LAB12: Componentes ASP.NET

Neste laboratório vamos criar uma aplicação **ASP.NET** que utiliza o componente **DropDownList**, diversas opções de associação de conteúdo ao componente visual, bem como o uso dos componentes básicos de validação.

Exercício 1 – Projeto Web Forms

1. Dentro do Visual Studio selecione o menu: *File | New | Project*. Na lista *Installed Templates* (modelos), selecione *ASP.NET Web Application*. Escolha o nome do projeto e a pasta e clique OK.

		New Project	? 🗙
▶ Recent	_	.NET Framework 4.5 - Sort by: Default	🗸 🏥 🔚 Search Installed Te 👂
 Installed 		ASP NET Web Application Visual C#	Type: Visual C#
 ✓ Templates ▷ Visual Basic ✓ Visual C# Windows St Windows ✓ Web ✓ Web ✓ Visual St ▷ Office/Shart Cloud LightSwitch 	tore tudio 2012 ePoint		A project template for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET. Add Application Insights to Project Microsoft recommends adding Application Insights telemetry to help you understand and optimize application performance.
▷ Online		Click here to go online and find templates.	
<u>N</u> ame:	WebApplication1		
Location: c:\users\tdykstra\documents\visual studio 2013\Projects •		Browse	
Solution name: WebApplication1		Create directory for solution	
			Add to source control
			OK Cancel

2. Selecione o template *Empty* e marque a opção WebForms e clique Ok,conforme figura abaixo:

New ASP.NET Project - V	NebApplication6 ? ×
Select a template:	An empty project template for creating ASP.NET
Empty Web Forms MVC Web API	applications. This template does not have any content in it. Learn more
Add folders and core references for: Web Forms MVC Web API Add unit tests Test project name: WebApplication6.Tests	Authentication: No Authentication
	OK Cancel

- No Solution Explorer, clique com o botão direito sobre o nome do projeto que você criou e selecione (*Add | New Item*). Adicione um novo arquivo, do tipo *Web Form*, nomeado "WebForm1.aspx".
- 4. Selecione o menu *View | Toolbox*. Quando uma página ASP.NET está aberta na interface de design, podem ser vistos os seguintes controles (ou componentes) na Janela de **Toolbox**.



 Arraste sobre a interface de design do um formulário criado, o controle *Label* na aba *Standard* da *Toolbox*. Após arraste um controle *Button*:

WebFor	m1.aspx*	₽	х	
div				
Label	Button	ļ		

6. Selecione o menu *View | Properties Window*. Nesta janela você pode visualizar e alterar as propriedades dos controles do formulário:

Properties	- ₽	×	
Label1 System.Web.UI.WebControls.Label			
B 🛛 🖡 🔎			
(Expressions)		*	
(ID)	Label1		
AccessKey			
AssociatedControlID			
BackColor			
BorderColor			
BorderStyle	NotSet		
BorderWidth			
ClientIDMode	Inherit		
CssClass			
Enabled	True		
EnableTheming	True		
EnableViewState	True		
⊞ Font			
ForeColor			
Height			
SkinID			
TabIndex	0		
Text	Label		
ToolTip			
ValidateRequestMode	Inherit		
ViewStateMode	Inherit		
Visible	True	Ψ.	
Text			
The text to be shown for the Label.			

Figura 7 - Janela Properties

 Clique sobre o rótulo (*Label*) de seu formulário altere as seguintes configurações (dentro da janela *Properties*):

ID	IblMensagem	
Text	Primeira Aula, S2B	

• Da mesma forma, altere as configurações do botão (Button)

ID	btnExibir
Text	Exibir

- 8. Execute e teste a aplicação. Note que não há "evento" associado ao clique do botão na aplicação.
- 9. Adicione ao formulário um novo Labele altere a propriedade ID para IblData.
- 10. Selecione o componente *btnExibir*, clique em *Eventos* (botão) da *Janela de Propriedades*. Nesta nova tela você pode acionar os eventos referentes ao *btnExibir*.
- 11. Escolha o evento *Click*. Dê duplo clique com o **botão esquerdo** do mouse sobre ele (o mesmo efeito é obitido com um duplo clique sobre o componente na janela de desgn). Adicione o código abaixo ao evento "*btnExibir_Click*":

```
publicpartialclassWebForm1 : System.Web.UI.Page
{
    protectedvoid Page_Load(object sender, EventArgs e)
        {
        }
    protectedvoid btnExibir_Click(object sender, EventArgs e)
        {
            lblHora.Text = DateTime.Now.ToLongTimeString();
        }
    }
}
```

12. Execute e teste.

Exercício 2 – Trabalhando com DropDownList

- 1. Selecione o projeto e adicione um novo formulário (Add |Web Form).
- Arraste para o novo formulário um componente do tipo *Dropdown list* da *Toolbox*, na aba *Standard*, para o formulário criado.
- Clique com o botão direito no arquivo e acesse View Code. Com isso deve ser exibido o code-behind do Webform.
- 4. No método Page_Load adicione o seguinte trecho de código:

```
if (!Page.IsPostBack)
{
    string[] tiposAutomovel = newstring[4] { "Carro", "Moto",
    "Caminhão", "Caminhonete" };
DropDownList1.DataSource = tiposAutomovel;
    DropDownList1.DataBind();
}
```

5. Aperte *Ctrl+F5* e veja o resultado. O *dropdown* deve conter os itens que estavam no array. O resultado deve ser similar à figura a seguir.

