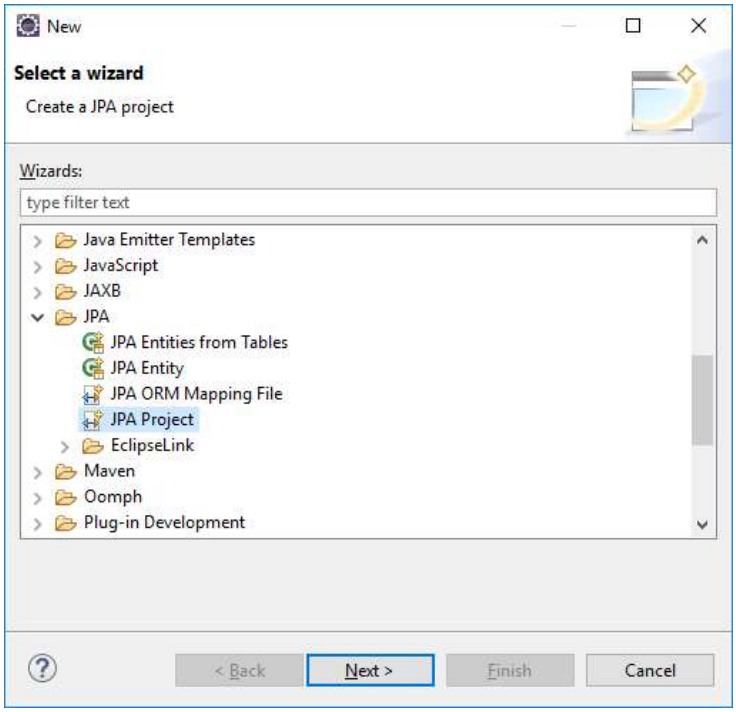
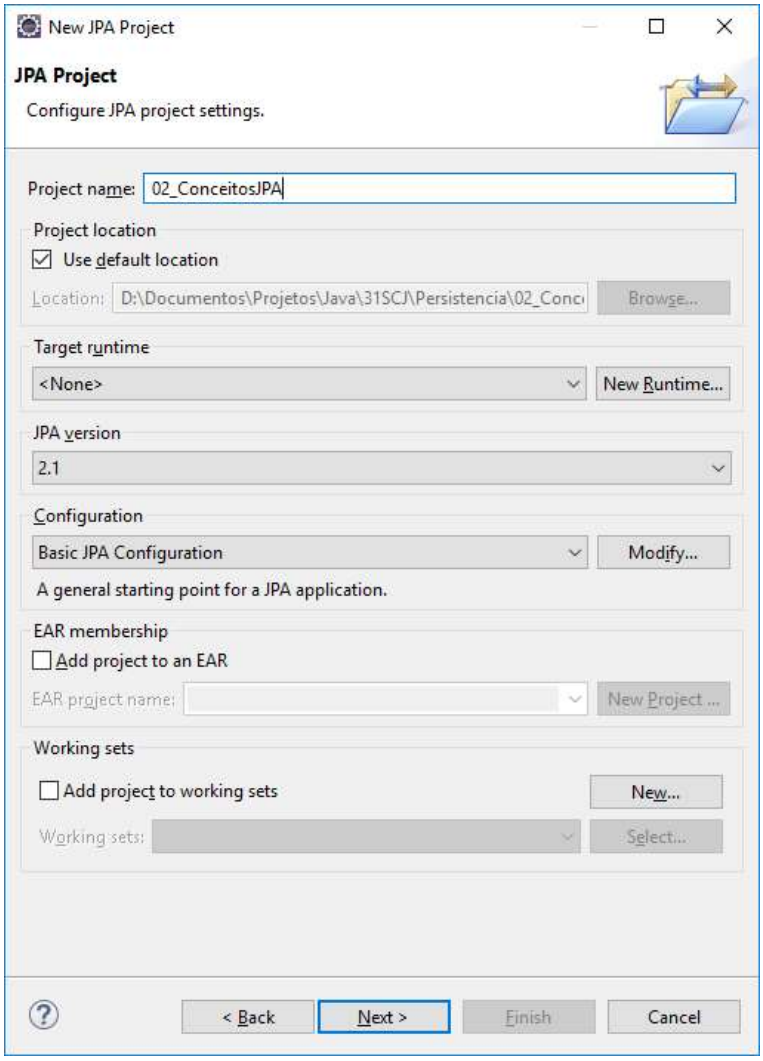
Roteiro para desenvolvimento da aula sobre JPA - Conceitos

