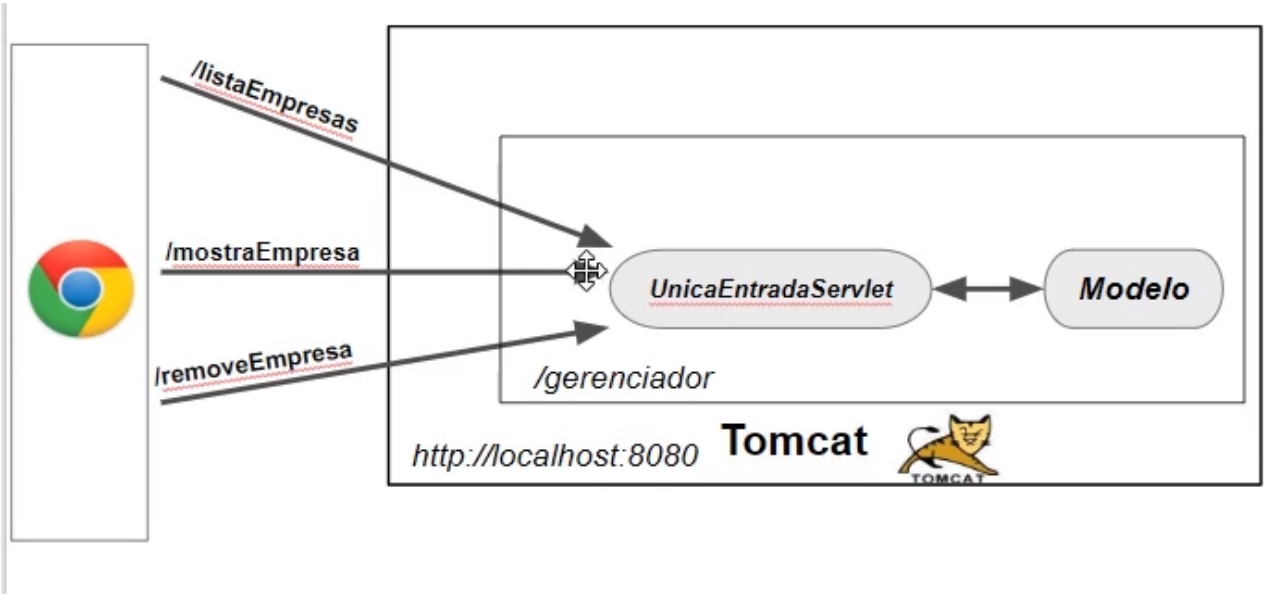
07

Implementando o controlador

Transcrição

Agora vamos trabalhar na nossa entrada única. Queremos que todas as nossas requisições, como `/listaEmpresas`, `/mostraEmpresa` e `/removeempresa`, cheguem pelo mesmo Servlet, para a partir disso tomarmos uma atitude. Agora vamos criar o Servlet que receberá essas requisições e que será a entrada para a nossa aplicação.



No pacote que limpamos anteriormente, criaremos esse novo Servlet clicando com o botão direito no pacote e em seguida em "New > Servlet". Por enquanto iremos nomeá-lo como `UnicaEntradaServlet`.

Na página "Create Servlet", renomearemos o mapeamento ("URL Mappings") para "/entrada" e clicaremos em next. Na próxima página, desmarcaremos as caixas `doPost` e `doGet`, e marcaremos a caixa `service`, pois queremos que esse Servlet atenda tanto `get` quanto `post` (ou seja, faça qualquer tipo de

requisição). Também podemos desmarcar a caixa `Constructors from superclass`, pois não precisaremos utilizá-la.

