

## Nesta seção você encontra artigos intermediários sobre Delphi Win32 e Delphi for .NET

### AJAX Control Toolkit

Turbine suas aplicações Web com ASP.NET 2.0



#### Resumo DevMan

O AJAX Control Toolkit é uma poderosa suíte de componentes para serem utilizadas em aplicações Web que utilizem ASP.NET. Inclui componentes para otimizar validações, calendários, controles de abas, janelas popups, animações e muito mais.

#### Nesse artigo veremos

- Utilização de Ajax em aplicações ASP.NET;
- Utilização da suíte de componentes Ajax Toolkit.

#### Qual a finalidade

Entender como e onde usar Ajax em aplicações Web.

#### Quais situações utilizam esses recursos?

Qualquer Web Site que necessite de velocidade de acesso e usabilidade.

**T**ornando cada vez mais indispensável em aplicações web, o uso de tecnologias AJAX torna páginas de sites e sistemas Web mais dinâmicas e mais rápidas. Um exemplo prático a ser citado é a atualização de uma página pela segunda vez, ou melhor, depois de carregada completamente a página, o usuário pede um serviço ao servidor e ao retornar a requisição, a atualização da página acontece somente no que há de diferente em relação ao que já tinha, não precisando atualizar toda ela, deixando assim bem mais rápida a navegação.

Com esse grande avanço do AJAX é que surgiu mais uma nova ferramenta para desenvolvimento de aplicações, é o *AJAX Control Toolkit*. Um conjunto de ferramentas para aplicações *ASP.NET* com suporte *AJAX* que foi desenvolvido em conjunto entre a comunidade e a Microsoft para versões do *Framework* acima do 2.0 com *ASP.NET AJAX Extensions 1*. Baseando-se em *AJAX*, uma das



#### Maikel Marcelo Scheid

(maikelscheid@gmail.com)

é técnico em Informática com ênfase em Análise e Programação de Sistemas. Atua na área de Desenvolvimento de Softwares em Delphi para plataforma Win32 e .NET com banco de dados Firebird e MS SQL. É membro da equipe Editorial ClubeDelphi.



#### Tiago José Pasioka

(tiagopasioka@gmail.com)

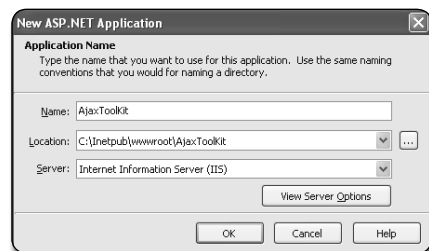
é técnico em informática, Analista e Programador de Sistemas com Delphi .NET, banco de dados Firebird e SQL Server.

*TextBox*, o *MaskEditExtender*, responsável por aplicar máscaras no *TextBox* e o *ValidatorCalloutExtender* responsável por exibir as mensagens dos componentes da *Web Validation*.

## Instalando o AJAX Control Toolkit

Para instalação do pacote de componentes *AJAX Toolkit*, faça o download no endereço [www.codeplex.com/Ajax-ControlToolkit/Release/ProjectReleases.aspx?ReleaseId=11121](http://www.codeplex.com/Ajax-ControlToolkit/Release/ProjectReleases.aspx?ReleaseId=11121), salve em alguma pasta na sua máquina e descompacte o arquivo de instalação. Localize na pasta [...] \SampleWebSite\bin\ a dll *AjaxControlToolkit.dll*, a qual é necessária para a instalação do pacote, copiando-a para o seguinte diretório *C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727*, local onde encontram-se todas as dlls da versão do Framework, pois se você excluir todo aquele pacote junto com a dll depois de o instalar, vai ocorrer o erro de que não está encontrando o caminho, que foi especificado no momento da instalação. Para que isso não aconteça, instale a partir dessa pasta.

Na instalação do componente, usaremos o *RAD 2007*, onde criaremos uma nova aplicação com suporte AJAX através do menu *File\New>Other>Delphi for .Net Projects>AJAX-Enable ASP.NET Web Application* definindo o nome do projeto como *AjaxToolkit* e no item *Location* selecione o diretório destino onde os arquivos do projeto serão criados. No item *Server* defina o servidor de aplicação para a hospedagem do sistema (**Figura 1**). Sendo uma aplicação com suporte AJAX, as páginas *Default* já serão exibidas contendo dois componentes: o *Script Manager* e o *UpdatePanel*, que são os dois componentes responsáveis pelo funcionamento e atualização do AJAX. Ao criar uma nova página, os componentes deverão ser adi-



**Figura 1.** Registrando a Aplicação

cionados manualmente, arrastando-os da paleta de componentes.

