Beleg 39467

1.0

Erzeugt von Doxygen 1.8.11

Inhaltsverzeichnis

Kapitel 1

Modul-Verzeichnis

1.1 Module

Hier folgt die Aufzählung aller Module:

Kapitel 2

Klassen-Verzeichnis

2.1 Auflistung der Klassen

Hier folgt die Aufzählung aller Klassen, Strukturen, Varianten und Schnittstellen mit einer Kurzbeschreibung:

Kapitel 3

Modul-Dokumentation

3.1 LendLib

Hauptmodul der Ausleihbibliothek.

Funktionen

- int main (int argc, char ∗argv[ ])
  Die Main-Funktion.

3.1.1 Ausführliche Beschreibung

Hauptmodul der Ausleihbibliothek.

3.1.2 Dokumentation der Funktionen

3.1.2.1 int main (int *argc,* char ∗ *argv[ ]*)

Die Main-Funktion.

- Ausgabevariablen vorbereiten
- Kopfzeilen ausgeben
- Einträge aus Datei einlesen (falls sie existiert)

CGI-Ausgabe:

Terminalausgabe:

- aus Datei erhaltene Medien anzeigen
- Nutzerinput verarbeiten

beide:

- Daten in Datei speichern
- Speicher freigeben
- Fußzeilen ausgeben

3.2 LendLibIn

Funktionen für jegliche Art von Ausgabe.

Makrodefinitionen

- #define **MALLOCERR** "Fehler bei der Speicherzuweisung"

Funktionen

- void readfile ()

  Liest Verleihmediendaten aus Datei aus.
- int getSize (FILE ∗libdb)

  Gibt Größe der gespeicherten CSV-Datenbank aus.
- void getInput ()

  Verarbeitet Nutzerinput im Terminal.
- void getPost ()

  Verarbeitet POSTs des CGI.
- char ∗ strtok2 (char ∗str, char const ∗delims)

  Eigenes strtok, welches auch leere Einträge ausgibt.

Variablen

- FILE ∗ **libdb**
- char **vbuf** [128]
- const char ∗ **filename**
- char **posted** [256]
- char **pTemp1** [256]
- char **pTemp2** [256]
- char **pTemp3** [256]

3.2.1 Ausführliche Beschreibung

Funktionen für jegliche Art von Ausgabe.

3.2.2 Dokumentation der Funktionen

3.2.2.1 void getInput ( )

Verarbeitet Nutzerinput im Terminal.

Hilfezeilen mit Anweisungen anzeigen

Dateien noch einmal auflisten

Ausleihmedium hinzufügen

- Eingabe Medienart
- (Festlegung der Medienart durch entsprechende Zahl aus mType oder Anfangsbuchstabe: Buch, Cd, Dvd)
- (Titel, Interpret/Autor und Ausleihender eingeben lassen) und jeweils in freigegebenen Speicher schreiben
- Eingegebene Daten als Medium speicher und in Liste einfügen

Medien sortieren, dazu nach Sortierkategorie fragen

Medium löschen, dazu ID abfragen

Medien finden, dazu nach Suchkategorie (Titel oder Ausleihender) und der zu suchenden Zeichenkette fragen

Ansonsten Programm beenden

3.2.2.2 void getPost ( )

Verarbeitet POSTs des CGI.

Der POST wird jeweils mit strtok2 auseinandergeschnitten und verarbeitet. Die vorgehensweise ist ähnlich wie bei der Terminaleingabe. POST einlesen

POSTs unterscheiden sich durch den Anfangsbuchstaben (wie bei Terminaleingabe)

- Sucheingabe verarbeiten
- Etwas finden
- Medium einfügen
- (Funktioniert wie bei Terminaleingabe: Es wird jeweils Speicher bereitgestellt und Angaben darin geschrieben.
- Löschen

3.2.2.3 int getSize (FILE * *libdb*)

Gibt Größe der gespeicherten CSV-Datenbank aus.

Parameter

libdb	Eine Datei

Rückgabe

Anzahl von Linien (also Einträgen)

Quelle: http://stackoverflow.com/a/1910795

- Anzahl der Zeilenumbrüche zählen
- Im Fall, dass die letzte Zeile keinen Umbruch enthält oder leer ist (bis auf den Umbruch), wird diese dazu gezählt oder eben nicht

3.2.2.4 void readfile ( )

Liest Verleihmediendaten aus Datei aus.

