

libgexf

0.1

Generated by Doxygen 1.5.6

Tue Jul 21 00:25:02 2009

Contents

1	Class Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	libgexf::AbstractIter Class Reference	7
4.1.1	Detailed Description	7
4.1.2	Member Function Documentation	8
4.1.2.1	begin	8
4.1.2.2	hasNext	8
4.1.2.3	next	8
4.2	libgexf::AbstractParser Class Reference	9
4.2.1	Detailed Description	9
4.2.2	Member Function Documentation	9
4.2.2.1	bind	9
4.2.2.2	processNode	9
4.3	libgexf::AttributeIter Class Reference	11
4.3.1	Detailed Description	11
4.3.2	Constructor & Destructor Documentation	11
4.3.2.1	AttributeIter	11
4.3.3	Member Function Documentation	12
4.3.3.1	begin	12
4.3.3.2	hasNext	12
4.3.3.3	next	12

4.3.3.4	currentTitle	12
4.3.3.5	currentType	13
4.4	libgexf::AttValueIter Class Reference	14
4.4.1	Detailed Description	14
4.4.2	Constructor & Destructor Documentation	15
4.4.2.1	AttValueIter	15
4.4.3	Member Function Documentation	15
4.4.3.1	begin	15
4.4.3.2	hasNext	15
4.4.3.3	next	15
4.4.3.4	currentValue	16
4.4.3.5	currentName	16
4.5	libgexf::Conv Class Reference	17
4.5.1	Detailed Description	17
4.6	libgexf::Data Class Reference	18
4.6.1	Detailed Description	19
4.6.2	Member Function Documentation	19
4.6.2.1	getLabel	19
4.6.2.2	hasLabel	20
4.6.2.3	setLabel	20
4.6.2.4	addNodeAttributeColumn	20
4.6.2.5	addEdgeAttributeColumn	20
4.6.2.6	setNodeAttributeDefault	20
4.6.2.7	setEdgeAttributeDefault	21
4.6.2.8	setNodeValue	21
4.6.2.9	setEdgeValue	21
4.6.2.10	getNodeAttributeColumn	21
4.6.2.11	getEdgeAttributeColumn	21
4.6.2.12	getNodeAttribute	22
4.6.2.13	getEdgeAttribute	22
4.6.2.14	getNodeAttributeRow	22
4.6.2.15	getEdgeAttributeRow	22
4.6.2.16	getNodeAttributeDefault	23
4.6.2.17	getEdgeAttributeDefault	23
4.6.2.18	hasNodeAttributeDefault	23
4.6.2.19	hasEdgeAttributeDefault	23

4.6.2.20	clearNodeAttributes	23
4.6.2.21	clearEdgeAttributes	24
4.6.3	Friends And Related Function Documentation	24
4.6.3.1	AttributeIter	24
4.6.3.2	AttValueIter	24
4.7	libgexf::DirectedGraph Class Reference	25
4.7.1	Detailed Description	26
4.7.2	Member Function Documentation	26
4.7.2.1	removeInEdges	26
4.7.2.2	removeOutEdges	26
4.7.2.3	getInEdges	26
4.7.2.4	getOutEdges	26
4.7.2.5	getSuccessors	27
4.7.2.6	getPredecessors	27
4.7.2.7	getInDegree	27
4.7.2.8	getOutDegree	27
4.7.2.9	isSuccessor	28
4.7.2.10	isPredecessor	28
4.8	libgexf::EdgeIter Class Reference	29
4.8.1	Detailed Description	29
4.8.2	Constructor & Destructor Documentation	29
4.8.2.1	EdgeIter	29
4.8.3	Member Function Documentation	30
4.8.3.1	begin	30
4.8.3.2	hasNext	30
4.8.3.3	next	30
4.8.3.4	currentSource	30
4.8.3.5	currentTarget	30
4.8.3.6	currentProperty	31
4.9	libgexf::FileReader Class Reference	32
4.9.1	Detailed Description	32
4.9.2	Constructor & Destructor Documentation	32
4.9.2.1	FileReader	32
4.9.3	Member Function Documentation	33
4.9.3.1	getGEXFCopy	33
4.9.3.2	init	33

4.10	libgexf::FileReaderException Class Reference	34
4.10.1	Detailed Description	34
4.11	libgexf::FileWriter Class Reference	35
4.11.1	Detailed Description	35
4.11.2	Constructor & Destructor Documentation	35
4.11.2.1	FileWriter	35
4.11.3	Member Function Documentation	36
4.11.3.1	getGEXFCopy	36
4.11.3.2	init	36
4.12	libgexf::FileWriterException Class Reference	37
4.12.1	Detailed Description	37
4.13	libgexf::GEXF Class Reference	38
4.13.1	Detailed Description	39
4.13.2	Member Function Documentation	39
4.13.2.1	getUndirectedGraph	39
4.13.2.2	getDirectedGraph	39
4.13.2.3	getData	39
4.13.2.4	getMetaData	39
4.13.2.5	setGraphType	40
4.13.2.6	getGraphType	40
4.13.2.7	checkIntegrity	40
4.13.3	Member Data Documentation	40
4.13.3.1	_graph	40
4.13.3.2	_type	40
4.13.3.3	_data	40
4.13.3.4	_meta	40
4.14	libgexf::GexfParser Class Reference	41
4.14.1	Detailed Description	41
4.14.2	Member Function Documentation	41
4.14.2.1	bind	41
4.14.2.2	processNode	41
4.15	libgexf::Graph Class Reference	43
4.15.1	Detailed Description	45
4.15.2	Member Function Documentation	45
4.15.2.1	addNode	45
4.15.2.2	addEdge	45

4.15.2.3	removeNode	45
4.15.2.4	removeEdge	46
4.15.2.5	containsNode	46
4.15.2.6	containsEdge	46
4.15.2.7	getNodes	46
4.15.2.8	getEdges	46
4.15.2.9	getNeighbors	47
4.15.2.10	getNodeCount	47
4.15.2.11	getEdgeCount	47
4.15.2.12	getDegree	47
4.15.2.13	clearEdges	47
4.15.2.14	readLock	48
4.15.2.15	writeLock	48
4.15.2.16	getInternalEdgeCount	48
4.15.3	Member Data Documentation	48
4.15.3.1	_nodes	48
4.15.3.2	_edges	48
4.15.3.3	_reverse_edges	48
4.15.3.4	_bloom_edges	48
4.15.3.5	_edges_properties	48
4.15.3.6	_rlock_count	49
4.15.3.7	_lock_flag	49
4.16	libgexf::LegacyParser Class Reference	50
4.16.1	Detailed Description	50
4.16.2	Member Function Documentation	50
4.16.2.1	bind	50
4.16.2.2	processNode	50
4.17	libgexf::MetaData Class Reference	52
4.17.1	Detailed Description	53
4.18	libgexf::NodeIter Class Reference	54
4.18.1	Detailed Description	54
4.18.2	Constructor & Destructor Documentation	54
4.18.2.1	NodeIter	54
4.18.3	Member Function Documentation	54
4.18.3.1	begin	54
4.18.3.2	hasNext	55

4.18.3.3	next	55
4.19	libgexf::ReadLockException Class Reference	56
4.19.1	Detailed Description	56
4.20	libgexf::UndirectedGraph Class Reference	57
4.20.1	Detailed Description	57
4.21	libgexf::WriteLockException Class Reference	58
4.21.1	Detailed Description	58
5	File Documentation	59
5.1	abstractiter.h File Reference	59
5.1.1	Detailed Description	59
5.2	abstractparser.h File Reference	60
5.2.1	Detailed Description	60
5.3	attributeiter.cpp File Reference	61
5.3.1	Detailed Description	61
5.4	attributeiter.h File Reference	62
5.4.1	Detailed Description	62
5.5	attvalueiter.cpp File Reference	63
5.5.1	Detailed Description	63
5.6	attvalueiter.h File Reference	64
5.6.1	Detailed Description	64
5.7	conv.cpp File Reference	65
5.7.1	Detailed Description	65
5.8	conv.h File Reference	66
5.8.1	Detailed Description	66
5.9	data.cpp File Reference	67
5.9.1	Detailed Description	67
5.10	data.h File Reference	68
5.10.1	Detailed Description	68
5.11	directedgraph.cpp File Reference	69
5.11.1	Detailed Description	69
5.12	directedgraph.h File Reference	70
5.12.1	Detailed Description	70
5.13	edgeiter.cpp File Reference	71
5.13.1	Detailed Description	71
5.14	edgeiter.h File Reference	72
5.14.1	Detailed Description	72

5.15	exceptions.h File Reference	73
5.15.1	Detailed Description	73
5.16	filereader.cpp File Reference	74
5.16.1	Detailed Description	74
5.17	filereader.h File Reference	75
5.17.1	Detailed Description	75
5.18	filewriter.cpp File Reference	76
5.18.1	Detailed Description	76
5.19	filewriter.h File Reference	77
5.19.1	Detailed Description	77
5.20	gexf.cpp File Reference	78
5.20.1	Detailed Description	78
5.21	gexf.h File Reference	79
5.21.1	Detailed Description	79
5.22	gexfparser.cpp File Reference	80
5.22.1	Detailed Description	80
5.23	gexfparser.h File Reference	81
5.23.1	Detailed Description	81
5.24	graph.cpp File Reference	82
5.24.1	Detailed Description	82
5.25	graph.h File Reference	83
5.25.1	Detailed Description	83
5.26	legacyparser.cpp File Reference	84
5.26.1	Detailed Description	84
5.27	legacyparser.h File Reference	85
5.27.1	Detailed Description	85
5.28	libgexf.h File Reference	86
5.28.1	Detailed Description	86
5.29	metadata.cpp File Reference	87
5.29.1	Detailed Description	87
5.30	metadata.h File Reference	88
5.30.1	Detailed Description	88
5.31	nodeiter.cpp File Reference	89
5.31.1	Detailed Description	89
5.32	nodeiter.h File Reference	90
5.32.1	Detailed Description	90

5.33	typedefs.h File Reference	91
5.33.1	Detailed Description	91
5.34	undirectedgraph.cpp File Reference	92
5.34.1	Detailed Description	92
5.35	undirectedgraph.h File Reference	93
5.35.1	Detailed Description	93