Dessa forma, teremos:

```
package br.com.alura.gerenciador.servlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/entrada")
public class UnicaEntradaServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;


    protected void service(HttpServletRequest request, HttpSer
        // TODO Auto-generated method stub
    }

}
```

COPIAR CÓDIGO

Nesse Servlet, precisaremos, de alguma forma, analisar a requisição. Por exemplo, quando chamamos `/listaEmpresas`, `/mostraEmpresa` ou `/removeEmpresa`, todas essas URLs na verdade se chamarão `/entrada`. Dessa forma, sempre chamaremos o mesmo Servlet. Mas como o Servlet irá saber se queremos listar as empresas, mostrar as empresas, remover uma empresa ou cadastrar uma nova empresa?

Além de chamar o nosso Servlet, a ideia é enviar, juntamento dos outros parâmetros que precisamos enviar, pelo menos um parâmetro que defina qual

ação queremos executar - por exemplo, `/entrada?acao=RemoveEmpresa`, `/entrada?acao=MostraEmpresa` e `/entrada?acao=ListaEmpresas`. Vamos implementar isso no nosso `UnicaEntradaServlet.java`.

Precisaremos ler o parâmetro que define, o que é feito a ação a partir de uma requisição. Como no exemplo anterior, chamaremos esse parâmetro de "acao". Dessa forma, teremos `String paramAcao = request.getParameter("acao")`.

Nessa primeira implementação, trabalharemos com o método `if()`. Se o nosso `paramAcao` for igual a `ListaEmpresas`, queremos gerar a saída "listando empresas". Assim, teremos:

```
if(paramAcao.equals("ListaEmpresa")) {
            System.out.println("listando empresas");
```

COPIAR CÓDIGO

Repetiremos essa construção para todas as ações, com a adição do `else`:

```
@WebServlet("/entrada")
public class UnicaEntradaServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;


    protected void service(HttpServletRequest request, HttpSer

        String paramAcao = request.getParameter("acao");

        if(paramAcao.equals("ListaEmpresas")) {
            System.out.println("listando empresas");
        } else if(paramAcao.equals("RemoveEmpresa")) {
            System.out.println("removendo empresa");
        } else if(paramAcao.equals("MostraEmpresa")) {
            System.out.println("mostrando dados da empresa")
        }
```

```
    }

}
```

COPIAR CÓDIGO

Ainda precisamos tomar alguma atitude para que o código realmente execute a ação correta, mas já podemos testá-lo subindo o Tomcat e acessando, por exemplo, a URL http://localhost:8080/gerenciador/entrada?acao=ListaEmpresas (http://localhost:8080/gerenciador/entrada?acao=ListaEmpresas). Ao fazermos isso, nada será mostrado no navegador, mas o console no Eclipse irá imprimir "listando empresas".

Pronto, nosso primeiro passo está feito! Agora vamos para o segundo passo, que será um pouco mais complicado. Preparado?

Vamos começar a trabalhar com a ação `ListaEmpresas`. Nela, precisaremos executar o código que está dentro do nosso `ListaEmpresasServlet`. Nossa primeira ideia (inocente) seria copiar o código de `ListaEmpresasServlet` e colá-lo dentro de `UnicaEntradaServlet`:

```
if(paramAcao.equals("ListaEmpresas")) {
    System.out.println("listando empresas");

    Banco banco = new Banco();
    List<Empresa> lista = banco.getEmpresas();

    request.setAttribute("empresas", lista);

    RequestDispatcher rd = request.getRequestDispatcher("/lista
    rd.forward(request, response);

//...
}
```

COPIAR CÓDIGO

Se testarmos no navegador, iremos reparar que isso até funcionou. No entanto, copiar e colar um código normalmente é uma má prática. Além disso, nosso `else if` ficará enorme, pois todo o código que estava dentro dos nossos Servlets, separadamente, será jogado em um único lugar. Ou seja, essa não parece uma boa solução.

Ao invés disso, colocaremos nosso código em uma classe separada, assim como fizemos anteriormente através dos nossos Servlets. Porém, essa classe separada não será um Servlet, mas uma classe um pouco mais simples.

Para isso, criaremos uma nova classe, clicando com o botão direito em `src` e em seguida em "New > Class". Chamaremos essa nova classe de `ListasEmpresas`, pois essa é a ação que ela irá executar.

