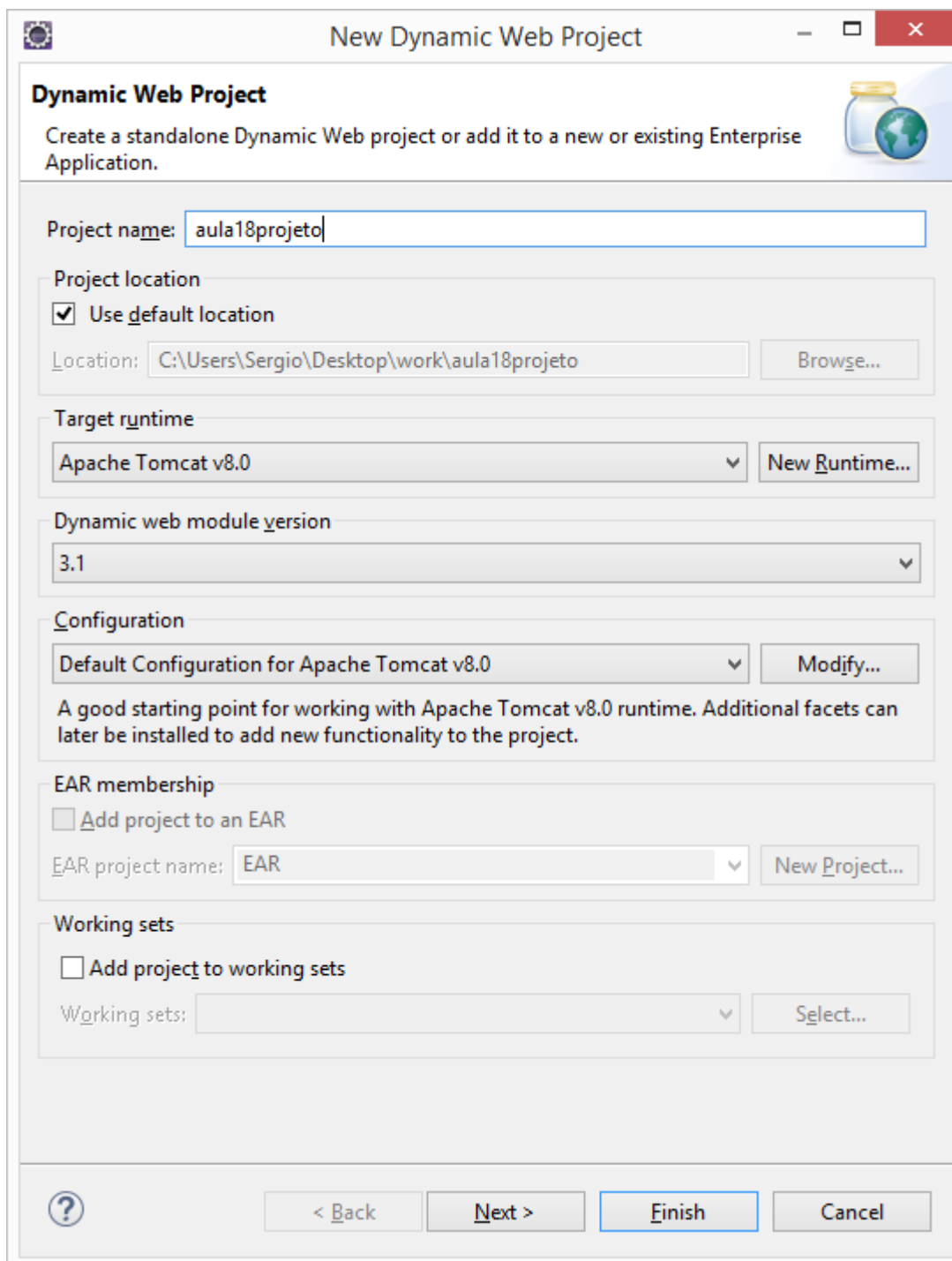


Criando o projeto:

- **File > New > Dynamic Web Project**



The screenshot shows the 'New Dynamic Web Project' dialog box in the Eclipse IDE. The dialog is titled 'New Dynamic Web Project' and has a subtitle 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' The 'Project name' field is filled with 'aula18projeto'. The 'Project location' section has the 'Use default location' checkbox checked, and the 'Location' field shows 'C:\Users\Sergio\Desktop\work\aula18projeto'. The 'Target runtime' is set to 'Apache Tomcat v8.0'. The 'Dynamic web module version' is set to '3.1'. The 'Configuration' section shows 'Default Configuration for Apache Tomcat v8.0'. The 'EAR membership' section has the 'Add project to an EAR' checkbox unchecked, and the 'EAR project name' is 'EAR'. The 'Working sets' section has the 'Add project to working sets' checkbox unchecked, and the 'Working sets' field is empty. At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (highlighted), and 'Cancel'.

Mapeando a entidade Pessoa

JPA – Java Persistence API

Mapeamento Objeto Relacional

```
package br.com.brq.entities;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GenerationType;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;

@Entity
@Table(name = "pessoa")
@NamedQueries({
    {
        @NamedQuery(name = Pessoa.FIND_ALL,
            query = "select p from Pessoa as p")
    }
})
public class Pessoa {

    public static final String FIND_ALL = "pessoa.findall";

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "idpessoa")
    private Integer idPessoa;

    @Column(name = "nome", length = 50, nullable = false)
    private String nome;

    @Column(name = "email", length = 50, nullable = false)
    private String email;

    public Pessoa() {
    }
}
```

```
public Pessoa(Integer idPessoa, String nome, String email) {
    super();
    this.idPessoa = idPessoa;
    this.nome = nome;
    this.email = email;
}

public Integer getIdPessoa() {
    return idPessoa;
}

public void setIdPessoa(Integer idPessoa) {
    this.idPessoa = idPessoa;
}

public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

public String getEmail() {
    return email;
}

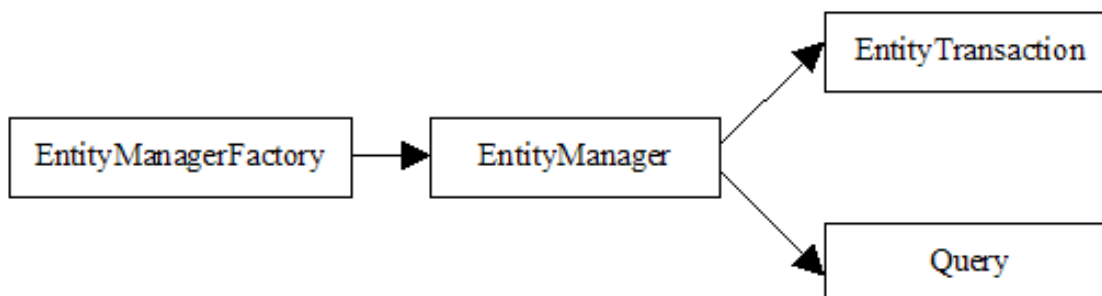
public void setEmail(String email) {
    this.email = email;
}

@Override
public String toString() {
    return "Pessoa [idPessoa=" + idPessoa + ", nome="
        + nome + ", email=" + email + "];"
}
}
```

Entity Manager

Na nova Java Persistence Specification, o EntityManager é o serviço central para todas as ações de persistência. Entidades são objetos de Java claros que são alocados como qualquer outro objeto Java. Eles não ficam persistentes explicitamente até seu código interagir com o EntityManager para os fazer persistente.

O EntityManager administra o O/R que o mapeia entre uma classe de entidade e uma fonte de dados subjacente. O EntityManager provê APIs para criar consultas, buscando objetos, sincronizando objetos, e inserindo objetos no banco de dados. Também pode prover caching e pode administrar a interação entre uma entidade e serviços transacionais em um ambiente Java EE.