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

libgexf::AbstractIter	7
libgexf::AttributeIter	11
libgexf::AttValueIter	14
libgexf::EdgeIter	29
libgexf::NodeIter	54
libgexf::AbstractParser	9
libgexf::GexfParser	41
libgexf::LegacyParser	50
libgexf::Conv	17
libgexf::Data	18
libgexf::FileReader	32
libgexf::FileReaderException	34
libgexf::FileWriter	35
libgexf::FileWriterException	37
libgexf::GEXF	38
libgexf::Graph	43
libgexf::DirectedGraph	25
libgexf::UndirectedGraph	57
libgexf::MetaData	52
libgexf::ReadLockException	56
libgexf::WriteLockException	58

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

libgexf::AbstractIter (Iterator Interface)	7
libgexf::AbstractParser (Parser Interface)	9
libgexf::AttributeIter (Iterator on attributes)	11
libgexf::AttValueIter (Iterator on attribute values)	14
libgexf::Conv (Utility class for transforming data)	17
libgexf::Data (Associated data and attributes on nodes and edges)	18
libgexf::DirectedGraph (Interpretation of the topology structure as a directed graph)	25
libgexf::EdgeIter (Iterator on edges)	29
libgexf::FileReader (Read a GEXF file)	32
libgexf::FileReaderException (Exception occuring when reading a file)	34
libgexf::FileWriter (Write a GEXF file)	35
libgexf::FileWriterException (Exception occuring when writing a file)	37
libgexf::GEXF (GEXF class, just a container)	38
libgexf::GexfParser (Parse a GEXF file)	41
libgexf::Graph (Topology structure of the graph)	43
libgexf::LegacyParser (Parse an old GEXF file (1.0))	50
libgexf::MetaData (Associated meta data and attributes on the graph)	52
libgexf::NodeIter (Iterator on nodes)	54
libgexf::ReadLockException (Exception occuring on a read-lock)	56
libgexf::UndirectedGraph (Interpretation of the topology structure as a undirected graph)	57
libgexf::WriteLockException (Exception occuring on a write-lock)	58

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

abstractiter.h	59
abstractparser.h	60
attributeiter.cpp	61
attributeiter.h	62
attvalueiter.cpp	63
attvalueiter.h	64
conv.cpp	65
conv.h	66
data.cpp	67
data.h	68
directedgraph.cpp	69
directedgraph.h	70
edgeiter.cpp	71
edgeiter.h	72
exceptions.h	73
filereader.cpp	74
filereader.h	75
filewriter.cpp	76
filewriter.h	77
gexf.cpp	78
gexf.h	79
gexfparser.cpp	80
gexfparser.h	81
graph.cpp	82
graph.h	83
legacyparser.cpp	84
legacyparser.h	85
libgexf.h	86
metadata.cpp	87
metadata.h	88
nodeiter.cpp	89
nodeiter.h	90
typedefs.h	91

undirectedgraph.cpp	92
undirectedgraph.h	93

Chapter 4

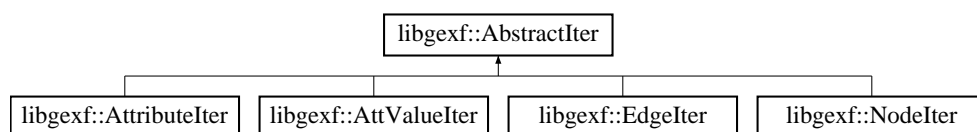
Class Documentation

4.1 libgexf::AbstractIter Class Reference

Iterator Interface.

```
#include <abstractiter.h>
```

Inheritance diagram for libgexf::AbstractIter::



Public Member Functions

- virtual [AbstractIter](#) * [begin](#) ()=0

Init.

- virtual bool [hasNext](#) () const =0

Test next element in collection.

- virtual libgexf::t_id [next](#) ()=0

Iterate.

4.1.1 Detailed Description

Iterator Interface.

4.1.2 Member Function Documentation

4.1.2.1 `virtual AbstractIter* libgexf::AbstractIter::begin ()` [pure virtual]

Init.

Initialize (or re-initialize) the iterator.

Returns:

an instance of the iterator

Implemented in [libgexf::AttributeIter](#), [libgexf::AttValueIter](#), [libgexf::EdgeIter](#), and [libgexf::NodeIter](#).

4.1.2.2 `virtual bool libgexf::AbstractIter::hasNext () const` [pure virtual]

Test next element in collection.

Verify if another element exists.

Returns:

true if another element exists, false otherwise.

Implemented in [libgexf::AttributeIter](#), [libgexf::AttValueIter](#), [libgexf::EdgeIter](#), and [libgexf::NodeIter](#).

4.1.2.3 `virtual libgexf::t_id libgexf::AbstractIter::next ()` [pure virtual]

Iterate.

Get next element in collection.

Returns:

The element ID.

Implemented in [libgexf::AttributeIter](#), [libgexf::AttValueIter](#), [libgexf::EdgeIter](#), and [libgexf::NodeIter](#).

The documentation for this class was generated from the following file:

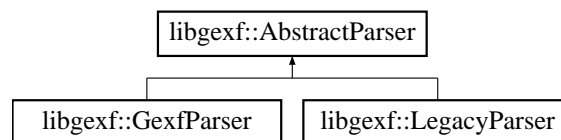
- [abstractiter.h](#)

4.2 libgexf::AbstractParser Class Reference

Parser Interface.

```
#include <abstractparser.h>
```

Inheritance diagram for libgexf::AbstractParser::



Public Member Functions

- virtual void [bind](#) (libgexf::GEXF *gexf)=0
Bind itself to a [GEXF](#) instance.
- virtual void [processNode](#) (xmlTextReaderPtr reader, const xmlChar *name)=0
Start a process on an XML element.

4.2.1 Detailed Description

Parser Interface.

4.2.2 Member Function Documentation

4.2.2.1 virtual void libgexf::AbstractParser::bind (libgexf::GEXF * *gexf*) [pure virtual]

Bind itself to a [GEXF](#) instance.

Parameters:

gexf : Reference to a [GEXF](#) object

Implemented in [libgexf::GexfParser](#), and [libgexf::LegacyParser](#).

4.2.2.2 virtual void libgexf::AbstractParser::processNode (xmlTextReaderPtr *reader*, const xmlChar * *name*) [pure virtual]

Start a process on an XML element.

Call the right processing method.

Parameters:

reader : Reference to the libxml TextReader instance

name : Name of the XML element to process

Implemented in [libgexf::GexfParser](#), and [libgexf::LegacyParser](#).

The documentation for this class was generated from the following file:

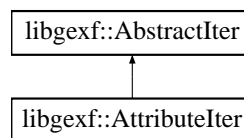
- [abstractparser.h](#)

4.3 libgexf::AttributeIter Class Reference

Iterator on attributes.

```
#include <attributeiter.h>
```

Inheritance diagram for libgexf::AttributeIter::



Public Types

- enum [Type](#) { **NODE**, **EDGE** }
Possible type of element.

Public Member Functions

- [AttributeIter](#) (const [Data](#) *d, const [AttributeIter::Type](#) t)
Constructor.
- [AttributeIter](#) * [begin](#) ()
Init.
- bool [hasNext](#) () const
Test next element in collection.
- libgexf::t_id [next](#) ()
Iterate.
- std::string [currentTitle](#) () const
Get title.
- libgexf::t_attr_type [currentType](#) () const
Get type.

4.3.1 Detailed Description

Iterator on attributes.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 libgexf::AttributeIter::AttributeIter (const Data * d, const AttributeIter::Type t)

Constructor.

Parameters:

d : Reference to the [Data](#) object

t : NODE or EDGE

4.3.3 Member Function Documentation

4.3.3.1 `AttributeIter * libgexf::AttributeIter::begin ()` [virtual]

Init.

Initialize (or re-initialize) the iterator.

Returns:

an instance of the iterator

Implements [libgexf::AbstractIter](#).

4.3.3.2 `bool libgexf::AttributeIter::hasNext () const` [virtual]

Test next element in collection.

Verify if another element exists.

Returns:

true if another element exists, false otherwise.

Implements [libgexf::AbstractIter](#).

4.3.3.3 `t_id libgexf::AttributeIter::next ()` [virtual]

Iterate.

Get next element in collection.

Returns:

The attribute ID.

Implements [libgexf::AbstractIter](#).

4.3.3.4 `string libgexf::AttributeIter::currentTitle () const`

Get title.

Returns:

The attribute title

4.3.3.5 `t_attr_type libgexf::AttributeIter::currentType () const`

Get type.

Returns:

The attribute type

The documentation for this class was generated from the following files:

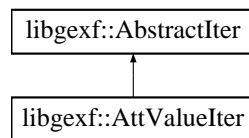
- [attributeiter.h](#)
- [attributeiter.cpp](#)

4.4 libgexf::AttValueIter Class Reference

Iterator on attribute values.

```
#include <attvalueiter.h>
```

Inheritance diagram for libgexf::AttValueIter::



Public Types

- enum [Type](#) { **NODE**, **EDGE** }

Possible type of element.

Public Member Functions

- [AttValueIter](#) (const [Data](#) *d, const libgexf::t_id id, const [AttValueIter::Type](#) t)

Constructor.

- [AttValueIter](#) * [begin](#) ()

Init.

- bool [hasNext](#) () const

Test next element in collection.

- libgexf::t_id [next](#) ()

Iterate.

- std::string [currentValue](#) () const

Get value.

- std::string [currentName](#) () const

Get attribute name.

4.4.1 Detailed Description

Iterator on attribute values.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 libgexf::AttValueIter::AttValueIter (const Data * *d*, const libgexf::t_id *id*, const AttValueIter::Type *t*)

Constructor.

Parameters:

d : Reference to the [Data](#) object

id : Node or edge ID

t : NODE or EDGE

4.4.3 Member Function Documentation

4.4.3.1 AttValueIter * libgexf::AttValueIter::begin () [virtual]

Init.

Initialize (or re-initialize) the iterator.

Returns:

an instance of the iterator

Implements [libgexf::AbstractIter](#).

4.4.3.2 bool libgexf::AttValueIter::hasNext () const [virtual]

Test next element in collection.

Verify if another element exists.

Returns:

true if another element exists, false otherwise.

Implements [libgexf::AbstractIter](#).

4.4.3.3 t_id libgexf::AttValueIter::next () [virtual]

Iterate.

Get next element in collection.

