Capítulo 8

ASP.NET

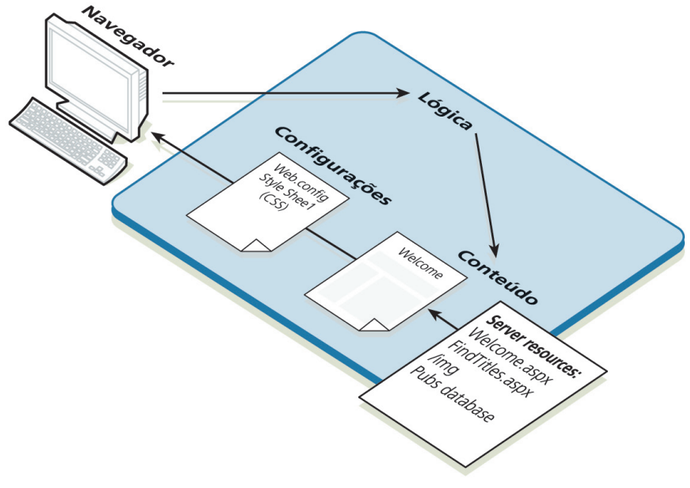
- Aplicação ASP.NET
- Eventos
- HTML Server Controls e Web Server Controls
- Sessões em ASP.NET
- Dados via URL

ASP.NET é uma plataforma de desenvolvimento usada para a construção de aplicações Web e Web Service, as quais serão executadas por um servidor, que, nesse caso, é o IIS (Internet Information Service – serviço de informação de Internet) da Microsoft. Não se trata apenas de uma tecnologia de desenvolvimento, mas de um conjunto de ferramentas que permite a integração entre servidores Microsoft, segurança, código compilado, e acesso a bancos de dados via ADO.NET e ao .NET Framework, bem como suporte total ao XML.

8.1. Aplicação ASP.NET

Uma aplicação ASP.NET é constituída por três partes (figura 304):
Conteúdo: arquivos do tipo Web Forms, HTML, imagens, áudio e vídeo, que determinam a aparência de uma aplicação Web.
Lógica: arquivos executáveis e script, que determinam como uma aplicação responderá às ações dos usuários.

Figura 304
Funcionamento de um Web Service.



Configuração: via arquivos WebConfig e CSS , que determinam como a aplicação vai ser executada.

Um dos principais caminhos para criar uma interface entre o usuário e a aplicação ASP.NET é o Web Form, cuja parte executável é armazenada em um assembly (.dll), porém, é executada no servidor e controlada por um work process (aspnet_wp.exe). Funciona em parceria com o **IIS**, que inicia a execução do ASP.NET (aspnet_wp.exe) carregando o assembly do Web Form, o qual, por sua vez, constrói a resposta para o usuário de acordo com sua requisição e envia uma resposta no formato HTML (figura 304).

Como ilustra a figura 304, partir da ação do usuário, o navegador cria a requisição de uma página Web ou ativa um determinado serviço. Isso faz com que a parte lógica da página entre em ação. A resposta será no formato HTML, devidamente configurada como, por exemplo, a forma gráfica que será apresentada ao usuário (formatação).

8.1.2. Web Form

Um Web Form pode conter os seguintes componentes: **Server Controls** (controles de servidor): TextBox, Label e Button, que permitem controlar e responder a determinados eventos do servidor. **HTML Controls** (controles de HTML): TextArea, Table e Image, que representam os elementos padrões do HTML. **Data Controls** (controles de dados): SqlConnection, SqlCommand, OleDbConnection, OleDbCommand e DataSet, que fornecerão mecanismos para manipulação de arquivos XML e conexão com bancos de dados (SQL). **System Components** (componentes de sistema): EventoLog, Message Queue e FileSystemWatcher, os quais permitem manipular eventos do servidor.

