DocumentClassHierarchy

Generated by Doxygen 1.8.6

Wed Oct 26 2016 15:39:34

Contents

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 5

Namespace Documentation

5.1 docs Namespace Reference

This namespace is used to collect all the document hierarchy classes and types.

Classes

- class DocType

  The class containing the definition of Document Types names.
- class Document

  This is the base class of the Document hierarchy.
- class WebPage

  It describes a web page as a Document extensions.
- class Spreadsheet

  This describes a spreadsheet as a Document extension.

Enumerations

- enum Type { DOCUMENT, WEB_PAGE, SPREADSHEET }

  Instances of document identifier values.

Functions

- std::ostream & operator<< (std::ostream &strm, const Document &doc)
- bool operator== (const Document &c1, const Document &c2)
- bool operator!= (const Document &c1, const Document &c2)

Variables

- const std::string DEFAULT_DOCUMENT_TITLE = "Default Title"

  The title, if not specified on Document construction.
- const int DEFAULT_DOCUMENT_KWSIZE = 3

  The size, of key words if not specified on Document construction.
- const std::string DEFAULT_WEBPAGE_URL = "www.defualt.com"

  The url, if not specified on WebPage construction.
- const int DEFAULT_WEBPAGE_TEXTSIZE = 10

The size of contents, if not specified on WebPage construction.

- const int DEFAULT_SPREADSHEET_COLCNT = 7

  The number of columns, if not specified on Spreadsheet construction.
- const int DEFAULT_SPREADSHEET_ROWCNT = 2

  The number of rows, if not specified on Spreadsheet construction.
- const std::string TYPE_NAME_DOCUMENT = "\"Document\""

  The name of the Type.DOCUMENT type values.
- const std::string TYPE_NAME_WEBPAGE = "\"Web Page\""

  The name of the Type.WEB_PAGE type values.
- const std::string TYPE_NAME_SPREADSHEET = "\"Spreadsheet\""

  The name of the Type.SPREADSHEET type values.
- const std::string LOG_HEADER = " DocumenHierarchy: "

  the debugging string to be appended on log head by this Document
- const std::string LOG_HEADER_TEST = "\t[TEST] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Test logs
- const std::string LOG_HEADER_WARNING = "[WARNING] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Warning logs
- const std::string LOG_HEADER_INFO = "[INFO] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Info logs
- const std::string LOG_HEADER_ERROR = "[ERROR] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Error logs

5.1.1 Detailed Description

This namespace is used to collect all the document hierarchy classes and types.

5.1.2 Enumeration Type Documentation

5.1.2.1 enum docs::Type

Instances of document identifier values.

Enumerator

DOCUMENT only Document objects will have this Document::getType() value

WEB_PAGE only WebPage objects will have this Document::getType() value

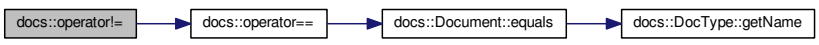
SPREADSHEET only Spreadsheet objects will have this Document::getType() value

5.1.3 Function Documentation

5.1.3.1 bool docs::operator!= (const Document & *c1*, const Document & *c2*)

it returns '!c1.equals(c2)'. This method should not be implemented on derived classes.

Here is the call graph for this function:



5.1.3.2 std::ostream& docs::operator<< (std::ostream & *strm,* const Document & *doc*)

it returns 'strm << doc.toString()'. This method should not be implemented on derived classes.

Here is the call graph for this function:

5.1.3.3 bool docs::operator== (const Document & *c1,* const Document & *c2*)

it returns 'c1.equals(c2)'. This method should not be implemented on derived classes.

Here is the call graph for this function:

5.2 parray Namespace Reference

it specify the dynamic array manager for a generic data T and a std::string specialisation.

Classes

- class PointerArray

  This class describes a manager of a dynamic array of generic type T∗.
- class StringPointerArray

  It implements a PointerArray with a string parameter (T).

Variables

- const std::string LOG_HEADER = " PointerArray: "

  the debugging string to be appended on log head by this list manager
- const std::string LOG_HEADER_TEST = "\t[TEST] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Test logs
- const std::string LOG_HEADER_WARNING = "[WARNING] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Warning logs
- const std::string LOG_HEADER_INFO = "[INFO] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Info logs
- const std::string LOG_HEADER_ERROR = "[ERROR] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Error logs

5.2.1 Detailed Description

it specify the dynamic array manager for a generic data T and a std::string specialisation.

5.3 patch Namespace Reference

to emulate C++11 and get to_string translator

Functions

- template<typename T >
  std::string to_string (const T &n)

  returns the description of the parameter by using << operator on a fresh std::ostringstream

5.3.1 Detailed Description

to emulate C++11 and get to_string translator

Chapter 6

Class Documentation

6.1 docs::DocType Class Reference

The class containing the definition of Document Types names.

`#include <Document.h>`

Public Member Functions

- DocType (Type t)

  constructors must provide a final type to be attached on document constructors.
- ~DocType ()

  empty since all instances of this class are suppose to be constant.
- const Type getType () const

  returns the final type ID.
- const std::string getName () const

  returns the name of the type of the document and "Invalid Type" if unknown.

Private Attributes

- const Type t

  the member type assign on constructor.

