clas-digital

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 CBook Class Reference

Public Member Functions

- CBook (nlohmann::json jMetadata)
- const std::string & getKey ()
- const std::string & getPath ()

  getter function to return the path to the directory of a book
- std::string getOcrPath ()
- bool getOcr ()
- CMetadata & getMetadata ()
- std::vector< std::string > getCollections ()
- std::string getAuthor ()
- std::string getTitle ()
- int getDate ()
- void setPath (std::string sPath)
- void createMapWords ()

  Create a map of all word of this book.
- void safePages ()

  safe map of all words and pages on which word occures to disc
- void loadPages (std::map< std::string, std::vector< size_t >> &mapWordsPages)

  load words and pages on which word occures into map
- std::list< size_t > * getPagesFull (std::string sWord)

  return list of all pages on which word occures.
- std::map< int, std::vector< std::string > > * getPagesContains (std::string sWord)

  return map of all pages + words found on page (Contains).
- std::map< int, std::vector< std::string > > * getPagesFuzzy (std::string sWord)

  return map of all pages + words found on page (fuzzy).
- std::map< int, std::vector< std::string > > * findPagesContains (std::string sWord, std::map< std::string, std::vector< size_t >> &mapWordsPages)

  Create map of pages and found words for i-word (Contains).
- std::map< int, std::vector< std::string > > * findPagesFuzzy (std::string sWord, std::map< std::string, std↩ ::vector< size_t >> &mapWordsPages)

  Create map of pages and found words for i-word (fuzzy).
- void removePages (std::map< int, std::vector< std::string >> *mapPages, std::map< int, std::vector< std::string >> *results2)

  Remove all elements from mapPages, which do not exist in results2.

4.1.1 Constructor & Destructor Documentation

4.1.1.1 CBook()

```
CBook::CBook (
            nlohmann::json jMetadata )
```

Constructor

Parameters

in	*sPath*	Path to book
in	*map*	map of words in book

4.1.2 Member Function Documentation

4.1.2.1 createMapWords()

```
void CBook::createMapWords ( )
```

Create a map of all word of this book.

checks whether book already has map of words, if not it create them

4.1.2.2 findPagesContains()

```
std::map< int, std::vector< std::string > > * CBook::findPagesContains (
            std::string sWord,
            std::map< std::string, std::vector< size_t >> & mapWordsPages )
```

Create map of pages and found words for i-word (Contains).

Returns

map of all pages on which word was found.

4.1.2.3 findPagesFuzzy()

```
std::map< int, std::vector< std::string > > * CBook::findPagesFuzzy (
            std::string sWord,
            std::map< std::string, std::vector< size_t >> & mapWordsPages )
```

Create map of pages and found words for i-word (fuzzy).

Returns

map of all pages on which word was found.

4.1.2.4 getAuthor()

```
std::string CBook::getAuthor ( )
```

Returns

lastName, or Name of author

4.1.2.5 getCollections()

```
std::vector< std::string > CBook::getCollections ( )
```

Returns

vector with all collections this book is in

4.1.2.6 getDate()

```
int CBook::getDate ( )
```

Returns

date or -1 if date does not exists or is currupted

4.1.2.7 getKey()

```
const std::string & CBook::getKey ( )
```

Returns

Key of the book, after extracting it from the path

4.1.2.8 getMetadata()

```
CMetadata & CBook::getMetadata ( )
```

Returns

info.json of book

4.1.2.9 getOcr()

```
bool CBook::getOcr ( )
```

Returns

Boolean, whether book contains ocr or not

4.1.2.10 getOcrPath()

```
std::string CBook::getOcrPath ( )
```

Returns

Path to directory of the book
Path to the ocr.txt file

4.1.2.11 getPagesContains()

```
std::map< int, std::vector< std::string > > * CBook::getPagesContains (
            std::string sWord )
```

return map of all pages + words found on page (Contains).

Parameters

in	*sWord*	searched word.

Returns

map of pages with vector of words found on this page.

4.1.2.12 getPagesFull()

```
std::list< size_t > * CBook::getPagesFull (
            std::string sWord )
```

return list of all pages on which word occures.

Parameters

in	*sWord*	searched word

Returns

list of pages on which searched word accures

4.1.2.13 getPagesFuzzy()

```
std::map< int, std::vector< std::string > > * CBook::getPagesFuzzy (
            std::string sWord )
```

return map of all pages + words found on page (fuzzy).

Parameters

in	*sWord*	searched word

Returns

map of pages with vector of words found on this page

4.1.2.14 getPath()

```
const std::string & CBook::getPath ( )
```

getter function to return the path to the directory of a book

Returns

string (Path to directory of the book)
Path to directory of the book

4.1.2.15 getTitle()

```
std::string CBook::getTitle ( )
```

Returns

title of book

4.1.2.16 setPath()

```
void CBook::setPath (
            std::string sPath )
```

Parameters

in	*path*	set Path to book)

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

- src/books/CBook.hpp
- src/books/Book.cpp

4.2 CBookManager Class Reference

Public Member Functions

- std::map< std::string, CBook > & getMapOfBooks ()
- bool initialize ()

  load all books.
- void updateZotero (nlohmann::json j_Items)

  parse json of all items. If item exists, change metadata of item, create new book.
- void addBook (std::string sKey)

  add a book, or rather: add ocr to book
- std::map< std::string, CBook * > * search (unsigned long long id)

  search function calling fitting function from search class
- void createMapWords ()

  create map of all words (key) and books in which the word occurs (value)
- void createMapWordsTitle ()

create map of all words (key) and book-titles in which the word occurs (value)

- void addSearch (CSearch *search)

  add a new search

- float getProgress (unsigned long long id)

  get progress of given search

- void deleteSearch (unsigned long long id)

  delete given search and erase from map

4.2.1 Member Function Documentation

4.2.1.1 addBook()

```
void CBookManager::addBook (
            std::string sKey )
```

add a book, or rather: add ocr to book

Parameters

in	*sKey*	key to book

4.2.1.2 getMapOfBooks()

```
std::map< std::string, CBook > & CBookManager::getMapOfBooks ( )
```

Returns

map of all book

4.2.1.3 getProgress()

```
float CBookManager::getProgress (
            unsigned long long id )
```

get progress of given search

Returns

float indicating progress

4.2.1.4 initialize()

```
bool CBookManager::initialize ( )
```

load all books.

Returns

boolean for successful of not

4.2.1.5 search()

```
std::map< std::string, CBook * > * CBookManager::search (
            unsigned long long id )
```

search function calling fitting function from search class

Returns

list of all found books

Parameters

in	*searchOPts*	

Returns

list of all found books

4.2.1.6 updateZotero()

```
void CBookManager::updateZotero (
            nlohmann::json j_items )
```

parse json of all items. If item exists, change metadata of item, create new book.

Parameters

in	*j_items*	json with all items

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

- src/books/CBookManager.hpp
- src/books/Bookmanager.cpp

4.3 CMetadata Class Reference

Public Member Functions

- CMetadata (nlohmann::json jMetadata)

  constructor
- void **setMetadata** (nlohmann::json jMetadata)
- nlohmann::json getMetadata ()

  return metadata as json
- std::string getMetadata (std::string sSearch)

  getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...)
- std::string getMetadata (std::string sSearch, std::string sFrom)

  getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom: from which json (f.e. title -> data -> title)
- std::string getMetadata (std::string sSearch, std::string sFrom1, std::string sFrom2)

  getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author)
- std::string getMetadata (std::string sSearch, std::string sFrom1, std::string sFrom2, int in)

  getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author) int (index: in case of list: which element from list)
- std::vector< std::string > getCollections ()
- std::string getAuthor ()
- std::string getTitle ()
- int getDate ()
- std::string getShow ()

4.3.1 Constructor & Destructor Documentation

4.3.1.1 CMetadata()

```
CMetadata::CMetadata (
            nlohmann::json jMetadata )
```

constructor

Parameters

in	*jMetadata*	json with metadata

4.3.2 Member Function Documentation

4.3.2.1 getAuthor()

```
std::string CMetadata::getAuthor ( )
```

Returns

lastName, or Name of author

4.3.2.2 getCollections()

```
std::vector< std::string > CMetadata::getCollections ( )
```

Returns

vector with all collections this book is in

4.3.2.3 getDate()

```
int CMetadata::getDate ( )
```

Returns

date or -1 if date does not exists or is currupted

4.3.2.4 getMetadata() [1/4]

```
std::string CMetadata::getMetadata (
            std::string sSearch )
```

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...)

Returns

string

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...)

Returns

string

4.3.2.5 getMetadata() [2/4]

```
std::string CMetadata::getMetadata (
            std::string sSearch,
            std::string sFrom )
```

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom: from which json (f.e. title -> data -> title)

Returns

string

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom: from which json (f.e. title -> data -> title)

Returns

string

4.3.2.6 getMetadata() [3/4]

```
std::string CMetadata::getMetadata (
            std::string sSearch,
            std::string sFrom1,
            std::string sFrom2 )
```

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author)

Returns

string

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author)

Returns

string

4.3.2.7 getMetadata() [4/4]

```
std::string CMetadata::getMetadata (
            std::string sSearch,
            std::string sFrom1,
            std::string sFrom2,
            int in )
```

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author) int (index: in case of list: which element from list)

Returns

string

getter function to return selected metadata string (sSearch: which metadata (f.e. title, date...) string (sFrom2: from which json (f.e. title -> data -> title) string (sFrom2: in json from which json (f.e. author -> data creators -> author) int (index: in case of list: which element from list)

Returns

string

4.3.2.8 getShow()

```
std::string CMetadata::getShow ( )
```

Returns

string with Auhtor + first 6 words 15 words of title + date

4.3.2.9 getTitle()

```
std::string CMetadata::getTitle ( )
```

Returns

title of book

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

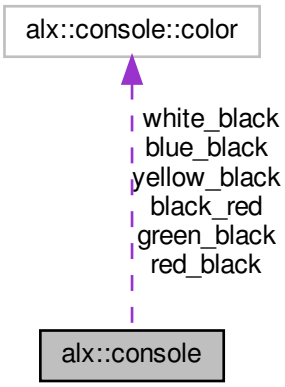
- src/books/CMetadata.hpp
- src/books/Metadata.cpp

4.4 alx::console Class Reference

Basic console class, creates a USER interface in the terminal based on the ncurses library.

