OurMovies

Christoph Pickl

Agenda

1. **OurMovies -** Das Programm kurz vorgestellt
2. **SwingX -** Die Zukunft von Swing; bereits jetzt
3. **QTJ -** QuickTime Integration für Java
4. **db4o -** Eine objektorientierte Datenbank

Profil

- Christoph Pickl
- 22 Jahre, Wien (urspr. Bgld)
- Bakk. Software Engineering
- Java Entwickler bei Erste Bank
- DJ bei Hörspiel Crew

OurMovies

OurMovies ist ...

- ... ein weiteres Tool um **Videos** zu verwalten
- ... ein Video-**Player** (*QuickTime vorausgesetzt*)
- ... **OpenSource** und auf SourceForge gehosted
- ... dem Look&Feel von **iTunes** nachempfunden

... entstandem aus dem einfachen Wunsch heraus,
den Prozess des Sharings zu vereinfachen:

1. HTML-Report erstellen
2. HTML verschicken und sich IDs schicken lassen (zb: "[[31,53,84]]")
3. mit SmartCopy Filme einfach anhand IDs kopieren (auf externe HD)
4. vorbereitete externe HD mitnehmen

Metriken

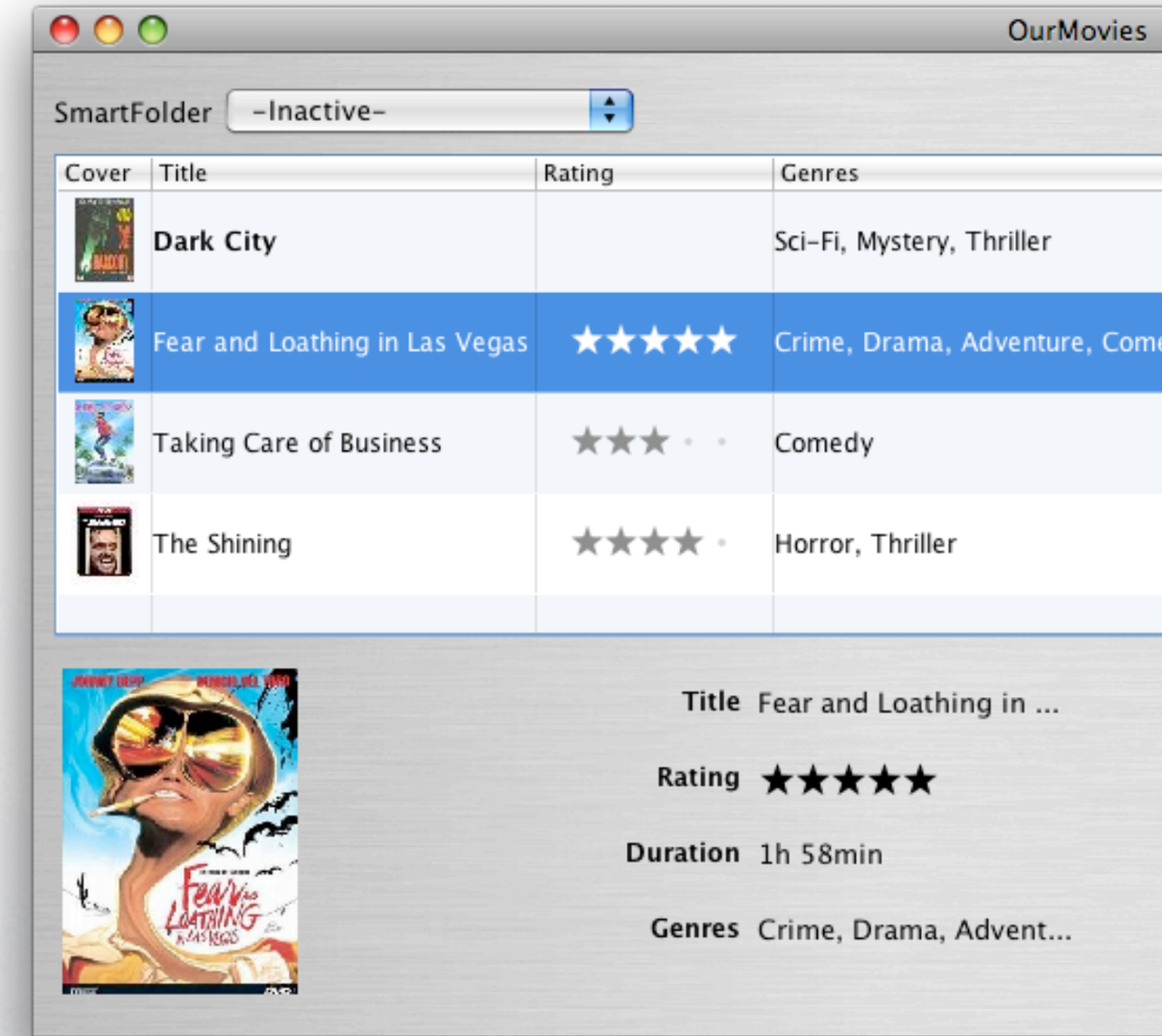
Modul	Klassen	LoC
omovApp	173	12,193
omovCore	143	9,065
omovQtjApi/Impl	19	1,399
omovWebApi/Impl	7	689
Summe	**342**	**23,346**