Friends

- std::ostream & operator<< (std::ostream &st, const DocType &t)

  prints the getName() result.
- bool operator== (const DocType &c1, const DocType &c2)

  return the comparison of getName() methods.
- bool operator!= (const DocType &c1, const DocType &c2)

  return the negation of the comparison of getName().

6.1.1 Detailed Description

The class containing the definition of Document Types names.

See Also

Type
TYPE_NAME_DOCUMENT
TYPE_NAME_WEBPAGE
TYPE_NAME_SPREADSHEET

It allows to print the name of the of type of document and at the same time make an easy comparison or perhaps a searching rule.

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

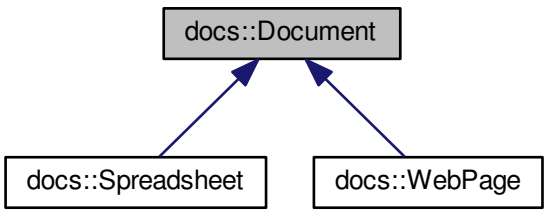
- src/Document.h

6.2 docs::Document Class Reference

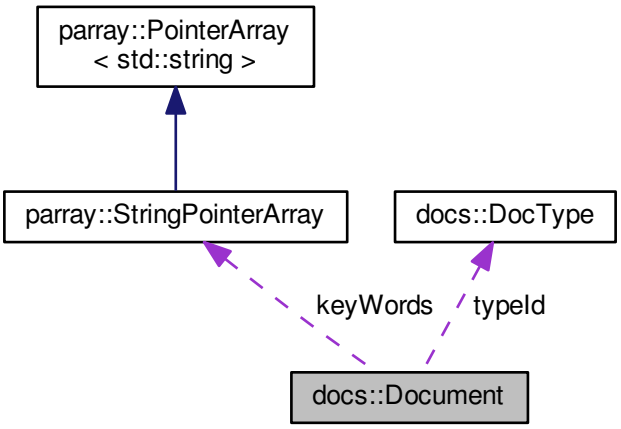
This is the base class of the Document hierarchy.

`#include <Document.h>`

Inheritance diagram for docs::Document:



Collaboration diagram for docs::Document:



Public Member Functions

- Document (std::string title=DEFAULT_DOCUMENT_TITLE, int keyWordSize=DEFAULT_DOCUMENT_KW-SIZE)

  construct by creating a new keyword list.
- Document (const parray::StringPointerArray &kws, std::string title=DEFAULT_DOCUMENT_TITLE)

  construct by using a copy of a PointerArray of string.
- Document (const Document &original)

  copy() constructor, to create an independent clone.
- virtual ~Document ()

  destructor, to clear the keywords list.
- Document & operator= (const Document &)

  to copy() this object
- virtual const std::string toString () const

  print the information of this object by listing all the data members.
- virtual bool const equals (const Document &) const

  check if all the data member are equals (type, title, keyWords).
- virtual std::string getTitle () const

  return the title of this document
- virtual void setTitle (const std::string)

  set a title to this document.
- virtual parray::StringPointerArray * getKeyWords () const

  returns the manager of the keyWords array of this document.
- virtual void setKeyWords (parray::StringPointerArray *)

  set a copy of the input keyWords and overwrites the previous.
- virtual const std::string * getKeyWordsArray () const

  returns the dynamic array containing the keyWords
- virtual const int getKeyWordsSize () const

  returns the getKeyWordsArray() size

- const DocType getType () const

  return the type identifier of this document assigned on custruction.

Protected Member Functions

- virtual void copy (const Document &)

  copy this object on a new memory location.
- Document (const DocType typeId, const std::string title=DEFAULT_DOCUMENT_TITLE, const int keyWordSize=DEFAULT_DOCUMENT_KWSIZE)

  The Document(std::string, int) constructor to be use on derived class for specifying the document typeId ID.
- Document (const DocType typeId, const parray::StringPointerArray &kws, const std::string title=DEFAULT_DOCUMENT_TITLE)

  The Document(const parray::StringPointerArray&, std::string) constructor to be use on derived class for specifying the document typeId ID.

Protected Attributes

- const DocType typeId

  the type ID assigned to this document on constructor.

Private Attributes

- std::string title

  the title of this Document.
- parray::StringPointerArray * keyWords

  the key word array manager of this Document.

Friends

- std::ostream & operator<< (std::ostream &, const Document &)

  return the result of toString().
- bool operator== (const Document &, const Document &)

  returns the result of equals().
- bool operator!= (const Document &, const Document &)

  returns the negation of equals()

6.2.1 Detailed Description

This is the base class of the Document hierarchy.

See Also

DocType
parray::StringPointerArray

It describes a generic document through: a Type, a title (string) and a keyword set (PointerArray). It implements default empty and copy constructors as well as basic operator overloading.

6.2.2 Friends And Related Function Documentation

6.2.2.1 bool operator!= (const Document & *c1*, const Document & *c2*) [friend]

returns the negation of equals()

it returns '!c1.equals(c2)'. This method should not be implemented on derived classes.

6.2.2.2 std::ostream& operator<< (std::ostream & *strm*, const Document & *doc*) [friend]

return the result of toString().

it returns 'strm << doc.toString()'. This method should not be implemented on derived classes.