褑 Alô Mundo - Windows Internet Explorer		
• http://localhost:64036/	🗕 😒 😽 🗙 📴 Bing	- م
🖕 Favorites 🛛 👍 🙋 CCO-610 Algoritmos	e Co 🙋 Home 🌄 Suggested Sites 👻 💋 Web Slice Gallery 👻	
🖉 Alô Mundo	📩 👻 🔜 👘 👻 Page 🕶 Sa	afety 🔻 Tools 🕶 🔞 👻 🦥
Carro Carro Moto Caminhão Caminhonete		
Done	💊 Local intranet Protected Mode: Off	🖓 🕶 🍕 100% 👻 💡

- 6. Com isso nós declaramos um array e inicializamos com os valores na própria declaração. Na sequência vamos fazer o mesmo porém inicializar o array após a declaração.
- 7. Troque o trecho de código anterior pelo seguinte trecho de código (mantendo o if(!isPostBack)):

```
string[] tiposAutomovel = newstring[4];
tiposAutomovel[0] = "Carro";
tiposAutomovel[1] = "Moto";
tiposAutomovel[2] = "Caminhão";
tiposAutomovel[3] = "Caminhonete";
DropDownList1.DataSource = tiposAutomovel;
DropDownList1.DataBind();
```

8. Aperte *Ctrl+F5* e veja o resultado. O *dropdown* deve conter os itens que estavam no array. O resultado deve ser similar à figura a seguir.

🖉 Alô Mundo - Windows Internet Explorer		
COO - Attp://localhost:64036/	👻 🗟 🐓 🗙 🔽 Bing	+ م
🙀 Favorites 🛛 👍 🔊 CCO-610 Algoritmos e Co	🙋 Home 🚺 Suggested Sites 🔻 🙋 Web Slice Gallery 👻	
🖉 Alô Mundo	🐴 🔻 🔝 👻 🚍 🗮 👻 Page 🕶 Safet	ty ▼ Tools ▼ 🕢 ▼ 🎽
Carro Moto Caminhão Caminhonete		
Done	🔩 Local intranet Protected Mode: Off 🛛 🛛	≩ ▼ 🔍 100% ▼

- 9. O resultado esperado é o mesmo, porém os valores do array foram inicializados de outra maneira.
- 10. Volte ao modo *Design* do *webform*.
- 11. Clique com o botão esquerdo do mouse ao lado do *Dropdown* e aperte enter para criar uma nova linha. Arraste um componente *Label* da *Toolbox* na aba *Standard* para a linha criada.
- 12. Volte para o **code-behind** do *webform* e adicione o seguinte trecho de código logo após a linha **DropDownList1.DataBind()**:

Label1.Text = "Total de itens: " + tiposAutomovel.Length.ToString();

- 13. Execute e teste.
- 14. Adicione um novo componente Label.
- 15. Coloque o texto "Seleção do usuário:" no label criado.
- 16. Adicione um novo label ao lado do anterior. Coloque o nome IblSelecao e o texto vazio.
- 17. Selecione o componente *DropDownList* e habilite o mecanismo de *postback*. Faça isso clicando sobre a seta de *atividades rápidas* e marcando *Enable AutoPostback*.
- 18. Dê um duplo clique sobre o *DropDownList* e acrescente o seguinte tratador de evento:

```
protectedvoid DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    lblSelecao.Text = DropDownList1.SelectedValue;
}
```

19. Aperte *Ctrl+F5* e veja o resultado.

Exercício 3 – Componentes básicos de validação

- 1. Adicione um novo formulário ao projeto.
- Adicione ao formulário quatro componentes do tipo TextBox (ID→ "txtNome", ID → "txtTelefone"; ID→ "txtEmail", ID→ "txtEndereco").
- 3. Adicione quatro Labels (ID→ "IbINome", Text→ "Nome"; ID→ "IbITelefone" Text → "Telefone"; ID→ "IbIEmail", Text→ "E-mail"; ID→ "IbIEndereco", Text→ "Endereço").
- 4. Adicione um botão (ID→ "btnEnviar", Text→ "Enviar").
- 5. Arranje a página para se parecer com a seguinte figura:

WebForm1.aspx 🗢 🗙
body
Nome:
Telefone
E-mail
Endereço
Enviar

Acrescentar componentes de validação

6. Adicione um *RequiredFieldValidator* da Toolbox na aba *Validation*, ao lado do *txtNome*. Mude suas propriedades da seguinte maneira:

ErrorMessage -> "Campo nome	obrigatório"
<pre>ControlToValidate ->txtNome</pre>	
ID ->rfvNome	
Display ->None	
Text ->*	

7. Adicione um *RegularExpressionValidator* da *Toolbox*, ao lado do *txtTelefone*. Mude suas propriedades da seguinte maneira:

```
ErrorMessage -> "Telefone inválido"
ControlToValidade ->txtTelefone
ValidationExpression -> Aperte nos três pontinhos e selecione P.R.C. Phone Number
ID -> revTelefone
Display -> None
Text -> *
```

8. Adicione um *RegularExpressionValidator* da *Toolbox*, ao lado do *txtEmail*. Mude suas propriedades da seguinte maneira:

```
ErrorMessage -> "E-mail inválido"
ControlToValidade ->txtEmail
ValidationExpression -> Aperte nos três pontinhos e selecione Internet E-mail Adress
ID -> revEmail
Display -> None
Text -> *
```

9. Adicione um *RequiredFieldValidator* da *Toolbox*, ao lado do *txtEndereco*. Mude suas propriedades da seguinte maneira:

ErrorMessage -> "Campo endereço obrigatório" ControlToValidate ->txtEndereco ID ->rfvEndereco Display ->None Text ->*

- 10. Adicione um ValidationSummary ao final da página.
- 11. Compile e execute o programa.

Exercício adicional:

 Adicione um Label que será preenchido com as informações "Processamento ok" ou "Erro" caso o evento de click do botão tenha processado os dados corretamente. Utilize a propriedade Page.IsValid para garantir que a página foi validade antes de tentar utilizar os dados inseridos pelo usuário.