`#include <console.hpp>`

Collaboration diagram for alx::console:



Public Member Functions

- ~console ()
  The destructor this resets the console to the original state and deletes all allocated objects.
- template<typename ... Args>
  void write (Args... args)
  This function atomically prints multiple arguments on screen and resets the color of the terminal the back to white_black.
- template<typename ... Args>
  void writeln (Args... args)
  This function is the same as write, but includes a new line at the end of the argument printing.
- std::string getCommand ()
  Reads a string from the user input and returns it on enter.
- void SetColor (color x)
  Sets the color to the specified color given to the function.
- console & operator<< (std::string strs)
  Mimicks the basic cout operator only used for basic printing.
- console & operator<< (int x)
  Mimicks the basic cout operator only used for basic printing.
- void flush ()
  Flushes all pending changes to the console write will do this automatically for you.

Static Public Member Functions

- static console & GetConsole ()
  This function manages the Singleton instance of the console as there can always only be one console at one time.

Static Public Attributes

- static color red_black

  The color foreground red, background black in the terminal.
- static color white_black

  The color foreground white, background black in the terminal, this is the default color.
- static color green_black

  The color foreground green, background black in the terminal.
- static color yellow_black

  The color foreground yellow, background black in the terminal.
- static color blue_black

  The color foreground blue, background black in the terminal.
- static color black_red

  The color foreground black, background red used for error messages!

4.4.1 Detailed Description

Basic console class, creates a USER interface in the terminal based on the ncurses library.

4.4.2 Member Function Documentation

4.4.2.1 getCommand()

```
std::string alx::console::getCommand ( )
```

Reads a string from the user input and returns it on enter.

Returns

The string the user entered

4.4.2.2 GetConsole()

```
console & alx::console::GetConsole ( )  [static]
```

This function manages the Singleton instance of the console as there can always only be one console at one time.

Returns

A reference to the only instance of the console

4.4.2.3 operator<<() [1/2]

```
console & alx::console::operator<< (
            std::string strs )
```

Mimicks the basic cout operator only used for basic printing.

Parameters

strs	The string to print on screen

Returns

A reference to itself for function chaining e. g.

```
alx::cout<<"Hallo "<<"Welt"<<"\n";
```

4.4.2.4 operator<<() [2/2]

```
console & alx::console::operator<< (
            int x )
```

Mimicks the basic cout operator only used for basic printing.

Parameters

x	The string to print on screen

Returns

A reference to itself for function chaining e. g.

```
alx::cout<<"Hallo "<<"Welt"<<10;
```

4.4.2.5 write()

```
template<typename ... Args>
void alx::console::write (
            Args... args ) [inline]
```

This function atomically prints multiple arguments on screen and resets the color of the terminal the back to white↩ _black.

Parameters

args	The arguments to print on screen

4.4.2.6 writeln()

```
template<typename ... Args>
```

```
void alx::console::writeln (
            Args...  args )  [inline]
```

This function is the same as write, but includes a new line at the end of the argument printing.

Parameters

args	The arguments to print on screen

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

- src/console/console.hpp
- src/console/console.cpp
- src/console/unittestconsole.cpp

4.5 CSearch Class Reference

Public Member Functions

- CSearch (CSearchOptions *searchOpts, unsigned long long sID)
  constructor
- unsigned long long getID ()
- std::string getSearchedWord ()
- float getProgress ()
- void setWord (std::string sWord)
  set searched word. param[in] searchedWord set searched word
- std::map< std::string, CBook * > * **search** (std::map< std::string, std::map< std::string, CBook *>> &mWs, std::map< std::string, std::map< std::string, CBook *>> &mWsTitle)
- void normalSearch (std::map< std::string, std::map< std::string, CBook *>> &mapWords, std::map< std::string, CBook *> *mapSR)
  search full-match
- void containsSearch (std::map< std::string, std::map< std::string, CBook *>> &mapWords, std::map< std::string, CBook *> *mapSR)
  search contains
- void fuzzySearch (std::map< std::string, std::map< std::string, CBook *>> &mapWords, std::map< std::string, CBook *> *mapSR)
  search fuzzy
- void removeBooks (std::map< std::string, CBook *> *mapSR)
  remove all books that do not match with searchoptions
- bool checkSearchOptions (CBook *book)
  check whether book-metadata matches with searchoptions
- std::list< CBook * > * convertToList (std::map< std::string, CBook *> *mapBooks)
  convert to list
- void deleteSearchOptions ()
  delete searchOptions

4.5.1 Member Function Documentation

4.5.1.1 checkSearchOptions()

```
bool CSearch::checkSearchOptions (
            CBook * book )
```

check whether book-metadata matches with searchoptions

Parameters

in	*book*	to be checked return Boolean

4.5.1.2 containsSearch()

```
void CSearch::containsSearch (
            std::map< std::string, std::map< std::string, CBook *>> & mapWords,
            std::map< std::string, CBook *> * mapSR )
```

search contains

Parameters

in	*mapWords*	map of all words with a list of books in which this where accures
in,out	*mapSR*	map of search results
in	*mapWords*	map of all words with a list of books in which this word accures
in,out	*mapSR*	searchresults

4.5.1.3 convertToList()

```
std::list< CBook * > * CSearch::convertToList (
            std::map< std::string, CBook *> * mapBooks )
```

convert to list

Returns

list of searchresulst

4.5.1.4 fuzzySearch()

```
void CSearch::fuzzySearch (
            std::map< std::string, std::map< std::string, CBook *>> & mapWords,
            std::map< std::string, CBook *> * mapSR )
```

search fuzzy

Parameters

in	*mapWords*	map of all words with a list of books in which this word accures
in,out	*mapSR*	searchresults

4.5.1.5 getID()

```
unsigned long long CSearch::getID ( )
```

Returns

id of search

4.5.1.6 getProgress()

```
float CSearch::getProgress ( )
```

Returns

progress

4.5.1.7 getSearchedWord()

```
std::string CSearch::getSearchedWord ( )
```

Returns

searched word from searchOptions

4.5.1.8 normalSearch()

```
void CSearch::normalSearch (
            std::map< std::string, std::map< std::string, CBook *>> & mapWords,
            std::map< std::string, CBook *> * mapSR )
```

search full-match

Parameters

in	*mapWords*	map of all words with a list of books in which this where accures
in,out	*mapSR*	map of search results
in	*mapWords*	map of all words with a list of books in which this word accures
in,out	*mapSR*	searchresults

4.5.1.9 removeBooks()

```
void CSearch::removeBooks (
            std::map< std::string, CBook *> * mapSR )
```

remove all books that do not match with searchoptions

Parameters

in,out	*mapSR*	map of search results

4.5.1.10 setWord()

```
void CSearch::setWord (
            std::string sWord )
```

set searched word. param[in] searchedWord set searched word

param[in] searchedWord set searched word

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

- src/books/CSearch.hpp
- src/books/Search.cpp

4.6 CSearchOptions Class Reference

Public Member Functions

- CSearchOptions ()
- CSearchOptions (std::string chSearchedWord, int fuzzyness, std::vector< std::string > sCollections, bool onlyTitle, bool onlyOCR, std::string slastName, int from, int to, bool full)

  Constructor.
- void initialise (std::string chSearchedWord, int fuzzyness, std::vector< std::string > pillar, bool onlyTitle, bool onlyOCR, std::string slastName, int from, int to, bool full)

  initialise search options outside of constructor

- std::string getSearchedWord () const
- int getFuzzyness () const
- std::vector< std::string > getCollections () const
- bool getOnlyTitle () const
- bool getOnlyOcr () const
- std::string getLastName () const
- int getFrom () const
- int getTo () const
- bool getAccess () const
- void setSearchedWord (std::string searchedWord)

4.6.1 Constructor & Destructor Documentation

4.6.1.1 CSearchOptions() [1/2]

```
CSearchOptions::CSearchOptions ( )
```

default constructor.

4.6.1.2 CSearchOptions() [2/2]

```
CSearchOptions::CSearchOptions (
            std::string chSearchedWord,
            int fuzzyness,
            std::vector< std::string > sCollections,
            bool onlyTitle,
            bool onlyOCR,
            std::string slastName,
            int from,
            int to,
            bool full )
```

Constructor.

Parameters

in	*chSearchedWord*	searched word
in	*fuzzyness*	value of fuzzyness
in	*sCollections*	collections in which to be searched
in	*onlyTitle*	search only in title?
in	*onlyOCR*	search only in ocr (if exists)
in	*slastName*	las name of author
in	*from*	date from which books shall be searched
in	*to*	date to which books shall be searched

4.6.2 Member Function Documentation

4.6.2.1 getAccess()

```
bool CSearchOptions::getAccess ( ) const
```

Returns

get user access

4.6.2.2 getCollections()

```
std::vector< std::string > CSearchOptions::getCollections ( ) const
```

Returns

selected pillars

4.6.2.3 getFrom()

```
int CSearchOptions::getFrom ( ) const
```

Returns

year from which books shall be searched in

4.6.2.4 getFuzzyness()

```
int CSearchOptions::getFuzzyness ( ) const
```

Returns

selected fuzzyness

4.6.2.5 getLastName()

```
std::string CSearchOptions::getLastName ( ) const
```

Returns

last name of selected author

4.6.2.6 getOnlyOcr()

```
bool CSearchOptions::getOnlyOcr ( ) const
```

Returns

whether search only in ocr (if exists)

4.6.2.7 getOnlyTitle()

```
bool CSearchOptions::getOnlyTitle ( ) const
```

Returns

whether search only in title

4.6.2.8 getSearchedWord()

```
std::string CSearchOptions::getSearchedWord ( ) const
```

Returns

searched word

4.6.2.9 getTo()

```
int CSearchOptions::getTo ( ) const
```

Returns

year to which books shall be searched

4.6.2.10 initialise()

```
void CSearchOptions::initialise (
            std::string chSearchedWord,
            int fuzzyness,
            std::vector< std::string > pillar,
            bool onlyTitle,
            bool onlyOCR,
            std::string slastName,
            int from,
            int to,
            bool full )
```

initialise search options outside of constructor

Parameters

in	*chSearchedWord*	searched word
in	*fuzzyness*	value of fuzzyness
in	*sCollections*	collections in which to be searched
in	*onlyTitle*	search only in title?
in	*onlyOCR*	search only in ocr (if exists)
in	*slastName*	las name of author
in	*from*	date from which books shall be searched
in	*to*	date to which books shall be searched

4.6.2.11 setSearchedWord()

```
void CSearchOptions::setSearchedWord (
            std::string searchedWord )
```

Parameters

in	*searchedWord*	new searched word

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

- src/books/CSearchOptions.hpp
- src/books/SearchOptions.cpp

4.7 debug::empty Struct Reference

This structure does not do anything with its constructor arguments it also does not print them.

```
#include <debug.hpp>
```

Public Member Functions

- template<typename ... Args>
  empty (Args...)

