

Datenbanken und Java

Die Abiturklassen im Zusammenspiel mit SQLite

Bohlen, Dr. Appel, vom Hau

Überblick

- 1 Abiturvorgaben
- 2 Abiturklassen
- 3 SQLite
- 4 In der Praxis
 - Vorbereitungen
 - Beispiele
 - Workshopphase
- 5 Weitere Anwendungen
 - Tabellenabfrage
 - Tabellendemo
 - Sportfestaufgabe
 - Verknüpfung zweier Datenbanken
 - Kiosk: Anknüpfen an Bekanntes
 - Optional: Workshopphase

Überblick

1 Abiturvorgaben

2 Abiturklassen

3 SQLite

4 In der Praxis

- Vorbereitungen
- Beispiele
- Workshopphase

5 Weitere Anwendungen

- Tabellenabfrage
- Tabellendemo
- Sportfestaufgabe
- Verknüpfung zweier Datenbanken
- Kiosk: Anknüpfen an Bekanntes
- Optional: Workshopphase

Grundkurs

| Daten und ihre Strukturierung* | Algorithmen* | Formale Sprachen und Automaten* | Informatiksysteme | Informatik, Mensch und Gesellschaft |
|--|--|---|-------------------------------------|-------------------------------------|
| Objekte und Klassen – Entwurfsdiagramme und Implementationsdiagramme – lineare Strukturen <ul style="list-style-type: none"> • array bis zweidimensional • Stapel (Klasse Stack) • Schlange (Klasse Queue) • lineare Liste (Klasse List) – nicht-lineare Strukturen <ul style="list-style-type: none"> • Binärbaum (Klasse BinaryTree) • binärer Suchbaum (Klasse BinarySearchTree) | Analyse, Entwurf und Implementierung von Algorithmen | Syntax und Semantik einer Programmiersprache – Java – SQL | Einzelrechner und Rechner-netzwerke | Wirkungen der Automatisierung |
| Datenbanken | Algorithmen in ausgewählten informatischen Kontexten | Endliche Automaten – Transformation eines nichtdeterministischen endlichen Automaten in einen deterministischen endlichen Automaten | Nutzung von Informatiksystemen | Grenzen der Automatisierung |
| | | Grammatiken regulärer Sprachen – Produktionen mit ϵ | Sicherheit | |
| | | Möglichkeiten und Grenzen von Automaten und formalen Sprachen | | |

* Materialien hierzu sind der Anlage im Bildungsportal (<http://www.schulentwicklung.nrw.de/lehreplaene/lehrplannavigator-s-ii/gymnasiale-oberstufe/informatik/hinweise-und-beispiele/>) zu entnehmen.

Leistungskurs

| Daten und ihre Strukturierung* | Algorithmen* | Formale Sprachen und Automaten* | Informatiksysteme | Informatik, Mensch und Gesellschaft |
|---|--|--|-------------------------------------|-------------------------------------|
| Objekte und Klassen - Entwurfsdiagramme und Implementationsdiagramme - lineare Strukturen <ul style="list-style-type: none"> • array bis zweidimensional • Stapel (Klasse Stack) • Schlange (Klasse Queue) • lineare Liste (Klasse List) - nicht-lineare Strukturen <ul style="list-style-type: none"> • Binärbaum (Klasse BinaryTree) • binärer Suchbaum (Klasse BinarySearchTree) • Graphen (Klassen Graph, Vertex, Edge) | Analyse, Entwurf und Implementierung von Algorithmen | Syntax und Semantik einer Programmiersprache - Java - SQL | Einzelrechner und Rechner-netzwerke | Wirkungen der Automatisierung |
| Datenbanken • Klassen DatabaseConnector/ QueryResult | Algorithmen in ausgewählten informati-schen Kontexten - Operationen der Datenstrukturen Stack und BinarySearchTree (nur search) - Algorithmen zur Kommunikation in Netzwerken (Klassen Connection, Client, Server) | Endliche Automaten und Kellerautomaten - Transformation eines nichtdeter-ministischen endlichen Automaten in einen deterministischen endlichen Automaten | Nutzung von Informatik-systemen | Grenzen der Automatisierung |
| | | Grammatiken regulärer und kontext-freier Sprachen - Produktionen mit ϵ | Sicherheit | |
| | | Scanner, Parser und Interpreter für eine reguläre Sprache | | |
| | | Möglichkeiten und Grenzen von Auto-maten und formalen Sprachen | | |

* Materialien hierzu sind der Anlage im Bildungsportal (<http://www.schulentwicklung.nrw.de/lehrplaene/lehrplannavigator-s-ii/gymnasiale-oberstufe/informatik/hinweise-und-beispiele/>) zu entnehmen.