8.1.3. Projeto Web Application (Aplicação Web)

Para iniciar uma Web Application, escolha, em Project Types, Visual Basic, e depois a opção Web na template ASP.NET Web Application (figura 305). Não

O IIS é um conjunto integrado de serviços de rede para a plataforma Windows, criado pela Microsoft. Sua primeira versão surgiu com o Windows NTServer 4 e passou por várias atualizações.
A versão de 2009 é o IIS 7.5 (disponível no Windows Server 2008 R2 e Windows 7). Uma de suas características mais utilizadas é a geração de páginas HTML dinâmicas, que, diferentemente de outros servidores web, funciona com tecnologia proprietária ASP (Active Server Pages, páginas de servidor ativas), mas também pode usar outras tecnologias com a adição de módulos de terceiros. Para ter acesso a essa ferramenta, é necessário adquirir licença de uso. E para cada instalação ou versão é exigido um pagamento. Depois do lançamento da plataforma .NET em 2002, o IIS ganhou também a função de gerenciar o ASP.NET, formado basicamente por dois tipos de aplicações: Páginas Web (acessadas por usuários com a extensão ASPX) e Web Services (funções disponibilizadas pela rede, chamadas de aplicativos ASMX).

Figura 305
Aplicação ASP.NET.

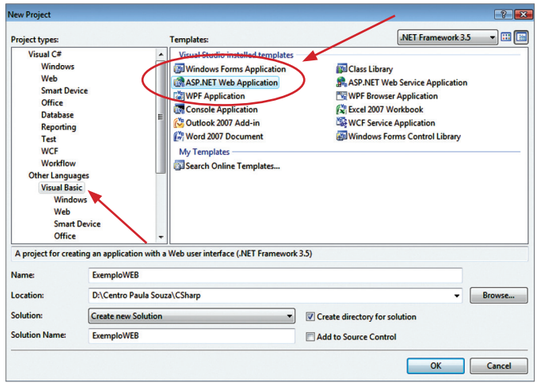
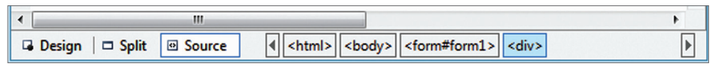


Figura 306
WebForm.



se esqueça de indicar o nome da Solution, que, nesse caso, é ExemploWeb, e o local em que ela será gravada.

Analisando a Solution Explorer, novos arquivos serão disponibilizados para aplicação em ASP, como o Default.aspx. A descrição visual de um Web Form que, uma vez ativado, permite visualizar na janela Code as opções Design, Split e Source (figura 306). Elas permitem modificar a janela de desenvolvimento dentro de três opções: design; design e código, e código. Além disso, possibilita o controle das tags de marcação.

8.1.4. Ciclo de vida da Aplicação

Uma aplicação Web começa no momento em que o navegador faz a requisição na sua página principal. Inicialmente, executa o assembly (.dll), criando uma instância do Web Form. Essa, por sua vez, responderá à requisição do usuário no formato HTML, sendo destruída posteriormente. Caso o usuário não realize nenhuma requisição, a instância poderá ser excluída depois de um determinado tempo.

8.1.5. Executando uma Application Service (Serviço de Aplicação)

Execute uma Application Service da mesma forma que faz em um Windows Form Application. Dessa vez, porém, não teremos a aplicação rodando como anteriormente, mas sim uma porta lógica que será criada para o servidor IIS. O navegador padrão da máquina será ativado para mostrar a aplicação, conforme ilustra a figura 307. Podemos observar que aparece na URL a expressão localhost, indicando que o serviço está operando como um servidor local. O número que surge após essa expressão representa a porta lógica da máquina, o qual não é padrão e poderá ser modificada a cada execução.

Toda e qualquer modificação no código deverá ser gravada por meio da atualização da página e poderá ser visualizada posteriormente. Para isso, basta pressionar a tecla F5. Caso seja usado o botão break, a janela do navegador será fechada.

Figura 307
Servidor em Execução.

