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* MAT\_DRAGON (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* property), 67  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 MAT\_ELEVATOR (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 MAT\_GLASS (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 MAT\_GLASS (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 MAT\_GRASS (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 MAT\_GRASS (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 MAT\_GRAVEL (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 MAT\_HEAVY\_METAL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 MAT\_HEAVY\_METAL (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199  
 MAT\_HEAVY\_STONE (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial* attribute), 89  
 MAT\_HEAVY\_STONE (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial* attribute), 199

## M

m\_11() (*pyffi.formats.nif.NifFormat.Matrix22* property), 99  
 m\_12() (*pyffi.formats.nif.NifFormat.Matrix22* property), 99  
 m\_21() (*pyffi.formats.nif.NifFormat.Matrix22* property), 99  
 m\_22() (*pyffi.formats.nif.NifFormat.Matrix22* property), 99  
 magnitude() (*pyffi.formats.nif.NifFormat.NiPSysFieldModifier* property), 151  
 malleable\_constraint() (*pyffi.formats.nif.NifFormat.bhkRDTConstraint* property), 222  
 map\_index() (*pyffi.formats.nif.NifFormat.ShaderTexDesc* property), 196

*attribute*), 199  
 MAT\_HEAVY\_WOOD (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_HEAVY\_WOOD (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_HOLLOW\_METAL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_ICE (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_LIGHT\_WOOD (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_LUNCHBOX (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_MATERIAL\_ARMOR\_HEAVY  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_ARMOR\_LIGHT  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_ARROW  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_AXE\_1HAND  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BASKET  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BLADE\_1HAND  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BLADE\_1HAND\_SMALL  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BLADE\_2HAND  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BLUNT\_2HAND  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BONE  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOOK  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOTTLE\_SMALL  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOULDER\_LARGE  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOULDER\_MEDIUM  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOULDER\_SMALL  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_BOWS\_STAVES  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_CARPET  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_CERAMIC\_MEDIUM  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_CHAIN  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 199  
 MAT\_MATERIAL\_CHAIN\_METAL  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_COIN  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_SHIELD\_HEAVY  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_SHIELD\_LIGHT  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_SKIN\_LARGE  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_SKIN\_SMALL  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_STONE\_AS\_STAIRS  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_MATERIAL\_WOOD\_AS\_STAIRS  
 (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_METAL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_MUD (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_ORGANIC (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_ORGANIC (*pyffi.formats.nif.NifFormat.SkyrimHavokMaterial*  
*attribute*), 200  
 MAT\_PISTOL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_RIFLE (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89  
 MAT\_RUBBER BALL (*pyffi.formats.nif.NifFormat.Fallout3HavokMaterial*  
*attribute*), 89



(*pyffi.formats.nif.NifFormat.NiNode* method), 133  
*merge\_skeleton\_roots()* (*pyffi.formats.nif.NifFormat.NiNode* method), 133  
*mesh\_description()* (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc* property), 166  
 MESH\_PRIMITIVE\_LINESTRIPS (*pyffi.formats.nif.NifFormat.MeshPrimitiveType* attribute), 101  
 MESH\_PRIMITIVE\_POINTS (*pyffi.formats.nif.NifFormat.MeshPrimitiveType* attribute), 101  
 MESH\_PRIMITIVE\_QUADS (*pyffi.formats.nif.NifFormat.MeshPrimitiveType* attribute), 101  
 MESH\_PRIMITIVE\_TRIANGLES (*pyffi.formats.nif.NifFormat.MeshPrimitiveType* attribute), 101  
 MESH\_PRIMITIVE\_TRISTRIPS (*pyffi.formats.nif.NifFormat.MeshPrimitiveType* attribute), 101  
*meshes()* (*pyffi.formats.nif.NifFormat.NiPSysMeshUpdateModifier* property), 153  
*MetaFileFormat* (class in *pyffi.object\_models*), 271  
*min\_distance()* (*pyffi.formats.nif.NifFormat.PrismaticDescription* property), 192  
*min\_num\_to\_spawn()* (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier* property), 155  
 MIP\_FMT\_DEFAULT (*pyffi.formats.nif.NifFormat.MipMapFormat* attribute), 102  
 MIP\_FMT\_NO (*pyffi.formats.nif.NifFormat.MipMapFormat* attribute), 102  
 MIP\_FMT\_YES (*pyffi.formats.nif.NifFormat.MipMapFormat* attribute), 102  
 MO\_QUAL\_BULLET (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 102  
 MO\_QUAL\_CHARACTER (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 102  
 MO\_QUAL\_CRITICAL (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_DEBRIS (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_FIXED (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_INVALID (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_KEYFRAMED (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_KEYFRAMED\_REPORT (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_MOVING (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_QUAL\_USER (*pyffi.formats.nif.NifFormat.MotionQuality* attribute), 103  
 MO\_SYS\_BOX (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_BOX\_STABILIZED (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_CHARACTER (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_DYNAMIC (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_FIXED (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_INVALID (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_KEYFRAMED (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_SPHERE (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_SPHERE\_INERTIA (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
 MO\_SYS\_THIN\_BOX (*pyffi.formats.nif.NifFormat.MotionSystem* attribute), 103  
*model\_projection\_matrix()* (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
*model\_projection\_transform()* (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
*modifier()* (*pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier* property), 65  
*modifier\_name()* (*pyffi.formats.nif.NifFormat.NiPSysModifierCtrl* property), 154  
*modifier\_vertices()* (*pyffi.formats.tri.TriFormat.ModifierRecord* property), 238  
*modifiers()* (*pyffi.formats.nif.NifFormat.NiMesh* property), 128  
*modifiers()* (*pyffi.formats.nif.NifFormat.NiParticleSystem* property), 158  
*mopp\_from\_tree()* (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape* method), 220  
*msg()* (*pyffi.spells.Toaster* method), 270  
*msgblockbegin()* (*pyffi.spells.Toaster* method), 270  
*msgblockend()* (*pyffi.spells.Toaster* method), 270  
*multi\_bound()* (*pyffi.formats.nif.NifFormat.BSMultiBoundNode* property), 64  
*multiplier()* (*pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolation* property), 109

## N

<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.AVObject</i> property), 53	<code>NifFormat.AdditionalDataBlock</code> (class in <i>pyffi.formats.nif</i> ), 53
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.bhkRagdollTemplate</i> property), 223	<code>NifFormat.AdditionalDataInfo</code> (class in <i>pyffi.formats.nif</i> ), 54
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.bhkRagdollTemplateData</i> property), 223	<code>NifFormat.AlphaFormat</code> (class in <i>pyffi.formats.nif</i> ), 54
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.BSTreadTransform</i> property), 75	<code>NifFormat.AnimationType</code> (class in <i>pyffi.formats.nif</i> ), 54
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.Ni3dsAnimationNode</i> property), 104	<code>NifFormat.ApplyMode</code> (class in <i>pyffi.formats.nif</i> ), 55
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiExtraData</i> property), 119	<code>NifFormat.ArkTexture</code> (class in <i>pyffi.formats.nif</i> ), 55
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSBombForce</i> property), 136	<code>NifFormat.ATextureRenderData</code> (class in <i>pyffi.formats.nif</i> ), 53
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSBoxEmitter</i> property), 136	<code>NifFormat.AVObject</code> (class in <i>pyffi.formats.nif</i> ), 53
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSMeshEmitter</i> property), 140	<code>NifFormat.AvoidNode</code> (class in <i>pyffi.formats.nif</i> ), 55
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSPlanarCollider</i> property), 143	<code>NifFormat.BallAndSocketDescriptor</code> (class in <i>pyffi.formats.nif</i> ), 76
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSSphereEmitter</i> property), 145	<code>NifFormat.bhkAabbPhantom</code> (class in <i>pyffi.formats.nif</i> ), 212
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSysModifier</i> property), 153	<code>NifFormat.bhkBallAndSocketConstraint</code> (class in <i>pyffi.formats.nif</i> ), 213
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiSequence</i> property), 172	<code>NifFormat.bhkBallSocketConstraintChain</code> (class in <i>pyffi.formats.nif</i> ), 213
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.NiShadowGenerator</i> property), 173	<code>NifFormat.bhkBlendCollisionObject</code> (class in <i>pyffi.formats.nif</i> ), 213
<code>name ()</code> ( <i>pyffi.formats.nif.NifFormat.SemanticData</i> property), 196	<code>NifFormat.bhkBlendController</code> (class in <i>pyffi.formats.nif</i> ), 213
<code>name ()</code> ( <i>pyffi.formats.tri.TriFormat.ModifierRecord</i> property), 238	<code>NifFormat.bhkBoxShape</code> (class in <i>pyffi.formats.nif</i> ), 214
<code>name_attribute ()</code> ( <i>pyffi.object_models.FileFormat</i> class method), 272	<code>NifFormat.bhkBreakableConstraint</code> (class in <i>pyffi.formats.nif</i> ), 214
<code>name_class ()</code> ( <i>pyffi.object_models.FileFormat</i> class method), 272	<code>NifFormat.bhkBvTreeShape</code> (class in <i>pyffi.formats.nif</i> ), 214
<code>name_parts ()</code> ( <i>pyffi.object_models.FileFormat</i> class method), 272	<code>NifFormat.bhkCapsuleShape</code> (class in <i>pyffi.formats.nif</i> ), 215
<code>near_extent ()</code> ( <i>pyffi.formats.nif.NifFormat.LODRange</i> property), 97	<code>NifFormat.bhkCMSDBigTris</code> (class in <i>pyffi.formats.nif</i> ), 214
<code>next_collider ()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSysCollider</i> property), 147	<code>NifFormat.bhkCMSDChunk</code> (class in <i>pyffi.formats.nif</i> ), 214
<code>next_controller ()</code> ( <i>pyffi.formats.nif.NifFormat.NiTimeController</i> property), 182	<code>NifFormat.bhkCMSDMaterial</code> (class in <i>pyffi.formats.nif</i> ), 215
<code>next_extra_data ()</code> ( <i>pyffi.formats.nif.NifFormat.NiExtraData</i> property), 119	<code>NifFormat.bhkCMSDTransform</code> (class in <i>pyffi.formats.nif</i> ), 215
<code>next_modifier ()</code> ( <i>pyffi.formats.nif.NifFormat.NiParticleModifier</i> property), 158	<code>NifFormat.bhkCollisionObject</code> (class in <i>pyffi.formats.nif</i> ), 215
<code>NifFormat</code> (class in <i>pyffi.formats.nif</i> ), 53	<code>NifFormat.bhkCompressedMeshShape</code> (class in <i>pyffi.formats.nif</i> ), 215
<code>NifFormat.AbstractAdditionalGeometryData</code> (class in <i>pyffi.formats.nif</i> ), 53	<code>NifFormat.bhkCompressedMeshShapeData</code> (class in <i>pyffi.formats.nif</i> ), 216
	<code>NifFormat.bhkConstraint</code> (class in <i>pyffi.formats.nif</i> ), 217
	<code>NifFormat.bhkConvexListShape</code> (class in <i>pyffi.formats.nif</i> ), 217

<i>pyffi.formats.nif</i> ), 217		<i>pyffi.formats.nif</i> ), 224
NifFormat.bhkConvexShape (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkSerializable (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkConvexTransformShape (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkShape (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkConvexVerticesShape (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkShapeCollection (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkEntity (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkShapePhantom (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkHingeConstraint (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkSimpleShapePhantom (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkLimitedHingeConstraint (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkSPCollisionObject (class in <i>pyffi.formats.nif</i> ), 224
NifFormat.bhkLiquidAction (class in <i>pyffi.formats.nif</i> ), 218		NifFormat.bhkSphereRepShape (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkListShape (class in <i>pyffi.formats.nif</i> ), 219		NifFormat.bhkSphereShape (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkMalleableConstraint (class in <i>pyffi.formats.nif</i> ), 219		NifFormat.bhkStiffSpringConstraint (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkMeshShape (class in <i>pyffi.formats.nif</i> ), 219		NifFormat.bhkTransformShape (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkMoppBvTreeShape (class in <i>pyffi.formats.nif</i> ), 219		NifFormat.bhkWorldObject (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkMultiSphereShape (class in <i>pyffi.formats.nif</i> ), 220		NifFormat.BillboardMode (class in <i>pyffi.formats.nif</i> ), 76
NifFormat.bhkNiCollisionObject (class in <i>pyffi.formats.nif</i> ), 220		NifFormat.BodyPartList (class in <i>pyffi.formats.nif</i> ), 77
NifFormat.bhkNiTriStripsShape (class in <i>pyffi.formats.nif</i> ), 220		NifFormat.BoneLOD (class in <i>pyffi.formats.nif</i> ), 77
NifFormat.bhkOrientHingedBodyAction (class in <i>pyffi.formats.nif</i> ), 220		NifFormat.bool (class in <i>pyffi.formats.nif</i> ), 225
NifFormat.bhkPackedNiTriStripsShape (class in <i>pyffi.formats.nif</i> ), 221		NifFormat.BoundingBox (class in <i>pyffi.formats.nif</i> ), 77
NifFormat.bhkPCollisionObject (class in <i>pyffi.formats.nif</i> ), 220		NifFormat.BoundingVolume (class in <i>pyffi.formats.nif</i> ), 78
NifFormat.bhkPhantom (class in <i>pyffi.formats.nif</i> ), 222		NifFormat.BoundVolumeType (class in <i>pyffi.formats.nif</i> ), 77
NifFormat.bhkPrismaticConstraint (class in <i>pyffi.formats.nif</i> ), 222		NifFormat.BoxBV (class in <i>pyffi.formats.nif</i> ), 78
NifFormat.bhkRagdollConstraint (class in <i>pyffi.formats.nif</i> ), 223		NifFormat.BSAnimNotes (class in <i>pyffi.formats.nif</i> ), 55
NifFormat.bhkRagdollTemplate (class in <i>pyffi.formats.nif</i> ), 223		NifFormat.BSBehaviorGraphExtraData (class in <i>pyffi.formats.nif</i> ), 55
NifFormat.bhkRagdollTemplateData (class in <i>pyffi.formats.nif</i> ), 223		NifFormat.BSBlastNode (class in <i>pyffi.formats.nif</i> ), 55
NifFormat.bhkRDTConstraint (class in <i>pyffi.formats.nif</i> ), 222		NifFormat.BSBoneLODExtraData (class in <i>pyffi.formats.nif</i> ), 55
NifFormat.bhkRDTMalleableConstraint (class in <i>pyffi.formats.nif</i> ), 222		NifFormat.BSBound (class in <i>pyffi.formats.nif</i> ), 56
NifFormat.bhkRefObject (class in <i>pyffi.formats.nif</i> ), 224		NifFormat.BSDamageStage (class in <i>pyffi.formats.nif</i> ), 56
NifFormat.bhkRigidBody (class in <i>pyffi.formats.nif</i> ), 224		NifFormat.BSDebrisNode (class in <i>pyffi.formats.nif</i> ), 56
NifFormat.bhkRigidBodyT (class in <i>pyffi.formats.nif</i> ), 224		NifFormat.BSDecalPlacementVectorExtraData (class in <i>pyffi.formats.nif</i> ), 56
		NifFormat.BSDismemberBodyPartType (class in <i>pyffi.formats.nif</i> ), 56
		NifFormat.BSDismemberSkinInstance (class

*in pyffi.formats.nif*), 59  
 NifFormat.BSDistantTreeShaderProperty (class in *pyffi.formats.nif*), 59  
 NifFormat.BSEffectShaderProperty (class in *pyffi.formats.nif*), 59  
 NifFormat.BSEffectShaderPropertyColorControl (class in *pyffi.formats.nif*), 60  
 NifFormat.BSEffectShaderPropertyFloatControl (class in *pyffi.formats.nif*), 60  
 NifFormat.BSFadeNode (class in *pyffi.formats.nif*), 60  
 NifFormat.BSFrustumFOVController (class in *pyffi.formats.nif*), 60  
 NifFormat.BSFurnitureMarker (class in *pyffi.formats.nif*), 60  
 NifFormat.BSFurnitureMarkerNode (class in *pyffi.formats.nif*), 60  
 NifFormat.BSInvMarker (class in *pyffi.formats.nif*), 61  
 NifFormat.BSKeyframeController (class in *pyffi.formats.nif*), 61  
 NifFormat.BSLagBoneController (class in *pyffi.formats.nif*), 61  
 NifFormat.BSLeafAnimNode (class in *pyffi.formats.nif*), 61  
 NifFormat.BSLightingShaderProperty (class in *pyffi.formats.nif*), 61  
 NifFormat.BSLightingShaderPropertyColorControl (class in *pyffi.formats.nif*), 63  
 NifFormat.BSLightingShaderPropertyFloatControl (class in *pyffi.formats.nif*), 63  
 NifFormat.BSLightingShaderPropertyShaderType (class in *pyffi.formats.nif*), 63  
 NifFormat.BSLODTriShape (class in *pyffi.formats.nif*), 61  
 NifFormat.BSMasterParticleSystem (class in *pyffi.formats.nif*), 63  
 NifFormat.BSMaterialEmittanceMultControl (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBound (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBoundAABB (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBoundData (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBoundNode (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBoundOBB (class in *pyffi.formats.nif*), 64  
 NifFormat.BSMultiBoundSphere (class in *pyffi.formats.nif*), 64  
 NifFormat.BSNiAlphaPropertyTestRefController (class in *pyffi.formats.nif*), 65  
 NifFormat.BSOrderedNode (class in *pyffi.formats.nif*), 65  
 NifFormat.BSPackedAdditionalDataBlock (class in *pyffi.formats.nif*), 67  
 NifFormat.BSPackedAdditionalGeometryData (class in *pyffi.formats.nif*), 68  
 NifFormat.BSParentVelocityModifier (class in *pyffi.formats.nif*), 68  
 NifFormat.BSPartFlag (class in *pyffi.formats.nif*), 68  
 NifFormat.BSProceduralLightningController (class in *pyffi.formats.nif*), 68  
 NifFormat.BSPSysArrayEmitter (class in *pyffi.formats.nif*), 65  
 NifFormat.BSPSysHavokUpdateModifier (class in *pyffi.formats.nif*), 65  
 NifFormat.BSPSysInheritVelocityModifier (class in *pyffi.formats.nif*), 65  
 NifFormat.BSPSysLODModifier (class in *pyffi.formats.nif*), 65  
 NifFormat.BSPSysMultiTargetEmitterCtrlr (class in *pyffi.formats.nif*), 66  
 NifFormat.BSPSysRecycleBoundModifier (class in *pyffi.formats.nif*), 66  
 NifFormat.BSPSysScaleModifier (class in *pyffi.formats.nif*), 66  
 NifFormat.BSPSysSimpleColorModifier (class in *pyffi.formats.nif*), 66  
 NifFormat.BSPSysStripUpdateModifier (class in *pyffi.formats.nif*), 67  
 NifFormat.BSPSysSubTexModifier (class in *pyffi.formats.nif*), 67  
 NifFormat.BSRefractionFirePeriodController (class in *pyffi.formats.nif*), 69  
 NifFormat.BSRefractionStrengthController (class in *pyffi.formats.nif*), 69  
 NifFormat.BSRotAccumTransfInterpolator (class in *pyffi.formats.nif*), 70  
 NifFormat.BSSegment (class in *pyffi.formats.nif*), 70  
 NifFormat.BSSegmentedTriShape (class in *pyffi.formats.nif*), 70  
 NifFormat.BSSegmentFlags (class in *pyffi.formats.nif*), 70  
 NifFormat.BSShaderFlags (class in *pyffi.formats.nif*), 70  
 NifFormat.BSShaderFlags2 (class in *pyffi.formats.nif*), 71  
 NifFormat.BSShaderLightingProperty (class in *pyffi.formats.nif*), 72  
 NifFormat.BSShaderNoLightingProperty (class in *pyffi.formats.nif*), 72  
 NifFormat.BSShaderPPLightingProperty (class in *pyffi.formats.nif*), 72  
 NifFormat.BSShaderProperty (class in

<i>pyffi.formats.nif</i> ), 73		<i>pyffi.formats.nif</i> ), 83	
NifFormat.BSShaderTextureSet (class in <i>pyffi.formats.nif</i> ), 73		NifFormat.CoordGenType (class in <i>pyffi.formats.nif</i> ), 84	
NifFormat.BSShaderType (class in <i>pyffi.formats.nif</i> ), 73		NifFormat.CStreamableAssetData (class in <i>pyffi.formats.nif</i> ), 79	
NifFormat.BSSkyShaderProperty (class in <i>pyffi.formats.nif</i> ), 74		NifFormat.CycleType (class in <i>pyffi.formats.nif</i> ), 84	
NifFormat.BSStripParticleSystem (class in <i>pyffi.formats.nif</i> ), 74		NifFormat.Data (class in <i>pyffi.formats.nif</i> ), 85	
NifFormat.BSStripPSysData (class in <i>pyffi.formats.nif</i> ), 74		NifFormat.Data.VersionUInt (class in <i>pyffi.formats.nif</i> ), 85	
NifFormat.BSTreadTransfInterpolator (class in <i>pyffi.formats.nif</i> ), 74		NifFormat.DataStreamAccess (class in <i>pyffi.formats.nif</i> ), 86	
NifFormat.BSTreadTransform (class in <i>pyffi.formats.nif</i> ), 75		NifFormat.DataStreamUsage (class in <i>pyffi.formats.nif</i> ), 86	
NifFormat.BSTreadTransformData (class in <i>pyffi.formats.nif</i> ), 75		NifFormat.DeactivatorType (class in <i>pyffi.formats.nif</i> ), 86	
NifFormat.BSTreeNode (class in <i>pyffi.formats.nif</i> ), 75		NifFormat.DecalVectorArray (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.BSValueNode (class in <i>pyffi.formats.nif</i> ), 75		NifFormat.DecayType (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.BSWArray (class in <i>pyffi.formats.nif</i> ), 75		NifFormat.DistantLODShaderProperty (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.BSWaterShaderProperty (class in <i>pyffi.formats.nif</i> ), 76		NifFormat.EffectShaderControlledColor (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.BSWindModifier (class in <i>pyffi.formats.nif</i> ), 76		NifFormat.EffectShaderControlledVariable (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.BSXFlags (class in <i>pyffi.formats.nif</i> ), 76		NifFormat.EffectType (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.ByteArray (class in <i>pyffi.formats.nif</i> ), 78		NifFormat.ElementReference (class in <i>pyffi.formats.nif</i> ), 87	
NifFormat.ByteColor3 (class in <i>pyffi.formats.nif</i> ), 78		NifFormat.EmitFrom (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.ByteColor4 (class in <i>pyffi.formats.nif</i> ), 79		NifFormat.EndianType (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.ByteMatrix (class in <i>pyffi.formats.nif</i> ), 79		NifFormat.ExportInfo (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.CapsuleBV (class in <i>pyffi.formats.nif</i> ), 79		NifFormat.ExtraMeshDataEpicMickey (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.ChannelConvention (class in <i>pyffi.formats.nif</i> ), 80		NifFormat.ExtraMeshDataEpicMickey2 (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.ChannelData (class in <i>pyffi.formats.nif</i> ), 80		NifFormat.ExtraVectorsFlags (class in <i>pyffi.formats.nif</i> ), 88	
NifFormat.ChannelType (class in <i>pyffi.formats.nif</i> ), 80		NifFormat.FaceDrawMode (class in <i>pyffi.formats.nif</i> ), 89	
NifFormat.CloningBehavior (class in <i>pyffi.formats.nif</i> ), 80		NifFormat.Fallout3HavokMaterial (class in <i>pyffi.formats.nif</i> ), 89	
NifFormat.CollisionMode (class in <i>pyffi.formats.nif</i> ), 81		NifFormat.Fallout3Layer (class in <i>pyffi.formats.nif</i> ), 90	
NifFormat.Color3 (class in <i>pyffi.formats.nif</i> ), 81		NifFormat.FieldType (class in <i>pyffi.formats.nif</i> ), 92	
NifFormat.Color4 (class in <i>pyffi.formats.nif</i> ), 81		NifFormat.FilePath (class in <i>pyffi.formats.nif</i> ), 92	
NifFormat.ComponentFormat (class in <i>pyffi.formats.nif</i> ), 81		NifFormat.FileVersion (class in <i>pyffi.formats.nif</i> ), 92	
NifFormat.ConsistencyType (class in <i>pyffi.formats.nif</i> ), 83		NifFormat.Flags (class in <i>pyffi.formats.nif</i> ), 92	
NifFormat.ControllerLink (class in <i>pyffi.formats.nif</i> ), 83		NifFormat.Footer (class in <i>pyffi.formats.nif</i> ), 92	