Grundkurs

| Daten und ihre Strukturierung* | Algorithmen* | Formale Sprachen und Automaten* | Informatiksysteme | Informatik, Mensch und Gesellschaft |
|---|---|---|------------------------------------|--|
| Objekte und Klassen — Entwurfsdiagramme und Implementationsdiagramme — lineare Strukturen <ul style="list-style-type: none"> • array bis zweidimensional • Stapel (Klasse Stack) • Schlange (Klasse Queue) • lineare Liste (Klasse List) — nicht-lineare Strukturen <ul style="list-style-type: none"> • Binärbaum (Klasse BinaryTree) • binärer Suchbaum (Klasse BinarySearchTree) | Analyse, Entwurf und Implementierung von Algorithmen — Struktogramme | Syntax und Semantik einer Programmiersprache — Java — SQL | Einzelrechner und Rechnernetzwerke | Wirkungen der Automatisierung — Grundprinzipien des Datenschutzes <ul style="list-style-type: none"> • Verbot mit Erlaubnisvorbehalt • Erforderlichkeit |
| Datenbanken — Klassen DatabaseConnector, QueryResult | Algorithmen in ausgewählten informatischen Kontexten | Endliche Automaten — Deterministische endliche Automaten — Nichtdeterministische endliche Automaten | Nutzung von Informatiksystemen | Grenzen der Automatisierung |
| | | Grammatiken regulärer Sprachen — Produktionen mit ϵ | Sicherheit | |
| | | Möglichkeiten und Grenzen von Automaten und formalen Sprachen | | |

* Materialien hierzu stehen unter der Adresse <https://www.schulentwicklung.nrw.de/lehrplaene/lehrplannavigator-s-l/gymnasiale-oberstufe/informatik/hinweise-und-beispiele/hinweise-und-beispiele.html> zur Verfügung.

Leistungskurs

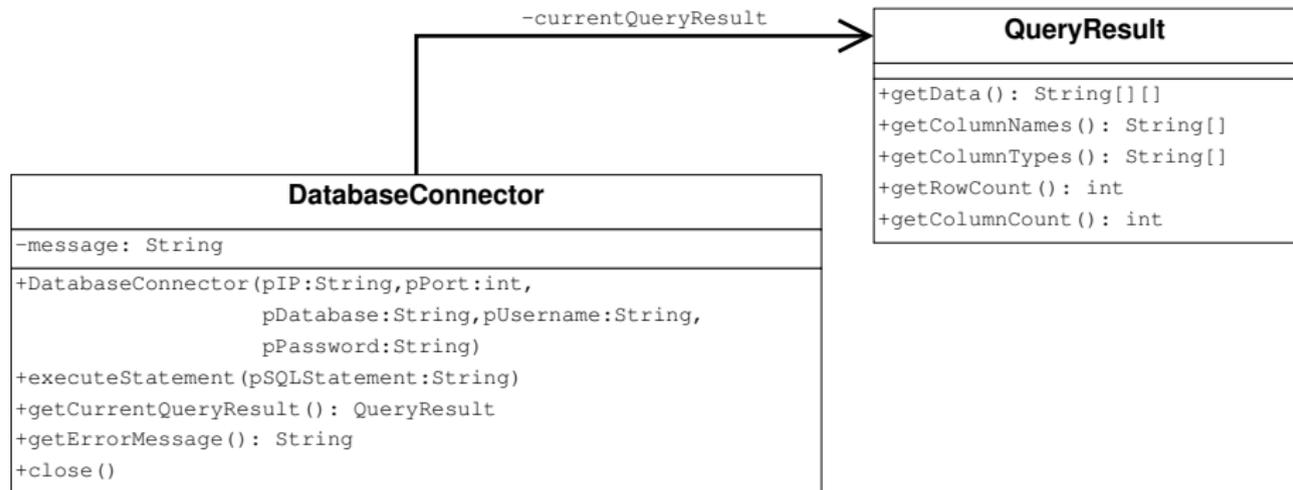
| Daten und ihre Strukturierung* | Algorithmen* | Formale Sprachen und Automaten* | Informatiksysteme | Informatik, Mensch und Gesellschaft |
|--|--|--|------------------------------------|--|
| Objekte und Klassen – Entwurfsdiagramme und Implementationsdiagramme – lineare Strukturen <ul style="list-style-type: none"> • array bis zweidimensional • Stapel (Klasse Stack) • Schlange (Klasse Queue) • lineare Liste (Klasse List) – nicht-lineare Strukturen <ul style="list-style-type: none"> • Binärbaum (Klasse BinaryTree) • binärer Suchbaum (Klasse BinarySearchTree) • Graphen (Klassen Graph, Vertex, Edge) | Analyse, Entwurf und Implementierung von Algorithmen – Struktogramme | Syntax und Semantik einer Programmiersprache – Java – SQL | Einzelrechner und Rechnernetzwerke | Wirkungen der Automatisierung – Grundprinzipien des Datenschutzes <ul style="list-style-type: none"> • Verbot mit Erlaubnisvorbehalt • Erforderlichkeit |
| Datenbanken – Klassen DatabaseConnector, QueryResult | Algorithmen in ausgewählten informatischen Kontexten – Algorithmen zur Kommunikation in Netzwerken (Klassen Connection, Client, Server) | Endliche Automaten und Kellerautomaten – Deterministische endliche Automaten – Nichtdeterministische endliche Automaten – Nichtdeterministische Kellerautomaten | Nutzung von Informatiksystemen | Grenzen der Automatisierung |
| | | Grammatiken regulärer und kontextfreier Sprachen – Produktionen mit ϵ | Sicherheit | |
| | | Scanner, Parser und Interpreter für eine reguläre Sprache | | |
| | | Möglichkeiten und Grenzen von Automaten und formalen Sprachen | | |