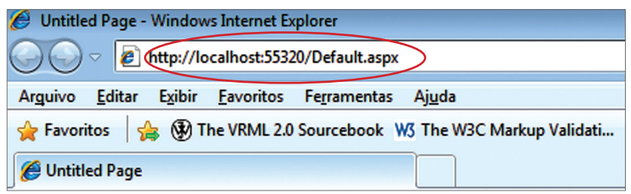
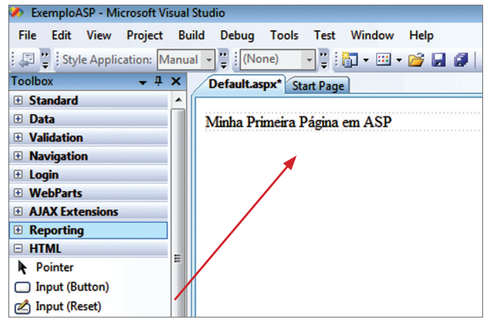


Figura 308
ToolBox para Aplicação Web.



8.1.6. Escrevendo a Aplicação

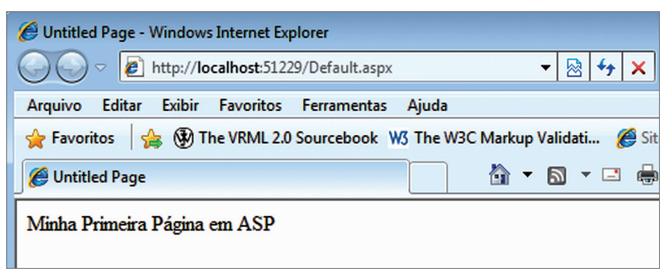
Assim como nas aplicações em C# ou VB.NET, podemos utilizar a ToolBox para arrastar os componentes usados no desenvolvimento do layout da página Web. Para isso, é preciso mudar a visão para Design (figura 308).

Se preferir ir diretamente ao código, mude para Source, escreva e grave o seu código e verifique o resultado final no navegador (figuras 309 a e b).

Figura 309a
Verificando o resultado final.

```
<html xmlns=”http://www.w3.org/1999/xhtml” >
<head runat=”server”>
  <title>Untitled Page</title>
</head>
<body>
  <form id=”form1” runat=”server”>
  <div>
    Minha Primeira Página em ASP</div>
  </form>
</body>
</html>
```

Figura 309b
Minha primeira aplicação.



8.1.7. Estruturando uma página ASP.NET

Antes, as páginas em ASP utilizavam as tags "<%...%>" para a inclusão do script, como podemos constatar no código ilustrado na figura 310.

Figura 310
Uso de tags "<%...%>".

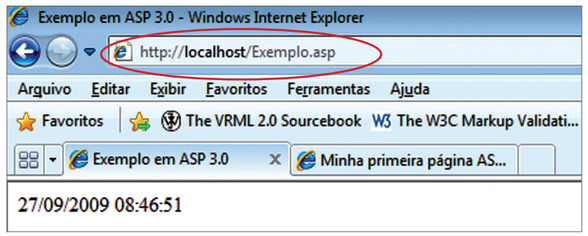
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Exemplo em ASP 3.0</title>
</head>
<% response.write(now()) %>
<body>
</body>
</html>
```

HTML Server Controls (Controles de servidor HTML): tags HTML tradicionais.
Web Server Controls (Controles de servidor Web): novas tags ASP.NET.
Validation Server Controls (Controles de servidor de validação): validação da entrada de dados.

O resultado pode ser visto na figura 311. Lembre-se de executar o código usando o IIS (localhost).

Dessa forma, as tags marcadoras do ASP eram colocadas onde se desejava que o código aparecesse. Elas ficavam misturadas com o HTML, gerando um código de difícil leitura e manutenção, que foi batizado de código espaguetti. O ASP.NET eliminou esse problema, utilizando os Server controls, ou seja, tags que podem ser interpretadas diretamente pelo servidor e estão divididas em três categorias de Server Controls.

Figura 311
Execução do ASP.



8.1.7.1. HTML Server Controls

São tags HTML padrão, criadas a partir da inclusão de um novo atributo runat="server". Veja o exemplo no código mostrado na figura 312.

Figura 312
Tags com inclusão de atributo runat="server".

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Exemplo em ASP - VB</title>
</head>
<% Tempo.InnerText = Now()%>
<body>
  <p id="Tempo" runat="server"></p>
</body>
</html>
```

