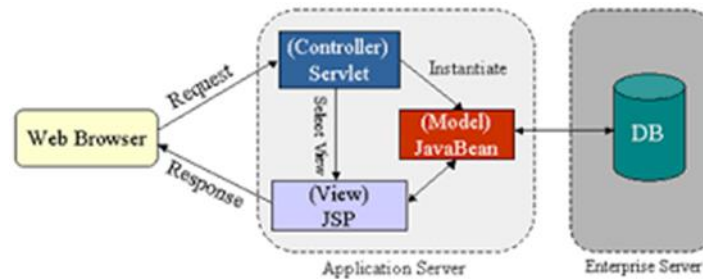


## Unidade 6

### Gerenciamento de Sessão

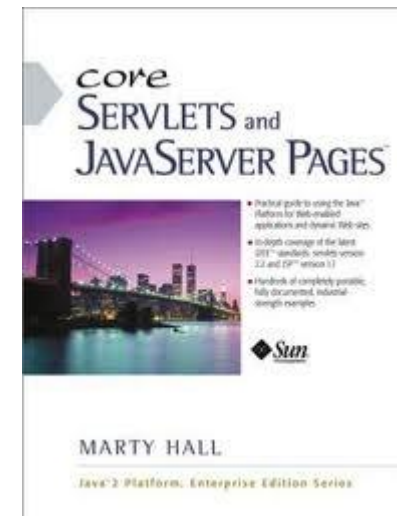
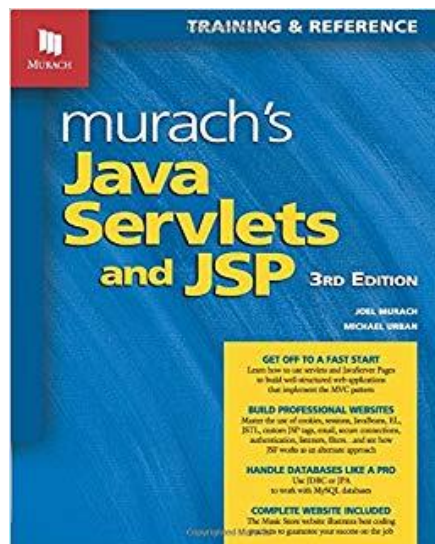


Prof. Aparecido V. de Freitas  
Doutor em Engenharia  
da Computação pela EPUSP



## Referências

- Head First Servlets & JSP – Bryan Basham, Kathy Sierra & Bert Bates
- Core Servlets and Java Server Pages – Marty Hall
- Java Servlets and JSP – Joel Murach – 3rd Edition



# Introdução

- HTTP é um protocolo “**stateless**”: cada vez que um cliente recupera uma página WEB, o cliente abre uma conexão separada para o WEB Server.
- Desse modo, o servidor não mantém, de forma automática, informação de contexto sobre o cliente.



## Quais as implicações disto ?

- ✱ Quando o cliente faz compras num site de comércio eletrônico, como o servidor sabe o que já foi comprado ?
- ✱ Quando cliente desejar fazer um checkout, como proceder para fechar a compra ?



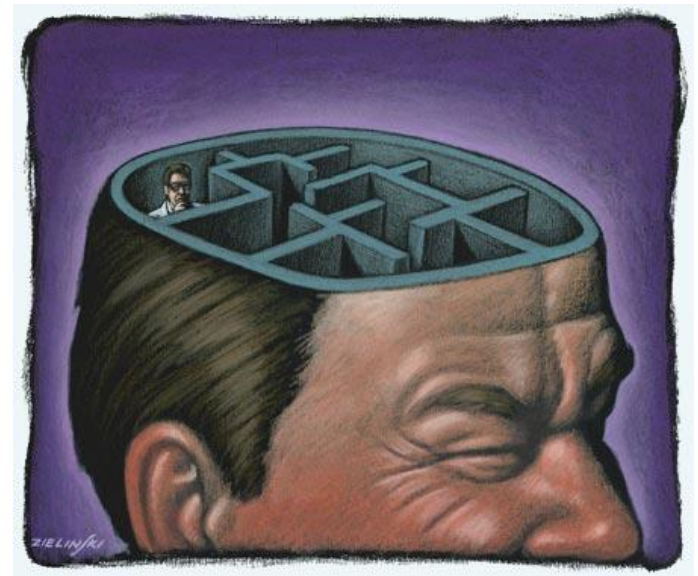
# Gerenciamento de Sessão

- ✿ Servlets disponibilizam uma API para gerenciamento de sessão: a API **HttpSession**.
- ✿ O uso de sessões em servlets é direto e envolve **quatro** passos básicos.



