Microsoft .net

Ľuboslav Lacko

Vývoj webových aplikácií pre začiatočníkov pomocou Express nástrojov

Microsoft®

Vývoj webových aplikácií pre začiatočníkov pomocou Express nástrojov

Príklady a pomocné súbory k brožúre si môžete stiahnuť na adrese
http://www.microsoft.cz/akce/msdn_brozury/

Microsoft®

Kapitola 1:
Úvod

Ak by sme sa pokúšali o kategorizáciu webových aplikácií jedným zo zaujímavých kritérií môže byť rozdelenie na komerčné a hobby aplikácie. Do druhej skupiny hobby aplikácií, ktorá bude hlavnou náplňou tejto publikácie patria stránky podporujúce najrôznejšie záujmy, zberateľské aktivity a podobne. Slúžia pre sprostredkovanie informácií v určitej väčšej alebo menšej komunite a asi nepreženieme že v súhrne oslovujú nás všetkých. Veď kto by sa nedal podľa svojich záujmov zaradiť aspoň do jednej zo skupín ktoré zbežne vymenujeme – zberateľ čohokoľvek, záhradkár, chovateľ, domáci kutil... vymenovať všetky záujmy snáď ani nie je možné. Sem patria aj stránky študentov a rôznych záujmových organizácií. V tejto skupine vývojárov a prevádzkovateľov webových aplikácií je zrejme najviac tvorivého potenciálu ale asi najmenej financií. Niečo sa dá získať na reklamách, no na nákup profesionálneho komerčného vývojového prostredia to nestačí, o databázových serveroch ani nehovoriac. Preto logicky v segmente hobby aplikácií kraľujú voľne šíriteľné produkty pričom azda najčastejšie sa využíva kombinácia operačného systému LINUX, webového servera APACHE, databázového servera MySQL a skriptového systému PHP. Toto je aj obsahom ponukových balíčkov väčšiny webhostingových firiem. Po tom, ako sa pred rokmi podarilo Microsoftu obsadiť so svojimi klientskými operačnými systémami Windows a prehliadačom webových stránok Internet Explorer rozhodujúci podiel na trhu sa teraz snaží Microsoft presadiť sa aj na serverovej strane, veď webovú stránku má dnes nielen každá firma ale aj veľa jednotlivcov takto prezentuje svoje záujmy. Navyše tieto kategórie sa samozrejme aj prelínajú, kedy napríklad úspešný manažér veľkej firmy môže byť aj zanieteným pestovateľom kaktusom, chovateľom exotických živočíchov a podobne. Pozícia vývojárskych nástrojov Micosoftu pre podnikové internetové a intranetové aplikácie je vďaka vývojovému prostrediu Visual Studio .NET 2003 (a pripravovanej verzie 2005 s kódovým označením Whidbey) pomerne pevná. Nakoľko takéto vývojové prostredie však stojí niekoľko desaťtisíc korún, hobby sféru samozrejme v tejto podobe nijako neosloví.
Microsoft je jedným z najvýznamnejších hráčov aj na databázovom trhu vďaka SQL Serveru 2000. Aj dosiaľ síce existovala odľahčená verzia jeho databázového jadra s označením MSDE, no obmedzenia na maximálne 5 súčasne vykonávaných dopytov jeho využitie vo webových aplikáciách dosť limitovala. Doptyty museli byť optimalizované tak aby ich vykonávanie trvalo čo najkratšiu dobu.

Technológie Microsoftu pre Hobby vývojárov

Disponujúc kvalitnými nástrojmi pre podnikovú sféru v snahe osloviť hobby vývojárov začal Microsoft vyvíjať „odľahčené“ verzie vývojárskych nástrojov a databázového servera. Prvým krokom na poli „hobby“ vývojárskych nástrojov bol v nedávnej minulosti projekt Web Matrix doplnený o ASP.NET Starter Kit. Web Matrix mal v porovnaní s Visual Studiom pomerne veľa obmedzení, hlavne to, že nebol projektovo, ale súborovo orientovaný a projekt vlastne tvorilo viac spolupracujúcich ASP.NET stránok v jednom adresári. Na druhej strane zasa ponúkal veľa šablón pre jednotlivé typy stránok, napríklad pre zobrazovanie obsahu databázových tabuliek a podobne. Tento projekt bol však utlmený a nahradený projektom Visual Web Developer 2005. Webová aplikácia bez databázy však príliš veľký význam nemá a preto vývojové prostredie pre takéto aplikácie bez databázového serveru by bolo značne obmedzené. V prípade Visual Web Developera 2005 sa problémov s databázou obávať nemusíme. Rada produktov edície Express obsahuje v obmedzenom rozsahu prakticky celé portfólio vývojárskych aplikácií a patrí sem aj SQL Server 2005 (kódové označenie Yukon) Express Edition.

Čo potrebujeme pre prevádzku jednoduchej webovej aplikácie

Webová aplikácia potrebuje na strane servera tieto služby:

- operačný systém podporujúci sieťové služby
- webový server
- databázový server

Môžu to byť napríklad prostriedky od Microsoftu: Operačný systém Windows 2000/XP/2003 Server, Internet Information Server (IIS) a databázový server Microsoft SQL Server 2000, alebo nová verzia SQL Server 2005 (kódové označenie Yukon). Niektoré z týchto programov sú alebo boli pomerne drahé, preto sa v hobby sfére používajú hlavne voľne šíriteľné programy najčastejšie prevádzkované pod operačným systémom LINUX, napríklad Apache web server a databázový server MySQL. Predmetom záujmu tejto publikácie bude kombinácia

- operačný systém Windows XP/2003 Server
- webový server IIS
- databázový server SQL Server 2005 Express Edition

Ako funguje webová aplikácia

Klient prostredníctvom prehliadača web stránok zadá adresu požadovanej stránky. Podľa navigačných prvkov na hlavnej stránke sa donaviguje na stránku obsahujúcu predmet jeho záujmu. Takto to vyzerá z pohľadu klienta. Čo sa však deje v pozadí na webovom serveri, kam sú smerované klientove požiadavky? Webový server na základe týchto požiadaviek postupne generuje HTML stránky, ktoré budú prostriedkami siete doručené klientovi. Stránky budú pravdepodobne obsahovať texty, tabuľky, obrázky, komponenty a skriptové kódy. Vývojári aplikácie síce poznali námet aplikácie (internetový časopis, internetový obchod...) samozrejme ale nemohli predpokladať, aké články sa budú na publikačnom portáli publikovať, prípadne aký konkrétny tovar sa bude predávať, ani čo si ktorý klient bude prezerať, alebo objednávať. Preto HTML stránky nie sú uložené na diskoch servera v statickej podobe, ale sa dynamicky generujú podľa požiadaviek klientov, na základe informácií o jednotlivých tituloch, uložených v databáze. Preto okrem HTML kódu budeme potrebovať jednak mať nainštalovaný a samozrejme ho do potrebnej hĺbku zvládnuť nejaký programovací jazyk, alebo skriptový systém, napríklad PHP, Javu, Perl, alebo ASP.NET, ktorý je predmetom tejto publikácie. Vo väčšine prípadov sa nezaobídeme ani bez základných znalostí databázového jazyka SQL. budeme potrebovať aj nejaký programovací jazyk alebo skriptový systém. A aby to nebolo také jednoduché, aj HTML stránky, generované serverom, ktoré sa potom zobrazujú u klienta obsahujú spravidla okrem HTML kódu aj kódy v skriptovom jazyku, ktoré sa vykonávajú pre zmenu na klientskom počítači. Na strane klienta sú najrozšírenejšie skriptové jazyky **JScript** a **VB Script**.

Vývojové prostredie Visual Web Developer poskytuje vynikajúce možnosti pre vývoj rôznych typov webových aplikácií. Dostupnosť beta verzií umožňuje vývojárom zvládnuť túto technológiu ešte v predstihu pred uvedením produkčnej verziu. Vzhľadom na to, že Visual Web Developer Express edície je kompatibilný s modulom pre vývoj webových aplikácií v plnej verzii Visual Studia 2005, znalosti získané pri vývoji hobby aplikácií môžeme neskôr zúročiť v aplikáciách firemných.

Kapitola 2

Vývojové prostredie Visual Web Developer 2005 Express Edition

Visual Studio.NET a WebMatrix, predchodcovia alebo súrodenci?

Ak by sme sa snažili nejako zaradiť produkt Visual Web Developer 2005 Express Edition do rodiny vývojárskych nástrojov Microsoftu, snažili by sme sa identifikovať jeho predchodcov a taktiež produkty tvoriace s Visual Web Developerom produktovú radu. Medzi predchodcov môžeme zaradiť komerčné vývojové prostredie Visual Studio .NET 2003 a voľne šíriteľný vývojársky nástroj pre vývoj webových aplikácií WebMatrix. V pozícii „veľkého súrodenca“ vystupuje nové vývojové prostredie Visual Studio 2005.
K dispozícii je taktiež projekt ASP.NET Starter Kit, koncipovaný ako množina príkladov alebo prototypových aplikácií pre technológiu ASP.NET. Jednotlivé časti projektu sa dajú získať z webu z adresy www.asp.net. Ak by sme porovnali možnosti predchádzajúcej verzie Visual Studia a Web Matrixu, Visual Studio ponúka oveľa väčšie možnosti, no Matrix má zasa bohatú ponuku šablón pre jednotlivé typy stránok. Na základe skúseností z obidvoch typov vývojových prostredí bol do nového vývojového prostredia zakomponovaný blok **Visual Web Developer „Whidbey“**, Je to nástroj pre vývoj internetových a Intranetových aplikácií a webových služieb využívajúcich technológie ASP.NET webových formulárov. Visual Web Developer využíva rovnaké IDE ako ostatné časti vývojového prostredia. Visual Web Developer je v súčastnosti dodávaný (v dobe písania publikácie ako beta verzia) aj ako samostatný produkt pre vývojárov zameraných výhradne na webové aplikácie, ktorý je možné nainštalovať a používať bez ostatných častí Visual Studia.

Najvýznamnejšie novinky rodiny vývojových prostredí Visual Studio 2005

Zhrnúť v priestore jedného odseku všetky novinky vývojového prostredia Visual Studio 2005 (kódové označenie „Whidbey“) pochopiteľne nie je možné a tak spomenieme len najvýznamnejšie, medzi ktoré patrí podpora nielen doterajšej 32 bitovej platformy ale aj 64 bitových platforiem IA-64 (Itanium) a x64 (Intel/AMD procesory s rozšírenou zbernicou. , spolupráca s novým databázovým serverom SQL Server 2005 (kódové označenie „Yukon“), podpora vývoja aplikácií pre spoluprácu s kancelárskym balíkom MS Office 2003, podpora vývoja aplikácií pre tímovú spoluprácu a vývoj aplikácií pre mobilné platformy Pocket PC a Smartphone.

Vývojové prostredie Visual Studio 2005 sa bude dodávať vo verziách
– Express
– Standard
– Professional
– Team

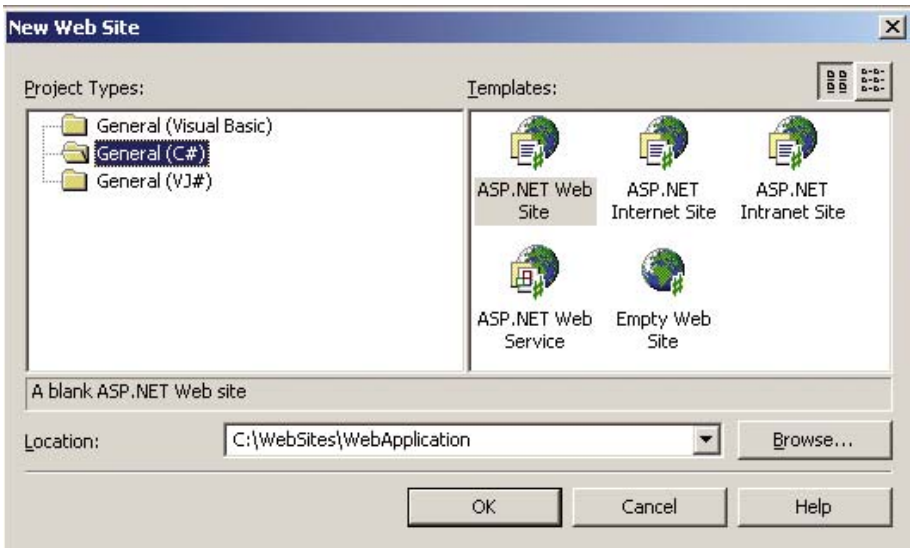
Pričom verzia Express bude k dispozícii zdarma, prípadne za cenu médií.

Za najvýznamnejšiu novinku vzhľadom na zameranie tejto publikácie môžeme pokladať zapúzdrenie všetkých častí pre vývoj webových aplikácií do jedného bloku s názvom Visual Web Developer. Vo verziách /Standard/Professional/Team je Visual Web Developer integrovaný do vývojového prostredia, vo verzii Express ide o samostatný produkt. Jedným z hlavných cieľov nového vývojového prostredia je uplatnenie jednoduchosti a bezpečnosti nasadenia webových aplikácií aj u klientských aplikácií. Skúsenosti s vývojom webových a klientských aplikácií priniesla so sebou vylepšenie grafického používateľského rozhrania (GUI) a ovládacích prvkov (vylepšený GridView, nová lišta „WinBar“, Splitter, komponenta pre asynchrónnu komunikáciu, komponenta Web Browser…), zjednodušenie práce s údajmi v databázach a výraznejšiu podporu RAD (Rapid Application Development), zjednodušene povedané – menej kódu a významnejšia podpora grafického návrhu používateľského rozhrania aplikácie.

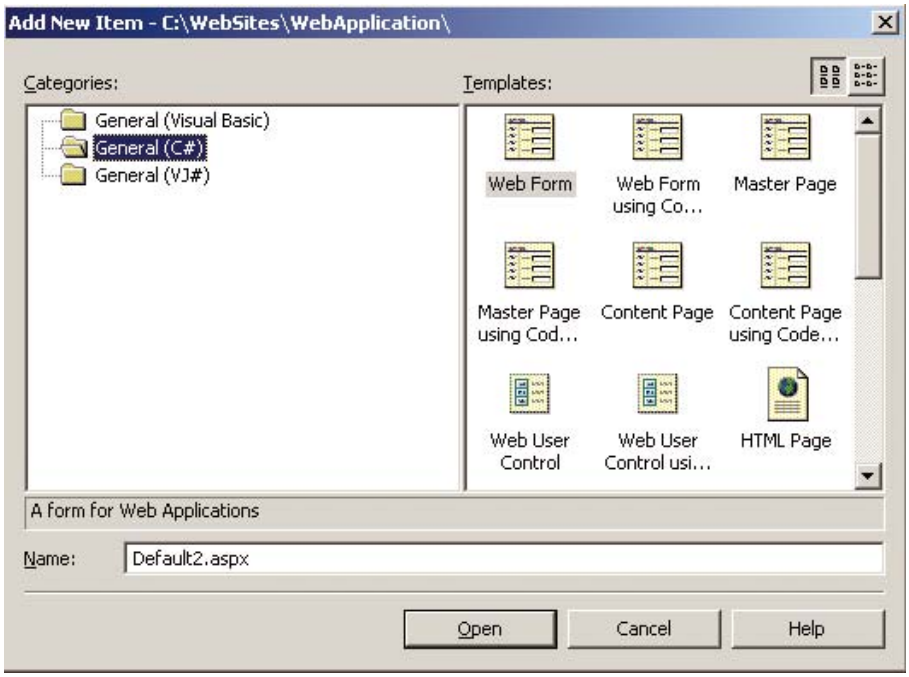
Podporované typy projektov

Pri vytváraní nových projektov v plnej verzii Visual Studia 2005 si môžeme v oknách šablón pre jednotlivé typy projektov všimnúť, že tam chýbajú ikony pre ASP.NET webové aplikácie, či už klasické, alebo webové aplikácie podporujúce prístup z mobilných klientských zariadení, ikony pre Webové služby... proste ikony pre všetky typy webových aplikácií. Ak sa však v menu pre vytvorenie nového projektu vrátime o úroveň vyššie („Web sites“), zistíme že tam pribudla položka pre vývoj webových projektov. V zložkách pre typy projektov v programovacích jazykoch Visual Basic, C# a J# sú šablóny

- ASP.NET Web Site
- ASP.NET Internet Site
- ASP.NET Intranet Site
- ASP.NET Web Service
- Empty Web Site



Web Developer – ponuka projektov

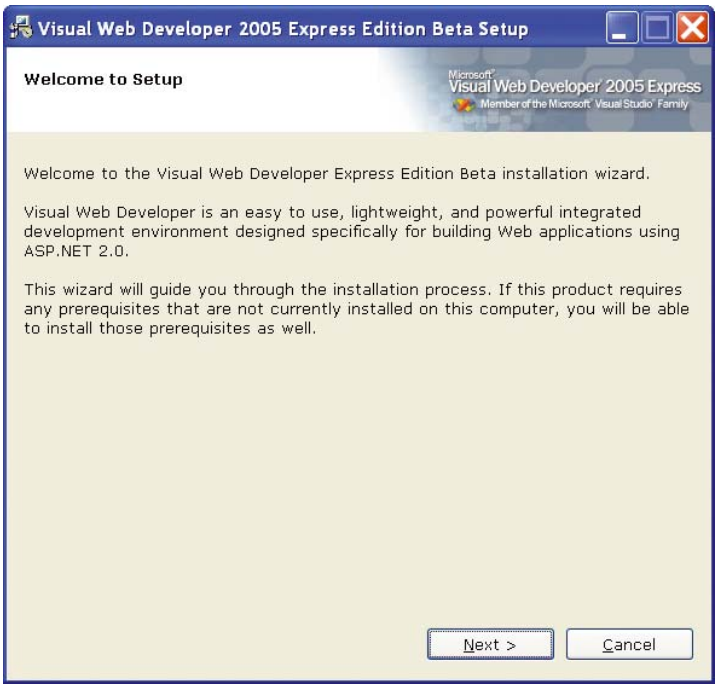


Web Developer – šablóny webových formulárov

Pri vývoji webových aplikácií prostredníctvom Visual Web Developera sa nevyžaduje mať nainštalovaný Internet Information Server (IIS) a toto vývojové prostredie rieši aj otázku umiestnenia aplikácie na server pomocou integrovaného FTP klienta, prípadne celý projekt môžeme pomocou XCOPY presunúť do požadovaného adresára.

Inštalácia

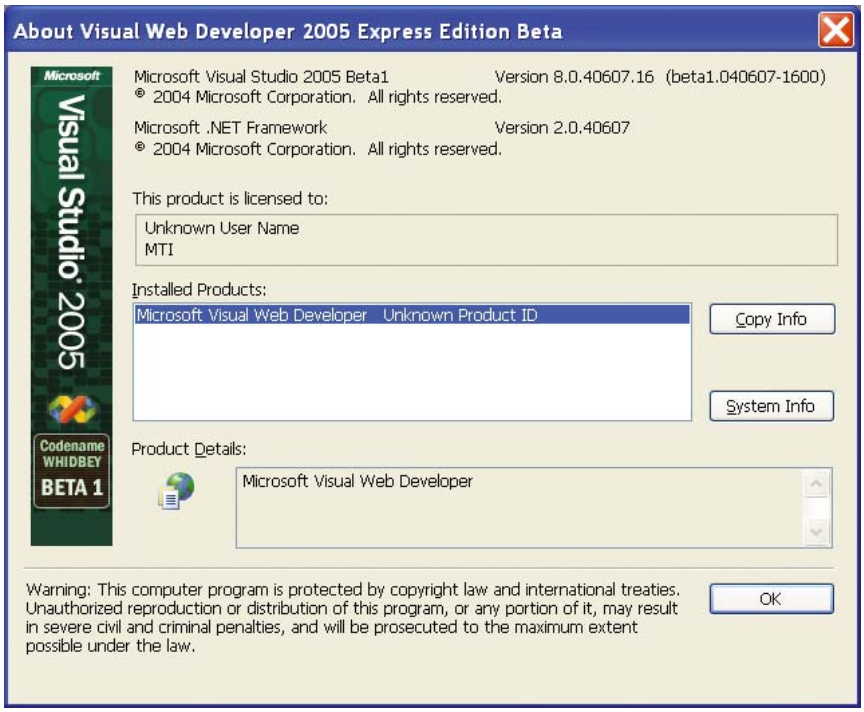
Vzhľadom na úzku spoluprácu webovej aplikácie s databázovým serverom budeme vo väčšine prípadov inštalovať Visual Web developer spolu s databázovým serverom. Tak je koncipovaný aj dialóg inštalačného programu Visual Web Developera pre voľbu inštalovaných komponentov.



Úvodný dialóg inštalácie Visual Web Developera

Inštalácia Visual Web Developer

Odporúčame nainštalovať aj dokumentáciu (Microsoft MSDN Express Library 2005). Záujemci o nainštalovanie „Express“ produktov, ktorí nedisponujú rýchlym pripojením na web sa možno spočiatku potešia tomu, že inštalačné súbory všetkých týchto produktov majú len okolo 2.5 megabajtov. Zatiaľ čo však inštalačný súbor o veľkosti približne 1.5 MB u Web Matrixu obsahoval prakticky celú inštaláciu (predinštalovať bolo potrebné len technologickú platformu .NET Framework), u „Express“ produktov tieto malé súbory len inicializujú inštaláciu a zvyšok (pri typickej inštalácii asi 240 MB) sa stiahne z webu počas procesu inštalácie. Inštalačný proces je po špecifikovaní súčastí, ktoré si želáme nainštalovať úplne v réžii sprievodcu inštaláciou.



Informačný dialóg beta verzie Visual Web Developera

Popis pracovnej obrazovky Visual Web Developera

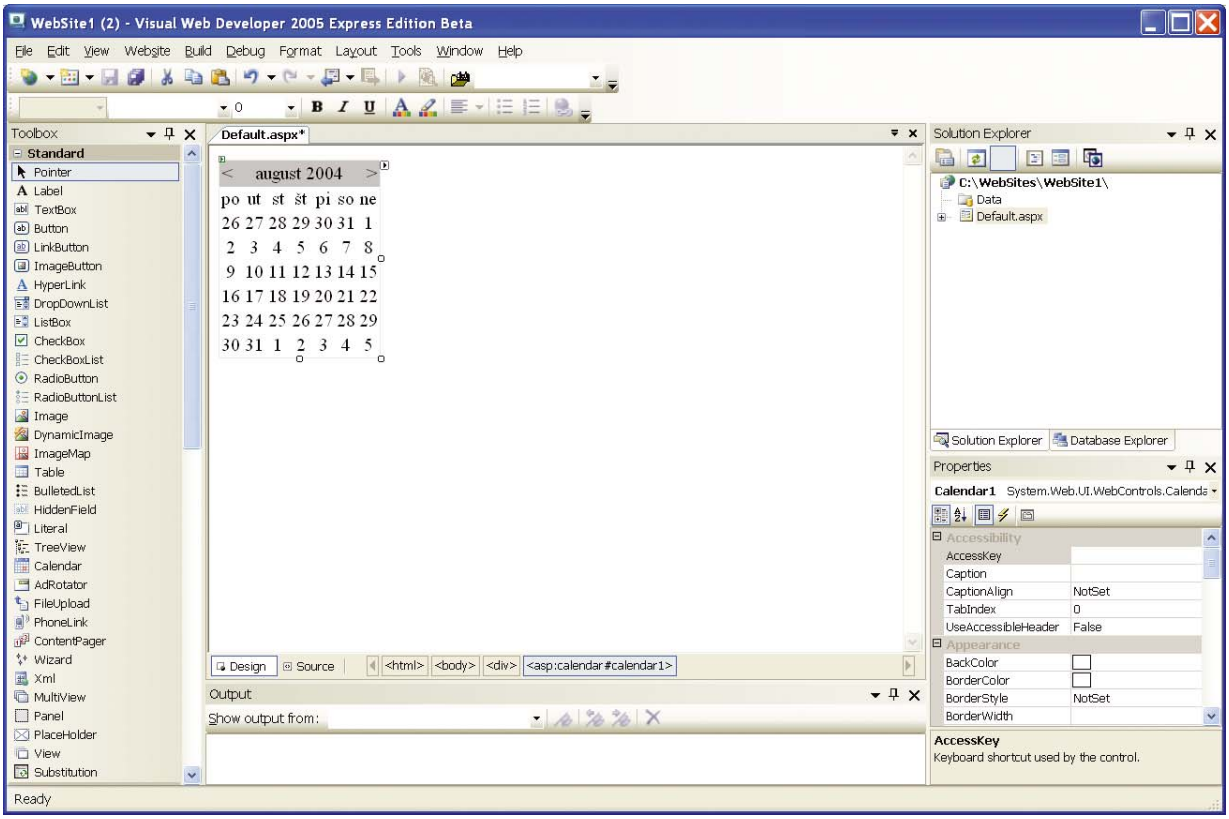
Základná konfigurácia pracovnej obrazovky vývojového prostredia zostala v porovnaní s plnou verziou Visual Studia či už 2005 alebo 2003 prakticky nezmenená. Osvedčené veci predsa meniť netreba. Dizajn pracovných vývojových prostredí všetkých firiem sa totiž postupom času vyprofiloval a ustálil, preto ani migrácia z iného nástroja nie je príliš problematická.

Pracovná obrazovka Visual Web Developera je rozdelená na panely

- Toolbox
- Solution Explorer
- Properties
- Output

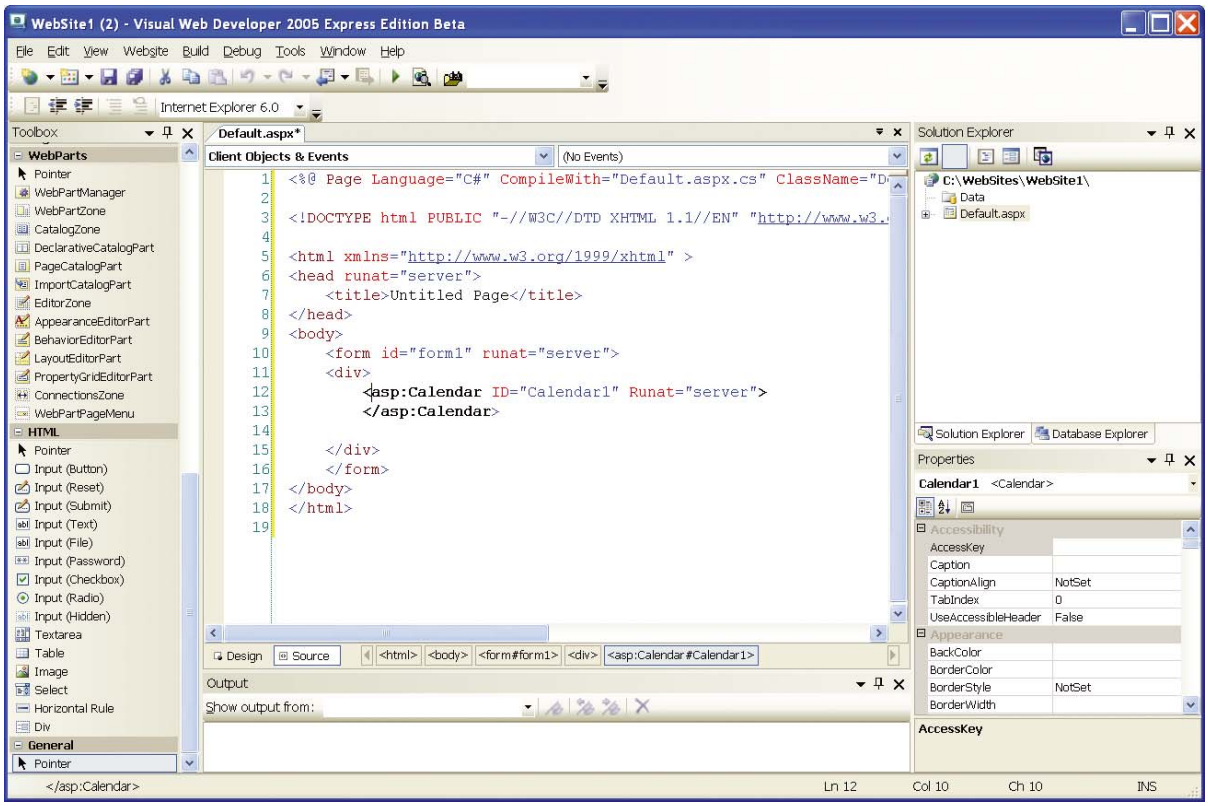
Stredné, najväčšie okno je možné prepínať medzi návrhovým zobrazením s možnosťou editovania pomocou grafického návrhového prostredia a editovaním zdrojového kódu v textovom móde

Možnosť prepínania stredného okna

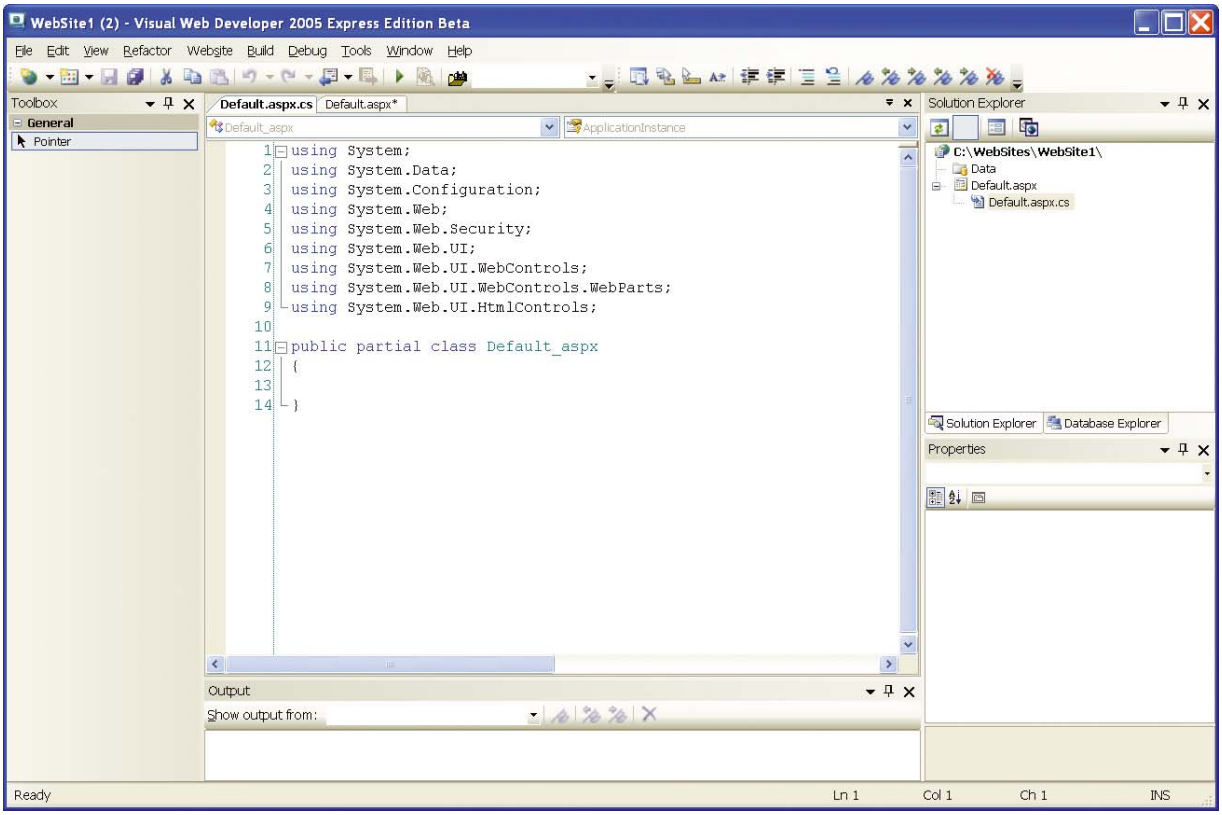


Pracovná obrazovka Visual Web Developera, v strednom okne je návrhové zobrazenie

Každý z týchto panelov sa obvykle skladá z niektorých záložiek, ktoré menia jeho aktuálny význam, čím je veľmi efektívne využitá zobrazovacia plocha monitora. V ľavej časti pracovnej obrazovky sa nachádza okno Toolbox. Vpravo sú okná Solution Explorer, Database Explorer (prepínajú sa pomocou záložiek v jednom okne) a Properties.

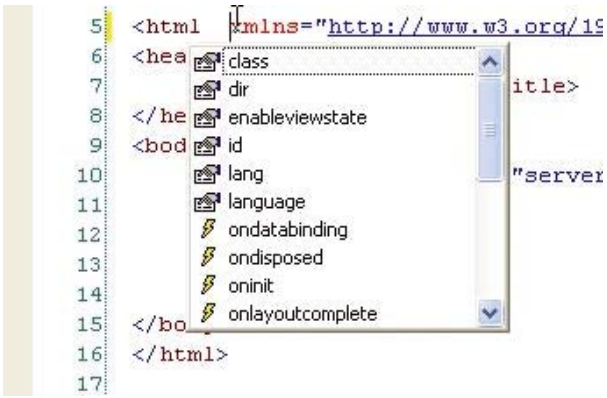


Pracovná obrazovka Visual Web Developera, v strednom okne je zdrojový kód ASP.NET stránky



Pracovná obrazovka Visual Web Developera, v strednom okne je zdrojový kód v jazyku C#.

Aj pri písaní a editovaní zdrojového kódu v textovom režime nám vývojové prostredie poskytuje cenné služby v podobe IntelliSense. Táto nápoveda funguje nielen pre kľúčové slová a objekty programovacích jazykov, ale aj pre HTML dokumenty, ASP.NET tagy a podobne



Interaktívna nápoveda IntelliSense

Programovacie jazyky

Vývojové prostredie Visual Studio .NET vo svojej prvej verzii prinieslo programovací jazyk C#. Vo verzii Whidbey (rovnako ako vo verzii 2003) sú k dispozícii štyri programovacie jazyky. Princíp výberu vhodného programovacieho jazyka pre ten ktorý typ aplikácie je otázkou jeho možností a skúseností vývojára. Pre každý programovací jazyk je potom akceptovaná trochu iná filozofia používateľského prostredia. Azda najstručnejšie informatívne doporučenie by mohlo vyzerať nasledovne:

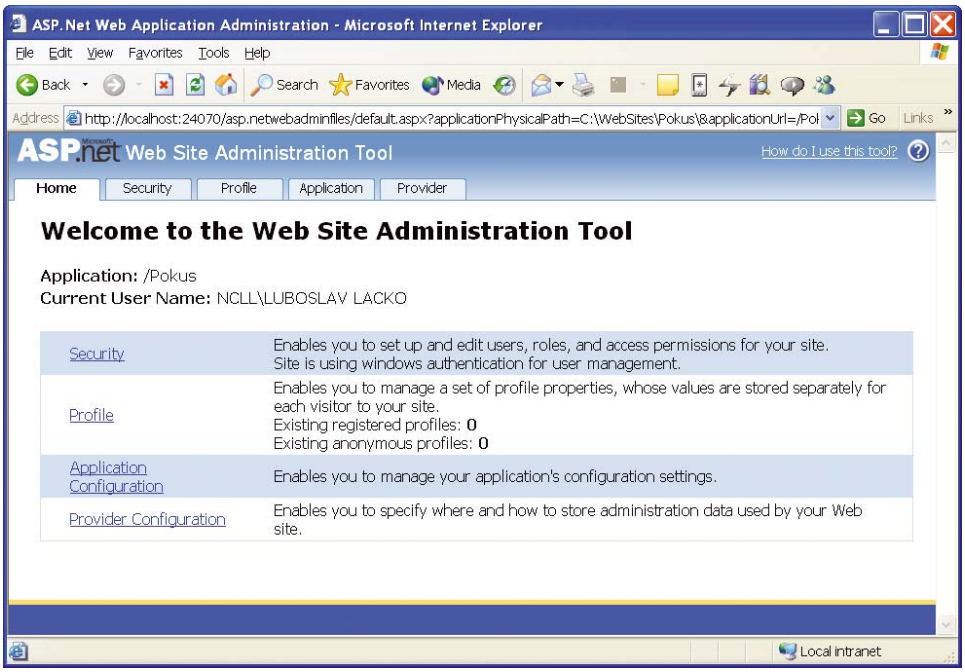
C# – nový moderný výkonný programovací jazyk, ktorý rešpektuje črty ako typovú bezpečnosť, „garbage collection“, teda upratanie nepotrebných objektov z pamäti, ľahký návrh aplikačných formulárov a podobne. Vo verzii „Whidbey“ pribudla podpora generického kódu, anonymné metódy, iterátory (kľúčové slovo Yield), čiastočné typy (definícia vo viacerých súboroch), statické triedy obsahujúce len statické členy a aliasy pre menné priestory (namespaces)

Visual Basic.NET – jednoduchosť návrhu predurčuje tento programovací jazyk pre jednoduché aplikácie, u ktorých je rozhodujúce, aby boli rýchlo hotové, napríklad rôzne testovacie aplikácie, aplikácie určené pre jednorazové použitie (vyhodnotenie nejakej špecifickej akcie) a podobne.

J# – umožňuje vývojárom použitie Javy v .NET prostredí

Administrátorský nástroj pre ASP.NET aplikáciu

Vo verzii ASP.NET 2.0 je k dispozícii aj administrátorský nástroj pre vyvíjanú aplikáciu. Aktivujeme ho v konkrétnom projekte prostredníctvom menu Website – ASP.NET Configuration.



Administrátorský nástroj pre ASP.NET aplikáciu

Administrátorský nástroj je webová aplikácia, pričom jednotlivé služby a funkcie sú prehľadne usporiadané do štyroch záložiek

Security – záložka integruje funkcie pre vytváranie a editovanie rolí, používateľov a prístupových práv k aplikácii. Môžeme nastaviť prístup len z lokálnej siete, alebo z celého webu. Výhodou hlavne pre začínajúcich administrátorov je možnosť využiť sprievodcu zabezpečením aplikácie, kde sú jednotlivé administrátorské úkony rozdelené do siedmych na seba nadcäzujúcich krokov, takže ak sa necháme viesť sprievodcom máme istotu, že na nič dôležité pri nastavovaní zabezpečenia aplikácie nezabudneme.

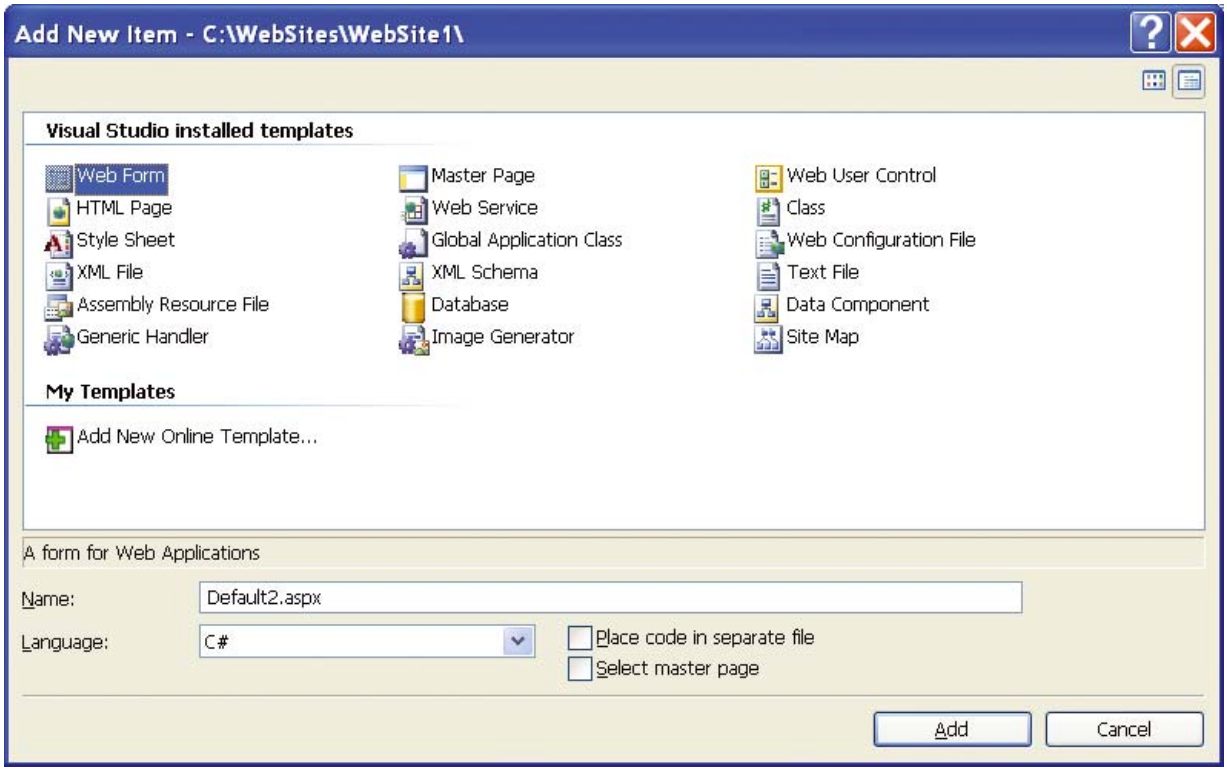
Profile – záložka slúži pre nastavenie profilov, ktoré môžu byť uložené separátne pre každého návštevníka vyvíjanej aplikácie. Význam to má hlavne pri podnikových portáloch.

Application Configuration – v tejto záložke je možné nastaviť parametre pre ladenie aplikácie a v prvej Beta verzii aj nakonfigurovať počítadlo a záznam štatistiky o návštevnosti jednotlivých stránok.

Provider Configuration – je azda najdôležitejšia záložka pre nastavenie spôsobu ukladania údajov, ktoré slúžia pre administráciu webovej aplikácie. Implicitne sa bude predpokladať použitie databázového servera SQL Server 2005 Express. Ak využívame iný databázový server, samozrejme môžeme nastaviť ukladanie konfiguračných a administrátorských údajov do tejto databázy.

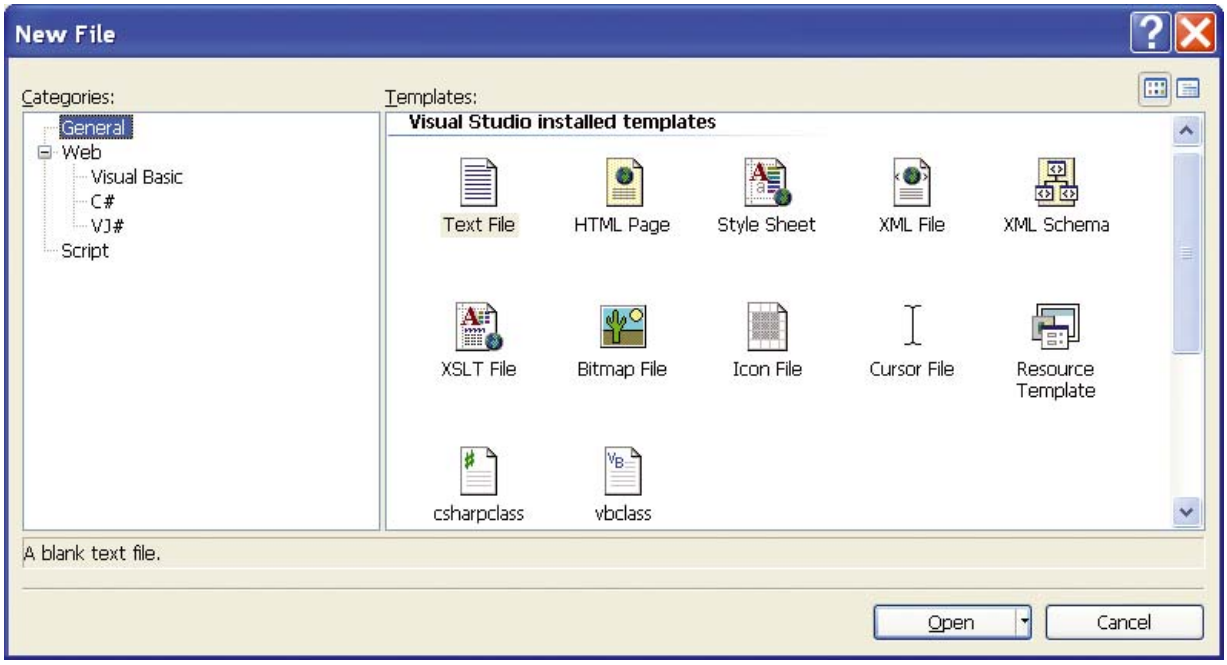
Vytváranie a editovanie súborov

Pomocou Visual Web Developera môžeme vytvárať a editovať pomerne veľa rôznych druhov súborov a dokumentov, ktoré sa nejakým spôsobom viažu k vývoju webových aplikácií, napríklad HTML stránky, webové formuláre, XML dokumenty, schémy a šablóny, zdrojové kódy, konfiguračné súbory a mnohé ďalšie.

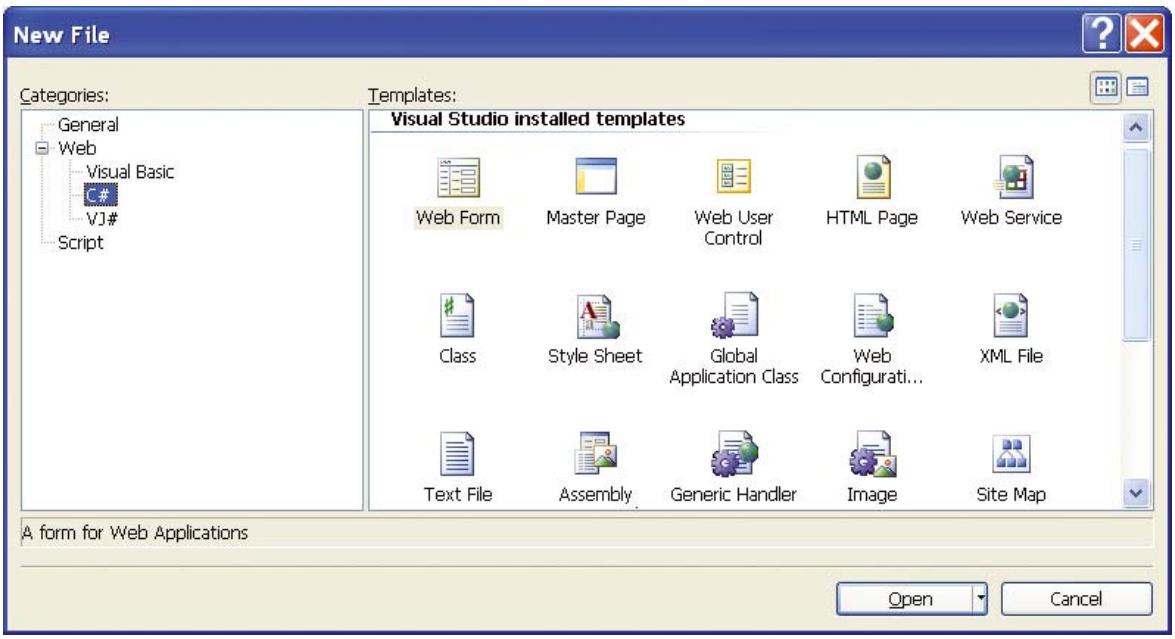


Typy súborov pre pridanie do projektu

Ikony pre jednotlivé typy dokumentov sú prehľadne usporiadané do zložiek, pričom niektoré dokumenty pre programovacie jazyky Visual Basic, C#, a J# sú rovnaké alebo veľmi podobné



Typy súborov v zložke General



Typy súborov v zložke C#

Pre ilustráciu ukážeme niektoré typy prázdnych dokumentov vytvorených pomocou funkcie New File.

Web Form – pomocou tejto voľby vytvoríme súbor s príponou **.aspx**. Súbor obsahuje direktívu @Page directive, HTML tagy a server-side <form>.