Returns:

The attribute ID.

Implements [libgexf::AbstractIter](#).

4.4.3.4 string libgexf::AttValueIter::currentValue () const

Get value.

Returns:

The attribute value of the node/edge

4.4.3.5 string libgexf::AttValueIter::currentName () const

Get attribute name.

Returns:

The attribute name

The documentation for this class was generated from the following files:

- [attvalueiter.h](#)
- [attvalueiter.cpp](#)

4.5 libgexf::Conv Class Reference

Utility class for transforming data.

```
#include <conv.h>
```

Public Member Functions

- **Conv** (const [Conv](#) &orig)

Static Public Member Functions

- static libgexf::t_id **xmlCharToId** (const xmlChar *str)
- static libgexf::t_id **strToId** (const std::string str)
- static std::string **xmlCharToStr** (const xmlChar *str)
- static unsigned int **xmlCharToUnsignedInt** (const xmlChar *str)
- static std::string **idToStr** (const libgexf::t_id id)
- static std::string **unsignedIntToStr** (const unsigned int i)
- static unsigned int **strToUnsignedInt** (const std::string str)
- static std::string **edgeTypeToStr** (const libgexf::t_edge_type t)
- static std::string **attrTypeToStr** (const libgexf::t_attr_type t)

4.5.1 Detailed Description

Utility class for transforming data.

The documentation for this class was generated from the following files:

- [conv.h](#)
- [conv.cpp](#)

4.6 libgexf::Data Class Reference

Associated data and attributes on nodes and edges.

```
#include <data.h>
```

Public Member Functions

- [Data](#) (const [Data](#) &orig)
Copy constructor.
- std::string [getLabel](#) (const libgexf::t_id node_id) const
Get node label.
- bool [hasLabel](#) (const libgexf::t_id node_id) const
Check if the node label exists.
- void [setLabel](#) (const libgexf::t_id node_id, const std::string label)
Set node label.
- void [addNodeAttributeColumn](#) (const libgexf::t_id id, const std::string title, const libgexf::t_attr_type type)
Add a node attribute column.
- void [addEdgeAttributeColumn](#) (const libgexf::t_id id, const std::string title, const libgexf::t_attr_type type)
Add an edge attribute column.
- void [setNodeAttributeDefault](#) (const libgexf::t_id attr_id, const std::string default_value)
Set the default value of a node attribute.
- void [setEdgeAttributeDefault](#) (const libgexf::t_id attr_id, const std::string default_value)
Set the default value of an edge attribute.
- void [setNodeValue](#) (const libgexf::t_id node_id, const libgexf::t_id attr_id, const std::string value)
Set the node value of an attribute.
- void [setEdgeValue](#) (const libgexf::t_id edge_id, const libgexf::t_id attr_id, const std::string value)
Set the edge value of an attribute.
- libgexf::AttributeIter * [getNodeAttributeColumn](#) () const
Get an iterator on the node attribute column.
- libgexf::AttributeIter * [getEdgeAttributeColumn](#) () const
Get an iterator on the edge attribute column.
- std::string [getNodeAttribute](#) (const libgexf::t_id node_id, const libgexf::t_id attr_id) const
Get the node attribute value.
- std::string [getEdgeAttribute](#) (const libgexf::t_id edge_id, const libgexf::t_id attr_id) const

Get the edge attribute value.

- [libgexf::AttValueIter](#) * [getNodeAttributeRow](#) (const libgexf::t_id node_id) const
Get an iterator on the node attribute row.
- [libgexf::AttValueIter](#) * [getEdgeAttributeRow](#) (const libgexf::t_id edge_id) const
Get an iterator on the edge attribute row.
- std::string [getNodeAttributeDefault](#) (const libgexf::t_id attr_id) const
Get the default value of a node attribute.
- std::string [getEdgeAttributeDefault](#) (const libgexf::t_id attr_id) const
Get the default value of an edge attribute.
- bool [hasNodeAttributeDefault](#) (const libgexf::t_id attr_id) const
Check if a node attribute has a default value.
- bool [hasEdgeAttributeDefault](#) (const libgexf::t_id attr_id) const
Check if an edge attribute has a default value.
- void [clearNodeAttributes](#) (const libgexf::t_id node_id)
Delete all attribute values for a node.
- void [clearEdgeAttributes](#) (const libgexf::t_id edge_id)
Delete all attribute values for an edge.
- void [clear](#) ()
Clear all attributes (columns an rows).
- void [clearEdgesAttributes](#) ()
Clear edge attributes (columns an rows).

Friends

- class [AttributeIter](#)
- class [AttValueIter](#)
- std::ostream & [operator<<](#) (std::ostream &os, const [Data](#) &o)

4.6.1 Detailed Description

Associated data and attributes on nodes and edges.

4.6.2 Member Function Documentation

4.6.2.1 string libgexf::Data::getLabel (const libgexf::t_id node_id) const

Get node label.

Parameters:

node_id : node ID

4.6.2.2 bool libgexf::Data::hasLabel (const libgexf::t_id *node_id*) const

Check if the node label exists.

Parameters:

node_id : node ID

4.6.2.3 void libgexf::Data::setLabel (const libgexf::t_id *node_id*, const std::string *label*)

Set node label.

Parameters:

node_id : node ID

label : node label

4.6.2.4 void libgexf::Data::addNodeAttributeColumn (const libgexf::t_id *id*, const std::string *title*, const libgexf::t_attr_type *type*)

Add a node attribute column.

Parameters:

id : attribute ID

title : name of the node attribute

type : type of attribute (integer, double, float, boolean, string or list-string)

4.6.2.5 void libgexf::Data::addEdgeAttributeColumn (const libgexf::t_id *id*, const std::string *title*, const libgexf::t_attr_type *type*)

Add an edge attribute column.

Parameters:

id : attribute ID

title : name of the edge attribute

type : type of attribute (integer, double, float, boolean, string or list-string)

4.6.2.6 void libgexf::Data::setNodeAttributeDefault (const libgexf::t_id *attr_id*, const std::string *default_value*)

Set the default value of a node attribute.

Parameters:

attr_id : attribute ID

default_value : default value

4.6.2.7 void libgexf::Data::setEdgeAttributeDefault (const libgexf::t_id *attr_id*, const std::string *default_value*)

Set the default value of an edge attribute.

Parameters:

attr_id : attribute ID
default_value : default value

4.6.2.8 void libgexf::Data::setNodeValue (const libgexf::t_id *node_id*, const libgexf::t_id *attr_id*, const std::string *value*)

Set the node value of an attribute.

Parameters:

node_id : node ID
attr_id : attribute ID
value : node value

4.6.2.9 void libgexf::Data::setEdgeValue (const libgexf::t_id *edge_id*, const libgexf::t_id *attr_id*, const std::string *value*)

Set the edge value of an attribute.

Parameters:

edge_id : edge ID
attr_id : attribute ID
value : edge value

4.6.2.10 AttributeIter * libgexf::Data::getNodeAttributeColumn () const

Get an iterator on the node attribute column.

Returns:

Attribute iterator instance

4.6.2.11 AttributeIter * libgexf::Data::getEdgeAttributeColumn () const

Get an iterator on the edge attribute column.

Returns:

Attribute iterator instance

4.6.2.12 `string libgexf::Data::getNodeAttribute (const libgexf::t_id node_id, const libgexf::t_id attr_id) const`

Get the node attribute value.

Parameters:

node_id : node ID
attr_id : attribute ID

Returns:

node attribute value

4.6.2.13 `string libgexf::Data::getEdgeAttribute (const libgexf::t_id edge_id, const libgexf::t_id attr_id) const`

Get the edge attribute value.

Parameters:

edge_id : edge ID
attr_id : attribute ID

Returns:

edge attribute value

4.6.2.14 `AttValueIter * libgexf::Data::getNodeAttributeRow (const libgexf::t_id node_id) const`

Get an iterator on the node attribute row.

Parameters:

node_id : node ID

Returns:

iterator instance on attribute values

4.6.2.15 `AttValueIter * libgexf::Data::getEdgeAttributeRow (const libgexf::t_id edge_id) const`

Get an iterator on the edge attribute row.

Parameters:

edge_id : edge ID

Returns:

iterator instance on attribute values

4.6.2.16 string libgexf::Data::getNodeAttributeDefault (const libgexf::t_id attr_id) const

Get the default value of a node attribute.

Parameters:

attr_id : attribute ID

Returns:

default value

4.6.2.17 string libgexf::Data::getEdgeAttributeDefault (const libgexf::t_id attr_id) const

Get the default value of an edge attribute.

Parameters:

attr_id : attribute ID

Returns:

default value

4.6.2.18 bool libgexf::Data::hasNodeAttributeDefault (const libgexf::t_id attr_id) const

Check if a node attribute has a default value.

Parameters:

attr_id : attribute ID

Returns:

true if the default value exists, false otherwise

4.6.2.19 bool libgexf::Data::hasEdgeAttributeDefault (const libgexf::t_id attr_id) const

Check if an edge attribute has a default value.

Parameters:

attr_id : attribute ID

Returns:

true if the default value exists, false otherwise

4.6.2.20 void libgexf::Data::clearNodeAttributes (const libgexf::t_id node_id)

Delete all attribute values for a node.

Parameters:

node_id : node ID

4.6.2.21 void libgexf::Data::clearEdgeAttributes (const libgexf::t_id *edge_id*)

Delete all attribute values for an edge.

