Webový vývoj v ASP.NET 2.0 pomocí bezplatných Express nástrojů pro úplné začátečníky

Microsoft®
Vaše možnosti. Naše inspirace.

WEBOVÝ VÝVOJ V ASP.NET 2.0 POMOCÍ BEZPLATNÝCH EXPRESS NÁSTROJŮ PRO ÚPLNÉ ZAČÁTEČNÍKY

Praktické seznámení s nejrychleji rostoucí webovou technologií. ASP.NET 2.0 umožňuje vytvářet aplikace s bohatou funkčností i úplným začátečníkům, neboť psaní kódu je velmi silně omezeno. V deseti přehledných lekcích se seznámíte se základními dovednostmi a vytvoříte svoji první mini-aplikaci.

Připravil Microsoft, s.r.o. ve spolupráci s redakcí Živě.cz[1]

1 http://www.zive.cz/

OBSAH

1. ÚVOD, INSTALACE

Vítejte u desetidílného kurzu, který je určen pro začátečnické seznámení s technologií ASP.NET 2.0. Tato technologie dnes nabízí nejvyšší komfort pro vývoj webových aplikací, díky které můžeme vyvíjet aplikace s překvapivě bohatou funkčností, aniž bychom napsali jediný řádek kódu. A pokud již nějaký kód píšete, vždy jde o tvůrčí činnost a ne o mechanické opakování stále stejných nudných postupů.

A co víc – vše, co budete ke své práci potřebovat, lze získat zdarma – databázi, webový server, komfortní vývojový nástroj i hosting na veřejném webu. Nikdy v minulosti nebylo možné vyvíjet webové aplikace tak snadno, rychle a levně.

Tento kurz bude mít deset lekcí:

1. Úvod, instalace
2. Zabezpečení počítače
3. Hello World aplikace v ASP.NET
4. Základní principy ASP.NET
5. Databáze SQL Express – základy
6. Práce s daty v ASP.NET I.
7. Práce s daty v ASP.NET II.
8. Master pages a themes
9. Autentizace, autorizace
10. Kde získat další informace

První dvě lekce jsou přípravné – provedou vás instalací a zabezpečením počítače, na kterém budeme webové aplikace vyvíjet. V dalších dvou lekcích se seznámíme se základy vývoje v ASP.NET – co jsou to ovládací prvky, jak se používají apod. Protože drtivá většina webů zobrazuje nějaká data – ceník, sportovní výsledky, seznam článků apod., budeme se v páté lekci věnovat databázi SQL Server 2005 Express Edition, kterou poté v šesté a sedmé lekci použijeme pro práci s daty, jejich zobrazování uživatelům, editaci apod. V osmé lekci si trochu pohrajeme se vzhledem našeho webu – dosáhneme jednotné grafiky pomocí témat a jednotného rozvržení pomocí tzv. *master pages*. Devátá lekce je věnována uživatelům webů a jejich rozlišení – jak autentizovat uživatele jménem a heslem, jak rozlišit, co uživatelé smí a nesmí vidět, jak ukládat preference uživatele do jeho profilu apod. Poslední lekce nabídne zájemcům paletu možností k dalšímu postupu a rozvoji.

Potřebné HW vybavení

Pro svoji práci budete potřebovat počítač s operačním systémem Windows 2000 nebo vyšším. Hardwarové a softwarové požadavky jsou upřesněny v následující tabulce:

Processor	Minimum: 600 MHz Pentium procesor Doporučeno: 1 GHz Pentium procesor
RAM	Minimum: 128 MB Doporučeno: 256 MB
Hard Disk	Až 1.3 GB (typicky méně)
Operační systém	Microsoft Windows 2003 Server nebo Windows XP, Service Pack 2 nebo Windows 2000, Service Pack 4

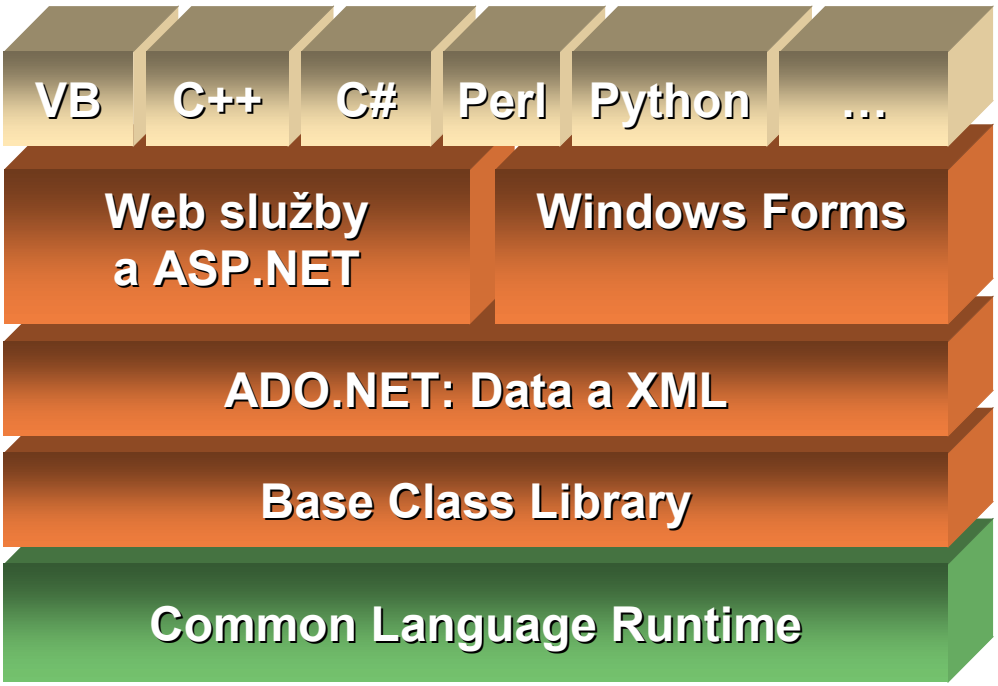
Jak je zřejmé, jedná se o požadavky, které splňuje velká většina počítačů zakoupených v uplynulých několika letech.

.NET framework

Pro činnost webových stránek v ASP.NET 2.0 je třeba komponenta zvaná *Microsoft .NET Framework 2.0* (můžete se setkat i se staršími verzemi 1.0 a 1.1, ty nejsou pro účely našeho kurzu dostatečné). K čemu vlastně .NET Framework slouží? Stará se o věci, které dříve museli vývojáři často řešit a dnes je považují za tak samozřejmé, že si je ani neuvědomují Teplo v bytě zpravidla také neřešíte, zkrátka otočíte kohoutkem a máte ho. Dokud bylo nutné nosit ze sklepa uhlí, každý o teple určitě více přemýšlel. .NET Framework se podobně „otočením kohoutku" postará o řadu nízkoúrovňových a nezáživných povinností jakými jsou:

- správa paměti, vytváření a rušení objektů
- spouštění a zastavování vláken kódu
- bezpečnost kódu a kontrola oprávnění k prováděným operacím
- natahování potřebných knihoven a komponent do paměti apod.

O tyto téměř neviditelné, ale velmi důležité operace se stará část zvaná *Common Language Runtime* (CLR) – viz obrázek celého .NET frameworku:



Měli bychom si vysvětlit význam dalších komponent. *Base Class Library* (BCL) je knihovna obsahující nejčastější pomocné funkce – práci se soubory, třídění, diagnostiku, síťovou komunikaci apod. ADO.NET je knihovna pro práci s daty s možností jejich XML reprezentace. V našem kurzu budeme velmi často jednotlivé třídy z ADO.NET a BCL nepřímo používat, i když budou skryty spíše pod povrchem než aby byly přímo viditelné. Dále jsou na obrázku dvě knihovny pro vývoj uživatelského rozhraní – Windows Forms pro desktopové aplikace a ASP.NET pro webové uživatelské rozhraní. Nás bude zajímat pouze druhá uvedená možnost, které budeme věnovat osm z deseti lekcí kurzu.

Velmi důležitou částí .NET frameworku jsou podporované jazyky. .NET framework je jazykově nezávislý, pro libovolnou úlohu lze v principu použít jakýkoliv z podporovaných jazyků. Ne všechny jazyky jsou ale vyvíjeny Microsoftem, řada jich je vyvíjena partnerskými firmami. Z jazyků vyvíjených Microsoftem nemají všechny stejnou vizuální podporu pro vývoj toho kterého typu aplikace. Po těchto vylučovacích kritériích nám zůstanou dva použitelné jazyky – C# a Visual Basic.NET. Z těchto dvou jazyků si může vybrat podle svých osobních preferencí, výsledek bude stejný pokud jde o funkčnost i výkonnost. Visual Basic.NET připomíná dobře známé skriptovací jazyky, je pro začátečníky lépe čitelnější, ovšem na úkor větší „ukecanosti". C# je elegantní, moderní, sevřený, ale pro začátečníka pravděpodobně obtížněji čitelný. Pokud v tomto kurzu budou ukázány fragmenty kódu, bude to vždy v obou těchto jazycích.

.NET Framework 2.0 lze samostatně stáhnout zde[2], Software Development Kit s některými nástroji a příklady zde[3].

SQL Server 2005 Express Edition

V typických aplikacích zpravidla potřebujete zobrazit data z relační databáze, ať už je to katalog zboží, seznam oblíbených písniček anebo sportovní výsledky. Webové stránky ASP.NET mohou zobrazit data z libovolné databáze, ke které je k dispozici ODBC nebo OLEDB ovladač – což by se v podstatě dalo zkrátit na „z libovolné databáze".

V našich příkladech budeme používat relační databázi *SQL Server 2005 Express Edition*. Lze ji stáhnout na adrese http://msdn.microsoft.com/vstudio/express/sql/ . Ke správě databáze je vhodné mít ještě *Microsoft SQL Server Management Studio Express*, v době psaní tohoto textu bylo ještě ve vývoji. Edice nazvaná *November CTP* byla k dispozici ke stažení zde[4].

Přestože je *SQL Server 2005 Express Edition* zadarmo, nabízí plnohodnotnou paletu funkcí. Jenom namátkou – referenční integrita mezi tabulkami, uložené procedury, triggery, uživatelsky definované funkce, přirozená podpora XML typu, možnost rozšiřování funkcí v .NET jazycích díky integraci .NET frameworku do databázového stroje atd.

Jak asi očekáváte, tato edice má i některá omezení. Dokáže využít pouze 1 GB paměti RAM, což je pro naše potřeby poměrně astronomická hodnota. Maximální velikost datového souboru je 4 GB, což pro většinu menších webových aplikací opět s rezervou stačí. Počet současných databázových připojení není (na rozdíl od předchůdce MSDE 2000) nijak omezen. Pokud bychom v budoucnu potřebovali přejít na některou z vyšších verzí[5] SQL Serveru 2005, např. protože nám výše uvedená omezení nestačí nebo potřebujeme funkce Business Intelligence, můžeme tak učinit velmi snadno. Všechny edice databáze totiž mají stejný formát souboru, které tak můžeme během okamžiku odpojit od edice Express a připojit k vyšší edici.

Webový server

K provozování webových aplikací v ASP.NET potřebujete samozřejmě nějaký webový server. Tento problém nemusíte řešit při vývoji, neboť vývojové prostředí, které budeme používat obsahuje vestavěný webový server tzv. *ASP.NET Development Server*. Pro vývoj je naprosto dostatečný, ale má určitá omezení, která znemožňují jeho použití pro provoz hotového webu. Podporuje totiž pouze lokální připojení (tj. prohlížeč a webový server musí být na stejném počítači), není automaticky spouštěn při startu apod.

Pro provozování hotového webu jej musíte umístit na počítač s *Internet Information Serverem* (IIS). Podporované verze jsou IIS 5.0 (Windows 2000), 5.1 (Windows XP) nebo 6.0 (Windows Server 2003). IIS nabízí vše, co pro provozování webu potřebujete, a to ve scénáři pro intranet i pro Internet. Publikování webu na IIS server je velmi snadné.

---

2 http://www.microsoft.com/downloads/details.aspx?FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5&DisplayLang=en
3 http://www.microsoft.com/downloads/details.aspx?FamilyID=fe6f2099-b7b4-4f47-a244-c96d69c35dec&DisplayLang=en
4 http://www.microsoft.com/downloads/details.aspx?FamilyId=82AFBD59-57A4-455E-A2D6-1D4C98D40F6E&displaylang=en
5 http://www.microsoft.com/sql/2005/productinfo/sql2005features.mspx

Pokud chcete provozovat web v internetovém scénáři a nemáte vlastní server, který by byl trvale připojený k Internetu a vhodný pro hostování vašeho webu, můžete využít služby hosterů. Řada z nich ASP.NET 2.0 a SQL Server 2005 Express podporuje za měsíční přijatelný peníz. Mnozí z nich dávají k dispozici omezené hostování aplikace zdarma. V době psaní tohoto textu to v ČR a SR byli, http://www.asp2.cz, http://www.qsh.sk/ a http://www.aspx.sk. Přenesení webu na server v hostingovém centru je možné pomocí FTP protokolu anebo (méně často) využitím FrontPage extenzí a protokolu HTTP. Tuto funkci si vyzkoušíme ve třetí lekci.

Visual Web Developer 2005 Express Edition

K vlastnímu vývoji budeme používat prostředí *Visual Studia 2005*, konkrétně edici *Visual Web Developer 2005 Express Edition* (dále jen VWD). Můžete si ji opět zdarma stáhnout z adresy http://msdn.microsoft.com/vstudio/express/vwd/. VWD je též distribuován jako příloha řady IT časopisů, takže je možné, že ji máte k dispozici bez nutnosti stahování. Pokud budete volit instalaci z webu, stáhnete pouze soubor *vwdsetup.exe* s velikostí necelé 3 MB, během instalace se pak stáhnou z webu další části.

Přestože je VWD zdarma, nabízí jednotlivci prakticky kompletní menu pro vytváření webových aplikací a stránek. Kromě vizuálního vytváření a editace stránek máte k dispozici i editor HTML dokumentů s podporou IntelliSense® nápovědy a podtrháváním chybných nebo nekompatibilních konstrukcí (volitelně se kontroluje i XHTML kompatibilita). Tam, kde nestačí vizuální průvodci a editory, je k dispozici editor kódu v jazycích C# nebo Visual Basic.NET, opět s podporou IntelliSense® nápovědy a podtrháváním nezkompilovatelných konstrukcí. Důležitou složkou je plná podpora ladění hotových stránek a integrovaná nápověda *MSDN Express*.

Profesionálové zřejmě časem budou chtít použít spíše vyšší edice *Visual Studio 2005 Standard*, *Professional* nebo *Team System*. Tyto edice jsou již zpoplatněny a přinášejí možnost vývoje dalších typů aplikací, vytváření instalačních balíčků, práci se vzdálenými databázemi, sdílení zdrojového kódu v týmu, modelování architektury, testování, komunikaci v týmu apod. Více informací lze nalézt např. na českém MSDN webu[6].

VWD bude po třiceti dnech provozu vyžadovat aktivaci. Aktivační kód získáte bezplatně výměnou za registraci a vyplnění profilu na webu MSDN. Více informací získáte ve VWD použitím nabídky *Help/Register Product...*

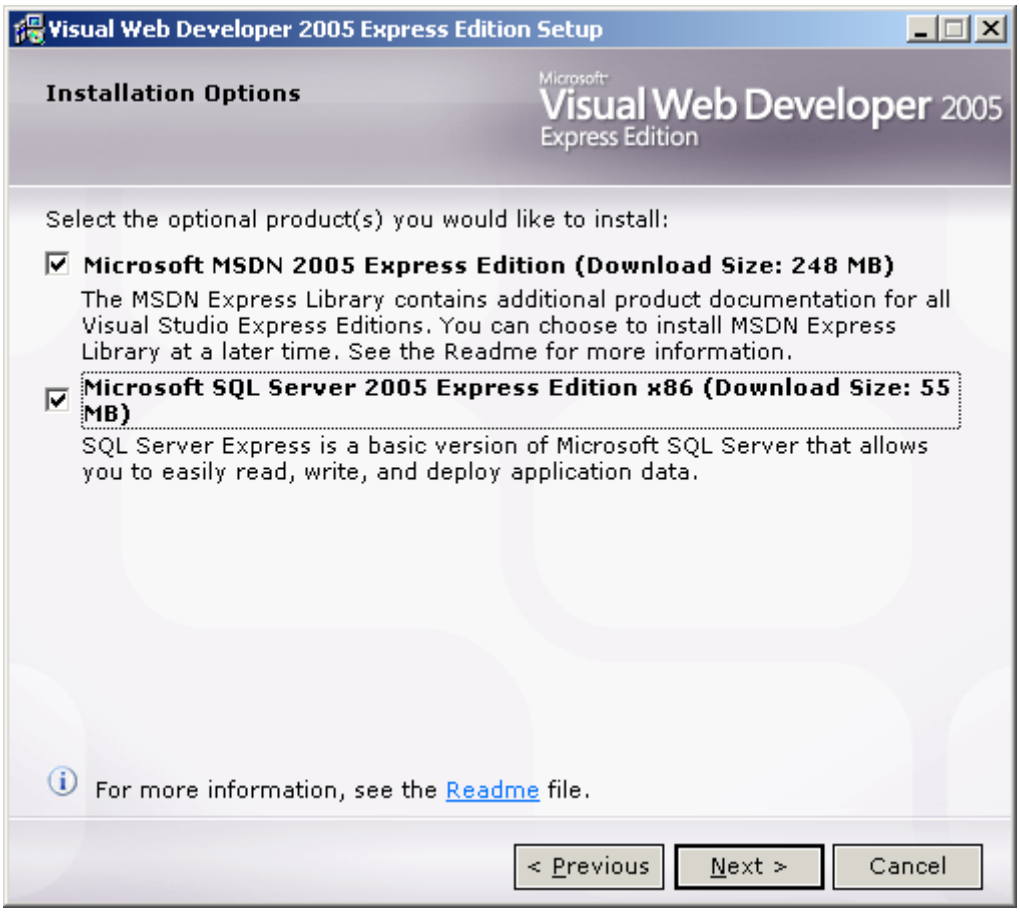
Průběh vlastní instalace

Vlastní instalace je velmi jednoduchá. Před instalací je nutné odstranit všechny předešlé nehotové verze produktu (zejména Beta 2 verze, popis je uveden zde[7]). Pokud instalujete stažené soubory např. z CD, instalujte komponenty v následujícím pořadí:

- .NET Framework 2.0
- SQL Server 2005 Express Edition
- Visual Web Developer 2005 Express Edition

Instalujete-li přímo z webu, máte situaci o mnoho jednodušší, neboť instalační program *vwdsetup.exe* je schopen stáhnout si veškeré potřebné komponenty přímo z webu. Vlastní instalace je pak velmi jednoduchá, spočívá v podstatě pouze v opakovaném mačkání tlačítka *Next*. Jediná obrazovka, která vyžaduje přemýšlení, je výběr komponent pro instalaci:

6 http://msdn.microsoft.cz/vstudio/
7 http://msdn.microsoft.com/vstudio/express/support/uninstall/



S nejvyšší pravděpodobností nebude třeba instalovat jazyk *Visual J#*, který je dnes silně menšinovým směrem. Instalaci *MSDN Express Library 2005*, což je nápověda a dokumentace lze určitě doporučit každému, kdo má dostatečné místo na disku. Databázový stroj *SQL Server 2005 Express Edition* budeme v našich příkladech potřebovat, takže jej určitě nainstalujte. Součástí celého procesu je též instalace *Microsoft .NET Frameworku 2.0*, který je vyžadovanou komponentou nezbytnou pro běh pro VWD.

Narazím-li na problém...

Je možné, že při instalaci narazíte na nějaký problém. Pak máte možnost využívat zdroje podpory pro vývojáře podle přehledu[8] na českém MSDN webu. Pokud nemáte MSDN předplatné, vhodnou formou pomoci pro vás bude pravděpodobně diskusní skupina *microsoft.public.cs.developer*, přístupná následujícími způsoby:

- Přes newsreader (NNTP) – news://news.microsoft.com/microsoft.public.cs.developer
- Ve webovém prohlížeči – http://support.microsoft.com/newsgroups/newsReader.aspx?lang=cs&cr=CZ&dg=microsoft.public.cs.developer&sloc=cs

Přestože odpověď na váš dotaz není garantována, je diskusní skupina pravidelně monitorována zaměstnanci Microsoftu ČR/Slovensko a velká většina dotazů je do tří pracovních dnů zodpovězena.

Závěrem

V tuto chvíli bychom měli mít plně připravené prostředí pro vývoj webových aplikací v ASP.NET 2.0. Pokud se již těšíte, že se v druhé lekci vrhneme na vývoj první aplikace, tak budete zřejmě malinko zklamáni – budete muset ještě počkat. Důvod je více než důležitý – bezpečnost. V příští lekci se dozvíte, jak zabezpečit počítač, na kterém vyvíjíte či provozujete webovou aplikaci, aby se nestal snadnou kořistí v internetové džungli plné lidí s nekalými úmysly.

8 http://msdn.microsoft.cz/support/

2. ZABEZPEČENÍ POČÍTAČE

Ve druhé lekci se budeme věnovat zabezpečení počítače. Pokud vám její obsah přijde triviální, v klidu ji přeskočte a připojte se k nám zase v třetím třetí lekci. V době všudypřítomného internetu je každý připojený počítač vystaven neustále hrozbě a podcenění zabezpečení může vést ke zkáze dat, ztrátě citlivých údajů apod. doslova během několika málo minut.

V ideálním světě, kde by vše fungovalo tak, jak má, by žádné bezpečnostní problémy vlastně ani neexistovaly. Bohužel náš svět se liší od ideálního v několika „drobných" detailech:

- Uživatelé se chovají nezodpovědně, v rozporu s doporučeními, která buď neznají anebo ignorují.
- Správci systému nedodržují doporučené postupy pro instalaci a konfiguraci aplikací a operačního systému.
- Programy mají chyby, které lze zneužít k útokům na bezpečnost počítače.

Vzhledem ke složitosti celé problematiky se Microsoft snaží co nejvíce zjednodušit, zlidštit a zpřístupnit zabezpečení počítače koncovým uživatelům. Tato iniciativa se nazývá 3 kroky k zabezpečení počítače[9] (používání firewallu, aktualizace počítače, používání antivirového programu). Jakkoliv se takovýto přístup může zdát počítačovému profesionálovi příliš zjednodušený, je faktem, že přinesl v poslední době razantní zvýšení základní úrovně bezpečnosti počítačů, z nichž mnoho je vlastněno laiky a nemá profesionálního správce.

Pokud jde o statistiky a přehledy bezpečnostních chyb v jednotlivých operačních systémech a softwarových balících, doporučuji jednoznačně vaší pozornosti databázi chyb v softwaru zvanou NVD (National Vulnerability Database) provozované americkou vládní organizací NIST (The National Institute of Standards and Technology). Databázi najdete na adrese http://nvd.nist.gov/. Databázi lze dobře využít ke srovnání počtu chyb v jednotlivých produktech. Zvolíte-li možnost Advanced Search, můžete porovnávat počet chyb v jednotlivých softwarových balících. Technologie, které budeme používat přitom ve srovnání se svými konkurenty – navzdory zažitým předsudkům – dopadají velmi dobře. V následující tabulce můžete vidět počet chyb za posledních necelých 8 měsíců v době psaní tohoto textu (tj. od 1. 1. 2005 do 22. 8. 2005):

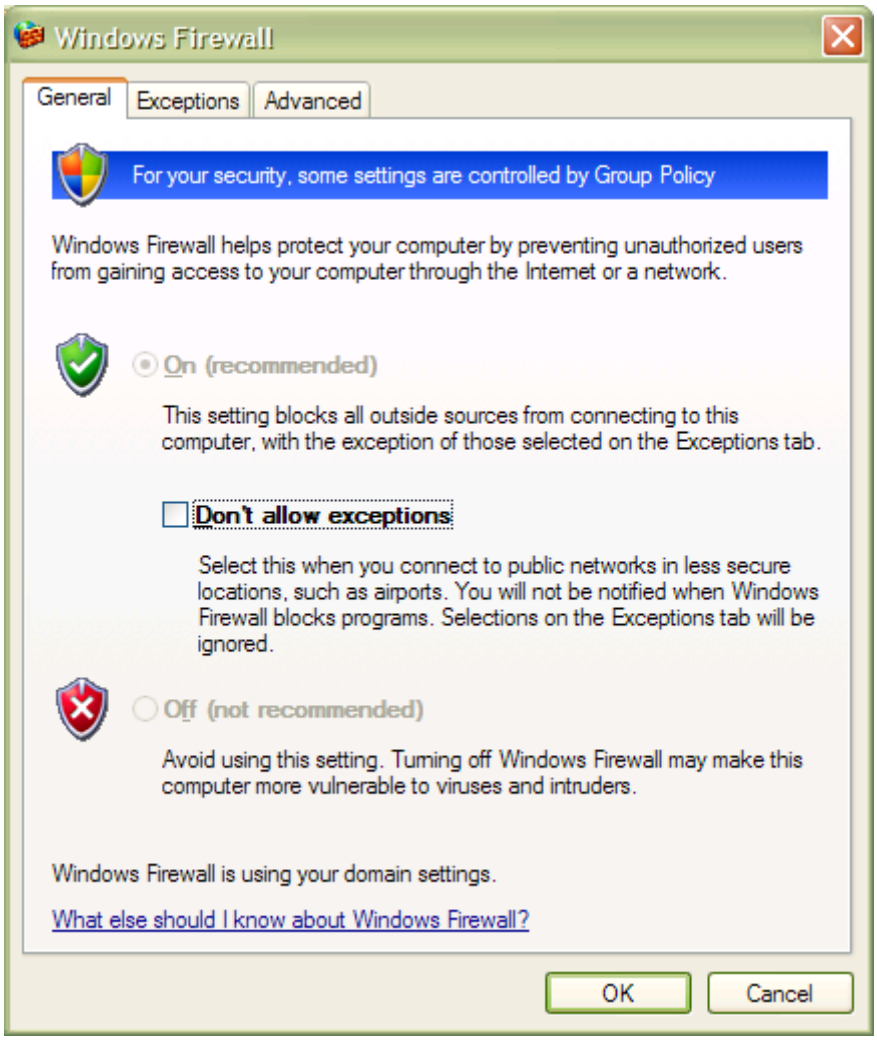
Microsoft produkt		**Konkurenční produkt**	
SQL Server	0	Oracle 10g Database Server (Oracle)	3
		MySQL (MySQL)	11
IIS	1	Apache (Apache Software Foundation)	5
Windows XP	43	Red Hat Desktop (Red Hat)	48
Windows Server 2003	37	Red Hat Enterprise Linux (Red Hat)	66
ASP.NET	4	PHP (PHP)	7

Určitě je velmi užitečné se s tímto webem seznámit a naučit se v něm vyhledávat. Lepší výsledky ale samozřejmě nejsou důvodem k jásotu. Čím větší podíl na trhu ten který produkt má, tím vyšší je možný dopad každé bezpečnostní chyby. Výše uvedená čísla jsou spíše závazkem do budoucna a uvědoměním si odpovědnosti za stav softwarové bezpečnosti. Jednoznačným cílem do budoucna není žádný závod s konkurencí, ale zvýšení bezpečnosti a důvěryhodnosti prostředí počítačových sítí. Vraťme se nyní zpátky na zem a podívejme se, jak může k bezpečnosti vlastního počítače přispět každý uživatel, ať už profesionál či začátečník.

9 http://www.microsoft.com/cze/athome/security/protect/default.mspx

Používání osobního firewallu

Firewall slouží jako první linie obrany. Filtruje síťovou komunikaci přicházející zvenčí (ze sítě, z internetu apod.) a zabraňuje veškerým pokusům o nepovolenou komunikaci. Zapnutí firewallu je jednoduché – v ovládacích panelech buď prostřednictvím položky *Windows Firewall* anebo ve vlastnostech síťové karty – viz obrázek:



Základní ovládání firewallu je jednoduché – zapnout a vypnout. Další operace, jako je nastavování výjimek pro aplikace a síťové karty nebo povolování některých příchozích komunikací jsou již za hranicí tohoto článku.

Samozřejmě nemusíte používat Windows Firewall, ale můžete místo něj použít osobní firewall jiného výrobce. Na trhu jich je celá řada, mnohé jsou dokonce k dispozici zdarma.

Je třeba si uvědomit, že firewall sám o sobě je zcela nedostatečnou ochranou, brání pouze před aktivním napadením počítače zvenčí. V žádném případě neochrání před jinými způsoby zavlečení nákazy – stažením a spuštěním infikované webové stránky nebo obsahu, prostřednictvím emailu, CD/DVD, diskety, USB klíče apod.

Antivirový software

Antivirový software se snaží likvidovat nežádoucí soubory, programy, skripty, dokumenty apod., které se na počítač již dostaly anebo se právě dostat snaží. Na trhu je velké množství antivirových programů, mnohé po určitou dobu zadarmo anebo za přijatelný peníz. Aby byla antivirová ochrana účinná, je nezbytná okamžitá aktualizace virových databází jakmile se objeví nová verze. Bez této funkce je

antivirový program téměř neúčinný. Výhodou antivirových programů je, že dokáží zčásti napravit chybné a lehkovážné chování uživatele, stoprocentní účinnost ale nelze čekat nikdy. Hlubší rozbor jednotlivých druhů nákaz a odpovídajících funkcí antivirových programů není účelem tohoto textu.

Správa oprav

Viry, červi, trojské koně a další zlovolný software lze rozdělit zhruba do dvou skupin. První využívají nevzdělanosti a naivity uživatele (lehkovážné chování, špatná konfigurace apod.). Druhá skupina využívá chyb a slabin v operačním systému nebo některé z nainstalovaných aplikací. Antivirové programy s rozumnou mírou úspěšnosti chrání proti oběma skupinám. Instalace bezpečnostních oprav je pak prakticky 100%ní ochranou proti druhé skupině.

Druhy softwarových oprav

Softwarové opravné balíčky se označují různými názvy. Dají se rozdělit do tří skupin: service packy, hotfixy a „něco mezi tím".

Service packy (SP) jsou známé největšímu počtu lidí. Nabízejí opravu většího množství chyb v jednom kroku.Ve většině případů se číslují vzestupnými čísly (SP1, SP2, SP3, ...) a označují se jménem produktu a tímto číslem (např. Windows Server 2003 Service Pack 4). Zpravidla obsahují všechny opravy uskutečněné od vydání produktu – říkáme, že jsou kumulativní. V praxi to znamená, že pokud například existuje SP4 na Windows Server 2000, postačí když nainstalujete operační systém z originálního média a poté Service Pack 4 coby poslední vydaný service pack – nemusíte tedy instalovat SP1, SP2 a SP3. Service packy na produkty Microsoftu jsou vydávány poměrně zřídka a s tím jak se zpřísňují kritéria a vylepšuje metodika a kvalita vývoje produktů, perioda jejich vydávání se dále zvyšuje. V tabulce vidíte dobu vydání příslušného SP od vydání produktu v měsících:

Produkt/SP	**SP1**	**SP2**	**SP3**	**SP4**	**SP5**	**SP6**
Windows NT 4	1	3	8	25	32	39
Windows 2000	5	15	30	40	-	-
Windows XP	12	36	-	-	-	-
Windows Server 2003	24	-	-	-	-	-

Service packy by v zásadě měli instalovat všichni uživatelé produktu, takže spíše než „zda" by bylo vhodné ptát se „kdy". Zejména ve velkých firmách je zpravidla nutné otestovat, zda instalace SP nenaruší funkci stávajících aplikací nebo produktů, což může někdy trvat měsíce. Drobní uživatelé by s instalací SP neměli příliš otálet – čekáním nic nezískají. SP jsou velmi důkladně testovány a rovněž srovnávají systémy s různě instalovanými opravami na jednotnou úroveň. Veškeré produkty a další vyvíjené opravy jsou typicky testovány oproti poslednímu a předposlednímu vydanému SP, takže instalací SP se rovněž snižuje riziko, že vaše aplikace budou mít funkční problémy.

Hotfixy (nazývané též patche nebo záplaty) naopak opravují většinou pouze jedinou objevenou chybu a jsou tedy vydávány výrazně častěji – pro jeden produkt může být klidně vydáno i několik hotfixů měsíčně. Plně opravený produkt tak může mít nainstalované i desítky hotfixů. Měli bychom totiž instalovat všechny hotfixy, které nebyly zahrnuty v posledním vydaném service packu, což nám znovu připomnělo důležitost instalace SP – je určitě výhodnější nainstalovat poslední SP a 5 hotfixů než instalovat 50 hotfixů. Hotfixy Microsoftu se označují písmenem Q a šestimístným číslem. Jejich popis je přístupný na webu Technetu[10], kde je umístěna tzv. Knowledge Base (KB). Ve vyhledávací stránce[11] můžete vyhledat hotfix podle čísla chyby (např. Q298936) anebo se pokuste popsat anglickými termíny chybu, její číslo apod. a použít full-textové hledání. Z hlediska doporučení se hotfixy dělí do dvou skupin: bezpečnostní a ostatní.

10 http://technet.microsoft.com/default.aspx
11 http://support.microsoft.com/default.aspx?scid=fh;EN-US;kbhowto&sd=GN&ln=EN-US&FR=0

Bezpečnostní hotfixy opravují chyby spojené s bezpečnostním rizikem, proto by je měl bezodkladně instalovat každý, kdo zasaženou komponentu používá. Bezpečnostní hotfixy jsou bez výjimky veřejné a jsou důkladně testovány. O vydání bezpečnostního hotfixu se nejsnáze dozvíte odebíráním bezpečnostních bulletinů (Security Bulletins), které si lze zdarma objednat na adrese http://www.microsoft.com/security/bulletins/default.mspx.

Ostatní hotfixy opravují problémy s funkcí, stabilitou, výkonností apod. Instalovat by je měl pouze ten, kdo v praxi zaznamená problém, jenž hotfix opravuje. Hotfixy opravující často se projevující chyby jsou zpravidla veřejně dostupné a důkladně otestované. Některé hotfixy opravují chyby vyskytující se poměrně vzácně. Tyto hotfixy neprocházejí tak důkladným testováním a jsou neveřejné – v popisu se pak můžete dočíst toto „contact Microsoft Product Support Services to obtain the hotfix" (viz např. Q298936[12]). Důvodem není žádné utajení, ale menší důkladnost testování, v jehož důsledku je z preventivních důvodů nutné udržovat seznam zákazníků, kteří si hotfix nainstalovali. Navzdory obecnému přesvědčení může neveřejný hotfix získat bez problémů kdokoliv. Stačí zavolat na Linku technické podpory (+420-841300300, více informací zde[13]) a sdělit Q-číslo problému, produkt nemusíte mít dokonce ani registrovaný. Adresu ke stažení hotfixu pak obdržíte e-mailem.

Poslední skupina oprav „něco mezi tím" nemá jednotné jméno – často se nazývá update pack, rollup pack, update rollup, rollup patch apod. Nevyskytují se příliš často, takže je dokonce možné, že jste se s ní nikdy nesetkali. Příkladem z poslední doby je Windows 2000 Update Rollup 1 for Service Pack 4[14]. Tyto balíčky představují více hotfixů zabalených do jednoho instalačního balíčku, účelem jejich vydávání je většinou snížení pracnosti instalace anebo dočasná (v případě Windows Serveru 2000 pravděpodobně trvalá) náhrada SP. Ke každému takovému balíčku je připojen popis, který doporučuje, kdo by jej měl instalovat.

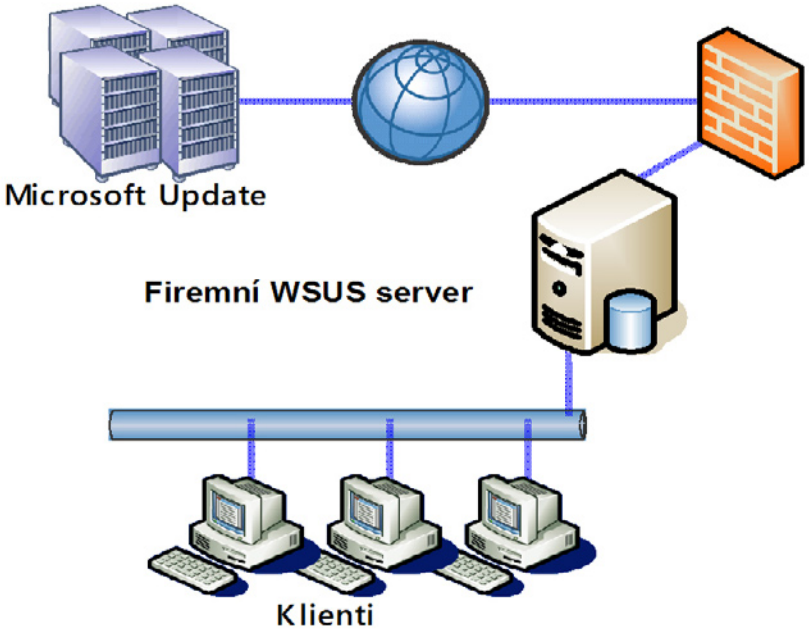
Kde získat opravy?