* Materialien hierzu stehen unter der Adresse <https://www.schulentwicklung.nrw.de/lehrplaene/lehrplannavigator-s-ii/gymnasiale-oberstufe/informatik/hinweise-und-beispiele/hinweise-und-beispiele.html> zur Verfügung.

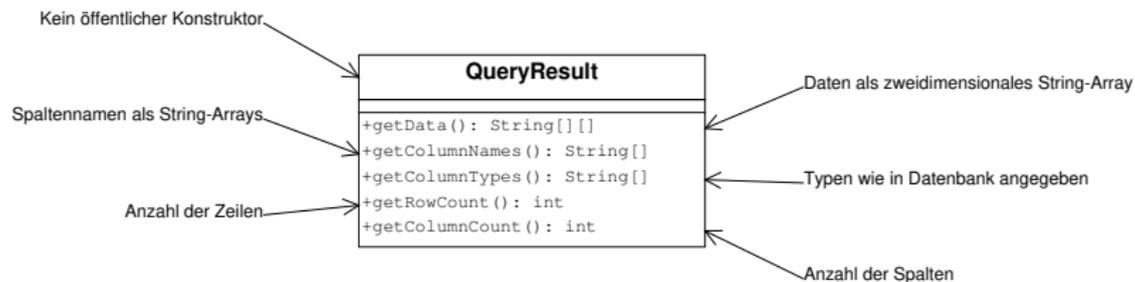
Überblick

- 1 Abiturvorgaben
- 2 Abiturklassen
- 3 SQLite
- 4 In der Praxis
 - Vorbereitungen
 - Beispiele
 - Workshopphase
- 5 Weitere Anwendungen
 - Tabellenabfrage
 - Tabellendemo
 - Sportfestaufgabe
 - Verknüpfung zweier Datenbanken
 - Kiosk: Anknüpfen an Bekanntes
 - Optional: Workshopphase