NifFormat.ForceType (class in *pyffi.formats.nif*), 93  
 NifFormat.FurnitureEntryPoints (class in *pyffi.formats.nif*), 93  
 NifFormat.FurniturePosition (class in *pyffi.formats.nif*), 93  
 NifFormat.FxButton (class in *pyffi.formats.nif*), 93  
 NifFormat.FxRadioButton (class in *pyffi.formats.nif*), 93  
 NifFormat.FxWidget (class in *pyffi.formats.nif*), 94  
 NifFormat.HairShaderProperty (class in *pyffi.formats.nif*), 94  
 NifFormat.HalfSpaceBV (class in *pyffi.formats.nif*), 94  
 NifFormat.HavokColFilter (class in *pyffi.formats.nif*), 94  
 NifFormat.HavokMaterial (class in *pyffi.formats.nif*), 95  
 NifFormat.Header (class in *pyffi.formats.nif*), 95  
 NifFormat.HeaderString (class in *pyffi.formats.nif*), 95  
 NifFormat.HingeDescriptor (class in *pyffi.formats.nif*), 95  
 NifFormat.hkConstraintType (class in *pyffi.formats.nif*), 226  
 NifFormat.hkPackedNiTriStripsData (class in *pyffi.formats.nif*), 226  
 NifFormat.hkResponseType (class in *pyffi.formats.nif*), 226  
 NifFormat.hkTriangle (class in *pyffi.formats.nif*), 227  
 NifFormat.ImageType (class in *pyffi.formats.nif*), 96  
 NifFormat.InertiaMatrix (class in *pyffi.formats.nif*), 96  
 NifFormat.Key (class in *pyffi.formats.nif*), 96  
 NifFormat.KeyGroup (class in *pyffi.formats.nif*), 97  
 NifFormat.KeyType (class in *pyffi.formats.nif*), 97  
 NifFormat.Lighting30ShaderProperty (class in *pyffi.formats.nif*), 97  
 NifFormat.LightingShaderControlledColor (class in *pyffi.formats.nif*), 98  
 NifFormat.LightingShaderControlledVariable (class in *pyffi.formats.nif*), 98  
 NifFormat.LightMode (class in *pyffi.formats.nif*), 97  
 NifFormat.LimitedHingeDescriptor (class in *pyffi.formats.nif*), 98  
 NifFormat.LineString (class in *pyffi.formats.nif*), 98  
 NifFormat.LODRange (class in *pyffi.formats.nif*), 97  
 NifFormat.MatchGroup (class in *pyffi.formats.nif*), 99  
 NifFormat.MaterialData (class in *pyffi.formats.nif*), 99  
 NifFormat.Matrix22 (class in *pyffi.formats.nif*), 99  
 NifFormat.Matrix33 (class in *pyffi.formats.nif*), 99  
 NifFormat.Matrix44 (class in *pyffi.formats.nif*), 100  
 NifFormat.MeshData (class in *pyffi.formats.nif*), 101  
 NifFormat.MeshPrimitiveType (class in *pyffi.formats.nif*), 101  
 NifFormat.MipMap (class in *pyffi.formats.nif*), 101  
 NifFormat.MipMapFormat (class in *pyffi.formats.nif*), 102  
 NifFormat.MoppDataBuildType (class in *pyffi.formats.nif*), 102  
 NifFormat.Morph (class in *pyffi.formats.nif*), 102  
 NifFormat.MorphWeight (class in *pyffi.formats.nif*), 102  
 NifFormat.MotionQuality (class in *pyffi.formats.nif*), 102  
 NifFormat.MotionSystem (class in *pyffi.formats.nif*), 103  
 NifFormat.MotorDescriptor (class in *pyffi.formats.nif*), 103  
 NifFormat.MTransform (class in *pyffi.formats.nif*), 99  
 NifFormat.MultiTextureElement (class in *pyffi.formats.nif*), 103  
 NifFormat.Ni3dsAlphaAnimator (class in *pyffi.formats.nif*), 104  
 NifFormat.Ni3dsAnimationNode (class in *pyffi.formats.nif*), 104  
 NifFormat.Ni3dsColorAnimator (class in *pyffi.formats.nif*), 105  
 NifFormat.Ni3dsMorphShape (class in *pyffi.formats.nif*), 105  
 NifFormat.Ni3dsParticleSystem (class in *pyffi.formats.nif*), 105  
 NifFormat.Ni3dsPathController (class in *pyffi.formats.nif*), 105  
 NifFormat.NiAdditionalGeometryData (class in *pyffi.formats.nif*), 106  
 NifFormat.NiAlphaController (class in *pyffi.formats.nif*), 107  
 NifFormat.NiAlphaProperty (class in *pyffi.formats.nif*), 107  
 NifFormat.NiAmbientLight (class in *pyffi.formats.nif*), 107  
 NifFormat.NiArkAnimationExtraData (class in *pyffi.formats.nif*), 107  
 NifFormat.NiArkImporterExtraData (class in *pyffi.formats.nif*), 107  
 NifFormat.NiArkShaderExtraData (class in *pyffi.formats.nif*), 108  
 NifFormat.NiArkTextureExtraData (class in

*pyffi.formats.nif*), 108  
 NifFormat.NiArkViewportInfoExtraData (class in *pyffi.formats.nif*), 108  
 NifFormat.NiAutoNormalParticles (class in *pyffi.formats.nif*), 108  
 NifFormat.NiAutoNormalParticlesData (class in *pyffi.formats.nif*), 108  
 NifFormat.NiAVObject (class in *pyffi.formats.nif*), 105  
 NifFormat.NiAVObjectPalette (class in *pyffi.formats.nif*), 106  
 NifFormat.NiBezierMesh (class in *pyffi.formats.nif*), 111  
 NifFormat.NiBezierTriangle4 (class in *pyffi.formats.nif*), 112  
 NifFormat.NiBillboardNode (class in *pyffi.formats.nif*), 112  
 NifFormat.NiBinaryExtraData (class in *pyffi.formats.nif*), 113  
 NifFormat.NiBinaryVoxelData (class in *pyffi.formats.nif*), 113  
 NifFormat.NiBinaryVoxelExtraData (class in *pyffi.formats.nif*), 113  
 NifFormat.NiBlendBoolInterpolator (class in *pyffi.formats.nif*), 113  
 NifFormat.NiBlendFloatInterpolator (class in *pyffi.formats.nif*), 113  
 NifFormat.NiBlendInterpolator (class in *pyffi.formats.nif*), 114  
 NifFormat.NiBlendPoint3Interpolator (class in *pyffi.formats.nif*), 114  
 NifFormat.NiBlendTransformInterpolator (class in *pyffi.formats.nif*), 114  
 NifFormat.NiBone (class in *pyffi.formats.nif*), 114  
 NifFormat.NiBoneLODController (class in *pyffi.formats.nif*), 114  
 NifFormat.NiBoolData (class in *pyffi.formats.nif*), 115  
 NifFormat.NiBooleanExtraData (class in *pyffi.formats.nif*), 115  
 NifFormat.NiBoolInterpController (class in *pyffi.formats.nif*), 115  
 NifFormat.NiBoolInterpolator (class in *pyffi.formats.nif*), 115  
 NifFormat.NiBoolTimelineInterpolator (class in *pyffi.formats.nif*), 115  
 NifFormat.NiBSAnimationNode (class in *pyffi.formats.nif*), 108  
 NifFormat.NiBSBoneLODController (class in *pyffi.formats.nif*), 108  
 NifFormat.NiBSPArrayController (class in *pyffi.formats.nif*), 108  
 NifFormat.NiBSPParticleNode (class in *pyffi.formats.nif*), 108  
 NifFormat.NiBSplineBasisData (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineCompFloatInterpolator (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineCompPoint3Interpolator (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineCompTransformEvaluator (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineCompTransformInterpolator (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineData (class in *pyffi.formats.nif*), 109  
 NifFormat.NiBSplineFloatInterpolator (class in *pyffi.formats.nif*), 111  
 NifFormat.NiBSplineInterpolator (class in *pyffi.formats.nif*), 111  
 NifFormat.NiBSplinePoint3Interpolator (class in *pyffi.formats.nif*), 111  
 NifFormat.NiBSplineTransformInterpolator (class in *pyffi.formats.nif*), 111  
 NifFormat.NiCamera (class in *pyffi.formats.nif*), 115  
 NifFormat.NiClod (class in *pyffi.formats.nif*), 116  
 NifFormat.NiClodData (class in *pyffi.formats.nif*), 116  
 NifFormat.NiClodSkinInstance (class in *pyffi.formats.nif*), 116  
 NifFormat.NiCollisionData (class in *pyffi.formats.nif*), 117  
 NifFormat.NiCollisionObject (class in *pyffi.formats.nif*), 117  
 NifFormat.NiColorData (class in *pyffi.formats.nif*), 117  
 NifFormat.NiColorExtraData (class in *pyffi.formats.nif*), 117  
 NifFormat.NiControllerManager (class in *pyffi.formats.nif*), 117  
 NifFormat.NiControllerSequence (class in *pyffi.formats.nif*), 117  
 NifFormat.NiDataStream (class in *pyffi.formats.nif*), 118  
 NifFormat.NiDefaultAVObjectPalette (class in *pyffi.formats.nif*), 118  
 NifFormat.NiDirectionalLight (class in *pyffi.formats.nif*), 118  
 NifFormat.NiDitherProperty (class in *pyffi.formats.nif*), 118  
 NifFormat.NiDynamicEffect (class in *pyffi.formats.nif*), 119  
 NifFormat.NiEnvMappedTriShape (class in *pyffi.formats.nif*), 119  
 NifFormat.NiEnvMappedTriShapeData (class in *pyffi.formats.nif*), 119  
 NifFormat.NiExtraData (class in

*pyffi.formats.nif*), 119  
 NifFormat.NiExtraDataController (class in *pyffi.formats.nif*), 119  
 NifFormat.NiError, 187  
 NifFormat.NiFlipController (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatData (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatExtraData (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatExtraDataController (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatInterpController (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatInterpolator (class in *pyffi.formats.nif*), 120  
 NifFormat.NiFloatsExtraData (class in *pyffi.formats.nif*), 121  
 NifFormat.NiFogProperty (class in *pyffi.formats.nif*), 121  
 NifFormat.NiFurSpringController (class in *pyffi.formats.nif*), 121  
 NifFormat.NiGeometry (class in *pyffi.formats.nif*), 122  
 NifFormat.NiGeometryData (class in *pyffi.formats.nif*), 124  
 NifFormat.NiGeomMorpherController (class in *pyffi.formats.nif*), 121  
 NifFormat.NiGravity (class in *pyffi.formats.nif*), 125  
 NifFormat.NiImage (class in *pyffi.formats.nif*), 125  
 NifFormat.NiInstancingMeshModifier (class in *pyffi.formats.nif*), 125  
 NifFormat.NiIntegerExtraData (class in *pyffi.formats.nif*), 125  
 NifFormat.NiIntegersExtraData (class in *pyffi.formats.nif*), 126  
 NifFormat.NiInterpController (class in *pyffi.formats.nif*), 126  
 NifFormat.NiInterpolator (class in *pyffi.formats.nif*), 126  
 NifFormat.NiKeyBasedInterpolator (class in *pyffi.formats.nif*), 126  
 NifFormat.NiKeyframeController (class in *pyffi.formats.nif*), 126  
 NifFormat.NiKeyframeData (class in *pyffi.formats.nif*), 126  
 NifFormat.NiLight (class in *pyffi.formats.nif*), 126  
 NifFormat.NiLightColorController (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLightDimmerController (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLightIntensityController (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLines (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLinesData (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLODDData (class in *pyffi.formats.nif*), 126  
 NifFormat.NiLODNode (class in *pyffi.formats.nif*), 126  
 NifFormat.NiLookAtController (class in *pyffi.formats.nif*), 127  
 NifFormat.NiLookAtInterpolator (class in *pyffi.formats.nif*), 127  
 NifFormat.NiMaterialColorController (class in *pyffi.formats.nif*), 128  
 NifFormat.NiMaterialProperty (class in *pyffi.formats.nif*), 128  
 NifFormat.NiMesh (class in *pyffi.formats.nif*), 128  
 NifFormat.NiMeshHWInstance (class in *pyffi.formats.nif*), 129  
 NifFormat.NiMeshModifier (class in *pyffi.formats.nif*), 129  
 NifFormat.NiMeshParticleSystem (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMeshPSysData (class in *pyffi.formats.nif*), 129  
 NifFormat.NiMorphController (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMorphData (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMorpherController (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMorphMeshModifier (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMorphWeightsController (class in *pyffi.formats.nif*), 130  
 NifFormat.NiMultiTargetTransformController (class in *pyffi.formats.nif*), 131  
 NifFormat.NiMultiTextureProperty (class in *pyffi.formats.nif*), 131  
 NifFormat.NiNode (class in *pyffi.formats.nif*), 131  
 NifFormat.NiObject (class in *pyffi.formats.nif*), 134  
 NifFormat.NiObjectNET (class in *pyffi.formats.nif*), 134  
 NifFormat.NiPalette (class in *pyffi.formats.nif*), 156  
 NifFormat.NiParticleBomb (class in *pyffi.formats.nif*), 157  
 NifFormat.NiParticleColorModifier (class in *pyffi.formats.nif*), 157  
 NifFormat.NiParticleGrowFade (class in *pyffi.formats.nif*), 157  
 NifFormat.NiParticleMeshes (class in *pyffi.formats.nif*), 158  
 NifFormat.NiParticleMeshesData (class in

*pyffi.formats.nif*), 158  
 NifFormat.NiParticleMeshModifier (class in *pyffi.formats.nif*), 157  
 NifFormat.NiParticleModifier (class in *pyffi.formats.nif*), 158  
 NifFormat.NiParticleRotation (class in *pyffi.formats.nif*), 158  
 NifFormat.NiParticles (class in *pyffi.formats.nif*), 161  
 NifFormat.NiParticlesData (class in *pyffi.formats.nif*), 161  
 NifFormat.NiParticleSystem (class in *pyffi.formats.nif*), 158  
 NifFormat.NiParticleSystemController (class in *pyffi.formats.nif*), 159  
 NifFormat.NiPathController (class in *pyffi.formats.nif*), 162  
 NifFormat.NiPathInterpolator (class in *pyffi.formats.nif*), 162  
 NifFormat.NiPersistentSrcTextureRenderer (class in *pyffi.formats.nif*), 162  
 NifFormat.NiPhysXActorDesc (class in *pyffi.formats.nif*), 163  
 NifFormat.NiPhysXBodyDesc (class in *pyffi.formats.nif*), 164  
 NifFormat.NiPhysXD6JointDesc (class in *pyffi.formats.nif*), 164  
 NifFormat.NiPhysXKinematicSrc (class in *pyffi.formats.nif*), 164  
 NifFormat.NiPhysXMaterialDesc (class in *pyffi.formats.nif*), 164  
 NifFormat.NiPhysXMeshDesc (class in *pyffi.formats.nif*), 164  
 NifFormat.NiPhysXProp (class in *pyffi.formats.nif*), 165  
 NifFormat.NiPhysXPropDesc (class in *pyffi.formats.nif*), 165  
 NifFormat.NiPhysXShapeDesc (class in *pyffi.formats.nif*), 166  
 NifFormat.NiPhysXTransformDest (class in *pyffi.formats.nif*), 167  
 NifFormat.NiPixelData (class in *pyffi.formats.nif*), 167  
 NifFormat.NiPlanarCollider (class in *pyffi.formats.nif*), 167  
 NifFormat.NiPoint3InterpController (class in *pyffi.formats.nif*), 168  
 NifFormat.NiPoint3Interpolator (class in *pyffi.formats.nif*), 168  
 NifFormat.NiPointLight (class in *pyffi.formats.nif*), 169  
 NifFormat.NiPortal (class in *pyffi.formats.nif*), 169  
 NifFormat.NiPosData (class in *pyffi.formats.nif*), 169  
 NifFormat.NiProperty (class in *pyffi.formats.nif*), 169  
 NifFormat.NiPSBombForce (class in *pyffi.formats.nif*), 136  
 NifFormat.NiPSBoundUpdater (class in *pyffi.formats.nif*), 136  
 NifFormat.NiPSBoxEmitter (class in *pyffi.formats.nif*), 136  
 NifFormat.NiPSCylinderEmitter (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSDragForce (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitParticlesCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterDeclinationCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterDeclinationVarCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterLifeSpanCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterPlanarAngleCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterPlanarAngleVarCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterRadiusCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterRotAngleCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterRotAngleVarCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterRotSpeedCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterRotSpeedVarCtrl (class in *pyffi.formats.nif*), 138  
 NifFormat.NiPSEmitterSpeedCtrl (class in *pyffi.formats.nif*), 139  
 NifFormat.NiPSFacingQuadGenerator (class in *pyffi.formats.nif*), 139  
 NifFormat.NiPSForceActiveCtrl (class in *pyffi.formats.nif*), 139  
 NifFormat.NiPSGravityForce (class in *pyffi.formats.nif*), 139  
 NifFormat.NiPSGravityStrengthCtrl (class in *pyffi.formats.nif*), 140  
 NifFormat.NiPSMeshEmitter (class in *pyffi.formats.nif*), 140  
 NifFormat.NiPSMeshParticleSystem (class in *pyffi.formats.nif*), 141  
 NifFormat.NiPSParticleSystem (class in *pyffi.formats.nif*), 141  
 NifFormat.NiPSPlanarCollider (class in *pyffi.formats.nif*), 143  
 NifFormat.NiPSResetOnLoopCtrl (class in

*pyffi.formats.nif*), 143  
 NifFormat.NiPSSimulator (class in *pyffi.formats.nif*), 143  
 NifFormat.NiPSSimulatorCollidersStep (class in *pyffi.formats.nif*), 143  
 NifFormat.NiPSSimulatorFinalStep (class in *pyffi.formats.nif*), 143  
 NifFormat.NiPSSimulatorForcesStep (class in *pyffi.formats.nif*), 143  
 NifFormat.NiPSSimulatorGeneralStep (class in *pyffi.formats.nif*), 144  
 NifFormat.NiPSSimulatorMeshAlignStep (class in *pyffi.formats.nif*), 144  
 NifFormat.NiPSSimulatorStep (class in *pyffi.formats.nif*), 145  
 NifFormat.NiPSSpawner (class in *pyffi.formats.nif*), 145  
 NifFormat.NiPSSphereEmitter (class in *pyffi.formats.nif*), 145  
 NifFormat.NiPSSphericalCollider (class in *pyffi.formats.nif*), 145  
 NifFormat.NiPSysAgeDeathModifier (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysAirFieldAirFrictionCtrl (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysAirFieldInheritVelocityCtrl (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysAirFieldModifier (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysAirFieldSpreadCtrl (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysBombModifier (class in *pyffi.formats.nif*), 146  
 NifFormat.NiPSysBoundUpdateModifier (class in *pyffi.formats.nif*), 147  
 NifFormat.NiPSysBoxEmitter (class in *pyffi.formats.nif*), 147  
 NifFormat.NiPSysCollider (class in *pyffi.formats.nif*), 147  
 NifFormat.NiPSysColliderManager (class in *pyffi.formats.nif*), 148  
 NifFormat.NiPSysColorModifier (class in *pyffi.formats.nif*), 148  
 NifFormat.NiPSysCylinderEmitter (class in *pyffi.formats.nif*), 148  
 NifFormat.NiPSysData (class in *pyffi.formats.nif*), 148  
 NifFormat.NiPSysDragFieldModifier (class in *pyffi.formats.nif*), 149  
 NifFormat.NiPSysDragModifier (class in *pyffi.formats.nif*), 149  
 NifFormat.NiPSysEmitter (class in *pyffi.formats.nif*), 149  
 NifFormat.NiPSysEmitterCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterCtrlData (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterDeclinationCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterDeclinationVarCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterInitialRadiusCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterLifeSpanCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterPlanarAngleCtrl (class in *pyffi.formats.nif*), 150  
 NifFormat.NiPSysEmitterPlanarAngleVarCtrl (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysEmitterSpeedCtrl (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysFieldAttenuationCtrl (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysFieldMagnitudeCtrl (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysFieldMaxDistanceCtrl (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysFieldModifier (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysGravityFieldModifier (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysGravityModifier (class in *pyffi.formats.nif*), 151  
 NifFormat.NiPSysGravityStrengthCtrl (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysGrowFadeModifier (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysInitialRotAngleCtrl (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysInitialRotAngleVarCtrl (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysInitialRotSpeedCtrl (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysInitialRotSpeedVarCtrl (class in *pyffi.formats.nif*), 152  
 NifFormat.NiPSysMeshEmitter (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysMeshUpdateModifier (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysModifier (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysModifierActiveCtrl (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysModifierBoolCtrl (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysModifierCtrl (class in *pyffi.formats.nif*), 153  
 NifFormat.NiPSysModifierFloatCtrl (class

