My Project

Generated by Doxygen 1.8.16

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 BAlgoritmosDeOrdenacion Namespace Reference

Functions

- template<class T >
  void algoritmoBurbuja (T ∗list, size_t size)
- template<class T >
  void algoritmoInserccionDirecta (T ∗list, size_t size)

  size_tercambiamos la posicion actual con el valor mas peque
  template<class T >
  void quickSort (T ∗list, size_t size)
  Elegimos un numero al alazar (Mitad de la cadena)
  template<class T >
  void countShort (T ∗list, size_t size)

5.1.1 Function Documentation

5.1.1.1 algoritmoBurbuja()

```
template<class T >
void BAlgoritmosDeOrdenacion::algoritmoBurbuja (
             T ∗ list,
             size_t size )
```

Algoritmo de burbuja Recorremos la lista de numeros comparando el numero actual con el siguiente, y si es el orden incorrecto los cambiamos Hasta que no cambiamos nada

5.1.1.2 algoritmoInserccionDirecta()

```
template<class T >
void BAlgoritmosDeOrdenacion::algoritmoInserccionDirecta (
             T ∗ list,
             size_t size )
```

size_tercambiamos la posicion actual con el valor mas pequelgoritmo de inserccion directa Recorremos la lista buscando el valor mas peque

5.1.1.3 countShort()

```
template<class T >
void BAlgoritmosDeOrdenacion::countShort (
             T * list,
             size_t size )
```

Necesita una lista de auxialr del tipo de dato Contamos cuantas veces aparece cada dato repetido Lista de igual tama

5.1.1.4 quickSort()

```
template<class T >
void BAlgoritmosDeOrdenacion::quickSort (
             T * list,
             size_t size )
```

Elegimos un numero al alazar (Mitad de la cadena)

Algoritmo de quick sort Llamamos a la funcion de ordenacion de quickShort para que lo reordene Cuando queden dos elementos, se retorna ordenandolos

5.2 glt Namespace Reference

5.3 rapidxml Namespace Reference

Classes

- class attribute_iterator

  Iterator of child attributes of xml_node.
- class file

  Represents data loaded from a file.
- class memory_pool
- class node_iterator

  Iterator of child nodes of xml_node.
- class parse_error
- class xml_attribute
- class xml_base
- class xml_document
- class xml_node

Enumerations

- enum node_type {
  node_document, node_element, node_data, node_cdata,
  node_comment, node_declaration, node_doctype, node_pi }

Functions

- template<class OutIt , class Ch >
  OutIt print (OutIt out, const xml_node< Ch > &node, int flags=0)
- template<class Ch >
  std::basic_ostream< Ch > & print (std::basic_ostream< Ch > &out, const xml_node< Ch > &node, int flags=0)
- template<class Ch >
  std::basic_ostream< Ch > & operator<< (std::basic_ostream< Ch > &out, const xml_node< Ch > &node)

- template<class Ch >
  std::size_t count_children (xml_node< Ch > *node)
- template<class Ch >
  std::size_t count_attributes (xml_node< Ch > *node)

Variables

- const int parse_no_data_nodes = 0x1
- const int parse_no_element_values = 0x2
- const int parse_no_string_terminators = 0x4
- const int parse_no_entity_translation = 0x8
- const int parse_no_utf8 = 0x10
- const int parse_declaration_node = 0x20
- const int parse_comment_nodes = 0x40
- const int parse_doctype_node = 0x80
- const int parse_pi_nodes = 0x100
- const int parse_validate_closing_tags = 0x200
- const int parse_trim_whitespace = 0x400
- const int parse_normalize_whitespace = 0x800
- const int parse_default = 0
- const int parse_non_destructive = parse_no_string_terminators | parse_no_entity_translation
- const int parse_fastest = parse_non_destructive | parse_no_data_nodes
- const int parse_full = parse_declaration_node | parse_comment_nodes | parse_doctype_node | parse_pi_nodes | parse_validate_closing_tags
- const int print_no_indenting = 0x1

  Printer flag instructing the printer to suppress indenting of XML. See print() function.

5.3.1 Enumeration Type Documentation

5.3.1.1 node_type

```
enum rapidxml::node_type
```

Enumeration listing all node types produced by the parser. Use xml_node::type() function to query node type.

Enumerator

node_document	A document node. Name and value are empty.
node_element	An element node. Name contains element name. Value contains text of first data node.
node_data	A data node. Name is empty. Value contains data text.
node_cdata	A CDATA node. Name is empty. Value contains data text.
node_comment	A comment node. Name is empty. Value contains comment text.
node_declaration	A declaration node. Name and value are empty. Declaration parameters (version, encoding and standalone) are in node attributes.
node_doctype	A DOCTYPE node. Name is empty. Value contains DOCTYPE text.
node_pi	A PI node. Name contains target. Value contains instructions.

5.3.2 Function Documentation

5.3.2.1 count_attributes()

```
template<class Ch >
std::size_t rapidxml::count_attributes (
             xml_node< Ch > * node )  [inline]
```

Counts attributes of node. Time complexity is O(n).

Returns

Number of attributes of node

5.3.2.2 count_children()

```
template<class Ch >
std::size_t rapidxml::count_children (
             xml_node< Ch > * node )  [inline]
```

Counts children of node. Time complexity is O(n).

Returns

Number of children of node

5.3.2.3 operator<<()

```
template<class Ch >
std::basic_ostream<Ch>& rapidxml::operator<< (
             std::basic_ostream< Ch > & out,
             const xml_node< Ch > & node )  [inline]
```

Prints formatted XML to given output stream. Uses default printing flags. Use print() function to customize printing process.

Parameters

out	Output stream to print to.
node	Node to be printed.

Returns

Output stream.

5.3.2.4 print() [1/2]

```
template<class OutIt , class Ch >
OutIt rapidxml::print (
             OutIt out,
             const xml_node< Ch > & node,
             int flags = 0 )  [inline]
```

Prints XML to given output iterator.

Parameters

out	Output iterator to print to.

Parameters

node	Node to be printed. Pass xml_document to print entire document.
flags	Flags controlling how XML is printed.

Returns

Output iterator pointing to position immediately after last character of printed text.

5.3.2.5 print() [2/2]

```
template<class Ch >
std::basic_ostream<Ch>& rapidxml::print (
             std::basic_ostream< Ch > & out,
             const xml_node< Ch > & node,
             int flags = 0 )  [inline]
```

Prints XML to given output stream.

Parameters

out	Output stream to print to.
node	Node to be printed. Pass xml_document to print entire document.
flags	Flags controlling how XML is printed.

Returns

Output stream.

5.3.3 Variable Documentation

5.3.3.1 parse_comment_nodes

```
const int rapidxml::parse_comment_nodes = 0x40
```

Parse flag instructing the parser to create comments nodes. By default, comment nodes are not created. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.2 parse_declaration_node

```
const int rapidxml::parse_declaration_node = 0x20
```

Parse flag instructing the parser to create XML declaration node. By default, declaration node is not created. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.3 parse_default

```
const int rapidxml::parse_default = 0
```

Parse flags which represent default behaviour of the parser. This is always equal to 0, so that all other flags can be simply ored together. Normally there is no need to inconveniently disable flags by anding with their negated (∼) values. This also means that meaning of each flag is a *negation* of the default setting. For example, if flag name is rapidxml::parse_no_utf8, it means that utf-8 is *enabled* by default, and using the flag will disable it.

See xml_document::parse() function.

5.3.3.4 parse_doctype_node

```
const int rapidxml::parse_doctype_node = 0x80
```

Parse flag instructing the parser to create DOCTYPE node. By default, doctype node is not created. Although W3C specification allows at most one DOCTYPE node, RapidXml will silently accept documents with more than one. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.5 parse_fastest

```
const int rapidxml::parse_fastest = parse_non_destructive | parse_no_data_nodes
```

A combination of parse flags resulting in fastest possible parsing, without sacrificing important data.

See xml_document::parse() function.

5.3.3.6 parse_full

```
const int rapidxml::parse_full = parse_declaration_node | parse_comment_nodes | parse_doctype_node
| parse_pi_nodes | parse_validate_closing_tags
```

A combination of parse flags resulting in largest amount of data being extracted. This usually results in slowest parsing.

See xml_document::parse() function.

5.3.3.7 parse_no_data_nodes

```
const int rapidxml::parse_no_data_nodes = 0x1
```

Parse flag instructing the parser to not create data nodes. Text of first data node will still be placed in value of parent element, unless rapidxml::parse_no_element_values flag is also specified. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.8 parse_no_element_values

```
const int rapidxml::parse_no_element_values = 0x2
```

Parse flag instructing the parser to not use text of first data node as a value of parent element. Can be combined with other flags by use of | operator. Note that child data nodes of element node take precendence over its value when printing. That is, if element has one or more child data nodes *and* a value, the value will be ignored. Use rapidxml::parse_no_data_nodes flag to prevent creation of data nodes if you want to manipulate data using values of elements.

See xml_document::parse() function.

5.3.3.9 parse_no_entity_translation

```
const int rapidxml::parse_no_entity_translation = 0x8
```

Parse flag instructing the parser to not translate entities in the source text. By default entities are translated, modifying source text. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.10 parse_no_string_terminators

```
const int rapidxml::parse_no_string_terminators = 0x4
```

Parse flag instructing the parser to not place zero terminators after strings in the source text. By default zero terminators are placed, modifying source text. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.11 parse_no_utf8

```
const int rapidxml::parse_no_utf8 = 0x10
```

Parse flag instructing the parser to disable UTF-8 handling and assume plain 8 bit characters. By default, UTF-8 handling is enabled. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.12 parse_non_destructive

```
const int rapidxml::parse_non_destructive = parse_no_string_terminators | parse_no_entity_translation
```

A combination of parse flags that forbids any modifications of the source text. This also results in faster parsing. However, note that the following will occur:

- names and values of nodes will not be zero terminated, you have to use xml_base::name_size() and xml_base::value_size() functions to determine where name and value ends
- entities will not be translated
- whitespace will not be normalized

See xml_document::parse() function.

5.3.3.13 parse_normalize_whitespace

```
const int rapidxml::parse_normalize_whitespace = 0x800
```

Parse flag instructing the parser to condense all whitespace runs of data nodes to a single space character. Trimming of leading and trailing whitespace of data is controlled by rapidxml::parse_trim_whitespace flag. By default, whitespace is not normalized. If this flag is specified, source text will be modified. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.14 parse_pi_nodes

```
const int rapidxml::parse_pi_nodes = 0x100
```

Parse flag instructing the parser to create PI nodes. By default, PI nodes are not created. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.15 parse_trim_whitespace

```
const int rapidxml::parse_trim_whitespace = 0x400
```

Parse flag instructing the parser to trim all leading and trailing whitespace of data nodes. By default, whitespace is not trimmed. This flag does not cause the parser to modify source text. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.16 parse_validate_closing_tags

```
const int rapidxml::parse_validate_closing_tags = 0x200
```

Parse flag instructing the parser to validate closing tag names. If not set, name inside closing tag is irrelevant to the parser. By default, closing tags are not validated. Can be combined with other flags by use of | operator.

See xml_document::parse() function.

5.3.3.17 print_no_indenting

```
const int rapidxml::print_no_indenting = 0x1
```

Printer flag instructing the printer to suppress indenting of XML. See print() function.

Chapter 6

Class Documentation

6.1 rapidxml::attribute_iterator< Ch > Class Template Reference

Iterator of child attributes of xml_node.

`#include <rapidxml_iterators.hpp>`

Public Types

- typedef xml_attribute< Ch > value_type
- typedef xml_attribute< Ch > & reference
- typedef xml_attribute< Ch > * pointer
- typedef std::ptrdiff_t difference_type
- typedef std::bidirectional_iterator_tag iterator_category

Public Member Functions

- attribute_iterator ()
- attribute_iterator (xml_node< Ch > *node)
- reference operator* () const
- pointer operator-> () const
- attribute_iterator & operator++ ()
- attribute_iterator operator++ (int)
- attribute_iterator & operator-- ()
- attribute_iterator operator-- (int)
- bool operator== (const attribute_iterator< Ch > &rhs)
- bool operator!= (const attribute_iterator< Ch > &rhs)

6.1.1 Detailed Description

template< class Ch >
class rapidxml::attribute_iterator< Ch >

Iterator of child attributes of xml_node.

6.1.2 Member Typedef Documentation

6.1.2.1 difference_type

```
template<class Ch>
typedef std::ptrdiff_t rapidxml::attribute_iterator< Ch >::difference_type
```

6.1.2.2 iterator_category

```
template<class Ch>
typedef std::bidirectional_iterator_tag rapidxml::attribute_iterator< Ch >::iterator_category
```

6.1.2.3 pointer

```
template<class Ch>
typedef xml_attribute<Ch>* rapidxml::attribute_iterator< Ch >::pointer
```

6.1.2.4 reference

```
template<class Ch>
typedef xml_attribute<Ch>& rapidxml::attribute_iterator< Ch >::reference
```

6.1.2.5 value_type

```
template<class Ch>
typedef xml_attribute<Ch> rapidxml::attribute_iterator< Ch >::value_type
```

6.1.3 Constructor & Destructor Documentation

6.1.3.1 attribute_iterator() [1/2]

```
template<class Ch>
rapidxml::attribute_iterator< Ch >::attribute_iterator ( )  [inline]
```

6.1.3.2 attribute_iterator() [2/2]

```
template<class Ch>
rapidxml::attribute_iterator< Ch >::attribute_iterator (
             xml_node< Ch > * node )  [inline]
```

6.1.4 Member Function Documentation

6.1.4.1 operator"!=()

```
template<class Ch>
bool rapidxml::attribute_iterator< Ch >::operator!= (
             const attribute_iterator< Ch > & rhs )  [inline]
```

6.1.4.2 operator∗()

```
template<class Ch>
reference rapidxml::attribute_iterator< Ch >::operator* ( ) const  [inline]
```

6.1.4.3 operator++() [1/2]

```
template<class Ch>
attribute_iterator& rapidxml::attribute_iterator< Ch >::operator++ ( )  [inline]
```

6.1.4.4 operator++() [2/2]

```
template<class Ch>
attribute_iterator rapidxml::attribute_iterator< Ch >::operator++ (
             int  )  [inline]
```

6.1.4.5 operator--() [1/2]

```
template<class Ch>
attribute_iterator& rapidxml::attribute_iterator< Ch >::operator-- ( )  [inline]
```

6.1.4.6 operator--() [2/2]

```
template<class Ch>
attribute_iterator rapidxml::attribute_iterator< Ch >::operator-- (
             int  )  [inline]
```

6.1.4.7 operator->()

```
template<class Ch>
pointer rapidxml::attribute_iterator< Ch >::operator-> ( ) const  [inline]
```

6.1.4.8 operator==()

```
template<class Ch>
bool rapidxml::attribute_iterator< Ch >::operator== (
             const attribute_iterator< Ch > & rhs )  [inline]
```

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

– D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml_iterators.hpp

6.2 BAudio Class Reference

```
#include <BAudio.hpp>
```

Classes

- struct BAudioInfo

Public Member Functions

- string setRelativePath (const char ∗_path)
- BAudio ()
- ∼BAudio ()
- Id loadMusic (const char ∗path)
- int loadSound (const char ∗path)
- int startMusic (Id id)
- int makeSound (Id id)
- void stopAllMusic ()
- void stopMusicId (Id id)
- void stopAllSounds ()
- void stopChanelId (Id id)
- void setMusicVolume (Id id, int volume)
- void setSoundVolume (Id id, int volume)

