Sborník vědeckých prací Vysoké školy báňské - Technické univerzity Ostrava
číslo 2, rok 2005, ročník LI, řada strojní
článek č. 1462

Marek BABIUCH*

NEW FEATURES OF ASP.NET 2.0 TECHNOLOGY

NOVÉ VLASTNOSTI TECHNOLOGIE ASP.NET 2.0

Abstract

This contribution describes new version of ASP.NET 2.0, which is expected this year together with new version of Visual Studio.NET 2005. Already Visual studio's face gives away emphatic changes. ASP.NET 2.0 Technology offers to us wide range of new possibilities, which will be described at huge number of books and guides, let's name shortly several new features: New properties of page, cross-page postbacks implementation, master page and content page strategy, wizard control, web parts and web zone for robust applications and several new login server control.

Abstrakt

Příspěvek se zabývá novou verzí technologie ASP.NET 2.0, která je očekávaná tento rok společně s novou verzí Visual Studia .NET 2005. Už první pohled na vzhled Visual Studia nám prozradí základní změny. Technologie ASP.NET 2.0 obsahuje širokou škálu nových možností, které budou opět náplní velkého počtu knih a průvodců, a tak článek popisuje zkráceně některé nové vlastnosti jako: nové vlastnosti objektu *Page*, cross-page postbacks implementaci, nové rozvržení strategie na *master page and content page*, tvorbu *wizard*ů, web part zón pro robustnější aplikace a také celou řadu nových *login* servrových ovládacích prvků.

1 NEW FACE OF MICROSOFT VISUAL STUDIO 2005

Visual Studio.NET 2005 with ASP 2.0 technology offers range of important changes. Visual Studio.NET 2005 contain built–in web server. This eliminates the need to re-compile your projects when you create or modify an application. Also, you do not need to open an entire project to make a modification; you can modify individual files. Already face itself gives away emphatic changes. IntelliSense works everywhere, including within data-binding expressions and page directives, even when you open a single file in Visual Studio 2005. You can open page, which you want without set her as start-up page. The page lifestyle is also changed and at first appearance we can see new toolbox with wide range of new advanced tools and controls ordered into categories.

2 NEW PROPERTIES OF PAGE

First novelty is paragraph about properties of *Page*. New attributes to the *@Page* directive as: *MasterPageFile* attribute for specification of *Master File path*, *EnablePersonalization* attribute for profile information, *Theme*, *Asyns* and *CompileWith* attribute. Page class also has several new methods; most of them are inherited from base *Control* class. Methods focus on new types of functionality, including control state, validation groups, and an enhanced script object model. At the following example we can see how to dynamically set properties of the head tag in a Web form. The *Header* property is a new *Page* property that inherits the *HtmlHead* class.

* Ing., Ph.D., Department of Control Systems and Instrumentation, Faculty of Mechanical Engineering, VŠB-TU Ostrava, 17.listopadu 15/2172, Ostrava-Poruba, tel. (+420) 59 732 4119, marek.babiuch@vsb.cz

```
<html xmlns=http://www.w3.org/1999/xhtml >
<head runat="server">
   <title>Untitled Page</title>
</head>
<script runat=server>
   Sub btnSetTitle(ByVal sender As Object, ByVal e As EventArgs)
      Header.Title = Msg.Text
   End Sub
</script>
```

The ability to post information from one Web form to another is a new feature of ASP.NET 2.0. It allows you to separate business logic and information processing from your user interface and design. By implementing a logical cross-page, post-back structure for your site, you cut down on redundant code and make controlling and updating code a simpler task. Following example demonstrates Cross-Page posting action.

```
<script runat=server>
   Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs)
      If PreviousPage Is Nothing Then
         Response.Write("need cross-page posting invoke.")
         Response.End()
         Return
      End If
      Dim Name As TextBox = CType(PreviousPage.FindControl("tbName"), TextBox)
      tb.Text = Name.Text
   End Sub
</script>
```

