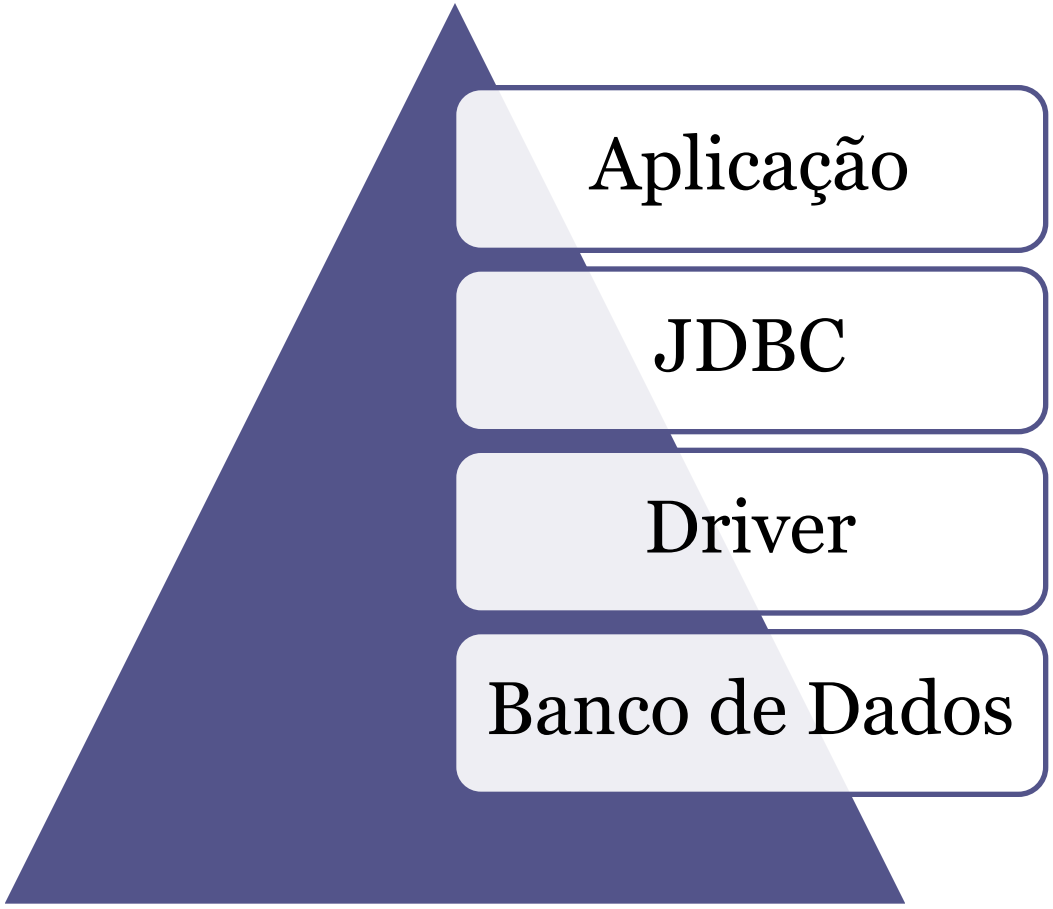
JPA – Java Persistence API

Prof. Ramon Chiara

JDBC

- Java DataBase Connectivity
- Acesso a bancos de dados
- “Independência” de banco de dados