Para a instalação do pacote de componentes, no menu *Component>Installed .Net Component* abra o gerenciador de componentes .NET (**Figura 2**), onde no campo *Category* digite uma identificação para o pacote *AjaxControlToolkit* e clique

no botão *Select an Assembly* localizando a dll de instalação que armazenamos a pouco no diretório de pacotes do *Framework 2.0*. Ao selecionar a DLL, toda a lista de componentes a ser instalada será listada, bastando confirmar o gerenciador para que os mesmos estejam disponíveis na *Tool Palette* (**Figura 3**).



## Nota do DevMan

AJAX (acrônimo em língua inglesa de Asynchronous Javascript And XML) é o uso sistemático de tecnologias providas por navegadores, como Javascript e XML, para tornar páginas mais interativas com o usuário, utilizando-se de solicitações assíncronas de informações. AJAX não é somente um novo modelo, é também uma iniciativa na construção de aplicações web mais dinâmicas e criativas. AJAX não é uma tecnologia, são realmente várias tecnologias conhecidas trabalhando juntas, cada uma fazendo sua parte, oferecendo novas funcionalidades. AJAX incorpora em seu modelo.:

- \* Apresentação baseada em padrões, usando XHTML e CSS;
- \* Exposição e interação dinâmica usando o DOM;
- \* Intercâmbio e manipulação de dados usando XML e XSLT;
- \* Recuperação assíncrona de dados usando o objeto XMLHttpRequest;
- \* e Javascript unindo todas elas em conjunto.

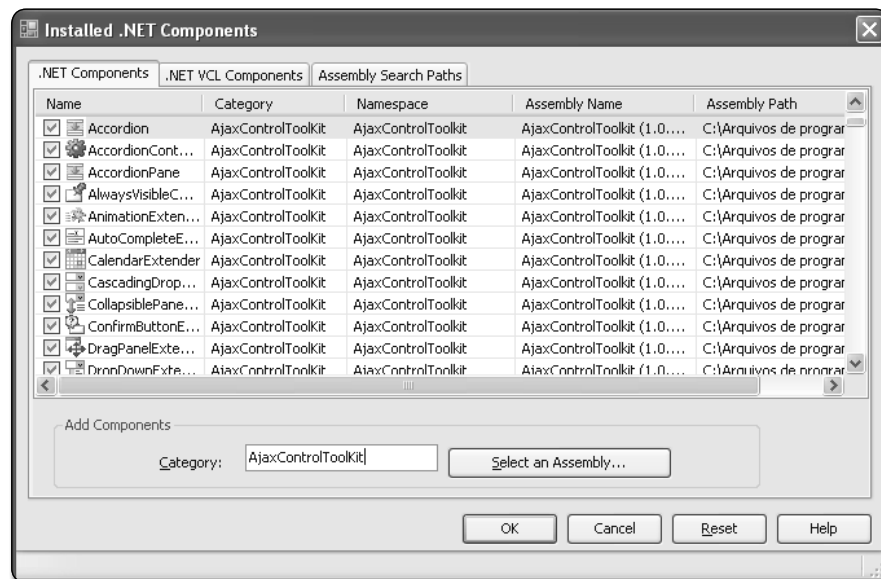
O modelo clássico de aplicação web trabalha assim: A maioria das ações do usuário na interface dispara uma solicitação HTTP para o servidor web. O servidor processa algo — recuperando dados, realizando cálculos, conversando com vários sistemas legados — e então retorna uma página HTML para o cliente. É

um modelo adaptado do uso original da Web como um agente de hipertexto, porém o que faz a Web boa para hipertexto não necessariamente faz ela boa para aplicações de software.

Esta aproximação possui muito dos sentidos técnicos, mas não faz tudo que um usuário experiente poderia fazer. Enquanto o servidor está fazendo seu trabalho, o que o usuário estará fazendo? O que é certo, esperando. E a cada etapa em uma tarefa, o usuário aguarda mais uma vez.

Obviamente, se nós estivéssemos projetando a Web a partir do zero para aplicações, não faríamos com que os usuários esperassem em vão. Uma vez que a interface está carregada, por que a interação do usuário deveria parar a cada vez que a aplicação precisasse de algo do servidor? Na realidade, por que o usuário deveria ver a aplicação ir ao servidor toda vez?

A maior vantagem das aplicações AJAX é que elas rodam no próprio navegador web. Então, para estar hábil a executar aplicações AJAX, basta possuir algum dos navegadores modernos, ou seja, lançados após 2001. São eles: Mozilla Firefox, Internet Explorer 5+, Opera, Konqueror e Safari.



**Figura 2.** Instalando pacote de componentes AJAX Toolkit

## Conhecendo e utilizando os componentes AJAX Toolkit

Antes de começar a adicionar componentes vamos aprimorar a aplicação, altere a página inicial, *Default.aspx* para *ajax.aspx*. Para isso clique com o botão direito do mouse no nome da página no *Project Manager* e selecione *Rename*. Antes de trabalharmos na página criada, iremos definir uma estrutura padrão para o cabeçalho do nosso sistema através de *User Control* na qual mais adiante faremos também o controle de usuários logados. Para adicionar o recurso é necessário criar uma página *.ascx* através do menu *File\New>Other>Delphi for .NET Projects>New ASP.NET Files>ASP.NET User Control*. Dessa forma o Delphi fará

a criação de uma nova página de edição do conteúdo. Altere no *Project Manager* o nome do *User Control* para *uctop.ascx*.