6.2.2.3 bool operator== (const Document & *c1*, const Document & *c2*) [friend]

returns the result of equals().

it returns 'c1.equals(c2)'. This method should not be implemented on derived classes.

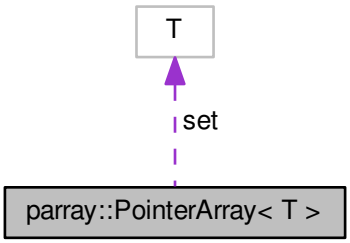
The documentation for this class was generated from the following files:

- src/Document.h
- src/Document.cpp

6.3 parray::PointerArray< T > Class Template Reference

This class describes a manager of a dynamic array of generic type T*.

Collaboration diagram for parray::PointerArray< T >:



Public Member Functions

- PointerArray ()
  does not initialise the set and invalidates the counter (generates also warning log to advertise the call of setSize())
- PointerArray (const int length)
  create a new empty array of specified size
- PointerArray (const PointerArray< T > &original)
  copy() constructors
- virtual ~PointerArray ()

destructors, invalidate counter, pointer and clear memory. Logs: "deleting", for debugging purposes.

- void setSize (const int length)

  set the size of a new empty array on memory and delete the previous.
- int getSize () const

  returns the actual size of the array.
- int getCnt () const

  returns the actual number of elements in the array.
- const T * getArray () const

  returns the head to the memory managed by this class.
- virtual const T & get (int idx) const

  returns an element of the array by index.
- const T & operator[] (size_t idx) const
- virtual bool add (const T toAdd)

  add an element into the array. Returns false if the array is full
- virtual const int find (const T toFind) const

  returns the index of an occurrence in the array. -1 if not found
- virtual void clear ()

  delete the array and create a new memory location of the same size.
- virtual const bool remove (T toRemove)

  remove an element from the array (based on find() and remove(int)).
- virtual const bool remove (int idx)

  remove an element from the array by index.
- virtual void resize (const int newLenght)

  replace the array with a new memory of the defined size. Possible old values are copied from start up to fill the array.
- virtual void pack ()

  replace the array by removing empty cell on tail (size will be equal to cnt). Based on resize().
- virtual const std::string toString () const

  returns a description of the memory managed by this class
- virtual bool equals (const PointerArray< T > &toCompare) const

  returns true if all the elements of this array are the only elements of the parameter.
- PointerArray< T > & operator= (const PointerArray< T > &newCopy)

  make a copy() of this array.
- virtual const bool isEmpty () const

  returns true if there are no elements in the array.
- bool const operator! () const

  returns isEmpty().

Private Member Functions

- void init (int length)

  initialises a new array of given lenght (based on newSet()).
- void newSet ()

  allocate a new memory of this object size. Delete old memory if is not invalidated.
- virtual void copy (const PointerArray< T > &original)

  make a new copy in memory (based on newSet()).

Private Attributes

- int size

  the size of the dynamic array

- int cnt

  the number of elements in the array

- T * set

  the head of the array

Friends

- std::ostream & operator<< (std::ostream &strm, const PointerArray< T > &par)

  calls toString() to describe the object.

- bool operator== (const PointerArray< T > &c1, const PointerArray< T > &c2)

  calls equals()

- bool operator!= (const PointerArray< T > &c1, const PointerArray< T > &c2)

  returns the negation of equals()

6.3.1 Detailed Description

template<class T>class parray::PointerArray< T >

This class describes a manager of a dynamic array of generic type T*.

It manage manipulation, creation, copy and destruction, as well as basic operator overloading. In details, the array is specified by: a pointer to the head, a size integer and a cnt meaning the number of elements currently in the array.

6.3.2 Member Function Documentation

6.3.2.1 template<class T> const T& parray::PointerArray< T >::operator[] (size_t *idx*) const [inline]

returns get().

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

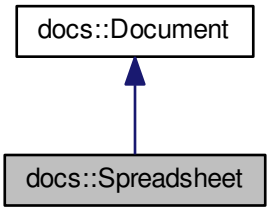
- src/PointerArray.cpp

6.4 docs::Spreadsheet Class Reference

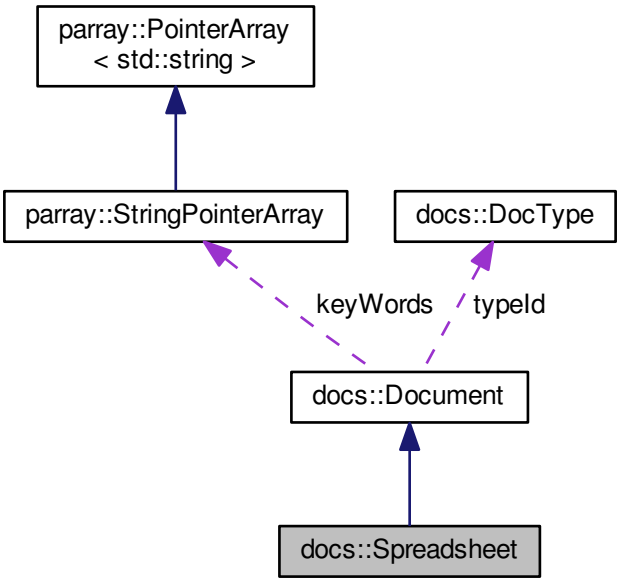
This describes a spreadsheet as a Document extension.

```
#include <Document.h>
```