6.2.1 Constructor & Destructor Documentation

6.2.1.1 BAudio()

```
BAudio::BAudio ( )
```

6.2.1.2 ∼BAudio()

```
BAudio::∼BAudio ( )
```

6.2.2 Member Function Documentation

6.2.2.1 loadMusic()

```
Id BAudio::loadMusic (
             const char ∗ path )
```

6.2.2.2 loadSound()

```
int BAudio::loadSound (
             const char ∗ path )
```

6.2.2.3 makeSound()

```
int BAudio::makeSound (
             Id id )
```

6.2.2.4 setMusicVolume()

```
void BAudio::setMusicVolume (
             Id id,
             int volume )
```

6.2.2.5 setRelativePath()

```
string BAudio::setRelativePath (
             const char * _path )  [inline]
```

6.2.2.6 setSoundVolume()

```
void BAudio::setSoundVolume (
             Id id,
             int volume )
```

6.2.2.7 startMusic()

```
int BAudio::startMusic (
             Id id )
```

6.2.2.8 stopAllMusic()

```
void BAudio::stopAllMusic ( )
```

6.2.2.9 stopAllSounds()

```
void BAudio::stopAllSounds ( )
```

6.2.2.10 stopChanelId()

```
void BAudio::stopChanelId (
             Id id )
```

6.2.2.11 stopMusicId()

```
void BAudio::stopMusicId (
             Id id )
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BAudio.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BAudio.cpp

6.3 BAudio::BAudioInfo Struct Reference

```
#include <BAudio.hpp>
```

Public Member Functions

- BAudioInfo (Mix_Music ∗_music)
- BAudioInfo (Mix_Chunk ∗_sound)

Public Attributes

- Mix_Music ∗ music = nullptr
- Mix_Chunk ∗ sound = nullptr
- int channel = -1

6.3.1 Constructor & Destructor Documentation

6.3.1.1 BAudioInfo() [1/2]

```
BAudio::BAudioInfo::BAudioInfo (
            Mix_Music ∗ _music )  [inline]
```

6.3.1.2 BAudioInfo() [2/2]

```
BAudio::BAudioInfo::BAudioInfo (
            Mix_Chunk ∗ _sound )  [inline]
```

6.3.2 Member Data Documentation

6.3.2.1 channel

```
int BAudio::BAudioInfo::channel = -1
```

6.3.2.2 music

```
Mix_Music∗ BAudio::BAudioInfo::music = nullptr
```

6.3.2.3 sound

```
Mix_Chunk∗ BAudio::BAudioInfo::sound = nullptr
```

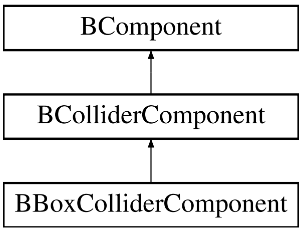
The documentation for this struct was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BAudio.hpp

6.4 BBoxColliderComponent Class Reference

```
#include <BBoxColliderComponent.hpp>
```

Inheritance diagram for BBoxColliderComponent:



Public Member Functions

- BBoxColliderComponent (shared_ptr< BEntity > parent)
- bool initialize ()
- bool parse_property (const string &name, const string &value)
- shared_ptr< BEntity > checkCollisions (shared_ptr< BEntity > other)

Public Attributes

- vec3< float > MaxOffset
- vec3< float > MinOffset

Additional Inherited Members

6.4.1 Constructor & Destructor Documentation

6.4.1.1 BBoxColliderComponent()

```
BBoxColliderComponent::BBoxColliderComponent (
            shared_ptr< BEntity > parent )
```

6.4.2 Member Function Documentation

6.4.2.1 checkCollisions()

```
shared_ptr< BEntity > BBoxColliderComponent::checkCollisions (
            shared_ptr< BEntity > other ) [virtual]
```

Implements BColliderComponent.

6.4.2.2 initialize()

```
bool BBoxColliderComponent::initialize ( ) [inline], [virtual]
```

Implements BColliderComponent.

6.4.2.3 parse_property()

```
bool BBoxColliderComponent::parse_property (
            const string & name,
            const string & value )  [inline], [virtual]
```

Implements BColliderComponent.

6.4.3 Member Data Documentation

6.4.3.1 MaxOffset

```
vec3<float> BBoxColliderComponent::MaxOffset
```

6.4.3.2 MinOffset

```
vec3<float> BBoxColliderComponent::MinOffset
```

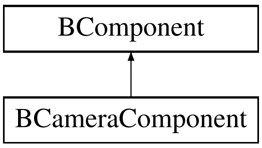
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BBoxColliderComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BBoxColliderComponent.cpp

6.5 BCameraComponent Class Reference

```
#include <BCameraComponent.hpp>
```

Inheritance diagram for BCameraComponent:



Public Member Functions

- BCameraComponent (shared_ptr< BEntity > parent)
- virtual bool initialize ()
- virtual bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.5.1 Constructor & Destructor Documentation

6.5.1.1 BCameraComponent()

```
BCameraComponent::BCameraComponent (
            shared_ptr< BEntity > parent )
```

6.5.2 Member Function Documentation

6.5.2.1 initialize()

```
bool BCameraComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.5.2.2 parse_property()

```
virtual bool BCameraComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

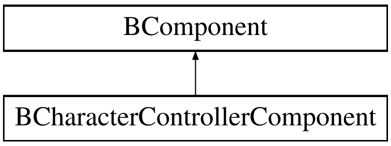
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BCameraComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BCameraComponent.cpp

6.6 BCharacterControllerComponent Class Reference

```
#include <BCharacterController.hpp>
```

Inheritance diagram for BCharacterControllerComponent:



Public Member Functions

- BCharacterControllerComponent (shared_ptr< BEntity > parent)
- virtual bool initialize ()
- virtual bool parse_property (const string &name, const string &value)

Public Attributes

- string Up
- string Down
- string Left
- string Right
- float speed

Additional Inherited Members

6.6.1 Constructor & Destructor Documentation

6.6.1.1 BCharacterControllerComponent()

```
BCharacterControllerComponent::BCharacterControllerComponent (
            shared_ptr< BEntity > parent )
```

6.6.2 Member Function Documentation

6.6.2.1 initialize()

```
bool BCharacterControllerComponent::initialize ( ) [virtual]
```

Implements BComponent.

6.6.2.2 parse_property()

```
virtual bool BCharacterControllerComponent::parse_property (
            const string & name,
            const string & value ) [inline], [virtual]
```

Implements BComponent.

6.6.3 Member Data Documentation

6.6.3.1 Down

```
string BCharacterControllerComponent::Down
```

6.6.3.2 Left

```
string BCharacterControllerComponent::Left
```

6.6.3.3 Right

```
string BCharacterControllerComponent::Right
```

6.6.3.4 speed

```
float BCharacterControllerComponent::speed
```

6.6.3.5 Up

```
string BCharacterControllerComponent::Up
```

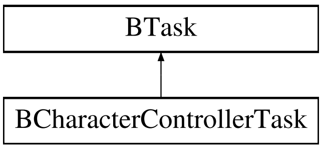
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BCharacterController.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BCharacterController.cpp

6.7 BCharacterControllerTask Class Reference

```
#include <BCharacterControllerTask.hpp>
```

Inheritance diagram for BCharacterControllerTask:



Public Member Functions

- BCharacterControllerTask (shared_ptr< BEntity > transfom, shared_ptr< BCharacterControllerComponent > component)

Additional Inherited Members

6.7.1 Constructor & Destructor Documentation

6.7.1.1 BCharacterControllerTask()

```
BCharacterControllerTask::BCharacterControllerTask (
            shared_ptr< BEntity > transfom,
            shared_ptr< BCharacterControllerComponent > component )
```

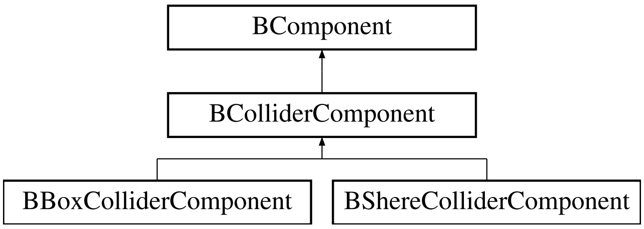
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BCharacterControllerTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BCharacterControllerTask.cpp

6.8 BColliderComponent Class Reference

```
#include <BColliderComponent.hpp>
```

Inheritance diagram for BColliderComponent:



Public Member Functions

- BColliderComponent (shared_ptr< BEntity > parent)
- void setFunction (std::function< void(shared_ptr< BEntity >, shared_ptr< BEntity >)> myFunction)
- virtual bool initialize ()=0
- virtual bool parse_property (const string &name, const string &value)=0
- virtual shared_ptr< BEntity > checkCollisions (shared_ptr< BEntity > other)=0
- COLLIDERTYPE getType ()

Protected Attributes

- COLLIDERTYPE type

6.8.1 Constructor & Destructor Documentation

6.8.1.1 BColliderComponent()

```
BColliderComponent::BColliderComponent (
             shared_ptr< BEntity > parent )
```

6.8.2 Member Function Documentation

6.8.2.1 checkCollisions()

```
virtual shared_ptr<BEntity> BColliderComponent::checkCollisions (
             shared_ptr< BEntity > other )  [pure virtual]
```

Implemented in BBoxColliderComponent, and BShereColliderComponent.

6.8.2.2 getType()

```
COLLIDERTYPE BColliderComponent::getType ( )  [inline]
```

6.8.2.3 initialize()

```
virtual bool BColliderComponent::initialize ( )  [pure virtual]
```

Implements BComponent.

Implemented in BBoxColliderComponent, and BShereColliderComponent.

6.8.2.4 parse_property()

```
virtual bool BColliderComponent::parse_property (
             const string & name,
             const string & value )  [pure virtual]
```

Implements BComponent.

Implemented in BBoxColliderComponent, and BShereColliderComponent.

6.8.2.5 setFunction()

```
void BColliderComponent::setFunction (
             std::function< void(shared_ptr< BEntity > ,
             shared_ptr< BEntity >  )
```

6.8.3 Member Data Documentation

6.8.3.1 type

```
COLLIDERTYPE BColliderComponent::type [protected]
```

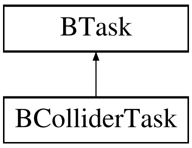
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BColliderComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BColliderComponent.cpp

6.9 BColliderTask Class Reference

```
#include <BSphereColliderTask.hpp>
```

Inheritance diagram for BColliderTask:



Public Member Functions

- BColliderTask (shared_ptr< BEntity > transfom, shared_ptr< BScene > scene)

Public Attributes

- shared_ptr< BEntity > entity
- shared_ptr< BScene > scene
- std::function< void(shared_ptr< BEntity >, shared_ptr< BEntity >)> onCollision

Additional Inherited Members

6.9.1 Constructor & Destructor Documentation

6.9.1.1 BColliderTask()

```
BColliderTask::BColliderTask (
            shared_ptr< BEntity > transfom,
            shared_ptr< BScene > scene )
```

6.9.2 Member Data Documentation

6.9.2.1 entity

```
shared_ptr<BEntity> BColliderTask::entity
```

6.9.2.2 onCollision

```
std::function<void(shared_ptr<BEntity>, shared_ptr<BEntity>)> BColliderTask::onCollision
```

6.9.2.3 scene

```
shared_ptr<BScene> BColliderTask::scene
```

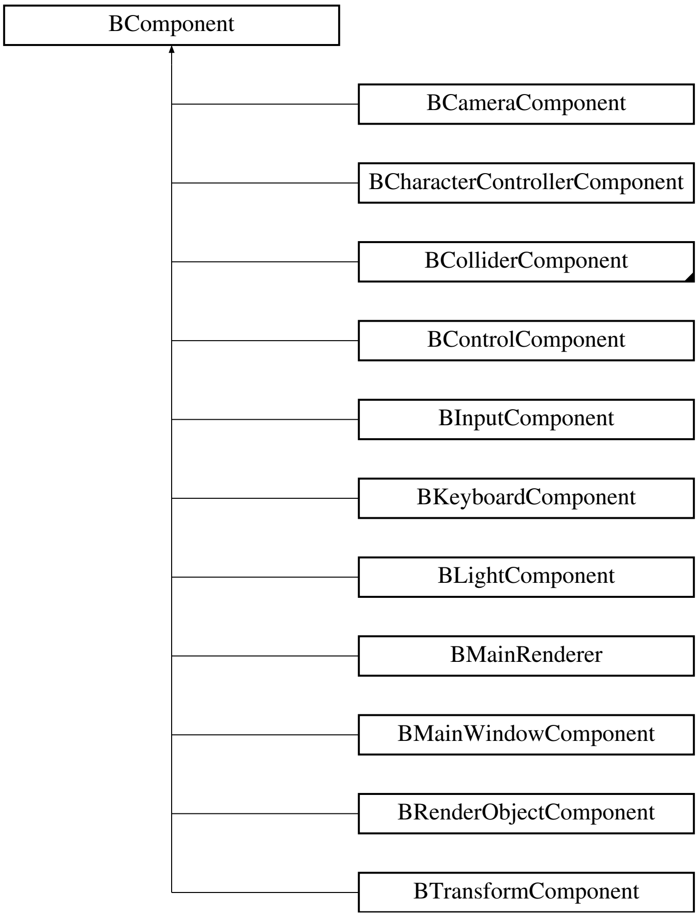
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BSphereColliderTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BColliderTask.cpp

6.10 BComponent Class Reference

```
#include <BComponent.hpp>
```

Inheritance diagram for BComponent:



Public Member Functions

- BComponent (shared_ptr< BEntity > parent)
- virtual ∼BComponent ()
- virtual bool initialize ()=0
- virtual bool parse_property (const string &name, const string &value)=0
- shared_ptr< BTask > getTask ()

Protected Attributes

- string id
- shared_ptr< BEntity > parent
- shared_ptr< BTask > task

6.10.1 Constructor & Destructor Documentation

6.10.1.1 BComponent()

```
BComponent::BComponent (
            shared_ptr< BEntity > parent )  [inline]
```

6.10.1.2 ∼BComponent()

```
virtual BComponent::∼BComponent ( )  [inline], [virtual]
```

6.10.2 Member Function Documentation

6.10.2.1 getTask()

```
shared_ptr<BTask> BComponent::getTask ( )  [inline]
```

6.10.2.2 initialize()

```
virtual bool BComponent::initialize ( )  [pure virtual]
```

Implemented in BColliderComponent, BControlComponent, BCharacterControllerComponent, BCameraComponent, BBoxColliderComponent, BShereColliderComponent, BKeyboardComponent, BRenderObjectComponent, BLightComponent, BTransformComponent, BMainWindowComponent, BInputComponent, and BMainRenderer.

6.10.2.3 parse_property()

```
virtual bool BComponent::parse_property (
            const string & name,
            const string & value )  [pure virtual]
```

Implemented in BColliderComponent, BControlComponent, BBoxColliderComponent, BCharacterControllerComponent, BCameraComponent, BShereColliderComponent, BTransformComponent, BKeyboardComponent, BRenderObjectComponent, BLightComponent, BMainWindowComponent, BInputComponent, and BMainRenderer.

6.10.3 Member Data Documentation

6.10.3.1 id

```
string BComponent::id [protected]
```

6.10.3.2 parent

```
shared_ptr<BEntity> BComponent::parent [protected]
```

6.10.3.3 task

```
shared_ptr<BTask> BComponent::task [protected]
```

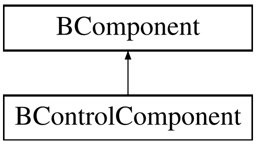
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BComponent.hpp

6.11 BControlComponent Class Reference

