



Listando empresas com servlets

Transcrição

Criamos a classe `Empresa` para representar nosso modelo, e a classe `Banco` para simular um banco de dados por meio de uma lista estática. Enquanto a máquina virtual estiver rodando, conseguiremos adicionar empresas dentro de `lista`. Porém, se reiniciarmos a máquina, a lista virtual sumirá, portanto não estamos trabalhando com 100% de persistência, mas podemos estudar e compreender os conceitos.

Nossa próxima meta é listar as empresas cadastradas no sistema acessando-o pelo navegador. Para isso, precisaremos trabalhar com protocolo HTTP, enviando uma requisição, recebendo-a no lado do servidor, buscando as empresas existentes e devolvendo o HTML para o navegador.

Na prática, fizemos procedimentos muito semelhantes, mas iremos revisar algumas ideias e fixar o conteúdo.

Em nosso pacote `br.com.alura.gerenciador.servlet` criaremos um Servlet chamado `ListaEmpresasServlet`. Na caixa de diálogo de configurações desse novo Servlet, ajustaremos a opção "URL Mappings", o endereço da URL, para `/listaEmpresas`. Pressionaremos o botão "Next" para escolher os métodos HTTP que utilizaremos, no caso, o mais correto seria usar o GET, afinal o navegador acessa o servidor, envia uma requisição e é devolvido a lista de empresas. Marcaremos, portanto, a opção "doGet".

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

```
import java.io.IOException;

/**
 *Servlet implementation class ListaEmpresasServlet
 */
@WebServlet("/listaEmpresas")
public class ListaEmpresasServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ListaEmpresasServlet() {
        super();
        // TODO auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getRequestURL());
    }
}
```

[COPIAR CÓDIGO](#)

Desse código, deletaremos construtor, os comentários e o método `response.getWriter()`.

Primeiramente, coletaremos a List de Empresa por meio da classe Banco, aquela que simula o banco de dados. Usaremos o método `getEmpresas()` para realizar essa ação. Lembrando que precisamos realizar a importação de `java.util` para que não haja problemas de compilação.

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

import java.io.IOException;

/**
 *Servlet implementation class ListaEmpresasServlet
 */
@WebServlet("/listaEmpresas")
public class ListaEmpresasServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

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

    }
}
```

[COPIAR CÓDIGO](#)

Temos a `lista` no Servlet, resta devolvermos conteúdo no formato HTML para o navegador. Para isso, usaremos o `PrintWriter`, com a referência `out`. A procedência de `PrintWriter` é `response`, e utilizaremos o método `getWriter()` para coletá-lo. Ao final, usaremos `out.println()` para escrever o código HTML que possibilitará a inserção da lista de empresas. Nossa lista ficará alocada dentro de um elemento ``.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

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

```
PrintWriter out = response.getWriter();

out.println("<html><body>");
out.println("<ul>");
out.println("</ul>");
out.println("</body></html>");

}
}
```

[COPIAR CÓDIGO](#)

Dentro desse elemento, criaremos um laço que mostra uma nova empresa em cada linha. Para isso, escreveremos `for (Empresa empresa : lista)`, em seguida imprimiremos o resultado utilizando `out.println()` recebendo o elemento `` que representa um item da lista no mundo HTML. Adicionaremos, ainda, `empresa.getNome()`, para coletar os nomes das empresas, e fecharemos o elemento com ``.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

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

    PrintWriter out = response.getWriter();
    out.println("<html><body>");
    out.println("<ul>");
    for (Empresa empresa : lista) {
        out.println("<li>" + empresa.getNome() + "</li>");
    }
    out.println("</ul>");
    out.println("</body></html>");
}
```

```
}  
}
```

[COPIAR CÓDIGO](#)

Utilizamos a classe `Banco` para coletar o nome das empresas e elaboramos uma resposta HTML, embora pareça errado escrever um código dessa natureza diretamente no Java. Agora, testaremos todas as modificações e inserções no navegador.

No navegador, escreveremos a URL

`localhost:8080/gerenciador/listaEmpresas` . Ao pressionarmos "Enter", veremos que nada é exibido na tela. Ao analisarmos o código fonte da página, veremos o HTML:

```
<html><body>  
<ul>  
</ul>  
</body></html>
```

[COPIAR CÓDIGO](#)

O HTML foi enviado, mas como não temos nenhuma empresa cadastrada, não existe qualquer conteúdo a ser exibido no navegador, portanto, primeiramente cadastraremos uma empresa no sistema, acessando a URL

`localhost:8080/gerenciador/formNovaEmpresa.html` .

No formulário "Nome" escreveremos "Alura" e pressionaremos o botão "Enviar". Voltaremos ao formulário mais uma vez e cadastraremos a empresa "Caelum". Dessa forma temos duas empresas cadastradas na lista.

Acessaremos novamente `localhost:8080/gerenciador/listaEmpresas` , e agora teremos o seguinte conteúdo exibido:

Alura

Caelum

É importante lembrar que essas informações não serão persistentes, pois não as gravamos em um HD. Ao reiniciarmos o servidor, esses dados serão excluídos. Todas as vezes teremos de recadastrar as empresas, o que é trabalhoso e dificultará nosso processo de aprendizagem.

Para resolvermos essa questão, adicionaremos um bloco estático de empresas na classe `Banco`. Assim como existem atributos estáticos, podemos criar códigos estáticos. O código estático que criaremos será executado quando a máquina virtual carregar a classe e a lista de empresas for inicializada. Neste ponto, serão cadastradas duas empresas nessa lista: `Alura` e `Caelum`.

```
public class Banco {  
  
    private static List<Empresa> lista = new ArrayList<>();  
  
    static {  
        Empresa empresa = new Empresa();  
        empresa.setNome("Alura");  
        Empresa empresa2 = new Empresa();  
        empresa2.setNome("Caelum");  
        lista.add(empresa);  
        lista.add(empresa2);  
    }  
  
    public void adiciona (Empresa empresa) {  
        Banco.lista.add(empresa);  
    }  
}
```

[COPIAR CÓDIGO](#)

Após salvarmos e recarregarmos a aplicação, poderemos fazer um novo teste no navegador, por meio da URL `localhost8080/gerenciador/listaEmpresas`.

Perceberemos que os itens `Alura` e `Caelum` já estão disponíveis na página por padrão. Contudo, ainda podemos adicionar novas empresas à lista.

Iremos até o formulário `localhost8080/gerenciador/formNovaEmpresa` e cadastraremos a empresa "Google". Ao acessarmos novamente a lista de empresas cadastradas, teremos:

`Alura`

`Caelum`

`Google`

Se recarregarmos a página, apenas `Alura` e `Caelum` permanecerão no HTML, afinal estamos simulando um banco de dados. Conseguimos chegar ao cadastro completo de uma empresa, nas próximas aulas estudaremos novas possibilidades de uso de Servlets!