Inheritance diagram for docs::Spreadsheet:



Collaboration diagram for docs::Spreadsheet:



Public Member Functions

- Spreadsheet (int coloumnCnt=DEFAULT_SPREADSHEET_COLCNT, int rowCnt=DEFAULT_SPREADSHEET_ROWCNT, std::string title=DEFAULT_DOCUMENT_TITLE, int keyWordSize=DEFAULT_DOCUMENT_KWSIZE)

  Construct by creating a new Spreadsheet. It is based on Document(const DocType typeId, std::string, int)
- Spreadsheet (const parray::StringPointerArray &kws, std::string title=DEFAULT_DOCUMENT_TITLE, int coloumnCnt=DEFAULT_SPREADSHEET_COLCNT, int rowCnt=DEFAULT_SPREADSHEET_ROWCNT)

  Construct by creating a new Spreadsheet. It is based on Document(const DocType typeId, const parray::StringPointerArray&, std::string title);.
- Spreadsheet (const Spreadsheet &original)

to copy() this object

- virtual ~Spreadsheet ()

  empty destructor,it relies on class hierarchy.

- virtual const std::string toString () const

  append to the result of Document::toString() the description of the number of columns and rows.

- virtual const bool equals (const Spreadsheet &) const

  return true if Document::equals() is true as well as the number of columns and rows are equals.

- const int getRowCnt () const

  get the number of rows (rowCnt) of this Spreadsheet.

- void setRowCnt (const int)

  set the number of rows (rowCnt) of this Spreadsheet.

- int getColoumnCnt () const

  get the number of columns (coloumnCnt) of this Spreadsheet.

- void setColoumnCnt (const int)

  set the number of columns (coloumnCnt) of this Spreadsheet.

Protected Member Functions

- virtual void copy (const Spreadsheet &)

  copy this object on a new memory location. It relies on Document::copy(), which is also used on copy constructor and assign operator.

Private Attributes

- int rowCnt

  the number of rows assigned to this Spreadsheet.

- int coloumnCnt

  the number of columns assigned to this Spreadsheet.

Additional Inherited Members

6.4.1 Detailed Description

This describes a spreadsheet as a Document extension.

See Also

Document
DocType
parray::StringPointerArray

It is a Document, which has also a specified number of columns and rows.

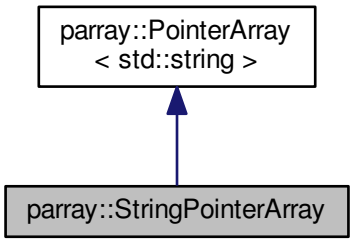
The documentation for this class was generated from the following files:

- src/Document.h
- src/Spreadsheet.cpp

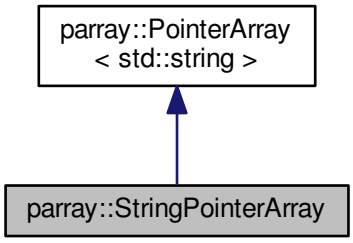
6.5 parray::StringPointerArray Class Reference

It implements a PointerArray with a string parameter (T).

Inheritance diagram for parray::StringPointerArray:



Collaboration diagram for parray::StringPointerArray:



Public Member Functions

- StringPointerArray ()

  empty constructor, based on PointerArray::PointerArray().
- StringPointerArray (const int length)

  constructor a new array of given length, based on PointerArray::PointerArray(int).
- StringPointerArray (const StringPointerArray &original)

  PointerArray::copy() constructor, based on PointerArray::PointerArray(PointerArray).
- virtual ~StringPointerArray ()

  empty destructor, it relies on the base implementation.
- virtual bool add (std::string toAdd)

  overload the PointerArray::add() method by call resize if the array is full (generate warning).

Static Public Member Functions

- static void tester ()

  static method to test the behavior of a PointerArray of strings.

Static Private Member Functions

- static void printTestDelitator ()

  prints a line to separe different tests used in tester() function

6.5.1 Detailed Description

It implements a PointerArray with a string parameter (T).

See Also

PointerArray

6.5.2 Member Function Documentation

6.5.2.1 static void parray::StringPointerArray::tester ( ) [inline], [static]

static method to test the behavior of a PointerArray of strings.

Creates (pa0) new StringPointerArray(const int) and populate with letters.

StringPointerArray::remove(T) an element from pa0 and access the first char (StringPointerArray::operator[]()) after the manipulation.

Creates (pa1) a new StringPointerArray() and set is size to be smaller than pa0.

Populate (StringPointerArray::add()) pa1 with some elements.

Make a comparison (StringPointerArray::operator==()) between two StringPointerArray (pa0 == pa1)

Creates (pa2) a copy (StringPointerArray::copy() and StringPointerArray(StringPointerArray)) of pa1.

StringPointerArray::remove() all its elements and makes a comparison.