JDBC



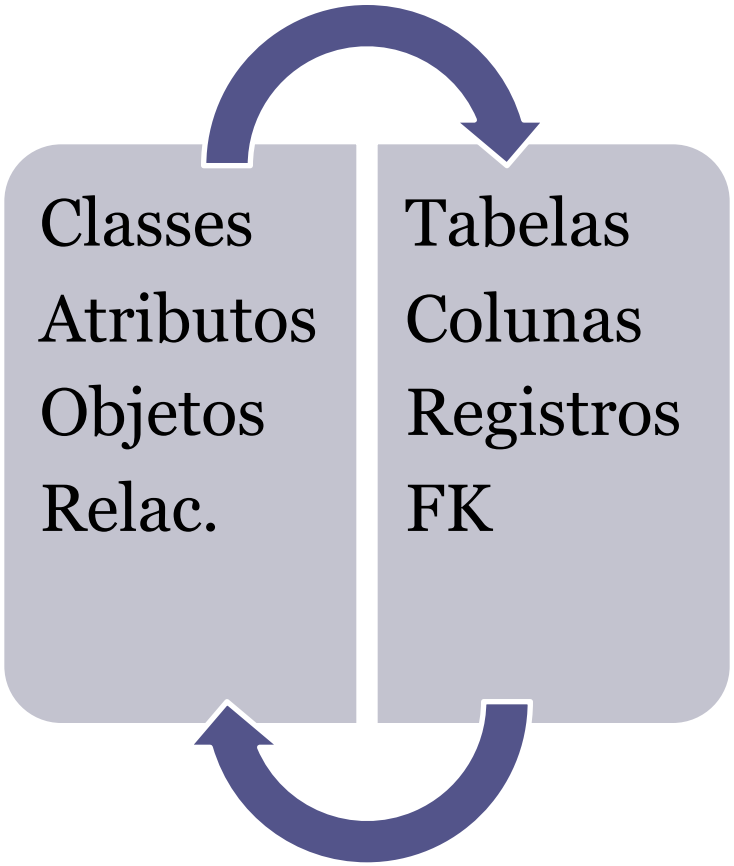
JDBC

- Java DataBase Connectivity
- Acesso a bancos de dados
- “Independência” de banco de dados
- Dificuldade?
  - ...
  - Passar atributos para campos
  - Passar campos para atributos

ORM

- Object-Relational Mapper

ORM


Classes
Atributos
Objetos
Relac.
Tabelas
Colunas
Registros
FK

ORM

- Object-Relational Mapper
- Implementações
  - JDO – Java DataBase Objects
  - Hibernate
  - EclipseLink
  - ...

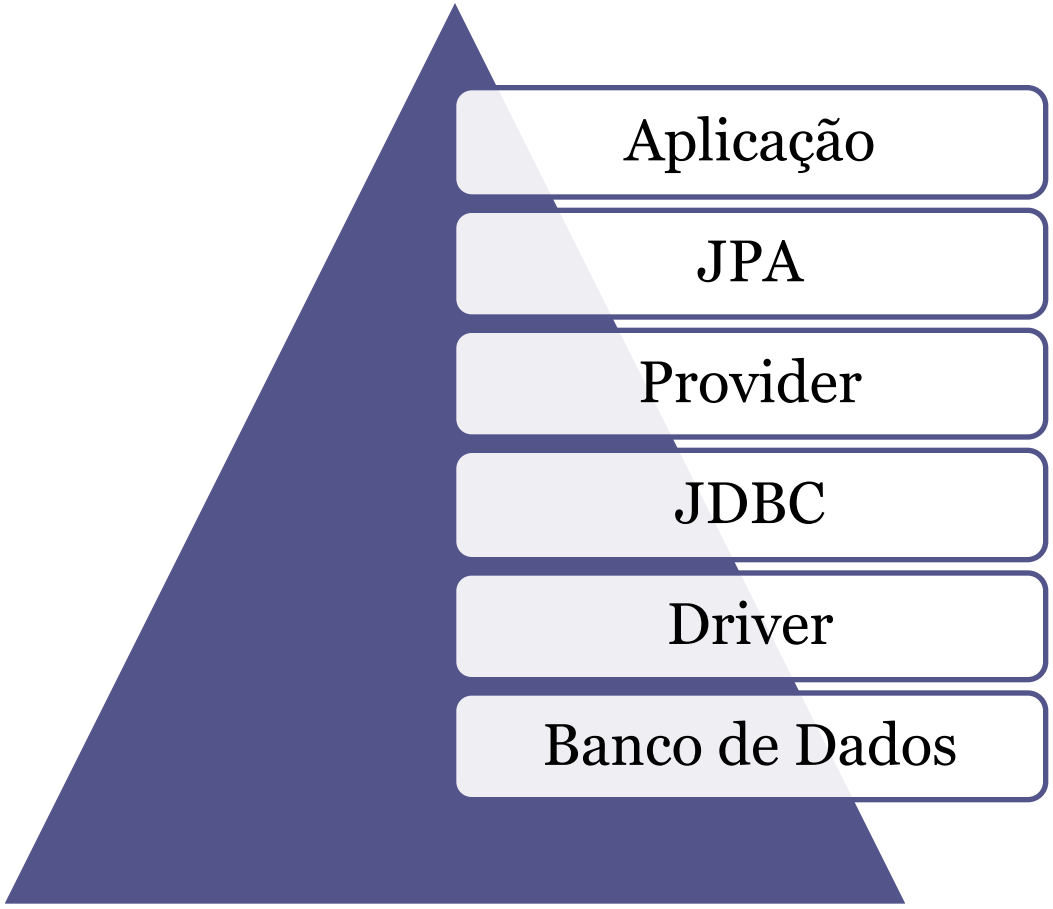
ORM

- Object-Relational Mapper
- Implementações
  - JDO – Java DataBase Objects
  - Hibernate
  - EclipseLink
  - ...
- Bagunça
  - Especificação

JPA

- Java Persistence API
  - Especificação

JPA



JPA

- Java Persistence API
  - Especificação
- Provider?
  - Implementações da Especificação:
    - Hibernate
    - EclipseLink
    - ...