App == rein die aufgesetzte GUI
Core == BusinessLogik & Co

-> viel Zeit in die GUI investiert

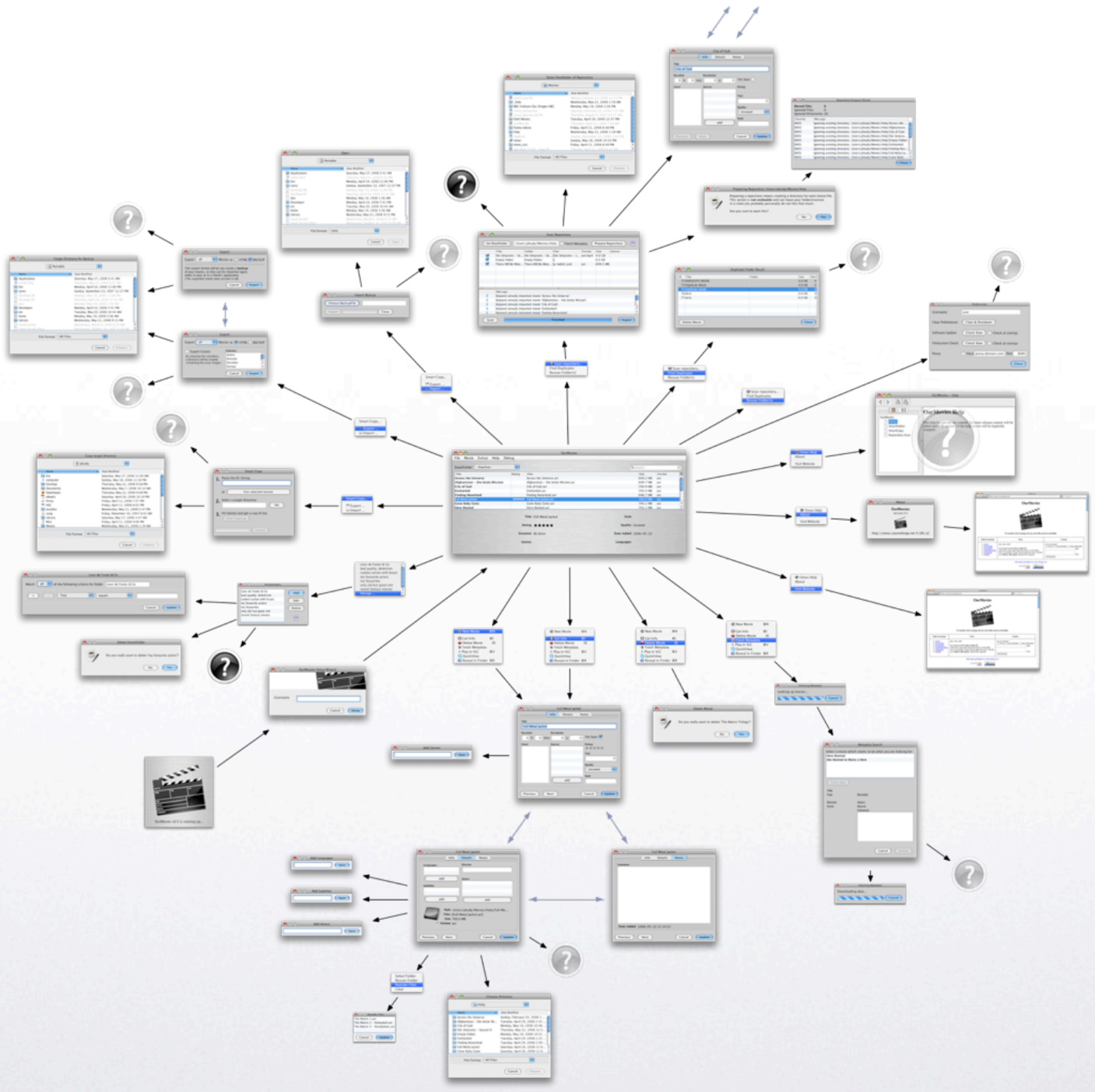
Keyfeatures

- Metadata Fetching
- SmartFolder
- SmartCopy
- QuickView



* Remote Feature in arbeit (mit v2.0 verfügbar)