*in pyffi.formats.nif*), 154  
 NifFormat.NiPSysPlanarCollider (class in *pyffi.formats.nif*), 154  
 NifFormat.NiPSysPositionModifier (class in *pyffi.formats.nif*), 154  
 NifFormat.NiPSysRadialFieldModifier (class in *pyffi.formats.nif*), 154  
 NifFormat.NiPSysResetOnLoopCtrlr (class in *pyffi.formats.nif*), 154  
 NifFormat.NiPSysRotationModifier (class in *pyffi.formats.nif*), 154  
 NifFormat.NiPSysSpawnModifier (class in *pyffi.formats.nif*), 155  
 NifFormat.NiPSysSphereEmitter (class in *pyffi.formats.nif*), 155  
 NifFormat.NiPSysSphericalCollider (class in *pyffi.formats.nif*), 155  
 NifFormat.NiPSysTrailEmitter (class in *pyffi.formats.nif*), 155  
 NifFormat.NiPSysTurbulenceFieldModifier (class in *pyffi.formats.nif*), 156  
 NifFormat.NiPSysUpdateCtrlr (class in *pyffi.formats.nif*), 156  
 NifFormat.NiPSysVolumeEmitter (class in *pyffi.formats.nif*), 156  
 NifFormat.NiPSysVortexFieldModifier (class in *pyffi.formats.nif*), 156  
 NifFormat.NiRangeLODDData (class in *pyffi.formats.nif*), 169  
 NifFormat.NiRawImageData (class in *pyffi.formats.nif*), 170  
 NifFormat.NiRenderObject (class in *pyffi.formats.nif*), 170  
 NifFormat.NiRollController (class in *pyffi.formats.nif*), 170  
 NifFormat.NiRoom (class in *pyffi.formats.nif*), 170  
 NifFormat.NiRoomGroup (class in *pyffi.formats.nif*), 171  
 NifFormat.NiRotatingParticles (class in *pyffi.formats.nif*), 171  
 NifFormat.NiRotatingParticlesData (class in *pyffi.formats.nif*), 171  
 NifFormat.NiScreenElements (class in *pyffi.formats.nif*), 171  
 NifFormat.NiScreenElementsData (class in *pyffi.formats.nif*), 171  
 NifFormat.NiScreenLODDData (class in *pyffi.formats.nif*), 172  
 NifFormat.NiSequence (class in *pyffi.formats.nif*), 172  
 NifFormat.NiSequenceData (class in *pyffi.formats.nif*), 173  
 NifFormat.NiSequenceStreamHelper (class in *pyffi.formats.nif*), 173  
 NifFormat.NiShadeProperty (class in *pyffi.formats.nif*), 173  
 NifFormat.NiShadowGenerator (class in *pyffi.formats.nif*), 173  
 NifFormat.NiSingleInterpController (class in *pyffi.formats.nif*), 173  
 NifFormat.NiSkinData (class in *pyffi.formats.nif*), 173  
 NifFormat.NiSkinInstance (class in *pyffi.formats.nif*), 174  
 NifFormat.NiSkinningLODController (class in *pyffi.formats.nif*), 175  
 NifFormat.NiSkinningMeshModifier (class in *pyffi.formats.nif*), 175  
 NifFormat.NiSkinPartition (class in *pyffi.formats.nif*), 175  
 NifFormat.NiSortAdjustNode (class in *pyffi.formats.nif*), 175  
 NifFormat.NiSourceCubeMap (class in *pyffi.formats.nif*), 175  
 NifFormat.NiSourceTexture (class in *pyffi.formats.nif*), 176  
 NifFormat.NiSpecularProperty (class in *pyffi.formats.nif*), 176  
 NifFormat.NiSphericalCollider (class in *pyffi.formats.nif*), 176  
 NifFormat.NiSpotLight (class in *pyffi.formats.nif*), 177  
 NifFormat.NiStencilProperty (class in *pyffi.formats.nif*), 177  
 NifFormat.NiStringExtraData (class in *pyffi.formats.nif*), 177  
 NifFormat.NiStringPalette (class in *pyffi.formats.nif*), 178  
 NifFormat.NiStringsExtraData (class in *pyffi.formats.nif*), 178  
 NifFormat.NiSwitchNode (class in *pyffi.formats.nif*), 178  
 NifFormat.NiTextKeyExtraData (class in *pyffi.formats.nif*), 178  
 NifFormat.NiTexture (class in *pyffi.formats.nif*), 178  
 NifFormat.NiTextureEffect (class in *pyffi.formats.nif*), 178  
 NifFormat.NiTextureModeProperty (class in *pyffi.formats.nif*), 179  
 NifFormat.NiTextureProperty (class in *pyffi.formats.nif*), 180  
 NifFormat.NiTextureTransformController (class in *pyffi.formats.nif*), 180  
 NifFormat.NiTexturingProperty (class in *pyffi.formats.nif*), 180  
 NifFormat.NiTimeController (class in *pyffi.formats.nif*), 182

NifFormat.NiTransformController (class in *pyffi.formats.nif*), 182  
 NifFormat.NiTransformData (class in *pyffi.formats.nif*), 182  
 NifFormat.NiTransformEvaluator (class in *pyffi.formats.nif*), 182  
 NifFormat.NiTransformInterpolator (class in *pyffi.formats.nif*), 182  
 NifFormat.NiTransparentProperty (class in *pyffi.formats.nif*), 182  
 NifFormat.NiTriBasedGeom (class in *pyffi.formats.nif*), 183  
 NifFormat.NiTriBasedGeomData (class in *pyffi.formats.nif*), 183  
 NifFormat.NiTriShape (class in *pyffi.formats.nif*), 184  
 NifFormat.NiTriShapeData (class in *pyffi.formats.nif*), 184  
 NifFormat.NiTriShapeSkinController (class in *pyffi.formats.nif*), 185  
 NifFormat.NiTriStrips (class in *pyffi.formats.nif*), 185  
 NifFormat.NiTriStripsData (class in *pyffi.formats.nif*), 185  
 NifFormat.NiUVController (class in *pyffi.formats.nif*), 185  
 NifFormat.NiUVData (class in *pyffi.formats.nif*), 185  
 NifFormat.NiVectorExtraData (class in *pyffi.formats.nif*), 186  
 NifFormat.NiVertexColorProperty (class in *pyffi.formats.nif*), 186  
 NifFormat.NiVertWeightsExtraData (class in *pyffi.formats.nif*), 186  
 NifFormat.NiVisController (class in *pyffi.formats.nif*), 186  
 NifFormat.NiVisData (class in *pyffi.formats.nif*), 186  
 NifFormat.NiWireframeProperty (class in *pyffi.formats.nif*), 187  
 NifFormat.NiZBufferProperty (class in *pyffi.formats.nif*), 187  
 NifFormat.NodeGroup (class in *pyffi.formats.nif*), 187  
 NifFormat.OblivionHavokMaterial (class in *pyffi.formats.nif*), 187  
 NifFormat.OblivionLayer (class in *pyffi.formats.nif*), 188  
 NifFormat.OblivionSubShape (class in *pyffi.formats.nif*), 190  
 NifFormat.OldSkinData (class in *pyffi.formats.nif*), 190  
 NifFormat.Particle (class in *pyffi.formats.nif*), 190  
 NifFormat.ParticleDesc (class in *pyffi.formats.nif*), 191  
 NifFormat.physXMaterialRef (class in *pyffi.formats.nif*), 227  
 NifFormat.PixelFormat (class in *pyffi.formats.nif*), 191  
 NifFormat.PixelLayout (class in *pyffi.formats.nif*), 191  
 NifFormat.Polygon (class in *pyffi.formats.nif*), 192  
 NifFormat.PrismaticDescriptor (class in *pyffi.formats.nif*), 192  
 NifFormat.PropagationMode (class in *pyffi.formats.nif*), 193  
 NifFormat.PSLoopBehavior (class in *pyffi.formats.nif*), 190  
 NifFormat.Ptr (class in *pyffi.formats.nif*), 193  
 NifFormat.QTransform (class in *pyffi.formats.nif*), 193  
 NifFormat.Quaternion (class in *pyffi.formats.nif*), 194  
 NifFormat.QuaternionXYZW (class in *pyffi.formats.nif*), 194  
 NifFormat.QuatKey (class in *pyffi.formats.nif*), 194  
 NifFormat.RagdollDescriptor (class in *pyffi.formats.nif*), 194  
 NifFormat.Ref (class in *pyffi.formats.nif*), 194  
 NifFormat.Region (class in *pyffi.formats.nif*), 195  
 NifFormat.RootCollisionNode (class in *pyffi.formats.nif*), 195  
 NifFormat.SemanticData (class in *pyffi.formats.nif*), 196  
 NifFormat.ShaderTexDesc (class in *pyffi.formats.nif*), 196  
 NifFormat.ShortString (class in *pyffi.formats.nif*), 196  
 NifFormat.SizedString (class in *pyffi.formats.nif*), 196  
 NifFormat.SkinData (class in *pyffi.formats.nif*), 197  
 NifFormat.SkinPartition (class in *pyffi.formats.nif*), 197  
 NifFormat.SkinShape (class in *pyffi.formats.nif*), 197  
 NifFormat.SkinShapeGroup (class in *pyffi.formats.nif*), 198  
 NifFormat.SkinTransform (class in *pyffi.formats.nif*), 198  
 NifFormat.SkinWeight (class in *pyffi.formats.nif*), 198  
 NifFormat.SkyObjectType (class in *pyffi.formats.nif*), 198  
 NifFormat.SkyrimHavokMaterial (class in *pyffi.formats.nif*), 199  
 NifFormat.SkyrimLayer (class in

- pyffi.formats.nif*), 200
- NifFormat.SkyrimShaderPropertyFlags1 (class in *pyffi.formats.nif*), 202
- NifFormat.SkyrimShaderPropertyFlags2 (class in *pyffi.formats.nif*), 203
- NifFormat.SkyrimWaterShaderFlags (class in *pyffi.formats.nif*), 204
- NifFormat.SkyShaderProperty (class in *pyffi.formats.nif*), 198
- NifFormat.SolverDeactivation (class in *pyffi.formats.nif*), 204
- NifFormat.SortingMode (class in *pyffi.formats.nif*), 204
- NifFormat.SphereBV (class in *pyffi.formats.nif*), 204
- NifFormat.StencilAction (class in *pyffi.formats.nif*), 204
- NifFormat.StencilCompareMode (class in *pyffi.formats.nif*), 205
- NifFormat.StiffSpringDescriptor (class in *pyffi.formats.nif*), 205
- NifFormat.string (class in *pyffi.formats.nif*), 227
- NifFormat.StringOffset (class in *pyffi.formats.nif*), 205
- NifFormat.StringPalette (class in *pyffi.formats.nif*), 205
- NifFormat.SubConstraint (class in *pyffi.formats.nif*), 207
- NifFormat.SymmetryType (class in *pyffi.formats.nif*), 207
- NifFormat.SyncPoint (class in *pyffi.formats.nif*), 207
- NifFormat.TallGrassShaderProperty (class in *pyffi.formats.nif*), 208
- NifFormat.TargetColor (class in *pyffi.formats.nif*), 208
- NifFormat.TBC (class in *pyffi.formats.nif*), 207
- NifFormat.TexClampMode (class in *pyffi.formats.nif*), 208
- NifFormat.TexCoord (class in *pyffi.formats.nif*), 208
- NifFormat.TexDesc (class in *pyffi.formats.nif*), 208
- NifFormat.TexFilterMode (class in *pyffi.formats.nif*), 209
- NifFormat.TexSource (class in *pyffi.formats.nif*), 209
- NifFormat.TexTransform (class in *pyffi.formats.nif*), 210
- NifFormat.TextureType (class in *pyffi.formats.nif*), 210
- NifFormat.TileShaderProperty (class in *pyffi.formats.nif*), 210
- NifFormat.Triangle (class in *pyffi.formats.nif*), 210
- NifFormat.UnionBV (class in *pyffi.formats.nif*), 211
- NifFormat.Vector3 (class in *pyffi.formats.nif*), 211
- NifFormat.Vector4 (class in *pyffi.formats.nif*), 211
- NifFormat.VelocityType (class in *pyffi.formats.nif*), 212
- NifFormat.VertMode (class in *pyffi.formats.nif*), 212
- NifFormat.VolumetricFogShaderProperty (class in *pyffi.formats.nif*), 212
- NifFormat.WaterShaderProperty (class in *pyffi.formats.nif*), 212
- NifFormat.ZCompareMode (class in *pyffi.formats.nif*), 212
- NIFXMLPATH, 7
- node() (*pyffi.formats.nif.NifFormat.NiPhysXTransformDest* property), 167
- node\_groups() (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 114
- nodes() (*pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier* property), 65
- nodes() (*pyffi.formats.nif.NifFormat.NodeGroup* property), 187
- NONCOLLIDABLE (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- NONCOLLIDABLE (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189
- NONCOLLIDABLE (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201
- None (*pyffi.formats.nif.NifFormat.ExtraVectorsFlags* attribute), 89
- norm() (*pyffi.formats.nif.NifFormat.Vector3* method), 211
- normal() (*pyffi.formats.nif.NifFormat.HalfSpaceBV* property), 94
- normal() (*pyffi.formats.nif.NifFormat.hkTriangle* property), 227
- NORMAL\_MAP (*pyffi.formats.nif.NifFormat.TextureType* attribute), 210
- normal\_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181
- normalize() (*pyffi.formats.nif.NifFormat.TextureCoord* method), 208
- normalize() (*pyffi.formats.nif.NifFormat.Vector3* method), 211
- normalize\_flag() (*pyffi.formats.nif.NifFormat.ElementReference* property), 87
- normalized() (*pyffi.formats.nif.NifFormat.Vector3* method), 211
- normals() (*pyffi.formats.nif.NifFormat.DecalVectorArray* property), 87
- NULL (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- NULL (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189
- NULL (*pyffi.formats.nif.NifFormat.SkyrimLayer* at-

*tribute*), 201  
*num\_1* () (*pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator* property), 104  
*num\_2* () (*pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator* property), 104  
*num\_active* () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161  
*num\_affected\_node\_list\_pointers* () (*pyffi.formats.nif.NifFormat.NiDynamicEffect* property), 119  
*num\_affected\_nodes* () (*pyffi.formats.nif.NifFormat.NiDynamicEffect* property), 119  
*num\_atoms* () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* property), 68  
*num\_bezier\_triangles* () (*pyffi.formats.nif.NifFormat.NiBezierMesh* property), 112  
*num\_big\_tris* () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
*num\_big\_verts* () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
*num\_block\_infos* () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData* property), 68  
*num\_block\_infos* () (*pyffi.formats.nif.NifFormat.NiAdditionalGeometryData* property), 106  
*num\_blocks* () (*pyffi.formats.nif.NifFormat.AdditionalDataBlock* property), 54  
*num\_blocks* () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* property), 68  
*num\_blocks* () (*pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData* property), 68  
*num\_blocks* () (*pyffi.formats.nif.NifFormat.NiAdditionalGeometryData* property), 107  
*num\_bones* () (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* property), 223  
*num\_bones* () (*pyffi.formats.nif.NifFormat.NiFurSpringControllerData* property), 121  
*num\_bones* () (*pyffi.formats.nif.NifFormat.NiSkinInstancemesh* property), 175  
*num\_bones* () (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier* property), 175  
*num\_bones* () (*pyffi.formats.nif.NifFormat.NiTriShapeSkinController* property), 185  
*num\_bones\_1* () (*pyffi.formats.nif.NifFormat.BSTreeNode* property), 75  
*num\_bones\_2* () (*pyffi.formats.nif.NifFormat.BSTreeNode* property), 75  
*num\_bones\_2* () (*pyffi.formats.nif.NifFormat.NiFurSpringController* property), 121  
*num\_buttons* () (*pyffi.formats.nif.NifFormat.FxRadioButton* property), 94  
*num\_bv* () (*pyffi.formats.nif.NifFormat.UnionBV* property), 211  
*num\_bytes* () (*pyffi.formats.nif.NifFormat.NiDataStream* property), 118  
*num\_bytes* () (*pyffi.formats.nif.NifFormat.NiVertWeightsExtraData* property), 186  
*num\_channel\_bytes* () (*pyffi.formats.nif.NifFormat.AdditionalDataInfo* property), 54  
*num\_channel\_bytes\_per\_element* () (*pyffi.formats.nif.NifFormat.AdditionalDataInfo* property), 54  
*num\_children* () (*pyffi.formats.nif.NifFormat.NiEnvMappedTriShape* property), 119  
*num\_chunks* () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
*num\_colliders* () (*pyffi.formats.nif.NifFormat.NiPSSimulatorColliders* property), 143  
*num\_colliders* () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneral* property), 144  
*num\_controls* () (*pyffi.formats.nif.NifFormat.NiMeshModifier* property), 129  
*num\_controls* () (*pyffi.formats.nif.NifFormat.MeshData* property), 101  
*num\_components* () (*pyffi.formats.nif.NifFormat.NiDataStream* property), 118  
*num\_control\_points* () (*pyffi.formats.nif.NifFormat.NiBSplineBasisData* property), 109  
*num\_controlled\_blocks* () (*pyffi.formats.nif.NifFormat.NiSequence* property), 117  
*num\_controller\_sequences* () (*pyffi.formats.nif.NifFormat.NiControllerManager* property), 117  
*num\_data* () (*pyffi.formats.nif.NifFormat.AdditionalDataBlock* property), 54  
*num\_data* () (*pyffi.formats.nif.NifFormat.NiMesh* property), 128  
*num\_dests* () (*pyffi.formats.nif.NifFormat.NiPhysXProp* property), 165  
*num\_descs* () (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc* property), 166  
*num\_elements* () (*pyffi.formats.nif.NifFormat.NiMorphMeshModifier* property), 130  
*num\_emitter\_meshes* () (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter* property), 153  
*num\_entities* () (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207  
*num\_entries* () (*pyffi.formats.nif.NifFormat.NiPalette* property), 156  
*num\_extra\_bytes* ()

(*pyffi.formats.nif.NifFormat.NiFloatExtraDataController* property), 170  
 (*pyffi.formats.nif.NifFormat.NiMultiTargetTransformController* property), 120  
 num\_extra\_targets() (*pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRenderData* property), 131  
 num\_faces() (*pyffi.formats.nif.NifFormat.NiPixelData* property), 163  
 num\_faces() (*pyffi.formats.nif.NifFormat.NiParticleSystemModifier* property), 167  
 num\_floats() (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain* property), 213  
 num\_floats() (*pyffi.formats.nif.NifFormat.BSPSysScaleModifier* property), 66  
 num\_floats() (*pyffi.formats.nif.NifFormat.NiFloatsExtraData* property), 121  
 num\_forces() (*pyffi.formats.nif.NifFormat.NiPSSimulatorForcesStep* property), 144  
 num\_in\_portals() (*pyffi.formats.nif.NifFormat.NiRoom* property), 170  
 num\_indices() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* property), 214  
 num\_indices() (*pyffi.formats.nif.NifFormat.Region* property), 195  
 num\_indices\_2() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* property), 214  
 num\_integers() (*pyffi.formats.nif.NifFormat.NiIntegersExtraData* property), 126  
 num\_interpolators() (*pyffi.formats.nif.NifFormat.NiGeomMorpherController* property), 122  
 num\_interpolators() (*pyffi.formats.nif.NifFormat.NiMorphWeightsController* property), 130  
 num\_items() (*pyffi.formats.nif.NifFormat.BSWArray* property), 76  
 num\_items() (*pyffi.formats.nif.NifFormat.NiRoom* property), 170  
 num\_joints() (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc* property), 166  
 num\_keys() (*pyffi.formats.nif.NifFormat.KeyGroup* property), 97  
 num\_keys() (*pyffi.formats.nif.NifFormat.Morph* property), 102  
 num\_keys() (*pyffi.formats.nif.NifFormat.NiVisData* property), 187  
 num\_link\_pairs() (*pyffi.formats.nif.NifFormat.SkinShapeGroup* property), 198  
 num\_links() (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain* property), 213  
 num\_links\_2() (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain* property), 213  
 num\_lod\_levels() (*pyffi.formats.nif.NifFormat.NiLODNode* property), 126  
 num\_lod\_levels() (*pyffi.formats.nif.NifFormat.NiRangeLODData* property), 170  
 num\_materials() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshSho* property), 217  
 num\_materials() (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc* property), 166  
 num\_meshes() (*pyffi.formats.nif.NifFormat.NiRenderObject* property), 170  
 num\_meshes() (*pyffi.formats.nif.NifFormat.NiPSysMeshUpdateModifier* property), 153  
 num\_modifiers() (*pyffi.formats.nif.NifFormat.NiMesh* property), 128  
 num\_modifiers() (*pyffi.formats.nif.NifFormat.NiParticleSystem* property), 158  
 num\_node\_groups() (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 114  
 num\_node\_groups\_2() (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 114  
 num\_nodes() (*pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier* property), 65  
 num\_nodes() (*pyffi.formats.nif.NifFormat.NodeGroup* property), 187  
 num\_objects() (*pyffi.formats.nif.NifFormat.NiDefaultAVObjectPalette* property), 118  
 num\_particle\_meshes() (*pyffi.formats.nif.NifFormat.NiParticleMeshModifier* property), 157  
 num\_particle\_systems() (*pyffi.formats.nif.NifFormat.BSMasterParticleSystem* property), 63  
 num\_particles() (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161  
 num\_particles() (*pyffi.formats.nif.NifFormat.NiParticleSystemControl* property), 159  
 num\_pixels() (*pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRende* property), 163  
 num\_pixels() (*pyffi.formats.nif.NifFormat.NiPixelData* property), 167  
 num\_polygons() (*pyffi.formats.nif.NifFormat.NiScreenElementsData* property), 171  
 num\_portals\_2() (*pyffi.formats.nif.NifFormat.NiRoom* property), 171  
 num\_positions() (*pyffi.formats.nif.NifFormat.BSFurnitureMarker* property), 60  
 num\_regions() (*pyffi.formats.nif.NifFormat.NiDataStream* property), 118  
 num\_rooms() (*pyffi.formats.nif.NifFormat.NiRoomGroup* property), 171  
 num\_rotation\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 144  
 num\_rotation\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep* property), 144