```
#include <BControlComponent.hpp>
```

Inheritance diagram for BControlComponent:



Public Member Functions

- BControlComponent (shared_ptr< BEntity > parent)
- void setFunction (std::function< void(float, shared_ptr< BEntity >)> myFunction)
- virtual bool initialize () override
- virtual bool parse_property (const string &name, const string &value) override

Additional Inherited Members

6.11.1 Constructor & Destructor Documentation

6.11.1.1 BControlComponent()

```
BControlComponent::BControlComponent (
            shared_ptr< BEntity > parent )
```

6.11.2 Member Function Documentation

6.11.2.1 initialize()

```
bool BControlComponent::initialize ( )  [override], [virtual]
```

Implements BComponent.

6.11.2.2 parse_property()

```
bool BControlComponent::parse_property (
            const string & name,
            const string & value )  [override], [virtual]
```

Implements BComponent.

6.11.2.3 setFunction()

```
void BControlComponent::setFunction (
            std::function< void(float, shared_ptr< BEntity >  )
```

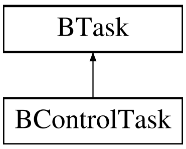
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BControlComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BControlComponent.cpp

6.12 BControlTask Class Reference

```
#include <BControlTask.hpp>
```

Inheritance diagram for BControlTask:



Public Member Functions

- BControlTask (shared_ptr< BEntity > entityReference)

Public Attributes

- shared_ptr< BEntity > entityReference
- std::function< void(float, shared_ptr< BEntity >)> myFunction

Additional Inherited Members

6.12.1 Constructor & Destructor Documentation

6.12.1.1 BControlTask()

```
BControlTask::BControlTask (
            shared_ptr< BEntity > entityReference )
```

6.12.2 Member Data Documentation

6.12.2.1 entityReference

```
shared_ptr<BEntity> BControlTask::entityReference
```

6.12.2.2 myFunction

```
std::function<void(float, shared_ptr<BEntity>)> BControlTask::myFunction
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BControlTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BControlTask.cpp

6.13 BDispacher Class Reference

```
#include <BDispacher.hpp>
```

Public Member Functions

- void add (BOrbserver &o, string id)
- void Send (BMessage &m)

Static Public Member Functions

- static shared_ptr< BDispacher > instance ()

6.13.1 Member Function Documentation

6.13.1.1 add()

```
void BDispacher::add (
             BOrbserver & o,
             string id )  [inline]
```

6.13.1.2 instance()

```
static shared_ptr<BDispacher> BDispacher::instance ( )  [inline], [static]
```

6.13.1.3 Send()

```
void BDispacher::Send (
             BMessage & m )
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BDispacher.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BDispacher.cpp

6.14 BEntity Class Reference

```
#include <BEntity.hpp>
```

Public Member Functions

- BEntity (string id, shared_ptr< BScene > scene)
- bool initialize ()
- shared_ptr< BTransformComponent > getTransform ()
- shared_ptr< BScene > getScene ()
- bool add_component (const string &type, shared_ptr< BComponent > &component)
- template<class T >
  shared_ptr< T > getComponent ()
- const string getId ()
- list< shared_ptr< BComponent > > getComponents ()

Public Attributes

- shared_ptr< BComponent > transform

6.14.1 Constructor & Destructor Documentation

6.14.1.1 BEntity()

```
BEntity::BEntity (
             string id,
             shared_ptr< BScene > scene )
```

6.14.2 Member Function Documentation

6.14.2.1 add_component()

```
bool BEntity::add_component (
             const string & type,
             shared_ptr< BComponent > & component )
```

6.14.2.2 getComponent()

```
template<class T >
shared_ptr<T> BEntity::getComponent ( )  [inline]
```

6.14.2.3 getComponents()

```
list< shared_ptr< BComponent > > BEntity::getComponents ( )
```

6.14.2.4 getId()

```
const string BEntity::getId ( ) [inline]
```

6.14.2.5 getScene()

```
shared_ptr<BScene> BEntity::getScene ( ) [inline]
```

6.14.2.6 getTransform()

```
shared_ptr< BTransformComponent > BEntity::getTransform ( )
```

6.14.2.7 initialize()

```
bool BEntity::initialize ( )
```

6.14.3 Member Data Documentation

6.14.3.1 transform

```
shared_ptr<BComponent> BEntity::transform
```

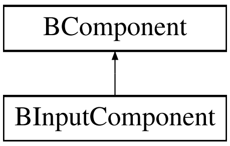
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BEntity.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BEntity.cpp

6.15 BInputComponent Class Reference

```
#include <BInputComponent.hpp>
```

Inheritance diagram for BInputComponent:



Public Member Functions

- BInputComponent (shared_ptr< BEntity > parent)
- virtual bool initialize ()
- virtual bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.15.1 Constructor & Destructor Documentation

6.15.1.1 BInputComponent()

```
BInputComponent::BInputComponent (
             shared_ptr< BEntity > parent )
```

6.15.2 Member Function Documentation

6.15.2.1 initialize()

```
bool BInputComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.15.2.2 parse_property()

```
bool BInputComponent::parse_property (
             const string & name,
             const string & value )  [virtual]
```

Implements BComponent.

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BInputComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BInputComponent.cpp

6.16 BInputMapper Class Reference

```
#include <BInputMapper.hpp>
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BInputMapper.hpp

6.17 BKernel Class Reference

```
#include <BKernel.hpp>
```

Public Member Functions

- BKernel (shared_ptr< BScene > _scene)
- void add_Task (shared_ptr< BTask > task)
- void run ()
- void stop ()
- void pause ()
- void resume ()
- shared_ptr< BScene > getScene ()

6.17.1 Constructor & Destructor Documentation

6.17.1.1 BKernel()

```
BKernel::BKernel (
             shared_ptr< BScene > _scene )  [inline]
```

6.17.2 Member Function Documentation

6.17.2.1 add_Task()

```
void BKernel::add_Task (
             shared_ptr< BTask > task )
```

6.17.2.2 getScene()

```
shared_ptr<BScene> BKernel::getScene ( )  [inline]
```

6.17.2.3 pause()

```
void BKernel::pause ( )  [inline]
```

6.17.2.4 resume()

```
void BKernel::resume ( )  [inline]
```

6.17.2.5 run()

```
void BKernel::run ( )
```

6.17.2.6 stop()

```
void BKernel::stop ( )  [inline]
```

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

– D:/GitHub/BarxEngine/BarxEngine/code/headers/BKernel.hpp
– D:/GitHub/BarxEngine/BarxEngine/code/source/BKernel.cpp

6.18 BKeyboard Class Reference

```
#include <BKeyboard.hpp>
```

Classes

– struct KEYCODE

Public Member Functions

- bool isKeyPresed (string letter)
- void setKeyDown (string letter)
- void setKeyUp (string letter)

Public Attributes

- KEYCODE keyMapper
- list< string > keyPresed

6.18.1 Member Function Documentation

6.18.1.1 isKeyPresed()

```
bool BKeyboard::isKeyPresed (
             string letter )
```

6.18.1.2 setKeyDown()

```
void BKeyboard::setKeyDown (
             string letter )
```

6.18.1.3 setKeyUp()

```
void BKeyboard::setKeyUp (
             string letter )
```

6.18.2 Member Data Documentation

6.18.2.1 keyMapper

```
KEYCODE BKeyboard::keyMapper
```

6.18.2.2 keyPresed

```
list<string> BKeyboard::keyPresed
```

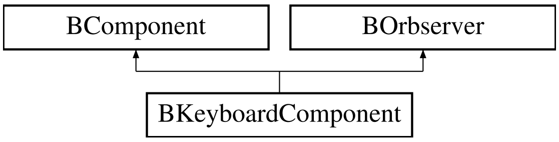
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BKeyboard.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BKeyboard.cpp

6.19 BKeyboardComponent Class Reference

```
#include <BKeyboardComponent.hpp>
```

Inheritance diagram for BKeyboardComponent:



Public Member Functions

- BKeyboardComponent (shared_ptr< BEntity > parent)
- virtual bool initialize ()
- virtual bool parse_property (const string &name, const string &value)
- void handle (const BMessage &m)

Public Attributes

- shared_ptr< BKeyboard > Keyboard

Additional Inherited Members

6.19.1 Constructor & Destructor Documentation

6.19.1.1 BKeyboardComponent()

```
BKeyboardComponent::BKeyboardComponent (
             shared_ptr< BEntity > parent )
```

6.19.2 Member Function Documentation

6.19.2.1 handle()

```
void BKeyboardComponent::handle (
             const BMessage & m )  [virtual]
```

Implements BOrbserver.

6.19.2.2 initialize()

```
bool BKeyboardComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.19.2.3 parse_property()

```
virtual bool BKeyboardComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

6.19.3 Member Data Documentation

6.19.3.1 Keyboard

```
shared_ptr< BKeyboard > BKeyboardComponent::Keyboard
```

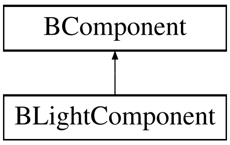
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BKeyboardComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BKeyboardComponent.cpp

6.20 BLightComponent Class Reference

```
#include <BLightComponent.hpp>
```

Inheritance diagram for BLightComponent:



Public Member Functions

- BLightComponent (shared_ptr< BEntity > parent)
- virtual bool initialize ()
- virtual bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.20.1 Constructor & Destructor Documentation

6.20.1.1 BLightComponent()

```
BLightComponent::BLightComponent (
             shared_ptr< BEntity > parent )
```

6.20.2 Member Function Documentation

6.20.2.1 initialize()

```
bool BLightComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.20.2.2 parse_property()

```
virtual bool BLightComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

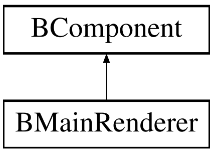
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BLightComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BLightComponent.cpp

6.21 BMainRenderer Class Reference

```
#include <BMainRenderer.hpp>
```

Inheritance diagram for BMainRenderer:



Public Member Functions

- BMainRenderer (shared_ptr< BEntity > parent)
- bool initialize ()
- bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.21.1 Constructor & Destructor Documentation

6.21.1.1 BMainRenderer()

```
BMainRenderer::BMainRenderer (
             shared_ptr< BEntity > parent )
```

6.21.2 Member Function Documentation

6.21.2.1 initialize()

```
bool BMainRenderer::initialize ( )  [virtual]
```

Implements BComponent.

6.21.2.2 parse_property()

```
bool BMainRenderer::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

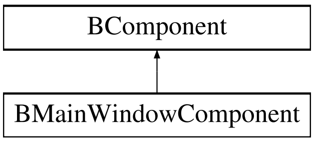
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BMainRenderer.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BMainRenderer.cpp

6.22 BMainWindowComponent Class Reference

`#include <BMainWindowComponent.hpp>`

Inheritance diagram for BMainWindowComponent:



Public Member Functions

- BMainWindowComponent (shared_ptr< BEntity > parent, string windowName="BarxEngine tool", int w=1200, int h=800, bool fs=false)
- bool initialize ()
- bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.22.1 Constructor & Destructor Documentation

6.22.1.1 BMainWindowComponent()

```
BMainWindowComponent::BMainWindowComponent (
             shared_ptr< BEntity > parent,
             string windowName = "BarxEngine tool",
             int w = 1200,
             int h = 800,
             bool fs = false )
```

6.22.2 Member Function Documentation

6.22.2.1 initialize()

```
bool BMainWindowComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.22.2.2 parse_property()

```
bool BMainWindowComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BMainWindowComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BMainWindowComponent.cpp

6.23 BMessage Class Reference

```
#include <BMessage.hpp>
```

Public Member Functions

- BMessage (const string &id)
- void add_parameter (const string &name, string value)
- const string getId ()

Public Attributes

- string id
- map< string, string > parameters

6.23.1 Constructor & Destructor Documentation

6.23.1.1 BMessage()

```
BMessage::BMessage (
             const string & id )  [inline]
```

6.23.2 Member Function Documentation

6.23.2.1 add_parameter()

```
void BMessage::add_parameter (
             const string & name,
             string value )  [inline]
```

6.23.2.2 getId()

```
const string BMessage::getId ( )  [inline]
```

6.23.3 Member Data Documentation

6.23.3.1 id

```
string BMessage::id
```

6.23.3.2 parameters

```
map< string, string > BMessage::parameters
```

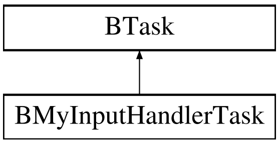
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BMessage.hpp

6.24 BMyInputHandlerTask Class Reference

```
#include <BInputHandlerTask.hpp>
```

Inheritance diagram for BMyInputHandlerTask:



Public Member Functions

- BMyInputHandlerTask (bool active)

Additional Inherited Members

6.24.1 Constructor & Destructor Documentation

6.24.1.1 BMyInputHandlerTask()

```
BMyInputHandlerTask::BMyInputHandlerTask (
            bool active )
```

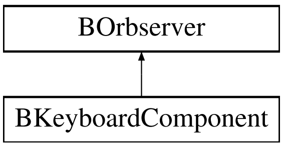
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BInputHandlerTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BInputHandlerTask.cpp

6.25 BOrbserver Class Reference

```
#include <BObserver.hpp>
```

Inheritance diagram for BOrbserver:



Public Member Functions

- virtual void handle (const BMessage &m)=0

6.25.1 Member Function Documentation

6.25.1.1 handle()

```
virtual void BOrbserver::handle (
            const BMessage & m )  [pure virtual]
```

Implemented in BKeyboardComponent.

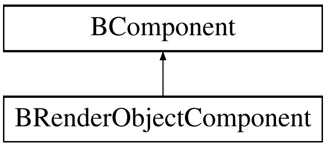
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BObserver.hpp

6.26 BRenderObjectComponent Class Reference

```
#include <BRenderObjectComponent.hpp>
```

Inheritance diagram for BRenderObjectComponent:



Public Member Functions

- BRenderObjectComponent (shared_ptr< BEntity > parent)
- bool initialize ()
- bool parse_property (const string &name, const string &value)

Additional Inherited Members

6.26.1 Constructor & Destructor Documentation

6.26.1.1 BRenderObjectComponent()

```
BRenderObjectComponent::BRenderObjectComponent (
            shared_ptr< BEntity > parent )
```

6.26.2 Member Function Documentation

6.26.2.1 initialize()

```
bool BRenderObjectComponent::initialize ( )  [virtual]
```

Implements BComponent.