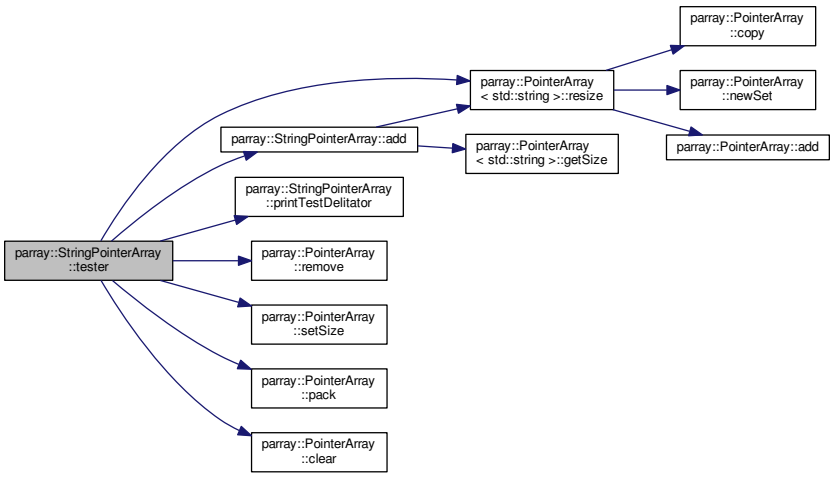
Adds two strings to pa2.

Creates (pa3) a copy() of pa2 and remove the first element.

Makes a comparison (pa3 == pa2) and pack() pa2.

resize() pa1 to be bigger and smaller.

Here is the call graph for this function:



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

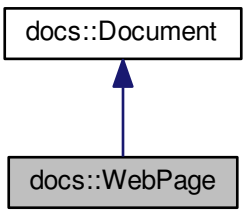
- src/PointerArray.cpp

6.6 docs::WebPage Class Reference

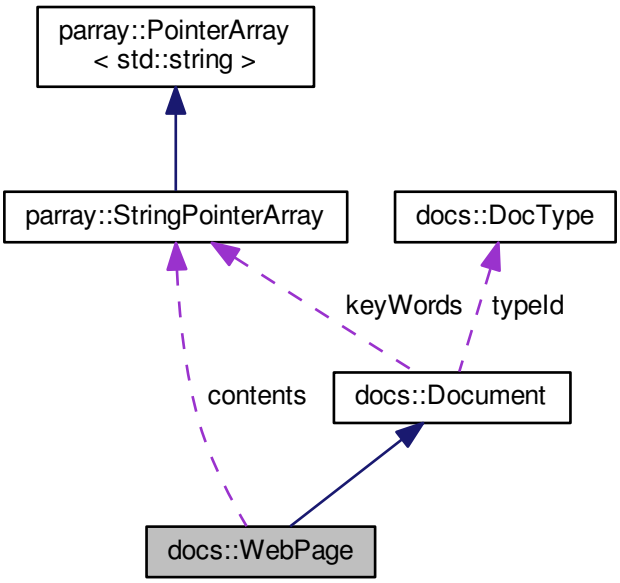
It describes a web page as a Document extensions.

`#include <Document.h>`

Inheritance diagram for docs::WebPage:



Collaboration diagram for docs::WebPage:



Public Member Functions

- WebPage (std::string title=DEFAULT_DOCUMENT_TITLE, std::string url=DEFAULT_WEBPAGE_URL, int keyWordSize=DEFAULT_DOCUMENT_KWSIZE, int textSize=DEFAULT_WEBPAGE_TEXTSIZE)

  construct by creating a new contents list. It is based on Document(const DocType typeId, std::string, int)
- WebPage (const parray::StringPointerArray &kws, const parray::StringPointerArray &tex=0, std::string title=DEFAULT_DOCUMENT_TITLE, std::string url=DEFAULT_WEBPAGE_URL)

  construct by using a copy of a PointerArray of contents. It is based on Document(const DocType typeId, const parray::StringPointerArray&, std::string title);
- WebPage (const WebPage &original)

  to copy() this object
- virtual ~WebPage ()

  destructor, to clear the contents list.
- virtual const std::string toString () const

  append to the result of Document::toString() the description of the url and contents.
- virtual const bool equals (const WebPage &) const

  return true if Document::equals() is true as well as urls and contends are equals.
- virtual const std::string getUrl () const

  returns the url assign to this web page.
- virtual void setUrl (const std::string)

  set the url for this web page
- virtual parray::StringPointerArray * getContents () const

  returns the manager of the contents array of this document.
- virtual void setContents (parray::StringPointerArray *)

  set a copy of the input contents and overwrites the previous.

- virtual const std::string * getContentsArray () const

  returns the dynamic array containing the contents.
- virtual const int getContentsSize () const

  returns the getContentsArray() size

Protected Member Functions

- virtual void copy (const WebPage &)

  copy this object on a new memory location. It relies on Document::copy(), which is also used on copy constructor and assign operator.

Private Attributes

- std::string url

  The url of this WebPage.
- parray::StringPointerArray * contents

  The contents of this WebPage.

Additional Inherited Members

6.6.1 Detailed Description

It describes a web page as a Document extensions.

See Also

Document
DocType
parray::StringPointerArray

This is a Document which have also an url and an array of contents (e.g., textual line).