Nessa *UserControl* (*uctop.ascx*), abra o código pressionando a tecla *F12*, e no evento da *procedure Page\_Load* adicione o código a seguir:

```
Page.Title := '
Trabalhando com Ajax Control Toolkit';
```

Esse termo definirá um título padrão para todas as páginas que tiverem essa *User Control*, não precisando assim definir em cada página. Se quiseres, pode criar um rodapé, da mesma forma como criamos o cabeçalho, mas não precisando colocar o código *Page.Title* na *procedure Page\_Load*.

Para aplicar as *User Control's* na sua página *ajax.aspx*, na janela *Project Manager*, procure o arquivo de *uctop.ascx*, selecione-o e arraste para a página acima do componente *UpdatePanel*, pois só ficarão dentro dela os controles da página. Se você criou a *User Control* representando o rodapé, siga os mesmos passos para arrastá-lo à página como rodapé da mesma.

### CalendarExtender

Antes de adicionar o componente *CalendarExtender* na página, através do *Windows Explorer*, procure a pasta onde são salvos os arquivos do projeto da aplicação (campo *location* da **Figura 1**) e crie uma nova pasta com o nome de imagens, pasta onde serão colocadas todas as imagens para a aplicação. Aproveitando, adicione nessa pasta um ícone que represente um calendário, que ocuparemos depois.

Agora no *Delphi*, dentro da *UpdatePanel*, da *Tool Palette* adicione da paleta *Web-Control* os componentes *TextBox*(*txbData*) e ao lado dele um *ImageButton*(*imgbData*). No *ImageButton* adicione o ícone de calendário, através da janela *Object Inspector*, na propriedade *ImageUrl*, procure o ícone na pasta imagens que criamos e adicionamos anteriormente. Depois adicione da paleta *AjaxControlToolkit* que adicionamos no início do artigo, o componente *CalendarExtender*, em qualquer parte do *design*, mas dentro do *UpdatePanel* (**Figura 4**).

**Nota:** A maioria dos componentes do *Toolkit* não precisa, necessariamente, estar ao lado do seu controlador, pois quando adicionados, no Delphi eles ficam ocupando espaço, mas quando executados no browser, eles ficam invisíveis, não percebendo que existe o componente.

Adicionado o componente de calendário, vamos configurar algumas propriedades na janela *Object Inspector* do *CalendarExtender*. Selecionando o componente, acesse suas propriedades e altere-as conforme configurações a seguir:

*TargetControlID*: aqui você digitará o ID do *TextBox* que irá receber a data (Ex: *txbData*). Este *TextBox* ao receber o foco do cursor exibirá automaticamente uma máscara de data que definiremos como componente de calendário;

*ID*: *calendar*, nome que será atribuído ao componente;

*Format*: o formato é uma máscara de formatação para a data ao ser digitada ou selecionada no *TextBox* (Ex: *dd/MM/yyyy*);

*FirstDayOfWeek*: Selecionará qual será o primeiro dia da semana que será exibido no calendário (Ex: *Sunday*, semana irá começar sempre no dia de domingo).

*PopupButtonPosition*: Digitará o ID dado ao *ImageButton* (Ex: *imgbData*). É o componente que deverá ser acionado para que o calendário seja exibido. Ao clique do botão *imgbData* o calendário ficará visível.

*PopupPosition*: Selecione uma posição para a exibição do calendário (Ex: *Right*). É a posição em relação ao componente de foco que o calendário irá assumir.

Realizadas as principais configurações necessárias para o funcionamento do componente, antes de ver o seu resultado (**Figura 5**), verifique se no código *.aspx* da sua página, a *tag HEAD* está como *runat="server"* conforme código a seguir. Realizada a verificação do trecho de código, execute sua página e clique sobre o ícone do calendário adicionando, exibindo-o.

```
<HEAD runat="server">
<TITLE></TITLE>
</HEAD>
```

