

Wydawnictwo Helion ul. Kościuszki 1c 44-100 Gliwice tel. 032 230 98 63 e-mail: helion@helion.pl



# Technologie ASP.NET i ADO.NET w Visual Web Developer

ion.nl

Autor: Jacek Matulewski, Sławomir Orłowski ISBN: 978-83-246-0738-9 Format: B5, stron: 296



Firma bez własnej witryn internetowych to obecnie rzadkość, a wiele aplikacji sieciowych korzysta z baz danych. Za pośrednictwem stron internetowych klienci mogą na przykład zamawiać usługi i kupować produkty oraz przeglądać katalogi z ofertą sklepu, a firmy mogą między innymi zarządzać magazynem czy danymi klientów. Technologia ASP.NET umożliwia błyskawiczne tworzenie rozbudowanych i nowoczesnych witryn internetowych, a dzięki ADO.NET można szybko dodać do nich obsługę baz danych. Bezpłatne środowisko Visual Web Developer Express pozwala każdemu przekonać się o ogromnych możliwościach tych narzędzi.

Książka "Technologie ASP.NET i ADO.NET w Visual Web Developer" zawiera wszechstronny opis technik i narzędzi do tworzenia zaawansowanych witryn internetowych za pomocą ASP.NET i ADO.NET. Dzięki niej nauczysz się korzystać z tych technologii w środowisku Visual Web Developer oraz używać podstawowych języków do programowania stron ASP.NET. Dowiesz się, jak wygodnie zarządzać bazami danych w aplikacjach sieciowych i jak sprawnie umieścić gotową witrynę na serwerze. Przykładowe studia przypadku pokazują, jak zastosować omawiane technologie w praktyce, a w części poświęconej AJAX-owi zobaczysz, jak przyspieszyć działanie swych witryn.

- Wprowadzenie do środowiska Visual Web Developer
- · Korzystanie z języków C# i XML w projektach ASP.NET
- · Zapewnianie spójnego stylu za pomocą wzorców stron
- Zarządzanie sesjami i plikami cookie
- Korzystanie z ADO.NET do obsługi baz SQL Server 2005, Access i XML
- Tworzenie i używanie usług sieciowych
- · Budowanie kompletnych rozwiązań sieciowych

2.123 C

- Przyspieszanie witryn za pomocą AJAX-a
- Dodawanie do witryny kontrolek AJAX-a

Wypróbuj ASP.NET wraz z ADO.NET i przekonaj się, że tworzenie rozbudowanych witryn internetowych z obsługą baz danych nie musi być trudne.

# Spis treści

	Wstęp, w którym namówimy Czytelnika na rendez-vous z ASP.NET 7			
Część I	ASP.NET i ADO.NET	9		
Rozdział 1.	ASP.NET i Visual Web Developer. Szybki start	11		
	Tworzenie projektu pierwszej strony ASP.NET	12		
	Projektowanie interfejsu strony	13		
	Edycja kodu w pliku . <i>aspx</i>	16		
	Programowanie silnika strony ASP.NET	17		
	Walidacja danych	19		
Rozdział 2.	Wieża Babel, czyli języki wykorzystywane w projektach ASP.NET	23		
	Język C#	24		
	Podstawowe typy danych	24		
	Sterowanie przepływem	31		
	Zwracanie wartości przez argument metody	34		
	Wyjątki	36		
		37		
	Język AML	44		
	Poustawy języka AIVIL Zanis danych do nliku XMI	44 /6		
	Odezyt danych z pliku XML o znanej strukturze	40		
	Analiza i odczyt pliku XML o nieznanej strukturze	40		
Rozdział 3.	Nowoczesna i funkcionalna strona ASP.NET	53		
	Rozmieszczenie kontrolek na stronie	53		
	Pozycjonowanie elementów serwisu za pomocą tabeli	56		
	Pozycjonowanie bezwzględne	63		
	Witryna wielojęzyczna	66		
	Zasoby lokalne	67		
	Zasoby globalne	73		
	Samodzielne wybieranie wersji językowej	76		
	Komponent MultiView	79		
	Wyświetlanie banerów reklamowych	82		
	Buforowanie zwracanych stron	83		