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<script runat="server">

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

    </div>
    </form>
</body>
</html>
```

HTML Page – pomocou tejto voľby vytvoríme súbor s príponou **.htm**. Súbor obsahuje otváracie a zatváracie tagy.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>Untitled Page</title>
</head><body>
</body></html>
```

XML File – pomocou tejto voľby vytvoríme súbor s príponou .xml.

```
<?xml version="1.0" encoding="utf-8" ?>
```

XML Web Service – pomocou tejto voľby vytvoríme súbor webovej služby s príponou .asmx v požadovanom programovacom jazyku. V komentári je aj príklad najjednoduchšieho kódu webovej služby.

```
<%@ WebService Language="C#" Class="WebService1" %>

using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;

public class WebService1

{
    public WebService1()
    {
        //
        // TODO: Add any constructor code required
        //
    }
    // WEB SERVICE EXAMPLE
    // The HelloWorld() example service returns the string Hello World.
    // To test, uncomment the following lines then save and run the ASMX file.
    // To run the ASMX file, select it in the solution explorer and press F5.
'

//    [WebMethod]
//    public string HelloWorld()
//    {
//        return "Hello World";
//    }
}
```

Global.asax – pomocou tejto voľby vytvoríme globálny ASP.NET súbor s príponou .asax

```
<%@ Application Language="C#" %>

<script runat="server">

    void Application_Start(Object sender, EventArgs e) {
        // Code that runs on application startup

    }

    void Application_End(Object sender, EventArgs e) {
        //  Code that runs on application shutdown

    }
```

```
    void Application_Error(Object sender, EventArgs e) {
     // Code that runs when an unhandled error occurs

    }

    void Session_Start(Object sender, EventArgs e) {
        // Code that runs when a new session is started

    }

    void Session_End(Object sender, EventArgs e) {
        // Code that runs when a session ends.
        // Note: The Session_End event is raised only when the sessionstate mode
        // is set to InProc in the Web.config file. If session mode is set to
StateServer
        // or SQLServer, the event is not raised.

    }

</script>
```

Web.Config – pomocou tejto voľby vytvoríme súbor s názvom *web.config*, ktorý obsahuje sekciu <configuration>, <appSettings>, <connectionStrings> a <system.web>. Súbor je vo formáte XML pričom jednotlivé sekcie a elementy sú okomentované

```
<?xml version="1.0" ?>

<!-- Note: As an alternative to hand editing this file you can use the web admin
tool to
       configure settings for your application. Use the Website->Asp.Net
Configuration option
       in Visual Studio.
     A full list of settings and comments can be found in machine.config.comments
usually
     located in \Windows\Microsft.Net\Frameworks\v2.x\Config -->

<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">

    <appSettings />
    <connectionStrings />

    <system.web>

        <!--
            Set compilation debug="true" to insert debugging symbols into the
compiled page.
            Because this affects performance, set this value to true only during
development.
        -->
        <compilation debug="false" />


        <!--
            The <authentication> section enables configuration of the security
authentication
            mode used by ASP.NET to identify an incoming user.
        -->
```

```
        <authentication mode="Windows" />


        <!--
            The <customErrors> section enables configuration of what to do
if/when an unhandled
            error occurs during the execution of a request.  Specifically, it
enables developers
            to configure html error pages to be displayed in place of a error
stack trace.
        -->
        <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
        <!--
            <error statusCode="403" redirect="NoAccess.htm"/>
            <error statusCode="404" redirect="FileNotFound.htm"/>
        -->
        </customErrors>

    </system.web>
</configuration>
```

Niektoré dokumenty, ako napríklad XML Schema a niektoré dokumenty typu Resource sa zasa navrhujú pomocou grafického návrhového prostredia.

Možnosti migrácie – Visual Studio.NET 2005

Hoci primárne je táto publikácia určená vývojárom „hobby“ aplikácií, niekedy sa stane, že naše hobby alebo pokusná webová aplikácia, napríklad zásielková služba a podobne prerastie do seriózneho biznisu a vtedy je potrebné webovú aplikáciu škálovať, prípadne migrovať do výkonnejšieho serverového prostredia. Pre vývoj rozsiahlejších, zložitejších a robustnejších ASP.NET aplikácií (a .NET aplikácií vo všeobecnosti) sa používa vývojové prostredie Visual Studio .NET, pričom jeho najnovšia verzia má označenie 2005.

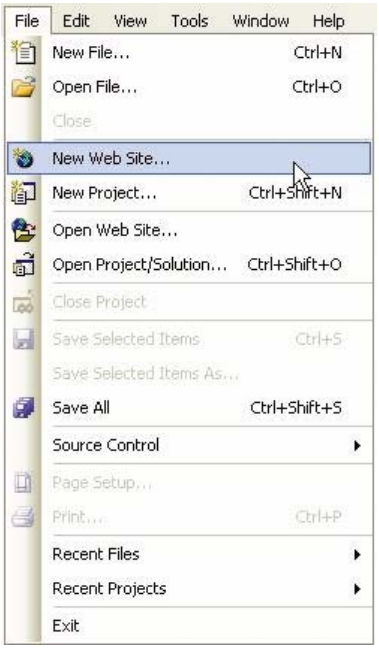
Kapitola 3

Vývoj ASP.NET aplikácií

Vývoj internetových a intranetových projektov je na jednej strane veľmi dynamicky sa rozvíjajúca oblasť, na druhej strane sa internetové a intranetové aplikácie na seba dosť podobajú. Už pri letmom pohľade na tieto aplikácie môžeme identifikovať ich „typ“. Buď sa jedná o internetový alebo intranetový publikačný a informačný portál, aplikácie pre elektronické obchodovanie, podnikové aplikácie a podobne. Aj vzhľad a ovládanie týchto aplikácií sa časom ustálili, čo je výhodné aj pre vývojára ale hlavne pre orientáciu klienta, prípadne zamestnanca.

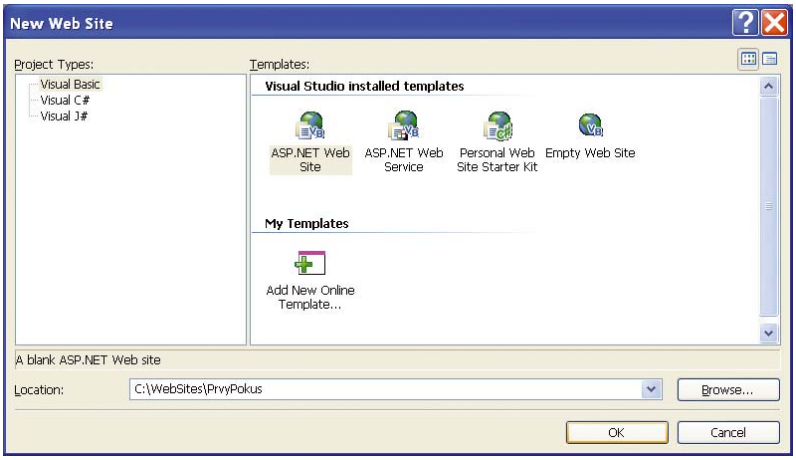
Prvá ASP.NET aplikácia

Kým sa však dostaneme k zložitejším aplikáciám, vytvoríme prvú cvičnú aplikáciu, od ktorej nebudeme požadovať žiadne závratné služby, jej hlavnou úlohou bude overenie správnej inštalácie a konfigurácie vývojového prostredia, vrátane možnosti spustenia a ladenia aplikácie, pričom sa „za pochodu“ zoznámime zo základnými ovládacími prvkami Visual Web Developera. V ponuke pre vytvorenie novej webovej aplikácie je nielen ASP.NET stránka, webová služba, prázdna stránka ale aj komplexný projekt pre vytvorenie osobnej webovej stránky – Personal Web Site Starter Kit.



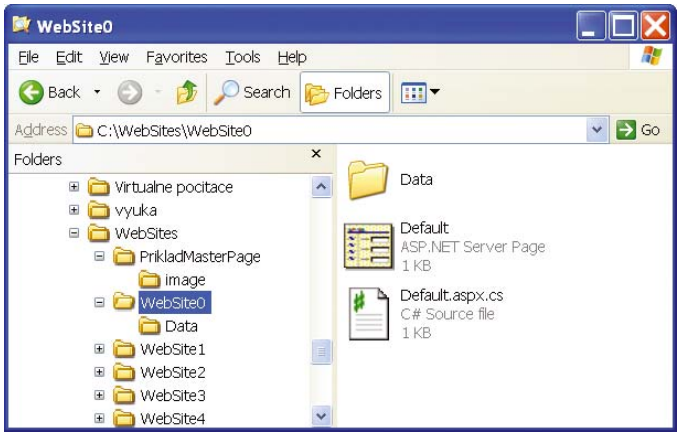
Menu FILE pre manipuláciu so súbormi a projektami

Po vytvorení nového projektu typu ASP.NET Web Site v svojom obľúbenom jazyku (Visual Basic, Visual C# alebo Visual J#) je výhodné v okne Solution Explorer vytvoriť nový podadresár s názvom Code. Vývojové prostredie takýto podadresár rozpoznáva a umiestňuje do neho kód, ktorý je oddelený (code behind) od návrhu formulárov a ostatných designových prvkov na ASP.NET stránkach. Aby bol prvý príklad čo najjednoduchší, zvolili sme programovací jazyk Visual Basic. Projekt bol nazvaný PrvyPokus.



Ponúkané typy projektov pre jednotlivé programovacie jazyky

Po vytvorení nového projektu sa zoznámime podrobnejšie, čo vlastne pojem projekt znamená. Najskôr si pozrime fyzický adresár v ktorom bol projekt vytvorený. Projekt bude implicitne fyzicky umiestnený do adresára WebSites a podadresára zhodného s názvom projektu. Tento podadresár je tvorený zložkou Data a súbormi default.aspx, ktorý obsahuje návrh ASP.NET stránky a súbor default.aspx.cs so separátne oddeleným kódom v jazyku C# (alebo default.aspx.vb s kódom v jazyku Visual Basic). Logicky je projekt prístupný vo virtuálnom adresári s identickým názvom ako je názov projektu.



Fyzický adresár obsahujúci súbory tvoriace projekt

Aby sme v návrhovom okne nemali čistú bielu plochu, umiestnili sme tam nejaké texty a prvok Label. Sprievodca vytvorením novej aplikácie vytvoril kód ASP.NET stránky a jadro zdrojového programu v jazyku Visual Basic (alebo C#, podľa toho aký jazyk sme pre projekt zvolili)

Kód ASP.NET stránky bude

```
<%@ Page Language="VB" AutoEventWireup="false" CompileWith="Default.aspx.vb"
ClassName="Default_aspx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1>
            Prvý projekt<br />
        </h1>
        Dnešný dátum je:
        <asp:Label ID="Label1" Runat="server" Text="Label" Width="168px"
Height="22px"></asp:Label>

    </div>
    </form>
</body>
</html>>
```

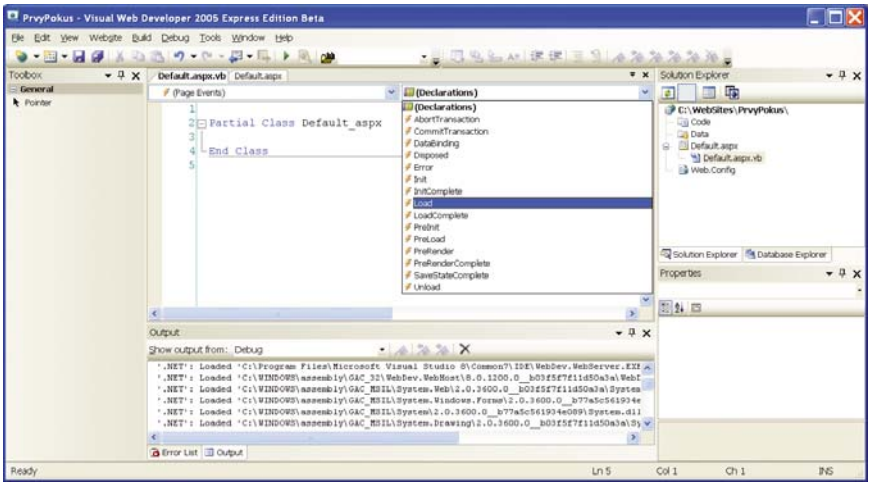
Po ukončení etapy návrhu základných aplikačných formulárov a časti aplikačnej logiky nastane onen „osudový“ okamih, kedy ideme s kožou na trh a začíname aplikáciu ladiť. Pre tento účel bude aktivovaný samostatný webový server, ktorý je integrálnou súčasťou vývojového prostredia

Visual Web Developer Web Server

Prvé spustenie projektu

Z hľadiska formálneho, teda správnosť kódu, fungovanie reťazca webový server, ASP.NET engine, klientský prehliadač je všetko v poriadku. Aplikácia však nerobí s výnimkou výpisu statického textu vôbec nič (to predsa aplikácie typu Hello World bežne robia), no my sme si uplietli pastičku sami na seba. Stránka sa podľa nadpisov tvári, že by mala vypísať aktuálny dátum a čas. Zatiaľ však len vypisuje statický implicitný text komponeny „Label“. Preto tento projekt doplníme o zobrazenie dátumu a času. Zistenie aktuálnych hodnôt dátumu a času urobíme v okamihu načítania stránky, teda ako obsluhu udalosti Page_Load.

Ak v návrhovom okne klikneme na komponentu Label, prepneme sa do zobrazenia zdrojového kódu. Nad oknom zdrojovky sú dva combo boxy. V prvom z nich nastavíme Page Events (udalosti stránky) a v druhom boxe nastavíme udalosť Load

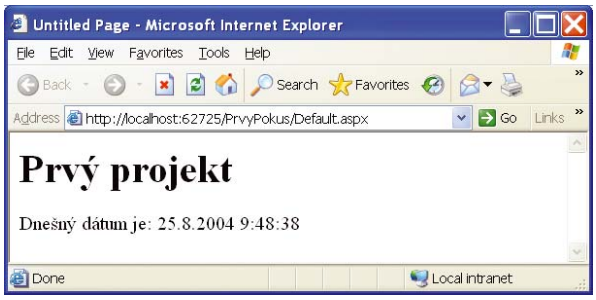


Výber typu udalosti, ktorú chceme obslúžiť

Vytvorí sa obslužná procedúra **`Private Sub Page_Load`**. Do tela tejto procedúry potom vložíme kód pre výpis dátumu a času

Zdrojový kód v jazyku Visual Basic

```
Partial Class Default_aspx

  Private Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load

    Label1.Text = DateTime.Now.ToString

    End Sub
End Class
```



Spustenie projektu

Webové formuláre

Okrem aplikačnej logiky budeme samozrejme potrebovať aj rôzne ovládacie prvky, napríklad pre zadávanie údajov a podobne. Na platforme ASP.NET sú k dispozícii všetky druhy ovládacích prvkov, ktoré poznáme z klasických HTML formulárov, ale aj z desktopových aplikácií napísaných vo vyšších programovacích jazykoch. Pretože jednotlivé komponenty formulárov sú vlastne základnými stavebnými kameňmi webových stránok, budeme im v tejto publikácii venovať väčšiu pozornosť.

Serverové ovládacie prvky

Jedná sa o ovládacie prvky, ktoré dôverne poznáme z HTML stránok, hlavne z rôznych formulárov. Pridaním klauzuly **runat=server** však zabezpečíme, že sa kód vykoná na strane servera a následne sa vyrendruje vizuálna podoba komponenty pre klienta. Najskôr ukážeme pravdepodobne najjednoduchší kód pre výpis naformátovaného textu, pričom použijeme jednu z najjednoduchších komponent – Label.

```
<%@ Page Language="C#" CompileWith="Default.aspx.cs" ClassName="Default_aspx" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" Runat="server" Text="Label" Width="134px"
Height="22px"></asp:Label>

    </div>
    </form>
</body>
</html>
```

Zatiaľ sme sa k žiadnej výraznej novej funkcionalite nedopracovali. Ak porovnáme zdrojový kód a serverom vygenerovanú HTML stránku u klienta

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head><title>
      Untitled Page
</title></head>
<body>
    <form method="post" action="Default.aspx" id="form1">
<div>
<input type="hidden" name="__VIEWSTATE"
value="/wEPDwUJODExMDE5NzY5ZGTSdUxXC2QPi6n9bTg==" />
</div>

    <div>
        <span id="Label1" style="height:22px;width:134px;">Label</span>

    </div>
    </form>
</body>
</html>
```

je to takmer to isté. Na tomto jednoduchom príklade sa to neprejaví, ale vo všeobecnosti nám serverové prvky umožňujú programovú manipuláciu s elementami HTML. Je to vlastne objekt (z hľadiska terminológie objektovo orientovaného programovania), inými slovami povedané - každý prvok má určité vlastnosti metódy a udalosti. Používať na príklady tohoto rozsahu vývojové prostredie snáď ani nie je potrebné, no pre napísanie takejto aplikácie vo Visual Web Developeri by sme museli urobiť jeden jediný úkon, presunúť ikonku Web komponenty Label na pracovnú plochu aplikácie a zmeniť text.

Serverové ovládacie prvky sú v toolbare rozdelené do logických skupín
- Standard
- Data
- Validation
- Navigation
- Login
- WebParts
- HTML

Prvky toolbaru pre vytváranie webovych formulárov

Ovládacie prvky zo skupiny Standard

Label – Prvok Label umožní zobrazenie textového výpisu, napríklad hodnoty nejakej textovej premennej a podobne.

TextBox – Jednoduché alebo viacriadkové editačné okno pre zadávanie textových údajov. Zmenu textu, teda napísanie alebo vymazanie niektorého znaku môžeme monitorovať pomocou metódy OnTextChanged.

Button – Klasické tlačidlo, ktoré poznáme z webových stránok aj z desktopových aplikácií, po zatlačení ktorého môžeme realizovať rôzne akcie.

ListBox – Pomocou tejto komponenty môžeme realizovať výber jednej alebo viacerých položiek z ponúkaných možností. Aj túto komponentu môžeme napojiť na vhodný stĺpec databázovej tabuľky.

CheckBox – Tento jednoduchý ovládací prvok umožňuje prepínať medzi dvomi hodnotami pravda - nepravda (TRUE-FALSE)

CheckBoxList – Ovládací prvok umožní výber jednej alebo viacerých možností z ponúkaného zoznamu. Výber sa vykonáva zaškrtnutím políčka vedľa textu. Komponentu je možné naviazať na databázovú tabuľku

RadioButton – Ovládací prvok RadioButton sám o sebe veľký zmysel nemá, je potrebné použiť týchto prvkov niekoľko, potom fungujú ako prepínač (odtiaľ analógia RadioButton - tlačidlová sada pre prepínanie rozsahov rozhlasového prijímača)

RadioButtonList – Ako vyplýva z názvu, tento ovládací prvok funguje podobne ako prepínacie tlačidlá vlnových rozsahov na starších rádioprijímačoch. Môžeme ho naplniť položkami priamo v kóde, alebo naviazať na databázovú tabuľku

Image – Ovládací prvok Image slúži pre zobrazenie obrázku

Literal – Táto komponenta sa používa pre rendrovanie statického textu na webovej stránke.
DropDownList Prvok umožňuje zoznam možností, ktorý je možné rozvinúť a vybrať jednu z možností. Možnosti môžeme definovať jednak priamo v kóde, prípadne môžeme DropDownList napojiť na vhodný stĺpec databázovej tabuľky.

ImageButton – Na rozdiel od prvku Image, prvok ImageButton umožňuje obslúžiť udalosť kliknutia.

AdRotator – Typickým príkladom použitia ovládacieho prvku AdRotator je rotácia reklamných bannerov.

Calendar – Okrem jednoduchých prvkov formulára pre zadávanie textu, alebo pre zapnutie, vypnutie, alebo prepnutie nejakých volieb veľmi často potrebujeme zadávať kalendárne údaje. Nie je problém požadovať zadanie týchto údajov v klasickom editačnom okne formulára, no jednak tým utrpí presnosť zadania (ako uvidíme neskôr, toto by sme pomerne pracne mohli ošetriť pomocou validátorov), no hlavne utrpí používateľský komfort. A to už nehovoríme o rozličných formátoch dátumu a času používaných v anglosaských krajinách a u nás, o oddeľovačoch dní a podobne. Ovládací prvok Calendar zobrazí mesačný kalendár, pomocou ktorého môže klient jednoducho a interaktívne zadať požadovaný dátum, alebo špecifikovať nejaké časové obdobie.

Table – Veľmi používaným zobrazovacím prvkom na webových stránkach sú tabuľky. Komponenta Table umožňuje programovo vytvárať rôzne tabuľky a manipulovať s nimi pomocou ovládacieho prvku Table. S týmto ovládacím prvkom úzko súvisia ďalšie dva ovládacie prvky TableRow a TableCell pre vytvorenie a manipuláciu riadkov a buniek tabuľky

Ovládacie prvky sa veľmi často zoskupujú do formulárov. Formulár sa zobrazí používateľovi na HTML stránke. Po jeho vyplnení používateľom zadané údaje odošleme na server a následne spracujeme.

Príklad formulára realizovaného pomocou ASP.NET

Pre riešenie tej istej úlohy pomocou ASP.NET potrebujeme len jednu stránku (súbor **formular1.aspx**). Táto stránka obsahuje jednak HTML kód formulára, jednak kód pre spracovanie používateľom zadaných údajov

```
<html>
     <script language="C#" runat="server">
          void Kliknutie(Object Src, EventArgs E)
          {
               Message.Text = "Nazdar " + Name.Text;
          }
     </script>

     <body>
        <h1>Príklad formulára</font></h1>
        <form action="controls3.aspx" runat=server>
           Vase meno: <asp:textbox id="Name" runat=server/>
                      <asp:button text="Potvrď"
                       Onclick=" Kliknutie " runat=server/>
                      <asp:label id="Message" runat=server/>
        </form>
     </body>
</html>
```

Najskôr sa presvedčíme, či to funguje, a potom sa budeme venovať vysvetleniu princípu.

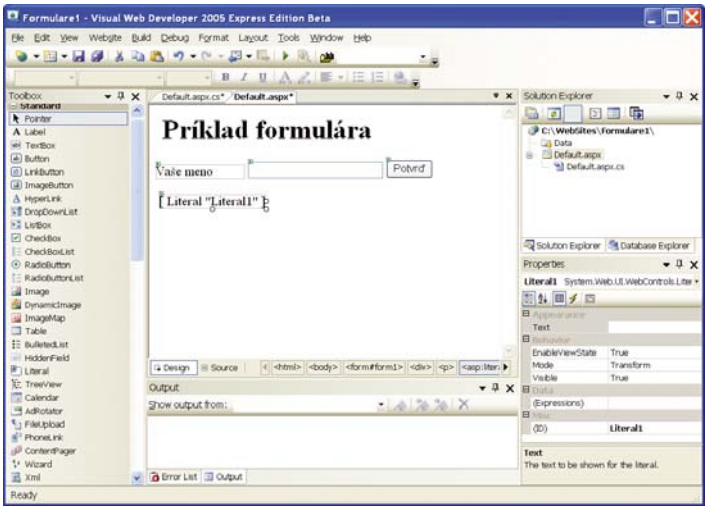
Formulár

Veľmi nám pomôže, ak si zobrazíme zdrojový kód stránky, ktorá bude vygenerovaná pre klientov prehliadač.

```
<html> <body>
        <h1>Príklad formulára</font></h1>
        <form name="_ctl0" method="post" action="formular1.aspx" id="_ctl0">
<input type="hidden" name="__VIEWSTATE"
value="dDwxMzc4MDMwNTk1O3Q8O2w8aTwyPjs+O2w8dDw7bDxpPDU+Oz47bDx0PHA8cDxsPFRleHQ7Pjt
sPE5hemRhciBNaWxhbjs+Pjs+Ozs+Oz4+Oz4+Oz5IITb0PWSI79ZAVATTIYfkzl7YDw==" />
           Vase meno: <input name="Name" type="text" value="Milan" id="Name" />
                      <input type="submit" name="_ctl1" value="Potvrď" />
                      <span id="Message">Nazdar Milan</span>
        </form>
</body></html>
```

V prvom rade je potrebné vyzdvihnúť, že pre klienta bola vygenerovaná úplne štandardná HTML stránka bez akýchkoľvek prvkov, ktoré by mohli spôsobovať potenciálnu nekompatibilitu pri jej zobrazení v rôznych prehliadačoch. Nie sú tu nijaké ActiveX komponenty, Java applety, len čistý HTML kód, s ktorým si rovnako dobre poradí akýkoľvek prehliadač webových stránok. Všetky informácie o druhu a stave komponenty budú serveru odovzdané pomocou skrytého parametra formulára s názvom __VIEWSTATE.

Už pre takýto jednoduchý formulár je výhodnejšie použiť vývojové prostredie Visual Web Developer. V okne Design ho vytvoríme pomocou Web komponentov Label, Text Box, Button a Literal. Nadpis bol vpísaný priamo do kontextu HTML stránky a naformátovaný ako <h1>.



Návrh formulára vo vývojovom prostredí Visual Web Developer

Po niekoľkých jednoduchých presunoch komponentov pomocou kurzoru myši vznikol prakticky totožný kód, z toho vyplýva, že vývojové prostredie pracuje pomerne efektívne.

```
<%@ Page Language="C#" CompileWith="Default.aspx.cs" ClassName="Default_aspx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1>
             Príklad formulára </h1>
        <p>
            <asp:Label ID="Label1" Runat="server" Height="22px" Width="134px"
Text="Vaše meno"></asp:Label>
            <asp:TextBox ID="TextBox1" Runat="server"></asp:TextBox>
            <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď" />
        </p>

    </div>
    </form>
</body>
</html>
```

Takto vytvorený kód zobrazí formulár, no zatiaľ neobsahuje kód pre spracovanie používateľom zadaných hodnôt.

V tomto okamihu by bolo ešte zbytočné pustiť sa do písania jadra procedúry, ktorou ošetrujeme udalosť zatlačenia tlačidla. Vývojové prostredie nám totiž dokáže vygenerovať aj jej kostru. Stačí dvojklik na tlačidlo v návrhovom formulári.

```
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Default_aspx
{
     void btPotvrd_Click(object sender, EventArgs e)
     {

     }

}
```

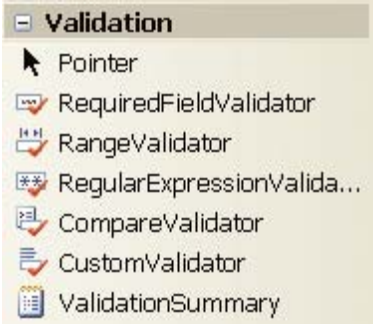
Telo procedúry už napíšeme sami. Zadaný text z komponenty TextBox vypíšeme do komponenty Literal

```
// Insert page code here
     //

     void Button1_Click(Object sender, EventArgs e) {
       Literal1.Text = "Nazdar " + Name.Text;
     }
```

Ovládacie prvky zo skupiny Validation – ASP.NET validátory

Na formuláre na webových stránkach sú kladené určité požiadavky, hlavne vzhľadom na komfort používateľov. Hlavne v aplikáciách typu e-business je dôležité, aby údaje zadané používateľom (údaje, rodné číslo, číslo objednávky, číslo kreditnej karty....) boli zadané správne. Od bežného klienta, napríklad návštevníka internetového obchodu samozrejme nemôžeme chcieť, aby poznal všetky záludnosti našej aplikácie. Pre vývojára je „samozrejmé“ že sa bude používať desatinná bodka, matematicky „odchovaný“ klient neprogramátor tam celkom pochopiteľne zadá čiarku, problémy sú zo zadávaním formátu dátumu a času a podobne. Preto musíme používateľovi pomôcť nielen vhodným a jednoznačným návrhom formulára pre zadávanie údajov, ale aj zadávané údaje kontrolovať a používateľa usmerňovať tak, aby výsledkom jeho snaženia bola úspešná transakcia. Na vytvorenie interaktívnej stránky sme pred érou .NET mali k dispozícii skripty na strane servera a taktiež skripty na strane klienta. Teoreticky by stačilo použiť len skripty na strane servera, ale to by vyžadovalo neustálu interakciu medzi klientom a serverom pri vyplňovaní a kontrole každého vstupného poľa formulára. Skripty na strane klienta na túto úlohu vždy nestačia, pretože niekedy je potrebné kontrolovať údaje aj na duplicitu. Ideálnym riešením je kombinácia skriptov na strane klienta so skriptami na strane servera. V ASP.NET pre tento účel slúžia objekty zvané validátory.

Validátory je možné do aplikácie vložiť metódou drag and drop z Toolbox vývojového prostredia



Validátory dostupné cez Toolbox vývojového prostredia Visual Web Developer

Najskôr uvedieme prehľad validátorov vo forme tabuľky

Typ validácie	Validátor
Nutnosť zadať hodnotu	**RequiredFieldValidator**
Porovnanie zadanej hodnoty	**CompareValidator**
Kontrola rozsahu	**RangeValidator**
Kontrola vzorov v reťazcoch	**RegularExpressionValidator**
Používateľsky definovaný	**CustomValidator**
Sumarizácia viacerých validáorov	**ValidationSumary**

RequiredFieldValidator

Ešte predtým, než budeme kontrolovať, či používateľ zadal správne údaje, začneme tou najjednoduchšou a najtriviálnejšou úlohou, totiž či používateľ zadal vôbec nejaké údaje. Hodí sa to napríklad pri zadávaní mena, kde iné kontrolné kritériá pravdepodobne ani neprichádzajú do úvahy. Princíp činnosti je tohoto validátora je veľmi jednoduchý. RequiredFieldValidator kontroluje neprázdnosť zadávacieho poľa.

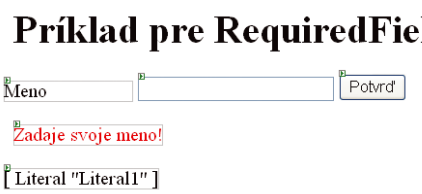
Definícia syntaxe

```
<asp:RequiredFieldValidator
      id="ID validátora"
      ControlToValidate=" ID prvku, ktorého obsah chceme kontrolova□"
      InitialValue="hodnota"
      ErrorMessage="Oznam, ktorý sa vypíše v prípade neplatnej validácie"
      Text="text prvku"
      ForeColor="hodnota"
      BackColor="hodnota" …
      runat="server" >
</asp:RequiredFieldValidator>
```

Príklad použitia

Vytvoríme najjednoduchší formulár pre zadanie mena. Pomocou RequiredFieldValidatora budeme kontrolovať, či používateľ do editačného okna napísal nejaký text, v tomto prípade meno

```
<form id="form1" runat="server">
    <div>
        <h1>
            Príklad pre RequiredFieldValidator </h1>
        <p>
            <asp:Label ID="Label1" Runat="server" Height="22px" Width="134px"
Text="Meno"></asp:Label>
            <asp:TextBox ID="TextBox1" Runat="server"></asp:TextBox>
            <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď"
OnClick="btPotvrd_Click" /> </p>
        <p>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
Runat="server" ErrorMessage="Zadajte svoje meno!" ControlToValidate="TextBox1">
            </asp:RequiredFieldValidator></p>
        <p>
            <asp:Literal ID="Literal1" Runat="server"></asp:Literal>
        </p>

  </div>
</form>
```



RequiredFieldValidator

CompareValidator

Tento validátor kontroluje zadanú hodnotu s referenčnou, prípadne s hodnotou iného poľa. Notoricky známym a v praxi veľmi často používaným príkladom je overovanie hesla pri jeho zadaní, či sme sa napríklad nedopustili preklepu, ktorý by nám mohol v budúcnosti veľmi znepríjemniť život. Všeobecný syntaktický predpis pre použitie tohoto validátora je nasledovný:

Definícia syntaxe

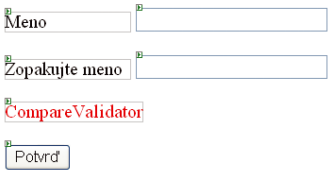
```
<asp:CompareValidator
      id="ID validátora"
      ControlToValidate="ID prvku, ktorého obsah chceme porovnáva"
      ValueToCompare="hodnota"
      ControlToCompare="hodnota"
      Type="dátový typ"
      Operator="hodnota operátora"
      ErrorMessage="Oznam, ktorý sa vypíše v prípade neplatnej validácie"
      Text="text prvku"
      ForeColor="hodnota"
      BackColor="hodnota" …
      runat="server" >
</asp:CompareValidator>
```

Príklad použitia

V tomto príklade vzájomne porovnávame dve zadané hodnoty, napríklad pri overovaní hesla.

```
<form id="form1" runat="server">
     <div>
          <h1>
               Príklad pre CompareValidator </h1>
          <p>
               <asp:Label ID="Label1" Runat="server" Height="22px" Width="134px"
Text="Meno"></asp:Label>
               <asp:TextBox ID="TextBox1" Runat="server"></asp:TextBox></p>
          <p><asp:Label ID="Label2" Runat="server" Height="22px" Width="134px"
Text="Zopakujte meno"></asp:Label>
               <asp:TextBox ID="TextBox2" Runat="server"></asp:TextBox><p>
               </p>
               <asp:CompareValidator ID="CompareValidator1" Runat="server"
ErrorMessage="CompareValidator"
                    ControlToValidate="TextBox2" ControlToCompare="TextBox1">
               </asp:CompareValidator>
          </p>
          <p>
               <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď"
OnClick="btPotvrd_Click" />
          </p>
          <p>
          </p>

   </div>
</form>
```

Príklad pre CompareValidator



CompareValidator

V tomto príklade sme ukázali porovnanie hodnôt dvoch prvkov medzi sebou. Častým prípadom bude určite porovnanie zadaného obsahu textového prvku s konštantou, alebo s obsahom premennej.

RangeValidator

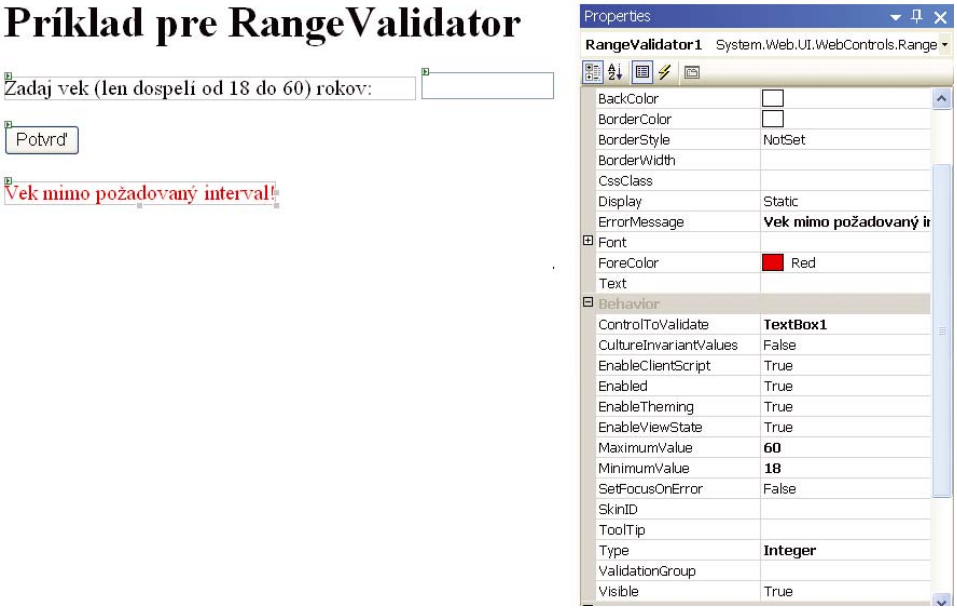
U číselných hodnôt veľmi často potrebujeme skontrolovať, či je zadaná hodnota v určitom číselnom intervale, ktorý je obvykle daný aplikačnou logikou.

Definícia syntaxe

```
<asp:RangeValidator
      id="ID alidatora"
      ControlToValidate=" ID prvku, ktorého rozsah chceme kontrolova "
      MinimumValue="hodnota"
      MaximumValue="hodnota"
      Type="dátový typ"
      ErrorMessage=" Oznam, ktorý sa vypíše v prípade neplatnej validácie "
      Text="text prvku"
      ForeColor="hodnota"
      BackColor="hodnota" …
      runat="server" >
</asp:RangeValidator>
```

Príklad použitia (validujeme čísla ako Integer v rozsahu od 18 do 60 rokov)

```
<form id="form1" runat="server">
    <div>
        <h1>
            Príklad pre RangeValidator </h1>
        <p>
            <asp:Label ID="Label1" Runat="server" Height="22px" Width="382px"
Text="Zadaj vek (len dospelí od 18 do 60) rokov: "></asp:Label>
            <asp:TextBox ID="TextBox1" Runat="server" Height="25px"
Width="123px"></asp:TextBox></p>
        <p>
            <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď"
OnClick="btPotvrd_Click" />
        </p>
        <p>
            <asp:RangeValidator ID="RangeValidator1" Runat="server"
ErrorMessage="Vek mimo požadovaný interval!"
                ControlToValidate="TextBox1" MaximumValue="60" MinimumValue="18"
Type="Integer">
            </asp:RangeValidator>
        </p>

  </div>
</form>
```



RangeValidator (vpravo je okno Properties)

RegularExpressionValidator

Ako vyplýva z názvu, tento validátor slúži pre overenie, či zadaná hodnota vyhovuje regulárnemu výrazu. Typickým príkladom regulárneho výrazu je napríklad mailová adresa.

Definícia syntaxe

```
<asp:RegularExpressionValidator
      id="ID validátora"
      ControlToValidate=" ID prvku, ktorého obsah chceme kontrolova□"
      ValidationExpression="výraz"
      ErrorMessage="Oznam, ktorý sa vypíše v prípade neplatnej validácie"
      Text="text prvku"
      ForeColor="hodnota"
      BackColor="hodnota" …
      runat="server">
</asp: RegularExpressionValidator>
```

Predpis pre overenie regulárneho výrazu sa vytvára pomocou kombinácie špeciálnych znakov. Ich kompletný zoznam je v dokumentácii. Vo forme tabuľky ukážeme niekoľko najpoužívanejších.

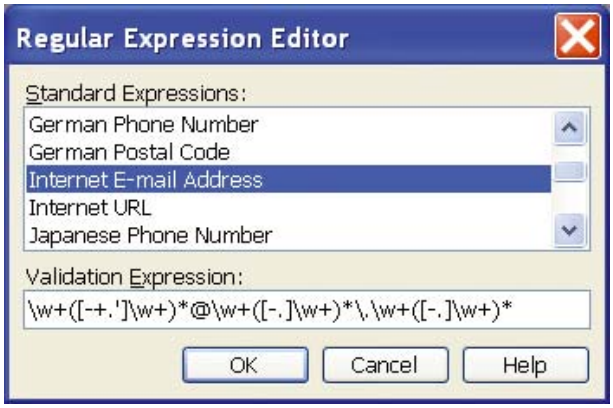
Znak	Popis
*	Žiadny znak, prípadne viac znakov.
+	Jeden alebo viac znakov.
\	Prefix pred špeciálnym znakom, napríklad '\n' znamená znak CR. Používa sa aj pre aplikáciu špeciálnych znakov ako bežných
.	Jeden ľubovoľný znak okrem CR.
\d	Číslica.
\D	Iný znak ako číslica
\t	Tabulátor.

Príklad použitia (kontrola používateľom zadanej mailovej adresy)
Kompletná tabuľka obsahuje niekoľko desiatok riadkov, no popis bez praktickej ukážky by nebol príliš účelný. Ukážeme to radšej na niekoľkých príkladoch. Predpis pre najjednoduchšiu kontrolu mailovej adresy bude: '.*@.*\..*', po preklade do ľudskej reči to znamená:

. jeden ľubovoľný znak
* ľubovoľný počet znakov (aj žiadne).
@ konkrétny znak, v našom prípade populárny zavináč
. jeden ľubovoľný znak
* ľubovoľný počet znakov (aj žiadne).
\. konkrétny znak, v tomto prípade bodka. Pretože bodka je aj špeciálny znak, je pred ňou lomítko
. jeden ľubovoľný znak
* ľubovoľný počet znakov (aj žiadne).

Reťazec teda musí začínať jedným, alebo viacerými znakmi, za nimi musí byť znak zavináč, pokračujeme ďalším minimálne jednoznakovým reťazcom za ktorým musí byť bodka. Mailová adresa končí ukončovacím, minimálne jednoznakovým reťazcom.

Tento najjednoduchší regulárny výraz `'.*@.*\..*'` ani zďaleka nevystihuje všetky varianty. Našťastie je tu iné riešenie. Pre najčastejšie sa vyskytujúce výrazy, napríklad mailovú a URL adresu, poštové kódy a telefónne čísla významných krajín je tu „nápoveda“. Aktivuje sa tlačidlom s tromi bodkami v poli pre zadanie regulárneho výrazu konkrétneho validátora



Editor regulárnych výrazov

Napríklad

Internet email: `\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*`

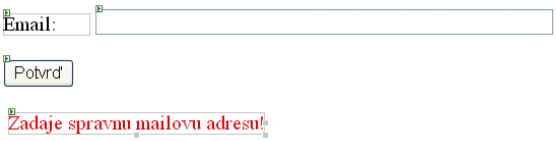
URL adresa: `http(s)?://([\w-]+\.)+[\w-]+(/[\w- ./?%&=]*)?`

Kód aplikácie potom bude

```
<form id="form1" runat="server">
    <div>
        <h1>
            Príklad pre RegularExpressionValidator </h1>
        <p>
            <asp:Label ID="Label1" Runat="server" Height="22px" Width="86px"
Text="Email:"></asp:Label>
            <asp:TextBox ID="TextBox1" Runat="server" Height="25px"
Width="451px"></asp:TextBox>  </p>
        <p>
            <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď"
```

```
OnClick="btPotvrd_Click" />
        </p>
        <p>
      <asp:RegularExpressionValidator ID="RegularExpressionValidator1"
Runat="server"
                  ErrorMessage="Zadajte spravnu mailovu adresu!"
ControlToValidate="TextBox1" ValidationExpression="\w+([-+.']\w+)*@\w+([-
.]\w+)*\.\w+([-.]\w+)*">
            </asp:RegularExpressionValidator>
    </p>

 </div>
</form>
```

Príklad pre RegularExpressionValidator



príklad použitia Regular Expres Validátora

Custom Validator

Doteraz preberané validátory boli svojimi tvorcami navrhnuté tak, aby pomáhali validovať najčastejšie sa vyskytujúce situácie. Veľakrát sme postavení pred situáciu, validovať nejakú súvislosť, ktorá vyplýva z aplikačnej logiky. Potrebujeme napríklad skontrolovať rodné číslo, číslo kreditnej karty a podobne. Napríklad Rodné číslo v Českej aj Slovenskej republike sa zadáva v tvare napríklad 650327/6153. Kontrolný mechanizmus u rodných čísel osôb po roku asi 1953 spočíva v kontrole deliteľnosti celého rodného čísla jedenástimi. Podobné závislosti platia aj pre základnú kontrolu čísla kreditnej karty a podobne. Nejedná sa o nijakú podrobnú kontrolu a už vôbec nie validáciu, no už aj táto jednoduchá kontrola v prípade rodného čísla s pomerne vysokou pravdepodobnosťou odhalí náhodne zadané rodné číslo a samozrejme pomôže aj odhaliť preklep pri jeho zadávaní.

Definícia syntaxe

```
<asp:CustomValidator
      id="ID validátora"
      ControlToValidate=" ID prvku, ktorého obsah chceme kontrolova"
      ClientValidationFunction="ID validačnej funkcie u klienta"
      OnServerValidate="Validačná_funkcia"
      ErrorMessage="Oznam, ktorý sa vypíše v prípade neplatnej validácie"
      Text="text prvku"
      ForeColor="hodnota"
      BackColor="hodnota" …
      runat="server" >
</asp:CustomValidator>
```

Pre napísanie vlastnej validačnej funkcie máme dve možnosti. Prostredníctvom parametra ClientValidationFunction naviažeme validátor na validačnú funkciu u klienta, ktorá býva napísaná obvykle v JavaScripte, alebo VB Scripte.

Validačná funkcia v JavaScripte:

```
Function ValidationFunctionName (source, arguments)
```

Validačná funkcia vo VBScripte:

```
Sub ValidationFunctionName (source, arguments)
```

Prostredníctvom parametra OnServerValidate môžeme naviazať validátor na validačnú funkciu na serveri.

Aj syntaktické formy pre serverové validačné funkcie sú predpísané. Pre obidva najpoužívanejšie programovacie jazyky Visual Basic a C# sú tieto formy nasledovné:

[Visual Basic]

```
Sub ServerValidation (source As object, args As ServerValidateEventArgs)
    args.IsValid = True/Talse
End Sub
```

[C#]

```
void ServerValidation (object source, ServerValidateEventArgs args)
{
    args.IsValid = true/false;
}
```

Ako príklad sme vybrali jednoduchý test, či je zadané číslo párne. Validačná funkcia v tomto prípade pobeží na serveri.