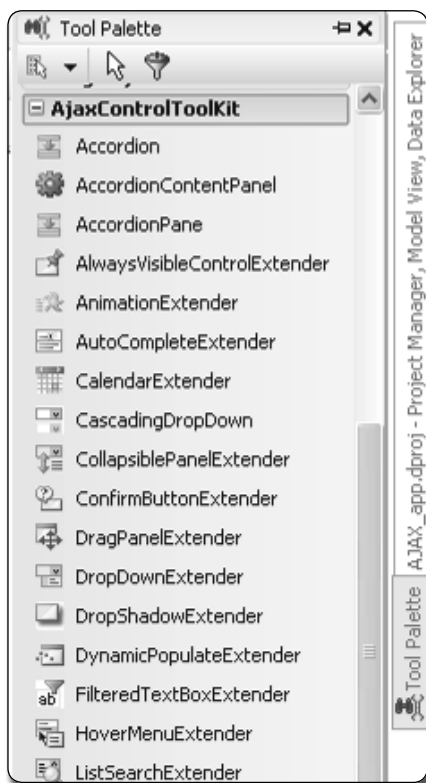


Figura 3. Paleta com componentes do pacote AJAX Toolkit

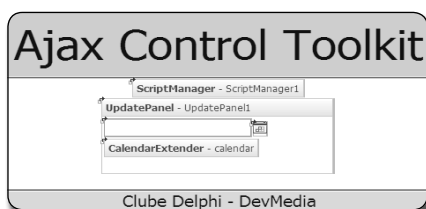


Figura 4. Adicionando o CalendarExtender à página

Uma outra opção de visualização disponível seria se você não quisesse exibir o calendário ao clique do mouse sobre o botão, mas sim quando o foco for jogado ao *TextBox* por clique ou *tab* na página, bastando deixar em branco a propriedade *PopupButtonPosition*. Perceba também ao executar a aplicação que o calendário está escrito em inglês e você gostaria de visualizá-lo em português, como solucionar isso? Simples, no *Delphi*, no *design* da sua página, selecione o componente *ScriptManager* e altere para *True* as propriedades *EnableScriptGlobalization* e *EnableScriptLocalization*. Depois abra o seu arquivo *Web.Config*, na janela *Project Manager*, procure a tag `<globalization>` e adicione dentro ao seu corpo os seguintes termos, fazendo com que o idioma de apresentação do calendário seja em *Pt-BR* (Figura 6):

```
culture="pt-BR" uiCulture="pt-BR"
```

## MaskEditExtender

Lembra a máscara de data que aplicamos em nossos sistemas *Win32* geralmente utilizando o componente

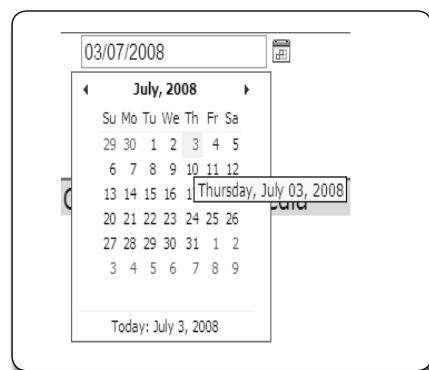


Figura 5. Calendário Padrão

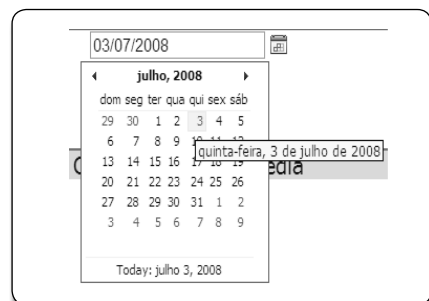


Figura 6. Calendário em Português

*MaskEdit* ou algum *DataSet* ligado a uma base de dados onde aparecem as barras pré-digítadas no componente?

Este mesmo recurso podemos utilizar agora na *web*, em nossas aplicações *for ASP.NET*. Aplicaremos através do componente *CalendaExtender* vinculado ao *TextBox*, uma máscara para quando o usuário não quiser selecionar uma data, mas sim digitá-la. Para isso, arraste da paleta *AjaxControlToolkit* o componente *MaskEditExtender* abaixo do componente *CalendaExtender* e altere as seguintes propriedades:

*TargetControlID*: Selecione o *TextBox* a ser controlado e deseja aplicar a máscara de data;

As duas próximas propriedades a serem adicionadas deverão ser incorporadas por meio de código diretamente no código *aspx* da página (Figura 7), onde deverá identificar o início das *tags* de criação do componente e adicionar as seguintes configurações (Listagem 1):

*MASK*: Formato do controle da máscara (Ex: 99/99/9999);

*MASKTYPE*= Qual é o tipo da máscara para o controle (Ex:Date).

O código adicionado fará com que, quando está executando a aplicação e posicionar com ponteiro dentro do *TextBox*, o formato correto de digitação da data será exibido, permitindo digitar somente números (Figura 8).

