

Mapping and Managing Entities



Antonio Goncalves

JAVA CHAMPION

@agoncal www.antoniogoncalves.org



Previous Module



Object-relational mapping tool

Object-programming languages

Relational databases

JPA is the standard specification

What is an entity?

Where to use it?

Overview



Managing entities

Persistence unit

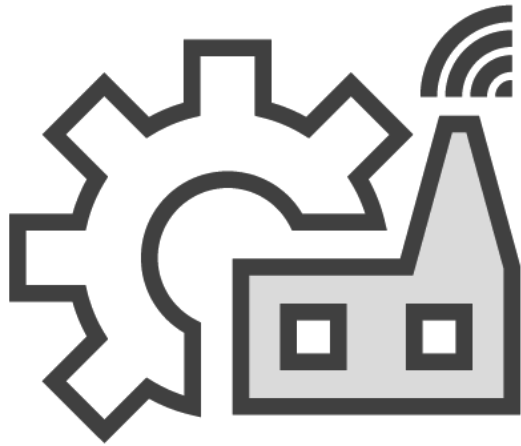
Mapping entities

- Annotations
- XML

Unit testing



Managing Entities



Entity manager

- EntityManagerFactory in Java SE
- Injected in Java EE

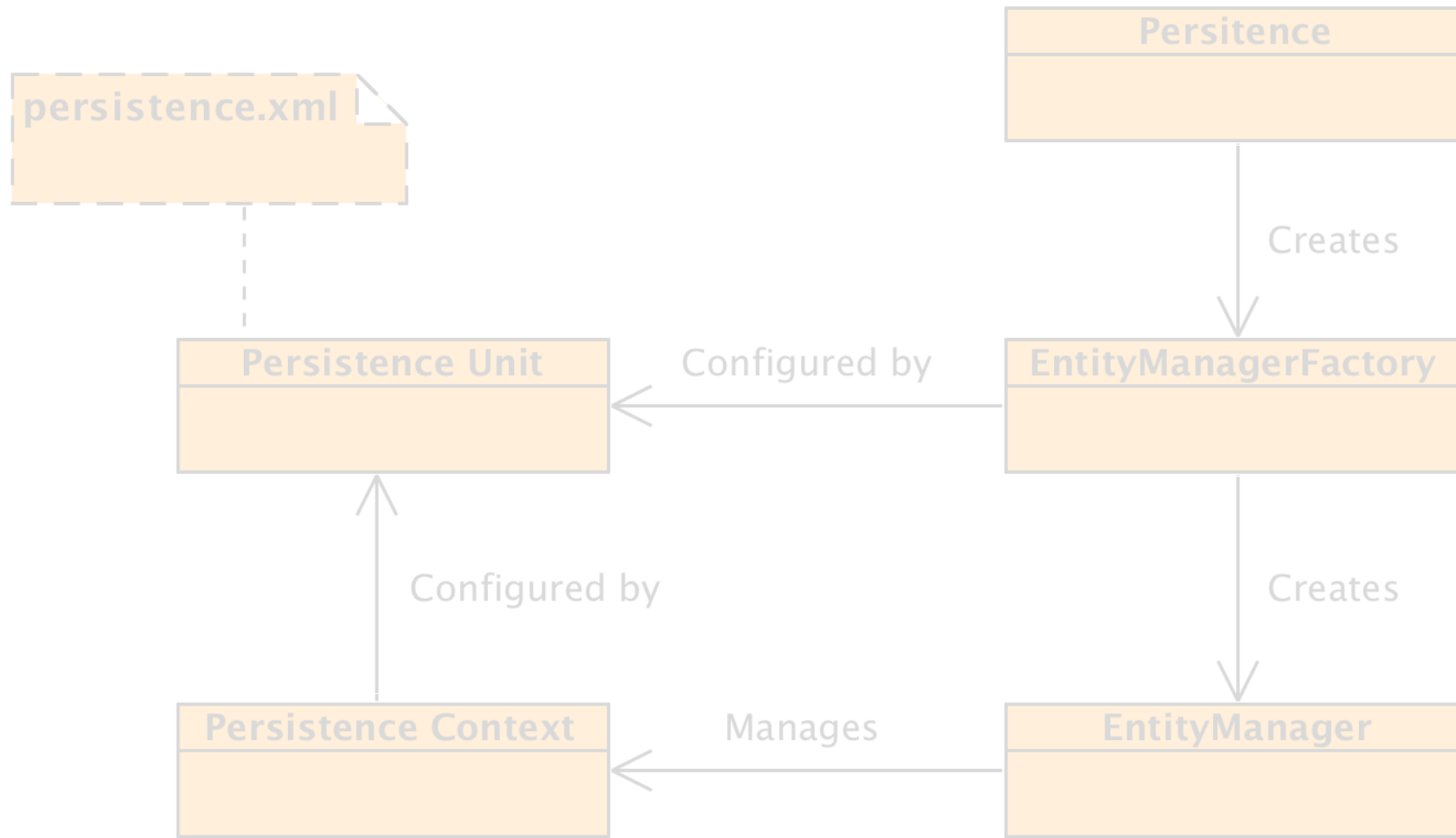
Persistence context

- Set of managed entity

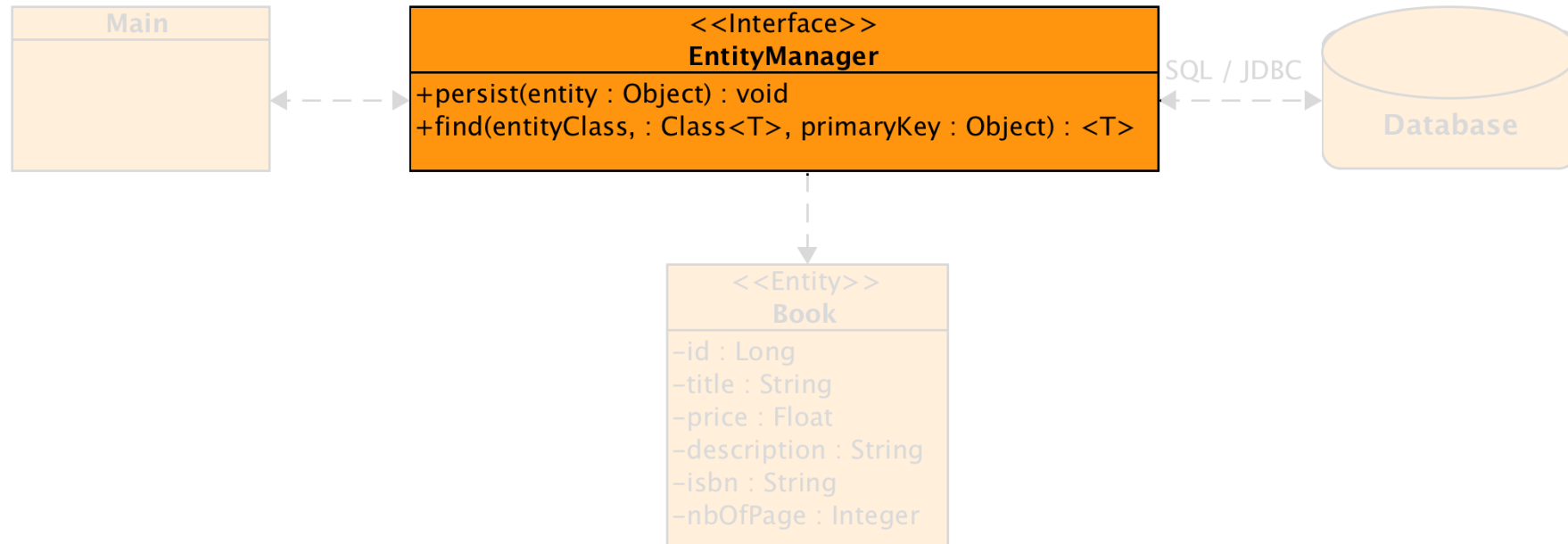
Persistence unit

- `persistence.xml`

Managing Entities



Entity Manager



Persistence Context

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>  
<persistence (...) version="2.2">
```

```
  <persistence-unit name="myPU" transaction-type="RESOURCE_LOCAL">  
    <class>com.pluralsight.model.Book</class>  
    <class>com.pluralsight.model.CD</class>
```

```
  </persistence-unit>  
</persistence>
```



Configuration in persistence.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<persistence (...) version="2.2">

  <persistence-unit name="myPU" transaction-type="RESOURCE_LOCAL">
    <class>com.pluralsight.model.Book</class>
    <class>com.pluralsight.model.CD</class>
    <properties>
      <property name="javax.persistence.jdbc.driver"
        value="org.h2.Driver" />
      <property name="javax.persistence.jdbc.url"
        value="jdbc:h2:mem:myDB" />
      <property name="javax.persistence.jdbc.user" value="app" />
      <property name="javax.persistence.jdbc.password" value="app" />
    </properties>
  </persistence-unit>
</persistence>
```