```
     void ServerValidation (object source, ServerValidateEventArgs arguments)
  {
       try
      {
         int i = int.Parse(arguments.Value);
      }

      catch (Exception e)
      {
         //
      }

       arguments.IsValid = ((i%2) == 0);
  }
```

Kompletný kód jednoduchej miniaplikácie na testovanie skutočnosti, či bolo zadané párne číslo je nasledovný (validačná funkcia je v samostatnom súbore so zdrojovým kódom):

```
<form id="form1" runat="server">
    <div>
        <h1>
            Príklad pre CustomValidator </h1>
        <p>
            <asp:Label ID="Label1" Runat="server" Height="21px" Width="166px"
Text="Zadaj párne číslo:"></asp:Label>
            <asp:TextBox ID="TextBox1" Runat="server" Height="25px"
Width="243px"></asp:TextBox>  </p>
        <p>
            <asp:Button ID="btPotvrd" Runat="server" Text="Potvrď"
OnClick="btPotvrd_Click" />
        </p>
        <p>
            <asp:CustomValidator ID="CustomValidator1" Runat="server"
ErrorMessage="Cislo nie je parne!"
                ControlToValidate="TextBox1"
ClientValidationFunction="ServerValidation">
            </asp:CustomValidator>
        </p>

  </div>
</form>
```

Custom validator – test na párne číslo

ValidationSummary Control

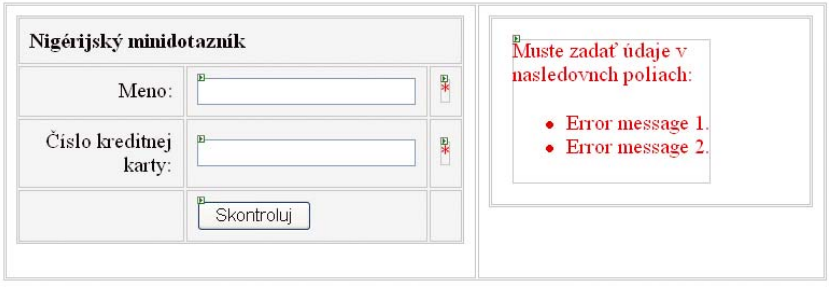
Zatiaľ sme testovali jednotlivé validátory izolovane po jednom. Ak však navrhujeme formulár na ASP.NET stránke, ktorý má klient vyplniť, tento formulár obsahuje spravila viac zadávacích prvkov. Keďže základom stabilnej a spoľahlivej aplikácie je určitá paranoja voči používateľovi, spravidla kontrolujeme väčšinu zadaných údajov. Každý z údajov však pritom môže byť ošetrený validátorom iného typu. Pomocou prvku ValidationSummary dokážeme vypísať oznamy všetkých validátorov na jednom mieste.

Definícia syntaxe

```
<asp:ValidationSummary
  id=" ID prvku"
  DisplayMode="BulletList | List | SingleParagraph"
  EnableClientScript="true | false"
  ShowSummary="true | false"
  ShowMessageBox="true | false"
  HeaderText="Text ktorý sa zobrazí v záhlaví výpisu oznamov validátorov"
  runat="server"/>
```

Funkciu tohto validátora najlepšie pochopíme na príklade jednoduchého formulára s dvoma polami pre zadanie údajov. Na každé z polí je naviazaný RequiredField Validator. V tomto prípade okrem správne vyplneného formulára, teda takého kde sú vyplnené obidva polia sa môže používateľ dopustiť troch druhov chýb. Môže vyplniť len jedno z polí, prípadne môže nechať obidva polia nevyplnené. Túto situáciu riešime pomocou ValidationSummary

Vo vývojovom prostredí vytvoríme formulár s dvoma textovými poliami, pričom na každé z nich bude naviazaný samostatný RequiredField Validator. Validátor umiestnime vedľa editačného poľa a ako text jeho chybovej správy dáme napríklad hviezdičku (*),



Vytvorenie príkladuna Validation Summary

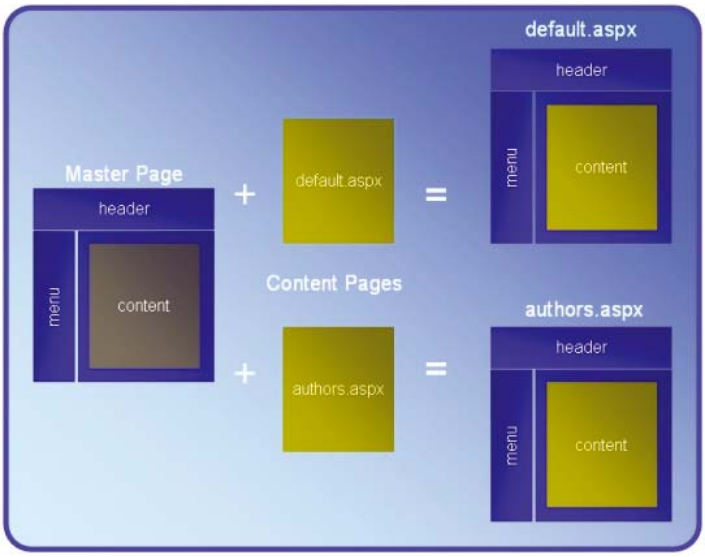
Výpis chybových stavov bude mať na starosti komponenta ValidationSummary. Kompletný kód miniaplikácie potom bude:

```
<form id="Form1" runat="server">
        <table cellpadding="10"><tbody>
            <tr><td><table cellpadding="10" bgcolor="#eeeeee"><tbody>
            <tr><td colspan="3"><b>Nigérijský minidotazník</b></td></tr>

            <tr><td align="right">Meno:</td>
                 <td><asp:TextBox id="TextBox1"
```

```
runat="server"></asp:TextBox></td>
                  <td align="middle" rowspan="1">
                        <asp:RequiredFieldValidator id="RequiredFieldValidator1"
                             runat="server" Text="*" Width="100%" InitialValue=""
                              Display="Static" ErrorMessage="Meno."
                              ControlToValidate="TextBox1">
                          </asp:RequiredFieldValidator></td></tr>

              <tr><td align="right">Číslo kreditnej karty: </td>
                 <td><asp:TextBox id="TextBox2" runat="server"></asp:TextBox></td>
                 <td><asp:RequiredFieldValidator id="RequiredFieldValidator2"
                           runat="server" Text="*" Width="100%" Display="Static"
                            ErrorMessage="Cislo karty. " ControlToValidate="TextBox2">
                      </asp:RequiredFieldValidator></td></tr>
          <tr><td></td>
          <td><asp:Button id="Button1" runat="server"
Text="Skontroluj"></asp:Button></td>
          <td></td></tr>
          </tbody></table></td>

          <td valign="top"><table cellpadding="20"><tbody>
               <tr><td><asp:ValidationSummary id="valSum" runat="server"
                           HeaderText="Muste zada  údaje v nasledovnch poliach:"
                           EnableClientScript="true" DisplayMode="BulletList">
                     </asp:ValidationSummary></td></tr>
           </tbody></table></td></tr>
        </tbody></table>
    </form>
```

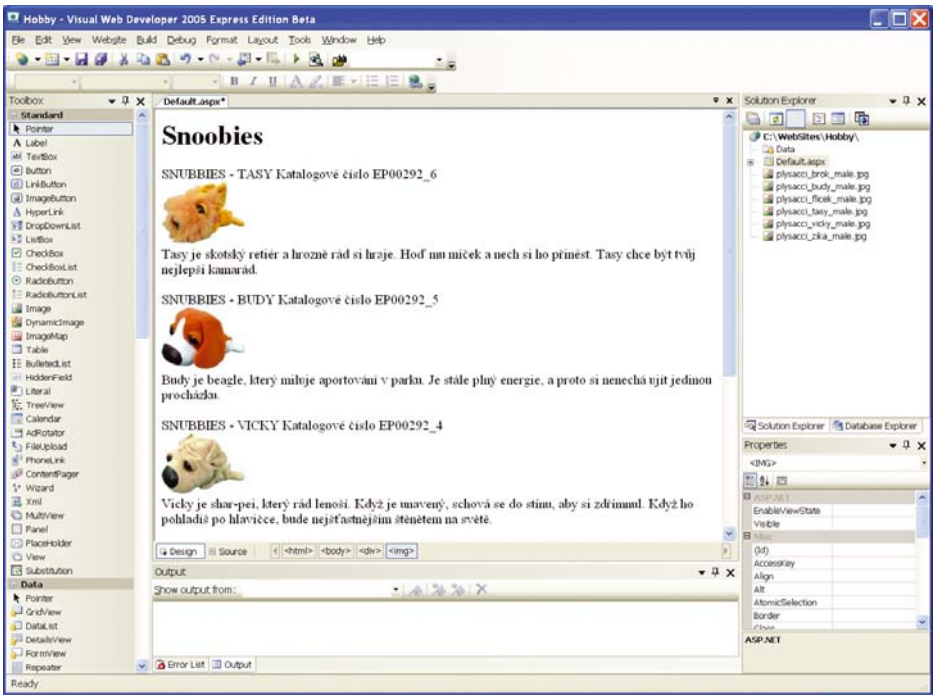
Kapitola 4
Master Page

Ak sa zamyslíte nad princípom zobrazovania obsahu mnohých webových aplikácií, určitá časť stránky má viac menej stály – unifikovaný obsah, ktorý tú ktorú aplikáciu odlišuje od iných (logo, banery, menu...) a podľa toho aký obsah si klient pozerá mení sa len určitá časť plochy stránky. ASP.NET 2.0 a teda aj Visual Web Developer ponúka pre vývoj webových aplikácií tohto typu črtu nazvanú Master Page. V praxi to funguje tak, že do projektu bude doplnený jeden súbor s príponou .master, ktorý môže obsahovať statický text, HTML elementy a serverové komponenty. Podľa tejto Master Page šablóny sa potom budú generovať stránky aplikácie. Asi to takto na prvé počutie nebude príliš jasné. Skúste sa teda zamyslieť nad schématickým obrázkom alebo najlepšie skúste si vytvoriť jednoduchú aplikáciu podľa nášho návodu.



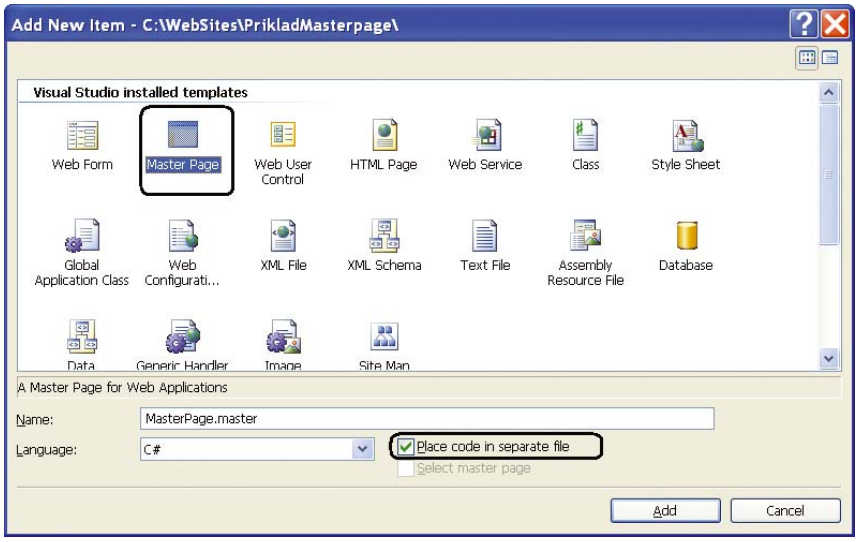
Princíp aplikácie využívajúcej Master Page

Aj táto kapitola by mohla začínať v zmysle „Vytvoríme nový projekt typu Empty Web Site...“ , pokúsime sa však výjsť z inej východiskovej situácie. Máme už hotovú nejakú hobby aplikáciu, v našom prípade najjednoduchšiu ASP.NET stránku, ktorá neobsahuje nijaké serverové komponenty, len hypertextový obsah ktorý sa povedzme vzťahuje k nášmu hobby. V našom prípade to bol zberateľský katalóg plyšových psíkov. Vo vašom prípade, jednoducho skopírujte napríklad nejaký obsah do stránky Default.aspx u novo vytvoreného projektu. Aplikácia môže samozrejme obsahovať viac stránok. Takúto „jestvujúcu“ aplikáciu doplníme o Master Page



Hobby aplikácia pred rozšírením o Master Page

V okne Solution Explorer v kontextovom menu aktivujeme funkciu Add New Items. V ponúkanom dialógu si zvolíme ikonu Master Page . Je výhodné označiť voľbu „Place code in separate file“, aby bol náš projekt prehľadnejší



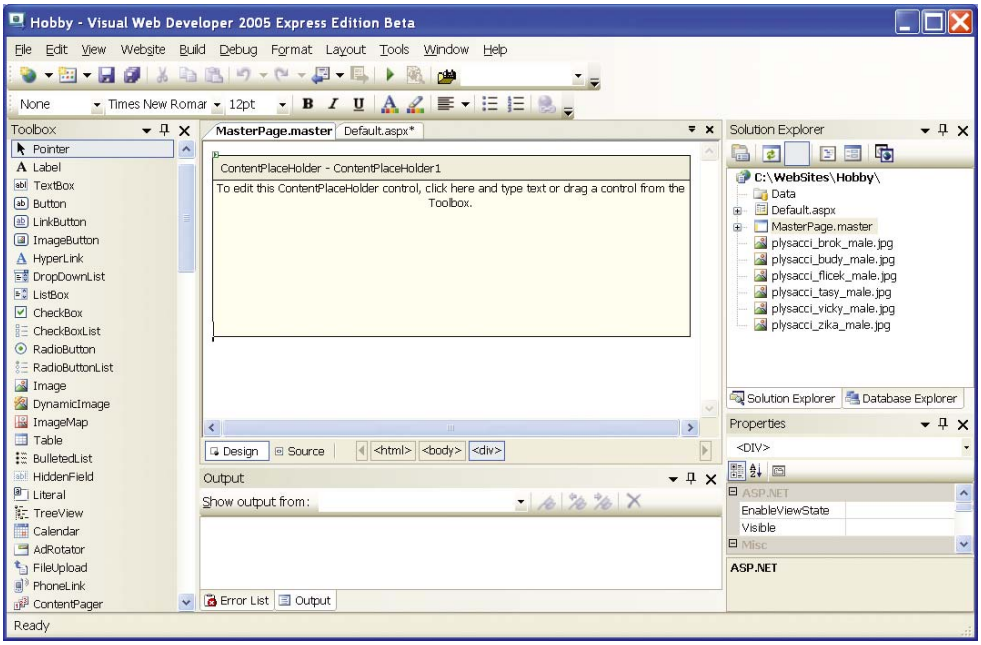
Vytvorenie dokumentu typu Master Page

Master Page dokument má príponu *. master a zatiaľ sprievodcom vygenerovaný kód bude

```
<%@ Master Language="C#" CompileWith="MasterPage.master.cs"
ClassName="MasterPage_master" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:contentplaceholder id="ContentPlaceHolder1" runat="server">
        </asp:contentplaceholder>
    </div>
    </form>
</body>
</html>
```

Všimnime si novú deklaráciu `<%@ Master %>` a nový tag `<asp:contentplaceholder>`.

V zložke Design sa zobrazí vizuálna podoba prvku ContentPlaceHolder. Je to zatiaľ jednoduché okno so šedým záhlavím.

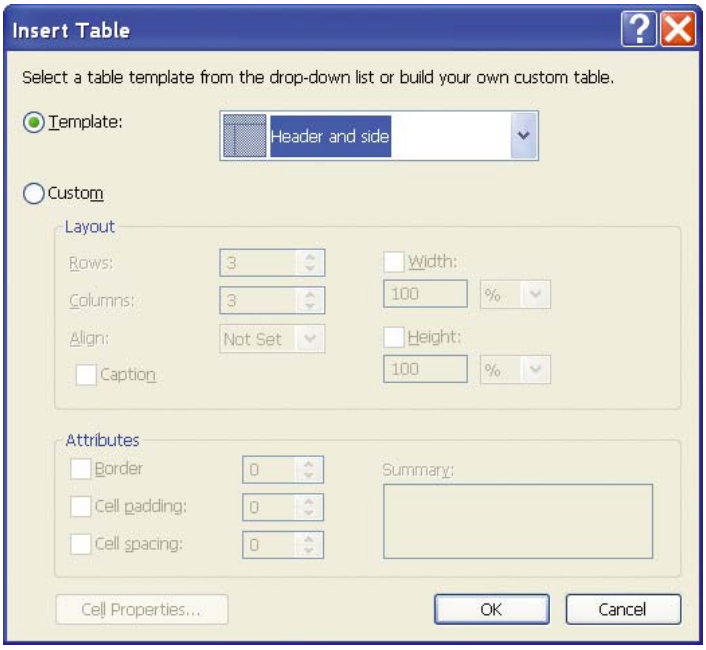


Vytvorenie dokumentu typu Master Page

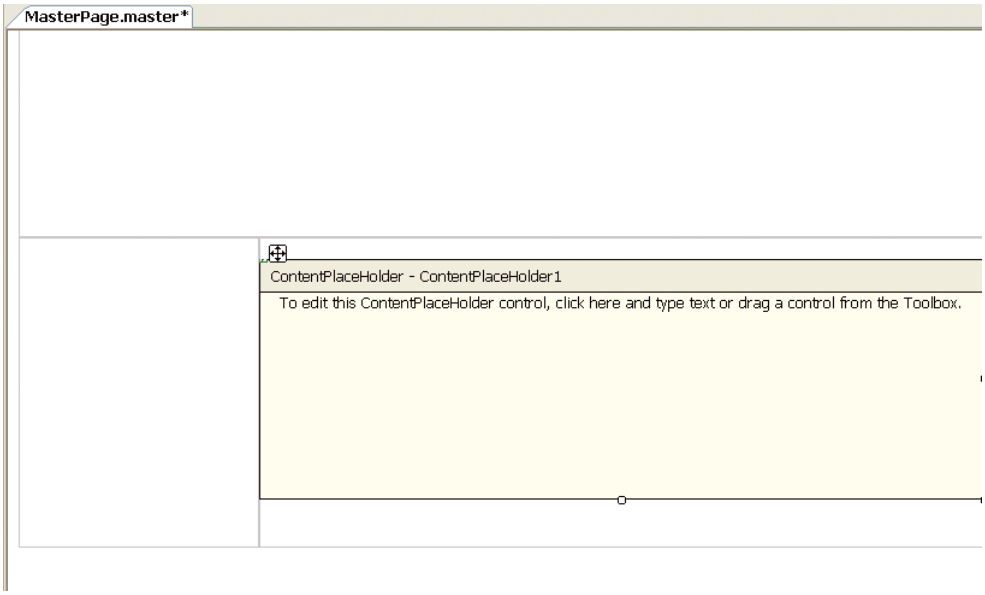
zároveň sa vytvorí aj súbor so separátnym kódom, v našom prípade v jazyku C#

```
public partial class MasterPage_master
{

}
```

V takejto podobe by nám Master Page veľa úžitku nepriniesla. Zatiaľ totiž komponenta ContentPlaceHolder, v ktorej sa budú meniť obsahové stránky „okupuje“ celú plochu aplikácie, takže nič iné okrem obsahovej stránky by sa nám nezobrazilo, a výmenu celých stránok dosiahneme inými triviálnymi spôsobmi. Na hlavnej stránke musíme vytvoriť takzvaný „vizuál“ aplikácie, ktorý je stále rovnaký, bez ohľadu na to, ktorá stránka sa v komponente ContentPlaceHolder práve zobrazuje. Najjednoduchšie rozčlenenie plochy dosiahneme pomocou tabuľky.
Z pripravených šablón tabuliek v dialógu Insert Table (z menu Layout) zvolíme „Header and side“



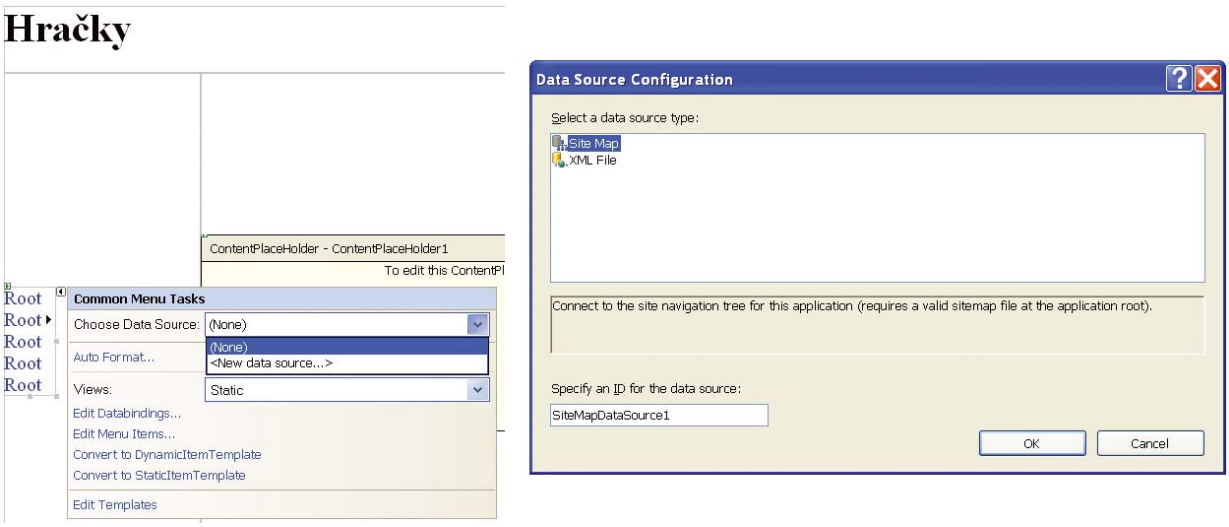
Vytvorenie tabuľky Master Page



Tabuľka na Master Page

Komponentu ContentPlaceHolder presunieme do najväčšieho políčka tabuľky. Potom horné a ľavé menšie políčko zostane rezervované napríklad pre menu, bannery, grafiku a podobne. Do horného úzkeho riadku tabuľky vložíme pre jednoduchosť v textovej podobe nejaký nadpis, ktorý bude našu aplikáciu charakterizovať, napríklad HRAČKY.
Z toolboxu zo zložky Navigation presunieme do ľavého stĺpca tabuľky komponentu menu. V Common Menu Task aktivujeme voľbu New Data Source, takže pokračujeme dialógom „Data Source Configuration“, kde máme na výber dve možnosti
– Site Map
– XML File

Aktivujeme voľbu Site Map. Na plochu aplikačného formulára po tomto úkone pribudla komponenta SiteMapDataSource.

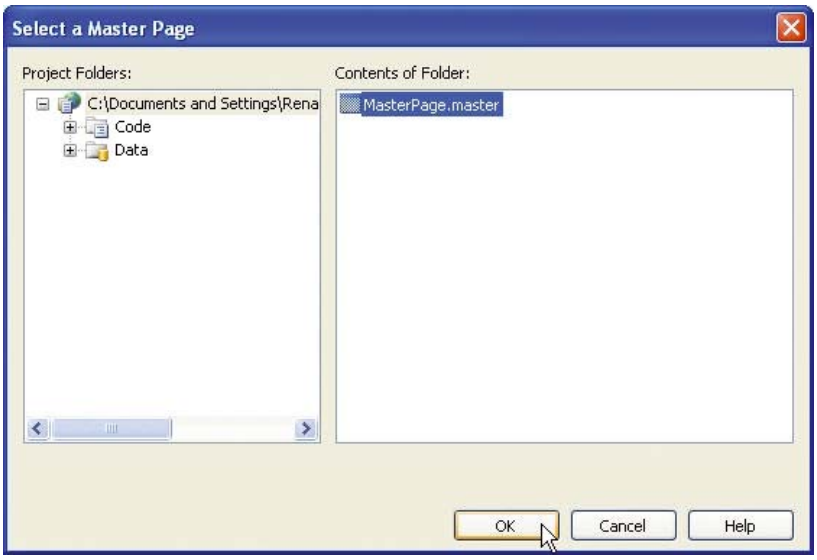


Menu

Aby to celé fungovalo, musíme súbor sitemap, pridať do projektu (voľba Add new item)

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap>
    <siteMapNode url="" title=""  description="" roles="">
      <siteMapNode url="" title=""  description="" roles="" />
      <siteMapNode url="" title=""  description="" roles="" />
    </siteMapNode>
</siteMap>
```

Následne pridáme do projektu aj súbor typu WebForm, ktorý nazveme ContentPage.aspx. Nezabudnime zaškrtnúť políčko Select Master Page

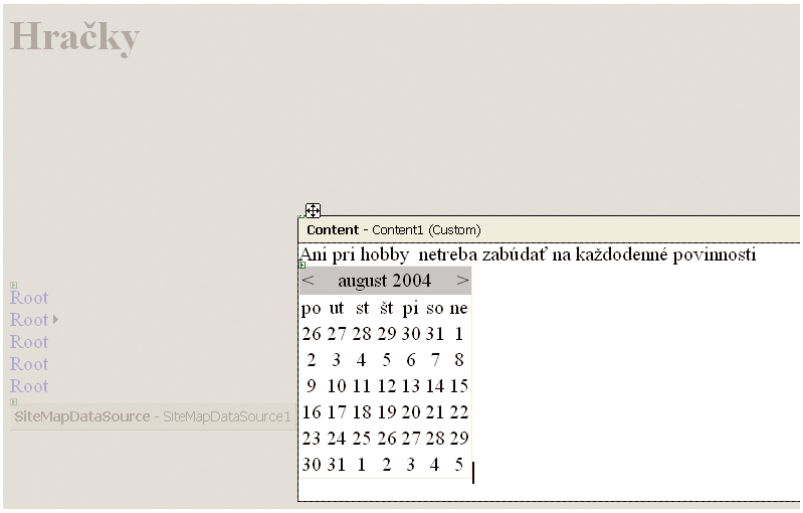


Select Master Page

Stránka bude obsahovať len jeden riadok kódu

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master" Title="Untitled Page"%>
```

a v návrhovom zobrazení sa zobrazí Master Page so zvýrazneným prvkom ContentPlaceholder. Môžeme sem pridať nejaký obsah (kontextové menu Create Custom Content), napríklad textový reťazec, prvky a podobne. My sme pridali textový reťazec a komponentu kalendár



Content Page

Aktualizujeme subor Web.Sitemap

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap>
    <siteMapNode url="Home.aspx" title="Home"  description="" roles="" >
       <siteMapNode url="Default.aspx" title="Psiky Snoobies"  description="" roles="" />
      <siteMapNode url="ContentPage.aspx" title="My Content Page"  description=""
roles="" />
    </siteMapNode>
</siteMap>
```

Nakoľko sme sa rozhodli na „Master Page“ migrovať už existujúcu ASP.NET aplikáciu, potrebujeme modifikovať stránku Default.ASPX, tak aby bola vztiahnutá k Master Page. To urobíme v dvoch krokoch, pričom ako základ použujeme stránku ContentPage.aspx.

Zo stránky ContentPage.aspx zoberieme do clipboardu reťazec **MasterPageFile="~/MasterPage.master"** a nakopírujeme ho do stránky Default.aspx

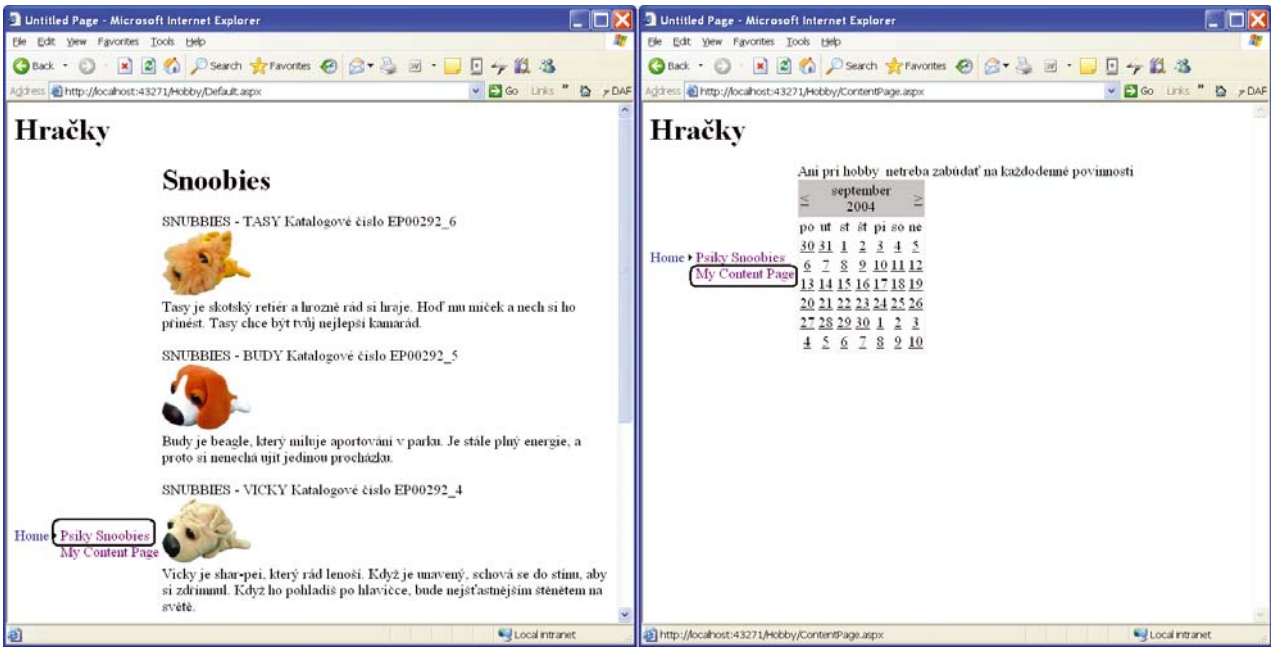
```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
CompileWith="Default.aspx.cs" ClassName="Default_aspx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1>
            Snoobies</h1>
        SNUBBIES - TASY Katalogové číslo EP00292_6 <br />
        <img src="plysacci_tasy_male.jpg" />
        <br />
        Tasy je skotský retiér a hrozně rád si
        hraje. Hoď mu míček a nech si ho přinést. Tasy chce být tvůj nejlepší
kamarád.
        <br />

    </div>
    </form>
</body>
</html>
```

To nie je ale všetko. Všimnite si HTML kód označený sivým pozadím.
tento kód je potrebné nahradiť kódom novým, ktorý taktiež zoberieme zo stránky ContentPage.aspx

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
CompileWith="Default.aspx.cs" ClassName="Default_aspx" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="server">
    <div>
        <h1>
```

```
  Snoobies</h1>
        SNUBBIES - TASY Katalogové číslo EP00292_6 <br />
        <img src="plysacci_tasy_male.jpg" />
        <br />
        Tasy je skotský retiér a hrozně rád si
       hraje. Hoď mu míček a nech si ho přinést. Tasy chce být tvůj nejlepší kamarád.
        <br />

    </div>
</asp:Content>
```

Po uvedených úpravách kódu môžeme aplikáciu otestovať



Spustenie aplikácie obsahujúcej Master Page. Vľavo a vpravo vidíme rôzny obsah ContentPage vybraný pomocou menu

Kapitola 5

Databázový server SQL Server 2005 Express Edition

Vo sfére „hobby“ webových aplikácií sa pre ukladanie údajov používa najmä voľne šíriteľný databázový server MySQL. Donedávna to bolo jedno z mála voľne dostupných riešení. Druhou alternatívou, ktorú pozná každý kto sa pokúšal niekedy vytvárať databázovú aplikáciu na platforme Microsoft a databázový server by bola vzhľadom k nasadeniu aplikácie príliš nákladná záležitosť, bolo MSDE (Microsoft Desktop Engine) - voľne šíriteľné, plne funkčné jadro MS SQL Serveru primárne určené pre vývoj a testovanie databázových aplikácií a pri prevádzke databázových aplikácií na lokálnych počítačoch. Hlavné úmyselne implementované obmedzenie MSDE spočívalo v možnom prístupe maximálne piatich procesov súčasne. Vzhľadom na nasadenie u webových projektov toto obmedzenie nebolo príliš šťastne vymyslené, no pri dobre navrhnutej a optimalizovanej webovej aplikácii sa dali obslúžiť desiatky prístupov za minútu. Obmedzená bola aj maximálna veľkosť databázových súborov na 1GB. V porovnaní s MySQL, ktorý okrem rôznych iných obmedzení napríklad nepodporoval transakcie bolo MSDN vynikajúcou alternatívou, hlavnou výhodou bola stopercentná kompatibilita s Microsoft SQL Serverom, takže kedykoľvek môžeme prakticky okamžite vymeniť databázový stroj MSDE za plnú verziu MS SQL Servera. Náš projekt pritom zostane úplne nezmenený.

Za „nástupníka“ MSDE môžeme považovať SQL Server 2005 Express Edition. Je to „odľahčená“ verzia nového databázového serveru SQL Server 2005 (kódové označenie „Yukon“). Uplatní sa pre budovanie jednoduchých dynamických aplikácií, ktoré predpokladajú zber ukladanie a manipuláciu s údajmi. Môžu to byť katalógové údaje alebo údaje o zákazníkoch a ich objednávky, prípadne textový a multimediálny obsah informačného internetového portálu a podobne. Požiadavky na spoľahlivosť, bezpečnosť a jednoduchú administráciu, ktoré sú úplne samozejmé u drahých komerčne dostupných databázových serverov prenikajú aj do sféry hobby aplikácií. Zo svojho „veľkého“ brata s kódovým označením Yukon si táto zjednodušená verzia ponechala všetky nové črty, hlavne vynikajúcu podporu najrozšírenejšieho formátu dokumentov pre medziplatformovú výmenu informácií - XML, dynamické ladenie parametrov databázy. Aby sa však databázový server edície „Express“ nenasadzoval do podnikovej sféry kam nie je určený, hlavne preto, že podniková sféra na rozdiel od hobby by mala prinášať zisk a profitovať na tom by pochopiteľne mal aj výrobca databázového serveru, boli zavedené niektoré obmedzenia. Obmedzenia v porovnaní s plnou verziou SQL Servera 2005 sa dajú zhrnúť do troch bodov

– podpora jedného procesora (produkt je síce možné nainštalovať aj na viacprocesorový server, no využívať sa bude len jeden procesor)
– podpora maximálne 1GB pamäti RAM
– maximálna veľkosť databázy 4 GB

Sami vidíte, že k týmto limitujúcim parametrom sa pri hobby (a ani pri mnohých iných) aplikáciách pravdepodobne ani zďaleka nepriblížime.

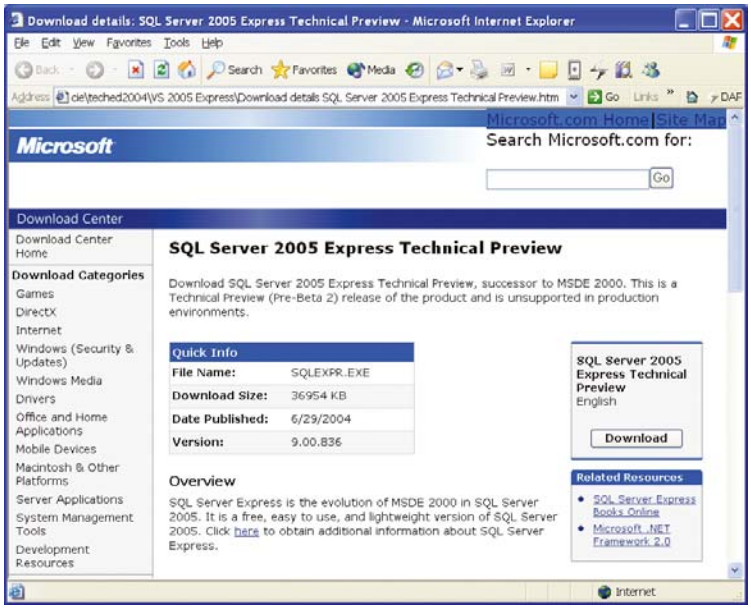
Administrácia, návrh databázových štruktúr a práca s údajmi v databáze sa vykonáva priamo z prostredia Visual Web Developera. Po vytvorení novej databázy v zložke Data Solution Explorera sa po kliknutí na názov databázy do okna Solution Explorera pridá záložka Database Explorer. Prakticky všetky úkony spojené s návrhom databázových štruktúr (vytváranie tabuliek, pohľadov, uložených procedúr) a tvorbou dopytov pre výber údajov je možné realizovať pomocou interaktívnych grafických dialógov.

Nové črty

Nakoľko SQL Server 2005 predstavuje novú výraznú inováciu, tieto nové črty (s výnimkou čisto podnikových záležitostí typu Business Intelligence) sú obsiahnuté aj v „Express edícii“, Jedná sa hlavne o podporu CLR (Common Language Runtime), čo umožňuje vytvárať a ladiť uložené procedúry priamo v .NET jazykoch, teda už nie len v T-SQL (Transact SQL) ale aj vo Visual Basicu, C# a ďalších jazykoch. Veľmi významná pre webové aplikácie je aj podpora XML ako natívneho dátového typu. Pre migráciu aplikácie na iný server je možné prekopírovať databázový súbor (xcopy) do inej destinácie, bez nutnosti ďalších úprav a zásahov. Inštalácia je okrem prehľadného sprievodcu možná aj v tichom (silent mode) móde bez sprievodných hlásení. Výhodou je aj pomerne hlboká integrácia podpory databázového servera SQL Server 2005 do vývojových prostredí Visual Studio .NET 2005 a Visual Web Developera. V novej verzii VS 2005 (od verzie Beta 3) je v prípade WinForms aplikácií možné aj jednoduché šírenie databázového enginu spolu s aplikáciou systémom ClickOnce.

Inštalácia

Inštalačný súbor produktu Microsoft SQL Server 2005 Express Edition má po stiahnutí z webu (http://msdn.microsoft.com/sql/express) necelých 38 megabajtov. Na disku počítača však musíme počítať s oveľa väčším priestorom, štandardný nástroj (súčasť Windows XP) pre odinštalovanie aplikácií, napríklad hlási pre tento program obsadenie takmer 600 megabajtov.



Download produktu Microsoft SQL Server 2005 Express Edition

Databázový server aj keď v odľahčenej verzii je predsa len program inej kategórie, než kalkulačka alebo katalogizátor mp3 súborov, ktoré sa nainštalujú na jedno kliknutie. Hoci jeho samotná inštalácia je jednoduchá, je potrebné splniť určité požiadavky.

Podporované operačné systémy: Windows 2000 Advanced Server, Windows 2000 Professional Edition , Windows 2000 Server, Windows 2000 Service Pack 4, Windows Server 2003, Windows XP Professional Edition , Windows XP Service Pack 1

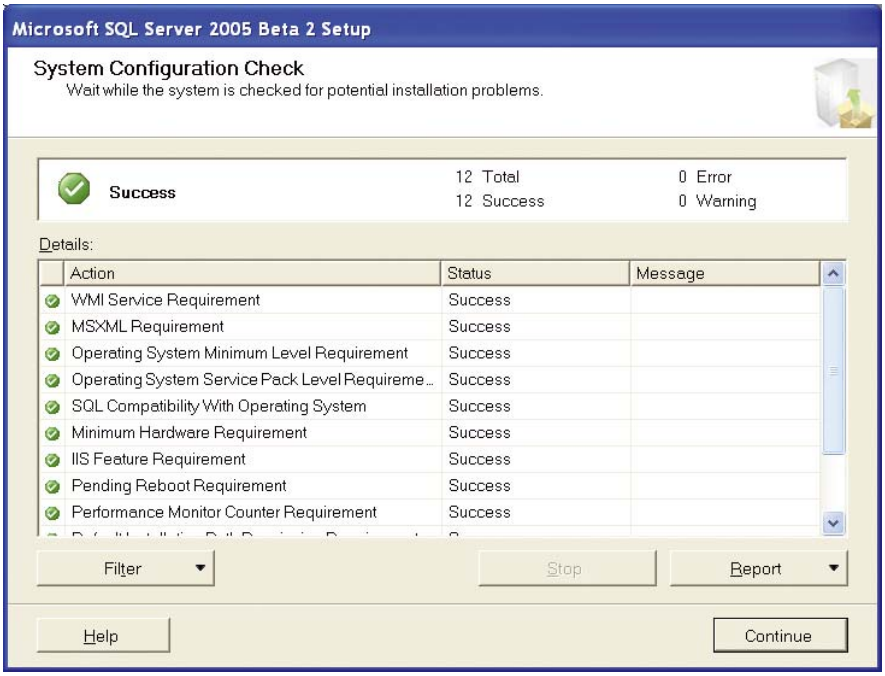
Požiadavky na počítač: procesor Intel Pentium kompatibilný, minimálne 600 MHz, odporúčaný je však minimálne 1 GHz. Z hľadiska operačnej pamäti je asi zbytočné uvažovať o minimálne požadovaných 256 MB, jednoznačne odporúčame 512 a viac megabajtov. Taktiež údaj 405 MB priestoru na pevnom disku je mierne podhodnotený a je potrebné počítať zhruba so 600 megabajtami. Tieto požiadavky sú na dnešnú dobu pomerne skromné a vyhovie im prakticky každý dnes predávaný počítač alebo notebook, nanajvýš bude za pár stovák potrebné rozšíriť pamäť RAM.

Kompatibilita: z hľadiska kompatibility je pred inštaláciou produktu nutné odinštalovať prípadné predchádzajúce beta, alebo neskôr aj ostré verzie SQL Servera 2005, teda Yukonu. Ostatné staršie verzie SQL Servera, počnúc SQL Serverom 2000 cez SQL Server 7.0, prípadne ešte staršie verzie môžeme na disku ponechať.

.NET Framework 2.0 Nutným predpokladom pre inštaláciu databázového servera SQL Server 2005 Express Edition je predinštalovaná technologická platforma Microsoft .NET Framework 2.0. Preto ak dvojkovú verziu .NET Frameworku nemáme, musíme ju stiahnuť z webu a inštalujeme ju ako prvú. Potom stiahneme a uložíme na lokálny počítač inštalačný súbor SQLExpr.exe. Jeho spustením začne samotný proces inštalácie. Ako bolo uvedené pri popise produktu Visual Web Developer, ak databázový server plánujeme používať z týmto vývojárskym nástrojom je výhodné nainštalovať SQL Server 2005 Express Edition v jednom inštalačnom procese spolu s Visual Web Developerom. Nasledujúce kroky sa potom taktiež odohrajú v plnom rozsahu a v nezmenenom poradí ako dielčí inštalačný proces. Sprievodca inštaláciou je riešený klasicky, preto ho popíšeme len heslovito, obrázky všetko vysvetlia.

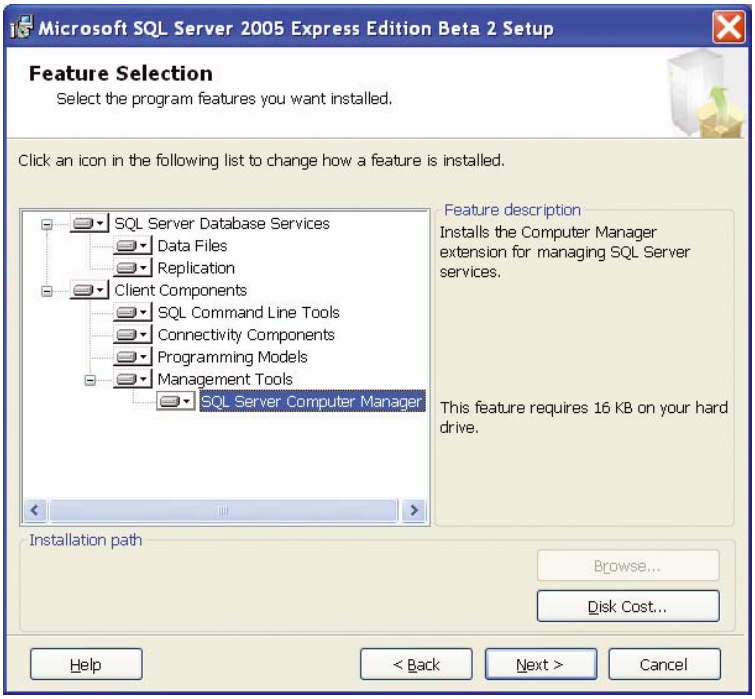
Úvodný dialóg pre inštaláciu produktu Microsoft SQL Server Express Edition

Po úvodnom dialógu pre inštaláciu produktu bude zahájená kontrola splnenia všetkých systémových požiadaviek. Na prípadné nezrovnalosti a možnosti ich odstránenia bude používateľ v tejto etape zrozumiteľne upozornený



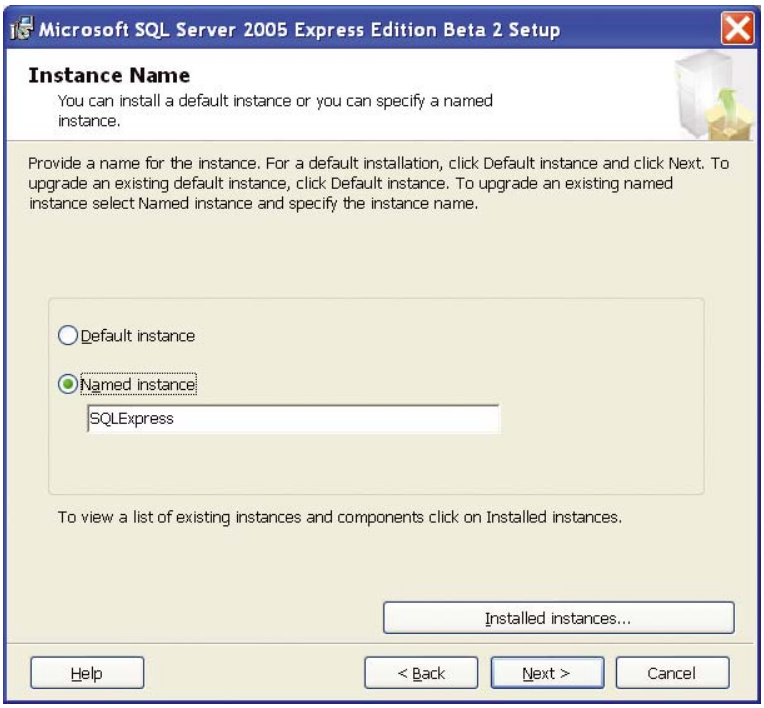
Inštalácia SQL Servera Express Edition – kontrola systémovej konfugurácie

V dialógu pre inštaláciu komponentov odporúčame začiatočníkom ponechať všetky implicitne označené položky. Pokročilejší používatelia si v prípade ak sa rozhodnú cielene niektorú súčasť produktu nenainštalovať určite budú vedieť poradiť



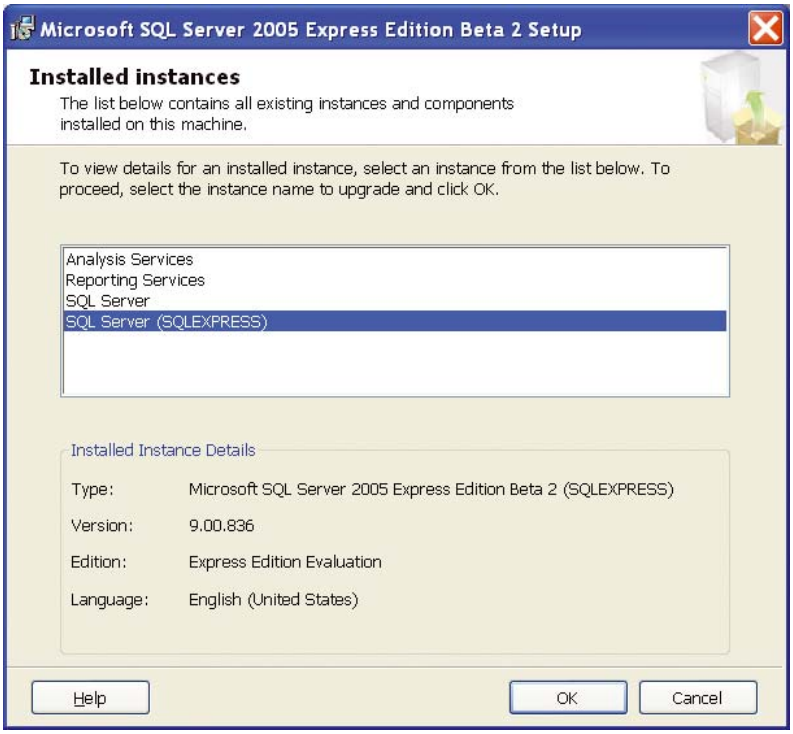
Inštalácia SQL Servera Express Edition – výber komponentov

Určitý význam má pomenovanie inštancie, implicitne bude v príslušnom inštalačnom dialógu ponúknutý názov SQLExpress. Tento názov budeme potrebovať pri vytváraní pripojovacích reťazcov aj pri pripojovaní pomocou klientskej konzolovej aplikácie (sqlcmd -S Server\Instance)



Inštalácia SQL Servera Express Edition – pomenovanie inštancie

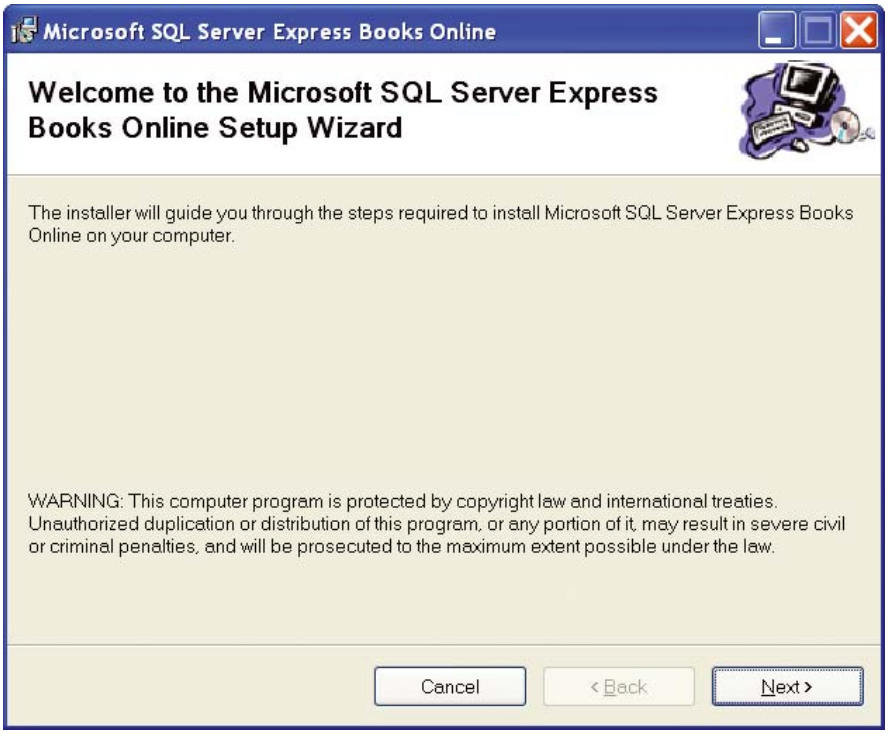
V záverečnom dialógu budeme informovaní o nainštalovaných inštanciách. V našom prípade bol na počítači predtým nainštalovaný SQL Server 2000 spolu s jeho analytickými a reportovacími službami.



Inštalácia SQL Servera Express Edition – detaily o inštancii

Dokumentácia

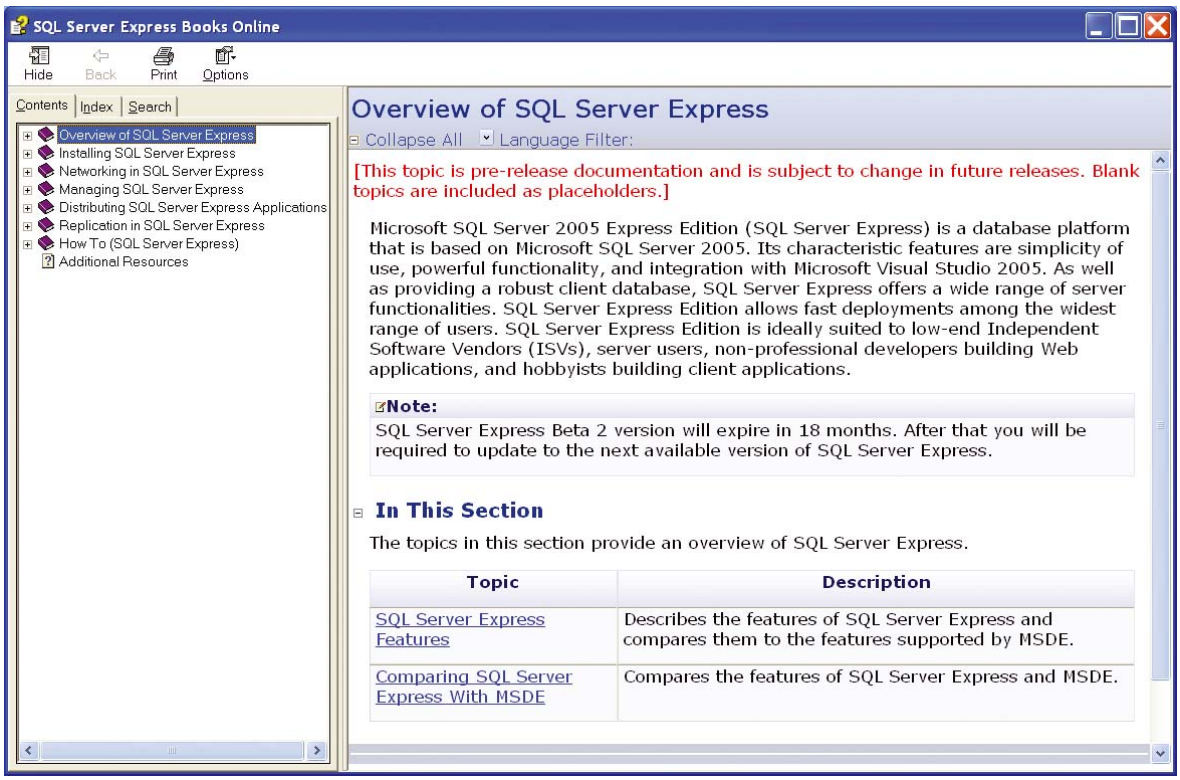
Okrem inštalačného súboru databázového servera je možné na uvedenej adrese stiahnuť aj dokumentáciu. Inštalačný súbor dokumentácie SqlExpressBOL.msi má niečo vyše 200 kilobajtov



Inštalácia dokumentácie

Dokumentácia bude implicitne nainštalovaná do adresára

c:\Program Files\Microsoft SQL Server\90\Tools\Books\1033\



Inštalácia dokumentácie

Základné črty SQL Servera 2005 Express Edition

Nový databázový server aj v odľahčenej verzii Express Edition umožňuje vytvárať robustné databázy pre prevádzku dynamických aplikácií. Obsahuje plnohodnotné jadro SQL Servera 2005 s integrovaným optimalizátorom dotazov. Stopercentná kompatibilita s plnou verziou sa týka aj jazyka T-SQL (rozšírená množina príkazov jazyka SQL), podpory pohľadov, uložených procedúr, kurzorov... Kapitolou samou o sebe vo všetkých verziách SQL Servera 2005 je rozšírená podpora XML, teda natívny XML dátový typ, podpora dotazovania XQuery a XML schém.

Aplikácia Express Manager obsahujúca sprievodcov najčastejšie vykonávaným úkonom nebol v dobe písania tejto publikácie ešte k dispozícii. Bude dostupný ako samostatný download. Pre návrh databázových objektov, pridávanie údajov a lokálne dotazovanie je možné použiť Visual Studio Data Explorer integrovaný do vývojového prostredia Visual Web developera.

Stručné minimum jazyka SQL

Nasledujúci prehľad základných príkazov jazyka SQL si nekladie za cieľ výklad syntaxe, poslúži skôr na získanie prehľadu možností tohoto príkazového jazyka.

Typický SQL príkaz pre výpis všetkých údajov z databázovej tabuľky má tvar:

```
SELECT * FROM dbo.zakaznici
```

V úvode tohoto odseku sme naznačili, že to s výkladom syntaxe nebudeme preháňať.
ale v tomto prípade urobíme výnimku. Príkaz SELECT si to skutočne zaslúži.

SELECT [**DISTINCT**] *položky* **FROM** *meno_tabulky* [**WHERE** *podmienka_výberu*] [**GROUP BY** položky, [**HAVING** *podmienka_agregácie*]] [**ORDER BY** *položky*]

Pomocou klauzule **DISTINCT** dokážeme zariadiť, že sa nám duplicitné riadky objavia vo výpise len raz. Kľúčovým slovom **FROM** vyšpecifikujeme tabuľku, z ktorej potrebujeme vybrať údaje. Ak potrebujeme vybrať len určitú skupinu zákazníkov, alebo dokonca jednotlivca, musíme to prostredníctvom podmienky v SQL príkaze jasne špecifikovať. Na jej vyjadrenie slúži klauzula **WHERE**. V našom prípade to bude napríklad

```
SELECT * FROM dbo.zakaznici WHERE vek > 30
```
alebo
```
SELECT * FROM dbo.zakaznici VHERE rodne_cislo = '651122/1234'
```

V prvom prípade dostaneme zoznam zákazníkov nad 30 rokov, v druhom prípade podmienke vyhovuje jeden jediný zákazník zo zadaným rodným číslom. Ak chceme výpis zákazníkov nad 30 rokov vylepšiť tým že ich zoradíme od najstaršieho po najmladšieho, použijeme klauzulu **ORDER BY**.