Parameters:

edge_id : edge ID

4.6.3 Friends And Related Function Documentation

4.6.3.1 friend class AttributeIter [friend]

[AttributeIter](#)

4.6.3.2 friend class AttValueIter [friend]

[AttValueIter](#)

The documentation for this class was generated from the following files:

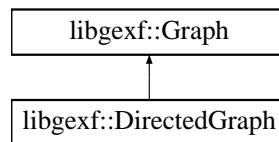
- [data.h](#)
- [data.cpp](#)

4.7 libgexf::DirectedGraph Class Reference

Interpretation of the topology structure as a directed graph.

```
#include <directedgraph.h>
```

Inheritance diagram for libgexf::DirectedGraph::



Public Member Functions

- [DirectedGraph](#) (const [DirectedGraph](#) &orig)
Copy constructor.
- void [removeInEdges](#) (const libgexf::t_id target_id)
Remove incoming edges from a node.
- void [removeOutEdges](#) (const libgexf::t_id source_id)
Remove outgoing edges from a node.
- std::set< libgexf::t_id > [getInEdges](#) (const libgexf::t_id node_id) const
Get incoming edges from a node.
- std::set< libgexf::t_id > [getOutEdges](#) (const libgexf::t_id node_id) const
Get outgoing edges from a node.
- std::set< libgexf::t_id > [getSuccessors](#) (const libgexf::t_id node_id) const
Get node successors.
- std::set< libgexf::t_id > [getPredecessors](#) (const libgexf::t_id node_id) const
Get node predecessors.
- unsigned int [getInDegree](#) (const libgexf::t_id node_id) const
Get indegree value.
- unsigned int [getOutDegree](#) (const libgexf::t_id node_id) const
Get outdegree value.
- bool [isSuccessor](#) (const libgexf::t_id node_id, const libgexf::t_id successor_id) const
Test a possible successor.
- bool [isPredecessor](#) (const libgexf::t_id node_id, const libgexf::t_id predecessor_id) const
Test a possible predecessor.

4.7.1 Detailed Description

Interpretation of the topology structure as a directed graph.

4.7.2 Member Function Documentation

4.7.2.1 `void libgexf::DirectedGraph::removeInEdges (const libgexf::t_id target_id)`

Remove incoming edges from a node.

Parameters:

target_id : node ID

4.7.2.2 `void libgexf::DirectedGraph::removeOutEdges (const libgexf::t_id source_id)`

Remove outgoing edges from a node.

Parameters:

source_id : node ID

4.7.2.3 `std::set< t_id > libgexf::DirectedGraph::getInEdges (const libgexf::t_id node_id) const`

Get incoming edges from a node.

Parameters:

node_id : node ID

Returns:

Set of IDs of incoming edges

4.7.2.4 `std::set< t_id > libgexf::DirectedGraph::getOutEdges (const libgexf::t_id node_id) const`

Get outgoing edges from a node.

Parameters:

node_id : node ID

Returns:

Set of IDs of outgoing edges

4.7.2.5 `std::set< t_id > libgexf::DirectedGraph::getSuccessors (const libgexf::t_id node_id) const`

Get node successors.

Parameters:

node_id : node ID

Returns:

Set of node IDs

4.7.2.6 `std::set< t_id > libgexf::DirectedGraph::getPredecessors (const libgexf::t_id node_id) const`

Get node predecessors.

Parameters:

node_id : node ID

Returns:

Set of node IDs

4.7.2.7 `unsigned int libgexf::DirectedGraph::getInDegree (const libgexf::t_id node_id) const`

Get indegree value.

Parameters:

node_id : node ID

Returns:

indegree value

4.7.2.8 `unsigned int libgexf::DirectedGraph::getOutDegree (const libgexf::t_id node_id) const`

Get outdegree value.

Parameters:

node_id : node ID

Returns:

outdegree value

4.7.2.9 `bool libgexf::DirectedGraph::isSuccessor (const libgexf::t_id node_id, const libgexf::t_id successor_id) const`

Test a possible successor.

Check if the *successor_id* is a successor of the node *node_id*.

Parameters:

node_id : node ID

successor_id : node ID of the tested successor

Returns:

true if *successor_id* is a successor of *node_id*

4.7.2.10 `bool libgexf::DirectedGraph::isPredecessor (const libgexf::t_id node_id, const libgexf::t_id predecessor_id) const`

Test a possible predecessor.

Check if the *predecessor_id* is a predecessor of the node *node_id*.

Parameters:

node_id : node ID

predecessor_id : node ID of the tested predecessor

Returns:

true if *predecessor_id* is a predecessor of *node_id*

The documentation for this class was generated from the following files:

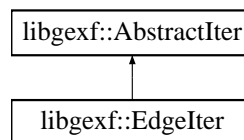
- [directedgraph.h](#)
- [directedgraph.cpp](#)

4.8 libgexf::EdgeIter Class Reference

Iterator on edges.

```
#include <edgeiter.h>
```

Inheritance diagram for libgexf::EdgeIter::



Public Member Functions

- [EdgeIter](#) (const [libgexf::Graph](#) *g)
Constructor.
- [EdgeIter](#) * [begin](#) ()
Init.
- bool [hasNext](#) () const
Test next element in collection.
- [libgexf::t_id](#) [next](#) ()
Iterate.
- [libgexf::t_id](#) [currentSource](#) () const
Get source node.
- [libgexf::t_id](#) [currentTarget](#) () const
Get target node.
- float [currentProperty](#) ([libgexf::t_edge_property](#) prop) const
Get topological property value.

4.8.1 Detailed Description

Iterator on edges.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 libgexf::EdgeIter::EdgeIter (const libgexf::Graph * g)

Constructor.

Parameters:

g : Reference to the [Graph](#) object

4.8.3 Member Function Documentation

4.8.3.1 `EdgeIter * libgexf::EdgeIter::begin ()` [virtual]

Init.

Initialize (or re-initialize) the iterator.

Returns:

an instance of the iterator

Implements [libgexf::AbstractIter](#).

4.8.3.2 `bool libgexf::EdgeIter::hasNext () const` [virtual]

Test next element in collection.

Verify if another element exists.

Returns:

true if another element exists, false otherwise.

Implements [libgexf::AbstractIter](#).

4.8.3.3 `t_id libgexf::EdgeIter::next ()` [virtual]

Iterate.

Get next element in collection.

Returns:

The edge ID.

Implements [libgexf::AbstractIter](#).

4.8.3.4 `t_id libgexf::EdgeIter::currentSource () const`

Get source node.

Returns:

The node id of the source

4.8.3.5 `t_id libgexf::EdgeIter::currentTarget () const`

Get target node.

Returns:

The node id of the target

4.8.3.6 float libgexf::EdgeIter::currentProperty (libgexf::t_edge_property *prop*) const

Get topological property value.

Parameters:

prop : Topological edge property name

Returns:

Property value or 0.0 by default

The documentation for this class was generated from the following files:

- [edgeiter.h](#)
- [edgeiter.cpp](#)

4.9 libgexf::FileReader Class Reference

Read a [GEXF](#) file.

```
#include <filereader.h>
```

Public Types

- enum [Version](#) { [_1_0](#), [_1_1](#) }
Possible version number of the [GEXF](#) format.

Public Member Functions

- [FileReader](#) (const std::string filepath, const [Version](#) v=[_1_1](#))
Constructor with init.
- [FileReader](#) (const [FileReader](#) &orig)
Copy constructor.
- [libgexf::GEXF](#) getGEXFCopy ()
Get a duplicated instance of the internal [GEXF](#) data.
- void [init](#) (const std::string filepath, const [Version](#) v=[_1_1](#))
Initialize the file reader.
- void [slurp](#) ()
Read the given file in one pass.

4.9.1 Detailed Description

Read a [GEXF](#) file.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 libgexf::FileReader::FileReader (const std::string *filepath*, const [Version](#) *v* = [_1_1](#))

Constructor with init.

Parameters:

- filepath* : Path to the written file
- v* : version number of the [GEXF](#) format

4.9.3 Member Function Documentation

4.9.3.1 GEXF libgexf::FileReader::getGEXFCopy ()

Get a duplicated instance of the internal [GEXF](#) data.

Returns:

[GEXF](#) instance

4.9.3.2 void libgexf::FileReader::init (const std::string *filepath*, const Version *v* = __1__1)

Initialize the file reader.

Parameters:

filepath : Path to the [GEXF](#) file
v : version number of the [GEXF](#) format

The documentation for this class was generated from the following files:

- [filereader.h](#)
- [filereader.cpp](#)

4.10 libgexf::FileReaderException Class Reference

Exception occurring when reading a file.

```
#include <exceptions.h>
```

Public Member Functions

- **FileReaderException** (const std::string what) throw ()
- virtual const char * **what** () const throw ()

4.10.1 Detailed Description

Exception occurring when reading a file.

The documentation for this class was generated from the following file:

- [exceptions.h](#)

4.11 libgexf::FileWriter Class Reference

Write a [GEXF](#) file.

```
#include <filewriter.h>
```

Public Types

- enum [ElemType](#) { **NODE**, **EDGE** }

Possible type of element.

Public Member Functions

- [FileWriter](#) (const std::string filepath, [GEXF](#) *gexf)
Constructor with init.
- [FileWriter](#) (const [FileWriter](#) &orig)
Copy constructor.
- [libgexf::GEXF](#) getGEXFCopy ()
Get a duplicated instance of the internal [GEXF](#) data.
- void [init](#) (const std::string filepath, [libgexf::GEXF](#) *gexf)
Initialize the file writer.
- void [write](#) ()
Write the file.

4.11.1 Detailed Description

Write a [GEXF](#) file.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 libgexf::FileWriter::FileWriter (const std::string *filepath*, [GEXF](#) * *gexf*)

Constructor with init.

Parameters:

filepath : Path to the written file

gexf : reference to a [GEXF](#) object

4.11.3 Member Function Documentation

4.11.3.1 GEXF libgexf::FileWriter::getGEXFCopy ()

Get a duplicated instance of the internal [GEXF](#) data.

Returns:

[GEXF](#) instance

4.11.3.2 void libgexf::FileWriter::init (const std::string *filepath*, libgexf::GEXF * *gexf*)

Initialize the file writer.

Parameters:

filepath : Path to the [GEXF](#) file
gexf : reference to a [GEXF](#) object

The documentation for this class was generated from the following files:

- [filewriter.h](#)
- [filewriter.cpp](#)

4.12 libgexf::FileWriterException Class Reference

Exception occurring when writing a file.

```
#include <exceptions.h>
```

Public Member Functions

- **FileWriterException** (const std::string what) throw ()
- virtual const char * **what** () const throw ()

4.12.1 Detailed Description

Exception occurring when writing a file.

The documentation for this class was generated from the following file:

- [exceptions.h](#)

4.13 libgexf::GEXF Class Reference

[GEXF](#) class, just a container.

```
#include <gexf.h>
```

Public Member Functions

- [GEXF](#) (const [GEXF](#) &orig)
Copy constructor.
- [libgexf::UndirectedGraph](#) & [getUndirectedGraph](#) ()
Get an undirected graph instance.
- [libgexf::DirectedGraph](#) & [getDirectedGraph](#) ()
Get a directed graph instance.
- [libgexf::Data](#) & [getData](#) ()
Get associated data instance.
- [libgexf::MetaData](#) & [getMetaData](#) ()
Get associated meta data instance.
- void [setGraphType](#) (libgexf::t_graph t)
Change the type of graph.
- libgexf::t_graph [getGraphType](#) ()
Get the type of graph.
- bool [checkIntegrity](#) ()
Check the data correctness.