*property*), 144  
 num\_segments() (*pyffi.formats.nif.NifFormat.BSSegmentedTriShape* *property*), 70  
 num\_shader\_textures() (*pyffi.formats.nif.NifFormat.NiTexturingProperty* *property*), 181  
 num\_shape\_groups() (*pyffi.formats.nif.NifFormat.NiBoneLODController* *property*), 114  
 num\_shape\_groups\_2() (*pyffi.formats.nif.NifFormat.NiBoneLODController* *property*), 114  
 num\_simulation\_steps() (*pyffi.formats.nif.NifFormat.NiPSSimulator* *property*), 143  
 num\_size\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* *property*), 144  
 num\_skin\_partition\_blocks() (*pyffi.formats.nif.NifFormat.NiSkinPartition* *property*), 175  
 num\_sources() (*pyffi.formats.nif.NifFormat.NiFlipController* *property*), 120  
 num\_spawn\_generations() (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier* *property*), 155  
 num\_strings() (*pyffi.formats.nif.NifFormat.NiStringsExtraData* *property*), 178  
 num\_strips() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* *property*), 215  
 num\_strips\_data() (*pyffi.formats.nif.NifFormat.bhkMeshShape* *property*), 219  
 num\_sub\_shapes() (*pyffi.formats.nif.NifFormat.bhkConvexListShape* *property*), 218  
 num\_submeshes() (*pyffi.formats.nif.NifFormat.MeshData* *property*), 101  
 num\_submeshes() (*pyffi.formats.nif.NifFormat.NiMesh* *property*), 128  
 num\_submit\_points() (*pyffi.formats.nif.NifFormat.NiMeshModifier* *property*), 129  
 num\_subtexture\_offset\_u\_vs() (*pyffi.formats.nif.NifFormat.NiPSysData* *property*), 148  
 num\_targets() (*pyffi.formats.nif.NifFormat.NiMorphMeshModifier* *property*), 130  
 num\_targets() (*pyffi.formats.nif.NifFormat.NiMorphWeightsController* *property*), 130  
 num\_text\_keys() (*pyffi.formats.nif.NifFormat.NiTextKeyExtraData* *property*), 178  
 num\_textures() (*pyffi.formats.nif.NifFormat.BSShaderTextureSet* *property*), 73  
 num\_textures() (*pyffi.formats.nif.NifFormat.NiArkTextureExtraData* *property*), 108  
 num\_total\_bytes() (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* *property*), 68  
 num\_total\_bytes\_per\_element() (*pyffi.formats.nif.NifFormat.AdditionalDataInfo* *property*), 54  
 num\_total\_bytes\_per\_element() (*pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock* *property*), 68  
 num\_transforms() (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape* *property*), 217  
 num\_tread\_transforms() (*pyffi.formats.nif.NifFormat.BSTreadTransfInterpolator* *property*), 74  
 num\_triangles() (*pyffi.formats.nif.NifFormat.Polygon* *property*), 192  
 num\_unknown\_bytes\_2() (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData* *property*), 113  
 num\_unknown\_floats() (*pyffi.formats.nif.NifFormat.bhkMeshShape* *property*), 219  
 num\_unknown\_ints() (*pyffi.formats.nif.NifFormat.NiGeomMorpherController* *property*), 122  
 num\_unknown\_ints\_1() (*pyffi.formats.nif.NifFormat.NiMeshPSysData* *property*), 129  
 num\_unknown\_links\_1() (*pyffi.formats.nif.NifFormat.NiShadowGenerator* *property*), 173  
 num\_unknown\_vectors() (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData* *property*), 113  
 num\_uv\_quadrants() (*pyffi.formats.nif.NifFormat.NiParticlesData* *property*), 161  
 num\_valid() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* *property*), 159  
 num\_vector\_blocks() (*pyffi.formats.nif.NifFormat.BSDecalPlacementVectorExtraData* *property*), 56  
 num\_vectors() (*pyffi.formats.nif.NifFormat.DecalVectorArray* *property*), 87  
 num\_vectors() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* *property*), 215  
 num\_vectors() (*pyffi.formats.nif.NifFormat.BSPackedAdditionalGeom* *property*), 68  
 num\_vectors() (*pyffi.formats.nif.NifFormat.MatchGroup* *property*), 99  
 num\_vectors() (*pyffi.formats.nif.NifFormat.NiAdditionalGeometryData* *property*), 107  
 num\_vectors() (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* *property*), 164

num\_vertices() (*pyffi.formats.nif.NifFormat.NiPortal property*), 169

num\_vertices() (*pyffi.formats.nif.NifFormat.NiVertWeightsExtraData attribute*), 91

num\_vertices() (*pyffi.formats.nif.NifFormat.OblivionSubShape property*), 190

num\_vertices() (*pyffi.formats.nif.NifFormat.Polygon property*), 192

num\_visibility\_keys() (*pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlData property*), 150

num\_walls() (*pyffi.formats.nif.NifFormat.NiRoom property*), 171

number() (*pyffi.formats.nif.NifFormat.physXMaterialRef property*), 227

**O**

object\_palette() (*pyffi.formats.nif.NifFormat.NiControllerManager property*), 117

objs() (*pyffi.formats.nif.NifFormat.NiDefaultAVObjectPalette property*), 118

offset() (*pyffi.formats.cgf.CgfFormat.BoneLink property*), 11

offset() (*pyffi.formats.cgf.CgfFormat.ChunkHeader property*), 12

offset() (*pyffi.formats.cgf.CgfFormat.Header property*), 15

offset() (*pyffi.formats.nif.NifFormat.FurniturePosition property*), 93

offset() (*pyffi.formats.nif.NifFormat.MipMap property*), 101

offset() (*pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator property*), 109

old\_emit\_rate() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 148

old\_speed() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159

only\_regexs (*pyffi.spells.Toaster attribute*), 270

openfile() (*pyffi.object\_models.MetaFileFormat static method*), 271

operation() (*pyffi.formats.nif.NifFormat.NiTextureTransformController property*), 180

options (*pyffi.spells.Toaster attribute*), 270

order() (*pyffi.formats.nif.NifFormat.NiPSysModifier property*), 153

orientation() (*pyffi.formats.nif.NifFormat.FurniturePosition property*), 93

origin() (*pyffi.formats.nif.NifFormat.CapsuleBV property*), 80

OTHER (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91

OTHER (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189

**P**

PACK (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91

palette() (*pyffi.formats.nif.NifFormat.NiPalette property*), 157

palette() (*pyffi.formats.nif.NifFormat.NiStringPalette property*), 178

parallax\_envmap\_strength() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 62

parallax\_inner\_layer\_texture\_scale() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 62

parallax\_inner\_layer\_thickness() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 62

parallax\_refraction\_scale() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 62

parent() (*pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator property*), 104

parent() (*pyffi.formats.nif.NifFormat.NiPSysCollider property*), 147

parent() (*pyffi.formats.nif.NifFormat.NiPSysDragModifier property*), 149

parse\_inifile() (*pyffi.spells.Toaster static method*), 271

parse\_mopp() (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method*), 220

part\_flag() (*pyffi.formats.nif.NifFormat.BodyPartList property*), 77

part\_descriptions() (*pyffi.formats.nif.NifFormat.NiPSysData property*), 148

particle\_extra() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159

particle\_lifetime() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159

particle\_link() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159

particle\_meshes() (*pyffi.formats.nif.NifFormat.NiParticleMeshModifier property*), 158

particle\_radius() (*pyffi.formats.nif.NifFormat.NiParticlesData property*), 161

particle\_systems() (*pyffi.formats.nif.NifFormat.BSMasterParticleSystem property*), 63

particle\_timestamp() (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 159

particle\_unknown\_short()

(*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 205  
 (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 159  
 particle\_unknown\_vector() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 160  
 particle\_velocity() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 160  
 particle\_vertex\_id() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 160  
 particles() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 160  
 pass\_action() (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 177  
 PATH\_PICK (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
 PATH\_PICK (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 PATH\_PICK (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201  
 percentage() (*pyffi.formats.nif.NifFormat.NiPSysDragModifier* property), 149  
 percentage\_spawned() (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier* property), 155  
 perp\_2\_axle\_in\_a\_1() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 perp\_2\_axle\_in\_a\_2() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 perp\_2\_axle\_in\_b\_1() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 perp\_2\_axle\_in\_b\_2() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 persist\_render\_data() (*pyffi.formats.nif.NifFormat.NiSourceTexture* property), 176  
 pf\_editor\_visible() (*pyffi.formats.nif.NifFormat.BSPartFlag* property), 68  
 pf\_start\_net\_boneset() (*pyffi.formats.nif.NifFormat.BSPartFlag* property), 68  
 phase() (*pyffi.formats.nif.NifFormat.NiTimeController* property), 182  
 pivot\_a() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 pivot\_a() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192  
 pivot\_a() (*pyffi.formats.nif.NifFormat.StiffSpringDescriptor* property), 114  
 pivot\_b() (*pyffi.formats.nif.NifFormat.HingeDescriptor* property), 96  
 pivot\_b() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192  
 pivot\_b() (*pyffi.formats.nif.NifFormat.StiffSpringDescriptor* property), 114  
 PIX\_LAY\_BUMPMAP (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 191  
 PIX\_LAY\_COMPRESSED (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 PIX\_LAY\_DEFAULT (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 PIX\_LAY\_HIGH\_COLOR\_16 (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 PIX\_LAY\_PALETTISED (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 PIX\_LAY\_PALETTISED\_4 (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 PIX\_LAY\_TRUE\_COLOR\_32 (*pyffi.formats.nif.NifFormat.PixelLayout* attribute), 192  
 pixel\_data() (*pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRender* property), 163  
 pixel\_data() (*pyffi.formats.nif.NifFormat.NiPixelData* property), 167  
 pixel\_data() (*pyffi.formats.nif.NifFormat.NiSourceTexture* property), 176  
 pixel\_data() (*pyffi.formats.nif.NifFormat.TexSource* property), 209  
 pixel\_layout() (*pyffi.formats.nif.NifFormat.NiSourceTexture* property), 176  
 PixelData (*pyffi.formats.dds.DdsFormat* attribute), 36  
 PixelData (*pyffi.formats.tga.TgaFormat* attribute), 235  
 planar\_angle() (*pyffi.formats.nif.NifFormat.NiPSysEmitter* property), 150  
 planar\_angle\_variation() (*pyffi.formats.nif.NifFormat.NiPSysEmitter* property), 150  
 PLANAR\_SYMMETRY (*pyffi.formats.nif.NifFormat.SymmetryType* attribute), 207  
 plane\_a() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192  
 plane\_b() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192  
 point\_3\_value() (*pyffi.formats.nif.NifFormat.NiPoint3Interpolator* property), 169  
 point\_value() (*pyffi.formats.nif.NifFormat.NiBlendPoint3Interpolator* property), 114

points() (*pyffi.formats.nif.NifFormat.DecalVectorArray* property), 87  
 points\_1() (*pyffi.formats.nif.NifFormat.NiBezierMesh* property), 112  
 points\_2() (*pyffi.formats.nif.NifFormat.NiBezierMesh* property), 112  
 polygon\_indices() (*pyffi.formats.nif.NifFormat.NiScreenElementsData* property), 171  
 polygons() (*pyffi.formats.nif.NifFormat.NiScreenElementsData* property), 171  
 PONYTAIL (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 PONYTAIL (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
 PORTAL (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 PORTAL (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
 PORTAL (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201  
 portals\_2() (*pyffi.formats.nif.NifFormat.NiRoom* property), 171  
 pos\_data() (*pyffi.formats.nif.NifFormat.NiPathController* property), 162  
 pos\_data() (*pyffi.formats.nif.NifFormat.NiPathInterpolator* property), 162  
 position() (*pyffi.formats.nif.NifFormat.BSMultiBoundAABB* property), 64  
 position() (*pyffi.formats.nif.NifFormat.NiGravity* property), 125  
 position() (*pyffi.formats.nif.NifFormat.NiParticleBomb* property), 157  
 position\_ref\_1() (*pyffi.formats.nif.NifFormat.FurniturePosition* property), 93  
 position\_ref\_2() (*pyffi.formats.nif.NifFormat.FurniturePosition* property), 93  
 positions() (*pyffi.formats.nif.NifFormat.BSFurnitureMarker2* property), 60  
 primitive\_type() (*pyffi.formats.nif.NifFormat.NiMesh* property), 128  
 priority() (*pyffi.formats.nif.NifFormat.bhkRDTConstraint* property), 222  
 priority() (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint* property), 223  
 priority() (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207  
 Prismatic (*pyffi.formats.nif.NifFormat.hkConstraintTypes* attribute), 226  
 prismatic() (*pyffi.formats.nif.NifFormat.bhkPrismaticConstraint* property), 222  
 prismatic() (*pyffi.formats.nif.NifFormat.SubConstraint* property), 207  
 PROJECTILE (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 PROJECTILE (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
 PROJECTILE (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201  
 PROJECTILEZONE (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 PROJECTILEZONE (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201  
 ps\_2\_description() (*pyffi.formats.nif.NifFormat.NiPhysXProp* property), 165  
 PROPAGATE\_ALWAYS (*pyffi.formats.nif.NifFormat.PropagationMode* attribute), 193  
 PROPAGATE\_NEVER (*pyffi.formats.nif.NifFormat.PropagationMode* attribute), 193  
 PROPAGATE\_ON\_FAILURE (*pyffi.formats.nif.NifFormat.PropagationMode* attribute), 193  
 PROPAGATE\_ON\_SUCCESS (*pyffi.formats.nif.NifFormat.PropagationMode* attribute), 193  
 propagation\_mode() (*pyffi.formats.nif.NifFormat.NiCollisionData* property), 117  
 proportion\_count() (*pyffi.formats.nif.NifFormat.NiScreenLODData* property), 172  
 proportion\_levels() (*pyffi.formats.nif.NifFormat.NiScreenLODData* property), 172  
 PROPS (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
 ps\_2\_position() (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
 ps\_2\_position() (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201  
 ps\_2\_k() (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104  
 ps\_2\_k() (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
 ps\_2\_k() (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 ps\_2\_k() (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.MultiTextureElement* property), 104  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
 ps\_2\_1() (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 PS\_LOOP\_AGESCALE (*pyffi.formats.nif.NifFormat.PSLoopBehavior* property), 209

- attribute*), 190
- PS\_LOOP\_CLAMP\_BIRTH (*pyffi.formats.nif.NifFormat.PSLoopBehavior attribute*), 190
- PS\_LOOP\_CLAMP\_DEATH (*pyffi.formats.nif.NifFormat.PSLoopBehavior attribute*), 190
- PS\_LOOP\_LOOP (*pyffi.formats.nif.NifFormat.PSLoopBehavior attribute*), 190
- PS\_LOOP\_REFLECT (*pyffi.formats.nif.NifFormat.PSLoopBehavior attribute*), 190
- PX\_FMT\_DXT1 (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PX\_FMT\_DXT5 (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PX\_FMT\_DXT5\_ALT (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PX\_FMT\_PAL8 (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PX\_FMT\_RGB8 (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PX\_FMT\_RGBA8 (*pyffi.formats.nif.NifFormat.PixelFormat attribute*), 191
- PyFFI, 304
- pyffi (*module*), 7
- pyffi.formats (*module*), 8
- pyffi.formats.bsa (*module*), 8
- pyffi.formats.cgf (*module*), 11
- pyffi.formats.dae (*module*), 32
- pyffi.formats.dds (*module*), 34
- pyffi.formats.egm (*module*), 37
- pyffi.formats.egt (*module*), 41
- pyffi.formats.esp (*module*), 44
- pyffi.formats.kfm (*module*), 48
- pyffi.formats.nif (*module*), 53
- pyffi.formats.tga (*module*), 233
- pyffi.formats.tri (*module*), 236
- pyffi.object\_models (*module*), 271
- pyffi.spells (*module*), 242
- pyffi.spells.cgf (*module*), 243
- pyffi.spells.dds (*module*), 243
- pyffi.spells.kfm (*module*), 243
- pyffi.spells.nif (*module*), 243
- pyffi.spells.nif.check (*module*), 243
- pyffi.spells.nif.dump (*module*), 243
- pyffi.spells.nif.fix (*module*), 243
- pyffi.spells.nif.modify (*module*), 254
- pyffi.spells.nif.optimize (*module*), 251
- pyffi.spells.tga (*module*), 264
- Q**
- quadratic\_attenuation () (*pyffi.formats.nif.NifFormat.NiPointLight property*), 169
- QUADRATIC\_KEY (*pyffi.formats.nif.NifFormat.KeyType attribute*), 97
- QUIVER (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- QUIVER (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R**
- r () (*pyffi.formats.nif.NifFormat.ByteColor3 property*), 79
- r () (*pyffi.formats.nif.NifFormat.ByteColor4 property*), 79
- r () (*pyffi.formats.nif.NifFormat.Color3 property*), 81
- r () (*pyffi.formats.nif.NifFormat.Color4 property*), 81
- R\_CALF (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_CALF (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R\_FOOT (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_FOOT (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R\_FORE\_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_FOREARM (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R\_HAND (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_HAND (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R\_THIGH (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_THIGH (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- R\_UPPER\_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 91
- R\_UPPER\_ARM (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 189
- radial\_type () (*pyffi.formats.nif.NifFormat.NiPSysRadialFieldModifier property*), 154
- radii () (*pyffi.formats.nif.NifFormat.NiParticlesData property*), 161
- radius () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property*), 216
- radius () (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property*), 223
- radius () (*pyffi.formats.nif.NifFormat.bhkSphereRepShape property*), 225
- radius () (*pyffi.formats.nif.NifFormat.BoundingBox property*), 77
- radius () (*pyffi.formats.nif.NifFormat.BSMultiBoundSphere property*), 64
- radius () (*pyffi.formats.nif.NifFormat.NiPSysCylinderEmitter property*), 148

radius () ( <i>pyffi.formats.nif.NifFormat.NiPSysSphereEmitter</i> property), 155	read () ( <i>pyffi.formats.egm.EgmFormat.FileSignature</i> method), 39
radius () ( <i>pyffi.formats.nif.NifFormat.NiPSysSphericalCollision</i> property), 155	read () ( <i>pyffi.formats.egm.EgmFormat.FileVersion</i> method), 39
radius () ( <i>pyffi.formats.nif.NifFormat.SphereBV</i> property), 204	read () ( <i>pyffi.formats.egt.EgtFormat.FileSignature</i> method), 42
radius_variation () ( <i>pyffi.formats.nif.NifFormat.NiPSysEmitter</i> property), 150	read () ( <i>pyffi.formats.egt.EgtFormat.FileVersion</i> method), 42
Ragdoll ( <i>pyffi.formats.nif.NifFormat.hkConstraintType</i> attribute), 226	read () ( <i>pyffi.formats.egt.EgtFormat.Header</i> method), 43
ragdoll () ( <i>pyffi.formats.nif.NifFormat.bhkRDTConstraint</i> property), 222	read () ( <i>pyffi.formats.esp.EspFormat.Data</i> method), 45
ragdoll () ( <i>pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint</i> property), 223	read () ( <i>pyffi.formats.esp.EspFormat.GRUP</i> method), 45
ragdoll () ( <i>pyffi.formats.nif.NifFormat.SubConstraint</i> property), 207	read () ( <i>pyffi.formats.esp.EspFormat.Record</i> method), 46
random_initial_axis () ( <i>pyffi.formats.nif.NifFormat.NiParticleRotation</i> property), 158	read () ( <i>pyffi.formats.esp.EspFormat.ZString</i> method), 47
random_initial_axis () ( <i>pyffi.formats.nif.NifFormat.NiPSysRotationModifier</i> property), 155	read () ( <i>pyffi.formats.kfm.KfmFormat.Data</i> method), 49
random_rot_speed_sign () ( <i>pyffi.formats.nif.NifFormat.NiPSysRotationModifier</i> property), 155	read () ( <i>pyffi.formats.kfm.KfmFormat.HeaderString</i> method), 49
range () ( <i>pyffi.formats.nif.NifFormat.NiPSysDragModifier</i> property), 149	read () ( <i>pyffi.formats.kfm.KfmFormat.SizedString</i> method), 50
range_falloff () ( <i>pyffi.formats.nif.NifFormat.NiPSysDragModifier</i> property), 149	read () ( <i>pyffi.formats.nif.NifFormat.bool</i> method), 226
RE_FILENAME ( <i>pyffi.formats.nif.NifFormat</i> attribute), 194	read () ( <i>pyffi.formats.nif.NifFormat.ByteArray</i> method), 78
RE_FILENAME ( <i>pyffi.object_models.FileFormat</i> attribute), 272	read () ( <i>pyffi.formats.nif.NifFormat.ByteMatrix</i> method), 79
read () ( <i>pyffi.formats.bsa.BsaFormat.BZString</i> method), 8	read () ( <i>pyffi.formats.nif.NifFormat.Data</i> method), 86
read () ( <i>pyffi.formats.bsa.BsaFormat.FileVersion</i> method), 8	read () ( <i>pyffi.formats.nif.NifFormat.FileVersion</i> method), 92
read () ( <i>pyffi.formats.bsa.BsaFormat.Header</i> method), 9	read () ( <i>pyffi.formats.nif.NifFormat.Footer</i> method), 92
read () ( <i>pyffi.formats.bsa.BsaFormat.ZString</i> method), 9	read () ( <i>pyffi.formats.nif.NifFormat.HeaderString</i> method), 95
read () ( <i>pyffi.formats.cgf.CgfFormat.Data</i> method), 13	read () ( <i>pyffi.formats.nif.NifFormat.LineString</i> method), 98
read () ( <i>pyffi.formats.cgf.CgfFormat.FileSignature</i> method), 14	read () ( <i>pyffi.formats.nif.NifFormat.Ref</i> method), 195
read () ( <i>pyffi.formats.cgf.CgfFormat.Ref</i> method), 27	read () ( <i>pyffi.formats.nif.NifFormat.ShortString</i> method), 196
read () ( <i>pyffi.formats.cgf.CgfFormat.SizedString</i> method), 28	read () ( <i>pyffi.formats.nif.NifFormat.SizedString</i> method), 197
read () ( <i>pyffi.formats.dae.DaeFormat.Data</i> method), 33	read () ( <i>pyffi.formats.nif.NifFormat.string</i> method), 227
read () ( <i>pyffi.formats.dds.DdsFormat.Data</i> method), 35	read () ( <i>pyffi.formats.tga.TgaFormat.Data</i> method), 234
read () ( <i>pyffi.formats.dds.DdsFormat.HeaderString</i> method), 35	read () ( <i>pyffi.formats.tga.TgaFormat.FooterString</i> method), 234
read () ( <i>pyffi.formats.egm.EgmFormat.Data</i> method), 38	read () ( <i>pyffi.formats.tri.TriFormat.FileSignature</i> method), 237
	read () ( <i>pyffi.formats.tri.TriFormat.FileVersion</i> method), 237
	read () ( <i>pyffi.formats.tri.TriFormat.Header</i> method), 238
	read () ( <i>pyffi.formats.tri.TriFormat.SizedStringZ</i> method), 239
	read () ( <i>pyffi.object_models.FileFormat.Data</i> method),