6.26.2.2 parse_property()

```
bool BRenderObjectComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

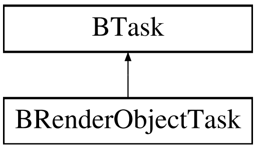
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BRenderObjectComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BRenderObjectComponent.cpp

6.27 BRenderObjectTask Class Reference

```
#include <BRenderObjectTask.hpp>
```

Inheritance diagram for BRenderObjectTask:



Public Member Functions

- BRenderObjectTask (string id, shared_ptr< BRenderTask > instance, shared_ptr< BTransformComponent > transformComponent)

Additional Inherited Members

6.27.1 Constructor & Destructor Documentation

6.27.1.1 BRenderObjectTask()

```
BRenderObjectTask::BRenderObjectTask (
             string id,
             shared_ptr< BRenderTask > instance,
             shared_ptr< BTransformComponent > transformComponent )
```

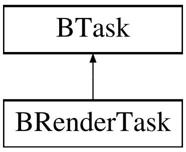
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BRenderObjectTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BRenderObjectTask.cpp

6.28 BRenderTask Class Reference

```
#include <BRenderTask.hpp>
```

Inheritance diagram for BRenderTask:



Public Member Functions

- BRenderTask (shared_ptr< BWindowTask > given_window)
- ∼BRenderTask ()
- void render ()
- shared_ptr< glt::Render_Node > getRenderer ()
- shared_ptr< BWindowTask > getWindow ()
- virtual bool initialize ()
- virtual bool finalize ()
- virtual bool execute (float time)

Static Public Attributes

- static shared_ptr< BRenderTask > instance = nullptr

Additional Inherited Members

6.28.1 Constructor & Destructor Documentation

6.28.1.1 BRenderTask()

```
BRenderTask::BRenderTask (
            shared_ptr< BWindowTask > given_window )
```

6.28.1.2 ∼BRenderTask()

```
BRenderTask::∼BRenderTask ( )
```

En este caso es necesario definir expltamente el destructor en el archivo de implementaciPP) para que el compilador pueda destruir el Render_Node. Si se deja que el compilador cree un destructor por defecto en el programa que use el engine, como solo tendra declaracielantada, no sabrmo destruirlo y ello dargar a un error de compilaci

6.28.2 Member Function Documentation

6.28.2.1 execute()

```
bool BRenderTask::execute (
            float time ) [virtual]
```

Implements BTask.

6.28.2.2 finalize()

```
bool BRenderTask::finalize ( ) [virtual]
```

Implements BTask.

6.28.2.3 getRenderer()

```
std::shared_ptr< glt::Render_Node > BRenderTask::getRenderer ( )
```

6.28.2.4 getWindow()

```
shared_ptr< BWindowTask > BRenderTask::getWindow ( )
```

6.28.2.5 initialize()

```
bool BRenderTask::initialize ( )  [virtual]
```

Implements BTask.

6.28.2.6 render()

```
void BRenderTask::render ( )
```

6.28.3 Member Data Documentation

6.28.3.1 instance

```
shared_ptr< BRenderTask > BRenderTask::instance = nullptr  [static]
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BRenderTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BRenderTask.cpp

6.29 BScene Class Reference

```
#include <BScene.hpp>
```

Public Member Functions

- BScene (const string &scene_description_file_path="")
- shared_ptr< BEntity > getEntity (string id)
- void run ()
- void reloadScene (const string &scene_description_file_path)
- template<class T >
  list< shared_ptr< BEntity > > entitesWithComponent ()
- shared_ptr< BDispacher > getDispacher ()
- shared_ptr< BEntity > getRootEntity ()
- shared_ptr< BKeyboard > getKeyBoardInput ()

6.29.1 Constructor & Destructor Documentation

6.29.1.1 BScene()

```
BScene::BScene (
            const string & scene_description_file_path = "" )
```

6.29.2 Member Function Documentation

6.29.2.1 entitesWithComponent()

```
template<class T >
list<shared_ptr<BEntity> > BScene::entitesWithComponent ( )  [inline]
```

6.29.2.2 getDispacher()

```
shared_ptr< BDispacher > BScene::getDispacher ( )  [inline]
```

6.29.2.3 getEntity()

```
shared_ptr< BEntity > BScene::getEntity (
            string id )
```

6.29.2.4 getKeyBoardInput()

```
shared_ptr< BKeyboard > BScene::getKeyBoardInput ( )
```

6.29.2.5 getRootEntity()

```
shared_ptr<BEntity> BScene::getRootEntity ( )  [inline]
```

6.29.2.6 reloadScene()

```
void BScene::reloadScene (
            const string & scene_description_file_path )
```

6.29.2.7 run()

```
void BScene::run ( )
```

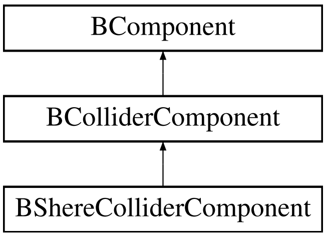
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BScene.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BScene.cpp

6.30 BShereColliderComponent Class Reference

```
#include <BShereColliderComponent.hpp>
```

Inheritance diagram for BShereColliderComponent:



Public Member Functions

- BShereColliderComponent (shared_ptr< BEntity > parent)
- bool initialize ()
- bool parse_property (const string &name, const string &value)
- shared_ptr< BEntity > checkCollisions (shared_ptr< BEntity > other)

Public Attributes

- float radius

Additional Inherited Members

6.30.1 Constructor & Destructor Documentation

6.30.1.1 BShereColliderComponent()

```
BShereColliderComponent::BShereColliderComponent (
            shared_ptr< BEntity > parent )
```

6.30.2 Member Function Documentation

6.30.2.1 checkCollisions()

```
shared_ptr< BEntity > BShereColliderComponent::checkCollisions (
            shared_ptr< BEntity > other ) [virtual]
```

Implements BColliderComponent.

6.30.2.2 initialize()

```
bool BShereColliderComponent::initialize ( ) [inline], [virtual]
```

Implements BColliderComponent.

6.30.2.3 parse_property()

```
bool BShereColliderComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BColliderComponent.

6.30.3 Member Data Documentation

6.30.3.1 radius

```
float BShereColliderComponent::radius
```

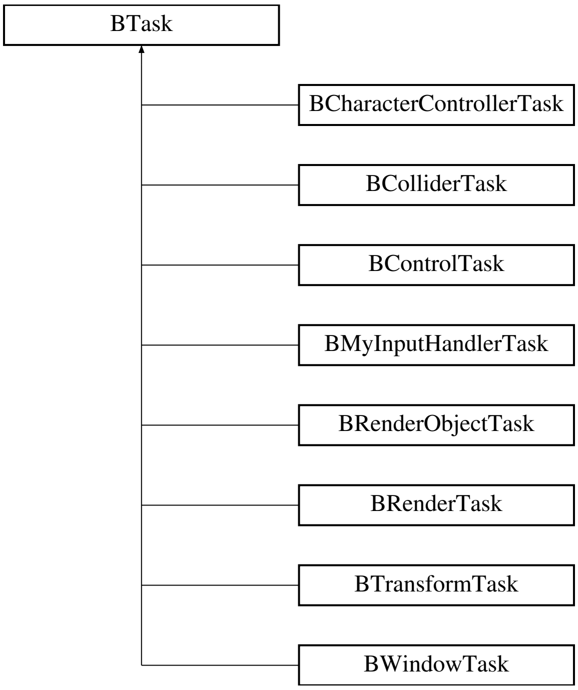
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BShereColliderComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BShereColliderComponent.cpp

6.31 BTask Class Reference

```
#include <BTask.hpp>
```

Inheritance diagram for BTask:



Public Member Functions

- BTask (int priority=0)
- virtual ∼BTask ()
- void set_kernel (BKernel ∗new_kernel)
- virtual bool initialize ()=0
- virtual bool finalize ()=0
- virtual bool execute (float time)=0
- bool operator< (const BTask &other) const

Public Attributes

- int priority

Protected Attributes

- BKernel ∗ kernel
- string id

6.31.1 Constructor & Destructor Documentation

6.31.1.1 BTask()

```
BTask::BTask (
             int priority = 0 )  [inline]
```

6.31.1.2 ∼BTask()

```
virtual BTask::∼BTask ( )  [inline], [virtual]
```

6.31.2 Member Function Documentation

6.31.2.1 execute()

```
virtual bool BTask::execute (
             float time )  [pure virtual]
```

Implemented in BWindowTask, and BRenderTask.

6.31.2.2 finalize()

```
virtual bool BTask::finalize ( )  [pure virtual]
```

Implemented in BWindowTask, and BRenderTask.

6.31.2.3 initialize()

```
virtual bool BTask::initialize ( )  [pure virtual]
```

Implemented in BWindowTask, and BRenderTask.

6.31.2.4 operator<()

```
bool BTask::operator< (
             const BTask & other ) const  [inline]
```

6.31.2.5 set_kernel()

```
void BTask::set_kernel (
             BKernel * new_kernel )  [inline]
```

6.31.3 Member Data Documentation

6.31.3.1 id

```
string BTask::id  [protected]
```

6.31.3.2 kernel

```
BKernel* BTask::kernel  [protected]
```

6.31.3.3 priority

```
int BTask::priority
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BTask.hpp

6.32 BTimer Class Reference

```
#include <BTimer.hpp>
```

Public Member Functions

- BTimer ()
- void start ()
- float elapsed_seconds () const
- uint32_t elapsed_milliseconds () const
- float timeDeltatime ()

6.32.1 Constructor & Destructor Documentation

6.32.1.1 BTimer()

```
BTimer::BTimer ( )  [inline]
```

6.32.2 Member Function Documentation

6.32.2.1 elapsed_milliseconds()

```
uint32_t BTimer::elapsed_milliseconds ( ) const
```

6.32.2.2 elapsed_seconds()

```
float BTimer::elapsed_seconds ( ) const  [inline]
```

6.32.2.3 start()

```
void BTimer::start ( )
```

6.32.2.4 timeDeltatime()

```
float BTimer::timeDeltatime ( )
```

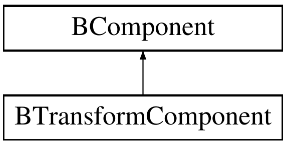
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BTimer.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BTimer.cpp

6.33 BTransformComponent Class Reference

```
#include <BTransformComponent.hpp>
```

Inheritance diagram for BTransformComponent:



Public Member Functions

- BTransformComponent (shared_ptr< BEntity > parent)
- bool initialize ()
- bool parse_property (const string &name, const string &value)

Public Attributes

- vec3< float > position
- vec3< float > rotation
- vec3< float > scale

Additional Inherited Members

6.33.1 Constructor & Destructor Documentation

6.33.1.1 BTransformComponent()

```
BTransformComponent::BTransformComponent (
             shared_ptr< BEntity > parent )
```

6.33.2 Member Function Documentation

6.33.2.1 initialize()

```
bool BTransformComponent::initialize ( )  [inline], [virtual]
```

Implements BComponent.

6.33.2.2 parse_property()

```
bool BTransformComponent::parse_property (
             const string & name,
             const string & value )  [inline], [virtual]
```

Implements BComponent.

6.33.3 Member Data Documentation

6.33.3.1 position

```
vec3<float> BTransformComponent::position
```

6.33.3.2 rotation

```
vec3<float> BTransformComponent::rotation
```

6.33.3.3 scale

```
vec3<float> BTransformComponent::scale
```

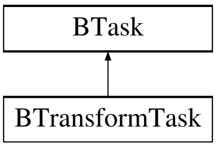
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BTransformComponent.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BTransformComponent.cpp

6.34 BTransformTask Class Reference

```
#include <BTranformTask.hpp>
```

Inheritance diagram for BTransformTask:



Public Member Functions

- BTransformTask (string id, shared_ptr< BTransformComponent > transformComponent)

Additional Inherited Members

6.34.1 Constructor & Destructor Documentation

6.34.1.1 BTransformTask()

```
BTransformTask::BTransformTask (
            string id,
            shared_ptr< BTransformComponent > transformComponent )
```

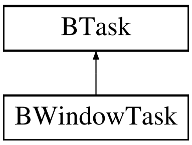
The documentation for this class was generated from the following files:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BTranformTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BTransformTask.cpp

6.35 BWindowTask Class Reference

`#include <BWindowTask.hpp>`

Inheritance diagram for BWindowTask:



Public Member Functions

- BWindowTask (const std::string &title, int _width, int _height, bool fullscreen=false)
- ∼BWindowTask ()
- void set_fullscreen (uint32_t type=0)
- void set_windowed ()
- unsigned get_width () const
- unsigned get_height () const
- void set_windowTitle (const char ∗title)
- void set_position (int new_left_x, int new_top_y)
- void set_size (int new_width, int new_height)
- void swap_buffers () const
- void clear () const
- virtual bool initialize ()
- virtual bool finalize ()
- virtual bool execute (float time)

Static Public Attributes

- static shared_ptr< BWindowTask > instance = nullptr

Additional Inherited Members

6.35.1 Constructor & Destructor Documentation

6.35.1.1 BWindowTask()

```
BWindowTask::BWindowTask (
             const std::string & title,
             int _width,
             int _height,
             bool fullscreen = false )
```

6.35.1.2 ∼BWindowTask()

```
BWindowTask::∼BWindowTask ( )
```

6.35.2 Member Function Documentation

6.35.2.1 clear()

```
void BWindowTask::clear ( ) const
```

6.35.2.2 execute()

```
bool BWindowTask::execute (
             float time ) [virtual]
```

Implements BTask.

6.35.2.3 finalize()

```
bool BWindowTask::finalize ( ) [virtual]
```

Implements BTask.

6.35.2.4 get_height()

```
unsigned BWindowTask::get_height ( ) const
```

6.35.2.5 get_width()

```
unsigned BWindowTask::get_width ( ) const
```

6.35.2.6 initialize()

```
bool BWindowTask::initialize ( ) [virtual]
```

Implements BTask.

6.35.2.7 set_fullscreen()

```
void BWindowTask::set_fullscreen (
             uint32_t type = 0 )
```

6.35.2.8 set_position()

```
void BWindowTask::set_position (
             int new_left_x,
             int new_top_y )
```

6.35.2.9 set_size()

```
void BWindowTask::set_size (
             int new_width,
             int new_height )
```

6.35.2.10 set_windowed()

```
void BWindowTask::set_windowed ( )
```

6.35.2.11 set_windowTitle()

```
void BWindowTask::set_windowTitle (
             const char * title )
```

6.35.2.12 swap_buffers()

```
void BWindowTask::swap_buffers ( ) const
```

6.35.3 Member Data Documentation

6.35.3.1 instance

```
shared_ptr< BWindowTask > BWindowTask::instance = nullptr  [static]
```

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BWindowTask.hpp
- D:/GitHub/BarxEngine/BarxEngine/code/source/BWindowTask.cpp

6.36 rapidxml::file< Ch > Class Template Reference

Represents data loaded from a file.

```
#include <rapidxml_utils.hpp>
```

Public Member Functions

- file (const char ∗filename)
- file (std::basic_istream< Ch > &stream)
- Ch ∗ data ()
- const Ch ∗ data () const
- std::size_t size () const

6.36.1 Detailed Description

template<class Ch = char>
class rapidxml::file< Ch >

Represents data loaded from a file.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 file() [1/2]

```
template<class Ch = char>
rapidxml::file< Ch >::file (
            const char ∗ filename )  [inline]
```

Loads file into the memory. Data will be automatically destroyed by the destructor.

Parameters

filename	Filename to load.

6.36.2.2 file() [2/2]

```
template<class Ch = char>
rapidxml::file< Ch >::file (
            std::basic_istream< Ch > & stream )  [inline]
```

Loads file into the memory. Data will be automatically destroyed by the destructor

Parameters

stream	Stream to load from

6.36.3 Member Function Documentation

6.36.3.1 data() [1/2]

```
template<class Ch = char>
Ch∗ rapidxml::file< Ch >::data ( )  [inline]
```

Gets file data.

Returns

Pointer to data of file.

6.36.3.2 data() [2/2]

```
template<class Ch = char>
const Ch* rapidxml::file< Ch >::data ( ) const  [inline]
```

Gets file data.

Returns

Pointer to data of file.

6.36.3.3 size()

```
template<class Ch = char>
std::size_t rapidxml::file< Ch >::size ( ) const  [inline]
```

Gets file data size.

Returns

Size of file data, in characters.