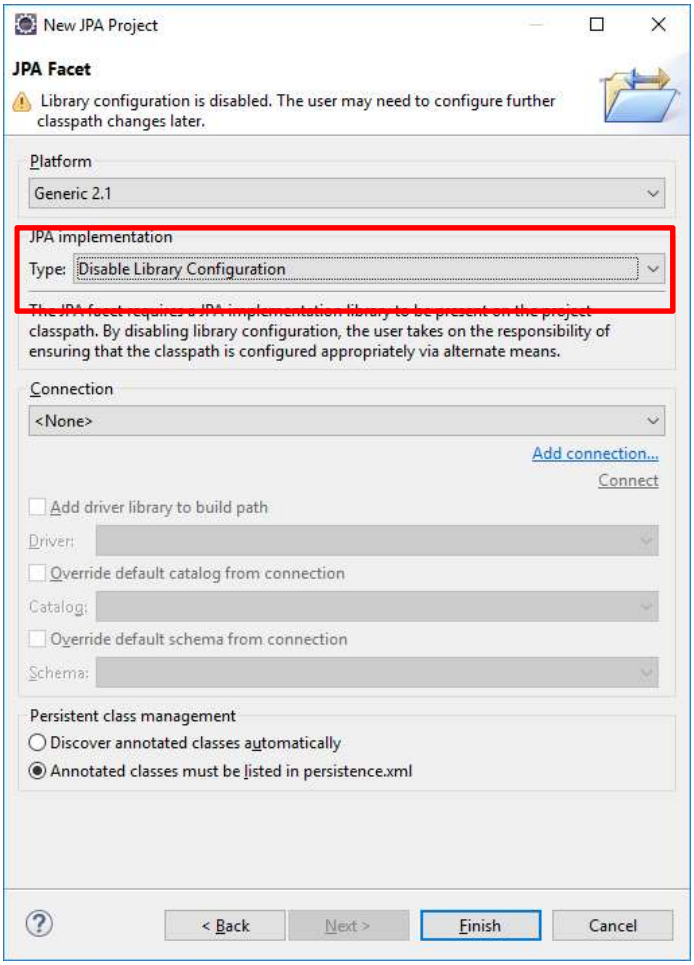
Neste roteiro desenvolveremos uma aplicação baseada em anotações JPA. Os passos são apresentados a seguir:

1. Criar um projeto **JPA Project** chamado **02_ConceitosJPA.**





2. Avance até a última etapa, e selecione a opção **Disable Library Configuration**.



Se for usar a API local:

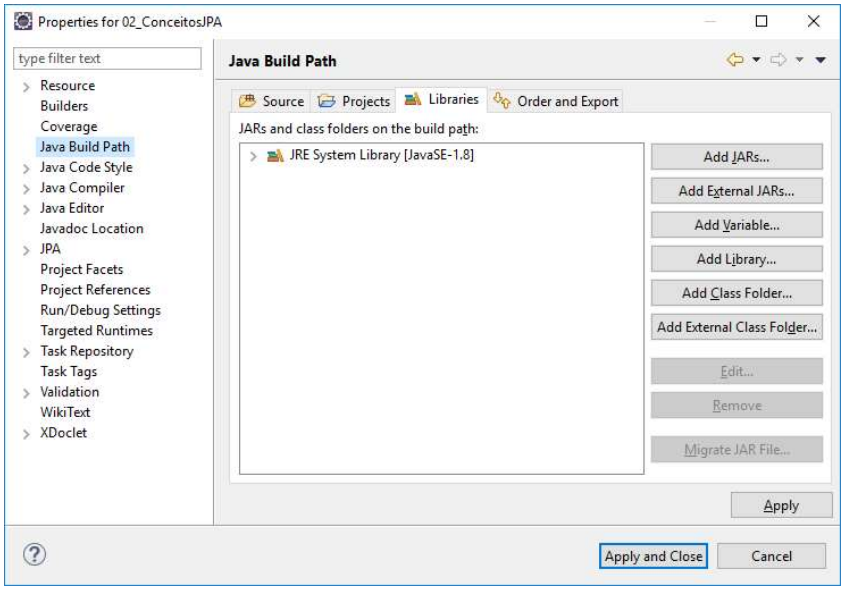
3. Obter a API do **Hibernate**, no link: **http://sourceforge.net/projects/hibernate/files/hibernate4/**

4. Descompacte o arquivo .zip em uma pasta da sua escolha.

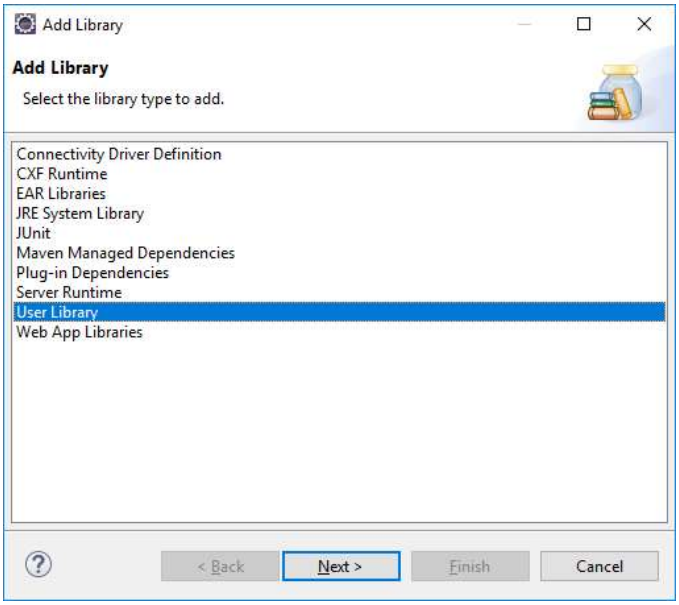
5. Usaremos os jars das pastas **required** e **jpa** do diretório **lib**.

6. **Opcionalmente, utilize a API disponibilizada no portal da Fiap**.

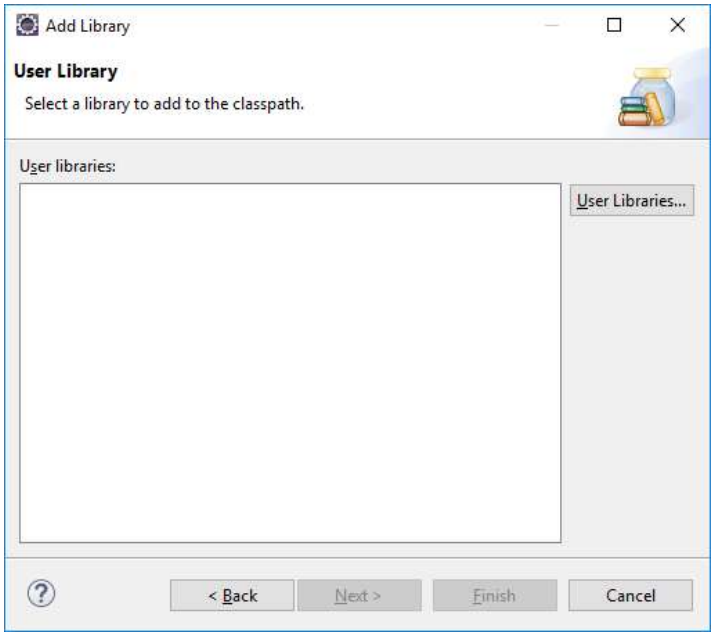
7. Selecione o projeto, clique com o botão direito do mouse, selecione **Properties** e, em seguida, **Java Build Path**.