  This function does not do anything at all.

4.7.1 Detailed Description

This structure does not do anything with its constructor arguments it also does not print them.

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

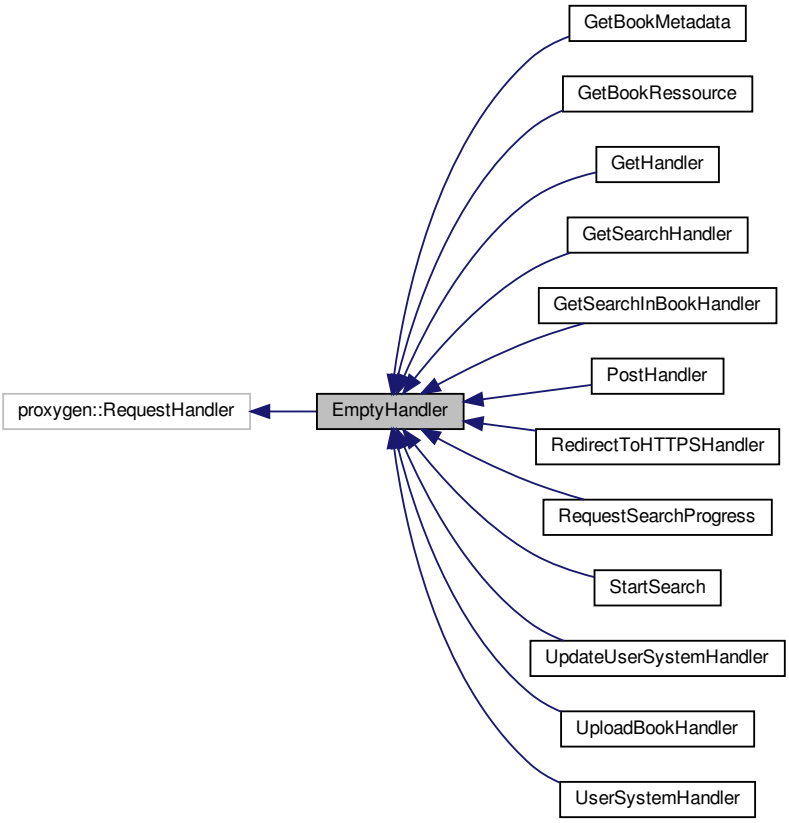
- src/util/debug.hpp

4.8 EmptyHandler Class Reference

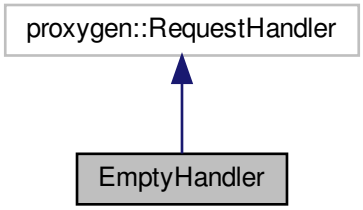
Small class used for setting the default empty method for every request handler.

`#include <BasicHandlers.hpp>`

Inheritance diagram for EmptyHandler:



Collaboration diagram for EmptyHandler:



Public Member Functions

- virtual void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Dummy function for the proxygen virtual function onRequest.
- virtual void onBody (std::unique_ptr< folly::IOBuf > body) noexcept override

  Dummy empty handler for the on body proxygen virtual function.
- virtual void onUpgrade (proxygen::UpgradeProtocol proto) noexcept override

  Dummy empty handler for the proxygen function onUpgrade.
- virtual void requestComplete () noexcept override

  The empty handler for the proxygen function requestComplete.
- virtual void onError (proxygen::ProxygenError err) noexcept override

  The dummy function for the proxygen function onError.
- virtual void onEgressPaused () noexcept override

  The dummy function for the proxygen function inEgressPaused.
- virtual void onEgressResumed () noexcept override

  The dummy function for the proxygen function onEgressResumed.
- virtual void onEOM () noexcept override

  The dummy function for the end of message function.

Public Attributes

- std::shared_ptr< User > _user

  The user this specific request is from.

4.8.1 Detailed Description

Small class used for setting the default empty method for every request handler.

This Handler is just used to provide an default empty method for the pure virtual class Request Handler, therefore it does not do anything at all expect implementing these empty methods Usage is as follows

```
class MyRequestHandler : public EmptyHandler
{
        public:
    void onRequest(std::unique_ptr<proxygen::HTTPMessage> headers)
        noexcept override;  //This is okay now just defining the function you use in your handler

};
```

4.8.2 Member Function Documentation

4.8.2.1 onBody()

```
virtual void EmptyHandler::onBody (
            std::unique_ptr< folly::IOBuf > body )  [inline], [override], [virtual], [noexcept]
```

Dummy empty handler for the on body proxygen virtual function.

Parameters

body	The body provided by proxygen for this message, can be called multiple times for the same request if there is a lot of data

Reimplemented in PostHandler, UploadBookHandler, and UpdateUserSystemHandler.

4.8.2.2 onError()

```
virtual void EmptyHandler::onError (
            proxygen::ProxygenError err )  [inline], [override], [virtual], [noexcept]
```

The dummy function for the proxygen function onError.

Parameters

err	The error that occured

4.8.2.3 onRequest()

```
virtual void EmptyHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [inline], [override], [virtual],
[noexcept]
```

Dummy function for the proxygen virtual function onRequest.

Parameters

headers	The HTTP Message provided by proxygen

Reimplemented in PostHandler, GetHandler, UploadBookHandler, GetBookMetadata, RedirectToHTTPSHandler, GetSearchInBookHandler, GetSearchHandler, GetBookRessource, and UserSystemHandler.

4.8.2.4 onUpgrade()

```
virtual void EmptyHandler::onUpgrade (
            proxygen::UpgradeProtocol proto )  [inline], [override], [virtual], [noexcept]
```

Dummy empty handler for the proxygen function onUpgrade.

Parameters

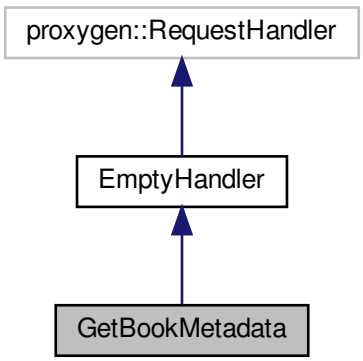
proto	The new protocol to follow from there on

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

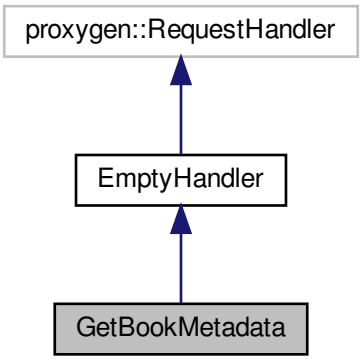
- src/server/BasicHandlers.hpp

4.9 GetBookMetadata Class Reference

Inheritance diagram for GetBookMetadata:



Collaboration diagram for GetBookMetadata:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Returns the metadata for a specified book request must be like /getBookMetadata?book=ITEM_KEY.

Additional Inherited Members

4.9.1 Member Function Documentation

4.9.1.1 onRequest()

```
void GetBookMetadata::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Returns the metadata for a specified book request must be like /getBookMetadata?book=ITEM_KEY.

Parameters

headers	The request headers send by the user

Reimplemented from EmptyHandler.

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

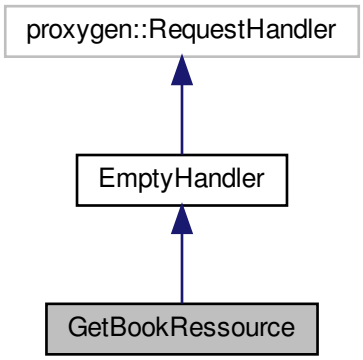
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.10 GetBookRessource Class Reference

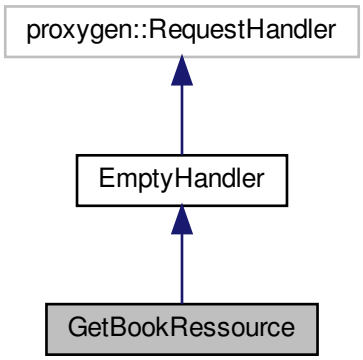
Returns either all the books in the server or all the files in one book or a specific ressource from a specific book.

`#include <URIObjects.hpp>`

Inheritance diagram for GetBookRessource:



Collaboration diagram for GetBookRessource:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Sends back specific informations regarding the user profile and profile changes.

Additional Inherited Members

4.10.1 Detailed Description

Returns either all the books in the server or all the files in one book or a specific ressource from a specific book.

4.10.2 Member Function Documentation

4.10.2.1 onRequest()

```
void GetBookRessource::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Sends back specific informations regarding the user profile and profile changes.

Parameters

headers	The headers provided by the proxygen library

Reimplemented from EmptyHandler.

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

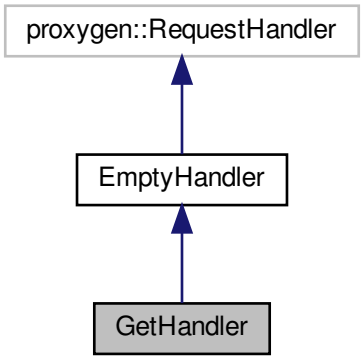
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.11 GetHandler Class Reference

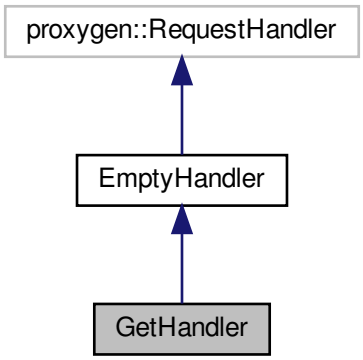
The Basic Get Handler which does almost all of the server disk IO acesses.

```
#include <BasicHandlers.hpp>
```

Inheritance diagram for GetHandler:



Collaboration diagram for GetHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Tries to satisfy a ressource request, do access right check and provide either an error response or the ressource response.

Additional Inherited Members

4.11.1 Detailed Description

The Basic Get Handler which does almost all of the server disk IO acesses.