Properties in persistence.xml

```
<property name="javax.persistence.
```

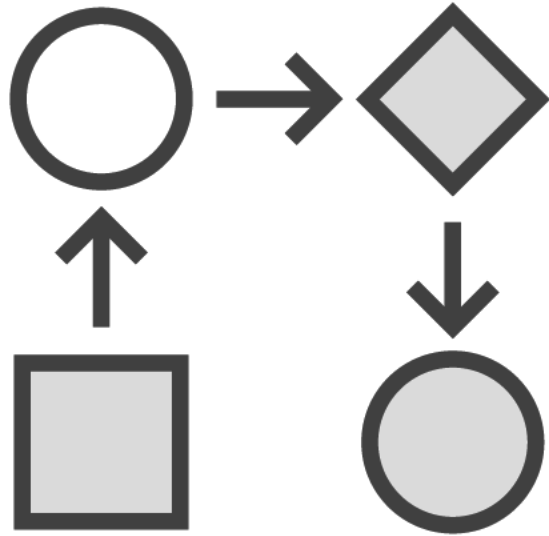


Properties in persistence.xml

```
<property name="javax.persistence.database-product-name" value="H2" />  
  
<property name="javax.persistence.schema-generation.database.action"  
    value="drop-and-create" />  
  
<property name="javax.persistence.schema-generation.scripts.action"  
    value="drop-and-create" />  
  
<property name="javax.persistence.schema-generation.scripts.create-target"  
    value="create.ddl" />  
  
<property name="javax.persistence.schema-generation.scripts.drop-target"  
    value="drop.ddl" />  
  
<property name="eclipselink.logging.level" value="INFO" />
```



CRUD Operations on an Entity



CRUD

- Create
- Read
- Update
- Delete

EntityManager

- persist
- find
- remove

A Book Entity

@Entity

```
public class Book {
```

@Id

```
private Long id;
```

```
private String title;
```

```
private String description;
```

```
private Float unitCost;
```

```
// Constructors, getters & setters
```

```
}
```



Persisting an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public Book createBook(Long id, String title, String desc, Float cost) {  
        Book book = new Book();  
        book.setId(id);  
        book.setTitle(title);  
        book.setDescription(desc);  
        book.setUnitCost(cost);  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
        return book;  
    }  
}
```



Persisting an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public Book createBook(Book book) {  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
        return book;  
    }  
  
}
```



Finding by ID

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public Book findBook(Long id) {  
        return em.find(Book.class, id);  
    }  
  
}
```



Removing an Entity by ID

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public void removeBook(Long id) {  
        Book book = em.find(Book.class, id);  
        if (book != null) {  
            tx.begin();  
            em.remove(book);  
            tx.commit();  
        }  
    }  
}
```



Removing an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public void removeBook(Book book) {  
        Book bookToBeDeleted = em.merge(book);  
        tx.begin();  
        em.remove(bookToBeDeleted);  
        tx.commit();  
    }  
  
}
```



Removing an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public void removeBook(Book book) {  
  
        tx.begin();  
        em.remove(em.merge(book));  
        tx.commit();  
    }  
  
}
```



Updating an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public Book raiseUnitCost(Long id, Float raise) {  
        Book book = em.find(Book.class, id);  
        if (book != null) {  
            tx.begin();  
            book.setUnitCost(book.getUnitCost() + raise);  
            tx.commit();  
        }  
        return book;  
    }  
}
```

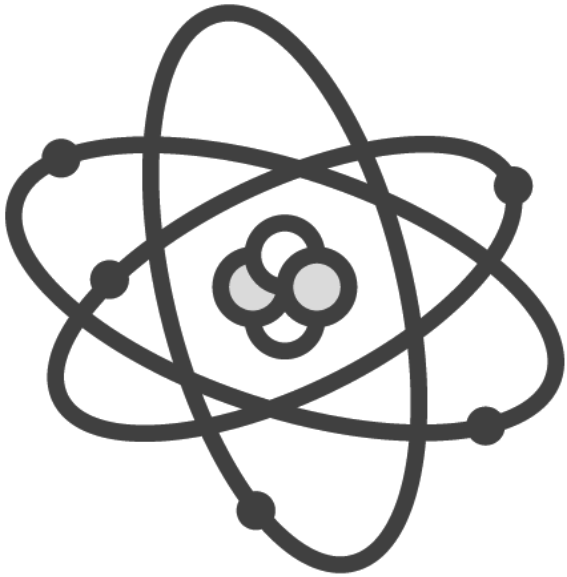


Updating an Entity