Public Attributes

- [libgexf::Graph _graph](#)
- [libgexf::t_graph _type](#)
- [libgexf::Data _data](#)
- [libgexf::MetaData _meta](#)

Friends

- std::ostream & **operator**<< (std::ostream &os, const [GEXF](#) &o)

4.13.1 Detailed Description

[GEXF](#) class, just a container.

- graph topology
- data attributes
- hierarchy (currently not available)
- viz data (currently not available)
- dynamics (currently not available)

4.13.2 Member Function Documentation

4.13.2.1 UndirectedGraph & libgexf::GEXF::getUndirectedGraph ()

Get an undirected graph instance.

Returns:

Undirected graph

4.13.2.2 DirectedGraph & libgexf::GEXF::getDirectedGraph ()

Get a directed graph instance.

Returns:

Directed graph

4.13.2.3 Data & libgexf::GEXF::getData ()

Get associated data instance.

Returns:

graph data

4.13.2.4 MetaData & libgexf::GEXF::getMetaData ()

Get associated meta data instance.

Returns:

graph meta data

4.13.2.5 void libgexf::GEXF::setGraphType (libgexf::t_graph t)

Change the type of graph.

Parameters:

t : Type of graph (directed, undirected or mixed)

4.13.2.6 t_graph libgexf::GEXF::getGraphType ()

Get the type of graph.

Returns:

Type of graph (directed, undirected or mixed)

4.13.2.7 bool libgexf::GEXF::checkIntegrity ()

Check the data correctness.

- verify if each node has a label
- verify if each attvalue has a value or a defaultvalue

4.13.3 Member Data Documentation

4.13.3.1 libgexf::Graph libgexf::GEXF::_graph

Topology structure

4.13.3.2 libgexf::t_graph libgexf::GEXF::_type

Default edge type

4.13.3.3 libgexf::Data libgexf::GEXF::_data

Associated data and attributes on nodes and edges

4.13.3.4 libgexf::MetaData libgexf::GEXF::_meta

Associated meta data

The documentation for this class was generated from the following files:

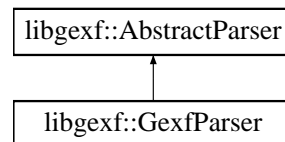
- [gexf.h](#)
- [gexf.cpp](#)

4.14 libgexf::GexfParser Class Reference

Parse a [GEXF](#) file.

```
#include <gexfparser.h>
```

Inheritance diagram for libgexf::GexfParser::



Public Member Functions

- [GexfParser](#) (const [GexfParser](#) &orig)
Copy constructor.
- void [bind](#) (libgexf::GEXF *gexf)
Bind itself to a [GEXF](#) instance.
- void [processNode](#) (xmlTextReaderPtr reader, const xmlChar *name)
Start a process on an XML element.

4.14.1 Detailed Description

Parse a [GEXF](#) file.

4.14.2 Member Function Documentation

4.14.2.1 void libgexf::GexfParser::bind (libgexf::GEXF *gexf) [virtual]

Bind itself to a [GEXF](#) instance.

Parameters:

gexf : Reference to a [GEXF](#) object

Implements [libgexf::AbstractParser](#).

4.14.2.2 void libgexf::GexfParser::processNode (xmlTextReaderPtr reader, const xmlChar *name) [virtual]

Start a process on an XML element.

Call the right processing method.

Parameters:

reader : Reference to the libxml TextReader instance

name : Name of the XML element to process

Implements [libgexf::AbstractParser](#).

The documentation for this class was generated from the following files:

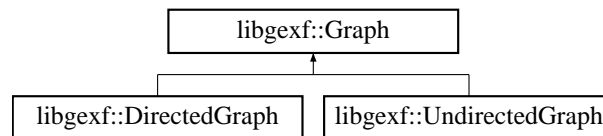
- [gexfparsers.h](#)
- [gexfparsers.cpp](#)

4.15 libgexf::Graph Class Reference

Topology structure of the graph.

```
#include <graph.h>
```

Inheritance diagram for libgexf::Graph::



Public Member Functions

- **Graph** (const **Graph** &orig)
Copy constructor.
- void **addNode** (const libgexf::t_id id)
Add a node.
- void **addEdge** (const libgexf::t_id id, const libgexf::t_id source_id, const libgexf::t_id target_id, const unsigned int cardinal=1, const libgexf::t_edge_type type=EDGE_UNDEF)
Add an edge.
- void **removeNode** (const libgexf::t_id id)
Remove a node.
- void **removeEdge** (const libgexf::t_id source_id, const libgexf::t_id target_id)
Remove an edge.
- bool **containsNode** (const libgexf::t_id id) const
Test node existence.
- bool **containsEdge** (const libgexf::t_id source_id, const libgexf::t_id target_id) const
Test edge existence.
- libgexf::NodeIter * **getNodes** () const
Get all nodes.
- libgexf::EdgeIter * **getEdges** () const
Get all edges.
- std::set< libgexf::t_id > **getNeighbors** (const libgexf::t_id node_id) const
Get node neighbors.
- unsigned int **getNodeCount** () const
Count the nodes.

- unsigned int [getEdgeCount](#) () const
Count the edges.
- unsigned int [getDegree](#) (const libgexf::t_id node_id) const
Get node degree.
- void [clearEdges](#) (const libgexf::t_id node_id)
Delete node links.
- void [clear](#) ()
Clear the graph.
- void [clearEdges](#) ()
Delete all edges.
- void [readLock](#) () throw (libgexf::ReadLockException)
Set a lock on reading.
- void [readUnlock](#) ()
Unset a lock on reading.
- void [writeLock](#) () throw (libgexf::WriteLockException)
Get a lock on writing.
- void [writeUnlock](#) ()
Unset a lock on writing.
- bool [isReadLock](#) ()
Test if a read lock exists.
- bool [isWriteLock](#) ()
Test if a write lock exists.
- bool [isUnlock](#) ()
Unset all locks.

Protected Member Functions

- unsigned int [getInternalEdgeCount](#) () const
Count edge rows.

Protected Attributes

- std::set< t_id > [_nodes](#)
- std::map< t_id, std::map< t_id, t_id > > [_edges](#)
- std::map< t_id, std::set< t_id > > [_reverse_edges](#)
- std::set< t_id > [_bloom_edges](#)

- `std::map< t_id, std::map< t_edge_property, t_edge_value > > _edges_properties`
- unsigned short int `_rlock_count`
- char `_lock_flag`

Flag used for determining the lock type:.

Friends

- class `NodeIter`
- class `EdgeIter`
- `std::ostream & operator<< (std::ostream &os, const Graph &o)`

4.15.1 Detailed Description

Topology structure of the graph.

4.15.2 Member Function Documentation

4.15.2.1 void libgexf::Graph::addNode (const libgexf::t_id id)

Add a node.

Parameters:

id : node ID

4.15.2.2 void libgexf::Graph::addEdge (const libgexf::t_id id, const libgexf::t_id source_id, const libgexf::t_id target_id, const unsigned int cardinal = 1, const libgexf::t_edge_type type = EDGE_UNDEF)

Add an edge.

Parameters:

id : edge ID

source_id : source node ID

target_id : target node ID

cardinal : number of edges (optional, 1 by default)

type : type of edge (optional, undef by default)

4.15.2.3 void libgexf::Graph::removeNode (const libgexf::t_id id)

Remove a node.

Parameters:

id : node ID

4.15.2.4 void libgexf::Graph::removeEdge (const libgexf::t_id *source_id*, const libgexf::t_id *target_id*)

Remove an edge.

Parameters:

source_id : source node ID

target_id : target node ID

4.15.2.5 bool libgexf::Graph::containsNode (const libgexf::t_id *id*) const

Test node existence.

Parameters:

id : node ID

Returns:

true if the node exists, false otherwise

4.15.2.6 bool libgexf::Graph::containsEdge (const libgexf::t_id *source_id*, const libgexf::t_id *target_id*) const

Test edge existence.

Parameters:

source_id : source node ID

target_id : target node ID

Returns:

true if the edge exists, false otherwise

4.15.2.7 NodeIter * libgexf::Graph::getNodes () const

Get all nodes.

Returns:

Iterator on the node collection

4.15.2.8 EdgeIter * libgexf::Graph::getEdges () const

Get all edges.

Returns:

Iterator on the edge collection

4.15.2.9 set< t_id > libgexf::Graph::getNeighbors (const libgexf::t_id *node_id*) const

Get node neighbors.

Parameters:

node_id : node ID

Returns:

Set of nodes directly linked to the node

4.15.2.10 unsigned int libgexf::Graph::getNodeCount () const

Count the nodes.

Returns:

Number of nodes

4.15.2.11 unsigned int libgexf::Graph::getEdgeCount () const

Count the edges.

Returns:

Number of edges

4.15.2.12 unsigned int libgexf::Graph::getDegree (const libgexf::t_id *node_id*) const

Get node degree.

Parameters:

node_id : node ID

Returns:

Degree

4.15.2.13 void libgexf::Graph::clearEdges (const libgexf::t_id *node_id*)

Delete node links.

Parameters:

node_id : node ID

4.15.2.14 void libgexf::Graph::readLock () throw (libgexf::ReadLockException)

Set a lock on reading.

Exceptions:

[ReadLockException](#) { Unable to set the lock }

4.15.2.15 void libgexf::Graph::writeLock () throw (libgexf::WriteLockException)

Get a lock on writing.

Exceptions:

[WriteLockException](#) { Unable to set the lock }

4.15.2.16 unsigned int libgexf::Graph::getInternalEdgeCount () const [protected]

Count edge rows.

Used for [EdgeIter](#); This is not the real number of edges (cardinals are not count).