3 MASTER PAGE AND CONTENT PAGE STRATEGY

The master page provides a single point of reference for all pages to display standardized Web content. Master pages are completely transparent to end users, and they allow developers to create Web sites where pages share a common layout. A master page is very similar to a normal Web form except that instead of the *@Page* directive, master pages use the *@Master* directive. Master pages contain HTML markup, user controls, and at least one *ContentPlaceHolder* control.

```
<%@ Master Language="VB" %>
<html>
<head runat="server"><title>MasterPage.master</title>
</head>
<body bgcolor="gainsboro">
   <form id="form1" runat="server">
         <asp:contentplaceholder id="masterHead" runat="server">
         <table…>…</table>
      </asp:contentplaceholder>
      <asp:ContentPlaceHolder ID="masterBody" Runat="Server" />
   </form>
</body>
</html>
```

A content page defines the content for each region of the master page. Content pages do not include the standard HTML of a Web form. The key feature of a content page is the *Content* control, which inherits the *Control class*. You define which master page to link to within the *@Page* directive.

```
<%@ Page Language="VB" MasterPageFile="MasterPage.master" %>
<script …>… </script>
<asp:Content ID="Content" ContentPlaceHolderID="masterBody" Runat="server">
<h2> body of the content page</h2>
<asp:Button ID="Button1" Runat="server" Text="Ok" OnClick="Button1_Click"/><br />
<asp:Label ID="lblMsg" Runat="server" Text=""> </asp:Label>
</asp:Content>
```

Advantages of master pages include: allowing you to centralize the common functionality of your pages so that you can make updates in just one place, making it easy to create one set of controls and code and apply the results to a set of pages, giving you fine-grained control over the layout of the final page by allowing you to control how the placeholder controls are rendered and providing an object model that allows you to customize the master page from individual content pages.