Optimální způsob získávání oprav závisí zejména na velikosti spravované sítě. Samozřejmě je možné sledovat bezpečnostní bulletiny, stahovat si jednotlivé opravy z webu a instalovat je ručně na jednotlivé počítače, ale tento postup zřejmě zvolí málokdo. Kromě vysoké pracnosti má i velké riziko chyby či opomenutí. Daleko lepší způsob je automatické získávání a instalace oprav.

Pro jednotlivé uživatele anebo malé firmy je určena služba Microsoft Update[15]. Počítač nastavený pro používání této bezplatné služby se jednou denně (při dostupnosti internetového spojení) napojí na servery Microsoftu, stáhne si aktuální seznam oprav a zkontroluje, zda jsou všechny opravy s vysokou prioritou nainstalovány na lokálním počítači. Zjistí-li, že některé opravy chybí, upozorní uživatele počítače, nebo je i sám stáhne či nainstaluje – podle nastavení. Doporučena je automatická instalace, která je nejméně pracná a nejbezpečnější. Tímto způsobem není aktualizován pouze operační systém (Windows 2000 nebo vyšší), ale i další produkty – Office 2003, Exchange Server 2003, SQL Server 2005 a časem budou přibývat další.

Pro menší a střední firmy je určena technologie Windows Server Update Services[16] (WSUS). WSUS jsou v mnohém podobné službě Microsoft Update – stahují totiž seznam oprav i vlastní opravy ze stejného webu Microsoft Update. Rozdíl je v kontrole nad celým procesem správy oprav. Zatímco Microsoft Update funguje „sám" bez nutnosti zásahů uživatele či správce, WSUS umožňuje plnou kontrolu nad procesem. Jednotlivé opravy jsou staženy na firemní server s běžícími WSUS – viz obrázek:

12 http://support.microsoft.com/default.aspx?scid=kb;en-us;298936
13 http://support.microsoft.com/gp/hotline
14 http://www.microsoft.com/windows2000/server/evaluation/news/bulletins/rollup.asp
15 ttp://update.microsoft.com/microsoftupdate/
16 http://www.microsoft.com/windowsserversystem/updateservices/default.mspx



Počítače ve firemní síti se připojují k tomuto serveru a získávají od něj seznam oprav i vlastní opravy. Administrátor WSUS má plnou kontrolu nad tím, které počítače v síti jsou, jaké opravy jsou na nich nainstalovány a které chybějí, může schvalovat instalaci jednotlivých oprav na všechny počítače nebo jenom na omezenou skupinu apod. WSUS kombinuje pohodlí a bezpečnost automatické správy s plnou kontrolou nad procesem instalace oprav.

Největší firmy zpravidla používají komplexní nástroje pro správu konfigurace svých stanic, které již obsahují nějakou funkčnost pro správu oprav. V tom případě je samozřejmě vhodné využívat již zakoupenou a rozběhnutou funkčnost. Např. zákazníci používající Microsoft Systems Management Serveru 2003 mohou využít funkčnost nástroje Systems Management Server 2003 Inventory Tool for Microsoft Updates[17].

Závěrem

Je možné, že se vám tato lekce zdála nudná, zbytečná, triviální. Možná jste se ale přece jenom dozvěděli něco nového. Ať už tak či onak, mějte na paměti, že zabezpečení jakéhokoliv počítače připojeného na internetu je absolutní prioritou a že jeho základem jsou tři jednoduché kroky, které zvládne opravdu každý uživatel. V příští lekci se již nebudeme věnovat přípravným činnostem – skočíme rovnýma nohama do vývoje vaší první aplikace.

17 http://www.microsoft.com/smserver/downloads/2003/tools/inventory.asp

3. HELLO WORLD APLIKACE V ASP.NET

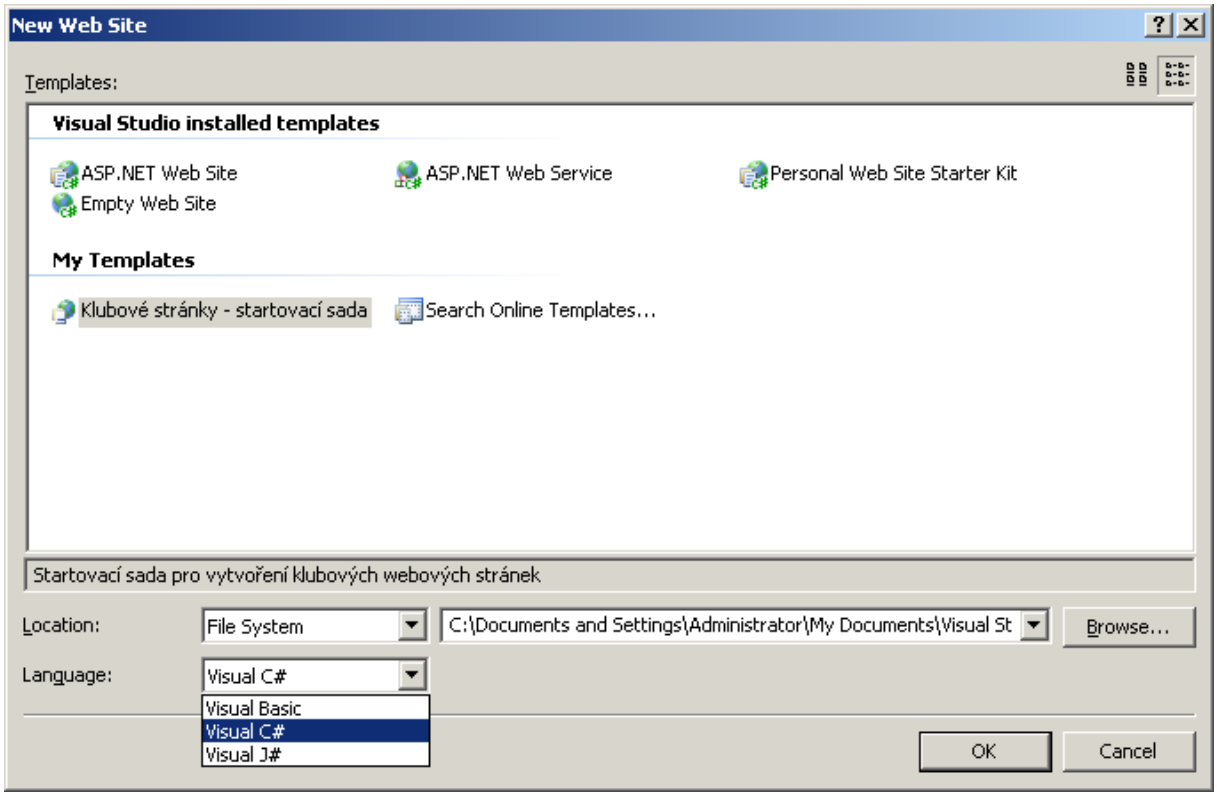
Ve třetí lekci začneme (konečně) vytvářet naši první pidi-aplikaci v ASP.NET 2.0. Bude věnována vysvětlení celého vývojového cyklu – založení projektu, seznámení s prostředím pro vývoj a ladění webových stránek, a konečně nasazení projektu na provozní server. Tak s chutí do toho. Začneme spuštěním *Visual Web Developer 2005 Express Edition* z nabídky *Start*.

Práce s projekty

Všichni asi cítíme, co je to projekt, přestože definice tohoto slova tak snadná není. Ve VWD je projektem web, který vytváříme. Po technické stránce je projekt složkou na disku obsahující celou řadu souborů – kód, stránky, obrázky, databáze apod. S projektem pracujeme budeme pracovat jako s nedělitelným celkem. V závislosti na fázi, ve které se nacházíme, můžeme zvolit buď založení nového projektu anebo otevření již existujícího projektu.

Založení projektu

V případě, že vytváříme či zakládáme nový projekt (*New Web Site*), musíme si vybrat z nabídky šablon (*Templates*):



Šablona není nic jiného, než zaZIPovaný soubor s počátečním obsahem webu. Nabídka dostupných šablon se může lišit podle zvoleného programovacího jazyka v levém dolním rohu dialogového okna. Šablony můžeme rozdělit do dvou skupin:

- „Prázdné" šablony – obsahují pouze minimální obsah, předvytvořené prázdné adresáře apod. Pro účely našich cvičení budeme používat šablonu *ASP.NET Web Site*. Šablona *Empty Web Site* neobsahuje vůbec nic, šablonu *ASP.NET Web Service* lze použít k vytváření webových služeb podle standardů SOAP a WSDL.

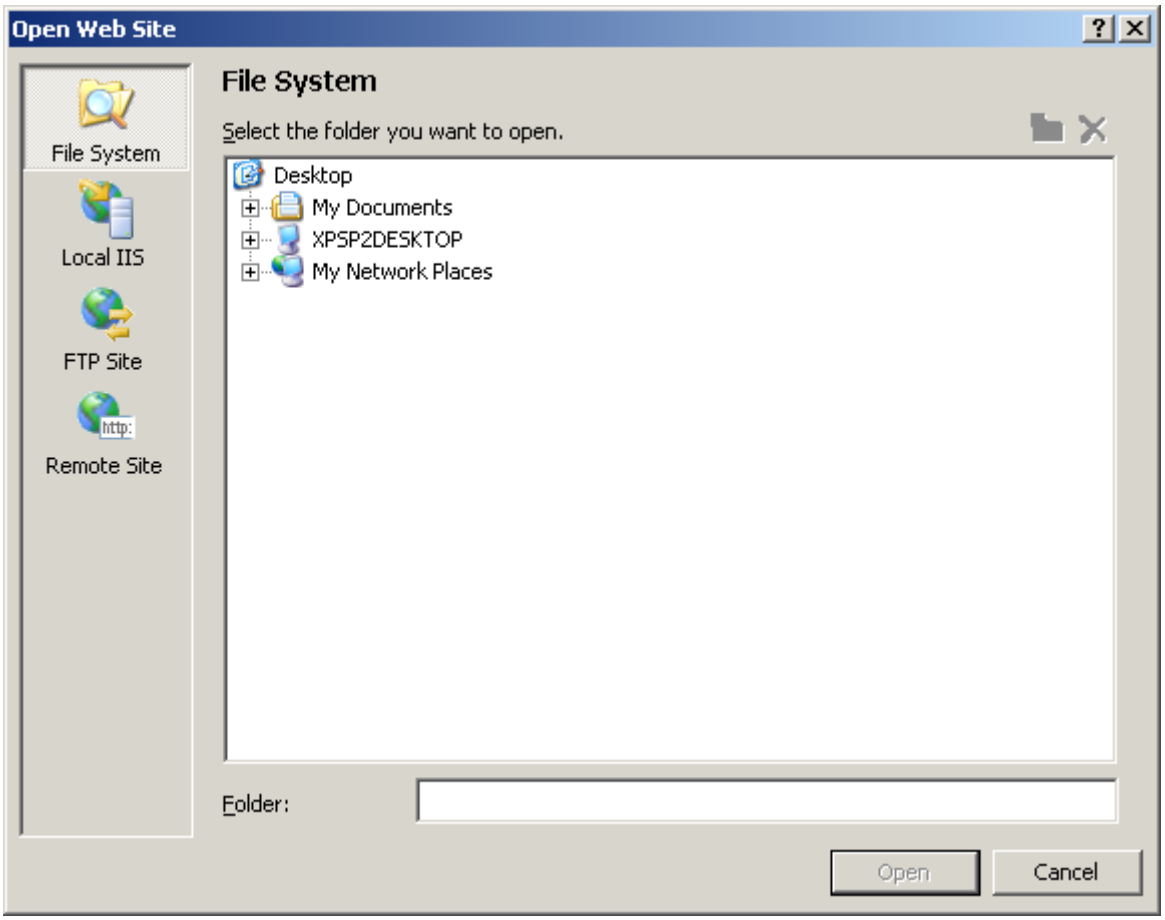
- Starter Kity – jsou v podstatě hotovými vzorovými weby, které můžeme upravovat a dodělávat podle vlastních potřeb. Představují tedy nejrychlejší cestu při vytváření specializovaných řešení. Součástí VWD je pouze jediný – *Personal Web Site Starter Kit*. Na webu www.asp.net jsou

k dispozici ještě další starter kity pro ASP.NET 2.0. V době psaní tohoto článku je možné získat *Club Site Starter Kit* pro vytváření stránek klubu, zájmového oddílu apod. a *Time Tracker Starter Kit* pro vedení výkazů práce odvedené jednotlivými lidmi na konkrétních projektech na této adrese[18]. Připravují se též další starter kity, některé z nich dokonce v češtině (informace o nich hledejte na http://msdn.microsoft.cz, jejich seznam lze nalézt též na http://starterkits.aspnet.cz/). Pozor, nabídka šablon Starter Kitů závisí na programovacím jazyce, který jste si vybrali.

Obě skupiny mají jedno společné – musíte zadat umístění pracovního adresáře projektu. Ten může být umístěn buď na lokálním disku vašeho počítače (doporučovaná možnost), na vzdáleném FTP serveru anebo HTTP serveru s instalovanými FrontPage extenzemi. Poslední dvě možnosti se doporučuje používat spíše pro nasazení hotového webu než pro vývoj.

Otevření existujícího projektu

Pokud otevíráme existující projekt (*Open Web Site*), máme situaci ještě jednodušší:



Jediné, co musíme zadat, je umístění složky webového projektu. Nejčastěji to bude složka v souborovém systém lokálního počítače, ale v některých případech může být web umístěn na lokálním webovém serveru IIS, vzdáleném FTP serveru anebo HTTP serveru s instalovanými FrontPage extenzemi – v tom případě samozřejmě musíme správně zadat adresu a další informace nezbytné pro připojení. Pro vývoj je nejjednodušší používat projekty v souborovém systému, neboť při otevírání vzdálených webů pomocí ostatních technologií je komfort práce nižší (typicky např. nebude fungovat vzdálené ladění).

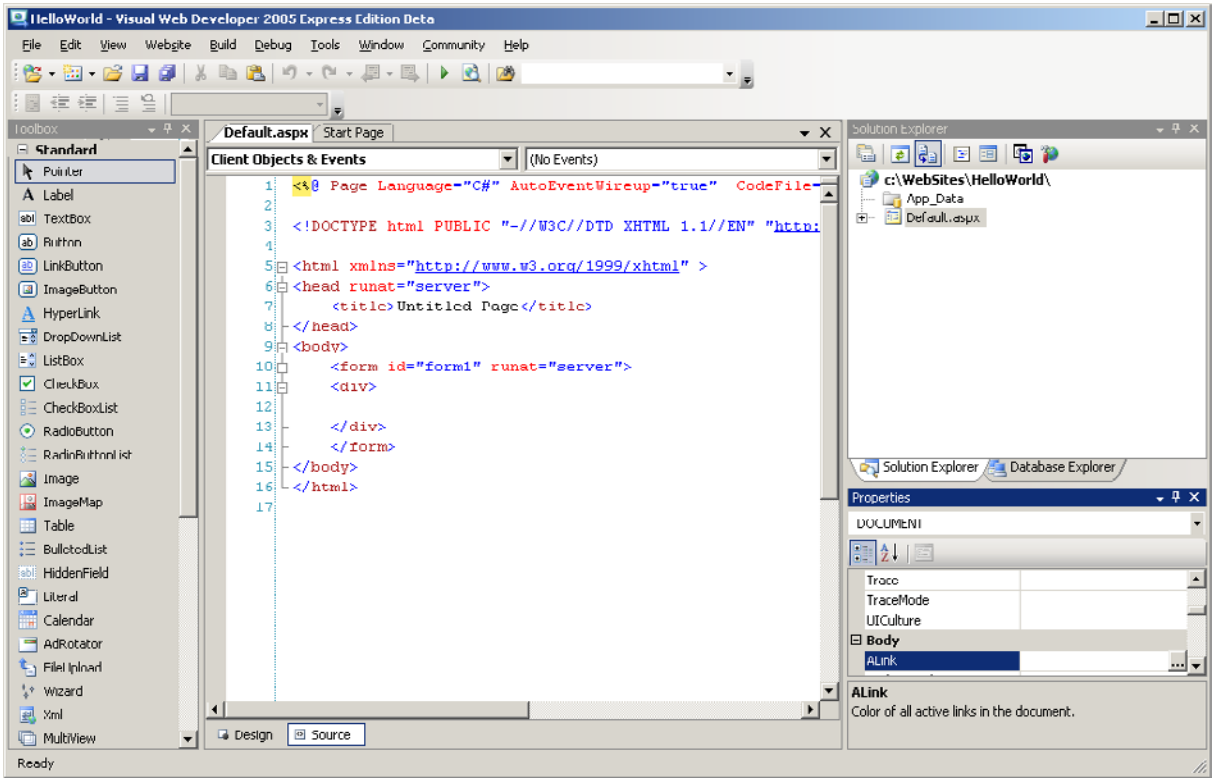
18 http://www.asp.net/vwd/starterkits.aspx?tabIndex=4&tabId=46

Vývoj Hello World

Z výše vyjmenovaných možností si zvolte si vytvoření nového webu (*New Web Site*), jako typ vyberte *ASP.NET Web Site* a jako cestu zadejte *c:\WebSites\HelloWorld*, čímž se vytvoří základní struktura prázdného webu s připravenou jedinou stránkou.

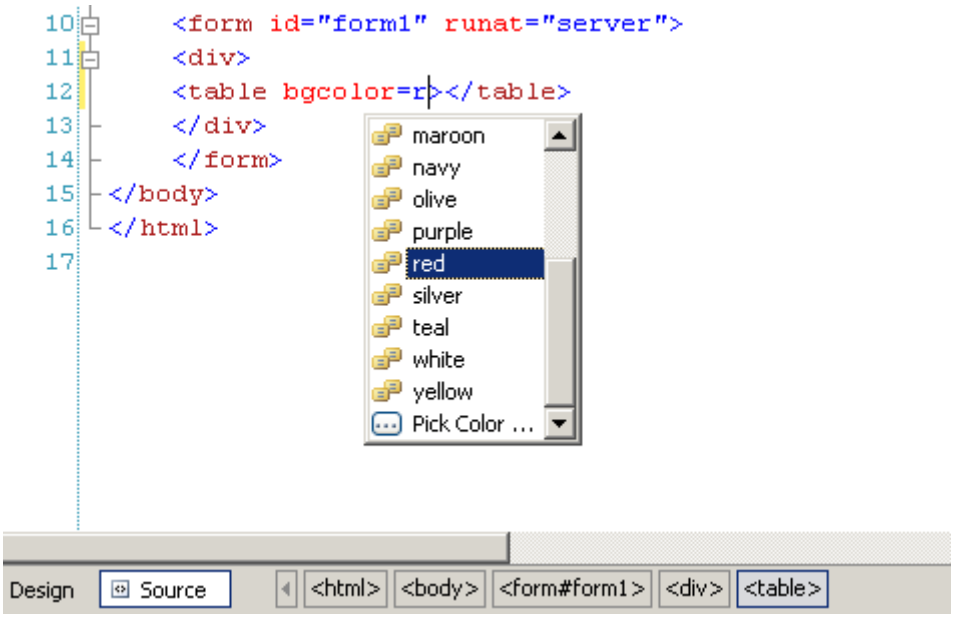
Okna ve VWD

Ve VWD bychom nyní měli vidět zhruba následující rozložení oken a prvků (pokud některé z oken nevidíte, můžete jej zobrazit pomocí nabídky *View*):



Nejdůležitější okno je to, které nemá jméno a je uprostřed obrazovky. V něm probíhá psaní kódu i vytváření designu webové stránky. Úplně nahoře vidíte záložky, z nichž každá znamená jeden otevřený soubor. Pro každou webovou stránku můžete otevřít dva soubory – jeden s kódem a jeden s definicí vzhledu.

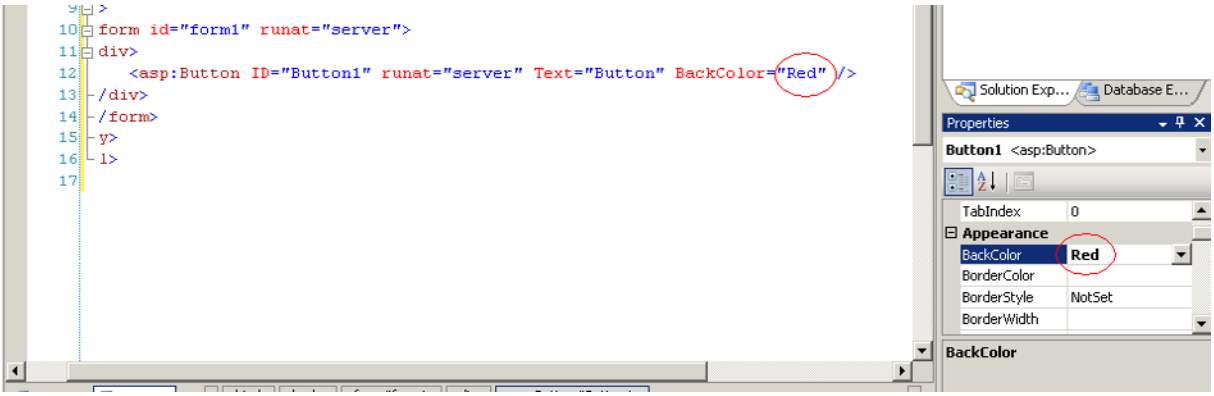
Pokud pracujeme s definicí vzhledu, máme k dispozici dva náhledy na stránku. Přepíná se mezi nimi výběrem režimu *Design* a *Source* v levé dolní části okna. Pokud pracujeme v režimu *Design*, vidíme vzhled stránky vizuálně. Můžeme upravovat rozmístění jednotlivých prvků, přetahovat je myší apod. Mírně pokročilejší vývojáři mohou v okně *Source* psát přímo HTML značky tvořící stránku. Pokud v tomto okně píšete, máte k dispozici tzv. *IntelliSense* neboli nápovědu, která předvídá vaše záměry, čímž šetří úhozy do klávesnice. Zkuste například kliknout na prázdném místě v řádku 12 a psát pomalu znak po znaku text pro vložení tabulky *<table bgcolor=yellow>* a uvidíte možnosti nápovědy (pokud chcete nápovědu použít, stiskněte na ní mezeru nebo jiný podobný znak) – viz obrázek:



Povšimněte si dále značek plus/minus těsně za číslem řádku. Pomocí těchto značek můžete rozbalovat a sbalovat celé pasáže textu, například celou definici tabulky. Dole na liště pak vidíte hierarchii HTML značek v dokumentu (<html><body><table> atd.). Kliknutím na těchto značkách můžete ve stránce vybrat celou značku včetně jejího vnitřního obsahu. Pokud píšete HTML značky, editor automaticky kontroluje a podtrhává odchylky od správné syntaxe podobně jako např. překlepy v MS Word. Přísnost a pravidla pro tuto kontrolu si na správnou úroveň můžete nastavit v nabídce *Tools/Options/Validation* – můžete ji například nastavit na striktní dodržení standardu XHTML 1.1.

Na pravé straně vidíte okno *Toolbox*, ve kterém jsou seskupeny jednotlivé ovládací prvky – tlačítka, kalendáře, přepínače, tabulky, obrázky apod. Jednotlivé skupiny jsou vyznačeny tučně a mají před sebou znak plus/minus sloužící k rozbalení/sbalení jejich obsahu. Libovolný z prvků (např. tlačítko, *Button*) můžeme přetažením myši přenést do okna editujícího HTML. Pokud to provedeme v režimu *Design*, uvidíme na cílovém místě námi přenesené tlačítko. Provedeme-li totéž v režimu *Source*, vidíme na cílovém místě vygenerované HTML značky odpovídající tlačítku. Můžete si to sami snadno vyzkoušet.

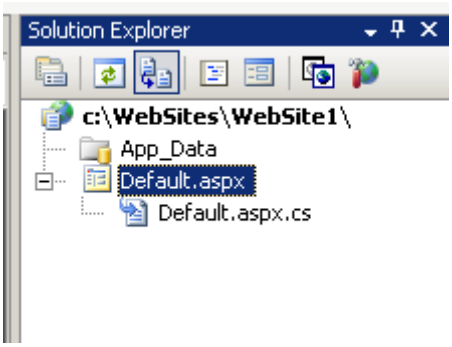
V pravém dolním rohu je důležité okno *Properties*. V něm se zobrazují nastavené vlastnosti aktuálně vybraného objektu. Pokud např. přepnete editaci HTML do režimu *Design*, někam na stránku vložíte tlačítko a vyberete jej myší, uvidíte v tomto okně celou řadu vlastností tlačítka – šířku, výšku, barvu pozadí, barvu okraje apod. (*Width, Height, BackColor, BorderColor*). Pokud výběrem ze seznamu možných hodnot nastavíte barvu pozadí (*BackColor*) např. na červenou (*Red*), změní se samozřejmě barva tlačítka na pracovní ploše. Přepnete-li nyní editační okno do režimu *Source*, uvidíte zhruba následující řádek:



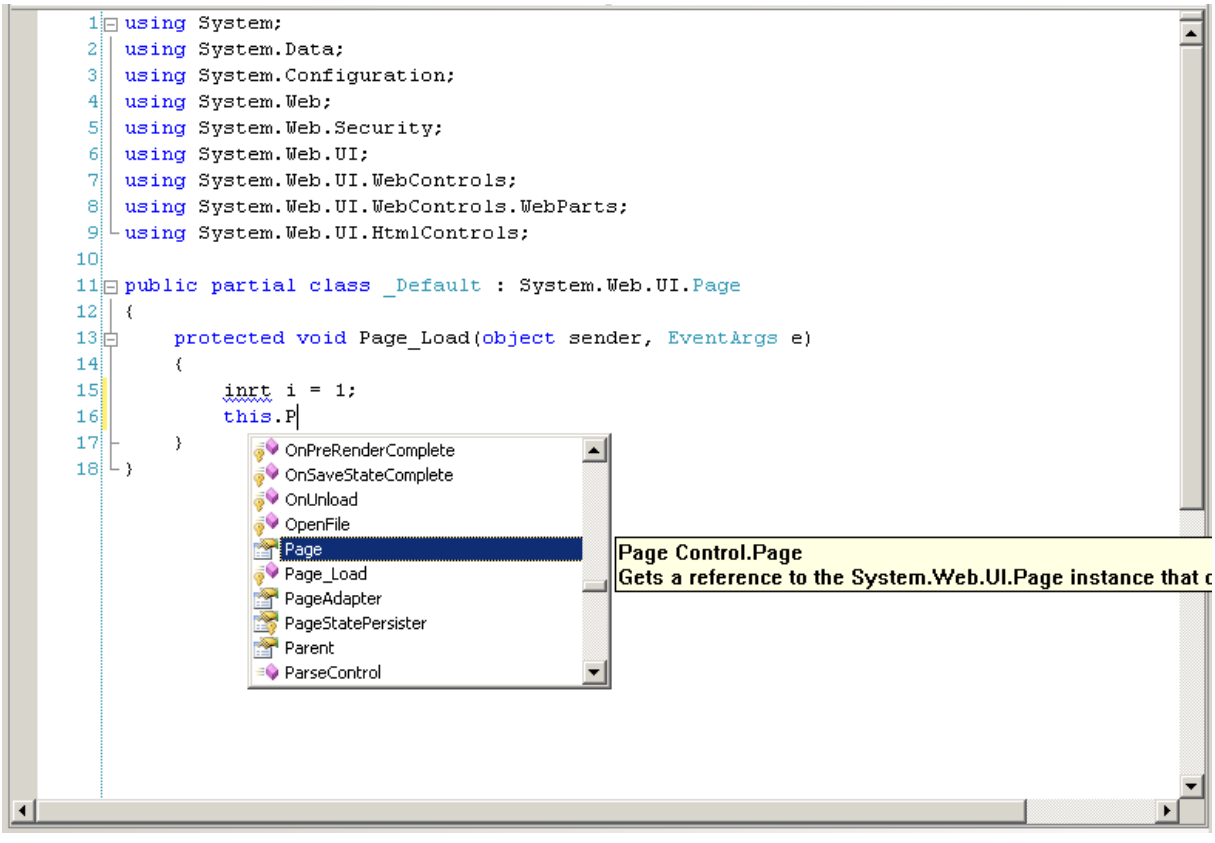
Všimněte si, že okno s vlastnostmi funguje i v režimu *Design*. Pokud někdo raději píše, než kliká myší, může barvu tlačítka samozřejmě změnit i přímo v editoru. Pokud v něm přepíšete červenou barvu (*Red*) na žlutou (*Yellow*), změní se vlastnost *BackColor* i v okně *Properties* a po přepnutí do režimu *Design* samozřejmě uvidíme tlačítko žluté.

Okno *Solution Explorer* vpravo nahoře slouží k organizaci jednotlivých souborů tvořících projekt. Můžeme v něm vytvářet nové položky (typicky webové stránky), přidávat existující položky, přidávat nové složky, mazat/přejmenovávat/přesouvat jednotlivé soubory či složky apod. Všechny tyto možnosti jsou přístupné v kontextové nabídce kliknutím na příslušné položce ve stromu pravým tlačítkem myši.

Posledním oknem, které si ukážeme, je editor kódu, ve kterém můžeme psát obsluhu událostí (stisknutí tlačítka apod.). Pokud máme otevřenou stránku, stačí stisknout tlačítko *View code* v horní liště okna *Solution Explorer*. Výsledek závisí na tom, zda máme kód v odděleném souboru anebo společně s HTML značkami v jednom souboru. V prvním případě má soubor stejné jméno jako stránka plus příponu podle použitého jazyka (vb pro Visual Basic a cs pro C#), v okně *Solution Explorer* je soubor jakoby zanořen pod příslušnou stránku (viz obrázek), ve druhém případě je kód vložen přímo do aspx souboru do sekce <script runat="server">.



Při psaní kódu máme k dispozici opět velký komfort. Na následujícím obrázku si povšimněte např. čísel řádků vlevo, barevného označování klíčových slov, znamének plus/minus pro rozbalení/sbalení bloků kódu, *IntelliSense* nápovědy při psaní kódu, podtrhávání špatných konstrukcí a syntaktických chyb apod.



První stránka krok za krokem

Nyní si vytvoříme první jednoduchou stránku. Nebudeme se zabývat otázkou „proč stránka funguje právě takto" – na to přijde řada v příští lekci. Cílem tohoto odstavce je prakticky si vyzkoušet práci s prostředím VWD.

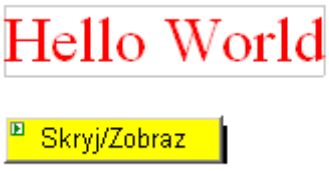
Pokud jste tak ještě neučinili, spusťte VWD. Vytvoříme nový projekt pomocí volby *File/New Web Site*. Jako šablonu zvolíme *ASP.NET Web Site*, výběr programovacího jazyka je na vašem uvážení, jako cestu zadejte *c:\WebSites\HelloWorld* a potvrďte. V projektu je již vytvořena výchozí stránka *default.aspx*, z cvičných důvodů si ale založíme novou stránku.

V okně *Solution Explorer* klikněte pravým tlačítkem na vašem projektu (kořen stromu) a zvolte *Add New Item...* Jako typ vyberte *Web Form*, jméno změňte např. na hodnotu *Hello.aspx*, programovací jazyk ponechte podle vlastního výběru, pro větší přehlednost zaškrtněte možnost *Place code in separate file* a potvrďte. Nyní bychom měli mít v editoru otevřenou novou prázdnou stránku *Hello.aspx*. Pokud chcete, můžete změnit titulek uzavřený ve značkách *<title>Untitled Page</title>* např. na *<title>Hello World</title>*. Pomocí tlačítka *Design* vlevo dole přepněte do režimu vizuální editace stránky.

Na stránce nyní vytvoříme dva vizuální prvky – textové pole bez editace (prvek *Label*) a tlačítko (prvek *Button*). Přetáhněte myší tyto dva prvky z okna *Toolbox* na pracovní plochu. Pro lepší vzhled doporučuji umístit kurzor mezi tyto dva prvky (např. kliknutím myši) a stisknout 2x *Enter* pro vložení nových řádků. Nyní nastavíme některé vlastnosti ovládacích prvků. Vyberte vždy příslušný ovládací prvek a nastavte jeho vlastnosti v okně *Properties* na nové hodnoty podle uvedené tabulky:

Prvek	**Vlastnost**	**Původní hodnota**	**Nová hodnota**
Label1	Text	Label1	Hello World
Label1	Font / Bold	False	True
Label1	ForeColor	-	Red
Label1	Font / Size	-	XX-Large
Label1	(ID)	Label1	LabelPozdrav
Button1	Text	Button	Skryj/Zobraz
Button1	BackColor	-	Yellow
Button1	(ID)	Button1	ButtonSkryjZobraz

Pokud vše nastavíme podle tabulky, pracovní plocha by měla vypadat zhruba takto (estetickou hodnotou našeho díla se teď raději nebudeme zabývat):



Třešničkou na dortu bude dopsání funkce tlačítka – požadavek zní, aby tlačítko při stisknutí střídavě skrývalo a zobrazovalo nápis „Hello World". Z vývojářského hlediska jde o velmi jednoduchou úlohu. Prvek *LabelPozdrav* má vlastnost *Visible*, jejíž hodnota určuje, zda je příslušný prvek viditelný. Chceme-li tlačítko zobrazit nebo skrýt, stačí nastavit tuto vlastnost na opak její současné hodnoty. Kód pro tuto akci je potřeba zapsat do události tlačítka nazvané *Click*, tato událost se volá vždy při stisknutí tlačítka. Seznam všech událostí ovládacího prvku lze zobrazit v okně *Properties*, zobrazení se aktivuje stisknutím tlačítka se symbolem blesku. Kód pro libovolnou událost začnete psát dvojklikem na příslušném řádku. Pro nejtypičtější událost (v případě tlačítka je to právě *Click*) je možné též dvojklik na příslušném ovládacím prvku (proveďte). Otevře se okno s kódem a vygenerovanou hlavičkou obsluhy události, do které dopište následující tučně zvýrazněný kód ve zvoleném jazyce:

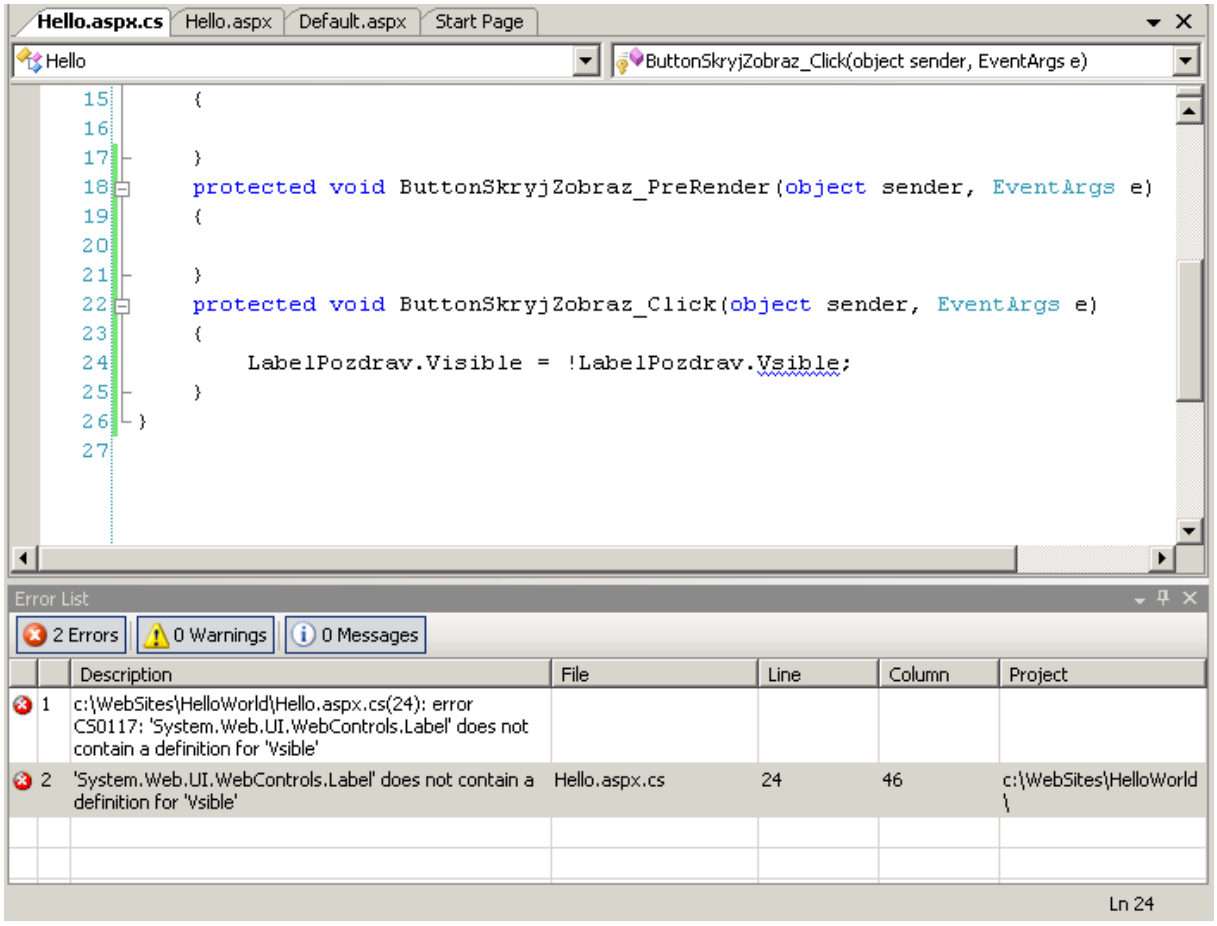
C#:

```
protected void ButtonSkryjZobraz_Click(object sender, EventArgs e)
{
    LabelPozdrav.Visible = !LabelPozdrav.Vsible;
}
```

Visual Basic:

```
Protected Sub ButtonSkryjZobraz _Click(ByVal sender As Object, ByVal e As System.EventArgs)
Handles ButtonSkryjZobraz.Click
    LabelPozdrav.Visible = Not LabelPozdrav.Vsible
End Sub
```

Při psaní si dáme pozor na malá a velká písmena – v jazyce C# je to nezbytnost, ve Visual Basicu bude kód čitelnější. Měl by vám rovněž fungovat *IntelliSense* a napovídat dokončení příkazu. V příkazu pro převrácení hodnoty viditelnosti je úmyslná chyba. Pokud nyní spustíme sestavení stránky či celého webu z nabídky *Build / Build Page* anebo *Build / Build Web Site*, zobrazí se nám chyby při kompilaci jako na následujícím obrázku:



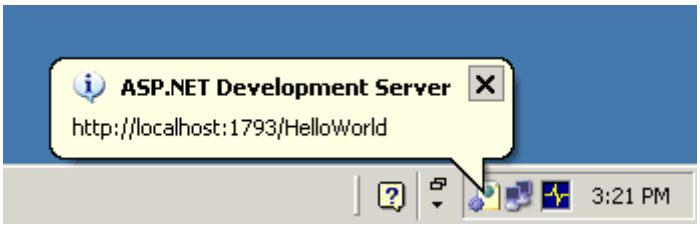
Jak je patrné, jedná se o chybu v souboru Hello.aspx.cs, v řádku 24 a sloupci 46 (pokud používáte Visual Basic, čísla budou jiná). V okně *Error List* je rovněž uveden její popis, chyba je navíc v editoru kódu označena vlnovkou. Náprava je nasnadě. Pokud provedeme dvojklik na řádku s popisem chyby, automaticky přeskočíme na příslušný řádek v editoru kódu. Opravíme-li nyní *Vsible* na *Visible* a provedeme znovu příkaz *Build*, neměla by se již žádná chyba objevit.