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

- src/Document.h
- src/WebPage.cpp

Chapter 7

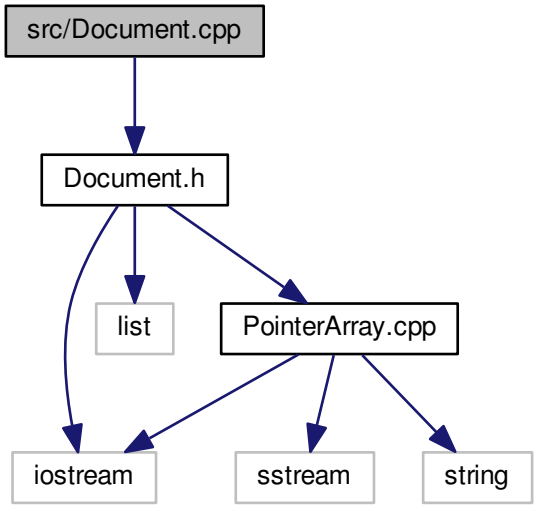
File Documentation

7.1 src/Document.cpp File Reference

implementation of the Document base class.

```
#include "Document.h"
```

Include dependency graph for Document.cpp:



Namespaces

- docs

  This namespace is used to collect all the document hierarchy classes and types.

Functions

- std::ostream & docs::operator<< (std::ostream &strm, const Document &doc)
- bool docs::operator== (const Document &c1, const Document &c2)

- bool docs::operator!= (const Document &c1, const Document &c2)

7.1.1 Detailed Description

implementation of the Document base class.

Author

Buoncompagni Luca

Date

Sep 14, 2016

See Also

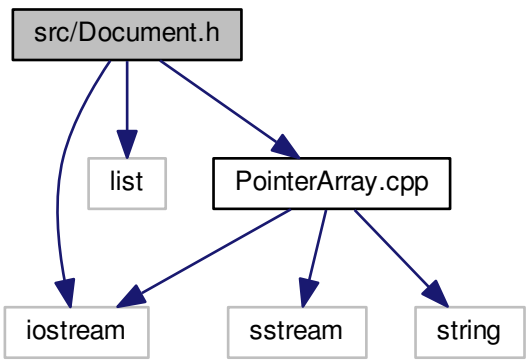
Document
docs

7.2 src/Document.h File Reference

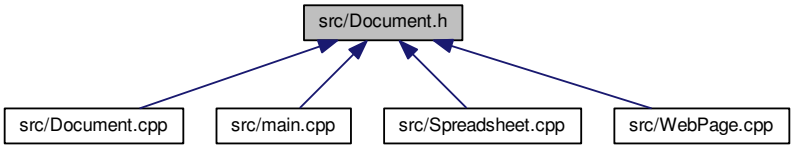
The complete Document Hierarchy namespce specification.

```
#include <iostream>
#include <list>
#include "PointerArray.cpp"
```

Include dependency graph for Document.h:



This graph shows which files directly or indirectly include this file:



Classes

- class docs::DocType

  The class containing the definition of Document Types names.
- class docs::Document

  This is the base class of the Document hierarchy.
- class docs::WebPage

  It describes a web page as a Document extensions.
- class docs::Spreadsheet

  This describes a spreadsheet as a Document extension.

Namespaces

- docs

  This namespace is used to collect all the document hierarchy classes and types.

Enumerations

- enum docs::Type { docs::DOCUMENT, docs::WEB_PAGE, docs::SPREADSHEET }

  Instances of document identifier values.

Variables

- const std::string docs::DEFAULT_DOCUMENT_TITLE = "Default Title"

  The title, if not specified on Document construction.
- const int docs::DEFAULT_DOCUMENT_KWSIZE = 3

  The size, of key words if not specified on Document construction.
- const std::string docs::DEFAULT_WEBPAGE_URL = "www.defualt.com"

  The url, if not specified on WebPage construction.
- const int docs::DEFAULT_WEBPAGE_TEXTSIZE = 10

  The size of contents, if not specified on WebPage construction.
- const int docs::DEFAULT_SPREADSHEET_COLCNT = 7

  The number of columns, if not specified on Spreadsheet construction.
- const int docs::DEFAULT_SPREADSHEET_ROWCNT = 2

  The number of rows, if not specified on Spreadsheet construction.
- const std::string docs::TYPE_NAME_DOCUMENT = "\"Document\""

  The name of the Type.DOCUMENT type values.
- const std::string docs::TYPE_NAME_WEBPAGE = "\"Web Page\""

The name of the Type.WEB_PAGE type values.

- const std::string docs::TYPE_NAME_SPREADSHEET = "\"Spreadsheet\""

  The name of the Type.SPREADSHEET type values.

- const std::string docs::LOG_HEADER = " DocumenHierarchy: "

  the debugging string to be appended on log head by this Document

- const std::string docs::LOG_HEADER_TEST = "\t[TEST] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Test logs

- const std::string docs::LOG_HEADER_WARNING = "[WARNING] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Warning logs

- const std::string docs::LOG_HEADER_INFO = "[INFO] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Info logs

- const std::string docs::LOG_HEADER_ERROR = "[ERROR] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Error logs

7.2.1 Detailed Description

The complete Document Hierarchy namespce specification.