- 272
- READONLY (*pyffi.spells.Spell* attribute), 264
- recurse() (*pyffi.spells.Spell* method), 266
- recurse() (*pyffi.spells.SpellGroupSeriesBase* method), 269
- refraction\_fire\_period() (*pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty* property), 73
- refraction\_strength() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62
- refraction\_strength() (*pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty* property), 73
- regions() (*pyffi.formats.nif.NifFormat.NiDataStream* property), 118
- remove\_child() (*pyffi.formats.nif.NifFormat.NiNode* method), 133
- remove\_effect() (*pyffi.formats.nif.NifFormat.NiNode* method), 133
- remove\_extra\_data() (*pyffi.formats.nif.NifFormat.NiObjectNET* method), 135
- remove\_if\_broken() (*pyffi.formats.nif.NifFormat.bhkBreakableConstraint* property), 214
- remove\_property() (*pyffi.formats.nif.NifFormat.NiAVObject* method), 106
- remove\_shape() (*pyffi.formats.nif.NifFormat.bhkListShape* method), 219
- replace\_global\_node() (*pyffi.formats.cgf.CgfFormat.Data* method), 13
- replace\_global\_node() (*pyffi.formats.nif.NifFormat.Data* method), 86
- replace\_global\_node() (*pyffi.formats.nif.NifFormat.Ptr* method), 193
- replace\_global\_node() (*pyffi.formats.nif.NifFormat.Ref* method), 195
- reserved\_bits\_0() (*pyffi.formats.nif.NifFormat.BSSegmentFlags* property), 70
- reserved\_bits\_1() (*pyffi.formats.nif.NifFormat.BSPartFlag* property), 68
- RESPONSE\_INVALID (*pyffi.formats.nif.NifFormat.hkResponseType* attribute), 226
- RESPONSE\_NONE (*pyffi.formats.nif.NifFormat.hkResponseType* attribute), 226
- RESPONSE\_REPORTING (*pyffi.formats.nif.NifFormat.hkResponseType* attribute), 226
- RESPONSE\_SIMPLE\_CONTACT (*pyffi.formats.nif.NifFormat.hkResponseType* attribute), 227
- restitution() (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* property), 224
- rotation() (*pyffi.formats.nif.NifFormat.ImageType* attribute), 96
- rotation() (*pyffi.formats.nif.NifFormat.NiRawImageData* property), 170
- RGBA (*pyffi.formats.nif.NifFormat.ImageType* attribute), 96
- rotation\_image\_data() (*pyffi.formats.nif.NifFormat.NiRawImageData* property), 170
- right() (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints* property), 93
- right\_eye\_reflection\_center() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62
- RIGID\_FACE\_CAMERA (*pyffi.formats.nif.NifFormat.BillboardMode* attribute), 77
- RIGID\_FACE\_CENTER (*pyffi.formats.nif.NifFormat.BillboardMode* attribute), 77
- rooms() (*pyffi.formats.nif.NifFormat.NiRoomGroup* property), 171
- root() (*pyffi.formats.nif.NifFormat.CStreamableAssetData* property), 79
- ROTATE\_ABOUT\_UP (*pyffi.formats.nif.NifFormat.BillboardMode* attribute), 77
- ROTATE\_ABOUT\_UP2 (*pyffi.formats.nif.NifFormat.BillboardMode* attribute), 77
- rotation() (*pyffi.formats.nif.NifFormat.bhkCMSDTransform* property), 215
- rotation() (*pyffi.formats.nif.NifFormat.BoundingBox* property), 77
- rotation() (*pyffi.formats.nif.NifFormat.BSMultiBoundOBB* property), 64
- rotation() (*pyffi.formats.nif.NifFormat.BSTreadTransformData* property), 75
- rotation() (*pyffi.formats.nif.NifFormat.MTransform* property), 99
- rotation() (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 127
- rotation() (*pyffi.formats.nif.NifFormat.QTransform* property), 193
- rotation\_a() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 192
- rotation\_angles() (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161
- rotation\_axes() (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 161

*property*), 161  
 rotation\_b() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* *attribute*), 58  
*property*), 192  
 rotation\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep*), 58  
*property*), 144  
 rotation\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep*), 58  
*property*), 145  
 rotation\_loop\_behavior() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* *property*), 144  
 rotation\_loop\_behavior() (*pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep* *attribute*), 58  
*property*), 145  
 rotation\_matrix\_a() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* *property*), 192  
 rotation\_speed() (*pyffi.formats.nif.NifFormat.NiParticleRotation* *property*), 158  
 rotation\_x() (*pyffi.formats.nif.NifFormat.BSInvMarkerSBP\_42\_CIRCLET* *property*), 61  
 rotation\_y() (*pyffi.formats.nif.NifFormat.BSInvMarkerSBP\_43\_EARS* *property*), 61  
 rotation\_z() (*pyffi.formats.nif.NifFormat.BSInvMarkerSBP\_44\_DRAGON\_BLOODHEAD\_OR\_MOD\_MOUTH* *property*), 61  
 rotations() (*pyffi.formats.nif.NifFormat.NiParticlesData* *property*), 161  
 rotations\_2() (*pyffi.formats.nif.NifFormat.NiRotatingParticlesData* *property*), 171

**S**

save\_as\_dds() (*pyffi.formats.nif.NifFormat.ATextureRenderData* *method*), 53  
 SBP\_130\_HEAD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_131\_HAIR (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_141\_LONGHAIR (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_142\_CIRCLET (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_143\_EARS (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_150\_DECAPITATEDHEAD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_230\_HEAD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_30\_HEAD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_31\_HAIR (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_32\_BODY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_33\_HANDS (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_34\_FOREARMS (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_35\_AMULET (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_36\_RING (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_37\_FEET (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_38\_CALVES (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_39\_SHIELD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_40\_TAIL (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_41\_LONGHAIR (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_42\_CIRCLET (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_43\_EARS (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_44\_DRAGON\_BLOODHEAD\_OR\_MOD\_MOUTH (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_45\_DRAGON\_BLOODWINGL\_OR\_MOD\_NECK (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 58  
 SBP\_46\_DRAGON\_BLOODWINGR\_OR\_MOD\_CHEST\_PRIMARY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_47\_DRAGON\_BLOODTAIL\_OR\_MOD\_BACK (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_48\_MOD\_MISC1 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_49\_MOD\_PELVIS\_PRIMARY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_50\_DECAPITATEDHEAD (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_51\_DECAPITATE (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_52\_MOD\_PELVIS\_SECONDARY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_53\_MOD\_LEG\_RIGHT (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59  
 SBP\_54\_MOD\_LEG\_LEFT (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *attribute*), 59

---

SBP\_55\_MOD\_FACE\_JEWELRY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_extra\_datas () (*pyffi.formats.nif.NifFormat.NiObjectNET* method), 135

SBP\_56\_MOD\_CHEST\_SECONDARY (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_geometry () (*pyffi.formats.cgf.CgfFormat.MeshChunk* method), 17
   
 set\_identity () (*pyffi.formats.cgf.CgfFormat.Matrix33* method), 16

SBP\_57\_MOD\_SHOULDER (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_identity () (*pyffi.formats.cgf.CgfFormat.Matrix44* method), 16

SBP\_58\_MOD\_ARM\_LEFT (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_identity () (*pyffi.formats.nif.NifFormat.InertiaMatrix* method), 96
   
 set\_identity () (*pyffi.formats.nif.NifFormat.Matrix33* method), 100

SBP\_59\_MOD\_ARM\_RIGHT (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_identity () (*pyffi.formats.nif.NifFormat.Matrix44* method), 101

SBP\_60\_MOD\_MISC2 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_identity () (*pyffi.formats.cgf.CgfFormat.Matrix44* method), 16

SBP\_61\_FX01 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* attribute), 59
   
 set\_identity () (*pyffi.formats.nif.NifFormat.Matrix44* method), 101

scale () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape* property), 216
   
 set\_node\_name () (*pyffi.formats.nif.NifFormat.ControllerLink* method), 84

scale () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62
   
 properties () (*pyffi.formats.nif.NifFormat.NiAVObject* method), 106

scale () (*pyffi.formats.nif.NifFormat.BSTreadTransformData* property), 75
   
 set\_property\_type () (*pyffi.formats.nif.NifFormat.ControllerLink* method), 84

scale () (*pyffi.formats.nif.NifFormat.MTransform* property), 99
   
 set\_rows () (*pyffi.formats.cgf.CgfFormat.Matrix44* method), 16

scale () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 127
   
 set\_rows () (*pyffi.formats.nif.NifFormat.Matrix44* method), 101

scale () (*pyffi.formats.nif.NifFormat.QTransform* property), 193
   
 set\_scale\_rotation ()

segment () (*pyffi.formats.nif.NifFormat.BSSegmentedTriShape* property), 70
   
 (*pyffi.formats.cgf.CgfFormat.Matrix33* method), 16

semantic () (*pyffi.formats.nif.NifFormat.ElementReference* property), 87
   
 set\_scale\_rotation () (*pyffi.formats.nif.NifFormat.Matrix33* method), 100

send\_bones\_to\_bind\_position () (*pyffi.formats.nif.NifFormat.NiGeometry* method), 123
   
 set\_scale\_rotation\_translation () (*pyffi.formats.nif.NifFormat.Matrix44* method), 101

send\_bones\_to\_bind\_position () (*pyffi.formats.nif.NifFormat.NiNode* method), 133
   
 set\_skin\_partition () (*pyffi.formats.nif.NifFormat.NiGeometry* method), 123

send\_detached\_geometries\_to\_node\_position () (*pyffi.formats.nif.NifFormat.NiNode* method), 134
   
 set\_strips () (*pyffi.formats.nif.NifFormat.NiTriShapeData* method), 185

send\_geometries\_to\_bind\_position () (*pyffi.formats.nif.NifFormat.NiNode* method), 134
   
 set\_strips () (*pyffi.formats.nif.NifFormat.NiTriStripsData* method), 185

set\_children () (*pyffi.formats.nif.NifFormat.NiNode* method), 134
   
 set\_target\_color () (*pyffi.formats.nif.NifFormat.NiMaterialColorController* method), 128

set\_controller\_type () (*pyffi.formats.nif.NifFormat.ControllerLink* method), 84
   
 set\_transform () (*pyffi.formats.nif.NifFormat.NiAVObject* method), 106

set\_effects () (*pyffi.formats.nif.NifFormat.NiNode* method), 134
   
 set\_transform () (*pyffi.formats.nif.NifFormat.NiSkinData* method), 174

set\_transform () (*pyffi.formats.nif.NifFormat.SkinData* method), 174

*method*), 197  
 set\_transform() (*pyffi.formats.nif.NifFormat.SkinTransform*  
*method*), 198  
 set\_translation() (*pyffi.formats.cgf.CgfFormat.Matrix44*  
*method*), 16  
 set\_translation() (*pyffi.formats.nif.NifFormat.Matrix44*  
*method*), 101  
 set\_triangles() (*pyffi.formats.nif.NifFormat.NiTriShapeData*  
*method*), 185  
 set\_triangles() (*pyffi.formats.nif.NifFormat.NiTriStripData*  
*method*), 185  
 set\_value() (*pyffi.formats.bsa.BsaFormat.ZString*  
*method*), 10  
 set\_value() (*pyffi.formats.cgf.CgfFormat.FileSignature*  
*method*), 14  
 set\_value() (*pyffi.formats.cgf.CgfFormat.Ref*  
*method*), 27  
 set\_value() (*pyffi.formats.cgf.CgfFormat.SizedString*  
*method*), 28  
 set\_value() (*pyffi.formats.egm.EgmFormat.FileVersion*  
*method*), 39  
 set\_value() (*pyffi.formats.egt.EgtFormat.FileVersion*  
*method*), 42  
 set\_value() (*pyffi.formats.esp.EspFormat.ZString*  
*method*), 47  
 set\_value() (*pyffi.formats.kfm.KfmFormat.HeaderString*  
*method*), 49  
 set\_value() (*pyffi.formats.kfm.KfmFormat.SizedString*  
*method*), 50  
 set\_value() (*pyffi.formats.nif.NifFormat.bool*  
*method*), 226  
 set\_value() (*pyffi.formats.nif.NifFormat.ByteArray*  
*method*), 78  
 set\_value() (*pyffi.formats.nif.NifFormat.ByteMatrix*  
*method*), 79  
 set\_value() (*pyffi.formats.nif.NifFormat.Data.VersionUInt*  
*method*), 85  
 set\_value() (*pyffi.formats.nif.NifFormat.FileVersion*  
*method*), 92  
 set\_value() (*pyffi.formats.nif.NifFormat.LineString*  
*method*), 98  
 set\_value() (*pyffi.formats.nif.NifFormat.Ptr*  
*method*), 193  
 set\_value() (*pyffi.formats.nif.NifFormat.Ref*  
*method*), 195  
 set\_value() (*pyffi.formats.nif.NifFormat.ShortString*  
*method*), 196  
 set\_value() (*pyffi.formats.nif.NifFormat.SizedString*  
*method*), 197  
 set\_value() (*pyffi.formats.tga.TgaFormat.FooterString*  
*method*), 234  
 set\_value() (*pyffi.formats.tri.TriFormat.FileVersion*  
*method*), 237  
 set\_variable\_1() (*pyffi.formats.nif.NifFormat.ControllerLink*  
*method*), 84  
 set\_variable\_2() (*pyffi.formats.nif.NifFormat.ControllerLink*  
*method*), 84  
 set\_vertices\_normals() (*pyffi.formats.cgf.CgfFormat.MeshChunk*  
*method*), 26  
 sf\_2\_1\_st\_light\_is\_point\_light() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_2\_nd\_light() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_3\_rd\_light() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_alpha\_decals() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_billboard\_and\_envmap\_light\_fade() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_envmap\_light\_fade() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_fit\_slope() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_lod\_building() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_lod\_landscape() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_no\_fade() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_no\_lod\_land\_blend() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_no\_transparency\_multisampling() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_premult\_alpha() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_refraction\_tint() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_show\_in\_local\_map() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71  
 sf\_2\_skip\_normal\_maps() (*pyffi.formats.nif.NifFormat.BSShaderFlags2*  
*property*), 71



*sf\_unknown\_3()* (*pyffi.formats.nif.NifFormat.BSShaderFlags* property), 73  
*sf\_unknown\_4()* (*pyffi.formats.nif.NifFormat.BSShaderFlags* attribute), 74  
*sf\_vertex\_alpha()* (*pyffi.formats.nif.NifFormat.BSShaderFlags* property), 71  
*sf\_window\_environment\_mapping()* (*pyffi.formats.nif.NifFormat.BSShaderFlags* property), 71  
*sf\_z\_buffer\_test()* (*pyffi.formats.nif.NifFormat.BSShaderFlags* property), 71  
SHADER\_DEFAULT (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 73  
*shader\_flags()* (*pyffi.formats.nif.NifFormat.BSShaderProperty* property), 73  
*shader\_flags\_1()* (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty* property), 60  
*shader\_flags\_1()* (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62  
*shader\_flags\_1()* (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty* property), 74  
*shader\_flags\_1()* (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty* property), 76  
*shader\_flags\_2()* (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty* property), 60  
*shader\_flags\_2()* (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 62  
*shader\_flags\_2()* (*pyffi.formats.nif.NifFormat.BSShaderProperty* property), 73  
*shader\_flags\_2()* (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty* property), 74  
*shader\_flags\_2()* (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty* property), 76  
SHADER\_LIGHTING30 (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 73  
SHADER\_NOLIGHTING (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 73  
SHADER\_SKIN (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 74  
SHADER\_SKY (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 74  
SHADER\_TALL\_GRASS (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 74  
*shader\_textures()* (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 181  
SHADER\_TILE (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 74  
*shader\_type()* (*pyffi.formats.nif.NifFormat.BSShaderProperty* property), 73  
SHADER\_WATER (*pyffi.formats.nif.NifFormat.BSShaderType* attribute), 74  
*shape()* (*pyffi.formats.nif.NifFormat.bhkWorldObject* property), 225  
*shape()* (*pyffi.formats.nif.NifFormat.SkinShape* property), 198  
*shape\_description()* (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
*shape\_groups\_1()* (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 114  
*shape\_groups\_2()* (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 114  
*shell\_link()* (*pyffi.formats.nif.NifFormat.NiRoomGroup* property), 171  
SHELLCASING (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91  
SHELLCASING (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 91  
SHIELD (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 189  
SHIELD (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
short (*pyffi.formats.cgf.CgfFormat* attribute), 29  
short (*pyffi.formats.dds.DdsFormat* attribute), 36  
short (*pyffi.formats.egm.EgmFormat* attribute), 40  
short (*pyffi.formats.egt.EgtFormat* attribute), 43  
short (*pyffi.formats.esp.EspFormat* attribute), 47  
short (*pyffi.formats.kfm.KfmFormat* attribute), 51  
short (*pyffi.formats.nif.NifFormat* attribute), 227  
short (*pyffi.formats.tga.TgaFormat* attribute), 235  
short (*pyffi.formats.tri.TriFormat* attribute), 239  
SHADER\_PROPERTY\_0 (*pyffi.formats.nif.NifFormat.bhkCMSDMaterial* property), 215  
shrink\_generation (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 144  
shrink\_time (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 144  
SIDE\_WEAPON (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189  
signature (*pyffi.formats.cgf.CgfFormat.Header* property), 15  
simulation\_steps (*pyffi.formats.nif.NifFormat.NiPSSimulator* property), 143  
simulator (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 141  
sit (*pyffi.formats.nif.NifFormat.AnimationType* attribute), 54  
size (*pyffi.formats.nif.NifFormat.BSMultiBoundOBB* property), 64  
size (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 141

*property*), 160  
size\_keys() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep*  
*property*), 144  
size\_loop\_behavior() (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep*  
*property*), 144  
sizes() (*pyffi.formats.nif.NifFormat.NiParticlesData*  
*property*), 161  
skeleton\_root() (*pyffi.formats.nif.NifFormat.NiSkinInstance*  
*property*), 175  
skeleton\_root() (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier*  
*property*), 175  
skeleton\_transform() (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier*  
*property*), 175  
skelrootentry() (*pyffi.spells.nif.fix.SpellSendBonesToBindPosition*  
*method*), 247  
skelrootentry() (*pyffi.spells.nif.fix.SpellSendDetachedGeometryToBindPosition*  
*method*), 247  
skelrootentry() (*pyffi.spells.nif.fix.SpellSendGeometryToBindPosition*  
*method*), 247  
skin\_instance() (*pyffi.formats.nif.NifFormat.SkinShape*  
*property*), 198  
skin\_partition() (*pyffi.formats.nif.NifFormat.NiSkinInstance*  
*property*), 175  
skin\_partition\_blocks() (*pyffi.formats.nif.NifFormat.NiSkinPartition*  
*property*), 175  
skin\_tint\_color() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty*  
*property*), 62  
skip\_regexs (*pyffi.spells.Toaster attribute*), 271  
sky\_object\_type() (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty*  
*property*), 74  
sky\_object\_type() (*pyffi.formats.nif.NifFormat.SkyShaderProperty*  
*property*), 198  
Sleep (*pyffi.formats.nif.NifFormat.AnimationType attribute*), 55  
sliding\_a() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor*  
*property*), 192  
sliding\_b() (*pyffi.formats.nif.NifFormat.PrismaticDescriptor*  
*property*), 192  
slsf\_1\_cast\_shadows() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_decals() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_dynamic\_decals() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_environment\_mapping() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_external\_emittance() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_eye\_environment\_mapping() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_face\_gen\_rgb\_tint() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_face\_gen\_detail\_map() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_fire\_refraction() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_greyscale\_to\_palette\_alpha() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_greyscale\_to\_palette\_color() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_hair\_soft\_lighting() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_landscape() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_localmap\_hide\_secret() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_model\_space\_normals() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_multiple\_textures() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_non\_projective\_shadows() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_own\_emit() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_parallax() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_parallax\_occlusion() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_projected\_uv() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202  
slsf\_1\_recieve\_shadows() (*pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1*  
*property*), 202

<i>property</i> ), 202	<i>property</i> ), 203
slsf_1_refraction() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_fit_slope() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_remappable_textures() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_glow_map() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_screendoor_alpha_fade() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_hd_lod_objects() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_skinned() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_hide_on_local_map() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_soft_effect() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_lod_landscape() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_specular() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_lod_objects() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_temp_refraction() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_multi_index_snow() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_use_falloff() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 202	slsf_2_multi_layer_parallax() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_vertex_alpha() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 203	slsf_2_no_fade() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_1_z_buffer_test() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1</i> <i>property</i> ), 203	slsf_2_no_lod_land_blend() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_anisotropic_lighting() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_no_transparency_multisampling() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_assume_shadowmask() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_packed_tangent() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_back_lighting() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_premult_alpha() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_billboard() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_rim_lighting() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_cloud_lod() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_soft_lighting() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_double_sided() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_tree_anim() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_effect_lighting() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_uniform_scale() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
slsf_2_env_map_light_fade() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203	slsf_2_unused_01() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203
	slsf_2_unused_02() ( <i>pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2</i> <i>property</i> ), 203

(pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2\_texture() (pyffi.formats.nif.NifFormat.NiTextureEffect property), 203  
 sources() (pyffi.formats.nif.NifFormat.NiFlipController property), 179  
 slsf\_2\_vertex\_colors() (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 120  
 slsf\_2\_vertex\_lighting() (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 203  
 slsf\_2\_weapon\_blood() (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 155  
 slsf\_2\_wireframe() (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 146  
 slsf\_2\_z\_buffer\_write() (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 203  
 sm\_group() (pyffi.formats.cgf.CgfFormat.Face property), 14  
 smooth() (pyffi.formats.nif.NifFormat.BSShaderProperty property), 73  
 soft\_falloff\_depth() (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 60  
 SOLVER\_DEACTIVATION\_HIGH (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SOLVER\_DEACTIVATION\_INVALID (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SOLVER\_DEACTIVATION\_LOW (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SOLVER\_DEACTIVATION\_MAX (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SOLVER\_DEACTIVATION\_MEDIUM (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SOLVER\_DEACTIVATION\_OFF (pyffi.formats.nif.NifFormat.SolverDeactivation attribute), 204  
 SORTING\_INHERIT (pyffi.formats.nif.NifFormat.SortingMode attribute), 204  
 sorting\_mode() (pyffi.formats.nif.NifFormat.NiSortAdjustment property), 175  
 SORTING\_OFF (pyffi.formats.nif.NifFormat.SortingMode attribute), 204  
 source() (pyffi.formats.nif.NifFormat.TexDesc property), 209  
 source\_texture() (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 60  
 source\_texture() (pyffi.formats.nif.NifFormat.BSSkyShaderProperty property), 74  
 specular\_color() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 63  
 specular\_color() (pyffi.formats.nif.NifFormat.NiLight property), 127  
 specular\_strength() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 63  
 speed() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 160  
 speed() (pyffi.formats.nif.NifFormat.NiPSysEmitter property), 150  
 speed\_random() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 160  
 speed\_variation() (pyffi.formats.nif.NifFormat.NiPSysEmitter property), 150  
 spell, 304  
 Spell (class in pyffi.spells), 264  
 SPELL (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 91  
 SPELL (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 189  
 SPELL (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 201  
 SPELL\_EXPLOSION (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 189  
 SpellAddStencilProperty (class in pyffi.spells.nif.modify), 262  
 SpellApplyStencilProperty (class in pyffi.spells.nif.modify), 244  
 SpellApplySkinDeformation (class in pyffi.spells), 264