META-INF/persistence.xml

Mapeando a conexão com o banco de dados

```

<?xml version="1.0" encoding="UTF-8" ?>
<persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
    http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
  version="2.0" xmlns="http://java.sun.com/xml/ns/persistence">

  <!-- nome da configuração mapeada : aula -->
  <persistence-unit name="aula" transaction-type="RESOURCE_LOCAL">

    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <properties>
      <property name="hibernate.dialect"
        value="org.hibernate.dialect.MySQL5InnoDBDialect" />
      <property name="javax.persistence.jdbc.driver"
        value="com.mysql.jdbc.Driver" />
      <property name="javax.persistence.jdbc.url"
        value="jdbc:mysql://localhost:3306/aula18" />
      <property name="javax.persistence.jdbc.user" value="root" />
      <property name="javax.persistence.jdbc.password" value="brqbrq" />
    
```

```
<property name="hibernate.show_sql" value="true" />
<property name="hibernate.format_sql" value="false" />
<property name="hibernate.use_sql_comments" value="false" />
<property name="hibernate.jdbc.wrap_result_sets" value="false" />
<property name="hibernate.hibernate.cache.use_query_cache" value="true"
/>
    <property name="hibernate.hbm2ddl.auto" value="update" />
</properties>
</persistence-unit>
</persistence>
```

HibernateUtil para gerar a instancia do EntityManager

```
package br.com.brq.persistence;

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

public class HibernateUtil {

    //gerenciador de entidades do Hibernate 4
    private static EntityManager entityManager;

    //método para configurar e retornar o EntityManager
    public static EntityManager getEntityManager() {

        //nome da configuração feita no arquivo /META-INF/persistence.xml
        EntityManagerFactory factory = Persistence
            .createEntityManagerFactory("aula");

        if(entityManager == null){ //se não foi inicializado..
            entityManager = factory.createEntityManager();
        }

        return entityManager;
    }
}
```

Classe de persistência para Pessoa: DAOPessoa.java

```
package br.com.brq.persistence;
```

```
import java.util.List;

import javax.persistence.EntityManager;

import br.com.brq.entities.Pessoa;

public class DAOPessoa {

    //declarar o EntityManager...
    private EntityManager entityManager;

    //construtor..
    public DAOPessoa() {
        //inicializar o entityManager..
        entityManager = HibernateUtil.getEntityManager();
    }

    //método para gravar uma pessoa..
    public void insert(Pessoa p) throws Exception{
        entityManager.getTransaction().begin();
        entityManager.persist(p); //gravando..
        entityManager.getTransaction().commit();
    }

    //método para atualizar..
    public void update(Pessoa p) throws Exception{
        entityManager.getTransaction().begin();
        entityManager.merge(p); //atualizando..
        entityManager.getTransaction().commit();
    }

    //método para excluir..
    public void delete(Pessoa p) throws Exception{
        entityManager.getTransaction().begin();
        entityManager.remove(p); //excluindo..
        entityManager.getTransaction().commit();
    }

    //método para buscar 1 pessoa pelo id..
    public Pessoa findById(Integer idPessoa) throws Exception{
        return entityManager.find(Pessoa.class, idPessoa);
    }

    //método para executar a namedquery..
```

```
@SuppressWarnings("unchecked")
public List<Pessoa> findAll() throws Exception{
    return entityManager.createNamedQuery(Pessoa.FIND_ALL)
        .getResultList();
}
}
```

Criando as páginas do projeto:

- index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css" href="css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css" href="css/bootstrap-
theme.min.css"/>

</head>
<body class="container">

    <h2>Controle de Pessoas</h2>
    Struts 2 Annotations e Hibernate 4 EntityManager
    <hr/>

    <ul>
        <li> <a href="cadastro.jsp">Cadastrar Pessoas</a> </li>
        <li> <a href="consultarpessoas.action">Consultar Pessoas</a> </li>
    </ul>