Most of the function

4.11.2 Member Function Documentation

4.11.2.1 onRequest()

```
void GetHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Tries to satisfy a ressource request, do access right check and provide either an error response or the ressource response.

Parameters

headers	The HTTP headers for this request provided by proxygen

Reimplemented from EmptyHandler.

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

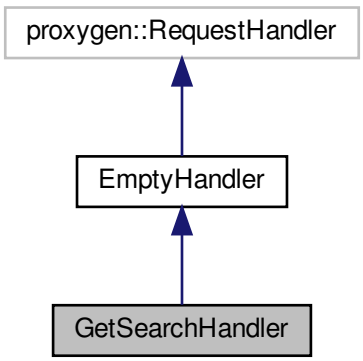
- src/server/BasicHandlers.hpp
- src/server/GetHandler.cpp

4.12 GetSearchHandler Class Reference

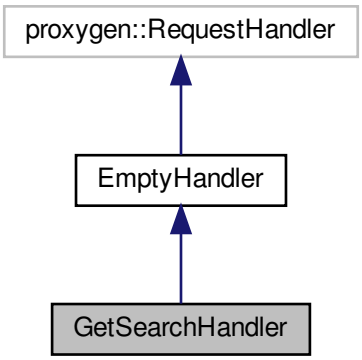
Handles the general search in all books.

```
#include <URIObjects.hpp>
```

Inheritance diagram for GetSearchHandler:



Collaboration diagram for GetSearchHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override
  Tries to satify the search request and send back an json with all found books to the server.

Static Public Member Functions

- static CBookManager & GetBookManager ()
  Returns an instance of the global book manager used to manage all books.

Additional Inherited Members

4.12.1 Detailed Description

Handles the general search in all books.

4.12.2 Member Function Documentation

4.12.2.1 GetBookManager()

```
CBookManager & GetSearchHandler::GetBookManager ( )  [static]
```

Returns an instance of the global book manager used to manage all books.

Returns

Returns the global book manager which manages all books

4.12.2.2 onRequest()

```
void GetSearchHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Tries to satify the search request and send back an json with all found books to the server.

Parameters

headers	The http message received from the the client with the parameters for the search

Reimplemented from EmptyHandler.

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

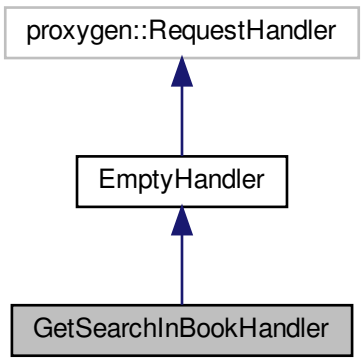
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.13 GetSearchInBookHandler Class Reference

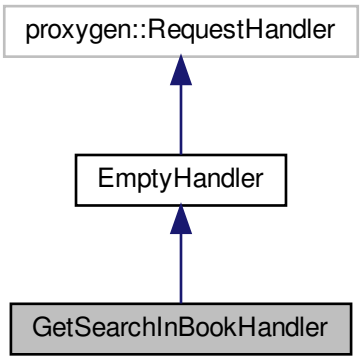
Searches a specific book for a specific word with the given fuzzyness.

```
#include <URIObjects.hpp>
```

Inheritance diagram for GetSearchInBookHandler:



Collaboration diagram for GetSearchInBookHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Searches in a specific book for a specifiifc word with the given fuzzynes and returns a json file with the results from the search.

Additional Inherited Members

4.13.1 Detailed Description

Searches a specific book for a specific word with the given fuzzyness.

4.13.2 Member Function Documentation

4.13.2.1 onRequest()

```
void GetSearchInBookHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Searches in a specific book for a specifiifc word with the given fuzzynes and returns a json file with the results from the search.

Parameters

in	*headers*	The headers for the http request received from the user

Reimplemented from EmptyHandler.

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

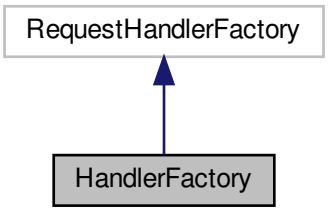
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.14 HandlerFactory Class Reference

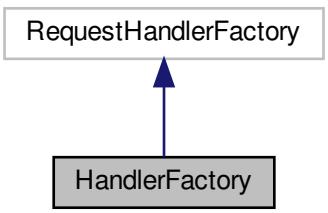
The HandlerFactory is used to instantiate the proxygen server and creates all request handler for every request directed to the server This class redirects every request to the right handler, in order to do so it keeps book of every URI object registered and sends the request to the URI object if there is any else the request goes to the default handler.

```
#include <HandlerFactory.hpp>
```

Inheritance diagram for HandlerFactory:



Collaboration diagram for HandlerFactory:



Public Member Functions

- void onServerStart (folly::EventBase ∗) noexcept override
- void onServerStop () noexcept override

  Handles the cleanup and behaviour if the server is stopped, in this case it does nothing at all because no cleanup is needed.
- RequestHandler ∗ onRequest (RequestHandler ∗, HTTPMessage ∗hdr) noexcept override
- template<typename T >
  RequestHandler ∗ onRequest (std::map< std::string, EmptyHandler ∗(∗)()> &mp, HTTPMessage ∗hdr)

  This function handles all requests to either getMap or postMap depending on the first parameter It creates the right RequestHandler and sets the user parameter in the handler class to the user that does the request.

Static Public Member Functions

- static void parseCommands (std::string command)

  Used to parse and execute commands from the user.

4.14.1 Detailed Description

The HandlerFactory is used to instantiate the proxygen server and creates all request handler for every request directed to the server This class redirects every request to the right handler, in order to do so it keeps book of every URI object registered and sends the request to the URI object if there is any else the request goes to the default handler.

4.14.2 Member Function Documentation

4.14.2.1 onRequest() [1/2]

```
RequestHandler* HandlerFactory::onRequest (
            RequestHandler * ,
            HTTPMessage * hdr )  [inline], [override], [noexcept]
```

As soon as an request is received this function gets called by the proxygen server and expects to return an Request handler. This function does nothing more than to select the correct request handler and return a new instance of him to the proxygen server.

Parameters

hdr	The header of the message received, is used to set identify the user the message was sent from

Returns

Returns the correct request handler for the requested URI

4.14.2.2 onRequest() [2/2]

```
template<typename T >
RequestHandler* HandlerFactory::onRequest (
            std::map< std::string, EmptyHandler *(*)()> & mp,
            HTTPMessage * hdr )  [inline]
```

This function handles all requests to either getMap or postMap depending on the first parameter It creates the right RequestHandler and sets the user parameter in the handler class to the user that does the request.

Parameters

mp	The map to use for looking up the URI function mapping
hdr	The http message received by the user

Returns

The request handler which is about to handle the received request

4.14.2.3 onServerStart()

```
void HandlerFactory::onServerStart (
            folly::EventBase *  )  [inline], [override], [noexcept]
```

Initialises the HandlerFactory and gets called by the proxygen server as the server starts up. Reserve all the post URI Objects as well as all the GET URI Objects.

4.14.2.4 parseCommands()

```
void HandlerFactory::parseCommands (
            std::string command )  [static]
```

Used to parse and execute commands from the user.

Parameters

command	The command to execute

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

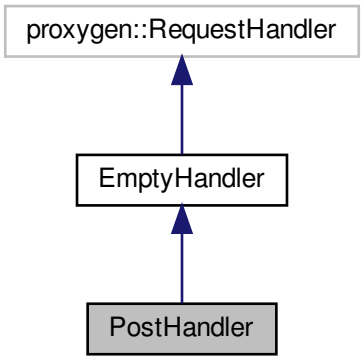
- src/server/HandlerFactory.hpp
- src/server/HandlerFactory.cpp

4.15 PostHandler Class Reference

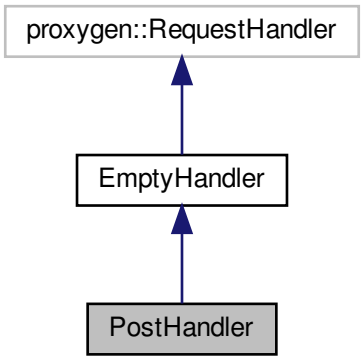
Handles the basic posts to the server mainly does the login and not much else.

`#include <BasicHandlers.hpp>`

Inheritance diagram for PostHandler:



Collaboration diagram for PostHandler:



Public Member Functions

- void onBody (std::unique_ptr< folly::IOBuf > body) noexcept override

  Proxygen callback for body data tries to parse user name and password from the data.
- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  The Function determines if the user wants to login or logout and handles the request accordingly.

Additional Inherited Members

4.15.1 Detailed Description

Handles the basic posts to the server mainly does the login and not much else.

4.15.2 Member Function Documentation

4.15.2.1 onBody()

```
void PostHandler::onBody (
            std::unique_ptr< folly::IOBuf > body )  [override], [virtual], [noexcept]
```

Proxygen callback for body data tries to parse user name and password from the data.

Parameters

body	The data send with the post request

Reimplemented from EmptyHandler.

4.15.2.2 onRequest()

```
void PostHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

The Function determines if the user wants to login or logout and handles the request accordingly.

Parameters

headers	The http message passed by proxygen to our handler

Reimplemented from EmptyHandler.

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

- src/server/BasicHandlers.hpp
- src/server/PostHandler.cpp

4.16 debug::print Struct Reference

This structure prints everything in order given to the constructor of the class and an endline at the end of all prints.

`#include <debug.hpp>`

Public Member Functions

- template<typename ... Args, typename T >
  print (T t1, Args... args)

  Overloaded constructor prints all arguments in order to stdout.

4.16.1 Detailed Description

This structure prints everything in order given to the constructor of the class and an endline at the end of all prints.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 print()

```
template<typename ...  Args, typename T >
debug::print::print (
             T t1,
             Args...  args )  [inline]
```

Overloaded constructor prints all arguments in order to stdout.

Parameters

t1	The argument to print now
args	The arguments to print next

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

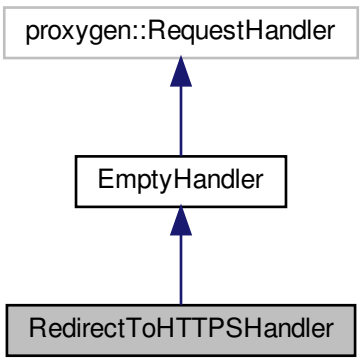
- src/util/debug.hpp

4.17 RedirectToHTTPSHandler Class Reference

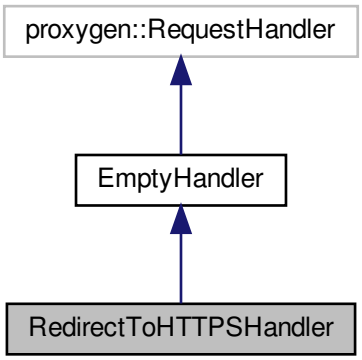
Redirects all request made to http port to the https website.