```
SELECT * FROM dbo.zakaznici WHERE vek > 30 ORDER BY vek DESCENDING
```

Dosiaľ popisovaná skupina príkazov slúži len na výber údajov z databázy. Nemôžeme zatiaľ ani mazať, ani vkladať údaje.

Pre vkladanie údajov slúži príkaz **INSERT**

```
INSERT INTO dbo.zakaznici VALUES ('Ignac Knihomol', '591111/1234',
  'Ustredny archiv', 'Bratislava' )
```

Neaktuálne údaje môžeme prepísať novými pomocou príkazu UPDATE.

```
UPDATE dbo.zakazníci SET pracovisko = 'Urad vlady'
WHERE RodneCislo = '591111/1234'
```

Vymazať údaje môže oprávnená osoba pomocou príkazu **DELETE FROM**. Zákazníka, ktorého údaje potrebujeme z databázy vymazať, špecifikujeme podmienkou v klauzuli **WHERE** podľa jedinečného kľúča, v našom prípade rodného čísla:

```
DELETE FROM dbo.zakaznici WHERE RodneCislo = '591111/1234'
```

Okrem dotazov typu SELECT môžeme používať aj takzvané AGREGAČNÉ DOTAZY, ktoré pomocou matematických, alebo štatistických príkazov spracujú hodnoty z celých stĺpcov. V SQL sú implementované napríklad tieto funkcie:

SUM() – súčet hodnôt v stĺpci,
MIN() – minimálna hodnota ,
MAX() – maximálna hodnota,
COUNT() – počet numerických hodnôt v stĺpci,
AVG() – aritmetický priemer numerických hodnôt v stĺpci.

Okrem manipulácie s údajmi môžeme pomocou SQL príkazov pracovať s celou databázou. Dokážeme vytvoriť tabuľky, indexy, dokonca je možné definovať aj používateľov vrátane ich oprávnení.

Na vytvorenie tabuľky slúži príkaz **CREATE TABLE**:

```
CREATE TABLE meno_tabulky
(
   meno_stlpca typ [integritne_obmedzenia],
     ...,
)
```

Napríklad tabuľku vodičov vytvoríme takto:

```
CREATE TABLE dbo.vodici
(
  vodic  VARCHAR(30)  NOT NULL,
  vozidlo_farba VARCHAR(15),
  vozidlo_typ VARCHAR(20)
)
```

Veľmi podobný príkaz **ALTER TABLE** slúži na vykonanie zmien v predtým navrhnutej tabuľke:

```
ALTER TABLE meno_tabulky
(
   ADD meno_stlpca typ [integritne_obmedzenia],
   MODIFY ...
   DROP ...
)
```

Klauzula **ADD** pridá stĺpec do tabuľky, **MODIFY** zmení definíciu stĺpca a pomocou klauzule **DROP** stĺpec odoberieme. Veľmi opatrní musíme byť pri použití klauzuly pre zmenu typu MODIFY. Databázový stroj musí byť schopný vykonať automatickú typovú konverziu. Nie je možná napríklad konverzia z typu VARCHAR na INTEGER.

Pripojovanie sa pomocou klientskej konzolovej aplikácie

Pre správu databázového serveru a ladenie databázovej časti aplikácií potrebujeme dva typy aplikácií. SQL príkazy môžeme pred ich zakomponovaní do ASP.NET kódu ladiť pomocou **klientskej konzolovej aplikácie**. Náplň jej činnosti je jednoduchá. Slúži pre zadávanie príkazov jazyka SQL databázovému serveru a okne tej istej aplikácie taktiež vidíme výstupy, ktoré databázový server vygeneruje ako odozvu na naše príkazy, teda napríklad výpis obsahu databázových tabuliek, potvrdenie vykonania našich príkazov, chybové hlásenia a podobne. Pre administráciu databázy, napríklad pre nastavovanie prístupových práv potrebujeme **aplikáciu pre správu databázy**. Pomocou nástroja pre správu databázy je možné nastaviť stratégiu údržby a zálohovania údajov v databáze a podobne.

SQLCMD

Nápovedu, k tejto jednoduchej konzolovej aplikácii, teda parametre pre spustenie aplikácie získame použitím parametra **sqlcmd -?**

```
C:\Program Files\Microsoft SQL Server\90\Tools\binn>sqlcmd -?
Microsoft (R) SQL Server Command Line Tool
Version 9.00.836 NT INTEL X86
Copyright (C) 2004 Microsoft Corporation.  All rights reserved.

usage: Sqlcmd             [-U login id]          [-P password]
   [-S server]            [-H hostname]          [-E trusted connection]
   [-d use database name] [-l login timeout]     [-t query timeout]
   [-h headers]           [-s colseparator]      [-w screen width]
   [-a packetsize]        [-e echo input]        [-I Enable Quoted Identifiers]
   [-c cmdend]            [-L[c] list servers[clean output]]
   [-q "cmdline query"]   [-Q "cmdline query" and exit]
   [-m errorlevel]        [-V severitylevel]     [-W remove trailing spaces]
   [-u unicode output]    [-r[0|1] msgs to stderr]
   [-i inputfile]         [-o outputfile]        [-z new password]
   [-f <codepage> | i:<codepage>[,o:<codepage>]]  [-Z new password and exit]
   [-k[1|2] remove[replace] control characters]
   [-y variable length type display width]
   [-Y fixed length type display width]
```

```
[-p[1] print statistics[colon format]]
[-R use client regional setting]
[-b On error batch abort]
[-v var = "value"...]  [-A dedicated admin connection]
[-X[1] disable commands[and exit with warning]]
[-? show syntax summary]
```

Ak sa chceme pripojiť ku konkrétnej inštancii databázového servera, použijeme príkaz

```
sqlcmd -S Server\Instance
```

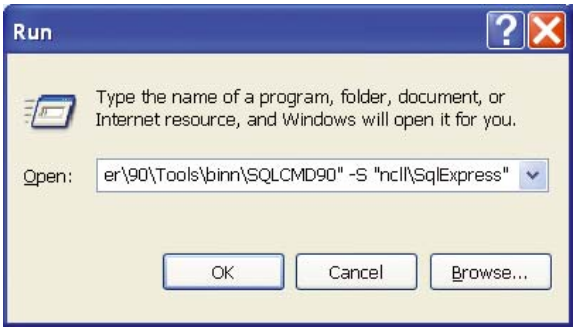
v našom prípade bol názov počítača ncll a názov inštancie implicitný SQLExpress, takže konkrétny príkaz bol

```
sqlcmd -S ncll\SQLExpress
```

Klientskú konzolovú aplikáciu SQLCMD90 je možné spustiť aj pomocou menu RUN operačného systému Windows

"C:\Program Files\Microsoft SQL Server\90\Tools\binn\SQLCMD90" -S "ncll\SqlExpress"

kde ncll bol v našom prípade názov lokálneho počítača

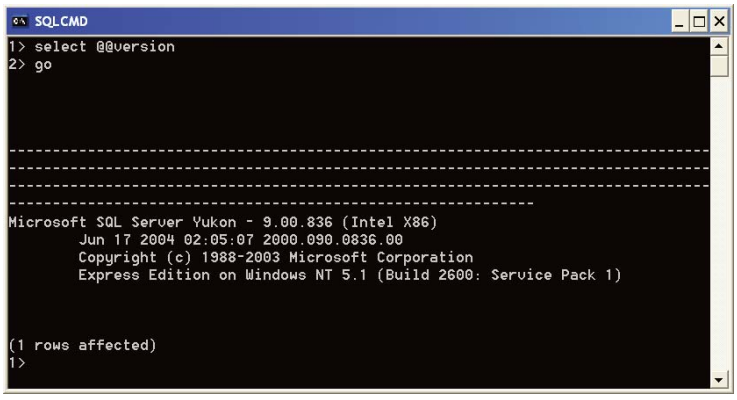


Spustenie klientskej konzolovej aplikácie pomocou menu RUN operačného systému Windows

Ak máme na vývojárskom počítači nainštalované aj iné SQL servery, je dôležité aby sme boli pripojení k správnej inštancii. Ak sme pripojení k inému serveru ľahko môžu nastať na prvý pohľad nevysvetliteľné situácie, kedy pomocou príkazov z konzoly (v inej databáze pod správou iného servera) vytvoríme a naplníme cvičné databázové tabuľky a z aplikácie ich potom nevidíme. Pre zistenie či sme pripojení k „Express Edition“ v konzolovej aplikácii napíšeme príkaz

```
select @@version

go
```

V okne klientskej konzolovej aplikácie bude vypísané



Informácie o verzii databázového servera

Ak sa vo výpise objaví Express Edition, sme pripojení k inštancii tohto správneho databázového enginu.

```
Microsoft SQL Server Yukon - 9.00.836 (Intel X86)
        Jun 17 2004 02:05:07 2000.090.0836.00
        Copyright (c) 1988-2003 Microsoft Corporation
        Express Edition on Windows NT 5.1 (Build 2600: Service Pack 1)
```

príkazy potvrdzujeme pomocou GO.

Zatiaľ sme si nerobili starosti, do ktorej databázy budú tieto údaje ukladané. Konzola sa ak nie je špecifikované inak pripojí k databáze MASTER. S touto databázou, kam sa ukladajú systémové údaje by sme v aplikáciách pracovať nemali, preto si vytvoríme novú databázu s názvom test.

Novú databázu vytvoríme príkazom

CREATE DATBASE TEST;
go

Teraz prepnime konzolovú aplikáciu do novovytvorenej databázy.

USE TEST;
go

Prácu s databázou pomocou j konzolovej aplikácie sqlcmd si môžeme vyskúšať na jednoduchom príklade

```
CREATE TABLE filmy
(
   cislo  INT PRIMARY KEY,
   nazov  VARCHAR(40),
   predstavitel  VARCHAR (10),
   rok    INT
);
```

potvrdíme príkazom GO

```
INSERT INTO filmy VALUES (1, "Dr. No","Connery", 1962);
INSERT INTO filmy VALUES (2, "From Russia With Love", "Connery", 1963);
INSERT INTO filmy VALUES (3, "Goldfinger", "Connery", 1964);
INSERT INTO filmy VALUES (4, "Thunderball", "Connery", 1965);
INSERT INTO filmy VALUES (5, "You Only Live Twice", "Connery", 1967);
INSERT INTO filmy VALUES (6, "On Her Majesty's Secret Service", "Lazenby", 1969);
INSERT INTO filmy VALUES (7, "Diamonds Are Forever", "Connery", 1971);
INSERT INTO filmy VALUES (8, "Live And Let Die", "Moore", 1973);
INSERT INTO filmy VALUES (9, "The Man With The Golden Gun", "Moore", 1974);
INSERT INTO filmy VALUES (10, "The Spy Who Loved Me", "Moore", 1977);
INSERT INTO filmy VALUES (11, "Moonraker" , "Moore", 1979);
INSERT INTO filmy VALUES (12, "For Your Eyes Only", "Moore", 1981);
INSERT INTO filmy VALUES (13, "Octopussy", "Moore", 1983);
INSERT INTO filmy VALUES (14, "A View To A Kill", "Moore", 1985);
INSERT INTO filmy VALUES (15, "The Living Daylights", "Dalton", 1987);
INSERT INTO filmy VALUES (16, "Licence To Kill", "Dalton", 1989);
INSERT INTO filmy VALUES (17, "GoldenEye", "Brosnan", 1995);
INSERT INTO filmy VALUES (18, "Tomorrow Never Dies", "Brosnan", 1997);
INSERT INTO filmy VALUES (19, "The World Is Not Enough", "Brosnan", 1999);
INSERT INTO filmy VALUES (20, "Die Another Day", "Brosnan", 2002);
```

dávku príkazov znovu potvrdíme príkazom GO. O tom, že sú údaje ukladané do databázy máme len nepriamy dôkaz, zobrazí sa séria potvrdzujúcich hlásení

```
(1 rows affected)
```

Priamym dôkazom bude výpis vložených údajov. Pre tento účel použijeme príkaz SELECT

```
select * from filmy;
```

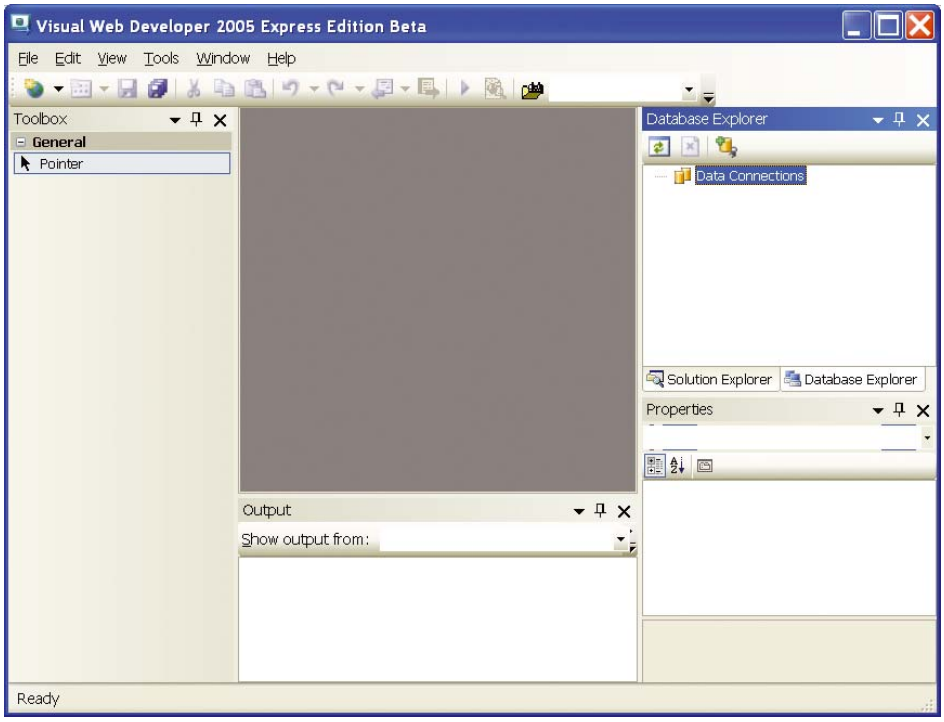
aj tento príkaz potvrdíme pomocou GO

```
cislo       nazov                                  predstavitel          rok
----------- -------------------------------------- ------------------------------
          1 Dr. No                                 Connery               1962
          2 From Russia With Love                  Connery               1963
          3 Goldfinger                             Connery               1964
          4 Thunderball                            Connery               1965
          5 You Only Live Twice                    Connery               1967
          6 On Her Majesty's Secret Service        Lazenby               1969
          7 Diamonds Are Forever                   Connery               1971
          8 Live And Let Die                       Moore                 1973
          9 The Man With The Golden Gun            Moore                 1974
         10 The Spy Who Loved Me                   Moore                 1977
         11 Moonraker                              Moore                 1979
         12 For Your Eyes Only                     Moore                 1981
         13 Octopussy                              Moore                 1983
         14 A View To A Kill                       Moore                 1985
         15 The Living Daylights                   Dalton                1987
         16 Licence To Kill                        Dalton                1989
         17 GoldenEye                              Brosnan               1995
         18 Tomorrow Never Dies                    Brosnan               1997
         19 The World Is Not Enough                Brosnan               1999
         20 Die Another Day                        Brosnan               2002
```

Výpis údajov z databázy pomocou klientskej konzolovej aplikácie

Práca s databázou pomocou Visual Web Developera

Pre prvé pokusy s databázovým serverom nám stačí spustiť vývojové prostredie Visual Web Developer 2005 Express Edition. Bez toho aby sme otvárali alebo vytvárali nejakú aplikáciu si v pravej hornej časti okna pracovnej obrazovky vývojového prostredia všimnime záložky Solution Explorer a Database Explorer. Po prepnutí do Database Explorera uvidíme v okne prázdnu (zatiaľ) zložku Data Connections

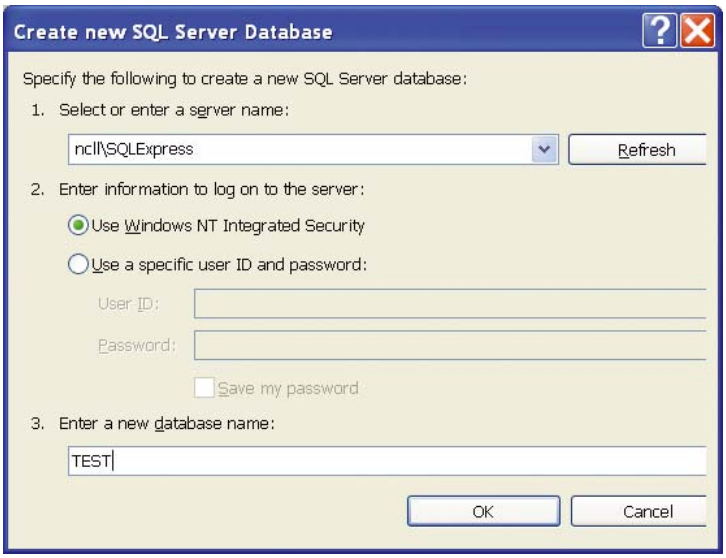


Visual Web Developer – zložka Database Explorer

V kontextovom menu tejto zložky sú pri prázdnom projekte dostupné dve položky:

- Add Connection
- Create New SQL Server Database

Výhodná je aj možnosť vytvorenia novej databázy pod správou SQL Servera 2005 Express Edition.



Dialóg pre vytvorenie novej databázy

Ak sa chceme pripojiť ku konkrétnej inštancii databázového servera, použijeme príkaz

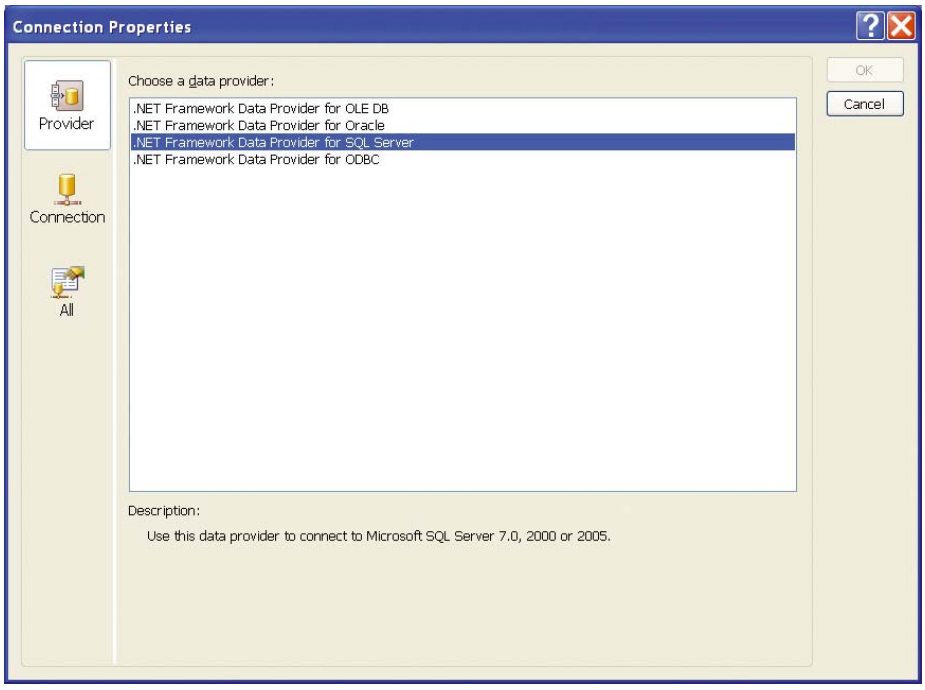
```
sqlcmd -S Server\Instance
```

s uvedením kompletnej cesty to bude

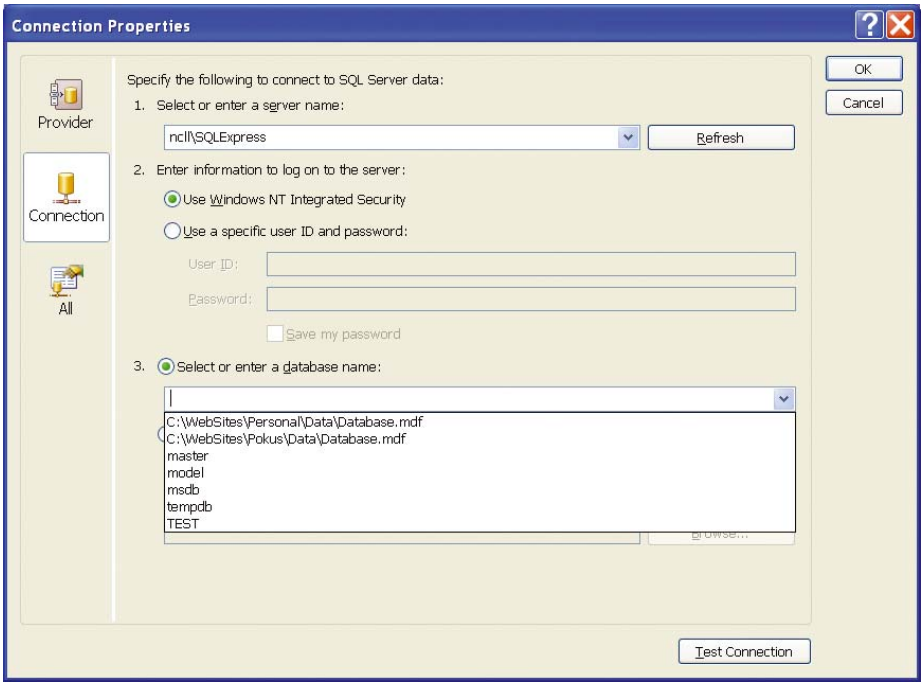
```
C:\Program Files\Microsoft SQL Server\90\Tools\binn\sqlcmd -S ncll\SQLExpress
```

Potom novú databázu vytvoríme príkazom

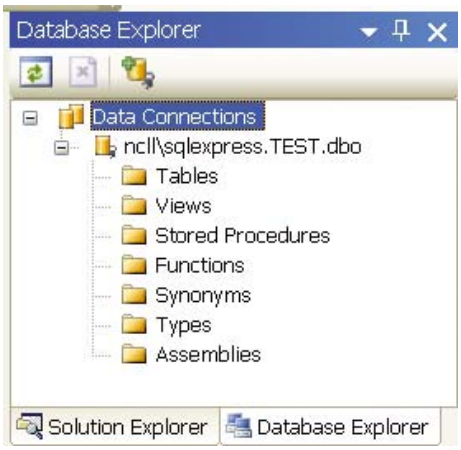
```
CREATE DATBASE TEST;
go
```



Dialóg Connection Properties – výber providera pre prístup k údajom



Dialóg Connection Properties – výber databázy



Okno Database Explorer – pripojenie k databáze TEST

Návrh databázovej tabuľky pomocou Visual Web Developera

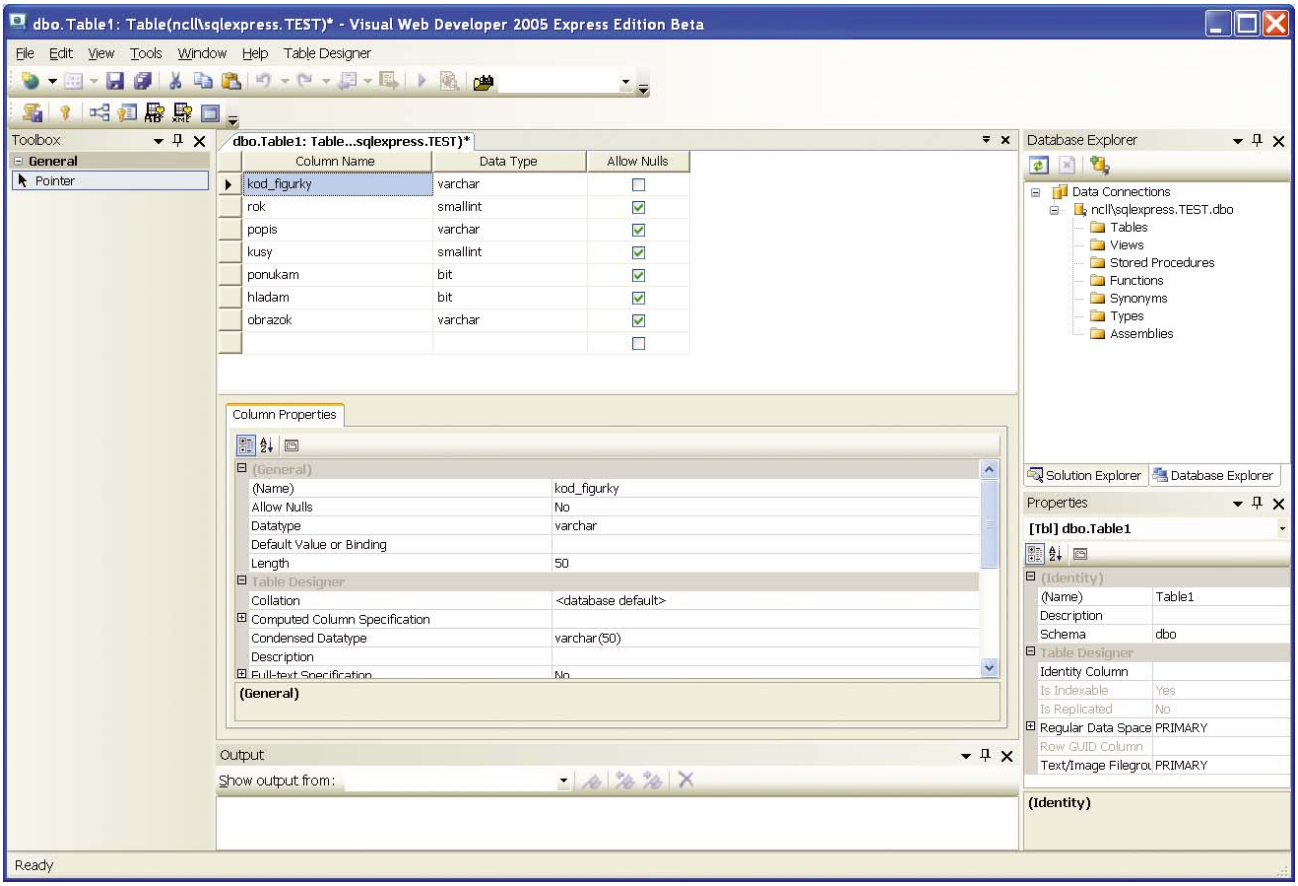
Po vytvorení pripojenia na databázu, môžeme zložku s názvom tohto pripojenia rozvinúť. Najviac nás asi bude zaujímať podzložka Tables. Je logické, že v novovytvorenej satabáze zatiaľ nie sú žiadne databázové tabuľky. Pomocou kontextového menu na zložku Tables zvolíme možnosť Add New Table. V strednom okne pracovnej obrazovky vývojového prostredia sa zobrazí grafické návrhové okno pre vytvorenie novej tabuľky.

Ako príklad jednoduchej webovej databázovej aplikácie sme vybrali námet z oblasti hobby aplikáciu zberateľa ručne maľovaných figúrok z vajíčok Kinder Surprise. Pre takúto aplikáciu bude vhodná (samozrejme len pre prvé pokusy) databázová tabuľka s nasledovnou štruktúrou

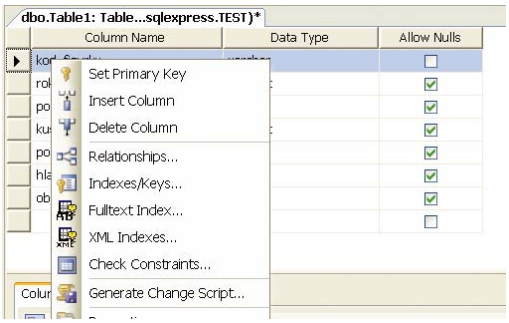
[kod_figurky]	[varchar]
[rok]	[smallint]
[popis]	[varchar]
[kusy]	[smallint]
[ponukam]	[bit]
[hladam]	[bit]
[obrazok]	[varchar]

Jednotlivé stĺpce tabuľky navrhujeme z ohľadom na logiku fungovania aplikácie. Pri zberateľských stránkach bez ohľadu na to, či sa jedná o známky, odznaky, mince, figúrky a rôzne iné predmety bude mať databáza viac menej rovnakú štruktúru. Každý jednotlivý predmet bude mať v databáze svoj záznam, ktorý okrem identifikácie predmetu obsahuje jeho popis, rok kedy bol daný predmet vyrobený, potom príznaky, či zberateľ tento predmet ponúka na výmenu, prípadne ho zháňa, a prípadne informáciu o tom, koľko kusov má zberateľ vo svojej zbierke.

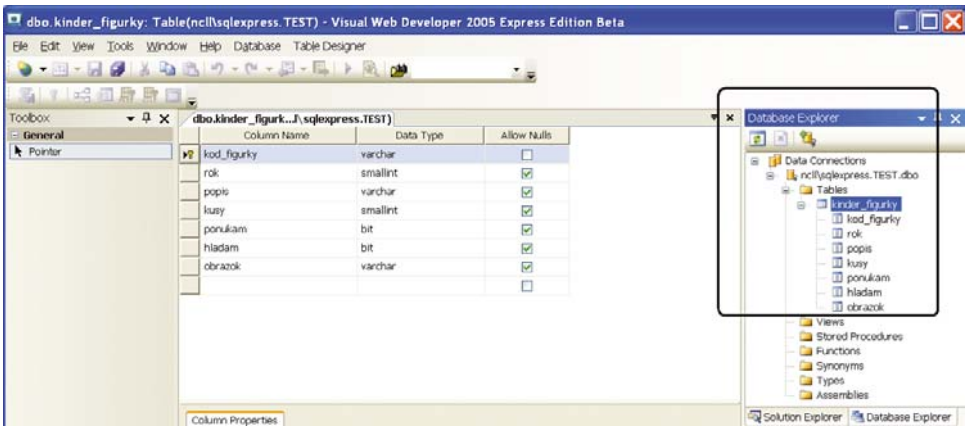
Podľa predbežného návrhu vyplníme údaje pre jednotlivé stĺpce novovytváranej tabuľky



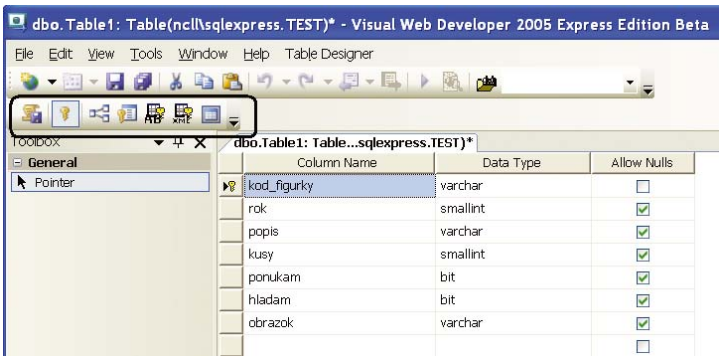
Vytvorenie novej databázovej tabuľky



Kontextové menu pre upresnenie návrhu stlpcov



Databázová tabuľka v okne Database Explorer



Tlačidlá pre návrh databázových štruktúr

Tlačidlá majú (zľava doprava) nasledovný význam
- Generovanie skriptu pre vykonanie zmeny vyplývajúcej z návrhu
- Nastavenie primárneho kľúča
- Relačné vzťahy v databáze
- Správa indexov a kľúčov
- Správa XML indexov
- Správa obmedzení (constraints)

Všimnime si funkciu tlačidla „Generovanie skriptu pre vykonanie zmeny vyplývajúcej z návrhu“ (prvé zľava) Po jeho aktivovaní sa zobrazí postupnosť SQL príkazov ktoré zmeny nami vykonané v etape návrhu zapíšu do databázy. Pre návrh tabuľky z predchádzajúceho odseku bol vygenerovaný skript

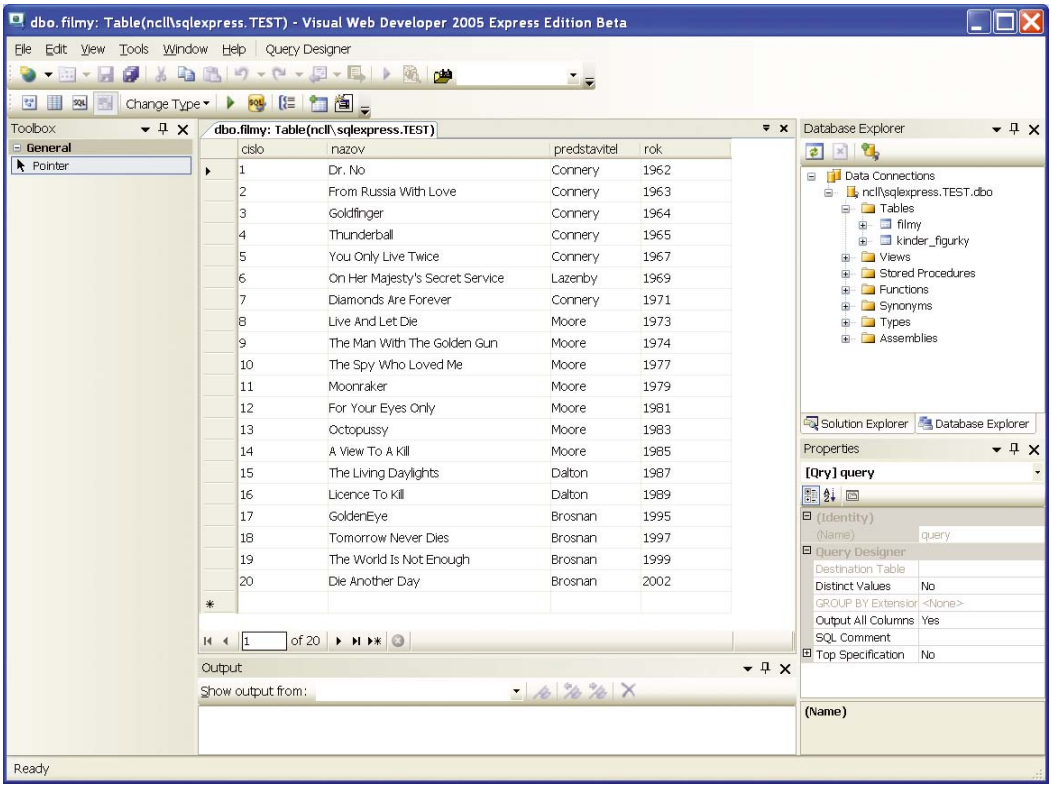
```
BEGIN TRANSACTION
SET QUOTED_IDENTIFIER ON
SET ARITHABORT ON
SET NUMERIC_ROUNDABORT OFF
SET CONCAT_NULL_YIELDS_NULL ON
SET ANSI_NULLS ON
SET ANSI_PADDING ON
SET ANSI_WARNINGS ON
COMMIT
BEGIN TRANSACTION
CREATE TABLE dbo.Table1
      (
      kod_figurky varchar(50) NOT NULL,
      rok smallint NULL,
      popis varchar(50) NULL,
      kusy smallint NULL,
      ponukam bit NULL,
      hladam bit NULL,
      obrazok varchar(50) NULL
      )  ON [PRIMARY]
GO
ALTER TABLE dbo.Table1 ADD CONSTRAINT
      PK_Table1 PRIMARY KEY CLUSTERED
      (
      kod_figurky
      ) WITH( STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]

GO
COMMIT
```

Ak zatvoríme návrhové okno v strednej časti pracovnej obrazovky grafické návrhové prostredie nám navrhne uloženie zmien do databázy. V našom prípade sme vytvárali novú databázovú tabuľku. Túto samozrejme nejako pomenujeme

Pomenovanie novej databázovej tabuľky

Výpis údajov z databázovej tabuľky

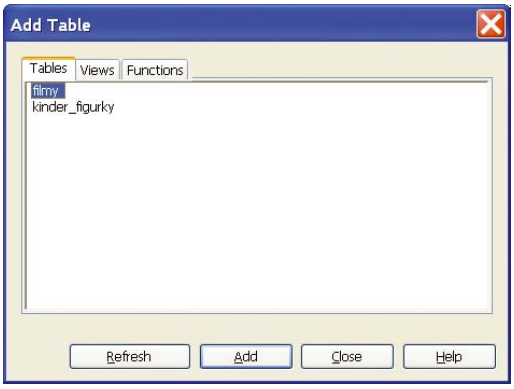
Databázový explorer integrovaný do Visual Web Developera nám poslúži pri väčšine úkonov pri práci s údajmi v databáze, samozrejme aj pri prehliadaní údajov. Najjednoduchšou metódou je výpis všetkých údajov z databázovej tabuľky. Dosiahneme to z kontextového menu príslušnej databázovej tabuľky pomocou funkcie Show table data. Napríklad pre tabuľku obsahujúcu 20 záznamov o filmoch, o štyroch stĺpcoch je to ideálna metóda.



Výpis údajov z databázovej tabuľky

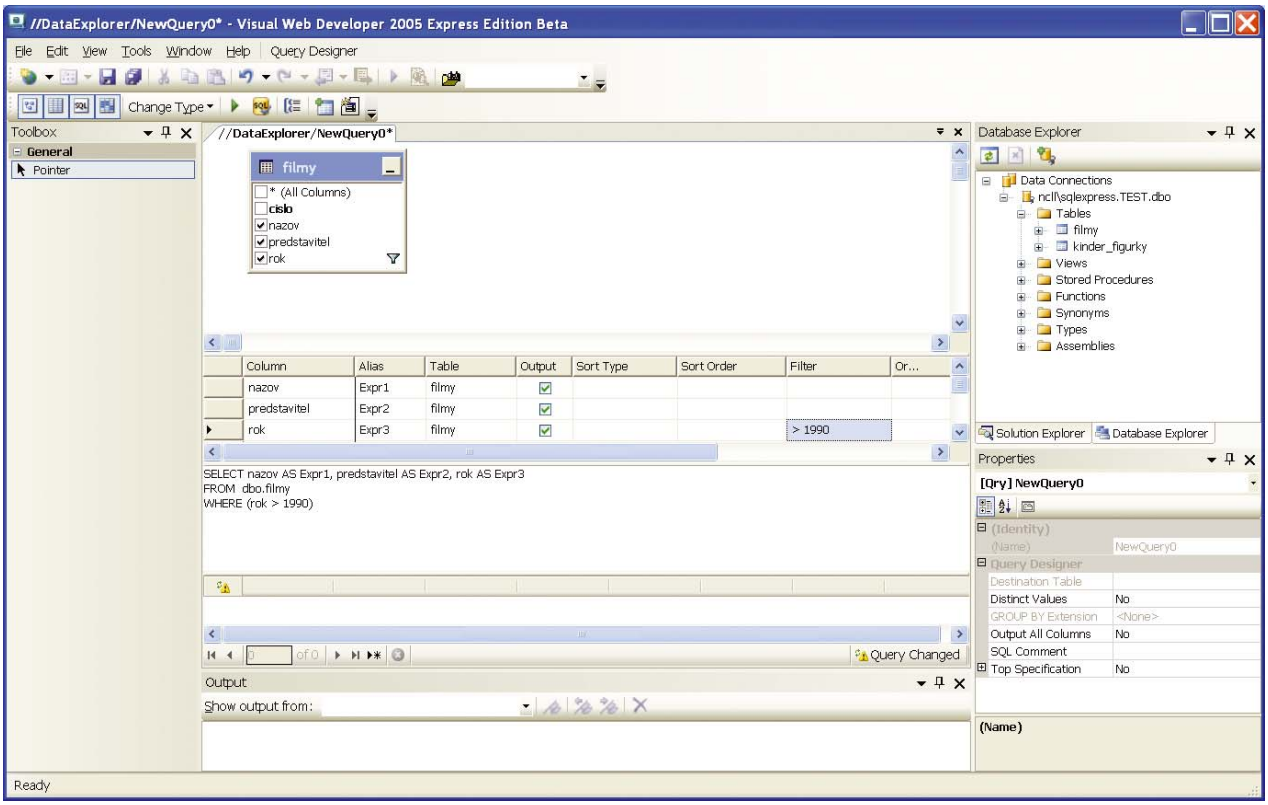
pre veľké tabuľky je takýto priamy a jednoduchý výpis prakticky nepoužiteľný. Ak by aj klientská aplikácia v spolupráci s databázovým serverom takéto veľké množstvo údajov v rozumnom čase vypísať dokázala, pre človeka to nemá žiadny význam, v tak veľkom množstve údajov sa ani nezorientuje, nie že by tam dokonca dokázal niečo vyhľadať. Množinu údajov z databázy môžeme presnejšie špecifikovať niekoľkými spôsobmi. Najčastejšie sa používa projekcia, reštrikcia a spájanie tabuliek. Pomocou **projekcie** môžeme množinu údajov obmedziť len na určité stĺpce (atribúty), ktoré sú v generovanom výpise predmetom nášho záujmu. Pod pojmom **reštrikcia** rozumieme výber definovaných záznamov. Pomocou príkazu SELECT môžeme presne špecifikovať, ktoré záznamy z tabuľky potrebujeme vybrať. V praxi sa veľmi často sa používa kombinácia projekcie a reštrikcie, kedy vypíšeme len určené stĺpce z vybraných záznamov

Preto Visual Web Developer obsahuje nástroje pre zadávanie SQL dopytu. Pokročilejší používatelia zadajú priamo textový reťazec s dopytom. Začiatočníci určite ocenia grafické návrhové prostredie pre tvorbu dopytu. Sprievodca vytvorením dopytu nám najskôr ponúkne dialóg pre výber tabuliek. Dopyt totiž môžeme vytvárať nad jednou, prípadne nad viacerými tabuľkami. Pre náš nasledujúci príklad sme vybrali námet kedy potrebujeme vypísať údaje o filmoch, teda názov filmu, priezvisko predstaviteľa hlavného hrdinu a rok uvedenia filmu, pričom nás zaujímajú len filmy uvedené po roku 1990. Bude to jednoduchý dotaz nad jednou tabuľkou, preto v dialógu pre výber tabuliek vyberieme tabuľku filmy.



Pridávanie tabuliek do dotazu

Po výbere tabuliek, ktorých sa bude dotaz týkať za v strednom okne vývojového prostredia zobrazí niekoľko vodorovne oddelených okien. V hornom okne vyberieme stĺpce, ktorých hodnoty chceme vypísať, toto okno slúži pre definovanie projekcie. Vybrané stĺpce sa zobrazia v tabuľke v druhom okne zhora. Tam môžedme definovať pravidlá pre reštrikciu, teda podmienky výberu záznamov. V vašom prípade sme pre rok definovali filter > 1990.



Grafický návrh dotazu

V ďalšom okne zhora sa zobrazuje aktuálny SQL dotaz, ktorý je výsledkom doterajšieho návrhu

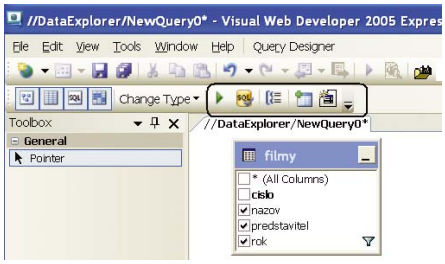
```
SELECT nazov AS Expr1, predstavitel AS Expr2, rok AS Expr3
FROM  dbo.filmy
WHERE (rok > 1990)
```

Pre prácu s dotazmi slúži tlačidlová lišta. Úplne vľavo sú tlačidlá pre zapínanie a vypínanie zobrazovania vodorovne rozdelených okien v strednej časti obrazovky. V poradí zľava doprava tlačidlá zapínajú a vypínajú zobrazovanie okien:

- diagram pre projekciu
- kritériá reštrikcie
- SQL dotaz v textovej podobe
- výsledky dopytovania

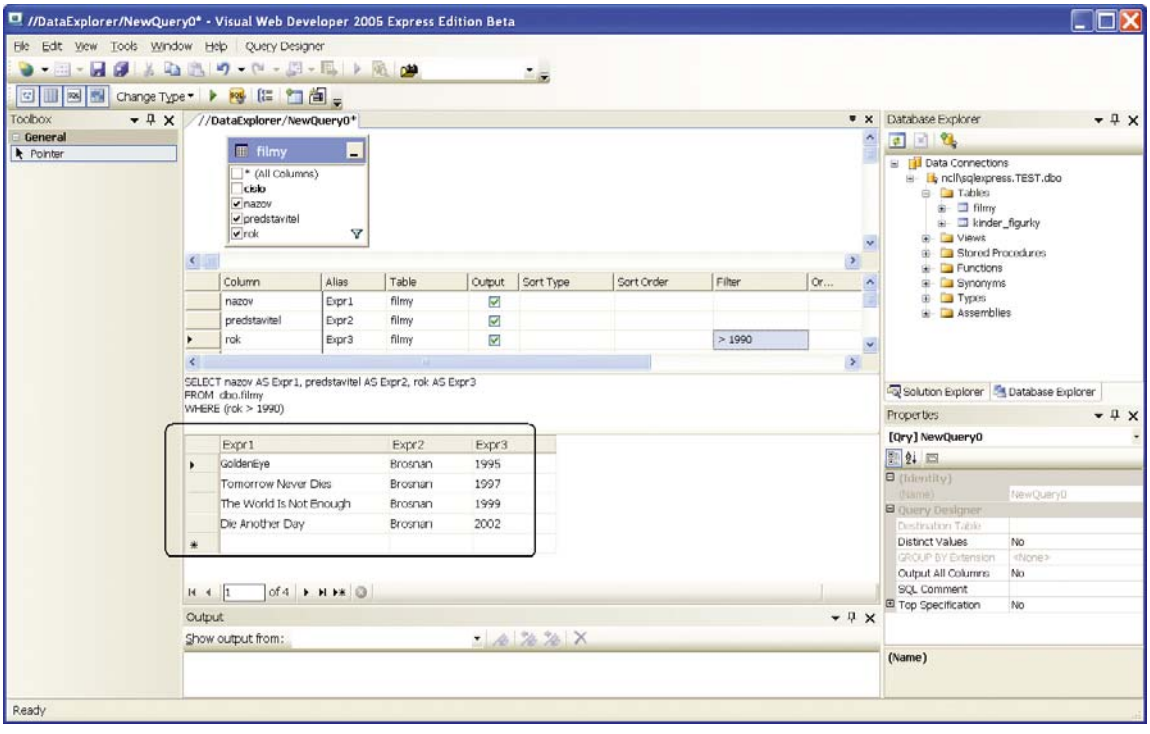
ďalšia skupina tlačidiel (na obrázku označená čiernym rámikom) slúži pre manipuláciu s dotazom. Tlačidlá majú (zľava doprava) tento význam:

- spustenie SQL dotazu
- kontrola syntaxe
- pridanie pravidiel pre zoskupovanie údajov (klauzula GROUP BY)
- pridanie novej tabuľky do dotazu
- pridanie novej odvodenej tabuľky



Tlačidlá pre SQL dotaz

po spustení dotazu sa v okne výsledkov zobrazia výsledky dotazovania.



Výpis výsledkov dopytovania

Kapitola 6:

Vývoj databázových aplikácií

Databázový server SQL Server 2005 Express Edition, ktorý predpokladáme použiť vo svojich databázových aplikáciách bol podrobnejšie popísaný v predchádzajúcej kapitole, takže nastal čas pre vývoj prvej cvičnej databázovej aplikácie. Nakoľko Visual Web Developer umožňuje vizuálny návrh jednoduchších databázových aplikácií, bez toho aby sme si zaťažovali hlavu prebytkom teórie, pustíme sa priamo do vizuálneho návrhu.

Prvá databázová aplikácia – prehliadanie tabuľky

Snáď len pripomenieme skript obsahujúci návrhovú štruktúru tabuľky filmov s ktorou budeme v aplikácii pracovať.

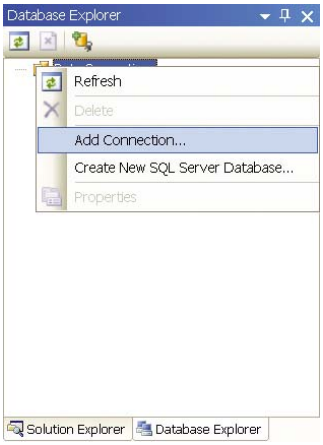
```
CREATE TABLE filmy
(
   cislo  INT PRIMARY KEY,
   nazov  VARCHAR(40),
   predstavitel  VARCHAR (10),
   rok    INT
);