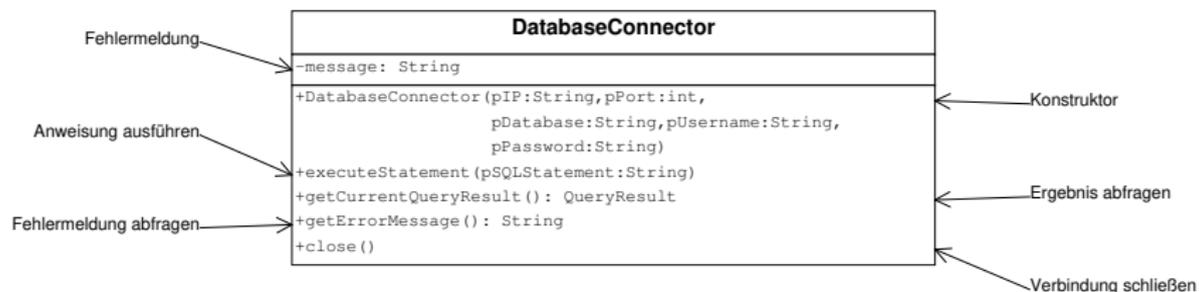
Die beiden Abiturklassen



Klasse QueryResult



Klasse DatabaseConnector



Konstruktor

```
DatabaseConnector (pIP:String,  
                  pPort:int,  
                  pDatabase:String,  
                  pUsername:String,  
                  pPassword:String)
```

← Dateiname

Überblick

- 1 Abiturvorgaben
- 2 Abiturklassen
- 3 SQLite
- 4 In der Praxis
 - Vorbereitungen
 - Beispiele
 - Workshopphase
- 5 Weitere Anwendungen
 - Tabellenabfrage
 - Tabellendemo
 - Sportfestaufgabe
 - Verknüpfung zweier Datenbanken
 - Kiosk: Anknüpfen an Bekanntes
 - Optional: Workshopphase

Vorteile von SQLite

- Datenbank nur eine Datei
- keine Installation nötig
- schlanke und portable GUI-Frontends verfügbar
 - ▶ DB Browser for SQLite
 - ▶ SQLiteStudio

DB Browser for SQLite - /home/daniel/Dropbox/Schule/Moderation und Beratung Info/Moderation (Dez. 46)/FoBi Java und DB/NetBeansWorkSpac

Datei Bearbeiten Ansicht Hilfe

New Database Open Database Write Changes Revert Changes

Datenbankstruktur Daten durchsuchen Pragmas bearbeiten SQL ausführen

SQL 1

```
1 select * from Schueler
```

| | SID | Vorname | Name | Geburtsdatum | Geschlecht | Wohnort | PLZ | Zeile |
|---|-----|---------|---------|--------------|------------|---------|-------|-------|
| 1 | 1 | Ann | Kopp | 2001-12-25 | w | Hanau | 63452 | Ze... |
| 2 | 2 | Karina | Winkler | 2001-02-18 | w | Hanau | 63450 | Ei... |

50 Rows returned from: select * from Schueler (took 3ms)

SQL Log

Zeige SQL von Benutzer Leeren

```
1 select * from Schueler
2
```

SQL Log Plot DB Schema UTF-8

<http://sqlitebrowser.org/>

Überblick

- 1 Abiturvorgaben
- 2 Abiturklassen
- 3 SQLite
- 4 In der Praxis**
 - Vorbereitungen
 - Beispiele
 - Workshopphase
- 5 Weitere Anwendungen
 - Tabellenabfrage
 - Tabellendemo
 - Sportfestaufgabe
 - Verknüpfung zweier Datenbanken
 - Kiosk: Anknüpfen an Bekanntes
 - Optional: Workshopphase

