UNIVERZITA PALACKÉHO V OLOMOUCI
KATEDRA MATEMATICKÉ INFORMATIKY

ZÁPOČTOVÁ PRÁCE
Softwarová laboratoř 6

Web Parts

Pavel Kubát
Informatika V. ročník

Říjen 2005

Abstract

The objective of this work is describing the Web parts – simple and great part of ASP.NET, which provides customization and personalization of the web sites. In the next few chapters you'll find a description of this technology, how to use it and the example of creating one site with web parts.

1 Úvod

Obsahem a cílem této práce je seznámit čtenáře s velmi zajímavou výhodou ASP.NETu – web party (webovými částmi), vysvětlit proč se webovým vývojářům nabízí podobná technologie až dnes, jaké má výhody, jak přesně funguje a na jednoduchém příkladu ukázat, jak jednoduše ji lze použít..

2 Vývoj

2.1. Uživatelsky modifikovaný vzhled

Myšlenka uživatelsky modifikovatelných webů není nikterak nová. Stejnou myšlenkou se zabývali vývojáři už mnoho let zpět, ale bohužel s nevelkým úspěchem – použití bylo neohrabané, rychlost většinou pomalá, funkčnost nedostačující a vývojářská práce o mnoho složitější. Proto byla tato myšlenka odložena a pouze špičkové firmy a vývojáři se pouštěli do webových stránek s možností uživatelsky si nastavit vzhled nebo pozměnit styl. Kódy byly ručně psané a v drtivé většině šité přímo na míru danému webu. Weby jsou v dnešní době jednostylové a uživatelsky těžko (nebo vůbec) modifikovatelné.

2.2. ASP.NET

Nerad bych v této práci popisoval jak přesně funguje ASP.NET jako celek, nebo popisoval jeho historii. To je otázka na úplně jiné téma a rozebírat to tedy nebudu. Rád bych zde jen uvedl jak došlo k začlenění Web partů do cílů vývojářů ASP.NETu.
ASP.NET ve svých prvních verzích (1.0) způsobil mezi vývojáři naprostou euforii. Tato technologie se nedala (a stále nedá) porovnávat s žádnou jinou v oblasti skriptování webových stránek. Nejenže naprosto jednoznačně předčí všechny konkurenční technologie svou rychlostí, ale také pojetí a tvorba stránek je najednou o mnoho jednoduší a přesto profesionálnější. Osobně jsem byl (jakožto vývojář v PHP) skutečně ohromen jak vývojáři Microsoftu bravurně zvládli naprosto změnit pohled na webové skriptovací jazyky.
Příchod ASP.NET 1.1 nebyl příliš zajímavý – bylo opraveno několik chyb a přidáno několik novinek. Po této verzi následovala několikaletá odmlka a v této chvíli se společně s .NET Frameworkem 2.0 objevuje i nová verze ASP.NETu. Vývojáři se nechali inspirovat svými kolegy venku a uživateli samotnými a do cílů druhé verze byla zařazena i vzhledová modifikovatelnost webů o kterou žádali. Cíl byl bravurně splněn a to hned několika způsoby. Byla přidána možnost tzv. Themes (témat) a skins (vzhled) což jsou dvě související technologie umožňující vývojářům i uživatelům rychle a efektivně vytvořit nebo změnit stávající vzhled stránek. Druhou novinkou pak byly právě web party.

3 Co je web part?

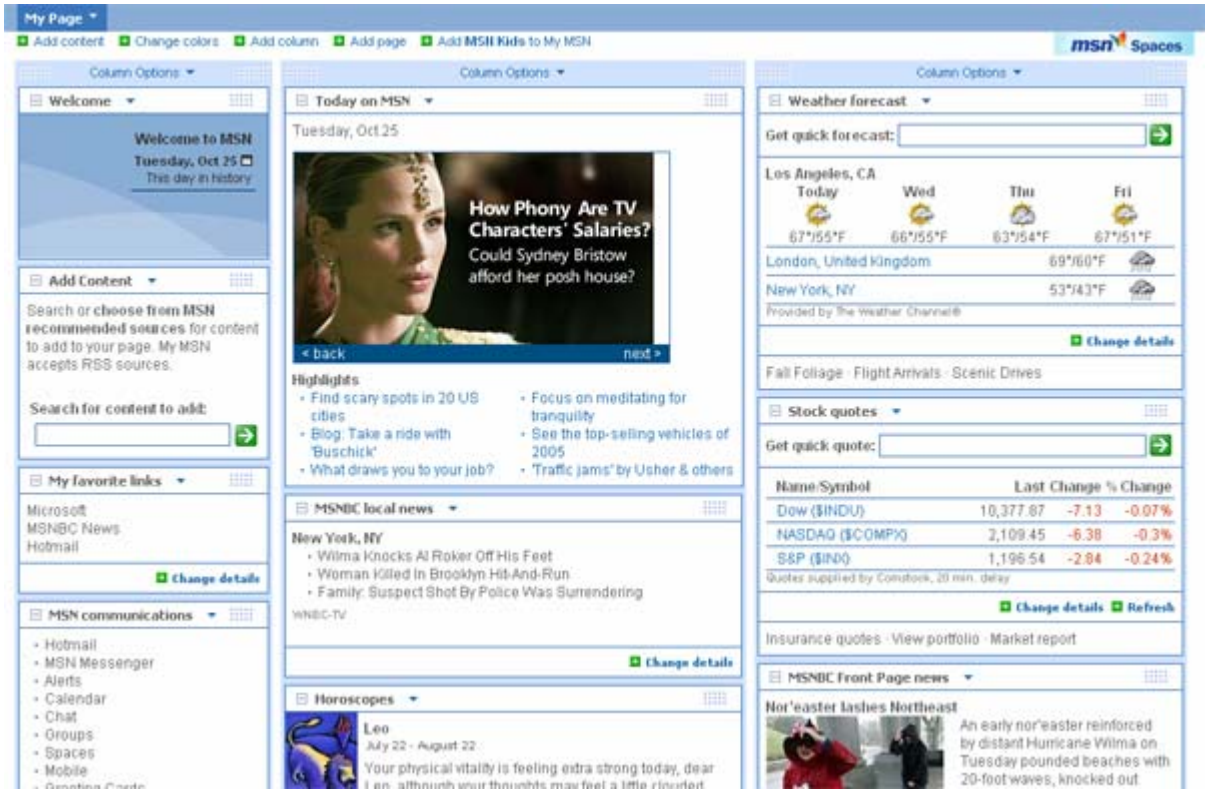
Pro web party platí několik základních pravidel. Web party

- Jsou vizuální moduly
- Mohou být pozicovány a jejich viditelnost (aktivita) vypínána nebo zapínána