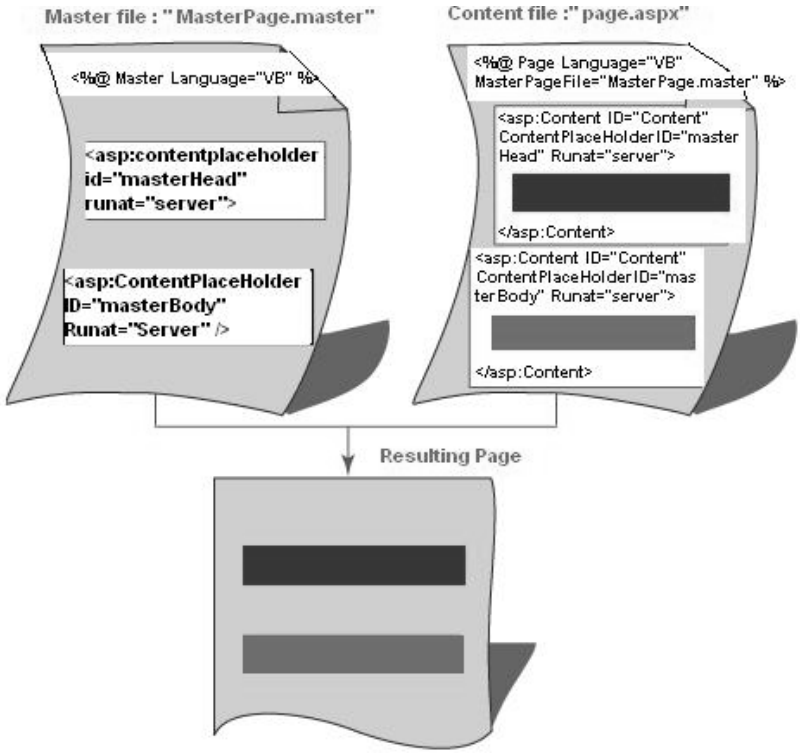

Fig. 1 Master and Content Page example

4 WIZARD CONTROL

This is another new ASP.NET 2.0 control for creating comfortable client forms. Wizard control enables sequence of related steps, which are associated with an input form and a user interface. Wizard form enables user to skip steps or modify entered values on previous steps and supports themes, styles, and templates. Wizard control automatically generates constituent controls. Wizard form usually contains navigation bar with buttons for previous and another steps, button for finish and close all operations, sidebar with all ordered steps and of course view of the content of active step.

```
<asp:Wizard Runat="Server" ID="MyWizard"  OnNextButtonClick="OnNext"
   OnFinishButtonClick="OnFinish">
   <SideBarStyle />     <HeaderStyle />     <NavigationButtonStyle />     <StepStyle />
  <SideBarTemplate>
          ……
</SideBarTemplate>
<WizardSteps>
<asp:wizardstep ID="Wizardstep1" runat="server" steptype="auto" title="…">
   <asp:textbox runat=server id="FirstName" />
    <asp:requiredfieldvalidator  />
…
  <asp:validationsummary runat="server" displaymode="List" id="Summary" />
</asp:wizardstep>
<asp:wizardstep runat="server" steptype="auto" title="">
…
</asp:wizardstep>
<asp:wizardstep runat="server" steptype="auto" title="Optional Information">
</asp:wizardstep>
<asp:wizardstep runat="server" steptype="auto" title="Finalizing...">
</asp:wizardstep>
</WizardSteps>
  </asp:Wizard>
```

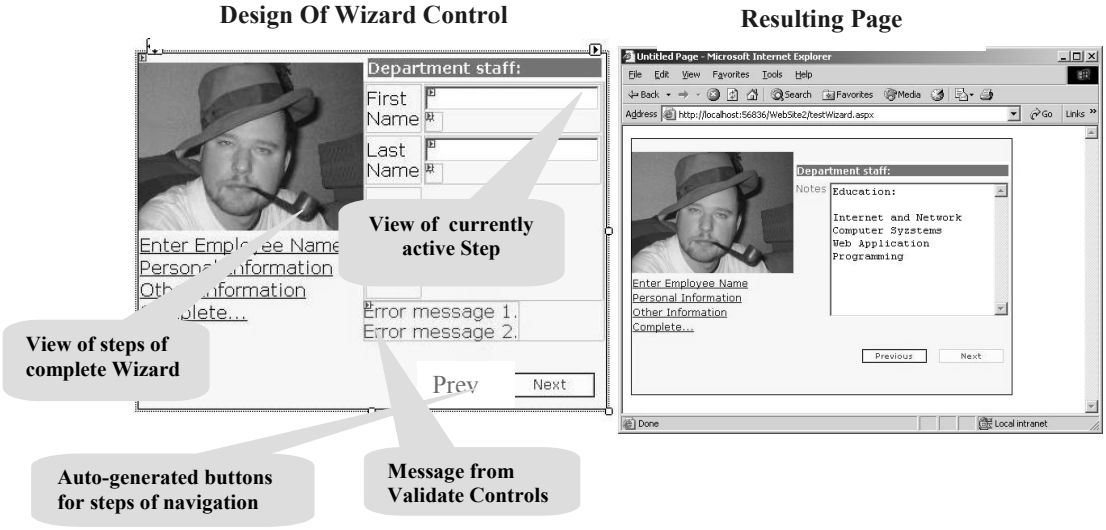

Fig. 2 Wizard Control example

5 WEB PARTS ZONE

In ASP.NET 2.0, Web Parts provide an infrastructure for creating robust web applications. A Web Part is a modular unit of information that has a single purpose and that forms the basic building block of a Web Part page. A Web Part page is a special type of Web page that consolidates data, such as lists and charts, and Web content, such as text and images, into a dynamic information portal built around a common task or special interest. Web Part page as nothing more than a special page designed to contain and support Web Parts. Web Parts are compartments of content that have the ability to be customized and personalized by the user. The users also can select, from a list of available Web Parts, those they want to be displayed on the page and how they want them displayed. They can also minimize, close, restore, or remove Web Parts from the page. Every Web Part page

must contain one and only one *WebPartManager* control. This control must appear before any Web Part zones in a page and is responsible for keeping track of the state of all the Web Parts in a page.