Returns:

number of edge rows

4.15.3 Member Data Documentation**4.15.3.1 std::set<t_id> libgexf::Graph::_nodes [protected]**

Set of all nodes

4.15.3.2 std::map<t_id,std::map<t_id,t_id> > libgexf::Graph::_edges [protected]

map<source_id, map<target_id, edge_id> >

4.15.3.3 std::map<t_id,std::set<t_id> > libgexf::Graph::_reverse_edges [protected]

map<target_id, set<source_id> >

4.15.3.4 std::set<t_id> libgexf::Graph::_bloom_edges [protected]

Set of all edge_id used as a (poor) bloom filter

**4.15.3.5 std::map<t_id,std::map<t_edge_property,t_edge_value> >
libgexf::Graph::_edges_properties [protected]**

Topological properties of edges

4.15.3.6 unsigned short int libgexf::Graph::_rlock_count [protected]

Number of read-locks

4.15.3.7 char libgexf::Graph::_lock_flag [protected]

Flag used for determining the lock type:.

- 0 = unlocked
- 1 = read locked
- 2 = write locked

The documentation for this class was generated from the following files:

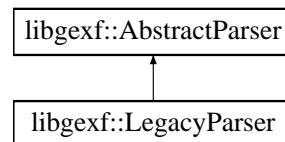
- [graph.h](#)
- [graph.cpp](#)

4.16 libgexf::LegacyParser Class Reference

Parse an old [GEXF](#) file (1.0).

```
#include <legacyparser.h>
```

Inheritance diagram for libgexf::LegacyParser::



Public Member Functions

- [LegacyParser](#) (const [LegacyParser](#) &orig)
Copy constructor.
- void [bind](#) (libgexf::GEXF *gexf)
Bind itself to a [GEXF](#) instance.
- void [processNode](#) (xmlTextReaderPtr reader, const xmlChar *name)
Start a process on an XML element.

4.16.1 Detailed Description

Parse an old [GEXF](#) file (1.0).

4.16.2 Member Function Documentation

4.16.2.1 void libgexf::LegacyParser::bind (libgexf::GEXF *gexf) [virtual]

Bind itself to a [GEXF](#) instance.

Parameters:

gexf : Reference to a [GEXF](#) object

Implements [libgexf::AbstractParser](#).

4.16.2.2 void libgexf::LegacyParser::processNode (xmlTextReaderPtr reader, const xmlChar *name) [virtual]

Start a process on an XML element.

Call the right processing method.

Parameters:

reader : Reference to the libxml TextReader instance

name : Name of the XML element to process

Implements [libgexf::AbstractParser](#).

The documentation for this class was generated from the following files:

- [legacyparser.h](#)
- [legacyparser.cpp](#)

4.17 libgexf::MetaData Class Reference

Associated meta data and attributes on the graph.

```
#include <metadata.h>
```

Public Member Functions

- [MetaData](#) (const [MetaData](#) &orig)
Copy constructor.
- std::string [getVersion](#) () const
Get the [GEXF](#) version.
- std::string [getXmlns](#) () const
Get the XMLNS.
- std::string [getXsi](#) () const
Get the XMLNS:XSI.
- std::string [getSchema](#) () const
Get the XML schema.
- std::string [getVariant](#) () const
Get the variant.
- std::string [getCreator](#) () const
Get the creator.
- std::string [getDescription](#) () const
Get the description.
- std::string [getKeywords](#) () const
Get the keywords.
- std::string [getLastModifiedDate](#) () const
Get the last modified date.
- void [setVersion](#) (const std::string version)
Set the [GEXF](#) version.
- void [setXmlns](#) (const std::string xmlns)
Set the XMLNS.
- void [setXsi](#) (const std::string xsi)
Set the XMLNS:XSI.
- void [setSchema](#) (const std::string schema)
Set the XML schema.

- void [setVariant](#) (const std::string variant)
Set the variant.
- void [setCreator](#) (const std::string creator)
Set the creator.
- void [setDescription](#) (const std::string description)
Set the description.
- void [setKeywords](#) (const std::string keywords)
Set the keywords.
- void [setLastModifiedDate](#) (const std::string lastmodifieddate)
Set the last modified date.

Friends

- std::ostream & **operator**<< (std::ostream &os, const [MetaData](#) &o)

4.17.1 Detailed Description

Associated meta data and attributes on the graph.

The documentation for this class was generated from the following files:

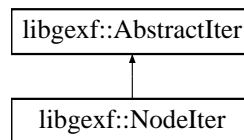
- [metadata.h](#)
- [metadata.cpp](#)

4.18 libgexf::NodeIter Class Reference

Iterator on nodes.

```
#include <nodeiter.h>
```

Inheritance diagram for libgexf::NodeIter::



Public Member Functions

- [NodeIter](#) (const [libgexf::Graph](#) *g)
Constructor.
- [NodeIter](#) * [begin](#) ()
Init.
- bool [hasNext](#) () const
Test next element in collection.
- [libgexf::t_id](#) [next](#) ()
Iterate.

4.18.1 Detailed Description

Iterator on nodes.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 libgexf::NodeIter::NodeIter (const libgexf::Graph * g)

Constructor.

Parameters:

g : Reference to the [Graph](#) object

4.18.3 Member Function Documentation

4.18.3.1 NodeIter * libgexf::NodeIter::begin () [virtual]

Init.

Initialize (or re-initialize) the iterator.

Returns:

an instance of the iterator

Implements [libgexf::AbstractIter](#).

4.18.3.2 bool libgexf::NodeIter::hasNext () const [virtual]

Test next element in collection.

Verify if another element exists.

Returns:

true if another element exists, false otherwise.

Implements [libgexf::AbstractIter](#).

4.18.3.3 t_id libgexf::NodeIter::next () [virtual]

Iterate.

Get next element in collection.

Returns:

The node ID.

Implements [libgexf::AbstractIter](#).

The documentation for this class was generated from the following files:

- [nodeiter.h](#)
- [nodeiter.cpp](#)

4.19 libgexf::ReadLockException Class Reference

Exception occurring on a read-lock.

```
#include <exceptions.h>
```

Public Member Functions

- **ReadLockException** (const std::string what) throw ()
- virtual const char * **what** () const throw ()

4.19.1 Detailed Description

Exception occurring on a read-lock.

The documentation for this class was generated from the following file:

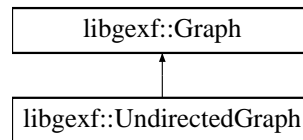
- [exceptions.h](#)

4.20 libgexf::UndirectedGraph Class Reference

Interpretation of the topology structure as a undirected graph.

```
#include <undirectedgraph.h>
```

Inheritance diagram for libgexf::UndirectedGraph::



Public Member Functions

- [UndirectedGraph](#) (const [UndirectedGraph](#) &orig)
Copy constructor.

4.20.1 Detailed Description

Interpretation of the topology structure as a undirected graph.

The documentation for this class was generated from the following files:

- [undirectedgraph.h](#)
- [undirectedgraph.cpp](#)

4.21 libgexf::WriteLockException Class Reference

Exception occurring on a write-lock.

```
#include <exceptions.h>
```

Public Member Functions

- **WriteLockException** (const std::string what) throw ()
- virtual const char * **what** () const throw ()

4.21.1 Detailed Description

Exception occurring on a write-lock.

The documentation for this class was generated from the following file:

- [exceptions.h](#)

Chapter 5

File Documentation

5.1 abstractiter.h File Reference

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::AbstractIter](#)
Iterator Interface.

5.1.1 Detailed Description

Author:

sebastien heyman

Date:

20 juillet 2009, 11:44

Version:

0.1

5.2 abstractparser.h File Reference

```
#include "gexf.h"
#include <libxml/xmlreader.h>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::AbstractParser](#)
Parser Interface.

5.2.1 Detailed Description

Author:

sebastien heymann

Date:

17 juillet 2009, 15:02

Version:

0.1

5.3 attributeiter.cpp File Reference

```
#include "attributeiter.h"
```

Namespaces

- namespace **libgexf**

5.3.1 Detailed Description

Author:

: sebastien heyman

Date:

15 juillet 2009, 14:27

Version:

5.4 attributeiter.h File Reference

```
#include "typedefs.h"
#include "data.h"
#include "abstractiter.h"
#include <map>
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::AttributeIter](#)
Iterator on attributes.

5.4.1 Detailed Description

Author:

: sebastien heymann

Date:

15 juillet 2009, 14:27

Version:

5.5 attvalueiter.cpp File Reference

```
#include "attvalueiter.h"  
#include <stdexcept>
```

Namespaces

- namespace **libgexf**

5.5.1 Detailed Description

Author:

sebastien heymann

Date:

17 juillet 2009, 22:35

Version:

0.1

5.6 attvalueiter.h File Reference

```
#include "typedefs.h"
#include "data.h"
#include "abstractiter.h"
#include <map>
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::AttValueIter](#)
Iterator on attribute values.

5.6.1 Detailed Description

Author:

sebastien heymann

Date:

17 juillet 2009, 22:35

Version:

0.1

5.7 conv.cpp File Reference

```
#include "conv.h"  
#include <iostream>  
#include <sstream>  
#include <string.h>  
#include <libxml/encoding.h>  
#include <libxml/xmlwriter.h>
```

Namespaces

- namespace **libgexf**

5.7.1 Detailed Description

Author:

sebastien heymann

Date:

15 juillet 2009, 18:36

Version:

0.1

5.8 conv.h File Reference

```
#include "typedefs.h"
#include <string>
#include <libxml/xmlstring.h>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::Conv](#)
Utility class for transforming data.

5.8.1 Detailed Description

Author:

sebastien heymann

Date:

15 juillet 2009, 18:36

Version:

0.1

5.9 data.cpp File Reference

```
#include <map>
#include "data.h"
#include "attributeiter.h"
#include <sstream>
```

Namespaces

- namespace **libgexf**

Functions

- `std::ostream & libgexf::operator<< (std::ostream &os, const Data &o)`

5.9.1 Detailed Description

Author:

sebastien heymann

Date:

30 juin 2009, 13:35

Version:

0.1

5.10 data.h File Reference

```
#include "typedefs.h"
#include "attributeiter.h"
#include "attvalueiter.h"
#include <string>
#include <map>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::Data](#)
Associated data and attributes on nodes and edges.

5.10.1 Detailed Description

Author:

sebastien heymann

Date:

30 juin 2009, 13:35

Version:

0.1

5.11 directedgraph.cpp File Reference

```
#include "directedgraph.h"
#include "exceptions.h"
#include <stdexcept>
#include <map>
#include <set>
```

Namespaces

- namespace **libgexf**

5.11.1 Detailed Description

Author:

sebastien heymann

Date:

8 juin 2009, 10:21

Version:

0.1

5.12 directedgraph.h File Reference

```
#include "graph.h"
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::DirectedGraph](#)
Interpretation of the topology structure as a directed graph.