## Passos para Gerenciamento de Sessão

- ✿ Acesso ao objeto **Session**;
- ✿ Pesquisa da Informação associada à sessão;
- ✿ Armazenamento da informação em uma sessão;
- ✿ Descarte dos dados da sessão.



# Acesso ao objeto Session

- ✿ Objetos **Session** são do tipo **HttpSession**;
- ✿ São basicamente **hash tables** que podem armazenar objetos do usuário (cada qual associado a uma chave);
- ✿ Podemos acessar objetos **Session** por meio da chamada:

```
HttpSession session = request.getSession();
```



## O que a API faz nos bastidores ?

- O sistema extrai um user **ID** de um cookie ou da URL.
- Este **ID** é usado como chave para acessar uma tabela previamente criada.
- No entanto, isto é transparente ao programador.
- Você simplesmente chama **getSession()**
- Se não houver uma Session **ID**, o sistema cria uma nova sessão vazia.





# getSession()

- **getSession(true)** cria uma nova sessão se a sessão não existir.
- **getSession(false)** não cria nova sessão. Apenas checa se a sessão existe.

```
HttpSession session=request.getSession(false);  
if (session == null) {  
    EnviaMensagemCarrinhoVazio();  
else  
    ExtraiInformacoesSession();
```



# Informações da Sessão

- Objetos **HttpSession** residem no servidor.
- Assim, estes objetos não trafegam na rede.
- O servidor os mantém por meio de **cookies** ou reescrita de **URL**.
- Têm uma estrutura (**hash table**) no qual armazenam-se qualquer número de chaves e valores associados.



# Associando informações à Sessão

- Para ler uma informação associada à uma **session**, usa-se a função **getAttribute()**.
- Para se gravar informação, usa-se a função **setAttribute()**.
- Esta função efetua um **update** nos valores prévios.
- Para remover-se a informação sem fornecer um substituto, usa-se a função **removeAttribute()**;



# Acesso às informações da Sessão

- Usa-se **session.getAttribute("key")** para pesquisa de valores previamente armazenados na sessão;
- O tipo retornado é **Object**. Assim, deve-se efetuar operação de **casting** para acerto de tipos de dados;
- Valor retornado é **null** caso a chave não exista.



## Acesso às informações da Sessão



```
HttpSession session=request.getSession();  
SomeClass value =  
    (SomeClass)session.getAttribute("identificador");  
if (value == null){  
    value = new SomeClass(...);  
    session.setAttribute("identificador",value);  
}  
facaAlgocomValor(value);
```



# Acesso às informações da Sessão

- Na maioria das vezes, temos um nome de atributo específico para recuperar;
- No entanto, podemos descobrir todos os nomes de atributos definidos em uma dada sessão por meio da chamada da função **getAttributeNames()** a qual retorna um objeto do tipo **Enumeration**.



## Descartando informações da Sessão

- Para se remover somente os dados criados no servlet, pode-se usar a função **`removeAttribute("key")`** ;
- Ao se usar esta função, os dados associados à chave especificada são descartados;
- Esta é a abordagem mais comum.