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml_utils.hpp

6.37 BKeyboard::KEYCODE Struct Reference

```
#include <BKeyboard.hpp>
```

Public Attributes

- const string A = "A"
- const string B = "B"
- const string C = "C"
- const string D = "D"
- const string E = "E"
- const string F = "F"
- const string G = "G"
- const string H = "H"
- const string I = "I"
- const string J = "J"
- const string K = "K"
- const string L = "L"
- const string M = "M"
- const string N = "N"
- const string O = "O"
- const string P = "P"
- const string Q = "Q"
- const string R = "R"
- const string S = "S"
- const string T = "T"
- const string U = "U"
- const string V = "V"
- const string W = "W"
- const string X = "X"

- const string Y = "Y"
- const string Z = "Z"
- const string N1 = "1"
- const string N2 = "2"
- const string N3 = "3"
- const string N4 = "4"
- const string N5 = "5"
- const string N6 = "6"
- const string N7 = "7"
- const string N8 = "8"
- const string N9 = "9"
- const string N0 = "0"

6.37.1 Member Data Documentation

6.37.1.1 A

```
const string BKeyboard::KEYCODE::A = "A"
```

6.37.1.2 B

```
const string BKeyboard::KEYCODE::B = "B"
```

6.37.1.3 C

```
const string BKeyboard::KEYCODE::C = "C"
```

6.37.1.4 D

```
const string BKeyboard::KEYCODE::D = "D"
```

6.37.1.5 E

```
const string BKeyboard::KEYCODE::E = "E"
```

6.37.1.6 F

```
const string BKeyboard::KEYCODE::F = "F"
```

6.37.1.7 G

```
const string BKeyboard::KEYCODE::G = "G"
```

6.37.1.8 H

```
const string BKeyboard::KEYCODE::H = "H"
```

6.37.1.9 I

```
const string BKeyboard::KEYCODE::I = "I"
```

6.37.1.10 J

```
const string BKeyboard::KEYCODE::J = "J"
```

6.37.1.11 K

```
const string BKeyboard::KEYCODE::K = "K"
```

6.37.1.12 L

```
const string BKeyboard::KEYCODE::L = "L"
```

6.37.1.13 M

```
const string BKeyboard::KEYCODE::M = "M"
```

6.37.1.14 N

```
const string BKeyboard::KEYCODE::N = "N"
```

6.37.1.15 N0

```
const string BKeyboard::KEYCODE::N0 = "0"
```

6.37.1.16 N1

```
const string BKeyboard::KEYCODE::N1 = "1"
```

6.37.1.17 N2

```
const string BKeyboard::KEYCODE::N2 = "2"
```

6.37.1.18 N3

```
const string BKeyboard::KEYCODE::N3 = "3"
```

6.37.1.19 N4

```
const string BKeyboard::KEYCODE::N4 = "4"
```

6.37.1.20 N5

```
const string BKeyboard::KEYCODE::N5 = "5"
```

6.37.1.21 N6

```
const string BKeyboard::KEYCODE::N6 = "6"
```

6.37.1.22 N7

```
const string BKeyboard::KEYCODE::N7 = "7"
```

6.37.1.23 N8

```
const string BKeyboard::KEYCODE::N8 = "8"
```

6.37.1.24 N9

```
const string BKeyboard::KEYCODE::N9 = "9"
```

6.37.1.25 O

```
const string BKeyboard::KEYCODE::O = "O"
```

6.37.1.26 P

```
const string BKeyboard::KEYCODE::P = "P"
```

6.37.1.27 Q

```
const string BKeyboard::KEYCODE::Q = "Q"
```

6.37.1.28 R

```
const string BKeyboard::KEYCODE::R = "R"
```

6.37.1.29 S

```
const string BKeyboard::KEYCODE::S = "S"
```

6.37.1.30 T

```
const string BKeyboard::KEYCODE::T = "T"
```

6.37.1.31 U

```
const string BKeyboard::KEYCODE::U = "U"
```

6.37.1.32 V

```
const string BKeyboard::KEYCODE::V = "V"
```

6.37.1.33 W

```
const string BKeyboard::KEYCODE::W = "W"
```

6.37.1.34 X

```
const string BKeyboard::KEYCODE::X = "X"
```

6.37.1.35 Y

```
const string BKeyboard::KEYCODE::Y = "Y"
```

6.37.1.36 Z

```
const string BKeyboard::KEYCODE::Z = "Z"
```

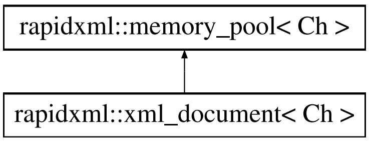
The documentation for this struct was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BKeyboard.hpp

6.38 rapidxml::memory_pool< Ch > Class Template Reference

`#include <rapidxml.hpp>`

Inheritance diagram for rapidxml::memory_pool< Ch >:



Public Member Functions

- memory_pool ()

  Constructs empty pool with default allocator functions.
- ∼memory_pool ()
- xml_node< Ch > ∗ allocate_node (node_type type, const Ch ∗name=0, const Ch ∗value=0, std::size_t name_size=0, std::size_t value_size=0)
- xml_attribute< Ch > ∗ allocate_attribute (const Ch ∗name=0, const Ch ∗value=0, std::size_t name_size=0, std::size_t value_size=0)
- Ch ∗ allocate_string (const Ch ∗source=0, std::size_t size=0)
- xml_node< Ch > ∗ clone_node (const xml_node< Ch > ∗source, xml_node< Ch > ∗result=0)
- void clear ()
- void set_allocator (alloc_func ∗af, free_func ∗ff)

6.38.1 Detailed Description

template<class Ch = char>
class rapidxml::memory_pool< Ch >

This class is used by the parser to create new nodes and attributes, without overheads of dynamic memory allocation. In most cases, you will not need to use this class directly. However, if you need to create nodes manually or modify names/values of nodes, you are encouraged to use memory_pool of relevant xml_document to allocate the memory. Not only is this faster than allocating them by using `new` operator, but also their lifetime will be tied to the lifetime of document, possibly simplyfing memory management.

Call allocate_node() or allocate_attribute() functions to obtain new nodes or attributes from the pool. You can also call allocate_string() function to allocate strings. Such strings can then be used as names or values of nodes without worrying about their lifetime. Note that there is no `free()` function – all allocations are freed at once when clear() function is called, or when the pool is destroyed.

It is also possible to create a standalone memory_pool, and use it to allocate nodes, whose lifetime will not be tied to any document.

Pool maintains `RAPIDXML_STATIC_POOL_SIZE` bytes of statically allocated memory. Until static memory is exhausted, no dynamic memory allocations are done. When static memory is exhausted, pool allocates additional blocks of memory of size `RAPIDXML_DYNAMIC_POOL_SIZE` each, by using global `new[]` and `delete[]` operators. This behaviour can be changed by setting custom allocation routines. Use set_allocator() function to set them.

Allocations for nodes, attributes and strings are aligned at `RAPIDXML_ALIGNMENT` bytes. This value defaults to the size of pointer on target architecture.

To obtain absolutely top performance from the parser, it is important that all nodes are allocated from a single, contiguous block of memory. Otherwise, cache misses when jumping between two (or more) disjoint blocks of memory can slow down parsing quite considerably. If required, you can tweak `RAPIDXML_STATIC_POOL_SIZE`, `RAPIDXML_DYNAMIC_POOL_SIZE` and `RAPIDXML_ALIGNMENT` to obtain best wasted memory to performance compromise. To do it, define their values before rapidxml.hpp file is included.

Parameters

Ch	Character type of created nodes.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 memory_pool()

```
template<class Ch = char>
rapidxml::memory_pool< Ch >::memory_pool ( )  [inline]
```

Constructs empty pool with default allocator functions.

6.38.2.2 ∼memory_pool()

```
template<class Ch = char>
rapidxml::memory_pool< Ch >::∼memory_pool ( )  [inline]
```

Destroys pool and frees all the memory. This causes memory occupied by nodes allocated by the pool to be freed. Nodes allocated from the pool are no longer valid.

6.38.3 Member Function Documentation

6.38.3.1 allocate_attribute()

```
template<class Ch = char>
xml_attribute<Ch>* rapidxml::memory_pool< Ch >::allocate_attribute (
             const Ch * name = 0,
             const Ch * value = 0,
             std::size_t name_size = 0,
             std::size_t value_size = 0 )  [inline]
```

Allocates a new attribute from the pool, and optionally assigns name and value to it. If the allocation request cannot be accomodated, this function will throw `std::bad_alloc`. If exceptions are disabled by defining RAPIDXML_NO_EXCEPTIONS, this function will call rapidxml::parse_error_handler() function.

Parameters

name	Name to assign to the attribute, or 0 to assign no name.
value	Value to assign to the attribute, or 0 to assign no value.
name_size	Size of name to assign, or 0 to automatically calculate size from name string.
value_size	Size of value to assign, or 0 to automatically calculate size from value string.

Returns

Pointer to allocated attribute. This pointer will never be NULL.

6.38.3.2 allocate_node()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::memory_pool< Ch >::allocate_node (
             node_type type,
             const Ch * name = 0,
             const Ch * value = 0,
             std::size_t name_size = 0,
             std::size_t value_size = 0 )  [inline]
```

Allocates a new node from the pool, and optionally assigns name and value to it. If the allocation request cannot be accomodated, this function will throw `std::bad_alloc`. If exceptions are disabled by defining RAPIDXML_NO_EXCEPTIONS, this function will call rapidxml::parse_error_handler() function.

Parameters

type	Type of node to create.
name	Name to assign to the node, or 0 to assign no name.
value	Value to assign to the node, or 0 to assign no value.
name_size	Size of name to assign, or 0 to automatically calculate size from name string.
value_size	Size of value to assign, or 0 to automatically calculate size from value string.

Returns

Pointer to allocated node. This pointer will never be NULL.

6.38.3.3 allocate_string()

```
template<class Ch = char>
Ch* rapidxml::memory_pool< Ch >::allocate_string (
             const Ch * source = 0,
             std::size_t size = 0 )  [inline]
```

Allocates a char array of given size from the pool, and optionally copies a given string to it. If the allocation request cannot be accomodated, this function will throw `std::bad_alloc`. If exceptions are disabled by defining RAPIDXML_NO_EXCEPTIONS, this function will call rapidxml::parse_error_handler() function.

Parameters

source	String to initialize the allocated memory with, or 0 to not initialize it.
size	Number of characters to allocate, or zero to calculate it automatically from source string length; if size is 0, source string must be specified and null terminated.

Returns

Pointer to allocated char array. This pointer will never be NULL.

6.38.3.4 clear()

```
template<class Ch = char>
void rapidxml::memory_pool< Ch >::clear ( )  [inline]
```

Clears the pool. This causes memory occupied by nodes allocated by the pool to be freed. Any nodes or strings allocated from the pool will no longer be valid.

6.38.3.5 clone_node()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::memory_pool< Ch >::clone_node (
             const xml_node< Ch > * source,
             xml_node< Ch > * result = 0 )  [inline]
```

Clones an xml_node and its hierarchy of child nodes and attributes. Nodes and attributes are allocated from this memory pool. Names and values are not cloned, they are shared between the clone and the source. Result node can be optionally specified as a second parameter, in which case its contents will be replaced with cloned source node. This is useful when you want to clone entire document.

Parameters

source	Node to clone.
result	Node to put results in, or 0 to automatically allocate result node

Returns

Pointer to cloned node. This pointer will never be NULL.

6.38.3.6 set_allocator()

```
template<class Ch = char>
void rapidxml::memory_pool< Ch >::set_allocator (
             alloc_func * af,
             free_func * ff )  [inline]
```

Sets or resets the user-defined memory allocation functions for the pool. This can only be called when no memory is allocated from the pool yet, otherwise results are undefined. Allocation function must not return invalid pointer on failure. It should either throw, stop the program, or use `longjmp()` function to pass control to other place of program. If it returns invalid pointer, results are undefined.

User defined allocation functions must have the following forms:

```
void *allocate(std::size_t size);
void free(void *pointer);
```

Parameters

af	Allocation function, or 0 to restore default function
ff	Free function, or 0 to restore default function

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

6.39 rapidxml::node_iterator< Ch > Class Template Reference

Iterator of child nodes of xml_node.

```
#include <rapidxml_iterators.hpp>
```

Public Types

- typedef xml_node< Ch > value_type
- typedef xml_node< Ch > & reference
- typedef xml_node< Ch > * pointer
- typedef std::ptrdiff_t difference_type
- typedef std::bidirectional_iterator_tag iterator_category

Public Member Functions

- node_iterator ()
- node_iterator (xml_node< Ch > *node)
- reference operator* () const
- pointer operator-> () const
- node_iterator & operator++ ()
- node_iterator operator++ (int)
- node_iterator & operator-- ()
- node_iterator operator-- (int)
- bool operator== (const node_iterator< Ch > &rhs)
- bool operator!= (const node_iterator< Ch > &rhs)

6.39.1 Detailed Description

template<class Ch>
class rapidxml::node_iterator< Ch >

Iterator of child nodes of xml_node.

6.39.2 Member Typedef Documentation

6.39.2.1 difference_type

```
template<class Ch>
typedef std::ptrdiff_t rapidxml::node_iterator< Ch >::difference_type
```

6.39.2.2 iterator_category

```
template<class Ch>
typedef std::bidirectional_iterator_tag rapidxml::node_iterator< Ch >::iterator_category
```

6.39.2.3 pointer

```
template<class Ch>
typedef xml_node<Ch>* rapidxml::node_iterator< Ch >::pointer
```

6.39.2.4 reference

```
template<class Ch>
typedef xml_node<Ch>& rapidxml::node_iterator< Ch >::reference
```

6.39.2.5 value_type

```
template<class Ch>
typedef xml_node<Ch> rapidxml::node_iterator< Ch >::value_type
```

6.39.3 Constructor & Destructor Documentation

6.39.3.1 node_iterator() [1/2]

```
template<class Ch>
rapidxml::node_iterator< Ch >::node_iterator ( )  [inline]
```

6.39.3.2 node_iterator() [2/2]

```
template<class Ch>
rapidxml::node_iterator< Ch >::node_iterator (
             xml_node< Ch > * node )  [inline]
```

6.39.4 Member Function Documentation

6.39.4.1 operator"!=()

```
template<class Ch>
bool rapidxml::node_iterator< Ch >::operator!= (
             const node_iterator< Ch > & rhs )  [inline]
```

6.39.4.2 operator*()

```
template<class Ch>
reference rapidxml::node_iterator< Ch >::operator* ( ) const  [inline]
```

6.39.4.3 operator++() [1/2]

```
template<class Ch>
node_iterator& rapidxml::node_iterator< Ch >::operator++ ( )  [inline]
```

6.39.4.4 operator++() [2/2]

```
template<class Ch>
node_iterator rapidxml::node_iterator< Ch >::operator++ (
             int  )  [inline]
```

6.39.4.5 operator--() [1/2]

```
template<class Ch>
node_iterator& rapidxml::node_iterator< Ch >::operator-- ( )  [inline]
```

6.39.4.6 operator--() [2/2]

```
template<class Ch>
node_iterator rapidxml::node_iterator< Ch >::operator-- (
             int  )  [inline]
```

6.39.4.7 operator->()

```
template<class Ch>
pointer rapidxml::node_iterator< Ch >::operator-> ( ) const  [inline]
```

6.39.4.8 operator==()

```
template<class Ch>
bool rapidxml::node_iterator< Ch >::operator== (
             const node_iterator< Ch > & rhs )  [inline]
```

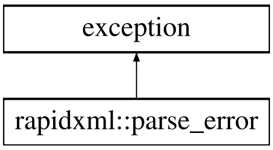
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml_iterators.hpp

6.40 rapidxml::parse_error Class Reference

```
#include <rapidxml.hpp>
```

Inheritance diagram for rapidxml::parse_error:



Public Member Functions

- parse_error (const char ∗what, void ∗where)

  Constructs parse error.
- virtual const char ∗ what () const throw ()
- template<class Ch >

  Ch ∗ where () const

6.40.1 Detailed Description

Parse error exception. This exception is thrown by the parser when an error occurs. Use what() function to get human-readable error message. Use where() function to get a pointer to position within source text where error was detected.