Para agrupar essas novas classes que irão executar ações, como `AlterarEmpresa`, `MostrarEmpresa`, `RemoverEmpresa` e `NovaEmpresa`, também criaremos um novo pacote. Na "New Java Class", preencheremos o campo "Package" com "br.com.alura.gerenciador.acao".

Agora removeremos o código que copiamos anteriormente do nosso `if()`, selecionando todo ele e pressionando "Ctrl + X" (recortar). Na classe `ListaEmpresas`, criaremos o método `executa()` (que poderia ter o nome que quiséssemos). Esse método irá encapsular o código para listar empresas, e nele colaremos o código que recortamos:

```
public class ListaEmpresas {

    public void executa() {
        Banco banco = new Banco();
        List<Empresa> lista = banco.getEmpresas();

        request.setAttribute("empresas", lista);

        RequestDispatcher rd = request.getRequestDispatcher(
```

```
            rd.forward(request, response);
        }

    }
```

COPIAR CÓDIGO

Porém, temos um problema. Repare que nosso código não está compilando, pois o método `forward()` precisa das referências de `request` e `response`.

Para consertarmos isso, no nosso `UnicaEntradaServlet`, vamos criar uma `ListaEmpresas acao = new ListaEmpresas()` e pressionaremos "Ctrl + Shit + O" para importar. A ideia aqui é criar um objeto que saiba fazer a ação e executá-la (`acao.executa()`). Futuramente faremos o mesmo com as outras ações.

Nosso método `executa()` ainda precisa das referências de `request` e `response`, vamos passá-las aqui:

```
if(paramAcao.equals("ListaEmpresas")) {
    System.out.println("listando empresas");

    ListaEmpresas acao = new ListaEmpresas();
    acao.executa(request, response);
    //...
}
```

COPIAR CÓDIGO

Porém, o método `executa()` ainda não recebe essas duas referências. Passaremos essas referências com `HttpServletRequest request` e `HttpServletResponse response`. Além disso, precisaremos importar dos nossos Servlets ("Ctrl + Shit + O") e criar uma exceção (`throws ServletException, IOException`):

```
public class ListaEmpresas {

    public void executa(HttpServletRequest request, HttpServlet
        Banco banco = new Banco();
        List<Empresa> lista = banco.getEmpresas();

        request.setAttribute("empresas", lista);

        RequestDispatcher rd = request.getRequestDispatcher("/
        rd.forward(request, response);
    }

}
```

COPIAR CÓDIGO

Vamos recaptular o que fizemos até agora: criamos uma classe que **não** é um Servlet, apesar das semelhanças (principalmente nos objetos que indicamos como referência do método `executa()`). Além disso, colocamos o código da ação (nesse caso, `ListaEmpresas`) dentro da classe. Mais tarde, ainda tentaremos simplificar esse método.

Agora que nosso código está encapsulado, não precisaremos mais colocar o código puro da ação na nossa `EntradaUnicaServlet`, apenas delegá-la.

Para testarmos se nossa classe está funcionando corretamente, vamos remover `System.out.println("listando empresas")` do método `if()` e colocá-lo diretamente na classe `ListaEmpresas`.

```
public class ListaEmpresas {

    public void executa(HttpServletRequest request, HttpServlet

        System.out.println("listando empresas");
```

```
        Banco banco = new Banco();
        List<Empresa> lista = banco.getEmpresas();
        //...
    }
}
```

COPIAR CÓDIGO

Quando executamos esse código e acessamos a URL http://localhost:8080/gerenciador/entrada?acao=ListaEmpresas (http://localhost:8080/gerenciador/entrada?acao=ListaEmpresas), a página continua funcionando corretamente e o console ainda imprime "listando empresas".

Agora precisaremos repetir o processo para todas as ações. Faremos isso juntos no próximo vídeo, mas se você já pode, sozinho, tentar criar as outras classes e as ações para chamar o método `executa()`. Até logo!