## Descartando informações da Sessão

- Pode-se deletar a sessão inteira na aplicação WEB corrente;
- Neste caso, usa-se a função **invalidate()** para se descartar os dados de toda a sessão;
- Cuidado! Lembre-se que ao fazer isto [todos os dados da sessão de seu Servlet ou JSP serão destruídos!](#)





## Atividade – 13

1. Escrever um servlet (**Atividade\_13**) que exibe informações básicas da sessão de um cliente;
2. Quando o cliente se conectar, o servlet usa **request.getSession()** ou para recuperar a sessão existente ou, para criar uma nova;
3. O servlet então pesquisa um atributo chamado **accessCount** do tipo **integer**. Se não encontrar o atributo, inicializa-o com o valor **0**;
4. Este valor é então incrementado e associado à sessão por meio da função **setAttribute()**;
5. Finalmente, o servlet exibe um tabela HTML mostrando informações da sessão.



```
package qualit;  
  
import java.io.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.util.*;  
  
public class Atividade_13 extends HttpServlet {  
  
    public void doGet(HttpServletRequest request,  
        HttpServletResponse response)  
        throws ServletException, IOException {  
  
        response.setContentType("text/html");  
        HttpSession session = request.getSession();  
        String heading;  
        Integer accessCount =  
            (Integer) session.getAttribute("accessCount");  
        if (accessCount == null) {  
            accessCount = new Integer(0);  
            heading = "Olá Cliente! Seja bem vindo....";  
        }  
        else {  
            heading = "Bemvindo novamente.... ";  
            accessCount = new Integer(accessCount.intValue() + 1);  
        }  
    }  
}
```



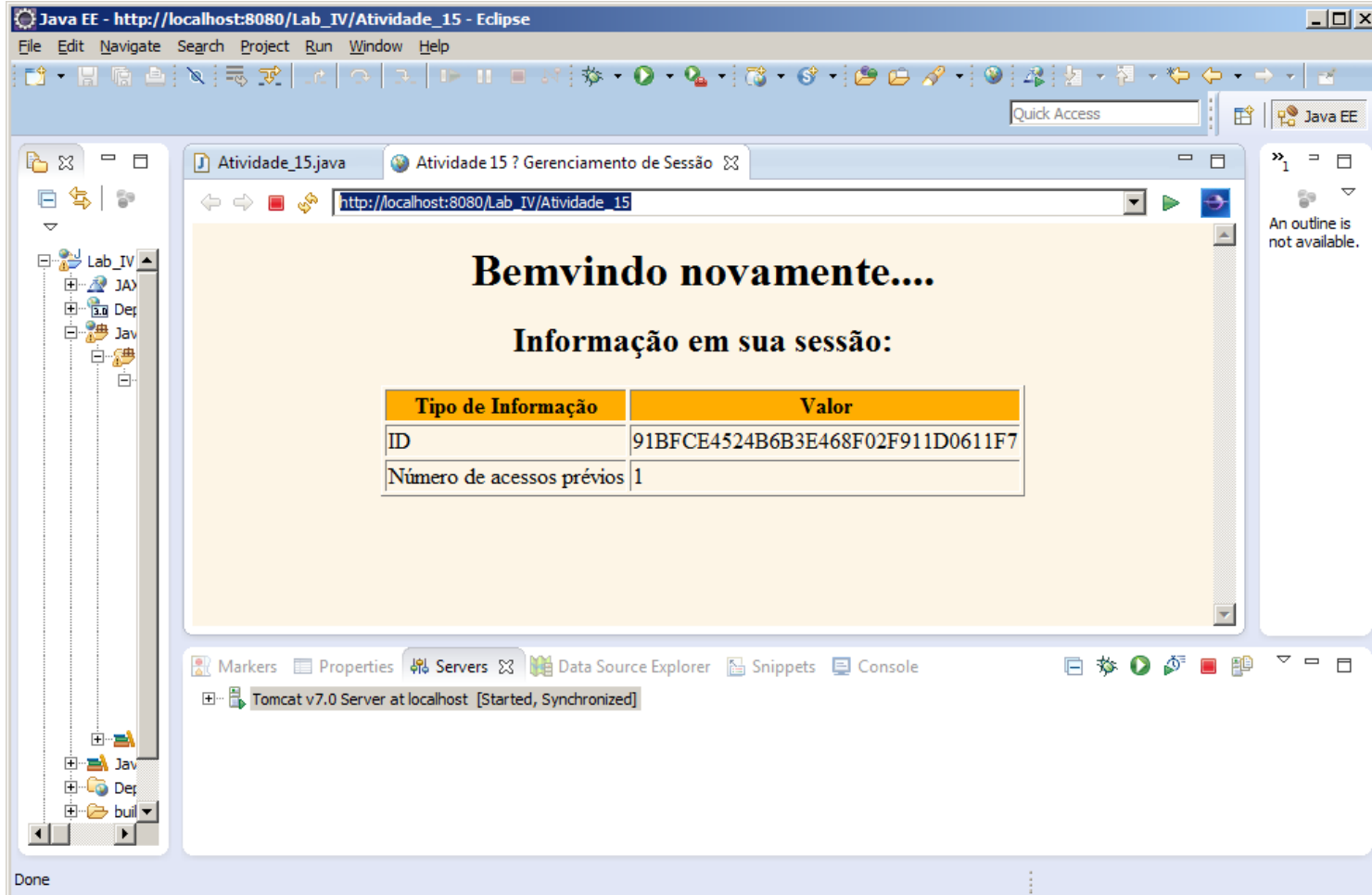
```
session.setAttribute("accessCount", accessCount);  
  
PrintWriter out = response.getWriter();  
  
String title = "Atividade 13 - Gerenciamento de Sessão";  
  
String docType =  
    "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 \" +  
    \"Transitional//EN\">\n";
```



```
out.println(docType +
    "<HTML>\n" +
    "<HEAD><TITLE>" + title + "</TITLE></HEAD>\n" +
    "<BODY BGCOLOR=\"#FDF5E6\"\>\n" +
    "<CENTER>\n" +
    "<H1>" + heading + "</H1>\n" +
    "<H2> Informação em sua sessão: </H2>\n" +
    "<TABLE BORDER=1>\n" +
    "<TR BGCOLOR=\"#FFAD00\"\>\n" +
    "  <TH> Tipo de Informação <TH>Valor \n" +
    "<TR>\n" +
    "  <TD>ID\n" +
    "  <TD>" + session.getId() + "\n" +
    "<TR>\n" +
    "  <TD> Número de acessos prévios \n" +
    "  <TD>" + accessCount + "\n" +
    "</TABLE>\n" +
    "</CENTER></BODY></HTML>");
}
```