INSERT INTO filmy VALUES (1, "Dr. No","Connery", 1962);
INSERT INTO filmy VALUES (2, "From Russia With Love", "Connery", 1963);
INSERT INTO filmy VALUES (3, "Goldfinger", "Connery", 1964);
INSERT INTO filmy VALUES (4, "Thunderball", "Connery", 1965);
INSERT INTO filmy VALUES (5, "You Only Live Twice", "Connery", 1967);
INSERT INTO filmy VALUES (6, "On Her Majesty's Secret Service", "Lazenby", 1969);
INSERT INTO filmy VALUES (7, "Diamonds Are Forever", "Connery", 1971);
INSERT INTO filmy VALUES (8, "Live And Let Die", "Moore", 1973);
INSERT INTO filmy VALUES (9, "The Man With The Golden Gun", "Moore", 1974);
INSERT INTO filmy VALUES (10, "The Spy Who Loved Me", "Moore", 1977);
INSERT INTO filmy VALUES (11, "Moonraker" , "Moore", 1979);
INSERT INTO filmy VALUES (12, "For Your Eyes Only", "Moore", 1981);
INSERT INTO filmy VALUES (13, "Octopussy", "Moore", 1983);
INSERT INTO filmy VALUES (14, "A View To A Kill", "Moore", 1985);
INSERT INTO filmy VALUES (15, "The Living Daylights", "Dalton", 1987);
INSERT INTO filmy VALUES (16, "Licence To Kill", "Dalton", 1989);
INSERT INTO filmy VALUES (17, "GoldenEye", "Brosnan", 1995);
INSERT INTO filmy VALUES (18, "Tomorrow Never Dies", "Brosnan", 1997);
INSERT INTO filmy VALUES (19, "The World Is Not Enough", "Brosnan", 1999);
INSERT INTO filmy VALUES (20, "Die Another Day", "Brosnan", 2002);
```

Východiskový stav je teda taký, že v databáze TEST máme uvedené údaje.

Vytvoríme novú aplikáciu typu ASP.NET Web Site (v našom prípade v jazyku C#). Po vytvorení aplikácie rešpektujeme dobré zvyky a v okne Solution Explorer vytvoríme podadresár Code, kam budú ukladané súbory zo zdrojovými kódmi. Po tomto prípravnom úkone zostaneme ešte v pravej hornej časti obrazovky vývojového prostredia, no okno Solution Explorer prepneme pomocou dáložky v dolnej časti na Database Explorer.

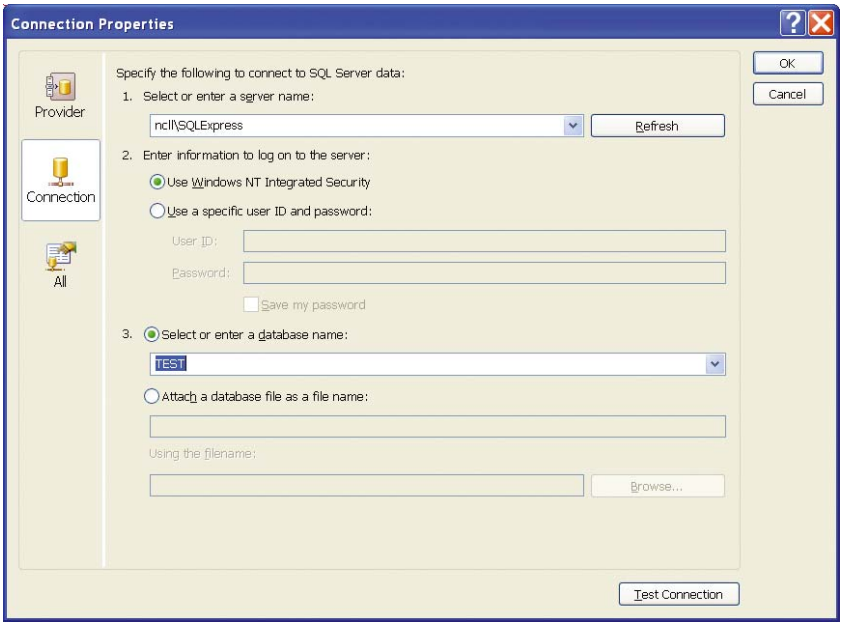
Vytvorenie pripojenia na databázu

V okne Database Explorer (ak sme dosiaľ žiadnu inú databázovú aplikáciu nevyvíjali) máme zatiaľ prázdnu zložku Data Connections . Pre pridanie priponenia na databázu TEST aktivujeme v kontextovom menu položku Add Connection.



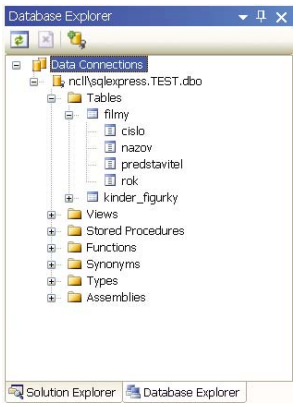
Database Explorer

V dialógu pre definovanie parametrov pripojenia najskôr nastavíme meno databázového servera v tvare meno_lokalneho_pocitaca\sqlexpress, v našom prípade ncll\sqlexpress. Použijeme integrovanú Windows NT bezpečnosť a vyberieme si databázu TEST.



Vytvorenie pripojenia na databázu

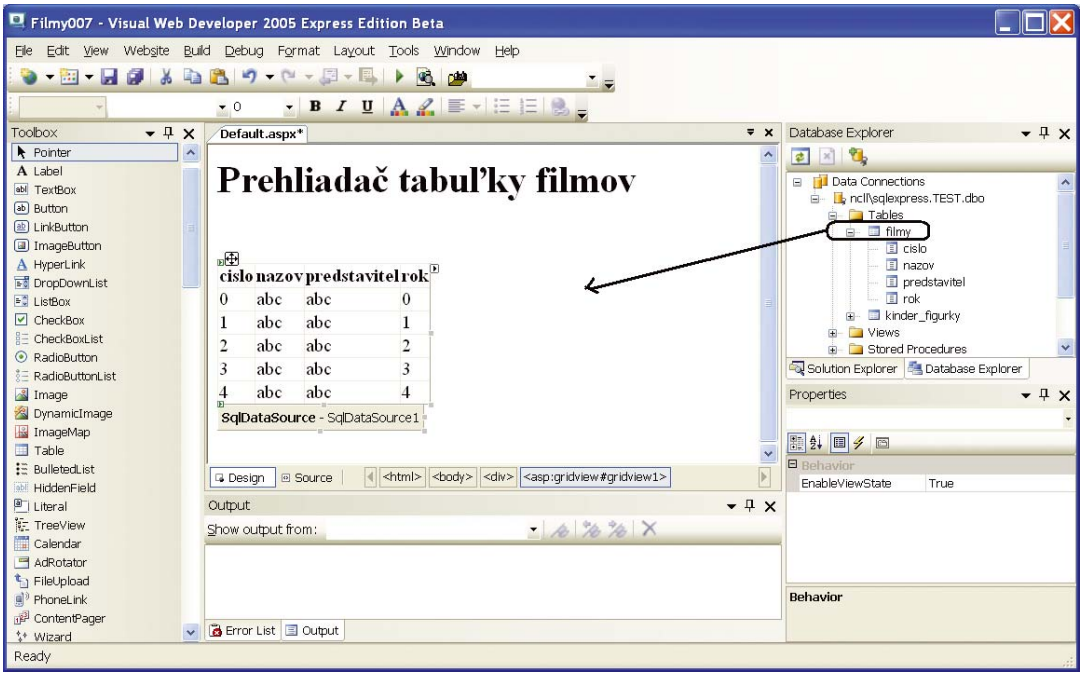
V prípade ak sa pripájame na inú databázu, prípadne sme databázu naplnili osobitne a metódou XCopy sme ju prekopírovali do novej destinácie, musíme tieto skutočnosti pri vytváraní pripojenia na zdroj údajov zohľadniť.



Okno Database Explorer po pripojení sa na konkrétnu databázu

Zobrazenie obsahu databázovej tabuľky

Pre vytvorenie najjednoduchšieho prehliadača postačí triviálny úkon. V okne Database Explorer klikneme na názov tabuľky a pri stlačenom tlačidle myši presunieme tento názov na plochu návrhového formulára (predtým sme tam vložili len nadpis). Túto jednoduchú metódu nazývame vo Windows *drag and drop*



Presun symbolu databázovej tabuľky a okna Database Explorer na návrhovú plochu webového formulára

Týmto jednoduchým úkonom sme na plochu aplikácie presunuli dve komponenty SqlDataSource a GridView.

Pre komponentu **SqlDataSource**, (nevizuálna komponenta v podobe malého šedého obdĺžnika pod databázovou tabuľkou), bol vygenerovaný kód

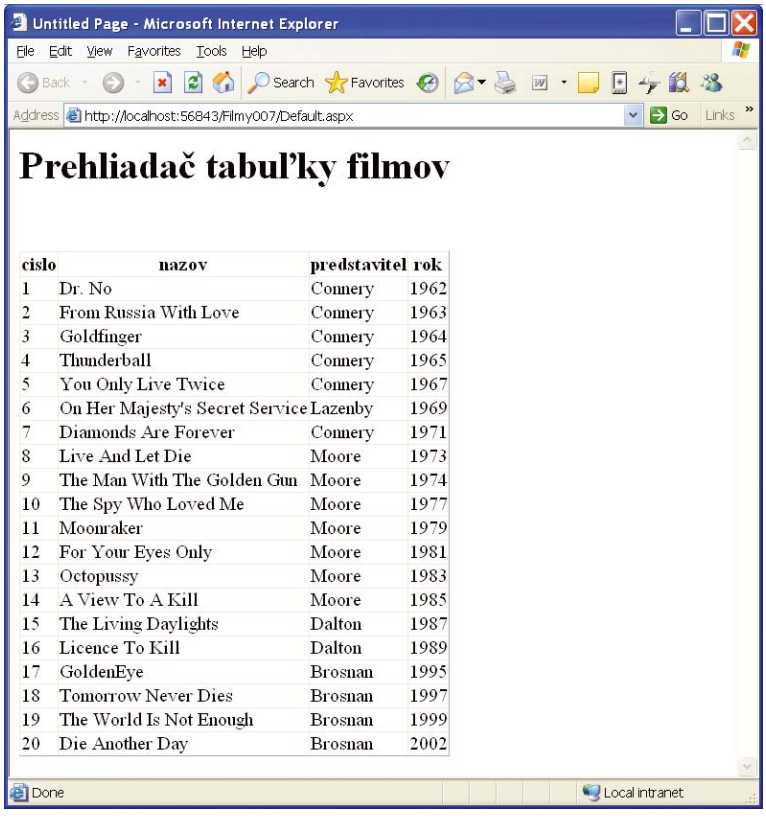
```
<asp:SqlDataSource ID="SqlDataSource1" Runat="server" ProviderName="<%$
ConnectionStrings:AppConnectionString1.ProviderName %>"
            UpdateCommand="UPDATE [filmy] SET [nazov] = @nazov, [predstavitel] =
@predstavitel, [rok] = @rok WHERE [cislo] = @original_cislo"
            InsertCommand="INSERT INTO [filmy] ([cislo], [nazov], [predstavitel],
[rok]) VALUES (@cislo, @nazov, @predstavitel, @rok)"
            DeleteCommand="DELETE FROM [filmy] WHERE [cislo] = @original_cislo"
SelectCommand="SELECT [cislo], [nazov], [predstavitel], [rok] FROM [filmy]"
            ConnectionString="<%$ ConnectionStrings:AppConnectionString1 %>">
            <DeleteParameters>
                <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
            </DeleteParameters>
            <InsertParameters>
                <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
                <asp:Parameter Type="String" Name="nazov"></asp:Parameter>
                <asp:Parameter Type="String"
Name="predstavitel"></asp:Parameter>
                <asp:Parameter Type="Int32" Name="rok"></asp:Parameter>
            </InsertParameters>
            <UpdateParameters>
                <asp:Parameter Type="String" Name="nazov"></asp:Parameter>
                <asp:Parameter Type="String"
Name="predstavitel"></asp:Parameter>
                <asp:Parameter Type="Int32" Name="rok"></asp:Parameter>
```

```
            <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
        </UpdateParameters>
    </asp:SqlDataSource>
```

Zobrazenie obsahu tabuľky má na starosti komponenta **GridView**. Je napojená na komponentu SqlDataSource a nastavením jej parametrov môžeme dosiahnuť zmenu zobrazenia tabuľky. Ak si pozrieme kompletný kód ktorý vývojové prostredie Visual Web Developer vygenerovalo po jednoduchom presune ikony databázovej tabuľky na pracovnú plochu aplikácie azda najlepšie oceníme jeho prínos.

```
<asp:GridView ID="GridView1" Runat="server" DataKeyNames="cislo"
DataSourceID="SqlDataSource1"
        EmptyDataText="There are no data records to display."
AutoGenerateColumns="False" AllowPaging="True" AllowSorting="True">
        <Columns>
            <asp:CommandField ShowDeleteButton="True" ShowEditButton="True"
ShowSelectButton="True"></asp:CommandField>
            <asp:BoundField ReadOnly="True" HeaderText="cislo"
DataField="cislo" SortExpression="cislo"></asp:BoundField>
            <asp:BoundField HeaderText="nazov" DataField="nazov"
SortExpression="nazov"></asp:BoundField>
            <asp:BoundField HeaderText="predstavitel"
DataField="predstavitel" SortExpression="predstavitel"></asp:BoundField>
            <asp:BoundField HeaderText="rok" DataField="rok"
SortExpression="rok"></asp:BoundField>
        </Columns>
    </asp:GridView>
```

Výsledkom bude databázová tabuľka zobrazená v okne klientovho prehliadača webových stránok

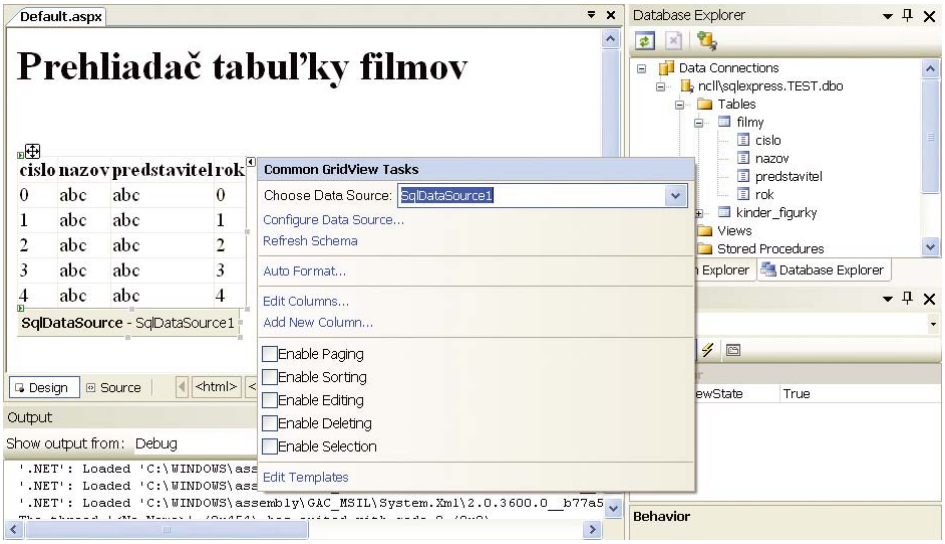


cislo	nazov	predstavitel	rok
1	Dr. No	Connery	1962
2	From Russia With Love	Connery	1963
3	Goldfinger	Connery	1964
4	Thunderball	Connery	1965
5	You Only Live Twice	Connery	1967
6	On Her Majesty's Secret Service	Lazenby	1969
7	Diamonds Are Forever	Connery	1971
8	Live And Let Die	Moore	1973
9	The Man With The Golden Gun	Moore	1974
10	The Spy Who Loved Me	Moore	1977
11	Moonraker	Moore	1979
12	For Your Eyes Only	Moore	1981
13	Octopussy	Moore	1983
14	A View To A Kill	Moore	1985
15	The Living Daylights	Dalton	1987
16	Licence To Kill	Dalton	1989
17	GoldenEye	Brosnan	1995
18	Tomorrow Never Dies	Brosnan	1997
19	The World Is Not Enough	Brosnan	1999
20	Die Another Day	Brosnan	2002

Zobrazenie databázovej tabuľky u klienta

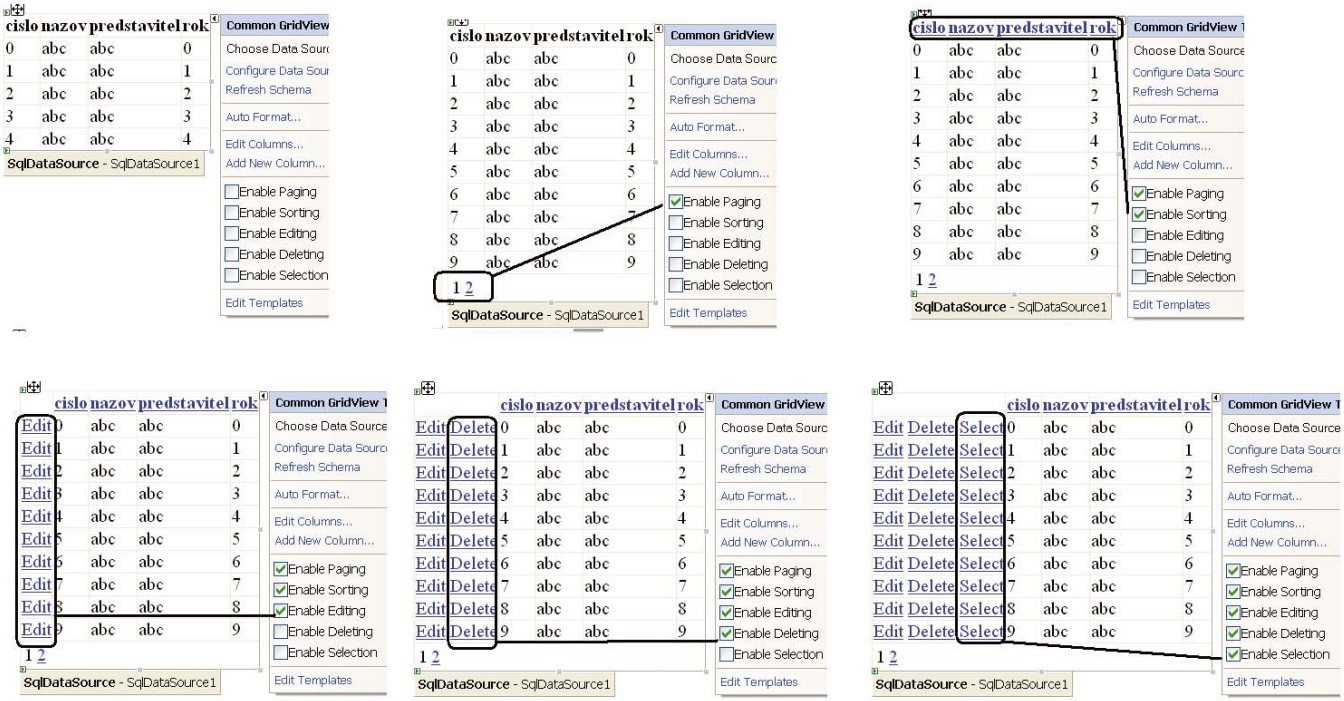
Ak si pamätníci starších (a aj mnohých súčastných) webových technológií, napríklad PHP a ASP spomenú, koľko kódu bolo potrebné napísať, aby bol dosiahnutý rovnaký výsledok - teda zobrazenie databázovej tabuľky v prehľadnej forme na HTML stránke. Naopak „migranti“ z Web Matrixu sa budú obzerať po možnosti stránkovania, utriedenia editovania údajov a podobne. Visual Web Developer tieto úkony ešte ďalej značne zjednodušil. Všimnime si u vizuálnej komponenty GridView malú šípku vpravo hore. Pomocou nej sa aktivuje Common Tasks menu pre danú komponentu. V tomto menu potom zaškrtnutím príslušných položiek aktivujeme jenotlivé funkcionality

– Enable Paging
– Enable Sorting
– Enable Editing
– Enable Selecting

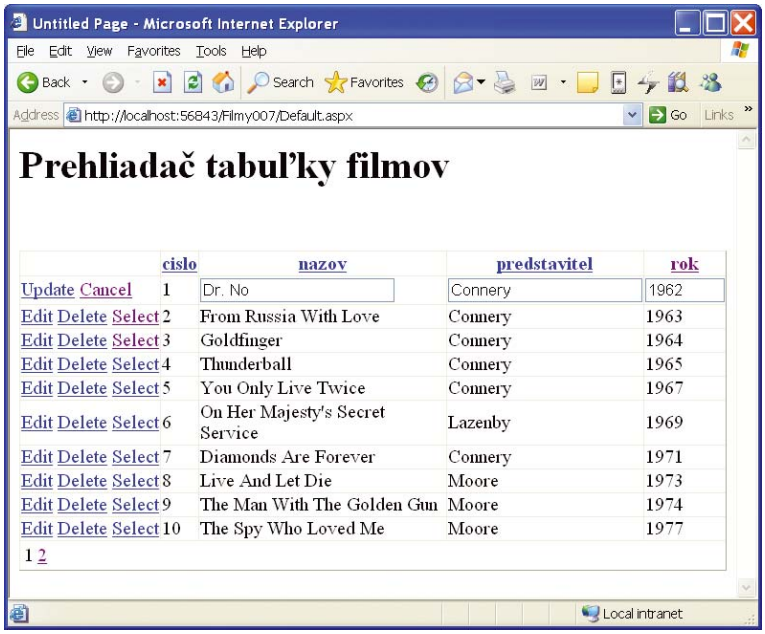


Common Tasks menu pre prvok Grid View

Pri aktivovaní jednotlivých volieb sa mení aj dizajn tabuľky pre zobrazenie údajov. Pribúdajú tam nové prvky pre aktiváciu spomenutých funkcií. Nasledujúci rozfázovaný obrázok ukazuje zmenu „vizuálu“ zobrazenia databázovej tabuľky pri postupnom zaškrtávaní jednotlivých funkcionalít

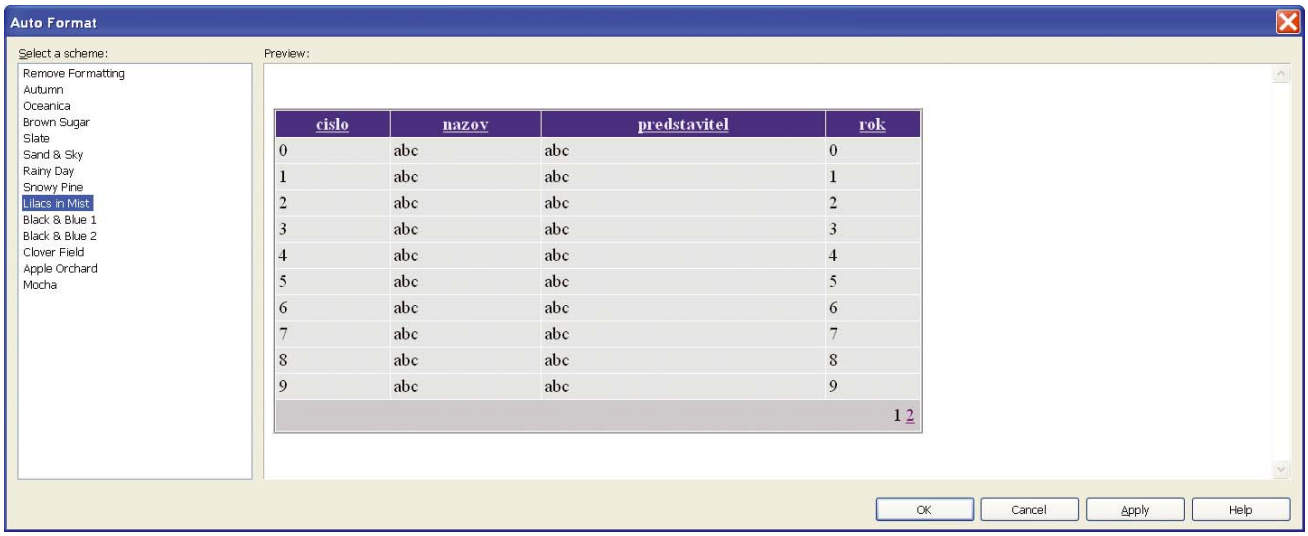


Aktivovanie vlastností GridView v menu Common Tasks



Zobrazenie databázovej tabuľky s rozšírenými vlastnosťami u klienta

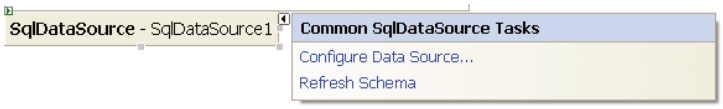
Pozornosť by sme mali venovať nielen funkčnosti, ale aj dizajnu. Ak je aplikácia dobre a esteticky navrhnutá, prispieva to k presnosti jej ovládania. Vzhľad komponenty GridView môžeme prispôsobiť dizajnu a farebnému vizuálu zbytku stránky pomocou Auto Formatu (menu Common Tasks)



Úprava dizajnu komponenty GridView

Práca so zdrojom údajov

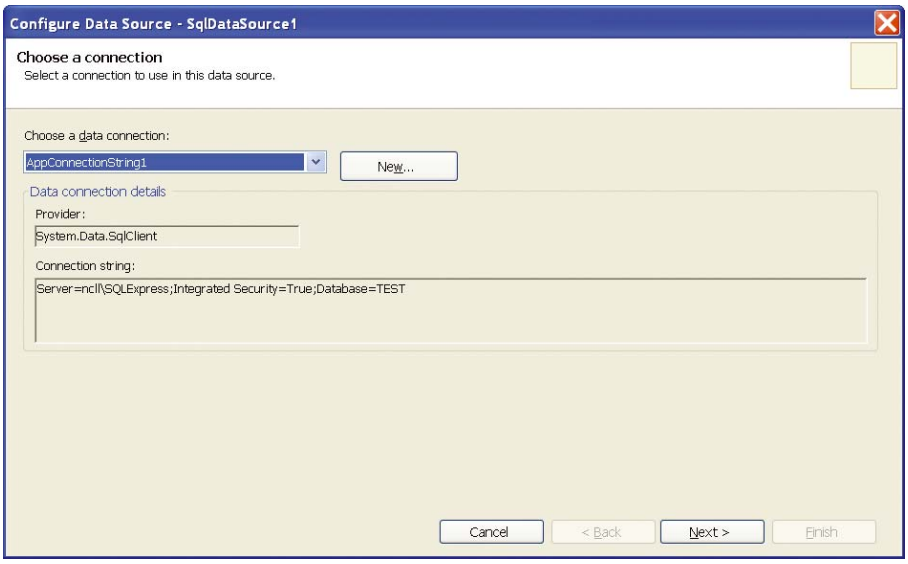
Podobne ako u vizuálnej komponenty GridView , aj u vizuálnej komponenty DataSource môžeme aktivovať Common Tasks menu (malú šípku vpravo hore od symbolu komponenty).



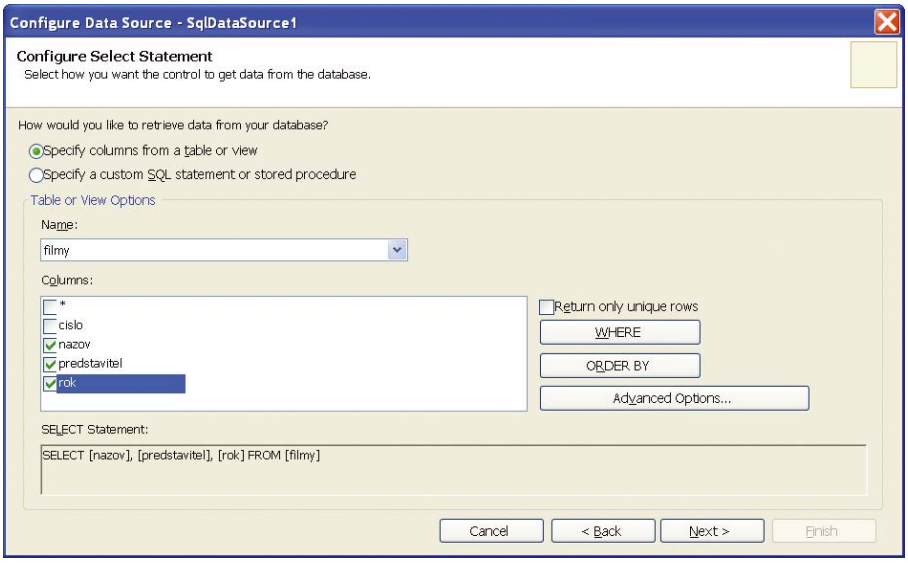
Common Tasks menu pre prvok SqlDataSource

Ak aktivujeme voľbu Configure Data Source, zobrazí sa dialóg pre vytváranie a editovania konfiguračného reťazca pre pripojenie sa k zdroju údajov. V našom prípade bol pripojovací reťazec konkrétne
Server=ncll\SQLExpress;Integrated Security=True;Database=TEST
Tento pripojovací reťazec je uložený v konfiguračnom súbore web.config v XML formáte

```
<connectionStrings>
  <add name="AppConnectionString1" connectionString="Server=ncll\SQLExpress;
    Integrated Security=True;Database=TEST" providerName="System.Data.SqlClient"/>
</connectionStrings>
```



Dialóg Configure Data Source

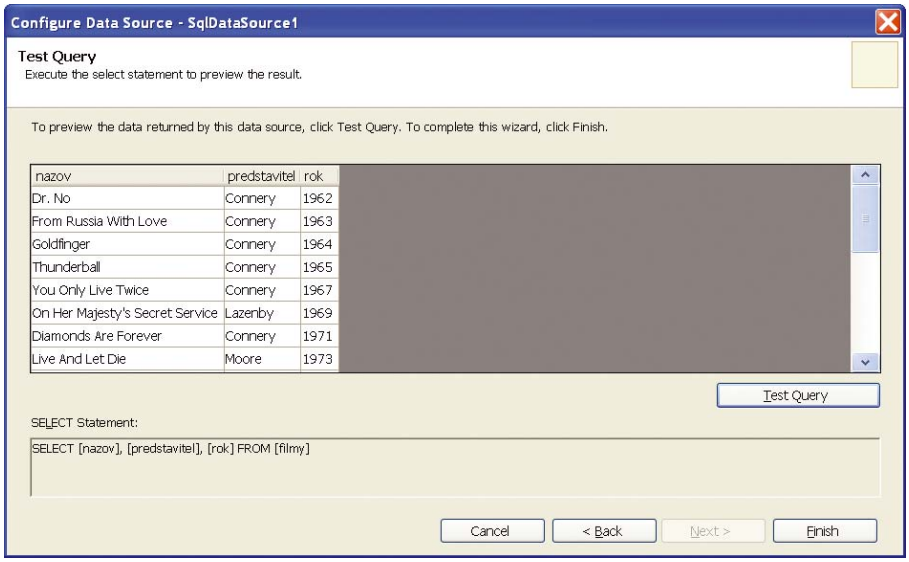


Dialóg pre návrh dotazu pre výber údajov

Zmenou pripojovacieho reťazca sa môžeme napríklad pripojiť na iný zdroj údajov. V tomto príklade sme spočiatku zobrazili databázovú tabuľku implicitne, to jest tak ako bola navrhnutá. Vo väčšine prípadov potrebujeme zobraziť len niektoré riadky a stĺpce, teda aplikovať kombináciu reštrikcie a projekcie, inými slovami vytvoriť SQL dotaz SELECT pomocou presne špecifikujeme, ktoré údaje nás zaujímajú. Ak v dialógu Configure Data Source klikneme na tlačidlo Next, môžeme v nasledujúcom dialógu takýto dotaz interaktívne vytvoriť. V našom prípade sme v zobrazení tabuľky vynechali stĺpec číslo.

Výsledkom návrhu bol SQL dotaz

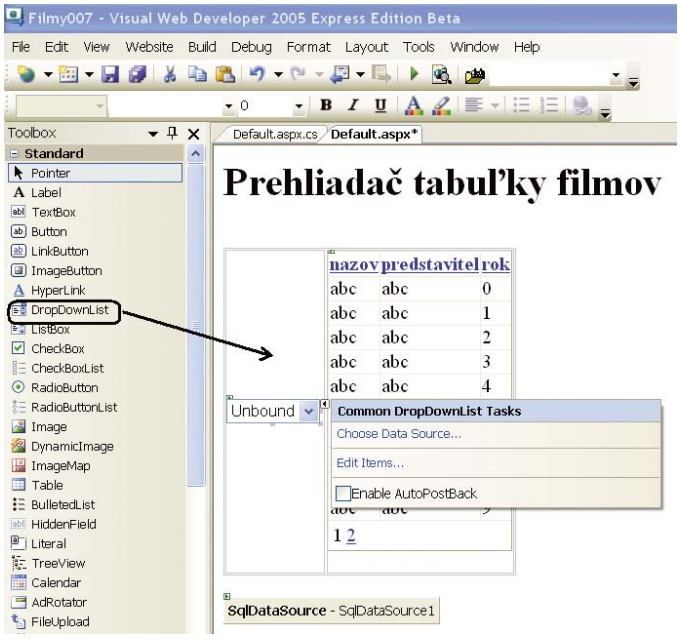
```
SELECT [nazov], [predstavitel], [rok] FROM [filmy]
```



Dialóg pre otestovanie navrhnutého dotazu

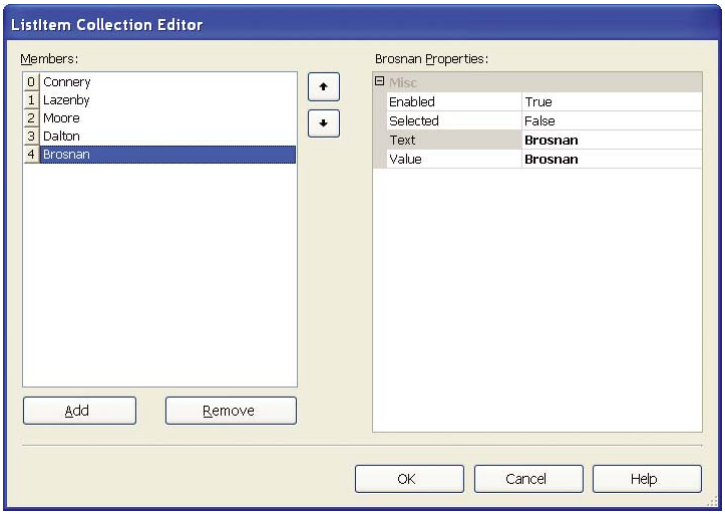
Interaktívny vyber zobrazovaných údajov

V predchádzajúcej stati sme ukázali možnosť výberu údajov pomocou SQL dotazu v etape návrhu aplikácie. často však chce kritériá pre výber údajov špecifikovať priamo používateľ, teda za behu aplikácie. Jednoducho, chce to určitú dynamiku. Naša jednoduchá tabuľka veľa možností neposkytuje, no nejaký jednoduchý príklad by sa vymyslieť predsa len dal. Trebárs by sme požadovali mať možnosť vybrať si herca a vypísať všetky filmy, v ktorých účinkoval. Možností pre zadávanie mena herca je mnoho, my sme vybrali komponentu DropDownList. Nakoľko je potrebné pracovať len s piatimi údajmi, zadáme ich ručne ako položky komponenty. Aby aplikácia niesla aj nejaké známky úpravy, vytvoríme na ploche formulára aplikácie HTML tabuľku, ktorá bude mať jeden riadok a dva stĺpce. Komponentu GridView metódou drag and drop jednoducho presunieme do druhého stĺpca tabuľky. Do prvého stĺpca (užšieho) umiestnime novú komponentu DropDownList.



Pridanie komponenty DropDownList pre špecifikovanie výberu údajov

V komponente DropDownList aktivujeme voľbu Enable AutoPostBack, čím zaistíme okamžitú reakciu na zmenu hodnoty prvku a ako jednotlivé položky zadáme priezviská hercov.



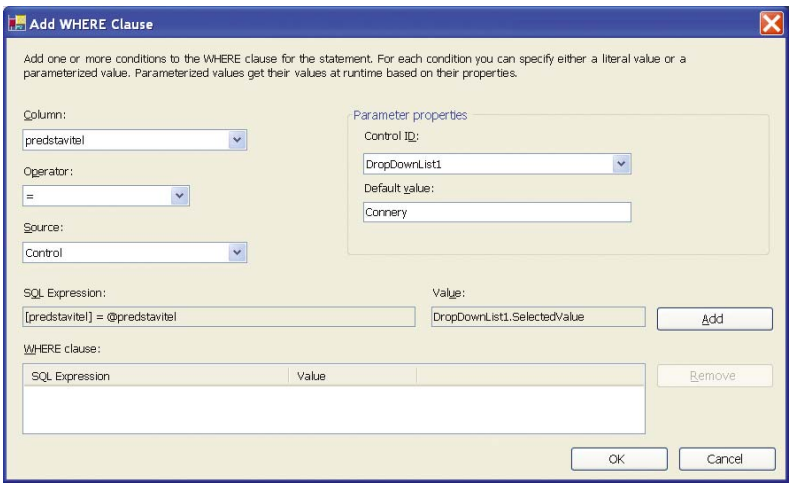
Zadanie položiek komponenty DropDownList

Takéto „statické“ naplnenie comboboxu má skôr nevýhody ako výhody. Hlavnou nevýhodou môže byť neaktuálnosť. Do filmotéky bondoviek pribudnú časom ďalšie prírastky a keďže Pierce Brosnan v ďalšej bondovke hrať odmietol, bude to niekto iný. Pri takto navrhnutej aplikácii by sme museli nielen pridávať údaje do databázy, ale čas od času aj modifikovať aplikáciu. Oveľa výhodnejšie bude naviazať ComboBox na databázu (Common Tasks menu) a zoznam hercov naplniť pomocou dopytu **`SELECT DISTINCT FROM dbo.filmy`**

Vráťme sa však k účelu Comboboxu. Už asi tušíte, že na základe zadaného priezviska herca sa bude zostavovať SQL dotaz pre výber filmov

```
SELECT * FROM [filmy] WHRERE predstavitel = 'Connery'
```

Vrátime sa preto k vizuálnej komponente DataSource, kde pomocou Common Tasks menu aktivujeme položku Configure Data Source. V Dialógu pre zostavovanie SQL dotazu nakoľko ideme konštruovať podmienku, zatlačíme tlačidlo Where, čím aktivujeme dialóg Add WHERE Clause. V tomto dialógu budeme zostavovať podmienku za klauzulou WHERE, teda konštruovať reťazec, na základe hodnoty zadanej v komponente DropDownList `predstavitel = 'ZADANA_HODNOTA'`
V dialógu pre zostavenie podmienky v klauzule preto nastavíme názov stĺpca ´predstaviteľ´, hodnotu budeme testovať na rovnosť (operátor =) voči hodnote ovládacieho prvku (Source: Control). Ak sme nezadlai inak, implicitné ID ovládacieho prvku budeDropDownList1. Aby sa nám na začiatku pri prvom načítaní stránky zobrazili nejaké údaje nastavíme aj implicitnú hodnotu na priezvisko nejakého herca.

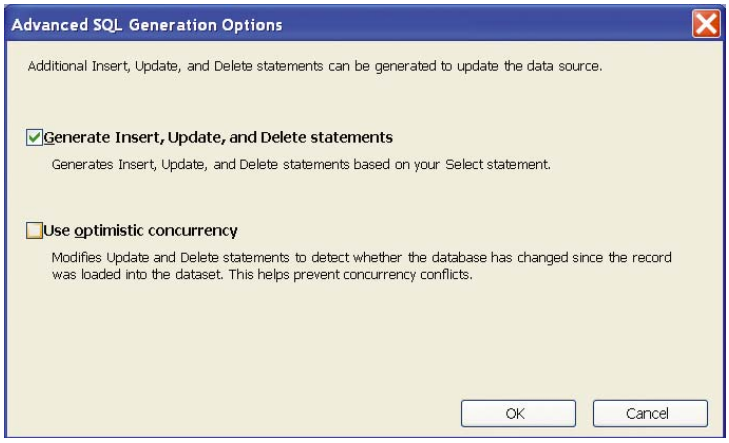


Zostavenie podmienky pre špecifikovanie výberu údajov

V dialógu sme naznačeným postupom skonštruovali SQL dotaz s parametrom `@predstavitel`.