Je nach Terminal- oder CGI-Anwendung ist der Dateipfad anders

- Datei öffnen
- Datei Zeilenweise durchgehen und als Medium in Liste einfügen

3.2.2.5 char * strtok2 (char * *str*, char const * *delims*)

Eigenes strtok, welches auch leere Einträge ausgibt.

Parameter

str	Der String
delims	Die Trennzeichen Quelle: http://stackoverflow.com/a/8706031

3.3 LendLibItem

Funktionen und Deklarationen rund um ausgeliehene Medien.

Klassen

- struct medium

  Das ausgeliehene Medium.
- struct LITEM

  Eintrag in der Liste der verliehenen Medien.
- struct theLib

  Verwaltet die Liste der verliehenen Medien.

Typdefinitionen

- typedef struct LITEM lItem

  Eintrag in der Liste der verliehenen Medien.

Aufzählungen

- enum mType { other, book, cd, dvd }

  Typ des ausgeliehenen Mediums.
- enum sBy { title, lendee, id }

  Möglichkeiten, nach denen Sortiert/gesucht werden kann.

Funktionen

- medium ∗ createItem (int ntype, char ∗ntitle, char ∗nauthor, char ∗nlendee)

  Erstellt einen neuen Medieneintrag.
- medium ∗ createItemF (FILE ∗libitem)

  Erstellt neuen Medieneintrag aus Dateizeile.
- void insertItem (medium ∗nMedium, theLib ∗inLib)

  Fügt Medium in Liste ein.
- char ∗ getmType (int type)

  Gibt String für enum mType aus.
- void deleteItem (int theID, theLib ∗inLib)

  Löscht Medium aus der Liste.
- void sortItems (sBy sortBy, theLib ∗inLib)

  Sortiert die Liste nach Mediumtitel/Leihendem.
- void findItem (char ∗sItem, sBy findBy, theLib ∗inLib)

  Suche, bzw. finde ein ausgeliehenes Medium.
- void freeAll (theLib ∗inLib)

  Den gesamten Speicher freigeben.
- int strcicmp (char const ∗a, char const ∗b)

  Vergleicht zwei "Strings" unabhängig von Groß- oder Kleinbuchstaben Quelle: http://stackoverflow.↩com/a/5820991.
- void **initLib** ()

Variablen

- theLib ∗ **currLib**
- theLib **myLib**
- lItem ∗ **myItems**
- medium ∗ **myMedia**
- int **myItemsCount**
- int **myMediaCount**
- unsigned int **idc**

3.3.1 Ausführliche Beschreibung

Funktionen und Deklarationen rund um ausgeliehene Medien.

3.3.2 Dokumentation der Aufzählungstypen

3.3.2.1 enum mType

Typ des ausgeliehenen Mediums.

Aufzählungswerte

other Medium ohne Kategorie.

book Bücher.

cd CDs.

dvd DVDs.

3.3.2.2 enum sBy

Möglichkeiten, nach denen Sortiert/gesucht werden kann.

Aufzählungswerte

title Nach Medientitel.

lendee Nach Leihendem.

id Nach ID.

3.3.3 Dokumentation der Funktionen

3.3.3.1 medium ∗ createItem (int *ntype,* char ∗ *ntitle,* char ∗ *nauthor,* char ∗ *nlendee*)

Erstellt einen neuen Medieneintrag.

Parameter

ntype	Typ des Mediums
ntitle	Titel des Mediums
nauthor	Autor/Interpret des Mediums
nlendee	Person, dem das Medium ausgeliehen wurde

Rückgabe

Gibt das erstellte medium zurück

- ID festlegen, dazu wird die Liste nach IDs sortiert und nach der ersten "Zahlenlücke" gesucht
- Type festlegen
- Titel festlegen
- Interpret/Autor festlegen
- Leihendem festlegen

3.3.3.2 medium * createItemF (FILE * *libitem*)

Erstellt neuen Medieneintrag aus Dateizeile.