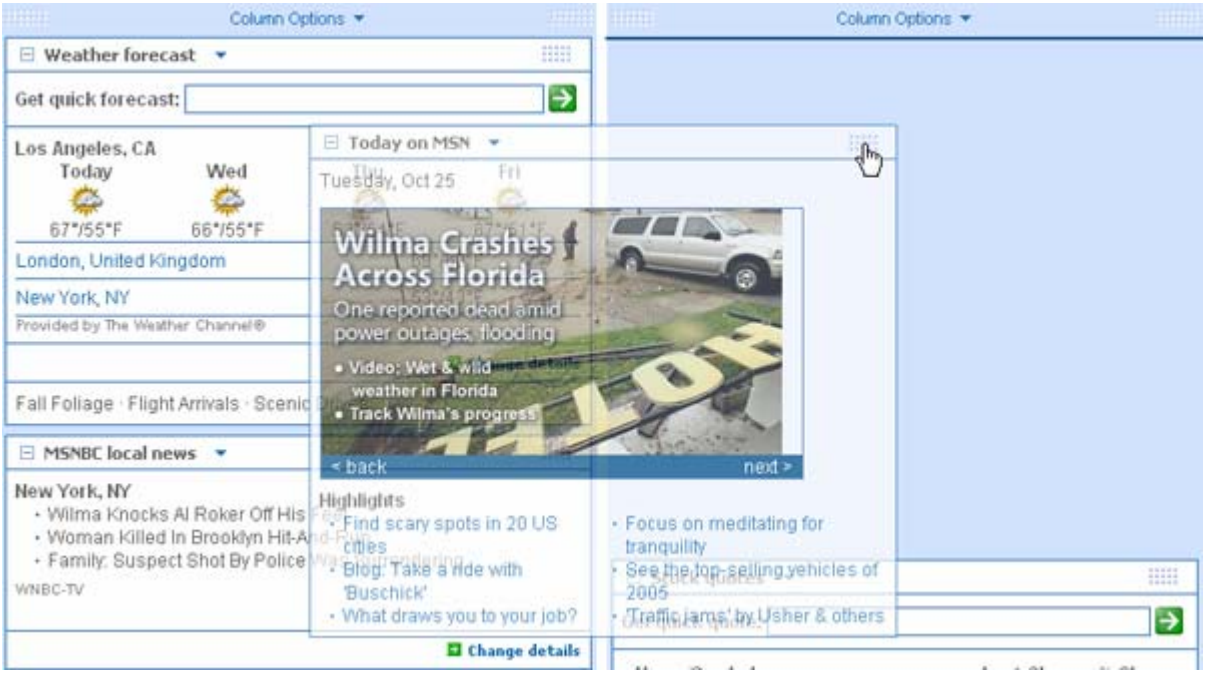
- Mohou být vytvářeny třetími stranami - skládání web partů
- Změny jsou uloženy a znovu použity pouze pro daného uživatele nebo pro skupinu uživatelů

Abych vše vysvětlil trošku podrobněji – web part je vlastně jakýsi modul (nebo část) stránek, obsahující informace o určitém tématu. Jednotlivé web party je možné nejen přesouvat mezi sebou, ale také měnit jejich velikost a aktivitu. Je velice snadné (pokud byla tato možnost vývojářem povolena) přidávat web party třetích stran, nebo přidávat web party s načtením Vámi zvoleného obsahu (např. RSS feed).

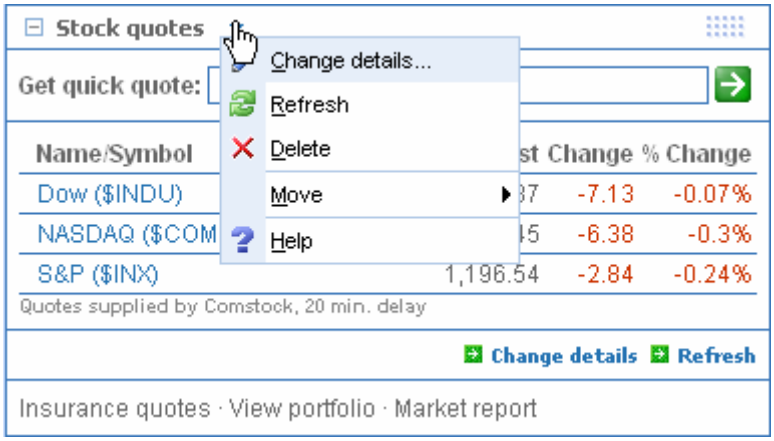
Pro ilustraci přikládám screenshoty z velmi povedené (a Microsoftem označované jako ukázkové) web part stránky My MSN (http://my.msn.com/).