Steps for creating a page with web parts:

- Creating a Web page
- Adding zones to the page
- Creating content for the zones

```
<asp:webpartmanager id="WebPartManager1" runat="server" />
<asp:webpartzone id="SideBarZone" runat="server" headertext="Sidebar" >
    <zonetemplate>
   …….
</zonetemplate>
      <PartChromeStyle /> <MenuLabelHoverStyle /> <EmptyZoneTextStyle />
      <MenuLabelStyle /> <MenuVerbHoverStyle /> <HeaderStyle />
      <MenuVerbStyle /> <PartStyle />
      <TitleBarVerbStyle /> <MenuPopupStyle /> <PartTitleStyle />
</asp:webpartzone>
   </td>
  <asp:webpartzone id="MainZone" runat="server" headertext="Main">
   <zonetemplate>
        …..
    <asp:Image title="picture" runat=server id="obr" ImageUrl="C:\Dokumenty\Visual Studio
2005\WebSites\WebSite4\bab75.jpg" />
 </zonetemplate>
 </asp:webpartzone>
```

Fig. 3 Web Part example

6 NEW LOGIN CONTROLS

ASP.NET 2.0 brings several new login controls as you can se on Figure 4. Here we can se *Login Control*, *CreateUserWizard Control, ChangePassword* and *PaswordRecovery Control*. *Login Control* provides login page functionality and ability to validate user against default membership provider *PasswordRecovery Control* represents form that enables a user to recover or reset a lost password and receive it back through an e-mail message. Membership class contains static methods that you use to obtain unique identity for each connected user.

```
<asp:Login ID="Login1" runat="server">
<TitleTextStyle /> <TextBoxStyle /><LoginButtonStyle /><InstructionTextStyle />
 </asp:Login>
<asp:ChangePassword ID="ChangePassword1"  runat="server">
  <CancelButtonStyle  />  <ChangePasswordButtonStyle  /> <ContinueButtonStyle  />
<TitleTextStyle  /> <PasswordHintStyle  /> <TextBoxStyle  /> <InstructionTextStyle  />
   </asp:ChangePassword>

<asp:PasswordRecovery ID="PasswordRecovery1" runat="server">
        <InstructionTextStyle /> <SuccessTextStyle  />  <TextBoxStyle />
        <TitleTextStyle />  <SubmitButtonStyle  />
      </asp:PasswordRecovery>
<asp:CreateUserWizard ID="CreateUserWizard1" runat="server">
        <WizardSteps>
          <asp:CreateUserWizardStep runat="server" Title="Sign Up for Your New Account">
          </asp:CreateUserWizardStep>
          <asp:CompleteWizardStep runat="server" Title="Complete">
          </asp:CompleteWizardStep>
        </WizardSteps>
        <SideBarStyle /><SideBarButtonStyle />  <NavigationButtonStyle /><HeaderStyle />
<CreateUserButtonStyle /> <ContinueButtonStyle /> <StepStyle /> <TitleTextStyle />
      </asp:CreateUserWizard>
```

Fig. 4 New ASP 2.0 Login Controls

REFERENCES

[1] Microsoft Learning 2005, *http://www.microsoft.com/learning/*

[2] FOJTÍK, D. Innovations of Microsoft Visual Basic .Net 2005 Language Ostrava. In *Proceedings of XXX. Seminary ASR '05 "Instruments and Control"*. Ostrava : Katedra ATŘ, VŠB-TU Ostrava, 29. 4. 2005, s. 155-162. ISBN 80-248-0774-2.

Reviewer: doc. Ing. Radim Farana, CSc., VŠB-Technical University of Ostrava