### Listagem 1. Propriedades do MaskEdit

```
<cc1:MaskEditExtender
  ID="MaskEditExtender1"
  RUNAT="server"
  TARGETCONTROLID="txbData"
  MASK="99/99/9999"
  MASKTYPE="Date"
></cc1:MaskEditExtender>
```

```
<asp:UpdatePanel id="UpdatePanel1" runat="server">
  <ContentTemplate>
    <asp:TextBox ID="txbData" RUNAT="server" WIDTH="167px"></asp:TextBox>
    <asp:ImageButton ID="imgbData" RUNAT="server" IMAGEURL="~/Images/Calendar.png" />
    <asp:RegularExpressionValidator ID="revData" RUNAT="server"
      VALIDATIONEXPRESSION="^(((0[1-9]|1[0-9]|2[0-9]|3[0-9])\/(0[1-9]|1[0-9]|2[0-9])\/([1-9]{1}|[1-9]{2}|[1-9]{3}))$)"
      ERRORMESSAGE="Data Inválida" />
    <cc1:CalendarExtender ID="calendar"
      RUNAT="server" CSSCLASS="calendar"
      FORMAT="dd/MM/yyyy" POPUPPOSITION="BottomRight"
      POPUPBUTTONID="imgbData" TARGETCONTROLID="txbData">
    </cc1:CalendarExtender>
    <cc1:MaskEditExtender ID="MaskEditExtender1"
      RUNAT="server" TARGETCONTROLID="txbData" MASK="99/99/9999"
      MASKTYPE="Date"></cc1:MaskEditExtender>
```

Figura 7. Localização da Tag do component MaskEdit

MASKTYPE	MASK	Descrição
Date	99/99/9999	Para data
None	999,999,999-02	Para CPF
None	99,999-999	Para CEP
Time	99:99:99	Para Hora
DateTime	99/99/9999-99:99	Para Data e Hora
Number	9999	Para Números

Tabela 1. Tipos de Máscaras

A **Tabela 1** mostra mais alguns tipos de máscaras que poderão ser configuradas no componente *MaskEditExtender*.

## ValidatorCalloutExtender

Outro recurso bastante útil para nossas aplicações é a validação das datas digitadas, verificar se o valor foi digitado corretamente, onde poderemos fazer o controle por meio do *RegularExpressionValidator* que é nativo do ASP.NET, responsável por validar a data informada e o *ValidatorCalloutExtender* do *Ajax Control Toolkit*, responsável por exibir a

mensagem do *RegularExpressionValidator* em caso de data inválida. Adicione os dois componentes abaixo do *MaskEditExtender* e configure o componente *RegularExpressionValidator* (*revData*) conforme propriedades listadas a seguir:

**ErrorMessage:** Digite uma mensagem de erro (Ex: Data Inválida);

**Display:** Selecione a opção *None*, ou seja, opção que não permitirá exibir a mensagem de erro no próprio componente;

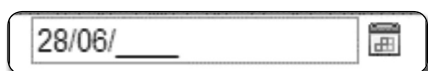
**ControlToValidate:** Selecione o *txtData*;

**ValidationExpression:** coloque a seguinte

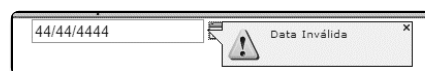
expressão para que seja feita a validação de data no formato “dd/MM/yyyy”;

```
^((((([1-9])|(1\d)|(2[0-8]))|((0[1-9])|(1[0-2]))|((31|((0[13578])|(1[02]))|((29|30)|((0[1,3-9])|(1[0-2])))))|((20[0-9])|(19[0-9][0-9]))|((29\ / 02\ / (19|20)(([02468][048])|([13579][26])))))$
```

Para o componente *ValidatorCalloutExtender*, a única propriedade a ser configurada é a *TargetControlID*, onde seleciona-se qual é o *RegularExpressionValidator* que o mesmo deverá controlar. Sempre lembrando que os componentes devem ficar dentro do *UpdatePanel*. Compile e veja o resultado (**Figura 9**).