- pyffi.spells.nif.fix*), 248
- SpellChangeBonePriorities (class in *pyffi.spells.nif.modify*), 259
- SpellClampMaterialAlpha (class in *pyffi.spells.nif.fix*), 246
- SPELLCLASSES (*pyffi.spells.SpellGroupBase* attribute), 267
- SpellCleanFarNif (class in *pyffi.spells.nif.modify*), 264
- SpellCleanRefLists (class in *pyffi.spells.nif.optimize*), 251
- SpellCleanStringPalette (class in *pyffi.spells.nif.fix*), 249
- SpellCollisionMaterial (class in *pyffi.spells.nif.modify*), 256
- SpellCollisionType (class in *pyffi.spells.nif.modify*), 255
- SpellDelBranches (class in *pyffi.spells.nif.modify*), 261
- SpellDelInterpolatorTransformData (class in *pyffi.spells.nif.modify*), 260
- SpellDelSkinShapes (class in *pyffi.spells.nif.modify*), 262
- SpellDelTangentSpace (class in *pyffi.spells.nif.fix*), 243
- SpellDelUnusedBones (class in *pyffi.spells.nif.optimize*), 253
- SpellDelUnusedRoots (class in *pyffi.spells.nif.fix*), 250
- SpellDelVertexColor (class in *pyffi.spells.nif.modify*), 263
- SpellDetachHavokTriStripsData (class in *pyffi.spells.nif.fix*), 245
- SpellDisableParallax (class in *pyffi.spells.nif.modify*), 262
- SPELLEXPLOSION (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- SPELLEXPLOSION (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 201
- SpellFFVT3RSkinPartition (class in *pyffi.spells.nif.fix*), 244
- SpellFixCenterRadius (class in *pyffi.spells.nif.fix*), 249
- SpellFixEmptySkeletonRoots (class in *pyffi.spells.nif.fix*), 250
- SpellFixMopp (class in *pyffi.spells.nif.fix*), 249
- SpellFixSkinCenterRadius (class in *pyffi.spells.nif.fix*), 249
- SpellFixTexturePath (class in *pyffi.spells.nif.fix*), 245
- SpellGroupBase (class in *pyffi.spells*), 267
- SpellGroupParallel () (in module *pyffi.spells*), 267
- SpellGroupParallelBase (class in *pyffi.spells*), 267
- SpellGroupSeries () (in module *pyffi.spells*), 267
- SpellGroupSeriesBase (class in *pyffi.spells*), 268
- SpellLowResTexturePath (class in *pyffi.spells.nif.modify*), 255
- SpellMakeFarNif (class in *pyffi.spells.nif.modify*), 264
- SpellMakeSkinlessNif (class in *pyffi.spells.nif.modify*), 264
- SpellMergeDuplicates (class in *pyffi.spells.nif.optimize*), 252
- SpellMergeSkeletonRoots (class in *pyffi.spells.nif.fix*), 247
- SPELLNAME (*pyffi.spells.Spell* attribute), 264
- spellnames (*pyffi.spells.Toaster* attribute), 271
- SpellOptimize (class in *pyffi.spells.nif.optimize*), 253
- SpellOptimizeGeometry (class in *pyffi.spells.nif.optimize*), 253
- SpellReverseAnimation (class in *pyffi.spells.nif.modify*), 258
- spells (*pyffi.spells.SpellGroupBase* attribute), 267
- SPELLS (*pyffi.spells.Toaster* attribute), 269
- SpellScale (class in *pyffi.spells.nif.fix*), 248
- SpellScaleAnimationTime (class in *pyffi.spells.nif.modify*), 257
- SpellSendBonesToBindPosition (class in *pyffi.spells.nif.fix*), 247
- SpellSendDetachedGeometriesToNodePosition (class in *pyffi.spells.nif.fix*), 247
- SpellSendGeometriesToBindPosition (class in *pyffi.spells.nif.fix*), 247
- SpellSetInterpolatorTransRotScale (class in *pyffi.spells.nif.modify*), 259
- SpellSubstituteStringPalette (class in *pyffi.spells.nif.modify*), 258
- SpellSubstituteTexturePath (class in *pyffi.spells.nif.modify*), 255
- SpellTexturePath (class in *pyffi.spells.nif.modify*), 254
- sphere () (*pyffi.formats.nif.NifFormat.BoundingBoxVolume* property), 78
- SPHERE\_BV (*pyffi.formats.nif.NifFormat.BoundingBoxVolumeType* attribute), 77
- SPHERICAL\_SYMMETRY (*pyffi.formats.nif.NifFormat.SymmetryType* attribute), 207
- SPINE1 (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- SPINE1 (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189
- SPINE2 (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 91
- SPINE2 (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 189

split\_triangles() (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape property), 152  
 (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShapeString (pyffi.formats.cgf.CgfFormat attribute), 28  
 method), 220  
 STAIRHELPER (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 201  
 STAIRS (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 189  
 start() (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey2 property), 69  
 (pyffi.formats.nif.NifFormat.NiParticleBomb property), 88  
 (pyffi.formats.nif.NifFormat.NiParticleBomb property), 157  
 start\_frame() (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 219  
 (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 67  
 start\_frame\_fudge() (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 214  
 (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 67  
 start\_index() (pyffi.formats.nif.NifFormat.Region property), 195  
 start\_random() (pyffi.formats.nif.NifFormat.NiParticleSystemConstraint01 property), 160  
 start\_time() (pyffi.formats.nif.NifFormat.NiTimeController property), 182  
 STATIC (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 91  
 STATIC (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 189  
 STATIC (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 201  
 stencil\_enabled() (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 stencil\_function() (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 stencil\_mask() (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 stencil\_ref() (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 (pyffi.formats.nif.NifFormat.NiStencilProperty property), 177  
 stiff\_spring() (pyffi.formats.nif.NifFormat.bhkStiffSpringConstraint) (pyffi.formats.nif.NifFormat.Matrix33 method), 225  
 (pyffi.formats.nif.NifFormat.SubConstraint) (pyffi.formats.nif.NifFormat.Matrix44 method), 207  
 StiffSpring (pyffi.formats.nif.NifFormat.hkConstraintType attribute), 226  
 stop\_time() (pyffi.formats.nif.NifFormat.NiTimeController property), 182  
 stream (pyffi.spells.Spell attribute), 266  
 stream() (pyffi.formats.nif.NifFormat.MeshData property), 101  
 streamable() (pyffi.formats.nif.NifFormat.NiDataStream property), 118  
 strength() (pyffi.formats.nif.NifFormat.BSWindModifier property), 76  
 (pyffi.formats.nif.NifFormat.NiP SysGravityModifier property), 152  
 string (pyffi.formats.cgf.CgfFormat attribute), 28  
 string\_data() (pyffi.formats.nif.NifFormat.NiStringExtraData property), 178  
 StringIndex (pyffi.formats.nif.NifFormat attribute), 205  
 strip\_width() (pyffi.formats.nif.NifFormat.BSPProceduralLightningConvex property), 215  
 strips() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 215  
 strips\_data() (pyffi.formats.nif.NifFormat.bhkMeshShape property), 219  
 sub\_constraint() (pyffi.formats.nif.NifFormat.bhkBreakableConstraint property), 214  
 sub\_shapes() (pyffi.formats.nif.NifFormat.bhkConvexListShape property), 218  
 submesh\_to\_region\_map() (pyffi.formats.nif.NifFormat.MeshData property), 101  
 submit\_points() (pyffi.formats.nif.NifFormat.NiMeshModifier property), 129  
 substitute() (pyffi.spells.nif.fix.SpellCleanStringPalette method), 250  
 substitute() (pyffi.spells.nif.fix.SpellFixTexturePath method), 245  
 substitute() (pyffi.spells.nif.modify.SpellLowResTexturePath method), 255  
 substitute() (pyffi.spells.nif.modify.SpellSubstituteStringPalette method), 258  
 substitute() (pyffi.spells.nif.modify.SpellSubstituteTexturePath method), 255  
 substitute() (pyffi.spells.nif.modify.SpellTexturePath method), 254  
 subtexture\_offset\_u\_vs() (pyffi.formats.nif.NifFormat.NiP SysData property), 148  
 tanh\_norm() (pyffi.formats.cgf.CgfFormat.Matrix44 method), 16  
 tanh\_norm() (pyffi.formats.nif.NifFormat.Matrix33 method), 100  
 tanh\_norm() (pyffi.formats.nif.NifFormat.Matrix44 method), 101  
 toggle\_state() (pyffi.formats.nif.NifFormat.NiDynamicEffect property), 119  
 tswsf\_1\_bypass\_refraction\_map() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 204  
 tswsf\_1\_enabled() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 204  
 tswsf\_1\_highlight\_layer\_toggle() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 204  
 tswsf\_1\_unknown\_0() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 204

*property*), 204  
 swsf\_1\_unknown\_3 () (*pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags* *property*), 204  
 swsf\_1\_unknown\_4 () (*pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags* *property*), 204  
 swsf\_1\_unknown\_5 () (*pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags* *property*), 204  
 swsf\_1\_water\_toggle () (*pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags* *property*), 204  
 symmetry\_type () (*pyffi.formats.nif.NifFormat.NiParticleBomb* *property*), 157  
 symmetry\_type () (*pyffi.formats.nif.NifFormat.NiPSysBombModifier* *property*), 147  
 SYNC\_ANY (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_PHYSICS\_COMPLETED (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_PHYSICS\_SIMULATE (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_POST\_UPDATE (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_REFLECTIONS (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_RENDER (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_UPDATE (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207  
 SYNC\_VISIBLE (*pyffi.formats.nif.NifFormat.SyncPoint* *attribute*), 207

**T**

t () (*pyffi.formats.nif.NifFormat.TBC* *property*), 208  
 t\_0 () (*pyffi.formats.cgf.CgfFormat.UVFace* *property*), 29  
 t\_1 () (*pyffi.formats.cgf.CgfFormat.UVFace* *property*), 29  
 t\_2 () (*pyffi.formats.cgf.CgfFormat.UVFace* *property*), 29  
 TAIL (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 92  
 TAIL (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 189  
 Tangents\_Bitangents (*pyffi.formats.nif.NifFormat.ExtraVectorsFlags* *attribute*), 89  
 target () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape* *property*), 216  
 target () (*pyffi.formats.nif.NifFormat.NiCollisionObject* *property*), 117  
 target () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* *property*), 128  
 target () (*pyffi.formats.nif.NifFormat.NiPortal* *property*), 169  
 target () (*pyffi.formats.nif.NifFormat.NiPSysModifier* *property*), 153  
 target () (*pyffi.formats.nif.NifFormat.NiShadowGenerator* *property*), 173  
 target () (*pyffi.formats.nif.NifFormat.NiTimeController* *property*), 182  
 target\_color () (*pyffi.formats.nif.NifFormat.NiPoint3InterpController* *property*), 168  
 target\_names () (*pyffi.formats.nif.NifFormat.NiMorphWeightsController* *property*), 130  
 tbc () (*pyffi.formats.nif.NifFormat.Key* *property*), 97  
 tbc () (*pyffi.formats.nif.NifFormat.QuatKey* *property*), 194  
 TBC\_KEY (*pyffi.formats.nif.NifFormat.KeyType* *attribute*), 97  
 TC\_AMBIENT (*pyffi.formats.nif.NifFormat.TargetColor* *attribute*), 208  
 TC\_DIFFUSE (*pyffi.formats.nif.NifFormat.TargetColor* *attribute*), 208  
 TC\_SELF\_ILUM (*pyffi.formats.nif.NifFormat.TargetColor* *attribute*), 208  
 TC\_SPECULAR (*pyffi.formats.nif.NifFormat.TargetColor* *attribute*), 208  
 TERRAIN (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 92  
 TERRAIN (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 189  
 TERRAIN (*pyffi.formats.nif.NifFormat.SkyrimLayer* *attribute*), 201  
 TEST\_ALWAYS (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_EQUAL (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_GREATER (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_GREATER\_EQUAL (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_LESS (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_LESS\_EQUAL (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_NEVER (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 TEST\_NOT\_EQUAL (*pyffi.formats.nif.NifFormat.StencilCompareMode* *attribute*), 205  
 text\_keys () (*pyffi.formats.nif.NifFormat.NiSequence* *property*), 172

text\_keys () (*pyffi.formats.nif.NifFormat.NiTextKeyExtraData* format .ImageType (class in *pyffi.formats.tga*),  
     property), 178 235  
 text\_keys\_name () (*pyffi.formats.nif.NifFormat.NiSequenceXMLPATH*, 7  
     property), 172 threshold () (*pyffi.formats.nif.NifFormat.bhkBreakableConstraint*  
     property), 214  
 TextString (*pyffi.formats.kfm.KfmFormat* attribute),  
     51 threshold () (*pyffi.formats.nif.NifFormat.NiAlphaProperty*  
     property), 107  
 texture\_clamp\_mode ()  
     (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty*  
     property), 60 tilling () (*pyffi.formats.nif.NifFormat.TexDesc* prop-  
     erty), 209  
 texture\_clamp\_mode ()  
     (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty*  
     property), 63 time () (*pyffi.formats.nif.NifFormat.Key* property), 97  
 texture\_clamp\_mode ()  
     (*pyffi.formats.nif.NifFormat.BSShaderLightingProperty*  
     property), 72 time () (*pyffi.formats.nif.NifFormat.QuatKey* property),  
     194  
 texture\_clamping ()  
     (*pyffi.formats.nif.NifFormat.NiTextureEffect*  
     property), 179 timestamp () (*pyffi.formats.nif.NifFormat.Particle*  
     property), 191  
 texture\_count () (*pyffi.formats.nif.NifFormat.NiTexturingProperty*  
     property), 182 toast () (*pyffi.spells.Toaster* method), 271  
 texture\_data () (*pyffi.formats.nif.NifFormat.ShaderTexDesc*  
     property), 196 toast\_archives () (*pyffi.spells.Toaster* method),  
     271  
 texture\_elements ()  
     (*pyffi.formats.nif.NifFormat.NiMultiTextureProperty*  
     property), 131 toastentry () (*pyffi.spells.nif.fix.SpellScale* class  
     method), 249  
 texture\_filtering ()  
     (*pyffi.formats.nif.NifFormat.NiTextureEffect*  
     property), 179 toastentry () (*pyffi.spells.nif.modify.SpellChangeBonePriorities*  
     class method), 259  
 texture\_name () (*pyffi.formats.nif.NifFormat.ArkTexture*  
     property), 55 toastentry () (*pyffi.spells.nif.modify.SpellCollisionMaterial*  
     class method), 257  
 texture\_set () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty*  
     property), 63 toastentry () (*pyffi.spells.nif.modify.SpellCollisionType*  
     class method), 256  
 texture\_set () (*pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty*  
     property), 73 toastentry () (*pyffi.spells.nif.modify.SpellDelInterpolatorTransformData*  
     class method), 261  
 texture\_slot () (*pyffi.formats.nif.NifFormat.NiFlipController*  
     property), 120 toastentry () (*pyffi.spells.nif.modify.SpellLowResTexturePath*  
     class method), 255  
 texture\_slot () (*pyffi.formats.nif.NifFormat.NiTextureTransformController*  
     property), 180 toastentry () (*pyffi.spells.nif.modify.SpellScaleAnimationTime*  
     class method), 257  
 texture\_type () (*pyffi.formats.nif.NifFormat.NiTextureEffect*  
     property), 179 toastentry () (*pyffi.spells.nif.modify.SpellSetInterpolatorTransRotScale*  
     class method), 260  
 textures () (*pyffi.formats.nif.NifFormat.BSShaderTextureSet*  
     property), 73 toastentry () (*pyffi.spells.nif.modify.SpellSubstituteStringPalette*  
     class method), 258  
 textures () (*pyffi.formats.nif.NifFormat.NiArkTextureExtraData*  
     property), 108 toastentry () (*pyffi.spells.nif.modify.SpellSubstituteTexturePath*  
     class method), 255  
 texturing\_property ()  
     (*pyffi.formats.nif.NifFormat.ArkTexture* prop-  
     erty), 55 toastentry () (*pyffi.spells.nif.modify.SpellTexturePath*  
     class method), 254  
 TgaFormat (class in *pyffi.formats.tga*), 233 toastentry () (*pyffi.spells.Spell* class method), 266  
 TgaFormat .ColorMapType (class in  
     *pyffi.formats.tga*), 233 toastentry () (*pyffi.spells.SpellGroupBase* class  
     method), 267  
 TgaFormat .Data (class in *pyffi.formats.tga*), 234 Toaster (class in *pyffi.spells*), 269  
 TgaFormat .FooterString (class in  
     *pyffi.formats.tga*), 234 toaster (*pyffi.spells.Spell* attribute), 266  
 TgaFormat .Image (class in *pyffi.formats.tga*), 234 toastexit () (*pyffi.spells.Spell* class method), 266  
     toastexit () (*pyffi.spells.SpellGroupBase* class  
     method), 267  
     top (*pyffi.spells.Toaster* attribute), 271  
     trailer () (*pyffi.formats.nif.NifFormat.NiParticleSystemController*  
     property), 160  
     transform\_1 () (*pyffi.formats.nif.NifFormat.BSTreadTransform*  
     property), 75  
     transform\_2 () (*pyffi.formats.nif.NifFormat.BSTreadTransform*

<i>property</i> ), 75	<code>tree()</code> ( <i>pyffi.formats.cgf.CgfFormat.Chunk</i> method), 12
<code>transform_dests()</code> ( <i>pyffi.formats.nif.NifFormat.NiPhysXProp</i> <i>property</i> ), 165	<code>tree()</code> ( <i>pyffi.formats.nif.NifFormat.NiObject</i> method), 134
<code>transform_index()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDChunk</i> <i>property</i> ), 215	TREES ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i> ), 92
<code>transform_type()</code> ( <i>pyffi.formats.nif.NifFormat.TexDesc</i> <i>property</i> ), 209	TREES ( <i>pyffi.formats.nif.NifFormat.OblivionLayer</i> <i>at-</i> <i>tribute</i> ), 190
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDChunk</i> <i>property</i> ), 215	TREES ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>at-</i> <i>tribute</i> ), 201
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDChunk</i> <i>property</i> ), 215	<code>triangle()</code> ( <i>pyffi.formats.nif.NifFormat.hkTriangle</i> <i>property</i> ), 227
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.BoundingBox</i> <i>property</i> ), 77	<code>triangle_1()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDBigTris</i> <i>property</i> ), 214
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.BSTreadTransformData</i> <i>property</i> ), 75	<code>triangle_2()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDBigTris</i> <i>property</i> ), 214
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.MTransform</i> <i>property</i> ), 99	<code>triangle_3()</code> ( <i>pyffi.formats.nif.NifFormat.bhkCMSDBigTris</i> <i>property</i> ), 214
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.NiLookAtInterpolator</i> <i>property</i> ), 128	<code>triangle_offset()</code> ( <i>pyffi.formats.nif.NifFormat.Polygon</i> <i>prop-</i> <i>erty</i> ), 192
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.ParticleDesc</i> <i>property</i> ), 191	<code>TriFormat</code> (class in <i>pyffi.formats.tri</i> ), 236
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.QTransform</i> <i>property</i> ), 194	<code>TriFormat.FileSignature</code> (class in <i>pyffi.formats.tri</i> ), 236
<code>translation()</code> ( <i>pyffi.formats.nif.NifFormat.TexDesc</i> <i>property</i> ), 209	<code>TriFormat.FileVersion</code> (class in <i>pyffi.formats.tri</i> ), 237
TRANSPARENT ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i> ), 92	<code>TriFormat.Header</code> (class in <i>pyffi.formats.tri</i> ), 237
TRANSPARENT ( <i>pyffi.formats.nif.NifFormat.OblivionLayer</i> <i>attribute</i> ), 190	<code>TriFormat.ModifierRecord</code> (class in <i>pyffi.formats.tri</i> ), 238
TRANSPARENT ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>attribute</i> ), 201	<code>TriFormat.MorphRecord</code> (class in <i>pyffi.formats.tri</i> ), 238
TRANSPARENT_SMALL ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i> ), 92	<code>TriFormat.QuadFace</code> (class in <i>pyffi.formats.tri</i> ), 239
TRANSPARENT_SMALL ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>attribute</i> ), 201	<code>TriFormat.SizedStringZ</code> (class in <i>pyffi.formats.tri</i> ), 239
TRANSPARENT_SMALL_ANIM ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i> ), 92	TRIGGER ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>at-</i> <i>tribute</i> ), 92
TRANSPARENT_SMALL_ANIM ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>attribute</i> ), 201	TRIGGER ( <i>pyffi.formats.nif.NifFormat.OblivionLayer</i> <i>at-</i> <i>tribute</i> ), 190
TRAP ( <i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>at-</i> <i>tribute</i> ), 92	TRIGGER ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>at-</i> <i>tribute</i> ), 201
TRAP ( <i>pyffi.formats.nif.NifFormat.OblivionLayer</i> <i>at-</i> <i>tribute</i> ), 190	TT_ROTATE ( <i>pyffi.formats.nif.NifFormat.TexTransform</i> <i>attribute</i> ), 210
TRAP ( <i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>at-</i> <i>tribute</i> ), 201	TT_SCALE_U ( <i>pyffi.formats.nif.NifFormat.TexTransform</i> <i>attribute</i> ), 210
<code>tread_transforms()</code> ( <i>pyffi.formats.nif.NifFormat.BSTreadTransformInterpolator</i> <i>property</i> ), 74	TT_SCALE_V ( <i>pyffi.formats.nif.NifFormat.TexTransform</i> <i>attribute</i> ), 210
	TT_TRANSLATE_U ( <i>pyffi.formats.nif.NifFormat.TexTransform</i> <i>attribute</i> ), 210
	TT_TRANSLATE_V ( <i>pyffi.formats.nif.NifFormat.TexTransform</i> <i>attribute</i> ), 210
	<code>turbulence()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSysGravityModifier</i> <i>property</i> ), 152
	<code>turbulence_scale()</code> ( <i>pyffi.formats.nif.NifFormat.NiPSysGravityModifier</i> <i>property</i> ), 152





property), 141  
 unknown\_18 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_19 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 136  
 unknown\_19 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 139  
 unknown\_19 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_19 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 142  
 unknown\_19 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_2 () (pyffi.formats.nif.NifFormat.bhkMeshShape property), 219  
 unknown\_2 () (pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimation property), 104  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiBezierTriangle property), 112  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiGeomMorpherController property), 122  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiMorphWeightsController property), 130  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 136  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSBoundUpdater property), 136  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 138  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSEmitterRadiusCone property), 138  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 139  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSForceActiveCtrl property), 139  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 139  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSGravityStrengthCtrl property), 140  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 144  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiPSSphericalCollision property), 145  
 unknown\_2 () (pyffi.formats.nif.NifFormat.NiTextureTransformController property), 180  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 139  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 139  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 142  
 unknown\_20 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_200 () (pyffi.formats.nif.NifFormat.NiMesh property), 129  
 unknown\_201 () (pyffi.formats.nif.NifFormat.NiMesh property), 129  
 unknown\_21 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_21 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140  
 unknown\_21 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_21 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 142  
 unknown\_21 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_22 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_22 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140  
 unknown\_22 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_22 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 142  
 unknown\_22 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145  
 unknown\_23 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_23 () (pyffi.formats.nif.NifFormat.NiPSCylinderEmitter property), 138  
 unknown\_23 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140  
 unknown\_23 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_23 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 141  
 unknown\_24 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_24 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140  
 unknown\_24 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141  
 unknown\_24 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 141  
 unknown\_25 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137  
 unknown\_25 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140  
 unknown\_25 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141