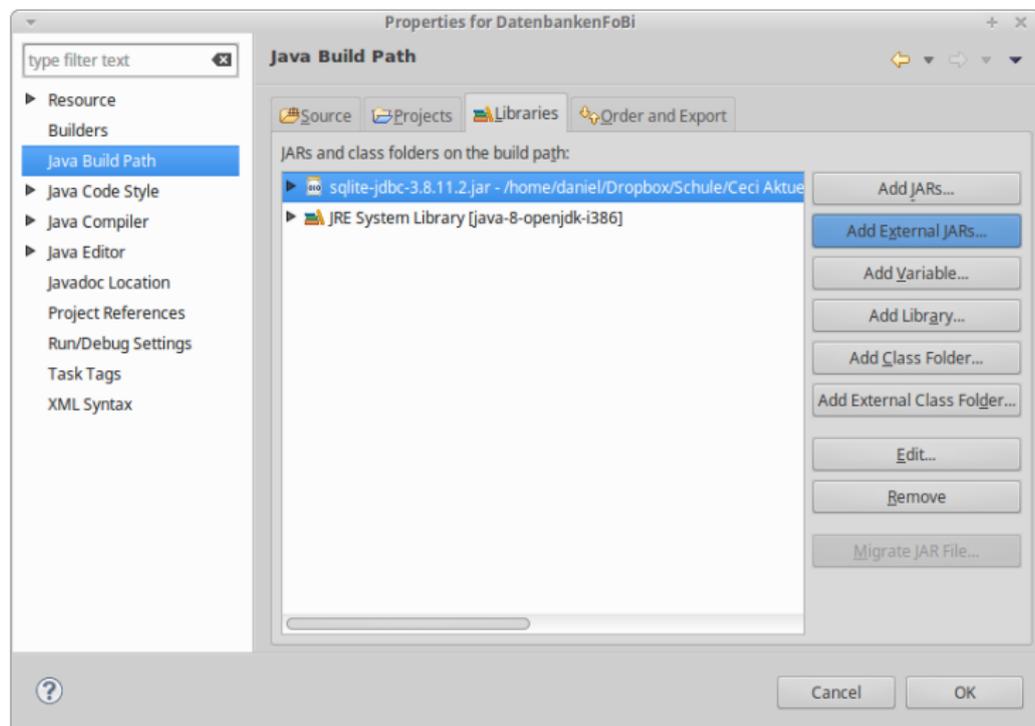
Lehrer – Vorbereitungen vor dem Unterricht

- Datenbank vorbereiten und als `.db`-Datei bereitstellen
- Abiturklassen bereitstellen (in unserem Fall Version für SQLite)
- Treiber `sqlite-jdbc-x.x.x.x.jar` bereitstellen

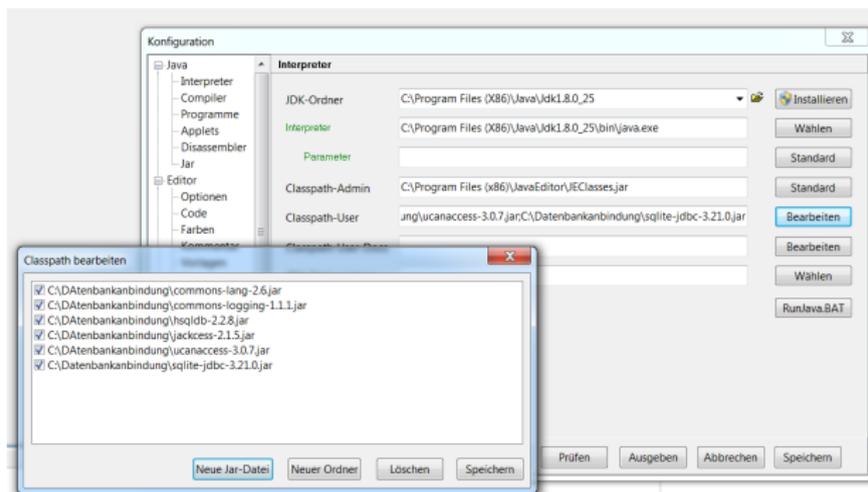
Schüler – Vorbereitungen im Unterricht

- `.db`-Datei in Projektordner kopieren
- Abiturklassen (inkl. **Queue**) in Projekt hinzufügen
- Treiber `sqlite-jdbc-x.x.x.x.jar` in Projekt einbinden

Schüler – Treiber einbinden in Eclipse



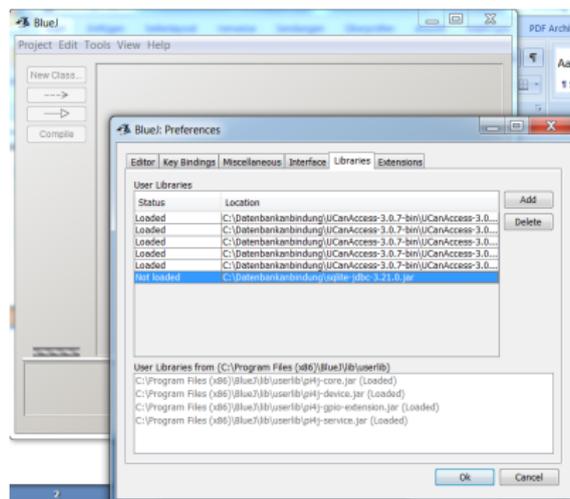
Schüler – Treiber einbinden im Java Editor



Bei Verwendung des Java-Editors von Gerhard Röhner <http://javaeditor.org/doku.php> (abgerufen am 01.03.2017) findet man die Möglichkeit zur Einbindung im Menü *Fenster* unter *Konfiguration*.