Spuštění a ladění Hello World

Hotovou aplikaci si nyní spustíme a vyzkoušíme si její ladění.

Spuštění hotové stránky

Nejprve stránku uložíme stisknutím tlačítka *Save* anebo z nabídky pomocí *File / Save <jméno stránky>*, případně *File / Save All*. Poté můžeme stránku spustit, nejsnáze tak, že klikneme pravým tlačítkem na příslušné stránce v okně *Solution Explorer* a zvolíme *View in Browser*. Naše „Hello World" stránka s textem a tlačítkem by se pak měla zobrazit v prohlížeči. Ještě předtím je ovšem spuštěn vývojový web server jménem *ASP.NET Development Server* – viz obrázek:

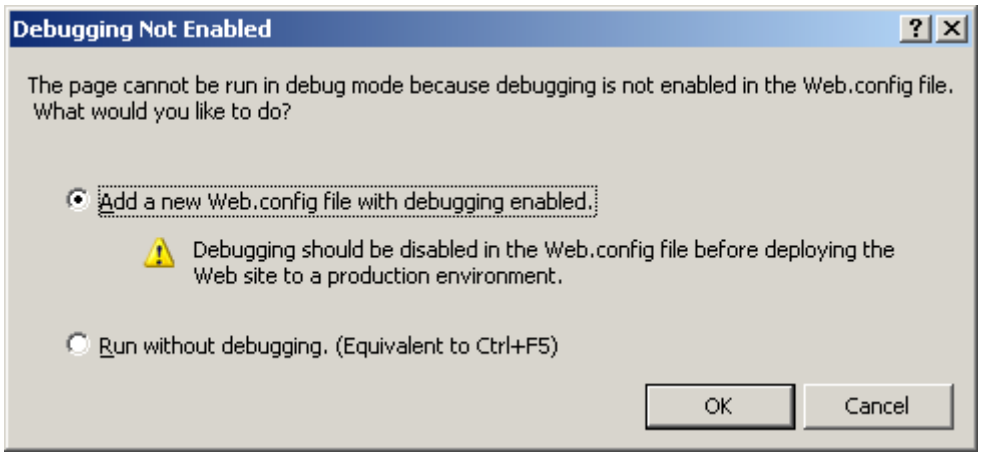


V mém případě běží server na portu 1793, ve vašem případě bude toto číslo pravděpodobně jiné. Pokud klikneme pravým tlačítkem myši na ikonce z předchozího obrázku, máme možnost webový server zastavit anebo se podívat na jeho detailní informace. Pokud chceme použít jiný port, můžeme v okně *Solution Explorer* zvolit myší webový projekt a v okně *Properties* nastavit *Use dynamic ports* na *False* a *Port Number* na požadovanou hodnotu. V mém případě je tedy adresa právě vytvořené webové stránky http://localhost:1793/HelloWorld/Hello.aspx. Stránka by se měla bez problémů zobrazit. Pokud budeme mačkat tlačítko na stránce, nápis „Hello World" by se měl střídavě skrývat a zase objevovat.

Ladění

Zatím jsme stránku spouštěli bez možnosti ladění kódu – je tak jednoduchá, že vlastně ani moc není co ladit. V praxi se ale setkáme se stránkami mnohem složitějšími, u nichž je pak možnost ladění neocenitelným pomocníkem při hledání chyb. Chceme-li stránku ladit, není to nic složitého. Prvním krokem je nastavení bodu přerušení (*Breakpoint*) na vhodném místě v kódu. Učiníme tak buď kliknutím na šedém poli úplně vlevo před příslušným řádkem anebo kliknutím pravým tlačítkem myši na příslušném řádku a volbou kontextové nabídky *Breakpoint / Insert Breakpoint*. Bod přerušení se projeví červeným pruhem pod zvoleným řádkem a symbolem „červená kulička" na začátku řádku. Odstranění bodu přerušení se provádí stejně. Druhým krokem je nastavení startovací stránky pro spuštění ladění. Je možné nastavit libovolnou stránku, v našem jednoduchém případě nastavíme naši stránku *Hello.aspx*. Klikneme na ní pravým tlačítkem myši v okně *Solution Explorer* a vybereme možnost *Set as Start Page*.

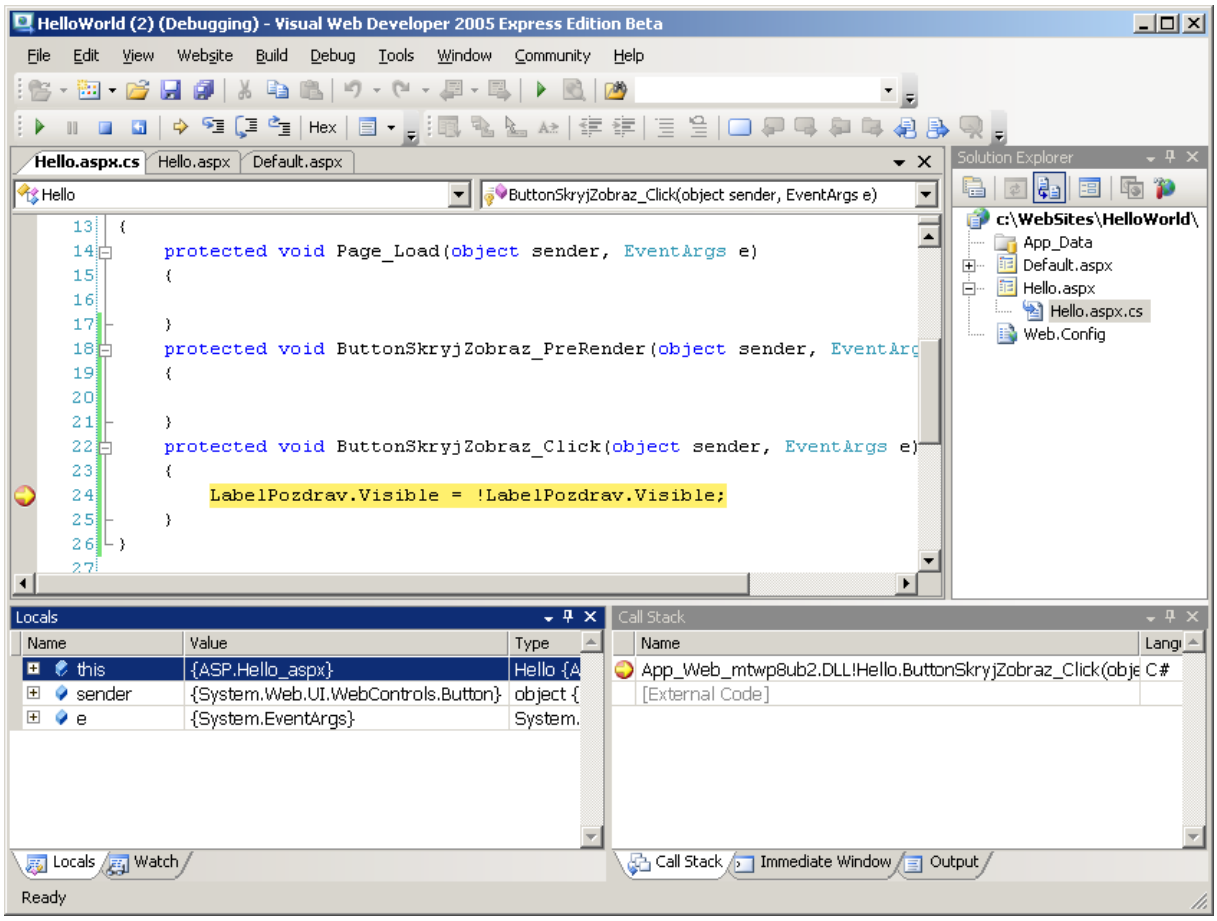
Vlastní ladění spustíme z nabídky *Debug / Start Debugging* (případně stisknutím tlačítka F5). Při prvním spuštění se nám zobrazí následující dialog:



Potvrďte jej beze změn, čímž vytvoříte nový konfigurační soubor *web.config*, ve kterém je povoleno ladění. Poté by se měla otevřít naše nová stránka. Stiskneme-li tlačítko, mělo by se nám „rozblikat“ okno VWD se zastaveným bodem přerušení.

Okna pro ladění

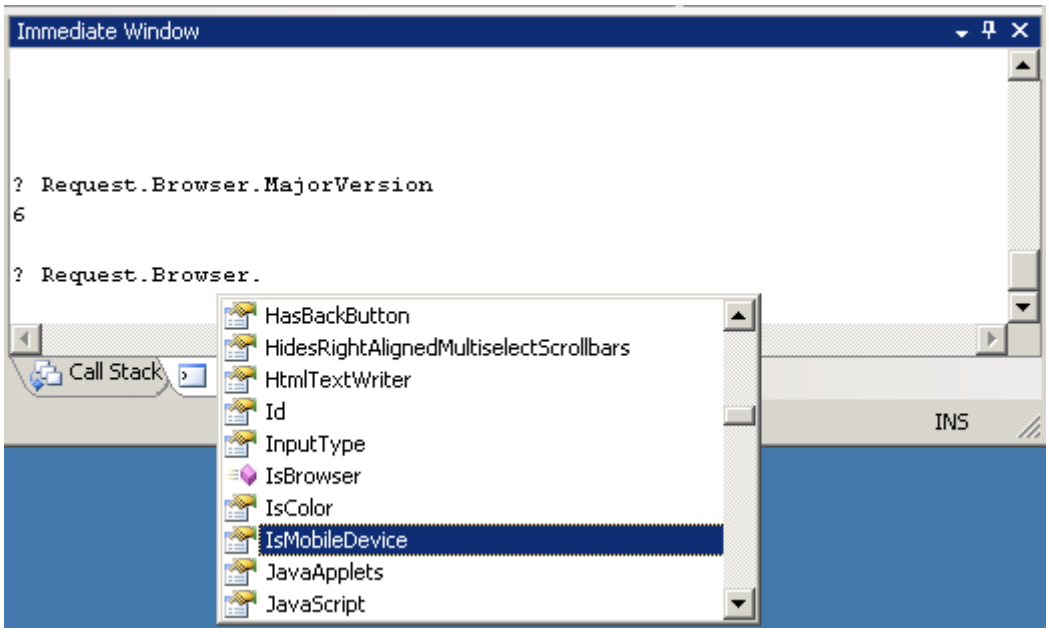
Nyní bychom měli vidět zhruba následující rozložení oken:



Zajímavé možnosti přináší již okno editoru kódu. Žlutě podsvícený řádek je ten, který by se měl právě vykonat, pokračování je možné použitím tlačítek *Step Into/Over/Out* anebo ekvivalentních příkazů v nabídce. Přemístíte-li kurzor nad některou proměnnou – např. nad *LabelPozdrav*, *Visible* nebo *sender*, zobrazí se hodnota dané proměnné – buď jako jednoduchá hodnota anebo rozbalovací strom, což je velmi praktické.

V okně *Locals* se můžeme podívat na aktuální hodnoty proměnných a dalších objektů, které používá blok kódu, ve kterém jsme právě zastaveni – opět jako jednoduché proměnné anebo jako rozbalovací strom hodnot. Záložkou dole můžeme přepnout na zobrazení okna *Watch*. Má podobnou funkci jako okno *Locals*, s tím rozdílem, že v něm jsou vidět hodnoty objektů a proměnných, které jsme se rozhodli trvale sledovat. Zkuste například kliknout pravým tlačítkem myši na proměnnou *LabelPozdrav* a zvolte z kontextové nabídky *Add Watch*. Objekt se přidá do okna *Watch* a můžeme jej během ladění trvale sledovat.

V pravé dolní polovině můžeme přepínat mezi třemi okny. Okno *Call Stack* obsahuje hierarchii volání mezi bloky kódu. Pokud například kód A volá kód B a ten zase volá kód C, uvidíme zde sekvenci C,B,A. V okně *Immediate Window* můžete zadat libovolný výraz, který bude okamžitě vyhodnocen a vrácen výsledek:



Všimněte si příjemné podpory *IntelliSense* nápovědy i v tomto okně. A konečně okno *Output* slouží k výstupu některých diagnostických informací. Pokud do něj chcete nechat vypsat vlastní diagnostický text, můžete tak snadno učinit např. příkazem:

System.Diagnostics.Debug.Print(„Nazdar")

Uvedení Hello World do provozu

Mini-aplikace nyní úspěšně běží na testovacím serveru a je čas přenést ji do provozního prostředí.

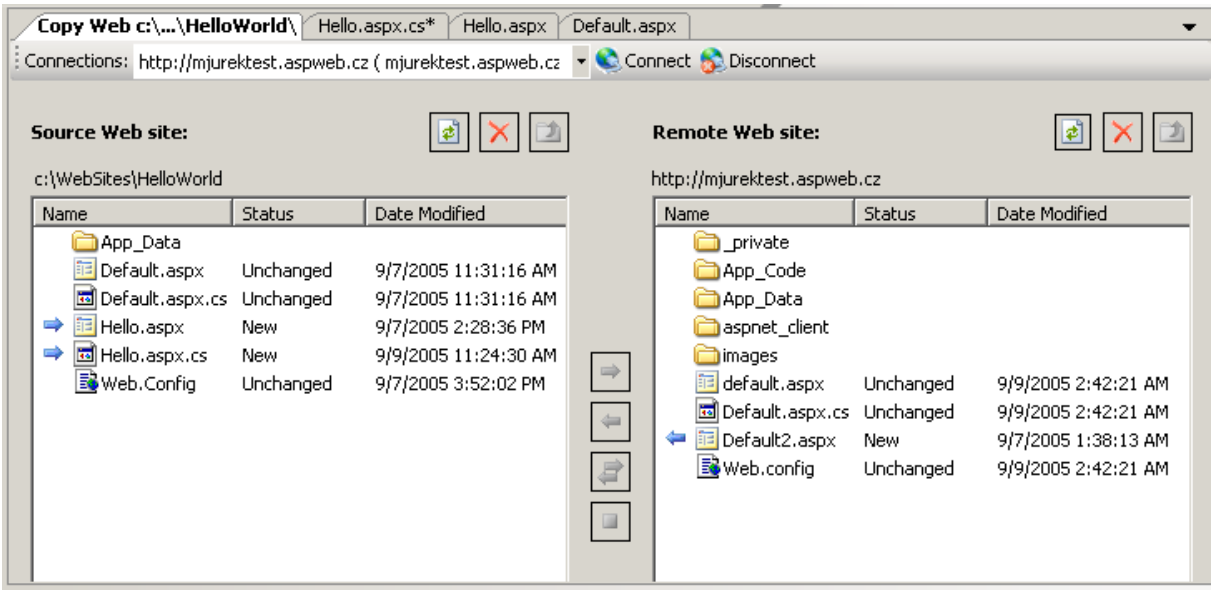
Nasazení na lokální síti

Pokud chceme aplikaci nasadit na lokální síti – typicky pro intranetové řešení, je postup velmi jednoduchý. Potřebujeme počítač s operačním systémem Windows 2000, Windows XP nebo Windows Server 2003 (preferovaná možnost) s běžícím Internet Information Serverem (IIS). Založíme složku na disku, vytvoříme virtuální adresář pro novou aplikaci a nakopírujeme do něj náš kompletní projekt. Pokud chceme aplikaci na serveru průběžně modifikovat, můžeme na serveru přidat podporu *FrontPage Extensions* anebo rozběhnout pro adresář aplikace FTP server. Nasazení aplikace je potom obdobné jako v následujícím odstavci. Přesný postup konfigurace serveru není obsahem tohoto kurzu.

Nasazení na vzdálený server

Webovou aplikaci je často potřeba nasadit na veřejném internetu. Pro tento účel potřebujete mít dohodu s firmou poskytujícím hostování aplikací – hosterem, který podporuje platformu ASP.NET 2.0. Většina hosterů nabízí různé úrovně služeb a od nich se odvíjející ceny. Někteří hosteři nabízejí pro jednoduché aplikace a testování jednu bezplatnou úroveň – v době psaní tohoto textu to v ČR a SR byli http://www.aspweb.cz/, http://www.asp2.cz, http://www.qsh.sk/ a http://www.aspx.sk. Aplikaci tak můžeme snadno publikovat na veřejném internetu.

Po podstoupení registrační procedury dostanete od hostera typicky k dispozici adresu URL, jméno a heslo. Ve VWD pak můžete využít funkce *Website/Copy Web Site...*, stisknout *Connect* a zadat potřebné údaje pro připojení.



Okno připomíná programy na organizaci souborů na disku. V levé části je lokální web (typicky adresářová složka), v pravé části vzdálený web (typicky web v hostingovém centru). Pomocí šipek je možné soubory přenášet mezi oběma weby. Důležitou funkcí je též detekce změněných souborů a data změny umožňující automatickou synchronizaci obou webů. Po přenosu na server hostera je aplikace veřejně dostupná na Internetu. V mém případě je na adrese http://mjurektest.aspweb.cz/Hello.aspx.

Závěrem

V dnešní lekci jsme se naučili základním úkonům při vývoji webové aplikace ASP.NET 2.0 pomocí nástroje VWD. Pokud něco nebude fungovat jak by mělo, zkuste využít možností podpory zmíněných ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

V příštím lekci si povíme něco více o „vnitřnostech" ASP.NET. Dozvíte se některé základní termíny a technologie, které umožňují fungování naší *Hello World* aplikace.

4. ZÁKLADNÍ PRINCIPY ASP.NET

V dnešní lekci se náhledneme pod povrch technologie ASP.NET. Vysvětlíme si, proč nám v předchozí lekci aplikace „Hello world" fungovala podle našich předpokladů. Pokud by vám některé věci v této lekci připadaly složité a zbytečné, můžete klidně dnešní lekci přeskočit a připojit se k nám v příští lekci.

Objektový model webových stránek

Do příchodu ASP.NET byl velký rozdíl mezi vývojem aplikací pro desktop a pro web. ASP.NET přineslo v tomto směru velkou změnu – vývoj webových a desktopových aplikací je založen na podobných principech.

Rozdíl mezi tradičním vývojem pro desktop a pro web

Desktopové aplikace se posledních 10-15 let vyvíjejí velmi snadno. K dispozici jsou moderní prostředí, jako je Visual Basic anebo Delphi. Aplikace se skládají z navazujících formulářů. Každý formulář je složen z ovládacích prvků – tlačítek, rozbalovacích seznamů, stromů, kalendářů apod. Vývojáři vytvářejí formuláře „přetahováním" jednotlivých prvků z palety dostupných prvků. Každý nástroj jich obsahuje omezené množství – pokud vám nabídka ovládacích prvků nestačí, můžete si za peníze či zadarmo sehnat další ovládací prvky a komponenty, anebo si je dokonce můžete napsat sami.

V pozadí ovládacích prvků je vyspělý objektový model. Každý ovládací prvek je po softwarové stránce objektem. Jeho vzhled a chování je určováno vlastnostmi (výška, šířka, barva pozadí, zobrazený text apod.). Prvky též na základě akcí uživatele vyvolávají události (stisknutí tlačítka, výběr prvku ze seznamu, změna textu v editačním poli apod.). Vlastní vývoj uživatelského rozhraní pak spočívá z velké části ve vytváření obslužného kódu událostí, např. při výběru prvku z rozbalovacího seznamu se aktualizují některá editační pole, při stisknutí tlačítka se zavolá zpracování právě prováděné úlohy apod.

Jednoduchost vývoje desktopových aplikací vedla k jejich velké popularitě, neboť vývoj uživatelského rozhraní byl pohodlný a rychlý. Také z hlediska koncových uživatelů jsou desktopové aplikace velmi příjemné – nabízejí velký komfort ovládání (klávesové zkratky, drag&drop myší apod.), skvěle využívají dostupný hardware a periférie, umožňují uložení dat na lokálním počítači apod. Naopak správci sítí desktopové aplikace nemilovali – instalace, podpora a aktualizace desktopových aplikací byla jejich noční můrou. A přestože Java i .NET dnes nabízejí technologie pro bezpracnou aktualizaci a údržbu desktopových aplikací, určitá averze vůči nim stále přetrvává.

Pracnost správy desktopových aplikací vedla v posledních cca 10 letech k velkému rozmachu webových aplikací, vyžadujících na straně klienta pouze jakýkoliv prohlížeč s podporou HTML. Množství webových intranetových i internetových aplikací tehdy rostlo geometrickou řadou. Ne každý si však plně uvědomoval negativa tohoto přístupu. Webové aplikace mají svoje limity pokud jde o komfort uživatelského rozhraní a využívání možností lokálního počítače. Například pokud chcete z webové aplikace používat nějakou periférii, musíte stejně na klienta nainstalovat nějakou komponentu (Java applet nebo ActiveX prvek), čímž do ní vnesete závislost na prostředí koncového uživatele a noční můra administrátorů je zpět.

Ještě větším problémem tradičně vyvíjených webových aplikací je neefektivita a nákladnost jejich vývoje. V prohlížeči jsou jednotlivé stránky zobrazovány na základě HTML značek v textovém souboru. Pomocí HTML lze snadno zobrazit pouze jednoduché vizuální prvky, k vytvoření složitějších prvků jako jsou kalendáře, menu nebo rozbalovací stromy, je již třeba velmi pokročilých znalostí. Vlastní vývoj webových aplikací v tradičních technologiích (PHP, ASP, Perl, JSP apod.) spočívá v míchání HTML značek s kódem v programovacím jazyce či skriptu. Výsledkem je velmi špatně čitelný kód (tzv. špagety kód), který se velmi pracně udržuje a mění, je velmi těžké sdílet ho mezi více projekty, vytvářet v týmu apod. Chybějící objektový model je důvodem pro chybějící kvalitní vývojové prostředí. Většina „vývojových" prostředí jsou ve skutečnosti pouze lepšími editory a ke komfortu vývojových prostředí pro desktopové aplikace mají stejně daleko jako Velorex k Mercedesu.

V čem je ASP.NET jiné?

ASP.NET není nic jiného, než přenesení osvědčených principů desktopového vývoje do prostředí vývoje webového. Podobně jako v desktopovém vývoji, ASP.NET stránka se skládá z ovládacích prvků – tlačítek, kalendářů, stromů apod., což jsou čistokrevné objekty na straně serveru. Jejich vzhled a chování je rovněž určováno vlastnostmi (výška, šířka, barva pozadí, zobrazený text apod.). Prvky stejně tak na základě akcí uživatele vyvolávají události (stisknutí tlačítka, výběr prvku ze seznamu, změna textu v editačním poli apod.). Vývojáři rovněž vytvářejí stránky „přetahováním" jednotlivých prvků z palety dostupných prvků v komfortním vývojovém prostředí. Pokud vám nabídka dostupných ovládacích prvků nestačí, můžete si rovněž za peníze či zadarmo sehnat další ovládací prvky a komponenty, anebo si je dokonce můžete napsat sami.

Na druhou stranu na klientovi je stejně jako u webových aplikací pouze čisté HTML verze 3.2, případně dokonce čisté XHTML. Na klienta se neinstalují žádné objekty – ani Java applety ani ActiveX ani .NET framework ani nic jiného. Jedinou interaktivitou na klientovi jsou jednoduché Java skripty, bez kterých se koneckonců dnes již téměř žádný web neobejde. Webové aplikace napsané v ASP.NET tak fungují v libovolném běžně dostupném prohlížeči s podporou HTML nebo XHTML a Java skriptu.

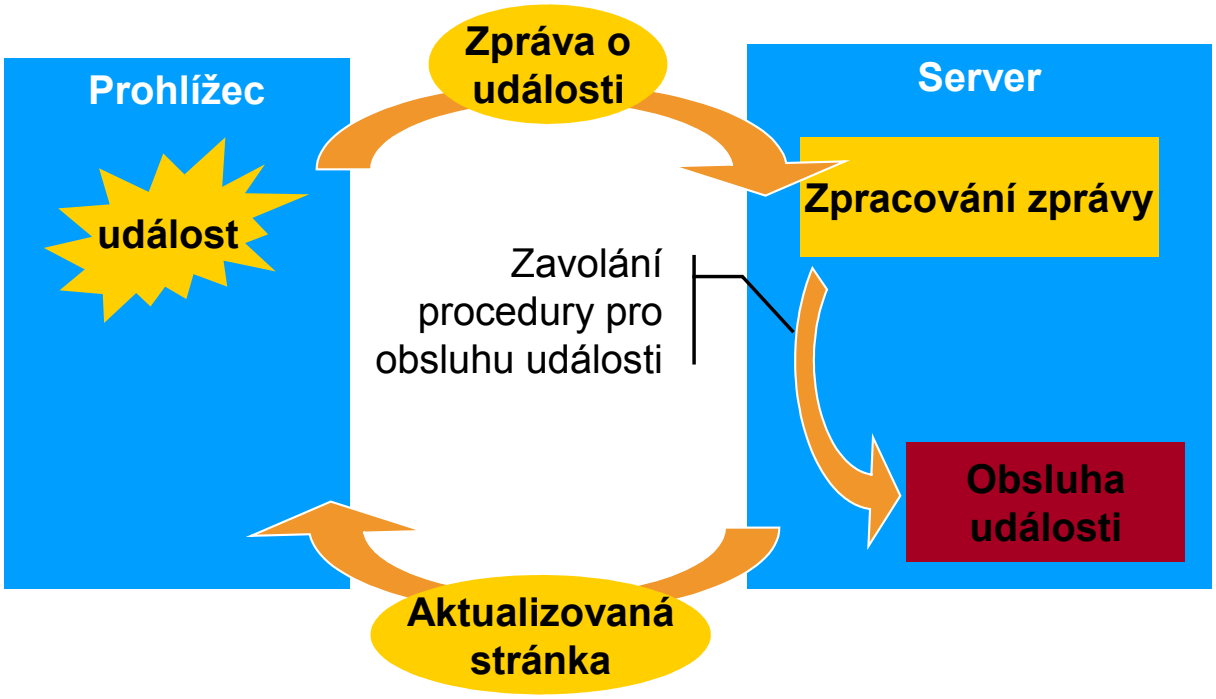
ASP.NET tedy ideálním způsobem kombinují rychlost a jednoduchost vývoje desktopových aplikací s minimálními požadavky na klientský počítač webových aplikací. Možná jste postřehli určitý rozpor – ASP.NET je na jednu stranu založená na objektech, ale na straně klienta žádné objekty nejsou. Kde jsou tedy objekty?

Vnitřnosti objektového modelu

Objekty v ASP.NET takříkajíc „žijí" na serveru, nikoliv na klientovi. Objekty jsou definovány v souboru aspx pomocí speciálních značek. Například v našem příkladu „Hello World" zde můžete najít definici objektu tlačítka vytvářeného na serveru jako:

<asp:Button ID="ButtonSkryjZobraz" runat="server" BackColor="Yellow" OnClick="ButtonSkryjZobraz_Click" Text="Skryj/Zobraz" />

Vztah mezi klientským prohlížečem a serverem s ASP.NET stránkou znázorňuje následující obrázek:



Objekty, jako jsou tlačítka či kalendáře, existují v paměti serveru pouze po dobu vykonávání žádosti klienta. Po vykonání celé stránky se na klienta pošle „obraz" jednotlivých objektů v podobě HTML značek. Například „obraz" objektu kalendář jsou HTML značky, které nám připomínají kalendář. Obraz žlutého tlačítka (objektu) s nápisem „Skryj/Zobraz" v předchozí lekci byly HTML značky zobrazující žluté tlačítko s příslušným nápisem, což můžeme snadno zjistit zobrazením zdroje v internetovém prohlížeči (*View Source*):

<input type="submit" name="ButtonSkryjZobraz" value="Skryj/Zobraz" id="ButtonSkryjZobraz" style="background-color:Yellow;" />

Obrazem červeného textu s viditelností (vlastnost *Visible*) nastavenou na *False* byl „neviditelný text" – tedy žádné HTML značky.

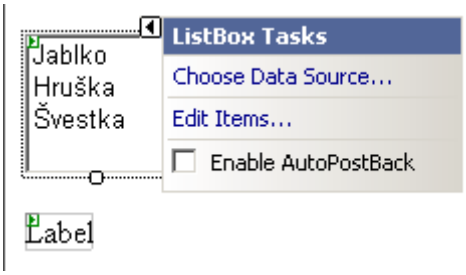
Pojem PostBack

Dobře, pokud objekty existují pouze na serveru a pouze po dobu vykonávání žádosti klienta, jak je možné, že tyto objekty reagují na událostí, jakými je stisknutí tlačítka nebo změna vybraného data v kalendáři?

Odpovědí je funkce nazvaná *PostBack* zobrazená na obrázku výše. Pokud dojde ke zmíněným událostem (např. výběr data v kalendáři), je prostřednictvím Java skriptu znovu zavolán server a jsou mu předány informace o proběhlé události. Na straně serveru se znovu vytvoří objektový model stránky (tlačítka, kalendáře apod.) a je zde vyvolána událost příslušného prvku (v případě kalendáře je to událost *SelectionChanged*). Pokud jste napsali kód obsluhující tuto událost, je tento kód vykonán. Na závěr je obraz celé (někdy aktualizované) stránky odeslán do klientského prohlížeče. V našem příkladě „Hello World" jsme přesně takto obsluhovali událost tlačítka *Click*, v jejíž obsluze jsme střídavě přepínali viditelnost – na klienta se tudíž střídavě odesílala stránka bez tlačítka a s tlačítkem.

Některé prvky mají provádění akce *PostBack* zapnuto, neboť by bez něj postrádaly smysl – typickým příkladem je tlačítko. Jiné prvky mohou mít tuto funkci zapnutou či vypnutou v závislosti na potřebách vývojáře. Typickým příkladem je seznam hodnot, u kterého se generování události při výběru hodnoty řídí nastavením vlastnosti *AutoPostBack*. Pojďme si toto chování vyzkoušet.

Založme v projektu novou stránku *PostBack.aspx* a přepněme ji do režimu *Design*. Myší přetáhneme na stránku prvek typu *ListBox* a prvek typu *Label*. *ListBox1* přejmenujeme nastavením vlastnosti *(ID)* na *DruhyOvoce*, *Label1* přejmenujeme stejným způsobem na *VybraneOvoce*. Pokud zvolíte prvek *DruhyOvoce* a kliknete na zobrazivším se trojúhelníčku vpravo nahoře, můžete do tohoto prvku pomocí volby *Edit Items...* přidat některé druhy ovoce jako na následujícím obrázku (stejnou akci lze provést i v okně *Properties* nastavením vlastnosti *Items*):



Nyní napíšeme kód pro obsluhu události – výběru nové položky ze seznamu hodnot a událost událost *SelectedIndexChanged*. Stačí dvakrát kliknout na seznamu hodnot a dopsat v editoru kódu tučně zvýrazněný kód ve zvoleném jazyce:

C#:

```
protected void DruhyOvoce_SelectedIndexChanged (object sender, EventArgs e)
{
    VybraneOvoce.Text = DruhyOvoce.SelectedValue;
}
```

Visual Basic:

```
Protected Sub DruhyOvoce_SelectedIndexChanged(ByVal sender As Object, ByVal e As System.EventArgs)
    VybraneOvoce.Text = DruhyOvoce.SelectedValue
End Sub
```

Pokud si nyní stránku uložíme, zobrazíme si ji a vybereme některou z hodnot, nemění se hodnota textu *VybraneOvoce* – je stále nastavena na původní hodnotu *Label*. Důvodem je vlastnost *AutoPostBack*, která je nastavena na hodnotu *False* – při změně vybrané hodnoty v seznamu tedy nedochází k zavolání serveru a akci *PostBack*, zkrátka „neděje se nic". Pokud však změníme u prvku *DruhyOvoce* v okně *Properties* anebo ve výše zobrazené rychlé nabídce vlastnost *AutoPostBack* na *True*, stránku uložíme a znovu zobrazíme v prohlížeči (musíte v něm dát *Refresh*), chování se změní. Při jakékoliv změně hodnoty vybrané ze seznamu se zavolá server, na něm se vyvolá námi napsaná obsluha události *DruhyOvoce_SelectedIndexChanged*, kde dojde k aktualizaci prvku *VybraneOvoce* s následným odesláním výsledku zpět do klientského prohlížeče.

Zároveň si můžete všimnout, že dojde k obnově stránky na klientovi, což se projeví „bliknutím" stránky v prohlížeči. Narazili jsme na úskalí tohoto postupu – každou takovouto akcí dochází k zatížení serveru a k malému zdržení na klientovi. Máme-li na serveru dostatečnou výkonovou rezervu a síťové spojení mezi serverem a prohlížečem je dostatečně rychlé, nezaznamenáme žádný pozorovatelný problém. Nejsou-li však tyto podmínky splněny, aplikace se může jevit jako pomalu reagující. Při využívání funkce *PostBack* je proto třeba přemýšlet nad funkcí uživatelského rozhraní a řídit se zásadou „všeho s mírou".

Pojem ViewState

Bystrý pozorovatel si možná všimnul dalšího rozporu – pokud objekty „žijí" na serveru pouze krátkodobě, jak server zjistí při příštím požadavku, zda je statický text právě zobrazen či nikoliv? Přesněji řečeno – kde se mezi jednotlivými PostBack události uchovává stav stránky?