Parameter

libitem	Die Datei

Rückgabe

Gibt das erstellte medium zurück

- ID festlegen
- Typ festlegen
- Titel festlegen
- Interpret/Autor festlegen
- Leihendem festlegen

3.3.3.3 void deleteItem (int *theID,* theLib * *inLib*)

Löscht Medium aus der Liste.

Parameter

theID	Das zu löschende Medium (ID)
inLib	in dieser Liste

- Medium mit ID finden
- Medium aus Liste löschen indem Zeiger der Nachbarn neu gesetzt werden und anschließend der Speicher freigegeben wird.

3.3.3.4 void findItem (char * *sItem,* sBy *findBy,* theLib * *inLib*)

Suche, bzw. finde ein ausgeliehenes Medium.

Parameter

sItem	Nach dieser Zeichenkette soll gesucht werden
findBy	Die Zeichenkette soll in diesem Eintrag des ausgeliehen Mediums gesucht werden (sBy)
inLib	in dieser Liste

Rückgabe

Gibt das gefundene Medium zurück

- Durch myLib theLib gehen und alle gefunden in eine neue, temporäre theLib Liste einfügen, die dann ausgegeben wird.

3.3.3.5 void freeAll (theLib * *inLib*)

Den gesamten Speicher freigeben.

Parameter

inLib	in dieser Liste

3.3.3.6 char * getmType (int *type*)

Gibt String für enum mType aus.

Parameter

type	Zahl von mType

Rückgabe

mType als String

3.3.3.7 void insertItem (medium * *nMedium,* theLib * *inLib*)

Fügt Medium in Liste ein.

Parameter

iMedium	Das einzufügende Medium

- Listenobjekt erstellen und Inhalt zuweisen

Je nach Sortierung der Liste, muss anders eingefügt werden, falls nach Titel sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

falls nach Ausleihendem sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

Falls nach ID sortiert:

- Falls Liste noch leer, diese initialisieren
- Falls Liste einelementig ist, das nächste Element einfach einfügen
- Alle anderen Elemente sortiert in Liste einfügen
- - Nach einem Element suchen, welches größer ist, damit man das einzufügende Element vor diesem einfügen kann.
- - Wenn größeres vorm Listenende gefunden wurde, Element davor einfügen
- - Ansonsten Element am Listenende je nach größe davor oder danach einfügen.

3.3.3.8 void sortItems (sBy *sortBy,* theLib * *inLib*)

Sortiert die Liste nach Mediumtitel/Leihendem.

Parameter

sortBy	Nach was sortiert werden soll (sBy)
inLib	in dieser Liste

Zum Sortieren wird eine medium Hilfsliste erstellt, in die alle Medien aus der theLib Originalliste kopiert werden, um sie dann wieder in die Originalliste einzufügen(dabei sortiert ::insertItem schon).

3.4 LendLibOut

Funktionen für jegliche Art von Ausgabe.

Aufzählungen

- enum ptype {
  in, out, error, status,
  extra }

  Verschieden Ausgabemöglichkeiten.

Funktionen

- void prepareOut ()

  Bereitet Variablen für den Output vor.
- void printItems ()

  Gibt die Liste der ausgeliehenen Medien aus.
- void libprint (ptype type, const char ∗printable,...)

  Terminalausgabe abhängig von (beim kompilieren) definierten Paramtern.
- void setColor (int pType)

  Setzt die Terminal-Schriftfarbe abhängig vom Typ.
- void resetColor ()

  Setzt die Terminal-Schriftfarbe zurück auf den Standard.
- void printHead ()

  Gibt einen Header aus.
- void printHTMLInter ()

  Gibt die HTML-Bedienelemente aus.
- void printFoot ()

  Gibt einen Footer aus.
- void printTLine (char type, int length)

  Gibt eine Linie im Terminal aus.
- void saveDBtoFile ()

  Speichert Liste in Datei.
- void sleep_ms (int milliseconds)

  Cross-Platform Sleep Funktion in Millisekunden.

3.4.1 Ausführliche Beschreibung

Funktionen für jegliche Art von Ausgabe.

Ausgabe abhängig von den übers kompilieren definierten Parameter:

Parameter

CGI	für eine .cgi-Datei
DCOLOR	für eingefärbte Terminalausgaben

3.4.2 Dokumentation der Aufzählungstypen

3.4.2.1 enum ptype

Verschieden Ausgabemöglichkeiten.