Schüler – Treiber einbinden in BlueJ

In BlueJ ist dies unter dem Menüpunkt *Preferences* im Menü *Tools* entsprechend möglich. Dort oben den Reiter *Libraries* anklicken und den Pfad der Treiberklasse hinzufügen.



Die Änderungen werden erst bei einem Neustart der Virtuellen Maschine bzw. des Programms aktiv. (Der Status ändert sich dann in *Loaded*.)

Beispiel – Anweisung ausführen

```
public class ErstesBeispiel {  
  
    public static void main(String[] args) {  
  
        // DatabaseConnector erstellen  
        DatabaseConnector myCon =  
            new DatabaseConnector("", 0, "Schuelerdaten.db", "", "");  
  
        // Anweisung ausfuehren  
        myCon.executeStatement("select Name from Schueler");  
  
        //...  
  
    }  
  
}
```

Beispiel – Ausgabe

```
public class ErstesBeispiel {

    public static void main(String[] args) {
        //...

        // Fehler anzeigen lassen
        System.out.println(myCon.getErrorMessage());

        // Ergebnis anzeigen lassen
        if(myCon.getErrorMessage()==null) {
            for(int i=0; i<myCon.getCurrentQueryResult().
                getRowCount(); i=i+1){
                System.out.println(myCon.getCurrentQueryResult().
                    getData()[i][0]);
            }
        }
    }
}
```

Beispiel – Spaltennamen und -typen

```
myCon.executeStatement("select * from Schueler");
for(int i=0; i<myCon.getCurrentQueryResult().
    getColumnCount(); i=i+1){
    System.out.println(myCon.getCurrentQueryResult().
        getColumnNames()[i] + "\t"+
        myCon.getCurrentQueryResult().getColumnTypes()[i]);
}
```

Ausgabe:

SID INTEGER

Vorname TEXT

Name TEXT

Geburtsdatum TEXT

Geschlecht TEXT

Wohnort TEXT

PLZ INTEGER

Strasse TEXT

Hausnummer INTEGER

Bezeichnung TEXT

Beispiel – Parsen

```
// Strings in Abfrage mit einfachen Anführungszeichen  
markieren!  
myCon.executeStatement("select * from Schueler where  
    Name = 'Nowak' and Vorname = 'Kai'");  
// Ergebnis ist zweidimensionales String-Array!  
  
int id = Integer.parseInt(  
    myCon.getCurrentQueryResult().getData()[0][0]  
);
```

Beispiel – INSERT

Beispieleintrag:

| | SID | Vorname | Name | Geburtsdatum | Geschlecht | Wohnort | PLZ | Strasse | Hausnummer | Bezeichnung |
|---|--------|---------|--------|--------------|------------|---------|--------|-----------|------------|-------------|
| | Fil... | Filter | Filter | Filter | Filter | Filter | Filter | Filter | Filter | Filter |
| 1 | 1 | Ann | Kopp | 2001-12-25 | w | Hanau | 63452 | Zeisigweg | 342 | Q1 |

Neuer Eintrag:

```
insert into Schueler
  values (99, 'Jesse', 'Pinkman', 1984-09-24, 'm',
         'Albuquerque', 87104, 'Margo Street', 9809,
         'Q1')
```

Es werden sowohl einfache als auch doppelte Anführungszeichen akzeptiert!

```
insert into Schueler (SID, Vorname, Name)
  values (100, "Walter Jr.", "White")
```

Beispiel – UPDATE

```
update Schueler  
  set Bezeichnung = 'EF'  
  where SID = 99
```

Beispiel – DELETE

```
delete from Schueler  
  where SID = 99 or SID = 100
```

Alle Einträge löschen:

```
delete from Schueler
```

Beispiel – testen Sie es selbst

Nun sind Sie dran :)

Überblick

- 1 Abiturvorgaben
- 2 Abiturklassen
- 3 SQLite
- 4 In der Praxis
 - Vorbereitungen
 - Beispiele
 - Workshopphase
- 5 Weitere Anwendungen
 - Tabellenabfrage
 - Tabellendemo
 - Sportfestaufgabe
 - Verknüpfung zweier Datenbanken
 - Kiosk: Anknüpfen an Bekanntes
 - Optional: Workshopphase