Rozdział 4.	Mechanizmy spajające witrynę ASP.NET	
	Wzorzec strony (master page)	87
	Przygotowywanie projektu	87
	Definiowanie wzorca	88
	Strona korzystająca ze wzorca	90
	Site map i komponent SiteMapPath	90
	Menu strony	
	Drzewo pokazujące strukture stron w witrynie	
	Dwa słowa na temat kaskadowych arkuszy stylów	
	Tworzenie arkuszy stylów	95
	Stosowanie arkuszy stylów	97
	Czas życja sesij i aplikacji ASP NET	98
	Sesia i dane sesii	99
	Pliki cookie	101
	Dane anlikacij	
		405
Rozdział 5.	WSpołpraca ASP.NET I ADO.NET. SQL Server 2005 I Access	105
	Moc ADU.NE1	105
	SOL Server 2005	100
	SQL Server 2005	107
	Tworzymy bazę danych na serwerze SQL Server 2005	10/
	Tworzymy tobalo w borie donych	109
	I worzymy tabelę w bazie danych	109
	Koniigurujemy kontroikę reprezentującą tabelę w aplikacji	111
	MICrosoft Access	
	I worzenie bazy danych Access	
	Dołączanie pliku bazy Access do projektu	
	Konfigurujemy kontrolkę reprezentującą tabelę w aplikacji	
	Bardzo krotki wstęp do języka SQL	
	Modyfikacje danych z poziomu aplikacji ASP.NET	
	Dodawanie adresu e-mail do listy	
	Walidacja danych	120
	Usuwanie adresu z listy	
	Podgląd listy adresów w siatce	122
Rozdział 6.	Współpraca ASP.NET i ADO.NET. Bazy danych XML	125
	Projekt	
	Plik XML	
	Konfiguracja objektu tvpu XmlDataSource	
	Edvcja pliku XML za pomoca klasv XmlDocument	
	Walidacia danych formularza	
	Zabezpieczenie przed podwóinym dodaniem	
	Usuwanie elementu	130
	Prezentacja tabeli w siatce	
D	lite stands with more start	400
Rozuział 7.	Daostępnianie witryny w sieci	<b>133</b>
	Dostawca usiug Przenoszenie witryny na serwer	133 134
	Tizenoszeme widyny na serwer	
Rozdział 8.	Wzorce projektowe ASP.NET	139
	Ping	
	Księga gości	141
	Prezentowanie danych umieszczonych w pliku XML	
	Walidacja wpisywanych danych	
	Formularz wysyłający wiadomości e-mail	
	Wiadomosc z załącznikami. Kontrolka FileUpload	151

	Identyfikacja i autoryzacja użytkowników	154
	Formularz rejestrujący nowych użytkowników	159
	Kontrolka LoginView	160
Rozdział 9.	Usługi sieciowe	163
Norarian or	Dierwsza usługa sieciowa	164
	Użycie usługi sieciowej	
Część II	Studia przypadków	171
Rozdział 10	). Studium przypadku: konferencja	173
	Część statyczna witryny (pliki .aspx)	174
	Szablon	174
	Strona główna	175
	Formularz	175
	Walidacja danych w formularzu	177
	Pozostałe strony formularza	178
	Dynamika witryny (pliki .cs)	182
	Struktury danych	182
	Przechowywanie danych sesji	183
	Powrót do edycji danych	185
	Zabezpieczenie przed jawnym wybraniem strony formularza w przeglądarce	187
	Zapis do bazy danych	188
	Szkodliwy znak apostrofu (')	191
	Wysłanie listu z potwierdzeniem zgłoszenia	193
	Czyszczenie danych	194
	Obsługa błędów	195
	Dynamiczna lista zgłoszonych wykładów	197
	Logowanie i edycja wcześniej zgłoszonych danych	197
	Wysyłanie listu z zapomnianym hasłem	203
	Uwierzytelnianie z użyciem mechanizmu Forms	203
	Estetyka witryny (pliki .css)	209
	Tworzenie tematu	209
	Formatowanie elementów HTML	210
	Formatowanie poszczególnych obiektów	212
	Klasy stylów	
	Plik "skórek"	215
	Możliwe drogi rozwoju witryny	
	Przeniesienie witryny na serwer IIS	217
Rozdział 11	Sklen internetowy	219
	Szahlon strony	219
	Baza danych	221
	Strong główna	224
	Koszyk	227
	Rejestracja użytkowników	232
	Zamawianie	
Część III	Podstawy AJAX dla ASP.NET	241
Rozdział 12	2. Cześciowa aktualizacia strony	243
	Kontrolka UndateProgress	243
	Wyzwalanie częściowej aktualizacji przez komponent spoza UndatePanel	245
	Svgnalizowanie częściowej aktualizacji	