  ... Filme über's Netzwerk synchronisieren
  ... nur Wunschliste (IDs für SmartCopy) erstellen
  keine Übermittlung der Filmdateien selbst (zaaach)

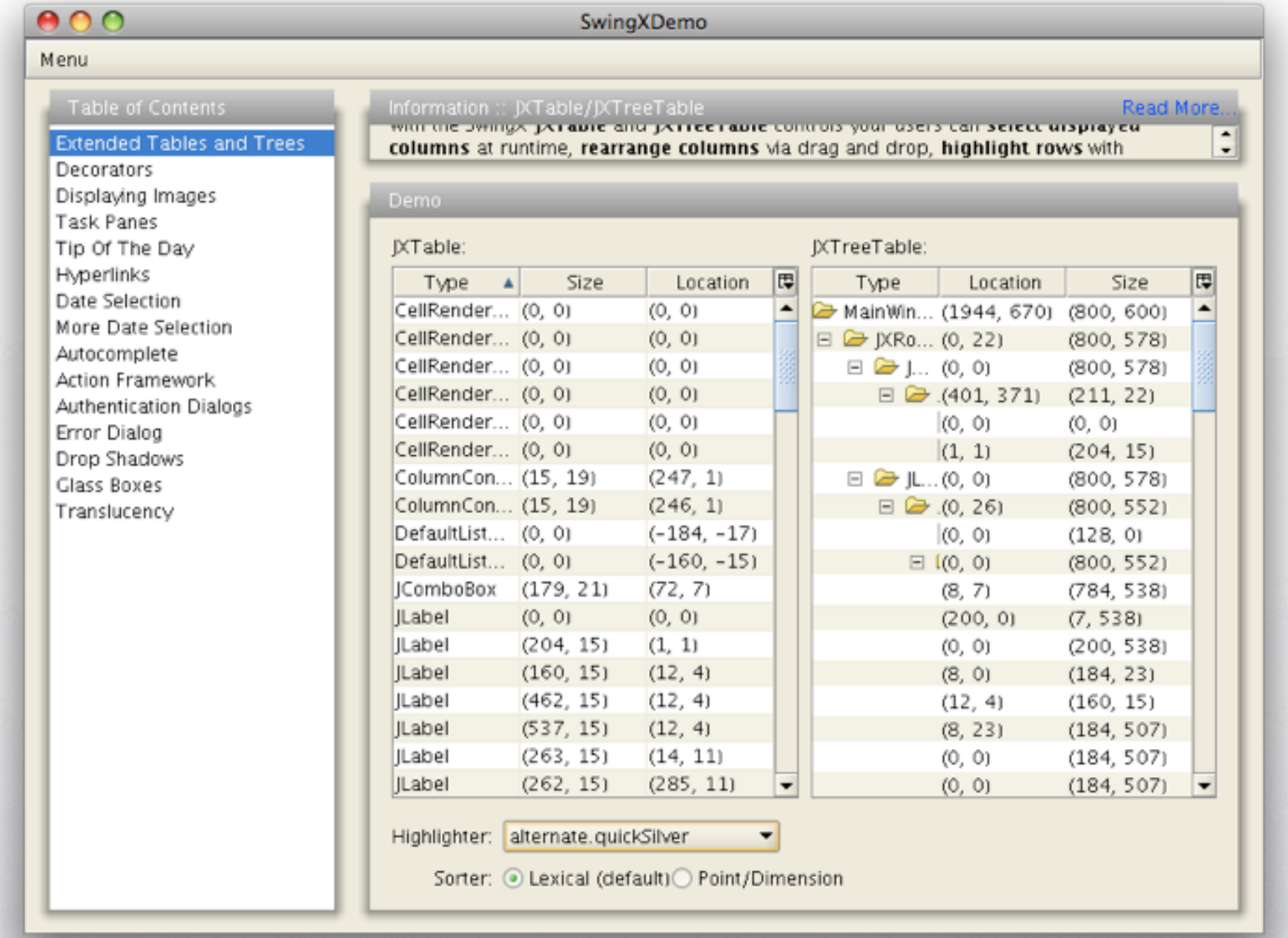
OurMovies v0.6 is starting up...