Author

Buoncompagni Luca

Date

Sep 14, 2016

This is a test implementation of a Document class, which contains an array of key words and a title and two derived classes. A WebPage representation, with an url and an array of lines contents (

See Also

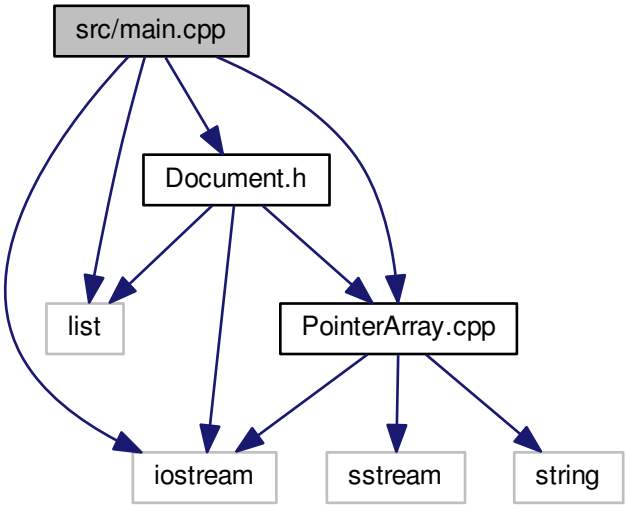
PointerArray), as well as a Spreadsheet, with related number of column and raw.
PointerArray

7.3 src/main.cpp File Reference

implements a main function to check the behavior of the parray template and docs hierarchy.

```
#include <iostream>
#include <list>
#include "Document.h"
#include "PointerArray.cpp"
```

Include dependency graph for main.cpp:



Functions

- void logTest (std::string msg)

  function used to highlights debugging logs by printing a frame of '#'.
- int main ()

  the main function to check project consistency by console logging.

Variables

- const std::string LOG_HEADER = " Main: "

  the debugging string to be appended on log head by this manager
- const std::string LOG_HEADER_TEST = "\t[TEST] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester()

7.3.1 Detailed Description

implements a main function to check the behavior of the parray template and docs hierarchy.

Author

Buoncompagni Luca

Date

Sep 17, 2016

See Also

StringPointerArray
plist
Document
SpreadSheet
WebPage
docs

7.3.2 Function Documentation

7.3.2.1 int main ( )

the main function to check project consistency by console logging.

check the behavior of the template implementation (parray) through the parray::StringPointerArray::tester() function.

check the behavior of the Document hierarchy (docs).

define a document with given title and keywords.

define another document with given title and keywords.

make a copy of the second Document, augment title and change keywords.

create a copy of the third Document. Overwrite the title and remove keywords.

create a Document with a given set of keywords.

create a WebPage. Add keywords and contents.

make a new Web Page by specifying only the title and url.

make a new Web Page by specifying the title, the url, the contents and keywords.

make a copy of the second WebPage. Augment the title, remove a keyword and add a contents.

Make a copy of the third WebPage. Set a new title and clear keywords.

create a new WebPage with a given keywords set.

create a new WebPage with a given keywords and contents sets. Set also its title.

create a new default Spreadsheet.

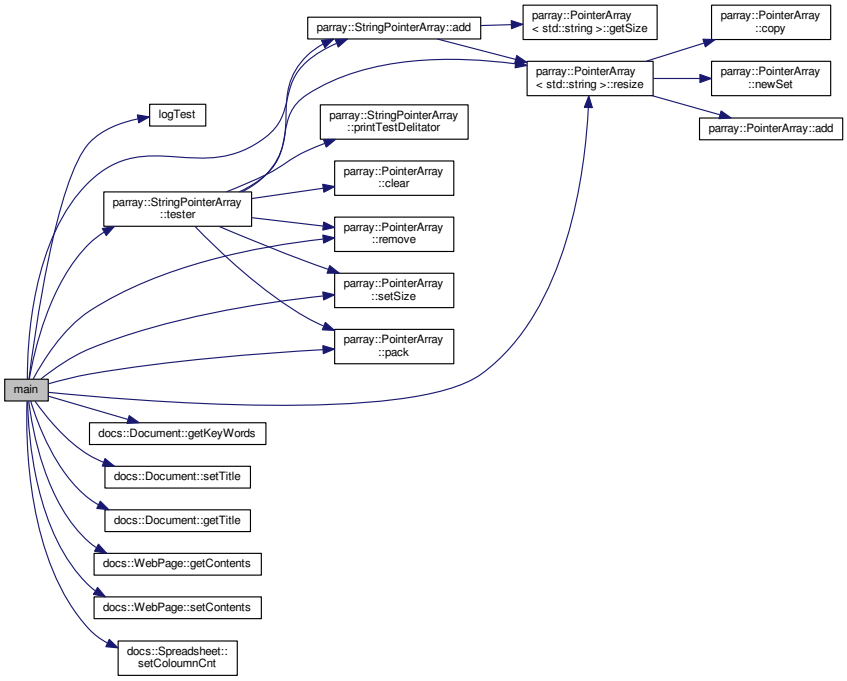
create a new Spreadsheet with given number of columns and rows.

create a new Spreadsheet with a given key words set, a title, the number of row and columns.

create a new SpreadSheet as the copy of the previous and set the title.

put all the created Document in a list and print its contents.