	Cykliczne wyzwalanie aktualizacji	
	Aktualizacja warunkowa	
Rozdział 13	. Migracja aplikacji ASP.NET do AJAX ASP.NET	249
Rozdział 14	. AJAX Control Toolkit	253
	Przegląd ACT	253
	Używanie kontrolek ACT we własnych projektach	255
	Instalacja kontrolek ACT w środowisku VWD	255
	Użycie rozszerzenia ConfirmButtonExtender	256
	Jak oni to zrobili?	257
	Suwaki	258
	Reklama	
	Akordeon	
Część IV	Dodatki	265
Dodatek A	Instalacja i konfigurowanie serwera IIS w systemie Windows	267
	Instalacja IIS i uruchamianie jego usług	
	Rejestrowanie ASP.NET 2.0 w IIS	270
Dodatek B	Instalacja protokołu SSL dla serwera IIS	273
Dodatek C	Uruchamianie projektów ASP.NET 2.0 w platformie Mono	277
Dodatek D	Dodatkowe projekty	283
	Skorowidz	285

# Rozdział 4. **Mechanizmy spajające** witrynę ASP.NET

Jacek Matulewski

# Wzorzec strony (master page)

W przypadku witryny zawierającej kilka stron warto posłużyć się wzorcem. **Wzorzec** (ang. *master page*) to zwykła strona ASP.NET zapisana do pliku z rozszerzeniem *.master*. Wyróżnia ją jednak to, że zawiera komponenty ContentPlaceHolder, które rezerwują miejsce do wypełnienia przez strony korzystające ze wzorca. Poza tym wzorzec może zawierać zwykłe elementy HTML, jak i komponenty ASP.NET.

Wzorzec służy jako szablon pozostałych stron projektu. Tworząc nowe strony projektu, możemy wskazać ich wzorzec, a wówczas w widoku projektowania, zamiast edytować całą stronę, będziemy edytować jedynie te miejsca, które we wzorcu zostały zarezerwowane komponentami ContentPlaceHolder.

W najprostszym przypadku można posłużyć się wzorcem do ujednolicenia nagłówków i stopek wszystkich stron witryny — wówczas wzorzec zawiera tylko jeden komponent ContentPlaceHolder. I właśnie na takim przykładzie nauczymy się teraz tworzenia wzorców i korzystania z nich.

### Przygotowywanie projektu

Nie można, a przynajmniej nie jest to proste, przypisać strony master od istniejącej witryny ASP.NET. Dlatego utworzymy zupełnie nowy projekt — Czytelnik odpocznie wreszcie od *Kolorów* — w którym pierwszą czynnością będzie usunięcie domyślnie utwo-rzonej strony *Default.aspx*.

- 1. Tworzymy nowy projekt:
  - a) z menu File wybieramy New Web Site ...,
  - b) zaznaczamy pozycję ASP.NET Web Site,
  - c) w rozwijanej liście Location wybieramy File System (wartość domyślna),
  - d) a w rozwijanej liście Language Visual C#,
  - e) klikamy OK.
- 2. Z projektu usuwamy stronę *Default.aspx*:
  - a) zaznaczamy ją w oknie projektu (podokno o nazwie Solution Explorer),
  - b) rozwijamy menu kontekstowe
  - c) i wybieramy z niego polecenie Delete;
  - d) pojawi się pytanie o potwierdzenie, w którym należy kliknąć przycisk OK.

To usunie plik strony nie tylko z projektu, ale także z dysku. W tym miejscu umieścimy nową wersję strony o nazwie *Default.aspx*, ale korzystającą ze wzorca. Wcześniej musimy oczywiście przygotować wzorzec. Nie będziemy w tym zbyt wymyślni — zdefiniujemy prosty nagłówek oraz stopkę strony i zadowolimy się jednym komponentem ContentPlaceHolder.

Uwaga

Nie należy mylić usuwania pliku (także z dysku), a więc polecenia *Delete*, z usuwaniem pliku z projektu, tj. z poleceniem *Exclude From Project*.