If throwing exceptions by the parser is undesirable, it can be disabled by defining RAPIDXML_NO_EXCEPTIONS macro before rapidxml.hpp is included. This will cause the parser to call rapidxml::parse_error_handler() function instead of throwing an exception. This function must be defined by the user.

This class derives from `std::exception` class.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 parse_error()

```
rapidxml::parse_error::parse_error (
             const char * what,
             void * where )  [inline]
```

Constructs parse error.

6.40.3 Member Function Documentation

6.40.3.1 what()

```
virtual const char* rapidxml::parse_error::what ( ) const throw ( )   [inline], [virtual]
```

Gets human readable description of error.

Returns

Pointer to null terminated description of the error.

6.40.3.2 where()

```
template<class Ch >
Ch* rapidxml::parse_error::where ( ) const  [inline]
```

Gets pointer to character data where error happened. Ch should be the same as char type of xml_document that produced the error.

Returns

Pointer to location within the parsed string where error occured.

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

6.41 vec3< T > Struct Template Reference

```
#include <BtypeDef.hpp>
```

Public Member Functions

- vec3 ()
- vec3 (T _x, T _y, T _z)
- void setValues (T _x, T _y, T _z)
- void normalize ()
- T inv_length ()

Public Attributes

- T x
- T y
- T z

6.41.1 Constructor & Destructor Documentation

6.41.1.1 vec3() [1/2]

```
template<class T>
vec3< T >::vec3 ( )  [inline]
```

6.41.1.2 vec3() [2/2]

```
template<class T>
vec3< T >::vec3 (
            T _x,
            T _y,
            T _z )  [inline]
```

6.41.2 Member Function Documentation

6.41.2.1 inv_length()

```
template<class T>
T vec3< T >::inv_length ( )  [inline]
```

6.41.2.2 normalize()

```
template<class T>
void vec3< T >::normalize ( )  [inline]
```

6.41.2.3 setValues()

```
template<class T>
void vec3< T >::setValues (
             T _x,
             T _y,
             T _z )  [inline]
```

6.41.3 Member Data Documentation

6.41.3.1 x

```
template<class T>
T vec3< T >::x
```

6.41.3.2 y

```
template<class T>
T vec3< T >::y
```

6.41.3.3 z

```
template<class T>
T vec3< T >::z
```

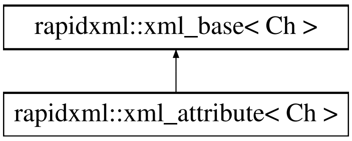
The documentation for this struct was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/BtypeDef.hpp

6.42 rapidxml::xml_attribute< Ch > Class Template Reference

```
#include <rapidxml.hpp>
```

Inheritance diagram for rapidxml::xml_attribute< Ch >:



Public Member Functions

- xml_attribute ()
- xml_document< Ch > * document () const
- xml_attribute< Ch > * previous_attribute (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_attribute< Ch > * next_attribute (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const

Friends

- class xml_node< Ch >

Additional Inherited Members

6.42.1 Detailed Description

template<class Ch = char>
class rapidxml::xml_attribute< Ch >

Class representing attribute node of XML document. Each attribute has name and value strings, which are available through name() and value() functions (inherited from xml_base). Note that after parse, both name and value of attribute will point to interior of source text used for parsing. Thus, this text must persist in memory for the lifetime of attribute.

Parameters

Ch	Character type to use.

6.42.2 Constructor & Destructor Documentation

6.42.2.1 xml_attribute()

```
template<class Ch = char>
rapidxml::xml_attribute< Ch >::xml_attribute ( )  [inline]
```

Constructs an empty attribute with the specified type. Consider using memory_pool of appropriate xml_document if allocating attributes manually.

6.42.3 Member Function Documentation

6.42.3.1 document()

```
template<class Ch = char>
xml_document<Ch>* rapidxml::xml_attribute< Ch >::document ( ) const  [inline]
```

Gets document of which attribute is a child.

Returns

Pointer to document that contains this attribute, or 0 if there is no parent document.

6.42.3.2 next_attribute()

```
template<class Ch = char>
xml_attribute<Ch>* rapidxml::xml_attribute< Ch >::next_attribute (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets next attribute, optionally matching attribute name.

Parameters

name	Name of attribute to find, or 0 to return next attribute regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found attribute, or 0 if not found.

6.42.3.3 previous_attribute()

```
template<class Ch = char>
xml_attribute<Ch>* rapidxml::xml_attribute< Ch >::previous_attribute (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets previous attribute, optionally matching attribute name.

Parameters

name	Name of attribute to find, or 0 to return previous attribute regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found attribute, or 0 if not found.

6.42.4 Friends And Related Function Documentation

6.42.4.1 xml_node< Ch >

```
template<class Ch = char>
friend class xml_node< Ch >  [friend]
```

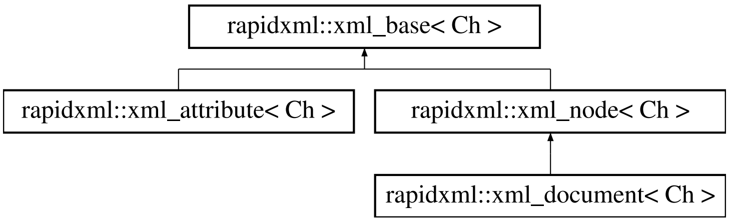
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

6.43 rapidxml::xml_base< Ch > Class Template Reference

```
#include <rapidxml.hpp>
```

Inheritance diagram for rapidxml::xml_base< Ch >:



Public Member Functions

- xml_base ()
- Ch ∗ name () const
- std::size_t name_size () const
- Ch ∗ value () const
- std::size_t value_size () const
- void name (const Ch ∗name, std::size_t size)
- void name (const Ch ∗name)
- void value (const Ch ∗value, std::size_t size)
- void value (const Ch ∗value)
- xml_node< Ch > ∗ parent () const

Static Protected Member Functions

- static Ch ∗ nullstr ()

Protected Attributes

- Ch ∗ m_name
- Ch ∗ m_value
- std::size_t m_name_size
- std::size_t m_value_size
- xml_node< Ch > ∗ m_parent

6.43.1 Detailed Description

template<class Ch = char>
class rapidxml::xml_base< Ch >

Base class for xml_node and xml_attribute implementing common functions: name(), name_size(), value(), value_size() and parent().

Parameters

Ch	Character type to use

6.43.2 Constructor & Destructor Documentation

6.43.2.1 xml_base()

```
template<class Ch = char>
rapidxml::xml_base< Ch >::xml_base ( )  [inline]
```

6.43.3 Member Function Documentation

6.43.3.1 name() [1/3]

```
template<class Ch = char>
Ch* rapidxml::xml_base< Ch >::name ( ) const  [inline]
```

Gets name of the node. Interpretation of name depends on type of node. Note that name will not be zero-terminated if rapidxml::parse_no_string_terminators option was selected during parse.

Use name_size() function to determine length of the name.

Returns

Name of node, or empty string if node has no name.

6.43.3.2 name() [2/3]

```
template<class Ch = char>
void rapidxml::xml_base< Ch >::name (
             const Ch * name )  [inline]
```

Sets name of node to a zero-terminated string. See also ownership_of_strings and xml_node::name(const Ch *, std::size_t).

Parameters

name	Name of node to set. Must be zero terminated.

6.43.3.3 name() [3/3]

```
template<class Ch = char>
void rapidxml::xml_base< Ch >::name (
             const Ch * name,
             std::size_t size )  [inline]
```

Sets name of node to a non zero-terminated string. See ownership_of_strings.

Note that node does not own its name or value, it only stores a pointer to it. It will not delete or otherwise free the pointer on destruction. It is reponsibility of the user to properly manage lifetime of the string. The easiest way to achieve it is to use memory_pool of the document to allocate the string - on destruction of the document the string will be automatically freed.

Size of name must be specified separately, because name does not have to be zero terminated. Use name(const Ch *) function to have the length automatically calculated (string must be zero terminated).

Parameters

name	Name of node to set. Does not have to be zero terminated.
size	Size of name, in characters. This does not include zero terminator, if one is present.

6.43.3.4 name_size()

```
template<class Ch = char>
std::size_t rapidxml::xml_base< Ch >::name_size ( ) const  [inline]
```

Gets size of node name, not including terminator character. This function works correctly irrespective of whether name is or is not zero terminated.

Returns

Size of node name, in characters.

6.43.3.5 nullstr()

```
template<class Ch = char>
static Ch* rapidxml::xml_base< Ch >::nullstr ( )  [inline], [static], [protected]
```

6.43.3.6 parent()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_base< Ch >::parent ( ) const  [inline]
```

Gets node parent.

Returns

Pointer to parent node, or 0 if there is no parent.

6.43.3.7 value() [1/3]

```
template<class Ch = char>
Ch* rapidxml::xml_base< Ch >::value ( ) const  [inline]
```

Gets value of node. Interpretation of value depends on type of node. Note that value will not be zero-terminated if rapidxml::parse_no_string_terminators option was selected during parse.

Use value_size() function to determine length of the value.

Returns

Value of node, or empty string if node has no value.

6.43.3.8 value() [2/3]

```
template<class Ch = char>
void rapidxml::xml_base< Ch >::value (
            const Ch * value )  [inline]
```

Sets value of node to a zero-terminated string. See also ownership_of_strings and xml_node::value(const Ch *, std::size_t).

Parameters

value	Vame of node to set. Must be zero terminated.

6.43.3.9 value() [3/3]

```
template<class Ch = char>
void rapidxml::xml_base< Ch >::value (
             const Ch * value,
             std::size_t size )  [inline]
```

Sets value of node to a non zero-terminated string. See ownership_of_strings.

Note that node does not own its name or value, it only stores a pointer to it. It will not delete or otherwise free the pointer on destruction. It is reponsibility of the user to properly manage lifetime of the string. The easiest way to achieve it is to use memory_pool of the document to allocate the string - on destruction of the document the string will be automatically freed.

Size of value must be specified separately, because it does not have to be zero terminated. Use value(const Ch *) function to have the length automatically calculated (string must be zero terminated).

If an element has a child node of type node_data, it will take precedence over element value when printing. If you want to manipulate data of elements using values, use parser flag rapidxml::parse_no_data_nodes to prevent creation of data nodes by the parser.

Parameters

value	value of node to set. Does not have to be zero terminated.
size	Size of value, in characters. This does not include zero terminator, if one is present.

6.43.3.10 value_size()

```
template<class Ch = char>
std::size_t rapidxml::xml_base< Ch >::value_size ( ) const  [inline]
```

Gets size of node value, not including terminator character. This function works correctly irrespective of whether value is or is not zero terminated.

Returns

Size of node value, in characters.

6.43.4 Member Data Documentation

6.43.4.1 m_name

```
template<class Ch = char>
Ch* rapidxml::xml_base< Ch >::m_name  [protected]
```

6.43.4.2 m_name_size

```
template<class Ch = char>
std::size_t rapidxml::xml_base< Ch >::m_name_size  [protected]
```

6.43.4.3 m_parent

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_base< Ch >::m_parent  [protected]
```

6.43.4.4 m_value

```
template<class Ch = char>
Ch* rapidxml::xml_base< Ch >::m_value  [protected]
```

6.43.4.5 m_value_size

```
template<class Ch = char>
std::size_t rapidxml::xml_base< Ch >::m_value_size  [protected]
```

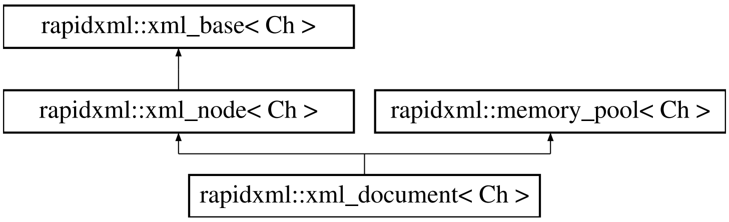
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

6.44 rapidxml::xml_document< Ch > Class Template Reference

```
#include <rapidxml.hpp>
```

Inheritance diagram for rapidxml::xml_document< Ch >:



Public Member Functions

- xml_document ()

  Constructs empty XML document.
- template<int Flags>
  void parse (Ch ∗text)
- void clear ()

Additional Inherited Members

6.44.1 Detailed Description

template<class Ch = char>
class rapidxml::xml_document< Ch >

This class represents root of the DOM hierarchy. It is also an xml_node and a memory_pool through public inheritance. Use parse() function to build a DOM tree from a zero-terminated XML text string. parse() function allocates memory for nodes and attributes by using functions of xml_document, which are inherited from memory_pool. To access root node of the document, use the document itself, as if it was an xml_node.

Parameters

Ch	Character type to use.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 xml_document()

```
template<class Ch = char>
rapidxml::xml_document< Ch >::xml_document ( )  [inline]
```

Constructs empty XML document.

6.44.3 Member Function Documentation

6.44.3.1 clear()

```
template<class Ch = char>
void rapidxml::xml_document< Ch >::clear ( )  [inline]
```

Clears the document by deleting all nodes and clearing the memory pool. All nodes owned by document pool are destroyed.

6.44.3.2 parse()

```
template<class Ch = char>
template<int Flags>
void rapidxml::xml_document< Ch >::parse (
             Ch * text )  [inline]
```

Parses zero-terminated XML string according to given flags. Passed string will be modified by the parser, unless rapidxml::parse_non_destructive flag is used. The string must persist for the lifetime of the document. In case of error, rapidxml::parse_error exception will be thrown.

If you want to parse contents of a file, you must first load the file into the memory, and pass pointer to its beginning. Make sure that data is zero-terminated.

Document can be parsed into multiple times. Each new call to parse removes previous nodes and attributes (if any), but does not clear memory pool.

Parameters

text	XML data to parse; pointer is non-const to denote fact that this data may be modified by the parser.

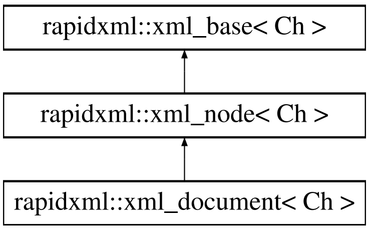
The documentation for this class was generated from the following file:

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

6.45 rapidxml::xml_node< Ch > Class Template Reference

```
#include <rapidxml.hpp>
```

Inheritance diagram for rapidxml::xml_node< Ch >:



Public Member Functions

- xml_node (node_type type)
- node_type type () const
- xml_document< Ch > * document () const
- xml_node< Ch > * first_node (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_node< Ch > * last_node (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_node< Ch > * previous_sibling (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_node< Ch > * next_sibling (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_attribute< Ch > * first_attribute (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- xml_attribute< Ch > * last_attribute (const Ch *name=0, std::size_t name_size=0, bool case_sensitive=true) const
- void type (node_type type)
- void prepend_node (xml_node< Ch > *child)
- void append_node (xml_node< Ch > *child)
- void insert_node (xml_node< Ch > *where, xml_node< Ch > *child)
- void remove_first_node ()
- void remove_last_node ()
- void remove_node (xml_node< Ch > *where)

  Removes specified child from the node.
- void remove_all_nodes ()

  Removes all child nodes (but not attributes).
- void prepend_attribute (xml_attribute< Ch > *attribute)
- void append_attribute (xml_attribute< Ch > *attribute)
- void insert_attribute (xml_attribute< Ch > *where, xml_attribute< Ch > *attribute)
- void remove_first_attribute ()
- void remove_last_attribute ()
- void remove_attribute (xml_attribute< Ch > *where)
- void remove_all_attributes ()

  Removes all attributes of node.