**Figura 8.** Resultado do *MaskEditExtender*



**Figura 9.** Resultado do *ValidatorCalloutExtender* ao digitar uma data inválida

### Listagem 2. Estilo css do *CalendarExtender*

```
.calendario.ajax__calendar_container
/*formatação da parte externa onde encontra-se o corpo do calendário */
{background-color:#E6E6FA; border:solid 1px #004000;}
.calendario.ajax__calendar_header
/*formatação do cabeçalho*/
{background-color:#ffffff; margin-bottom:3px; border:solid 1px #004000; width:170px;}
.calendario.ajax__calendar_title
/*formatação do texto que encontra-se no cabeçalho*/
{color:black;}
.calendario.ajax__calendar_body
/*formatação interna de onde se encontra o calendário(onde estão os dias do respectivo
mês)*/
{background-color : white ;border : solid 1px #004000 ;}
.calendario.ajax__calendar_dayname
/*formatação da descrição dos dias da semana*/
{text-align : center ; font-weight : bold ; margin-bottom : 4px ;margin-top : 2px ;color
: red;}
.calendario.ajax__calendar_day
/*formatação dos dias do mês*/
{text-align : center ;color: black;}
.calendario.ajax__calendar_hover .ajax__calendar_day
/*formatação para quando passar o mouse sobre um determinado dia*/
{color: red;}
.calendario.ajax__calendar_hover .ajax__calendar_month
/*formatação para quando passar o mouse sobre um determinado mês*/
{color: red;}
.calendario.ajax__calendar_hover .ajax__calendar_year
/*formatação para quando passar o mouse sobre um determinado ano*/
{color: red;}
.calendario.ajax__calendar_active
/*formatação do dia, quando já possui uma data no textbox de controle*/
{background-color : #D4D0C8 ;font-weight : bold ; }
.calendario.ajax__calendar_today
/*destaque para a data atual(Today)*/
{ font-weight : bold ;color : red;}
.calendario.ajax__calendar_other .ajax__calendar_day
/*formatação dos dias que não pertencem ao mês selecionado*/
{color:#6A5ACD;}
.calendario.ajax__calendar_hover .ajax__calendar_today
/*formatação para quando passar o mouse sobre a data atual(Today)*/
{color: black}
.calendario.ajax__calendar_hover .ajax__calendar_title
/*formatação para quando passar o mouse sobre o texto do título*/
{color : red ; }
```

## Customizações com CSS

Depois de aplicados os componentes ao nosso formulário, podemos ainda observar que seus estilos estão em padrões pouco variados dos que costumamos ver em aplicações atuais, tais como cores e formas de apresentação. Da mesma forma como é possível aplicar estilos em componentes *web*, poderemos utilizar do recurso para os componentes da paleta *Ajax Control Toolkit*. Criaremos estilos variados que serão aplicados aos componentes *CalendarExtender* e o *ValidatorCalloutExtender* alterando sua forma de apresentação *default*.

Para início da construção dos estilos, crie uma nova pasta chamada “estilos”, dentro da pasta do projeto da aplicação no mesmo nível de diretório onde criamos a pasta imagens anteriormente. Em seguida faremos uso do bloco de notas para criação de um novo arquivo, ao qual deverá ser adicionado o código da **Listagem 2** que encontra-se comentado em alguns trechos de acordo com a funcionalidade das linhas de código relacionadas.

Todas estas linhas de código serão responsáveis pela aparência dos nossos componentes *web*. Salve o arquivo dentro dessa pasta estilos recém criada com o nome de *estilos.css*.

Observe que os estilos criados anteriormente independem da linguagem de ASP.NET. Estamos referenciando a cada uma das funcionalidades do calendário a confecção de um estilo, aplicando a ele formatações de fonte, tamanho e cor.

Depois de digitado e já ter salvo o estilo, retorne ao *Delphi* e no código *.aspx* da sua página, dentro da *tag HEAD*, adicione o código a seguir imediatamente abaixo da *tag TITLE*. Este código a ser adicionado será o responsável por referenciar os estilos recém criados, deixando os mesmos disponíveis para serem utilizados nos componentes da página.

```
<LINK rel="stylesheet"
href="estilos/estilos.css" type="text/css" />
```

Adicionado a referência ao arquivo de estilos no código *aspx* da página, retorne ao *design* pressionando a tecla F12 e selecione o componente *Calendar Extender*, adicionando em sua propriedade *CssClass* no *Object Inspector* o termo "calendario", nome dado ao estilo no arquivo *estilos.css*. Execute novamente sua aplicação comparando o novo resultado obtido (**Figura 10**) ao primeiro resultado visto na execução anterior, observando que os estilos alteraram a apresentação do componente.

Para aplicarmos também um estilo ao componente *ValidatorCalloutExtender* Criaremos um novo bloco de códigos, o qual terá a finalidade de alterar o design da *popup* que exibe a mensagem dos componentes da *Web Validation*. Retomando o arquivo *css estilos.css* anteriormente criado, adicione ao final do estilo para o calendário o código da **Listagem 3**, encontrando-se também comentado de acordo com a funcionalidade de cada seção de códigos empregados.

Realizada a digitação do código, que se utiliza também de efeitos comuns já conhecidos nos padrões *HTML*, salve as alterações do arquivo e retorne ao formulário da aplicação, onde deverá observar se a propriedade *TargetControlID* do componente *ValidatorCallout* está relacionada ao componente de validação *RegularExpressionValidator*. O mesmo deverá ser selecionado e na janela *Object Inspector* deverá localizar suas propriedades *Extenders* (**Figura 11**). Lembrando que as propriedades apenas estarão visíveis após ambos os componentes estarem interligados, onde as seguintes propriedades deverão ser alteradas conforme itens a seguir:



**Figura 10.** Resultado do Calendário com Estilo



## Nota do DevMan

Cascading Style Sheets, ou simplesmente CSS, é uma linguagem de estilo utilizada para definir a apresentação de documentos escritos em uma linguagem de marcação, como HTML ou XML. Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento.