#### Definiowanie wzorca

- W oknie projektu zaznaczamy główną gałąź reprezentującą projekt całej witryny (a nie katalog App\_Data).
- 2. Z menu File wybieramy New File ....
- 3. W oknie Add New Item (rysunek 4.1) zaznaczamy pozycję Master Page.
- **4.** Klikamy *Add*. Do projektu dodany zostanie nowy plik *MasterPage.master*. W edytorze zobaczymy jego kod.
- 5. Przejdźmy od razu do widoku projektowania nowej strony (zakładka Design w dole okna). Zobaczymy na niej tylko jeden komponent klasy ContentPlaceHolder to jest miejsce, które będzie zapełniane przez strony korzystające z naszego wzorca. Jeżeli chcemy dodać ich więcej, to komponent ten znajdziemy na zakładce Standard.
- **6.** Umieśćmy powyżej i poniżej tego komponentu jakiś tekst pełniący rolę nagłówka i stopki stron naszej witryny. Przykład widoczny jest na rysunku 4.2.

Add New Item - 7	F:\Wydawnictwa\Wyda	awnictwo Helion\w	przygotowaniu\Visua	l Web Developer	\zrodla\R3_Wzorce\	? ×
Templates:						00 00 00 00 00 00 00
Visual Studio	installed templates					
Web Form HTML Page Class Keb Configu Resource Fill Generic Han YBScript File Mobile Web Browser File My Templates Search Onlin	uration File le dler e Configuration File e Templates	Master Page WCF Service Style Sheet Style Sheet SOL Database SOL Database Site Map JScript File		<ul> <li>Web User Cont</li> <li>Web Service</li> <li>Global Applica</li> <li>Text File</li> <li>DataSet</li> <li>Mobile Web Fice</li> <li>Mobile Web Us</li> <li>Skin File</li> </ul>	rol lion Class rm er Control	
A Master Page for	Web Applications					
<u>N</u> ame:	MasterPage.master					
Language:	Visual C#	•	☐ Place code in sepa ☐ Select master page	ırate file		
					Add	Cancel

**Rysunek 4.1.** *Polecenie dodawania pliku jest czułe na zaznaczoną pozycję w oknie projektu — aby zobaczyć wszystkie możliwe rodzaje plików, należy zaznaczyć pozycję odpowiadającą całemu projektowi* 



Rysunek 4.2. Osadzanie komponentu ContentPlaceHolder we wzorcu witryny

#### Strona korzystająca ze wzorca

Nasz wzorzec jest prosty, żeby nie powiedzieć prymitywny, ale nie o estetykę teraz chodzi, a o ideę. Stworzymy zatem przykładowe strony, które będą z tego wzorca korzystać.

- 1. Z menu File wybieramy ponownie pozycję New File.
- 2. Tym razem zaznaczamy ikonę Web Form.
- **3.** Koniecznie musimy zaznaczyć pole opcji *Select master page*. Tylko w momencie tworzenia strony można wskazać jej wzorzec.
- **4.** Zalecam również zaznaczenie opcji *Place code in separate file*, dzięki której ewentualne metody zdarzeniowe będą umieszczone w osobnym pliku.
- 5. Musimy wskazać jeszcze nazwę pliku strony domyślnie jest to *Default.aspx* — oraz język użyty do programowania metod zdarzeniowych. Jak już się pewnie Czytelnik zorientował, w tej książce zalecamy używanie C#.
- 6. Wreszcie klikamy Add.
- 7. Natychmiast pojawi się okno Select a Master Page. Wskazujemy w nim stronę MasterPage.master i klikamy OK.

Po utworzeniu strony znajdziemy się w edytorze kodu. Przejdźmy niezwłocznie do widoku projektowania. Zobaczymy w nim stronę wzorca, ale poza obszarem wyznaczonym przez komponent ContentPlaceHolder jest ona niedostępna do edycji (rysunek 4.3). Miejscem, w którym możemy umieszczać nasze komponenty, jest wyłącznie miejsce zarezerwowane wcześniej we wzorcu.