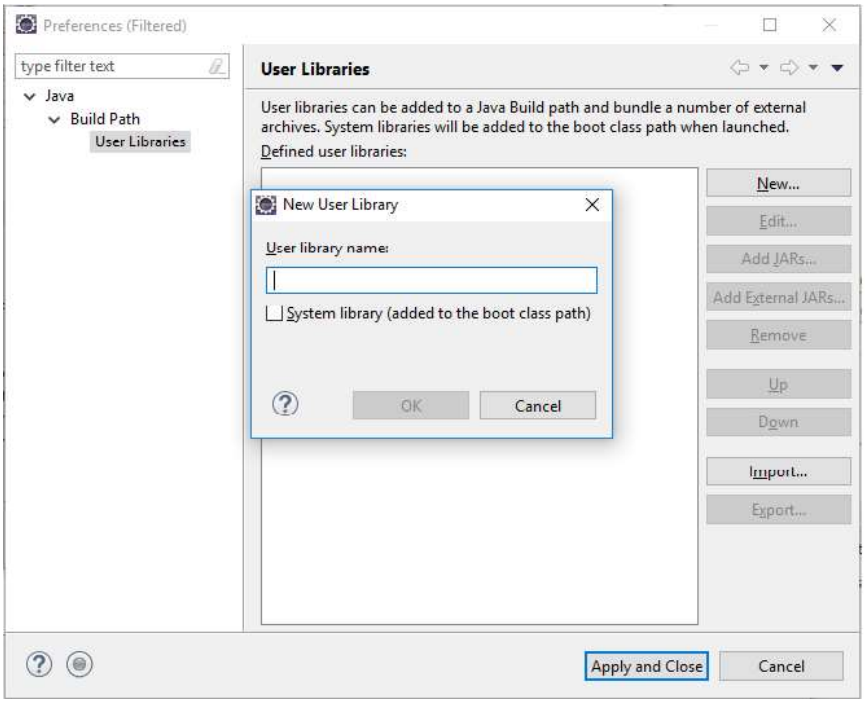
8. Clique em **Add Library**. Selecione **User Library**.



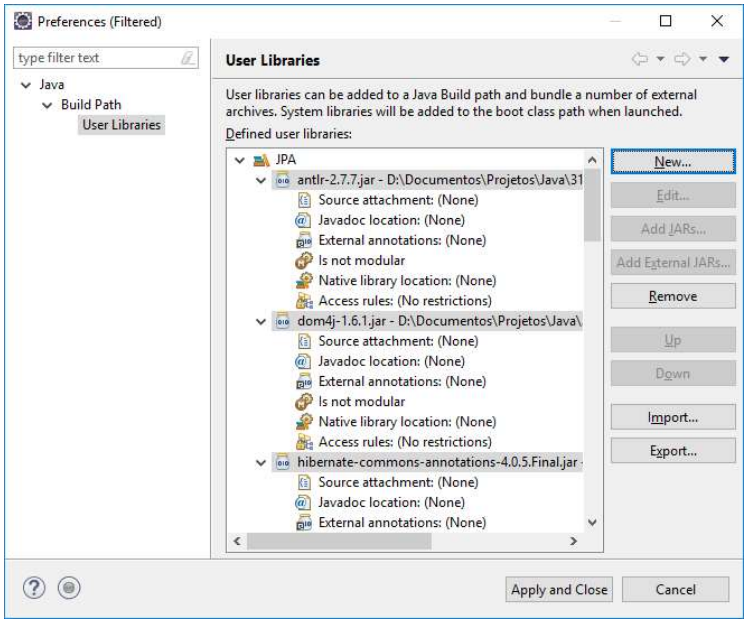
9. Na tela seguinte, selecione o botão **User Libraries**:



10. Na próxima tela, selecione new e, em seguida, forneça um nome (sugestão: **JPA**)



11. No próximo passo deveremos adicionar as bibliotecas do JPA. Clique em Add External Jars, buscando pelo local onde as bibliotecas estão disponibilizadas. O resultado deve ser semelhante a:

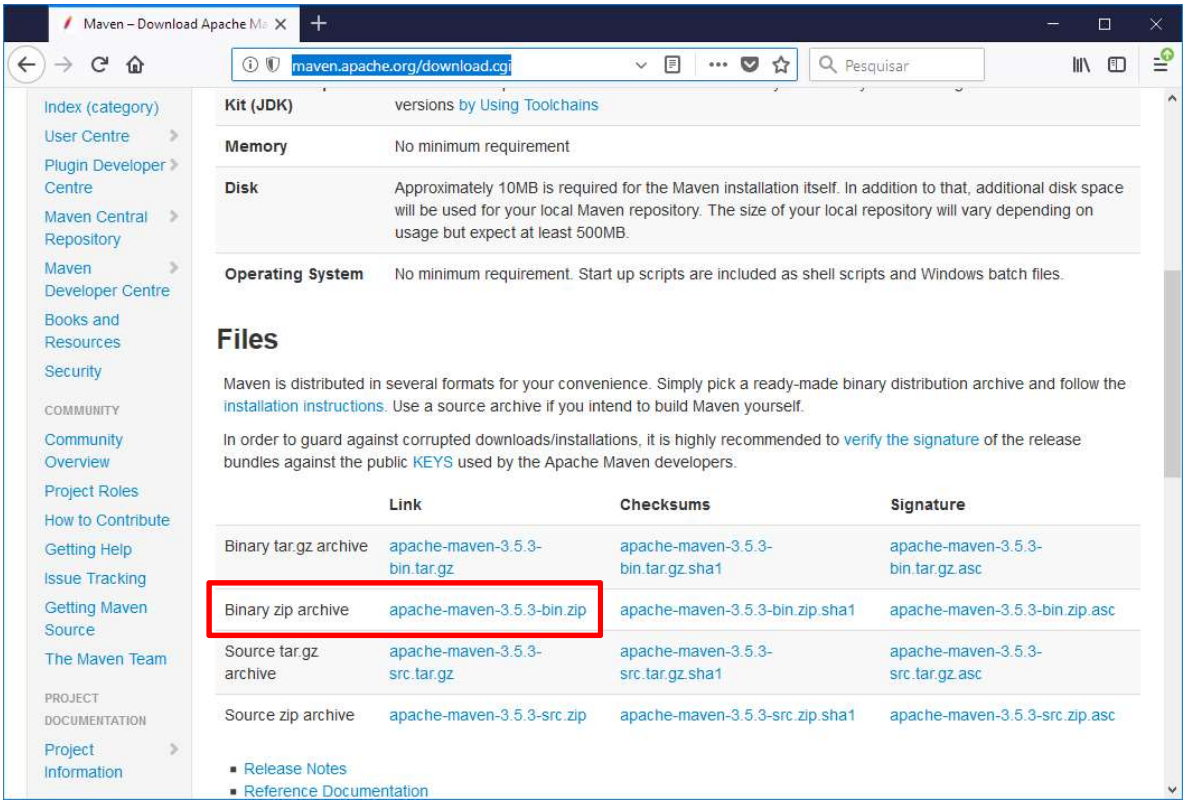


12. Fechar as janelas, e verificar que a API foi adicionada ao projeto

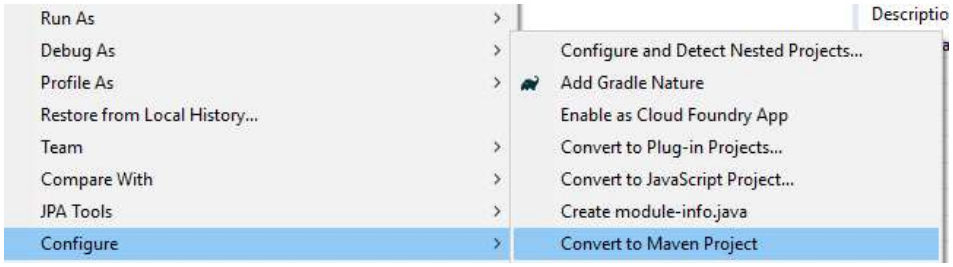
Usando o Maven:

13. Certificar-se de que o maven esteja instalado e incluído no path do seu sistema. O maven pode ser obtido no link: http://maven.apache.org/download.cgi

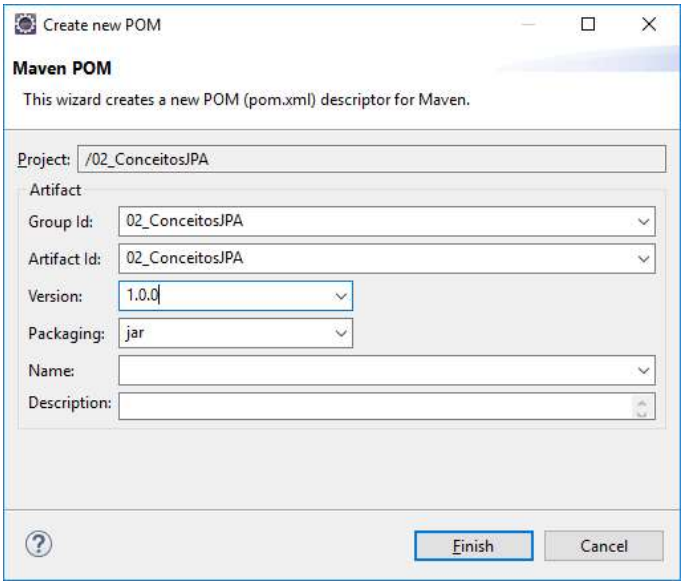
14. Fazer o download e descompactar o arquivo mostrado:



15. Clicar com o botão direito do mouse no projeto, selecione **Configure** e, em seguida, **Convert to Maven Project**.



16. Configurar as opções como mostrado abaixo.



17. Este procedimento criará o arquivo **pom.xml**, onde incluímos as dependências.

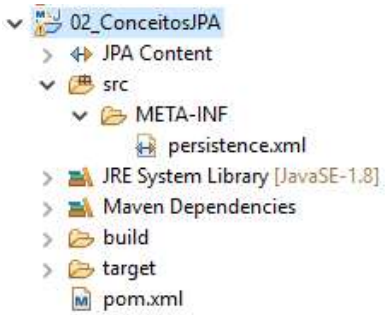
18. Acessar o site: mvnrepository.com e buscar as dependências: **hibernate-core** e **mysql-connector-java** (ficar atento às versões, pois tem grande possibilidade de incompatibilidades).

19. Incluir as dependências como mostrado a seguir.

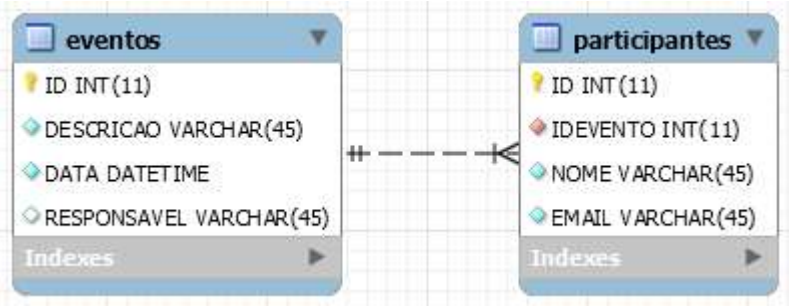
```
<dependencies>
  <dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>5.2.10.Final</version>
  </dependency>
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.42</version>
  </dependency>
</dependencies>
```