```
public class BookService {  
  
    private EntityManagerFactory emf =  
        Persistence.createEntityManagerFactory("myPU");  
    private EntityManager em = emf.createEntityManager();  
    private EntityTransaction tx = em.getTransaction();  
  
    public Book raiseUnitCost(Book book, Float raise) {  
        Book bookToBeUpdated = em.merge(book);  
        tx.begin();  
        bookToBeUpdated.setUnitCost(bookToBeUpdated.getUnitCost() + raise);  
        tx.commit();  
        return book;  
    }  
  
}
```



Transactions



Consistent state

Changes succeed or fail atomically

Transactional operations

- Persist
- Remove
- Update

Find can be non-transactional

EntityTransaction

Demo



Book entity

Book service

Entity manager

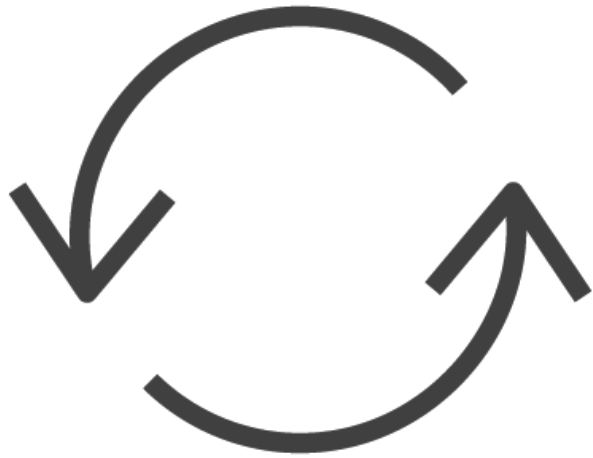
CRUD operations

Database structure

Transaction management



Default Entity Mapping



`@javax.persistence.Entity`

`@javax.persistence.Id`

No-arg constructor

Concrete or abstract class

Enums or interfaces cannot be entities

All attributes become persistent

Configuration by Exception

@Entity

```
public class Author {
```

@Id

```
private Long id;
```

```
private String firstName;
```

```
private String lastName;
```

```
private String bio;
```

```
private LocalDate dateOfBirth;
```

```
private Integer age;
```

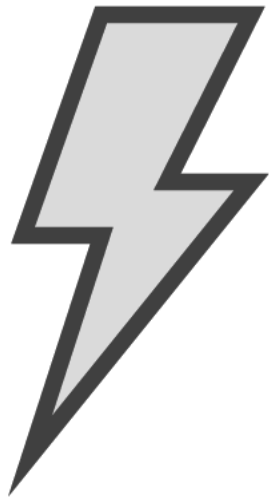
```
private Language language;
```

```
// Constructors, getters & setters
```

```
}
```



Configuration by Exception



I.e. convention over configuration

JPA provider applies the default rules








Entity name ► Table name

Attribute name ► Column name

JDBC rules for mapping primitives

- String ► VARCHAR(255)
- Long ► BIGINT
- Boolean ► SMALLINT
- ...

Database Representation

| AUTHOR | | | |
|---|-------------|--------------|---|
|  | ID | bigint | |
|  | AGE | integer | N |
|  | BIO | varchar(255) | N |
|  | DATEOFBIRTH | date | N |
|  | FIRSTNAME | varchar(255) | N |
|  | LANGUAGE | integer | N |
|  | LASTNAME | varchar(255) | N |

Customize Mapping with Metadata

@Entity

```
public class Author {  
  
    @Id  
    private Long id;  
  
    private String firstName;  
  
    private String lastName;  
  
    private String bio;  
  
    private LocalDate dateOfBirth;  
  
    private Integer age;  
  
    private Language language;  
}
```



Customizing Table

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Table

```
@Entity
@Table(name = "T_AUTHOR", catalog = "CAT", schema = "APP")
public class Author {

    @Id
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue(strategy = GenerationType.TABLE)
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Generated Value

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;

    private String firstName;

    private String lastName;

    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Column

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", nullable = true)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Temporal Types

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Legacy Temporal Types

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;
    @Temporal(TemporalType.DATE)
    private Date dateOfBirth;

    private Integer age;

    private Language language;
```



Legacy Temporal Types

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;
    @Temporal(TemporalType.TIME)
    private Date dateOfBirth;

    private Integer age;

    private Language language;
```



Legacy Temporal Types

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;
    @Temporal(TemporalType.TIMESTAMP)
    private Date dateOfBirth;

    private Integer age;

    private Language language;
```



Legacy Temporal Types

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;

    private Integer age;

    private Language language;
```