[http://localhost:8080/Lab\\_IV/Atividade\\_13](http://localhost:8080/Lab_IV/Atividade_13)



Java EE - [http://localhost:8080/Lab\\_IV/Atividade\\_15](http://localhost:8080/Lab_IV/Atividade_15) - Eclipse

File Edit Navigate Search Project Run Window Help

Quick Access

Atividade\_15.java Atividade 15 ? Gerenciamento de Sessão

[http://localhost:8080/Lab\\_IV/Atividade\\_15](http://localhost:8080/Lab_IV/Atividade_15)

**Bemvindo novamente....**

**Informação em sua sessão:**

Tipo de Informação	Valor
ID	91BFCE4524B6B3E468F02F911D0611F7
Número de acessos prévios	1

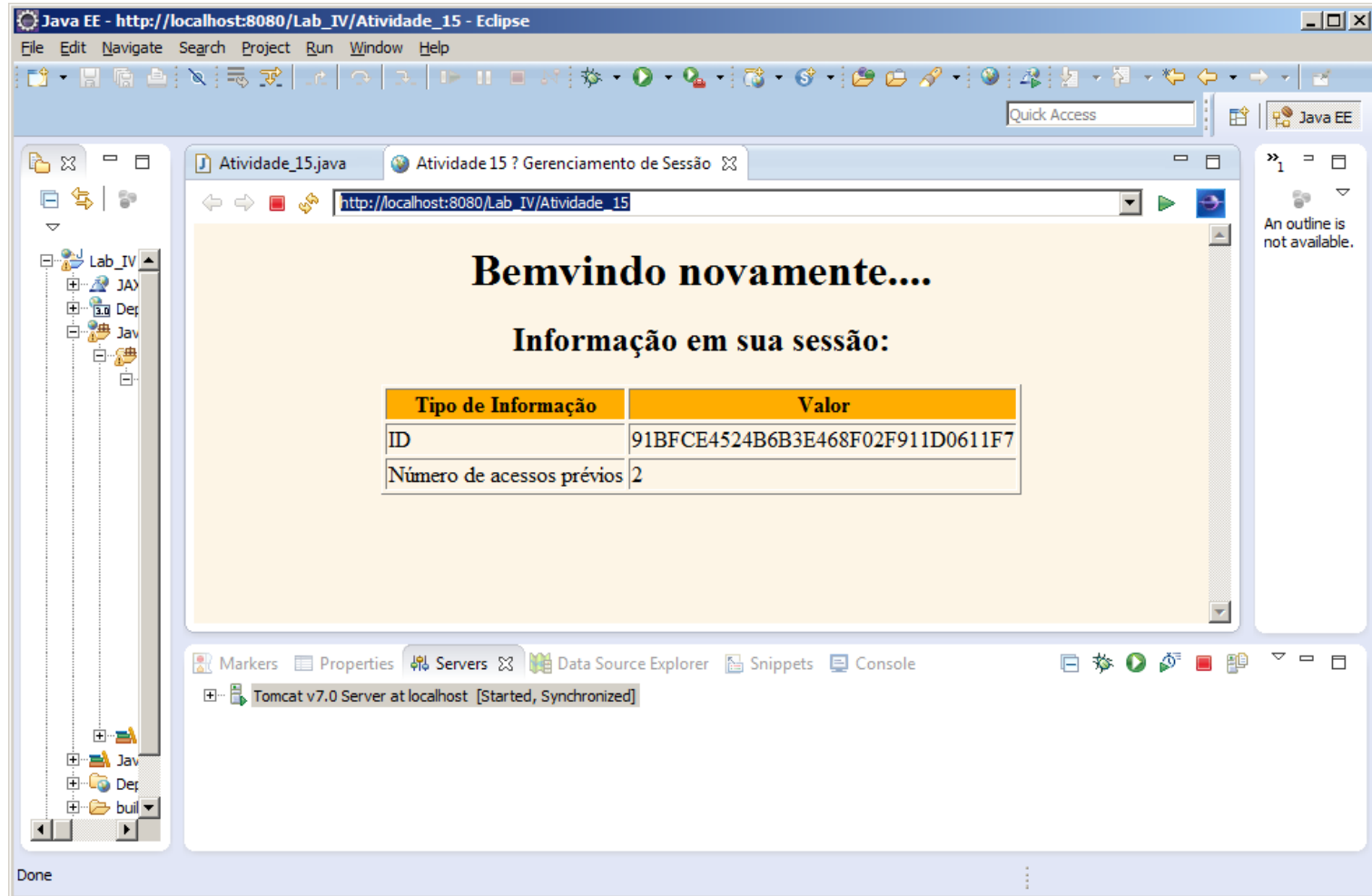
Markers Properties Servers Data Source Explorer Snippets Console

Tomcat v7.0 Server at localhost [Started, Synchronized]

Done



[http://localhost:8080/Lab\\_IV/Atividade\\_13](http://localhost:8080/Lab_IV/Atividade_13)



Java EE - [http://localhost:8080/Lab\\_IV/Atividade\\_15](http://localhost:8080/Lab_IV/Atividade_15) - Eclipse

File Edit Navigate Search Project Run Window Help

Quick Access

Atividade\_15.java Atividade\_15 ? Gerenciamento de Sessão

[http://localhost:8080/Lab\\_IV/Atividade\\_15](http://localhost:8080/Lab_IV/Atividade_15)

**Bemvindo novamente....**

**Informação em sua sessão:**

Tipo de Informação	Valor
ID	91BFCE4524B6B3E468F02F911D0611F7
Número de acessos prévios	2

Markers Properties Servers Data Source Explorer Snippets Console

Tomcat v7.0 Server at localhost [Started, Synchronized]

Done