`#include <URIObjects.hpp>`

Inheritance diagram for RedirectToHTTPSHandler:



Collaboration diagram for RedirectToHTTPSHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Sends back an redirect response no matter what the request is!

Additional Inherited Members

4.17.1 Detailed Description

Redirects all request made to http port to the https website.

4.17.2 Member Function Documentation

4.17.2.1 onRequest()

```
void RedirectToHTTPSHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Sends back an redirect response no matter what the request is!

Parameters

in	*headers*	The http headers send from the user

Reimplemented from EmptyHandler.

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

- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.18 Zotero::Request Struct Reference

Defines the most Basic requests to the zotero API.

`#include <zotero.hpp>`

Static Public Member Functions

- static std::string GetSpecificItem (std::string key)

  The zotero request to get a specific item from the zotero api.
- static std::string GetItemsInSpecificPillar (std::string key)

  The zotero request to get all items from a specific collection out of zotero.

Static Public Attributes

- static constexpr const char GetAllItems [ ] = "/items/top?format=json&include=data,bib,citation&limit=100"

  The zotero request to get all items in the zotero library from zotero.
- static constexpr const char GetAllPillars [ ] = "/collections/top?format=json"

  The zotero request to get all collections from the zotero api.

4.18.1 Detailed Description

Defines the most Basic requests to the zotero API.

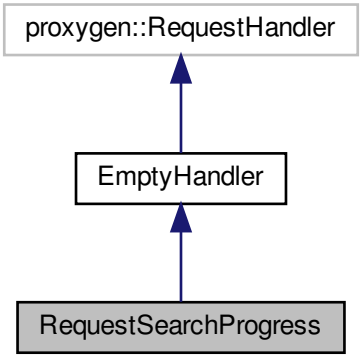
```
Zotero::SendRequest(Zotero::Request::GetAllItems);
//Or this
Zotero::SendRequest(Zotero::Request::GetSpecificItem("
      X2DEFG"));
```

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

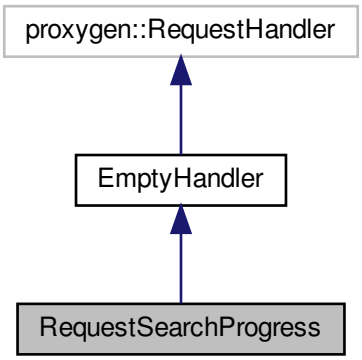
- src/zotero/zotero.hpp
- src/zotero/zotero.cpp

4.19 RequestSearchProgress Class Reference

Inheritance diagram for RequestSearchProgress:



Collaboration diagram for RequestSearchProgress:



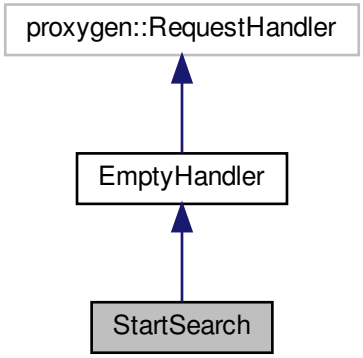
Additional Inherited Members

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

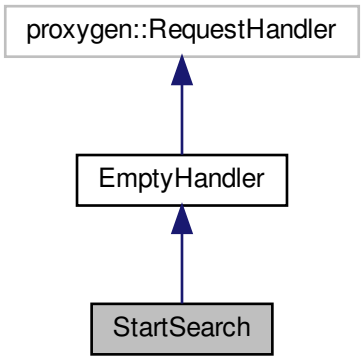
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.20 StartSearch Class Reference

Inheritance diagram for StartSearch:



Collaboration diagram for StartSearch:



Additional Inherited Members

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

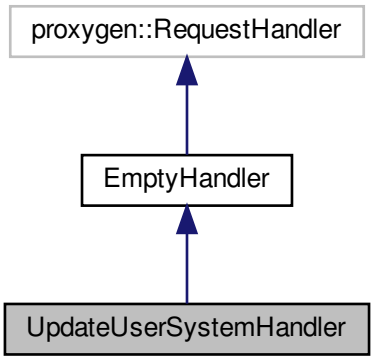
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.21 UpdateUserSystemHandler Class Reference

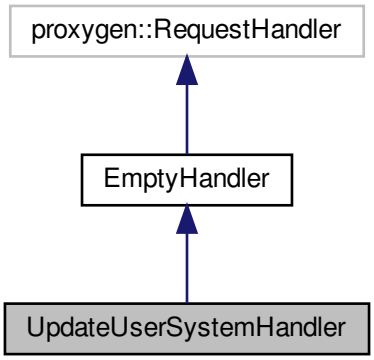
Handles all changes to the user table like create delete and change access.

`#include <URIObjects.hpp>`

Inheritance diagram for UpdateUserSystemHandler:



Collaboration diagram for UpdateUserSystemHandler:



Public Member Functions

- void onBody (std::unique_ptr< folly::IOBuf > body) noexcept override

  Only needs the body which contains the data to change the access rights and create the user. The supported actions are create delete and change user rights.

Additional Inherited Members

4.21.1 Detailed Description

Handles all changes to the user table like create delete and change access.

4.21.2 Member Function Documentation

4.21.2.1 onBody()

```
void UpdateUserSystemHandler::onBody (
            std::unique_ptr< folly::IOBuf > body )  [override], [virtual], [noexcept]
```

Only needs the body which contains the data to change the access rights and create the user. The supported actions are create delete and change user rights.

Parameters

body	The body which contains an array of json with the commands that should be executed

Returns

200 Ok to the client if everything worked

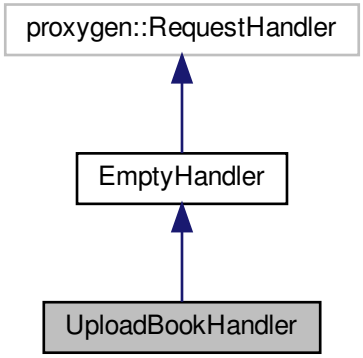
Reimplemented from EmptyHandler.

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

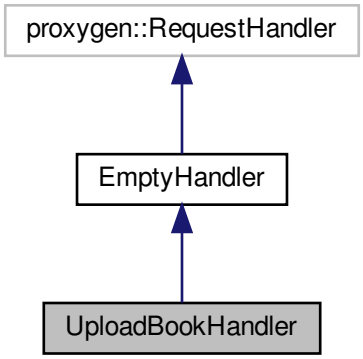
- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.22 UploadBookHandler Class Reference

Inheritance diagram for UploadBookHandler:



Collaboration diagram for UploadBookHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override

  Dummy function for the proxygen virtual function onRequest.
- void onBody (std::unique_ptr< folly::IOBuf > body) noexcept override

  Dummy empty handler for the on body proxygen virtual function.
- void onEOM () noexcept override

  The dummy function for the end of message function.

Additional Inherited Members

4.22.1 Member Function Documentation

4.22.1.1 onBody()

```
void UploadBookHandler::onBody (
            std::unique_ptr< folly::IOBuf > body ) [override], [virtual], [noexcept]
```

Dummy empty handler for the on body proxygen virtual function.

Parameters

body	The body provided by proxygen for this message, can be called multiple times for the same request if there is a lot of data

Reimplemented from EmptyHandler.

4.22.1.2 onRequest()

```
void UploadBookHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers ) [override], [virtual], [noexcept]
```

Dummy function for the proxygen virtual function onRequest.

Parameters

headers	The HTTP Message provided by proxygen

Reimplemented from EmptyHandler.

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

- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.23 URIFile Class Reference

The class contains basic information about the URI file.

```
#include <BasicHandlers.hpp>
```

Public Member Functions

- URIFile (std::string path, int accessRights=0)

  The constructor tells the URI Object has which file path on the disk and the access rights needed to access it.
- URIFile (const URIFile &fl)

  Copy constructur, careful never ever use that!!! It was just created because std::unordered_map needs a copy constructor.
- URIFile (URIFile &&mvConst)

  The move constructor constructs the new object by moving all the data out of the other URIFile object.
- bool doAccessCheck (int acc) const

  Performs an access check on this file with the given access rights.
- const std::string & getPath () const

  Returns a const reference to the path the file points to.
- const std::string & getMimeType () const

  Returns the mime type of the given file path in html reprensentation.
- std::unique_ptr< folly::IOBuf > getBuffer ()

  Returns a unqiue ptr to the clone of the IOBuf so that in can be send to the client in a timely manner.
- std::unique_ptr< folly::IOBuf > & getBufferReference ()

  This function is mainly used if one wants to move the content of the buffer to a file, eg. when the URI File object is short lived.

Static Public Member Functions

- static std::string PrepareDataForSending (nlohmann::json &&js)

4.23.1 Detailed Description

The class contains basic information about the URI file.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 URIFile() [1/3]

```
URIFile::URIFile (
            std::string path,
            int accessRights = 0 )
```

The constructor tells the URI Object has which file path on the disk and the access rights needed to access it.

Parameters

path	The path to the file to load
accessRights	The access rights needed to access the file, the accessRights must be a power of two!

4.23.2.2 URIFile() [2/3]

```
URIFile::URIFile (
            const URIFile & fl )  [inline]
```

Copy constructur, careful never ever use that!!! It was just created because std::unordered_map needs a copy constructor.

Parameters

fl	The file to create this file from

4.23.2.3 URIFile() [3/3]

```
URIFile::URIFile (
            URIFile && mvConst )  [inline]
```

The move constructor constructs the new object by moving all the data out of the other URIFile object.

Parameters

mvConst	The object to move the data away from

4.23.3 Member Function Documentation

4.23.3.1 doAccessCheck()

```
bool URIFile::doAccessCheck (
            int acc ) const
```

Performs an access check on this file with the given access rights.

Parameters

acc	The access rights trying to access the file, can be any positive integer

4.23.3.2 getBuffer()

```
std::unique_ptr< folly::IOBuf > URIFile::getBuffer ( )
```

Returns a unqiue ptr to the clone of the IOBuf so that in can be send to the client in a timely manner.

```
URIFile file("web/index.html",0);
ResponseBuilder(downstream_)
    .status(200,"Ok")
    .header("Content-Type",file.getMimeType())
    .body(file.getBuffer());    //Dont move the buffer as it would remove the data from the buffer inside
       the URIFile class
```

Returns

A unique ptr to a clone of the IOBuf which holds the file data

4.23.3.3 getBufferReference()

```
std::unique_ptr< folly::IOBuf > & URIFile::getBufferReference ( )
```

This function is mainly used if one wants to move the content of the buffer to a file, eg. when the URI File object is short lived.