Beliebige Datenbank, beliebige SQL-Anfrage

Diese Anwendung wird Ihnen live vorgeführt :)

Änderungen in Tabellen in Datenbank speichern

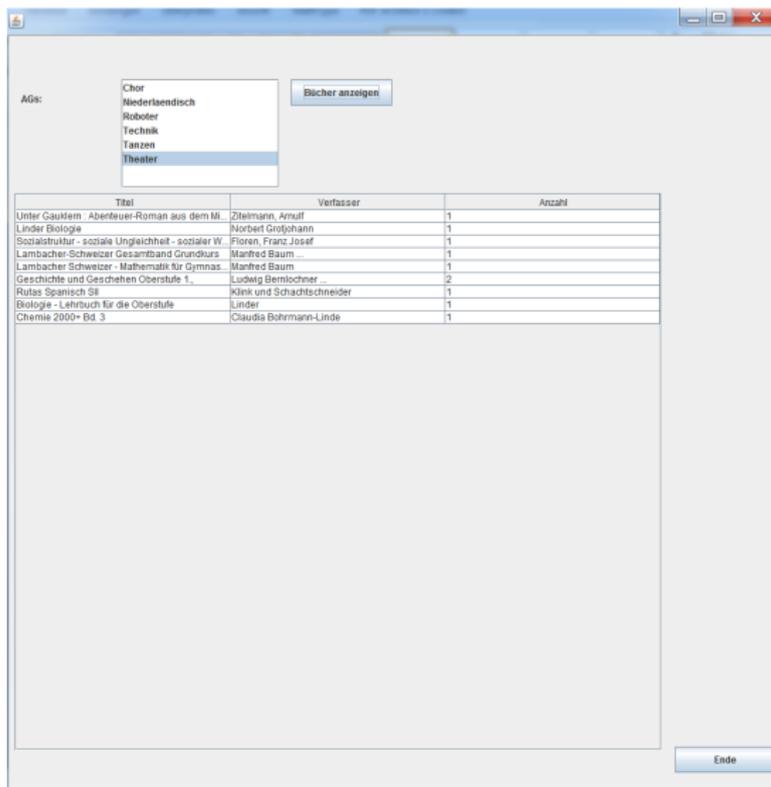
Diese Anwendung wird Ihnen live vorgeführt :)

Teil einer offiziellen Beispielaufgabe

| | WettkampfID | Bezeichnung |
|---|-------------|-------------------|
| | Filter | Filter |
| 1 | 1 | Hürdenlauf |
| 2 | 2 | Slalomlauf |
| 3 | 3 | Sprint |
| 4 | 4 | Abenteuerparcours |

Diese Anwendung wird Ihnen live vorgeführt :)

AGs und Schulbuchausleihe



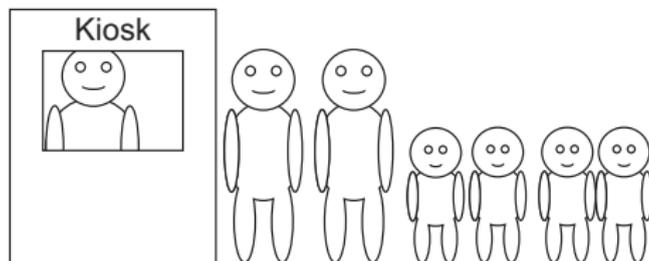
The screenshot shows a software application window with a search interface. On the left, there is a label 'AGs:' followed by a list of categories: 'Chor', 'Niederländisch', 'Roboter', 'Technik', 'Tanzen', and 'Theater'. The 'Theater' category is selected and highlighted. To the right of this list is a button labeled 'Bücher anzeigen'. Below the search area is a table with three columns: 'Titel', 'Verfasser', and 'Anzahl'. The table contains the following data:

| Titel | Verfasser | Anzahl |
|---|----------------------------|--------|
| Unter Cauhem - Abenteuer-Roman aus dem 18. | Zielmann, Arnulf | 1 |
| Linder Biologie | Norbert Großjohann | 1 |
| Sozialstruktur - soziale Ungleichheit - sozialer W. | Floren, Franz Josef | 1 |
| Lambacher-Schweizer Gesamtband Grundkurs | Manfred Baum ... | 1 |
| Lambacher-Schweizer - Mathematik für Gymnas. | Manfred Baum | 1 |
| Geschichte und Geschichten Oberstufe 1. | Ludwig Bernthoener | 2 |
| Rutas Spanisch SII | Klink und Schachtschneider | 1 |
| Biologie - Lehrbuch für die Oberstufe | Linder | 1 |
| Chemie 2000* Bd. 3 | Claudia Bohrmann-Linde | 1 |

At the bottom right of the application window, there is a button labeled 'Ende'.