SwingX

Enhanced Components

- Sorting/Filtering for Tables/Trees/Lists
- Find/Search
- Auto-Completion
- Login/authentication framework
- Date picker component
- Drop Shadows, Translucency, ...


SwingXDemo
Menu
Table of Contents
Extended Tables and Trees
Decorators
Displaying Images
Task Panes
Tip Of The Day
Hyperlinks
Date Selection
More Date Selection
Autocomplete
Action Framework
Authentication Dialogs
Error Dialog
Drop Shadows
Glass Boxes
Translucency
Information :: JXTable/JXTreeTable
Read More...
columns at runtime, rearrange columns via drag and drop, highlight rows with
Demo
JXTable:
Type
Size
Location
CellRender... (0, 0) (0, 0)
CellRender... (0, 0) (0, 0)
CellRender... (0, 0) (0, 0)
CellRender... (0, 0) (0, 0)
CellRender... (0, 0) (0, 0)
CellRender... (0, 0) (0, 0)
ColumnCon... (15, 19) (247, 1)
ColumnCon... (15, 19) (246, 1)
DefaultList... (0, 0) (-184, -17)
DefaultList... (0, 0) (-160, -15)
JComboBox (179, 21) (72, 7)
JLabel (0, 0) (0, 0)
JLabel (204, 15) (1, 1)
JLabel (160, 15) (12, 4)
JLabel (462, 15) (12, 4)
JLabel (537, 15) (12, 4)
JLabel (263, 15) (14, 11)
JLabel (262, 15) (285, 11)
JXTreeTable:
Type
Location
Size
MainWin... (1944, 670) (800, 600)
JXRo... (0, 22) (800, 578)
J... (0, 0) (800, 578)
(401, 371) (211, 22)
(0, 0) (0, 0)
(1, 1) (204, 15)
JL... (0, 0) (800, 578)
(0, 26) (800, 552)
(0, 0) (128, 0)
(0, 0) (800, 552)
(8, 7) (784, 538)
(200, 0) (7, 538)
(0, 0) (200, 538)
(8, 0) (184, 23)
(12, 4) (160, 15)
(8, 23) (184, 507)
(0, 0) (184, 507)
(0, 0) (184, 507)
Highlighter:
alternate.quickSilver
Sorter:
Lexical (default)
Point/Dimension

JXTable vs JTable

- Alternierende Hintergrundfarbe der Zeilen
- Tabellenspalten ein-/ausblenden
- Nach Spalten sortieren (Indizes?!)
- Tabellenhintergrund “weiterzeichnen”
- Vertikale Linien des Grids zeichnen

Auto-Completion

- Vorschläge für JTextField/JComboBox/JList
- strikte und nicht-strikte Vorschläge
- Vorgeschlagener Text vormarkiert

QTJ

QuickTime for Java

- Plattform-unabhängige API
- Manipulieren von Bildern/Audio/Video
- Erstellung von 2D/3D-Animationen
- Integration von QuickTime Funktionalität, nicht reines API für die Anwendung selbst

```
QTSession.open();

File file = new File("/some_movie.mpg");
OpenMovieFile openMovieFile =
    OpenMovieFile.asRead(new QTFile(file));
Movie movie = Movie.fromFile(openMovieFile);

MoviePlayer player = new MoviePlayer(movie);
QTJComponent qtjPlayer = QTFactory.makeQTJComponent(player);
JComponent playerComponent = qtjPlayer.asJComponent();

JPanel somePanel = new JPanel();
somePanel.add(playerComponent);

// ...

QTSession.close();
```

```
MovieController controller = new MovieController(movie);
controller.setKeysEnabled(true);
QTComponent qtControllerComponent = QTFactory.makeQTComponent(controller);
Component controllerComponent = qtControllerComponent.asComponent();
```

db4o

db4o ist ...