Ao invés de colocar a formatação dentro do documento, o desenvolvedor cria um link (ligação) para uma página que contém os estilos, procedendo de forma idêntica para todas as páginas de um portal. Quando quiser alterar a aparência do portal basta portanto modificar apenas um arquivo.

### Listagem 3. Estilo css do ValidatorCalloutExtender

```
.validar div, .validar td
/*Formatação do visual da popup(EX: borda e cor de fundo)*/
{border:solid 1px red; background-color:white;}
.validar .ajax__validatorcallout_callout_cell
/*Manter padrão*/
{width:80px; height:100%; text-align:right; vertical-align:top; border:none; background-
color:transparent; padding:0px;}
.validar .ajax__validatorcallout_callout_table
/* Manter padrão */
{height:100%; border:none;background-color:transparent;padding:0px;}
.validar .ajax__validatorcallout_callout_arrow_cell
/* Manter padrão */
{padding:8px 0px 0px 0px; text-align:right; vertical-align:top; font-size:1px;
border:none; background-color:transparent;}
.validar .ajax__validatorcallout_callout_arrow_cell .ajax__validatorcallout_innerdiv
/* Manter padrão */
{font-size:1px; position:relative; left:1px; border-bottom:none; border-right:none;
border-left:none; width:15px; background-color:transparent; padding:0px;}
.validar .ajax__validatorcallout_callout_arrow_cell .ajax__validatorcallout_innerdiv div
/* Manter padrão */
{height:1px; overflow:hidden; border-top:none; border-bottom:none; border-right:none;
padding:0px; margin-left:auto;}
.validar .ajax__validatorcallout_error_message_cell
/*Formatação do texto dentro da popup */
{font-family:Verdana; font-size:10px; border-right:none; border-left:none; width:100%;
color:red; font-weight:bold; text-align:left;}
.validar .ajax__validatorcallout_icon_cell
/* Manter padrão */
{width:20px; padding:5px; border-right:none;}
.validar .ajax__validatorcallout_close_button_cell
/* Manter padrão */
{vertical-align:top; padding:0px; text-align:right; border-left:none;}
.validar .ajax__validatorcallout_close_button_cell .ajax__validatorcallout_innerdiv
/* Manter padrão */
{border:none; text-align:center; width:16px; height:16px; padding:2px; cursor:pointer;}
```



**CloseImageUrl:** coloque o caminho de onde o componente buscará a imagem para fechar a popup (se não colocar nada, ficará a imagem padrão);

**CssClass:** digitar o nome do estilo ("validar") criado para sua formatação;

**WarningIconImageUrl:** caminho da imagem que é exibida na frente do texto da popup;

**Width:** Tamanho da popup (em caso da mensagem ser muito grande).

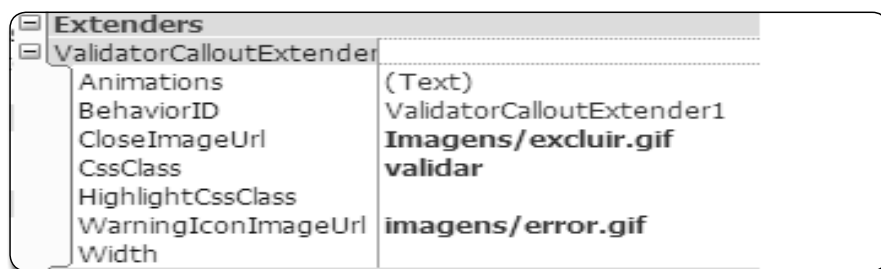
O resultado da aplicação de estilos ao componente de validação poderá

ser observado na **Figura 12**, onde um formato incorreto de data está sendo ilustrado. Estas variações de estilo você poderá aplicar a seus formulários de acordo com a necessidade e usabilidade dos formulários.

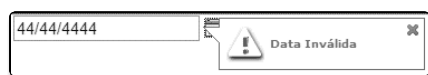
Problemas ocorrem e muitas vezes são inevitáveis, mas sempre há formas de facilitar sua resolução. Nos exemplos vistos durante a criação do artigo, há possibilidades de que possíveis problemas de *javascript* em virtude da compatibilidade dos componentes de

AJAX possam estar ocorrendo. No link a seguir está disponível para download um componente para mapeamento e validação dos códigos, uma ferramenta que será bastante útil para estes casos ([blogs.msdn.com/mattgi/archive/2007/01/23/asp-net-ajax-validators.aspx](http://blogs.msdn.com/mattgi/archive/2007/01/23/asp-net-ajax-validators.aspx)).