```
SELECT * FROM [filmy] WHERE ([predstavitel] = @predstavitel)
```

Aby sprievodca (wizard) vygeneroval aj príslušné príkazy pre akcie Insert, Update a Delete, pomocou tlačidla Advanced Option nastavíme vygenerovanie týchto príkazov

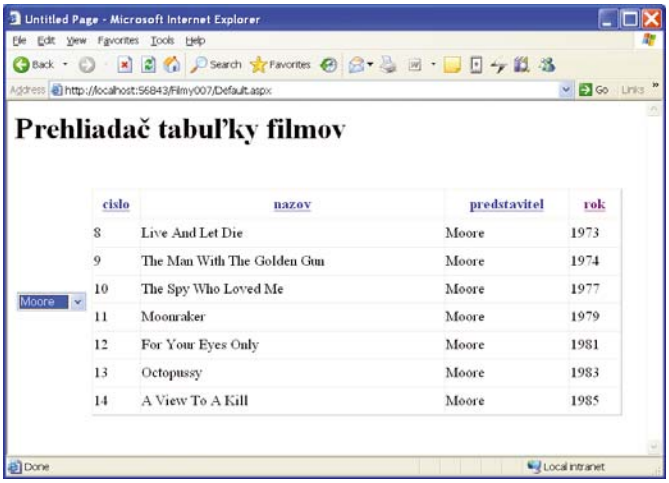


Pridanie komponenty DropdownList pre špecifikovanie výberu údajov

Po preklade a zostavení aplikácie si v okne zdrojového kódu môžeme pozrieť vygenerované reťazce

```
<asp:SqlDataSource ID="SqlDataSource1" Runat="server"
  SelectCommand="SELECT * FROM [filmy] WHERE ([predstavitel] = @predstavitel)"
  ConnectionString="<%$ ConnectionStrings:AppConnectionString1 %>"
  DeleteCommand="DELETE FROM [filmy] WHERE [cislo] = @original_cislo"
InsertCommand="INSERT INTO [filmy] ([cislo], [nazov], [predstavitel], [rok])
VALUES (@cislo, @nazov, @predstavitel, @rok)"
UpdateCommand="UPDATE [filmy] SET [nazov] = @nazov, [predstavitel] =
@predstavitel, [rok] = @rok WHERE [cislo] = @original_cislo">
            <DeleteParameters>
                <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
            </DeleteParameters>
            <UpdateParameters>
                <asp:Parameter Type="String" Name="nazov"></asp:Parameter>
                <asp:Parameter Type="String" Name="predstavitel"></asp:Parameter>
                <asp:Parameter Type="Int32" Name="rok"></asp:Parameter>
                <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
            </UpdateParameters>
            <SelectParameters>
                <asp:ControlParameter Name="predstavitel" DefaultValue="Connery"
Type="String" ControlID="DropDownList1"
                    PropertyName="SelectedValue"></asp:ControlParameter>
            </SelectParameters>
            <InsertParameters>
                <asp:Parameter Type="Int32" Name="cislo"></asp:Parameter>
                <asp:Parameter Type="String" Name="nazov"></asp:Parameter>
                <asp:Parameter Type="String" Name="predstavitel"></asp:Parameter>
                <asp:Parameter Type="Int32" Name="rok"></asp:Parameter>
            </InsertParameters>
</asp:SqlDataSource>
```

Teraz samozrejme už nič nebráni tomu aby sme aplikáciu spustili a otestovali



Spustenie aplikácie

Kapitola 7:
Autentifikácia a autorizácia

Pokiaľ sa pohybujeme vo sférach verejných a snažíme sa pomocou webovej aplikácie spropagovať svoju spoločnosť, svoje hobby (prípadne úchylku), zverejňujeme katalóg produktov a podobne, teda keď mottom aplikácie je zverejňovať a zviditeľňovať, otázkou autentifikácie a autorizácie sa vôbec nemusíme trápiť. Skôr naopak, snažíme sa aby naša aplikácia bola na zoznamoch webu, aby si k nej našli cestu vyhľadávacie portály a podobne. Tento trend platí (snáď z výnimkou teroristov, nacionalistov, galerky a iných živlov) pre všetky aplikácie. No vo veľa prípadoch sa potrebujeme na stránkach aplikácie presunúť do „privátnej zóny“ a tam napríklad pri internet bankingu vykonať požadované transakcie, v internetovom obchode si niečo záväzne objednať, prípadne to pomocou kreditnej karty aj zaplatiť, prečítať si na mailovom portáli svoje vlastné maily a podobne.

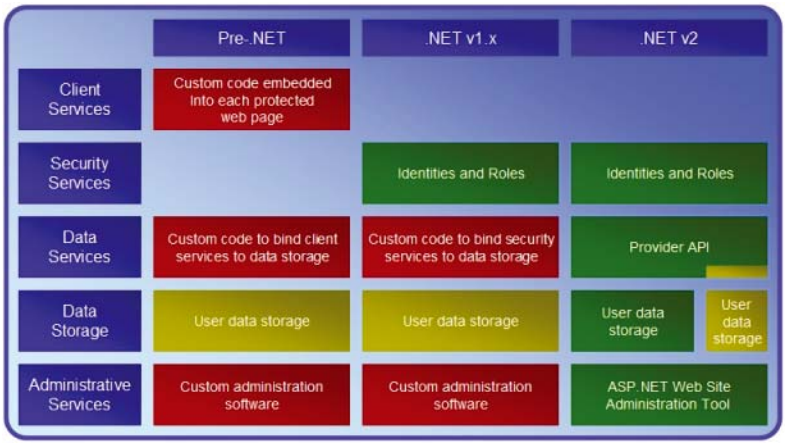
Na úvod zdôrazníme jednu dôležitú a osvedčenú zásahu, ktorou je inak škodlivá paranoja. Pri administrácii aplikácie alebo databázy a nastavovaní prístupových práv neverte nikomu a používateľom už vôbec nie. Nemôžete sa spoliehať na to, že sa budú správať zodpovedne a už vôbec nie na to, že sa budú správať korektne. Vytvoriť webovú databázovú aplikáciu nad účtom systémového administrátora a zverejniť pre všetkých klientov jeho prístupové parametre určite nezaistí bezpečnú prevádzku databázy. Ani opačný extrém nie je dobrý. Napríklad dĺžka hesla v náväznosti na ľudskú psychiku je problém sám o sebe. Ak je heslo jednoduché a ľahko zapamätovateľné, ľahko ho odhalíme. Ak je to naopak 25 znaková nezmyselná kombinácia malých a veľkých písmen a číslic, pravdepodobne si každý heslo niekam napíše a tak stačí skúmať žlté papieriky nalepené na stole, prípadne nápisy ceruzou na okraji monitora, alebo na spodnej strane klávesnice...

Taktiež je potrebné vysvetliť pojmy autentifikácia a autorizácia. Klient sa po prístupe na webové stránky väčšinou neautentifikuje, nepreukazuje (**anonymný prístup**), a pristupuje len k verejne prístupným informáciám – zistí si predpoveď počasia, bankové kurzy, prečíta správy, popozerá nové fotky modeliek, proste každý podľa svojho gusta. Ak však klient chce pristupovať k privátnym informáciám, napríklad k svojmu účtu, alebo musí zadať dôverné informácie musí sa nejakým spôsobom preukázať (**autentifikácia**). Na základe autentifikačných údajov sa zistí, či ten ktorý klient má alebo nemá k požadovanej službe prístup (**autorizácia**).

Role

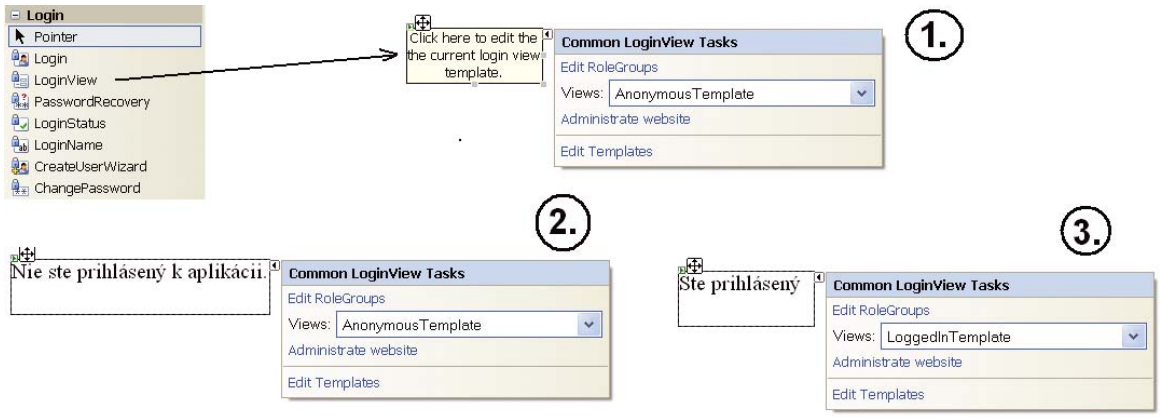
Ak začneme vytvárať používateľov pre zložitejší projekt a prideľovať im prístupové práva, čoskoro prídeme na to, že sa nám vytvárajú skupinky používateľov, ktoré majú tieto práva rovnaké. Je to analógia s bežnou hierarchiou vo firmách, kde pracuje viac pracovníkov v rovnakom pracovnom zaradení. Role umožňujú združovať používateľov do skupín. Viacero používateľov môže byť priradených k jednej role, ale aj opačne, jeden používateľ môže mať prístup k viacerým rolám.

Ako sa situácia s prístupovými právami a rolami vyvíjala pred „érou“ .NET a vo verziách .NET 1.X a .NET 2.0



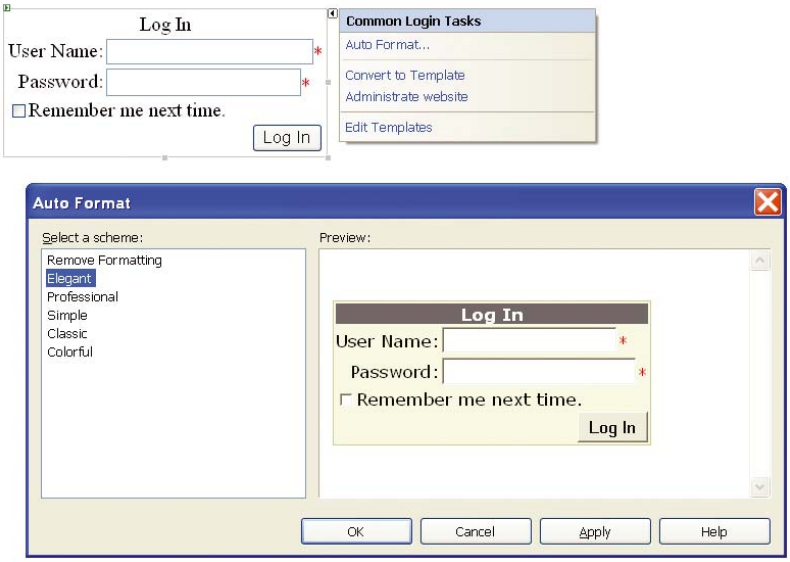
Porovnanie verzií.NET frameworku z hľadiska autentifikácie a autorizácie

Z toolboxu zo sekcie Login pridáme na plochu aplikácie komponentu LoginView. V menu Common Tasks je potrebné zvoliť najskôr Anonymous Template a na plochu komponenty doplniť text v zmysle „Nie ste prihlásený k aplikácii.“ Potom prepneme na LoggedIn Template a na plochu komponenty umiestnime text v zmysle „Ste prihlásený“.



Komponenta LoginView

Samotný proces autentifikácie a autorizácie sa bude odobhrávať na základe prihlasovacích údajov (meno, heslo), ktoré budeme zadávať na samostatnej stránke. Pridajme teda do projektu nový WebForm a nazvime ho Login.aspx.



Komponenta Login

V súbore Web.Config zmeníme spôsob autentifikácie a

```
<authentication mode="Windows"/>
```

na

```
<authentication mode="Forms" />
```

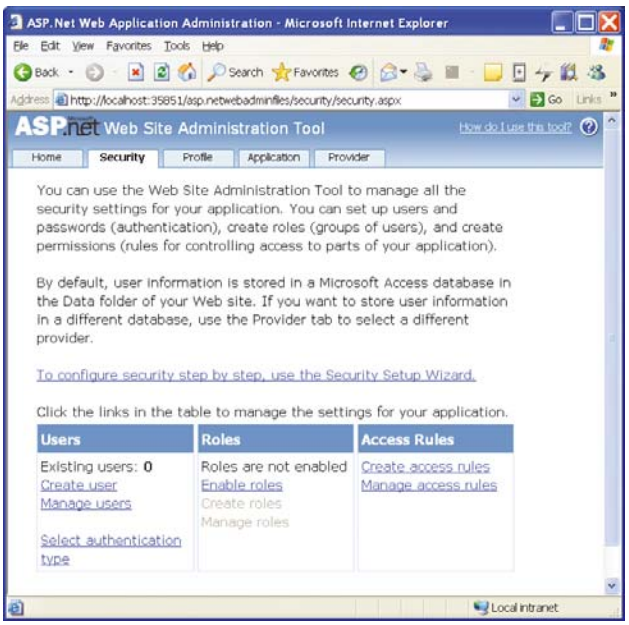
V projekte (okno Solution Explorer) vytvoríme nový adresár Administration. V tomto adresári vytvoríme nový webový formulár Default.aspx, ktorý už bude obsahovať zabezpečené údaje.

Ak máme vytvorené navigačné menu na MasterPage (podľa štvrtej kapitoly) nezabudneme do súboru Sitemap pridať položky so zabezpečenými stránkami v adresári Administration

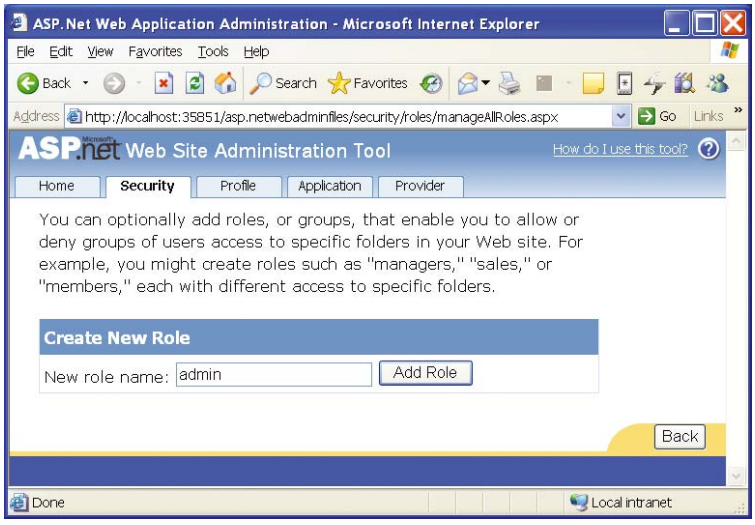
```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap>
    <siteMapNode url="" title=""  description="" roles="">
        <siteMapNode url="" title=""  description="" roles="" />
        <siteMapNode url="" title=""  description="" roles="" />
        <siteMapNode url="~/Administration" title="Admin" description="" roles="" />
        ...
   </siteMapNode>
</siteMap>
```

Vo verzii ASP.NET 2.0 máme k dispozícii administrátorský nástroj, ktorý aktivujeme buď v komponente LoginView pomocou Common Tasks menu (položka Administrate website) alebo v projekte prostredníctvom menu Website – ASP.NET Configuration.

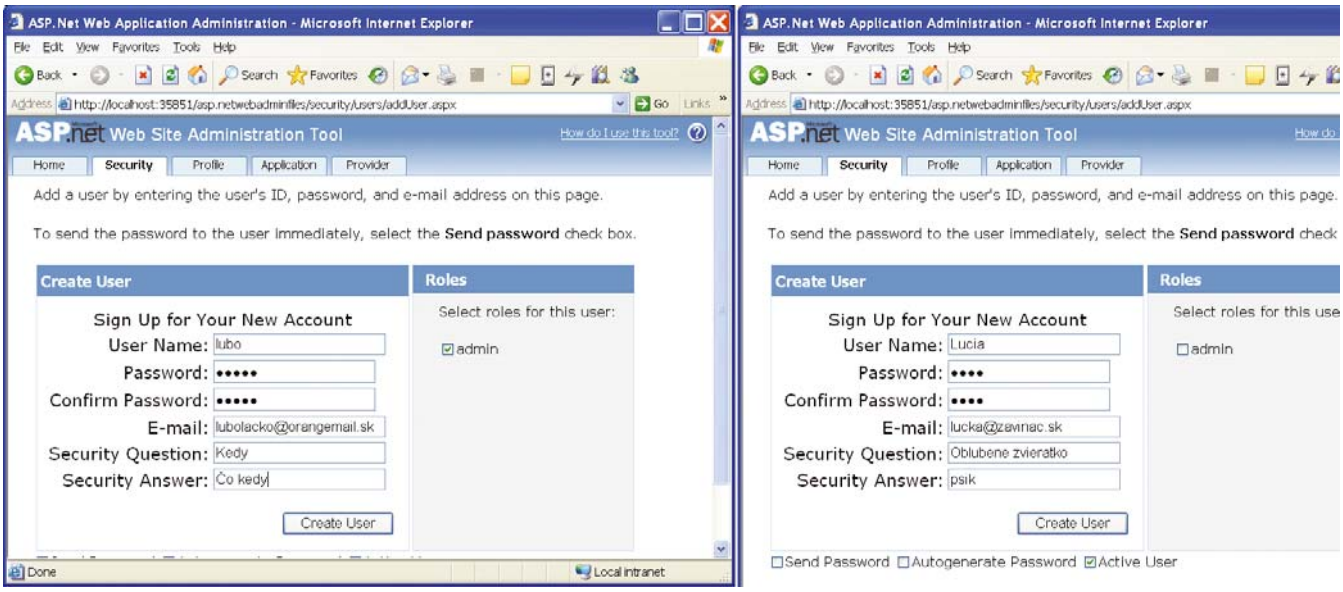
Aktivácia administrácii webového sídla



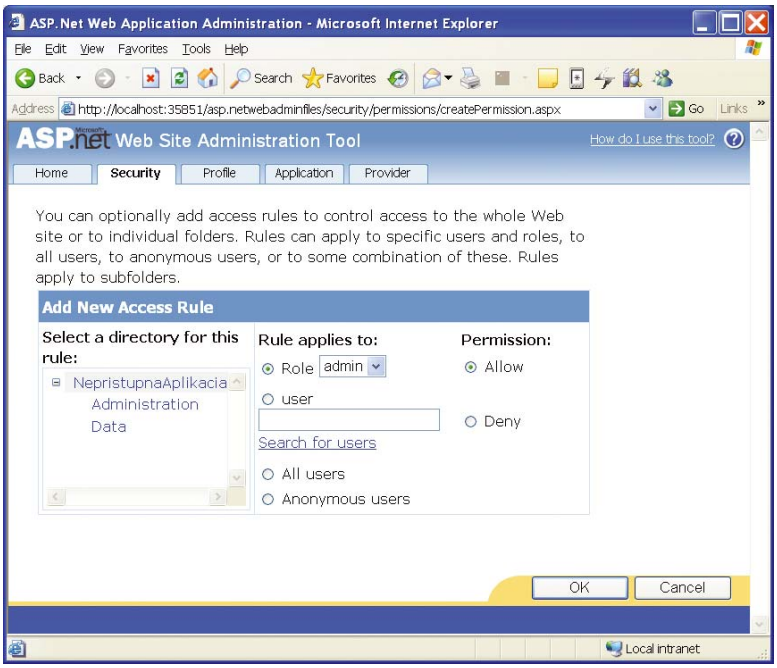
Administrácia webového sídla – záložka Security



Vytvorenie novej role



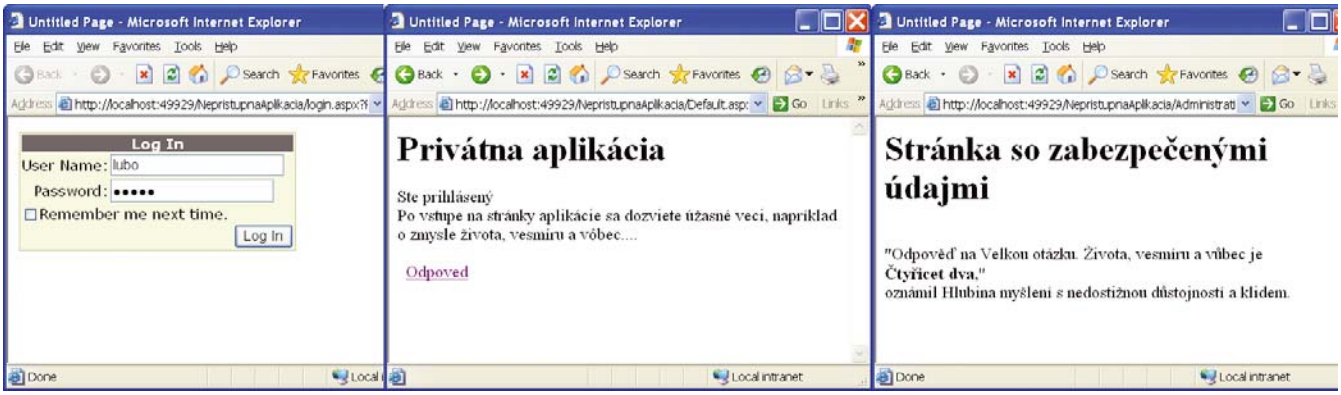
Vytvorenie nového používateľa



Vytvorenie prístupového pravidla

Toto bolo povoľovacie pravidlo pre administrátora. Následne vytvoríme aj opačné pravidlo, kedy ostatným klientom prístup zakážeme – nastavíme permission Deny pre AllUser. V súbore Web Config sa nami vykonané zmeny prejavia nasledovne

```
...
<authorization>
    <allow roles="admin" />
    <deny users="*" />
</authorization>
...
```



Spustenie zabezpečenej aplikácie

Ak tomuto spôsobu zabezpečenia neveríme, môžeme sa pokúsiť o priamy prístup na jednotlivé zabezpečené stránky aplikácie pomocou URL adries

http://localhost:49929/NepristupnaAplikacia/Default.aspx

alebo

http://localhost:49929/NepristupnaAplikacia/Administration/Default.aspx

no neuspejeme, zakaždým musíme prejsť cez dialóg pre zadanie mena a hesla

Uvedená metóda autentifikácie posiela heslo cez HTTP protokol, takže nie je až takým problémom toto heslo zachytiť. Pre hobby aplikácie to nepredstavuje žiadny problém, no pokiaľ by šlo o aplikácie typu internetový obchod alebo bankovníctvo (a teda o peniaze) je bezpečnejšie použiť HTTPS

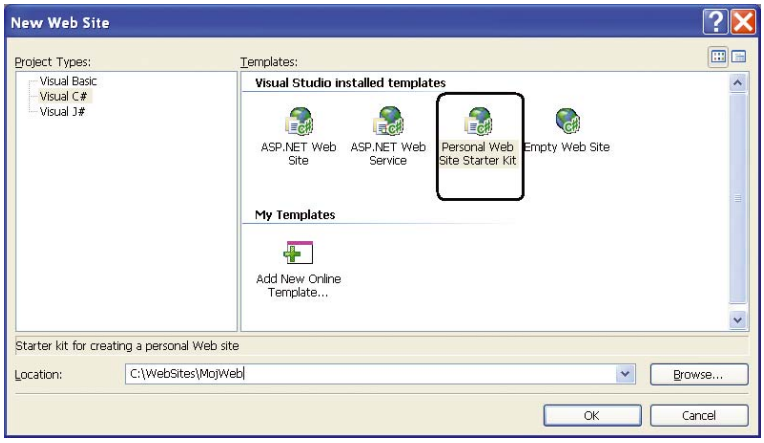
Kapitola 8:

Personal Web Site Starter Kit

Ak si pozrieme na webe osobné stránky a informačné stránky zamerané na hobby, zábavu a podobne, zistíme, že sú si do značnej miery podobné. Na tomto poli málokto vymyslí niečo nové a tak sa z výnimkou zopár kreatívne zameraných ľudí ktorí sa budú snažiť prísť s úplne novou koncepciou osobných stránok pustí do návrhu klasického dizajnu takejto stránky. Skrátka a dobre pravdepodobne sa chceme naplno zamerať na subjekt nášho záujmu, preto návrh kostry takejto aplikácie by sme až do okamihu, kedy ju začneme modifikovať do podoby, ktorá vyplýva z obsahových kritérií považovali za nudnú záležitosť, jednoducho nutné zlo. Samozrejme môžeme použiť a zmodifikovať veľké množstvo príkladov a prototypových aplikácií dostupných na webe (asp.net), no tvorcovia nového vývojové prostredia Visual Web Design mali na mysli jeho najčastejšie nasadenie do oblasti personálnych stránok, a doplnili šablónu pre typ aplikácie Personal Web Site Starter Kit.

Prototypová aplikácia je prehľadne rozčlenená do sekcií pre administráciu a prihlasovanie sa používateľov, publikovanie textových a obrazových informácií, prácu s albumom fotografií a ... nakoľko sa jedná o beta verziu, tvorcovia ju možno na základe beta testovania rozšíria o ďalšie funkcie. Modifikácia je veľmi jednoduchá a prehľadná.

Osobnú stránku, alebo stránku hobby vytvoríme s použitím šalóny Personal Web Site Starter Kit v preferovanom programovacom jazyku.



Vytvorenie aplikácie typu Personal Web Site Starter Kit

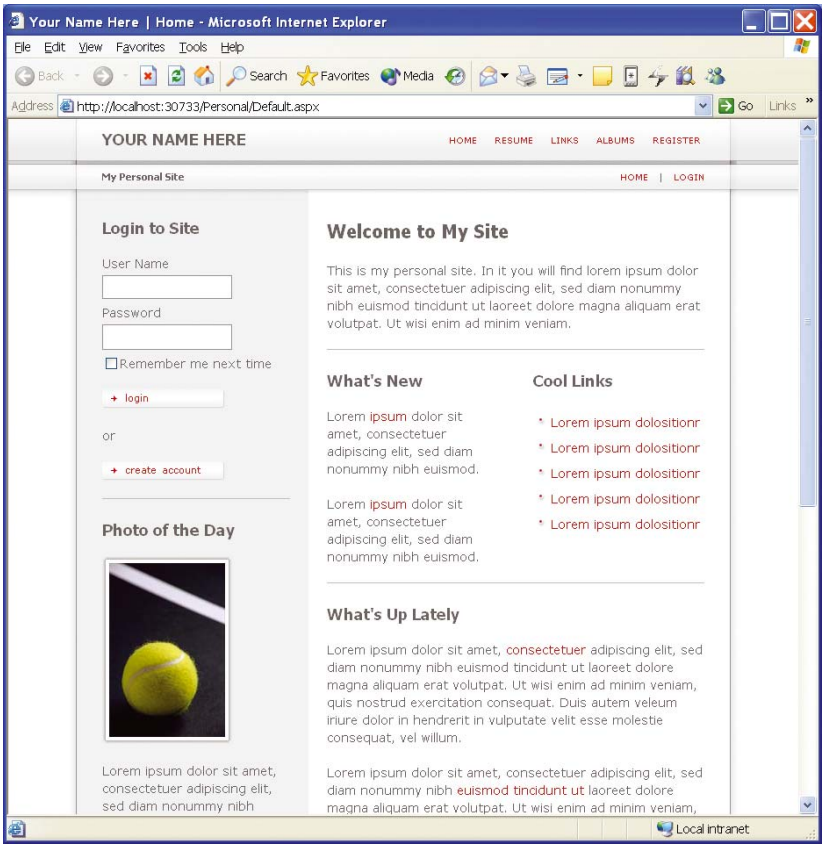
Po vytvorení projektu sa zobrazí stránka Welcome.html, ktorá obsahuje informácie o postupe modifikácie kostry osobných stránok. Stránka je rozdelená do viacerých sekcií

- Introduction
- Creating an Administrative User
- Managing the Personal Web Site
- Designating Guest Users as Friends
- Working with Photo Albums
- Publishing your Site
- Further Resources

Potom už snáď stačí len zmodifikovať odkazy a texty (implicitne je tam skomolený text Cicerovho citátu „Lorem ipsum...“) a prvá funkčná verzia osobnej stránky je na svete. Podľa zoznamu úkonov v predchádzajúcom odstavci to zasa až také jednoduché nebude, zatiaľ si môžeme aplikáciu spustiť a poprezerať šablóny stránok v záložkách

– HOME
– RESUME
– LINKS
– ALBUMS
– REGISTER

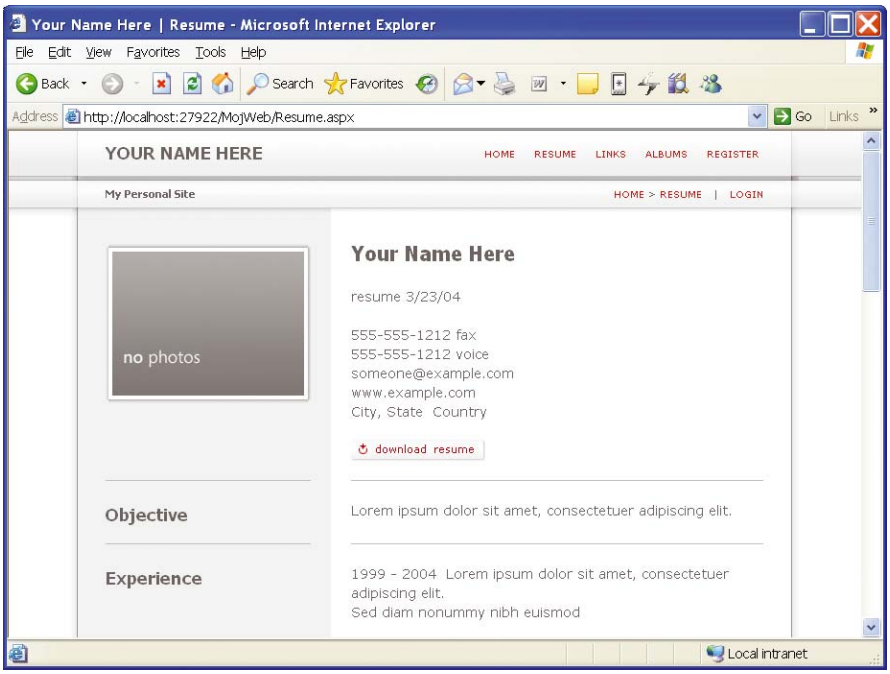
Záložka HOME je hlavnou stránkou projektu. Sem umiestnime v prípade osobnej stránky hlavnú ideu našej stránky, novinky, linky na relevantné odkazy, prípadne na čom práve pracujeme. Z funkčného hľadiska tu dominuje prihlasovací dialóg. Po prihlásení bude administrátor stránky, prípadne priatelia alebo členovia zberateľského klubu môcť vykonávať niektoré modifikačné úkony, napríklad vytvárať albumy a umiestňovať do nich fotografie



Personal Web Site Starter Kit

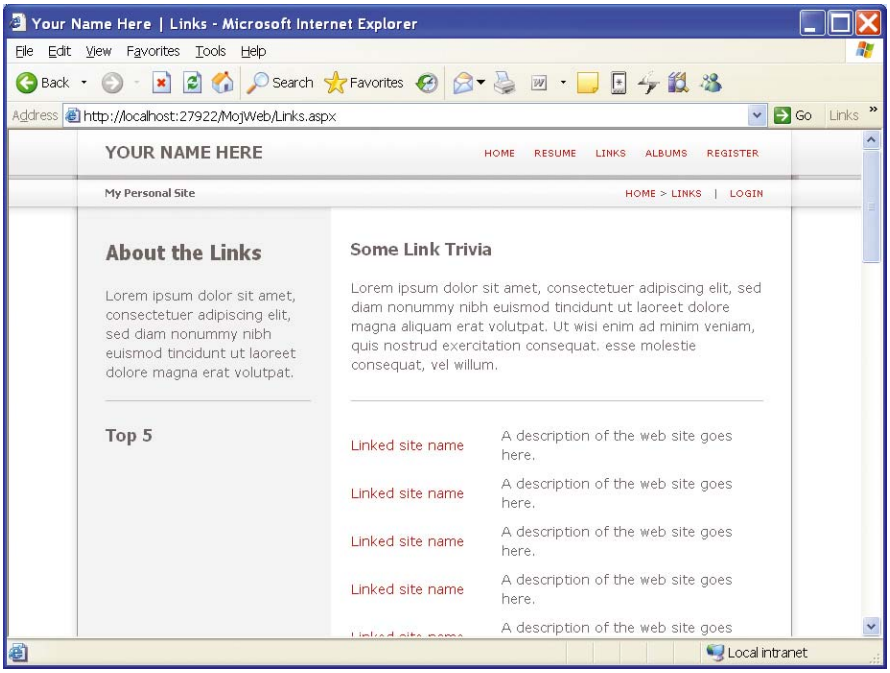
Záložka RESUME je v prípade osobných stánok určená na zhrnutie faktov o danej osobe, prípadne o časovej genéze hobby, ktoré je na stránke prezentované. Ako približný návod, aké údaje sem umiestniť uvedieme takýto príklad. Máme trebárs osobnú stránku amatérskeho fotografa. No a pre prípad, že by naša stránka zaujala nejakú agentúru, prípadne redakciu časopisu ktorá hľadá fotografa, do záložky RESUME uvedieme náš stručný profesný životopis a úspechy dosiahnuté v oblasti hobby a samozrejme kontaktnú adresu. Potencionálny záujemca si tu tieto údaje spoľahlivo nájde. Sú aj príklady ako sa to robiť radšej nemá, napríklad životopis takého typu ako na stránke www.martinjakubec.sk by mohol zaujať jedine v prípade kastingu na prezidenta galaxie.

Všimnite si ešte raz text v predchádzajúcom odstavci. Aj napriek tomu, že sa jedná o vývojársku brožúru, vôbec neriešime aké triedy a objekty použijeme, kde budú uložené údaje, ale zameriavame sa výhradne na obsah. Už podľa toho je zrejmé že tvorcom šablóny Personal Web Site Starter Kit sa ich zámer vydaril.



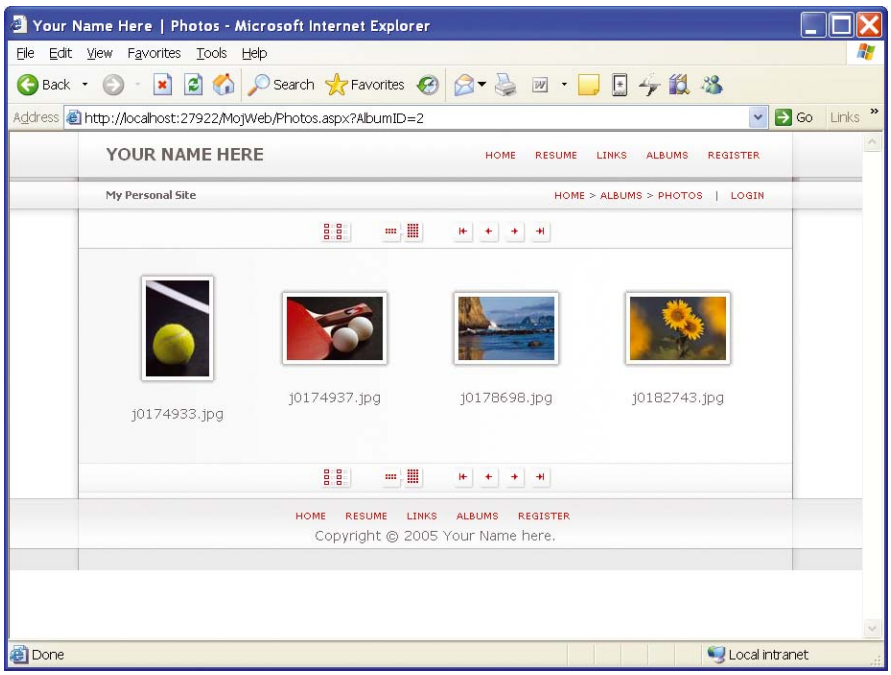
Personal Web Site Starter Kit – záložka RESUME

V záložke LINKS umiestnime linky na stránky priateľov, prípadne ľudí s podobnými záujmami, linkami na internetové obchody a burzy viažúce sa k predmetu hobby



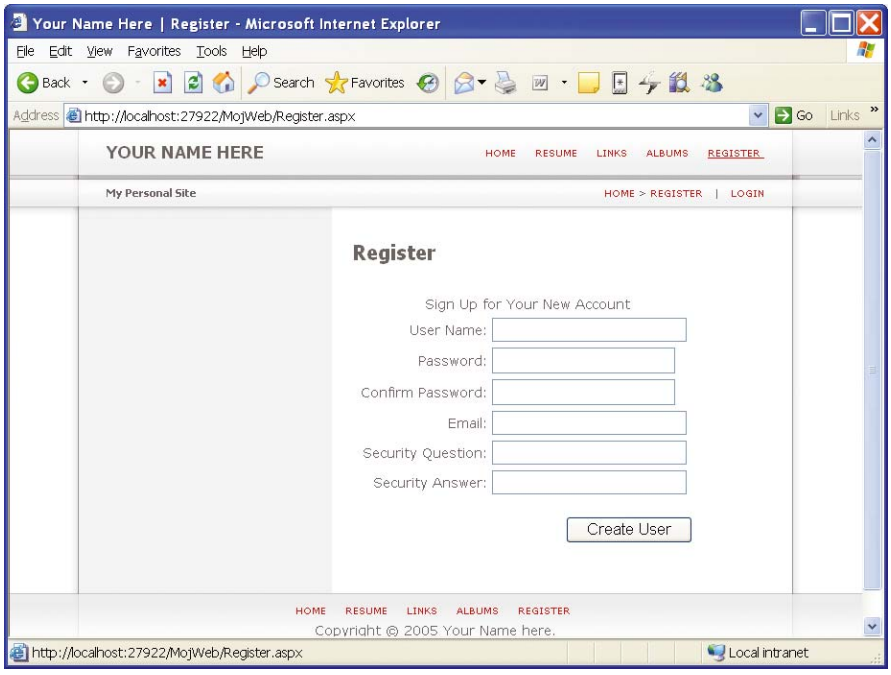
Personal Web Site Starter Kit – záložka LINKS

Aj záložka ALBUMS je viacúčelová. Sem môžeme umiestniť fotografie či už rodinné, alebo dovolenkové, športové a podobne. Druhou možnosťou je vytvoriť tu zberateľský katalóg



Personal Web Site Starter Kit – záložka ALBUMS

V záložke REGISTER je možné „vstúpiť do klubu“, a teda prihlásiť sa do užšej privátnejšej skupiny,s prístupom ku „klubovej“ časti našej stránky.



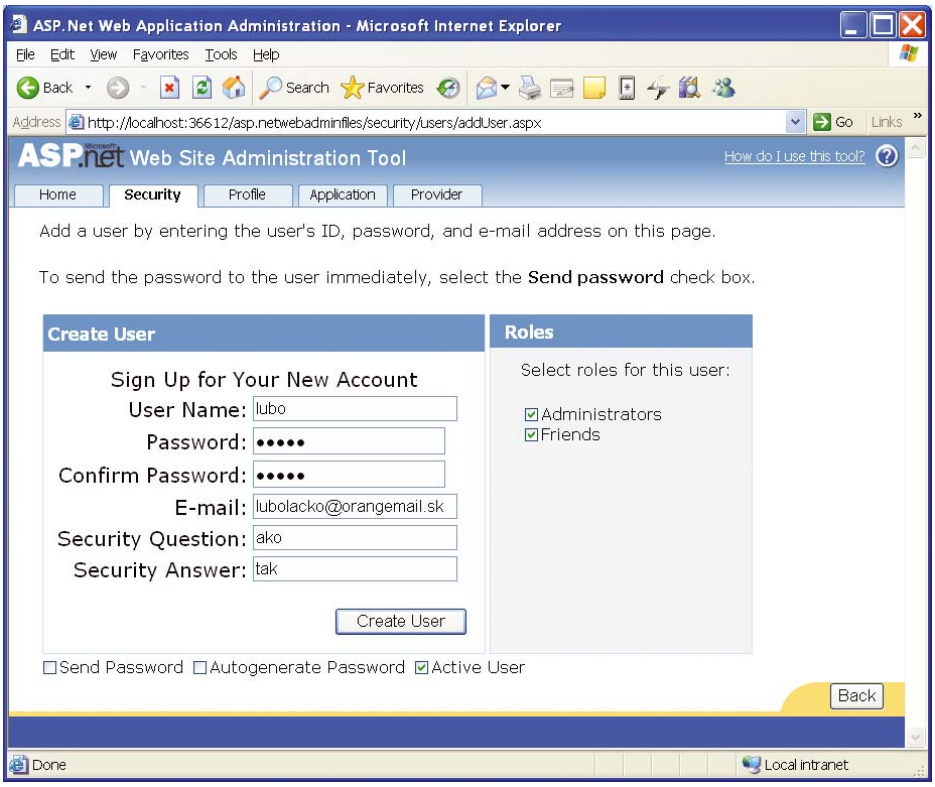
Personal Web Site Starter Kit – záložka REGISTER

Po stručnej exkurzii po jednotlivých záložkách osobnej stránky v hrubých rysoch ukážeme jej „ovládnutie“, teda prispôsobenie našim požiadavkám a naplnenie obsahom.

Vytvorenie administrátora webového sídla

Tento krok (ktorý je podrobne popísaný v predchádzajúcej kapitole) musíme vykonať ako prvý, nakoľko len administrátor má napríklad právo vytvárať albumy a posielať do nich fotografie cez upload.

V menu Website aktivujeme voľbu ASP.NET Configuration. Zobrazí sa administračná HTML stránka v ktorej sa prepneme do zložky Security, kde vytvoríme nového používateľa pomocou linku Create User. Vyplníme všetky potrebné údaje (meno, heslo, mail, kontrolnú otázku pre prípad zabudnutia hesla...) a nakoľko tento používateľ bude administrátorom zaškrtneme obidve políčka v pravej časti stránky, teda Administrators aj Friends.

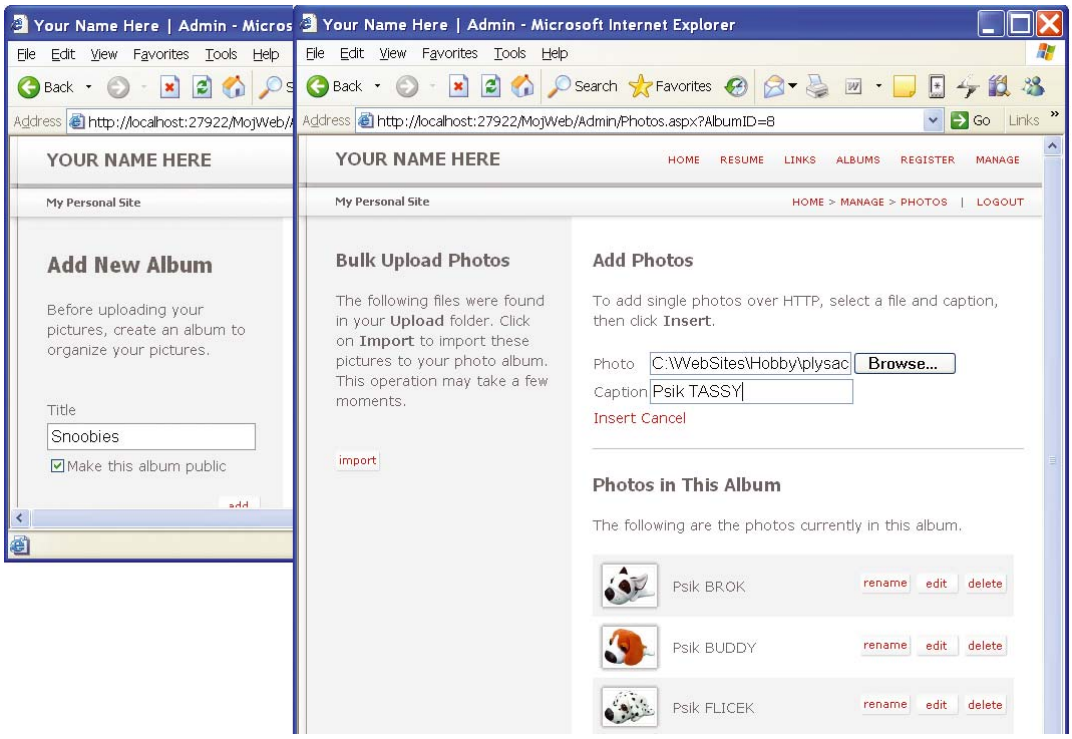


Vytvorenie používateľa

Potom pridáme ostatných používateľov (členov rodiny, priateľov, zamestnancov...) a pridelíme im rolu Friends, prípadne vytvoríme iné role a nastavíme pre ne prístupové parametre.

Práca s albumami fotografií

Ak budeme postupovať podľa metodiky na úvodnej stránke projektu (welcome.html), mali by sme oživiť album fotografií. A nakoniec prečo nie, veď v multimediálnej ére umocnenej digitálnymi fotoaparátmi integrovanými do mobilov je to pomerne silný výrazový prostriedok. Nový album vytvoríme a fotografie do neho natiahneme v zložke MANAGE



Upload fotografií do albumu

Default.master

Web.sitemap

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap>
      <siteMapNode title="Home" url="Default.aspx">
            <siteMapNode title="Resume" url="Resume.aspx" />
            <siteMapNode title="Links" url="Links.aspx" />
            <siteMapNode title="Albums" url="Albums.aspx" >
                  <siteMapNode title="Photos" url="Photos.aspx" >
                        <siteMapNode title="Details" url="Details.aspx" />
                  </siteMapNode>
            </siteMapNode>
            <siteMapNode title="Register" url="Register.aspx" />
            <siteMapNode title="Manage" url="Admin/Albums.aspx" >
                  <siteMapNode title="Photos" url="Admin/Photos.aspx" >
                        <siteMapNode title="Details" url="Admin/Details.aspx" />
                  </siteMapNode>
```

```
            </siteMapNode>
        </siteMapNode>
</siteMap>
```

Albums : Tabulka

	AlbumID	Caption	Public
+	1	System Folder	☑
+	2	Sample Album	☑
*	tomatické číslo)		☐

Záznam: 1 z 2

Photos : Tabulka

	PhotoID	AlbumID	BytesOriginal	BytesFull	BytesPoster	BytesThumb	Caption
	1	1	ıhá binární data	ıhá binární data	ıhá binární data	ıhá binární data	No Photos
	32	2	ıhá binární data	ıhá binární data	ıhá binární data	ıhá binární data	j0174933.jpg
	33	2	ıhá binární data	ıhá binární data	ıhá binární data	ıhá binární data	j0174937.jpg
	34	2	ıhá binární data	ıhá binární data	ıhá binární data	ıhá binární data	j0178698.jpg
	35	2	ıhá binární data	ıhá binární data	ıhá binární data	ıhá binární data	j0182743.jpg
*	tomatické číslo)	0					

Záznam: 1 z 5

Databázové tabuľky

Publikovanie webovej aplikácie

Po prispôsobení osobnej stránky a jej naplnení údajmi môžeme ešte na lokálnom vývojárskom počítači starostlivo skontrolovať funkčnosť všetkých odkazov a záložiek a po drobných úpravách ju môžeme publikovať na wem. Pre tento účel použijeme funkciu Copy Web Site (dostupnú z menu Website vývojového prostredia). Kopírovať aplikáciu môžeme buď na reálny server cez FTP, alebo do inej lokality na lokálnom počítači.

Ukážeme si prenesenie osobnej stánky do inej lokality na vývojárskom počítači. Zdanlivo to nemá význam, ale doteraz sme našu aplikáciu spúšťali len z vývojového prostredia. Súbory, ktoré ju tvoria sú implicitne v adresári

```
C:\website\MojWeb
```

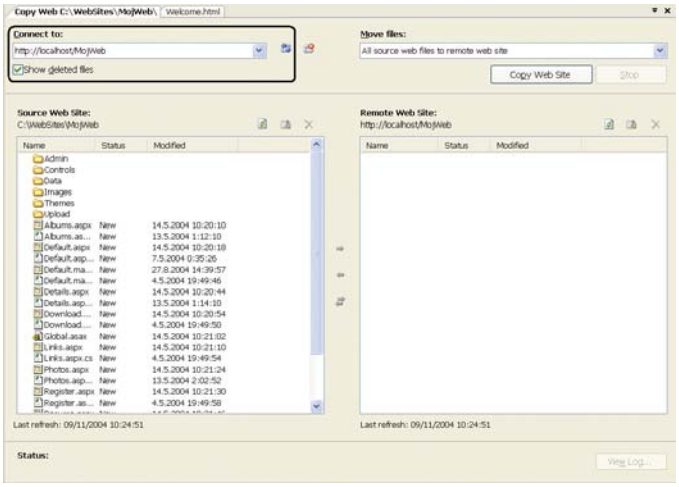
Aby sme mohli aplikáciu spúšťať cez Internet Information Server, umiestnime ju do implicitného adresára pre publikovanie webových aplikácií.

```
C:\inetpub\
```

Poznámka
Tento príklad ukážeme len pre ilustráciu publikovania webovej aplikácie do inej lokality. Ako uvidíme neskôr, pokojne by aplikácia mohla zostať, tam kde bola vytvorená a v pracovnom adresári C:\inetpub\ by sme vytvorili len virtuálny adresár.

Pri publikovaní na reálny webový server si najskôr necháme od administrátora vytvoriť konto a získame od neho parametre pre FTP pripojenie. U lokálnej aplikácie vytvoríme v pracovnom adresári podadresár
`C:\inetpub\MojWeb\`

Spojenie nadviažeme pomocou malého tlačidla s dvomi vodorovnými protismernými šípkami vľavo hore, vedľa comboboxu Connect to:



Pracovná obrazovka Visual Web Developera pre publikovanie webovej aplikácie

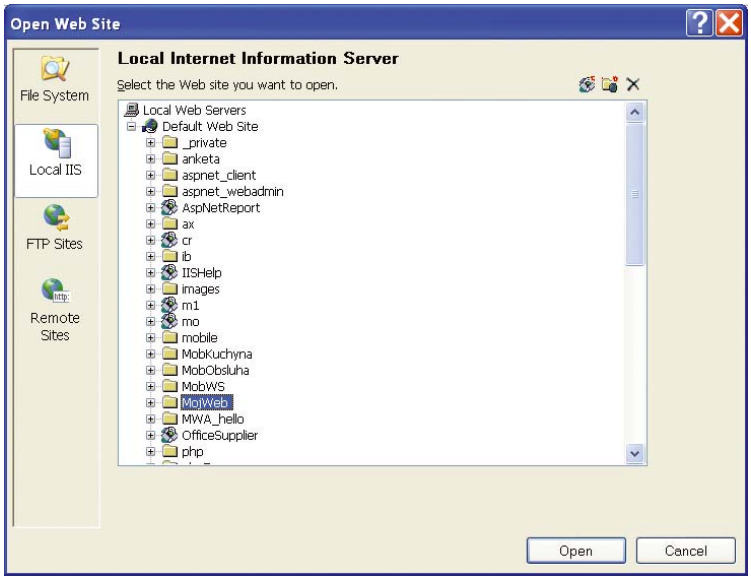
Zobrazí sa dialóg Open Web Site, v ktorom si najskôr pomocou tlačidiel v ľavej časti vyberieme druh a spôsob prenosu a v stromovej štruktúre v pravej časti upresníme lokalizáciu. V našom prípade Local IIS a adresár

```
C:\inetpub\MojWeb\
```

prístupová adresa bude

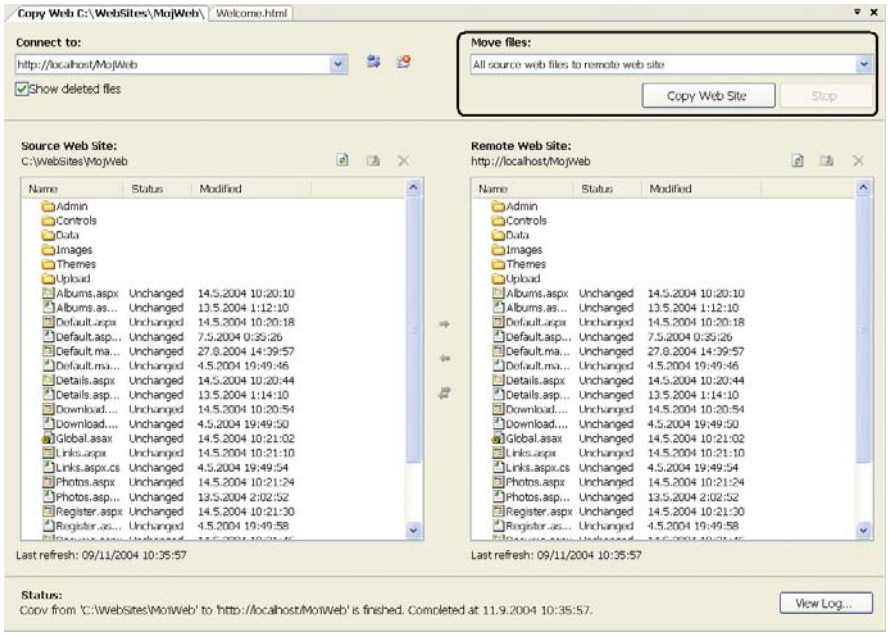
```
http://localhost/MojWeb/
```

V prípade FTP prenosu bude adresa cieľovej destinácie v tvare
`ftp://ftp.servername/foldername`



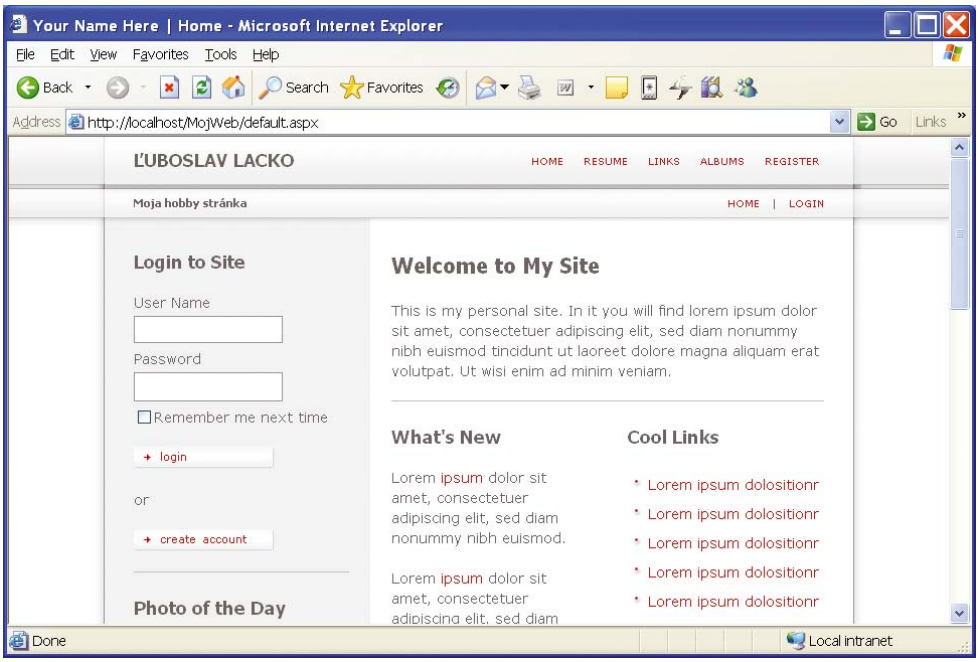
Dialóg pre vytvorenie pripojenia pre publikovanie webovej aplikácie

Prostredníctvom pracovnej obrazovky Visual Web Developera pre publikovanie webovej aplikácie, ktorá má dizajn ako väčšina známych súborových manažérov môžeme vybrať súbory a adresáre, ktoré chceme kopírovať. Nakoľko aplikáciu prenášame po prvý krát, a potrebujeme preniesť všetky jej súbory a adresáre, aktivujeme v Comboboxe Move files: voľbu All source files to remote Web site. Následne stlačíme tlačítko Copy Web Site.