```
URIFile file("web/index.html",0);
ResponseBuilder(downstream_)
    .status(200,"Ok")
    .header("Content-Type",file.getMimeType())
    .body(std::move(file.getBufferReference()));
    return; //The URIFile object is short lived, no reason to copy the whole buffer so just move the loaded
       file buff away
```

Returns

The reference to a buffer

4.23.3.4 getMimeType()

```
const std::string & URIFile::getMimeType ( ) const
```

Returns the mime type of the given file path in html reprensentation.

```
//This is how to use it
URIFile file("web/index.html",0);
file.getMimeType(); //Will be "text/html"
```

Returns

A const reference to the detected mime type

4.23.3.5 getPath()

```
const std::string & URIFile::getPath ( ) const
```

Returns a const reference to the path the file points to.

Returns

A const reference to the path the file points to

4.23.3.6 PrepareDataForSending()

```
std::string URIFile::PrepareDataForSending (
            nlohmann::json && js )  [static]
```

Prepares the json data for sending must be called after adding a html file with body!!

Parameters

js	The json to be appended as serverJS Object

Returns

The string to get appended to the http message

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

- src/server/BasicHandlers.hpp
- src/server/GetHandler.cpp

4.24 User Class Reference

The basic user class this represents a basic user and stores email password and access rights for this user as well as the current session.

`#include <user_system.hpp>`

Public Member Functions

- User ()
- User (const char *email, const char *pass, int access)
- std::string toJSON () const
- int GetAccessRights () const
- void SetAccessRights (int acc)
- const std::string & GetEmail () const
- const std::string & GetPassword () const
- const std::string & GetSessid () const

  Returns the current session id of the user logged in at the moment.
- void SetSessionId (std::string sessid)

  Sets the session id to a new session id.
- bool DoesMatch (std::string email, std::string passwd) const

Static Public Member Functions

- static bool AccessCheck (const std::shared_ptr< User > &usr, int accRequired)

  Check if the user has the necessary access rights to access this ressource.

4.24.1 Detailed Description

The basic user class this represents a basic user and stores email password and access rights for this user as well as the current session.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 User() [1/2]

```
User::User ( )
```

Default constructor for the User class.

4.24.2.2 User() [2/2]

```
User::User (
            const char * email,
            const char * pass,
            int access )
```

Constructs a user with a given email, password and access rights.

Parameters

email	The email the user got
pass	The password the user will use to login
access	The access rights the user got.

4.24.3 Member Function Documentation

4.24.3.1 AccessCheck()

```
bool User::AccessCheck (
            const std::shared_ptr< User > & usr,
            int accRequired )  [static]
```

Check if the user has the necessary access rights to access this ressource.

Parameters

usr	The user the message is one can be a nullptr as well!
accRequired	The access rights required to access this ressource

Returns

true if the user has enough access rights false otherwise

4.24.3.2 DoesMatch()

```
bool User::DoesMatch (
            std::string email,
            std::string passwd ) const
```

Checks if the given password and email matches the users credentials.

Parameters

email	The email to check against
passwd	The Password to check against

Returns

Returns true if the given credentials matches the user credentials

4.24.3.3 GetAccessRights()

```
int User::GetAccessRights ( ) const
```

Getter for the access rights of the user.

4.24.3.4 GetEmail()

```
const std::string & User::GetEmail ( ) const
```

The getter for the email the user uses.

4.24.3.5 GetPassword()

```
const std::string & User::GetPassword ( ) const
```

The getter for the password of the user.

4.24.3.6 GetSessid()

```
const std::string & User::GetSessid ( ) const
```

Returns the current session id of the user logged in at the moment.

Returns

The session id for the user

4.24.3.7 SetAccessRights()

```
void User::SetAccessRights (
            int acc )
```

Setter for the access rights of the user.

Parameters

acc	The new access rights for the user.

4.24.3.8 toJSON()

```
std::string User::toJSON ( ) const
```

Returns the user information as json file. User information means: email and access rights.

Returns

A string in the json format containing email and access rights of the user

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

- src/login/user_system.hpp
- src/login/user.cpp

4.25 UserHandler Class Reference

The user handler got a list of all available users and manages creating and deleting new users.

```
#include <user_system.hpp>
```

Public Member Functions

- UserHandler (std::string filePath)
  Loads the user table from the specified path and initialises it.
- bool AddUser (std::string email, std::string password, int access)
  Adds a user to the map of current users.
- void SetAccessRights (std::string email, int newAccess)
- std::string toJSON ()
- void RemoveUser (std::string email)
- std::shared_ptr< User > GetUserBySessid (std::string x)
  Returns a shared ptr to the User associated with the session id if it exists.
- std::shared_ptr< User > GetUserByName (std::string email)
  Returns a shared pointer to the user associated with the given email returns a nullptr otherwise.
- void RemoveSession (std::string x)
  Removes the session by the given id.

Static Public Member Functions

- static UserHandler & GetUserTable ()
  Returns the global user table used to manage all users in the server.

4.25.1 Detailed Description

The user handler got a list of all available users and manages creating and deleting new users.

4.25.2 Constructor & Destructor Documentation

4.25.2.1 UserHandler()

```
UserHandler::UserHandler (
            std::string filePath )
```

Loads the user table from the specified path and initialises it.

Parameters

filePath	The path to the saved user table

4.25.3 Member Function Documentation

4.25.3.1 AddUser()

```
bool UserHandler::AddUser (
            std::string email,
            std::string password,
            int access )
```

Adds a user to the map of current users.

Parameters

email	The email with which the user gets created
password	The password the user accounts has got
access	The access rights the user has

4.25.3.2 DoLogin()

```
std::string UserHandler::DoLogin (
            std::string email,
            std::string password )
```

Checks if a given set of email and password matches an existing user and returns the user if the password and login matches.

Parameters

email	The email address of the user.
password	The Password of the user

Returns

returns either the user if the password/email matches an user or zero if there is no user with this password and/or email.

4.25.3.3 GetUserByName()

```
std::shared_ptr< User > UserHandler::GetUserByName (
            std::string email )
```

Returns a shared pointer to the user associated with the given email returns a nullptr otherwise.

Parameters

email	The email of the user which should be found

Returns

A shared pointer to the user associated with the given email

4.25.3.4 GetUserBySessid()

```
std::shared_ptr< User > UserHandler::GetUserBySessid (
            std::string x )
```

Returns a shared ptr to the User associated with the session id if it exists.

Parameters

x	The session id

Returns

The user associated with the session id

4.25.3.5 GetUserTable()

```
static UserHandler& UserHandler::GetUserTable ( )  [inline], [static]
```

Returns the global user table used to manage all users in the server.

Returns

A reference to the global user table

4.25.3.6 RemoveSession()

```
void UserHandler::RemoveSession (
            std::string x )
```

Removes the session by the given id.

Parameters

x	The session id to remove

Returns

4.25.3.7 RemoveUser()

```
void UserHandler::RemoveUser (
            std::string email )
```

Removes a user from the user table and deletes all files and all folder associated with him.

Parameters

email	The email from the user who is about to be removed from the map of users

4.25.3.8 SetAccessRights()

```
void UserHandler::SetAccessRights (
            std::string email,
            int newAccess )
```

Set the access rights for a specific user and save the changes instantly to disk.

Parameters

email	The email of the user who gets the access rights changed
newAccess	The new access rights the user gets granted

4.25.3.9 toJSON()

```
std::string UserHandler::toJSON ( )
```

Converts the complete user table to a string formatted in json style. The json contains only email name and access rights.

Returns

The string containing the UserTable in json format

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

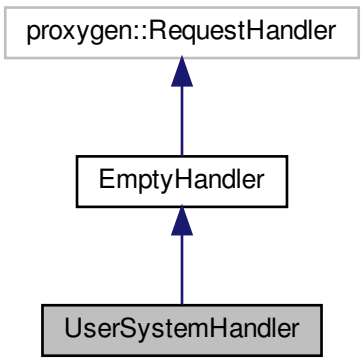
- src/login/user_system.hpp
- src/login/usertable.cpp

4.26 UserSystemHandler Class Reference

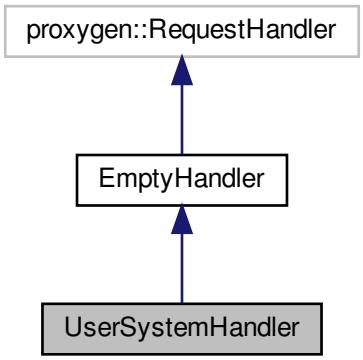
Handles all read accesses to the user system.

```
#include <URIObjects.hpp>
```

Inheritance diagram for UserSystemHandler:



Collaboration diagram for UserSystemHandler:



Public Member Functions

- void onRequest (std::unique_ptr< proxygen::HTTPMessage > headers) noexcept override
  Sends back specific informations regarding the user profile and profile changes.

Additional Inherited Members

4.26.1 Detailed Description

Handles all read accesses to the user system.

4.26.2 Member Function Documentation

4.26.2.1 onRequest()

```
void UserSystemHandler::onRequest (
            std::unique_ptr< proxygen::HTTPMessage > headers )  [override], [virtual], [noexcept]
```

Sends back specific informations regarding the user profile and profile changes.

Parameters

headers	The http request provided by the proxygen library

Reimplemented from EmptyHandler.

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

- src/server/URIObjects.hpp
- src/server/URIObjects.cpp

4.27 Zotero Class Reference

The zotero class connects the server to the zotero api and requests metadata from the server to keep the metadata up to date.

`#include <zotero.hpp>`

Classes

- struct Request

  Defines the most Basic requests to the zotero API.

Public Member Functions

- ~Zotero ()

Static Public Member Functions

- static std::string SendRequest (std::string requestURI)

  This is just a wrapper for the _sendRequest function.
- static const nlohmann::json & GetPillars ()

Friends

- size_t zoteroHeaderReader (char *, size_t, size_t, void *)

  Needed as curl callback on header receiving.
- size_t zoteroReadBuffer (void *, size_t, size_t, void *)

  Needed as curl callback on data receiving.

4.27.1 Detailed Description

The zotero class connects the server to the zotero api and requests metadata from the server to keep the metadata up to date.

4.27.2 Constructor & Destructor Documentation

4.27.2.1 ~Zotero()

```
Zotero::~Zotero ( )
```

Closes all open connection and cleans everything up

4.27.3 Member Function Documentation

4.27.3.1 GetPillars()

```
const nlohmann::json & Zotero::GetPillars ( )  [static]
```

Returns a json with all the pillars used by the current zotero implementation

Returns

The pillars of zotero

4.27.3.2 SendRequest()