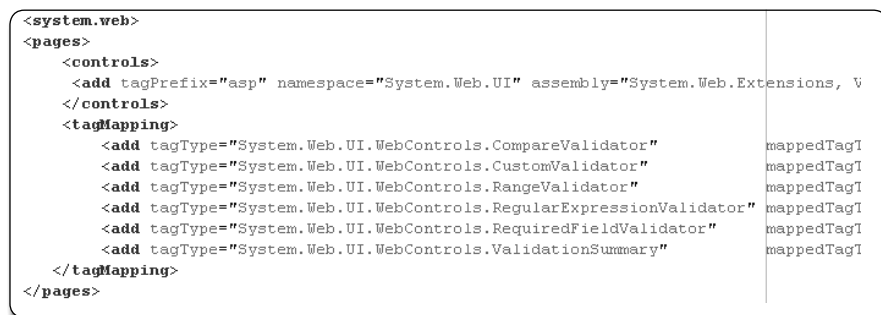
Realizado o download dos componentes (*Validators.zip*), descompacte-os e localize na pasta uma *dll* chamada *Validattors.dll*. Copie e cole dentro da pasta *Bin* do seu projeto e adicione nas referências da aplicação, onde no *Project Manager* com um clique do botão direito do mouse sobre a *dll* principal do projeto da aplicação, deverá selecionar o item *Add Reference*, buscando na caixa de diálogo da pasta *Bin* do projeto o arquivo *Validattors.dll* recém adicionado. Para que as configurações e validações entrem em prática de funcionamento, uma alteração no arquivo *Web.Config* deverá ser realizada, adicionando o código da **Listagem 4** dentro da tag *<pages>* localizada entre as demais configurações da declaração *<system.web>* (**Figura 13**).



**Figura 11.** Propriedades Extenders do RegularExpressionValidator



**Figura 12.** Resultado do ValidatorCalloutExtender com estilo



**Figura 13.** Localização na Web.Config

#### Listagem 4. TagMapping

```
<tagMapping>
<add tagType= "System.Web.UI.WebControls.CompareValidator" mappedTagType= "Sample.Web.
  UI.Compatibility.CompareValidator, Validators, Version=1.0.0.0"/>
<add tagType= "System.Web.UI.WebControls.CustomValidator" mappedTagType= "Sample.Web.
  UI.Compatibility.CustomValidator, Validators, Version=1.0.0.0"/>
<add tagType= "System.Web.UI.WebControls.RangeValidator" mappedTagType= "Sample.Web.
  UI.Compatibility.RangeValidator, Validators, Version=1.0.0.0"/>
<add tagType= "System.Web.UI.WebControls.RegularExpressionValidator" mappedTagType=
  "Sample.Web.UI.Compatibility.RegularExpressionValidator, Validators, Version=1.0.0.0"/>
<add tagType= "System.Web.UI.WebControls.RequiredFieldValidator" mappedTagType= "Sample.
  Web.UI.Compatibility.RequiredFieldValidator, Validators, Version=1.0.0.0"/>
<add tagType= "System.Web.UI.WebControls.ValidationSummary" mappedTagType= "Sample.Web.
  UI.Compatibility.ValidationSummary, Validators, Version=1.0.0.0"/>
</tagMapping>
```

## Conclusão

Vimos no decorrer deste artigo a aplicação de alguns dos componentes disponíveis na tecnologia AJAX. A usabilidade destes componentes na sua aplicação torna-a mais leve e dinâmica, facilitando a distribuição dos processos e navegação do usuário. Inúmeros outros recursos poderão ser aplicados com a utilização desta tecnologia, visite o *website* ([www.asp.net/ajax/ajaxcontrol-toolkit/samples/](http://www.asp.net/ajax/ajaxcontrol-toolkit/samples/)) de apresentação das demais ferramentas e faça a implementação em outras aplicações, criando controles inovadores para suas aplicações. Abraço e até o próximo artigo. ●

#### Dê seu feedback sobre esta edição!

A ClubeDelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

[www.devmedia.com.br/clubedelphi/feedback](http://www.devmedia.com.br/clubedelphi/feedback)



# Já pensou em hospedar todos os seus sites em um único plano de hospedagem?

## PLANO PROFISSIONAL 1

3 domínios independentes  
50 GB de transferência  
30 GB de e-mails com SSL  
1 GB de espaço  
Painel de controle  
ASP, ASP.NET 3.5 E PHP 5  
Access ilimitado  
3 bancos MYSQL 5  
1 banco SQL Server 2005 Express

**30 DIAS GRÁTIS**

Ao assinar, digite o código de desconto exclusivo para leitores desta revista e ganhe!

**RVSTMEDIA**

Tudo isso  
por apenas R\$

**25,90** por mês

**Na Hospedix você ainda ganha registro de domínio (.com .net .org ou .info) grátis e isento de taxas anuais enquanto for cliente.**

Acesse agora mesmo e descubra por que Hospedix é a hospedagem de sites preferida entre os profissionais.

**[hospedix.com.br](http://hospedix.com.br)**



**HOSPEDIX**  
HOSPEDAGEM PROFISSIONAL