

## Data Persistence and Object-Relational Mapping

Slides by James Brucker, used with his permission

1

### Goal

- Applications need to save data to **persistent storage**.
- Persistent storage can be database, directory service, or other.
- For O-O programming, we'd like to save and retrieve **objects** to/from storage.

2

### The Problem with Databases

- Databases store data in rows in **tables**, which are not like objects.
- We can simulate object **associations** and **collections** using **relations** between rows in tables.
- Preserving uniqueness of objects and some object properties using persistence is difficult.
- Some conceptual differences exist, referred to as the **Object-Relational Paradigm Mismatch**

3

### Object-Relational Mapping

**Purpose:**

- save object as a row in a database table
- retrieve data from tables and create objects
- save and recreate **associations** between objects

**Design Goals:**

- separate object-relational mapping services from the rest of our program
- minimize the impact of changing database vendor or database schema

4

### An Example

An Event Manager application with these classes:

5

### Object-Relational Mapping

Map between an object and a row in a database table.

**Location**

id: int  
name: String  
address: String

**Class**  
should have an *identifier* attribute

**Data Mapper**  
convert *object* to *table row data*,  
convert *data types*,  
instantiates *objects*

**LOCATIONS**

PK	id	INTEGER
	name	VARCHAR(80)
	address	VARCHAR(160)

**Database Table**  
*identifier* is usually the *primary key* of table

6

### Mapping an Object

```
ku : Location
id = 101
name = "Kasetsart University"
address = "90 Pahonyotin ..."
```

object diagram

save()

LOCATIONS		
id	name	address
101	Kasetsart University	90 Pahonyotin ...
102	Seacon Square	120 Srinakarin ...

7

### O-R Mapping Code for Location (1)

```
Location ku = new Location( "Kasetsart University" );
ku.setAddress( "90 Pahonyotin Road; Bangkok" );
// save the location
dataMapper.save( location );
```

Issues:

- data mapper should choose a unique ID for persisted objects
- what happens if same data is already in the table?

8

### O-R Mapping Code for Location (2)

```
// retrieve the location
Location ku1 = dataMapper.find( "Kasetsart University" );
Location ku2 = dataMapper.find( "Kasetsart University" );
```

- how to we tell find what field to search for? id? name?
- our code does same find twice, does mapper return the same object?  
( ku1 == ku2 ) => true or false?

```
// update the address
ku1.setAddress( "Kampaengsaen Road; Kampaengsaen" );
```

- is the address updated automatically in the database?

9

### Transparent Persistence

With *transparent persistence*, any changes to a persistent object are automatically propagated to the database.

```
Location ku = new Location( "Kasetsart University" );
ku.setAddress( "90 Pahonyotin Road; Bangkok" );
// save the location
dataMapper.save( ku );
// change the address
ku.setAddress( "Kampaengsaen, Nakorn Pathom" );
```

LOCATIONS		
id	name	address
101	Kasetsart University	Kampaengsaen ...
102	Seacon Square	120 Srinakarin ...

10

### O-R Mapping of n-to-1 Associations

**Event**

```
id: int
name: String
startDate: Date
location: Location
```

\*

→

1

**Location**

```
id: int
name: String
address: String
```

11

### O-R Mapping of n-to-1 Associations

**Event**

```
id: int
name: String
startDate: Date
location: Location
```

\*

→

1

**Location**

```
id: int
name: String
address: String
```

The Data Mapper converts a n-to-1 association to a foreign key relation (persist) or foreign key to object (retrieve).

EVENTS	
PK id	INTEGER
name	VARCHAR
start_date	TIMESTAMP
FK location_id	INTEGER

LOCATIONS	
PK id	INTEGER
name	VARCHAR
address	VARCHAR

12

### O-R Mapping Code for Event

```

Event event = new Event( "Java Days" );
Location ku = new Location( "Kasetsart University" );
ku.setAddress( "90 Pahonyotin Road; Bangkok" );
event.setLocation( ku );
event.setStartDate( new Date(108,Calendar.JULY, 1) );
// save the event
dataMapper.save( event );
    
```

- when we save the event, does dataMapper save the location, too?

13

### O-R Mapping Code for Event

```

// retrieve the event
Event evt = dataMapper.find( "Java Days" );
Location location = evt.getLocation(); // null?
    
```

- when we get the event, does the dataMapper create the location, too?

```

// delete the event
Event evt = dataMapper.find( "Java Days" );
dataMapper.delete( evt );
    
```

- does the dataMapper delete the location, too?
- what if other events (still in database) also refer to this location?