Odpověď je jednodušší, než byste čekali. Server na konci zpracování získá od jednotlivých ovládacích prvků jejich stav a pošle ho zakódovaný a digitálně podepsaný ve skrytém vstupním poli prohlížeči. V našem případě „Hello World" můžete ve zdrojovém kódu najít něco jako:

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/
wEPDwUKLTE2MjY5MTY1NQ9kFgICAw9kFgICAQ8PFgQeBFRleHQFC0hlbGxvIFdvcmxkHgdWaXNp
YmxlaGRkZCCePWo7WfgvTZyzwtsWDEHgrvFY" />
```

Tento řetězec je pak při PostBack události předán zpět na server, kde slouží k rekonstrukci stavu ovládacích prvků.

Pojďme si nyní význam *ViewState* pro aplikaci vysvětlit na jednoduchém příkladě, při kterém si vyzkoušíme práci s vlastnostmi *IsPostBack* a *EnableViewState*. Budeme pokračovat v práci na předchozím příkladu s ovocem. Nejprve vymažte ze seznamu ovoce všechny druhy, které jste tam zapsali (vlastnost *Items* v okně *Properties*). Nyní dvakrát klikněte někde na prázdném místě stránky, čímž se dostanete k editaci události *Load* pro celou stránku. Tato událost se vyvolá vždy na začátku provádění stránky a často se používá pro složitější nastavování vlastností (doplňování položek z databáze apod.). Do této události vepíšeme kód pro vložení dvou položek do seznamu *DruhyOvoce*:

C#:

```
protected void Page_Load(object sender, EventArgs e)
{
    DruhyOvoce.Items.Add("Jablko");
    DruhyOvoce.Items.Add("Hruška");
}
```

Visual Basic:

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    DruhyOvoce.Items.Add(„Jablko")
    DruhyOvoce.Items.Add(„Hruška")
End Sub
```

Pokud si nyní vyzkoušíte funkci stránky, budete možná překvapeni. Při prvním zobrazení stránky uvidíte dva druhy ovoce (Jablko, Hruška). Pokud zvolíte některé ovoce, dojde k události *PostBack* po níž jsou ovšem druhy ovoce již čtyři (Jablko, Hruška, Jablko, Hruška). Po další *PostBack* události jich bude již šest a tak dále. Důvod je celkem jednoduchý. Obsah seznamu je uchováván jako stav (*ViewState*) zachovávaný mezi jednotlivými žádostmi a při každé *PostBack* události jsou do seznamu přidány nové dva druhy ovoce. Tato nežádoucí situace má dvě možná řešení.

Prvním je vypnout *ViewState* pro náš prvek *DruhyOvoce*, čímž se seznam prvků nebude ukládat do stavu žádosti a zároveň se zmenší velikost komunikace mezi klientem a serverem. Server pak při každé žádosti přidá dva druhy ovoce do prázdného seznamu. Toho dosáhnete snadno – zvolte prvek *DruhyOvoce* a v okně *Properties* nastavte vlastnost *EnableViewState* na hodnotu *False* (vyzkoušejte). Když nyní stránku vyzkoušíte, měli byste vidět stále pouze dva druhy ovoce. Vypnutí *ViewState* má ale jeden nepříjemný efekt, který se projeví při volbě některého druhu – prvek si „nepamatuje" naposledy zvolený druh ovoce a neaktualizuje se tedy hodnota textu pod ním. Přístup lze tedy použít pouze v těch případech, kdy jsou prvky jakoby pasivní a nejsou součástí nějaké pokročilejší funkčnosti.

Druhý přístup je elegantnější a univerzálnější – *ViewState* znovu zapneme nastavením vlastnosti *EnableViewState* na hodnotu *True* (proveďte) a poté provedeme následující úpravu kódu

C#:

```
protected void Page_Load(object sender, EventArgs e)
{
    if( ! IsPostBack)
        DruhyOvoce.Items.Add("Jablko");
        DruhyOvoce.Items.Add("Hruška");
    }
}
```

Visual Basic:

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    If Not IsPostBack Then
        DruhyOvoce.Items.Add(„Jablko")
        DruhyOvoce.Items.Add(„Hruška")
    End If
End Sub
```

Nejde tedy o nic složitého – přidání položek do seznamu hodnot jsme podmínili pomocí vlastnosti *IsPostBack*, která má při prvním zobrazení stránky hodnotu *False* a při následných *PostBack* událostech hodnotu *True*.

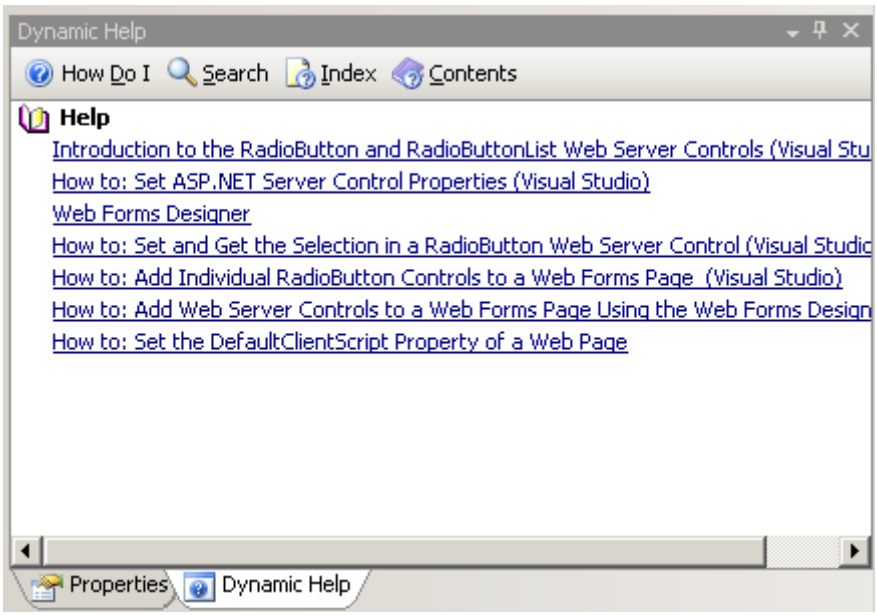
Ovládací prvky

Pojďme si nyní udělat malou přehlídku dostupných ovládacích prvků. Nejprve se podíváme na prvky dostupné takříkajíc „v krabici“, které jsou součástí ASP.NET 2.0 a poté alespoň naznačíme, jak vypadá svět vně krabice.

Prvky v krabici

Součástí ASP.NET 2.0 je několik desítek dostupných ovládacích prvků. Pro pohodlí vývojářů jsou ve VWD seskupeny do několika kategorií, které můžete vidět a rozbalovat v okně *Toolbox* (nevidíte-li, zvolte z nabídky *View/Toolbox*).

Bezkonkurenčně nejvíce prvků najdete v kategorii *Standard*. Zde najdete nejčastěji používané ovládací prvky – tlačítka, textové vstupy, rozbalovací seznamy, kalendáře apod. Všechny dosud použité prvky patřily do této kategorie. Jednotlivé prvky si můžeme snadno vyzkoušet jejich přetažením na stránku. Pokud si nejste jisti, jak který z prvků použít, navolte prvek myší a poté zvolte z nabídky *Help/ Dynamic Help*. Otevře se okno, které zobrazí nejpravděpodobnější odkazy z integrované nápovědy (za předpokladu, že jste integrovanou nápovědu *MSDN Express* nainstalovali). Na obrázku vidíte obsah okna při vybraném ovládacím prvku *RadioButton*:

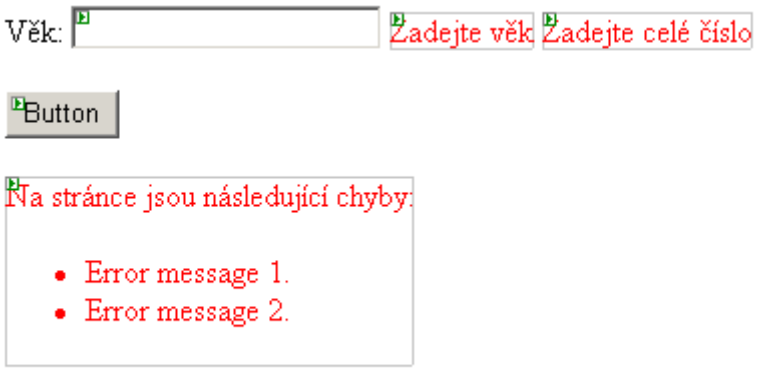


Druhou skupinou v pořadí (a druhou nejčastěji používanou) skupinou je *Data*. Zde se nachází ovládací prvky pro práci s daty. S některými z těchto prvků se seznámíme v šesté a sedmé lekci kurzu.

Zajímavou a nesmírně užitečnou skupinou jsou validátory v sekci *Validators*. Validátory slouží ke kontrole zadaných hodnot na stránce. Pojďme se na validátory podívat na malém příkladu – vyrobíme stránku pro zadání věku, což může být pouze celé číslo od 0 do 130 (necháváme si raději rezervu). Vytvořme proto novou stránku v oblíbeném jazyku a nazvěme ji *ZadaniVeku.aspx*. Na stránku nyní napíšeme „Věk: “, přesuneme prvky *TextBox*, *RequiredFieldValidator* a *RangeValidator*. Stiskneme 2x *Enter* pro odřádkování a přeneseme myší prvek *Button*, znovu stiskneme 2x *Enter* pro odřádkování a vložíme prvek *ValidationSummary*. Nyní nastavíme některé vlastnosti ovládacích prvků. Vyberte vždy příslušný ovládací prvek a nastavte jeho vlastnosti v okně *Properties* na nové hodnoty podle uvedené tabulky:

Prvek	Vlastnost	Původní hodnota	Nová hodnota
TextBox1	(ID)	TextBox1	Vek
RequiredFieldValidator1	ControlToValidate	-	Vek
RequiredFieldValidator1	ErrorMessage	RequiredFieldValidator	Zadejte věk
RequiredFieldValidator1	Display	Static	Dynamic
RangeValidator1	ControlToValidate	-	Vek
RangeValidator1	ErrorMessage	RangeValidator	Zadejte celé číslo
RangeValidator1	Type	String	Integer
RangeValidator1	MinimumValue	-	0
RangeValidator1	MaximumValue	-	130
RangeValidator1	Display	Static	Dynamic
ValidationSummary1	HeaderText	-	Na stránce jsou následující chyby

Pokud jsme vše udělali podle návodu, měli bychom vidět zhruba následující obrázek:



Pokud si nyní zobrazíme stránku v prohlížeči a stiskneme tlačítko, měli bychom na stránce dostat upozornění na nutnost zadání věku. A pokud vyplníme řetězec, který nepředstavuje celé číslo, dostaneme zase upozornění na špatný formát zadaných dat. Dole na stránce je navíc umístěn nepovinný prvek *ValidationSummary*, který přehledně vypisuje varování ze všech validátorů na stránce. Pokud bychom naopak chtěli zobrazovat pouze tento přehled a nikoliv jednotlivé validátory, můžeme jejich zobrazení potlačit (nastavením vlastnosti *Display* na *None*). Povšimněme si rovněž, že nebylo nutné napsat ani jediný řádek kódu. Pozor, validátory nejsou ochranou proti zlovolným uživatelům, kteří mohou vypnutím JavaScriptu validátory vyřadit. Pro kontrolu bychom proto na serveru měli vždy před zpracováním dat zkontrolovat platnost dat (validátorů) pomocí kontroly vlastnosti stránky *IsValid*.

Další skupinu tvoří prvky v kategorii *Navigation*. Zde se nachází prvky pro zobrazování hierarchických dat – mapy webu, struktury katalogu zboží, obsahu knihy apod. Se třemi zde zastoupenými prvky – nabídka *Menu*, rozbalovací strom *TreeView* a zobrazení cesty *SiteMapPath* se setkáme v sedmé lekci v pokročilejší kapitole o práci s daty.

V kategorii *Login* se setkáme s prvky pracujícími s přihlašovacími účty uživatelů, přihlášení a odhlášení, založení nového účtu, změnu hesla apod. S mnohými z těchto ovládacích prvků se setkáme v deváté lekci.

V kategorii *WebParts* jsou ovládací prvky pro vytváření personalizovaných stránek. Tyto ovládací prvky se nejčastěji používají pro vytváření portálů, u kterých si uživatelé mohou sami do určité míry měnit rozmístění a nastavení jednotlivých webových dílců (*web parts*). Toto téma je za hranicí našeho kurzu, pokud se chcete dozvědět víc, doporučujeme například tento odkaz[19]. V kategorii *HTML* jsou prvky pomáhající při převodu existujících stránek do ASP.NET, při běžné práci se zpravidla nepoužívají.

19 http://msdn.microsoft.com/msdnmag/issues/05/09/WebParts/toc.asp

Nenašel jsem prvek ...

Co dělat, pokud potřebujeme nějaký ovládací prvek a nenašli jste jej v *Toolboxu*? Máme v zásadě dvě možnosti: napsat si ho sami anebo se podívat, zda to již neudělal někdo jiný.

Méně pracná je druhá možnost. Spousta lidí a firem vytvořila velké množství různých ovládacích prvků, které se od sebe výrazně liší. Některé jsou vysoce kvalitní, jiné méně. Některé podporují řadu prohlížečů, u jiných je to s kompatibilitou horší. Některé jsou zcela zdarma, jiné si musíte koupit (i když zpravidla za cenu v řádu tisíců korun, za kterou je určitě nejste schopní sami napsat). Některé jsou dodávány i se zdrojovým kódem, jiné bez něj. Katalog stovek ovládacích prvků najdete na webu http://www.asp.net v sekci *Control Gallery*.

Pokud si chceme napsat ovládací prvek sami, potřebujeme již mírně pokročilé znalosti. Vytváření vlastních ovládacích prvků ale není možné ve VWD edici. Buď musíme mít některou vyšší edici Visual Studia 2005 (alespoň Standard) anebo používat kompilátor z příkazové řádky, což je poměrně nepohodlné řešení. Vývoj ovládacích prvků využívá principy objektově orientovaného programování – nové prvky vytváříme pomocí dědičnosti buď z abstraktních základních tříd anebo z některého konkrétního prvku. Více informací o této vývojářské disciplíně naleznete například zde[20].

Závěrem

V této lekci jsme si řekli něco málo o technologických základech, ze kterých ASP.NET vychází. Protože většina aplikací potřebuje nějaká data, jako jsou katalog výrobků, seznam fotek, sportovní výsledky apod., seznámíme se v příští lekci s databázovou platformou SQL Server 2005, konkrétně bezplatnou edicí Express. V přespříští a přespřespříští lekci pak budeme data z databáze zobrazovat na webových stránkách.

A ještě připomenutí – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

20 http://msdn.microsoft.com/library/en-us/dnvs05/html/custwebcon.asp

5. DATABÁZE SQL EXPRESS – ZÁKLADY

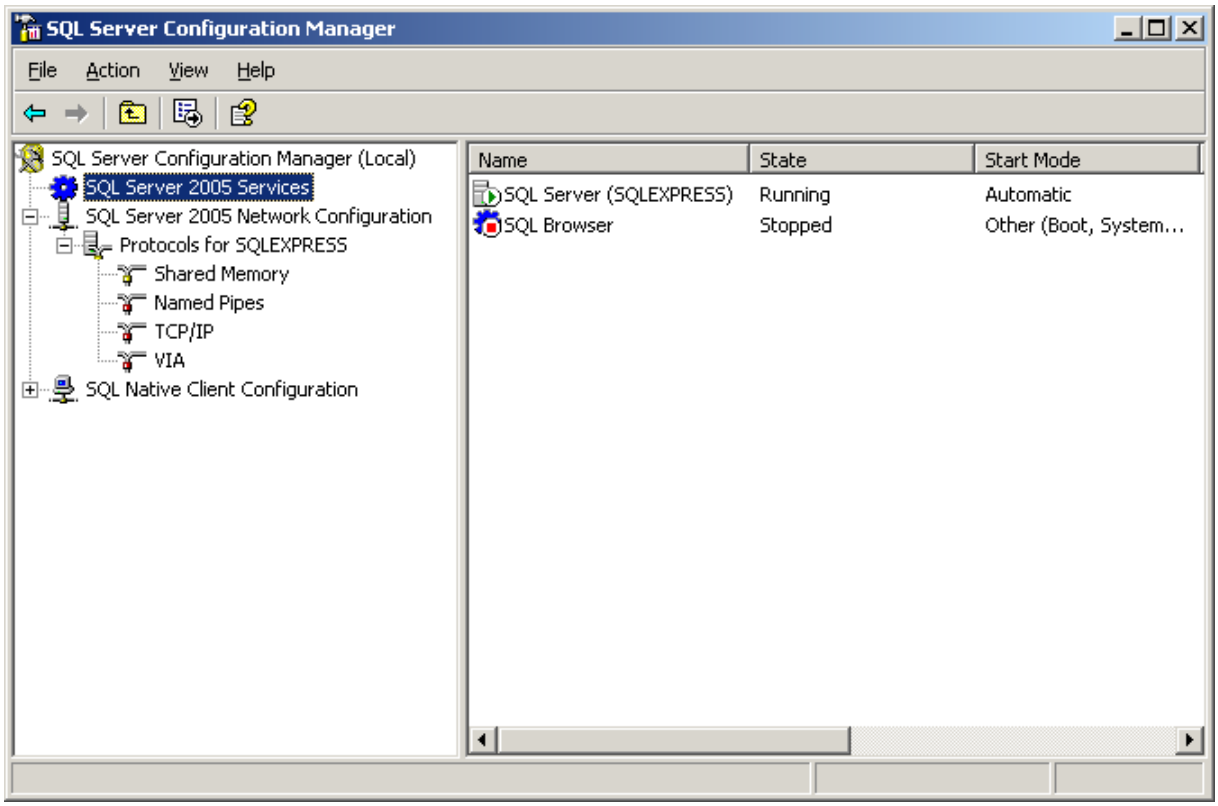
V páté lekci změníme téma – přejdeme od vývoje webových aplikací k práci s databází, neboť většina webů s daty v nějaké formě pracuje. V příští lekci se k ASP.NET opět vrátíme – v lekci o práci s daty v ASP.NET skloubíme znalosti webového vývoje s databázovými znalostmi. Budeme používat databází *SQL Server Express 2005* (budeme používat zkrácený název *SQL Express*). Nejprve si povíme něco o dostupných nástrojích pro vývoj a správu databáze a poté se naučíme základním úkonům.

Dostupné nástroje

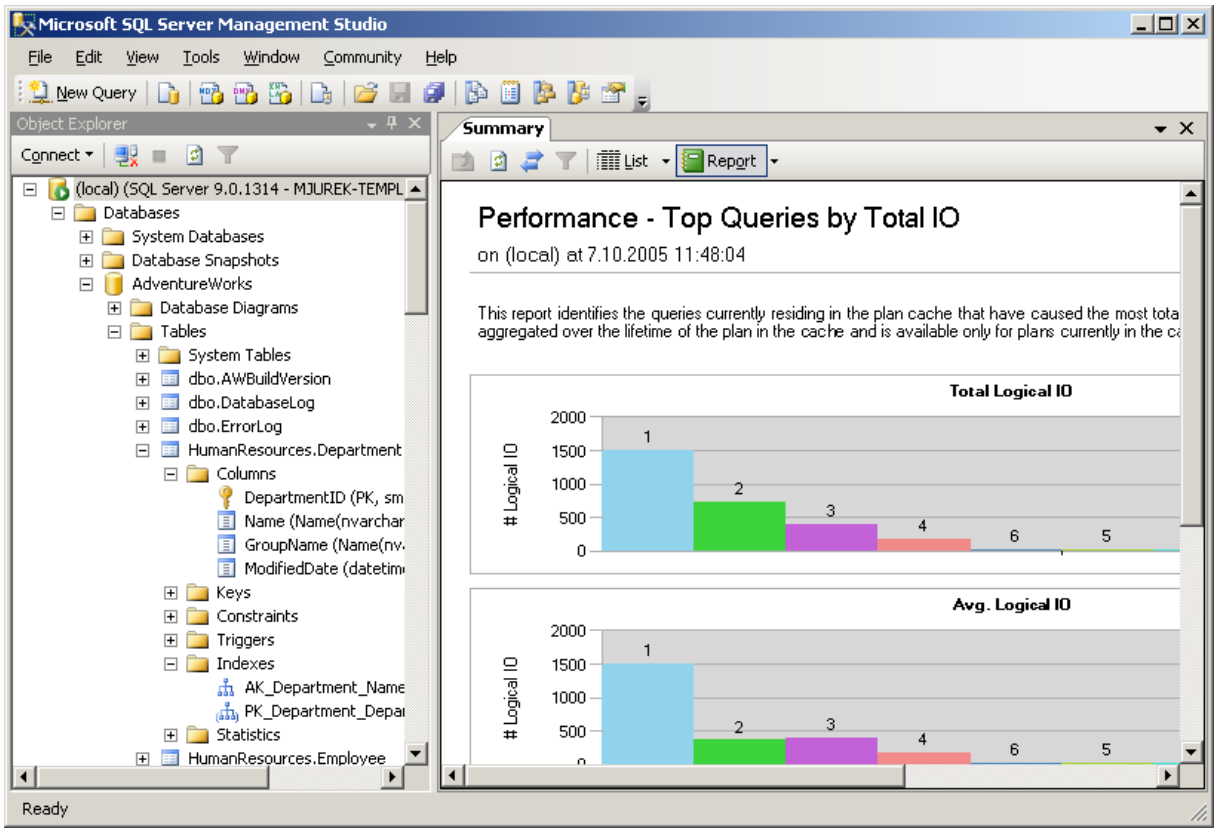
K SQL Serveru existuje celá řada nástrojů a je vhodné vybrat si ten správný. Můžeme je rozdělit na nástroje s grafickým rozhraním a nástroje příkazové řádky.

Grafické nástroje

Prvním nástrojem, s kterým se seznámíme je *SQL Configuration Manager*. Jedná se o nástroj k základní správě a konfiguraci (viz obrázek). Najdete ho ve Start menu pod *Microsoft SQL Server 2005/ Configuration Tools*. Pomocí tohoto nástroje lze zastavovat a spouštět jednotlivé nainstalované instance SQL Serveru a měnit účet, pod kterým služba běží. V nástroji lze rovněž určovat používané síťové protokoly a jejich nastavení. Můžete si sami ověřit, že žádný z protokolů není z důvodu bezpečnosti po instalaci povolen. K SQL Serveru se tak můžeme připojit pouze lokálně – ze stejného počítače, kde běží. Pro zastavování/spouštění služby a změnu jejího účtu můžeme rovněž použít standardní nástroj Windows jménem *Services/Služby* v nabídce *Administrative Tools/Nástroje pro správu*.



K vyšším edicím SQL Serveru 2005 (Workgroup, Standard, Enterprise) je dodáván nástroj *SQL Server Management Studio*. Tento nástroj je vůbec nejlepším nástrojem pro správu SQL Serveru – můžeme s ním administrovat a měnit všechny jeho aspekty a nastavení, generovat skripty, plánovat provádění úloh a další (viz obrázek). Pokud vlastníte licenci k vyšší verzi SQL Serveru anebo třeba máte nainstalovanou zkušební verzi tohoto produktu, můžete tento nástroj používat i pro správu Express verze (pozor, pokud chcete SQL Express spravovat vzdáleně, musíte na něm nejprve povolit nějaký síťový protokol).



K předběžným Beta verzím SQL Express se dal stáhnout přímo z webu Microsoftu nástroj *SQL Express Manager*, který umožňoval pokládat dotazy do databáze a několik dalších věcí. Podle nedávno zveřejněných informací byl vývoj tohoto nástroje zcela zastaven. Místo něj by se měla na jaře 2006 objevit druhá edice SQL Express obsahující nový nástroj pro správu, který bude odlehčenou edicí nástroje *SQL Server Management Studio* z komerčních verzí. Jeho předběžná verze se nazývá *SQL Server Management Studio Express (SSMSE) – Community Technical Preview November 2005*, lze ji stáhnout zde[21].

Pro většinu základních vývojářských úkonů nad databází vystačíme s prostředím, které nám nabízí *Visual Web Developer (VWD)*, kde je k dispozici okno *Database Explorer*, které lze vyvolat z nabídky *View*. V něm je možné vytvářet tabulky, pohledy, uložené procedury a funkce. Nejzajímavější funkcí je možnost editace databáze prostřednictvím databázového diagramu. Vzhledem k tomu, že se práci s databází ve VWD budeme věnovat celý zbytek dnešní lekce, pokročíme nyní dále.

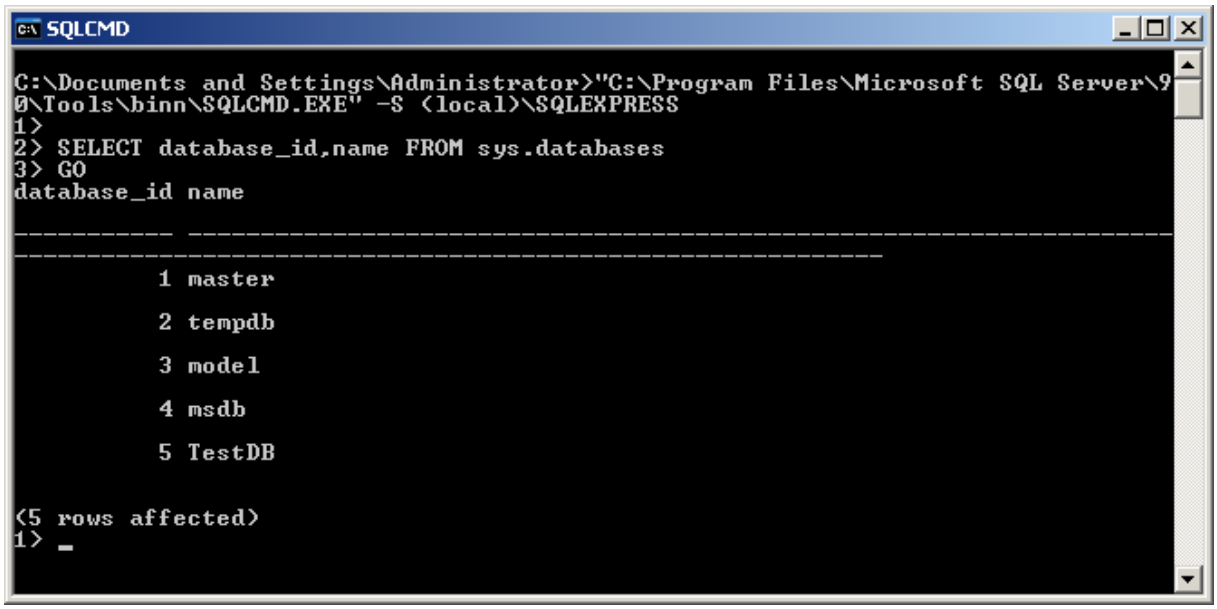
Existují též nástroje pro dálkovou administraci SQL serveru pomocí webového rozhraní. Tyto nástroje se liší schopnostmi i cenou (mnohé jsou zcela zdarma), v našem kurzu se jim nebudeme věnovat. Namátkou lze uvést:

- http://microsoftsqltools.com/
- http://www.microsoft.com/downloads/details.aspx?FamilyId=C039A798-C57A-419E-ACBC-2A332CB7F959&displaylang=en
- http://www.aspenterprisemanager.com

21 http://www.microsoft.com/downloads/details.aspx?amp;displaylang=en&familyid=82AFBD59-57A4-455E-A2D6-1D4C98D40F6E&displaylang=en

Nástroje příkazové řádky

Pro pokročilejší uživatele jsou k dispozici nástroje příkazové řádky. Nejčastěji používaným nástrojem je *sqlcmd.exe*, který najdete ve složce *C:\Program Files\Microsoft SQL Server\MSSQL.?\MSSQL\Binn* Pomocí tohoto nástroje můžete vykonávat libovolné příkazy ve skriptovacím jazyce T-SQL (viz obrázek s výpisem databází). Dokumentaci k jazyku T-SQL můžete stáhnout z webu Microsoftu pod jménem SQL Server 2005 Books Online[22].



Základní úkony

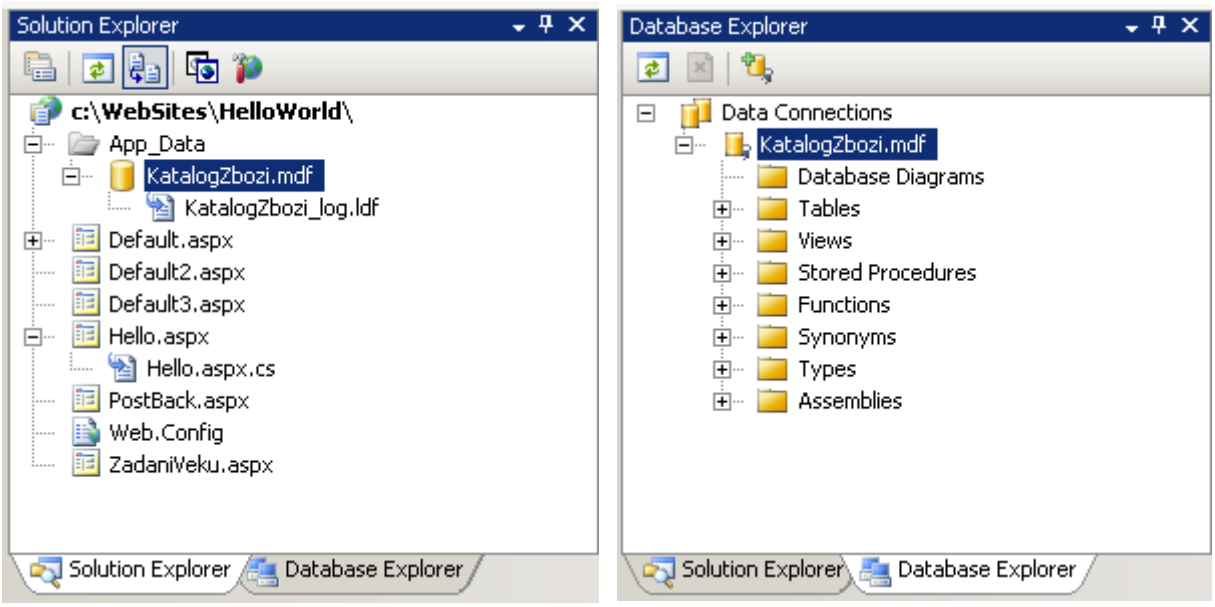
Nyní si projdeme nejzákladnější úlohy, které budeme pro práci s databázemi potřebovat.

Založení databáze a diagramu

Databázi lze založit dvojím způsobem. V prvním případě vytváříme pojmenovanou databázi, která je trvale připojena k serveru a může být sdílena více aplikacemi. Tento způsob se používá u komerčních verzí databází (kde ani jiný způsob není). Volba pro vytvoření sdílené pojmenované databáze je v okně *Data Explorer* v kontextové nabídce na složce *Data Connection* označena *Create New SQL Server Database...* Bohužel ve finální verzi VWD je znepřístupněná (pravděpodobně je jenom ve vyšších edicích Visual Studia), proto musíme použít jiný nástroj – buď příkaz *CREATE* DATABASE v řádkovém *sqlcmd.exe* anebo příslušnou funkci v nástroji *SQL Management Studio*.

Pro nás ale bude daleko zajímavější druhý způsob – vytváření databáze pro jednu aplikaci, přičemž tato databáze nemá sdílené jméno – je to zkrátka soubor uložený v adresáři aplikace. Pojďme na to – ve VWD v okně *Solution Explorer* zobrazíme kontextovou nabídku na složce s webovým projektem, zvolíme *Add New Item...*, vybereme typ *SQL Database*, jako jméno zadáme *KatalogZbozi.mdf* a potvrdíme, stejně jako odsouhlasíme upozornění, že databáze bude raději umístěna ve speciální složce *App_Data*. Po malé chvilce uvidíte nově vytvořenou databázi v okně *Solution Explorer*, pod ní můžete vidět soubor *KatalogZbozi_log.ldf*, což je tzv. transakční log pro zajištění neporušenosti dat v případě havárie (viz obrázek). Pokud se vám samo neotevřelo okno *Database Explorer*, můžete ho zobrazit dvojím kliknutím na databázi (mdf souboru, viz obrázek).

22 http://www.microsoft.com/technet/prodtechnol/sql/2005/downloads/books.mspx



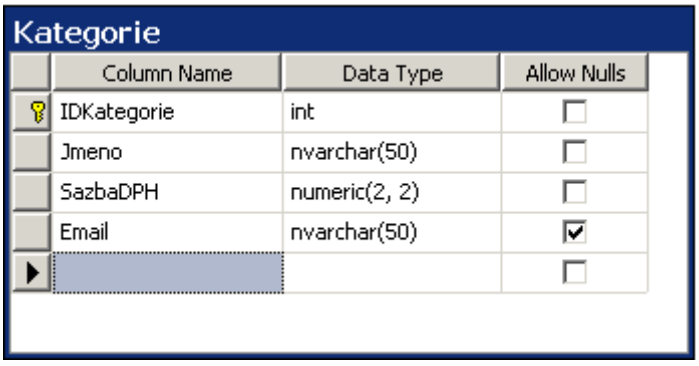
Nejčastější úlohy lze s výhodou provádět v zobrazení databázového diagramu, který ukazuje strukturu tabulek a jejich vzájemné vazby. Vytvoříme si proto nový databázový diagram pro provádění dalších prací. Zobrazíme si kontextovou nabídku na uzlu *Database Diagrams* a zobrazíme si nabídku *Add New Diagram*. Pokud jsme dotázáni na vytvoření databázových objektů, můžeme bez obav potvrdit jejich vytvoření. Dialog *Add Table* zavřeme pomocí *Close*, čímž máme k dispozici panenský databázový diagram.

Založení tabulky

Do diagramu můžeme přidat novou tabulku. Budeme vytvářet fiktivní katalog zboží, který bude obsahovat dvě tabulky – tabulku kategorií zboží a tabulku produktů, přičemž každý produkt bude zařazen právě v jedné kategorii. Začneme tabulkou kategorií, která bude obsahovat:

- Číslo kategorie – jednoznačný identifikátor (ID), který se nemění
- Jméno kategorie (např. nápoje)
- Sazbu DPH pro danou kategorii
- E-mail na kontaktní osobu pro danou kategorii

Tabulku vytvoříme zobrazením nabídky kdekoliv na prázdné ploše diagramu a volbou *New Table*... Jako název tabulky zvolíme *Kategorie*. Zobrazí se nám obdélník symbolizující danou tabulku, ale zatím bez řádků zastupujících jednotlivá datová pole. Jednotlivé řádky vyplníme podle následujícího obrázku:



V prvním sloupci jsou uvedena jména datových polí (raději se vyvarujeme diakritiky). V druhém sloupci je uveden typ. Nejčastěji budeme používat následující typy:

- řetězec – typ *nvarchar*, kde v závorce je uveden maximální počet znaků. Nepoužívejte typ *varchar*, má sice menší nároky na uložení dat, ale mívá potíže s diakritikou.
- celé číslo – typ *int*
- číslo s pevnou desetinou tečkou – *numeric*, v závorce je celkový počet číslic a počet desetinných míst; vhodné pro uložení finančních a jiných přesných čísel
- číslo s plovoucí desetinnou tečkou – *float*, vhodné pro inženýrské a vědecké výpočty
- pravda/nepravda – typ *bit*, jež může nabývat hodnoty 1 nebo 0
- datum a čas – typ *datetime*

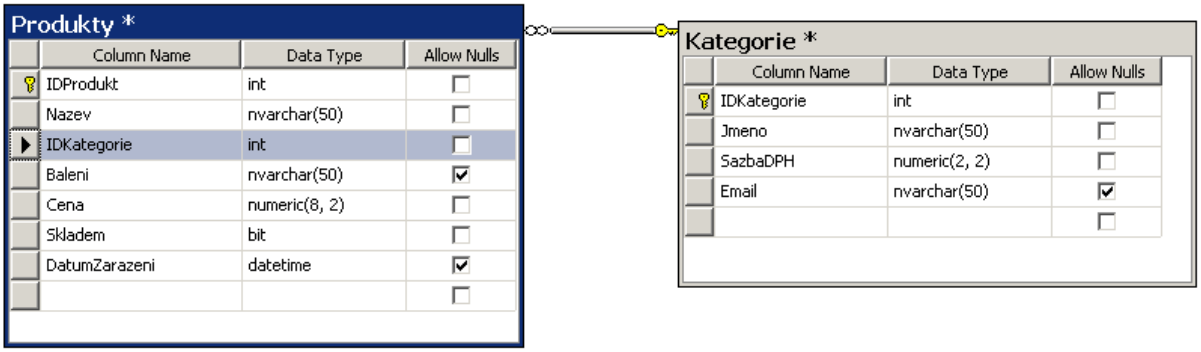
Úplný výčet možných typů polí a jejich přesnější popis najdete v SQL Server 2005 Books Online[23]. Třetí sloupeček pak vyjadřuje, zda musí být datové pole vyplněno (*Allow Nulls* není zaškrtnuté) anebo zda může být prázdné.

Pro většinu tabulek je doporučené a vhodné, aby měl každý řádek zvolený jednoznačný identifikátor nebo identifikátory, takzvaný primární klíč. V našem případě zvolíme primární klíč snadno. Zobrazíme si kontextovou nabídku na definici pole *IDKategorie* a z nabídky vybereme *Set Primary Key*. Změna se projeví symbolem klíče na příslušném řádku jak je to vidět na předchozím obrázku.

Nyní můžeme změny navržené v diagramu uložit, tedy promítnout do databáze. K tomu slouží možnost *Save*. Po jeho stisknutí budeme dotázáni na název diagramu, kam můžeme zadat celkem libovolný text, např. *Diagram Katalogu*

Založení druhé tabulky a vytvoření relace

Velmi podobným způsobem vytvoříme i druhou tabulku pro jednotlivé produkty. Zobrazte si nabídku kdekoliv na prázdné ploše diagramu a zvolte *New Table...* jako název tabulky zvolíme *Produkty*.
V tabulce založte celkem 7 datových polí různých typů podle následujícího obrázku:



Jako primární klíč zvolte sloupec *IDProdukt* – stejným způsobem jako při vytváření klíče v první tabulce. Nyní si ukážeme ještě několik pokročilejších věcí. Pokud vybereme některé datové pole, v okně *Properties* se nám zobrazí řada dalších vlastností, které můžeme nastavit. Pokud se vám okno nezobrazuje, zvolte z nabídky *View/Properties*. Například můžeme nastavit automatické přiřazování nového čísla výrobku – u sloupečku *IDProduct* nastavíme vlastnost *Identity/(Is Identity)* na *Yes* (proveďte). Rovněž můžeme nastavit výchozí hodnotu některého pole, která se použije, není-li pole vyplněno a řadu dalších věcí.

23 http://www.microsoft.com/technet/prodtechnol/sql/2005/downloads/books.mspx

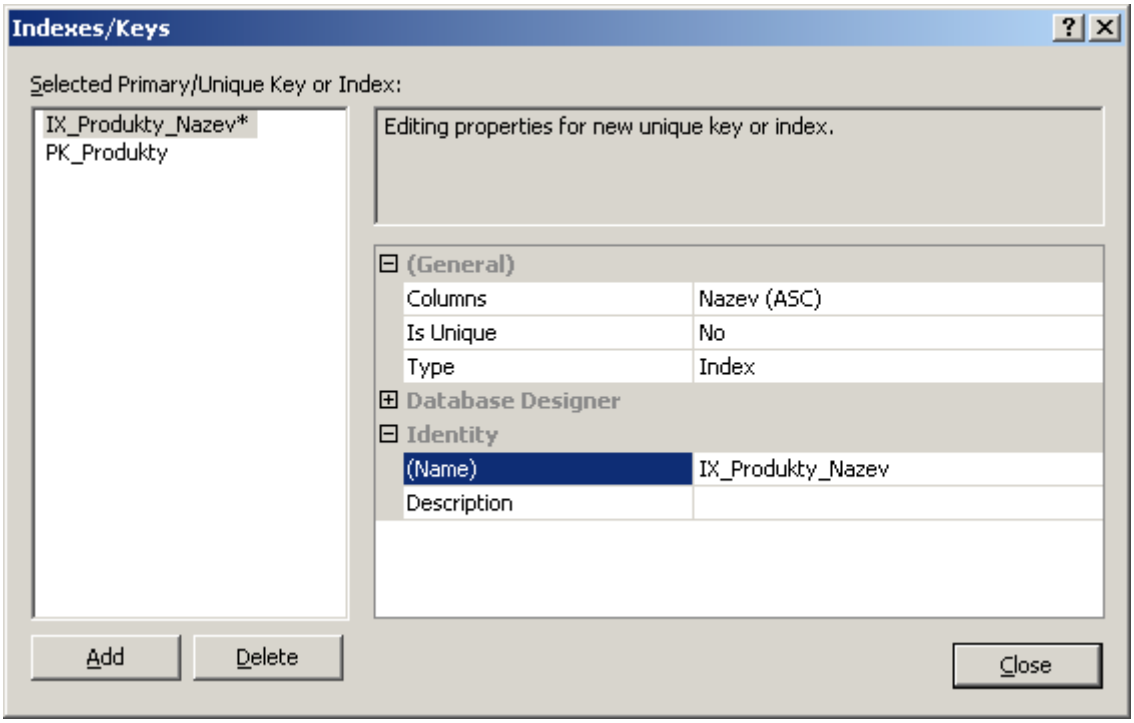
Nyní vytvoříme vztah či relaci mezi tabulkami. Tato relace nám zaručuje, že v tabulce *Produkty* bude ve sloupečku *IDKategorie* vždy správná hodnota – tj. číslo existující kategorie v tabulce *Kategorie*. Jakékoliv změny dat, které by vedly k narušení tohoto stavu nebudou připuštěny – tato vlastnost se odborně nazývá referenční integrita. Chytneme myší šedý obdélníček před sloupcem *IDKategorie* v tabulce *Produkty* a přetáhneme jej na sloupeček *IDKategorie* v tabulce *Kategorie*. Po dvojím stisknutí *OK* bychom měli vidět vztah vidět vyznačený obdobně jako na předchozím obrázku.

Nyní změny uložíme pomocí *Save* a potvrzením *Yes*, čímž se nová tabulka i vztah mezi tabulkami vytvoří v databázi.

Indexovaní

Pokud budeme v tabulce ukládat větší množství dat, je vhodné si vytvořit takzvané indexy, které urychlují vyhledávání nebo třídění dat. Indexy fungují podobně jako papírové kartotéky u lékaře. Pokud přijdete a řeknete své příjmení, vaše lékařská dokumentace je snadno nalezena v setříděné řadě – takto funguje index. Pokud byste svoje příjmení neznali, ale znali rodné číslo, musela by sestra procházet postupně všechny lékařské karty a hledat shodu, což by bylo mnohem pomalejší – takto funguje hledání bez indexu. Na rozdíl od lékařské dokumentace ale v tabulce můžete mít i více než jeden index.

Spíše z cvičných důvodů vytvoříme index nad tabulkou *Produkty*, který bude setříděn podle názvu produktu (sloupec *Nazev*). V diagramu klikneme kdekoliv na tabulce *Produkty* pravým tlačítkem myši a zvolíme *Indexes/Keys...*, kde již uvidíme index *PK_Produkty*, což je index založený automaticky při vytvoření primárního klíče. Stiskneme *Add* pro nový index, jako sloupce (*Columns*) vybereme sloupec *Nazev* a jako jméno zadáme například *IX_Produkty_Nazev* podle následujícího obrázku:



Zavřeme pomocí *Close*, změny promítneme do databáze volbou *Save* a potvrdíme změny v tabulkách (*Yes*). Je ovšem nutné říci, že vytvořit indexy optimálně vyžaduje určitou praxi, náš případ je příliš zjednodušený. Příliš málo indexů může znamenat pomalé vykonávání dotazů, příliš mnoho indexů naopak zdržuje při vkládání nebo úpravě dat. Pro větší databázi si na vytváření indexů raději přizveme odborníka.

Vkládání a úpravy dat

Vkládání a úpravy dat v tabulkách jsou poměrně jednoduché. V okně *Database Explorer* klikneme pravým tlačítkem myši na vybranou tabulku a vybereme možnost *Show Table Data*. Zřejmým způsobem zadáme tři kategorie zboží podle následujícího obrázku:

	IDKategorie	Jmeno	SazbaDPH	Email
	1	Potraviny	0.05	karel@firma.cz
	2	Nápoje	0.19	josef@firma.cz
▶	3	Drogerie	0.19	marie@firma.cz
*	NULL	NULL	NULL	NULL

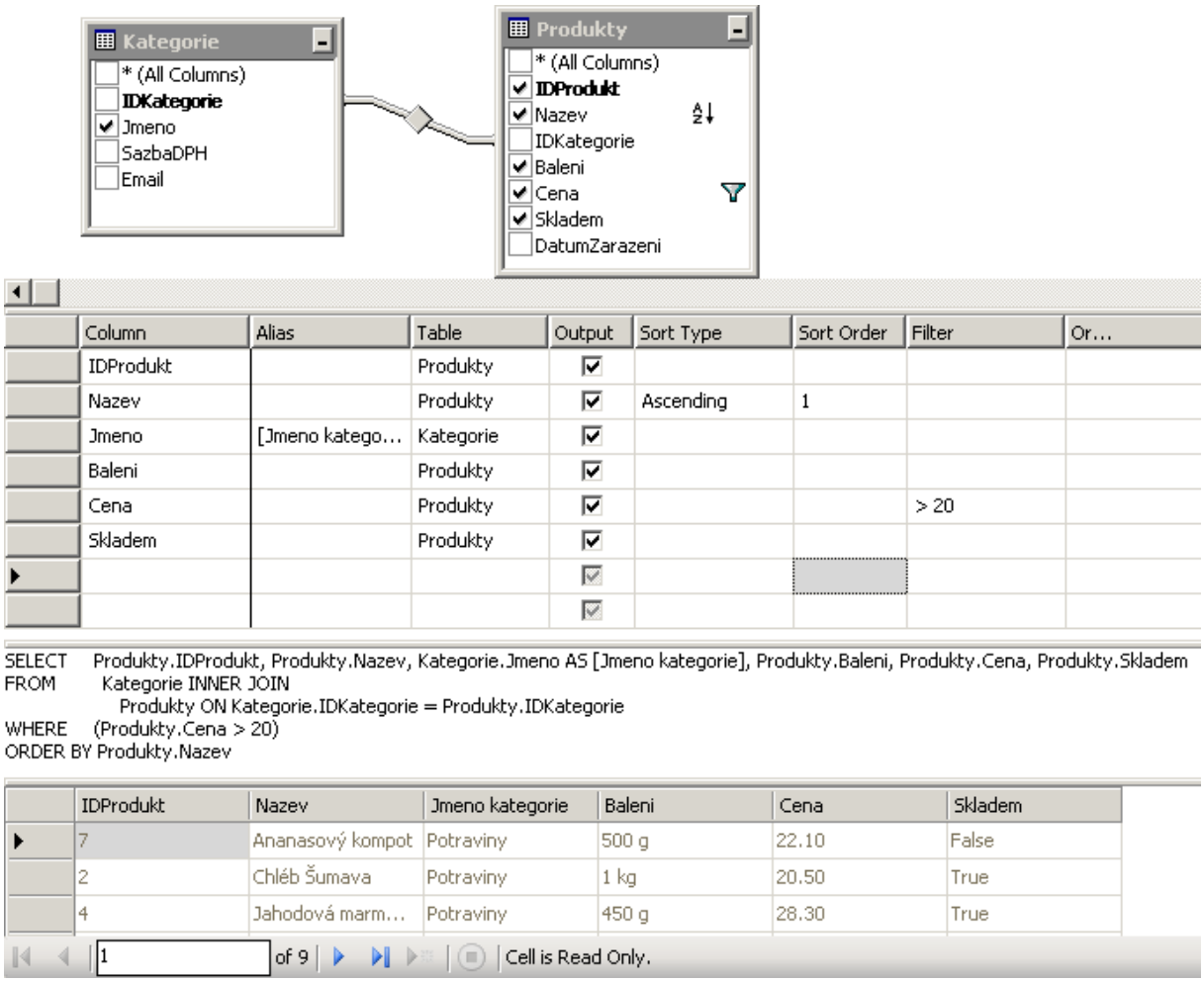
Můžeme si vyzkoušet, že databáze nám nepovolí neplatné údaje – pokud některý z údajů vynecháme anebo třeba do sloupce *SazbaDPH* vepíšeme údaj, který není číslem, řádek se nám nepodaří do databáze uložit. Tabulku nyní zavřeme a stejným způsobem vepíšeme několik údajů do tabulky *Produkty*. Doplníme několik produktů do každé kategorie označené 1 až 3 podle vlastní fantazie, podobně jako na následujícím obrázku:

	IDProdukt	Nazev	IDKategorie	Baleni	Cena	Skladem	DatumZarazeni
	2	Chléb Šumava	1	1 kg	20.50	True	10/10/2005 12:0...
	3	Mléko plnotučné	1	1 l	16.50	False	10/10/2005 12:0...
	4	Jahodová marmeláda	1	450 g	28.30	True	10/10/2005 12:0...
	5	Rohlík	1	NULL	1.50	True	10/10/2005 12:0...
	7	Ananasový kompot	1	500 g	22.10	False	10/10/2005 12:0...
	8	Pivo	2	500 ml	12.40	True	10/10/2005 12:0...
	10	Rum	2	500 ml	92.00	True	10/2/2005 12:00...
	11	Víno červené	2	1 l	68.00	False	5/5/2005 12:00:...
	12	Víno bílé	2	0.7 l	64.00	True	6/8/2005 12:00:...
	14	Tekuté mýdlo	3	250 ml	39.00	False	10/10/2005 12:0...
	15	Zubní pasta	3	100 g	24.00	True	3/9/2005 12:00:...
	16	Kartáček na ruce	3	NULL	27.50	True	8/7/2005 12:00:...
▶	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Všimněme si, že do sloupečku *IDProdukt* nemusíme zadávat žádnou hodnotu, neboť tato se doplňuje automaticky. Podobně do sloupečku *Baleni* není nutné zadávat hodnotu, neboť při vytváření tabulky bylo pro tento sloupeček povoleno *Allows Nulls*. A naopak, pokud ve sloupečku *IDKategorie* zadáme hodnotu např. 8, nepodaří se řádek uložit, neboť kategorie s tímto číslem neexistuje.

Vykonání dotazu

Pokud máme v tabulkách vložená data, můžeme ve VWD pokládat též dotazy do dat, abychom zjistili aktuální stav databáze anebo pak dotazy použili v našich aplikacích. Zobrazíme-li si kontextovou nabídku prakticky kdekoliv v okně *Database Explorer* je jednou z možností *New Query* – nový dotaz do databáze. V nabídce tabulek vybereme dvojklikem myši obě vytvořené tabulky (*Kategorie* a *Produkty*). Zatržením políček před názvem sloupce se sloupec objeví ve druhé sekci, kde můžeme měnit jeho pojmenování, nastavovat třídění, podmínky pro filtrování výběru apod. Ve třetí sekci se objevuje výběrový dotaz v jazyce T-SQL (pouze pro pokročilejší) a ve čtvrté sekci si můžeme nechat zobrazit výsledky dotazu volbou *Execute SQL* (tlačítko s červeným vykřičníkem).



Na předchozím obrázku jsou například zobrazeny produkty s cenou vyšší než 20 setříděné podle názvu produktu a s doplněným jménem kategorie přejmenované na *Jmeno kategorie*. Pokročilejší uživatelé mohou vytvářet dotazy se seskupováním a agregací dat, dotazy na mazání, aktualizaci či vkládání dat apod.

Zálohovaní

Posledním úkonem, o kterém se zmíníme, je zálohování dat. Zálohování pojmenovaných databází se provádí bez jakéhokoliv přerušení jejich provozu T-SQL příkazem *BACKUP DATABASE*, například z nástroje *sqlcmd.exe* anebo nepřímo prostřednictvím nástroje *SQL Management Studio*. My ale používáme nepojmenovanou databázi, která je pouhým souborem (přesněji dvěma soubory) v adresáři aplikace *App_Data* – v našem případě se jedná o soubory *KatalogZbozi.mdf* a *KatalogZbozi_log.ldf*. Zálohování tedy provedeme prostým zkopírováním těchto souborů na bezpečné místo.

Jediným úskalím je možnost, že některá aplikace databázi používá a má tudíž otevřené její soubory – za této situace není možné soubory kopírovat. Chceme-li databázi kopírovat, musíme uzavřít všechny aplikace, které databázi používají, případně je donutit, aby přestali databázi používat. V praxi to může znamenat:

- Máme-li databázi otevřenou ve VWD v okně *Database Explorer*, použít příkaz *Detach Database*, případně zavřít všechna okna s databází pracující, případně ukončit práci s VWD.
- Dočasně zastavit webový server – *ASP.NET Development Server* anebo příslušný web IIS
- Dočasně zastavit službu *SQL Server Express* – např. použitím standardního nástroje Windows jménem *Services/Služby* v nabídce *Administrative Tools/Nástroje pro správu* – toto je nejradikálnější a zcela spolehlivý krok.

Závěrem

V této lekci jsme se naučili základům práce s databází SQL Express. Odbočka od webového vývoje nebyla samoúčelná. Vytvořenou databázi *KatalogZbozi* budeme používat v příštích dvou lekcích zabývajících se prací s daty na ASP.NET stránkách. Komu dnešní látka nestačila chtěl by se dozvědět něco více, může se podívat na článek SQL Server 2005 Express Edition Overview[24].

Opakování je matka moudrosti – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

24 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/sseoverview.asp

6. PRÁCE S DATY V ASP.NET I.

Posledně jsme vytvořili databázi *KatalogZbozi* za pomoci SQL Serveru 2005 Express. Vytvořená data dnes použijeme na vytvářených webových stránkách pomocí tzv. data-bindingu. Mimo jiné si ukážeme, jak lze daty naplnit rozbalovací seznam, jak lze data zobrazit v datové mřížce anebo editovat ve formuláři.

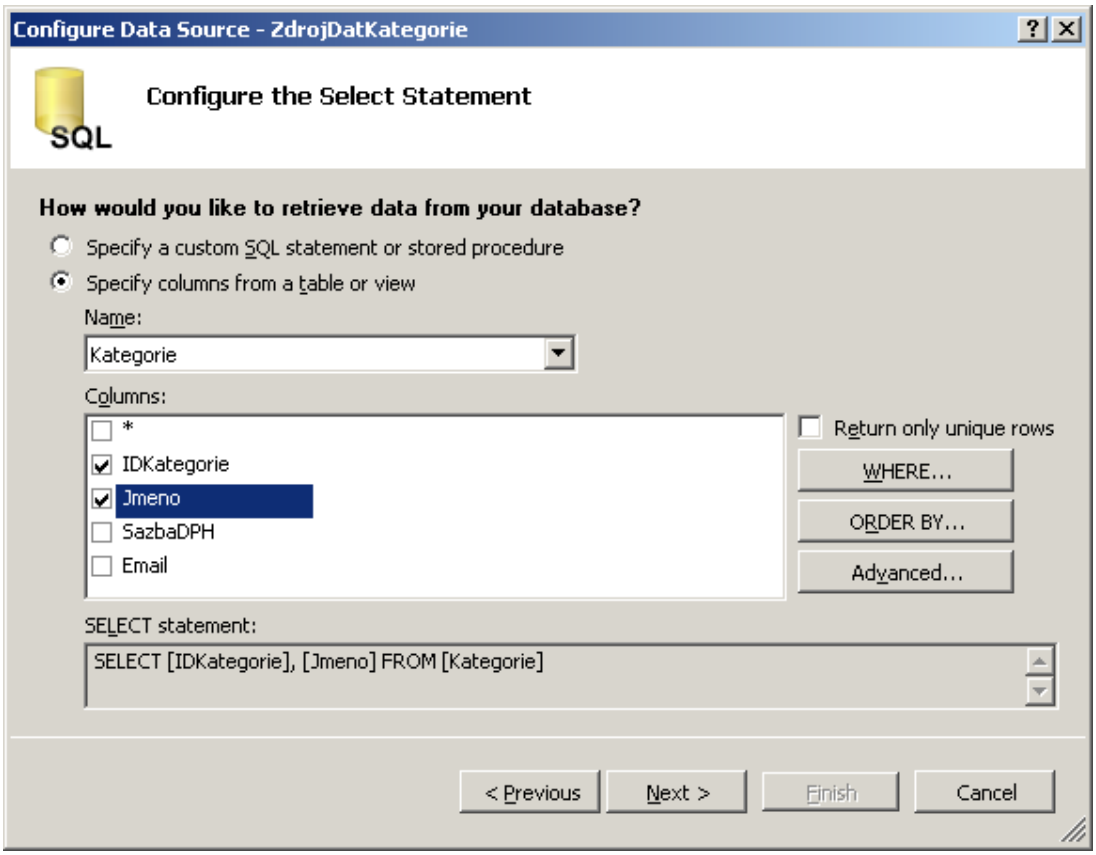
Pojem data-binding

Data-binding je princip, který umožňuje spojovat poskytovatele dat (typicky databázi) s konzumenty dat (typicky ovládacím prvky umístěné na formuláři). Důležité je, že při data-bindingu není nutné psát žádný kód. Data-binding je záležitost nastavení datového zdroje a konzumenta dat a typicky se provádí prostřednictvím pohodlného vizuálního rozhraní (odborným názvem – deklarativně).

Vytvoření zdroje dat

Zdrojem dat bude v prvním jednoduchém případě tabulka *Kategorie* z naší databáze *KatalogZbozi*. V ASP.NET je několik druhů zdrojů dat (*Data Source*), které si můžete prohlédnout v panelu nástrojů. V dnešní lekci budeme používat datový zdroj *SqlDataSource*, který lze – navzdory poněkud zavádějícímu názvu – využít nejenom pro přístup k datům SQL serveru, ale i k dalším relačním datovým zdrojům přístupným přes OLEDB/ODBC rozhraní.

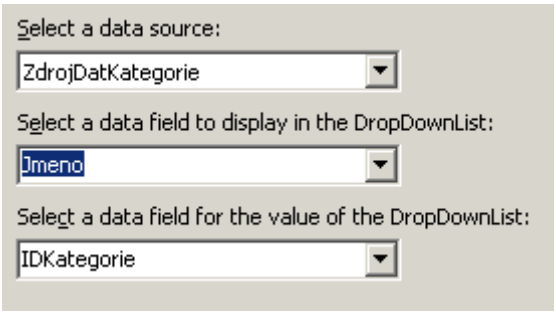
Pojďme na datový zdroj. Vytvoříme novou stránku a nazveme ji *ProchazeniKatalogu.aspx*. Z panelu nástrojů na ni myší přetáhneme prvek *SqlDataSource*. Jedná se o ovládací prvek jako každý jiný, takže jej můžeme přejmenovat – navolíme jej myší a v okně *Properties* změníme *(ID)* z původního *SqlDataSource1* na *ZdrojDatKategorie*. Jsme-li v pohledu *Design* a navolíme datový zdroj myší, zobrazí se malá šipka, pomocí které rozbalíme smart tag s nabídkou *Configure Data Source* (stejného efektu lze dosáhnout zobrazením kontextové nabídky na tomto prvku). Nejprve musíme vybrat datové spojení – v rozbalovacím seznamu bychom měli najít naši databázi *KatalogZbozi.mdf*. Na další stránce potvrdíme uložení připojovacího řetězce k databázi pro další použití – můžeme použít nějaké popisnější jméno, např. *SpojeniKatalogZbozi* a pokračujeme dále. Třetí stránka je nejdůležitější (viz obrázek).



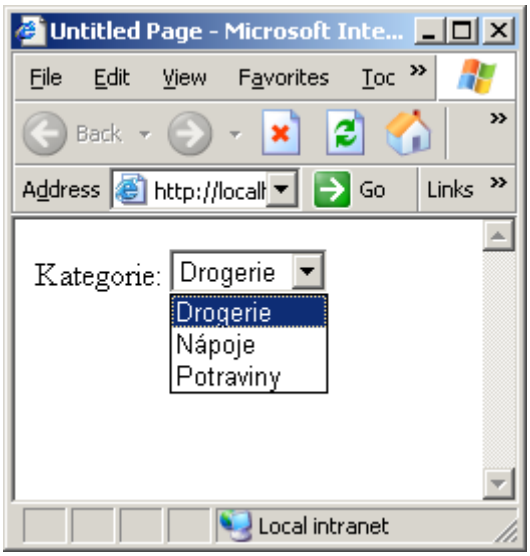
Zkušenější uživatelé mohou napsat vlastní SQL dotaz do databáze, my si vybereme pohodlnější možnost – zvolíme tabulku *Kategorie* a sloupce *IDKategorie* a *Jmeno* a pokračujeme dále (volitelně můžete též nastavit třídění pomocí tlačítka *ORDER BY*). Na následující stránce si můžete zkontrolovat výsledek dotazu (*Test Query*) a po ukončení průvodce máme datový zdroj hotový.

Nastavení konzumenta dat

Datový zdroj máme hotový, zbývá vytvořit alespoň jednoho konzumenta. V našem případě to bude rozbalovací seznam. Napíšeme na stránku text *Kategorie:* a za tento text přetáhneme z panelu nástrojů prvek *DropDownList*. V okně *Properties* jej přejmenujeme změnou vlastnosti *(ID)* z *DropDownList1* na *DDLKategorie*. Navolením prvku se zobrazí šipka zobrazující smart tag, v němž je i možnost *Choose Data Source...*, jednotlivá pole nastavíme podle následujícího obrázku:



A to je vše, co jsme museli udělat. Zobrazíme-li si nyní stránku v prohlížeči, měl by být rozbalovací seznam naplněn třemi kategoriemi jako na následujícím obrázku.



GridView

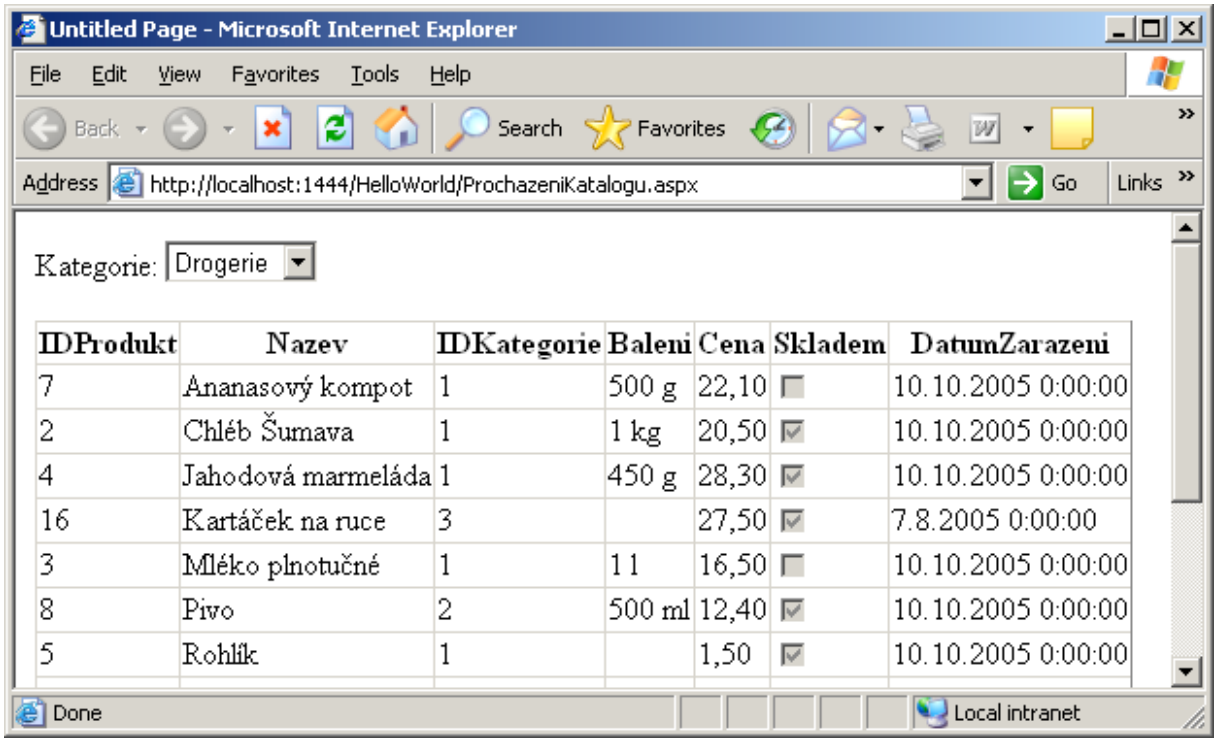
Ovládacích prvků, které lze použít pro data-binding, je celá řada. Nejpokročilejším z nich je *GridView* umožňující zobrazovat data v datové mřížce s možností výběru, třídění, mazání, upravování, stránkování apod.

Základní použití

Na stejné stránce *ProchazeniKatalogu.aspx* si vytvoříme druhý datový zdroj. Použijeme obdobný postup jako v předchozím případě. Z panelu nástrojů na ni myší přetáhneme druhý prvek *SqlDataSource*, navolíme jej myší a v okně *Properties* změníme *(ID)* z původního *SqlDataSource1* na *ZdrojDatProdukty*. Opět přepneme do pohledu *Design*, zobrazíme smart tag a vybereme *Configure Data Source*. Při

výběru datového spojení najdeme v seznamu již připravené spojení *SpojeniKatalogZbozi*. Jako tabulku vybereme tentokrát *Produkty* a pro urychlení můžeme vybrat všechna pole (*). Pomocí tlačítka *ORDER BY* nastavíme pole pro třídění na *Nazev* a dokončíme průvodce.

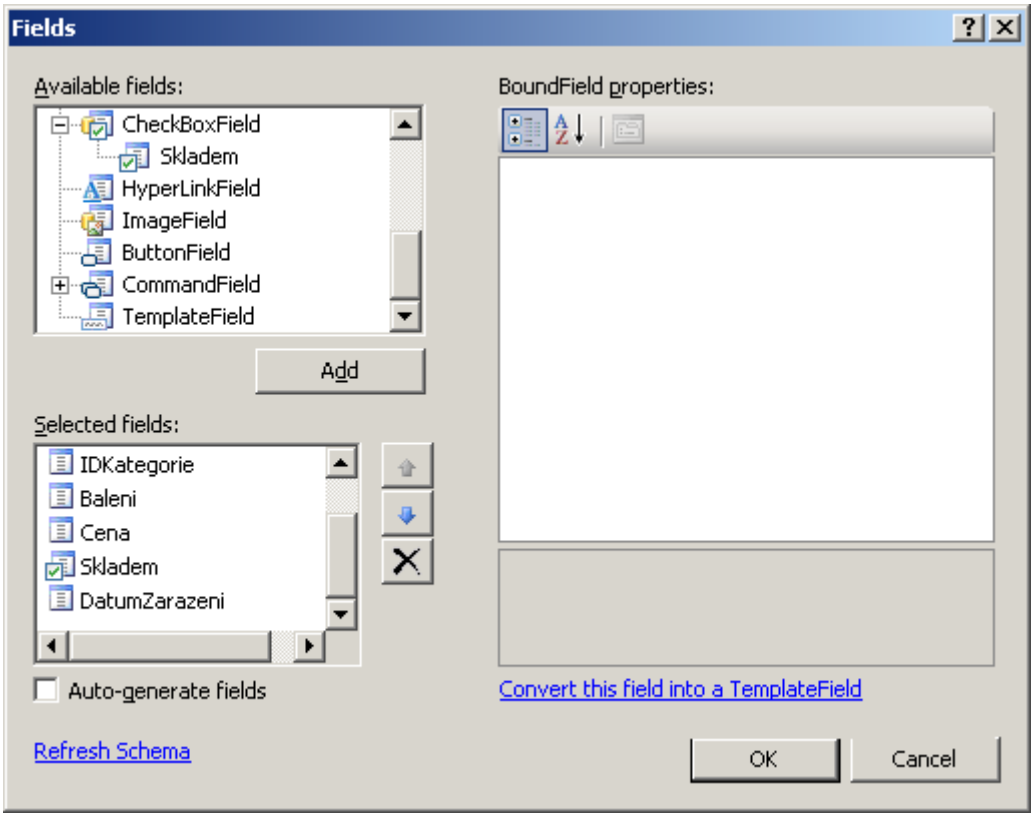
Jak asi tušíte, přidáme nyní konzumenta dat v podobě prvku *GridView*. V kontextové nabídce uvidíme možnost *Choose Data Source*, kde ze seznamu vyberte *ZdrojDatProdukty*. V okně *Properties* přejmenujeme naši datovou mřížku změnou vlastnosti *(ID)* z *GridView1* na *GVProdukty*. Pokud si zobrazíte stránku v prohlížeči, měla by nyní vypadat zhruba takto:



Zapnutí pokročilejších vlastností

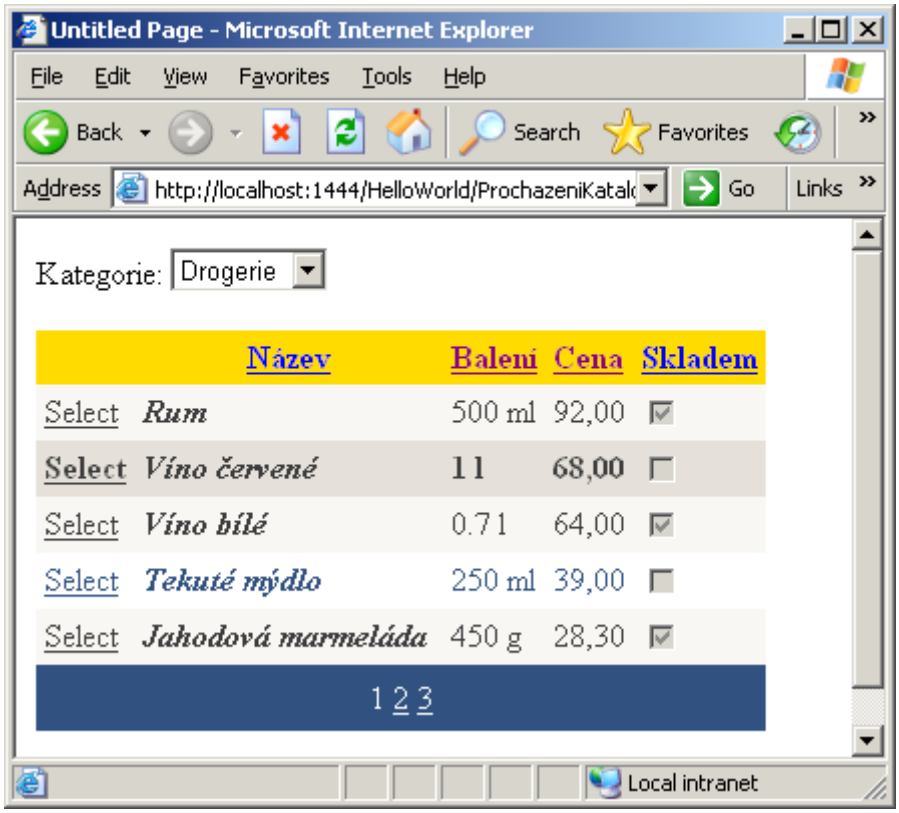
Naše datová mřížka je sice funkční, ale dá se na ní ještě spousta věcí vylepšit – začneme vzhledem. V kontextové nabídce prvku je možnost *Auto Format*, ve které můžete zvolit z mnoha přednastavených barevných kombinací. Každému, co jeho jest – já jsem zvolil formát *Professional*. Vizuální nastavení lze měnit i jednotlivě – v okně *Properties* je sekce *Styles*, kde je nepřeberné množství formátovacích možností – např. můžeme změnit barvu pozadí nadpisů sloupců (položka *BackColor* v podsekci *HeaderStyle*) na zlatou (*Gold*).

Další zajímavá nastavení se týkají sloupců. Některé možnosti jsou přístupné přímo z kontextové nabídky, všechny možnosti si pak zobrazíme kliknutím na symbol ... v položce *Columns* v okně *Properties* – viz obrázek:



Začneme tím, že odstraníme nepotřebná pole *IDProdukt*, *IDKategorie* a *DatumZarazeni*. Stačí příslušná pole vybrat v seznamu *Selected fields* a použít tlačítko se symbolem X pro mazání. Můžeme též měnit formátování jednotlivých sloupců. Chceme-li např. mít název vypsaný tučnou kurzívou, vybereme pole *Nazev* v seznamu *Selected fields* a ve vlastnostech vpravo změníme v položce *Styles/Item Style/Font* hodnoty *Bold* a *Italic* na *true*. Názvy sloupečků můžeme počeštit jejich výběrem v seznamu *Selected fields* a opravením vlastnosti *HeaderText* v sekci *Appearance*. V dialogu lze rovněž vytvářet nová pole v datové mřížce, ale to zatím dělat nebudeme.

Posledním úkonem bude povolení třídění, stránkování a výběru konkrétního řádku. Zvolíme-li datovou mřížku a rozbalíme smart tag nahoře vpravo, stačí zaškrtnout možnosti *Enable paging*, *Enable sorting* a *Enable selection* (alternativně lze nastavit i v okně *Properties*). V okně *Properties* ještě zmenšíme velikost stránky na 5 řádků na stránku pomocí volby *PageSize* v sekci *Paging*. Pokud si nyní stránku zobrazíme, měla by vypadat zhruba takto:



Všimněme si, že vzhled je přesně podle našeho nastavení – aplikoval se auto formát, názvy jsou tučnou italikou, pozadí hlaviček sloupců má zlatavou barvu, zobrazeny jsou pouze relevantní sloupečky. Kliknutím na titulku sloupečku lze data řadit podle obsahu sloupce vzestupně i sestupně. V dolním řádku jsou odkazy na jednotlivé stránky mřížky. Kliknutím na odkaz *Select* v úvodu každého řádku lze daný řádek vybrat/zvýraznit (text *Select* lze změnit v dialogu pro editaci sloupců, na výběr řádku se lze navázat událostí *SelectedIndexChanged* anebo jej použít pro data binding dalšího prvku).

Pokud se chcete seznámit s dalšími možnostmi prvku *GridView*, doporučujeme pro další studium následující dva odkazy:

http://msdn.microsoft.com/msdnmag/issues/04/08/GridView/default.aspx

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnaspp/html/GridViewEx.asp

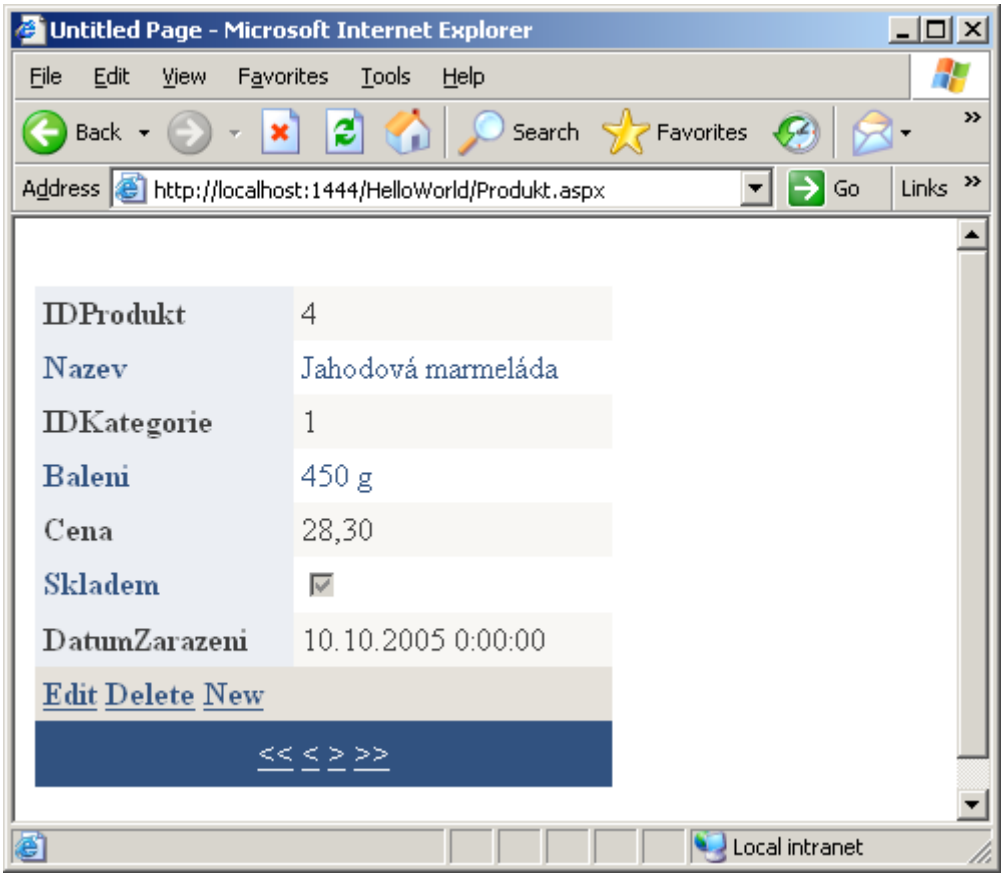
DetailsView

Druhým důležitým prvkem je *DetailsView*. Mnohé jeho rysy jsou shodné s prvkem *GridView*, od něhož se však liší v jedné důležité věci – zatímco *GridView* zobrazuje více datových řádků najednou, *DetailsView* zobrazuje právě jeden řádek. Velmi často se proto používá k editaci dat (k čemuž lze ale použít i *GridView*) anebo ke vkládání dat. Podíváme se nyní na *DetailsView* prakticky.

Založíme novou stránku *Produkt.aspx*. Vytvoříme na ní nový zdroj dat *ZdrojDatProdukty* – použijeme velmi podobný postup jako při práci s datovou mřížkou. Z panelu nástrojů na plochu přetáhneme prvek *SqlDataSource*, navolíme jej myší a změníme *(ID)* z původního *SqlDataSource1* na *ZdrojDatProdukty*. Opět přepneme do pohledu *Design*, zobrazíme nabídku a vybereme *Configure Data Source*. Při výběru datového spojení najdeme v seznamu již připravené spojení *SpojeniKatalogZbozi*. Jako tabulku vybereme tentokrát *Produkty* a pro urychlení můžeme vybrat všechna pole (*). Pomocí tlačítka *ORDER BY* nastavíme pole pro třídění na *Nazev* a dokončíme průvodce. A nyní rozdíl oproti předchozímu postupu – stiskneme tlačítko *Advanced...* a vybereme možnost *Generate INSERT, UPDATE, and DELETE statements*.

Na stránku přeneseme prvek *DetailsView* a přejmenujeme ho změnou vlastnosti *(ID)* z původního *DetailsView1* na *DVProdukty*. Zobrazíme jeho smart tag a v rozbalovacím seznamu *Choose Data*

Source zvolíme *ZdrojDatProdukty*. Rovněž zde povolíme stránkování, vkládání, úpravy a mazání dat zatržením *Enable Paging, Inserting, Deleting, Updating*. Stejně jako v předchozím případě můžeme použít *Auto Format* pro rychlou a snadnou úpravu vzhledu. Místo číselného stránkování můžeme zvolit symboly šipek – ve vlastnostech nastavíme *Paging/PagerSettings/Mode* například na hodnotu *NextPreviousFirstLast*. Zobrazená stránka může vypadat například takto:



Můžeme si vyzkoušet funkci stránkování, editace, vkládání i mazání záznamů. Podobně jako v případě prvku *GridView* můžeme měnit popisy hlaviček, formáty apod.

Další prvky

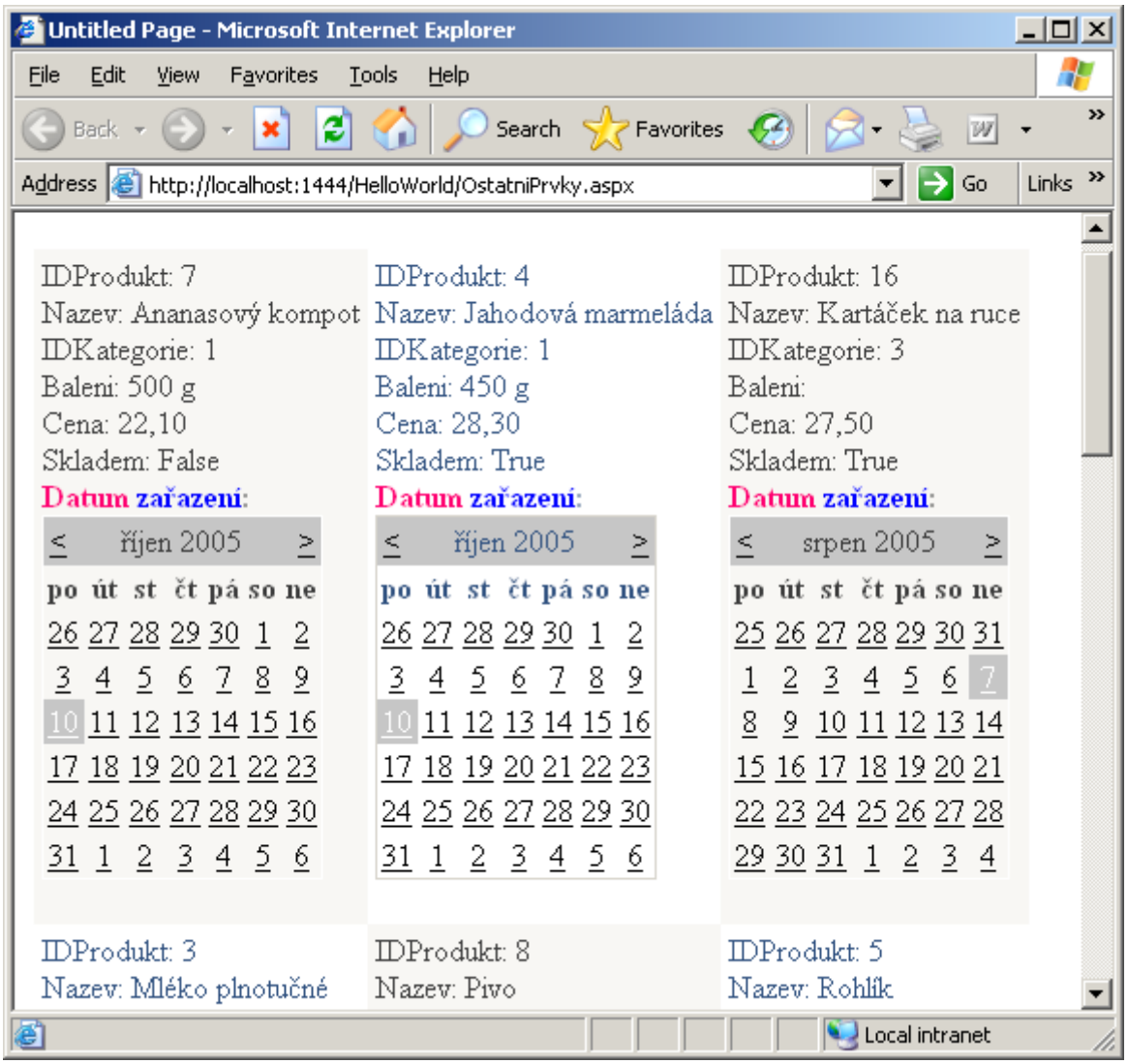
K zobrazování dat lze použít i další prvky – *DataList*, *Repeater* a *ListView*. Na rozdíl od předchozích prvků jsou mnohem flexibilnější a lze s nimi dosáhnout mnohem více, ovšem na úkor poněkud větší složitosti. Pro experimentování s těmito prvky si založíme novou stránku *OstatniPrvky.aspx* a na ní vytvoříme a nakonfigurujeme zdroj dat *ZdrojDatProdukty* přesně stejným postupem jako u prvku *DetailsView*.

DataList

DataList je podobný prvku *GridView*, ale na rozdíl od něj vykresluje pouze pole tabulky, jejichž vnitřek se vykresluje tzv. šablonami. Šablona je při navázání na zdroj dat vygenerována ze zdroje dat a poté si ji můžeme sami upravovat. Přetáhneme na plochu prvek *DataList* a jako jeho zdroj dat vybereme *ZdrojDatProdukty*. Můžeme rovněž zkusit *Auto Format*, ale jeho možnosti jsou zde omezené. Na rozdíl od prvku *GridView* může *DataList* mít více položek v jednom řádku. V okně *Properties* nastavíme v sekci *Layout* vlastnost *RepeatColumns* na 3 a vlastnost *RepeatDirection* na *Horizontal*.

Podíváme se nyní na šablony. Ze smart tagu zvolíme možnost *Edit Templates*. Šablona je jakousi mini-stránkou – je na ní text a různé ovládací prvky, které můžeme libovolně měnit. Například můžeme opravit titulek *DatumZarazeni*, vymyslet mu divoké formátování apod. Výpis data zařazení můžeme nahradit kalendářem. Smažeme prvek *DatumZarazeniLabel* a na jeho místo přetáhneme z panelu

nástrojů prvek *Calendar*, který musíme navázat na data. Zobrazíme jeho smart tag a zvolíme *Edit Databindings*. V zobrazeném dialogu vybereme vlastnost *SelectedDate* a navážeme ji (*Bound to:*) na pole *DatumZarazeni*, totéž provedeme s vlastností *VisibleDate*. Na kalendář můžeme též použít *Auto Format*. Editování šablony ukončíme ve smart tagu pomocí volby *End Template Editing*. Podobně jako jsme editovali šablonu pro položku, můžeme editovat i šablonu pro záhlaví, zápatí anebo oddělovač položek. Stránka bude po zobrazení vypadat zhruba takto:



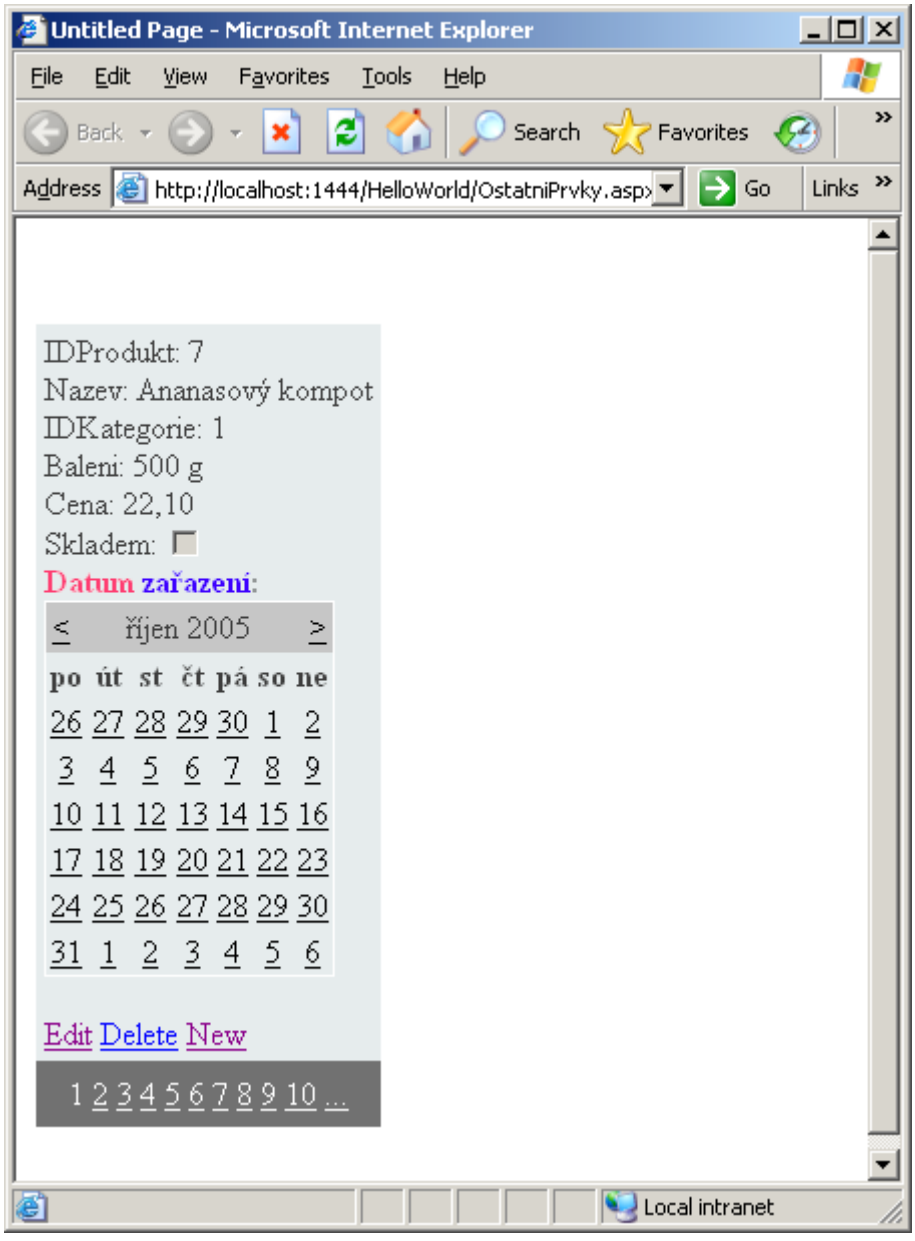
Jistě, výsledek oku příliš nelahodí, ale jako ukázka flexibility prvku *DataList* je snad dostatečný.

FormView

Prvek *FormView* má podobný vztah k prvku *DetailsView*, jako má prvek *DataList* k prvku *GridView* – viz tabulka:

	Současná práce s jedním záznamem	**Současná práce s více záznamy**
Velké množství přednastavené automatické funkčnosti	DetailsView	GridView
Větší flexibilita na úkor menší funkčnosti a větší pracnosti	FormView	DataList

FormView je prvku *DataList* v mnohém podobný, ale slouží pro práci pouze s jedním datovým záznamem. Je rovněž flexibilnější a lze s ním dosáhnout mnohem více. Ze stránky *OstatniPrvky.aspx* můžeme smazat *DataList* a místo něj přetáhnout na stránku *FormView*, jako jeho zdroj dat vybereme opět *ZdrojDatProdukty* a povolíme stránkování pomocí *Enable Paging*. Šablony se editují obdobně jako u prvku *ListView*, ale jejich typy jsou jiné. *FormView* obsahuje šablony pro hlavičku, patičku, stránkovací oblast, zobrazení položky, editaci položky, vkládání položky a pro prázdný datový zdroj. Jejich úpravami lze dosáhnout opravdu neomezených možností. Pokud provedeme obdobné úpravy jako u prvku *ListView* můžeme dosáhnout následujícího vzhledu stránky:



Repeater

Prvek *Repeater* má ze všech datových prvků nejmenší funkčnost. Nabízí vlastně jedinou funkci – opakování fragmentů stránky pro každý řádek dat. Prvek *Repeater* nelze upravovat v režimu *Design* a tento prvek se nejvíce blíží tradičnímu způsobu vytváření webových stránek pomocí HTML značek protkaných kousky kódu, proto se mu dále nebudeme věnovat.

Závěrem

V dnešní lekci jsme se naučili základním způsobům práce s daty na webových stránkách – jejich zobrazování a editaci. V další se podíváme na některé pokročilejší postupy, zejména vytváření master-detail zobrazení (dokončíme stránku *ProchazeniKatalogu.aspx*), použití vazeb mezi stránkami s daty a práci s hierarchickými ovládacími prvky a daty.

Tradiční připomínka – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

7. PRÁCE S DATY V ASP.NET II.

V sedmé lekci budeme pokračovat v práci s daty na webových stránkách. Vyzkoušíme si některé pokročilejší postupy, a to master-detail zobrazení, omezení zobrazovaných dat s použitím vazby mezi stránkami a práci s hierarchickými ovládacími prvky a daty.

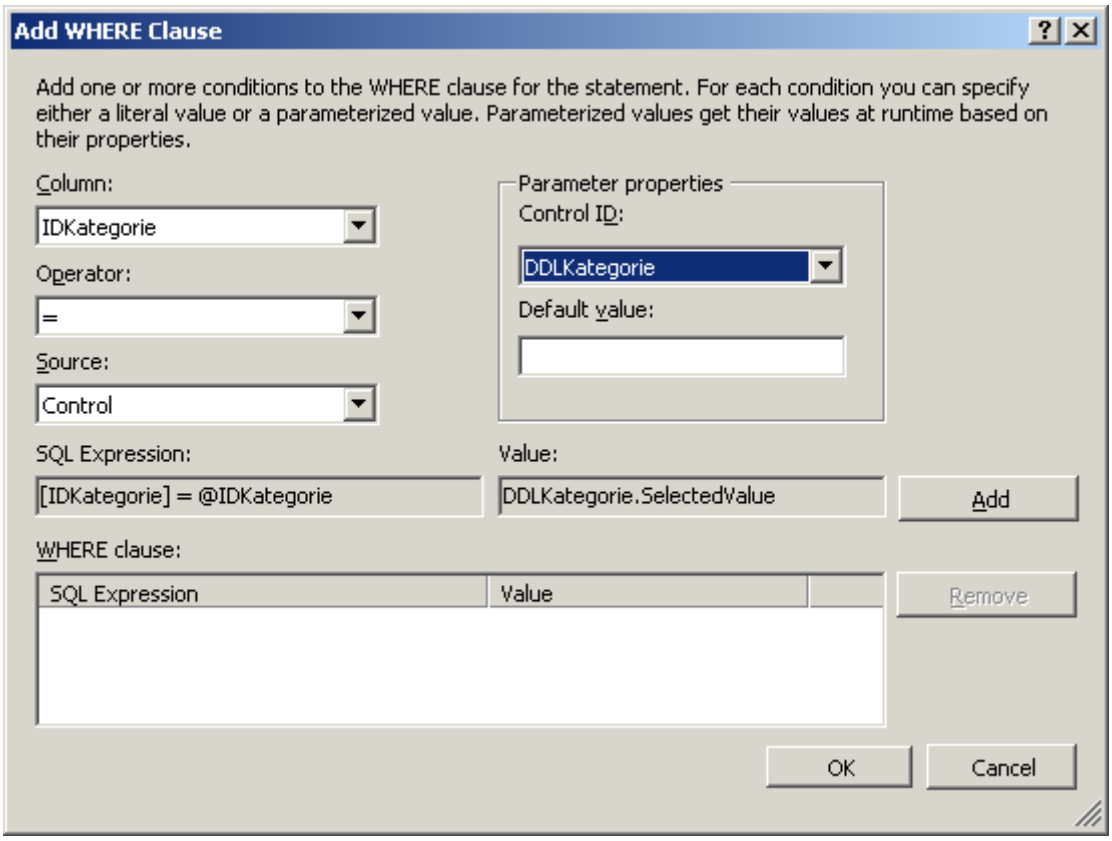
Master-detail zobrazení na stránce

Master-detail je zaběhnutý termín pro princip běžně používaný při vytváření uživatelského rozhraní. Jeden ovládací prvek se nazývá master a slouží k výběru určité hodnoty. Na základě tohoto výběru se potom v jiném prvku (zvaném detail) zobrazí hodnoty vztahující se k vybrané položce. V našem příkladě provedeme úpravu stránky *ProchazeniKatalogu.aspx* tak, aby se při výběru kategorie v rozbalovacím seznamu (master), objevily v datové mřížce (detail) pouze produkty, které do této kategorie patří.

Stačí jeden úkon...

Vytváření vazeb master-detail je v ASP.NET 2.0 překvapivě jednoduché, postačí nám k tomu jediný úkon. Tímto úkonem je úprava datového zdroje pro prvek detail tak, aby tento prvek nezobrazoval všechny hodnoty, ale pouze hodnoty vyfiltrované na základě výběrové podmínky (příslušnost do určité kategorie).

Provedení je jednoduché. Navolíme myší datový zdroj prvku detail *ZdrojDatProdukty* na stránce *ProchazeniKatalogu.aspx* vytvořené v minulé lekci a zvolíme možnost *Configure Data Source...* První stránce průvodce přejdeme a na druhé stránce specifikujeme výběrovou podmínku stiskem tlačítka *WHERE...* Filtrovaný sloupeček bude *IDKategorie*, operátor bude =, zdroj hodnoty, podle které se filtruje bude jiný ovládací prvek (*Control*), konkrétně rozbalovací seznam jménem *DDLKategorie* – jako na následujícím obrázku:



Nezapomeneme stisknout tlačítko *Add*, tento dialog je trochu matoucí a pouhé stisknutí *OK* podmínku nepřidá! Potvrdíme *OK* a přejdeme na další stránku průvodce, kde si můžeme prohlédnout nově vygenerovaný dotaz a vyzkoušet jeho funkci (zkoušejte hodnotu *IDKategorie* buď 1, 2 nebo 3).

Pokud jste si již nedočkavě stránku vyzkoušeli, budete zklamáni – zobrazuje seznam produktů z první kategorie (v mém případě Drogerie), ale při výběru jiné kategorie se seznam produktů nepřekreslí. Důvod je jednoduchý a pro vysvětlení můžete znovu nahlédnout do čtvrtého lekce. Rozbalovací seznam *DDLKategorie* nemá standardně zapnuté vyvolání akce *PostBack*, tudíž se při výběru hodnoty stránka nepřekresluje a seznam produktů neaktualizuje. Náprava je snadná, rozbalovací seznam navolíme a v okně *Properties* nastavíme jeho vlastnost *AutoPostBack* na *true* (případně zobrazit jeho smart tag a zaškrtnout *Enable AutoPostBack*). Nyní si již můžeme stránku zobrazit v prohlížeči a vyzkoušet její funkci – při výběru kategorie se v mřížce zobrazí pouze produkty v příslušné kategorii.

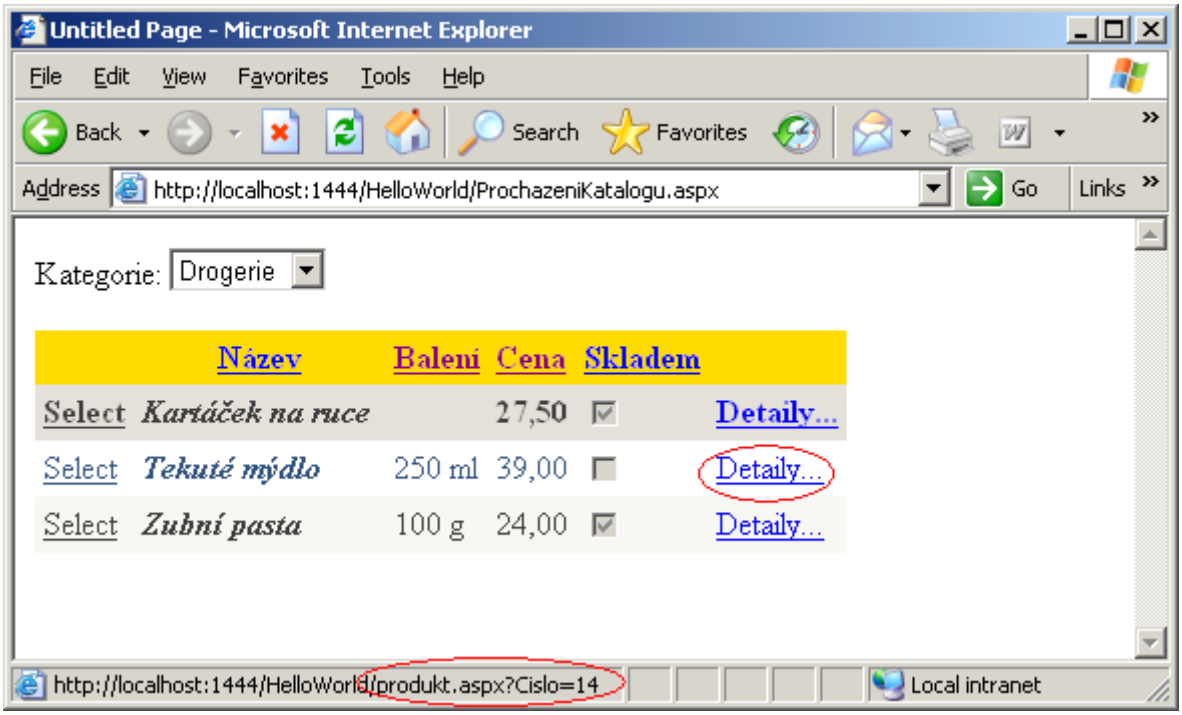
Omezení zobrazovaných záznamů parametrem

V předchozím příkladě jsme provazovali datové zdroje na jedné stránce. Často se setkáváme s požadavkem provázat následující stránky mezi sebou, což bude naším úkolem i v této sekci. Cílem bude upravit stránku *ProchazeniKatalogu.aspx* tak, aby se při kliknutí na produkt zobrazený v datové mřížce zobrazila v novém okně prohlížeče stránka *Produkt.aspx*, ale právě s tímto jediným vybraným produktem. Jediným rozhodnutím je, jak přenést číslo vybraného produktu mezi oběma stránkami. Z možných řešení se nejčastěji používá přenos přímo v hyperlinku pomocí tzv. *query string* řetězce, např. pro produktu číslo 10 pak může být odkaz *Produkt.aspx?Cislo=10*.

Úprava prvku generujícího odkaz

Začneme úpravou stránky *ProchazeniKatalogu.aspx*, kde do *GridView* jménem *GVProdukty* přidáme nový sloupeček s textem *Detaily...*, který bude obsahovat odkaz na stránku s detailem produktu a příslušným číslem např. *Produkt.aspx?Cislo=10*.

Navolíme tedy *GVProdukty*, zobrazíme si smart tag a vybereme možnost *Edit Columns...* V horním okně *Available fields...* vybereme typ *HyperLinkField* a stiskneme *Add*. Přidá se nový sloupec, jehož vlastnosti musíme upravit. Jako popisek (*Text*) vepíšeme „Detaily...". Jako *Target* zadáme *_blank*, čímž docílíme otevření odkazu v novém okně. Jako *DataNavigateUrlField* doplníme pole, které se bude v hyperlinku přenášet – v našem případě tedy *IDProdukt*. A konečně, místo *DataNavigateUrlFormat* vepíšeme formátovací řetězec *produkt.aspx?Cislo={0}* – za nulu ve složených závorkách se dosadí první datové pole uvedené v *DataNavigateUrlField* (*IDProdukt*, čísluje se od 0).Pokud jsme vše provedli správně, po uložení a zobrazení stránky byste měli vidět následující (povšimněte si dvou červených elips):

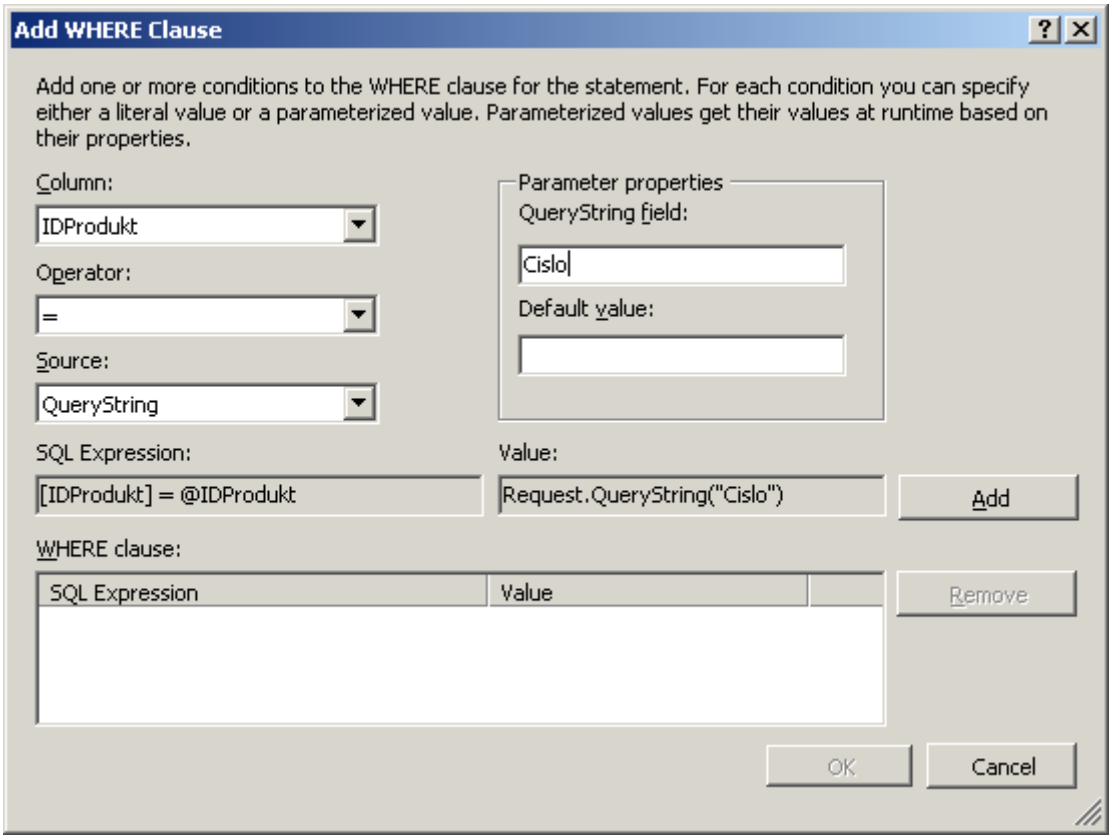


Při přemístění kurzoru myši nad nový sloupeček *Detaily...* se dole v prohlížeči zobrazí odkaz s příslušným číslem produktu, které je pro každý řádek jiné. Při kliknutí na odkaz by se mělo otevřít nové okno se správnou adresou včetně parametru v adresové řádce prohlížeče, ovšem zatím s prvním produktem (v závislosti na setřídění) namísto námi vybraného produktu.

Úprava stránky zobrazující výsledek

Druhým krokem bude filtrování datového zdroje *ZdrojDatProdukty* na stránce *Produkt.aspx* tak, aby zobrazoval pouze vybraný záznam produktu.

Postup je velmi podobný jako v předchozí úloze. Navolíme myší zmíněný datový zdroj a z kontextové nabídky nebo smart tagu zvolíme možnost *Configure Data Source...* První stránce průvodce přejdeme a na druhé stránce specifikujeme výběrovou podmínku stiskem tlačítka *WHERE...* Filtrovaný sloupeček bude *IDProdukt*, operátor bude =, zdroj hodnoty, podle které se filtruje bude URL požadavku (*QueryString*) a jako *QueryString field* zvolíme jméno parametru v adresní řádce, tedy *Cislo* (viz obrázek).



Opět nezapomeneme stisknout tlačítko *Add*, potvrdíme *OK* a přejdeme na další stránku průvodce, kde si můžeme prohlédnout nově vygenerovaný dotaz a vyzkoušet jeho funkci pro některá zadaná platná čísla produktu.

Pokud si nyní znovu zobrazíme stránku *ProchazeniKatalogu.aspx* a pokusíme se zobrazit detail některého z produktů, měli by se nám otevřít nové okno, ale nyní již s detailem správného produktu. Puntičkáři si ještě mohou nechat zobrazit smart tag prvku *DetailsView* se jménem *DVProdukty* a zrušit možnosti *Enable Paging* a *Enable Inserting*.

Práce s hierarchickými daty

Dosud jsme se zabývali pouze plochými nebo chcete-li pravoúhlými daty – byla uspořádána do řádků a sloupečků. V praxi se však často setkáme se stromovým uspořádáním, např. při hledání v dokumentaci nebo v katalozích produktů, kde jsou produkty ve skupinách, podskupinách a někdy

i podpodskupinách. Ve webových aplikacích se často setkáváme s hierarchickou navigací pomocí mapy webu, zobrazené jako rozbalovací strom nebo hierarchická nabídka menu.

Pro tyto účely můžeme v ASP.NET 2.0 použít datový zdroj typu *XmlDataSource*, kde jako zdroj dat může sloužit libovolný XML soubor, z nějž provádíme výběr pomocí XPath výrazů. Druhou možností je použít *SiteMapDataSource*, kde má XML soubor s hierarchickými daty pevně danou strukturu – není tedy tak flexibilní, ale pro začátečníky je vhodnější, neboť se s ním snadněji pracuje.

Vytvoření mapy webu

První úlohou je vytvoření XML souboru s mapou webu. Pokud si zvolíme možnost *Add New Item...* z kontextové nabídky našeho webového projektu, je jednou z možností položka *Site Map* jménem *Web.sitemap*. Automaticky se nám otevře editor XML souboru, který podle pevného schématu kontroluje správnost našeho XML dokumentu. Práce s XML dokumentem předpokládá alespoň minimální znalost XML, výuka XML není součástí našeho mini kurzu. XML soubor upravte na následující text (samozřejmě můžete text též zkopírovat):

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="default.aspx" title="Firma X"  description="Firma X
je nejlepší na světě">
        <siteMapNode url="" title="Produkty"  description="">
          <siteMapNode url="ProchazeniKatalogu.aspx" title="Procházení
katalogu" description="Projděte si katalog"/>
          <siteMapNode url="Produkt.aspx" title="Detaily produktů"
description="Do posledního detailu..."/>
        </siteMapNode>
      <siteMapNode url="" title="Ukázky"  description="Technologická dema">
        <siteMapNode url="UkazkaNavigace.aspx" title="Ukázka navigace"
description="Navigace na webu"/>
        <siteMapNode url="Ukazka2.aspx" title="Ukázka 2"
description="Podívejte se na ukázku 2"/>
        <siteMapNode url="Ukazka3.aspx" title="Ukázka 3"
description="Podívejte se na ukázku 3"/>
        <siteMapNode url="Ukazka4.aspx" title="Ukázka 4"
description="Podívejte se na ukázku 4"/>
      </siteMapNode>
    </siteMapNode>
</siteMap>
```