- ... OpenSource (+ kommerzielle Lizenz)
- ... für Java und .Net (C#) verfügbar
- ... einfach installiert/konfiguriert/verwendet
  - Jar runterladen und in den Klassenpfad
  - Keine Konfiguration/Treiber/Mappingfiles

```
Db4o.openFile("db.yap").set(new Object());
```

db4objects

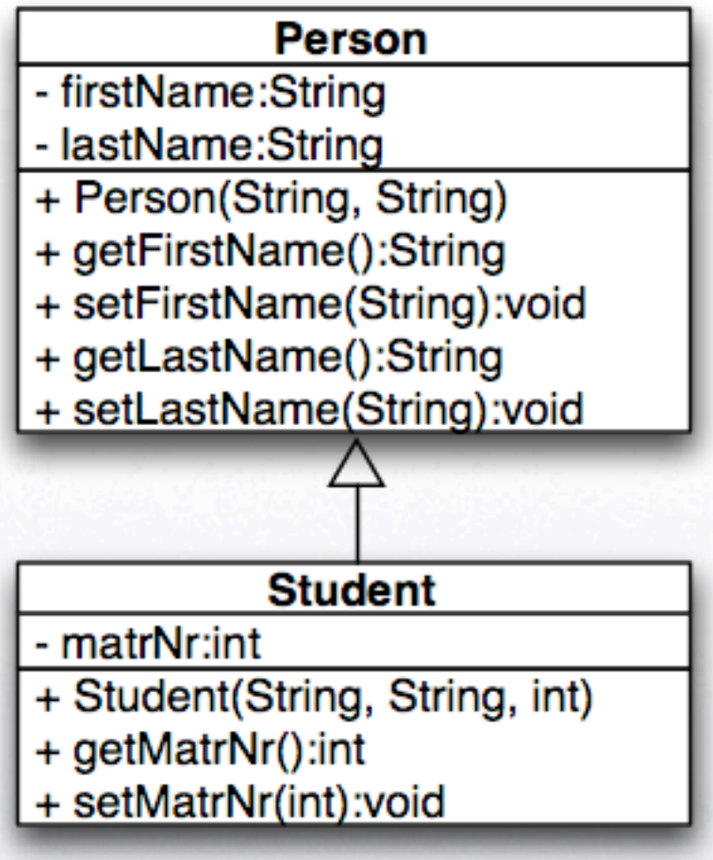
Wie geeignet sind RDBMS für OOP?

- komplexe Objekte in flache Relationen
- Mapping ist mühevoll/lästig/umständlich (trotz Hibernate & Co)
- wir kennen alle RDBMS, deshalb benützen wir sie (weil wir sie gewohnt sind)
- performance von RDBMS ist aber -derzeit- besser als von OODBMS

Erste Schritte

```
package at.ac.tuwien.e0525580.jsuga;

import com.db4o.*;

public class Db4oFirstSteps {
  public static void main(String[] args) {
    ObjectContainer db = Db4o.openFile("db.yap");
    try {
      // database operations
    } finally {
      db.close();
    }
  }
}
```

(einfaches) Datenmodell



Insert/Update/Delete

```
// insert
Person person = new Person(“Anna”, “Nym”);
db.set(person);


// update
person.setLastName(“Nüm”);
db.set(person);


// delete
db.delete(person);
```

nicht ganz so einfach,
da es haken gibt bzgl ObjektIdentität!

woher weiss db4o welches objekt gemeint ist,
ohne primary key (int ID) ?

Abfragesprachen

1. **QbE** (Query by Example)
   - Abfrage anhand eines Prototyps

2. **SODA** (Simple Object Database Access)
   - Verknüpfung von Constraints

3. **NQ** (Native Queries)
   - Filterung durch `Predicate` Interface

QbE - Simple Select

```
// fetch all objects by given Class instance
ObjectSet<Person> result = db.get(Person.class);

// ObjectSet implements List and Iterable interface
// therefore foreach loop can be used
for(Person p : result) {

  // process persons (and students!)

}
```

QbE - By Example

```
// default values (null, false, 0, 0.0) will be ignored
Person prototype = new Person("Anna", null);

// pass ObjectContainer the example
ObjectSet<Person> result = db.get(prototype);

while(result.hasNext()) {
  Person p = result.next();
  // process every person with firstName equals "Anna"
}
```

SODA - Simple Select

```
// create a Query object (obtained from ObjectContainer)
Query query = db.query();

// define a constrain
query.constrain(Student.class);

// result contains all students
ObjectSet<Student> result = query.execute();
```

- abstrakte Spezifikation durch Constrain-Graph
- ...