Customizing Transient State

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;
    @Transient
    private Integer age;

    private Language language;
```



Customizing Enumeration

```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;
    @Transient
    private Integer age;
    @Enumerated(EnumType.ORDINAL)
    private Language language;
```



Customizing Enumeration








```
@Entity
@Table(name = "T_AUTHOR")
public class Author {

    @Id @GeneratedValue
    private Long id;
    @Column(name = "first_name", length = 50)
    private String firstName;
    @Column(name = "last_name", length = 50)
    private String lastName;
    @Column(length = 5000)
    private String bio;

    private LocalDate dateOfBirth;
    @Transient
    private Integer age;
    @Enumerated(EnumType.STRING)
    private Language language;
```









Default Database Representation

| AUTHOR | | | |
|---|-------------|--------------|---|
|  | ID | bigint | |
|  | AGE | integer | N |
|  | BIO | varchar(255) | N |
|  | DATEOFBIRTH | date | N |
|  | FIRSTNAME | varchar(255) | N |
|  | LANGUAGE | integer | N |
|  | LASTNAME | varchar(255) | N |



Default Database Representation

| T_AUTHOR | | | |
|--|---------------|---------------|---|
|  | ID | bigint | |
|  | BIO | varchar(5000) | N |
|  | DATE_OF_BIRTH | date | N |
|  | FIRST_NAME | varchar(50) | N |
|  | LANGUAGE | varchar(255) | N |
|  | LAST_NAME | varchar(255) | |

Demo



Existing author table

Map author entity

Default mapping

JPA annotations

Customize the mapping



Mapping Metadata



Annotations



XML



Annotations on Attributes

```
@Entity
public class CD {

    @Id @GeneratedValue
    private Long id;
    @Column(length = 100)
    private String title;
    @Column(length = 3000)
    private String description;
    @Column(name = "total_duration")
    private Float totalDuration;

    private String genre;

    // Constructors, getters & setters
}
```



Annotations on Getters

```
@Entity
public class CD {

    @Id @GeneratedValue
    public Long getId() {return id;}
    @Column(length = 100)
    public String getTitle() {return title;}
    @Column(length = 3000)
    public String getDescription() {return description;}
    @Column(name = "total_duration")
    public Float getTotalDuration() {return totalDuration;}

    public String getGenre() {return genre;}

    // Attributes and Constructors
}
```



Mapping with XML



XML deployment descriptor

Each annotation has an XML equivalent

XML overrides annotations

When to use annotations over XML?

- Matter of taste
- Metadata really coupled == annotation
- Depending on environment == XML

XML Mapping File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<entity-mappings (...) version="2.2">
  <entity class="com.pluralsight.CD">
    <table name="T_CD" />
    <attributes>
      <basic name="title">
        <column nullable="false" />
      </basic>
      <basic name="description">
        <column length="5000" />
      </basic>
      <basic name="genre">
        <column length="50" />
      </basic>
    </attributes>
  </entity>
</entity-mappings>
```



Referencing in persistence.xml

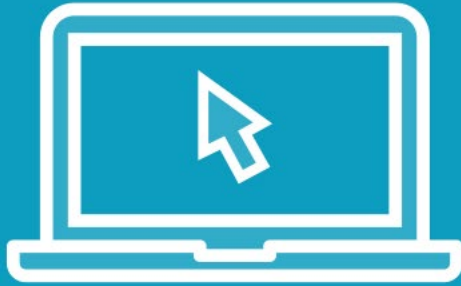
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<persistence (...) version="2.2">
  <persistence-unit name="myPU" transaction-type="RESOURCE_LOCAL">

    <mapping-file>META-INF/cd_mapping.xml</mapping-file>

    <class>com.pluralsight.model.Book</class>
    <class>com.pluralsight.model.CD</class>
    <properties>
      <property name="javax.persistence.jdbc.driver"
        value="org.h2.Driver" />
      <property name="javax.persistence.jdbc.url"
        value="jdbc:h2:mem:myDB" />
      <property name="javax.persistence.jdbc.user" value="app" />
      <property name="javax.persistence.jdbc.password" value="app" />
    </properties>
  </persistence-unit>
</persistence>
```



Demo



CD entity

Annotations on getters

Different environments

Different persistent units

External XML mapping file



Summary



EntityManager, PersistenceContext,
PersistentUnit

CRUD operations

Default mapping

Annotation metadata

XML metadata

Use JPA in Java SE

Unit testing



Next Module



Relationships and inheritance

Cardinality and direction

Cascading events

Fetching

Inheritance mapping strategies