*property*), 141  
 unknown\_25 () (*pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem*  
*property*), 141  
 unknown\_250 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_251 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_26 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_26 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_26 () (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter*  
*property*), 140  
 unknown\_26 () (*pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem*  
*property*), 141  
 unknown\_26 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 141  
 unknown\_27 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_27 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_27 () (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter*  
*property*), 141  
 unknown\_27 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_28 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_28 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_28 () (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter*  
*property*), 141  
 unknown\_28 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 129  
 unknown\_28 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 129  
 unknown\_29 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_29 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_29 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_292\_bytes ()  
 (*pyffi.formats.nif.NifFormat.FxWidget* *prop-*  
*erty*), 94  
 unknown\_2\_float ()  
 (*pyffi.formats.nif.NifFormat.NiTexturingProperty*  
*property*), 182  
 unknown\_2\_texture ()  
 (*pyffi.formats.nif.NifFormat.NiTexturingProperty*  
*property*), 182  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.FxWidget*  
*property*), 94  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiBezierMesh*  
*property*), 112  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiBezierTriangle*  
*property*), 112  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSBombForce*  
*property*), 136  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSDragForce*  
*property*), 138  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSEmitterSpeedCtrl*  
*property*), 139  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator*  
*property*), 139  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSGravityStrengthCtrl*  
*property*), 140  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter*  
*property*), 141  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep*  
*property*), 144  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSSphereEmitter*  
*property*), 145  
 unknown\_3 () (*pyffi.formats.nif.NifFormat.NiPSSphericalCollider*  
*property*), 146  
 unknown\_30 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_30 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_30 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_300 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_301 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_302 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_303 () (*pyffi.formats.nif.NifFormat.NiMesh*  
*property*), 129  
 unknown\_31 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_31 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_31 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_32 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_32 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_32 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142  
 unknown\_33 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter*  
*property*), 137  
 unknown\_33 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce*  
*property*), 140  
 unknown\_33 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem*  
*property*), 142

unknown\_34 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_34 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce* property), 140  
 unknown\_34 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_35 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_35 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce* property), 140  
 unknown\_35 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_350 () (*pyffi.formats.nif.NifFormat.NiMesh* property), 129  
 unknown\_351 () (*pyffi.formats.nif.NifFormat.NiMesh* property), 129  
 unknown\_36 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_36 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce* property), 140  
 unknown\_36 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_37 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_37 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_38 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_38 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_39 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_39 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiBezierMesh* property), 112  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiBezierTriangle4* property), 112  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSBombForce* property), 136  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSDragForce* property), 138  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator* property), 139  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* property), 137  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSDragForce* property), 138  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator* property), 139  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSGravityForce* property), 140  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter* property), 141  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 142  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSSphereEmitter* property), 145  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiPSSphericalCollider* property), 146  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiMesh* property), 129  
 unknown\_4 () (*pyffi.formats.nif.NifFormat.NiMesh* property), 129

unknown_53 ()	(pyffi.formats.nif.NifFormat.NiMesh property), 129	property), 136
unknown_54 ()	(pyffi.formats.nif.NifFormat.NiMesh property), 129	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137
unknown_55 ()	(pyffi.formats.nif.NifFormat.NiMesh property), 129	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 138
unknown_56 ()	(pyffi.formats.nif.NifFormat.NiMesh property), 129	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 139
unknown_5_ints ()	(pyffi.formats.nif.NifFormat.NiBinaryVoxelData property), 113	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiBezierMesh property), 112	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiBezierTriangle4 property), 112	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 143
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSBombForce property), 136	unknown_8 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 136
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSDragForce property), 138	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 138
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 139	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 138
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 139
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 142	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 143
unknown_6 ()	(pyffi.formats.nif.NifFormat.NiPSSphericalCollider property), 146	unknown_9 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSBombForce property), 136	unknown_array () (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property), 104
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 137	unknown_boolean_1 ()
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSDragForce property), 138	(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 146
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 139	unknown_boolean_2 ()
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSGravityForce property), 140	(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 146
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 141	unknown_boolean_3 ()
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 143	(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 146
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 145	unknown_byte () (pyffi.formats.nif.NifFormat.BSValueNode property), 75
unknown_7 ()	(pyffi.formats.nif.NifFormat.NiPSSphericalCollider property), 146	unknown_byte () (pyffi.formats.nif.NifFormat.NiArkTextureExtraData property), 108
unknown_7_floats ()	(pyffi.formats.nif.NifFormat.NiBinaryVoxelData property), 113	unknown_byte () (pyffi.formats.nif.NifFormat.NiPalette property), 157
unknown_8 ()	(pyffi.formats.nif.NifFormat.NiPSBombForce property), 136	unknown_byte () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 160
		unknown_byte () (pyffi.formats.nif.NifFormat.NiPhysXProp property), 165
		unknown_byte () (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 152

unknown\_byte () (*pyffi.formats.nif.NifFormat.NiSourceTexture* property), 176  
 unknown\_byte () (*pyffi.formats.nif.NifFormat.TexSource* property), 210  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.AdditionalDataInfo* property), 54  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.bhkCompressedListShape* property), 218  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.BSBlastNode* property), 55  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.BSDamageStage* property), 56  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.BSDebrisNode* property), 56  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.BSSegment* property), 70  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.ChannelData* property), 80  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.MotorDescription* property), 103  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.NiParticleData* property), 161  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDescription* property), 163  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXMaterialDesc* property), 164  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 164  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXTransformDesc* property), 167  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.physXMaterialRef* property), 227  
 unknown\_byte\_1 () (*pyffi.formats.nif.NifFormat.PrismaticDescription* property), 193  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData* property), 217  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 162  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXMaterialDesc* property), 164  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 164  
 unknown\_byte\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXTransformDesc* property), 167  
 unknown\_byte\_3 () (*pyffi.formats.nif.NifFormat.NiMeshPSysData* property), 129  
 unknown\_byte\_4 () (*pyffi.formats.nif.NifFormat.NiPSPlanarCollision* property), 143  
 unknown\_byte\_4 () (*pyffi.formats.nif.NifFormat.NiPSysData* property), 148  
 unknown\_byte\_6 () (*pyffi.formats.nif.NifFormat.BSStripPSysData* property), 74  
 unknown\_byte\_6 () (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc* property), 166  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.ArkTexture* property), 55  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.CStreamableAssetData* property), 79  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiArkAnimationExtraData* property), 107  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiArkImporterExtraData* property), 108  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiArkViewportInfoExtraData* property), 108  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiFloatExtraDataControl* property), 120  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiPhysXBodyDesc* property), 164  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiPhysXD6JointDesc* property), 164  
 unknown\_bytes () (*pyffi.formats.nif.NifFormat.NiPhysXKinematicSrc* property), 164  
 unknown\_bytes\_0 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 164  
 unknown\_bytes\_1 () (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData* property), 113  
 unknown\_bytes\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 164  
 unknown\_bytes\_2 () (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData* property), 113  
 unknown\_bytes\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 165  
 unknown\_bytes\_3 () (*pyffi.formats.nif.NifFormat.NiClodData* property), 116  
 unknown\_bytes\_3 () (*pyffi.formats.nif.NifFormat.NiClodData* property), 116  
 unknown\_bytes\_3 () (*pyffi.formats.nif.NifFormat.NiClodData* property), 116  
 unknown\_color () (*pyffi.formats.nif.NifFormat.NiParticleSystemControl* property), 160  
 unknown\_count\_1 () (*pyffi.formats.nif.NifFormat.NiClodData* property), 116

unknown_count_2 () ( <i>pyffi.formats.nif.NifFormat.NiClodData</i> <i>property</i> ), 116	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.CapsuleBV</i> <i>property</i> ), 80
unknown_count_3 () ( <i>pyffi.formats.nif.NifFormat.NiClodData</i> <i>property</i> ), 116	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.HalfSpaceBV</i> <i>property</i> ), 94
unknown_extra_bytes () ( <i>pyffi.formats.nif.NifFormat.NiFloatExtraDataController</i> <i>property</i> ), 120	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103
unknown_flags () ( <i>pyffi.formats.nif.NifFormat.NiPortal</i> <i>property</i> ), 169	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiGravity</i> <i>property</i> ), 125
unknown_flags () ( <i>pyffi.formats.nif.NifFormat.NiShadowGenerator</i> <i>property</i> ), 173	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiParticleSystemController</i> <i>property</i> ), 160
unknown_flags_1 () ( <i>pyffi.formats.nif.NifFormat.NiSwitchNode</i> <i>property</i> ), 178	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPathInterpolator</i> <i>property</i> ), 162
unknown_float () ( <i>pyffi.formats.nif.NifFormat.bhkSimpleShapePlane</i> <i>property</i> ), 224	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXMeshDesc</i> <i>property</i> ), 165
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiClodData</i> <i>property</i> ), 116	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXProp</i> <i>property</i> ), 165
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiFurSpringController</i> <i>property</i> ), 121	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXShapeDesc</i> <i>property</i> ), 166
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiImage</i> <i>property</i> ), 125	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 167
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiSpotLight</i> <i>property</i> ), 177	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 155
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiTextureEffect</i> <i>property</i> ), 179	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.NiSphericalCollider</i> <i>property</i> ), 176
unknown_float () ( <i>pyffi.formats.nif.NifFormat.NiVectorExtraData</i> <i>property</i> ), 186	unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.ParticleDesc</i> <i>property</i> ), 191
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain</i> <i>property</i> ), 213	unknown_float_10 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.bhkBlendCollisionObject</i> <i>property</i> ), 213	unknown_float_11 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i> <i>property</i> ), 216	unknown_float_12 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.bhkConvexListShape</i> <i>property</i> ), 218	unknown_float_13 () ( <i>pyffi.formats.nif.NifFormat.NiParticleSystemController</i> <i>property</i> ), 160
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.bhkLiquidAction</i> <i>property</i> ), 219	unknown_float_13 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.BSDecalPlacementVectorExtraData</i> <i>property</i> ), 56	
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier</i> <i>property</i> ), 65	
unknown_float_1 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66	

unknown_float_14 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiSphericalCollider</i> <i>property</i> ), 176
unknown_float_15 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.ParticleDesc</i> <i>property</i> ), 191
unknown_float_16 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i> <i>property</i> ), 216
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain</i> <i>property</i> ), 213	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.bhkLiquidAction</i> <i>property</i> ), 219
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.bhkBlendCollisionObject</i> <i>property</i> ), 213	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier</i> <i>property</i> ), 65
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.bhkLiquidAction</i> <i>property</i> ), 219	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier</i> <i>property</i> ), 65	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiPathController</i> <i>property</i> ), 162
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.CapsuleBV</i> <i>prop-</i> <i>erty</i> ), 80	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXShapeDesc</i> <i>property</i> ), 166
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiFurSpringController</i> <i>property</i> ), 121	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier</i> <i>property</i> ), 146
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPathController</i> <i>property</i> ), 162	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 156
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPathInterpolator</i> <i>property</i> ), 162	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.NiSphericalCollider</i> <i>property</i> ), 176
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXMeshDesc</i> <i>property</i> ), 165	unknown_float_3 () ( <i>pyffi.formats.nif.NifFormat.ParticleDesc</i> <i>property</i> ), 191
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPhysXShapeDesc</i> <i>property</i> ), 166	unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i> <i>property</i> ), 216
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.bhkLiquidAction</i> <i>property</i> ), 219
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier</i> <i>property</i> ), 146	unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66
unknown_float_2 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 155	unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty</i> <i>property</i> ), 73

unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103	unknown_float_8 () ( <i>pyffi.formats.nif.NifFormat.BSStripPSysData</i> <i>property</i> ), 74
unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_float_8 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier</i> <i>property</i> ), 146	unknown_float_9 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168
unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 156	unknown_floats () ( <i>pyffi.formats.nif.NifFormat.bhkConvexListShape</i> <i>property</i> ), 218
unknown_float_4 () ( <i>pyffi.formats.nif.NifFormat.NiSphericalCollider</i> <i>property</i> ), 177	unknown_floats () ( <i>pyffi.formats.nif.NifFormat.bhkMeshShape</i> <i>property</i> ), 219
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i> <i>property</i> ), 216	unknown_floats () ( <i>pyffi.formats.nif.NifFormat.NiArkImporterExtraData</i> <i>property</i> ), 108
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66	unknown_floats () ( <i>pyffi.formats.nif.NifFormat.NiBSplinePoint3Interp</i> <i>property</i> ), 111
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty</i> <i>property</i> ), 73	unknown_floats_1 () ( <i>pyffi.formats.nif.NifFormat.BallAndSocketDescriptor</i> <i>property</i> ), 76
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103	unknown_floats_1 () ( <i>pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i> <i>property</i> ), 216
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_floats_1 () ( <i>pyffi.formats.nif.NifFormat.Ni3dsAnimationNode</i> <i>property</i> ), 105
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 156	unknown_floats_1 () ( <i>pyffi.formats.nif.NifFormat.ParticleDesc</i> <i>property</i> ), 191
unknown_float_5 () ( <i>pyffi.formats.nif.NifFormat.NiSphericalCollider</i> <i>property</i> ), 177	unknown_floats_2 () ( <i>pyffi.formats.nif.NifFormat.BallAndSocketDescriptor</i> <i>property</i> ), 76
unknown_float_6 () ( <i>pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i> <i>property</i> ), 66	unknown_floats_2 () ( <i>pyffi.formats.nif.NifFormat.bhkSimpleShapePhantom</i> <i>property</i> ), 225
unknown_float_6 () ( <i>pyffi.formats.nif.NifFormat.MotorDescriptor</i> <i>property</i> ), 103	unknown_floats_2 () ( <i>pyffi.formats.nif.NifFormat.Ni3dsAnimationNode</i> <i>property</i> ), 105
unknown_float_6 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_floats_2 () ( <i>pyffi.formats.nif.NifFormat.NiParticleSystemController</i> <i>property</i> ), 160
unknown_float_6 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 156	unknown_floats_3 () ( <i>pyffi.formats.nif.NifFormat.NiPSysData</i> <i>property</i> ), 148
unknown_float_7 () ( <i>pyffi.formats.nif.NifFormat.NiPlanarCollider</i> <i>property</i> ), 168	unknown_floats_5 () ( <i>pyffi.formats.nif.NifFormat.NiPSPlanarCollider</i> <i>property</i> ), 143
unknown_float_7 () ( <i>pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i> <i>property</i> ), 156	unknown_int () ( <i>pyffi.formats.nif.NifFormat.bhkBlendController</i> <i>property</i> ), 214
	unknown_int () ( <i>pyffi.formats.nif.NifFormat.bhkRagdollTemplateData</i> <i>property</i> ), 224
	unknown_int () ( <i>pyffi.formats.nif.NifFormat.bhkRDTConstraint</i> <i>property</i> ), 222
	unknown_int () ( <i>pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint</i> <i>property</i> ), 222

*property*), 223  
 unknown\_int () (*pyffi.formats.nif.NifFormat.BoundingBox*  
     *property*), 78  
 unknown\_int () (*pyffi.formats.nif.NifFormat.BSMultiBoundNode*  
     *property*), 64  
 unknown\_int () (*pyffi.formats.nif.NifFormat.Morph*  
     *property*), 102  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiArkShade*  
     *property*), 108  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiBinaryVoxel*  
     *property*), 113  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiBlendInterp*  
     *property*), 114  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiCamera*  
     *property*), 116  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiDefaultAV*  
     *property*), 118  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiImage*  
     *property*), 125  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiMultiTexture*  
     *property*), 131  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiPathInterp*  
     *property*), 162  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiPhysXMaterial*  
     *property*), 164  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiPhysXProp*  
     *property*), 165  
 unknown\_int () (*pyffi.formats.nif.NifFormat.NiPSysSpawn*  
     *property*), 155  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BallAndSocket*  
     *property*), 76  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.bhkBallSocket*  
     *property*), 213  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.bhkCMSDBigTois*  
     *property*), 214  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.bhkCompressedMesh*  
     *property*), 216  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.bhkLiquidAction*  
     *property*), 219  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BSMultiBoundSphere*  
     *property*), 64  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BSPacked*  
     *property*), 68  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BSPSysIn*  
     *property*), 65  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BSPSysMulti*  
     *property*), 66  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.BSPSysRecycle*  
     *property*), 66  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.ExtraMeshDataEpic*  
     *property*), 88  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.FxRadioButton*  
     *property*), 94  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiArkImporter*  
     *property*), 108  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiBoneLODController*  
     *property*), 114  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiParticleSystem*  
     *property*), 158  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiParticleSystemController*  
     *property*), 160  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPathController*  
     *property*), 162  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc*  
     *property*), 163  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc*  
     *property*), 165  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXProp*  
     *property*), 165  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc*  
     *property*), 166  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc*  
     *property*), 166  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPSPlanarCollider*  
     *property*), 143  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiPSysTrailEmitter*  
     *property*), 156  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiSequence*  
     *property*), 172  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiSwitchNode*  
     *property*), 178  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.NiTextKeyExtraData*  
     *property*), 178  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.ParticleDesc*  
     *property*), 191  
 unknown\_int\_1 () (*pyffi.formats.nif.NifFormat.bhkCompressedMeshShape*  
     *property*), 217  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraint*  
     *property*), 213  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.bhkLiquidAction*  
     *property*), 219  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.BSMultiBoundSphere*  
     *property*), 65  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey*  
     *property*), 88  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.FxRadioButton*  
     *property*), 94  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiAlphaProperty*  
     *property*), 107  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiArkImporterExtraData*  
     *property*), 108  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiArkTextureExtraData*  
     *property*), 108  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiBoneLODController*  
     *property*), 114  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiCamera*  
     *property*), 116  
 unknown\_int\_2 () (*pyffi.formats.nif.NifFormat.NiFlipController*  
     *property*), 116

property), 120  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiMeshPSysData property), 130  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 160  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 163  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 165  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 166  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 166  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPSPlanarCollider property), 143  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 156  
 unknown\_int\_2 () (pyffi.formats.nif.NifFormat.NiSortAdjustMode property), 175  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.ArkTexture property), 55  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintClat property), 213  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 217  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.bhkLiquidAction property), 219  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.BSMultiBoundSphere property), 65  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey property), 88  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.FxRadioButton property), 94  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.NiBoneLODController property), 115  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.NiCamera property), 116  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 166  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 166  
 unknown\_int\_3 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 156  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.ArkTexture property), 55  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 217  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey property), 88  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 163  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 165  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 220  
 property), 167  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiPSysData property), 148  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 156  
 unknown\_int\_4 () (pyffi.formats.nif.NifFormat.NiSequence property), 173  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 217  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey property), 88  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 163  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 166  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 167  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.NiPSysData property), 148  
 unknown\_int\_5 () (pyffi.formats.nif.NifFormat.NiSequence property), 173  
 unknown\_int\_6 () (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureR property), 163  
 unknown\_int\_6 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 163  
 unknown\_int\_6 () (pyffi.formats.nif.NifFormat.NiPSysData property), 149  
 unknown\_int\_7 () (pyffi.formats.nif.NifFormat.BSStripPSysData property), 74  
 unknown\_int\_7 () (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureR property), 163  
 unknown\_int\_7 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 167  
 unknown\_int\_8 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 167  
 integer () (pyffi.formats.nif.NifFormat.NiTimeController property), 182  
 unknown\_ints () (pyffi.formats.nif.NifFormat.LODRange property), 97  
 unknown\_ints () (pyffi.formats.nif.NifFormat.NiArkAnimationExtraData property), 127  
 unknown\_ints () (pyffi.formats.nif.NifFormat.NiGeomMorpherController property), 122  
 unknown\_ints () (pyffi.formats.nif.NifFormat.NiTextureModeProperty property), 179  
 unknown\_ints\_1 () (pyffi.formats.nif.NifFormat.bhkAabbPhantom property), 213  
 unknown\_ints\_1 () (pyffi.formats.nif.NifFormat.bhkOrientHingedBody property), 220

unknown\_ints\_1 () (*pyffi.formats.nif.NifFormat.NiArkTextureExportData* property), 108  
 unknown\_ints\_1 () (*pyffi.formats.nif.NifFormat.NiMeshPSysData* property), 130  
 unknown\_ints\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 165  
 unknown\_ints\_1 () (*pyffi.formats.nif.NifFormat.NiTextureProperty* property), 180  
 unknown\_ints\_2 () (*pyffi.formats.nif.NifFormat.NiTextureProperty* property), 180  
 unknown\_link () (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 unknown\_link () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 162  
 unknown\_link () (*pyffi.formats.nif.NifFormat.NiParticleSystemComponent* property), 160  
 unknown\_link () (*pyffi.formats.nif.NifFormat.NiSourceTexture* property), 176  
 unknown\_link () (*pyffi.formats.nif.NifFormat.TexSource* property), 210  
 unknown\_link\_1 () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 128  
 unknown\_link\_2 () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 128  
 unknown\_link\_2 () (*pyffi.formats.nif.NifFormat.NiParticleMeshesData* property), 158  
 unknown\_link\_2 () (*pyffi.formats.nif.NifFormat.NiParticleSystemComponent* property), 160  
 unknown\_link\_3 () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 128  
 unknown\_link\_6 () (*pyffi.formats.nif.NifFormat.NiPSPlanarCollider* property), 143  
 unknown\_links\_1 () (*pyffi.formats.nif.NifFormat.NiShadowGenerator* property), 173  
 unknown\_matrix () (*pyffi.formats.nif.NifFormat.NiEnvMappedTriShape* property), 119  
 unknown\_node () (*pyffi.formats.nif.NifFormat.NiMeshPSysData* property), 130  
 unknown\_normal () (*pyffi.formats.nif.NifFormat.NiParticleSystemComponent* property), 160  
 unknown\_quat\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_quat\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc* property), 167  
 unknown\_quat\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_quat\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc* property), 167  
 unknown\_quat\_3 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_quat\_3 () (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc* property), 167  
 unknown\_ref\_0 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_ref\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_ref\_2 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 163  
 unknown\_refs\_1 () (*pyffi.formats.nif.NifFormat.NiPhysXProp* property), 165  
 unknown\_refs\_3 () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* property), 164  
 unknown\_short () (*pyffi.formats.nif.NifFormat.HavokColFilter* property), 94  
 unknown\_short () (*pyffi.formats.nif.NifFormat.Ni3dsAnimationNode* property), 105  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiBlendInterpolator* property), 114  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiClodData* property), 116  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator* property), 128  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiPathController* property), 162  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiPathInterpolator* property), 162  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiPlanarCollider* property), 168  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiTextureEffect* property), 179  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* property), 179  
 unknown\_short () (*pyffi.formats.nif.NifFormat.NiUVController* property), 185  
 unknown\_short () (*pyffi.formats.nif.NifFormat.Particle* property), 191  
 unknown\_short () (*pyffi.formats.nif.NifFormat.TexDesc* property), 209  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.bhkCMSDBigTris* property), 214  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* property), 215  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.BSAnimNotes* property), 55  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.BSProceduralLightningController* property), 69  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrlr* property), 66  
 unknown\_short\_1 () (*pyffi.formats.nif.NifFormat.NiAlphaProperty* property), 66