Publikovanie webovej aplikácie

Zostáva len zadať správnu URL adresu a projekt vyskúšať.



Spustenie webovej aplikácie v novej lokalite

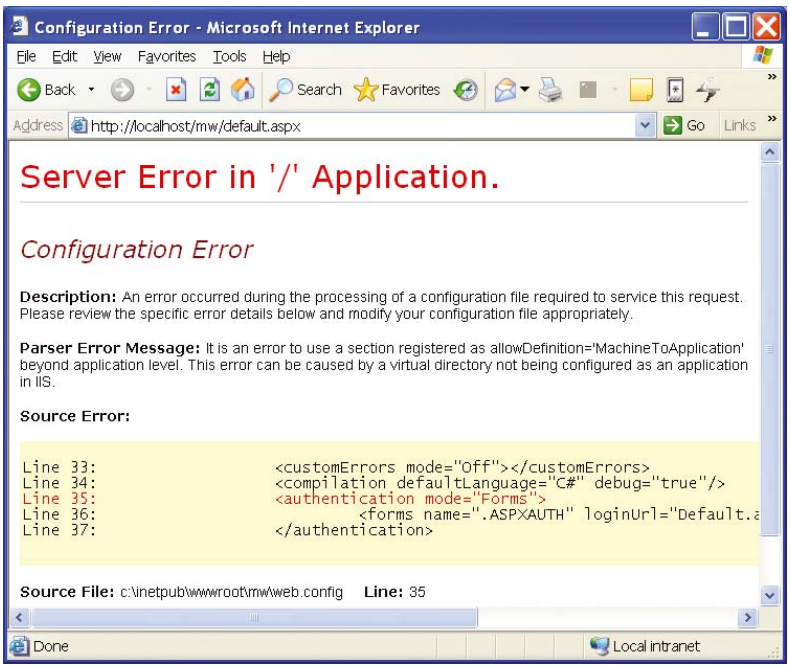
Po tomto pokuse na lokálnom počítači by nás mohlo napadnúť preniesť aplikáciu mimo vývojového prostredia, napríklad pomocou súborového manažéra. Ak by sme len prekopírovali súbory do adresára napríklad

```
C:\inetpub\MojWeb\
```

a aplikáciu spustili prístupom cez URL adresu

```
http://localhost/mw/default.aspx
```

dospeli by sme k chybovému hláseniu



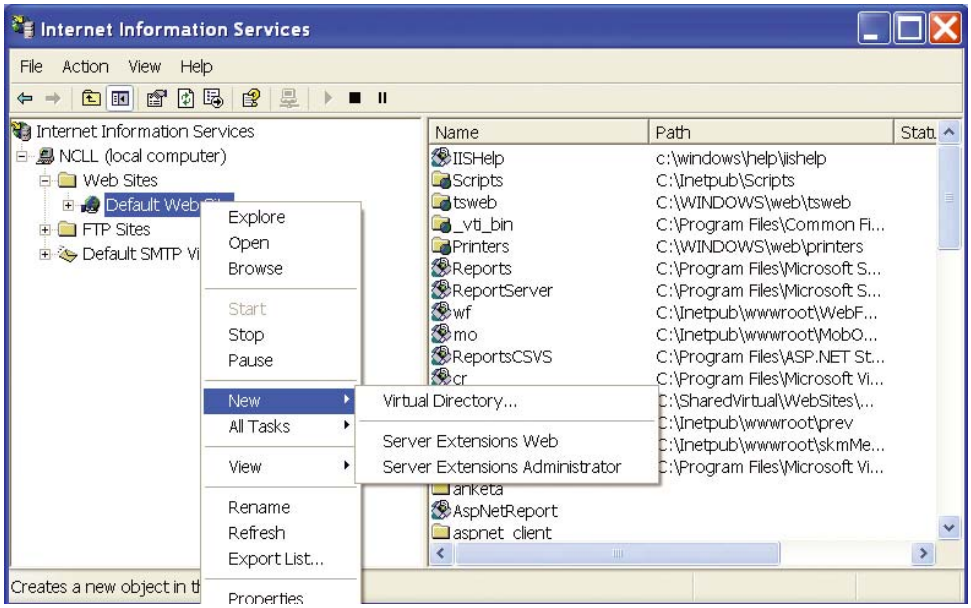
Spustenie webovej aplikácie v novej lokalite

Na základe konfiguračných parametrov v súbore web.config sa totiž kvôli bezpečnosti predpokladá umiestnenie webovej aplikácie do virtuálneho adresára. Pri predchádzajúcom pokuse - publikovaní webovej aplikácie prostredníctvom vývojového prostredia bol virtuálny adresár vytvorený automaticky.

Umiestnením aplikačného kódu do virtuálneho adresára dosiahneme dôsledné oddelenie logickej súborovej štruktúry od fyzického súborového systému servera prípadne vývojárskeho počítača. Pre ilustráciu v stručnosti ukážeme po jednotlivých krokoch postup pre vytvorenie a nakonfigurovanie virtuálneho adresára na vývojárskom počítači s operačným systémom Windows XP.

1. Vytvorenie fyzického adresára
Na disku servera, alebo vývojárskeho počítača vytvorime fyzický adresár napríklad `C:\inetpub\mw\` a do neho skopírujeme súbory tvoriace aplikáciu. Prípadne môžeme aplikáciu ponechať aj na povodnom mieste, teda v adresári `C:\website\MojWeb`. Na tento fyzický adresár v ďalšom postupe namapujeme virtuálny adresár. V prípade potreby môžeme v takomto adresári vytvárať aj podadresáre

2. Vytvorenie virtuálneho adresára v IIS
V hlavnom menu operačného systému, konkrétne v zložke Control Panel – *Administrative Tools* aktivujeme položku *Internet Information Services* a v kontextovom menu vytvoríme virtuálny adresár pomocou POLOŽKY *New – Virtual Directory*.



Menu pre vytvorenie virtuálneho adresára

Pre novovytváraný virtuálny adresár určený pre webovú aplikáciu zadáme vhodný alias, v našom prípade trebárs „mw“. Novo vytváraný virtuálny adresár následne namapujeme na fyzický adresár. Napokon môžeme nastaviť prístupové privilégiá pre virtuálny adresár.

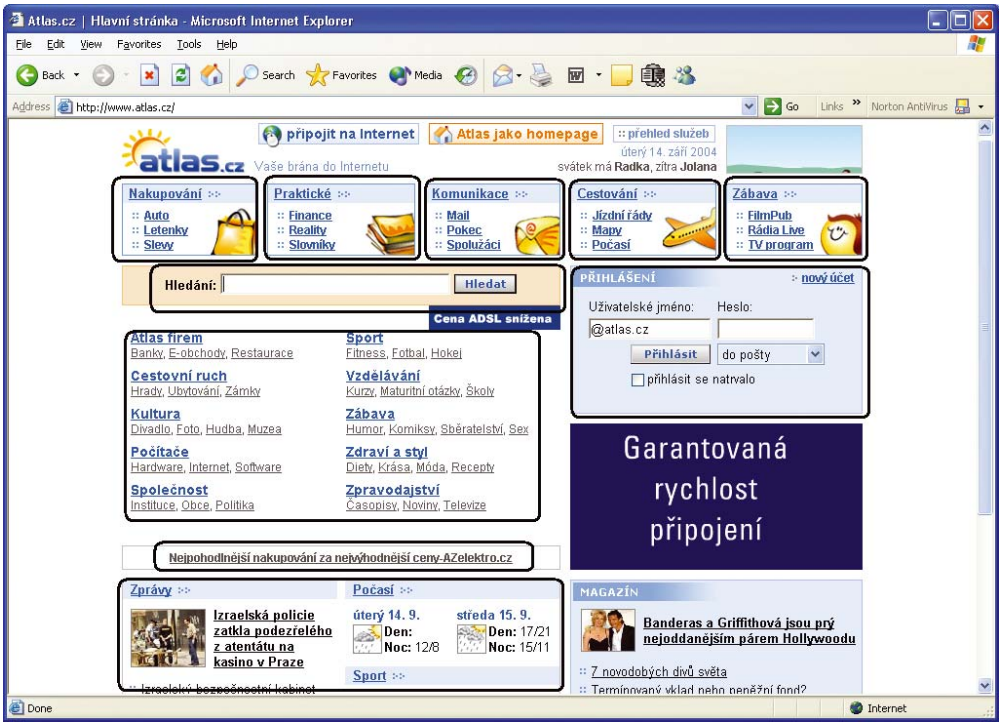
3. Zabezpečenie virtuálnych adresárov
Na vývojárskom počítači otázka zabezpečenia virtuálneo adresára nie je až taká pálčivá a administrácia serverov je mimo záber tejto publikácie, takže teroreticky by sme mohli pracovať aj s nezabezpečeným virtuálnym adresárom. Samozrejme je lepšie dodržiavať zásady bzpečnosti vždy, aj pri vývoji aplikácie. Na vývojárskom počítači postačí použiť integrovanú Windows autentizáciu.

Globálny stupeň zabezpečenia nastavujeme podľa požiadaviek pre každý virtuálny adresár osobitne. V kontextovom menu každého virtuálneho adresára nastavíme parameter *Application Protection* na požadovanú hodnotu *Low (IIS)*, *Medium (Pooled)*, prípadne *High (Isolated)*.

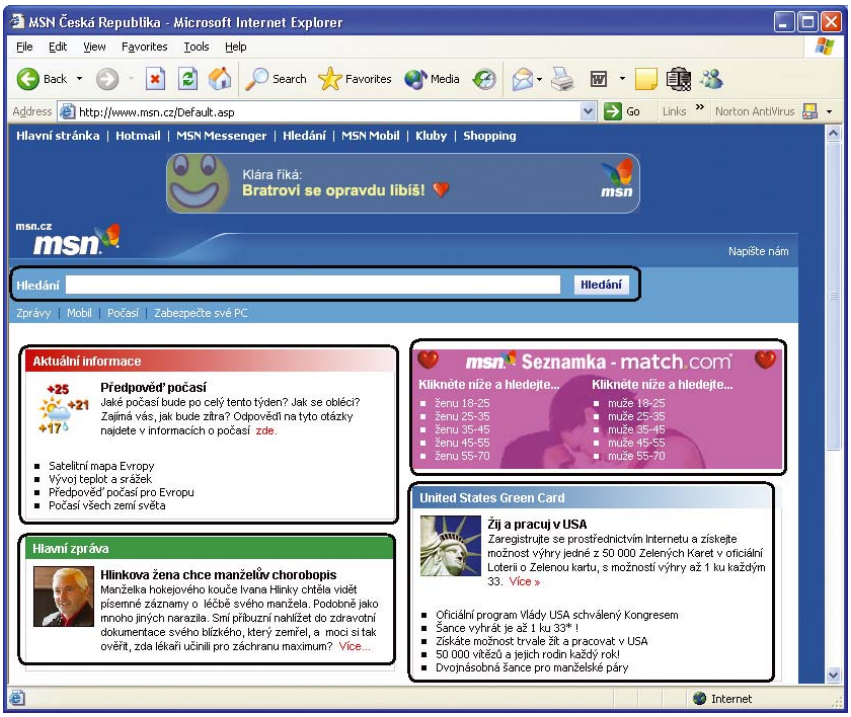
Kapitola 9:

Vytvorenie portálu pomocou ASP.NET 2.0

Pojem portál v doslovnom preklade znamená „vznešenú bránu“ a brány tohto typu sa spravidla používali pre prístup k niečomu dôležitému. V dnešnej dobe sú azda najcennejšie informácie, čiže má význam budovať k nim ani nie tak vznešené, ako skôr funkčné a bezpečné centrálne prístupové brány. Hlavnou úlohou portálov je sprístupniť užívateľom požadované informácie tak, aby s nimi mohli pracovať v okamihu, kedy ich potrebujú. Ak si všimnete väčšinu portálov na webe, zistíte, že ich zobrazovacia plocha je rozdelená na viacero častí - modulov. Tieto moduly môžeme premiestňovať. meniť ich rozmery. V ponuke portálu je niekedy k dispozícii spravidla modulov viac, než je zobrazené takže používateľ si môže vybrať o ktoré moduly (správy, predpovede počasia, bankové kurzy, erotický obrázok dňa...) má záujem.



Príklad rozčlenenia portálu na moduly



Iný príklad rozčlenenia portálu na moduly

Preto aj podpora budovania portálov na platforme ASP.NET 2.0 sa zameriava práve smerom podpory modularity. Jednotlivé moduly zobrazovacej plochy portálu zobrazujú informácie, ktoré spolu nejako súvisia.

Vybudovanie portálu môžeme rozčleniť na etapy

Vytvorenie obsahu portálu.
Tok dokumentov pri príprave obsahu portálu je prakticky totožný s klasickým „papierovaním“ Mierne zjednodušujúc môžeme tento proces rozdeliť do niekoľkých krokov.

Vytvorenie dokumentu
V prvom kroku autor, alebo autorský kolektív dokument vytvorí, skontroluje a umiestni na portál do jeho „neverejnej časti“.

Diskusia a schvaľovanie
Každý dôležitejší dokument sa spravidla stane predmetom diskusie, schvaľovania a podobných obsahových aj formálnych zásahov. Táto etapa diskusie a schvaľovania dokumentu taktiež prebieha v neverejnej časti portálu, kam majú prístup len zainteresovaní „kritici“ dokumentu.

Korektúry
Po zásahoch oponentov, jazykových korektorov, právnikov... sa upraveného dokumentu ujme opäť autor a vykoná takzvanú **autorskú korektúru**. Jazykový korektor mohol totiž napríklad v snahe o čistotu jazyka a štýlu narušiť odbornú terminológiu a podobne,

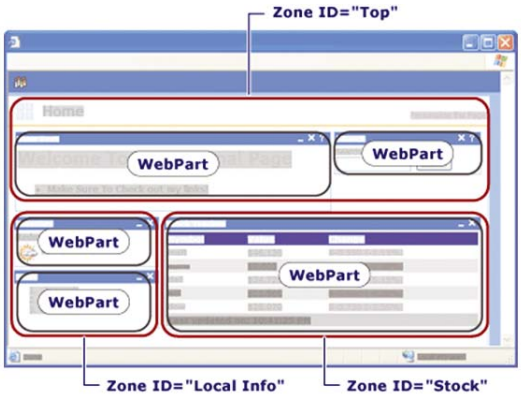
Publikovanie dokumentu
Po vykonaní všetkých doteraz opísaných krokov je možné dokument zverejniť, to znamená, že okrem autorského a oponentského kolektívu má k nemu prístup jeho adresát.

Publikovaním sa „kariéra“ dokumentu obvykle nekončí. Dokumenty spravidla „žijú“ to znamená, že ich autorský kolektív mení a aktualizuje.

Web Parts

V ASP.NET 2.0 je budovanie portálov realizované prostredníctvom Web Parts, ktoré pozostávajú z množiny spolupracujúcich komponentov

WebPartManager – jeho úlohou je manažovanie všetkých WebParts prvkov na stránke
WebPartZone – táto kontejnerová komponenta zapúzdruje WebParts prvkov, pričom tieto nadobúdajú niektoré spoločné črty. Stránka môže obsahovať jednu, alebo viacero zón.
CatalogWebPart – zoznam dostupných WebPart prvkov, ktoré je možné pridať na stránku
CatalogZone – kontejner pre moduly StaticCatalog a PageCatalog.
EditorPart – používateľské rozhranie umožňujúce nastavovanie parametrov pre vybraný WebPart prvok. EditorPart obsahuje prvky AppearanceEditorPart, LayoutEditorPart, BehaviorEditorPart, a PropertyGridEDitorPart.
EditorZone – kontejner pre prvky EditorPart.



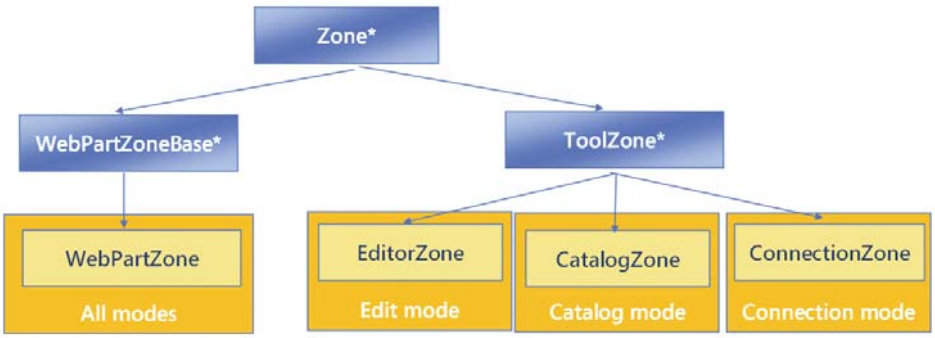
Princíp rozčlenenia portálu na zóny a umiestnenie prvkov.

WebPartManager

WebPartManager je nevizuálna komponenta pre uvedenie stránky do jedného s nasledujúcich stavov

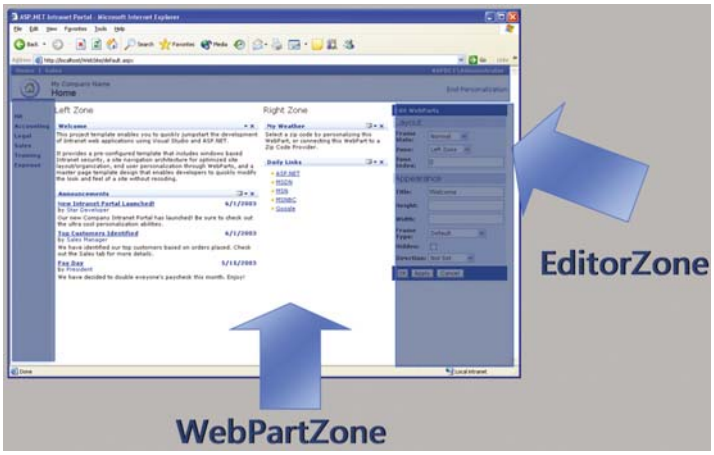
- Normal,
- Design,
- Edit,
- Catalog,
- Connect

Do požadovaného stavu budú prepnuté všetky časti v každej zóne



Rozdelenie na zóny

Možno viac napovie konkrétny obrázok hotového riešenia.

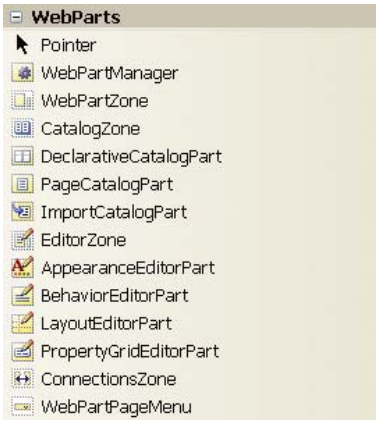


Zóna Web Part a zóna editora

Web Parts ako úplne nová črta ASP.NET 2.0 by pre svoj podrobnejší popis sama spotrebovala publikáciu rozsahu tohto zborníka, preto si radšej na praktickom príklade ukážeme jej použitie

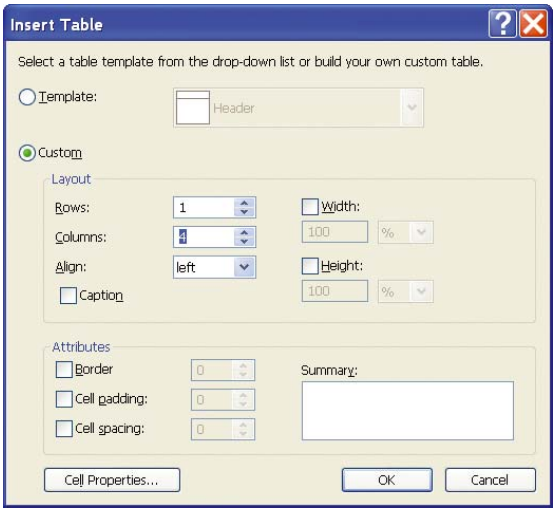
Vytvorenie portálovej aplikácie s využitím WebParts

V úvodnej fáze tohto cvičného projektu vytvoríme novú aplikáciu typ Empty Web Site. Môžeme ju nazvať trebárs NasPortal. Do prázdneho projektu pridáme (kontextové menu Add New Item) webový formulár s názvom povedzme WebParts.aspx, pričom zaškrtneme voľbu, aby bol programový kód v separátnom súbore. Následne budeme pracovať s prvkami WebParts. Ich ikony pre vizuálny návrh sú v rovnomennej zložke Toolboxu



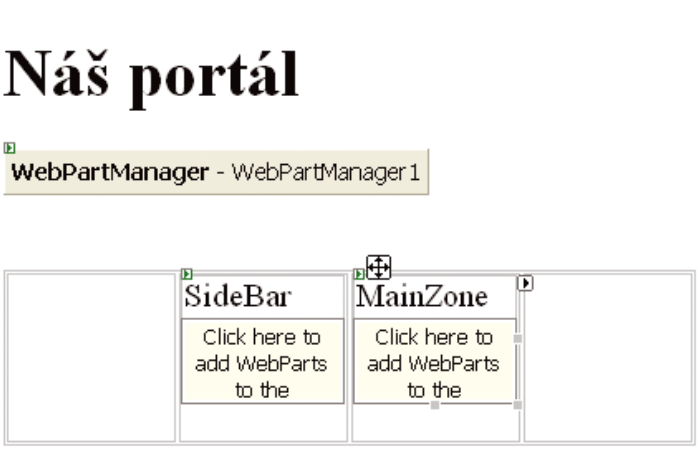
Zložka Tolboxu WebParts

Ako prvý umiestnime na plochu aplikácie prvok WebPartManager. Tento sa ocitne v hornej časti. Následne budeme potrebovať rozčleniť stránku na časti, to znamená fixovať polohu jednotlivých zón Pre tento účel slúži na HTML stránkach tabuľka. Pomocou menu (Layout, Insert Table) vložíme na stránku tabuľku, ktorá bude mať jeden riadok, štyri stĺpce a prvky v bunkách budú zarovnávané doľava.



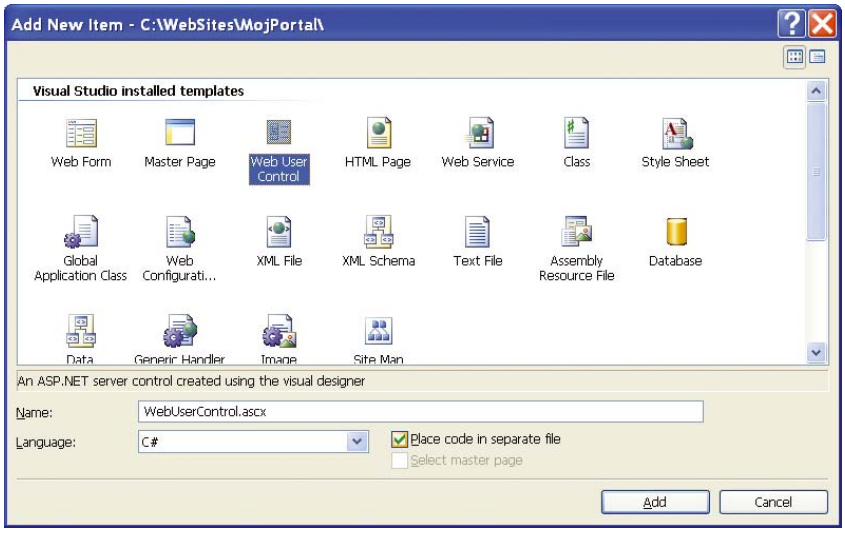
Nastavenie parametrov tabuľky.

Vychádzame pritom z plánu, že v ľavej časti (prvá bunka tabuľky zľava) bude prihlasovací dialóg, potom budú nasledovať dve informačné zóny a úplne vpravo bude zóna editačná.
Do druhého a tretieho poľa tabuľky (zľava) vložíme prvky WebPartZone. Môžeme ponechať pôvodné názvy WebPartZone1 a WebPartZone2, alebo lepším riešením bude použiť výstižnejšie názvy, napríklad ľavý prvok nazveme SiedbarZone a pravý MainZone.



Umiestnenie prvkov WebPartZone do buniek tabuľky tabuľky.

Pre umiestnenie prvkov do vnútra zón máme dve možnosti. Ak sa nám to hodí, môžeme do ich vnútra presúvať prvky z Toolboxu. Do vnútra elementu SidebarZone takto môžeme presunúť z Toolboxu prvok Calendar. Druhá možnosť je vytvoriť si vlastný prvok. Túto možnosť použijeme pre zónu v strednom poli tabuľky (MainZone). Najskôr pomocou kontextového menu Add New Item pridáme do projektu Web User Control.



Nastavenie parametrov tabuľky.

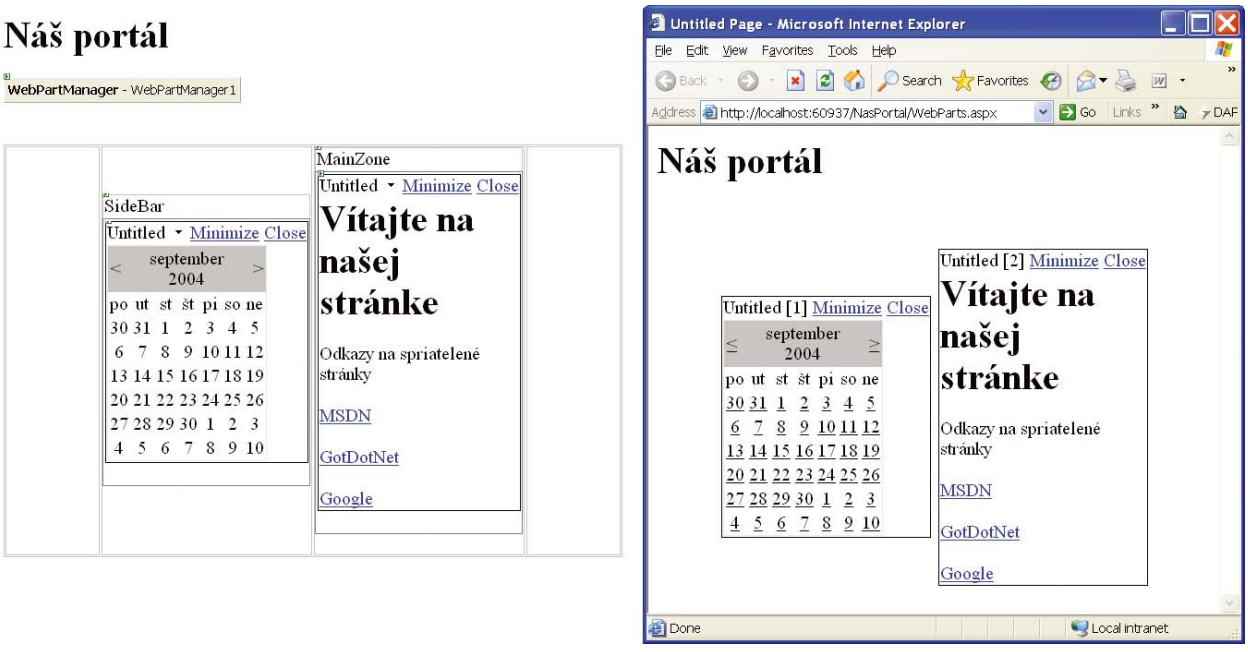
Po tomto úkone pribudne do projektu súbor WebUserControl.ascx. V návrhovom móde nadizajnujeme tento prvok. V našom prípade sme vytvorili odkazy na spriatelené stránky

Návrh Web User Control.

kód v súbore WebUserControl.ascx pri takto navrhnutom dizajne prvku bude

```
<%@ Control Language="C#" CompileWith="WebUserControl.ascx.cs"
ClassName="WebUserControl_ascx" %>

<h1>Vítajte na našej stránke</h1>
<p>Odkazy na spriatelené stránky</p>
<p>
    <asp:HyperLink ID="HyperLink1" Runat="server"
NavigateUrl="http://msdn.microsoft.com">MSDN</asp:HyperLink></p>
<p>
    <asp:HyperLink ID="HyperLink2" Runat="server"
NavigateUrl="www.gotdotnet.com">GotDotNet</asp:HyperLink>
</p>
<p>
    <asp:HyperLink ID="HyperLink3" Runat="server"
NavigateUrl="www.google.sk">Google</asp:HyperLink>
</p>
```

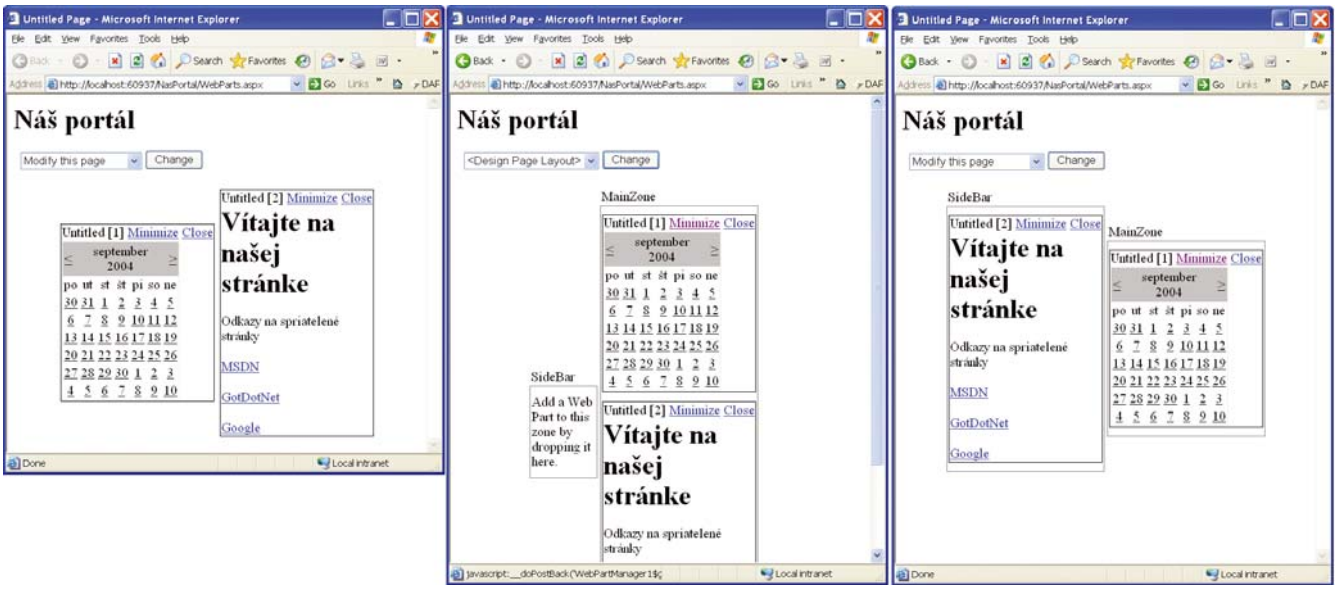
Ako už bolo avizované, tento WebUserControl umiestnime do zóny v poli tabuľky MainZone. Realizácia tejto myšlienky je nanajvýš jednoduchá, stačí presunúť ikonu súboru WebUserControl.ascx z okna Solution Explorer na plochu MainZone. Situáciu znázorňuje obrázok. Po týchto úpravách je možné projekt po prvý krát spustiť a prezrieť si výsledky doterajšieho snaženia. Zatiaľ sme nedosiahli nič, žiadnu pridanú hodnotu ani funkcionalitu čo vy sme neboli dokázali vytvoriť pomocou vizuálneho návrhu jednoduchej aplikácie. No k funkcionalite portálu, alebo portálového segmentu to má pomerne ďaleko.



Umiestnenie prvkov Calendar a Web User Control do Web Part Zone.

Prvé výhody Web Parts sa však začnú črtať po pridaní prvku WebPartPageMenu. Tento prvok sme v našej aplikácii umiestnili pod prvok WebPartManager a vizuálne sa javí ako Drop Down List s preddefinovanými možnosťami.
– Modify This Page
– Browse This Page
– Design Page Layout

Pridaním prvku WebPartPageMenu sme sa posunuli o veľký kus vpred, takže je dokonca na mieste znovuspustenie aplikácie a vyskúšanie si jej fungovania.
Po prepnutí Web Part Managera do módu Design Page Layout môžeme prispôsobiť stránku alebo jej časť svojim predstavám a napríklad metódou drag and drop presúvať prvky medzi jednotlivými zónami. Na obrázku vidíme časť možností, kedy sme (vstrede) umiestnili obidva prvky - kalendár aj nami nadefinovaný prvok s odkazmi do jednej zóny. Na obrázku vpravo sme tieto prvky navzájom medzi zónami vymenili. Samozrejme zo vzrastajúcim počtom zón a prvkov na skutočných portáloch vzrastá aj počet kombinatorických možností.



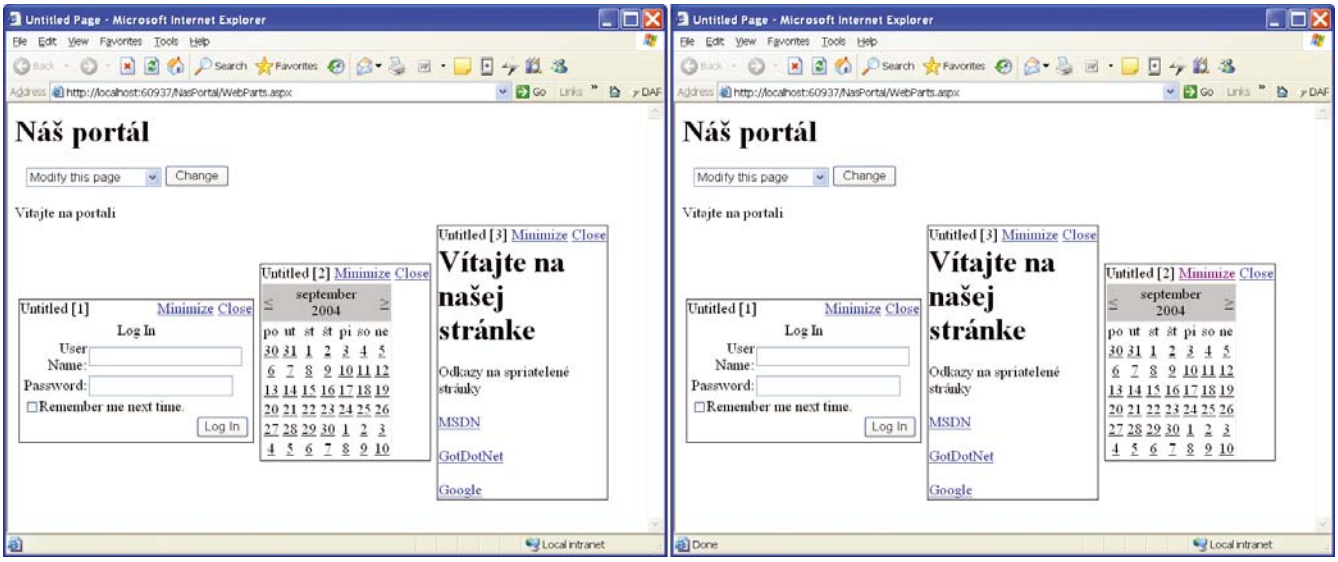
On - line zmena návrhu WebPartPageMenu

Takýmto spôsobom si môže každý prispôsobiť vizuál portálu svojim požiadavkám. Personalizácia samozrejme nie je možná bez autentifikácie. Veď prečo by sme sa trápili s prispôsobením vizuálu portálu na ktorý často chodím, keby naše zmeny neboli zapamätané a viazané práve na naše prihlásenie sa. Zopakujeme si vytvorenie prihlasovacieho dialógu zo siedmej kapitoly.

Z toolboxu zo sekcie Login pridáme na plochu aplikácie komponentu LoginView. Môžeme ju umiestniť niekam do záhlavia na viditeľné miesto napríklad nad tabuľku. V menu Common Tasks je potrebné zvoliť najskôr Anonymous Template a na plochu komponenty doplniť napríklad text „Nie ste prihlásený k aplikácii.“ Potom prepneme na LoggedIn Template a na plochu komponenty umiestnime povedzme text „Vitajte na portáli“.

Do ľavého, zatiaľ voľného miesta umiestnime Prvok WebPartZone a do jeho vnútra komponentu Login. Nezabudnime v súbore Web.Config zmeniť spôsob autentifikácie z `<authentication mode="Windows"/>` na `<authentication mode="Forms" />`

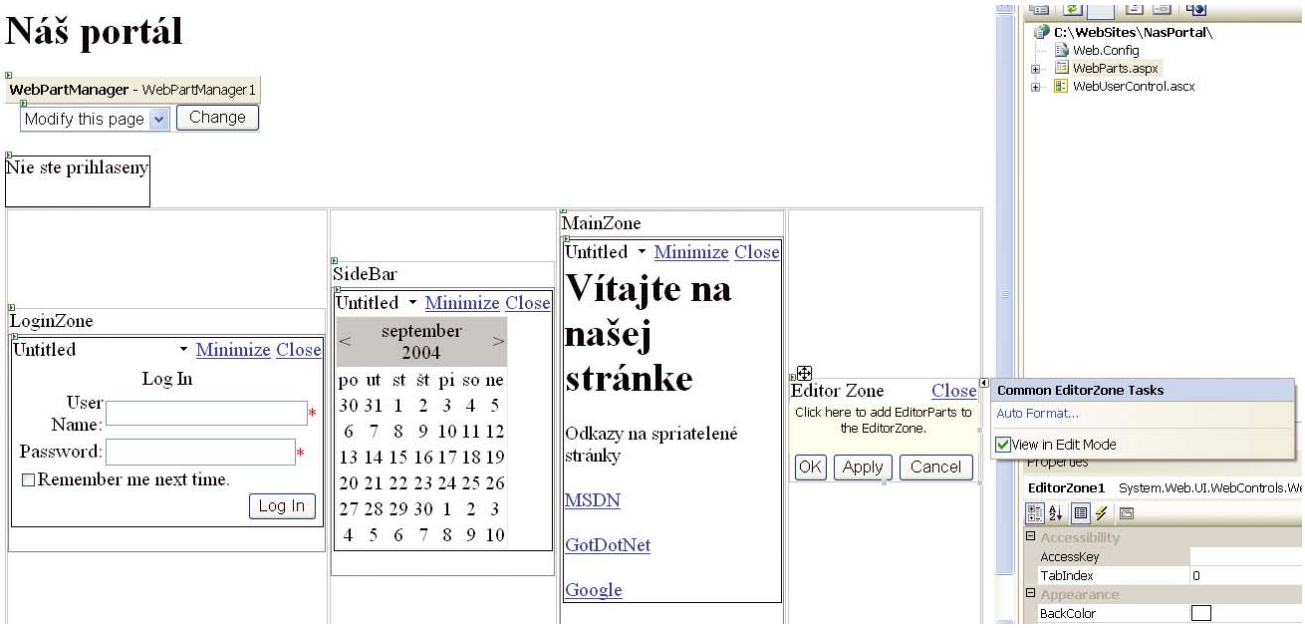
Prístupové parametre nadefinujeme pomocou administrátorského nástroja, ktorý aktivujeme buď v komponente LoginView pomocou Common Tasks menu (položka Administrate website) alebo v projekte prostredníctvom menu Website - ASP.NET Configuration. V tomto nástroji môžeme pre demonštráciu možností personalizácie vytvoriť dvoch používateľov. Následný test je jednoduchý. Najskôr sa prihlásime ako prvý používateľ a upravíme vizuál portálu. Potom sa prihlásime ako druhý používateľ a urobíme iné zmeny. Od tohoto okamihu si každý z používateľov pri ďalšom prihlásení nájde portál v takom stave ako si ho prispôsobil. Pre internetové portály je zaujímavá aj skutočnosť, že ASP.NET poskytuje aj možnosť identifikácie pomocou automaticky vydaného Cookie. V takomto prípade je možná personalizácia aj bez priamej autentifikácie pomocou mena a hesla.



Porovnanie personalizovaného vizuálu portálu pre dvoch rôznych používateľov

EditorZone

V doterajšej časti príkladu sme ukázali len zlomok možností personalizácie. Nastal čas použiť štvrtú dosiaľ prázdnu bunku tabuľky úplne vpravo. Do nej presunieme prvok EditorZone. V jeho Common Task menu zaškrtneme možnosť View in Edit Mode. Do vnútra zóny umiestnime prvok AppearanceEditorPart.



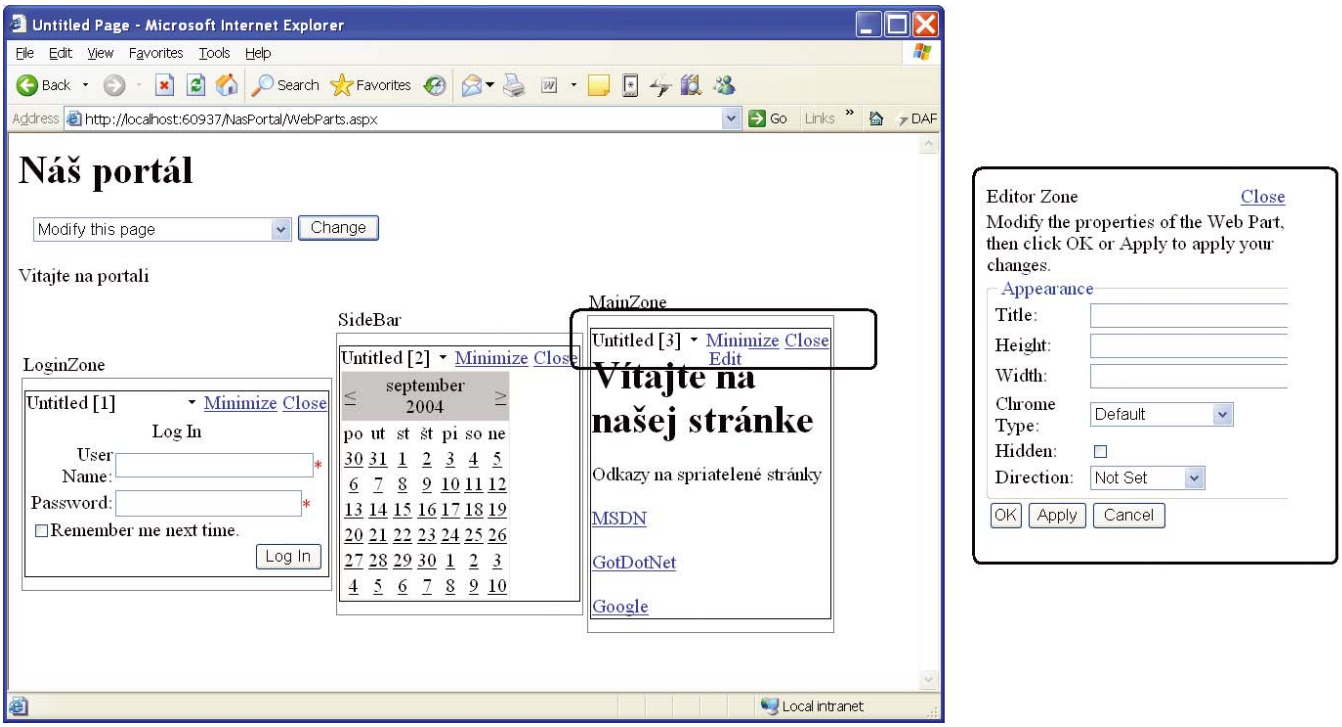
Pridanie komponenty EditorZone

Po spustení aplikácie sa žiadne zmeny súvisiace s pridaním editačnej zóny neprejavia a pokiaľ sa k potrálu neprihlásime, tak ani metódou pokus - omyl neprídeme na spôsob aktivovania editačnej zóny pre personalizáciu niiektorej WebParts komponenty. Ale veď to ani nie je potrebné. Personalizácia má predsa význam len po autentifikácii. Aby sme sa priblížili k cieľu, môžeme porovnať možnosti ponúkané v comboboxe prvku

WebPartPageMenu. Pokiaľ nie sme k portálu prihlásení máme k dispozícii tie isté možnosti, ktoré boli popísané v predchádzajúcom odseku:

– Modify This Page
– Browse This Page
– Design Page Layout

Po prihlásení sa ako ľubovoľný z nadefinovaných používateľov k týmto trom položkám pribudne ďalšia - Modify the Web Part Settings. Ak ju aktivujeme vedľa názvu zóny sa zobrazí malý nadol smerujúci trojuholník, ktorý nám sprístupní položku menu Edit. Keď sa nad tým zamyslíme, je to logický a prepracovaný postup.

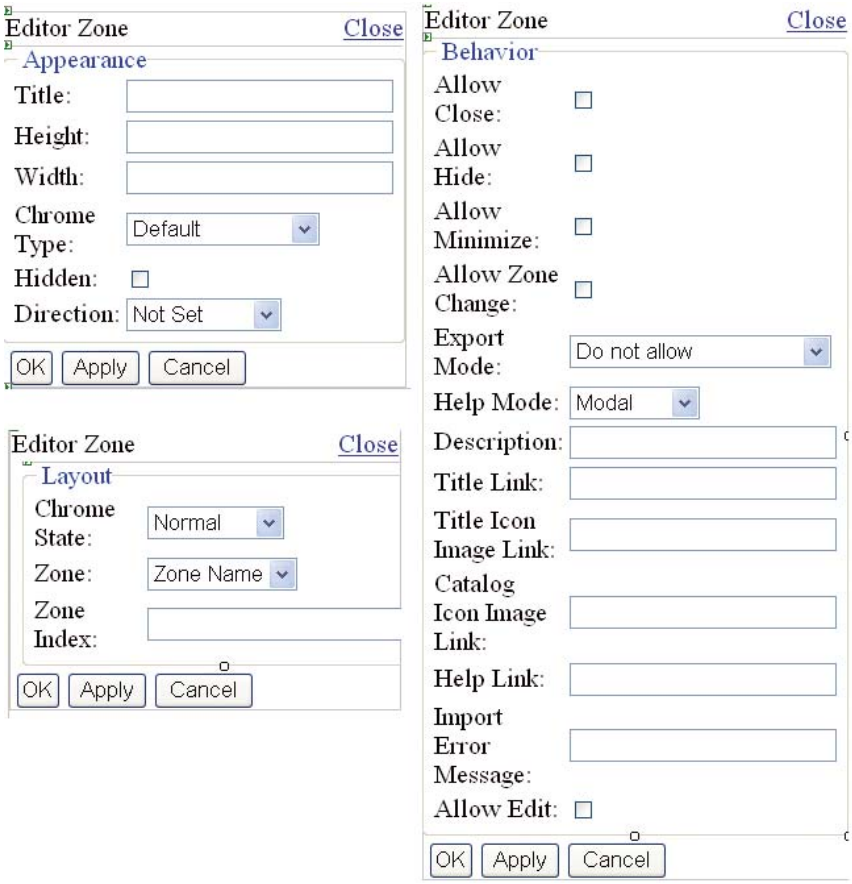


Aktivovanie komponenty EditorZone

V takejto podobe editačnej zóny s jedným prvkom typu EditorPart dokážeme príslušnú zónu personalizovať len čiastočne. Prvky typu Editor Part sú však až tri

– AppearanceEditorPart
– BehaviorEditorPart
– LayoutEditorPart

a pri správnej kombinácii už ponúkajú slušnú paletu možností.



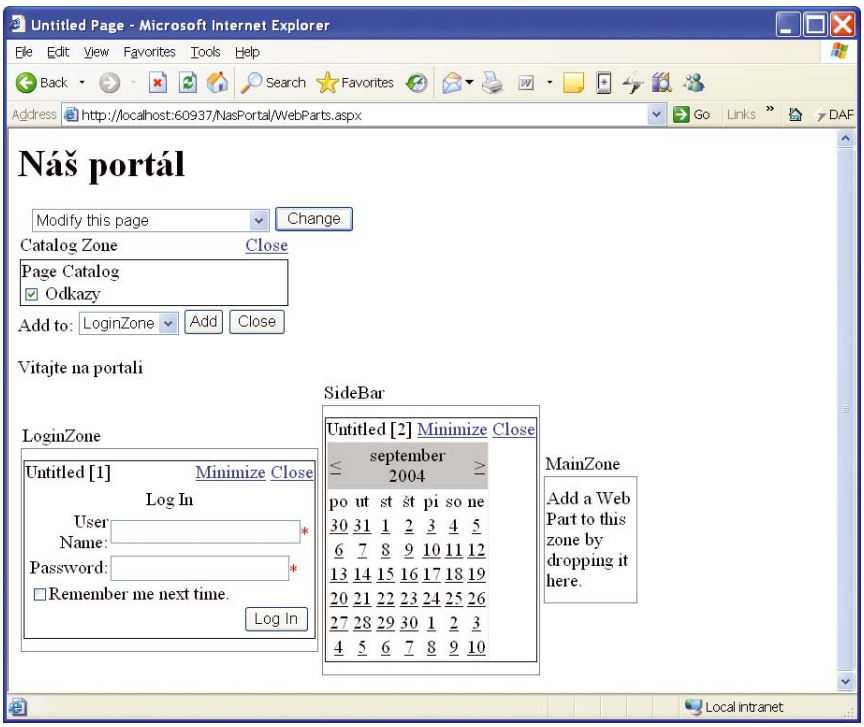
Možnosti editácie ponúkané cez Appearance, Behavior a Layout Editor Part

CatalogZone

Určite ste si všimli, že v modifikačnom móde je v každej zóne aj link Close. Po jeho aktivovaní zóna z zobrazovky portálu jednoducho zmizla. Toto kúzlo bolo dosiaľ bohužiaľ nevratné, presnejšie povedané trvalo až do ďalšieho spustenia aplikácie, alebo prihlásení sa nového používateľa. Asi tušíte, že bude k dispozícii nejaký kontejner, v ktorom sú nezobrazené zóny uložené a z tohoto kontejnera je ich možné vybrať a zobraziť. Pre tento účel slúži catalogZone. Postup pri pridávaní katalógovej zóny bude veľmi podobný doterajším postupom. Zónu umiestnime napríklad do priestoru nad tabuľku. V jej Common Tasks menu označíme voľbu View in Catalog menu. Po tomto úkone sa vytvorí priestor pre vloženie komponenty PageCatalogPart.