Web part stránka My MSN

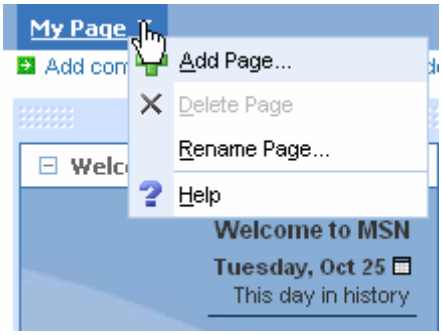


Jednoduché přesunování web partů

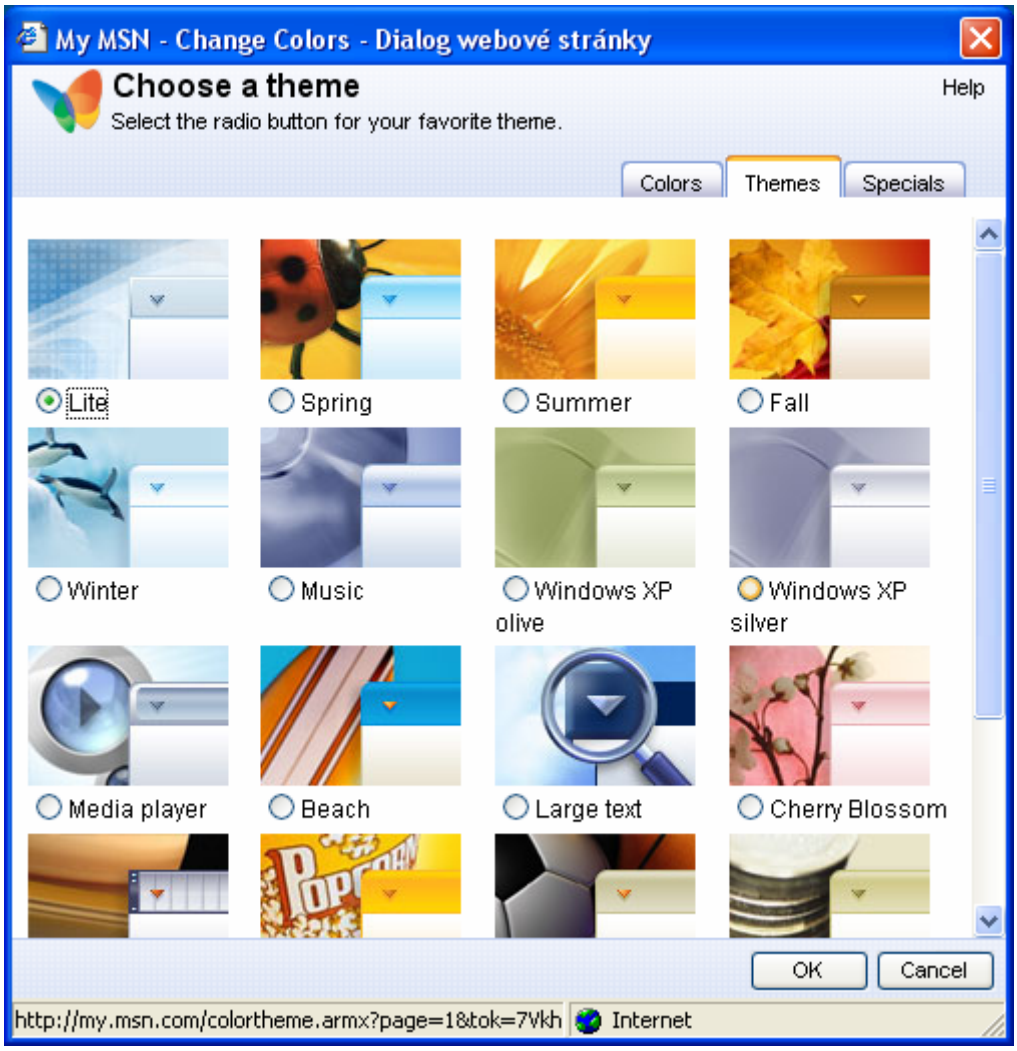


Základní možnosti Web partu na My MSN

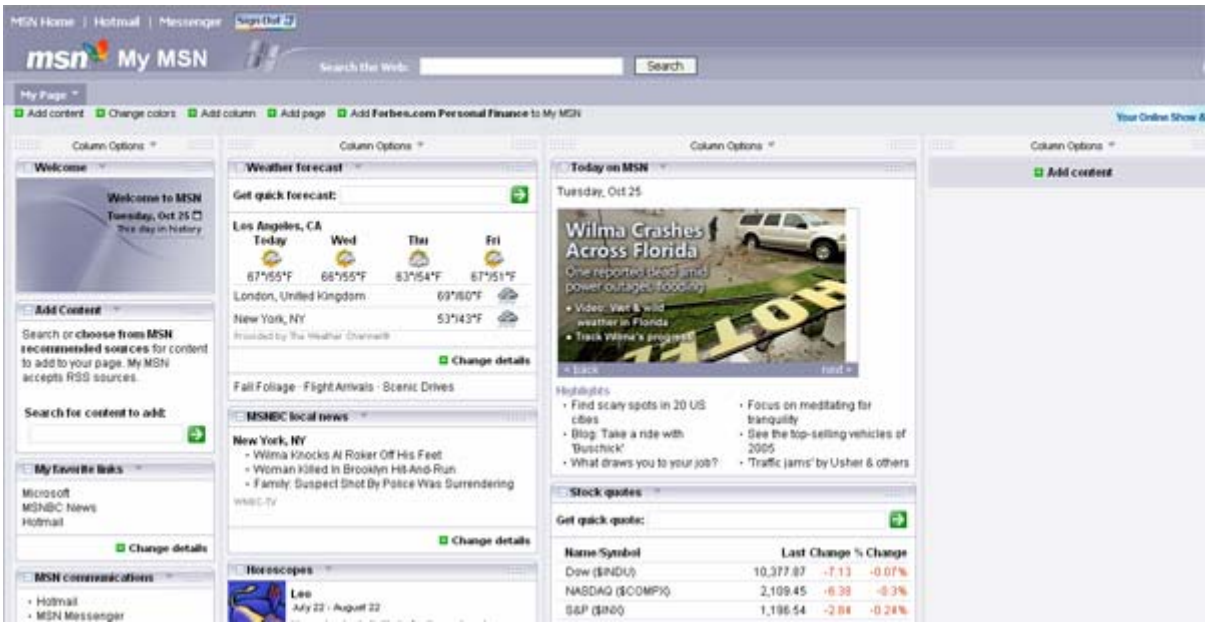
Základní možnosti sloupce web partů na My MSN



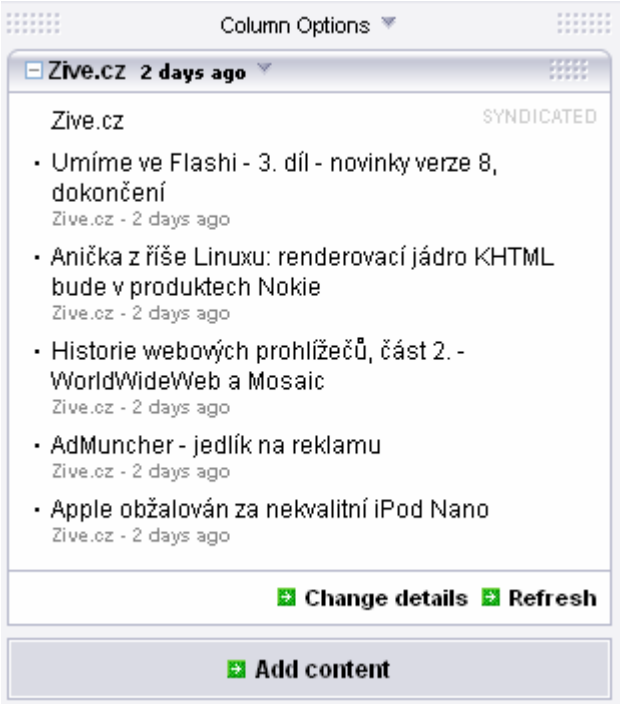
Základní možnosti web part stránky



Výběr tématu stránky na My MSN



Změna tématu My MSN



Přidání Web partu s načtením RSS Feed z Zive.cz

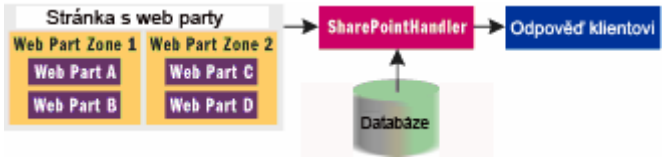
4 Princip

Části stránek by se dali, v jednom z dělení obsahu webových stránek, dělit na web party a ostatní části (ostatními částmi budeme rozumět jakýkoliv serverový i klientský prvek použitý mimo oblast web partů).
Oblasti web partů (nazývané zóny) tvoří jakýsi zastřešovací prvek (kontejner) pro web party, do nichž je lze přidávat, přidávat user controly, custom controly i jakékoliv jiné serverové prvky. Rozdíl mezi prvkem v zóně a mimo ni je tedy pouze v „zastřešení“ prvku a s tím přicházející funkčností (manipulací) tohoto prvku na stránce.

Samotná třída zóny „`WebPartZone`“ i ostatní zóny (které popíšu níže) jsou potomky třídy `WebZone` a ta je potomkem `CompositeControl` známého již z dřívější verze ASP.NETu. Všechny WebParty jsou zase potomky UserControl, který existuje v ASP.NETu také od začátku a vývojáři je často používán. Co tedy vlastně kromě jakési funkčnosti, kterou většinou obstarává JavaScript (a musím říct, že je to opravdu výborná funkčnost), přináší tato technologie navíc?
Jde tu o prolnutí dvou technologií a to technologie ASP.NET a tzv. Microsoft SharePoint Services. Samotný popis a princip této technologie je opět na jiné téma a proto se odvolávám na jiné zdroje, které dokáží celou tuto oblast popsat lépe. Ve spojení s WebParty má z této technologie význam především tzv. SharePointHandler, který tvoří jakéhosi prostředníka mezi WebPartem na serveru a výstupem u klienta. WebParty totiž nabízejí naprosto automatizovanou personalizaci webových stránek, pro jejichž fungování nemusí vývojář udělat prakticky nic. Tato personalizace může být použita v jiných technologiích, které také používají web party.

Tvorba web partu probíhá na serveru, kde ve chvíli kdy je na serveru vytvořena zóna i všechny její součásti (web party) je předáno řízení SharePointHandleru, který automaticky načte informace o uživatelských změnách z databáze, upraví výslednou odpověď (pozici webPartů, titulky, velikosti atd…) a následně je takto upravená odpověď

poslána na klienta. Klient tedy vždy dostane stránku upravenou tak jak ji naposledy zanechal. Automaticky také dochází při změnách web partů k uložení informací zpět na server s tím, že vývojář nemusí napsat ani čárku.
Celá databáze se tvoří sama a není třeba žádných změn v její struktuře ani obsahu. Ve stručnosti tedy vlastně dává technologie web partů vývojářům do ruky jednoduchý a efektivní nástroj, jak vytvořit modifikovatelný web bez jakéhokoliv přičinění.



Průběh zpracování web partů

Aby tento princip správně fungoval, musí existovat na daném počítači nebo na jiném, síťově propojeném s tímto, Windows SharePoint Server. Výše uvedené schéma a jeho princip má význam především v tzv. web farmách, nebo ve firmách, kde je zapotřebí propojit informace z SharePoint serveru s webem.

V ostatních případech lze toto omezení existence další služby (nebo serveru) jednoduše obejít a je nám nabízena druhá možnost – personalizace web partů může být jednoduše uložena v podadresáři webové aplikace. Aby daná věc fungovala, stačí jednou spustit webovou aplikaci, a vše je vytvořeno automaticky. V této práci i v jejích příkladech je použita tato možnost, protože nebyl k dispozici SharePoint Server.