A tag <p> tornou-se um controle do servidor. Sua identificação foi representada pelo atributo id, que permitirá futuras referências por meio do código executável, permanecendo agora fora do HTML.

8.1.7.2. Web Server Controls

São semelhantes ao HTML Server Controls, pois agem como parte de um código HTML. No entanto, são independentes e podem ser utilizados em aplicações interativas, o que significa que eles englobam serviços como, por exemplo, a programação de eventos. Isso porque dispõem de componentes de calendário, gridview, treeview, entre outros. Os Web Server Controls são mais complexos que o HTML Server Controls e devem iniciar com <asp:.../>, como mostra o código ilustrado pela figura 313.

Figura 313
Código com uso de <asp:.../>.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Exemplo em ASP - VB</title>
</head>
<% Tempo.Text = Now()%>
<body>
  <p></p><asp:label id="Tempo" runat="server"/></p>
</body>
</html>
```

8.1.7.3. Validation Server Controls

Se voltarmos ao conceito das tags em HTML, lembraremos que o grande problema era o controle de entrada de dados, ou seja, aquilo que o usuário poderia digitar ou não. Pois os controles de validação do servidor (validation server controls) permitem justamente validar essas entradas e ainda exibem mensagens. Cada controle executa uma validação específica, após uma ação do usuário por meio de controle Button, ImageButton, ou LinkButton.

8.2. Eventos

Para que o código seja executado no momento correto, podemos utilizar os manipuladores de evento. O código seguinte determinará o momento em que a verificação da data/hora deverá ser lida no servidor (figura 314). O manipulador

realizará uma sub-rotina quando determinado evento ocorrer. No exemplo anterior, o evento Page_Load é executado quando uma página é carregada.

Figura 314
Verificação de data e hora.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Exemplo em ASP - VB</title>
</head>
<script runat="server">
 Sub Page_Load(Sender As Object,E As EventArgs)
    Tempo.Text = Now()
 End Sub
</script>
<body>
  <p><asp:label id="Tempo" runat="server"/></p>
</body>
</html>
```

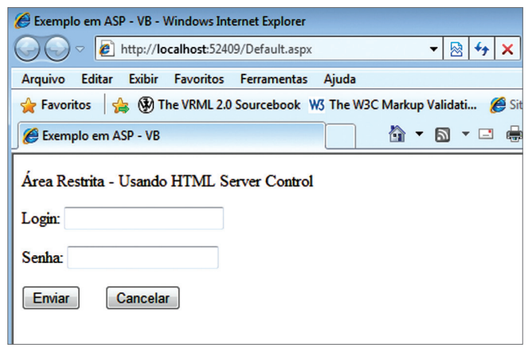
8.3. HTML Server Controls e Web Server Controls

A construção de uma página Web poderá ser feita de duas formas: por meio do HTML Server Controls ou do Web Server Controls. Isso depende diretamente dos componentes da Toolbox. Não existe regra para se utilizar um ou outro. Porém, é recomendável optar pelo HTML Controls, em caso de migração do ASP para o ASP.NET. Já o Web Controls é indicado para situações em que os componentes precisam de mais funcionalidade, pois ele permite a programação e incorporação de recursos do Framework.

8.3.1. HTML Server Controls

O exemplo ilustrado pelas figuras 315 a e b mostra uma entrada simples de login e senha.