Aufzählungswerte

in folgende Eingabe

out normale Ausgabe

error Fehlermeldungen.

status Statusmeldungen.

extra Fehlerbehebung.

3.4.3 Dokumentation der Funktionen

3.4.3.1 void libprint (ptype *type*, const char * *printable*, ...)

Terminalausgabe abhängig von (beim kompilieren) definierten Paramtern.

Parameter

type	Ausgabetyp (ptype)

Quelle: http://www.ozzu.com/cpp-tutorials/tutorial-writing-custom-printf-wrapper-function-t
html

- Im Falle CGI werden die Ausgaben in verschieden Formatierte divs gefüllt
- Im Terminal wird je nach definierten Strings etwas mit oder Farbe ausgegeben - oder nicht

Falls Fehlerausgabe deaktiviert ist, keine Fehler ausgeben!

3.4.3.2 void prepareOut ( )

Bereitet Variablen für den Output vor.

- Berechnung der Terminalbreite (Quelle: http://stackoverflow.com/a/1022961)

3.4.3.3 void printFoot ( )

Gibt einen Footer aus.

Im Terminal ein Abschlussstrich, im CGI den HTML-Footer

3.4.3.4 void printHead ( )

Gibt einen Header aus.

Im Terminal ein paar erste Zeilen, im CGI den HTML-Header

3.4.3.5 void printItems ( )

Gibt die Liste der ausgeliehenen Medien aus.

Im CGI durch eine Tabelle (in der auch schon Bedienelemente fürs löschen, sortieren usw. eingeschlossen sind)

Im Terminal in einer Tabellenimitation

3.4.3.6 void printTLine (char *type,* int *length*)

Gibt eine Linie im Terminal aus.

Parameter

type	Ein char, der die Linie visuell bestimmt
length	Die Länge der Linie, bei der Länge 0 wird die Breite des Terminals

3.4.3.7 void resetColor ( )

Setzt die Terminal-Schriftfarbe zurück auf den Standard.

Nur, falls beim Compilieren DCOLOR definiert wurde.

3.4.3.8 void setColor (int *pType*)

Setzt die Terminal-Schriftfarbe abhängig vom Typ.

Parameter

pType	Ausgabetyp (ptype)

Nur, falls beim Compilieren DCOLOR definiert wurde.

3.4.3.9 void sleep_ms (int *milliseconds*)

Cross-Platform Sleep Funktion in Millisekunden.

Quelle: http://stackoverflow.com/a/28827188

Kapitel 4

Klassen-Dokumentation

4.1 LITEM Strukturreferenz

Eintrag in der Liste der verliehenen Medien.

`#include <lendlibitem.h>`

Öffentliche Attribute

- struct LITEM * next

  Zeigt auf vorhergehenden Eintrag.
- struct LITEM * prev

  Zeigt auf folgenden Eintrag.
- medium * item

  Inhalt.

4.1.1 Ausführliche Beschreibung

Eintrag in der Liste der verliehenen Medien.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- htw/Programmierung 1/Beleg/lendlibitem.h

4.2 medium Strukturreferenz

Das ausgeliehene Medium.

`#include <lendlibitem.h>`

Öffentliche Attribute

- int **id**
- mType type

  Typ des ausgeliehen Mediums.
- char * title

  Titel des Mediums.
- char * author

  Autor bzw. Interpret des Mediums.
- char * lendee

  Name der Person, dem das Medium verliehen wurde.

4.2.1 Ausführliche Beschreibung

Das ausgeliehene Medium.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- htw/Programmierung 1/Beleg/lendlibitem.h

4.3 theLib Strukturreferenz

Verwaltet die Liste der verliehenen Medien.

```
#include <lendlibitem.h>
```

Öffentliche Attribute

- lItem * first

  Erster Eintrag.
- lItem * curr

  aktueller Eintrag
- unsigned int size

  größe der Liste
- unsigned int sort

  Liste ist sortiert nach 0: Titel, 1:Ausleihenden.

4.3.1 Ausführliche Beschreibung

Verwaltet die Liste der verliehenen Medien.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

- htw/Programmierung 1/Beleg/lendlibitem.h