4.1. Módy prohlížení

Web party se mohou nacházet v několika módech. Každý mód nabízí jiné funkčnosti a trošku odlišný způsob zobrazení. V zásadě je možné rozlišovat mezi pěti módy

- `BrowseDisplayMode` – klasický mód. Zobrazeny jsou pouze web party, uživatel má možnost prohlížet informace. Vetší funkčnost chybí.
- `ConnectDisplayMode` – propojovací mód. Slouží k propojování jednotlivých web partů a web zón.
- `CatalogDisplayMode` – katalog mód. Zobrazeny jsou jak web party tak web zóny, uživatel má možnost vrátit dříve vypnutý web part.
- `DesignDisplayMode` – design mód. Je možné měnit design web partů.
- `EditDisplayMode` – editovací mód. Je možné změnit některé informace na web partech, velikost, zónu apod.

4.2. Módy změn web partů

Web party mohou být pro uživatele konfigurovány dvěma způsoby, podle kterých ovlivní změna web partu pouze lokálního uživatele, nebo uživatelů jako celku. Pokud vývojář umožní některým uživatelům používat „`share`“ mód, znamená to, že tito uživatelé mohou provádět globální změny web partů, které se promítnou do nastavení všech uživatelů. Opakem je pak „`user`“ mód, jehož změny budou uloženy pouze pro aktuálního uživatele a nijak neovlivní obsah ani vzhled ostatních.

4.3. Módy controlů

Vývojář může při tvorbě aplikace povolit uživateli tzv. web party třetích stran, kde uživatel importuje nastavení web partu a následně celý web part do svého nastavení. To v důsledku znamená, že uživatel může zcela pozměnit obsah své aplikace tvorbou nových, nebo importováním cizích, web partů. Tyto web party se nazývají `per-user` control. Ty, které jsou uživatelům přednabízeny se nazývají `shared` controly.

4.4. Rozšíření web partů

Protože by technologie web partů přišla o velkou část svého kouzla, kdyby bylo možné do zón přidávat pouze potomky `webPart` třídy, bylo do technologie zahrnuto i rozšíření jakéhokoliv web server controlu, user controlu a custom controlu jako webpartu. Princip je jednoduchý – ve chvíli kdy JIT Compiler odhalí v zóně jakýkoliv prvek, který není potomkem třídy WebPart, „obalí“ ho funkčností `webpart` třídy, tak jako by skutečně jejím potomkem byl. To vše probíhá v JIT, tj. vývojář nemusí napsat opět ani čárku – celé obalení za něj udělá kompilátor. Jakýkoliv z těchto controlů může být prostě jednoduše přidán do zóny a je z něj automaticky webpart vytvořen. V samotné funkčnosti jsou nepatrné rozdíly – zatímco skutečný webPart je potomkem této třídy, obalený prvek je potomkem GenericWebPart a je zde několik menších omezení pro tyto prvky.

4.5. Web part controly

Na následujících řádcích bych rád popsal základní prvky web part technologie. V průběhu přepracování z ASP.NET 2 beta 1 do beta 2 došlo ke změně, vyloučení i přidání některých prvků. Já budu popisovat pouze prvky ve verzi beta 2, která je blíže finální podobě nové verze ASP.NET.

WebPartManager

Každá stránka obsahující jakýkoliv prvek web part technologie musí mít svého WebPartManagera (právě jednoho), který propojuje jednotlivé prvky i celé zóny v SharePointHandleru s databází. Tento manažer udržuje a následně ukládá všechny informace o WebPartZónách i WebPartech na stránce.
Není možné vytvořit web part stránku s jakýmkoliv obsahem bez tohoto prostředníka. Naštěstí je jeho použití tak snadné, že o jeho přítomnosti prakticky nemusíte vědět.

WebZone

Jednotný předek pro všechny zóny. Pravděpodobně se nikdy nesetkáte s jeho použitím.

WebPartZone

Základní zóna pro vkládání web partů, tvoří kontejner pro tyto prvky a udává chování i zobrazení těchto prvků jako celku. Web Part Zóna je viditelná pouze v některých módech.

WebPart

Tvoří základní stavební kámen celé technologie - vizuální modul obsahující informace. Jako s modulem je možné s ním také zacházet - přesouvat ho, měnit velikost, vypínat, měnit obsah i styl zobrazení. V existujících webech ho lze přirovnat k IFrame, s tím rozdílem, že nabízí mnohem větší možnosti a není externě načítán z jiného souboru - je to klasická HTML součást stránek se vším všudy.

CatalogZone

Stejně tak jako WebPartZone tvoří „zastřešovací kontejner“ pro webParty, tvoří CatalogZone stejné výhody pro CatalogParty. Účelem CatalogPartů je především spravovat existující webParty na stránkách – jejich zobrazení (vypnutí/zapnutí).

DeclarativeCatalogPart

Správa deklarovaných web partů – web partů, které nejsou implicitně přidány do stránek, ale uživatel je může podle své vůle přidat do svých stránek.

PageCatalogPart

Web part s výčtem webových partů na stránce. Zde je možné jednotlivé web party znovu aktivovat z dřívější deaktivace.

ImportCatalogPart

Správa importovaných partů.

EditorZone

Zóna pro editační web party. Platí pro ni stejná pravidla jako pro zóny předchozí. Zobrazuje se pouze v editačním módu.

AppearanceEditorPart

Prvek umožňující změnit vzhled web partu.

BehaviourEditorPart

Prvek ovlivňující chování web partu.

LayoutEditorPart

Prvek umožňující nastavení statusu web partu – minimalizovaný, maximalizovaný, nebo zavřený.

PropertyGridEditorPart

Flexibilní prvek, nabízející změnu uživatelsky nadefinovaných proměnných.

ProxyWebPartManager

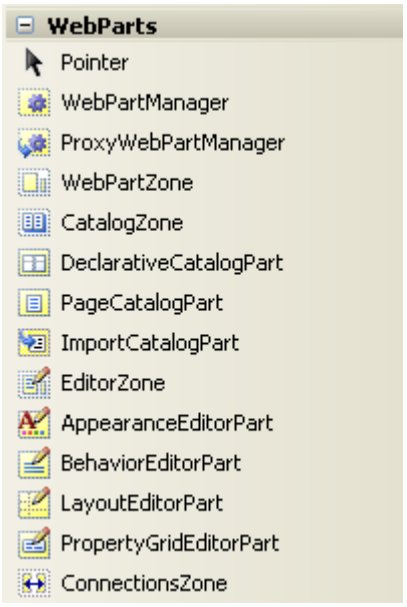
Manažer starající se o propojení importovaných web partů s aktuální stránkou.

Verbs

Verbs jsou příkazy, které je možné jednotlivým web partům udělit. Výčet příkazů se může měnit (a zpravidla mění) s různými typy módů. K implicitním příkazům patří Close a Minimalize. Vývojář může web partům přidělovat příkazy nové, ovlivňovat chování existujících, měnit zobrazení i ikony atd..