5.12.1 Detailed Description

Author:

sebastien heymann

Date:

8 juin 2009, 10:21

Version:

0.1

5.13 edgeiter.cpp File Reference

```
#include <map>
#include "edgeiter.h"
```

Namespaces

- namespace **libgexf**

5.13.1 Detailed Description

Author:

sebastien heymann

Date:

9 juillet 2009, 17:38

Version:

0.1

5.14 edgeiter.h File Reference

```
#include "typedefs.h"
#include "graph.h"
#include "abstractiter.h"
#include <map>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::EdgeIter](#)
Iterator on edges.

5.14.1 Detailed Description

Author:

sebastien heymann

Date:

9 juillet 2009, 17:38

Version:

0.1

5.15 exceptions.h File Reference

```
#include <exception>
#include <string>
#include <sstream>
#include "typedefs.h"
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::ReadLockException](#)
Exception occurring on a read-lock.
- class [libgexf::WriteLockException](#)
Exception occurring on a write-lock.
- class [libgexf::FileWriterException](#)
Exception occurring when writing a file.
- class [libgexf::FileReaderException](#)
Exception occurring when reading a file.

5.15.1 Detailed Description

Author:

sebastien heymann

Date:

4 juin 2009, 10:36

Version:

0.1

5.16 filereader.cpp File Reference

```
#include "filereader.h"
#include "gexfparser.h"
#include "legacyparser.h"
#include "exceptions.h"
#include <stdio.h>
#include <iostream>
#include <libxml/xmlreader.h>
```

Namespaces

- namespace **libgexf**

5.16.1 Detailed Description

Author:

sebastien heymann

Date:

19 juin 2009, 12:37

Version:

0.1

5.17 filereader.h File Reference

```
#include "gexf.h"
#include "abstractparser.h"
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::FileReader](#)
Read a [GEXF](#) file.

5.17.1 Detailed Description

Author:

sebastien heymann

Date:

19 juin 2009, 12:37

Version:

0.1

5.18 filewriter.cpp File Reference

```
#include "metadata.h"
#include "filewriter.h"
#include "exceptions.h"
#include "typedefs.h"
#include "conv.h"
#include <stdexcept>
#include <libxml/xmlwriter.h>
```

Namespaces

- namespace **libgexf**

5.18.1 Detailed Description

Author:

sebastien heymann

Date:

8 juillet 2009, 17:58

Version:

0.1

5.19 filewriter.h File Reference

```
#include "gexf.h"
#include <libxml/xmlwriter.h>
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class **libgexf::FileWriter**
Write a [GEXF](#) file.

5.19.1 Detailed Description

Author:

sebastien heymann

Date:

8 juillet 2009, 17:58

Version:

0.1

5.20 gexf.cpp File Reference

```
#include "data.h"  
#include "gexf.h"
```

Namespaces

- namespace **libgexf**

Functions

- `std::ostream & libgexf::operator<< (std::ostream &os, const GEXF &o)`

5.20.1 Detailed Description

Author:

sebastien heymann

Date:

17 avril 2009, 17:28

5.21 gexf.h File Reference

```
#include <iostream>
#include "graph.h"
#include "undirectedgraph.h"
#include "directedgraph.h"
#include "data.h"
#include "metadata.h"
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::GEXF](#)
[GEXF](#) class, just a container.

5.21.1 Detailed Description

Author:

sebastien heymann

Date:

17 avril 2009, 17:28

Version:

0.1

5.22 gexfparser.cpp File Reference

```
#include "data.h"  
#include "gexfparser.h"  
#include "filereader.h"  
#include "conv.h"  
#include <string>
```

Namespaces

- namespace **libgexf**

5.22.1 Detailed Description

Author:

sebastien heymann

Date:

22 juin 2009, 17:20

Version:

0.1

5.23 gexfparser.h File Reference

```
#include "gexf.h"
#include "typedefs.h"
#include "abstractparser.h"
#include <libxml/xmlreader.h>
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::GexfParser](#)
Parse a [GEXF](#) file.

5.23.1 Detailed Description

Author:

sebastien heymann

Date:

22 juin 2009, 17:20

Version:

0.1

5.24 graph.cpp File Reference

```
#include "graph.h"
#include "exceptions.h"
#include <stdexcept>
#include <map>
#include <set>
#include <string>
#include <sstream>
#include <iostream>
```

Namespaces

- namespace **libgexf**

Functions

- ostream & **libgexf::operator**<< (ostream &os, const Graph &o)

5.24.1 Detailed Description

Author:

sebastien heymann

Date:

17 avril 2009, 17:27

Version:

0.1

5.25 graph.h File Reference

```
#include <set>
#include <map>
#include <iostream>
#include "typedefs.h"
#include "exceptions.h"
#include "nodeiter.h"
#include "edgeiter.h"
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::Graph](#)
Topology structure of the graph.

5.25.1 Detailed Description

Author:

sebastien heymann

Date:

17 avril 2009, 17:27

Version:

0.1

5.26 legacyparser.cpp File Reference

```
#include "data.h"
#include "legacyparser.h"
#include "filereader.h"
#include "conv.h"
#include <string>
```

Namespaces

- namespace **libgexf**

5.26.1 Detailed Description

Author:

sebastien heymann

Date:

22 juin 2009, 17:20

Version:

0.1

5.27 legacyparser.h File Reference

```
#include "gexf.h"
#include "typedefs.h"
#include "abstractparser.h"
#include <libxml/xmlreader.h>
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::LegacyParser](#)
Parse an old [GEXF](#) file (1.0).

5.27.1 Detailed Description

Author:

sebastien heymann

Date:

22 juin 2009, 17:20

Version:

0.1

5.28 libgexf.h File Reference

```
#include <libgexf/typedefs.h>
#include <libgexf/exceptions.h>
#include <libgexf/gexf.h>
#include <libgexf/graph.h>
#include <libgexf/metadata.h>
#include <libgexf/data.h>
#include <libgexf/abstractiter.h>
#include <libgexf/nodeiter.h>
#include <libgexf/edgeiter.h>
#include <libgexf/directedgraph.h>
#include <libgexf/undirectedgraph.h>
#include <libgexf/abstractparser.h>
#include <libgexf/filereader.h>
#include <libgexf/filewriter.h>
```

5.28.1 Detailed Description

Author:

sebastien heyman

Date:

22 juin 2009, 16:45

Version:

0.1

5.29 metadata.cpp File Reference

```
#include "metadata.h"  
#include <sstream>
```

Namespaces

- namespace **libgexf**

Functions

- `std::ostream & libgexf::operator<< (std::ostream &os, const MetaData &o)`

5.29.1 Detailed Description

Author:

sebastien heymann

Date:

30 juin 2009, 14:14

Version:

0.1

5.30 metadata.h File Reference

```
#include <string>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::MetaData](#)
Associated meta data and attributes on the graph.

5.30.1 Detailed Description

Author:

sebastien heymann

Date:

30 juin 2009, 14:14

Version:

0.1

5.31 nodeiter.cpp File Reference

```
#include "nodeiter.h"
```

Namespaces

- namespace **libgexf**

5.31.1 Detailed Description

Author:

sebastien heymann

Date:

9 juillet 2009, 15:48

Version:

0.1

5.32 nodeiter.h File Reference

```
#include "typedefs.h"
#include "graph.h"
#include "abstractiter.h"
#include <set>
```

Namespaces

- namespace **libgexf**

Classes

- class [libgexf::NodeIter](#)
Iterator on nodes.

5.32.1 Detailed Description

Author:

sebastien heymann

Date:

9 juillet 2009, 15:48

Version:

0.1

5.33 typedefs.h File Reference

```
#include <string>
```

Namespaces

- namespace **libgexf**

Typedefs

- typedef std::string **libgexf::t_id**
- typedef float **libgexf::t_edge_value**

Enumerations

- enum **t_graph** { **GRAPH_DIRECTED**, **GRAPH_UNDIRECTED**, **GRAPH_MIXED** }
Available graph types.
- enum **t_edge_property** { **EDGE_TYPE**, **EDGE_COUNT**, **EDGE_WEIGHT** }
Available edge topological properties.
- enum **t_edge_type** { **EDGE_UNDEF**, **EDGE_DIRECTED**, **EDGE_UNDIRECTED**, **EDGE_DOUBLE** }
Available edge types.
- enum **t_attr_type** {
 INTEGER, **DOUBLE**, **FLOAT**, **BOOLEAN**,
 STRING, **LIST_STRING** }
Available types of attributes.

5.33.1 Detailed Description

Author:

sebastien

Date:

8 juin 2009, 14:22

5.34 undirectedgraph.cpp File Reference

```
#include "undirectedgraph.h"
```

Namespaces

- namespace **libgexf**

5.34.1 Detailed Description

Author:

sebastien heymann

Date:

8 juin 2009, 10:21

Version:

0.1

5.35 undirectedgraph.h File Reference