Figura 315a
Usando Server Control.




```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Exemplo em ASP - VB</title>
</head>
<body>
 <p id="lblLogin">Área Restrita - Usando Web Control</p>
 <p>Login:
   <input id="txtLogin" type="text" /></p>
 <p id="lblSenha">
   Senha:
   <input id="pwdSenha" type="password" /></p>
 <p>
 <input id="btnEnviar" type="button" value="Enviar" />
    
  <input id="btnCancelar" type="button" value="Cancelar" /></p>
</body>
</html>
```

Figura 315b
Detalhamento do uso do Server Control.

Assim como ocorreu em C# e VB.NET, a janela de Properties deverá ser utilizada para ajustes e identificação dos componentes (figura 316).

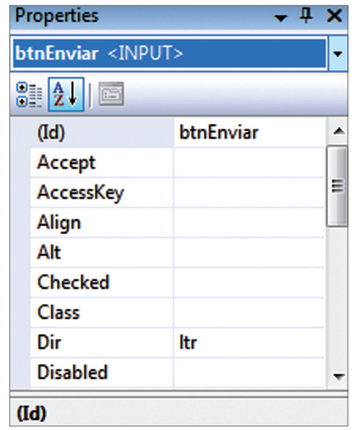


Figura 316
Janela Properties.

8.3.2. Web Server Controls

Usando o mesmo exemplo do item anterior, agora com o Web Server Controls, podemos verificar que não existe mudança de layout (figuras 317 a e b).

Figura 317a
Web Server Controls.

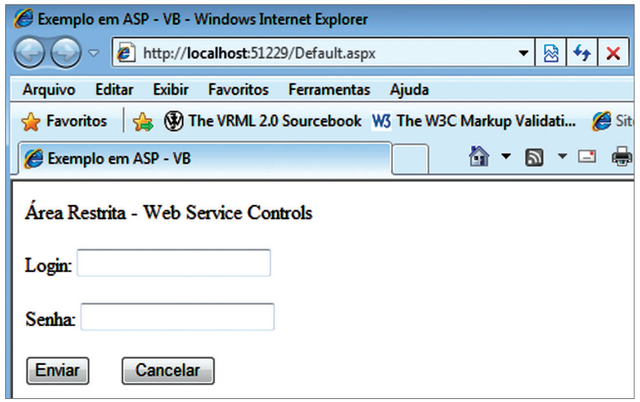


Figura 317b
Detalhamento do login e senha.

```
<html xmlns=”http://www.w3.org/1999/xhtml” >
<head runat=”server”>
  <title>Exemplo em ASP - VB</title>
</head>
<body>
  <form id=”form1” runat=”server”>
  <p id=”lblLogin”>Área Restrita - Web Service Controls</p>
  Login:
  <asp:TextBox ID=”TextBox1” runat=”server”></asp:TextBox>
  <p>
    Senha:
    <asp:TextBox ID=”TextBox2” runat=”server”></asp:TextBox>
  </p>
  <asp:Button ID=”btnEnviar” runat=”server” Text=”Enviar” />
    
  <asp:Button ID=”btnCancelar” runat=”server” Text=”Cancelar” />
</form>
</body>
</html>
```

8.4. Sessões em ASP.NET

Em alguns momentos, será necessário deixar um valor disponível para toda aplicação. Isso é fácil quando se trata de programação para ambiente Windows, como o VB.NET. Mas, no caso de uma aplicação Web, temos que utilizar os recursos das variáveis de sessão, ou seja, definir uma chave e atribuir um valor para ela. As sessões são criadas pelo comando Session(), que identifica as variáveis criadas pelo programador. Exemplo:

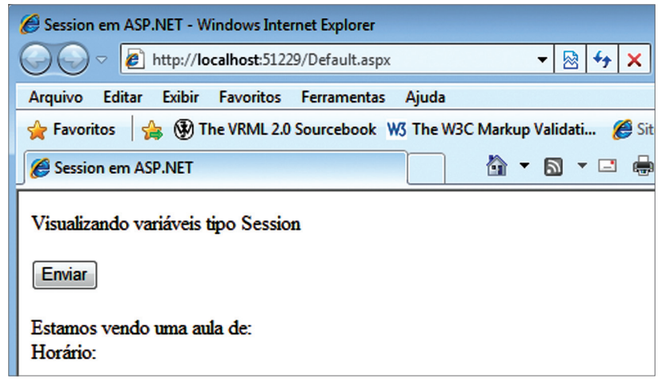


Figura 318
Tela antes de pressionar o botão Enviar.

```
Session(“aula”) = “ASP.NET”
Session(“data”) = Now()
```

Após a definição da palavra-chave da sessão, será determinado o seu valor. A recuperação dos dados será feita da mesma forma, identificando a palavra-chave. No exemplo da figura 318, vamos criar as duas variáveis mencionadas e recuperá-las após o clique do botão.

Ao clicarmos no botão Enviar, os itens referentes a Estamos vendo uma aula de: e Horário: serão preenchidos de acordo com o código mostrado na figura 319.

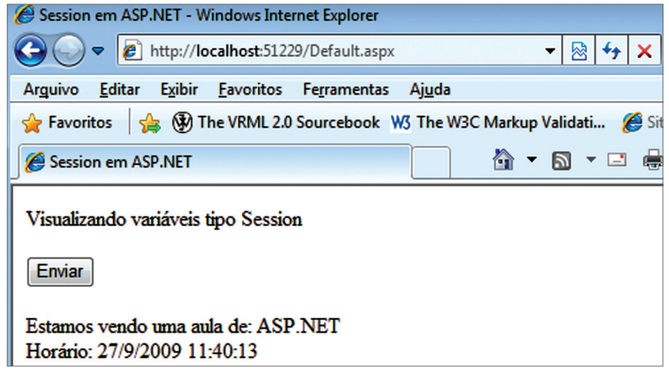
```
<html xmlns=”http://www.w3.org/1999/xhtml” >
<head runat=”server”>
  <title>Session em ASP.NET</title>
</head>
<script runat=”server”>
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    Session(“aula”) = “ASP.NET”
    Session(“data”) = Now()
  End Sub

  Protected Sub btnEnviar_Click(ByVal sender As Object, ByVal e As
System.EventArgs)
    Mostrar1.Text = Session(“aula”)
```

Figura 319
Preenchendo itens.

```
    Mostrar2.Text = Session("data")
  End Sub
</script>
<body>
  <form name="formulario" runat="server">
  Visualizando variáveis tipo Session<br />
  <br/>
  <asp:Button ID="btnEnviar" runat="server" Text="Enviar"
onclick="btnEnviar_Click" /><br/>
  <br/> Estamos vendo uma aula de:
  <asp:Label ID="Mostrar1" runat="server" Text=""></asp:Label>
  <br/>Horário:
  <asp:Label ID="Mostrar2" runat="server" Text=""></asp:Label>
  </form>
</body>
</html>
```

As variáveis de sessão foram criadas no evento Page_Load e a recuperação dos valores ocorrerá quando o evento btnEnviar_Click for acionado (figura 320).

Figura 320
Variável Session carregada.



8.4.1. Recuperando sessão em outra página por meio de um evento

Depois que uma sessão foi criada, podemos recuperar os valores em outra página. Vamos utilizar o exemplo anterior e dividi-lo em duas páginas – a primeira para criação da sessão e a segunda para visualização. Na primeira, vamos retirar o botão e incluir um link para que o usuário seja direcionado para a próxima página (figura 321).

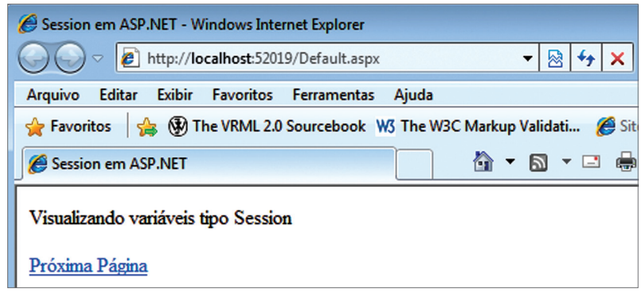


Figura 321
Link para visualização.

