dataManagement

Interface IDataBase

All Known Implementing Classes:

MySqlDatabase

```
public interface IDataBase
```

Dieses Interface stellt die Schnittstelle zur Datenbank her. Alle Methoden verwenden keinerlei Logik, sondern übersetzen lediglich eine elementare Programmanweisung in die Sprache des DMBS und leiten diese an das DMBS zur Ausführung weiter. Die Überprüfung der korrekten Arbeitsweise dieser Methoden ist somit nicht Aufgabe der Klassen die dieses Interface implementieren, sondern muss von den verwendenden Klassen geleistet werden. Dieses Interface führt somit die nötige Abstraktion ein um verschiedene Datenbanken zu verwenden oder den Umstieg auf eine andere Datenbank zu erleichtern.

Method Summary

Methods

Modifier and Type	Method and Description
void	**createBook**(java.lang.String bookName) Diese Methode legt ein neues Buch in der Tabelle "Books" an
void	**createChapter**(java.lang.String chapterName, int bookID) Diese Methode legt ein neues Kapitel in der Tabelle "Chapter" an
void	**createContent**(byte[] contentBytes, int contentNumber, int pageID) Diese Methode legt einen neuen Seiteninhalt in der Tabelle "Content" an
void	**createPage**(**PageInformation** pageInfo, int chapterID) Diese Methode legt eine neue Seite in der Tabelle "Pages" an
void	**deleteBook**(int bookID) Diese Methode löscht ein Buch aus der Tabelle "Books"
void	**deleteChapter**(int chapterID) Diese Methode löscht ein Kapitel aus der Tabelle "Chapter"
void	**deleteContent**(int contentID) Diese Methode löscht einen Seiteninhalt aus der Tabelle "Content"
void	**deletePage**(int pageID) Diese Methode löscht eine Seite aus der Tabelle "Pages"
boolean	**existsBook**(java.lang.String bookName) Diese Methode gibt zurück, ob es in der Tabelle "Books" ein Buch mit dem Namen "bookName" gibt

boolean	**existsChapter**(java.lang.String chapterName, int bookID) Diese Methode gibt zurück, ob es in der Tabelle "Chapter" ein Kapitel mit dem Namen "chapterName" gibt
boolean	**existsContent**(int contentNumber, int pageID) Diese Methode gibt zurück, ob es in der Tabelle "Content" ein Eintrag mit der Nummer "contentNumber" gibt
boolean	**existsPage**(java.lang.String pageName, int chapterID) Diese Methode gibt zurück, ob es in der Tabelle "Pages" eine Seite mit dem Namen "pageName" gibt
java.lang.String[]	**getAllBooks**() Diese Methode liefert den Namen aller Bücher welche in der Datenbank angelegt sind
java.lang.String[]	**getBookChildren**(int bookID) Diese Methode liefert zur id eines Buches alle Kapitel welche zu diesem Buch gehören
int	**getBookID**(java.lang.String bookName) Diese Methode liefert zu dem Namen eines Buches die entsprechende id in der Datenbank
java.lang.String[]	**getChapterChildren**(int chapterID) Diese Methode liefert zur id eines Kapitels alle Seiten welche zu diesem Kapitel gehören
int	**getChapterID**(java.lang.String chapterName, int bookID) Diese Methode liefert zu dem Namen eines Kapitels die entsprechende id in der Datenbank
int	**getContentID**(int contentNumber, int pageID) Diese Methode liefert zu der Nummer eines Seiteinhalts die entsprechende id in der Datenbank
int[]	**getPageChildren**(int pageID) Diese Methode liefert zur id einer Seite alle Inhalte welche zu dieser Seite gehören
int	**getPageID**(java.lang.String pageName, int chapterID) Diese Methode liefert zu dem Namen einer Seite die entsprechende id in der Datenbank
byte[]	**loadContent**(int contentID) Diese Methode läd einen Seiteninhalt aus der Tabelle "Content"
PageInformation	**loadPageInformation**(int pageID) Diese Methode lädt die Seiteninformation aus der Tabelle "Pages"
void	**saveContent**(byte[] contentBytes, int contentID) Diese Methode speichert einen Seiteninhalt in der Tabelle "Content"
void	**savePage**(**PageInformation** pageInfo, int chapterID, int pageID) Diese Methode speichert eine Seite in der Tabelle "Pages"

Method Detail

createBook

```
void createBook(java.lang.String bookName)
                throws java.sql.SQLException
```

Diese Methode legt ein neues Buch in der Tabelle "Books" an

Parameters:

`bookName` - der Name des anzulegenden Buches

Throws:

`java.sql.SQLException` - Datenbankfehler

createChapter

```
void createChapter(java.lang.String chapterName,
                   int bookID)
                     throws java.sql.SQLException
```