```
#include "graph.h"
```

Namespaces

- namespace **libgexf**

Classes

- class **libgexf::UndirectedGraph**

Interpretation of the topology structure as a undirected graph.

5.35.1 Detailed Description

Author:

sebastien heymann

Date:

8 juin 2009, 10:21

Version:

0.1

Index

- `_bloom_edges`
 - `libgexf::Graph`, 48
 - `_data`
 - `libgexf::GEXF`, 40
 - `_edges`
 - `libgexf::Graph`, 48
 - `_edges_properties`
 - `libgexf::Graph`, 48
 - `_graph`
 - `libgexf::GEXF`, 40
 - `_lock_flag`
 - `libgexf::Graph`, 49
 - `_meta`
 - `libgexf::GEXF`, 40
 - `_nodes`
 - `libgexf::Graph`, 48
 - `_reverse_edges`
 - `libgexf::Graph`, 48
 - `_rlock_count`
 - `libgexf::Graph`, 48
 - `_type`
 - `libgexf::GEXF`, 40
- `abstractiter.h`, 59
- `abstractparser.h`, 60
- `addEdge`
 - `libgexf::Graph`, 45
- `addEdgeAttributeColumn`
 - `libgexf::Data`, 20
- `addNode`
 - `libgexf::Graph`, 45
- `addNodeAttributeColumn`
 - `libgexf::Data`, 20
- `AttributeIter`
 - `libgexf::AttributeIter`, 11
 - `libgexf::Data`, 24
- `attributeiter.cpp`, 61
- `attributeiter.h`, 62
- `AttValueIter`
 - `libgexf::AttValueIter`, 15
 - `libgexf::Data`, 24
- `attvalueiter.cpp`, 63
- `attvalueiter.h`, 64
- `begin`
 - `libgexf::AbstractIter`, 8
 - `libgexf::AttributeIter`, 12
 - `libgexf::AttValueIter`, 15
 - `libgexf::EdgeIter`, 30
 - `libgexf::NodeIter`, 54
- `bind`
 - `libgexf::AbstractParser`, 9
 - `libgexf::GexfParser`, 41
 - `libgexf::LegacyParser`, 50
- `checkIntegrity`
 - `libgexf::GEXF`, 40
- `clearEdgeAttributes`
 - `libgexf::Data`, 23
- `clearEdges`
 - `libgexf::Graph`, 47
- `clearNodeAttributes`
 - `libgexf::Data`, 23
- `containsEdge`
 - `libgexf::Graph`, 46
- `containsNode`
 - `libgexf::Graph`, 46
- `conv.cpp`, 65
- `conv.h`, 66
- `currentName`
 - `libgexf::AttValueIter`, 16
- `currentProperty`
 - `libgexf::EdgeIter`, 30
- `currentSource`
 - `libgexf::EdgeIter`, 30
- `currentTarget`
 - `libgexf::EdgeIter`, 30
- `currentTitle`
 - `libgexf::AttributeIter`, 12
- `currentType`
 - `libgexf::AttributeIter`, 12
- `currentValue`
 - `libgexf::AttValueIter`, 15
- `data.cpp`, 67
- `data.h`, 68
- `directedgraph.cpp`, 69
- `directedgraph.h`, 70
- `EdgeIter`

- libgexf::EdgeIter, 29
- edgeiter.cpp, 71
- edgeiter.h, 72
- exceptions.h, 73
- FileReader
 - libgexf::FileReader, 32
- filereader.cpp, 74
- filereader.h, 75
- FileWriter
 - libgexf::FileWriter, 35
- filewriter.cpp, 76
- filewriter.h, 77
- getData
 - libgexf::GEXF, 39
- getDegree
 - libgexf::Graph, 47
- getDirectedGraph
 - libgexf::GEXF, 39
- getEdgeAttribute
 - libgexf::Data, 22
- getEdgeAttributeColumn
 - libgexf::Data, 21
- getEdgeAttributeDefault
 - libgexf::Data, 23
- getEdgeAttributeRow
 - libgexf::Data, 22
- getEdgeCount
 - libgexf::Graph, 47
- getEdges
 - libgexf::Graph, 46
- getGEXFCopy
 - libgexf::FileReader, 33
 - libgexf::FileWriter, 36
- getGraphType
 - libgexf::GEXF, 40
- getInDegree
 - libgexf::DirectedGraph, 27
- getInEdges
 - libgexf::DirectedGraph, 26
- getInternalEdgeCount
 - libgexf::Graph, 48
- getLabel
 - libgexf::Data, 19
- getMetaData
 - libgexf::GEXF, 39
- getNeighbors
 - libgexf::Graph, 46
- getNodeAttribute
 - libgexf::Data, 21
- getNodeAttributeColumn
 - libgexf::Data, 21
- getNodeAttributeDefault
 - libgexf::Data, 22
- getNodeAttributeRow
 - libgexf::Data, 22
- getNodeCount
 - libgexf::Graph, 47
- getNodes
 - libgexf::Graph, 46
- getOutDegree
 - libgexf::DirectedGraph, 27
- getOutEdges
 - libgexf::DirectedGraph, 26
- getPredecessors
 - libgexf::DirectedGraph, 27
- getSuccessors
 - libgexf::DirectedGraph, 26
- getUndirectedGraph
 - libgexf::GEXF, 39
- gexf.cpp, 78
- gexf.h, 79
- gexfparser.cpp, 80
- gexfparser.h, 81
- graph.cpp, 82
- graph.h, 83
- hasEdgeAttributeDefault
 - libgexf::Data, 23
- hasLabel
 - libgexf::Data, 20
- hasNext
 - libgexf::AbstractIter, 8
 - libgexf::AttributeIter, 12
 - libgexf::AttValueIter, 15
 - libgexf::EdgeIter, 30
 - libgexf::NodeIter, 55
- hasNodeAttributeDefault
 - libgexf::Data, 23
- init
 - libgexf::FileReader, 33
 - libgexf::FileWriter, 36
- isPredecessor
 - libgexf::DirectedGraph, 28
- isSuccessor
 - libgexf::DirectedGraph, 27
- legacyparser.cpp, 84
- legacyparser.h, 85
- libgexf.h, 86
- libgexf::AbstractIter, 7
 - begin, 8
 - hasNext, 8
 - next, 8
- libgexf::AbstractParser, 9
 - bind, 9

- processNode, 9
- libgexf::AttributeIter, 11
 - AttributeIter, 11
 - begin, 12
 - currentTitle, 12
 - currentType, 12
 - hasNext, 12
 - next, 12
- libgexf::AttValueIter, 14
 - AttValueIter, 15
 - begin, 15
 - currentName, 16
 - currentValue, 15
 - hasNext, 15
 - next, 15
- libgexf::Conv, 17
- libgexf::Data, 18
 - addEdgeAttributeColumn, 20
 - addNodeAttributeColumn, 20
 - AttributeIter, 24
 - AttValueIter, 24
 - clearEdgeAttributes, 23
 - clearNodeAttributes, 23
 - getEdgeAttribute, 22
 - getEdgeAttributeColumn, 21
 - getEdgeAttributeDefault, 23
 - getEdgeAttributeRow, 22
 - getLabel, 19
 - getNodeAttribute, 21
 - getNodeAttributeColumn, 21
 - getNodeAttributeDefault, 22
 - getNodeAttributeRow, 22
 - hasEdgeAttributeDefault, 23
 - hasLabel, 20
 - hasNodeAttributeDefault, 23
 - setEdgeAttributeDefault, 20
 - setEdgeValue, 21
 - setLabel, 20
 - setNodeAttributeDefault, 20
 - setNodeValue, 21
- libgexf::DirectedGraph, 25
 - getInDegree, 27
 - getInEdges, 26
 - getOutDegree, 27
 - getOutEdges, 26
 - getPredecessors, 27
 - getSuccessors, 26
 - isPredecessor, 28
 - isSuccessor, 27
 - removeInEdges, 26
 - removeOutEdges, 26
- libgexf::EdgeIter, 29
 - begin, 30
 - currentProperty, 30
 - currentSource, 30
 - currentTarget, 30
 - EdgeIter, 29
 - hasNext, 30
 - next, 30
- libgexf::FileReader, 32
 - FileReader, 32
 - getGEXFCopy, 33
 - init, 33
- libgexf::FileReaderException, 34
- libgexf::FileWriter, 35
 - FileWriter, 35
 - getGEXFCopy, 36
 - init, 36
- libgexf::FileWriterException, 37
- libgexf::GEXF, 38
 - _data, 40
 - _graph, 40
 - _meta, 40
 - _type, 40
 - checkIntegrity, 40
 - getData, 39
 - getDirectedGraph, 39
 - getGraphType, 40
 - getMetaData, 39
 - getUndirectedGraph, 39
 - setGraphType, 39
- libgexf::GexfParser, 41
 - bind, 41
 - processNode, 41
- libgexf::Graph, 43
 - _bloom_edges, 48
 - _edges, 48
 - _edges_properties, 48
 - _lock_flag, 49
 - _nodes, 48
 - _reverse_edges, 48
 - _rlock_count, 48
 - addEdge, 45
 - addNode, 45
 - clearEdges, 47
 - containsEdge, 46
 - containsNode, 46
 - getDegree, 47
 - getEdgeCount, 47
 - getEdges, 46
 - getInternalEdgeCount, 48
 - getNeighbors, 46
 - getNodeCount, 47
 - getNodes, 46
 - readLock, 47
 - removeEdge, 45
 - removeNode, 45
 - writeLock, 48

- libgexf::LegacyParser, [50](#)
 - bind, [50](#)
 - processNode, [50](#)
- libgexf::MetaData, [52](#)
- libgexf::NodeIter, [54](#)
 - begin, [54](#)
 - hasNext, [55](#)
 - next, [55](#)
 - NodeIter, [54](#)
- libgexf::ReadLockException, [56](#)
- libgexf::UndirectedGraph, [57](#)
- libgexf::WriteLockException, [58](#)

- metadata.cpp, [87](#)
- metadata.h, [88](#)

- next
 - libgexf::AbstractIter, [8](#)
 - libgexf::AttributeIter, [12](#)
 - libgexf::AttValueIter, [15](#)
 - libgexf::EdgeIter, [30](#)
 - libgexf::NodeIter, [55](#)
- NodeIter
 - libgexf::NodeIter, [54](#)
- nodeiter.cpp, [89](#)
- nodeiter.h, [90](#)

- processNode
 - libgexf::AbstractParser, [9](#)
 - libgexf::GexfParser, [41](#)
 - libgexf::LegacyParser, [50](#)

- readLock
 - libgexf::Graph, [47](#)
- removeEdge
 - libgexf::Graph, [45](#)
- removeInEdges
 - libgexf::DirectedGraph, [26](#)
- removeNode
 - libgexf::Graph, [45](#)
- removeOutEdges
 - libgexf::DirectedGraph, [26](#)

- setEdgeAttributeDefault
 - libgexf::Data, [20](#)
- setEdgeValue
 - libgexf::Data, [21](#)
- setGraphType
 - libgexf::GEXF, [39](#)
- setLabel
 - libgexf::Data, [20](#)
- setNodeAttributeDefault
 - libgexf::Data, [20](#)
- setNodeValue
 - libgexf::Data, [21](#)

- typedefs.h, [91](#)

- undirectedgraph.cpp, [92](#)
- undirectedgraph.h, [93](#)

- writeLock
 - libgexf::Graph, [48](#)