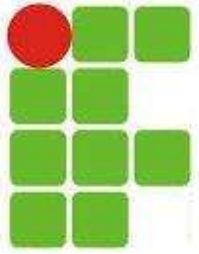

Programação Web

Professor: Diego Oliveira



**Conteúdo 11:
Controle de Sessão**





Controle de Sessão

- Servidores Web não armazenam informações de um cliente entre uma sessão e outra
- Há duas maneiras de guardar informações sobre as ações do cliente na página:
 - Fazer com que o cliente guarde as informações
 - Fazer com que o servidor as guarde





Controle de Sessão

- A primeira abordagem é mais simples e não exige ativação de capacidades especiais no lado do servidor
- A desvantagem é que exige a transmissão de dados do cliente para o servidor, o que diminui o desempenho





Controle de Sessão

- A segunda abordagem, do lado servidor, oferece mais funcionalidades
- Uma vez iniciada a sessão, o cliente a aceita e o servidor mantém as informações, precisando apenas de uma CHAVE para distinguir uma sessão de outra





Controle de Sessão

- Há quatro técnicas utilizadas para manter informações sobre sessões no lado servidor:
 - Campos Ocultos
 - Reescrita de URL
 - Cookies
 - Utilizando a API de Sessão HTTP
- A seguir veremos detalhadamente cada uma dessas quatro técnicas juntamente com um exemplo!





Campos Ocultos

- Basicamente são os campos `<INPUT TYPE="HIDDEN">`
- Estes campos não aparecem nos formulários das páginas, porém podem guardar informações
- Já utilizamos estes campos para fazer formulários divididos em várias páginas utilizando modelo WIZARD (next, next...)





Campos Ocultos

- Para recuperar o valor de *dados* e também para adicionar mais conteúdo nesta variável utilizamos este código anteriormente:

```
<%  
    String valor = request.getParameter("valor");  
    String dados = request.getParameter("dados");  
    String botoes = request.getParameter("adicionar");  
    String elementos[] = null;  
    if(dados != null){  
        dados += "," + botoes;  
        elementos = dados.split(",");  
    }  
>%
```





Campos Ocultos

```
<form action="index.jsp" method="post">
  <select name="adicionar">
    <option value="button">button</option>
    <option value="radio">radio</option>
    <option value="checkbox">checkbox</option>
    <option value="textarea">textarea</option>
  </select>
  <select name="valor">
    <option value="1">1</option>
    <option value="2">2</option>
    <option value="3">3</option>
  </select>
  <input type="hidden" value="<%=dados%>" name="dados" />
  <input type="submit" value="ADICIONAR" name="enviar" />
</form>
```





Reescrita de URL

- Como já sabemos, uma URL pode ter parâmetros
- Eles são formados por chaves 'nome=valor' separados por um caractere de '&'
- Para recuperar os valores no JSP fazemos:
 - `String nome1 = request.getParameter("nome1");`
- Páginas geradas dinamicamente podem 'lembrar' das informações utilizando esta técnica simples





Reescrita de URL

- Para repassar dados pela URL, podemos utilizar o atributo HREF dos links:
 - ``
 - ``
 - `<A HREF="pagina3.jsp?valor=<%=valor%>">`
- Para resgatar os valores nas páginas seguintes basta utilizarmos:
 - `String contador = request.getParameter("contador");`
 - `String nome = request.getParameter("nome");`
 - `String valor = request.getParameter("valor");`

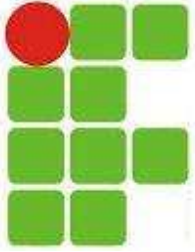




Cookies

- É a técnica mais utilizada para manter informações de sessões
- Um cookie é um pequeno arquivo enviado ao cliente pelo servidor que contém informações sobre o acesso
- Além do nome e valor, podem conter:
 - Uma data de validade
 - Um nome de domínio
 - Um caminho que restringe a URL
 - Um atributo de segurança (SSL)





Cookies

- O maior problema da utilização de cookies é que o usuário pode desativar sua utilização pelo navegador
- Normalmente isso é feito por questões de privacidade ou segurança
- Isso deve ser levado em consideração no momento da escolha do meio de persistência de informações no seu site





Cookies

- Salvando valores nos Cookies:

```
<%  
Cookie cookie = new Cookie("nome", "Diego");  
Cookie cookie2 = new Cookie("sobrenome", "Oliveira");  
cookie.setMaxAge(60*60); //60 minutos  
cookie2.setMaxAge(60*60); //60 minutos  
response.addCookie(cookie);  
response.addCookie(cookie2);  
%>
```