Wypełnijmy miejsce przeznaczone na stronę jakąś przykładową zawartością. Umieśćmy w nim np. komponent HyperLink. Za pomocą okna *Properties* do jego właściwości ImageUrl przypiszmy adres *http://helion.pl//img/logo162\_35.gif*, natomiast do właściwości NavigateUrl adres *http://helion.pl.* Na podglądzie powinniśmy zobaczyć natych-miast logo Wydawnictwa Helion (por. rysunek 4.3). Możemy stworzyć teraz kolejne strony korzystające z tego samego wzorca, który ujednolici ich wygląd. Wzorzec po-prawia zatem spójność całej witryny. Nie do przecenienia jest fakt, że stopkę i nagłówek wszystkich stron witryny kontrolujemy z jednego pliku, zatem jeżeli chcemy coś do nich dodać lub je zmienić, wystarczy edytować tylko plik wzorca.

# Site map i komponent SiteMapPath

Pozostańmy przy tym samym projekcie, a nawet dorzućmy do niego jeszcze kilka stron. Jeżeli witryna ma więcej stron (mowa o kilkunastu, kilkudziesięciu), warto pomyśleć o *site map* — mapie witryny. W ASP.NET przygotowanie takiej mapy polega na utworzeniu pliku XML o nazwie *Web.sitemap*, w którym znajduje się zhierarchizowana grupa elementów siteMapNode. W atrybutach każdego z nich wskazujemy adres strony z witryny, jej tytuł i ewentualnie opis. Struktura znaczników ma odzwierciedlać logiczną strukturę strony, wskazując strony nadrzędne i ich podstrony. Ilość stopni hierarchii jest w zasadzie dowolna.



**Rysunek 4.3.** Edycja stron korzystających ze wzorca ogranicza się do obszarów wyznaczonych na wzorcu przez komponent ContentPlaceHolder

W naszym projekcie jest tylko kilka stron (załóżmy, że trzy: *Default.aspx*, *Default2. aspx* i *Default3.aspx*), ale i my zdefiniujemy plik *Web.sitemap*. Przyjmijmy, że *Default1. aspx* jest stroną tytułową jakiegoś działu witryny o nazwie "Łącza do ważnych stron", a *Default2.aspx* i *Default3.aspx* są zwykłymi stronami tego działu. Z menu *File* wybieramy pozycję *New File...* i w oknie *Add New Item* wskazujemy pozycję *Site Map*. Powstanie plik *Web.sitemap*. Uzupełniamy go według wzoru z listingu 4.1.

**Listing 4.1.** *Plik Web.sitemap to plik XML opisujący logiczną strukturę witryny ASP.NET na potrzeby komponentów nawigacyjnych* 



W elemencie siteMap może być tylko jeden element siteMapNode, więc ewentualną rozbudowę powyższej struktury należy zacząć od trzeciego poziomu zagnieżdżenia elementów XML.

Plik *Web.sitemap* może być źródłem danych dla komponentów umieszczanych na stronach, które pozwalać będą internaucie na zorientowanie się w pozycji oglądanej strony w strukturze całej witryny i szybkie przejście do innych jej stron. Na początek przyjrzyjmy się komponentowi SiteMapPath.

- 1. Przejdźmy do widoku projektowania wzorca MasterPage.master.
- **2.** Umieśćmy na nim komponent SiteMapPath z zakładki *Navigation*. Komponent ten pokazuje ścieżkę aktualnej strony w strukturze zdefiniowanej w pliku *Web.sitemap* (rysunek 4.4).



**3.** Zwróćmy uwagę na mały trójkącik widoczny po prawej stronie górnej krawędzi nowego komponentu, jeżeli ten jest zaznaczony. Jeżeli go klikniemy, pojawi się lista typowych zadań dotyczących tego komponentu (rysunek 4.5). W tym przypadku składa się ona z dwóch pozycji *Auto Format* oraz *Edit Templates*. Pierwsze polecenie służy do niemal automatycznego konfigurowania wyglądu komponentu. Wybierzmy np. szablon *Colorful*.

# Menu strony

Innym zastosowaniem mapy witryny jest automatycznie tworzone menu i drzewo zawierające strony uwzględnione w tym pliku. Zacznijmy od menu. Dodamy je do wzorca — tym razem ponad komponentem rezerwującym miejsce dla stron.

- 1. Przechodzimy do widoku projektowania wzorca MasterPage.master.
- **2.** Ponad komponentem ContentPlaceHolder umieszczamy komponent Menu z zakładki *Navigation*.