14

### O-R Mapping of 1-to-n Associations

15

### O-R Mapping of 1-to-n Associations

EVENTS		SPEAKERS	
PK id	INTEGER	PK id	INTEGER
name	VARCHAR	name	VARCHAR
start_date	TIMESTAMP	telephone	VARCHAR
FK location_id	INT	FK event_id	INTEGER

16

### O-R Mapping Code for Collections (1)

```

Event event = new Event( "Java Days" );
event.setLocation( ku );
// add event speakers
Speaker gosling = new Speaker( "James Gosling" );
Speaker yuen = new Speaker( "Prof. Yuen" );
event.getSpeakers().add( gosling );
event.getSpeakers().add( yuen );
// save the event
dataMapper.save( event );
    
```

Issues:

- same issues as many-to-1 association

17

### O-R Mapping Code for Collections (2)

```

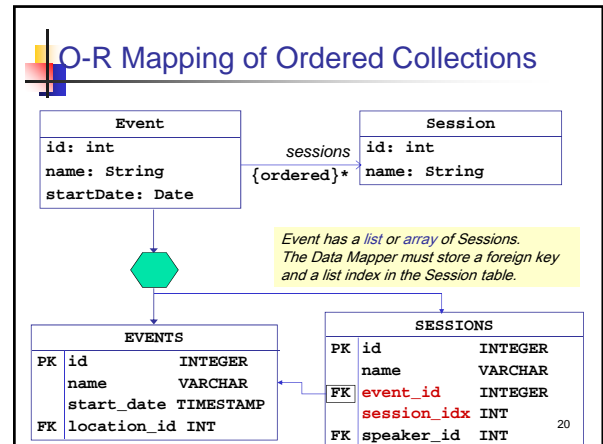
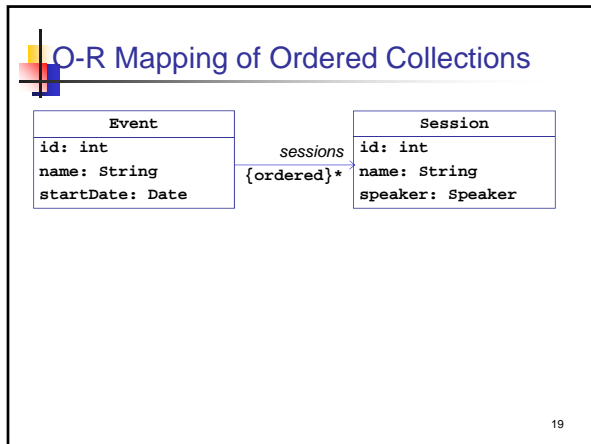
// retrieve the event
Event evt = dataMapper.find( "Java Days" );
Set speakers = evt.getSpeakers();
out.println( "Speakers for " + evt.getName() );
for( Speaker spkr : speakers ) out.print( spkr.getName() );
    
```

- what kind of collection does dataMapper return?
- can we use any collection we want in the Event class?

```

public class Event {
    private Set speakers = new ____; // ? what kind of collection ?
    public setSpeakers( Set speakers ) { this.speakers = speakers; }
}
    
```

18

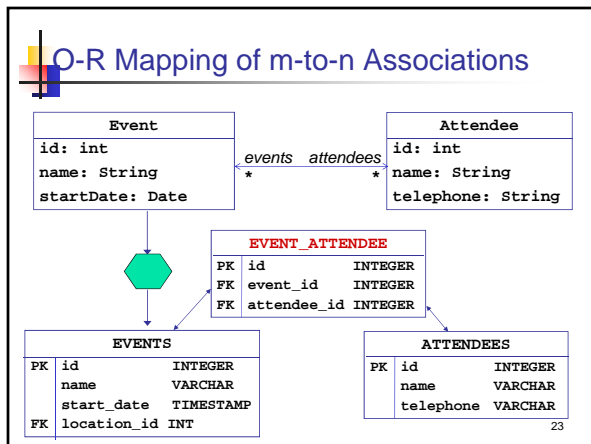
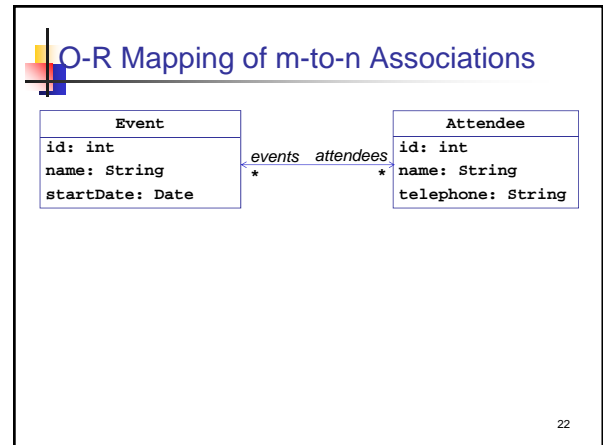


### O-R Mapping Code for a List

```

// add sessions to the event we already saved
Event event = dataMapper.find( "Java Days" );
Speaker gosling = dataMapper.find( "James Gosling" );
Session opening = new Session( "Opening Ceremony" );
opening.setSpeaker( gosling );
// make opening be the 1st session (sessions is a List)
event.getSessions().add( 0, opening );
... add more sessions ...
// update the event
dataMapper.update( event );
    
```

- does dataMapper use the existing data for gosling in the new Session?
- what if our update *changes the indices* of other objects in the list?