```
std::string Zotero::SendRequest (
            std::string requestURI )  [static]
```

This is just a wrapper for the _sendRequest function.

Parameters

requestURI	The zotero items uri which is about to be received

Returns

The json for the request or an empty string on error

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

- src/zotero/zotero.hpp
- src/zotero/zotero.cpp

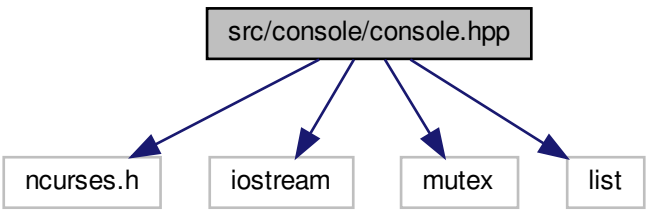
Chapter 5

File Documentation

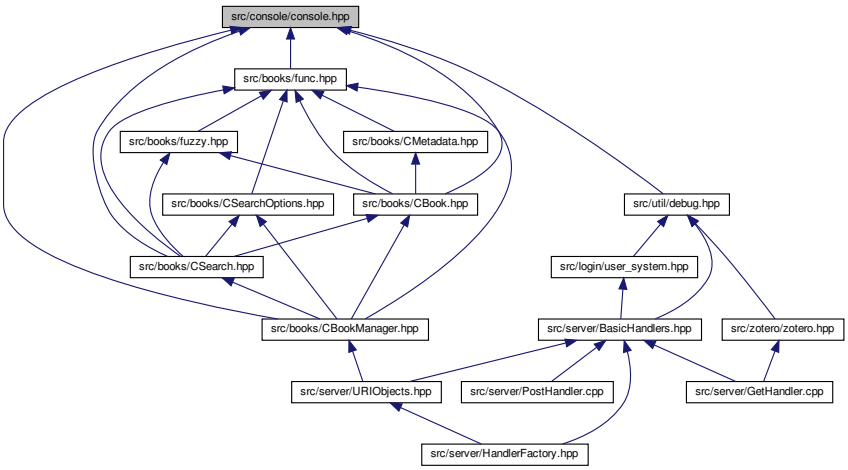
5.1 src/console/console.hpp File Reference

```
#include <ncurses.h>
#include <iostream>
#include <mutex>
#include <list>
```

Include dependency graph for console.hpp:



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



Classes

- class alx::console

  Basic console class, creates a USER interface in the terminal based on the ncurses library.

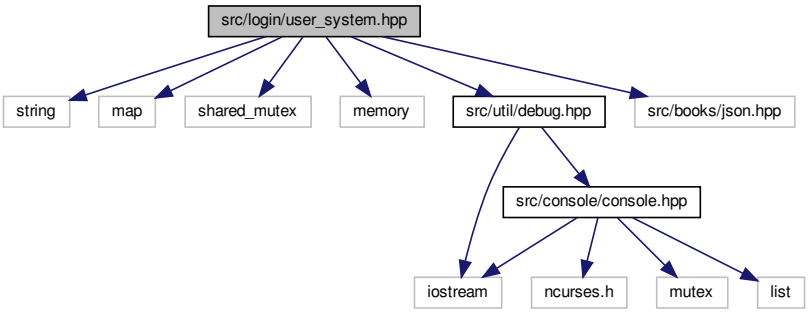
5.1.1 Detailed Description

Basic console file defines interfaces for outputting to the terminal and reading user input from the terminal

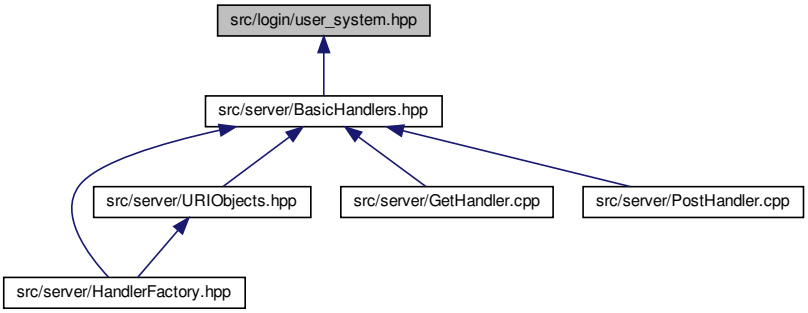
5.2 src/login/user_system.hpp File Reference

```
#include <string>
#include <map>
#include <shared_mutex>
#include <memory>
#include "src/util/debug.hpp"
#include "src/books/json.hpp"
```

Include dependency graph for user_system.hpp:



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



Classes

- class User

  The basic user class this represents a basic user and stores email password and access rights for this user as well as the current session.
- class UserHandler

  The user handler got a list of all available users and manages creating and deleting new users.

Enumerations

- enum AccessRights { USR_READ = 1, USR_WRITE = 2, USR_ADMIN = 4 }

  Defines the basic a access rights a user can have at the moment.

5.2.1 Detailed Description

This file defines the interface for the basic user class

5.2.2 Enumeration Type Documentation

5.2.2.1 AccessRights

```
enum AccessRights
```

Defines the basic a access rights a user can have at the moment.

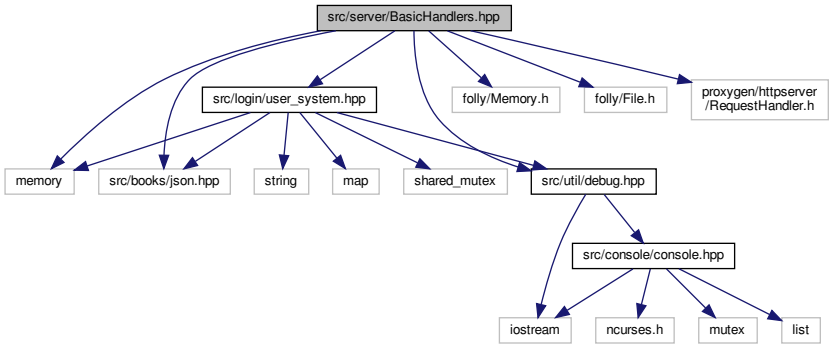
Enumerator

USR_READ	The user has read access means, he can access all books for reading.
USR_WRITE	The user has write access he upload new books and change existing ones.
USR_ADMIN	The user is an admin he can create new users and give all users new rights, he can delete users as well.

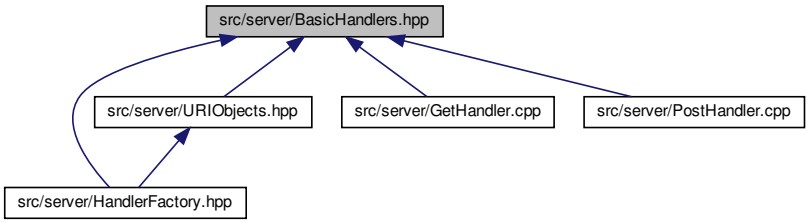
5.3 src/server/BasicHandlers.hpp File Reference

```
#include <memory>
#include <folly/Memory.h>
#include <folly/File.h>
#include <proxygen/httpserver/RequestHandler.h>
#include "src/books/json.hpp"
#include "src/login/user_system.hpp"
#include "src/util/debug.hpp"
```

Include dependency graph for BasicHandlers.hpp:



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



Classes

- class EmptyHandler

  Small class used for setting the default empty method for every request handler.
- class GetHandler

  The Basic Get Handler which does almost all of the server disk IO acesses.
- class URIFile

  The class contains basic information about the URI file.
- class PostHandler

  Handles the basic posts to the server mainly does the login and not much else.

Functions

- void SendErrorNotFound (proxygen::ResponseHandler *rsp, std::string message=" <h1>Not found!</h1> ")

  Sends an 404 not found message to the client with the given message.
- void SendAccessDenied (proxygen::ResponseHandler *rsp, std::string message=" <h1>Access denied</h1> ")

  Sends an 401 access denied message to the client with the given message.
- void ReloadAllFiles ()

  Reloads all cached files in the gethandler do not use this on active server!

5.3.1 Detailed Description

Defines the interface to the Basic Get Handler which does a lot of the servers disk IO

5.3.2 Function Documentation

5.3.2.1 SendAccessDenied()

```
void SendAccessDenied (
            proxygen::ResponseHandler * rsp,
            std::string message = "<center><h1>Access denied</h1></center>" )
```

Sends an 401 access denied message to the client with the given message.

Parameters

rsp	The downstream_ Response Builder ever RequestHandler has got
message	The message to set the body to, the format of the body will always be html `SendAccessDenied(downstream_); //Can be used like this in every handler inheriting from proxygen::RequestHandler or EmptyHandler SendAccessDenied(downstream_, "<h1>My special error</h1>"); //Or specify a string to send a specific error message back`

5.3.2.2 SendErrorNotFound()

```
void SendErrorNotFound (
            proxygen::ResponseHandler * rsp,
            std::string message = "<center><h1>Not found!</h1></center>" )
```

Sends an 404 not found message to the client with the given message.

Parameters

rsp	The downstream_ Response Builder ever RequestHandler has got

Parameters

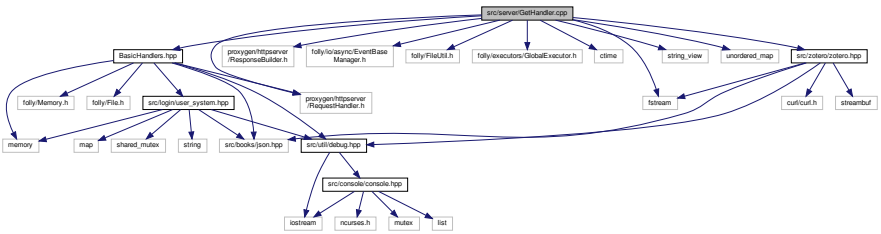
message	The message to set the body to, the format of the body will always be html `SendErrorNotFound(downstream_); //Can be used like this in every handler inheriting from proxygen::RequestHandler or EmptyHandler SendErrorNotFound(downstream_, "<h1>My special error</h1>"); //Or specify a string to send a specific error message back`

5.4 src/server/GetHandler.cpp File Reference

Implements the interface of the GetHandler class and the interface of the URIFile class.