SODA - Where Clause

```
Query query = db.query();

query.constrain(Student.class);

// no getters necessary (private variables are accessible)
query.descend("firstName").constrain("Anna");

// will be and-connected with previous constrain (default)
query.descend("matrNr").constrain(525580).or(
    query.descend("matrNr").constrain(525581)
  );

ObjectSet<Student> result = query.execute();
```

SODA - Where Clause

```
// SELECT *
Query query = db.query();

// FROM student
query.constrain(Student.class);

// WHERE firstName like 'A%'
query.descend("firstName").constrain("A").startsWith(true);
// AND matrNr >= 500000
query.descend("matrNr").constrain(500000).greater().equal();
// ORDER BY lastName ASC
query.descend("lastName").orderAscending();

ObjectSet<Student> result = query.execute();
```

SODA - Evaluation

```
Query query = db.query();

query.constrain(Student.class);
query.constrain(new Evaluation() {
  public void evaluate(Candidate candidate) {
    Student student = (Student) candidate.getObject();
    boolean match = (student.getMatrNr() >= 500000);
    candidate.include(match)
  }
});

// performance of Evaluation mechanism is very bad :(
ObjectSet<Student> result = query.execute();
```

NQ - Simple Select

```
ObjectSet<Person> result = db.query(
  new Predicate<Person>() {
    public boolean match(Person candidate) {
      return candidate.getFirstName().startsWith("A");
      // or fetch all: return true;
    }
  }
);
```

- NQ wird (optional) in äquivalente SODA-Darstellung transformiert
  -> NQ sind auch gegen Feldindizes ausführbar

NQ - Sortiert

```
Predicate<Student> predicate = new Predicate<Student>() {
  public boolean match(Student candidate) {
    boolean matched =
      candidate.getMatrNr() >= 500000 &&
      candidate.getMatrNr() <  600000;
    return matched;
  }
};
Comparator<Student> comparator = new Comparator<Student>() {
  public int compare(Student s1, Student s2) {
    return s1.getLastName().compareTo(s2.getLastName());
  }
};
ObjectSet<Student> result = db.query(predicate, comparator);
```

	+	-
QbE	einfach zu erlernen	nicht nach Defaultwerte suchbar
		äußerst ausdrucksschwach
SODA	Feldindizes werden genutzt	Attributname als Strings (Tippfehler erst zur Laufzeit)
	etwas performanter als NQ	bricht das Geheimnisprinzip
NQ	in Programmiersprache definiert (Typsicherheit, refaktorisierbar)	viel Schreibarbeit notwendig (evtl mit Java 7 closures verfübar)
	sehr ausdrucksstark	

Konfiguration - Global

```
Configuration config = Db4o.configure();

config.generateUUIDs(Integer.MAX_VALUE);
config.detectSchemaChanges(true);
config.callConstructors(true);
config.messageLevel(2);
config.readOnly(false);
config.exceptionsOnNotStorable(true);
config.optimizeNativeQueries(true);
config.setBlobPath("/db4o/blob/path");
// ...

// open db-file afterwards!
ObjectContainer db = Db4o.openFile("db4o.yap");
```