🖳 R04_Wzorce - Visual W	eb Developer 2005 Express Edition		<u> </u>
<u>F</u> ile <u>E</u> dit ⊻iew Web <u>s</u> ite	e <u>B</u> uild <u>D</u> ebug F <u>o</u> rmat Layout <u>T</u> ools <u>W</u> indow <u>C</u> ommunity <u>H</u> elp		
🐻 • 🗃 • 💕 🖬 🕼	👗 🗈 🛍 🗠 🕫 - 🖓 - 🖳 - 🖳 🕨 💽 💋 timeout	• -	
- Verdana	• 0.8em • B I U A 2 = • = = = 0.5		
Toolbox 🗸 🕈 🗙	MasterPage.master Default.aspx.cs Global.asax Web.Config	₹×	Solution Explorer 🗸 🕂 🗙
🗆 Standard 🔶			
Revinter	E ContantDiacol Ioldor - ContantDiacol Ioldor 1		T:\\B04 Wzorce\
A Label	Content-laceHolder - ContentPlaceHolder1		App_Data
abl TextBox			🖃 🛅 Default.aspx
ab Button			Default.aspx.cs
(ab) LinkButton			Default2.aspx
ImageButton			Web Config
A HyperLink			Web.sitemap
∎ DropDownList			
🛋 ListBox 🗕			Solution Explorer Database Exp
CheckBox			Properties - 7 ×
E CheckBoxList			SiteMapPath1 System.Web.UI.Wet -
③ RadioButton			2↓ ■ 4
§⊟ RadioButtonList		- II	
📓 Image	Root Node : Parent Node : Current Node		(ID) SiteMapPath1
📓 ImageMap	E di Temelata		□ Styles 🗾
Table	(c) Jacek Matulewski 2007		Auto Format
i∃ BulletedList		_	((D)
HiddenField	4	<u>×</u>	Programmatic name of the control.
ILiteral	Generation         Genera	Þ	
Ready			li

**Rysunek 4.5.** *Podręczna lista zadań to zbiór najczęściej wykorzystywanych kreatorów związanych z komponentem* 

- **3.** W liście podręcznej z rozwijanej listy *Choose Data Source* wybieramy <*New data source*...>.
- 4. Pojawi się kreator Data Source Configuration Wizard pozwalający na wybór źródła danych, na podstawie których utworzone zostanie menu. Może ono zostać zbudowane w oparciu o dowolny plik XML lub nasz gotowy plik Web.sitemap. Wybieramy oczywiście tę drugą możliwość, zaznaczając ikonę Site Map. Klikamy OK. Utworzony zostanie komponent SiteMapDataSource1.
- **5.** Nam pozostaje tylko sformatować wygląd menu. Proponuję również tym razem wybrać szablon *Colorful* (rysunek 4.6).

#### Rysunek 4.6.

Użycie tego samego stylu do formatowania menu i ścieżki pokazującej pozycję w strukturze witryny to zalążek estetycznego i spójnego wizerunku wszystkich stron witryny

🚰 Untitled Page - Microsoft Internet	Explorer			
Plik Edycja Widok Ulubione Nar:	zędzia Pomoc 🛛 🦉			
🕲 Wstecz 🔻 🕑 👻 📓 🏠	🖓 Wyszukaj 🥎 Ulubione 😧 🔗 🔹 👋			
Adres 🗃 http://localhost:3743/R4_Wzon	ce/Default2.aspx 🔹 💽 Przejdź 🛛 Łącza			
Google ▼	ukaj w Sieci 🔍 Przeszukaj Serwis 🛛 🎴 🍟			
Przykładowa witryna korzystają	ce ze wzorca			
ASP.net	Link do MSDN Link do strony dokumentacji MSDN2			
Nazwa witryny : <u>Linki do stron</u> : Link do ASP.NET				
(c) Jacek Matulewski 2007	-			
🖨 http://localhost:3743/	Substance Lokalny intranet			

### Drzewo pokazujące strukturę stron w witrynie

Na zakładce *Navigation* jest jeszcze jeden komponent, na który też warto zwrócić uwagę. Jest to drzewo TreeView, które prezentuje strukturę witryny. Nadaje się bardziej na osobną stronę niż do umieszczenia w nagłówku lub stopce stron.