Cookies

- Resgatando valores dos Cookies:

```
<%  
Cookie cookie = null;  
Cookie[] cookies = null;  
cookies = request.getCookies();  
if (cookies != null) {  
    out.println("<h2>Cookies encontrados :) </h2>");  
    for (int i = 0; i < cookies.length; i++) {  
        cookie = cookies[i];  
        out.print("Nome : " + cookie.getName() + ", ");  
        out.print("Valor: " + cookie.getValue() + " <br/>");  
    }  
} else {  
    out.println("<h2>Sem cookies :( </h2>");  
}
```

```
<%>
```



API HTTP Session

- Por padrão o rastreamento de sessão no JSP é ativado
- Cada vez que um cliente novo se conecta um novo objeto HttpSession é instanciado
- Para desabilitar o rastreamento de sessão:
 - `<%@ page session="false" %>`
- Há vários métodos importantes no objeto HttpSession, vamos vê-los nos slides seguintes

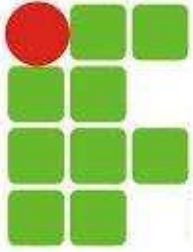




API HTTP Session

- `public Object getAttribute(String name)`
 - Retorna um objeto com o nome especificado
- `public Enumeration getAttributeNames()`
 - Retorna todos os objetos da sessão
- `public long getCreationTime()`
 - Retorna o horário em que a sessão foi criada
- `public String getId()`
 - Retorna o ID da sessão atual
- `public long getLastAccessedTime()`
 - Retorna o horário do último acesso





API HTTP Session

- `public int getMaxInactiveInterval()`
 - Retorna o tempo máximo de inatividade permitido
- `public void invalidate()`
 - Invalida a sessão atual
- `public boolean isNew()`
 - Retorna verdadeiro se o cliente é novo na sessão
- `public void removeAttribute(String name)`
 - Remove um objeto associado à sessão
- `public void setAttribute(String name, Object value)`
 - Define um objeto associado à sessão
- `public void setMaxInactiveInterval(int interval)`
 - Define o tempo máximo de inatividade permitido





API HTTP Session

- Verificando se é o primeiro acesso:

```
<%  
Date createTime = new Date(session.getCreationTime());  
Date lastAccessTime = new Date(session.getLastAccessedTime());  
String title = "Seja Bem Vindo Novamente";  
String userIDKey = new String("userID");  
String userID = new String("Diego Oliveira");  
if (session.isNew()) {  
    title = "Bem Vindo!";  
    session.setAttribute(userIDKey, userID);  
}  
userID = (String)session.getAttribute(userIDKey);  
%>
```





API HTTP Session

- Verificando se é o primeiro acesso:

```
<html>
<head><title>Session Tracking</title></head>
<body>
<center>
<h1><%out.println(title) ;%></h1>
</center>
<table border="1" align="center">
<tr bgcolor="GRAY"><th>Session info</th><th>Value</th></tr>
<tr><td>id</td><td><% out.print( session.getId()); %></td></tr>
<tr><td>Creation Time</td><td><% out.print(createTime); %></td></tr>
<tr><td>Time of Last Access</td><td><% out.print(lastAccessTime); %></td></tr>
<tr><td>User ID</td><td><% out.print(userID); %></td></tr>
</table>
</body>
</html>
```





API HTTP Session

- Verificando se é o primeiro acesso:

Bem Vindo! ←

| Session info | Value |
|---------------------|------------------------------------|
| id | A1AA861BCB0D09DADF470F443E7063A7 |
| Creation Time | Mon Jan 18 12:33:55 GMT-03:00 2016 |
| Time of Last Access | Mon Jan 18 12:33:55 GMT-03:00 2016 |
| User ID | Diego Oliveira |


Seja Bem Vindo Novamente

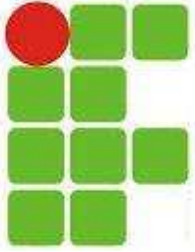
| Session info | Value |
|---------------------|------------------------------------|
| id | A1AA861BCB0D09DADF470F443E7063A7 |
| Creation Time | Mon Jan 18 12:33:55 GMT-03:00 2016 |
| Time of Last Access | Mon Jan 18 12:34:50 GMT-03:00 2016 |
| User ID | Diego Oliveira |





Exercício

- Crie um conjunto de formulários de cadastro de um novo usuário no site contendo:
 - Informações Pessoais (next usando hidden)
 - Informações Acadêmicas (next)
 - Informações Profissionais (next)
- Salve as principais informações do formulário utilizando Cookies
- Guarde informações da sessão como login e horário de acesso usando HttpSession
-  Mostre as informações em uma página!



Perguntas?