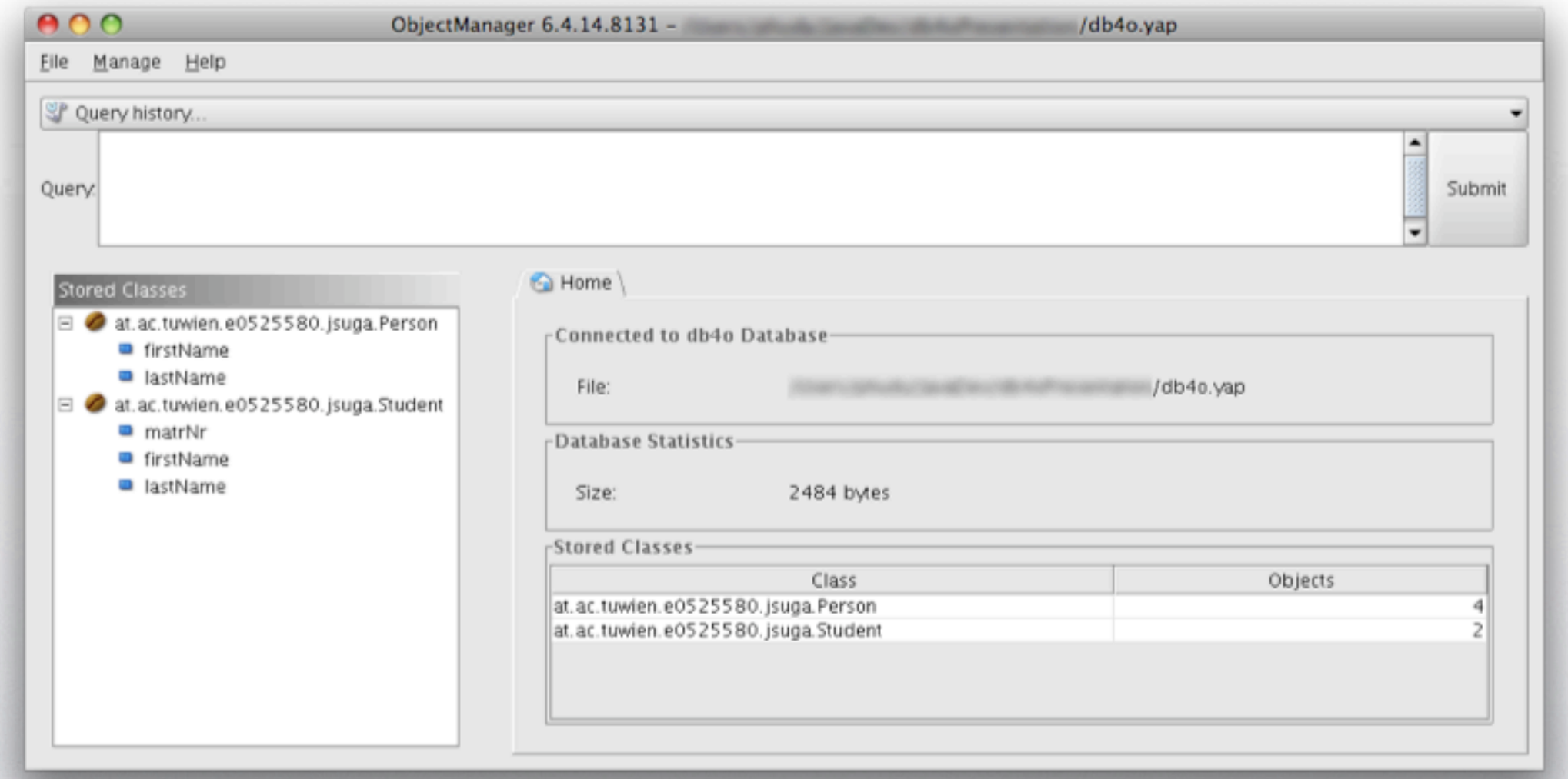
Konfiguration - Objekte

```
Configuration config = Db4o.configure();

config.objectClass(Person.class).cascadeOnActivate(true);
config.objectClass(Person.class).cascadeOnUpdate(true);
config.objectClass(Person.class).cascadeOnDelete(true);
config.objectClass(Person.class).storeTransientFields(true);
config.objectClass(Person.class).
  objectField(“lastName”).indexed(true);
// ...

// open db-file afterwards!
ObjectContainer db = Db4o.openFile(“db4o.yap”);
```

- gibt auch Annotation “Indexed”

```
public class Person {
  @Indexed
  private String firstName;
}
```

ObjectManager



Offene Themen

- Aktivierungstiefe
- Objektidentität
- Transaktionen
- Server
  (embedded/networking)
- ...

Links

http://omov.sourceforge.net

http://db4o.com

http://swinglabs.org

http://developer.apple.com/quicktime/qtjava

JavaDeus

- 19. Juni 2008
- FH St. Pölten (gratis Bustransfer)
- keine Teilnahmegebühr
- Präsentation von openXMA

http://at.sun.com/sunnews/events/2008/jun/javadeus08/