Here is the call graph for this function:

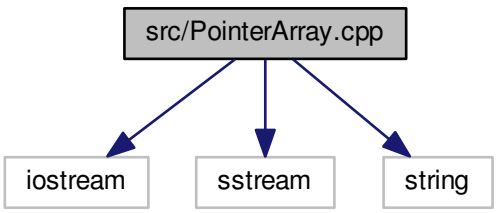


7.4 src/PointerArray.cpp File Reference

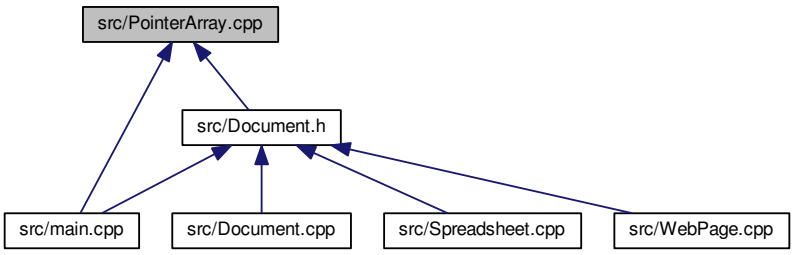
The complete parray (parray::PointerArray<T>) and patch names paces definition.

```
#include <iostream>
#include <sstream>
#include <string>
```

Include dependency graph for PointerArray.cpp:



This graph shows which files directly or indirectly include this file:



Classes

- class parray::PointerArray< T >

  This class describes a manager of a dynamic array of generic type T.*
- class parray::StringPointerArray

  It implements a PointerArray with a string parameter (T).

Namespaces

- patch

  to emulate C++11 and get to_string translator
- parray

  it specify the dynamic array manager for a generic data T and a std::string specialisation.

Functions

- template<typename T >
  std::string patch::to_string (const T &n)

  returns the description of the parameter by using << operator on a fresh std::ostringstream

Variables

- const std::string parray::LOG_HEADER = " PointerArray: "

  the debugging string to be appended on log head by this list manager
- const std::string parray::LOG_HEADER_TEST = "\t[TEST] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Test logs
- const std::string parray::LOG_HEADER_WARNING = "[WARNING] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Warning logs
- const std::string parray::LOG_HEADER_INFO = "[INFO] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Info logs
- const std::string parray::LOG_HEADER_ERROR = "[ERROR] " + LOG_HEADER

  the debugging string to be appended on log head by the StringPointerArray::tester() for Error logs

7.4.1 Detailed Description

The complete parray (parray::PointerArray<T>) and patch names paces definition.

Author

Buoncompagni Luca

Date

Sep 14, 2016

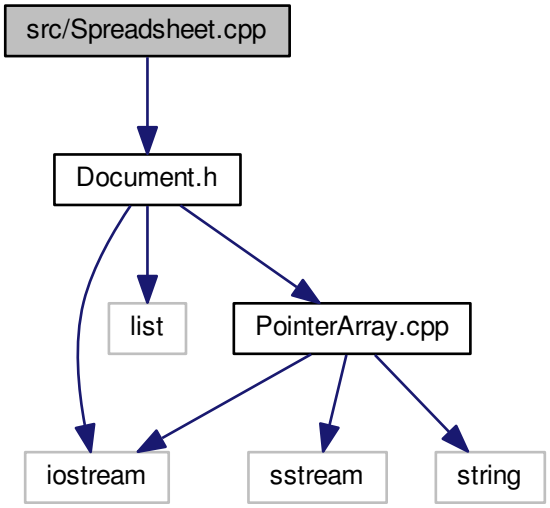
It describes a common templated method to convert an object into a string (patch::to_string()). Also, it defines a templated manager to a dynamic array (parray::PointerArray) as well as a specialisation that deal with arrays of string (parray::StringPointerArray). For the last, also a static method for behavior testing is provided (StringPointerArray::tester()).

7.5 src/Spreadsheet.cpp File Reference

implementation of the docs::Spreadsheet class.

```
#include "Document.h"
```

Include dependency graph for Spreadsheet.cpp:



Namespaces

- docs

  This namespace is used to collect all the document hierarchy classes and types.

7.5.1 Detailed Description

implementation of the docs::Spreadsheet class.

Author

Buoncompagni Luca

Date

Sep 14, 2016

See Also

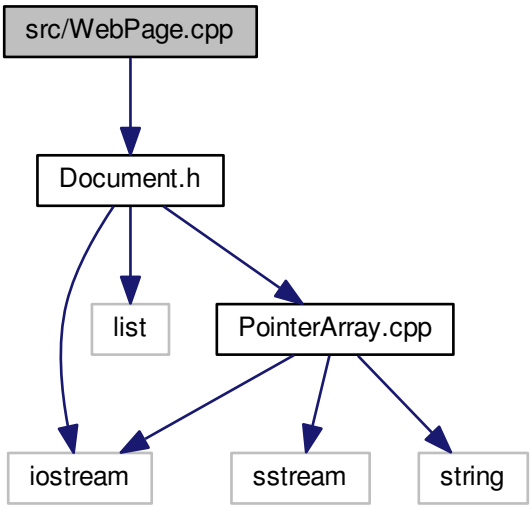
Spreadsheet
Document
docs

7.6 src/WebPage.cpp File Reference

implementation of the docs::WebPage class.

```
#include "Document.h"
```

Include dependency graph for WebPage.cpp:



Namespaces

- docs

  This namespace is used to collect all the document hierarchy classes and types.

7.6.1 Detailed Description

implementation of the docs::WebPage class.

Author

Buoncompagni Luca

Date

Sep 14, 2016

See Also

WebPage
Document
docs

Index