</body>
</html>
```

- cadastro.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<html>
<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css"
        href="css/bootstrap.min.css"/>
```

```
<link rel="stylesheet" type="text/css"
      href="css/bootstrap-theme.min.css"/>

</head>
<body class="container">

    <h2>Cadastro de Pessoas</h2>
    <a href="index.jsp">Voltar</a> para a página inicial.
    <hr/>

    <div class="col-md-4">

        <form name="formulario" method="post"
              action="cadastrarpessoa.action">

            Para cadastrar uma pessoa, informe os dados abaixo:
            <br/><br/>

            <label>Nome da Pessoa:</label>
            <input type="text" name="pessoa.nome" class="form-control"
                  placeholder="Digite aqui"/>
            <br/>

            <label>Email:</label>
            <input type="text" name="pessoa.email"
                  class="form-control"
                  placeholder="Digite aqui"/>
            <br/>

            <input type="submit" value="Cadastrar Pessoa"
                  class="btn btn-success btn-sm"/>
            <br/><br/>

            ${mensagem}

        </form>

    </div>

</body>
</html>
```

- consulta.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
      pageEncoding="ISO-8859-1"%>

<%@ taglib uri="/struts-tags" prefix="s" %>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>
```



```
<link rel="stylesheet" type="text/css" href="css/bootstrap.min.css"/>
<link rel="stylesheet" type="text/css" href="css/bootstrap-
theme.min.css"/>

</head>
<body class="container">

    <h2>Consulta de Pessoas</h2>
    <a href="index.jsp">Voltar</a> para a página inicial.
    <hr/>

    <div>
        ${mensagem}
    </div>

    <table class="table table-hover">
        <thead>
            <tr>
                <th>Código</th>
                <th>Nome da Pessoa</th>
                <th>Email</th>
                <th>Operações</th>
            </tr>
        </thead>
        <tbody>

            <s:iterator value="ListagemPessoas">
                <tr>
                    <td> <s:property value="idPessoa"/> </td>
                    <td> <s:property value="nome"/> </td>
                    <td> <s:property value="email"/> </td>
                    <td>

                        <a href="exibirpessoa.action?
                            id=${idPessoa}"
                            class="btn btn-primary btn-sm">
                                Exibir Dados
                        </a>

                        <a href="excluirpessoa.action?
                            id=${idPessoa}"
                            class="btn btn-danger btn-sm">
                                Excluir Registro
                        </a>

                    </td>
                </tr>
            </s:iterator>

        </tbody>

        <tfoot>
            <tr>
                <td colspan="4">
                    Quantidade de registro(s):
                </td>
            </tr>
        </tfoot>
    </table>
```

```
                <s:property value="ListagemPessoas.size()"/>
            </td>
        </tr>
    </tfoot>
</table>

</body>
</html>
```

- edicao.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css" href="css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css" href="css/bootstrap-
theme.min.css"/>

</head>
<body class="container">

    <h2>Editar dados de Pessoa</h2>
    <a href="index.jsp">Voltar</a> para a página inicial.
    <hr/>

    <div class="col-md-4">

        <form name="formulario" method="post"
            action="atualizarpessoa.action">

            Para atualizar uma pessoa, altere os dados abaixo:
            <br/><br/>

            <label>Código:</label>
            <input type="text" name="pessoa.idPessoa"
                class="form-control"
                readonly="readonly"
                value="${pessoa.idPessoa}"/>
            <br/>

            <label>Nome da Pessoa:</label>
            <input type="text" name="pessoa.nome" class="form-control"
                placeholder="Digite aqui" value="${pessoa.nome}"/>
            <br/>

            <label>Email:</label>
            <input type="text" name="pessoa.email"
                class="form-control"
                placeholder="Digite aqui"
```

```
        value="${pessoa.email}"/>
    <br/>

    <input type="submit" value="Atualizar Pessoa"
        class="btn btn-primary btn-sm"/>
    <br/><br/>

    ${mensagem}

</form>

</div>

</body>
</html>
```

Configurando o struts no projeto web: **web.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    id="WebApp_ID" version="2.5">
    <display-name>struts2estudo</display-name>
    <filter>
        <filter-name>struts2</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter
            .StrutsPrepareAndExecuteFilter</filter-class>
    </filter>

    <filter-mapping>
        <filter-name>struts2</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