5 Příklad vytvoření web part stránky ve Visual Studiu 2005

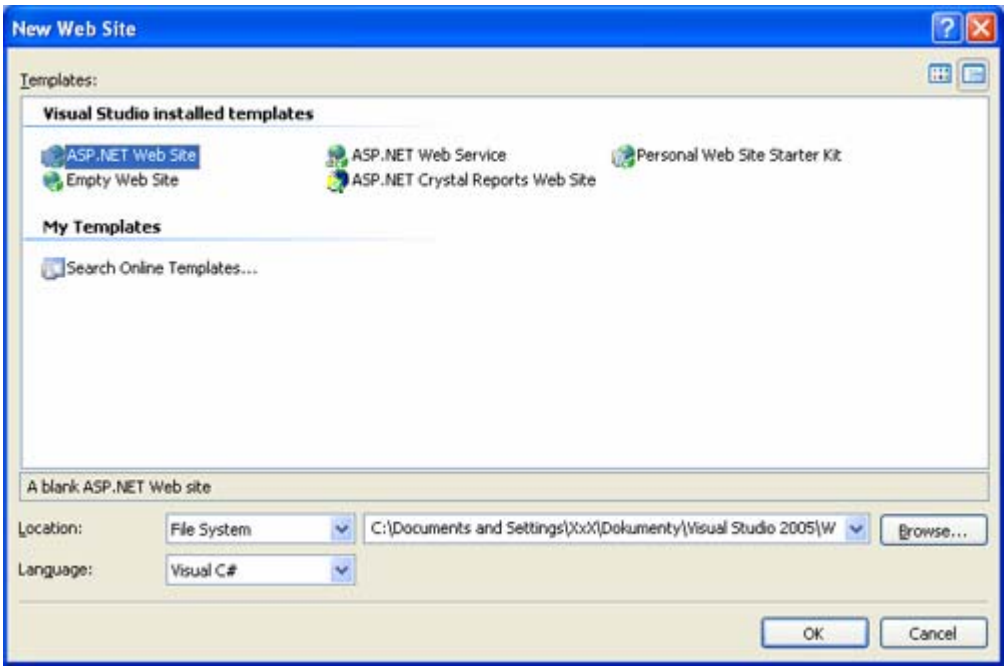
V následujícím příkladu se pokusím popsat a názorně ukázat vytvoření web part stránky, na které vytvořím několik webPartZone, přidám do těchto zón několik userControlů, web controlů, vytvořím a následně přidám jeden webPart a přidám mu několik verbs s určitým chováním. Na závěr pak předvedu chování v prohlížeči v jednotlivých módech.



Web part controly ve Visual Studiu 2005

5.1. Založení projektu

Otevřeme Visual Studio a zvolíme „New web site“.



Zvolíme „ASP.NET Web Site“, jazyk (v tomto případě Visual C#) a lokaci. Je prakticky úplně jedno kam svůj nový projekt umístíte. Není třeba ho, tak jako v dřívějších verzích ASP.NET, umísťovat do prostoru webového serveru, nebo vytvářet webovou aplikaci. Webová aplikace je vytvořena okamžitě sama, ať už máte svůj projekt kdekoliv a navíc nemusíte mít IIS. Novinkou v ASP.NET 2.0 je existence tzv. ASP.NET Development Serveru. Je to plnohodnotný IIS server, který funguje a zpracovává požadavky vzniklé pouze na localhostu a funguje na náhodně vybraných portech.Výhodou tedy je, že vývojář nemusí při vývoji zajišťovat bezpečnost nutnou pro provoz neodladěné aplikace a dokonce ani jakkoliv ladit IIS, protože prostě není potřeba.
V tuto chvíli tedy máme založen nový projekt.

5.2. Aktivace manažera

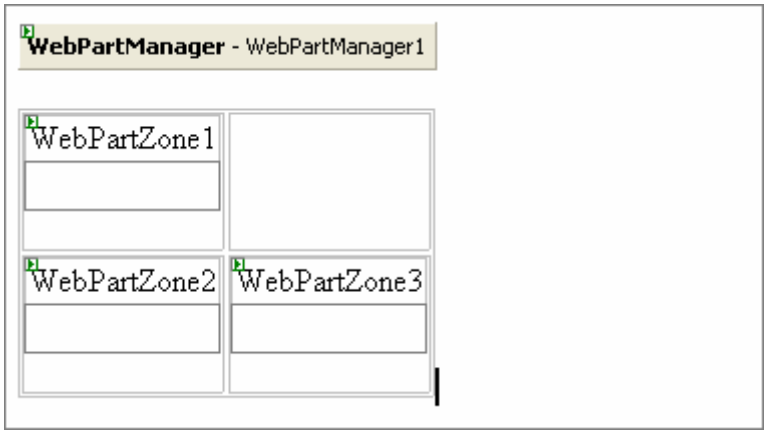
Jak již bylo psáno výše, je třeba u každé stránky používající web party vytvořit WebPartManagera. Není nic snazšího. Prostě si v toolboxu rozbalte záložku „WebParts“ a přetáhněte WebPartManagera na stránku. V nové verzi Visual Studia je úplně jedno zda přetáhnete prvek v Source View, nebo Design View. Oba pohledy spolu velice úzce spolupracují a můžete klidně vytvořit vše stejně lehce v kterémkoliv z nich.

Zatím prázdná stránka s WebPartManagerem

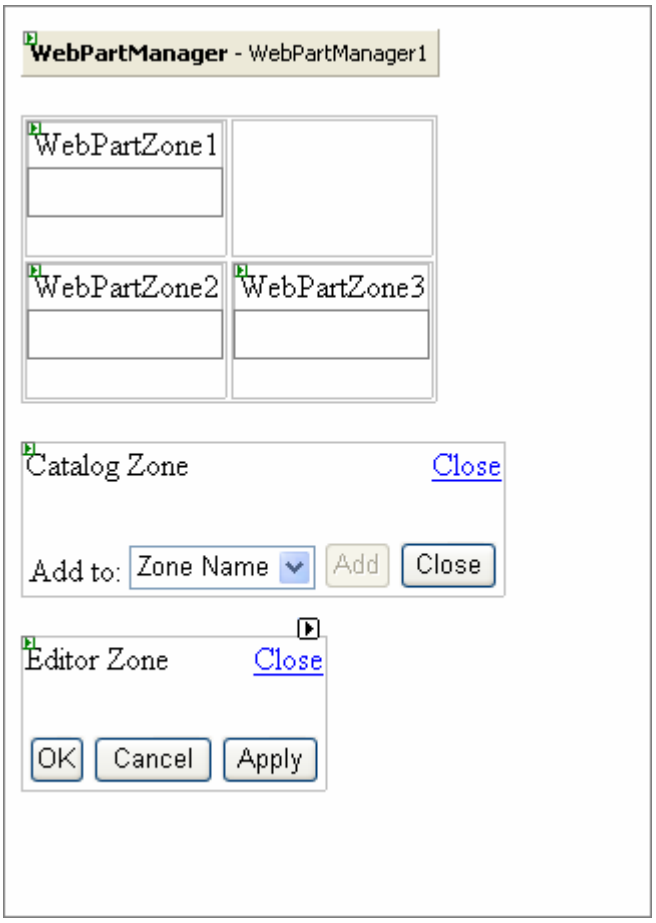
WebPartManager nebude ve výsledné stránce vidět. Takže ho je možné umístit kamkoliv na stránku.

5.3. Vytvoření zón

Zóny jsou vlastně jakési části stránek, do kterých se webParty umísťují, takže je pravděpodobné, že těchto částí budete mít víc. Vytvoříme proto tabulku, do které několik těchto zón přidáme.
V HTML prvcích vybereme Table a přetáhneme ji na stránku. Následně pak do těchto buněk naskládáme tak tři až čtyři WebPartZone, například takto



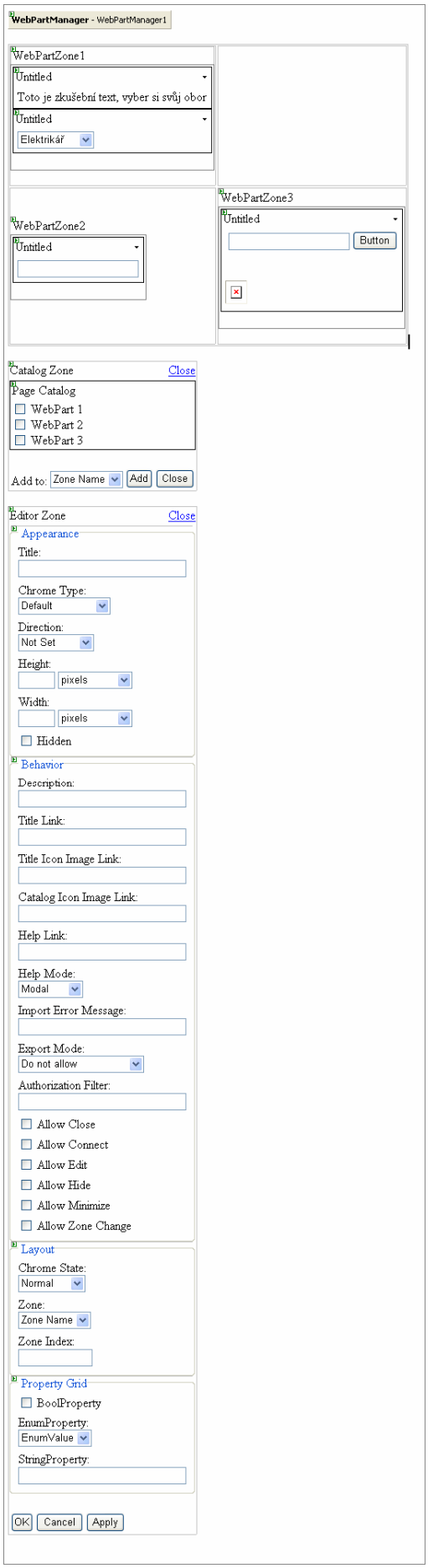
Následně přesuneme na stránku také EditorZone (chceme měnit vlastnosti jednotlivých web partů) a CatalogZone (chceme spravovat jednotlivé web party). Na konci tohoto bodu by tedy měl projekt vypadat přibližně takto



V této chvíli tedy máme fungující web part aplikaci. Máme managera starajícího se o spojení web partů s databází, zóny do kterých je možné přidávat jednotlivé party a následně pak také zóny připravené na editovací a katalogové prvky. Připomínám, že jsme zatím nemuseli napsat ani řádku kódu.

5.4. Přesun prvků do zón

Abych připomněl, co je popsáno výše – do zón lze vkládat jakýkoliv webový prvek. Ideální je, přidávat do zón potomky `webPart`, není to však podmínkou a v případě, že do zóny vložíte jiný, než web part prvek, Visual Studio za Vás tento „nedostatek“ vyřeší a změní typ z Control na WebPart s tím, že doplní chybějící části, nutné ke splnění podmínek WebPart. Můžeme tedy s klidem přesunout do zón jiné než webPart prvky. V příkladu přesouvám Label, TextBox, DropDownList a jeden narychlo vytvoření UserControl obsahující Buton, Image a TextBox, jehož úkolem je v případě správně zadané cesty v TextBoxu a kliku na Buton vložit obrázek do Image. Do PageZone vložte PageCatalogPart a do EditorZone můžete vložit všechny čtyři editorParty – AppearanceEditorPart, BehaviourEditorPart, LayoutEditorPart i PropertyGridEditorPart. Aplikace by měla v této chvíli vypadat asi takto


WebPartManager - WebPartManager1
WebPartZone1
Untitled
Toto je zkušební text, vyber si svůj obor
Untitled
Elektrikář
WebPartZone2
Untitled
WebPartZone3
Untitled
Button
Catalog Zone
Close
Page Catalog
WebPart 1
WebPart 2
WebPart 3
Add to:
Zone Name
Add
Close
Editor Zone
Close
Appearance
Title:
Chrome Type:
Default
Direction:
Not Set
Height:
pixels
Width:
pixels
Hidden
Behavior
Description:
Title Link:
Title Icon Image Link:
Catalog Icon Image Link:
Help Link:
Help Mode:
Modal
Import Error Message:
Export Mode:
Do not allow
Authorization Filter:
Allow Close
Allow Connect
Allow Edit
Allow Hide
Allow Minimize
Allow Zone Change
Layout
Chrome State:
Normal
Zone:
Zone Name
Zone Index:
Property Grid
BoolProperty
EnumProperty:
EnumValue
StringProperty:
OK
Cancel
Apply

Jak je vidět, Visual Studio za nás vytvořilo z daných controlů webParty. U každého z nich je vlevo nahoře šipečka na spouštění akcí a každému prvku byl přidán titulek, zatím nastavený na „Untitled“.
Nezapomeňte, že dané party je třeba vždy umisťovat do správných typů zón. Tj. editovací do EditorZone a katalogové do CatalogZone.

5.5. Tvorba web partu

V tuto chvíli vytvoříme jeden skutečný webPart. Tvorbou skutečného Webpartu (ne jen jeho „náhražky) získáváte některé výhody, které později ukážu.
Tvorba je velmi jednoduchá. Nejdříve přidáme do projektu nový soubor (třídu) a následně zkopírujeme do souboru následující obsah (který bude níže v této a následující kapitole podrobně vysvětlen).

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Security.Permissions;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace sample
{
    public class MyWebPart : WebPart
    {

        WebPartVerbCollection _verbs = null;
        private String _contentText = null;
        TextBox input;
        Label DisplayContent;

        public MyWebPart()
        {

        }

        protected override void CreateChildControls()
        {
            Controls.Clear();
            DisplayContent = new Label();
            DisplayContent.BackColor =
              System.Drawing.Color.LightBlue;
            DisplayContent.Text = this.ContentText;
            this.Controls.Add(DisplayContent);
            input = new TextBox();
            this.Controls.Add(input);
            Button update = new Button();
            update.Text = "Ulož podtitulek web partu";
            update.Click += new EventHandler(this.submit_Click);
            this.Controls.Add(update);

            for (int i = 0; i < CountOfTextBoxs; i++)
```

```
            {
                Label lbl = new Label();
                lbl.Text = "<br />Label " + i.ToString()+ " - ";
                TextBox tb = new TextBox();
                this.Controls.Add(lbl);
                this.Controls.Add(tb);

            }

            ChildControlsCreated = true;
        }

        private void submit_Click(object sender, EventArgs e)
        {
            if (input.Text != String.Empty)
            {
                this.ContentText = Page.Server.HtmlEncode(input.Text)
                                   + @"<br />";
                input.Text = String.Empty;
                DisplayContent.Text = this.ContentText;
            }
        }


        [
          Personalizable(PersonalizationScope.User, true),
          WebBrowsable()
        ]
        public String ContentText
        {
            get { return _contentText; }
            set { _contentText = value; }
        }

        private int CountOfTextBoxs
        {
            get
            {
                if (ViewState["countOfTextBoxs"] != null)
                {
                    return (int)ViewState["countOfTextBoxs"];
                }
                else return 0;
            }
            set
            {
                ViewState["countOfTextBoxs"] = value;
            }
        }




        public override WebPartVerbCollection Verbs
        {
            get
            {
                if (_verbs == null)
                {
                    WebPartVerb myVerb =
```

```
                    new WebPartVerb("O autorovi", OnWebPartAbout);
                myVerb.Description = "Informace o autorovi";
                myVerb.Text = "O autorovi";
                myVerb.ImageUrl = "~/images/admin.gif";

                WebPartVerb myVerb2 =
                    new WebPartVerb("Přidej Textbox", OnWebPartAdd);
                myVerb2.Description = "Přidání textboxu do web partu";
                myVerb2.Text = "Přidej textbox";
                myVerb2.ImageUrl = "~/images/docnew.gif";

                WebPartVerb myVerb3 =
                    new WebPartVerb("Odeber Textbox", OnWebPartRem);
                myVerb3.Description = "Odebrání textboxu z web partu";
                myVerb3.Text = "Odeber textbox";
                myVerb3.ImageUrl = "~/images/docdel.gif";

                _verbs = new WebPartVerbCollection(base.Verbs,
                    new WebPartVerb[] { myVerb, myVerb2,myVerb3 });
            }
            return _verbs;
        }
    }

    public void OnWebPartAbout(object sender, WebPartEventArgs e)
    {
        Page.Response.Write(@"<script>alert('Toto je příklad "+
           "vytvořený Pavlem Kubátem');</script>");

    }

    public void OnWebPartAdd(object sender, WebPartEventArgs e)
    {
        CountOfTextBoxs++;
        CreateChildControls();

    }

    public void OnWebPartRem(object sender, WebPartEventArgs e)
    {
        CountOfTextBoxs--;
        CreateChildControls();

    }

}
}
```

Jak je hned na první pohled vidět, je tato třída (MyWebPart) zděděna z třídy `WebPart` a obsahující kolekci (přesněji `WebPartVerbCollection`) příkazů (funkčností), řetězec a dva web controly.
Jednou z nejdůležitějších metod v potomkovy `WebPart` třídy je přetížení metody `CreateChildControls`, ve které se prakticky všechen obsah WebPartu vytváří.