JPA – Estrutura

META-INF/persistence.xml

- Unidade de Persistência (Persistence Unit)
- Entidades ←→ Banco de Dados

EntityManagerFactory emf =

- Persistence.createEntityManagerFactory("PU");

EntityManager em =

- emf.createEntityManager();

JPA – Estrutura

EntityTransaction tx =

- em.getTransaction();

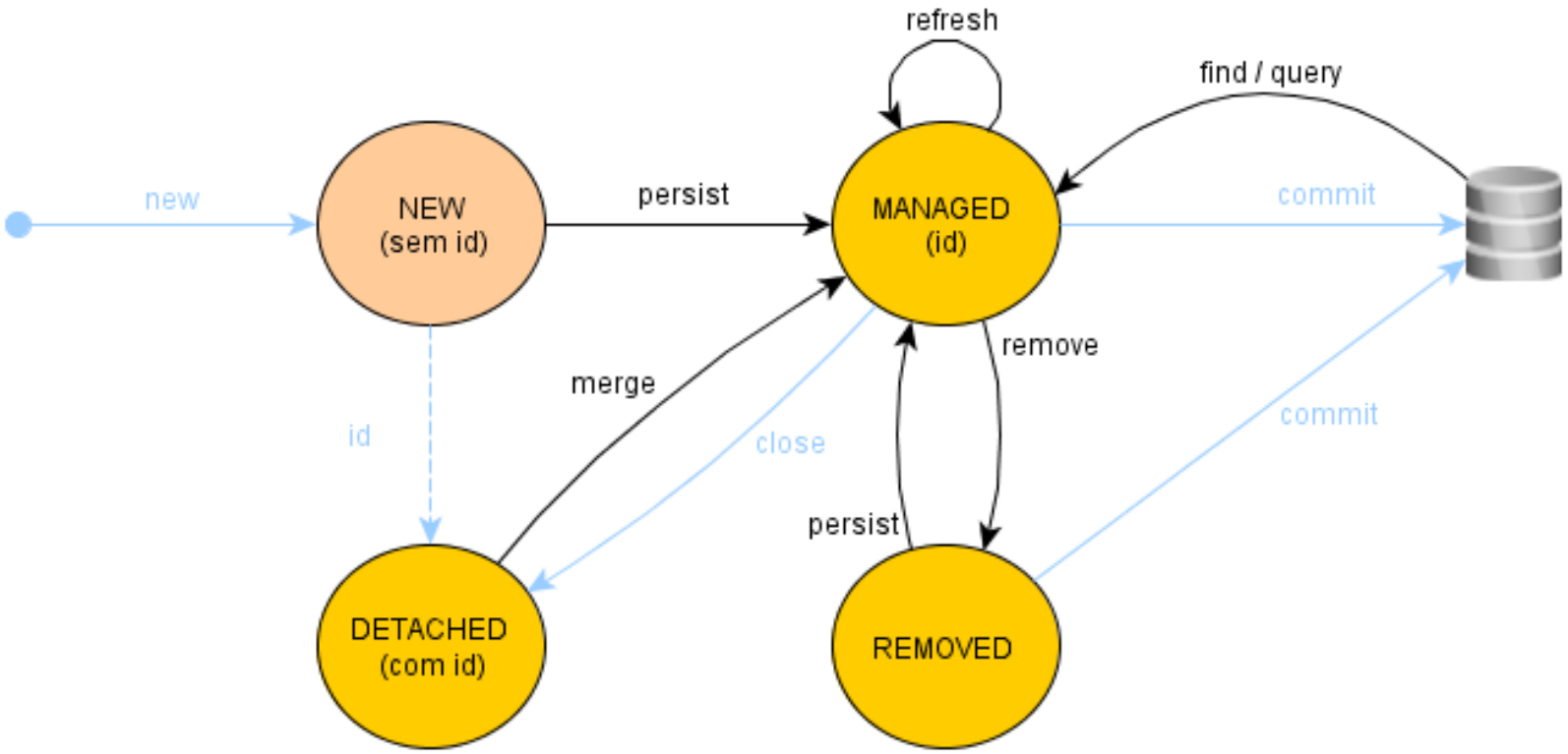
```
try {
    tx.begin();
    ...
    tx.commit();
}
```

```
catch (Exception ex) {
    tx.rollback();
}
```