- 1. Z menu File wybieramy New File.
- 2. W oknie Add New Item zaznaczamy Web Form, wybierając opcję Select master page, i zmieniamy nazwę pliku na MapaWitryny.aspx.
- 3. Klikamy Add. W nowym oknie wybieramy wzorzec i klikamy OK.
- 4. Przechodzimy do widoku projektowania nowej strony.
- **5.** Na dostępnym obszarze umieszczamy komponent TreeView z zakładki *Navigation*.
- **6.** Postępując identycznie jak w przypadku menu, tworzymy źródło danych korzystające z mapy witryny (niestety, nie można użyć gotowego komponentu SiteMapDataSource1 widocznego w obszarze wzorca).
- **7.** Formatujemy drzewo, wybierając z podręcznej listy zadań pozycję *Auto Format...*. Proponuję użycie stylu *Arrows 2* (rysunek 4.7).

Rysunek 4.7.	🖉 Untitled Page - Microsoft Internet Explorer 📃 🗆 🗙
Automatycznie	Plik Edycja Widok Ulubione Narzędzia Pomoc 🛛 🖉
generowana mana witrymy	🛛 🚱 Wstecz 🔻 🕤 👻 👔 🏠 🔎 Wyszukaj 📩 Ulubione 🧭 🔗 🔻 👋
тара wiiryny	Adres 🗃 http://localhost:3743/R4_Wzorce/MapaWitryny.aspx 💌 💽 Przejdź 🛛 Łącza
	Google - 🔂 Szukaj w Sieci 🔍 Przeszukaj Serwis   PageBank 😕
	Przykładowa witryna korzystające ze wzorca
	Nazwa witryny
	Mapa witryny:
	▼ Nazwa witryny
	✓ Linki do stron
	▶ Link do ASP.NET
	Link do MSDN Linki do oficjalnej strony ASP.NET
	Nazwa witryny : Mapa witryny
	A http://localhost.3743/

**8.** Nową stronę warto dopisać do mapy witryny, tworząc węzeł równorzędny do *Default.aspx* (por. źródła dołączone do książki).



Poza zwykłą nawigacją, jaką umożliwia komponent TreeView, można definiować metody zdarzeniowe związane z kliknięciem różnych pozycji drzewa (zdarzenie SelectedNodeChanged).

# Dwa słowa na temat kaskadowych arkuszy stylów

Wiemy już, że w projektach ASP.NET możemy w znacznym stopniu, w zasadzie nawet całkowicie, odseparować kod C# od szablonu HTML strony. W ten sposób oddzielony zostaje kod odpowiedzialny za statyczny wygląd stron witryny od metod określających ich dynamikę. Do tych dwóch etapów projektowania dochodzi trzeci, w którym za pomocą kaskadowych arkuszy stylów określamy estetyczny aspekt witryny. Podobnie jak w przypadku kodu C#, także arkusze stylu mogą być całkowicie odseparowane w plikach .*css*, a przez to ich rozwój, podobnie jak kodu C#, może być z łatwością powierzony innym osobom niż rozwój kodu z plików .*aspx*.

W kilku poniższych przykładach przedstawię podstawowe narzędzia służące do budowania kaskadowych arkuszy stylów. Nieco dodatkowych przykładów znajdzie Czytelnik także w rozdziale 10.

### Tworzenie arkuszy stylów

Kaskadowe arkusze stylów (ang. *cascade style sheet*) to kolejne obok wzorca narzędzie ujednolicenia stron witryny, a jednocześnie zcentralizowania kontroli nad ich wyglądem. I w tym przypadku wsparcie ze strony VWD jest godne pochwały.

- 1. Z menu File wybieramy New File ....
- **2.** W oknie *Add New Item* zaznaczamy pozycję *Style Sheet*, jeżeli odczuwamy taką potrzebę, zmieniamy nazwę pliku i klikamy *OK*.

W edytorze zobaczymy niemal pusty plik, który zawiera jedynie tekst:

To zalążek stylu związanego ze zwykłym tekstem umieszczonym na stronie (tekst między znacznikami BODY w kodzie HTML). Na szczęście nie musimy znać się na formacie arkuszy stylów, bo VWD zawiera proste narzędzie pozwalające na ich definiowanie. Zacznijmy od rozbudowania reguły formatowania dotyczącej prostego tekstu.