20. Salve o projeto, que o maven irá atualizar as dependências.

21. A estrutura do projeto é semelhante a esta:



22. Utilizar o banco de dados **dbeventos**:



23. Utilizar as classes **Evento** e **Participante**, incluindo as anotações JPA adequadas para o mapeamento com o banco de dados:

```
package br.com.fiap.entity;

import java.util.Date;
import java.util.HashSet;
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

@Entity
@Table(name="eventos")
public class Evento {

   @Id
   @GeneratedValue(strategy=GenerationType.AUTO)
```

```
   @Column(name="ID")
   private int id;

   @Column(name="DESCRICAO", length=45)
   private String descricao;

   @Temporal(value=TemporalType.TIMESTAMP)
   @Column(name="DATA", length=45)
   private Date data;

   @Column(name="RESPONSAVEL", length=45)
   private String responsavel;

   @OneToMany(cascade=CascadeType.ALL, fetch = FetchType.LAZY,
mappedBy="evento")
   private Set<Participante> participantes = new HashSet<>();

   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public String getDescricao() {
      return descricao;
   }
   public void setDescricao(String descricao) {
      this.descricao = descricao;
   }
   public Date getData() {
      return data;
   }
   public void setData(Date data) {
      this.data = data;
   }
   public String getResponsavel() {
      return responsavel;
   }
   public void setResponsavel(String responsavel) {
      this.responsavel = responsavel;
   }

   @Override
   public String toString() {
      return this.getDescricao();
   }
   public Set<Participante> getParticipantes() {
      return participantes;
   }
   public void setParticipantes(Set<Participante> participantes) {
      this.participantes = participantes;
```

```
   }
}
```

```
package br.com.fiap.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name="participantes")
public class Participante {

   @Id
   @GeneratedValue(strategy=GenerationType.AUTO)
   @Column(name="ID")
   private int id;

   @ManyToOne(fetch=FetchType.LAZY)
   @JoinColumn(name="IDEVENTO")
   private Evento evento;

   @Column(name="NOME")
   private String nome;

   @Column(name="EMAIL")
   private String email;

   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public Evento getEvento() {
      return evento;
   }
   public void setEvento(Evento evento) {
      this.evento = evento;
   }
   public String getNome() {
      return nome;
```

```
   }
   public void setNome(String nome) {
      this.nome = nome;
   }
   public String getEmail() {
      return email;
   }
   public void setEmail(String email) {
      this.email = email;
   }
}
```

24. Manter o arquivo **persistence.xml** com o seguinte conteúdo:

```
<?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="jpaPU" transaction-type="RESOURCE_LOCAL">
      <class>br.com.fiap.entity.Evento</class>
      <class>br.com.fiap.entity.Participante</class>

      <properties>
         <property name="hibernate.hbm2ddl.auto" value="update" />
         <property name="hibernate.format_sql" value="true" />
         <property name="hibernate.dialect"
            value="org.hibernate.dialect.MySQL5InnoDBDialect" />

         <property name="javax.persistence.jdbc.url"
            value="jdbc:mysql://localhost:3306/dbeventos" />

         <property name="javax.persistence.jdbc.user" value="root" />
         <property name="javax.persistence.jdbc.password" value="fiap" />

         <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
      </properties>
   </persistence-unit>
</persistence>
```