```
    protected override void CreateChildControls()
    {
        Controls.Clear();
        DisplayContent = new Label();
        DisplayContent.BackColor =
          System.Drawing.Color.LightBlue;
```

```
            DisplayContent.Text = this.ContentText;
            this.Controls.Add(DisplayContent);
            input = new TextBox();
            this.Controls.Add(input);
            Button update = new Button();
            update.Text = "Ulož podtitulek web partu";
            update.Click += new EventHandler(this.submit_Click);
            this.Controls.Add(update);

            for (int i = 0; i < CountOfTextBoxs; i++)
            {
                Label lbl = new Label();
                lbl.Text = "<br />Label " + i.ToString()+ " - ";
                TextBox tb = new TextBox();
                this.Controls.Add(lbl);
                this.Controls.Add(tb);

            }

            ChildControlsCreated = true;
        }
```

Zde je velmi jednoduchý kód, který vytvoří Label a TextBox s určitými vlastnostmi a událostí, přidaných do Controls kolekce. V cyklu je pak do této kolekce přidáván Label a TextBox s určitým popisem v takovém počtu jako je velikost `CountOfTextBoxs`. Na konci je příznaku `ChildControlsCreated` přiřazeno `true`. V této metodě probíhá veškeré tvoření WebPartu a je třeba tuto metodu prakticky vždy přetížit.
Na následujících řádcích je pak docela nezajímavá metoda ošetřující klik na tlačítko, přiřazující určitý text Labelu.

```
        private void submit_Click(object sender, EventArgs e)
        {
            if (input.Text != String.Empty)
            {
                this.ContentText = Page.Server.HtmlEncode(input.Text)
                                   + @"<br />";
                input.Text = String.Empty;
                DisplayContent.Text = this.ContentText;
            }
        }
```

Vlastnost ContentText také prakticky není příliš zajímavá až na dva atributy, které jsou k ní přiřazeny.

```
        [
          Personalizable(PersonalizationScope.User, true),
          WebBrowsable()
        ]
        public String ContentText
        {
            get { return _contentText; }
            set { _contentText = value; }
        }
```

Tyto atributy totiž velkou měrou přispívají k personalizaci stránek. Atribut `Personalizable` udává že, tato vlastnost se má ukládát a lze u ni specifikovat, zda je měnitelná pro každého uživatele - user, nebo pouze pro všechny – shared.
Atribut `WebBrowsable` udává, že je vlastnost měnitelná v režimu edit.

Soukromá vlastnost CountOfTextBoxs také není příliš zajímavá, pouze pro začátečníky v ASP.NETu dodávám, že způsobem popsaným v této metodě je možné jednoduše ukládat proměnné (mohou být ale stejně jednoduše ukládány do Session objektu).

```
private int CountOfTextBoxs
{
    get
    {
        if (ViewState["countOfTextBoxs"] != null)
        {
            return (int)ViewState["countOfTextBoxs"];
        }
        else return 0;
    }
    set
    {
        ViewState["countOfTextBoxs"] = value;
    }
}
```

5.6. Vytvoření nové funkčnosti – verbs

V tuto chvíli je vytvořen jednoduchý WebPart. My ale chceme našemu WebPartu přidat další ze skvělých vymožeností – Verbs neboli funkčnosti.

V překryté vlastnosti Verbs je vracena kolekce funkcí, které může daný webPart plnit. Pokud chcete přidávat nebo naopak odebírat některé funkce (odebírání většinou není třeba díky možnosti změny příznaků `AllowClose`, `AllowMinimize`, atd.měnit), musíte překrýt tuto metodu a nadefinovat ji sami.

```
public override WebPartVerbCollection Verbs
{
    get
    {
        if (_verbs == null)
        {
            WebPartVerb myVerb =
               new WebPartVerb("O autorovi", OnWebPartAbout);
            myVerb.Description = "Informace o autorovi";
            myVerb.Text = "O autorovi";
            myVerb.ImageUrl = "~/images/admin.gif";

            WebPartVerb myVerb2 =
               new WebPartVerb("Přidej Textbox", OnWebPartAdd);
            myVerb2.Description = "Přidání textboxu do web partu";
            myVerb2.Text = "Přidej textbox";
            myVerb2.ImageUrl = "~/images/docnew.gif";

            WebPartVerb myVerb3 =
               new WebPartVerb("Odeber Textbox", OnWebPartRem);
```

```
                myVerb3.Description = "Odebrání textboxu z web partu";
                myVerb3.Text = "Odeber textbox";
                myVerb3.ImageUrl = "~/images/docdel.gif";

                _verbs = new WebPartVerbCollection(base.Verbs,
                    new WebPartVerb[] { myVerb, myVerb2,myVerb3 });
            }
            return _verbs;
        }
    }
```

V této metodě dochází (v případě, že je kolekce stále prázdná) k naplnění kolekce bázovými funkcemi a přidáním tří nových, kde jsou u každé udány metody ošetřující vznikl události.

V následujících metodách ošetřující vznik událostí je vytvořena pouze jednoduchá funkčnost pro ilustraci chodu web partu.

```
    public void OnWebPartAbout(object sender, WebPartEventArgs e)
    {
        Page.Response.Write(@"<script>alert('Toto je příklad "+
           "vytvořený Pavlem Kubátem');</script>");

    }
```

Jednoduchým způsobem, využívající JavaScript, je vytvořeno okénko, ve kterém se zobrazuje zpráva o autorovi.

```
    public void OnWebPartAdd(object sender, WebPartEventArgs e)
    {
        CountOfTextBoxs++;
        CreateChildControls();
    }
```

V metodě OnWebPartAdd a OnWebPartRem je inkrementována resp. dekrementována hodnota CountOfTextBoxs a následně znovu volána metoda CreateChildControls.

```
    public void OnWebPartRem(object sender, WebPartEventArgs e)
    {
        CountOfTextBoxs--;
        CreateChildControls();

    }
```

Ve shrnutí se tedy dá říci, že máme WebPart s určitým titulkem a třemi web controly, (z nichž díky dvěma můžeme měnit obsah třetího - jehož obsah je díky atributům automaticky ukládán do personalizační databáze a znovu vracen). Přidali jsme několik nových funkčností našemu WebPartu, kde jedna zobrazuje informace o autorovi a další dvě pak snižují nebo zvyšují počet pomocných prvků ve webPartu.

5.7. Změna módů

Protože budeme chtít web part i měnit a ovlivňovat jeho vlastnosti (proto jsme také přidávali jiné než webPart zóny), musíme na hlavní stránce vytvořil prvek, který bude módy měnit.
Nejdříve tedy přidáme na stránku `DropDownList` a pojmenujeme ho např. `DisplayModeDropdown`. Následně pak přidáme do metody `Page_Load` kód, který naplní tento `DropDownList` hodnotami. Metoda tedy bude vypadat následovně

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        foreach (WebPartDisplayMode mode in
          WebPartManager1.SupportedDisplayModes)
        {
            string modeName = mode.Name;
            if (mode.IsEnabled(WebPartManager1))
            {
                ListItem item = new ListItem(modeName, modeName);
                DisplayModeDropdown.Items.Add(item);
            }
        }

    }
}
```

Metoda má v celku jednoduchý kód. Všechny podporované a právě povolené módy přidej do `DropDownListu`.