No código principal, podemos verificar somente a criação das chaves (figura 322).

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
  <title>Session em ASP.NET</title>
</head>
<script runat="server">
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    Session("aula") = "ASP.NET"
    Session("data") = Now()
  End Sub
</script>
<body>
  <form id="Form1" name="formulario" runat="server">
  Visualizando variáveis tipo Session<br />
  <br/>
  <asp:linkbutton runat="server" PostBackUrl="Recuperar.
aspx">Próxima Página</asp:linkbutton>
  </form>
</body>
</html>
```

Figura 322
Código principal com criação das chaves.

De acordo com o código mostrado no quadro anterior, o link aponta para um arquivo chamado Recuperar.aspx, que deverá mostrar o conteúdo da sessão. Após o clique no botão, observe o resultado (figura 323).

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
  <title>Session em ASP.NET</title>
</head>
<script runat="server">
```

Figura 323
Resultado após o clique no botão.

```
  Protected Sub btnEnviar_Click(ByVal sender As Object, ByVal e As
System.EventArgs)
    Mostrar1.Text = Session("aula")
    Mostrar2.Text = Session("data")
  End Sub
</script>
<body>
  <form id="Form1" name="formulario" runat="server">
  Clique no Botão Para Recuperar os Dados:<br />
  <br/>
  <asp:Button ID="btnEnviar" runat="server" Text="Enviar"
onclick="btnEnviar_Click" /><br/>
  <br/> Estamos vendo uma aula de:
  <asp:Label ID="Mostrar1" runat="server" Text=""></asp:Label>
  <br/>Horário:
  <asp:Label ID="Mostrar2" runat="server" Text=""></asp:Label>
  </form>
</body>
</html>
```

8.4.2. Recuperando sessão em outra página automaticamente

É importante que as variáveis de sessão sejam recuperadas automaticamente, para que possam ser utilizadas nas páginas subsequentes. Vamos utilizar o código do exemplo anterior e eliminar o botão para visualização (figuras 324 a e b).

Figura 324a
Recuperação direta.

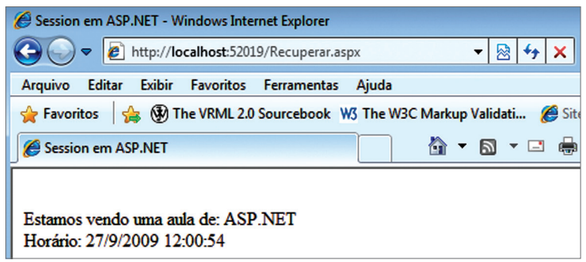


Figura 324b
Eliminado o botão para visualização.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
  <title>Session em ASP.NET</title>
</head>
<script runat="server">
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    Mostrar1.Text = Session("aula")
    Mostrar2.Text = Session("data")
  End Sub
</script>
<body>
  <form id="Form1" name="formulario" runat="server">
  <br/> Estamos vendo uma aula de:
  <asp:Label ID="Mostrar1" runat="server" Text=""></asp:Label>
  <br/>Horário:
  <asp:Label ID="Mostrar2" runat="server" Text=""></asp:Label>
  </form>
</body>
</html>
```

8.5. Dados via URL

Outro mecanismo para transportar informações de uma página Web para outra é o URL (Uniform Resource Locator ou localizador de recurso universal). Pode-se utilizar o seguinte artifício: após o nome do arquivo que receberá as informações, digite o sinal de interrogação ? e indique as variáveis e seus respectivos conteúdos. Se houver mais de uma variável, separe com o sinal de &, como no exemplo seguinte:

```
"Menu.aspx?op=1"
"Menu.aspx?op=1&nome=José&cod=3456"
```

Para receber as informações, usaremos o comando Request.QueryString() no evento Page_Load da página que receberá os dados, indicando o nome da variável que se deseja recuperar. A figura 325 mostra um exemplo de menu de opções.

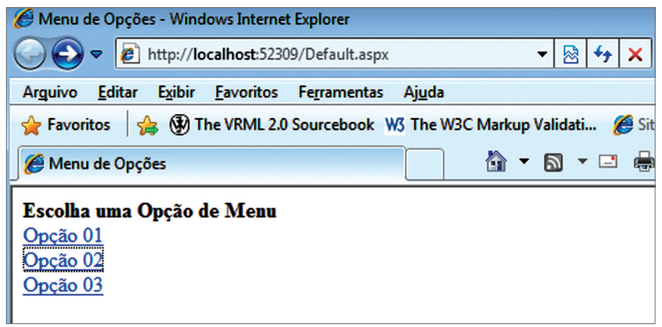


Figura 325
Menu de opções.

Uma URL é o endereço de um recurso (um arquivo, uma impressora, etc.) disponível em uma rede, – seja internet, rede corporativa ou intranet. A URL tem a seguinte estrutura: protocolo:// máquina/caminho/recurso. O protocolo poderá ser HTTP, FTP, entre outros. O campo máquina indica o servidor que disponibiliza o documento ou recurso designado. E o caminho especifica o local (geralmente em um sistema de arquivos) onde se encontra o recurso dentro do servidor.

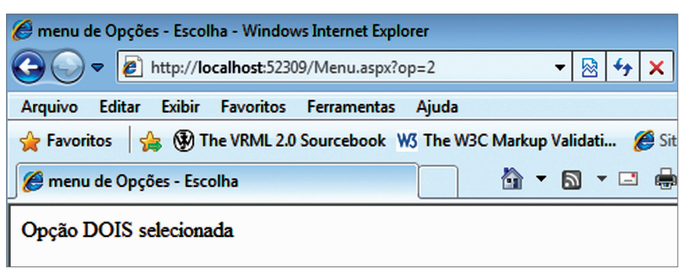
No código que estamos utilizando como exemplo, encontramos para cada link a identificação do arquivo e o valor a ser transferido via URL (figura 326).

Figura 326
Identificação de arquivo e valor a transferir.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Menu de Opções</title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <b>Escolha uma Opção de Menu </b> <br />
  <asp:LinkButton ID="lnkOpcao01" runat="server"
PostBackUrl="Menu.aspx?op=1">Opção 01</asp:LinkButton><br />
  <asp:LinkButton ID="lnkOpcao02" runat="server"
PostBackUrl="Menu.aspx?op=2">Opção 02</asp:LinkButton><br />
  <asp:LinkButton ID="lnkOpcao03" runat="server"
PostBackUrl="Menu.aspx?op=3">Opção 03</asp:LinkButton><br />
  </div>
  </form>
</body>
</html>
```

No arquivo Menu.aspx, teremos como resposta a indicação da opção escolhida pelo usuário (figura 327).

Figura 327
Opção escolhida via menu.



No código (figura 328), a função Request.QueryString() realizará a captura da variável no evento Load_Page e carregará a frase no controle Label, de acordo com a escolha do usuário.

Figura 328
Captura da variável no evento Load_Page.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitio-
nal//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>menu de Opções - Escolha</title>
</head>
<script runat="server">
  Sub Page_Load(ByVal Sender As Object, ByVal E As EventArgs)
    Dim opcao = Request.QueryString("op")
    Select Case opcao
      Case "1"
        Escolha.Text = "Opção UM selecionada"
      Case "2"
        Escolha.Text = "Opção DOIS selecionada"
      Case "3"
        Escolha.Text = "Opção TRÊS selecionada"
    End Select
  End Sub
</script>
<body>
  <form id="form1" runat="server">
  <asp:Label ID="Escolha" runat="server" Text=""></asp:Label>
  </form>
</body>
</html>
```