```
#include "BasicHandlers.hpp"
#include <proxygen/httpserver/RequestHandler.h>
#include <proxygen/httpserver/ResponseBuilder.h>
#include <folly/io/async/EventBaseManager.h>
#include <folly/FileUtil.h>
#include <folly/executors/GlobalExecutor.h>
#include <ctime>
#include <fstream>
#include <string_view>
#include <unordered_map>
#include "src/zotero/zotero.hpp"
```

Include dependency graph for GetHandler.cpp:



Functions

- void ReloadAllFiles ()

  Reloads all cached files in the gethandler do not use this on active server!
- void SendErrorNotFound (proxygen::ResponseHandler ∗rsp, std::string message)

  Sends an 404 not found message to the client with the given message.
- void SendAccessDenied (proxygen::ResponseHandler ∗rsp, std::string message)

  Sends an 401 access denied message to the client with the given message.

Variables

- std::unordered_map< std::string, URIFile > fileAccess

  The map which caches all file the get handler will return and also saves the access rights to acces these files.

5.4.1 Detailed Description

Implements the interface of the GetHandler class and the interface of the URIFile class.

Implements the interface of the gethandler class and the default response functions also hosts the file Map which mappes almost all files to a specific URI with the given access rights so one can do access checks on files

5.4.2 Function Documentation

5.4.2.1 SendAccessDenied()

```
void SendAccessDenied (
            proxygen::ResponseHandler * rsp,
            std::string message = "<center><h1>Access denied</h1></center>" )
```

Sends an 401 access denied message to the client with the given message.

Parameters

rsp	The downstream_ Response Builder ever RequestHandler has got
message	The message to set the body to, the format of the body will always be html `SendAccessDenied(downstream_); //Can be used like this in every handler inheriting from proxygen::RequestHandler or EmptyHandler SendAccessDenied(downstream_, "<h1>My special error</h1>"); //Or specify a string to send a specific error message back`

5.4.2.2 SendErrorNotFound()

```
void SendErrorNotFound (
            proxygen::ResponseHandler * rsp,
            std::string message = "<center><h1>Not found!</h1></center>" )
```

Sends an 404 not found message to the client with the given message.

Parameters

rsp	The downstream_ Response Builder ever RequestHandler has got
message	The message to set the body to, the format of the body will always be html `SendErrorNotFound(downstream_); //Can be used like this in every handler inheriting from proxygen::RequestHandler or EmptyHandler SendErrorNotFound(downstream_, "<h1>My special error</h1>"); //Or specify a string to send a specific error message back`

5.4.3 Variable Documentation

5.4.3.1 fileAccess

```
std::unordered_map<std::string,URIFile> fileAccess
```

Initial value:

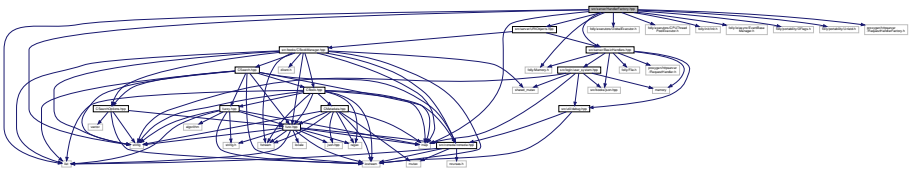
```
{
    {"/",URIFile("web/guest_index.html",0)},
    {"/favicon.ico",URIFile("web/favicon.png",0)},
    {"/home",URIFile("web/index.html",1)},
    {"/search",URIFile("web/Search.html",1)},
    {"/administration",URIFile("web/Administration.html",4)},
    {"/uploadbook",URIFile("web/UploadBook.html",2)},
    {"/managebooks",URIFile("web/ManageBooks.html",2)},
    {"/GetBooks",URIFile("web/GetBooks.html",1)},
    {"/ShowMetadata", URIFile("web/ShowMetadata.html",1)},
    {"/scan.png",URIFile("web/scan.png",1)},
    {"/404.jpeg",URIFile("web/404.jpeg",0)},
    {"/volltext.png",URIFile("web/volltext.png",1)},
    {"/jszip.js",URIFile("web/jszip.js",2)}
}
```

The map which caches all file the get handler will return and also saves the access rights to acces these files.

5.5 src/server/HandlerFactory.hpp File Reference

```
#include <folly/Memory.h>
#include <folly/executors/GlobalExecutor.h>
#include <folly/executors/CPUThreadPoolExecutor.h>
#include <folly/init/Init.h>
#include <folly/io/async/EventBaseManager.h>
#include <folly/portability/GFlags.h>
#include <folly/portability/Unistd.h>
#include <proxygen/httpserver/RequestHandlerFactory.h>
#include <list>
#include <string>
#include <map>
#include "src/server/BasicHandlers.hpp"
#include "src/server/URIObjects.hpp"
```

Include dependency graph for HandlerFactory.hpp:



Classes

- class HandlerFactory

  The HandlerFactory is used to instantiate the proxygen server and creates all request handler for every request directed to the server This class redirects every request to the right handler, in order to do so it keeps book of every URI object registered and sends the request to the URI object if there is any else the request goes to the default handler.

Functions

- template<typename T >
  EmptyHandler * CreateHandler ()

  Defines a template function which creates a new instance of the given type is used to easily create multiple functions which create the new Request handlers.

5.5.1 Detailed Description

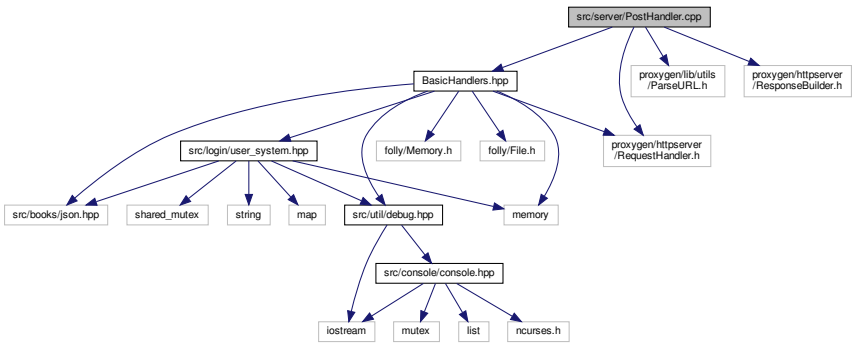
This file contains the basic static handler factory used to instantiate the proxygen server The Handler Factory selects based on the request the appropriate request handler class

5.6 src/server/PostHandler.cpp File Reference

Implements the interface for the PostHandler class and handles all user logins.

```
#include "BasicHandlers.hpp"
#include <proxygen/lib/utils/ParseURL.h>
#include <proxygen/httpserver/RequestHandler.h>
#include <proxygen/httpserver/ResponseBuilder.h>
```

Include dependency graph for PostHandler.cpp:



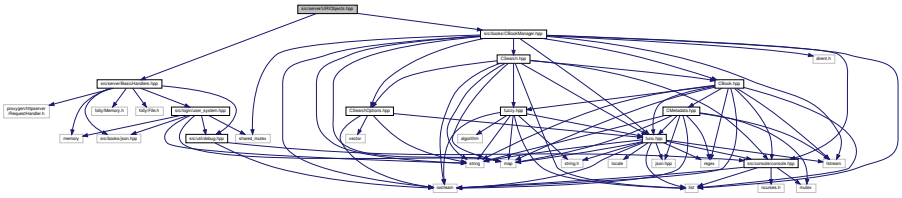
5.6.1 Detailed Description

Implements the interface for the PostHandler class and handles all user logins.

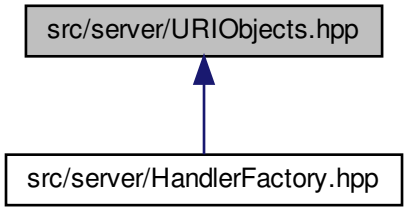
5.7 src/server/URIObjects.hpp File Reference

```
#include "src/server/BasicHandlers.hpp"
#include "src/books/CBookManager.hpp"
```

Include dependency graph for URIObjects.hpp:



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



Classes

- class UserSystemHandler

  Handles all read accesses to the user system.
- class UpdateUserSystemHandler

  Handles all changes to the user table like create delete and change access.
- class GetBookRessource

  Returns either all the books in the server or all the files in one book or a specific ressource from a specific book.
- class GetSearchHandler

  Handles the general search in all books.
- class GetSearchInBookHandler

  Searches a specific book for a specific word with the given fuzzyness.
- class RedirectToHTTPSHandler

  Redirects all request made to http port to the https website.
- class GetBookMetadata
- class UploadBookHandler
- class StartSearch
- class RequestSearchProgress

5.7.1 Detailed Description

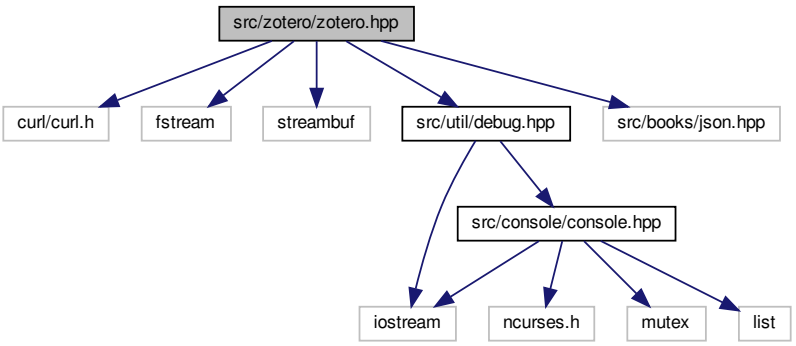
Defines the basic URI Objects /search and /getprofileinfo etc.

5.8 src/zotero/zotero.hpp File Reference

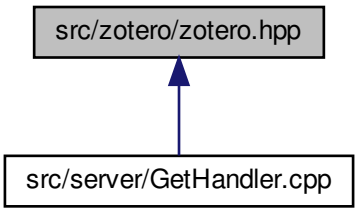
Contains the zotero interface, with which the server communicates with the zotero server.

```
#include <curl/curl.h>
#include <fstream>
#include <streambuf>
#include "src/util/debug.hpp"
#include "src/books/json.hpp"
```

Include dependency graph for zotero.hpp:



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



Classes

- class Zotero

  The zotero class connects the server to the zotero api and requests metadata from the server to keep the metadata up to date.
- struct Zotero::Request

  Defines the most Basic requests to the zotero API.

Variables

- constexpr const char ZOTERO_API_ADDR [ ] = "https://api.zotero.org"

  The zotero api server address where to send all requests to.
- constexpr const char ZOTERO_API_KEY_FILE_PATH [ ] = "bin/zoteroKey.json"

  The file path at which the access key and the group number can be found.

5.8.1 Detailed Description

Contains the zotero interface, with which the server communicates with the zotero server.