CatalogZone

Po spustení aplikácie a prihlásení sa k portálu sa kľúč k využitiu novo pridanej zóny skrýva v comboboxe prvku WebPartPageMenu. Pribudla tam možnosť - Add Web Parts to this Page. Predtým ako ju aktivujeme môžeme niektoré zóny pozatvárať. Tieto sa objavia v ponuke katalógu a odtiaľ ich môžeme opäť zobraziť.



Pridávanie a odoberanie Web Parts

Po predstavení niektorých možností Web Parts si teraz už určite dokážeme predstaviť akým spôsobom by sme ich vedeli využiť pri tvorbe personalizovanej portálovej aplikácie.

Pridávaním ďalších zón a navrhovaním ich obsahu postupne dotvoríme celý portál. No keď sa na spustenú aplikáciu pozrieme z hľadiska vkusu a estetiky, portál v takejto podobe by zaujal možno len spolok askétov, prípadne technologicky orientovaných scientológov. Aj keď vo funkčnosti je určitý druh krásy, v takejto podobe by portál svojou estetikou nezaujal. To že sme sa dosiaľ estetikou jednotlivých prvkov príliš nezaoberali, naznačuje že sa to bude dať kedykoľvek jednoducho napraviť.

Všimnime si prvý riadok v súboru WebParts.aspx

```
<%@ Page Language="C#" CompileWith="WebParts.aspx.cs" ClassName="WebParts_aspx" %>
```

do tohto riadku pridáme definíciu štýlu

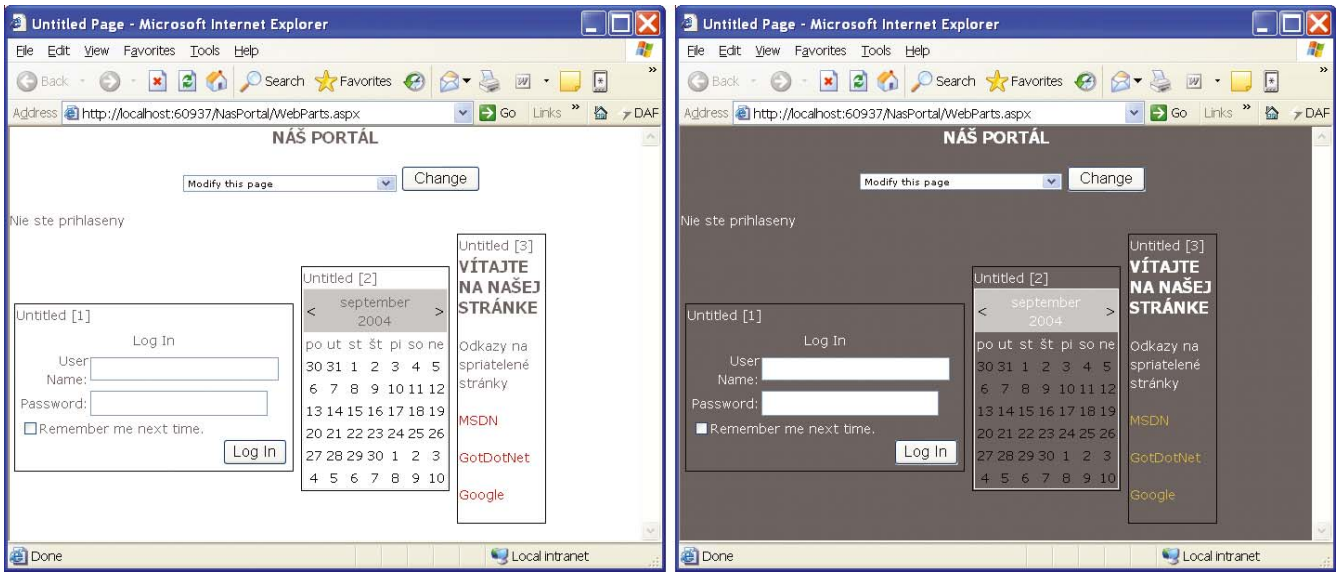
```
<%@ Page Language="C#" StylesheetTheme="White" CompileWith="WebParts.aspx.cs"
ClassName="WebParts_aspx" %>
```

alebo

```
<%@ Page Language="C#" StylesheetTheme="Black" CompileWith="WebParts.aspx.cs"
ClassName="WebParts_aspx" %>
```

Toto je však len prvý krok, použitie témy. Ak by sme sa v tomto okamihu pokúsili projekt preložiť a zostaviť, nepodarilo by sa nám to. Požadovanú tému musíme do projektu najskôr pridať. Najjednoduchšia cesta ako získať dve témy je niekde pomimo vytvoriť nový projekt typu Personal Web Site a z adresára tohto projektu prekopírovať do adresára projektu portálu podadresár Themes, v ktorom sú dve jednoduché témy White a Black. Tieto témy môžeme neskôr (samozrejme zo znalosťou veci a hlavne kaskádových štýlov) modifikovať, prípadne nejaké témy stiahneme z iných cvičných projektov z webu. Pomerne rozšírená je napríklad téma BasicBlue.

Po spustení vidíme, že nastal významný krok k vylepšeniu vzhľadu aplikácie. Samozrejme veľa zostáva aj na návrhárovom grafickom cítení, aby výsledný dizajn portálu bol účelný a estetický



Pridávanie a odoberanie Web Parts

Kapitola 10:

Vývoj ASP.NET aplikácií pre mobilné zariadenia

Jednou z najdynamickejšie sa rozvíjajúcou oblasťou webových aplikácií sú aplikácie pre mobilné zariadenia. Aj pri vývoji webových aplikácií by sme mali počítať s tým, že ak vytvoríme skutočne úspešnú aplikáciu, budú k nej chcieť pristupovať aj klienti z mobilných zariadení. Okrem aplikácií pre „klasického“ tenkého klienta teda prehliadač webových stránok bude čoraz častejšie potrebné vyvíjať aj webové aplikácie ku ktorým sa bude pristupovať z mobilných tenkých klientov, teda z prístrojov PDA (osobný digitálny asistent) a mobilných telefónov. Preto by sme mali počítať aj s variantov ASP.NET stránok pre mobilné zariadenia. Tieto zariadenia sa v ASP.NET, teda webových, alebo intranetových aplikáciách využívajú ako tenký klient. Pod pojmom tenký klient rozumieme spravidla počítač, kde sa pre prístup k údajom používa prehliadač webových stránok. To znamená, že celá aplikačná logika beží na serveri a výpočtové možnosti klientského počítača viac - menej ležia ľadom, využívajú sa hlavne pre zobrazenie a prípadný beh skriptov na strane klienta.
Cieľom nášho snaženia v tejto kapitole bude vývoj serverovej aplikácie s použitím technológie ASP.NET 2.0 vo vývojovom prostredí Visual Web Developer

Typy a platformy mobilných zariadení

Aby sme mali prehľad o jednotlivých triedach mobilných klientov uvedieme najskôr stručný prehľad platforiem z dielne Microsoftu, teda Pocket PC 2003 , Pocket PC Phone Edition a Smartphone 2003

Platformy mobilných zariadení s operačným systémom od Microsoftu

Pocket PC 2003

Túto platformu azda netreba príliš predstavovať. Je to prístroj s farebným dotykovým displejom a niekoľkými ďalšími ovládacími prvkami. Presadil sa ako personálny digitálny asistent. Rozšírenie komunikačných možností, napríklad o Wireles LAN, Bluetooth, GSM/GPRS docielime externou kartou (PCMCIA, CF,ba dokonca aj SD). Novšie prístroje majú niektoré zo spomínaných komunikačných modulov už vstavané. Ak by sme odbočili mimo platforiem Microsoftu veľmi úspešné zariadenia podobného form faktora ale s úplne inými operačnými systémami sú dodávané aj inými firmami (Palm, Handspring). Napriek úplnej nekompatibilite operačných systémov, môžeme vyvíjať mobilné ASP.NET aplikácie aj pre tieto prístroje, pretože záleží len na tom aby mobilné zariadenie malo prehliadač webového obsahu, alebo WAP stránok.

Pocket PC Phone Edition

Zjednodušene sa dá povedať, že prístroj triedy Pocket PC 2003 Phone Edition je klasické PDA s integrovaným GPRS mobilným telefónom. Predchádzajúce typy bolo možné na prvý pohľad spoznať podľa anténky. Najnovšie prístroje už majú anténu integrovanú priamo do tela prístroja. Možno lepším vodítkom pre rozpoznanie by bol slot na SIM kartu a pochopiteľne technické údaje prístroja.

Smartphone 2003

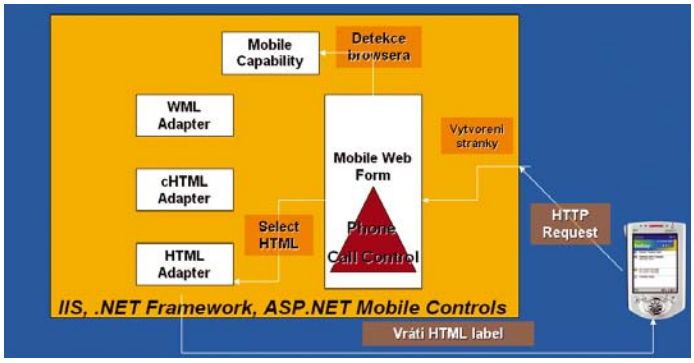
Jedná sa o mobilný telefón, teda presnejšie o prístroj ktorý konštrukciou oveľa viac pripomína mobilný telefón než PDA. Platforma Smartphone 2003 vychádza v podstate z Windows CE (Pocket PC) pričom by mala kombinovať to najlepšie z PDA a mobilných telefónov. Táto platforma by mala poskytovať inteligentné spojenie, to jest hlasom, e-mailom alebo inými prostriedkami a čo je najdôležitejšie kdekoľvek a kedykoľvek. Smartphone okrem elektronickej pošty ponúka aj možnosť prehliadania webového obsahu. Využíva pritom také používateľské rozhraní, aké zákazníci od mobilného telefónu očakávajú. Platforma Smartphone 2003 nakoľko je implementovaná v mobilných telefónoch má určité špecifiká, hlavným rozdielom oproti Pocket PC 2002 je absencia dotykového displeja. Mobilné telefóny sa totiž ovládajú hlavne klávesnicou a aj tá býva počas nosenia vo vrecku zablokovaná, takže dotykový displej bez odklopného krytu by bol nezmysel. Displej bez možnosti ovládania dotykom môže mať jednak odolné čelné sklo, ale zníži sa tým aj cena zariadenia a jeho spotreba. Rozlíšenie displeja je spravidla 176 x 220 pixelov. Displej môže byť farebný alebo čiernobiely.

Mobilný telefón podporujúci protokol WAP

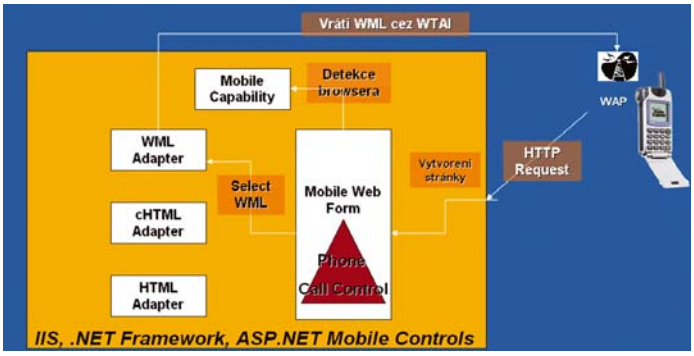
História protokolu WAP (Wireless Application Protocol) siaha len do roku 1997. Vtedy, 26 júna najvýznamnejší výrobcovia mobilných telefónov založili Fórum pre WAP. Definícia protokolu popisuje dátovú komunikáciu medzi mobilným zariadením a sieťou mobilného operátora. Ako ekvivalent internetových prehliadačov známych z prostredia PC (Internet Explorer, Netscape Navigator, Opera...) bol navrhnutý prehliadač pre bezdrôtové aplikačné prostredie WAE (Wireless Application Environment). Prehliadač je založený na princípe klient/server architektúry a „tenkého“ klienta. Mobilný telefón musí mať aj primerane veľký displej a vhodne navrhnuté ovládacie prvky pre surfovanie. Typické rozlíšenie displejov mobilných telefónov je asi 90 x 60 pixelov. Takéto rozlíšenie displeja samozrejme na zobrazenie klasickej HTML stránky nestačí. Praktické minimum pre zobrazenie HTML stránky, vrátane grafiky, zrejme predstavuje displej handheldu, ktorý má rozlíšenie 240 x 320 pixelov. Preto musia byť webové stránky prenášané protokolom WAP na mobilné telefóny navrhnuté s prihliadnutím na rozlišovaciu schopnosť ich zobrazovacích jednotiek.

Stručná história vývoja aplikácií pre mobilných tenkých klientov

Prvá verzia .NET Frameworku spojená s prvou verziou vývojového prostredia Visual Studio.NET vývoj mobilných ASP.NET aplikácií nepodporovala. Neskôr sa objavila beta verzia nástroja Microsoft Mobile Internet Toolkit (MMIT). Táto verzia nikdy nebola nasadená komerčne, nakoľko pod názvom ASP.NET Mobile Controls bola priamo implementovaná do .NET Frameworku 1.1 a nasledujúcej verzie vývojového prostredia Visual Studio .NET 2003. Pre prípadných záujemcov z radov používateľov starého Visual Studia, MMIT je možné stiahnuť z webu Microsoftu, konkrétne zo sekcie Download z adresy http://download.microsoft.com/download/VisualStudioNET/Install/RC/NT45XP/EN-US/MobileIT.exe Inštalačný súbor má len niečo vyše 4MB. Pre vývoj a ladenie aplikácií pre platformu Pocket PC budeme ešte potrebovať pravdepodobne emulátor Pocket PC. Môžeme využiť emulátor z voľne šíriteľného vývojového prostredia eMbedded Visual Tools. Tento emulátor môžeme voľne stiahnuť z webovej adresy http://www.microsoft.com/mobile/downloads/emvt30.asp Pre emulovanie WAP telefónov môžeme použiť napríklad Openwave emulátor, ktorý môžeme získať z webu z adresy www.openwave.com. V najnovšej verzii ASP.NET stránok bola prijatá nová koncepcia. Namiesto špeciálnych ovládacích prvkov pre vytváranie formulárov určených pre mobilné zariadenia je možné použiť tie isté prvky ako pre klasické aplikácie. Na základe rozpoznaného typu klientovho prehliadača webových stránok z ktorého bola vznesená požiadavka na vygenerovanie stránky. sa zapojí do hry niektorý z adaptérov, v našich končinách najčastejšie HTML adaptér, alebo WML adaptér pre WAP stránky a vyrendruje príslušný typ stránky. Túto technológiu implementovanú v ASP.NET 2.0 nazývame adaptívnym rendrovaním. Na obrázkoch vidíme generovanie WML stránky pre WAP telefón a HTML stránky pre Pocket PC alebo pre klasické PC

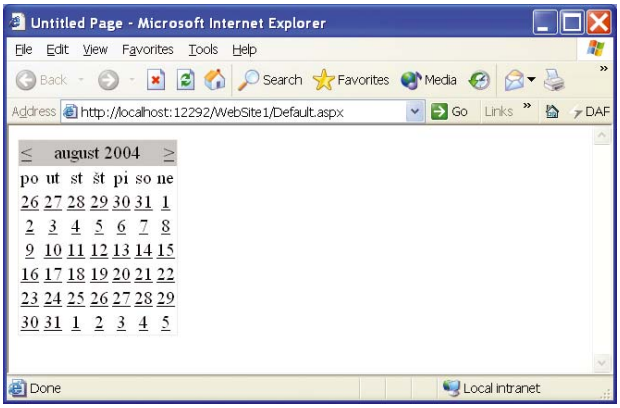


Rendrovanie HTML stránky pre Pocket PC



Rendrovanie WML stránky pre WAP telefón

Ako to vyzerá v praxi sa môžeme presvedčiť na jednoduchom príklade ASP.NET aplikácie, kedy na plochu formulára pridáme kalendárovú komponentu. Táto komponenta je pomerne zložitá a zapúzdruje niekoľko činností, ktoré vedú k výberu konkrétneho dátumu. Po spustení aplikácie si necháme zobraziť formulár najskôr v klasickom prehliadači HTML stránok. Zobrazená komponenta je presne taká istá ako v návrhovom zobrazení.



Zobrazenie komponenty Calendar v klasickom prehliadači HTML stránok

Na prvý pohľad sa môžeme domnievať že sa jedná o ActiveX komponentu, alebo Flash, ak si pozrieme zrojový kód stránky - jedná sa o poctivý HTML kód v kombinácii s JavaScriptom.

Skrátená zdrojovka HTML stránky s kalendárom, ktorá sa zobrazí u klienta bude

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head><title>
      Untitled Page
</title></head>
<body>
    <form method="post" action="Default.aspx" id="form1">
<div>
<input type="hidden" name="__EVENTTARGET" value="" />
<input type="hidden" name="__EVENTARGUMENT" value="" />
<input type="hidden" name="__VIEWSTATE"
value="/wEPDwULLTE4MTQ4MDc0MDRkZJrFZNMN6Lo0dFpqG/HuqMgpsdob" />
</div>

<script type="text/javascript">
<!--
var theForm = document.forms['form1'];
function __doPostBack(eventTarget, eventArgument) {
```

```
    if (theForm.onsubmit == null || theForm.onsubmit()) {
        theForm.__EVENTTARGET.value = eventTarget;
        theForm.__EVENTARGUMENT.value = eventArgument;
        theForm.submit();
    }
}
// -->
</script>


    <div>
        <table id="Calendar1" cellspacing="0" cellpadding="2" border="0"
style="border-width:1px;border-style:solid;border-collapse:collapse;">
     <tr><td colspan="7" style="background-color:Silver;"><table cellspacing="0"
border="0" style="width:100%;border-collapse:collapse;">
          <tr><td
style="width:15%;"><a href="javascript:__doPostBack('Calendar1','V1643')"
style="color:Black">&lt;</a></td><td align="center" style="width:70%;">august
2004</td><td align="right" style="width:15%;"><a ...
...
</table>

    </div>
    </form>
</body>
</html>
```

Ak na tú istú adresu pristúpime pomocou mobilného zariadenia Pocket PC, alebo jeho emulátora, zobrazená komponenta bude vyzerať prakticky rovnako.

Zobrazenie komponenty Calendar v emulátore Pocket PC 2003

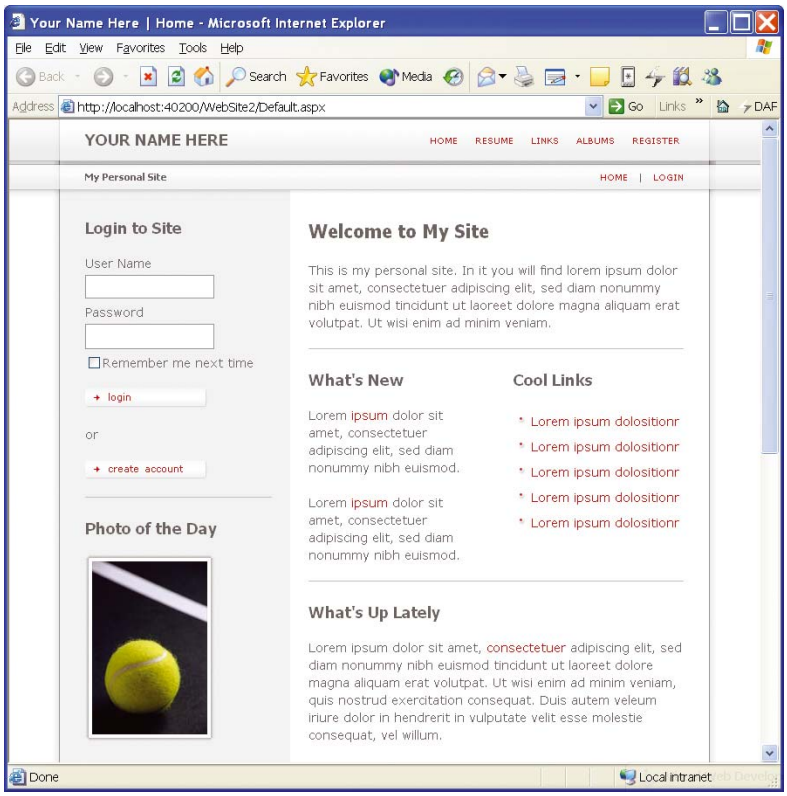
Ak však k tej istej stránke s kalendárovou komponentou pristúpime z mobilného telefónu podporujúceho WAP, prípadne z emulátora takéhoto telefónu, vyrendrovaná stránka pre takéto zariadenie bude akceptovať monochromatický displej s nízkym rozlíšením a obmedzenia vyplývajúce z ovládania WAP telefónu

Zobrazenie komponenty Calendar v emulátore Openwave

Pre zaujímavosť zdrojový kód stránky bude

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Untitled Page</title>
</head>
<body>
<form method="post" action="Default.aspx" id="form1">
<div>

<input type="hidden" name="__VIEWSTATE"
value="/wEPaA8FDzhjNjRiM2FkMmEzNTRlMGRVasLvbuq+4q23p0lqodT7Qk4p3g==" />

    <div>

<a href="Default.aspx?__VIEWSTATE=%2fwEPaA8FDzhjNjRiM2FkMmEzNTRlMGRVasLvbuq%2b4q23p0l
qodT7Qk4p3g%3d%3d&amp;__EVENTTARGET=Calendar1&amp;__EVENTARGUMENT=1">Calendar</a>

    </div>
    </div>
</form>
</body>
</html>
```

Ak urobíme ten istý pokus s typickou HTML stránkou, napríklad z internetovým portálom alebo s osobnu stránkou vygenerovanou Personal Web Site Starter kitu, navrhnutej tak aby maximálne využila displej klasického PC



Zobrazenie osobnej stránky v klasickom prehliadači

zobrazenie na displeji Pocket PC bude na hranici používateľského komfortu

Zobrazenie osobnej stránky v emulátore Pocket PC 2003

Zobrazenie na displeji WAP telefónu však už bude prakticky nepoužiteľné.

Zobrazenie osobnej stránky v emulátore WAP telefónu

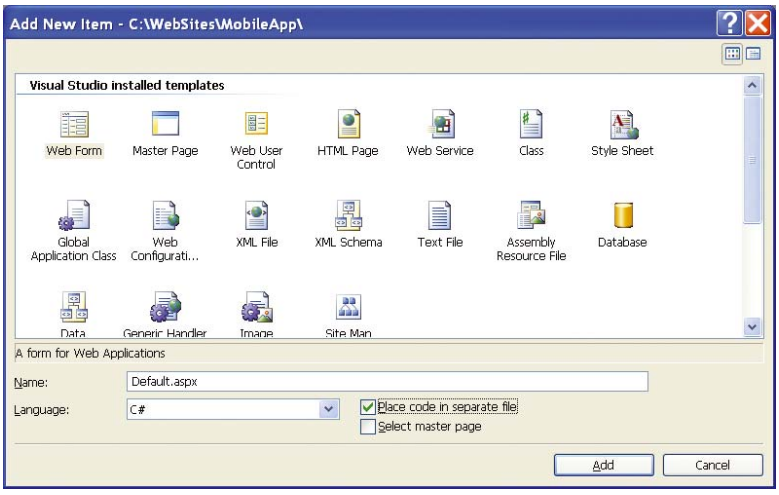
Návrh aplikácie typu „Multiview“ pre mobilné zariadenia

Na základe predpokladov a kritérií z predchádzajúcej state, ktoré by sme mohli zhrnúť do dvoch bodov

– malý displej
– obmedzené možnosti ovládania

sa pokúsime vytvoriť aplikáciu, ktorá bude vhodná (ak nie priam predurčená) pre mobilné zariadenia. Dôležitá je technológia, nie námet, preto námetom bude tak jednoduchá vec ako jednoúčelová kalkulačka, ktorá z dvoch zadaných čísel vypočíta aritmetický priemer.

Začiatok je štandardný - vytvoríme aplikáciu typu Empty Web Site, napríklad v jazyku C#. Do aplikácie pridáme webový formulár, pričom zaškrtneme voľbu Place code in separate file.

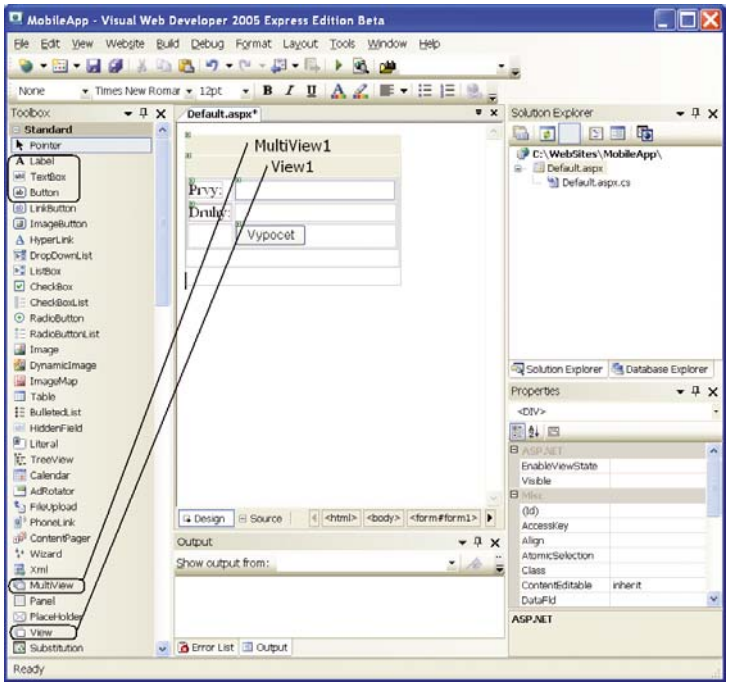


Pridanie webového formulára do aplikácie

Návrh zadávacieho formulára

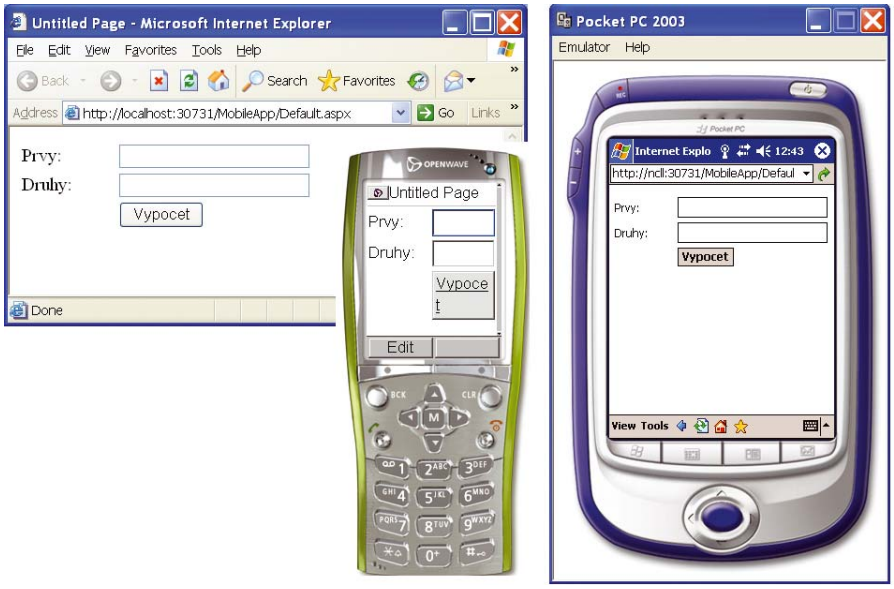
Dizajnovú kostru aplikácie bude tvoriť ovládací prvok MultiView. Tento prvok presunieme z toolboxu na plochu formulára. Do vnútra prvku MultiView presunieme prvok View. Po tomto úkone zmeníme v okne Properties hodnotu parametra Active View Index u prvku MultiView z -1 na hodnotu 0. Po týchto prípravných úkonoch sa môžeme pustiť do vlastného návrhu formulára. Grafickú úpravu formulára, kedy vľavo bude popis a vpravo editačné okienko pre

zadávanie údajov nám pomôže dodržať tabuľka. Ako aktívny označíme prvok View a pomocou menu Layout - Insert Table vložíme tabuľku s tromi riadkami, dvomi stĺpcami a zarovnávaním buniek doľava. Následne pomocou prvkov Label, TextBox a Button navrhneme formulár pre zadávanie údajov.



Návrh dizajnu formulára aplikácie

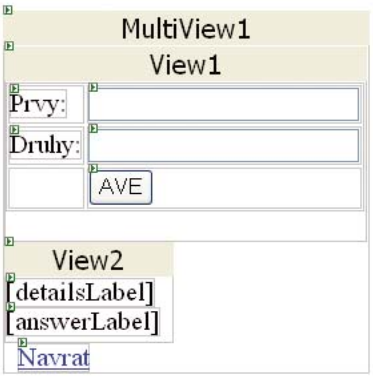
Keď máme kostu grafického návrhu hotovú, môžeme sa pokúsiť aplikáciu spustiť. Môžeme tak urobiť priamo z vývojového prostredia Visual Web Developer, kedy sa aplikácia spustí v okne Internet Explorera. URL adresu z Explorera môžeme potom použiť v emulátoroch ostatných mobilných zariadení, prípadne v pripojených reálnych zariadeniach. Na základe zobrazenia úsúdime, či je aplikácia na týchto typoch zariadení použiteľná. V našom prípade sa na emulátore mobilného telefónu nezmestilo tlačidlo „Výpočet“ na displej (bolo zobrazené v dvoch riadkoch), preto skrátime text na tomto tlačidla ne AVE (od anglického average), čím sa zmenší šírka tlačidla.



Zobrazenie na rôznych typoch zariadení

Návrh druhého formulára pre zobrazenie výsledkov

Robiť si starosti s aplikačnou logikov v prípade aplikácie počítajúcej priemer dvoch čísel naozaj nemusíme, preto môžeme pristúpiť k návrhu druhého pohľadu pre zobrazenie výsledkov. Do vnútra prvku MultiView presunieme ďalší prvok View



Návrh dizajnu druhého formulára vo vnútri prvku MultiView

Teraz už stačí doplniť aplikačnú logiku do obsluhy udalosti zatlačenia tlačidla

```
void Button1_Click(object sender, EventArgs e)
{
     double prvy = double.Parse(this.TextBox1.Text);
     double druhy = double.Parse(this.TextBox2.Text);
     double priemer = (prvy + druhy) / 2;
     this.detailsLabel.Text = "Priemer cisel " + this.TextBox1.Text + " a " +
this.TextBox1.Text + " je:";
     this.answerLabel.Text = priemer.ToString();
     this.MultiView1.ActiveViewIndex = 1;
}
```

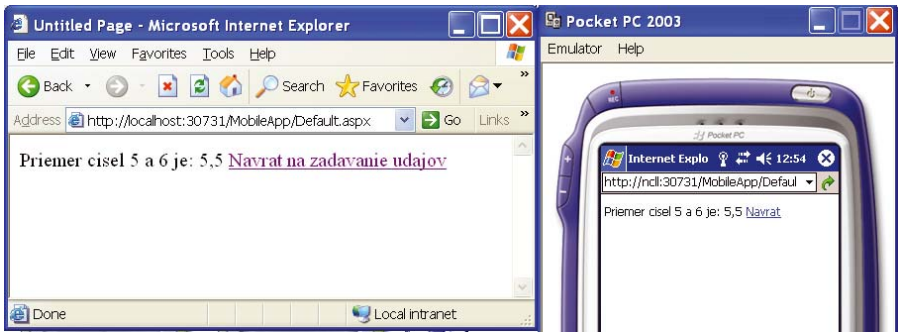
Beh aplikácie na rôznych typoch zariadení

V úvode kapitoly sme hovorili o rozdielnom rendrovaní stránok aplikácie pre rôzne typy zariadení. Toto rendrovanie môžeme aj sami ovplyvniť

```
<asp:LinkButton ID="LinkButton1" Runat="server"
OnClick="LinkButton1_Click">Navrat</asp:LinkButton>
```

stačí ak predpíšeme pre Internet Explorer iný text

```
<asp:LinkButton ID="LinkButton1" Runat="server" OnClick="LinkButton1_Click" Text
="Navrat" ie:Text="Navrat na zadavanie udajov"></asp:LinkButton>
```



Rôzne zobrazovanie na rôznych typoch zariadení

Vývoj databázovej aplikácie pre mobilné zariadenia

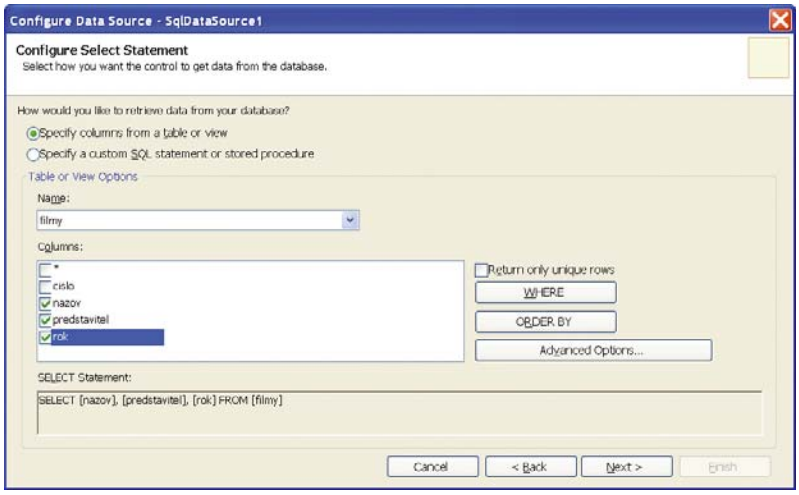
Nakoľko ASP.NET aplikácia beží na serveri, môže pristupovať prakticky k neobmedzenému množstvu údajov v rôznych databázach a tieto údaje vo vhodnej forme sprístupniť klientovi na jeho mobilné zariadenie. Postup pri vývoji je veľmi podobný vývoju klasickej databázovej ASP.NET aplikácie, rozdiel bude len v zobrazovaní údajov.

Začíname podobne ako v predchádzajúcom prípade - vytvoríme aplikáciu typu Empty Web Site, napríklad v jazyku C#. Do aplikácie pridáme webový formulár, pričom zaškrtneme voľbu Place code in separate file.
Špecifiká návrhu databázovej aplikácie sa začínajú až v okamihu presunu komponenty SqlDataSource na plochu formulára. V menu Common tasks tejto komponenty aktivujeme položku Configure Data Source. V našom prípade budeme pracovať s vlastnou tabuľkou filmy (vytvorená v kapitole venovanej vývoju databázových aplikácií) , takže pripojovací reťazec v našom prípade bol

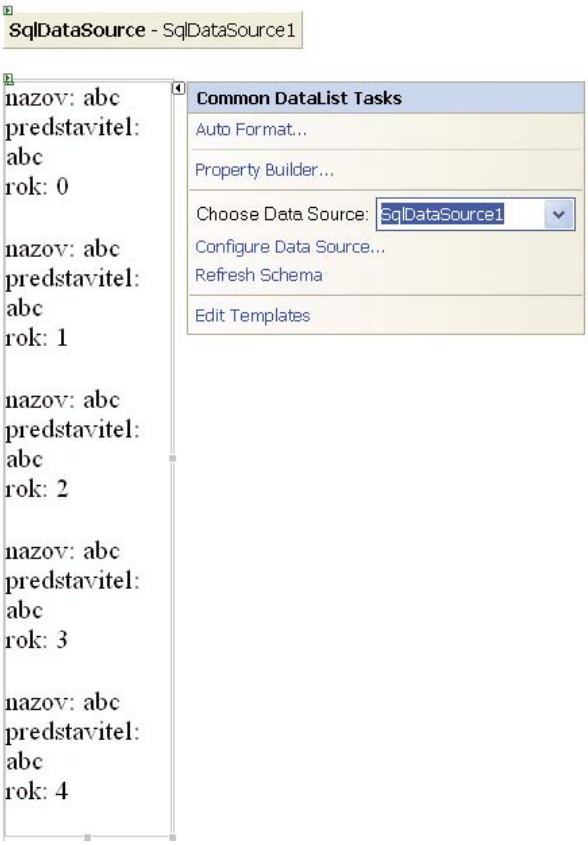
```
Server=ncll\SQLExpress;Integrated Security=True;Database=TEST
```

a SQL dotaz navrhnutý v dialógu Configure Data Source

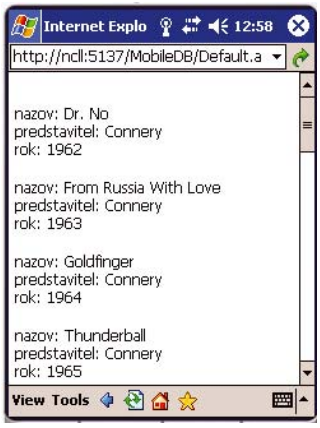
```
SELECT [nazov], [predstavitel], [rok] FROM [filmy]
```



Výber údajov z databázovej tabuľky



Komponenta DataList vo vnútri komponenty Panel pripojená na zdroj údajov



Zobrazenie výsledkov

Kapitola 11:
Webové služby

Skôr než vedieť presne a exaktne definovať webovú službu, je užitočnejšie vedieť na čo sa dajú webové služby použiť a ako sa dajú vytvárať. Napriek komplexnosti riešení, ktoré webové služby umožňujú, ide z hľadiska vývojára (aspoň v začiatkoch) o veľmi jednoduchú problematiku. Všeobecný princíp je asi takýto. Naša spoločnosť má prívlastok informačná a preto ľudia aj firmy sa delia do dvoch kategórií. Na tých čo informácie majú a sú ochotní (väčšinou za nejakú protihodnotu, alebo z propagačných a marketingových dôvodov) ich poskytnúť, a tých, ktorí o tieto informácie majú záujem. Vysvetlíme to na jednoduchom príklade. Na Slovensku vychádza veľa internetových denníkov, a nachádza sa tu aj mnoho portálov, na ktoré denne pristupujú klienti. Všetci prevádzkovatelia takýchto denníkov a portálov zverejňujú denne správy o počasí. Získavajú ich rôzne a vkladajú na svoje stránky. Tieto informácie však majú jednotný zdroj, buď sa jedná o hydrometeorologický ústav, alebo nejakú inú firmu, ktorá sa zaoberá predpoveďou počasia. Ak by takáto firma poskytla za rozumných obchodných podmienok údaje o predpovedi počasia, prevádzkovatelia internetových denníkov a portálov by ju určite radi využívali. Ide len o to, ako na to? Momentálny stav je zrejme taký, že poskytovateľ predpovede poskytne túto v nejakom formáte, napríklad textovom, alebo binárnom a webmastri to po prípadnej konverzii povkladajú na svoje stránky, každý v tom svojom formáte a grafickej úprave. Podobne sa to deje aj s kurzami valút u jednotlivých bánk, programom televíznych staníc... Kvalitatívne novým riešením je použitie webovej služby, kedy poskytovateľ poskytne svoje údaje prostredníctvom webovej služby vo formáte XML. Technológia XML je totiž otvorená a platformovo nezávislá. XML súbor obsahuje okrem samotných údajov aj ich štruktúru.. Zjednodušene by sa dalo povedať, že tento formát pre výmenu údajov je rovnako dobre čitateľný pre počítače aj pre ľudí, aj keď ideálny stav je taký, kedy si stroje medzi sebou vymenia údaje a používateľ o tom ani nevie. Údaje môžeme ľahko načítať a identifikovať pomocou pomerne jednoduchej rutiny v ľubovoľnom programovacom jazyku, alebo po definovaní formátu zobrazenia priamo publikovať na webovej stránke. XML dokument sa skladá z elementov. Každý element obsahuje počiatočnú a koncovú značku (tag) Obidva tagy obsahujú názov elementu, koncový tag obsahuje navyše pred názvom elementu lomítko. Typický príklad XML elementu môže byť napríklad:

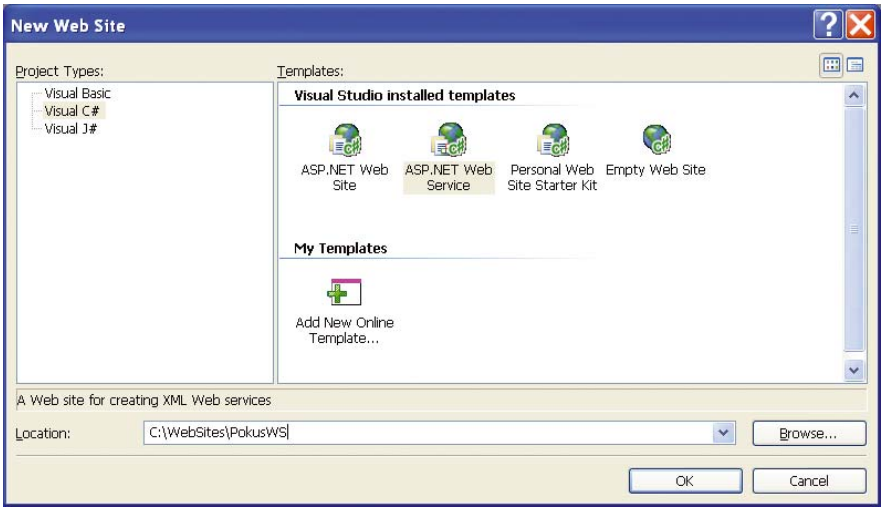
<teplota>**36**</teplota>

Každý element môže v sebe obsahovať ďalšie takzvané vnorené elementy.

```
<pocasie>
    <vietor>severozapadny</vietor>
    <teplota>36</teplota>
</pocasie >
```

Toľko stačilo ako teoretický úvod. Na záver úvodného odstavca venovaného webovým službám však spomenieme aj aktuálny stav. Fungujúce reálne webové služby je možné v dobe písania tejto publikácie spočítať na prstoch. Niekoho by to mohlo odradiť, no naopak, každá webová služba bude vlastne pionierskym činom a s rozvojom SOA (Serviced Oriented Architecture) sa očakáva dynamický nárast webových služieb. Technológie XML a SOAP sú naproti tomu masovo používané v integračných projektoch

Pokúsme sa vytvoriť prvú webovú službu vo vývojovom prostredí Visual Web Developer.



Vytvorenie projektu typu ASP.NET Web Services

Aplikácia obsahuje zdrojovku ukážky jednoduchej webovej služby, ktorej úlohou vypísať text „Hello World“.

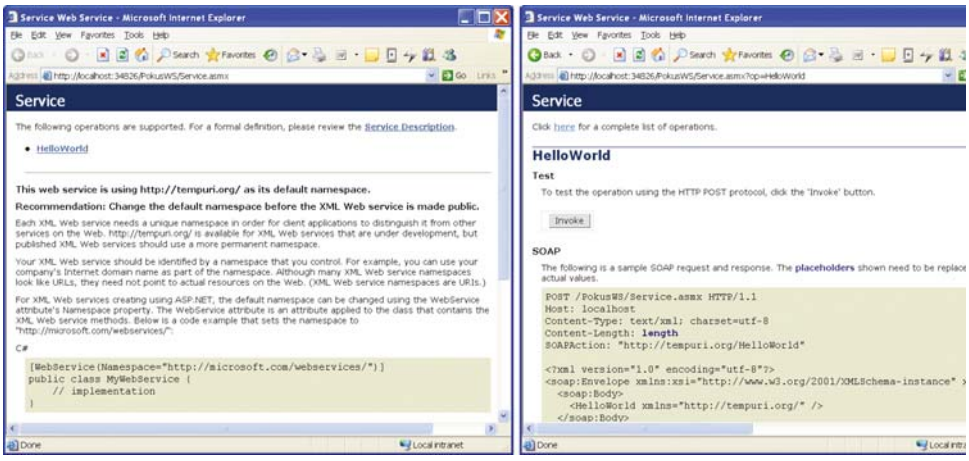
Príklad v programovacom jazyku Visual Basic .NET

```
Imports System.Web
Imports System.Web.Services
Imports System.Web.Services.Protocols

<WebServiceBinding(ConformanceClaims:=WsiClaims.BP10,
EmitConformanceClaims:=True)> _
Public Class Service
     Inherits System.Web.Services.WebService

    <WebMethod()> _
    Public Function HelloWorld() As String
        Return "Hello World"
    End Function

End Class
```

Príklad v programovacom jazyku C#

```
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;

[WebServiceBinding(ConformanceClaims=WsiClaims.BP10,EmitConformanceClaims = true)]
public class Service : System.Web.Services.WebService {

    [WebMethod]
    public string HelloWorld() {
        return "Hello World";
    }

}
```

Túto jednoduchú webovú službu, ktorú vygeneroval sprievodca vytvorením aplikácie môžeme ihneď otestovať



Testovanie webovej služby

Po zatlačení tlačidla Invoke nám webová služba vráti výsledok, samozrejme vo formáte XML

```
<?xml version="1.0" encoding="utf-8" ?>
  <string xmlns="http://tempuri.org/">Hello World</string>
```

Príklad webovej služby – pôžičkový kalkulátor

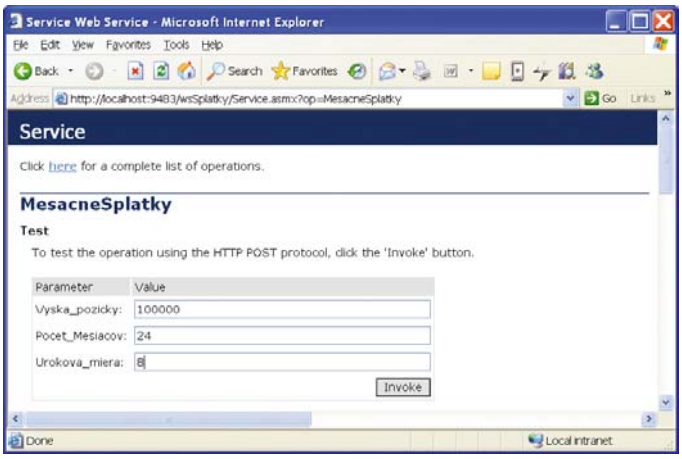
V tomto príklade vytvoríme jednoduchú webovú službu **wsSplatky**, ktorá spočíta výšku mesačnej splátky pri zadaní výšky pôžičky, úrokovej miery a počtu mesiacov na ktoré je pôžička poskytnutá. Založíme nový projekt typu WebServices, nazveme ho wsSplatky.asmx a umiestnime ho do adresára napríklad *C:\Inetpub\wwwroot\ASP_zbornik\k7*. Pre jednoduchosť použijeme ako programovací jazyk Visual Basic .NET nakoľko má implementovanú funkciu Pmt pre výpočet úroku.

Vývojové prostredie nám vygeneruje kostru aplikácie, kde doplníme kód tela webovej službu (hrubým fontom). Aby sme mohli využívať funkcie Pmt pre výpočet úroku a FormatCurrency pre formátovanie „obeživa“ musíme importovať namespace **`Microsoft.VisualBasic.Strings`** a **`Microsoft.VisualBasic.Strings`**.

Kompletný kód webovej služby bude:

```
Imports System.Web
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports Microsoft.VisualBasic.Strings
Imports Microsoft.VisualBasic.Financial


<WebServiceBinding(ConformanceClaims:=WsiClaims.BP10,
EmitConformanceClaims:=True)> _
Public Class Service
     Inherits System.Web.Services.WebService


    <WebMethod()> Public Function MesacneSplatky(ByVal Vyska_pozicky As Double,
ByVal Pocet_Mesiacov As Integer, ByVal Urokova_miera As Double) As String
        Dim dblSplatka As Double
        dblSplatka = -Pmt((Urokova_miera / 1200), Pocet_Mesiacov, Vyska_pozicky)
        Return FormatCurrency(dblSplatka)
    End Function


End Class
```

Webovú službu môžeme znovu jednoducho otestovať



Testovanie webovej služby Pôžičkový kalkulátor

Po zadaní parametrov pôžičky a zatlačení tlačidla Invoke nám webová služba vráti výsledok vo formáte XML

```
<?xml version="1.0" encoding="utf-8" ?>
  <string xmlns="http://tempuri.org/">4 522,73 Sk</string>
```

Vydal: Microsoft, s. r. o., Vyskočilova 1461/2a, 140 00 Praha 4, tel.: +420 - 261 197 111, fax: +420 - 261 197 100
http://www.microsoft.com/cze, http://www.microsoft.com/slovakia