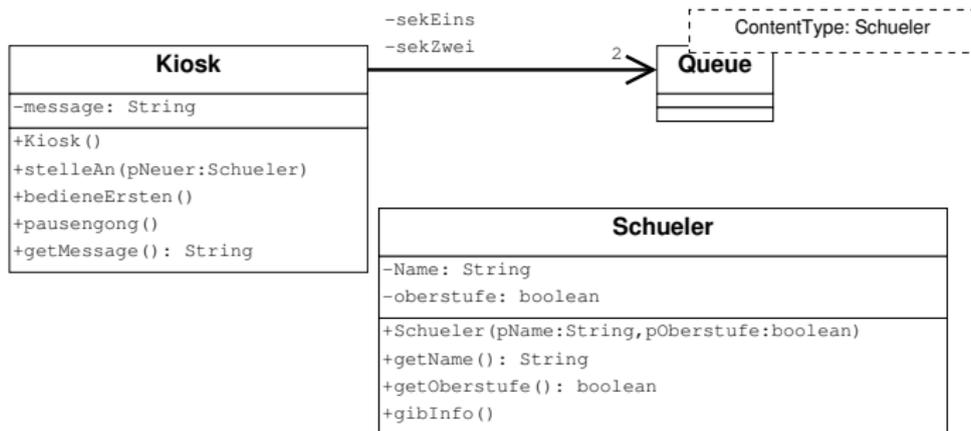
Diese Anwendung wird Ihnen live vorgeführt :)

Schulkiosk – Anknüpfen an bekannte Modellierung



- Die Schüler sollen sich einer Warteschlange anreihen.
- Die Schüler der Sekundarstufe I halten sich an diese Regel.
- Ein Oberstufenschüler stellt sich immer hinter den momentan letzten Oberstufenschüler. Gibt es noch keine anderen Oberstufenschüler, ist er der neue Erste.
- Beim Pausengang verlassen alle Schüler der Sek. I die Schlange, die Oberstufenschüler bleiben.

Schulkiosk – Diagramm



Schulkiosk – Beispielmethode ohne Datenbank

```
public void stelleAn(Schueler pNeuer) {  
  
    message = "Es hat sich angestellt: " + pNeuer.  
        getName();  
    if (pNeuer.getObertufe()) {  
        sekZwei.enqueue(pNeuer);  
        message = message + " Oberstufe";  
    } else {  
        sekEins.enqueue(pNeuer);  
        message = message + " Mittelstufe";  
    }  
  
}
```

Schulkiosk – Unterklasse mit Datenbank

```
import databases.DatabaseConnector;

public class DatenbankKiosk extends Kiosk {

    DatabaseConnector myCon;

    // Konstruktor
    public DatenbankKiosk(String pDB) {
        super();
        myCon = new DatabaseConnector("", 0, pDB, "", "");

        if (meinConnector.getErrorMessage() != null) {
            System.out.println(meinConnector.getErrorMessage());
        }

    }
    //...
}
```

Schulkiosk – Beispielmethode mit Datenbank

| SID | Vorname | Name | Geburtsdatum | Geschlecht | Wohnort | PLZ | Strasse | Hausnummer | Bezeichnung |
|-----|---------|------|--------------|------------|---------|-------|-----------|------------|-------------|
| 1 | Ann | Kopp | 2001-12-25 | w | Hanau | 63452 | Zeisigweg | 342 | Q1 |

```
public void stelleAn(int pId) {  
    // Anfrage wird ausgeführt:  
    meinConnector.executeStatement("select Vorname, Name,  
        Bezeichnung from Schueler where SID=" + pId);  
  
    // Hier fehlt noch was :)  
  
}
```

Nun sind wieder Sie dran...

wenn Sie möchten :)