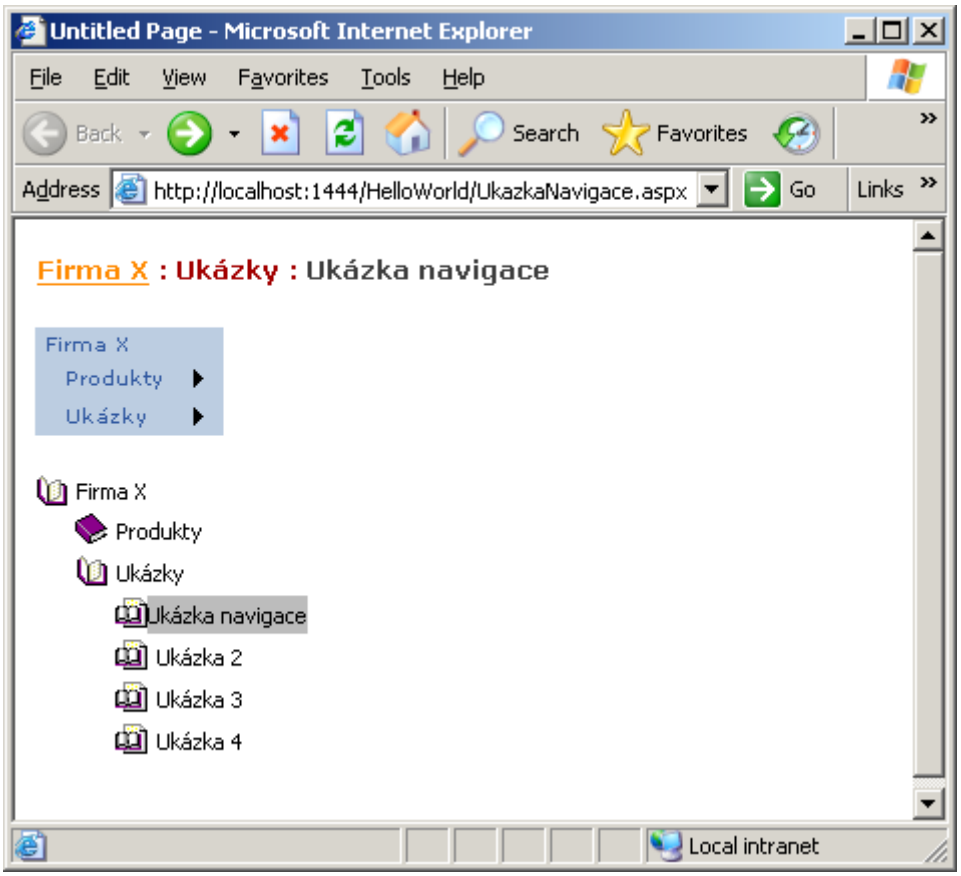
Každý prvek *siteMapNode* v XML dokumentu má několik atributů – *url* je nepovinné a určuje adresu stránky, *title* je povinný (silně doporučený) titulek a *description* je nepovinný doplňující popis, který se zobrazí krátkým podržením kurzoru myši nad příslušným uzlem.