Diese Methode legt ein neues Kapitel in der Tabelle "Chapter" an

Parameters:

`chapterName` - der Name des anzulegenden Kapitels

`bookID` - die id des Buches zu dem das anzulegende Kapitel gehören soll

Throws:

`java.sql.SQLException` - Datenbankfehler

createPage

```
void createPage(PageInformation pageInfo,
                int chapterID)
                  throws java.sql.SQLException
```

Diese Methode legt eine neue Seite in der Tabelle "Pages" an

Parameters:

`pageInfo` - informationen zur Seite

`chapterID` - die id des Kapitels zu dem die anzulegende Seite gehören soll

Throws:

`java.sql.SQLException` - Datenbankfehler

createContent

```
void createContent(byte[] contentBytes,
                   int contentNumber,
                   int pageID)
                     throws java.sql.SQLException
```

Diese Methode legt einen neuen Seiteninhalt in der Tabelle "Content" an

Parameters:

`contentBytes` - das serialisierte Objekt

`contentNumber` - die nummer des Seiteninhalts innerhalb seiner Seite

`pageID` - die id der Seite zu welcher der Inhalt gehören soll

Throws:

`java.sql.SQLException` - Datenbankfehler

deleteBook

```
void deleteBook(int bookID)
                throws java.sql.SQLException
```

Diese Methode löscht ein Buch aus der Tabelle "Books"

Parameters:

`bookID` - Die id des Buches, welches gelöscht werden soll

Throws:

`java.sql.SQLException` - Datenbankfehler

deleteChapter

```
void deleteChapter(int chapterID)
                   throws java.sql.SQLException
```

Diese Methode löscht ein Kapitel aus der Tabelle "Chapter"

Parameters:

`chapterID` - Die id des Kapitels, welches gelöscht werden soll

Throws:

`java.sql.SQLException` - Datenbankfehler

deletePage

```
void deletePage(int pageID)
                throws java.sql.SQLException
```

Diese Methode löscht eine Seite aus der Tabelle "Pages"

Parameters:

`pageID` - Die id der Seite, welche gelöscht werden soll

Throws:

`java.sql.SQLException` - Datenbankfehler

deleteContent

```
void deleteContent(int contentID)
                   throws java.sql.SQLException
```

Diese Methode löscht einen Seiteninhalt aus der Tabelle "Content"

Parameters:

`contentID` - Die id des Seiteninhaltes, welcher gelöscht werden soll

Throws:

`java.sql.SQLException` - Datenbankfehler

saveContent

```
void saveContent(byte[] contentBytes,
                 int contentID)
                   throws java.sql.SQLException
```

Diese Methode speichert einen Seiteninhalt in der Tabelle "Content"

Parameters:

`contentBytes` - das serialisierte Objekt

`contentID` - die innerhalb der Tabelle "Content"

Throws:

`java.sql.SQLException` - Datenbankfehler

savePage

```
void savePage(PageInformation pageInfo,
              int chapterID,
              int pageID)
                throws java.sql.SQLException
```

Diese Methode speichert eine Seite in der Tabelle "Pages"

Parameters:

`pageInfo` - die Information zur Seite

`chapterID` - das Kapitel zu welchem die Seite gehört

`pageID` - die id der Seite innerhalb der Tabelle "Pages"

Throws:

`java.sql.SQLException` - Datenbankfehler

loadContent

```
byte[] loadContent(int contentID)
                   throws java.sql.SQLException
```

Diese Methode läd einen Seiteninhalt aus der Tabelle "Content"

Parameters:

`contentID` - die id des angefragten Seiteninhalts

Returns:

das serialisierte Objekt zur angefragten id

Throws:

`java.sql.SQLException` - Datenbankfehler

loadPageInformation

```
PageInformation loadPageInformation(int pageID)
                                    throws java.sql.SQLException
```

Diese Methode lädt die Seiteninformation aus der Tabelle "Pages"

Parameters:

`pageID` - die id der angefragten Seite

Returns:

die Seiteninformation zur angefragten id

Throws:

`java.sql.SQLException` - Datenbankfehler

existsBook

```
boolean existsBook(java.lang.String bookName)
                   throws java.sql.SQLException
```

Diese Methode gibt zurück, ob es in der Tabelle "Books" ein Buch mit dem Namen "bookName" gibt

Parameters:

`bookName` - der Name des Buchs

Returns:

zum angefragten Buch existiert ein Eintrag in der Datenbank

Throws:

`java.sql.SQLException` - Datenbankfehler

existsChapter

```
boolean existsChapter(java.lang.String chapterName,
                      int bookID)
                        throws java.sql.SQLException
```

Diese Methode gibt zurück, ob es in der Tabelle "Chapter" ein Kapitel mit dem Namen "chapterName" gibt

Parameters:

`chapterName` - der Name des Kapitels

`bookID` - die id des Buches zu welchem das Kapitel gehört

Returns:

zum angefragten Kapitel existiert ein Eintrag in der Datenbank

Throws:

`java.sql.SQLException` - Datenbankfehler

existsPage

```
boolean existsPage(java.lang.String pageName,
                   int chapterID)
                     throws java.sql.SQLException
```

Diese Methode gibt zurück, ob es in der Tabelle "Pages" eine Seite mit dem Namen "pageName" gibt

Parameters:

`pageName` - der Name der Seite

`chapterID` - die id des Kapitels zu welchem die Seite gehört

Returns:

zur angefragten Seite existiert ein Eintrag in der Datenbank

Throws:

`java.sql.SQLException` - Datenbankfehler

existsContent

```
boolean existsContent(int contentNumber,
                      int pageID)
                        throws java.sql.SQLException
```

Diese Methode gibt zurück, ob es in der Tabelle "Content" ein Eintrag mit der Nummer "contentNumber" gibt

Parameters:

`contentNumber` - die Nummer des Seiteninhalts

`pageID` - die id der Seite zu welcher der Inhalt gehört

Returns:

zum angefragten Seiteninhalt existiert ein Eintrag in der Datenbank

Throws:

`java.sql.SQLException` - Datenbankfehler

getBookID

```
int getBookID(java.lang.String bookName)
              throws java.sql.SQLException
```

Diese Methode liefert zu dem Namen eines Buches die entsprechende id in der Datenbank

Parameters:

`bookName` - Der Name des angefragten Buches

Returns:

die id des angefragten Buches

Throws:

`java.sql.SQLException` - Datenbankfehler

getChapterID

```
int getChapterID(java.lang.String chapterName,
                 int bookID)
                   throws java.sql.SQLException
```

Diese Methode liefert zu dem Namen eines Kapitels die entsprechende id in der Datenbank

Parameters:

`chapterName` - Der Name des angefragten Kapitels

`bookID` - die id des Buches zu welchem das Kapitel gehört

Returns:

die id des angefragten Kapitels

Throws:

`java.sql.SQLException` - Datenbankfehler

getPageID

```
int getPageID(java.lang.String pageName,
              int chapterID)
                throws java.sql.SQLException
```

Diese Methode liefert zu dem Namen einer Seite die entsprechende id in der Datenbank

Parameters:

`pageName` - Der Name der angefragten Seite

`chapterID` - die id des Kapitels zu welchem die Seite gehört

Returns:

die id der angefragten Seite

Throws:

`java.sql.SQLException` - Datenbankfehler

getContentID

```
int getContentID(int contentNumber,
                 int pageID)
                   throws java.sql.SQLException
```

Diese Methode liefert zu der Nummer eines Seiteinhalts die entsprechende id in der Datenbank

Parameters:

`contentNumber` - Die Nummer des angefragten Seiteninhalts

`pageID` - die id der Seite zu welchem dir Inhalt gehört

Returns:

die id ders angefragten Seiteinhalts

Throws:

`java.sql.SQLException` - Datenbankfehler

getBookChildren

```
java.lang.String[] getBookChildren(int bookID)
                                     throws java.sql.SQLException
```

Diese Methode liefert zur id eines Buches alle Kapitel welche zu diesem Buch gehören

Parameters:

`bookID` - Die id des angefragten Buches

Returns:

der Name aller Kapitel, welche zum angefragten Buch gehören

Throws:

`java.sql.SQLException` - Datenbankfehler

getChapterChildren

```
java.lang.String[] getChapterChildren(int chapterID)
                                       throws java.sql.SQLException
```

Diese Methode liefert zur id eines Kapitels alle Seiten welche zu diesem Kapitel gehören

Parameters:

`chapterID` - Die id des angefragten Kapitels

Returns:

der Name aller Seiten, welche zum angefragten Kapitel gehören

Throws:

`java.sql.SQLException` - Datenbankfehler

getPageChildren

```
int[] getPageChildren(int pageID)
                      throws java.sql.SQLException
```

Diese Methode liefert zur id einer Seite alle Inhalte welche zu dieser Seite gehören

Parameters:

`pageID` - Die id der angefragten Seite

Returns:

der Name aller Inhalte, welche zur angefragten Seite gehören

Throws:

`java.sql.SQLException` - Datenbankfehler

getAllBooks

```
java.lang.String[] getAllBooks()
                                throws java.sql.SQLException
```

Diese Methode liefert den Namen aller Bücher welche in der Datenbank angelegt sind

Returns:

ein String[] mit den Namen aller angelegten Büchern

Throws:

`java.sql.SQLException` - Datenbankfehler

Overview Package Class Use Tree Deprecated Index Help

Prev Class **Next Class** Frames No Frames All Classes