*property*), 107  
 unknown\_short\_1 (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData property*), 113  
 unknown\_short\_1 (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property*), 165  
 unknown\_short\_1 (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property*), 167  
 unknown\_short\_1 (*pyffi.formats.nif.NifFormat.NiPSysData property*), 149  
 unknown\_short\_1 (*pyffi.formats.nif.NifFormat.NiSphericalCollider property*), 177  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.BSBlastNode property*), 55  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.BSDamageStage property*), 56  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.BSDebrisNode property*), 56  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.BSProceduralLightningController property*), 69  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData property*), 113  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiParticleSystem property*), 158  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 160  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPathController property*), 162  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPathInterpolator property*), 162  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property*), 165  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property*), 167  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPlanarCollider property*), 168  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPortal property*), 169  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiPSysData property*), 149  
 unknown\_short\_2 (*pyffi.formats.nif.NifFormat.NiSphericalCollider property*), 177  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.BSProceduralLightningController property*), 69  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty property*), 76  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.MultiTextureElement property*), 104  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData property*), 113  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.NiParticleSystem property*), 158  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 160  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.NiPSPlanarCollider property*), 143  
 unknown\_short\_3 (*pyffi.formats.nif.NifFormat.NiPSysData property*), 149  
 unknown\_short\_5 (*pyffi.formats.nif.NifFormat.BSStripPSysData property*), 74  
 unknown\_shorts (*pyffi.formats.nif.NifFormat.ExtraMeshDataEpicM property*), 88  
 unknown\_shorts (*pyffi.formats.nif.NifFormat.NiClodData property*), 116  
 unknown\_shorts\_1 (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property*), 165  
 unknown\_string (*pyffi.formats.nif.NifFormat.NiArkShaderExtraData property*), 108  
 unknown\_string\_4 (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc property*), 166  
 unknown\_u\_short\_1 (*pyffi.formats.nif.NifFormat.NiScreenElementsData property*), 172  
 unknown\_u\_short\_2 (*pyffi.formats.nif.NifFormat.NiScreenElementsData property*), 172  
 unknown\_u\_short\_3 (*pyffi.formats.nif.NifFormat.NiScreenElementsData*

*property*), 172  
 unknown\_vector() (*pyffi.formats.nif.NifFormat.NiTextureEffect*  
*property*), 179  
 unknown\_vector() (*pyffi.formats.nif.NifFormat.OldSkinData*  
*property*), 190  
 unknown\_vector() (*pyffi.formats.nif.NifFormat.Particle*  
*property*), 191  
 unknown\_vectors() (*pyffi.formats.nif.NifFormat.NiBinaryVoxelData*  
*property*), 113  
 unknown\_byte\_5() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_byte\_9() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_float\_4() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_floats() (*pyffi.formats.nif.NifFormat.bhkSimpleShapePlane*  
*property*), 225  
 unknown\_int\_2() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_int\_6() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_int\_7() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 unknown\_int\_8() (*pyffi.formats.nif.NifFormat.NiShadowGenerator*  
*property*), 173  
 up() (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints*  
*property*), 93  
 update\_a\_b() (*pyffi.formats.nif.NifFormat.bhkLimitedHingeConstraint*  
*method*), 218  
 update\_a\_b() (*pyffi.formats.nif.NifFormat.bhkMalleableConstraint*  
*method*), 219  
 update\_a\_b() (*pyffi.formats.nif.NifFormat.bhkRagdollConstraint*  
*method*), 223  
 update\_a\_b() (*pyffi.formats.nif.NifFormat.LimitedHingeDescriptor*  
*method*), 98  
 update\_a\_b() (*pyffi.formats.nif.NifFormat.RagdollDescriptor*  
*method*), 194  
 update\_bind\_position() (*pyffi.formats.nif.NifFormat.NiGeometry*  
*method*), 123  
 update\_center\_radius() (*pyffi.formats.nif.NifFormat.NiGeometryData*  
*method*), 125  
 update\_delta\_time() (*pyffi.formats.nif.NifFormat.BSPSysStripUpdateModifier*  
*property*), 67  
 update\_mass\_center\_inertia() (*pyffi.formats.nif.NifFormat.bhkRigidBody*  
*method*), 224  
 update\_mopp() (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape*  
*method*), 220  
 update\_mopp\_welding() (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape*  
*method*), 220  
 update\_origin\_scale() (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape*  
*method*), 220  
 update\_skin\_center\_radius() (*pyffi.formats.nif.NifFormat.NiTriBasedGeom*  
*method*), 183  
 update\_skin\_partition() (*pyffi.formats.nif.NifFormat.NiTriBasedGeom*  
*method*), 183  
 update\_skip() (*pyffi.formats.nif.NifFormat.NiPSysBoundUpdateModifier*  
*property*), 147  
 update\_tangent\_space() (*pyffi.formats.cgf.CgfFormat.MeshChunk*  
*method*), 26  
 update\_tangent\_space() (*pyffi.formats.nif.NifFormat.NiTriBasedGeom*  
*method*), 183  
 update\_versions() (*pyffi.formats.cgf.CgfFormat.Data*  
*method*), 13  
 usage() (*pyffi.formats.nif.NifFormat.NiDataStream*  
*property*), 118  
 USAGE\_SHADER\_CONSTANT (*pyffi.formats.nif.NifFormat.DataStreamUsage*  
*attribute*), 86  
 USAGE\_USER (*pyffi.formats.nif.NifFormat.DataStreamUsage*  
*attribute*), 86  
 USAGE\_VERTEX (*pyffi.formats.nif.NifFormat.DataStreamUsage*  
*attribute*), 86  
 USAGE\_VERTEX\_INDEX (*pyffi.formats.nif.NifFormat.DataStreamUsage*  
*attribute*), 86  
 use\_external() (*pyffi.formats.nif.NifFormat.NiCollisionData*  
*property*), 117  
 use\_external() (*pyffi.formats.nif.NifFormat.NiPSysDragFieldModifier*  
*property*), 149  
 use\_external() (*pyffi.formats.nif.NifFormat.NiImage*  
*property*), 125  
 use\_external() (*pyffi.formats.nif.NifFormat.NiSourceTexture*  
*property*), 176  
 use\_external() (*pyffi.formats.nif.NifFormat.TextSource*  
*property*), 210  
 use\_max\_distance() (*pyffi.formats.nif.NifFormat.NiPSysFieldModifier*  
*property*), 151  
 use\_mipmaps() (*pyffi.formats.nif.NifFormat.NiSourceTexture*  
*property*), 176  
 use\_orthographic\_projection() (*pyffi.formats.nif.NifFormat.NiCamera*  
*property*), 116  
 use\_triangle\_points() (*pyffi.formats.nif.NifFormat.NiScreenElementsData*  
*property*), 172  
 used\_vertices() (*pyffi.formats.nif.NifFormat.NiScreenElementsData*

- property), 172
- user\_version (*pyffi.object\_models.FileFormat.Data attribute*), 272
- user\_version() (*pyffi.formats.nif.NifFormat.Data property*), 86
- user\_version\_2() (*pyffi.formats.nif.NifFormat.Data property*), 86
- ushort (*pyffi.formats.cgf.CgfFormat attribute*), 29
- ushort (*pyffi.formats.dds.DdsFormat attribute*), 36
- ushort (*pyffi.formats.egm.EgmFormat attribute*), 40
- ushort (*pyffi.formats.egt.EgtFormat attribute*), 43
- ushort (*pyffi.formats.esp.EspFormat attribute*), 47
- ushort (*pyffi.formats.kfm.KfmFormat attribute*), 51
- ushort (*pyffi.formats.nif.NifFormat attribute*), 227
- ushort (*pyffi.formats.tga.TgaFormat attribute*), 235
- ushort (*pyffi.formats.tri.TriFormat attribute*), 239
- uv\_groups() (*pyffi.formats.nif.NifFormat.NiUVData property*), 186
- uv\_offset() (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty property*), 60
- uv\_offset() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 63
- uv\_offset() (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty property*), 74
- uv\_offset() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty property*), 76
- uv\_quadrants() (*pyffi.formats.nif.NifFormat.NiParticlesData property*), 162
- uv\_scale() (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty property*), 60
- uv\_scale() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 63
- uv\_scale() (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty property*), 74
- uv\_scale() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty property*), 76
- uv\_set() (*pyffi.formats.nif.NifFormat.MultiTextureElement property*), 104
- uv\_set() (*pyffi.formats.nif.NifFormat.TexDesc property*), 209
- V**
- v\_0() (*pyffi.formats.cgf.CgfFormat.Face property*), 14
- v\_1() (*pyffi.formats.cgf.CgfFormat.Face property*), 14
- v\_1() (*pyffi.formats.nif.NifFormat.Triangle property*), 210
- v\_2() (*pyffi.formats.cgf.CgfFormat.Face property*), 14
- v\_2() (*pyffi.formats.nif.NifFormat.Triangle property*), 211
- v\_3() (*pyffi.formats.nif.NifFormat.Triangle property*), 211
- value() (*pyffi.formats.nif.NifFormat.BSValueNode property*), 75
- value() (*pyffi.formats.nif.NifFormat.Key property*), 97
- value() (*pyffi.formats.nif.NifFormat.QuatKey property*), 194
- vector\_1() (*pyffi.formats.nif.NifFormat.NiBezierTriangle4 property*), 112
- vector\_2() (*pyffi.formats.nif.NifFormat.NiBezierTriangle4 property*), 112
- vector\_blocks() (*pyffi.formats.nif.NifFormat.BSDecalPlacementVector property*), 56
- vector\_data() (*pyffi.formats.nif.NifFormat.NiVectorExtraData property*), 186
- vectors() (*pyffi.formats.nif.NifFormat.Morph property*), 102
- velocity() (*pyffi.formats.nif.NifFormat.Particle property*), 191
- VELOCITY\_USE\_DIRECTION (*pyffi.formats.nif.NifFormat.VelocityType attribute*), 212
- VELOCITY\_USE\_NORMALS (*pyffi.formats.nif.NifFormat.VelocityType attribute*), 212
- VELOCITY\_USE\_RANDOM (*pyffi.formats.nif.NifFormat.VelocityType attribute*), 212
- version (*pyffi.object\_models.FileFormat.Data attribute*), 272
- version() (*pyffi.formats.cgf.CgfFormat.ChunkHeader property*), 12
- version() (*pyffi.formats.cgf.CgfFormat.Header property*), 15
- version() (*pyffi.formats.nif.NifFormat.Data property*), 86
- version\_number() (*pyffi.formats.bsa.BsaFormat static method*), 10
- version\_number() (*pyffi.formats.cgf.CgfFormat static method*), 29
- version\_number() (*pyffi.formats.dds.DdsFormat static method*), 36
- version\_number() (*pyffi.formats.egm.EgmFormat static method*), 40
- version\_number() (*pyffi.formats.egt.EgtFormat static method*), 43
- version\_number() (*pyffi.formats.esp.EspFormat static method*), 47
- version\_number() (*pyffi.formats.kfm.KfmFormat static method*), 51
- version\_number() (*pyffi.formats.nif.NifFormat static method*), 228
- version\_number() (*pyffi.formats.tri.TriFormat static method*), 239
- version\_number() (*pyffi.object\_models.FileFormat static method*), 273
- version\_string() (*pyffi.formats.kfm.KfmFormat.HeaderString static method*), 49
- version\_string() (*pyffi.formats.nif.NifFormat.HeaderString static method*), 49

*static method*), 95  
 versions (*pyffi.formats.nif.NifFormat* attribute), 228  
 VERT\_MODE\_SRC\_AMB\_DIF (*pyffi.formats.nif.NifFormat.VertMode* attribute), 212  
 VERT\_MODE\_SRC\_EMISSIVE (*pyffi.formats.nif.NifFormat.VertMode* attribute), 212  
 VERT\_MODE\_SRC\_IGNORE (*pyffi.formats.nif.NifFormat.VertMode* attribute), 212  
 vertex\_counts() (*pyffi.formats.nif.NifFormat.NiTriShapeSkinController* property), 185  
 vertex\_id() (*pyffi.formats.nif.NifFormat.Particle* property), 191  
 vertex\_index() (*pyffi.formats.nif.NifFormat.OldSkinData* property), 190  
 vertex\_indices() (*pyffi.formats.nif.NifFormat.MatchGroup* property), 99  
 vertex\_mode() (*pyffi.formats.nif.NifFormat.NiVertexColorProperty* property), 186  
 vertex\_offset() (*pyffi.formats.nif.NifFormat.Polygon* property), 192  
 vertex\_weight() (*pyffi.formats.nif.NifFormat.OldSkinData* property), 190  
 vertical\_angle() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 160  
 vertical\_direction() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 161  
 vertices() (*pyffi.formats.nif.NifFormat.bhkCMSDChunk* property), 215  
 vertices() (*pyffi.formats.nif.NifFormat.NiPhysXMeshDesc* property), 165  
 vertices() (*pyffi.formats.nif.NifFormat.NiPortal* property), 169  
 vertices\_to\_modify() (*pyffi.formats.tri.TriFormat.ModifierRecord* property), 238  
 viewport\_bottom() (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 viewport\_left() (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 viewport\_right() (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 viewport\_top() (*pyffi.formats.nif.NifFormat.NiCamera* property), 116  
 visibility\_interpolator() (*pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrl* property), 66  
 visibility\_interpolator() (*pyffi.formats.nif.NifFormat.NiPSysEmitterCtrl* property), 150  
 visibility\_keys() (*pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlData* property), 150  
**W**  
 w() (*pyffi.formats.cgf.CgfFormat.Quat* property), 26  
 w() (*pyffi.formats.cgf.CgfFormat.Tangent* property), 29  
 w() (*pyffi.formats.nif.NifFormat.Quaternion* property), 194  
 w() (*pyffi.formats.nif.NifFormat.QuaternionXYZW* property), 194  
 w() (*pyffi.formats.nif.NifFormat.NiRoom* property), 171  
 walk() (*pyffi.object\_models.FileFormat* class method), 273  
 walkData() (*pyffi.object\_models.FileFormat* class method), 273  
 walk\_plane() (*pyffi.formats.nif.NifFormat.NiRoom* property), 171  
 water() (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 202  
 WATER (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 92  
 WATER (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 190  
 WATER (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 202  
 water\_direction() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty* property), 76  
 water\_shader\_flags() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty* property), 76  
 WEAPON (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 92  
 WEAPON (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 190  
 WEAPON (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 202  
 weight() (*pyffi.formats.nif.NifFormat.MorphWeight* property), 102  
 weight() (*pyffi.formats.nif.NifFormat.NiVertWeightsExtraData* property), 186  
 weight() (*pyffi.formats.nif.NifFormat.SkinWeight* property), 198  
 welding\_info() (*pyffi.formats.nif.NifFormat.hkTriangle* property), 227  
 width() (*pyffi.formats.nif.NifFormat.MipMap* property), 102  
 width() (*pyffi.formats.nif.NifFormat.NiPSysBoxEmitter* property), 147  
 width() (*pyffi.formats.nif.NifFormat.NiPSysPlanarCollider* property), 154

width() (*pyffi.formats.nif.NifFormat.NiRawImageData* property), 170

WING (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 92

WING (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 190

world\_center() (*pyffi.formats.nif.NifFormat.NiScreenLODDData* property), 172

world\_radius() (*pyffi.formats.nif.NifFormat.NiScreenLODDData* property), 172

world\_space() (*pyffi.formats.nif.NifFormat.NiParticleSystem* property), 158

WorldMap1 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* attribute), 63

WorldMap2 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* attribute), 63

WorldMap3 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* attribute), 63

WorldMap4 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* attribute), 63

WRAP\_S\_CLAMP\_T (*pyffi.formats.nif.NifFormat.TexClampMode* attribute), 208

WRAP\_S\_WRAP\_T (*pyffi.formats.nif.NifFormat.TexClampMode* attribute), 208

write() (*pyffi.formats.bsa.BsaFormat.BZString* method), 8

write() (*pyffi.formats.bsa.BsaFormat.FileVersion* method), 8

write() (*pyffi.formats.bsa.BsaFormat.Header* method), 9

write() (*pyffi.formats.bsa.BsaFormat.ZString* method), 10

write() (*pyffi.formats.cgf.CgfFormat.Data* method), 13

write() (*pyffi.formats.cgf.CgfFormat.FileSignature* method), 14

write() (*pyffi.formats.cgf.CgfFormat.Ref* method), 27

write() (*pyffi.formats.cgf.CgfFormat.SizedString* method), 28

write() (*pyffi.formats.dae.DaeFormat.Data* method), 33

write() (*pyffi.formats.dds.DdsFormat.Data* method), 35

write() (*pyffi.formats.dds.DdsFormat.HeaderString* method), 35

write() (*pyffi.formats.egm.EgmFormat.Data* method), 38

write() (*pyffi.formats.egm.EgmFormat.FileSignature* method), 39

write() (*pyffi.formats.egm.EgmFormat.FileVersion* method), 39

write() (*pyffi.formats.egt.EgtFormat.FileSignature* method), 42

write() (*pyffi.formats.egt.EgtFormat.FileVersion* method), 42

write() (*pyffi.formats.egt.EgtFormat.Header* method), 43

write() (*pyffi.formats.esp.EspFormat.Data* method), 45

write() (*pyffi.formats.esp.EspFormat.GRUP* method), 46

write() (*pyffi.formats.esp.EspFormat.Record* method), 46

write() (*pyffi.formats.esp.EspFormat.ZString* method), 47

write() (*pyffi.formats.kfm.KfmFormat.Data* method), 49

write() (*pyffi.formats.kfm.KfmFormat.HeaderString* method), 50

write() (*pyffi.formats.kfm.KfmFormat.SizedString* method), 51

write() (*pyffi.formats.nif.NifFormat.bool* method), 226

write() (*pyffi.formats.nif.NifFormat.ByteArray* method), 228

write() (*pyffi.formats.nif.NifFormat.ByteMatrix* method), 79

write() (*pyffi.formats.nif.NifFormat.Data* method), 86

write() (*pyffi.formats.nif.NifFormat.FileVersion* method), 92

write() (*pyffi.formats.nif.NifFormat.Footer* method), 93

write() (*pyffi.formats.nif.NifFormat.HeaderString* method), 95

write() (*pyffi.formats.nif.NifFormat.LineString* method), 99

write() (*pyffi.formats.nif.NifFormat.Ref* method), 195

write() (*pyffi.formats.nif.NifFormat.ShortString* method), 196

write() (*pyffi.formats.nif.NifFormat.SizedString* method), 197

write() (*pyffi.formats.nif.NifFormat.string* method), 227

write() (*pyffi.formats.tga.TgaFormat.Data* method), 234

write() (*pyffi.formats.tga.TgaFormat.FooterString* method), 234

write() (*pyffi.formats.tri.TriFormat.FileSignature* method), 237

write() (*pyffi.formats.tri.TriFormat.FileVersion* method), 237

write() (*pyffi.formats.tri.TriFormat.Header* method), 238

write() (*pyffi.formats.tri.TriFormat.SizedStringZ* method), 239

write() (*pyffi.object\_models.FileFormat.Data* method), 272

write() (*pyffi.spells.Toaster* method), 271

writepatch() (*pyffi.spells.Toaster* method), 271

**X**

- `x()` (`pyffi.formats.cgf.CgfFormat.Quat` property), 26
- `x()` (`pyffi.formats.nif.NifFormat.Quaternion` property), 194
- `x()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` property), 194
- `x_axis()` (`pyffi.formats.nif.NifFormat.NiPSysPlanarCollider` property), 154
- `xml_alias` (`pyffi.formats.nif.NifFormat` attribute), 228
- `xml_bit_struct` (`pyffi.formats.nif.NifFormat` attribute), 228
- `xml_enum` (`pyffi.formats.nif.NifFormat` attribute), 228
- `xml_file_name` (`pyffi.formats.nif.NifFormat` attribute), 228
- `xml_file_path` (`pyffi.formats.nif.NifFormat` attribute), 228
- `xml_struct` (`pyffi.formats.nif.NifFormat` attribute), 228
- `XYZ_ROTATION_KEY` (`pyffi.formats.nif.NifFormat.KeyType` attribute), 97

**Y**

- `y()` (`pyffi.formats.cgf.CgfFormat.Quat` property), 26
- `y()` (`pyffi.formats.nif.NifFormat.Quaternion` property), 194
- `y()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` property), 194
- `y_axis()` (`pyffi.formats.nif.NifFormat.NiPSysPlanarCollider` property), 154

**Z**

- `z()` (`pyffi.formats.cgf.CgfFormat.Quat` property), 26
- `z()` (`pyffi.formats.nif.NifFormat.Quaternion` property), 194
- `z()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` property), 194
- `z_fail_action()` (`pyffi.formats.nif.NifFormat.NiStencilProperty` property), 177
- `ZCOMP_ALWAYS` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_GREATER` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_GREATER_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_LESS` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_LESS_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212
- `ZCOMP_NEVER` (`pyffi.formats.nif.NifFormat.ZCompareMode` attribute), 212