Soubor uložte a zavřete. Tím máme připravenou stromovou strukturu dat pro naše ovládací prvky.

Použití hierarchických navigačních prvků

Nyní si můžeme vyzkoušet ovládací prvky *Menu*, *TreeView* a *SiteMapPath*, které všechny pracují právě s hierarchickými daty. Založíme si novou webovou stránku a nazveme ji *UkazkaNavigace.aspx*. Na stránku přetáhneme z panelu nástrojů zdroj dat *SiteMapDataSource* a přejmenujeme ho nastavením vlastnosti *(ID)* z původního *SiteMapDataSource1* na *ZdrojDatNavigace*. Datový zdroj není třeba konfigurovat, neboť bude automaticky používat soubor *Web.sitemap*.

Z panelu nástrojů na plochu přetáhneme ovládací prvky *SiteMapPath*, *Menu* a *TreeView*, které pro přehlednost oddělíme prázdnými řádky. U prvku *Menu1* a *TreeView1* si postupně zobrazíme smart tag a nastavíme u nich zdroj dat (*Choose Data Source:*) na *ZdrojDatNavigace*. Tím je vše hotové, stránku si můžeme zobrazit v prohlížeči a měla by vypadat o něco hůře než následující obrázek:



Prvek *SiteMapPath1* ukazuje polohu současné adresy ve stromu položek. Vyzkoušíme si zbylé dva prvky – nabídku *Menu1* a rozbalovací strom *TreeView1*:

- jednotlivé uzly se rozbalují a sbalují
- položky s atributem *url* fungují jako odkazy (vyzkoušejte volbu položky *Procházení katalogu*)
- položka, u níž *url* odpovídá současné adrese (*Ukázka navigace*), je barevně odlišena
- krátkým podržením kurzoru myši nad uzlem zobrazíte popisný text *description* jako tzv. *tooltip*

Jistě jste si povšimli, že na obrázku jsou jednotlivé prvky vizuálně vylepšené. Toho dosáhneme experimentováním s úpravou vlastností jednotlivých prvků. Pomocí funkce *AutoFormat* z nabídky smart tag lze rychle vylepšit jejich vzhled. U prvku *Menu1* je zajímavá volba *Views* ve smart tagu (možnosti jsou *Static* a lepší *Dynamic*), vlastnost *StaticDisplayLevels* pro počáteční zobrazení statických úrovní a řada dalších zajímavých vlastností ovlivňujících vzhled. U prvku *TreeView1* jsou příklady zajímavých vizuálních vlastností *ExpandDepth* anebo *ShowCheckBoxes*.

Závěrem

Dnešní lekcí jsme ukončili práci s daty. Příště se budeme věnovat jednotnému vzhledu aplikace. Přestože ASP.NET nám nijak nepomůže s naším estetickým cítěním, umožní nám dosáhnout jednotného vzhledu stránek pomocí témat (skinování) a používání tzv. master stránek, které můžeme například vhodně zkombinovat s dnes použitými navigačními prvky.

A opět připomínka – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

8. JEDNOTNÝ VZHLED ASP.NET APLIKACÍ

V minulých dvou lekcích jsme se věnovali práci s daty na stránkách ASP.NET. Dnes zabrousíme do zcela jiné oblasti – zajištění jednotného vzhledu ASP.NET aplikací. Pro tento účel jsou zajímavé dvě technologie. První je skinování pomocí témat, které se stará o jednotný vzhled ovládacích prvků (barev, druhů písma apod.). Druhou technologií jsou tzv. master pages, které mají za úkol zajistit jednotné rozložení potřebných sekcí na stránce jako celku.

Skinování (themes)

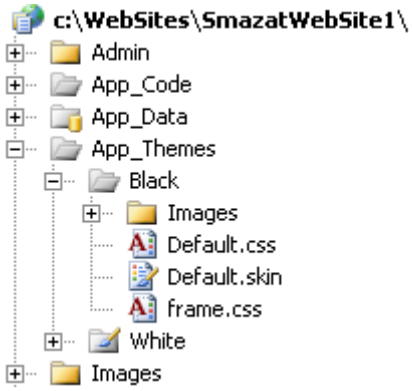
Prakticky každý, kdo vytvářel webové stránky, se někdy setkal s technologií *Cascading Style Sheets* (CSS). CSS jsou v podstatě šablonou, která na jednom místě nastavuje vzhled a vlastnosti HTML značek. CSS například může říkat, že hlavička úrovně 2 je napsána velkým tučným modrým fontem Arial. Skinování v ASP.NET funguje na podobném principu, ale na vyšší úrovni.

Princip a použití témat

Skinování totiž nepracuje s jednotlivými HTML značkami, ale s ovládacími prvky ASP.NET jako s objekty. Skin mi například říká, že všechna tlačítka v mé aplikaci mají pozadí (vlastnost *BackColor*) nastavenou na námořnickou modř, anebo že všechny kalendáře mají vybraný den znázorněný tučně (vlastnost *SelectedDayStyle.Font.Bold* je nastavena na *True*).

Skinování v ASP.NET není v rozporu s technologií CSS. Jako součást tématu můžeme mít CSS soubor, kterým se pak zobrazení stránky rovněž řídí, anebo při definici skinu můžeme definovat, kterou CSS třídu bude vybraný den v kalendáři používat (vlastnost *SelectedDayStyle.CssClass*). CSS sjednocuje vzhled na úrovni HTML značek, zatímco skinování dělá totéž na úrovni objektů ovládacích prvků, mohou se tedy výborně doplňovat.

Pro první pokusy využijeme dvou připravených témat definovaných ve starter kitu *Personal Web Site*. Ve VWD si vytvoříme nový projekt podle tohoto starter kitu a v okně *Solution Explorer* se zaměříme na složku *App_Themes*:



V této systémové složce jsou umístěna témata, každá podsložka odpovídá jednomu tématu – v našem případě máme tedy k dispozici témata *Black* a *White*. V každém tématu jsou typicky obsaženy:

- skiny – soubory s příponou *.skin*
- CSS stylesheety, které jsou automaticky vloženy do hlaviček stránek, jež téma používají
- další pomocné soubory (např. obrázky), na které se skiny a CSS stylesheety odkazují

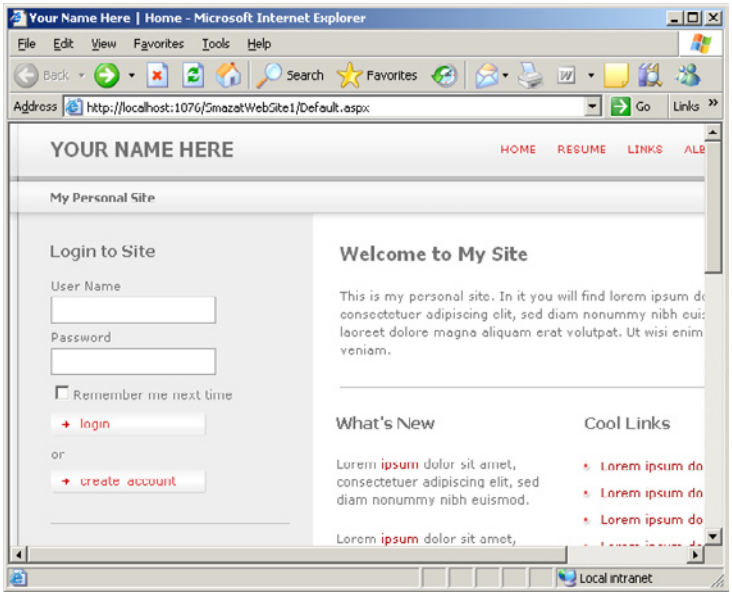
Použití určitého tématu je možné nastavit buď pro celý web anebo pro každou stránku. Nastavení na úrovni webu se provádí v konfiguračním souboru *web.config*, kde je uvnitř elementu <*system.web*> vložen element <*pages*> s atributem *theme*. Přepnutí na černé téma tedy provedeme nastavením řádku:

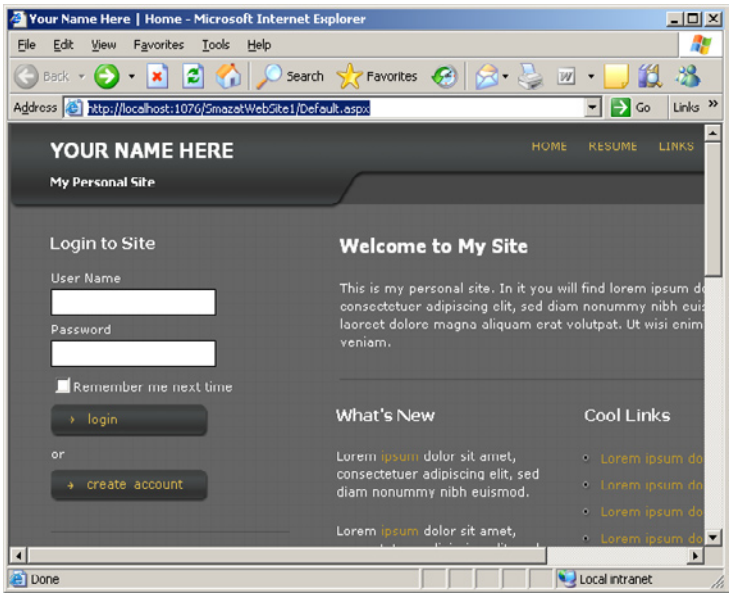
```
<pages theme="Black"/>
```

a přepnutí na bílé téma přepsáním na:

<pages theme="White"/>

Úvodní stránka osobního portálu pak vypadá jedním z těchto způsobů.





Nastavení pro celý web je možné „přebít" na úrovni jedné stránky v direktivě *@Page*, pomocí vlastnosti *Theme* například takto:

<%@ Page Language="C#" MasterPageFile="~/Default.master" Title="Titulek" CodeFile="Default.aspx.cs" Inherits="Default_aspx" ***Theme="Black"*** *%>*

Vytvoření vlastního tématu

Opustíme nyní ukázkové použití témat v personálním webu. Místo toho se vrátíme k našemu webu, který jsme používali pro zkoušení až do 7. lekce. Vytvoříme si vlastní téma jménem *Divocina* (to abychom neměli výčitky svědomí kvůli estetickému dojmu) a ukážeme si, jak pomocí něj můžeme změnit a sjednotit vzhled stránky *ProchazeniKatalogu.aspx*.

Nejprve si vytvoříme příslušnou složku. Zobrazíme-li kontextovou nabídku na našem webu v okně *Solution Explorer*, najdeme zde nabídku *Add Folder/Theme Folder*. Tím vytvoříme na webu složku *App_Themes* a v ní podsložku *Theme1*, kterou ihned přejmenujeme na *Divocina*. Zobrazíme si kontextovou nabídku na složce *Divocina* a vybereme si možnost *Add New Item...*, kde z nabídky typů vybereme *Skin File*. Jméno můžeme nechat libovolné, např. *MujSkin.skin*.

Skin soubory obsahují definice objektů s nastavením jejich vlastností ve tvaru velmi podobném jako ASPX stránky. Protože pro skin soubory není k dispozici grafický designér a ani editor nenabízí při psaní nápovědu, uchýlíme se k jednoduchému triku. Vytvoříme si novou stránku *Pomocna.aspx*, přepneme do zobrazení *Design* a z palety nástrojů na ni přeneseme prvek *DropDownList*. Nyní můžeme nastavit některé jeho vizuální vlastnosti – např. pozadí (*BackColor*) nastavíme na zelenou (*Green*), barvu písma (*ForeColor*) na červenou (*Red*) a tučnost písma (*Font/Bold*) na *True*. Nyní přepneme do zobrazení *Source* a myší navolíme celý prvek *asp:DropDownList*, který by měl vypadat zhruba takto:

```
<asp:DropDownList ID="DropDownList1" runat="server" BackColor="Green" Font-Bold="True"
ForeColor="Red"></asp:DropDownList>
```

Tento text pak nakopírujeme do *.skin* souboru (např. *MujSkin.skin*) a vymažeme z něj text:

```
ID="DropDownList1"
```

takže ve *.skin* souboru zůstane pouze:

```
<asp:DropDownList runat="server" BackColor="Green" Font-Bold="True" ForeColor="Red"></asp:
DropDownList>
```

Dalším prvkem, který upravíme bude *GridView*. Stránku *Pomocna.aspx* přepneme opět do zobrazení *Design* a přeneseme na ni prvek *GridView* z panelu nástrojů. Nastavíme některé jeho vizuální vlastnosti, např. barvu pozadí pro hlavičky řádků (*HeaderStyle/BackColor*) na červenou (*Red*), pozadí řádků (*RowStyle/BackColor*) na žlutou a pozadí střídavých řádků (*AlternatingRowStyle/BackColor*) na světle zelenou (*LightGreen*). Opět přepneme do pohledu *Design*, nakopírujeme celý prvek *asp:GridView* do našeho *.skin* souboru a analogicky z něj vymažeme

```
ID="GridView1"
```

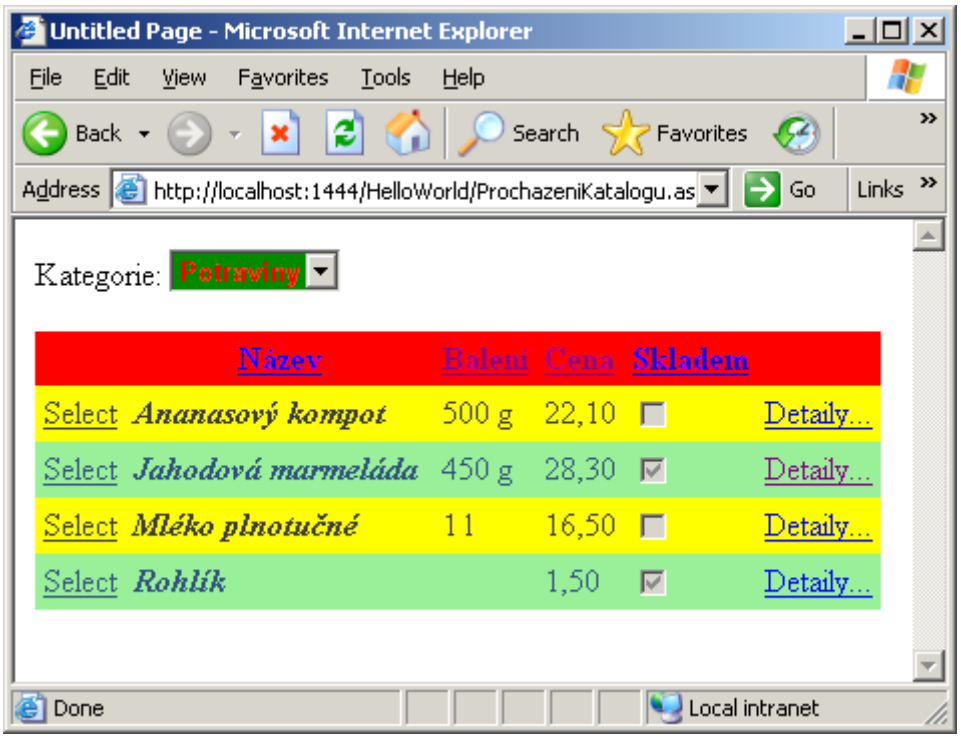
takže nám zůstane

```
<asp:GridView runat="server">
    <RowStyle BackColor="Yellow" />
    <HeaderStyle BackColor="Red" ForeColor="Yellow" />
    <AlternatingRowStyle BackColor="LightGreen" />
</asp:GridView>
```

Zbývá nám pouze zapnout naše téma *Divocina*, aby se projevilo na všech prvcích *DropDownList* a *GridView* na našem webu. Ještě před tím si můžeme v prohlížeči zobrazit stránku *ProchazeniKatalogu.aspx*, abychom si ověřili, že na ní zatím nejsou žádné změny patrné. Používání tématu zapneme v konfiguračním souboru *web.config* (pokud ho ve svém projektu ještě nemáme, vytvoříme si nový pomocí *Add New Item.../Web Configuration File*). Soubor otevřeme a do prvku *system.web* dopíšeme tučně zvýrazněný text:

```
<system.web>
    <pages theme="Divocina"/>
```

Po uložení změn ve všech souborech si znovu necháme zobrazit stránku *ProchazeniKatalogu.aspx*:



Jak vidíme, stránka nyní připomíná barvy amazonského pralesa. Jednoduchým použitím témat a skinů jsem zajistili jednotný (byť poněkud divoký) vzhled všech prvků *DropDownList* a *GridView* na našem webu.

Pokročilejší funkce témat

V práci se skiny lze použít i některé pokročilejší možnosti. První je možnost pojmenování skinů, kdy při definici skinu nastavíme vlastnost *SkinID* na nějaký identifikátor např. *DuleziteTlacitko*. Takový skin se pak nazývá *named skin*, zatímco pro skin bez jména je označení *default skin*. Při vytváření ovládacích prvků pak můžeme nastavit jejich vlastnost *SkinID*, čímž určíme, který pojmenovaný skin se použije – pro důležitá tlačítka tedy nastavíme *SkinID="DuleziteTlacitko"*. Pokud tuto vlastnost nevyplníme, použije se automaticky *default skin* pro tlačítko.

Druhou možností je rozlišování vlastností *Theme* a *StyleSheetTheme* na celém webu nebo na stránce. Obě vlastnosti mají podobný efekt, liší se pouze svojí váhou při rozhodování o tom, jaká vlastnost prvku bude nakonec použita. ASP.NET používá pro rozhodnutí o vzhledu ovládacího prvku následující postup:

1. Je-li vlastnost definovaná v rámci skinů v *Theme* tématu, použije se
2. Není-li tomu tak a je-li vlastnost definovaná přímo na stránce u ovládacího prvku, použije se
3. Není-li tomu tak a je-li vlastnost definovaná v rámci skinů *StyleSheetTheme* tématu, použije se

Jinými slovy, *Theme* má přednost před nastavením jednotlivých prvků na stránce a zaručuje nám jednotný vzhled ovládacích prvků za každých okolností. Naproti tomu *StyleSheetTheme* je jakési doporučené výchozí nastavení, které může vývojář nastavením jednotlivých ovládacích prvků přepsat.

Více informací o použití skinů můžete najít např. v tomto článku[25].

25 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/themes.asp

Jednotné rozložení (master pages)

Master pages zajišťují jednotné rozložení webových stránek tak, aby jednotlivé stránky webu působily jednotným dojmem.

Jak fungují?

Master stránka je vlastně jakousi šablonou určující vzhled stránek, jež ji používají. Typická master stránka definuje hlavičku, patičku a levý pruh webové stránky. Stránky, jež tento master používají, pak zpravidla doplňují „vnitřek" stránky k vnějším oblastem definovaných masterem. Podobný princip používají např. venkovské hudební skupiny, které si nechávají předtisknout plakáty s bílými místy, do kterých pak vepisují, kdy a kde se jejich produkce koná – master stránky fungují analogicky.

Dobrým příkladem je personální web, který jsme zkoumali i v případě témat. Vzhled master stránky je v souboru s příponou *.master* – otevřeme tedy soubor *Default.master* a přepneme do zobrazení *Design*. Na první pohled nepoznáme žádný rozdíl mezi normální stránkou a masterem – můžete na ně umisťovat ovládací prvky, nastavovat jejich vlastnosti, psát události apod. Jeden rozdíl ale existuje – existence prvku *ContentPlaceHolder*. Jedná se o speciální prvek, který bude vyplněn skutečným obsahem na stránkách používajících příslušný master (na každé master stránce může být i více pojmenovaných prvků typu *ContentPlaceHolder*).

Zajímavější bude otevřít některou stránku používající master. Otevřeme například stránku *Resume.aspx* a přepneme ji do zobrazení *Design*:

Vidíme, že oblasti definované na masteru jsou zašedlé a nemůžeme je upravovat. Naproti tomu *ContentPlaceHolder* označuje oblast, se kterou můžeme na stránce pracovat tak, jak jsme zvyklí – vytvářet zde ovládací prvky apod.

Zobrazíme-li si stránku v režimu *Source*, vidíme vazbu na master stránku hned na prvním řádku v direktivě *@Page*:

<%@ Page Language="C#" ***MasterPageFile="~/Default.master"*** *... %>*

Za povšimnutí stojí ještě následující řádek:

<asp:content id="Content1" ***contentplaceholderid="Main"*** *runat="server">*

asp:content označuje speciální ovládací prvek, jehož jedinou úlohou je vyplnit prvek *ContentPlaceHolder* definovaný v master stránce, jehož *ID* je shodné s *contentplaceholderid* vlastností zde definovanou. Při používání VWD nás ovšem tyto podrobnosti nemusíš až tak zajímat.

Vytvoření master stránky

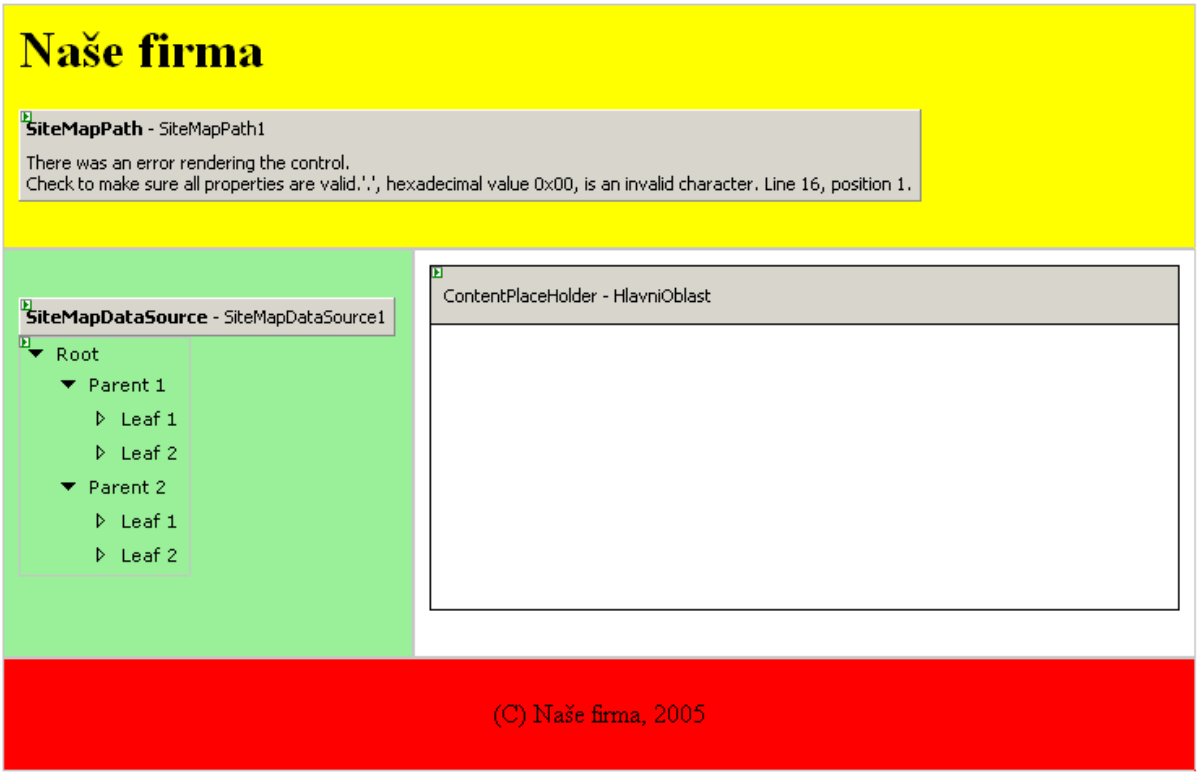
Po vysvětlení principů se pustíme do úpravy našeho pokusného projektu pro používání masteru. Vytvoření master stránky je jednoduché – v okně *Solution Explorer* vybereme z kontextové nabídky *Add New Item...*, jako typ vybereme *Master Page* a jméno zvolte *Sablona.master*.

Stránka bude mít hlavičku, patičku, boční lištu vlevo a hlavní obsah vpravo. Přepneme stránku *Sablona.master* do zobrazení *Design* a vymažeme z ní vložený prvek *ContentPlaceHolder1*. Z nabídky vložíme tabulku volbou *Layout/ Insert Table*, kde z nabídky šablon (*Template*) zvolíme *Header, footer and side*.

Nejprve upravíme hlavičku stránky. Navolíme myší horní pole tabulky a v okně *Properties* nastavíme pozadí (*BgColor*) na žlutou (*Yellow*). Na žlutě označeném poli tabulky si zobrazíme kontextovou nabídku a vybereme *Resize/Resize Row*, kde pak výšku nastavíme na 100px. Do políčka vepíšeme nějaký text (např. „Naše firma"), který můžeme pomocí nabídky na formátovací liště zvětšit na hlavičku první úrovně (*Heading 1*). Vytvoříme nový řádek a vložíme prvek *SiteMapPath* pro zobrazení navigace (můžeme pro něj použít *Auto Format...* pro vylepšení vzhledu) – viz obrázek dále.

Upravíme též patičku. Navolíme myší dolní pole tabulky a v okně *Properties* nastavíme pozadí (*BgColor*) na červenou (*Red*). Na červeně označeném poli tabulky si zobrazíme kontextovou nabídku a vybereme *Resize/Resize Row*, kde pak výšku nastavíme na 50px. Do políčka vepíšeme nějaký text (např. „ (C) Naše firma, 2005"), pro lepší efekt ho můžeme umístit na střed (*Justify Center*) tlačítkem ve formátovací liště nahoře – viz obrázek dále.

Nyní vytvoříme hlavní oblast. Stačí přetáhnout myší z panelu nástrojů prvek *ContentPlaceHolder* do pravého prostředního políčka tabulky a jeho jméno *(ID)* změníme z *ContentPlaceHolder1* na *HlavniObsah* – viz obrázek:



Zbývá vytvoření levé navigační lišty. Navolíme myší levé prostřední pole tabulky a v okně *Properties* nastavíme pozadí (*BgColor*) na světle zelenou (*LightGreen*). Na světle zeleném označeném poli tabulky si zobrazíme kontextovou nabídku a vybereme *Resize/Resize Column*, kde pak šířku nastavíme na 150px. Do pole tabulky vložíme datový zdroj *SiteMapDataSource* a hned pod něj ovládací prvek *TreeView*. U prvku *TreeView1* si zobrazíme jeho smart tag a zdroj dat (*Choose Data Source*) nastavíme na *SiteMapDataSource1*, rovněž můžeme upravit jeho vzhled použitím volby *Auto Format...* ze smart tagu nastavením např. na volbu *Arrows*.

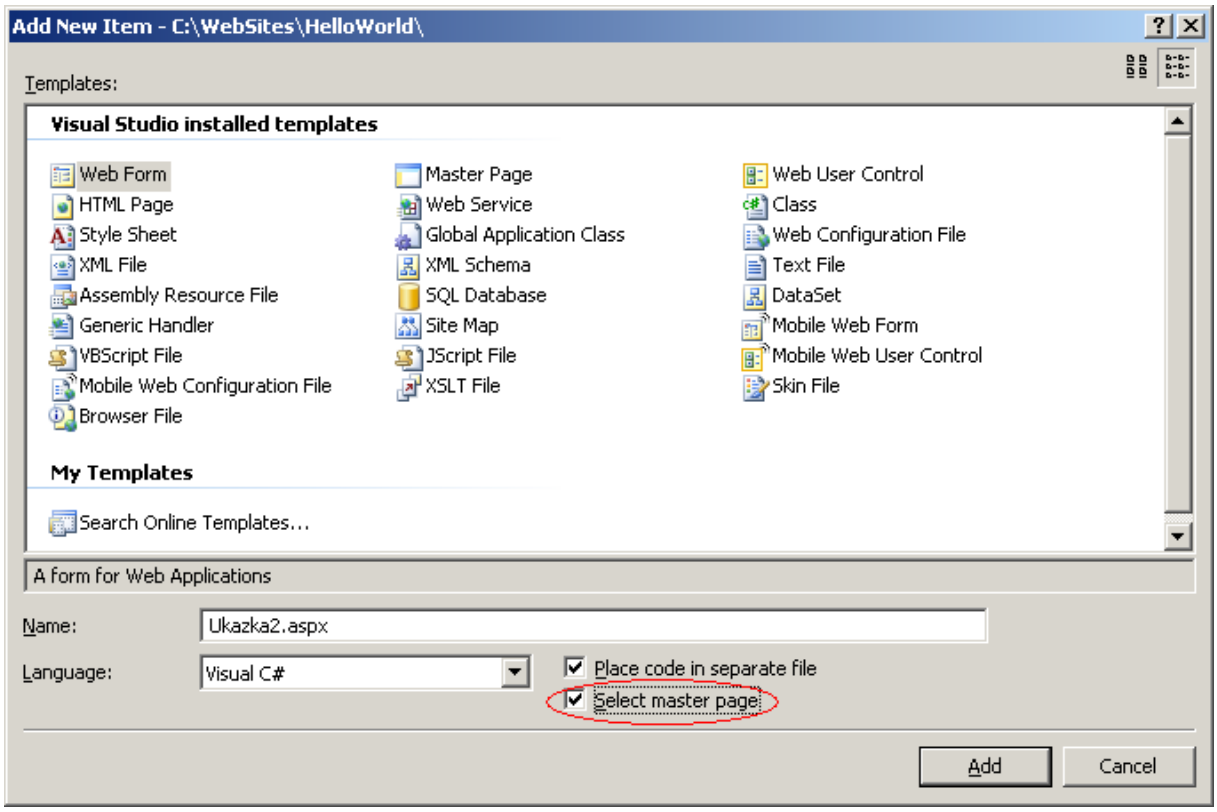
Poslední úpravou bude nastavení odstupů (*cellpadding*) na 10 bodů a zrušení maximální výšky a šířky tabulky úpravou prvku <*table*> v zobrazení *Source* na:

<table border="0" cellpadding="10" cellspacing="0">

Master stránka by nyní po přepnutí do režimu *Design* měla vypadat podobně jako na obrázku výše. Stránku uložíme.

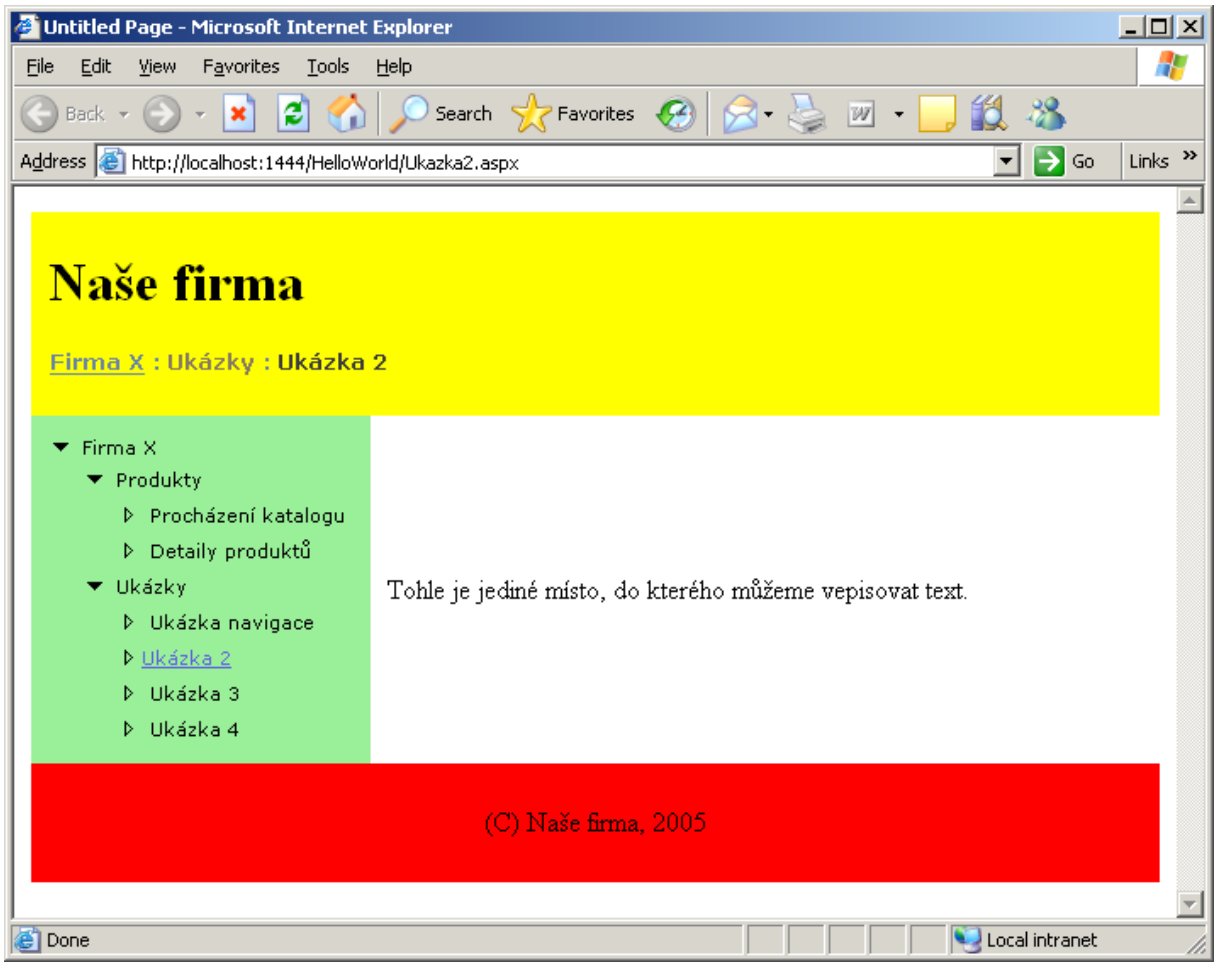
Stránky používající master

Vytvoření stránky používající master je velmi snadné. Založíme novou stránku, nazveme ji *Ukazka2.aspx* a na dialogu *Add New Item* zatrhněte možnost *Select master page* (viz obrázek):



V dalším dialogu *Select a Master Page* pak vybereme master stránku *Sablona.master*. Pokud stránku *Ukazka2.aspx* přepneme do zobrazení *Design*, můžeme editovat pouze oblast hlavního obsahu stránky. Hlavička, patička a levá lišta jsou zašedlé a nelze je měnit. Do pole *Content1* vepíšeme libovolný text. Zobrazíme-li si stránku v prohlížeči, vidíme správnou funkci master stránky:

Poslední úlohou bude úprava existující stránky *ProchazeniKatalogu.aspx* pro používání naší master stránky. Stránku *ProchazeniKatalogu.aspx* si otevřeme v zobrazení *Source* a provedeme několik jednoduchých úprav, spočívajících ve vymazání standardních HTML značek a uzavření obsahu stránky do prvku *asp:content*.