Následně pak vytvoříme metodu událost `SelectIndexChanged` jako

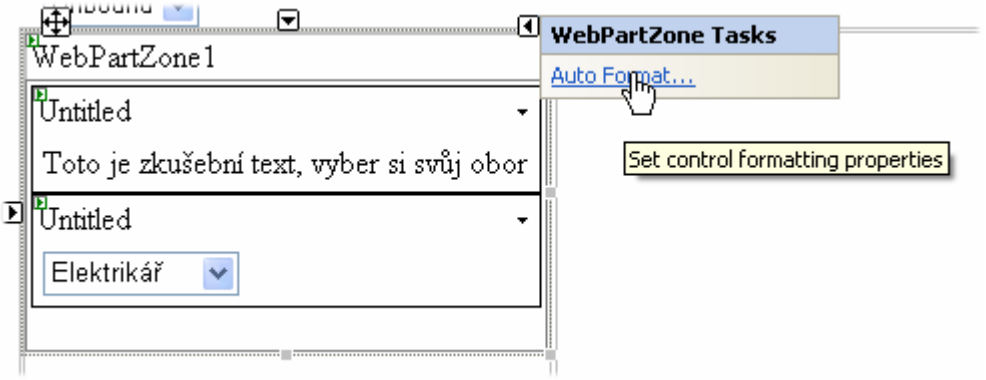
```
public void DisplayModeDropdown_SelectedIndexChanged(object
      sender,EventArgs e)
{
    string selectedMode = DisplayModeDropdown.SelectedValue;

    WebPartDisplayMode mode =
     WebPartManager1.SupportedDisplayModes[selectedMode];
    if (mode != null)
        WebPartManager1.DisplayMode = mode;
}
```

kde je zjištěn přepnutý mód, vybrán z podporovaných formátů a v případě úspěšného nalezení také přepnut.

Máme tedy vytvořenu web part stránku i s přepínáním módů.

5.8. Formátování

Příkladový web part je prakticky dokončen. Nevypadá ale příliš vábně, proto ho budeme trošku formátovat. Formátovat budeme pouze základními formáty, které mají web part zóny implicitně nastaveny. Ve Visual Studiu tedy stačí pouze označit v designeru danou zónu a kliknout na malou šipku ukazující se vpravo nahoře.



Vybrání formátování zóny

Pro funkčnost příkladu není důležité zvolené formátování, proto nebudu detailně popisovat jeho volbu. Já zvolil pro všechny web zóny styl `classic` a pro editační a katalogové zóny styl `colorful`.

5.9. Chování aplikace za běhu

Aplikace je vytvořena. Spusťme ji v prohlížeči. Pokud jste postupovali správně, měla by se Vám zobrazit stránka podobné této

Výsledná aplikace

Nahoře je DropDownList na přepínání módu, stránka se skládá ze tří zón a v nich je pět webpartů - čtyři vytvořil JIT compiler a jeden my.
Hned první věc, kterou můžeme vyzkoušet je minimalizace a maximalizace jednotlivých webpartů, dále pak jejich zavření a změna podtitulku námi vytvořeného webpartu. Nejen, že by všechno mělo automaticky fungovat (prakticky bez našeho

zásahu), ale také jakákoliv z výše uvedených změn bude i po zavření a znovuotevření znovu načtena, tak jako by jsme aplikaci vůbec nevypnuli.
Nyní můžeme přepnout do jiného módu, z nichž zajímavé pro nás jsou `Catalog` a `Edit` módy. Mód `Design` pro nás není v tuto chvíli příliš zajímavý, protože nám neposkytuje žádnou funkčnost navíc. Vše co můžeme změnit v módu `Design` lze provést také v módu `Browse`. Mód `Design` existuje především proto, aby v případě, že bude chtít autor designovat stránku ve větší míře, přidal funkčnost do tohoto módu (funkčnosti se dají přidávat do každého módu zvlášť).
Ve chvíli kdy v módu `Browse` zavřeme některý z partů, zjistíme, že ho nelze v tomto módu opětovně otevřít. Je proto třeba přepnout mód na `Catalog`, zde vybrat příslušný part a zónu, do které chcete part přiřadit.

Nejzajímavější je ale asi mód `Edit`. Po jeho přepnutí si asi nevšimnete žádné změny. Jsou sice zobrazeny zóny (což v módu `Browse` nejsou), ale jinak se žádný z partů vytvořených v `edit` zóně neobjevil. Je totiž třeba nejdřív vybrat jeden z web partů. Editovat není možné všechny party, ale jen ty, kde vývojář příznakem `allowEdit` povolil jejich editaci. V případě, že vyberete jeden z web controlů, otevřou se Vám pravděpodobně pouze party `AppearanceEditPart` a `LayoutEditPath`, kde je možné měnit titulek, ohraničení (u web partů chrome), šířku a výšku, zónu atd.. V případě, že ale vyberete web part, mající `webbrowseable` atribut u některé vlastnosti (editujte třeba námi vytvořený web part), otevře se Vám i `PropertyGridEditPart`, kde můžete hodnotu s touto vlastností měnit (v našem příkladu je editovanou hodnotou podtitulek ve vytvořeném webpartu, který lze přiřadit i přímo ve web partu).

Námi vytvořený web part v módu editace (přidána funkce Edit)

Funkčnost web partů lze dále rozšiřovat, ale není v rozsahu této práce všechny možnosti vyzkoušet. Jako základní příklad použití web partů je předchozí ukázka dostatečným vodítkem k dalšímu studování.

6 Nasazení

Vývojáři ASP.NETu se snažili zpřístupnit svoji technologii co největšímu množství uživatelů internetu a proto jsou Web party (stejně jako celý ASP.NET) zcela nezávislé na platformě i typu prohlížeče klienta. Jediné co je skutečně potřeba (aby webová aplikace fungovala jak má) je prohlížeč HTML/XHTML spolu s podporou JavaScriptu (bez něhož se už většina moderních prohlížečů neobejde). Výhody této technologie jsou tedy potlačeny pouze u textových prohlížečů typu Lynx nebo velmi starých typů grafických prohlížečů.
Přestože se Microsoft snaží své technologie jednoznačně upřednostňovat před konkurenčními, v tomto případě udělal výjimku a aby bylo možné prohlížet ASP.NETovou webovou aplikaci i na jiných prohlížečích než je IE (a jeho nadstavbách), zahrnul do této technologie i multi-browser podporu, kde se skripty JavaScriptu generují dynamicky podle typu prohlížeče, tak aby byla vždy zachována funkčnost aplikace.
Web party tedy fungují v jakémkoliv prohlížeči a pouze s několika výjimkami naprosto totožně jako v IE (kde je funkčnost stoprocentní). Asi nejmarkantnější omezení, kterého si bezpochyby všimne každý uživatel téměř okamžitě je, že při přesunu v jiných než IE prohlížečích nebude přesouvaný web part průhledný. Důvod je samozřejmě v CSS a přidaných „filtrech“ firmy Microsoft, kterou podporuje jenom IE. Ostatní změny jsou prakticky zanedbatelné.

7 Licence a práva

8 Závěr

Web Party (ale i ASP.NET 2.0 jako celek) mě skutečně nadchly. Něco jako možnost jednotného vzhledu (nebo aspoň stylu vzhledu) webových aplikací tady dlouho chybělo. Tvorba těchto aplikací je opravdu snadná a vývojář se nemusí zabývat detaily funkčnosti těchto web partů a soustředí se na obsah a vytvoření informačně bohatého webu, o což jde nejen autorům ASP.NETu, ale i nám – vývojářům.

9 Literatura

[1] Microsoft developer's resources:
http://www.msdn.com
[2] Clipcode, Web parts:
http://www.clipcode.biz/stream/07_Web_Parts.pdf
[3] QuickStart intelligence, Introduction to ASP.NET 2.0:
http://www.quickstart.com/training/seminars/ netsql_thx/ASP.Net_2.0_Seminar_New.pdf
[4] www.asp.net, An introduction to ASP.NET 2.0:
http://www.asp.net/whidbey/misc/chap1.pdf
[5] DevX:
http://www.devx.com
[6] Internet obecně, většina poznatků:
http://www.google.com