Additional Inherited Members

6.45.1 Detailed Description

template< class Ch = char >
class rapidxml::xml_node< Ch >

Class representing a node of XML document. Each node may have associated name and value strings, which are available through name() and value() functions. Interpretation of name and value depends on type of the node. Type of node can be determined by using type() function.

Note that after parse, both name and value of node, if any, will point interior of source text used for parsing. Thus, this text must persist in the memory for the lifetime of node.

Parameters

Ch	Character type to use.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 xml_node()

```
template<class Ch = char>
rapidxml::xml_node< Ch >::xml_node (
            node_type type )  [inline]
```

Constructs an empty node with the specified type. Consider using memory_pool of appropriate document to allocate nodes manually.

Parameters

type	Type of node to construct.

6.45.3 Member Function Documentation

6.45.3.1 append_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::append_attribute (
            xml_attribute< Ch > * attribute )  [inline]
```

Appends a new attribute to the node.

Parameters

attribute	Attribute to append.

6.45.3.2 append_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::append_node (
            xml_node< Ch > * child )  [inline]
```

Appends a new child node. The appended child becomes the last child.

Parameters

child	Node to append.

6.45.3.3 document()

```
template<class Ch = char>
xml_document<Ch>* rapidxml::xml_node< Ch >::document ( ) const  [inline]
```

Gets document of which node is a child.

Returns

Pointer to document that contains this node, or 0 if there is no parent document.

6.45.3.4 first_attribute()

```
template<class Ch = char>
xml_attribute<Ch>* rapidxml::xml_node< Ch >::first_attribute (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets first attribute of node, optionally matching attribute name.

Parameters

name	Name of attribute to find, or 0 to return first attribute regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found attribute, or 0 if not found.

6.45.3.5 first_node()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_node< Ch >::first_node (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets first child node, optionally matching node name.

Parameters

name	Name of child to find, or 0 to return first child regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found child, or 0 if not found.

6.45.3.6 insert_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::insert_attribute (
            xml_attribute< Ch > * where,
            xml_attribute< Ch > * attribute )  [inline]
```

Inserts a new attribute at specified place inside the node. All attributes after and including the specified attribute are moved one position back.

Parameters

where	Place where to insert the attribute, or 0 to insert at the back.
attribute	Attribute to insert.

6.45.3.7 insert_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::insert_node (
            xml_node< Ch > * where,
            xml_node< Ch > * child )  [inline]
```

Inserts a new child node at specified place inside the node. All children after and including the specified node are moved one position back.

Parameters

where	Place where to insert the child, or 0 to insert at the back.
child	Node to insert.

6.45.3.8 last_attribute()

```
template<class Ch = char>
xml_attribute<Ch>* rapidxml::xml_node< Ch >::last_attribute (
            const Ch * name = 0,
            std::size_t name_size = 0,
            bool case_sensitive = true ) const  [inline]
```

Gets last attribute of node, optionally matching attribute name.

Parameters

name	Name of attribute to find, or 0 to return last attribute regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found attribute, or 0 if not found.

6.45.3.9 last_node()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_node< Ch >::last_node (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets last child node, optionally matching node name. Behaviour is undefined if node has no children. Use first_node() to test if node has children.

Parameters

name	Name of child to find, or 0 to return last child regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found child, or 0 if not found.

6.45.3.10 next_sibling()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_node< Ch >::next_sibling (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets next sibling node, optionally matching node name. Behaviour is undefined if node has no parent. Use parent() to test if node has a parent.

Parameters

name	Name of sibling to find, or 0 to return next sibling regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found sibling, or 0 if not found.

6.45.3.11 prepend_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::prepend_attribute (
             xml_attribute< Ch > * attribute )  [inline]
```

Prepends a new attribute to the node.

Parameters

attribute	Attribute to prepend.

6.45.3.12 prepend_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::prepend_node (
             xml_node< Ch > * child )  [inline]
```

Prepends a new child node. The prepended child becomes the first child, and all existing children are moved one position back.

Parameters

child	Node to prepend.

6.45.3.13 previous_sibling()

```
template<class Ch = char>
xml_node<Ch>* rapidxml::xml_node< Ch >::previous_sibling (
             const Ch * name = 0,
             std::size_t name_size = 0,
             bool case_sensitive = true ) const  [inline]
```

Gets previous sibling node, optionally matching node name. Behaviour is undefined if node has no parent. Use parent() to test if node has a parent.

Parameters

name	Name of sibling to find, or 0 to return previous sibling regardless of its name; this string doesn't have to be zero-terminated if name_size is non-zero
name_size	Size of name, in characters, or 0 to have size calculated automatically from string
case_sensitive	Should name comparison be case-sensitive; non case-sensitive comparison works properly only for ASCII characters

Returns

Pointer to found sibling, or 0 if not found.

6.45.3.14 remove_all_attributes()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_all_attributes ( )  [inline]
```

Removes all attributes of node.

6.45.3.15 remove_all_nodes()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_all_nodes ( )  [inline]
```

Removes all child nodes (but not attributes).

6.45.3.16 remove_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_attribute (
            xml_attribute< Ch > * where )  [inline]
```

Removes specified attribute from node.

Parameters

where	Pointer to attribute to be removed.

6.45.3.17 remove_first_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_first_attribute ( )  [inline]
```

Removes first attribute of the node. If node has no attributes, behaviour is undefined. Use first_attribute() to test if node has attributes.

6.45.3.18 remove_first_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_first_node ( )  [inline]
```

Removes first child node. If node has no children, behaviour is undefined. Use first_node() to test if node has children.

6.45.3.19 remove_last_attribute()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_last_attribute ( )  [inline]
```

Removes last attribute of the node. If node has no attributes, behaviour is undefined. Use first_attribute() to test if node has attributes.

6.45.3.20 remove_last_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_last_node ( )  [inline]
```

Removes last child of the node. If node has no children, behaviour is undefined. Use first_node() to test if node has children.

6.45.3.21 remove_node()

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::remove_node (
            xml_node< Ch > * where )  [inline]
```

Removes specified child from the node.

6.45.3.22 type() [1/2]

```
template<class Ch = char>
node_type rapidxml::xml_node< Ch >::type ( ) const  [inline]
```

Gets type of node.

Returns

Type of node.

6.45.3.23 type() [2/2]

```
template<class Ch = char>
void rapidxml::xml_node< Ch >::type (
             node_type type )  [inline]
```

Sets type of node.

Parameters

type	Type of node to set.

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

- D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml.hpp

Chapter 7

File Documentation

7.1 D:/GitHub/BarxEngine/BarxEngine/code/headers/BAlgoritmos↩ DeOrdenacion.hpp File Reference

```
#include <iostream>
#include <vector>
#include <stdlib.h>
#include <time.h>
```

Namespaces

- BAlgoritmosDeOrdenacion

Functions

- template<class T >
  void BAlgoritmosDeOrdenacion::algoritmoBurbuja (T ∗list, size_t size)
- template<class T >
  void BAlgoritmosDeOrdenacion::algoritmoInserccionDirecta (T ∗list, size_t size)

  size_tercambiamos la posicion actual con el valor mas peque

  template<class T >
  void BAlgoritmosDeOrdenacion::quickSort (T ∗list, size_t size)

  Elegimos un numero al alazar (Mitad de la cadena)

  template<class T >
  void BAlgoritmosDeOrdenacion::countShort (T ∗list, size_t size)

7.2 D:/GitHub/BarxEngine/BarxEngine/code/headers/BAudio.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BAudio
- struct BAudio::BAudioInfo

Typedefs

- typedef struct _Mix_Music Mix_Music

7.2.1 Typedef Documentation

7.2.1.1 Mix_Music

```
typedef struct _Mix_Music Mix_Music
```

7.3 D:/GitHub/BarxEngine/BarxEngine/code/headers/BBox↩ ColliderComponent.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BBoxColliderComponent

7.4 D:/GitHub/BarxEngine/BarxEngine/code/headers/BCamera↩ Component.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BObserver.hpp"
```

Classes

- class BCameraComponent

7.5 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ CharacterController.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BObserver.hpp"
```

Classes

- class BCharacterControllerComponent

7.6 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ CharacterControllerTask.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTask.hpp"
```

Classes

- class BCharacterControllerTask

7.7 D:/GitHub/BarxEngine/BarxEngine/code/headers/BCollider↩ Component.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BComponent.hpp"
```

Classes

- class BColliderComponent

Enumerations

- enum COLLIDERTYPE { SPHERE = 0, BOX = 1 }

7.7.1 Enumeration Type Documentation

7.7.1.1 COLLIDERTYPE

```
enum COLLIDERTYPE
```

Enumerator

SPHERE	
BOX	

7.8 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ Component.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BComponent

7.9 D:/GitHub/BarxEngine/BarxEngine/code/headers/BControl↵ Component.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BControlComponent

7.10 D:/GitHub/BarxEngine/BarxEngine/code/headers/BControl↵ Task.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTask.hpp"
```

Classes

- class BControlTask

7.11 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↵ Dispacher.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BDispacher

7.12 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↵ Engine.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BScene.hpp"
#include "BEntity.hpp"
#include "BComponent.hpp"
#include "BControlComponent.hpp"
#include "BColliderComponent.hpp"
#include "BTransformComponent.hpp"
#include "BKeyboard.hpp"
#include "BAudio.hpp"
#include "BWindowTask.hpp"
```

Macros

- #define SDL_MAIN_HANDLED

Typedefs

- typedef BKeyboard Input

7.12.1 Macro Definition Documentation

7.12.1.1 SDL_MAIN_HANDLED

```
#define SDL_MAIN_HANDLED
```

7.12.2 Typedef Documentation

7.12.2.1 Input

```
typedef BKeyboard Input
```

7.13 D:/GitHub/BarxEngine/BarxEngine/code/headers/B←Entity.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BEntity

7.14 D:/GitHub/BarxEngine/BarxEngine/code/headers/BInput←Component.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BInputComponent

7.15 D:/GitHub/BarxEngine/BarxEngine/code/headers/BInput←HandlerTask.hpp File Reference

```
#include "BTask.hpp"
```

Classes

- class BMyInputHandlerTask

7.16 D:/GitHub/BarxEngine/BarxEngine/code/headers/BInput↩ Mapper.hpp File Reference

Classes

- class BInputMapper

7.17 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ Kernel.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTimer.hpp"
```

Classes

- class BKernel

7.18 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ Keyboard.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BKeyboard
- struct BKeyboard::KEYCODE

7.19 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ KeyboardComponent.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BObserver.hpp"
```

Classes

- class BKeyboardComponent

7.20 D:/GitHub/BarxEngine/BarxEngine/code/headers/BLight↵ Component.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BLightComponent

7.21 D:/GitHub/BarxEngine/BarxEngine/code/headers/BMain↵ Renderer.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BMainRenderer

7.22 D:/GitHub/BarxEngine/BarxEngine/code/headers/BMain↵ WindowComponent.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BMainWindowComponent

7.23 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↵ Message.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BMessage

7.24 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↵ Observer.hpp File Reference

```
#include "BMessage.hpp"
```

Classes

- class BOrbserver

7.25 D:/GitHub/BarxEngine/BarxEngine/code/headers/BRender↩ObjectComponent.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BRenderObjectComponent

7.26 D:/GitHub/BarxEngine/BarxEngine/code/headers/BRender↩ObjectTask.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BRenderObjectTask

7.27 D:/GitHub/BarxEngine/BarxEngine/code/headers/BRender↩Task.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTask.hpp"
```

Classes

- class BRenderTask

7.28 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩Scene.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BEntity.hpp"
#include "BDispacher.hpp"
```

Classes

- class BScene

7.29 D:/GitHub/BarxEngine/BarxEngine/code/headers/BShere↩ ColliderComponent.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BShereColliderComponent

7.30 D:/GitHub/BarxEngine/BarxEngine/code/headers/BSphere↩ Collider.cpp File Reference

7.31 D:/GitHub/BarxEngine/BarxEngine/code/headers/BSphere↩ ColliderTask.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTask.hpp"
```

Classes

- class BColliderTask

7.32 D:/GitHub/BarxEngine/BarxEngine/code/headers/BTask.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BTask

7.33 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ Timer.hpp File Reference

```
#include "BtypeDef.hpp"
#include <cstdint>
```

Classes

- class BTimer

7.34 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ TranformTask.hpp File Reference

```
#include "BtypeDef.hpp"
```

Classes

- class BTransformTask

7.35 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↩ TransformComponent.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BComponent.hpp"
```

Classes

- class BTransformComponent

7.36 D:/GitHub/BarxEngine/BarxEngine/code/headers/Btype↩ Def.hpp File Reference

```
#include <map>
#include <set>
#include <algorithm>
#include <memory>
#include <thread>
#include <functional>
#include <list>
#include <string>
#include <sstream>
#include <cstdlib>
#include <assert.h>
#include <vector>
#include <iterator>
#include <iostream>
```

Classes

- struct vec3< T >

Namespaces

- glt

Typedefs

- typedef int_fast16_t Id
- typedef short byte
- typedef struct SDL_Window SDL_Window
- typedef void ∗ SDL_GLContext

Enumerations

- enum TASKPRIORITY {
  WINDOW = 0, INPUTSYSTEM = 1, COLLISIONS = 2, TRANSFORM = 3,
  ENTITYUPDATES = 4, RENDEROBJECT = 5, RENDERGENERAL = 6 }

7.36.1 Typedef Documentation

7.36.1.1 byte

```
typedef short byte
```

7.36.1.2 Id

```
typedef int_fast16_t Id
```

7.36.1.3 SDL_GLContext

```
typedef void* SDL_GLContext
```

7.36.1.4 SDL_Window

```
typedef struct SDL_Window SDL_Window
```

7.36.2 Enumeration Type Documentation

7.36.2.1 TASKPRIORITY

```
enum TASKPRIORITY
```

Enumerator

WINDOW	
INPUTSYSTEM	
COLLISIONS	
TRANSFORM	
ENTITYUPDATES	
RENDEROBJECT	
RENDERGENERAL	

7.37 D:/GitHub/BarxEngine/BarxEngine/code/headers/B↵ WindowTask.hpp File Reference

```
#include "BtypeDef.hpp"
#include "BTask.hpp"
```

Classes

- class BWindowTask

7.38 D:/GitHub/BarxEngine/Barx↵ Engine/code/headers/rapidxml.hpp File Reference