25. Escrever a classe **EventoHelper**, contendo métodos auxiliares para realizar a persistência:

```
package br.com.fiap.helper;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.TypedQuery;

import br.com.fiap.entity.Evento;
import br.com.fiap.entity.Participante;

public class EventoHelper {
```

```
  private EntityManager em;

  public EventoHelper(EntityManager em) {
    this.em = em;
  }

  public String salvar(Evento evento) {
    try {
      em.getTransaction().begin();
      em.persist(evento);
      em.getTransaction().commit();
      return "Evento incluído com sucesso!";
    } catch (Exception e) {
      return e.getMessage();
    }
  }

  public String adicionarParticipante(int idEvento, Participante
participante) {
    try {
      Evento evento = em.find(Evento.class, idEvento);
      participante.setEvento(evento);
      evento.getParticipantes().add(participante);
      em.getTransaction().begin();
      em.persist(evento);
      em.getTransaction().commit();
      return "Evento atualizado com sucesso!";
    } catch (Exception e) {
      return e.getMessage();
    }
  }

  public List<Evento> listarEventos(){
    TypedQuery<Evento> query = em.createQuery("Select e from Evento e",
Evento.class);
    return query.getResultList();
  }
  public List<Participante> listarParticipantes(int idEvento){
    TypedQuery<Participante> query = em.createQuery("Select p from
Participante p Where p.evento.id = :idevento", Participante.class);
    query.setParameter("idevento", idEvento);
    return query.getResultList();
  }

}
```

26. Para testar a aplicação, use a classe abaixo:

```
package br.com.fiap.aplicacao;
```

```
import java.util.Date;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import br.com.fiap.entity.Evento;
import br.com.fiap.entity.Participante;
import br.com.fiap.helper.EventoHelper;

public class AppEventos {
   public static void main(String[] args) {
      //incluirEvento();
      //listarEventos();
      listarParticipantes(1);
   }

   private static void incluirEvento() {
      EntityManagerFactory emf =
Persistence.createEntityManagerFactory("jpaPU");
      EntityManager em = emf.createEntityManager();

      EventoHelper helper = new EventoHelper(em);

      Evento evento = new Evento();
      evento.setDescricao("Novo curso disponivel");
      evento.setResponsavel("Juvenal Santos");
      evento.setData(new Date());

      Participante p1 = new Participante();
      p1.setNome("Jose Antonio");
      p1.setEmail("jantonio@fap.com.br");
      p1.setEvento(evento);

      Participante p2 = new Participante();
      p2.setNome("Camila");
      p2.setEmail("camila@fap.com.br");
      p2.setEvento(evento);

      Participante p3 = new Participante();
      p3.setNome("Bonifacio");
      p3.setEmail("bonifacio@fap.com.br");
      p3.setEvento(evento);

      evento.getParticipantes().add(p1);
      evento.getParticipantes().add(p2);
      evento.getParticipantes().add(p3);

      System.out.println(helper.salvar(evento));
   }
```

```
  private static void listarEventos() {
    EntityManagerFactory emf =
Persistence.createEntityManagerFactory("jpaPU");
    EntityManager em = emf.createEntityManager();

    EventoHelper helper = new EventoHelper(em);

    for (Evento evento : helper.listarEventos()) {
      System.out.println("Id: " + evento.getId());
      System.out.println("Descrição: " + evento.getDescricao());
      System.out.println("Responsável: " + evento.getResponsavel());
      System.out.println("-------------------------------------");
    }
  }

  private static void listarParticipantes(int idEvento) {
    EntityManagerFactory emf =
Persistence.createEntityManagerFactory("jpaPU");
    EntityManager em = emf.createEntityManager();

    EventoHelper helper = new EventoHelper(em);

    for (Participante participante : helper.listarParticipantes(idEvento)) {
      System.out.println("Id: " + participante.getId());
      System.out.println("Nome: " + participante.getNome());
      System.out.println("Email: " + participante.getEmail());
      System.out.println("-------------------------------------");
    }
  }
}
```

Bom trabalho a todos!

Prof. Rafael Matsuyama