- **1.** W edytorze ustawiamy kursor (edycji, nie myszy) między nawiasami istniejącej reguły stylu.
- 2. Klikamy na pasku narzędzi przycisk Build Style ....
- **3.** W oknie *Style Builder* (rysunek 4.8) możemy wybrać format i kolor czcionki, tła, list i innych elementów umieszczonych na stronie. My ograniczymy się do sformatowania czcionki, dlatego klikamy pozycję *Font* na liście zakładek widocznej z lewej strony okna.

body {
}

Style Builder - body	?	×
Style Builder - Body	Eont name  Font name  Family:  System font:  Font attributes  Color:  #cd853f   Effects  Size  Effects  Size  Ffects  Absolute:  Cabsolute:  Cabsolute: C	
	Absolute:     C Relative:     Sample text	_
	OK Cancel	

Rysunek 4.8. Definiowanie reguły stylu dla znacznika body

- **4.** W części zatytułowanej *Font attributes* klikamy przycisk z trzema kropkami pozwalający na swobodny wybór koloru:
  - a) na zakładce Named Colors odnajdujemy kolor Peru (jeden z brązowych), który będzie dobrze współgrał ze stylem formatowania wybranym w menu i innych komponentach nawigacyjnych;
  - b) odznaczamy pole opcji Use color names, aby użyć określenia koloru poprzez składowe zamiast nazwy<sup>1</sup>, i klikamy OK, aby zamknąć kreator stylu.
- **5.** Następnie korzystając z ikony *Add Style Rule* na pasku narzędzi, tworzony nowy styl dla znacznika A (tj. dla umieszczonych na stronie łączy):
  - a) tym razem wybierzmy kolor SaddleBrown (pamiętajmy o wyłączeniu opcji Use color names);
  - b) w oknie Build Style w części Effects zaznaczamy pole None.
- **6.** Po tym zdefiniujmy jeszcze jedną regułę formatowania dla A:hover (łącze po najechaniu na niego kursorem), w którym kolor ustalamy na *Orange*, a w części *Effects* (por. rysunek 4.8) włączamy opcję *Underline*.

Po tych czynnościach plik kaskadowego arkusza stylów (plik z rozszerzeniem .css) powinien wyglądać jak na listingu 4.2:

<sup>&</sup>lt;sup>1</sup> Bardziej egzotyczne nazwy kolorów mogą być niezrozumiałe dla starszych przeglądarek.

```
body
{
    color: #cd853f;
}
A
{
    color: #8b4513;
    text-decoration: none;
}
A:hover
{
    color: #ffa500;
    text-decoration: underline;
}
```

## Stosowanie arkuszy stylów

Czas, aby arkusz wykorzystać do formatowania stron naszej przykładowej witryny:

- 1. Przejdźmy na stronę Default.aspx.
- 2. W widoku projektowania dodajmy do niej prosty tekst (wpisując go w widoku projektowania w polu Content) oraz komponent HyperLink ze skonfigurowaną właściwością NavigateUrl i etykietą (właściwość Text).
- 3. Teraz przejdźmy do widoku projektowania pliku wzorca MastepPage.master.
- **4.** Przeciągnijmy z okna projektu (*Solution Explorer*) utworzony plik .*css*. Do kodu strony dodany zostanie element <link href="StyleSheet.css" rel="stylesheet" type="text/css" />, dzięki któremu wzorzec i wszystkie używające go strony będą korzystały z arkusza i zdefiniowanych w nim stylów.

W podglądzie wzorca i podglądzie stron, które z niego korzystają, zobaczymy zmianę — tekst zmieni kolor na brązowy, łącza na jasnobrązowy. Ponadto łącza pozbawione zostały podkreślenia. Jeżeli obejrzymy stronę w przeglądarce, to zobaczymy, że kolor łączy zmienia się na pomarańczowy, jeżeli najechać na nie myszą, oraz że pojawia się wówczas pod nimi podkreślenie.



Jeżeli witryna nie ma wzorca, arkusz należy dodać do każdej strony osobno. To samo dotyczy stron w naszej witrynie, które nie korzystają ze wzorca.

Można również edytować indywidualny styl poszczególnych komponentów na stronach. W ich menu kontekstowym znajduje się pozycja *Style...*, która uruchamia okno *Style Builder* z rysunku 4.8 lub, jeżeli chcemy użyć istniejących klas stylu, w oknie *Properties* odnajdujemy właściwość CssClass i tam wpisujemy nazwę klasy zdefiniowanej w arkuszu stylu. Klasy można definiować w pliku .*css*, dodając regułę i zaznaczając opcję *Class name* (zob. rozdział 10.).