### Design of a Data Mapper

Problem:

- What behavior do we need for a data mapper?
- What operations should it perform?

## Object-Relational Operations: CRUD

Common O-R operations are:

- C**reate - save (persist) a new object in the database
- R**etrieve an object from the database
- U**ppdate data for an object already saved in database
- D**elate object's data from the database

25

## Design Model for Data Mapper

```

classDiagram
    class DataMapper {
        load(id) : T
        find(query : String) : T[*]
        findAll() : T[*]
        save(object : T)
        update(object : T)
        delete(object : T)
    }
    class T
    DataMapper --> T : A UML Type Parameter
    
```

The method to "load" an Object by its identifier is sometimes named:

- `load( id )` the Hibernate method name and Spring name
- `find( id ), findById( id )`
- `get( id )` similar to `load( id )` but no exception if id is not found

26

## A Data Mapping for Event Class

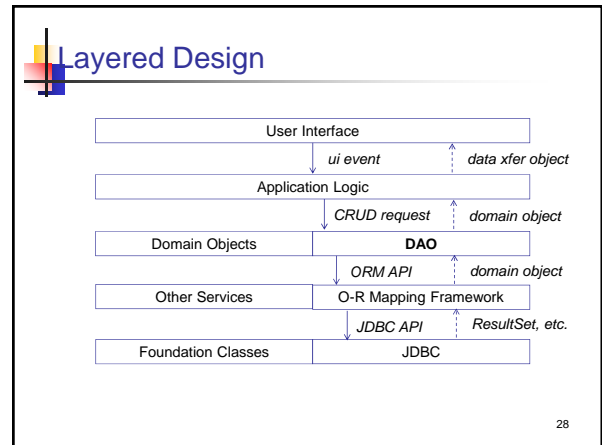
Data Mapper is also called "Data Access Object" (DAO).

- ▣ Hibernate uses the term data access object.
- ▣ We use DAO in data mapper names, e.g. `EventDao`.

```

classDiagram
    class EventDao {
        findById(id : int) : Event
        find(query : String) : Event[*]
        save(evt : Event)
        update(evt : Event)
        delete(evt : Event)
    }
    
```

27



## What's Next?

The Choices:

1. Write the OR Mapping yourself using Java and JDBC
  - ▣ SQL Fundamentals
  - ▣ JDBC Fundamentals
  - ▣ Configure Database
2. Use an existing O-R Framework
  - ▣ Compare O-R frameworks
  - ▣ Learn to use one framework
  - ▣ Configure Database

29

## Transactions

- ▣ These frameworks generally go beyond CRUD
- ▣ They also allow something called a "transaction"
  - `begin_xtn()`;
    - operations on the database
  - `commit_xtn()` or `abort_xtn()`;
- ▣ Has **all or nothing** guarantees.
- ▣ And they support locking for concurrency control.

30

## Persistence Frameworks

- **Hibernate** - widely used open-source persistence framework for Java. Persistence of POJOs, uses mapping files and object-query language to decouple Java from database. **NHibernate** for .Net languages.
- **iBatis** - simple, uses SQL maps. Database schema not transparent to Java code.
- **Entity Enterprise JavaBeans** - uses EJB container services to perform persistence. Resource hog.
- **Cayenne** - Apache project, has GUI modeler that eliminates need to write xml files. Can reverse engineer database or generate database schema & Java code.
- **TopLink** (Oracle), **Torque** (Apache DB), **Castor**, ...

31

## Standards and APIs

- **Java Data Objects (JDO)** - transparent persistence of POJOs; defines query language (JDOQL) and standard for XML descriptors.
  - implementations: **Kodo**, **JPOX**
- **Java Persistence API (JPA)** - part of the EJB 3.0, defines OR standard, query language (JPQL), and standalone POJO or EJB server-based persistence.
  - implementations: **TopLink Essentials** (Glassfish project), **OpenJPA**. **Hibernate** is JPA compliant.

Article: [Adopting a Java Persistence Framework](http://today.java.net/pub/a/today/2007/12/18/adopting-java-persistence-framework.html),  
<http://today.java.net/pub/a/today/2007/12/18/adopting-java-persistence-framework.html>

32