```
#include <cstdlib>
#include <cassert>
#include <new>
#include <exception>
```

Classes

- class rapidxml::parse_error
- class rapidxml::xml_node< Ch >
- class rapidxml::xml_attribute< Ch >
- class rapidxml::xml_document< Ch >
- class rapidxml::memory_pool< Ch >
- class rapidxml::xml_base< Ch >
- class rapidxml::xml_attribute< Ch >
- class rapidxml::xml_node< Ch >
- class rapidxml::xml_document< Ch >

Namespaces

- rapidxml

Macros

- #define RAPIDXML_PARSE_ERROR(what, where) throw parse_error(what, where)
- #define RAPIDXML_STATIC_POOL_SIZE (64 ∗ 1024)
- #define RAPIDXML_DYNAMIC_POOL_SIZE (64 ∗ 1024)
- #define RAPIDXML_ALIGNMENT sizeof(void ∗)

Enumerations

- enum rapidxml::node_type {
  rapidxml::node_document, rapidxml::node_element, rapidxml::node_data, rapidxml::node_cdata,
  rapidxml::node_comment, rapidxml::node_declaration, rapidxml::node_doctype, rapidxml::node_pi
  }

Variables

- const int rapidxml::parse_no_data_nodes = 0x1
- const int rapidxml::parse_no_element_values = 0x2
- const int rapidxml::parse_no_string_terminators = 0x4
- const int rapidxml::parse_no_entity_translation = 0x8

- const int rapidxml::parse_no_utf8 = 0x10
- const int rapidxml::parse_declaration_node = 0x20
- const int rapidxml::parse_comment_nodes = 0x40
- const int rapidxml::parse_doctype_node = 0x80
- const int rapidxml::parse_pi_nodes = 0x100
- const int rapidxml::parse_validate_closing_tags = 0x200
- const int rapidxml::parse_trim_whitespace = 0x400
- const int rapidxml::parse_normalize_whitespace = 0x800
- const int rapidxml::parse_default = 0
- const int rapidxml::parse_non_destructive = parse_no_string_terminators | parse_no_entity_↩ translation
- const int rapidxml::parse_fastest = parse_non_destructive | parse_no_data_nodes
- const int rapidxml::parse_full = parse_declaration_node | parse_comment_nodes | parse_↩ doctype_node | parse_pi_nodes | parse_validate_closing_tags

7.38.1 Detailed Description

This file contains rapidxml parser and DOM implementation

7.38.2 Macro Definition Documentation

7.38.2.1 RAPIDXML_ALIGNMENT

```
#define RAPIDXML_ALIGNMENT sizeof(void *)
```

7.38.2.2 RAPIDXML_DYNAMIC_POOL_SIZE

```
#define RAPIDXML_DYNAMIC_POOL_SIZE (64 * 1024)
```

7.38.2.3 RAPIDXML_PARSE_ERROR

```
#define RAPIDXML_PARSE_ERROR(
            what,
            where ) throw parse_error(what, where)
```

7.38.2.4 RAPIDXML_STATIC_POOL_SIZE

```
#define RAPIDXML_STATIC_POOL_SIZE (64 * 1024)
```

7.39 D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml↩ _iterators.hpp File Reference

```
#include "rapidxml.hpp"
```

Classes

- class rapidxml::node_iterator< Ch >
  Iterator of child nodes of xml_node.
- class rapidxml::attribute_iterator< Ch >
  Iterator of child attributes of xml_node.

Namespaces

- rapidxml

7.39.1 Detailed Description

This file contains rapidxml iterators

7.40 D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml↩ _print.hpp File Reference

```
#include "rapidxml.hpp"
#include <ostream>
#include <iterator>
```

Namespaces

- rapidxml

Functions

- template<class OutIt , class Ch >
  OutIt rapidxml::print (OutIt out, const xml_node< Ch > &node, int flags=0)
- template<class Ch >
  std::basic_ostream< Ch > & rapidxml::print (std::basic_ostream< Ch > &out, const xml_node< Ch > &node, int flags=0)
- template<class Ch >
  std::basic_ostream< Ch > & rapidxml::operator<< (std::basic_ostream< Ch > &out, const xml↩ _node< Ch > &node)

Variables

- const int rapidxml::print_no_indenting = 0x1
  Printer flag instructing the printer to suppress indenting of XML. See print() function.

7.40.1 Detailed Description

This file contains rapidxml printer implementation

7.41 D:/GitHub/BarxEngine/BarxEngine/code/headers/rapidxml↩ _utils.hpp File Reference

```
#include "rapidxml.hpp"
#include <vector>
#include <string>
#include <fstream>
#include <stdexcept>
```

Classes

- class rapidxml::file< Ch >
  Represents data loaded from a file.

Namespaces

- rapidxml

Functions

- template<class Ch >
  std::size_t rapidxml::count_children (xml_node< Ch > ∗node)
- template<class Ch >
  std::size_t rapidxml::count_attributes (xml_node< Ch > ∗node)

7.41.1 Detailed Description

This file contains high-level rapidxml utilities that can be useful in certain simple scenarios. They should probably not be used if maximizing performance is the main objective.

7.42 D:/GitHub/BarxEngine/BarxEngine/code/source/BAudio.cpp File Reference

```
#include "..\headers\BAudio.hpp"
#include <SDL.h>
#include <SDL_mixer.h>
```

7.43 D:/GitHub/BarxEngine/BarxEngine/code/source/BBox↵ ColliderComponent.cpp File Reference

```
#include "..\headers\BtypeDef.hpp"
#include "..\headers\BColliderComponent.hpp"
#include "..\headers\BBoxColliderComponent.hpp"
#include "..\headers\BShereColliderComponent.hpp"
#include "../headers/BSphereColliderTask.hpp"
#include "../headers/BTransformComponent.hpp"
#include "../headers/BEntity.hpp"
```

7.44 D:/GitHub/BarxEngine/BarxEngine/code/source/BCamera↵ Component.cpp File Reference

```
#include "..\headers\BComponent.hpp"
#include "..\headers\BCameraComponent.hpp"
#include "..\headers\BRenderTask.hpp"
#include "..\headers\BEntity.hpp"
#include "..\headers\BTransformComponent.hpp"
#include <Camera.hpp>
#include <Render_Node.hpp>
```

7.45 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ CharacterController.cpp File Reference

```
#include "..\headers\BtypeDef.hpp"
#include "..\headers\BComponent.hpp"
#include "..\headers\BCharacterController.hpp"
#include "..\headers\BEntity.hpp"
#include "..\headers\BTask.hpp"
#include "..\headers\BObserver.hpp"
#include "..\headers\BScene.hpp"
#include "..\headers\BTransformComponent.hpp"
#include "..\headers\BCharacterControllerTask.hpp"
```

7.46 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ CharacterControllerTask.cpp File Reference

```
#include "..\headers\BTask.hpp"
#include "..\headers\BCharacterControllerTask.hpp"
#include "..\headers\BEntity.hpp"
#include "..\headers\BTransformComponent.hpp"
#include "../headers/BCharacterController.hpp"
#include "../headers/BKeyboardComponent.hpp"
#include "../headers/BScene.hpp"
#include "../headers/BKeyboard.hpp"
```

7.47 D:/GitHub/BarxEngine/BarxEngine/code/source/BCollider↩ Component.cpp File Reference

```
#include "..\headers\BColliderComponent.hpp"
#include "..\headers\BSphereColliderTask.hpp"
#include "..\headers\BEntity.hpp"
```

7.48 D:/GitHub/BarxEngine/BarxEngine/code/source/BCollider↩ Task.cpp File Reference

```
#include "..\headers\BSphereColliderTask.hpp"
#include "..\headers\BComponent.hpp"
#include "..\headers\BTransformComponent.hpp"
#include "..\headers\BScene.hpp"
#include "../headers/BEntity.hpp"
#include "../headers/BColliderComponent.hpp"
```

7.49 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ Component.cpp File Reference

```
#include "../headers/BTask.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BTranformTask.hpp"
#include "../headers/BEntity.hpp"
#include <Model.hpp>
#include <Cube.hpp>
#include <Node.hpp>
#include <Drawable.hpp>
#include <Render_Node.hpp>
#include <Model_Obj.hpp>
#include "../headers/BRenderObjectTask.hpp"
```

7.50 D:/GitHub/BarxEngine/BarxEngine/code/source/BControl↩ Component.cpp File Reference

```
#include "..//headers/BComponent.hpp"
#include "..//headers/BControlComponent.hpp"
#include "..//headers/BControlTask.hpp"
```

7.51 D:/GitHub/BarxEngine/BarxEngine/code/source/BControl↩ Task.cpp File Reference

```
#include "../headers/BScene.hpp"
#include "../headers/BTask.hpp"
#include "../headers/BControlTask.hpp"
#include "../headers/BWindowTask.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BRenderObjectTask.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include <Render_Node.hpp>
#include <SDL.h>
```

7.52 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ Dispacher.cpp File Reference

```
#include "..\headers\BDispacher.hpp"
#include "..//headers/BMessage.hpp"
#include "..//headers/BObserver.hpp"
```

7.53 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩Engine.cpp File Reference

```
#include "..//headers/BScene.hpp"
#include "..//headers/BEntity.hpp"
#include "..//headers/BComponent.hpp"
#include "..//headers/BControlComponent.hpp"
#include "..//headers/BTransformComponent.hpp"
```

7.54 D:/GitHub/BarxEngine/BarxEngine/code/source/BEntity.cpp File Reference

```
#include "..\headers\BEntity.hpp"
#include "../headers/BComponent.hpp"
#include "..//headers/BtypeDef.hpp"
#include "..//headers/BTransformComponent.hpp"
```

7.55 D:/GitHub/BarxEngine/BarxEngine/code/source/BInput↩Component.cpp File Reference

```
#include "..//headers/BComponent.hpp"
#include "..\headers\BInputComponent.hpp"
#include "..//headers/BEntity.hpp"
#include "../headers/BTask.hpp"
#include "../headers/BInputHandlerTask.hpp"
#include "../headers/BControlTask.hpp"
```

7.56 D:/GitHub/BarxEngine/BarxEngine/code/source/BInput↩HandlerTask.cpp File Reference

```
#include "..\headers\BInputHandlerTask.hpp"
#include "..\headers\BTask.hpp"
#include "..\headers\BKernel.hpp"
#include "..\headers\BMessage.hpp"
#include "..\headers\BScene.hpp"
#include <SDL.h>
```

7.57 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩Kernel.cpp File Reference

```
#include "..\headers\BKernel.hpp"
#include "..\headers\BTask.hpp"
```

```
#include "..\headers\BAlgoritmosDeOrdenacion.hpp"
```

7.58 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ Keyboard.cpp File Reference

```
#include "..\headers\BtypeDef.hpp"
#include "..\headers\BKeyboard.hpp"
#include <SDL.h>
```

7.59 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ KeyboardComponent.cpp File Reference

```
#include "..\headers\BComponent.hpp"
#include "..\headers\BKeyboardComponent.hpp"
#include "..\headers\BObserver.hpp"
#include "..\headers\BEntity.hpp"
#include "..\headers\BTask.hpp"
#include "..\headers\BScene.hpp"
#include "..\headers\BKeyboard.hpp"
```

7.60 D:/GitHub/BarxEngine/BarxEngine/code/source/BLight↩ Component.cpp File Reference

```
#include "..\headers\BtypeDef.hpp"
#include "..\headers\BComponent.hpp"
#include "..\headers\BLightComponent.hpp"
#include "..\headers\BRenderTask.hpp"
#include "..\headers\BEntity.hpp"
#include "..\headers\BTransformComponent.hpp"
#include <Camera.hpp>
#include <Render_Node.hpp>
#include <Light.hpp>
```

7.61 D:/GitHub/BarxEngine/BarxEngine/code/source/BMain↩ Renderer.cpp File Reference

```
#include "..//headers/BComponent.hpp"
#include "..//headers/BMainRenderer.hpp"
#include "..//headers/BRenderTask.hpp"
#include "..//headers/BWindowTask.hpp"
```

7.62 D:/GitHub/BarxEngine/BarxEngine/code/source/BMain↩ WindowComponent.cpp File Reference

```
#include "..//headers/BComponent.hpp"
#include "..//headers/BMainWindowComponent.hpp"
#include "..//headers/BWindowTask.hpp"
```

7.63 D:/GitHub/BarxEngine/BarxEngine/code/source/BRender↩ ObjectComponent.cpp File Reference

```
#include "../headers/BComponent.hpp"
#include "../headers/BRenderObjectComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BRenderObjectTask.hpp"
#include "../headers/BEntity.hpp"
#include <Model.hpp>
#include <Cube.hpp>
#include <Node.hpp>
#include <Drawable.hpp>
#include <Render_Node.hpp>
#include <Model_Obj.hpp>
```

7.64 D:/GitHub/BarxEngine/BarxEngine/code/source/BRender↩ ObjectTask.cpp File Reference

```
#include "../headers/BScene.hpp"
#include "../headers/BTask.hpp"
#include "../headers/BTranformTask.hpp"
#include "../headers/BWindowTask.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BRenderObjectTask.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include <Render_Node.hpp>
#include <SDL.h>
```

7.65 D:/GitHub/BarxEngine/BarxEngine/code/source/BRender↩ Task.cpp File Reference

```
#include "..//headers/BTask.hpp"
#include <Cube.hpp>
#include <Model.hpp>
#include <Light.hpp>
```

```
#include <Render_Node.hpp>
#include "..//headers/BtypeDef.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BWindowTask.hpp"
```

7.66 D:/GitHub/BarxEngine/BarxEngine/code/source/BScene.cpp File Reference

```
#include "..//headers/BScene.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include "../headers/BRenderObjectComponent.hpp"
#include "../headers/BColliderComponent.hpp"
#include "../headers/BControlComponent.hpp"
#include "../headers/BKernel.hpp"
#include "../headers/BEntity.hpp"
#include "../headers/BWindowTask.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BMainRenderer.hpp"
#include "../headers/BMainWindowComponent.hpp"
#include "../headers/BInputComponent.hpp"
#include "../headers/rapidxml.hpp"
#include "../headers/rapidxml_utils.hpp"
#include "../headers/BCharacterController.hpp"
#include "../headers/BKeyboardComponent.hpp"
#include "../headers/BKeyboard.hpp"
#include "../headers/BCameraComponent.hpp"
#include "../headers/BLightComponent.hpp"
#include "../headers/BShereColliderComponent.hpp"
#include "../headers/BBoxColliderComponent.hpp"
```

7.67 D:/GitHub/BarxEngine/BarxEngine/code/source/BShere↩ ColliderComponent.cpp File Reference

```
#include "..\headers\BColliderComponent.hpp"
#include "..\headers\BShereColliderComponent.hpp"
#include "../headers/BSphereColliderTask.hpp"
#include "../headers/BTransformComponent.hpp"
#include "../headers/BEntity.hpp"
#include "../headers/BBoxColliderComponent.hpp"
```

7.68 D:/GitHub/BarxEngine/BarxEngine/code/source/BTask.cpp File Reference

7.69 D:/GitHub/BarxEngine/BarxEngine/code/source/BTimer.cpp File Reference

```
#include "..\headers\BTimer.hpp"
#include <SDL.h>
```

7.70 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ TransformComponent.cpp File Reference

```
#include "../headers/BTask.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include "../headers/BEntity.hpp"
#include "../headers/BTranformTask.hpp"
```

7.71 D:/GitHub/BarxEngine/BarxEngine/code/source/B↩ TransformTask.cpp File Reference

```
#include "../headers/BScene.hpp"
#include "../headers/BTask.hpp"
#include "../headers/BTranformTask.hpp"
#include "../headers/BWindowTask.hpp"
#include "../headers/BRenderTask.hpp"
#include "../headers/BRenderObjectTask.hpp"
#include "../headers/BComponent.hpp"
#include "../headers/BTransformComponent.hpp"
#include <Render_Node.hpp>
#include <SDL.h>
```

7.72 D:/GitHub/BarxEngine/BarxEngine/code/source/BWindow↩ Task.cpp File Reference

```
#include <SDL.h>
#include "..//headers/BTask.hpp"
#include "../headers/BWindowTask.hpp"
#include "..//headers/BtypeDef.hpp"
#include <OpenGL.hpp>
```

Enumerations

- enum Fullscreen_Type { REAL = SDL_WINDOW_FULLSCREEN, DESKTOP = SDL_WINDOW↩ _FULLSCREEN_DESKTOP }

7.72.1 Enumeration Type Documentation

7.72.1.1 Fullscreen_Type

```
enum Fullscreen_Type
```

Enumerator

REAL	
DESKTOP	

Index