JPA – Estrutura

```
finally {
    em.close();
    emf.close();
}
```

JPA – Ciclo de Vida



JPA – Ciclo de Vida

- Insert → em.persist(e)
- Select → e = em.find(E.class, id)
- Select → TypedQuery<E> q = em.createQuery("JPQL", E.class)
  - q.setParameter("p", valor)
  - List<E> l = q.getResultList()
  - e = q.getSingleResult()
    - NoResultException
    - NonUniqueResultException
- Update → e.setXXX(...)
- Delete → em.remove(e)

- e = em.merge(e)
- em.refresh(e)

JPA – Estrutura

META-INF/persistence.xml

- Unidade de Persistência (Persistence Unit)
- Entidades ←→ Banco de Dados

EntityManagerFactory emf =

- Persistence.createEntityManagerFactory("PU");

EntityManager em =

- emf.createEntityManager();

JPA – Entidades

- Obrigatório
  - @Entity
  - @Id
  - Construtor sem parâmetros
- Recomendável
  - implements Serializable

JPA – Entidades

- @Entity
- @Table(name="")

- @Id
- @GeneratedValue(strategy = GerationType.IDENTITY)

- @Column(name="", nullable=false, unique=false)
- @JoinColumn(name="", nullable=false, unique=false)

- @ManyToOne
- @OneToMany
- @ManyToMany
- @OneToOne

- ...

JPA

- Show me the code!

JPA – Show me the code!

1. Fazer o download do Driver JDBC (MySQL)
   - http://dev.mysql.com/downloads/connector/j/
2. Fazer o download de um JPA Provider (Hibernate)
   - http://hibernate.org/orm/
3. Descompactar os downloads
4. Criar um projeto Java
5. Configurar o Class Path (Build Path no Eclipse)
   - mysql/*.jar
   - hibernate/lib/jpa/*.jar
   - hibernate/lib/required/*.jar

Não esqueça de criar um banco de dados com usuário e senha.

JPA – Show me the code!

6. Criar o arquivo META-INF/persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
    <persistence-unit name="PU" transaction-type="RESOURCE_LOCAL">
        <provider>org.hibernate.ejb.HibernatePersistence</provider>
        <properties>
            <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
            <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/database"/>
            <property name="javax.persistence.jdbc.user" value="usuario"/>
            <property name="javax.persistence.jdbc.password" value="senha"/>
            <property name="javax.persistence.schema-generation.database.action" value="create"/>
            <property name="hibernate.show_sql" value="true"/>
            <property name="hibernate.format_sql" value="true"/>
        </properties>
    </persistence-unit>
</persistence>
```

Configurar com o banco de dados, usuário e senha criados no slide anterior.

JPA – Show me the code!

7. Criar entidades. Exemplo:

```
@Entity
public class Aluno implements Serializable {
    @Id
    private String nome;
    private double p1;
    private double p2;

    protected Aluno() {}

    public Aluno(String nome, double p1, double p2) {
        this.nome = nome;
        this.p1 = p1;
        this.p2 = p2;
    }

    public String getNome() { return nome; }
    public void setNome(String nome) { this.nome = nome; }
    public double getP1() { return p1; }
    public void setP1(double p1) { this.p1 = p1; }
    public double getP2() { return p2; }
    public void setP2(double p2) { this.p2 = p2; }

    public double getMedia() { return (p1 + p2) / 2; }
    public boolean getPassou() { return getMedia() >= 7; }
}
```

Cuidado com os imports!
Usar:
import javax.persistence.*;

JPA – Show me the code!

8. Criar um programa. Exemplo:

```
public class Program {
    public static void main(String[] args) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("PU");
        EntityManager em = emf.createEntityManager();
        EntityTransaction tx = em.getTransaction();
        try {
            tx.begin();

            Aluno u = new Aluno("João", 8, 5.5);
            em.persist(u);

            tx.commit();
        }
        catch (Exception e) {
            tx.rollback();
        }
        finally {
            em.close();
            emf.close();
        }
    }
}
```

Cuidado com os imports!
Usar:
import javax.persistence.*;

JPA – Show me the code!

9. Verificar no MySQL (ou no banco usado) que deu certo!!! ☺

Thats's all folks!