Nejprve vymažeme na začátku stránky text:

```
<!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“>
```

a nahradíme ho textem

```
<asp:Content ID=“Content1“ ContentPlaceHolderID=“HlavniOblast“ Runat=“Server“>
```

Podobně na konci stránky vymažeme značky

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

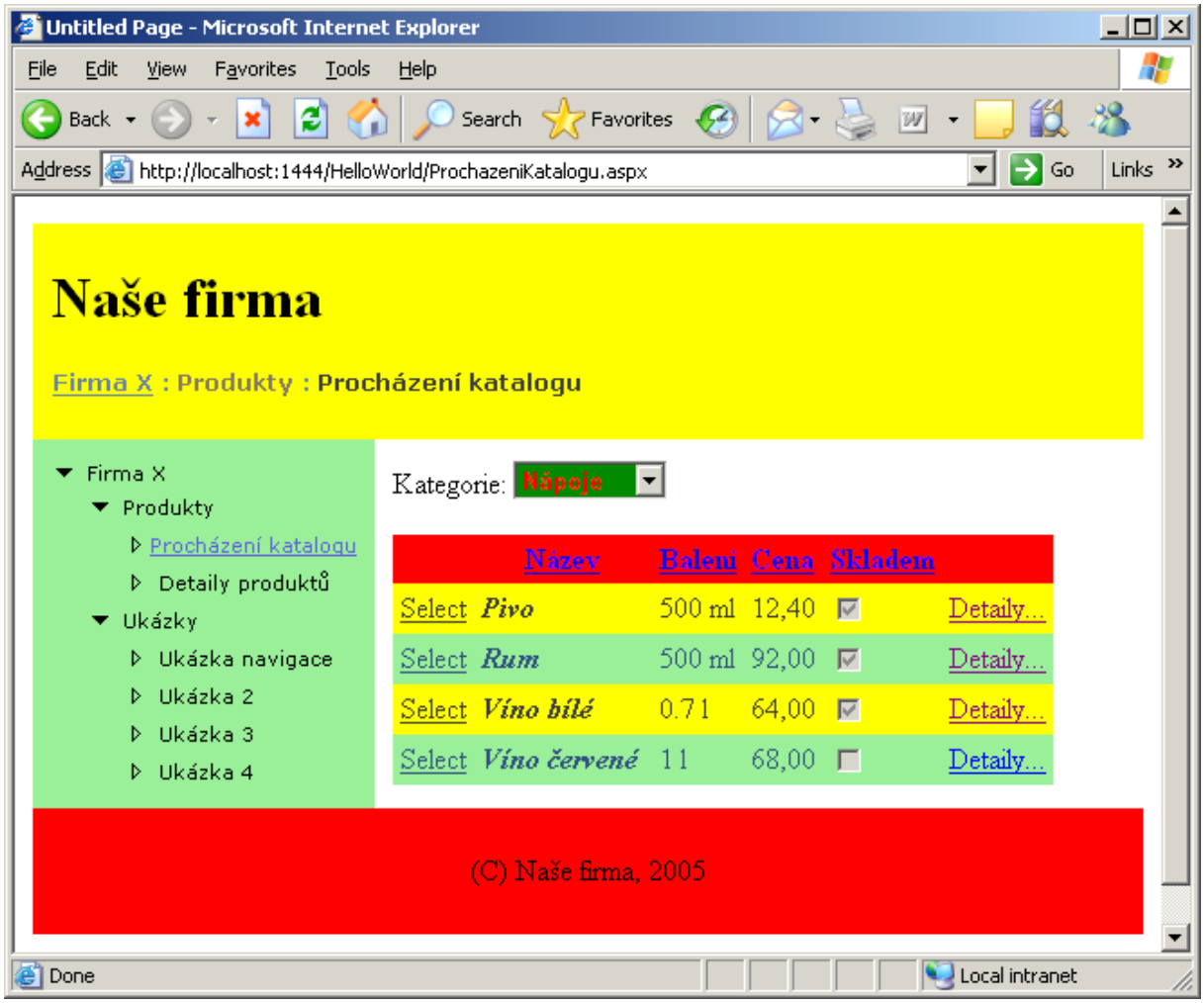
a nahradíme je značkou

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

Poslední nutnou úpravou je dopsání odkazu na master stránku do direktivy *@Page*, kam stačí doplnit tučně označený text:

*<%@ Page Language="C#" **MasterPageFile="~/Sablona.master"** AutoEventWireup="true" CodeFile= "ProchazeniKatalogu.aspx.cs" Inherits="ProchazeniKatalogu" %>*

Tím jsme provedli nezbytné úpravy, po nichž můžeme stránku upravovat i v režimu *Design* anebo si ji v novém rozložení podle master stránky zobrazit v prohlížeči:



Úprava existující stránky tedy vyžadovala trochu ruční editace, ale výsledek stojí za to. Více informací o master pages lze najít např. v tomto článku[26].

Závěrem

Tentokrát jsme se věnovali sjednocení vzhledu stránek – jednotný vzhled ovládacích prvků lze zajistit pomocí témat a skinů, jednotné rozložení stránek pomocí master stránek. V předposlední lekci se pro změnu budeme věnovat zabezpečení webu – definici uživatelů, přihlašování na web, vytváření rolí a zabezpečení obsahu pouze pro uživatele v určité roli.

Neustále se opakující uzavření – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

26 http://msdn.microsoft.com/library/en-us/dnvs05/html/masterpages.asp

9. AUTENTIZACE, AUTORIZACE

V dnešní předposlední lekci se pro změnu budeme věnovat zabezpečení webu pro různé uživatele. V první části se budeme věnovat autentizaci – v našem případě přihlášení uživatele jménem a heslem. Ve druhé části se zaměříme na autorizaci – v našem případě nastavení přístupu uživatelů k jednotlivým stránkám webu.

Autentizace

Autentizace je proces, při kterém má přistupující uživatel prokázat, že je tím, za koho se vydává. Nejčastěji to bývá prokázání znalosti jména a hesla anebo předložení certifikátu. U webových aplikací v ASP.NET se pro autentizaci používají zpravidla dva mechanismy.

U intranetových aplikací se pro přihlášení používají většinou účty, kterými se uživatelé přihlašují do síťového prostředí. Na platformě Windows jde o účty ve Windows doméně, kterými se uživatelé přihlašují „při příchodu do práce" a dále se tento účet víceméně bez vědomí uživatele používá pro přístup ke všem dalším zdrojům v síti – mluvíme o tzv. integrované autentizaci, v ASP.NET se označuje jako *Windows autentizace*.

Druhý mechanismus je typicky používán u aplikací na Internetu. Při tomto způsobu jsou jména a hesla uživatelů uložena v rámci aplikace, zpravidla v relační databázi. Při přihlašování pak uživatel musí vyplnit formulář se jménem a heslem, na základě kterého se pak rozhoduje o jeho přihlášení, přičemž přihlašovací jméno se mezi jednotlivými stránkami uchovává většinou v podepsaném řetězci cookie. Tento způsob se v ASP.NET nazývá *Forms autentizace* a budeme jej používat v našich experimentech.

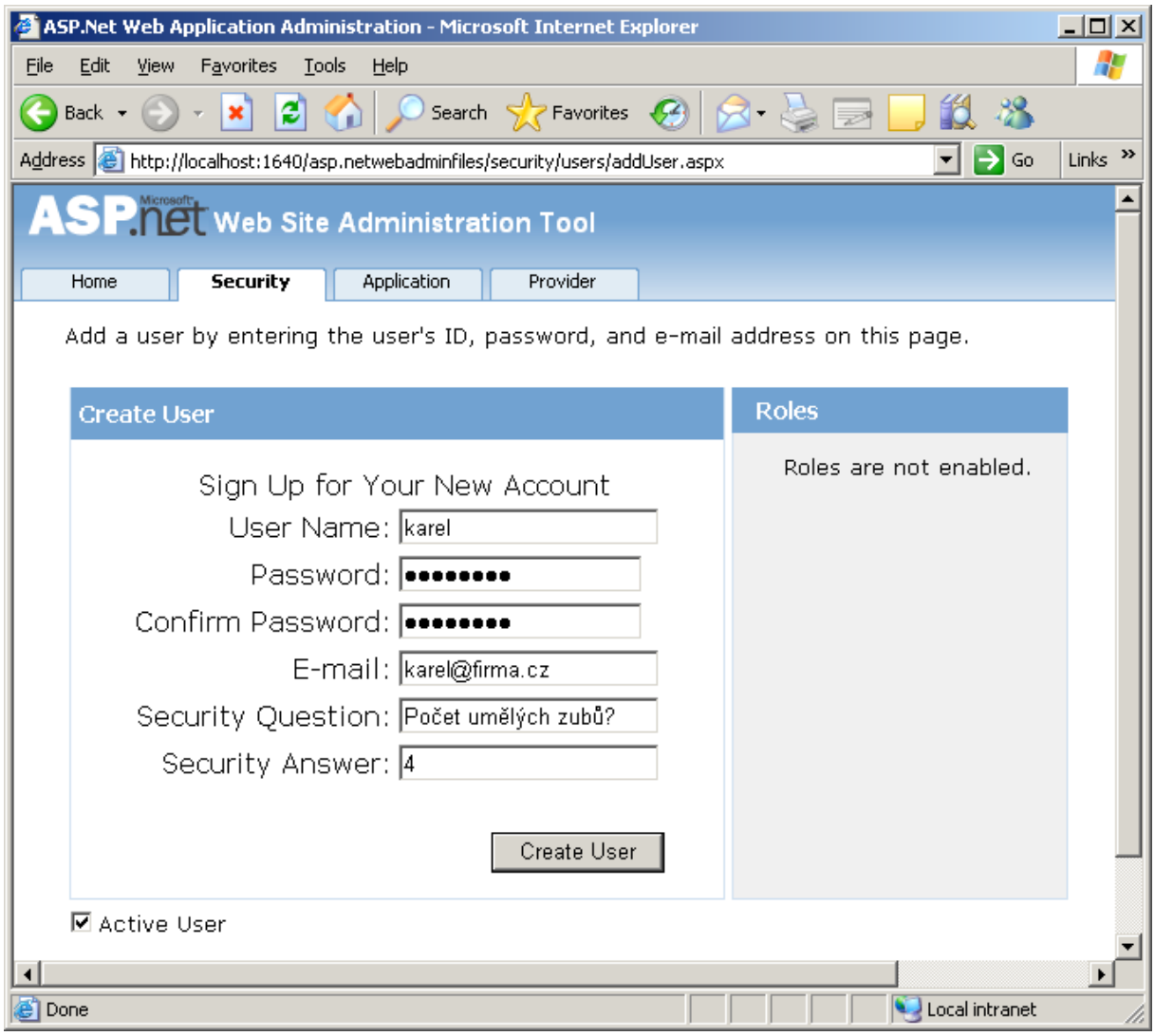
Definice uživatelských účtů

Pro používání Forms autentizace musíme nejprve vytvořit tabulku v databázi a vložit do ní data o účtech uživatelů. Pro tuto úlohu je k dispozici konfigurační nástroj, který z VWD spustíme nabídkou *Website/ASP.NET Configuration* a přepneme na záložku *Security*. Jak vidíme, web je nyní nastaven na používání Windows autentizace. Přepnutí na Forms autentizaci provedeme volbou *Select authentication type* a zvolíme možnost *From the Internet* a potvrdíme *Done*. Tato akce přidá do konfiguračního souboru *web.config* řádek

<authentication mode="Forms" />

a zároveň se ve složce *App_Data* vytvoří nová databáze *SQL Express* jménem *ASPNETDB.MDF*, ve které budou účty ukládány.

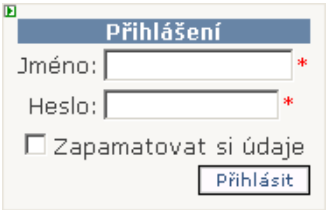
V konfigurační aplikaci zároveň přibyly možnosti *Create User* a *Manage Users*. Nového uživatele založíme samozřejmě volbou *Create User*:



Jako jméno zadáme *karel* a jako heslo *karel.123*, druhého založeného uživatele pojmenujeme *marie* a přiřadíme mu heslo *marie.123*.

Vytvoření přihlašovací stránky

Druhým krokem bude vytvoření přihlašovací stránky. Založíme novou stránku *Prihlasovani.aspx*. Pokud chceme, můžeme zvolit *Select master page* a založit ji na naší stránce *Sablona.master*. Na stránku přeneseme z palety nástrojů prvek *Login*, který udělá veškerou práci za nás. Zobrazíme jeho nabídku smart tag a upravíme vzhled volbou *Auto Format...* například na vzhled *Professional*. Pokud chceme, můžeme též počeštit texty ovládacího prvku pomocí vlastností *LoginButtonText*, *PasswordLabelText*, *RememberMeText*, *TitleText* a *UserNameLabelText*:



Zbývá ještě zaregistrovat naši přihlašovací stránku v konfiguračním souboru *web.config*, což provedeme nahrazením původního řádku

<authentication mode="Forms"/>

opravenou verzí

```
<authentication mode="Forms">
    <forms loginUrl="~/Prihlaseni.aspx"/>
</authentication>
```

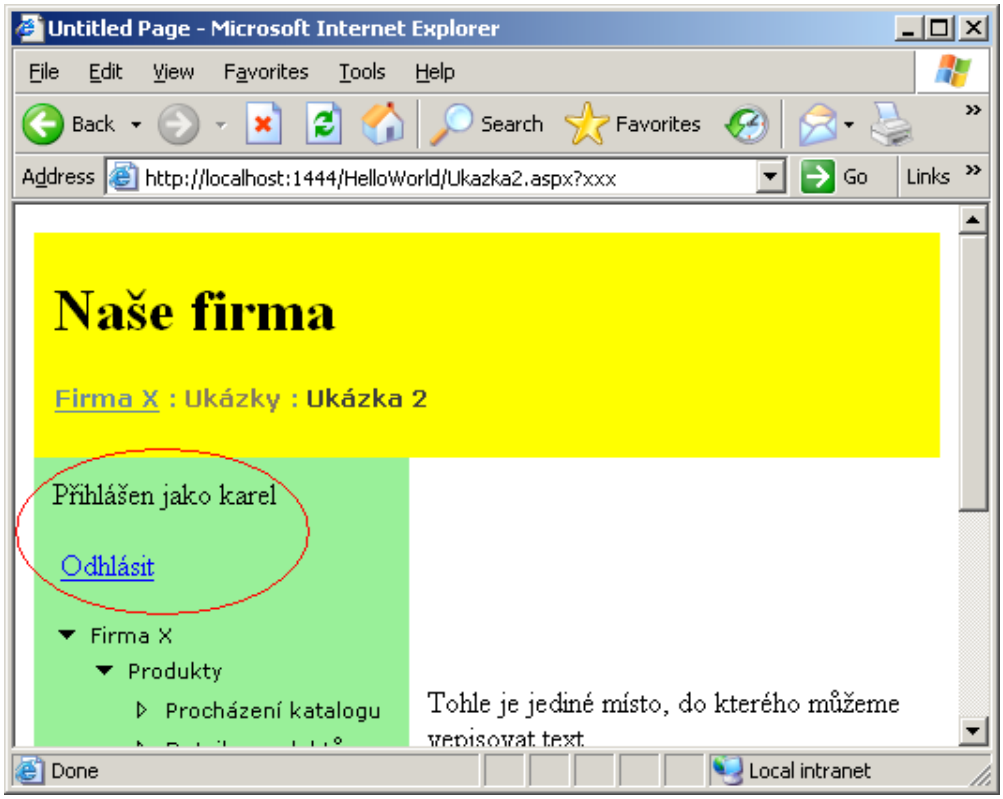
Stránku si nyní můžeme zobrazit v prohlížeči. Pokud zadáme neplatné jméno a heslo, dostaneme po stisknutí tlačítka chybové hlášení o neplatných přihlašovacích údajích. Po zadání správných přihlašovacích údajů dojde k přesměrování na stránku *default.aspx*

Ovládací prvky pro přihlašování

Na závěr první části zakomponujeme přihlašování do master stránky *Sablona.master*. Do levé sekce přeneseme z panelu nástrojů prvek *LoginStatus*, který slouží k provedení přihlášení a odhlášení uživatele, můžeme ho počeštit nastavením vlastností *LoginText* a *LogoutText* na *Přihlásit* a *Odhlásit*.

Nad něj můžeme z panelu nástrojů přesunout prvek *LoginView*, který zobrazuje různý text v závislosti na tom, zda uživatel je či není přihlášen. K tomu slouží šablony *AnonymousTemplate* a *LoggedInTemplate*, mezi kterými můžete přepínat pomocí smart tagu a do kterých můžeme přímo vepisovat text anebo přenášet ovládací prvky podobně jako na stránku. Prvek standardně zobrazuje anonymní šablonu, takže můžeme vepsat například *Nikdo není přihlášen*. Pomocí smart tagu přepneme do zobrazení *LoggedInTemplate*, vepíšeme text *Přihlášen jako* a myší přetáhneme za tento text prvek *LoginName*, který zobrazuje jméno aktuálně přihlášeného uživatele.

Pokud si teď zobrazíme jakoukoliv stránku používající *Sablona.master* (např. stránku *Ukazka2.aspx*), na levé straně uvidíme text *Nikdo není přihlášen* a odkaz *Přihlásit*, který nás přenese na přihlašovací stránku. Pokud správně zadáme přihlašovací údaje uživatele *karel*, jsme přesměrováni zpět na původní stránku. Na levé straně je pak již text *Přihlášen jako karel* a odkaz se změnil na *Odhlásit* (viz obrázek), což koneckonců můžeme také vyzkoušet:



Pokud chcete sami pokračovat v dalším zkoušení, jsou vám k dispozici další prvky související s autentizací – *CreateUserWizard* pro samoobslužné vytvoření uživatelského účtu, *PasswordRecovery* pro objednání zaslání zapomenutého hesla e-mailem a *ChangePassword* pro změnu hesla k existujícímu účtu.

Autorizace

Přihlášení je pouze prvním krokem, který je sám o sobě málo užitečný. Obvykle na něj navazuje rozlišení uživatelů různými právy pro provádění úkonů nebo prohlížení informací – takzvaná autorizace.

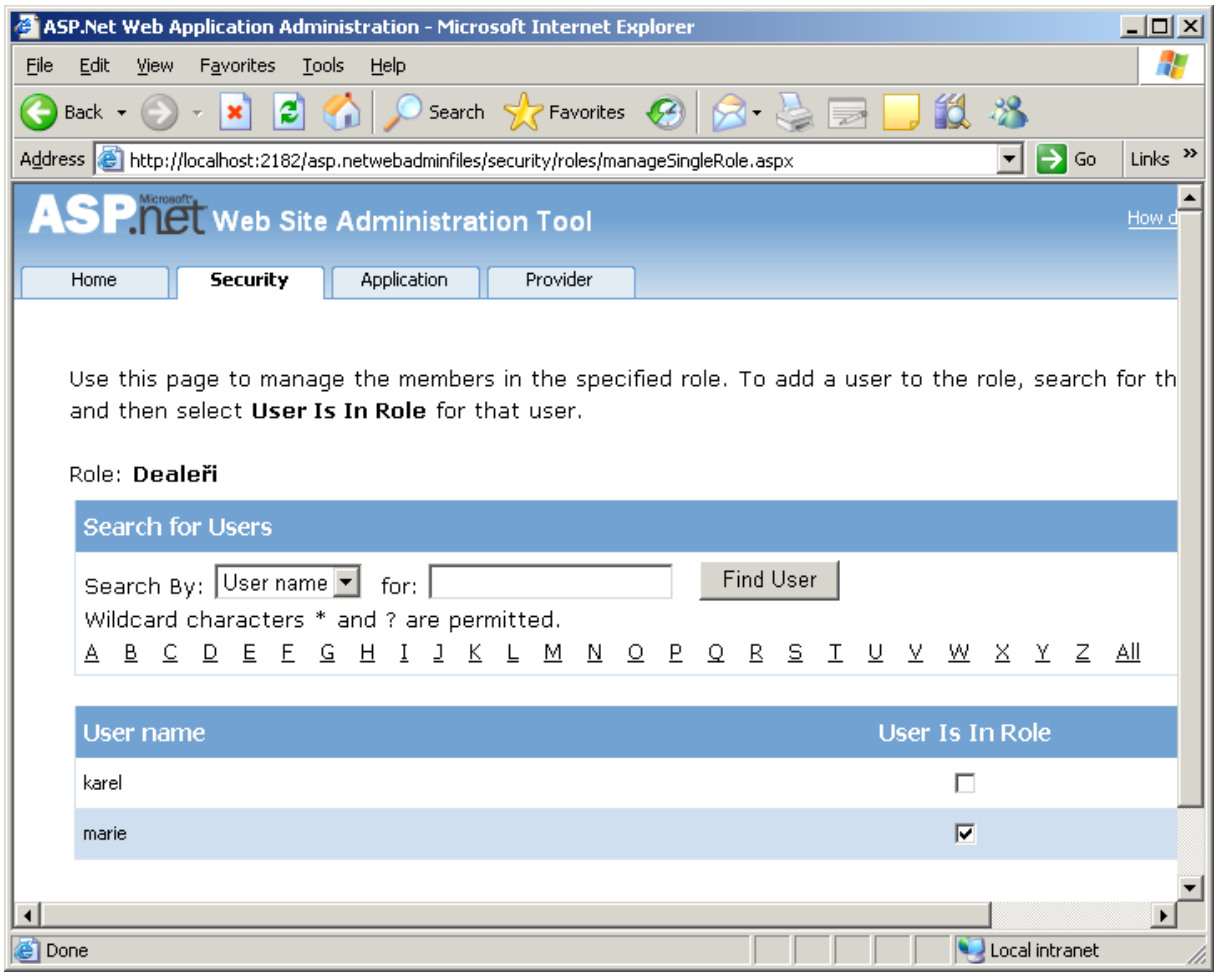
Definice rolí

Protože nastavování práv jednotlivým uživatelům by bylo ve většině případů příliš pracné, používají se pro přiřazování práv tzv. role. Role je ve většině případů pojmenovaná skupina uživatelů, např. manažeři, čtenáři, registrovaní uživatelé, fakturantky, redaktoři apod. Správce systému potom většinou přiřazuje jednotlivé uživatele do rolí, takže například do role *redaktoři* přiřadí uživatele *karel* a *josef*.

Role jsou většinou definovány v jedné databázové tabulce a v druhé databázové tabulce je pak přiřazení uživatelů do těchto rolí. Nejinak je tomu i ve výchozí používané databázi *ASPNETDB.MDF*, kterou ASP.NET používá jako standardní možnost (kdo nevěří, může se do databáze podívat). Abychom nemuseli vkládat data do tabulky ručně, je práce s rolemi jednou z možností konfiguračního nástroje, který jsme již jednou použili (v nabídce *Website/ASP.NET Configuration*). V záložce *Security* stačí kliknout na volbě *Enable roles*, čímž se do konfiguračního souboru přidá řádek:

<roleManager enabled="true" />

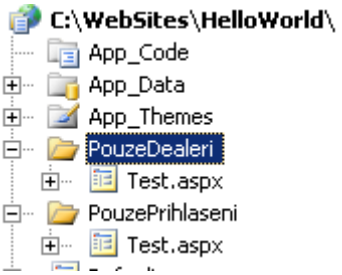
Zároveň se v konfiguračním nástroji objeví možnost *Create or Manage Roles*, pomocí které lze vytvářet a mazat role a určovat jejich členy. Zvolíme ji tedy a jako název nové role zadáme *Dealeři* (nejedná se o prodej drog!) a stiskneme *Add Role*. Po založení role se vedle ní zpřístupní možnost *Manage*, kterou zvolíme. Prokliknutím *All* se zobrazí seznam všech uživatelů, kde zaškrtneme pouze uživatele *marie* – viz obrázek:



Takže v roli *Dealeři* bude pouze *marie*, ale nikoliv *karel*.

Nastavení oprávnění

Po vytvoření rolí zbývá nastavit oprávnění pro přístup ke stránkám, přesněji řečeno složkám, ve kterých se stránky nachází. Zvolíme jednoduchý případ. Založíme ve struktuře webu novou složku *PouzePrihlaseni*, v níž založíme stránku *Test.aspx*, a dále složku *PouzeDealeri*, v níž založíme stránku *Test.aspx*. Obě stránky založíme na naší master stránce *Sablona.master* a do hlavního regionu (prvku *Content1*) stránek vepíšeme libovolné texty (např. *Tato stránka je určena pouze pro dealery*)

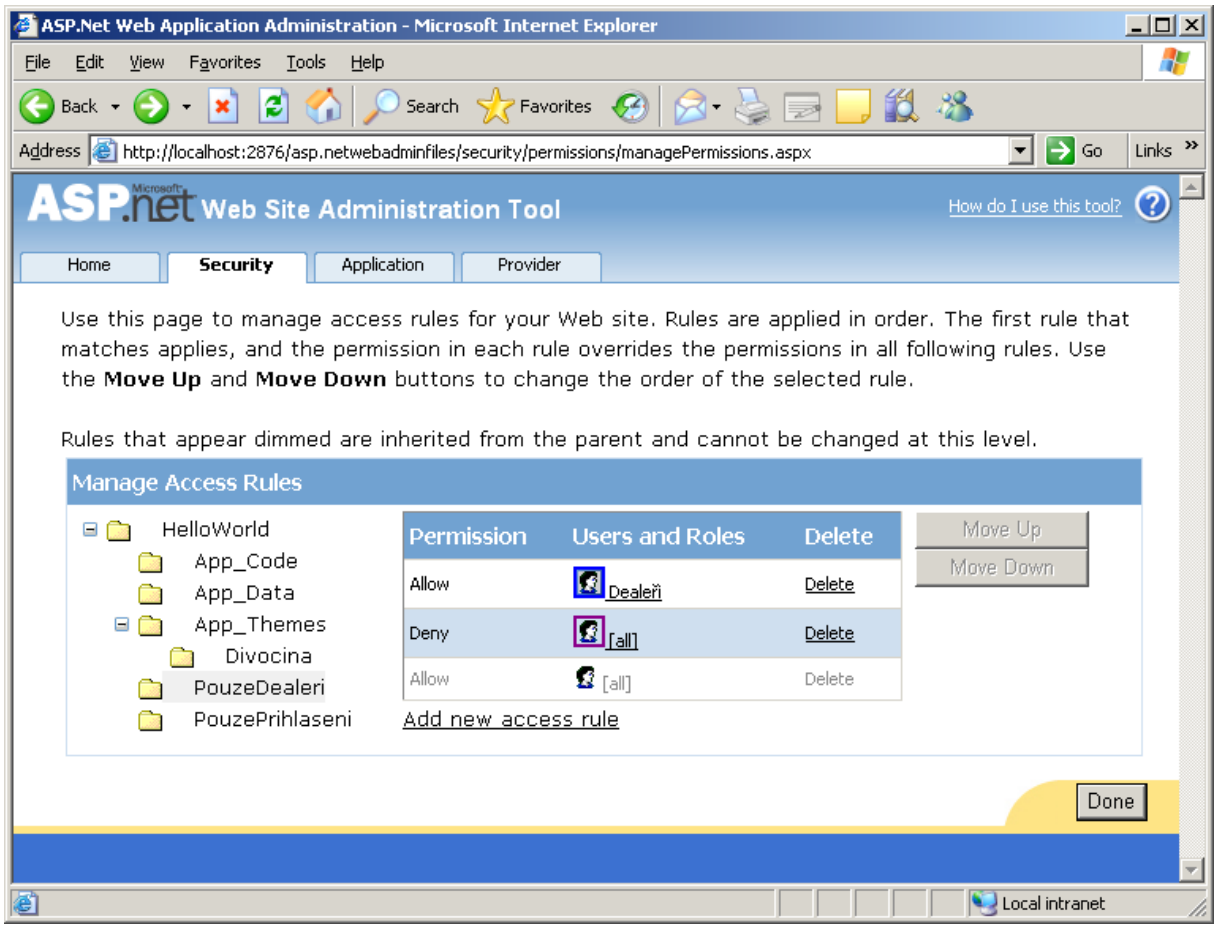


Jak asi tušíte, ke stránkám ve složce *PouzePrihlaseni* budou mít přístup pouze přihlášení uživatelé (ale nikoliv nepřihlášený uživatel). Ke stránkám ve složce *PouzeDealeri* budou mít (překvapivě) přístup pouze uživatelé v roli *Dealeři*.

Nastavení provedeme opět pomocí přístupových pravidel prostřednictvím konfiguračního nástroje ASP.NET pomocí volby *Create access rules* na záložce *Security*, kterou provedeme celkem 3x:

Select a directory for this rule	***Rule applies to***	***Permission***
PouzePrihlaseni	*Anonymous users*	*Deny*
PouzeDealeri	*Role: Dealeři*	*Allow*
PouzeDealeri	*All users*	*Deny*

Pro prohlížení nastavených práv můžeme použít volbu *Manage Access Rules*, kterou najdeme rovněž v administrativním nástroji na záložce *Security*:



Pokud nyní v okně *Solution Explorer* zvolíme *Refresh*, můžeme si povšimnout, že v těchto složkách vznikly nové konfigurační soubory *web.config*, které vytvořil konfigurační web. Můžeme si je otevřít, prohlédnout a případně upravit (? znamená anonymního uživatele, * znamená všechny uživatele).

Zabezpečení stránek si můžeme vyzkoušet v prohlížeči. Pokusíme-li se zobrazit stránku ze složky *PouzePrihlaseni* aniž bychom byli přihlášeni, jsme automaticky přesměrováni na stránku *Prihlaseni.aspx*. Pokud zadáme jakékoliv platné přihlašovací údaje (*karel* nebo *marie*), jsme přesměrování zpátky a zobrazí se požadovaná stránka. Podobně bude vypadat pokus o zobrazení stránky ze složky *PouzeDealeri*, která se ovšem zobrazí pouze po zadání účtu patřící do role *Dealeři* (v našem případě pouze *marie*).

Závěrem

Vyzkoušeli jsme si zabezpečení webu, které sestává ze tří kroků. V prvním jsme zajistili autentizaci uživatelů – v našem případě jménem a heslem. Ve druhém jsme pro snazší správu vytvořili role a přiřadili uživatele do těchto rolí. Ve třetím kroku jsme vytvořili pravidla zpřístupňující nebo zabraňující přístupu jednotlivých rolí ke složkám webu.

A opravdu již naposledy – pokud něco nebude fungovat jak by mělo, zkuste využít možnosti podpory zmíněné ve druhé lekci, zejména diskusní skupinu *microsoft.public.cs.developer*.

10. KDE ZÍSKAT DALŠÍ INFORMACE

Po seznámení s autorizací a autentizací nás dnes čeká poslední lekce, která bude zcela netypická. Místo výkladu nějakého konkrétního tématu je pouhým přehledem zdrojů a odkazů, které se nám při vývoji ASP.NET aplikací mohou hodit.

Seznam zdrojů

Základní weby firmy Microsoft

www.asp.net

Oficiální web technologie ASP.NET. Najdete zde články, software ke stažení, galerii ovládacích prvků, vysoce specializovaná diskusní fóra, návody, starter kity apod. Zajímavé jsou zejména:

- Tutoriály[27] – průvodce nejčastějšími úlohami
- Galerie ovládacích prvků[28] – v různé kvalitě, zdarma i za peníze...

ASP.NET centrum na MSDN[29]

Microsoft Developer Network (MSDN) je primárním zdrojem informací vývojářů na platformě Microsoft. Jedno z vývojářských center je věnováno ASP.NET. Například seznam všech dostupných článků na téma ASP.NET lze najít zde[30].

Český web MSDN[31]

Na českém webu MSDN můžete najít zajímavé materiály v češtině nebo slovenštině, lokální nabídky knih, brožur, DVD, soutěže, kalendář akcí, prezentace z již proběhnuvších akcí apod.

Možnosti podpory

Přehled dostupných možností podpory[32]

Přehled všech možností dostupných v ČR/SR – od bezplatných komunitních skupin až po placenou profesionální podporu.

Znalostní databáze (Knowledge Base)[33]

Samoobslužné vyhledávání v databázi známých problémů je vhodné zejména pro řešení dotazů typu „nefunguje mi...". Vyhledává se zpravidla podle chybového hlášení. Databáze zahrnuje všechny podporované produkty.

Profesionální podpora pro MSDN předplatitele a certifikované partnery[34]

Máte-li aktivní MSDN předplatné anebo pracujete-li pro certifikovaného partnera firmy Microsoft, můžete na problémy typu „nefunguje mi..." využít profesionální podporu v podobě incidentů pro řešení problémů, jež máte k dispozici.

Diskusní skupina[35] *microsoft.public.cs.developer*

---

27 http://www.asp.net/Tutorials/quickstart.aspx
28 http://www.asp.net/default.aspx?tabindex=2&tabid=31
29 http://msdn.microsoft.com/asp.net/
30 http://msdn.microsoft.com/asp.net/archive/default.aspx
31 http://msdn.microsoft.cz/
32 http://msdn.microsoft.cz/support/
33 http://support.microsoft.com/search/?adv=1
34 http://msdn.microsoft.cz/support/msdn.asp
35 http://support.microsoft.com/newsgroups/newsReader.aspx?lang=cs&cr=CZ&dg=microsoft.public.cs.developer&sloc=cs

Bezplatná podpora v diskusní skupině monitorované Microsoftem s.r.o. v České republice je vhodná zejména pro dotazy „jak na to“. Je přístupná následujícími způsoby:

- Přes newsreader (NNTP)[36]
- Ve webovém prohlížeči[37]

MSDN library[38]

Samoobslužné vyhledávání v základním zdroji obsahujícím řadu článků týkajících se vývoje s použitím Microsoft produktů. Vhodná zejména pro dotazy typu „jak na to“.

ASP.NET fóra[39]

Máte-li specializovanější dotaz, můžete zkusit fóra ASP.NET v angličtině. Jejich výhodou je vysoká specializace. Do fór často odpovídají přímo členové produktového týmu z Redmondu.

Starter Kity

www.asp.net

Seznam dostupných anglických starter kitů je dostupný na oficiálním webu www.asp.net, například je zde zařazen DotNetNuke starter kit pro snadné vytváření webových portálů.

starterkits.aspnet.cz

V době psaní textu zde byly k dispozici 4 české Starter Kity, z toho 2 lokalizace anglických starter kitů a 2 starter kity od Michala Valáška.

Knihy a vzdělávání

Elektronické PDF brožury v češtině[40]

Řada elektronických PDF brožur zdarma ke stažení. Jednou z brožur je i vývoj ASP.NET 2.0 aplikací pomocí Express nástroje (ve slovenštině).

Multimediální prezentace v češtině a slovenštině[41]

Multimediální prezentace v češtině/slovenštině s tématy SQL Serveru 2005 a Visual Studia 2005. Mezi nimi najdete i dvě prezentace s tématem ASP.NET 2.0 v češtině.

Anglické knihy o ASP.NET[42]

Nejúplnější soubor knih o ASP.NET v angličtině lze získat na webu Amazon, stačí zadat ve vyhledávání „ASP.NET“. Připravte se ovšem na ceny v dolarech a poplatky za poštovné.

Knihy v Computer Pressu[43]

V Computer Pressu si lze objednat mnohé z anglických knih bez nutnosti platit drahé poštovné. Mnohé knihy jsou přeloženy do češtiny a začínají se již objevovat i původní české knihy o ASP.NET 2.0, například tato[44].

36 news://news.microsoft.com/microsoft.public.cs.developer
37 http://support.microsoft.com/newsgroups/newsReader.aspx?lang=cs&cr=CZ&dg=microsoft.public.cs.developer&sloc=cs
38 http://msdn.microsoft.com/library/
39 http://forums.asp.net/
40 https://s.microsoft.com/cze/msdn/connection/default.mspx
41 http://msdn.microsoft.cz/webcasts/
42 http://www.amazon.com
43 http://knihy.cpress.cz/pocitac/
44 http://knihy.cpress.cz/pocitac/Book.asp?ID=1792

Komunitní weby

www.aspnet.cz

Český web Michala Valáška je plně věnovaný ASP.NET technologii.

www.vyvojar.cz

Komunitní web vývojářů na platformě .NET provozovaný Michalem Bláhou.

www.aspnet.sk

Slovenský komunitní web věnovaný hlavně ASP.NET, který vytvořil Igor Stanek.

www.asp101.com

Web v angličtině je užitečný především velkým množstvím fragmentů kódu a řešených úloh.

www.15seconds.com

Web 15seconds je zajímavým zdrojem informací pro vývojáře, nejenom k tématu ASP.NET, ale i k Windows Vista a dalším tématům.

www.aspfree.com

Web se výrazně zaměřuje na ASP.NET, ale přináší i články o souvisejících technologiích na platformě Microsoft.

aspnet.4guysfromrolla.com

Přestože většina článků se zabývá ASP.NET verzí 1.0 nebo 1.1, najdete zde celou řadu odkazů na články o různých tématech. Výhodou je přehledné řazení.

Free .NET hosting v ČR/SR

V tuto chvíli si můžete vyzkoušet .NET verze 2.0 zdarma u těchto hosterů:

http://www.aspweb.cz/

http://www.asp2.cz

http://www.qsh.sk/

http://www.aspx.sk.

Závěrem

ASP.NET 2.0 je natolik široká oblast, že deset lekcí kurzu pokryje pouze velmi úzkou část. Tento kurz by klidně mohl mít i sto lekcí, a stále by bylo o čem psát. Psaní dalších 90 lekcí bohužel není v našich časových možnostech, a zřejmě by nebylo ani příliš efektivní. Na webech uvedených v dnešní lekci najdete více než dostatek informací pro další pokračování a vývoj v ASP.NET 2.0, v čemž vám přejeme hodně štěstí.