Criando a classe de controle com Struts: PessoaAction.java

```
package br.com.brq.action;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.apache.struts2.ServletActionContext;
import org.apache.struts2.convention.annotation.Action;
import org.apache.struts2.convention.annotation.Result;

import br.com.brq.entities.Pessoa;
import br.com.brq.persistence.DAOPessoa;
import net.sf.ehcache.search.Results;

//classe de controle padrão struts..
public class PessoaAction {

    private static final String SUCCESS = "sucesso"; //constante..

    private Pessoa pessoa; // receber os dados do formulario de cadastro
    private List<Pessoa> listagemPessoas; //lista exibida na página de consulta..

    // construtor..
    public PessoaAction() {
        pessoa = new Pessoa(); // instanciando..
    }

    @Action(
        value = "/cadastrarpessoa",
        results = {
            @Result(name = SUCCESS, location = "/cadastro.jsp")
        }
    )
    public String cadastrar(){

        HttpServletRequest request = ServletActionContext.getRequest();

        try{
            DAOPessoa d = new DAOPessoa(); //persistencia..
            d.insert(pessoa); //gravando..
        }
    }
}
```

```
request.setAttribute("mensagem", "Pessoa " + pessoa.getNome()
                                + ", cadastrado com sucesso.");

pessoa = new Pessoa(); //novo espaço de memória..
}
catch(Exception e){
    request.setAttribute("mensagem", e.getMessage());
}

return SUCCESS;
}
```

```
@Action(
    value = "/consultarpessoas",
    results = {
        @Result(name = SUCCESS, location = "/consulta.jsp")
    }
)
public String consultar(){
```

```
    HttpServletRequest request = ServletActionContext.getRequest();

    try{
        DAOPessoa d = new DAOPessoa(); //persistencia..
        listagemPessoas = d.findAll(); //executando a consulta..
    }
    catch(Exception e){
        request.setAttribute("mensagem", e.getMessage());
    }

    return SUCCESS;
}
```

```
@Action(
    value = "/excluirpessoa",
    results = {
        @Result(name = SUCCESS, location = "/consulta.jsp")
    }
)
public String excluir(){
```

```
    HttpServletRequest request = ServletActionContext.getRequest();
```

```
try{
    //resgatando o id enviado pela URL..
    Integer idPessoa = Integer.parseInt(request.getParameter("id"));

    DAOPessoa d = new DAOPessoa(); //persistencia..
    d.delete(d.findById(idPessoa)); //buscando e excluindo pessoa..

    listagemPessoas = d.findAll(); //nova consulta...

    request.setAttribute("mensagem",
        "Pessoa excluido com sucesso.");
}
catch(Exception e){
    request.setAttribute("mensagem", e.getMessage());
}

return SUCCESS;
}

@Action(
    value = "/exibirpessoa",
    results = {
        @Result(name = SUCCESS, location = "/edicao.jsp")
    }
)
public String exibir(){

    HttpServletRequest request = ServletActionContext.getRequest();

    try{
        //resgatar o id enviado pela página..
        Integer idPessoa = Integer.parseInt(request.getParameter("id"));

        DAOPessoa d = new DAOPessoa(); //persistencia..
        pessoa = d.findById(idPessoa); //buscando pessoa pelo id..
    }
    catch(Exception e){
        request.setAttribute("mensagem", e.getMessage());
    }

    return SUCCESS;
}
```

```
@Action(
    value = "/atualizarpessoa",
    results = {
        @Result(name = SUCCESS, location = "/edicao.jsp")
    }
)
public String atualizar(){

    HttpServletRequest request = ServletActionContext.getRequest();

    try{
        DAOPessoa d = new DAOPessoa(); //persistencia..
        d.update(pessoa); //atualizando..

        request.setAttribute("mensagem", "Pessoa " + pessoa.getNome()
            + ", atualizado com sucesso.");
    }
    catch(Exception e){
        request.setAttribute("mensagem", e.getMessage());
    }

    return SUCCESS;
}

public Pessoa getPessoa() {
    return pessoa;
}

public void setPessoa(Pessoa pessoa) {
    this.pessoa = pessoa;
}

public List<Pessoa> getListagemPessoas() {
    return listagemPessoas;
}

public void setListagemPessoas(List<Pessoa> listagemPessoas) {
    this.listagemPessoas = listagemPessoas;
}
}
```

Estrutura do projeto:

