



Dobrowi autorzy – wymienita książka.  
Wszystko, co powinieneś wiedzieć o ASP.NET!

- Jakie kontrolki serwerowe udostępni ASP.NET 4?
- Jak monitorować stan aplikacji ASP.NET?
- Jak zapewnić najwyższą wydajność aplikacji?

Zaawansowane programowanie

# ASP.NET 4

z wykorzystaniem

# C# i VB

Bill Evjen, Scott Hanselman, Devin Rader



## » Idź do

- Spis treści
- Przykładowy rozdział

## » Katalog książek

- Katalog online
- Zamów drukowany katalog

## » Twój koszyk

- Dodaj do koszyka

## » Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

## » Czytelnia

- Fragmenty książek online

## » Kontakt

Helion SA  
ul. Kościuszki 1c  
44-100 Gliwice  
tel. 32 230 98 63  
e-mail: helion@helion.pl  
© Helion 1991–2010

## ASP.NET 4 z wykorzystaniem C# i VB. Zaawansowane programowanie

Autor: Bill Evjen, Scott Hanselman, Devin Rader

Tłumaczenie: Wojciech Moch, Tomasz Walczak

ISBN: 978-83-246-2846-9

Tytuł oryginału: [Professional ASP.NET 4 in C# and VB](#)

Format: 172×245, stron: 1792



### Doborowi autorzy – wymienita książka. Wszystko, co powinieneś wiedzieć o ASP.NET!

- Jakie kontrolki serwerowe udostępnia ASP.NET 4?
- Jak monitorować stan aplikacji ASP.NET?
- Jak zapewnić najwyższą wydajność aplikacji?

Platforma ASP.NET to główny konkurent języka Java w zakresie tworzenia aplikacji internetowych oraz dynamicznych stron internetowych. Każda jej wersja dostarcza wiele interesujących ulepszeń, a wśród nich te najważniejsze – pozwalające na zdjęcie z programisty obowiązku pisania dużych ilości nudnego kodu, bez którego jeszcze niedawno aplikacja nie mogłaby istnieć.

Niniejsza książka została napisana przez grupę wyjątkowych autorów. Bill Evjen to najaktywniejszy promotor technologii .NET, Scott Hanselman to główny menedżer w jednym z działów firmy Microsoft, prowadzący szkolenia dotyczące ASP.NET na całym świecie, a Devin Rader to pracownik firmy Infragistics. Ta doborowa trójka stworzyła świetny podręcznik, w całości poświęconą ASP.NET. Znajdziesz w niej informacje na temat stosowania języków VisualBasic oraz C# do tworzenia dynamicznych stron. Dowiesz się, jak wykorzystać kontrolki serwerowe, budować aplikacje z wykorzystaniem wzorca MVC oraz tchnąć życie w strony za pomocą technologii AJAX. Te i wiele innych bezcennych informacji, porad i wskazówek odkryjesz dzięki tej wyjątkowej książce!

- Produktywność programistów
- Infrastruktura i tworzenie aplikacji ASP.NET
- Środowisko ADO.NET
- Kompilacja aplikacji ASP.NET
- Monitorowanie stanu aplikacji
- Kontrolki serwerowe
- Wykorzystanie stron wzorcowych
- Kompozycje i skórki
- Wykonywanie zapytań z wykorzystaniem LINQ
- Wykorzystanie formatu XML
- Bezpieczeństwo aplikacji ASP.NET
- Instrumentacja

**Sprawdź, jaka moc drzemie w platformie ASP.NET!**

# Spis treści

<b>Wstęp .....</b>	<b>29</b>
<b>Rozdział 1. Środowiska do tworzenia aplikacji i stron .....</b>	<b>53</b>
Opcje lokalizacji aplikacji .....	54
Wbudowany serwer WWW .....	54
IIS .....	56
FTP .....	57
Strony internetowe wymagające FrontPage Extensions .....	57
Opcje struktury strony ASP.NET .....	58
Model inline .....	60
Model code-behind .....	62
Dyrektywy strony ASP.NET 4 .....	64
@Page .....	65
@Master .....	69
@Control .....	70
@Import .....	71
@Implements .....	74
@Register .....	74
@Assembly .....	75
@PreviousPageType .....	75
@MasterType .....	76
@OutputCache .....	76
@Reference .....	77
Zdarzenia strony ASP.NET .....	77
Praca z mechanizmem postback .....	79
Mechanizm cross-page posting .....	79
Katalogi aplikacji ASP.NET .....	85
Katalog AppCode .....	86
Katalog App_Data .....	90
Katalog App_Themes .....	90
Katalog App_GlobalResources .....	91
App_LocalResources .....	91
App_WebReferences .....	91
App_Browsers .....	91
Kompilacja .....	92

Build Providers .....	96
Korzystanie z wbudowanych klas BuildProvider .....	97
Korzystanie z własnych klas BuildProvider .....	98
Global.asax .....	103
Praca z klasami w VS 2010 .....	107
Podsumowanie .....	110
<b>Rozdział 2. Kontrolki serwerowe ASP.NET i skrypty po stronie klienta .....</b>	<b>111</b>
Kontrolki serwerowe ASP.NET .....	112
Typy kontrolek serwerowych .....	112
Tworzenie aplikacji z wykorzystaniem kontrolek serwerowych .....	114
Praca ze zdarzeniami kontrolek serwerowych .....	116
Dodawanie stylu do kontrolek serwerowych .....	118
Przegląd wspólnych właściwości kontrolek .....	118
Zmiana stylu za pomocą kaskadowych arkuszy stylów .....	120
Zmiany w CSS wprowadzone w ASP.NET 4 .....	123
Kontrolki serwerowe HTML .....	124
Omówienie klasy bazowej HtmlControl .....	126
Omówienie klasy HtmlContainerControl .....	127
Omówienie wszystkich klas HTML .....	127
Praca z klasą HtmlGenericControl .....	128
Identyfikowanie kontrolek serwerowych ASP.NET .....	129
Zarządzanie stronami i kontrolkami serwerowymi za pomocą JavaScriptu .....	132
Korzystanie z Page.ClientScript.RegisterClientScriptBlock .....	133
Korzystanie z Page.ClientScript.RegisterStartupScript .....	135
Korzystanie z Page.ClientScript.RegisterClientScriptInclude .....	137
Funkcja zwrotna po stronie klienta .....	137
Porównanie postback z funkcją zwrotną .....	137
Korzystanie z możliwości funkcji zwrotnej — proste podejście .....	140
Korzystanie z funkcji zwrotnych z jednym parametrem .....	144
Użycie mechanizmu funkcji zwrotnej — przykład bardziej zaawansowany ..	147
Podsumowanie .....	152
<b>Rozdział 3. Kontrolki serwerowe Web ASP.NET .....</b>	<b>153</b>
Wprowadzenie do kontrolek serwerowych Web .....	154
Kontrolka serwerowa Label .....	155
Kontrolka serwerowa Literal .....	156
Kontrolka serwerowa TextBox .....	157
Użycie metody Focus() .....	158
Użycie AutoPostBack .....	159
Użycie AutoCompleteType .....	160
Kontrolka serwerowa Button .....	161
Właściwość CausesValidation .....	162
Właściwość CommandName .....	162
Przyciski, które współpracują z JavaScriptem po stronie klienta .....	163
Kontrolka serwerowa LinkButton .....	165

Kontrolka serwerowa ImageButton .....	165
Kontrolka serwerowa HyperLink .....	166
Kontrolka serwerowa DropDownList .....	167
Wizualne usuwanie elementów z kolekcji .....	169
Kontrolka serwerowa ListBox .....	171
Umożliwienie wyboru kilku pozycji .....	172
Przykład użycia kontrolki ListBox .....	172
Dodawanie elementów do kolekcji .....	174
Kontrolka serwerowa CheckBox .....	174
W jaki sposób sprawdzić, czy pole wyboru jest zaznaczone .....	176
Przypisanie wartości do pola wyboru .....	176
Wyrównywanie tekstu kontrolki CheckBox .....	176
Kontrolka serwerowa CheckBoxList .....	177
Kontrolka serwerowa RadioButton .....	179
Kontrolka serwerowa RadioButtonList .....	181
Kontrolka serwerowa Image .....	182
Kontrolka serwerowa Table .....	184
Kontrolka serwerowa Calendar .....	186
Wybieranie daty za pomocą kontrolki Calendar .....	186
Wybieranie formatu daty pobieranej z kalendarza .....	188
Wybór dni, tygodni lub miesięcy .....	188
Praca z zakresami dat .....	189
Zmiana stylu i zachowania kalendarza .....	190
Kontrolka serwerowa AdRotator .....	193
Kontrolka serwerowa Xml .....	195
Kontrolka serwerowa Panel .....	196
Kontrolka serwerowa Placeholder .....	198
Kontrolka serwerowa BulletedList .....	199
Kontrolka serwerowa HiddenField .....	203
Kontrolka serwerowa FileUpload .....	205
Pobieranie plików za pomocą kontrolki FileUpload .....	205
Nadawanie ASP.NET właściwych praw do pobierania plików .....	208
Zrozumienie limitów rozmiaru plików .....	209
Wczytywanie wielu plików na tej samej stronie .....	211
Przekazywanie pobranego pliku do obiektu Stream .....	213
Przenoszenie zawartości pliku z obiektu Stream do tablicy bajtów .....	214
Kontrolki serwerowe MultiView oraz View .....	214
Kontrolka serwerowa Wizard .....	218
Dostosowanie nawigacji po stronach .....	219
Użycie atrybutu AllowReturn .....	220
Praca z atrybutem StepType .....	220
Wstawianie nagłówka w kontrolce Wizard .....	221
Praca z systemem nawigacji kontrolki Wizard .....	221
Obsługa zdarzeń kontrolki Wizard .....	222
Użycie kontrolki Wizard do pokazania elementów formularza .....	224

Kontrolka serwerowa ImageMap .....	228
Kontrolka serwerowa Chart .....	230
Podsumowanie .....	234
<b>Rozdział 4. Uwierzytelniające kontrolki serwerowe .....</b>	<b>235</b>
Zrozumienie procesu walidacji .....	235
Walidacja po stronie klienta a walidacja po stronie serwera .....	236
Kontrolki walidacyjne ASP.NET .....	237
Przyczyny walidacji .....	239
Kontrolka serwerowa RequiredFieldValidator .....	239
Kontrolka serwerowa CompareValidator .....	245
Kontrolka serwerowa RangeValidator .....	248
Kontrolka serwerowa RegularExpressionValidator .....	251
Kontrolka serwerowa CustomValidator .....	253
Kontrolka serwerowa ValidationSummary .....	258
Wyłączanie walidacji po stronie klienta .....	261
Korzystanie z obrazków i dźwięków w powiadomieniach o błędach .....	262
Praca z grupami walidacyjnymi .....	263
Podsumowanie .....	267
<b>Rozdział 5. Praca ze stronami wzorcowymi .....</b>	<b>269</b>
Do czego potrzebne są strony wzorcowe? .....	269
Podstawy stron wzorcowych .....	272
Pisanie kodu stron wzorcowych .....	273
Pisanie kodu strony z zawartością .....	276
Łączenie różnych typów stron i języków .....	280
Określanie, której strony wzorcowej użyć .....	281
Praca z tytułem strony .....	282
Praca z kontrolkami i właściwościami strony wzorcowej .....	283
Określanie domyślnej zawartości na stronie wzorcowej .....	290
Programowe przypisywanie strony wzorcowej .....	291
Osadzanie stron wzorcowych .....	293
Strony wzorcowe dostosowane do przeglądarek .....	296
Porządek wywoływania zdarzeń .....	298
Buforowanie stron wzorcowych .....	298
ASP.NET AJAX i strony wzorcowe .....	299
Podsumowanie .....	302
<b>Rozdział 6. Kompozycje i skórki .....</b>	<b>303</b>
Korzystanie z kompozycji ASP.NET .....	303
Przypisywanie kompozycji pojedynczej stronie ASP.NET .....	303
Stosowanie stylów do całej aplikacji .....	305
Usuwanie kompozycji z kontroltek serwerowych .....	306
Usuwanie kompozycji ze stron .....	307
Stosowanie kompozycji podczas korzystania ze stron wzorcowych .....	307
Działanie atrybutu StyleSheetTheme .....	308

Tworzenie własnych kompozycji .....	308
Tworzenie właściwej struktury katalogów .....	308
Tworzenie skórk .....	309
Umieszczanie w kompozycjach plików CSS .....	311
Wstawianie do kompozycji obrazków .....	314
Definiowanie wielu opcji skórek .....	317
Programowa praca z kompozycjami .....	319
Programowe przypisywanie kompozycji strony .....	319
Programowe przypisanie właściwości SkinID kontrolki .....	320
Kompozycje, skórki i własne kontrolki .....	320
Podsumowanie .....	324
<b>Rozdział 7. Wiązanie danych .....</b>	<b>325</b>
Kontrolki źródeł danych .....	326
Kontrolka SqlDataSource .....	328
Kontrolka AccessDataSource .....	337
Kontrolka LinqDataSource .....	337
Kontrolka EntityDataSource .....	344
Kontrolka XmlDataSource .....	346
Kontrolka ObjectDataSource .....	347
Kontrolka SiteMapDataSource .....	351
Konfiguracja buforowania kontrolek źródła danych .....	351
Przechowywanie informacji o połączeniu .....	352
Użycie kontrolek list umożliwiających wiązanie z kontrolkami źródeł danych ...	354
GridView .....	354
Edycja danych wierszy kontrolki GridView .....	370
Usuwanie danych w kontrolce GridView .....	377
DetailsView .....	380
Wstawianie, modyfikacja i usuwanie danych za pomocą kontrolki DetailsView .....	384
ListView .....	385
FormView .....	394
Inne kontrolki umożliwiające wiązanie danych .....	399
TreeView .....	399
AdRotator .....	400
Menu .....	400
Składnia rozwijanego wiązania danych .....	401
Zmiany w składni wiązania danych .....	401
Wiązanie danych w formacie XML .....	402
Wyrażenia i klasy do budowania wyrażeń .....	403
Podsumowanie .....	408
<b>Rozdział 8. Zarządzanie danymi w ADO.NET .....</b>	<b>409</b>
Podstawowe możliwości ADO.NET .....	410
Podstawowe zadania ADO.NET .....	410
Pobieranie danych .....	410

Podstawowe przestrzenie nazw i klasy ADO.NET .....	415
Korzystanie z obiektu Connection .....	416
Korzystanie z obiektu Command .....	418
Korzystanie z obiektu DataReader .....	419
Korzystanie z klas DataAdapter .....	422
Korzystanie z parametrów .....	424
Opis obiektów DataSet oraz DataTable .....	427
Typowany obiekt DataSet .....	432
Korzystanie z bazy danych Oracle w ASP.NET .....	433
Kontrolka serwerowa DataList .....	434
Przegląd dostępnych wzorców .....	434
Praca z ItemTemplate .....	434
Praca z innymi wzorcami układów graficznych .....	438
Praca z wieloma kolumnami .....	439
Kontrolka serwerowa ListView .....	441
Podłączanie kontrolki ListView do bazy danych .....	442
Tworzenie wzorca układu graficznego .....	443
Tworzenie ItemTemplate .....	445
Tworzenie EditItemTemplate .....	446
Tworzenie EmptyItemTemplate .....	446
Tworzenie InsertItemTemplate .....	447
Wyświetlanie wyników .....	447
Wykorzystanie Visual Studio do zadań związanych z ADO.NET .....	449
Tworzenie połączenia ze źródłem danych .....	450
Praca z projektantem DataSet .....	451
Korzystanie z obiektu DataSet CustomersOrders .....	457
Asynchroniczne wywoływanie poleceń .....	460
Asynchroniczne metody klasy SqlCommand .....	460
Interfejs IAsyncResult .....	462
AsyncCallback .....	463
Klasa WaitHandle .....	463
Sposoby przetwarzania asynchronicznego w ADO.NET .....	464
Anulowanie przetwarzania asynchronicznego .....	481
Asynchroniczne połączenia .....	481
Podsumowanie .....	482
<b>Rozdział 9. Zapytania w technologii LINQ .....</b>	<b>483</b>
LINQ to Objects .....	483
Tradycyjne metody zapytań .....	484
Zamiana tradycyjnych zapytań na zapytania w LINQ-u .....	491
Grupowanie danych .....	497
Inne operatory technologii LINQ .....	498
Złączenia w technologii LINQ .....	499
Paginacja za pomocą technologii LINQ .....	501
LINQ to XML .....	501
Łączenie danych w formacie XML .....	505



LINQ to SQL .....	506
Zapytania Insert, Update oraz Delete z wykorzystaniem technologii LINQ .....	514
Rozszerzanie technologii LINQ .....	518
Podsumowanie .....	519
<b>Rozdział 10. Praca z formatem XML i technologią LINQ to XML .....</b>	<b>521</b>
Podstawy języka XML .....	523
XML InfoSet .....	525
Definicja schematu XSD-XML .....	526
Edycja plików XML oraz schematów XML w Visual Studio 2010 .....	527
Klasy XmlReader oraz XmlWriter .....	530
Korzystanie z XmlDocument zamiast XmlReader .....	533
Korzystanie ze schematu oraz XmlTextReader .....	534
Walidacja względem schematu przy użyciu XmlDocument .....	536
Korzystanie z optymalizacji NameTable .....	538
Pobieranie typów .NET CLR z dokumentów XML .....	540
ReadSubtree oraz XmlSerialization .....	541
Tworzenie obiektów CLR z dokumentów XML za pomocą LINQ to XML .....	543
Tworzenie danych w formacie XML za pomocą XmlWriter .....	544
Tworzenie danych w formacie XML za pomocą LINQ to XML .....	546
Udoskonalenia obiektów XmlReader oraz XmlWriter .....	549
XmlDocument oraz XPathDocument .....	550
Problemy z modelem DOM .....	550
XPath, XPathDocument oraz XmlDocument .....	551
Obiekty DataSet .....	555
Zapisywanie obiektów DataSet w formacie XML .....	555
XmlDataDocument .....	557
Kontrolka XmlDataSource .....	559
XSLT .....	563
XslCompiledTransform .....	564
Debugowanie kodu w języku XSLT .....	569
XML i bazy danych .....	570
FOR XML AUTO .....	570
SQL Server oraz typy danych w języku XML .....	574
Podsumowanie .....	581
<b>Rozdział 11. Wprowadzenie do modelu dostawców .....</b>	<b>583</b>
Zrozumienie modelu dostawców .....	584
Model dostawców w ASP.NET 4 .....	585
Ustawianie dostawcy, aby współpracował z Microsoft SQL Server 7.0, 2000, 2005 lub 2008 .....	587
Dostawcy członkostwa .....	593
Dostawcy ról .....	597
Dostawca personalizacji .....	601
Dostawca SiteMap .....	602
Dostawcy SessionState .....	603

Dostawcy WebEvent .....	606
Dostawcy konfiguracji .....	614
Dostawca WebParts .....	617
Konfigurowanie dostawców .....	619
Podsumowanie .....	619

## **Rozdział 12. Rozszerzanie modelu dostawców .....621**

Dostawcy są jedną warstwą w rozbudowanej architekturze .....	621
Modyfikacja programowa z wykorzystaniem atrybutów .....	622
Ułatwienie wprowadzania hasła za pomocą SqlMembershipProvider .....	623
Nakładanie silnych restrykcji na hasło za pomocą SqlMembershipProvider ....	626
Analiza ProviderBase .....	627
Tworzenie własnych klas dostawców .....	628
Tworzenie aplikacji CustomProvider .....	629
Tworzenie wymaganego szkieletu klasy .....	630
Tworzenie magazynu danych w formacie XML .....	633
Definiowanie egzemplarza dostawcy w pliku web.config .....	634
Niezaimplementowane metody i właściwości klasy MembershipProvider .....	635
Implementacja metod i właściwości klasy MembershipProvider .....	636
Korzystanie z XmlMembershipProvider podczas logowania użytkownika .....	643
Rozszerzanie istniejących dostawców .....	645
Ograniczenie możliwości zarządzania rolami za pomocą nowego dostawcy LimitedSqlRoleProvider .....	645
Korzystanie z nowej klasy dostawcy LimitedSqlRoleProvider .....	649
Podsumowanie .....	653

## **Rozdział 13. Nawigacja witryny .....655**

Mapy witryny w postaci plików XML .....	656
Kontrolka serwerowa SiteMapPath .....	658
Właściwość PathSeparator .....	660
Właściwość PathDirection .....	662
Właściwość ParentLevelsDisplayed .....	662
Właściwość ShowToolTips .....	663
Elementy potomne kontrolki SiteMapPath .....	663
Kontrolka serwerowa TreeView .....	663
Wbudowane style kontrolki TreeView .....	668
Badanie składników kontrolki TreeView .....	669
Wiązanie kontrolki TreeView z plikiem XML .....	669
Wybór wielu opcji w kontrolce TreeView .....	672
Przypisywanie do kontrolki TreeView własnych ikon .....	675
Używanie linii w celu połączenia węzłów .....	677
Programistyczna praca z kontrolką TreeView .....	678
Kontrolka serwerowa Menu .....	683
Przypisywanie do kontrolki Menu różnych stylów .....	685
Zdarzenia kontrolki Menu .....	690
Wiązanie kontrolki Menu z plikiem XML .....	690

Dostawca danych SiteMap .....	692
ShowStartingNode .....	693
StartFromCurrentNode .....	693
StartingNodeOffset .....	694
StartingNodeUrl .....	695
SiteMap API .....	695
Mapowanie adresów URL .....	698
Lokalizacja mapy witryny .....	699
Tworzenie pliku Web.sitemap korzystającego z lokalizacji .....	699
Wprowadzanie modyfikacji w pliku Web.config .....	700
Tworzenie plików podzespołów z zasobami (.resx) .....	701
Testowanie wyników .....	702
Security trimming .....	704
Ustawienie zarządzania rolami dla administratorów .....	704
Ustawianie sekcji administratorów .....	705
Włączanie security trimming .....	707
Zagnieżdżanie plików SiteMap .....	708
Podsumowanie .....	710
<b>Rozdział 14. Personalizacja .....</b>	<b>711</b>
Model personalizacji .....	712
Tworzenie właściwości personalizacji .....	713
Dodawanie prostej właściwości personalizacji .....	713
Korzystanie z właściwości personalizacji .....	714
Dodawanie grup właściwości personalizacji .....	718
Korzystanie z grupowanych właściwości personalizacji .....	719
Definiowanie typów właściwości personalizacji .....	719
Korzystanie z własnych typów .....	720
Ustawianie wartości domyślnych .....	722
Tworzenie właściwości personalizacji tylko do odczytu .....	723
Personalizacja anonimowa .....	723
Umożliwienie anonimowej identyfikacji użytkowników .....	723
Praca z anonimową identyfikacją .....	726
Anonimowe opcje właściwości personalizacji .....	727
Uwagi na temat przechowywania profilów anonimowych użytkowników .....	728
Programowy dostęp do personalizacji .....	729
Migracja użytkowników anonimowych .....	729
Personalizacja profili .....	731
Określanie, czy korzystać z automatycznego zapisu .....	732
Dostawcy personalizacji .....	733
Praca z bazą SQL Server Express Edition .....	733
Praca z Microsoft SQL Server 7.0, 2000, 2005, 2008 .....	734
Korzystanie z wielu dostawców .....	736
Zarządzanie profilami aplikacji .....	737
Właściwości klasy ProfileManager .....	737

Metody klasy ProfileManager .....	738
Tworzenie strony ProfileManager.aspx .....	739
Omówienie kodu strony ProfileManager.aspx .....	742
Uruchomienie strony ProfileManager.aspx .....	743
Podsumowanie .....	744

## **Rozdział 15. Członkostwo i zarządzanie rolami ..... 745**

Uwierzytelnianie w ASP.NET 4 .....	746
Konfigurowanie systemu członkostwa w witrynie .....	746
Wstawianie użytkowników .....	749
Pobieranie danych uwierzytelniających .....	763
Praca z zarejestrowanymi użytkownikami .....	771
Pokazywanie liczby użytkowników online .....	773
Obsługa haseł .....	775
Autoryzacja w ASP.NET 4 .....	779
Korzystanie z kontrolki serwerowej LoginView .....	780
Konfiguracja systemu zarządzania rolami w witrynie .....	782
Dodawanie i pobieranie ról w aplikacji .....	785
Usuwanie ról .....	788
Dodawanie użytkowników do ról .....	789
Pobieranie wszystkich użytkowników określonej roli .....	789
Pobieranie wszystkich ról określonego użytkownika .....	791
Usuwanie użytkowników z ról .....	792
Sprawdzanie, czy użytkownicy przypisani są do ról .....	792
Wyjaśnienie sposobu buforowania ról .....	793
Korzystanie z narzędzia Web Site Administration Tool .....	794
Publiczne metody interfejsu API członkostwa .....	794
Publiczne metody interfejsu API ról .....	794
Podsumowanie .....	796

## **Rozdział 16. Platforma portalowa i kontrolki Web Parts ..... 797**

Wprowadzenie do kontrolki Web Parts .....	798
Tworzenie dynamicznych i modularnych portali .....	799
Wprowadzenie do kontrolki WebPartManager .....	800
Praca z układami stref .....	801
Omówienie kontrolki WebPartZone .....	804
Zezwolenie użytkownikowi na zmianę trybu strony .....	806
Modyfikacja stref .....	817
Praca z klasami platformy portalowej .....	823
Tworzenie własnych kontrolki Web Parts .....	827
Łączenie kontrolki Web Parts .....	833
Tworzenie dostawcy Web Part .....	834
Tworzenie kontrolki Web Part konsumenta .....	837
Łączenie kontrolki Web Parts na stronie ASP.NET .....	838
Trudności podczas łączenia kontrolki Web Parts przy stosowaniu stron wzorcowych .....	841
Podsumowanie .....	842

<b>Rozdział 17. Projektowanie za pomocą języków HTML i CSS w ASP.NET .....</b>	<b>843</b>
Uwagi .....	844
Ogólne informacje na temat HTML-a oraz CSS-a .....	844
Tworzenie arkuszy stylów .....	846
Reguły języka CSS .....	848
Dziedziczenie w języku CSS .....	857
Układ i położenie elementów .....	858
Praca z HTML-em oraz CSS-em w Visual Studio .....	866
Zarządzanie względnymi hiperłączami do plików CSS na stronach wzorcowych .....	872
Przypisywanie stylu do kontrolek ASP.NET .....	872
Podsumowanie .....	873
<b>Rozdział 18. ASP.NET AJAX .....</b>	<b>875</b>
Zrozumienie potrzeby stosowania AJAX-a .....	875
Przed technologią AJAX .....	876
AJAX zmienia ten stan rzeczy .....	877
ASP.NET AJAX oraz Visual Studio 2010 .....	880
Technologie po stronie klienta .....	880
Technologie działające po stronie serwera .....	881
Tworzenie aplikacji za pomocą ASP.NET AJAX .....	882
Tworzenie aplikacji ASP.NET AJAX .....	883
Tworzenie prostej strony ASP.NET niekorzystającej z AJAX-a .....	884
Tworzenie prostej strony ASP.NET z użyciem AJAX-a .....	886
Kontrolki ASP.NET AJAX po stronie serwera .....	891
Kontrolka ScriptManager .....	892
Kontrolka ScriptManagerProxy .....	894
Kontrolka Timer .....	896
Kontrolka UpdatePanel .....	897
Kontrolka UpdateProgress .....	901
Korzystanie z wielu kontrolek UpdatePanel .....	904
Praca z historią stron .....	908
Łączenie skryptów .....	913
Podsumowanie .....	917
<b>Rozdział 19. ASP.NET AJAX Control Toolkit .....</b>	<b>919</b>
Pobieranie i instalowanie zestawu AJAX Control Toolkit .....	921
Kontrolki ASP.NET AJAX .....	922
Kontrolki rozszerzające ASP.NET AJAX Control Toolkit .....	924
AlwaysVisibleControlExtender .....	924
AnimationExtender .....	927
AutoCompleteExtender .....	928
CalendarExtender .....	931
CollapsiblePanelExtender .....	933
ColorPickerExtender .....	935
ConfirmButtonExtender oraz ModalPopupExtender .....	936

DragPanelExtender .....	938
DropDownExtender .....	940
DropShadowExtender .....	942
DynamicPopulateExtender .....	944
FilteredTextBoxExtender .....	948
HoverMenuExtender .....	949
ListSearchExtender .....	951
MaskedEditExtender oraz MaskedEditValidator .....	952
MutuallyExclusiveCheckBoxExtender .....	954
NumericUpDownExtender .....	956
PagingBulletedListExtender .....	957
PopupControlExtender .....	958
ResizableControlExtender .....	960
RoundedCornersExtender .....	962
SliderExtender i MultiHandleSliderExtender .....	963
SlideShowExtender .....	964
TextBoxWatermarkExtender .....	967
ToggleButtonExtender .....	968
UpdatePanelAnimationExtender .....	970
ValidationCalloutExtender .....	971
Kontrolki serwerowe ASP.NET AJAX Control Toolkit .....	972
Kontrolka Accordion .....	973
CascadingDropDown .....	975
Kontrolka NoBot .....	978
Kontrolka PasswordStrength .....	980
Kontrolka Rating .....	981
Kontrolka TabContainer .....	982
Podsumowanie .....	984
<b>Rozdział 20. Bezpieczeństwo .....</b>	<b>985</b>
Techniki uwierzytelniania .....	986
Węzeł <authentication> .....	987
Uwierzytelnianie Windows .....	988
Uwierzytelnianie na podstawie formularzy .....	997
Uwierzytelnianie z wykorzystaniem mechanizmu Microsoft Passport .....	1008
Uwierzytelnianie w dostępie do określonych plików i katalogów .....	1008
Autoryzacja programowa .....	1009
Właściwość User.Identity .....	1009
Metoda User.IsInRole() .....	1011
Uzyskiwanie dodatkowych informacji z obiektu WindowsIdentity .....	1012
Element <identity> i tryb personifikacji .....	1014
Zabezpieczenia serwera IIS .....	1016
Ograniczenie zakresu adresów IP i nazw domenowych .....	1016
Rozszerzenia plików .....	1018
Korzystanie z konsoli ASP.NET MMC .....	1019
Konsola menedżera usługi IIS 7.0 .....	1022
Podsumowanie .....	1022

<b>Rozdział 21. Zarządzanie informacjami o stanie aplikacji .....</b>	<b>1023</b>
Jakie opcje są do wyboru? .....	1024
Obiekt Session platformy ASP.NET .....	1027
Sesje a model zdarzeń .....	1027
Konfiguracja mechanizmu zarządzania sesją .....	1029
Sesje wewnętrzne .....	1030
Sesje zewnętrzne .....	1039
Sesje zapisywane w serwerach SQL .....	1044
Rozszerzenie sesji o inne mechanizmy dostawców danych .....	1048
Sesje bez plików cookie .....	1049
Wybór odpowiedniego sposobu podtrzymywania sesji .....	1050
Obiekt Application .....	1051
Łącuchy zapytania .....	1052
Dane cookie .....	1053
Odsyłanie danych i przekazywanie danych między stronami .....	1053
Ukryte pola formularza, mechanizmy ViewState oraz ControlState .....	1056
Wykorzystanie kolekcji HttpContext.Current.Items do przechowywania krótkookresowych wartości .....	1061
Podsumowanie .....	1062
<b>Rozdział 22. Buforowanie .....</b>	<b>1063</b>
Buforowanie .....	1064
Buforowanie danych wyjściowych .....	1064
Buforowanie części strony (kontrolki użytkownika) .....	1068
Podmiana wartości w buforowanej treści .....	1069
Buforowanie po stronie klienta i obiekt HttpCachePolicy .....	1071
Buforowanie programowe .....	1074
Buforowanie danych za pomocą obiektu Cache .....	1074
Nadzorowanie pracy pamięci podręcznej środowiska ASP.NET .....	1075
Zależności wpisów pamięci podręcznej .....	1075
Nowe możliwości obiektu Caching w .NET 4 .....	1081
Zależności bufora SQL .....	1084
Włączanie unieważniania bufora SQL dla baz danych .....	1085
Dodanie tabeli do list tabel uwzględnianych w zależnościach bufora SQL ...	1085
SQL Server 2000 .....	1086
Analiza włączonych tabel .....	1087
Usunięcie tabeli z listy tabel uwzględnianych w zależnościach bufora SQL .....	1087
Usunięcie bazy danych z listy baz uwzględnianych w zależnościach bufora SQL .....	1088
Zależności bufora SQL w bazach danych SQL Server 2005 i 2008 .....	1088
Konfiguracja aplikacji ASP.NET .....	1089
Testowanie mechanizmu unieważniania danych bufora SQL .....	1091
Odwołanie do więcej niż jednej tabeli w kodzie strony .....	1093
Powiązanie zależności bufora SQL z obiektem Request .....	1094
Powiązanie zależności bufora SQL z obiektem Cache .....	1094
Podsumowanie .....	1098

<b>Rozdział 23. Debugowanie i obsługa błędów .....</b>	<b>1099</b>
Wsparcie w czasie projektowania .....	1100
Powiadomienia o błędach składni .....	1100
Okna Immediate i Command .....	1102
Lista zadań .....	1103
Śledzenie kodu .....	1104
Klasy System.Diagnostics.Trace oraz Page.Trace w ASP.NET .....	1104
Śledzenie kodu na poziomie strony .....	1104
Śledzenie pracy aplikacji .....	1105
Przeglądanie danych wynikowych .....	1105
Śledzenie pracy komponentów .....	1109
Przekazywanie danych ze śledzenia kodu .....	1111
Obiekty TraceListener .....	1112
Przełączniki diagnostyczne .....	1116
Zdarzenia sieciowe .....	1118
Debugowanie .....	1120
Potrzebne elementy .....	1120
Usługi IIS i ASP.NET Development Server .....	1121
Uruchomienie sesji debugowania .....	1123
Narzędzia ułatwiające debugowanie .....	1126
Debugowanie historii za pomocą mechanizmu IntelliTrace .....	1130
Debugowanie kodu wielowątkowego .....	1132
Debugowanie klienckiego kodu w języku JavaScript .....	1132
Debugowanie procedur składowanych SQL .....	1134
Wyjątki i obsługa błędów .....	1135
Przechwytywanie wyjątku na stronie .....	1136
Obsługa wyjątków aplikacji .....	1137
Kody statusowe HTTP .....	1138
Podsumowanie .....	1140
<b>Rozdział 24. Pliki i strumienie .....</b>	<b>1141</b>
Dyski, katalogi i pliki .....	1141
Klasa DriveInfo .....	1142
Klasy Directory i DirectoryInfo .....	1146
Klasy File i FileInfo .....	1153
Przetwarzanie ścieżek dostępu .....	1159
Właściwości plików i katalogów, ich atrybuty oraz listy kontroli dostępu ....	1163
Odczyt i zapis plików .....	1170
Strumienie .....	1171
Obiekty odczytu i zapisu .....	1178
Kompresowanie danych strumieni .....	1184
Pliki odwzorowane w pamięci .....	1186
Wykorzystanie portów szeregowych .....	1188
Komunikacja międzyprocesowa z wykorzystaniem potoków .....	1190



Komunikacja sieciowa .....	1191
Klasy WebRequest i WebResponse .....	1191
Przesyłanie poczty elektronicznej .....	1197
Podsumowanie .....	1198
<b>Rozdział 25. Kontrolki użytkownika i kontrolki serwerowe .....</b>	<b>1199</b>
Kontrolki użytkownika .....	1200
Utworzenie kontrolki użytkownika .....	1200
Interakcje z kontrolkami użytkownika .....	1202
Dynamiczne ładowanie kontrolek użytkownika .....	1204
Kontrolki serwerowe .....	1209
Projekty kontrolek serwerowych .....	1210
Atrybuty sterujące .....	1214
Wyświetlanie kontrolki .....	1215
Dodawanie atrybutów znaczników .....	1220
Definicje stylu HTML .....	1222
Motywy tematyczne i skórki .....	1225
Dodanie elementów kodu klienckiego .....	1226
Wykrywanie parametrów przeglądarki .....	1236
Mechanizm ViewState .....	1238
Wywoływanie zdarzeń powodujących odesłanie strony .....	1242
Obsługa odsyłanych danych .....	1246
Kontrolki złożone .....	1248
Kontrolki szablonowe .....	1251
Zachowanie kontrolki w środowisku projektowym .....	1256
Podsumowanie .....	1273
<b>Rozdział 26. Moduły i obsługa żądań .....</b>	<b>1275</b>
Przetwarzanie żądań HTTP .....	1275
IIS 6 i ASP.NET .....	1276
IIS 7 i ASP.NET .....	1276
Przetwarzanie żądań ASP.NET .....	1277
Moduły HTTP .....	1278
Procedury obsługi żądań HTTP .....	1283
Standardowe mechanizmy obsługi żądań .....	1283
Odwzorowanie rozszerzenia pliku w serwerze IIS .....	1287
Podsumowanie .....	1290
<b>Rozdział 27. ASP.NET MVC .....</b>	<b>1291</b>
Definiowanie modelu MVC .....	1292
Model MVC w dzisiejszej sieci WWW .....	1292
Wzorzec MVC i ASP.NET .....	1294
Dostarczanie metod, a nie plików .....	1294
Czy to już Web Forms 4.0? .....	1294
A dlaczego nie Web Forms? .....	1295
ASP.NET MVC to coś zupełnie innego! .....	1295

Dlaczego „(ASP.NET > ASP.NET MVC) == True” .....	1296
Konwencja przed konfiguracją .....	1298
Do trzech żądań sztuka .....	1301
Ścieżki i adresy URL .....	1304
Routing adresów a ich przepisywanie .....	1305
Definiowanie ścieżek .....	1305
Kontrolery .....	1310
Definiowanie kontrolera: interfejs IController .....	1310
Klasa Controller i akcje .....	1311
Praca z parametrami .....	1312
Praca z wieloma parametrami .....	1313
Widoki .....	1313
Definiowanie widoku .....	1314
Widoki o zdefiniowanym typie .....	1316
Stosowanie metod pomocniczych HTML .....	1317
Klasa HtmlHelper i metody rozszerzeń .....	1317
Podsumowanie .....	1319
<b>Rozdział 28. Obiekty biznesowe .....</b>	<b>1321</b>
Korzystanie z obiektów biznesowych w środowisku ASP.NET 4 .....	1322
Tworzenie wstępnie skompilowanych obiektów biznesowych platformy .NET .....	1322
Wykorzystanie wstępnie skompilowanych obiektów biznesowych w aplikacji ASP.NET .....	1325
Wykorzystanie komponentów COM w środowisku .NET .....	1326
Komponent Runtime Callable Wrapper .....	1326
Wykorzystanie obiektów COM w kodzie ASP.NET .....	1328
Obsługa błędów .....	1332
Wdrażanie komponentów COM w aplikacjach .NET .....	1335
Odwołania do kodu .NET z poziomu kodu niezarządzanego .....	1337
Moduł COM-Callable Wrapper .....	1338
Współdziałanie komponentów .NET z obiektami COM .....	1340
Wczesne czy późne wiązanie .....	1343
Obsługa błędów .....	1344
Wdrażanie komponentów .NET z aplikacjami COM .....	1346
Podsumowanie .....	1347
<b>Rozdział 29. Platforma ADO.NET Entity .....</b>	<b>1349</b>
Czy możemy zacząć mówić tym samym językiem? .....	1350
Warstwy pojęciowa i logiczna .....	1351
Odzworowywanie między warstwami .....	1351
Tworzenie pierwszego modelu EDM .....	1352
Korzystanie z kreatora EDM .....	1353
Używanie okna projektowego platformy ADO.NET Entity .....	1355
Budowanie strony ASP.NET korzystającej z modelu EDM .....	1356

Wprowadzenie do relacji .....	1358
Relacje jeden do jednego i jeden do wielu .....	1359
Relacje wiele do jednego i wiele do wielu .....	1362
Stosowanie dziedziczenia w modelu EDM .....	1365
Stosowanie procedur składowanych .....	1368
Stosowanie kontrolki EntityDataSource .....	1371
Tworzenie podstawowej strony .....	1372
Konfigurowanie kontrolki EntityDataSource .....	1373
Podsumowanie .....	1376
<b>Rozdział 30. ASP.NET Dynamic Data .....</b>	<b>1377</b>
Tworzenie aplikacji bazowej za pomocą Visual Studio 2010 .....	1377
Podstawowe pliki umieszczone w domyślnej aplikacji .....	1378
Aplikacja Dynamic Data .....	1379
Podłączanie bazy danych .....	1386
Rejestrowanie modelu danych w pliku Global.asax .....	1389
Style i układ strony .....	1391
Efekt działania aplikacji .....	1391
Praca z dynamicznymi ścieżkami .....	1395
Kontrola wyświetlania .....	1399
Dodawanie technologii Dynamic Data do istniejących stron .....	1401
Podsumowanie .....	1404
<b>Rozdział 31. Budowanie i wykorzystywanie usług .....</b>	<b>1405</b>
Komunikacja między rozproszonymi systemami .....	1406
Budowa prostej XML-owej usługi sieciowej .....	1408
Dyrektywa WebService .....	1409
Plik klasy bazowej usługi sieciowej .....	1409
Udostępnianie niestandardowych zbiorów danych w formie dokumentów SOAP .....	1411
Interfejs usługi sieciowej .....	1413
Korzystanie z nieskomplikowanych XML-owych usług sieciowych .....	1416
Dodawanie odwołań .....	1417
Wywoływanie usługi sieciowej w kodzie aplikacji klienckiej .....	1419
Przeciążanie metod sieciowych .....	1421
Buforowanie odpowiedzi usług sieciowych .....	1424
Nagłówki SOAP .....	1424
Tworzenie usług sieciowych uwzględniających nagłówki SOAP .....	1425
Wykorzystanie nagłówków SOAP w odwołaniach do usługi sieciowej .....	1427
Wykorzystanie żądań SOAP 1.2 .....	1429
Asynchroniczne odwołania do usług sieciowych .....	1431
Windows Communication Foundation .....	1434
Krok w stronę architektury opartej na usługach .....	1434
Przegląd technologii WCF .....	1435
Tworzenie usług WCF .....	1436

Aplikacja korzystająca z usługi WCF .....	1443
Dodanie odwołania do usługi .....	1444
Kontrakty danych .....	1447
Przestrzenie nazw .....	1451
Korzystanie z usług WCF Data Service .....	1452
Tworzenie pierwszej usługi .....	1453
Dodawanie modelu EDM .....	1453
Tworzenie usługi .....	1455
Kierowanie zapytań do interfejsu .....	1460
Wczytywanie tabel z danymi .....	1461
Wczytywanie konkretnych elementów z tabeli .....	1462
Zarządzanie relacjami .....	1464
Rozwijanie powiązań .....	1467
Porządkowanie zbiorów wyników .....	1470
Poruszanie się po zbiorach wyników .....	1471
Filtrowanie danych .....	1471
Używanie usług WCF Data Service w aplikacjach ASP.NET .....	1473
Podsumowanie .....	1476

## **Rozdział 32. Budowanie aplikacji międzynarodowych ..... 1477**

Ustawienia kulturowe i regionalne .....	1477
Typy kulturowe .....	1478
Wątki ASP.NET .....	1480
Ustawienia kulturowe serwera .....	1482
Ustawienia kulturowe klienta .....	1484
Tłumaczenie wartości i zmiana sposobu zachowania aplikacji .....	1485
Pliki zasobów ASP.NET 4 .....	1493
Wykorzystanie zasobów lokalnych .....	1493
Wykorzystanie zasobów globalnych .....	1499
Edytor zasobów .....	1502
Podsumowanie .....	1502

## **Rozdział 33. Konfiguracja ..... 1503**

Ogólne informacje na temat konfiguracji .....	1504
Pliki konfiguracyjne serwera .....	1505
Plik konfiguracyjny aplikacji .....	1507
W jaki sposób są odczytywane ustawienia konfiguracyjne? .....	1508
Wykrywanie zmian w plikach konfiguracyjnych .....	1509
Format pliku konfiguracyjnego .....	1509
Wspólne ustawienia konfiguracyjne .....	1510
Łącuchy połączeń .....	1510
Konfiguracja stanu sesji .....	1511
Konfiguracja kompilacji .....	1516
Parametry przeglądarki .....	1518
Niestandardowe komunikaty o błędach .....	1520
Uwierzytelnianie .....	1521

Identyfikacja użytkowników anonimowych .....	1525
Autoryzacja .....	1526
Blokowanie ustawień konfiguracyjnych .....	1528
Konfiguracja strony ASP.NET .....	1529
Włączane pliki .....	1531
Parametry pracy środowiska ASP.NET .....	1532
Konfiguracja procesu roboczego ASP.NET .....	1535
Przechowywanie ustawień aplikacji .....	1538
Programowe przetwarzanie plików konfiguracyjnych .....	1538
Ochrona ustawień konfiguracyjnych .....	1545
Edycja pliku konfiguracyjnego .....	1549
Tworzenie własnych sekcji konfiguracyjnych .....	1551
Wykorzystanie obiektu NameValueCollectionHandler .....	1551
Wykorzystanie obiektu DictionarySectionHandler .....	1553
Wykorzystanie obiektu SingleTagSectionHandler .....	1554
Wykorzystanie własnej procedury obsługi ustawień konfiguracyjnych .....	1555
Podsumowanie .....	1556
<b>Rozdział 34. Instrumentacja .....</b>	<b>1559</b>
Dzienniki zdarzeń .....	1559
Odczytywanie informacji z dziennika zdarzeń .....	1560
Zapis informacji w dzienniku zdarzeń .....	1562
Wskaźniki wydajności .....	1565
Przeglądanie wskaźników wydajności za pomocą narzędzi administracyjnych .....	1566
Narzędzie administracyjne uruchamiane w przeglądarce .....	1568
Śledzenie kodu aplikacji .....	1573
Monitorowanie kondycji aplikacji .....	1574
Model dostawcy danych systemu monitorowania kondycji aplikacji .....	1574
Konfiguracja systemu monitorowania kondycji aplikacji .....	1576
Zapis zdarzeń na podstawie parametrów konfiguracyjnych — uruchomienie przykładowej aplikacji .....	1583
Przekazywanie zdarzeń do serwera SQL .....	1584
Buforowanie zdarzeń sieciowych .....	1587
Wysyłanie informacji o zdarzeniach za pomocą poczty elektronicznej .....	1590
Podsumowanie .....	1594
<b>Rozdział 35. Administracja i zarządzanie .....</b>	<b>1595</b>
Aplikacja ASP.NET Web Site Administration Tool .....	1595
Zakładka Home .....	1597
Zakładka Security .....	1597
Zakładka Application .....	1605
Zakładka Provider .....	1609
Konfiguracja środowiska ASP.NET w usługach IIS w systemie Windows 7 ...	1610
Kompilacja platformy .NET .....	1612
Globalizacja platformy .NET .....	1613
Profil platformy .NET .....	1613

Role platformy .NET .....	1614
Poziomy zaufania platformy .NET .....	1614
Użytkownicy platformy .NET .....	1615
Ustawienia aplikacji .....	1616
Ciągi połączenia .....	1616
Strony i formanty .....	1617
Dostawcy .....	1617
Stan sesji .....	1617
Poczta e-mail SMTP .....	1618
Podsumowanie .....	1619
<b>Rozdział 36. Pakowanie i instalacja aplikacji ASP.NET .....</b>	<b>1621</b>
Instalowane elementy .....	1622
Czynności poprzedzające instalację .....	1622
Metody instalowania aplikacji WWW .....	1623
Program XCopy .....	1624
Opcja Copy Web Site środowiska Visual Studio .....	1626
Instalowanie wstępnie skompilowanej aplikacji WWW .....	1630
Budowanie pakietów z witrynami ASP.NET .....	1631
Utworzenie programu instalatora .....	1635
Szczegółowa analiza opcji instalatora .....	1642
Praca nad projektem instalacyjnym .....	1642
Edytor systemu plików .....	1646
Edytor rejestru .....	1650
Edytor typów plików .....	1651
Edytor interfejsu użytkownika .....	1652
Edytor niestandardowych operacji .....	1654
Edytor warunków uruchomienia .....	1655
Podsumowanie .....	1656
<b>Dodatek A Wykorzystanie projektów wcześniejszych wersji ASP.NET .....</b>	<b>1657</b>
Przenoszenie nie jest trudne .....	1657
Łączenie wersji — uwierzytelnianie na podstawie formularzy .....	1660
Aktualizacja — zarezerwowane foldery ASP.NET .....	1661
Format XHTML stron ASP.NET 4 .....	1662
Brak plików .js w ASP.NET 4 .....	1664
Konwertowanie aplikacji ASP.NET 1.x w środowisku Visual Studio 2010 .....	1664
Przeniesienie aplikacji ze środowiska ASP.NET 2.0 lub 3.5 do 4 .....	1668
Podsumowanie .....	1669
<b>Dodatek B Najważniejsze narzędzia w ASP.NET .....</b>	<b>1671</b>
Łatwiejsze debugowanie .....	1671
Źródła informacji .....	1677
Porządkowanie kodu .....	1678
Rozszerzanie środowiska ASP.NET .....	1681
Narzędzia programistyczne ogólnego przeznaczenia .....	1685
Podsumowanie .....	1688

---

<b>Dodatek C</b>	<b>Silverlight 3 i ASP.NET .....</b>	<b>1689</b>
	Wprowadzenie .....	1690
	Korzystanie z dodatku Silverlight .....	1692
	Silverlight i JavaScript .....	1700
	Podsumowanie .....	1710
<b>Dodatek D</b>	<b>Dynamiczne typy i języki .....</b>	<b>1711</b>
	Typy domniemane .....	1711
	Dynamic Language Runtime .....	1712
	Dynamiczne wyszukiwanie .....	1715
	Podsumowanie .....	1719
<b>Dodatek E</b>	<b>Serwisy internetowe o ASP.NET .....</b>	<b>1721</b>
	Blogi autorów książki oraz identyfikatory Twittera .....	1721
	Inne blogi na temat ASP.NET .....	1721
	Witryny internetowe .....	1722
	Kanały Twittera warte zasubskrybowania .....	1722
	<b>Skorowidz .....</b>	<b>1723</b>

---

# Praca ze stronami wzorcowymi

## ZAWARTOŚĆ ROZDZIAŁU:

- Tworzenie stron wzorcowych i stron z zawartością
- Stosowanie stron wzorcowych do określania domyślnych elementów
- Programowe przypisywanie stron wzorcowych
- Zagnieżdżanie stron wzorcowych
- Strony wzorcowe dla różnych przeglądarek
- Używanie stron wzorcowych z technologią ASP.NET AJAX

Dziedziczenie wizualne jest wspaniałym usprawnieniem, z którego można korzystać w ASP.NET przy budowaniu stron WWW. Taka możliwość została wprowadzona do ASP.NET w wersji 2.0. W efekcie otrzymujemy szansę utworzenia pojedynczej strony wzorcowej, która może być potem użyta jako podstawa dla dowolnej ilości zwykłych stron z zawartością w aplikacjach ASP.NET. Takie wzorce, zwane **stronami wzorcowymi**, zwiększają produktywność, sprawiając, że aplikacje tworzy się łatwiej. Prostsze jest także późniejsze zarządzanie takimi aplikacjami. Visual Studio 2010 daje nam pełne wsparcie przy tworzeniu stron wzorcowych za pomocą projektanta formularzy. Możliwości projektowe są teraz większe niż kiedykolwiek wcześniej. W niniejszym rozdziale pokazane zostanie, jak wykorzystać strony wzorcowe w aplikacjach w najlepszy możliwy sposób. Zaczniemy jednak od omówienia zalet wykorzystania stron wzorcowych.

## Do czego potrzebne są strony wzorcowe?

W dzisiejszych czasach większość portali internetowych posiada wspólne elementy, które wykorzystywane są przez większość stron aplikacji. Przyjrzyjmy się dla przykładu stronie głównej serwisu Reuters News (pod adresem [www.reuters.com](http://www.reuters.com)). Z łatwością można wyróżnić wspólne elementy używane na całym portalu. Oznaczono je na rysunku 5.1.



**Nagłówek** — **REUTERS** | LATEST NEWS: U.S., BRITISH FORCES BATTLE MEHDI ARMY IN BAGHDAD, BASRA | [Outlet](#) [News](#) [Pictures](#) [Video](#) [SEARCH](#) [Login](#)

DUA: 13587.28 • | Nasdaq: 2557.19

**HOME**  
**BUSINESS**  
 Industries  
 Companies  
 More...  
**INVESTING**  
 Markets  
 Stocks  
 Funds  
 More...  
**NEWS**  
 U.S.  
 International  
 Video  
 Pictures  
 Blogs  
 More...

**U.S., British forces battle Mehdi Army in Baghdad, Basra** 11:08am  
 BAGHDAD (Reuters) - U.S. and British forces battled Mehdi Army fighters in Baghdad and the southern city of Basra after their leader, Shiite cleric Moqtada al-Sadr, made a rare public appearance and called on U.S. troops to get out of Iraq. [Full Article](#)

**Israel pounds Gaza, arrests Palestinian minister** 12:17pm  
 GAZA (Reuters) - Israel stepped up its campaign against Hamas Islamists on Saturday, killing at least five fighters in the Gaza Strip in a wave of air strikes and seizing a Palestinian cabinet minister in the occupied West Bank. [Full Article](#) | [Video](#)

**Reuters Weekend**  
**Return to Manchuria**  
 American vets return to a Japanese POW camp as China decides to turn it into a museum. [Full Article](#) | [Watch Video](#)

**Edgy skiing vacation**  
 As ski resorts go, Kashmir's Gulmarg must rank as the most militarized on earth. [Full Article](#)

**A dream job?**  
 Good pay and friends draw even female workers to an Arctic coal mine. [Full Article](#)

**Investing**  
**MARKETS:**  
**May jobs data to test stocks' rally**  
 If May 25, 2007 09:56PM EDT  
 NEW YORK (Reuters) - Wall Street's merger-fueled rally may continue next week, so long as Friday's payroll report and other economic news don't spoil the party. [Full Article](#) | [Video](#)  
 - Wall St closes up as Nasdaq, Coke spur S&P optimism | [Video](#)  
 - U.S. crude jumps \$1 on gasoline worries | [Video](#)  
 - Dollar slips after weak house data  
 - Amgen to appeal as EU experts reject Vedibot drug  
 - More Investing News...

**COMMENT & ANALYSIS**  
 If world has one rate, it looks too low  
 The world economy is booming, financial markets are supercharged, energy prices are sky-high and credit growth is accelerating. If that spells inflation to you, global monetary policy looks way too loose. [Full Article](#)  
 Banks flock to booming carbon trade  
 Banks are jostling for a piece of what may be the world's fastest growing market, trading carbon emissions permits. Citigroup has just waded in, Bank of America is set to and Deutsche has doubled its team. [Full Article](#)

**MARKET UPDATE** sponsored by **Scottrade**  
 US | GB | UK | JP | More Indices...  
 DUA: 13587.28 06:18  
 S&P 500: 2557.19 -19.27  
 NASDAQ: 1916.73 -9.22  
 As of 12:18pm EDT, 05/20/07  
 11520  
 100%  
 2548.5  
 2543.5  
 8 10 11 12 13 14 15 2007.5  
 + UPDATE: 1-Wall St Week Ahead: May jobs data to test stocks' rally  
 + FTSE ends flat as oils offset gains in miners

**GET A QUOTE**  
 Enter a symbol:  | Create a Portfolio

**CURRENCY \$** Server  
 1.00 USD \$ to GBP £ [Convert](#)

**Przerznię na reklamy**

**Wspólne elementy strony**

**Stopka**

Reuters.com: [Help](#) [Contact Us](#) [Advertise With Us](#) [Mobile](#) [Newsletters](#) [RSS](#) [Widgets](#) [Interactive TV](#) [Labs](#) [Reuters in Second Life](#) [Site Index](#)  
 Reuters Corporate: [Copyright](#) [Disclaimer](#) [Privacy](#) [Professional Products](#) [Professional Products Support](#) [About Reuters](#) [Careers](#)

**International Editions:** Africa Arabic Argentina Brazil Canada Chinese Simplified Chinese Traditional France Germany India Italy Japan Latin America Mexico Russia Spain United Kingdom United States

Reuters is the world's largest international multimedia news agency, providing investing news, world news, business news, technology news, headline news, small business news, news alerts, personal finance, stock market, and mutual funds information available on Reuters.com, video, mobile, and interactive television platforms. Reuters journalists are subject to the Reuters Editorial Handbook which requires fair presentation and disclosure of relevant interests.

Rysunek 5.1

Na pokazanym zrzucie ekranu warto zwrócić uwagę między innymi na sekcję nagłówka, sekcję nawigacji i sekcję stopki. Większość stron aplikacji korzysta z tych samych elementów. Przed erą stron wzorcowych także istniały sposoby umieszczania tych samych elementów na wszystkich stronach. W większości przypadków przy stosowaniu różnych rozwiązań pojawiały się problemy.

Niektórzy programiści zwyczajnie kopiowali i wklejali kod wspólnych sekcji do każdej strony, która go potrzebowała. To działało, ale wymagało dużego nakładu pracy. Istnieje jednak problem poważniejszy. Gdy potrzebna była zmiana w jednej z tych sekcji, wtedy programista zmuszony był do przejrzania wszystkich stron portalu i wprowadzenia do każdej ze stron podobnych zmian. To nie jest oczywiście przyjemne. Istnieją lepsze sposoby wykorzystania czasu.

W czasach klasycznego języka ASP (ang. *Active Server Pages*) popularnym rozwiązaniem było umieszczenie wspólnych sekcji w **pliku dołączanym**. Taki plik mógł być potem umieszczany na stronie w taki oto sposób:

```
<!-- #include virtual="/myIncludes/header.asp" -->
```

Problem podczas korzystania z plików dołączanych polegał na tym, że należało brać pod uwagę otwarte znaczniki języka HTML z nagłówkowego pliku dołączanego. Znaczniki musiały być zamknięte w głównym dokumencie lub w kolejnym pliku dołączanym (ze stopką). Bardzo trudno było w takich sytuacjach utrzymać porządek znaczników języka HTML, zwłaszcza wtedy, gdy nad jednym projektem pracowało kilka osób. Strony internetowe czasami były pokazywane w dziwny sposób. Gdzieś bowiem po drodze mógł znaleźć się niewłaściwy lub nieistniejący znacznik zamykający lub otwierający. Trudno było także pracować z plikami dołączanymi w projektancie formularzy. Użycie plików dołączanych powodowało, że projektant nie był w stanie wyświetlić strony tak, jak powinna być pokazywana w przeglądarce. Programista kończył pisanie strony w sekcjach i *miał nadzieję*, że wszystkie kawałki w jakiś sposób poskładają się zgodnie z planem. Wiele godzin było marnowane na „ściganie tabel” otwartych w pliku dołączanym i prawdopodobnie zamkniętych później!

Wraz z wprowadzeniem ASP.NET 1.0 w 2000 roku programiści zaczęli używać **kontrolki użytkownika** do ukrywania powtarzających się sekcji stron WWW. Można było na przykład stworzyć stronę, która zawierała nagłówek, panel nawigacji i stopkę, przeciągając i upuszczając te sekcje kodu na każdą stronę, która tych elementów potrzebowała.

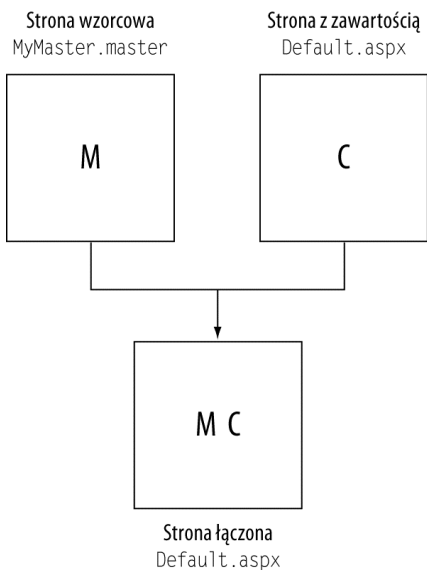
Ta technika działała, ale także powodowała pewne problemy. Przed pojawieniem się środowiska Visual Studio 2005 oraz technologii ASP.NET 2.0 kontrolki użytkownika prowadziły do problemu omówionego przy okazji plików dołączanych. Podczas pracy w widoku projektanta stron w środowisku Visual Studio .NET 2002 i 2003 wspólne obszary były wyświetlane w postaci szarych prostokątów. To utrudniało tworzenie strony. Nie dało się zobaczyć, jak tworzona właśnie strona wygląda, dopóki projekt nie został skompilowany i uruchomiony w przeglądarce. Kontrolki użytkownika nie były też wolne od drugiego problemu dotyczącego plików dołączanych — należało właściwie sparować otwierające i zamykające znaczniki języka HTML umieszczone w dwóch różnych plikach. Generalnie kontrolki użytkownika są lepsze niż pliki dołączane, ale to także nie jest doskonały sposób na rozwiązanie omawianych problemów w aplikacjach. Można zauważyć, że w środowisku Visual Studio poprawiono kilka problemów związanych z renderowaniem zawartości kontrolki użytkownika w oknie projektowym. Kontrolki użytkownika są dobre wtedy, gdy trzeba umieścić na stronach jakieś niewielkie sekcje. Problem w dalszym ciągu nie jest rozwiązany, gdy zachodzi potrzeba zastosowania ich jako większych wzorców.

W świetle problemów pojawiających się przy plikach dołączanych i kontrolkach użytkownika grupa odpowiedzialna za rozwój ASP.NET wpadła na pomysł **stron wzorcowych** — to nowy sposób stosowania wzorców w aplikacjach. *Zmieniono* dzięki temu sposób, w jaki programiści atakowali ten problem. Strony wzorcowe umieszczone są *poza* tworzonymi stronami. Jest to przeciwieństwo kontrolki użytkownika, które umieszczane były na stronie i były wielokrotnie powielane. Strony wzorcowe pozwalają oddzielić wspólne obszary, które umieszczane są na każdej stronie, i obszary, które są unikatowe dla każdej ze stron. Wkrótce pokażemy, że praca ze stronami wzorcowymi jest łatwa i przyjemna. W kolejnym podpunkcie omówione zostaną podstawowe elementy pracy ze stronami wzorcowymi ASP.NET.

## Podstawy stron wzorcowych

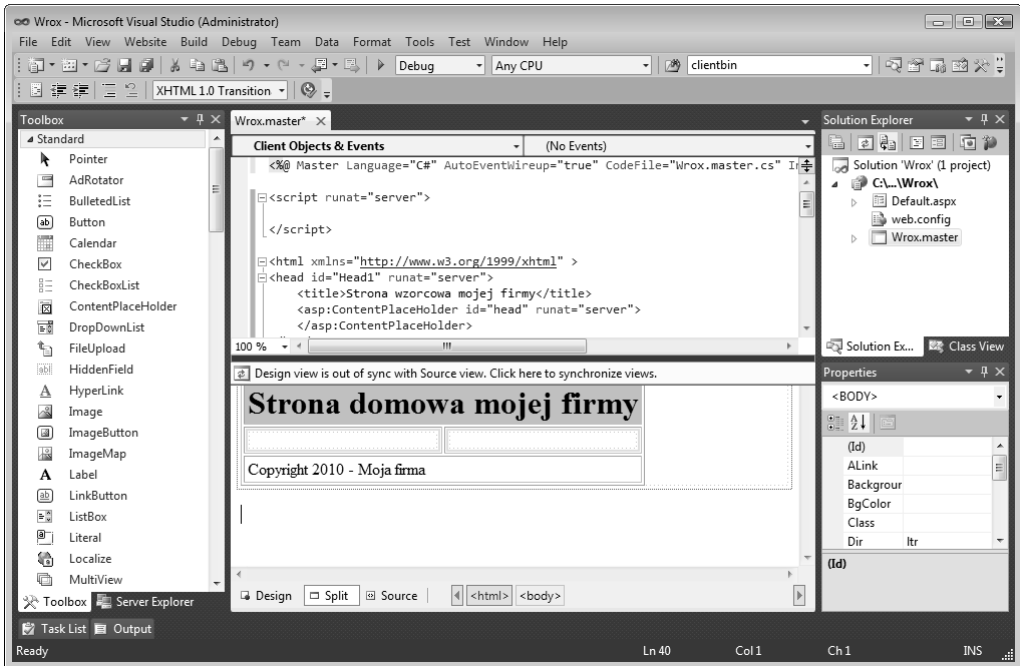
Strony wzorcowe w łatwy sposób udostępniają pewien wzorzec, który może być stosowany przez dowolną ilość stron ASP.NET w aplikacji. Pracę ze stronami wzorcowymi rozpoczyna się od utworzenia pliku wzorcowego, który będzie wskazywany przez **podstronę**, zwaną również **stroną z zawartością**. Strony wzorcowe mają rozszerzenie `.master`, podczas gdy strony z zawartością używają znanego już rozszerzenia `.aspx`. Strony z zawartością jako takie deklaruje się wewnątrz dyrektywy `Page`.

W pliku `.master`, który jest stroną wzorcową, można umieścić praktycznie wszystko. Może to być nagłówek, panel nawigacyjny lub stopka, które wykorzystywane są w całej aplikacji. Strona z zawartością obejmuje wtedy wszystkie elementy składające się na jej treść, ale bez elementów umieszczonych na stronie wzorcowej. W czasie wykonywania aplikacji silnik ASP.NET łączy te elementy dla użytkownika w jedną stronę. Na rysunku 5.2 pokazano diagram, który przedstawia zasadę działania tego mechanizmu.



**Rysunek 5.2**

Jedną z przyjemniejszych rzeczy w trakcie pracy ze stronami wzorcowymi jest to, że podczas tworzenia stron z zawartością w IDE można zobaczyć podgląd wzorca. W związku z tym, że podczas pracy widoczna jest cała strona, znacznie łatwiej można ją rozwijać. Podczas pracy ze stroną z zawartością wszystkie elementy pochodzące ze wzorca są wyszarzone i nie można ich modyfikować. Elementy, które można edytować, są wyróżnione. Te obszary robocze, zwane **obszarami zawartości**, są zdefiniowane w samej stronie wzorcowej. Na stronie wzorcowej określa się obszary strony, które mogą być używane przez strony z zawartością. Na stronie wzorcowej, jeżeli zachodzi taka potrzeba, może znaleźć się więcej takich obszarów zawartości. Na rysunku 5.3 pokazano stronę wzorcową z kilkoma obszarami zawartości.



Rysunek 5.3

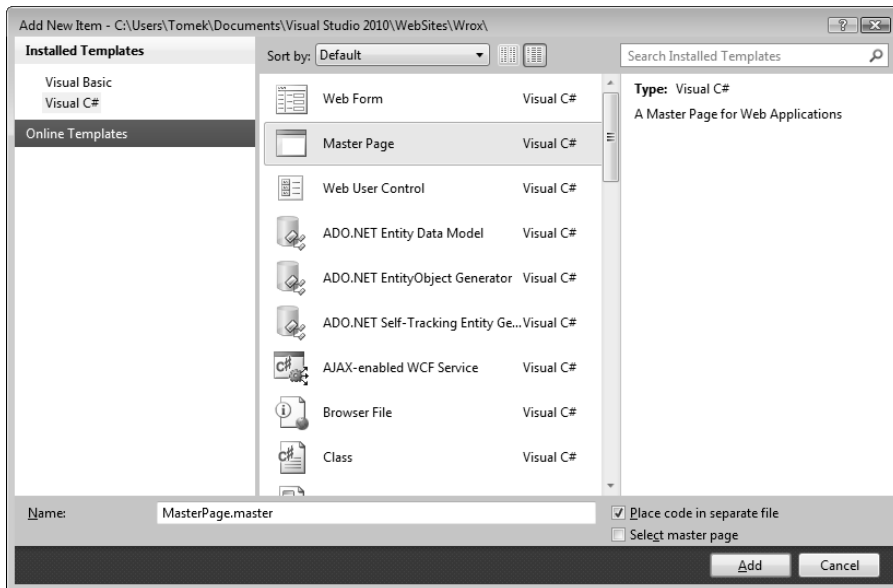
Patrząc na zrzut ekranu zaprezentowany na rysunku 5.3, można zauważyć dwa zdefiniowane obszary — obszary zawartości. Obszar zawartości wyróżniany jest w widoku *Design* za pomocą jasnego prostokąta z ramką w postaci punktów. W ten sposób reprezentowana jest kontrolka `ContentPlaceholder`. Oprócz tego, jeżeli najedziemy kursorem myszy nad obszar zawartości, nad kontrolką pojawi się jej nazwa (półprzezroczysta). Moment najechania wskaźnikiem mysy nad kontrolkę jest także ujęty na rysunku 5.3.

Dla firm i instytucji wykorzystanie stron wzorcowych to idealne rozwiązanie. Technologia ta doskonale odpowiada typowym wymaganiom biznesowym. Wiele firm tworzy wszystkie swoje strony intranetowe tak, że ich wygląd i obsługa są podobne. Mogą one teraz udostępnić działom firmy stronę wzorcową *.master* do stworzenia strony wydziału w intranecie. Taki proces zdecydowanie ułatwia zachowanie jednolitego wyglądu i podobnej obsługi strony w całym intranecie.

## Pisanie kodu stron wzorcowych

Przyjrzyjmy się teraz poszczególnym etapom budowania strony wzorcowej pokazanej wcześniej na rysunku 5.3. Stronę można stworzyć w dowolnym edytorze tekstowym, na przykład w Notatniku lub w narzędziu Visual Web Developer Express Edition, ale można także użyć nowego środowiska Visual Studio 2010. W niniejszym rozdziale pokażemy, jak to się robi właśnie w środowisku Visual Studio 2010.

Strony wzorcowe dodawane są do projektów w taki sam sposób jak zwykle strony *.aspx* — wystarczy podczas dodawania pliku do aplikacji wybrać opcję *Master Page*. Pokazano to na rysunku 5.4.



**Rysunek 5.4**

Dodawanie stron wzorcowych podobne jest do tworzenia zwykłych stron *.aspx*. W oknie dialogowym *Add New Item* znajduje się element, który pozwala utworzyć strony wzorcowe. Można skorzystać z modelu code-inline lub umieścić kod dla strony w oddzielnym pliku. Jeżeli nie umieścimy kodu serwera w oddzielnym pliku, będzie to oznaczało, że podczas tworzenia strony wzorcowej wykorzystywany jest model code-inline. Pozwala to utworzyć pojedynczy plik *.master*. Wybranie opcji *Place code in separate file* oznacza, że podczas tworzenia strony wykorzystywany jest model code-behind. Zaznaczenie pola wyboru *Place code in separate file* pozwala utworzyć pojedynczą stronę *.master* oraz skojarzony z nią plik *.master.vb* lub *.master.cs*. Istnieje także możliwość osadzenia strony wzorcowej w innej stronie wzorcowej poprzez wybranie opcji *Select master page*. Pokazane jest to w dalszej części rozdziału.

Przykładowa strona wzorcowa, która wykorzystuje model code-inline, pokazana jest na listingu 5.1.

**Listing 5.1.** Przykładowa strona wzorcowa

```
<%@ Master Language="VB" %>

<script runat="server">

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Strona wzorcowa mojej firmy</title>
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
        <table cellpadding="3" border="1">
            <tr style="background:silver">
```

```

        <td colspan="2">
            <h1>Strona domowa mojej firmy</h1>
        </td>
    </tr>
    <tr>
        <td>
            <asp:ContentPlaceHolder ID="ContentPlaceHolder1"
                runat="server">
            </asp:ContentPlaceHolder>
        </td>
        <td>
            <asp:ContentPlaceHolder ID="ContentPlaceHolder2"
                runat="server">
            </asp:ContentPlaceHolder>
        </td>
    </tr>
    <tr>
        <td colspan="2">
            Copyright 2010 - Moja firma
        </td>
    </tr>
</table>
</form>
</body>
</html>

```

Jest to prosta strona wzorcowa. Wspaniałe przy tworzeniu stron wzorcowych w Visual Studio 2010 jest to, że można pracować z nią w widoku kodu, ale można również przełączyć się do widoku projektanta i utworzyć tę stronę tak, jak tworzy się każdą inną stronę ASP.NET.

Rozpocniemy od analizy kodu strony wzorcowej. Pierwszy wiersz to dyrektywa:

```
<%@ Master Language="VB" %>
```

Zamiast używać dyrektywy `Page` tak, jak w zwykłych stronach `.aspx`, dla stron wzorcowych używa się dyrektywy `Master`. Pokazana strona wzorcowa korzysta tylko z jednego atrybutu, `Language`. Wartością atrybutu `Language` jest w tym przypadku `VB`, ale oczywiście można użyć `C#`, jeżeli tworzy się strony wzorcowe z wykorzystaniem tego języka.

Pozostała część kodu strony wzorcowej wygląda tak samo jak każda inna strona `.aspx`. Można korzystać z kontrolek serwerowych, zwykłego kodu w HTML-u i tekstu, obrazków, zdarzeń oraz wszystkich innych składników wykorzystywanych na stronach `.aspx`. Oznacza to, że strona wzorcowa może obsługiwać zdarzenie `Page_Load` oraz inne potrzebne zdarzenia.

W kodzie pokazanym na listingu 5.1 warto zwrócić uwagę na wykorzystanie nowej kontrolki serwerowej — kontrolki `<asp:ContentPlaceHolder>`. Kontrolka definiuje obszary strony wzorcowej, w którym strona z zawartością może umieszczać swoje elementy:

```

<tr>
    <td>
        <asp:ContentPlaceHolder ID="ContentPlaceHolder1"
            runat="server">
        </asp:ContentPlaceHolder>
    </td>
    <td>
        <asp:ContentPlaceHolder ID="ContentPlaceHolder2"

```

```

        runat="server">
        </asp:ContentPlaceHolder>
    </td>
</tr>

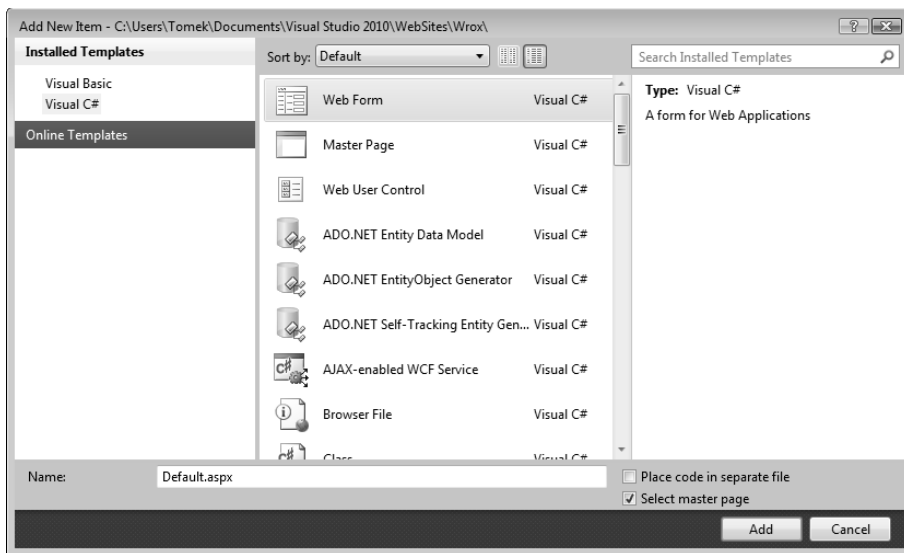
```

W pokazanej stronie wzorcowej istnieją dwa zdefiniowane obszary, gdzie można umieszczać zawartość. Strona zawiera oprócz tego obszar nagłówka i obszar stopki. Zdefiniowane są także dwa obszary, w których strony dziedziczące mogą wstawiać swoje własne elementy. Przyjrzyjmy się teraz, w jaki sposób strona z zawartością wykorzystuje tę stronę wzorcową.

## Pisanie kodu strony z zawartością

Gdy w aplikacji umieszczono już stronę wzorcową, to można użyć jej w stronach z zawartością w danej aplikacji. Kliknijmy prawym przyciskiem myszy w oknie *Solution Explorer* i wybierzmy opcję *Add New Item* w celu utworzenia w aplikacji nowej strony z zawartością.

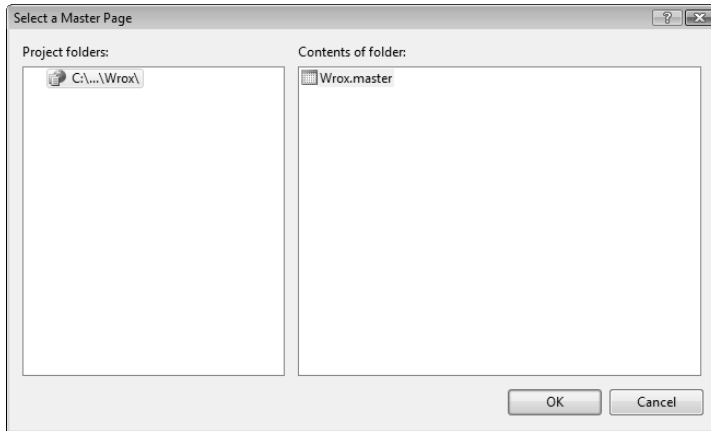
Aby utworzyć stronę z zawartością lub stronę, która wykorzystuje stronę wzorcową, w oknie dialogowym *Add New Item* spośród wielu opcji należy wybrać *Web Form* (zobacz rysunek 5.5). Zamiast jednak tworzyć typową stronę, trzeba zaznaczyć pole *Select master page*. Dzięki temu można połączyć tworzony formularz z wybraną stroną wzorcową.



**Rysunek 5.5**

Po nazwaniu strony z zawartością i kliknięciu przycisku *Add* w oknie dialogowym *Add New Item* pojawi się okno dialogowe *Select a Master Page* pokazane na rysunku 5.6.

Okno dialogowe pozwala wybrać stronę wzorcową, na podstawie której tworzona będzie strona z zawartością. Wyboru można dokonać spośród wszystkich stron wzorcowych dostępnych w aplikacji. W tym przypadku wybierzmy stronę wzorcową pokazaną na listingu 5.1 i kliknijmy OK. W ten sposób tworzona jest nowa strona z zawartością. Nowa strona jest zwykłą stroną *.aspx*. Zawiera tylko kilka wierszy kodu. Pokazano to na listingu 5.2.



Rysunek 5.6

**Listing 5.2.** Utworzona strona z zawartością

```

VB <%@ Page Language="VB" MasterPageFile="~/Wrox.master" Title="" %>

<script runat="server">

</script>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
  Runat="Server">
</asp:Content>

```

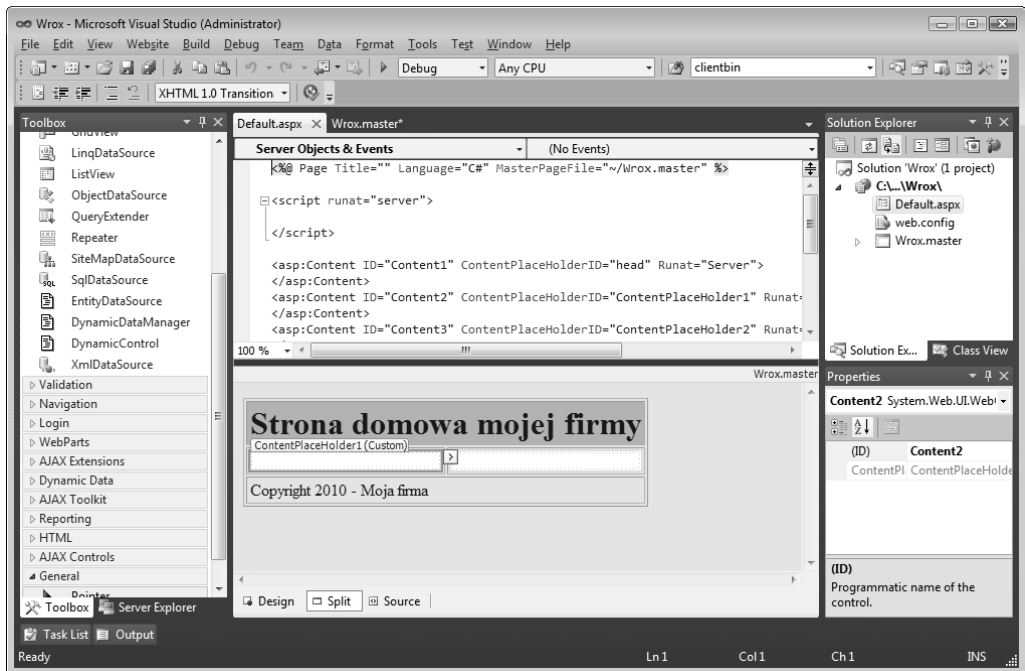
Strona z zawartością nie różni się bardzo od typowej strony *.aspx*, którą mieliśmy okazję stworzyć wcześniej. Dużą różnicą jest obecność w dyrektywie *Page* atrybutu *MasterPageFile*. Dzięki wykorzystaniu tego atrybutu można wskazać, że ta konkretna strona *.aspx* tworzona jest na podstawie innej strony. Położenie strony wzorcowej w aplikacji określone jest przez wartość atrybutu *MasterPageFile*.

Kolejną dużą różnicą jest to, że strona nie zawiera żadnych znaczników `<form id="form1" runat="server">` ani żadnych otwierających i zamykających znaczników języka HTML, które w normalnej stronie *.aspx* powinny być obecne.

Strona z zawartością może się wydawać prosta, ale jeżeli przełączymy się na widok projektanta w Visual Studio 2010, wtedy będzie można dostrzec możliwości, jakie dają strony wzorcowe. Dzięki dziedziczeniu wizualnemu otrzymujemy rezultat pokazany na rysunku 5.7.

Na pokazanym zrzucie ekranu można zauważyć, że aby wykorzystać stronę wzorcową z pliku *Wrox.master* poprzez dziedziczenie wizualne, wystarczy dodać atrybut *MasterPageFile* do dyrektywy *Page*. Korzystając z widoku projektanta w środowisku Visual Studio, można także zobaczyć, która strona wzorcowca jest obecnie używana. Nazwa aktywnej strony wzorcowej pokazywana jest w prawym górnym rogu widoku *Design*. Jeżeli spróbujemy kliknąć na wyszarzonym obszarze reprezentującym część dziedziczoną ze strony wzorcowej, wtedy będzie można zobaczyć, że kursor się zmienia. Sygnalizuje to, że wykonanie tej operacji jest niemożliwe. Pokazano to na rysunku 5.8 (kursor znajduje się nad słowem „firmy”).





Rysunek 5.7



Rysunek 5.8

Wszystkie wspólne obszary zdefiniowane w stronie wzorcowej są wyszarzone. Obszary zawartości oznaczone w stronie wzorcowej za pomocą kontrolki serwerowej `<asp:ContentPlaceHolder>` są pokazane wyraźnie i umożliwiają wstawienie dodatkowych elementów. Do tych obszarów zawartości można wstawić dowolne elementy. Odbyna się to tak samo, jak w przypadku zwykłych stron `.aspx`. Przykład wykorzystania strony wzorcowej w stronie z zawartością przedstawiono na listingu 5.3.

**Listing 5.3.** Strona z zawartością korzystająca ze strony wzorcowej `Wrox.master`

```
<%@ Page Language="VB" MasterPageFile="~/Wrox.master" %>
```

```
<script runat="server">
    Protected Sub Button1_Click(ByVal sender As Object,
        ByVal e As System.EventArgs)

        Label1.Text = "Witaj, " & TextBox1.Text & "!"
    End Sub
```

```

</script>

<asp:Content ID="Content1" ContentPlaceHolderId="ContentPlaceHolder1"
  runat="server">
  <b>Wpisz swoje imię:</b><br />
  <asp:Textbox ID="TextBox1" runat="server" />
  <br />
  <br />
  <asp:Button ID="Button1" runat="server" Text="Zatwierdź"
    OnClick="Button1_Click" /><br />
  <br />
  <asp:Label ID="Label1" runat="server" Font-Bold="True" />
</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderId="ContentPlaceHolder2"
  runat="server">
  <asp:Image ID="Image1" runat="server" ImageUrl="wrox.gif" />
</asp:Content>

```

```

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

```

```

<script runat="server">
  protected void Button1_Click(object sender, System.EventArgs e)
  {
    Label1.Text = "Witaj, " + TextBox1.Text + "!";
  }
</script>

```

Od razu można zauważyć różnice. Jak już wspomnieliśmy, strona nie posiada znacznika `<form id="form1" runat="server">` ani żadnego otwierającego lub zamykającego znacznika `<html>`. Te znaczniki nie są umieszczone na tej stronie, ponieważ znajdują się na stronie wzorcowej. Warto także zwrócić uwagę na nową kontrolkę serwerową `<asp:Content>`.

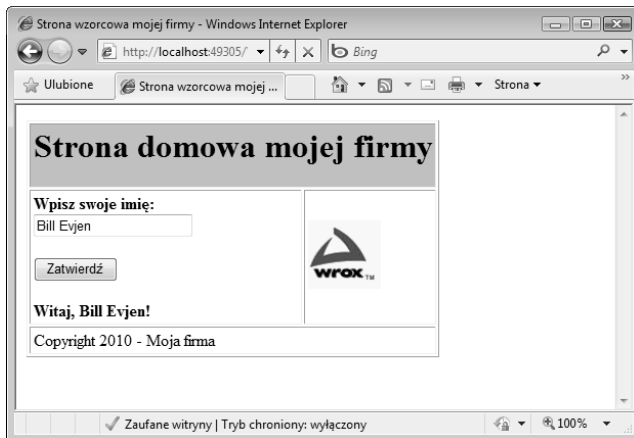
```

<asp:Content ID="Content1" ContentPlaceHolderId="ContentPlaceHolder1"
  runat="server">
  ...
</asp:Content>

```

Kontrolka serwerowa `<asp:Content>` jest zdefiniowanym obszarem zawartości, który pokrywa się z określoną kontrolką serwerową `<asp:ContentPlaceHolder>` strony wzorcowej. W pokazanym przykładzie można zauważyć, że kontrolka serwerowa `<asp:Content>` łączy się z kontrolką serwerową `<asp:ContentPlaceHolder>` strony wzorcowej posiadającą wartość ID równą `ContentPlaceHolder1`. Na stronie z zawartością nie trzeba się martwić ustawianiem położenia zawartości, ponieważ obszar zawsze jest definiowany przez stronę wzorcową. W związku z tym jedynym zmartwieniem jest umieszczenie właściwych elementów wewnątrz dostępnych sekcji. Pozostałą część pracy należy pozostawić stronie wzorcowej.

Tak samo jak w przypadku pracy ze zwykłymi stronami `.aspx`, tak i w stronach z zawartością można tworzyć dowolne procedury obsługi zdarzeń. W tym przypadku wykorzystujemy tylko jedną procedurę obsługi zdarzenia — wywoływana jest w momencie kliknięcia przycisku zatwierdzającego formularz. Utworzona strona `.aspx`, która korzysta ze strony wzorcowej oraz kilku wstawionych elementów, pokazana jest na rysunku 5.9.



Rysunek 5.9

## Łączenie różnych typów stron i języków

Jest jedna interesująca rzecz: podczas korzystania ze stron wzorcowych nie ma obowiązku korzystania z jednego określonego modelu pisania kodu (code-inline lub code-behind), nie ma też obowiązku stosowania jednego określonego języka. Można śmiało łączyć te elementy w aplikacji, ponieważ wszystko będzie działało doskonale.

Można zatem wykorzystać wcześniej utworzoną stronę wzorcową, jeśli się wie, że była utworzona z wykorzystaniem modelu code-inline, a następnie utworzyć stronę z zawartością za pomocą modelu code-behind. Na listingu 5.4 pokazano stronę, która została utworzona w modelu code-behind.

### Listing 5.4. Strona z zawartością korzystająca z modelu code-behind

#### .aspx (VB)

```
<%@ Page Language="VB" MasterPageFile="~/Wrox.master" AutoEventWireup="false"
    CodeFile="MyContentPage.aspx.vb" Inherits="MyContentPage" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderId="ContentPlaceHolder1"
    runat="server">
    <b>Wpisz swoje imię:</b><br />
    <asp:Textbox ID="TextBox1" runat="server" />
    <br />
    <br />
    <asp:Button ID="Button1" runat="server" Text="Zatwierdź" /><br />
    <br />
    <asp:Label ID="Label1" runat="server" Font-Bold="True" />
</asp:Content>

<asp:Content ID="Content3" ContentPlaceHolderId="ContentPlaceHolder2"
    runat="server">
    <asp:Image ID="Image1" runat="server" ImageUrl="wrox.gif" />
</asp:Content>
```

**VB (plik code-behind)**

```

Partial Class MyContentPage
    Inherits System.Web.UI.Page

    Protected Sub Button1_Click(ByVal sender As Object, _
        ByVal e As System.EventArgs) Handles Button1.Click

        Label1.Text = "Witaj, " & TextBox1.Text & "!"
    End Sub
End Class

```

**C# (plik code-behind)**

```

public partial class MyContentPage : System.Web.UI.Page
{
    protected void Button1_Click(object sender, EventArgs e)
    {
        Label1.Text = "Witaj, " + TextBox1.Text + "!";
    }
}

```

Pomimo że strona wzorcowa korzysta z modelu code-inline, można z łatwością tworzyć strony z zawartością (na przykład takie, jak ta pokazana na listingu 5.4), które wykorzystują model code-behind. Strony w dalszym ciągu będą działały doskonale.

Samo łączenie dwóch modeli pisania kodu to nie wszystko, co można zrobić przy stosowaniu stron wzorcowych. Można także mieszać języki programowania używane podczas pisania stron wzorcowych i stron z właściwą zawartością. To, że strona wzorcowa utworzona jest za pomocą języka C#, nie oznacza wcale, że język C# musi być używany na wszystkich stronach, które tej strony wzorcowej używają. Strony z zawartością mogą być tworzone także w języku Visual Basic. Jako dobry przykład stwórzmy stronę wzorcową w języku C#, która korzysta ze zdarzenia Page\_Load, a następnie dodajmy do tego stronę z zawartością, która została napisana w języku Visual Basic. Po zrobieniu wszystkiego jak należy spróbujmy uruchomić stronę. Działa ona doskonale. Oznacza to, że nawet wtedy, gdy strona wzorcowa napisana jest w jednym z dostępnych języków .NET, grupy programistyczne korzystające podczas tworzenia aplikacji ze strony wzorcowej mogą używać takiego języka .NET, jakiego tylko chcą. Warto pokochać otwartość, którą oferuje nam platforma .NET.

**Określanie, której strony wzorcowej użyć**

Pokazaliśmy już, że wskazanie wykorzystywanej strony wzorcowej z poziomu strony jest łatwe. W dyrektywie Page strony z zawartością wystarczy zastosować atrybut MasterPageFile:

```
<%@ Page Language="VB" MasterPageFile="~/Wrox.master" %>
```

Oprócz wskazywania używanej strony wzorcowej na poziomie strony istnieje drugi sposób wykonania tego samego zadania. Stronę wzorcową można określić za pomocą pliku konfiguracyjnego aplikacji *web.config*. Pokazano to na listingu 5.5.

**Listing 5.5.** Wskazywanie strony wzorcowej w pliku web.config

```
<configuration>
  <system.web>
    <pages masterPageFile="~/Wrox.master" />
  </system.web>
</configuration>
```

Wskazanie strony wzorcowej w pliku *web.config* spowoduje, że każda strona z zawartością tworzona w aplikacji będzie dziedziczyła po określonej stronie wzorcowej. Jeżeli stronę zadeklarujemy w pliku *web.config*, wtedy będzie można tworzyć dowolną ilość stron z zawartością i wszystkie one będą korzystały z podanej strony wzorcowej. Po wskazaniu strony w ten sposób dyrektywę Page można skonstruować w następujący sposób:

```
<%@ Page Language="VB" %>
```

Można w łatwy sposób przesłonić to ustawienie dla całej aplikacji, deklarując na stronie z zawartością inną stronę wzorcową:

```
<%@ Page Language="VB" MasterPageFile="~/MyOtherCompany.master" %>
```

Wskazanie strony wzorcowej w pliku *web.config* nie oznacza, że *wszystkie* strony *.aspx* muszą korzystać z tej strony wzorcowej. Jeżeli stworzymy i uruchomimy zwykłą stronę Web Form, środowisko ASP.NET będzie wiedziało, że nie jest to strona z zawartością. Strona ta zostanie przetworzona tak samo, jak zwykła strona *.aspx*.

Jeżeli zachodzi potrzeba zastosowania strony wzorcowej tylko do określonego zestawu stron (na przykład do stron umieszczonych w określonym katalogu aplikacji), wtedy można użyć elementu `<location>` w pliku *web.config*. Pokazano to na listingu 5.6.

**Listing 5.6.** Wskazywanie strony wzorcowej dla określonego katalogu w pliku web.config

```
<configuration>

  <location path="AdministrationArea">
    <system.web>
      <pages masterPageFile="~/WroxAdmin.master" />
    </system.web>
  </location>

</configuration>
```

Dodając w pliku *web.config* sekcję `<location>`, można wskazać, że określony katalog (AdministrationArea) będzie używał innego pliku strony wzorcowej. Zrealizowane jest to poprzez użycie atrybutu `path` elementu `<location>`. Wartością atrybutu `path` może być tak jak tutaj nazwa katalogu, ale może to być także określona strona — na przykład *AdminPage.aspx*.

## Praca z tytułem strony

Tworząc w aplikacji strony z zawartością, należy zwrócić uwagę na jedną rzecz. Domyślnie wszystkie takie strony posiadają tytuł zadeklarowany na stronie wzorcowej. Pierwotnie używaliśmy strony wzorcowej z tytułem Strona wzorcowa mojej firmy. Każda strona z zawartością tworzona na podstawie tej strony wzorcowej także miała tytuł Strona wzorcowa mojej firmy. Można to zmienić, korzystając z atrybutu `Title` dyrektywy `@Page` na stronie z zawartością. Można także programowo zmodyfikować tytuł wybranych stron z zawartością. Aby tego doko-

nać, należy w kodzie strony z zawartością użyć obiektu `Master`. Obiekt `Master` posiada wygodną właściwość o nazwie `Title`. Wartością tej właściwości jest tytuł strony używany przez strony z zawartością. Można go zmienić w sposób pokazany na listingu 5.7.

**Listing 5.7.** Kod zmieniający tytuł strony z zawartością

```
VB <%@ Page Language="VB" MasterPageFile="~/Wrox.master" %>

<script runat="server">
    Protected Sub Page_LoadComplete(ByVal sender As Object, _
        ByVal e As System.EventArgs)

        Master.Page.Title = "Strona została wygenerowana: " & _
            DateTime.Now.ToString()
    End Sub
</script>
```

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

<script runat="server">
    protected void Page_LoadComplete(object sender, EventArgs e)
    {
        Master.Page.Title = "Strona została wygenerowana: " +
            DateTime.Now.ToString();
    }
</script>
```

## Praca z kontrolkami i właściwościami strony wzorcowej

Pracując ze stronami wzorcowymi z poziomu stron z zawartością, można w łatwy sposób dostać się do kontrolki i właściwości udostępnionych przez stronę wzorcową. Strona wzorcowca wskazywana przez stronę z zawartością posiada właściwość `Master`. Dzięki tej właściwości można pobrać wartości kontrolki lub innych właściwości umieszczonych na samej stronie wzorcowej.

Aby zobaczyć przykład tego mechanizmu, stwórzmy na stronie wzorcowej GUID (unikatowy identyfikator), który będzie pobierany przez stronę z zawartością korzystającą z tej strony wzorcowej. Dla celów przykładu skorzystajmy ze strony wzorcowej pokazanej na listingu 5.1. Dodajmy jednak do niej kontrolkę serwerową `Label` oraz zdarzenie `Page_Load`. Pokazano to na listingu 5.8.

**Listing 5.8.** Strona wzorcowca tworząca identyfikator GUID podczas pierwszego żądania

```
VB <%@ Master Language="VB" %>

<script runat="server">
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
        If Not Page.IsPostBack Then
            Label1.Text = System.Guid.NewGuid().ToString()
        End If
    End Sub
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
```



```
ByVal e As System.EventArgs)
```

```
    Label1.Text = CType(Master.FindControl("Label1"), Label).Text
End Sub
```

```
Protected Sub Button1_Click(ByVal sender As Object, _
    ByVal e As System.EventArgs)
```

```
    Label2.Text = "Witaj, " & TextBox1.Text & "!"
```

```
End Sub
```

```
</script>
```

```
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
```

```
<asp:Content ID="Content2" ContentPlaceHolderId="ContentPlaceHolder1"
runat="server">
```

```
    <b>Twój numer GUID ze strony wzorcowej to:<br />
```

```
    <asp:Label ID="Label1" runat="server" /></b><p>
```

```
    <b>Wpisz swoje imię:</b><br />
```

```
    <asp:Textbox ID="TextBox1" runat="server" />
```

```
    <br />
```

```
    <br />
```

```
    <asp:Button ID="Button1" runat="server" Text="Zatwierdź"
```

```
        OnClick="Button1_Click" /><br />
```

```
    <br />
```

```
    <asp:Label ID="Label2" runat="server" Font-Bold="True" /></p>
```

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

```
<asp:Content ID="Content3" ContentPlaceHolderId="ContentPlaceHolder2"
runat="server">
```

```
    <asp:Image ID="Image1" runat="server" ImageUrl="Wrox.gif" />
```

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

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

```
<script runat="server">
```

```
    protected void Page_LoadComplete(object sender, EventArgs e)
```

```
    {
```

```
        Label1.Text = (Master.FindControl("Label1") as Label).Text;
```

```
    }
```

```
    protected void Button1_Click(object sender, EventArgs e)
```

```
    {
```

```
        Label2.Text = "<b>Witaj " + TextBox1.Text + "!"</b>";
```

```
    }
```

```
</script>
```

W przykładzie pokazanym na listingu 5.8 strona wzorcowa tworzy identyfikator GUID i zapamiętuje go w postaci wartości tekstowej w kontrolce serwerowej Label1. Sama kontrolka jest także umieszczona na stronie wzorcowej. Wartością ID kontrolki Label1 jest Label1. Strona wzorcowa generuje identyfikator GUID tylko podczas obsługi pierwszego żądania tej konkretnej strony. Od tego momentu strona generowana jest wraz z kontrolką uzupełnioną tą wartością.





```

                <asp:Label ID="Label1" runat="server" /></b>
            </td>
        </tr>
    </tr>
    <tr>
        <td>
            <asp:ContentPlaceHolder ID="ContentPlaceHolder1"
                runat="server">
            </asp:ContentPlaceHolder>
        </td>
        <td>
            <asp:ContentPlaceHolder ID="ContentPlaceHolder2"
                runat="server">
            </asp:ContentPlaceHolder>
        </td>
    </tr>
</tr>
<tr>
    <td colspan="2">
        Copyright 2010 - Moja firma
    </td>
</tr>
</table>
</form>
</body>
</html>

```

**C#**

```

<%@ Master Language="C#" %>

<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            Label1.Text = System.Guid.NewGuid().ToString();
        }
    }

    string m_PageHeadingTitle = "Moja firma";

    public string PageHeadingTitle
    {
        get
        {
            return m_PageHeadingTitle;
        }
        set
        {
            m_PageHeadingTitle = value;
        }
    }
</script>

```

W powyższym przykładzie pokazano stronę wzorcową, która udostępnia własną właściwość `PageHeadingTitle`. Domyślną wartością właściwości jest „Moja firma”. Wartość właściwości umieszczana jest w kodzie w HTML-u strony wzorcowej, pomiędzy znacznikami `<h1>`. Dzięki temu domyślna wartość staje się nagłówkiem używanym przez stronę wzorcową. Pomimo że strona wzorcowca posiada już wartość używaną jako tytuł, to każda strona z zawartością korzystająca ze strony wzorcowej może nadpisać nagłówek `<h1>`. Cały proces pokazany jest na listingu 5.11.

**Listing 5.11.** Strona z zawartością, która nadpisuje właściwość strony wzorcowej

```

VB <%@ Page Language="VB" MasterPageFile="~/Wrox.master" %>
<%@ MasterType VirtualPath="~/Wrox.master" %>

<script runat="server">
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
        Master.PageHeadingTitle = "Moja firma – Oddział X"
    End Sub
</script>

```

```

C# <%@ Page Language="C#" MasterPageFile="~/Wrox.master" %>
<%@ MasterType VirtualPath="~/Wrox.master" %>

<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        Master.PageHeadingTitle = "Moja firma – Oddział X";
    }
</script>

```

Z poziomu strony z zawartością można przypisać do właściwości udostępnianej przez stronę wzorcową pewną wartość. Umożliwia to właściwość `Master`. Jak widać, jest to całkiem proste. Należy pamiętać, że dostęp można uzyskać nie tylko do publicznych właściwości strony wzorcowej. Równie dobrze można wykorzystać metody, które zostały umieszczone na stronie wzorcowej.

Elementem, który umożliwia zastosowanie takich rozwiązań, jest dyrektywa `MasterType`. Dyrektywa `MasterType` pozwala utworzyć referencję ze ścisłą kontrolą typów do strony wzorcowej. Dzięki temu możliwe są odwołania do właściwości strony wzorcowej poprzez właściwość `Master`.

Wcześniej pokazaliśmy sposób uzyskania dostępu do umieszczonych na stronie wzorcowej kontrolki serwerowych za pomocą metody `FindControl()`. Metoda `FindControl()` działa dobrze, ale stosuje mechanizm późnego wiązania i w związku z tym jej wywołanie może się nie powieść, jeżeli kontrolka została usunięta ze znaczników strony. Należy zatem stosować technikę programowania defensywnego i zawsze sprawdzać, czy wartość zwrócona przez metodę `FindControl()` nie jest równa `null`. Korzystając z pokazanych mechanizmów (użycie publicznych właściwości zaprezentowano na listingu 5.10), można zastosować inny sposób udostępniania kontrolki serwerowych na stronie wzorcowej. Takie podejście okaże się bardziej efektywne.

Należy w tym celu udostępnić kontrolki serwerowe pod postacią publicznych właściwości. Pokazano to na listingu 5.12.

**Listing 5.12.** Udostępnianie kontrolki serwerowych strony wzorcowej poprzez publiczną właściwość

```

VB <%@ Master Language="VB" %>

<script runat="server">
    Public Property MasterPageLabel1() As Label
        Get
            Return Label1
        End Get
        Set(ByVal Value As Label)
            Label1 = Value
        End Set
    End Property

```

```

    End Set
  End Property
</script>

```

```

C# <%@ Master Language="C#" %>

<script runat="server">
    public Label MasterPageLabel1
    {
        get
        {
            return Label1;
        }
        set
        {
            Label1 = value;
        }
    }
</script>

```

W tym przypadku publiczna właściwość o nazwie `MasterPageLabel1` umożliwia uzyskanie dostępu do kontrolki `Label`, której ID jest równe `Label1`. Można teraz utworzyć egzemplarz właściwości `MasterPageLabel1` na stronie z zawartością i nadpisać dowolny atrybut kontrolki serwerowej `Label`. Jeżeli trzeba na przykład zmienić rozmiar identyfikatora GUID, który jest tworzony przez stronę wzorcową i wyświetlany przez kontrolkę serwerową `Label1`, wtedy wystarczy zwyczajnie nadpisać atrybut `Font.Size` tej kontrolki `Label`. Pokazano to na listingu 5.13.

#### Listing 5.13. Nadpisywanie atrybutu kontrolki `Label` umieszczonej na stronie wzorcowej

```

VB <%@ Page Language="VB" MasterPageFile="~/Wrox.master" %>
<%@ MasterType VirtualPath="~/Wrox.master" %>

<script runat="server">
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
        Master.MasterPageLabel1.Font.Size = 25
    End Sub
</script>

```

```

C# <%@ Page Language="C#" MasterPageFile="~/Wrox.master" %>
<%@ MasterType VirtualPath="~/Wrox.master" %>

<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        Master.MasterPageLabel1.Font.Size = 25;
    }
</script>

```

Takie podejście może być najbardziej efektywne spośród tych, które pozwalają pobrać referencję do dowolnej kontrolki serwerowej udostępnianej przez stronę wzorcową.

## Określanie domyślnej zawartości na stronie wzorcowej

Jak już mogliśmy się przekonać, strona wzorcowa pozwala określić obszary zawartości, które mogą być wykorzystywane przez strony z zawartością. Strony wzorcowe mogą zawierać tylko jeden obszar zawartości, ale równie dobrze mogą obejmować wiele takich obszarów. Dość interesującym rozwiązaniem związanym z obszarem zawartości jest to, że podczas tworzenia stron wzorcowych można zdefiniować domyślną treść dla tych obszarów. Ta domyślna zawartość może być zostawiona na swoim miejscu i nie musi być nadpisywana przez stronę z zawartością. Na listingu 5.14 pokazano stronę wzorcową z domyślną treścią umieszczaną w obszarze zawartości.

### Listing 5.14. Definiowanie domyślnej zawartości strony wzorcowej

```
<%@ Master Language="VB" %>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Moja firma</title>
  <asp:ContentPlaceHolder id="head" runat="server">
</asp:ContentPlaceHolder>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
      Tu jest jakaś domyślna zawartość.
    </asp:ContentPlaceHolder><p>
    <asp:ContentPlaceHolder ID="ContentPlaceHolder2" runat="server">
      Tu jest jeszcze więcej domyślnej zawartości.
    </asp:ContentPlaceHolder></p>
  </form>
</body>
</html>
```

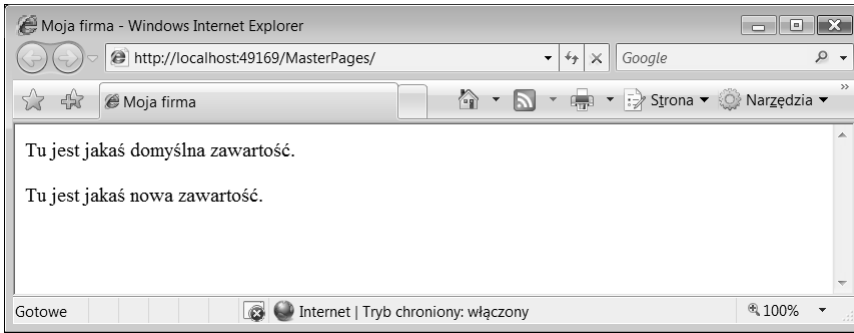
Aby umieścić domyślną treść w obszarze zawartości na stronie wzorcowej, wystarczy na samej stronie wzorcowej umieścić kontrolkę serwerową ContentPlaceHolder. Każda strona z zawartością dziedzicząca po tej stronie wzorcowej dziedziczy także domyślną treść. Na listingu 5.15 pokazano stronę z zawartością, która przesłania tylko jeden obszar zawartości strony wzorcowej.

### Listing 5.15. Przesłanie domyślnej zawartości na stronie z zawartością

```
<%@ Page Language="VB" MasterPageFile="~/MasterPage.master" %>

<asp:Content ID="Content3" ContentPlaceHolderId="ContentPlaceHolder2"
  runat="server">
  Tu jest jakaś nowa zawartość.
</asp:Content>
```

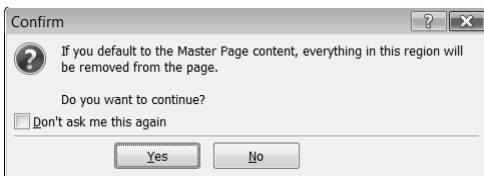
Pokazany kod pozwala utworzyć stronę z jednym obszarem zawartości, który dziedziczony jest po samej stronie wzorcowej, oraz z drugim obszarem, pochodzącym ze strony z zawartością (zobacz rysunek 5.10).



**Rysunek 5.10**

Kolejnym interesującym udogodnieniem jest to, że podczas pracy z obszarami zawartości w widoku projektanta Visual Studio 2010 pojawiają się inteligentne znaczniki, które ułatwiają pracę z domyślną treścią.

Rozpoczynając pracę ze stronami z zawartością, można zauważyć, że w kontrolkach serwerowych Content na początku pojawia się domyślna zawartość. Można zmienić tę zawartość, klikając inteligentny znacznik i wybierając z menu podręcznego opcję *Create Custom Content*. Opcja pozwala przesłonić zawartość strony wzorcowej i wstawić swoją własną zawartość. Po umieszczeniu własnych elementów wewnątrz obszaru zawartości inteligentny znacznik pokaże inną opcję — *Default to Master's Content*. Opcja pozwala przywrócić domyślną zawartość strony wzorcowej i usunąć wszystko, co zostało do tej pory umieszczone w obszarze zawartości. Skutkuje to powrotem do zawartości domyślnej. Po wybraniu tej opcji pojawi się ostrzeżenie informujące o tym, że wszystkie elementy wstawione w kontrolce serwerowej zostaną usunięte. Pokazano to na rysunku 5.11.

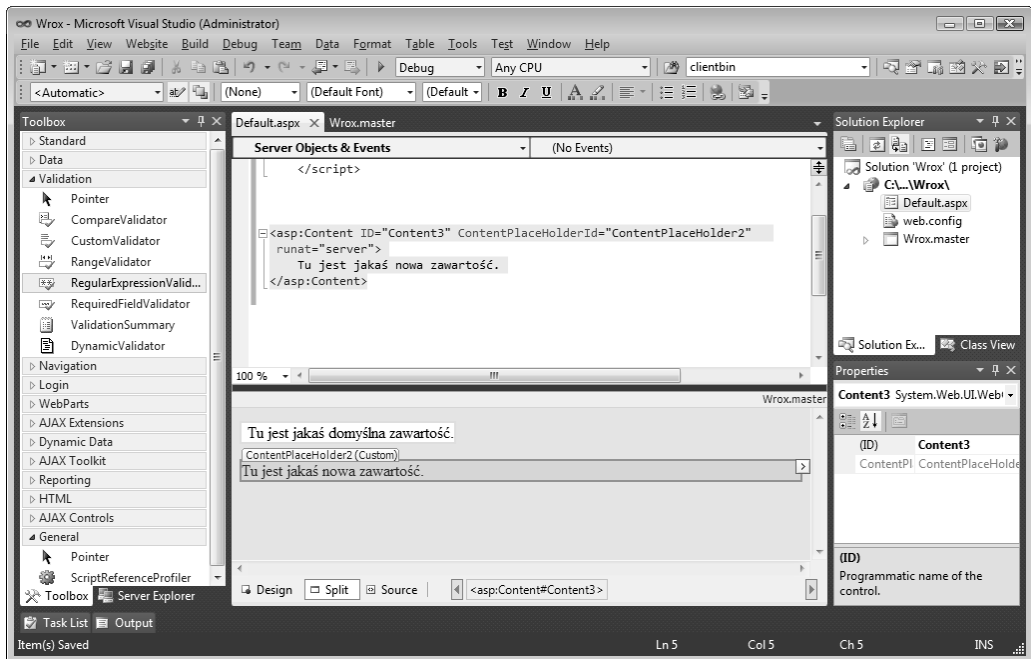


**Rysunek 5.11**

Po zmianie domyślnej zawartości kontrolki Content strona będzie wyglądała podobnie do tej z rysunku 5.12.

## Programowe przypisywanie strony wzorcowej

Z poziomu każdej strony z zawartością można w łatwy sposób programowo przypisać stronę wzorcową. Wykonuje się to za pomocą właściwości `Page.MasterPageFile`. Taka opcja może być użyta bez względu na to, czy inna strona wzorcowa została już przypisana w dyrektywie `@Page`.



Rysunek 5.12

by wykonać to zadanie, można skorzystać z właściwości `Page.MasterPageFile` w zdarzeniu `PreInit`. Zdarzenie `PreInit` jest pierwszym zdarzeniem w całym cyklu przetwarzania strony, w którym można uzyskać dostęp do właściwości strony. Z tego powodu jest to najlepsze miejsce, w którym do stron z zawartością można przypisać stronę wzorcową. `PreInit` to warte uwagi zdarzenie. Nabiera ono szczególnego znaczenia podczas pracy ze stronami wzorcowymi, ponieważ jest to jedyny moment, gdy zmiana może mieć wpływ zarówno na stronę wzorcową, jak i stronę z zawartością, zanim zostaną one połączone w jedną stronę właściwą. Na listingu 5.16 pokazano sposób programowego przypisania strony wzorcowej z poziomu strony z zawartością.

**Listing 5.16.** Korzystanie z procedury obsługi zdarzenia `Page_PreInit` w celu programowego przypisania strony wzorcowej

```

VB <%@ Page Language="VB" %>

<script runat="server">
    Protected Sub Page_PreInit(ByVal sender As Object, ByVal e As System.EventArgs)
        Page.MasterPageFile = "~/MyMasterPage.master"
    End Sub
</script>

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

<script runat="server">
    protected void Page_PreInit(object sender, EventArgs e)
    {
        Page.MasterPageFile = "~/MyMasterPage.master";
    }
</script>

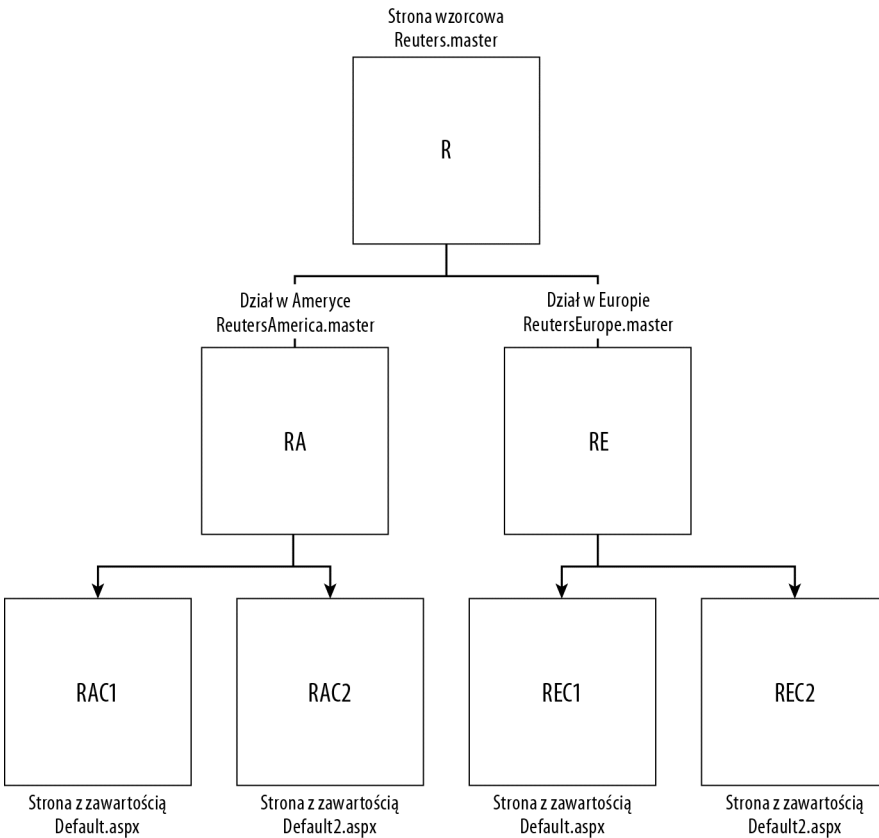
```

W tym przypadku strona jest generowana dynamicznie. Strona wzorcowa przypisywana jest do strony z zawartością na początku procesu konstrukcji strony. Należy zwrócić szczególną uwagę na to, że strona z zawartością musi się spodziewać kontrolerek Content, w przeciwnym razie pojawi się błąd.

## Osadzanie stron wzorcowych

Mamy nadzieję, że widoczna jest już potęga stron wzorcowych w obszarze tworzenia aplikacji sieciowych korzystających z jednego schematu. Do tej pory tworzyliśmy pojedynczą stronę wzorcową, z której korzystały strony z zawartością. Jednak większość portali firm i organizacji nie składa się z tylko dwóch warstw. W większych organizacjach istnieją różne oddziały i grupy, które mogą korzystać z różnych odmian stron wzorcowych. Jedna strona wzorcowa może wtedy zostać umieszczona w innej stronie wzorcowej. W ASP.NET jest to możliwe.

Przypuśćmy, że Reuters tworzy stronę wzorcową przeznaczoną do wykorzystania w intranecie całej firmy. Cała agencja Reuters może utworzyć jedną stronę wzorcową dla wszystkich w firmie, ale każdy dział agencji Reuters może udostępnić własne wzorce dla poszczególnych grup intranetu, które pozostają pod jej kontrolą. Dział Reutersa w Europie i dział Reutersa w Ameryce mogą mieć swoje własne strony wzorcowe. Pokazano to na rysunku 5.13.



**Rysunek 5.13**



Aby utworzyć strony działów Reutersa dla Europy i Reutersa dla Ameryki, programiści tworzą zwyczajne strony wzorcowe, które dziedziczą po globalnej stronie wzorcowej. Pokazano to na listingu 5.17.

#### Listing 5.17. Główna strona wzorcowca

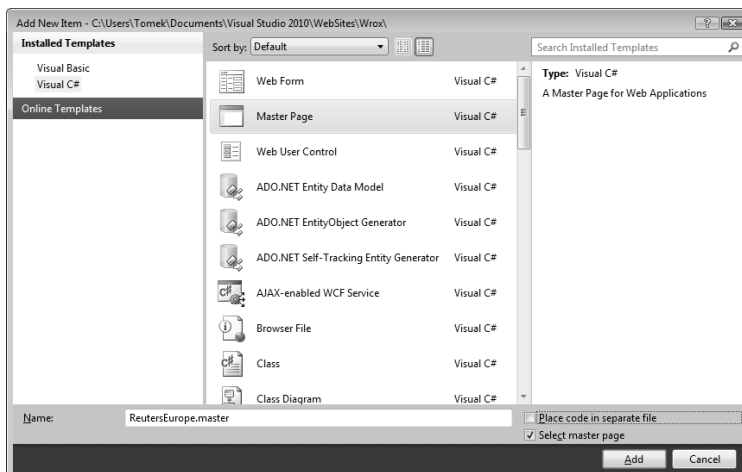
```
<%@ Master Language="VB" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Reuters</title>
  <asp:ContentPlaceHolder id="head" runat="server">
  </asp:ContentPlaceHolder>
</head>
<body>
  <form id="form1" runat="server">
    <p><asp:Label ID="Label1" runat="server" BackColor="LightGray"
      BorderColor="Black" BorderWidth="1px" BorderStyle="Solid"
      Font-Size="XX-Large"> Główna strona wzorcowca Reuters </asp:Label></p>
    <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
    </asp:ContentPlaceHolder>
  </form>
</body>
</html>
```

*Plik ReutersMain.master*

To prosta strona wzorcowca, ale doskonale nadaje się do pokazania funkcji osadzania. Główna strona wzorcowca jest używana globalnie w całej firmie. Obejmuje ona kontrolkę serwerową ContentPlaceHolder. Wartością ID kontrolki jest ContentPlaceHolder1.

Podczas tworzenia podstrony wzorcowej lub osadzonej strony wzorcowej wykonuje się te same zadania i w ten sam sposób, który był stosowany przy tworzeniu zwykłej strony wzorcowej. Z okna dialogowego *Add New Item* wybierzmy opcję *Master Page* i upewnijmy się, że zaznaczona jest opcja *Select master page*. Pokazano to na rysunku 5.14. W ten sposób po raz kolejny przejdziemy do okna dialogowego, które pozwala wybrać stronę wzorcową.



Rysunek 5.14

Na listingu 5.18 pokazano, w jaki sposób można korzystać ze strony wzorcowej w pliku podstrony wzorcowej.

#### Listing 5.18. Podstrona wzorcowa

```
<%@ Master Language="VB" MasterPageFile="-/ReutersMain.master" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

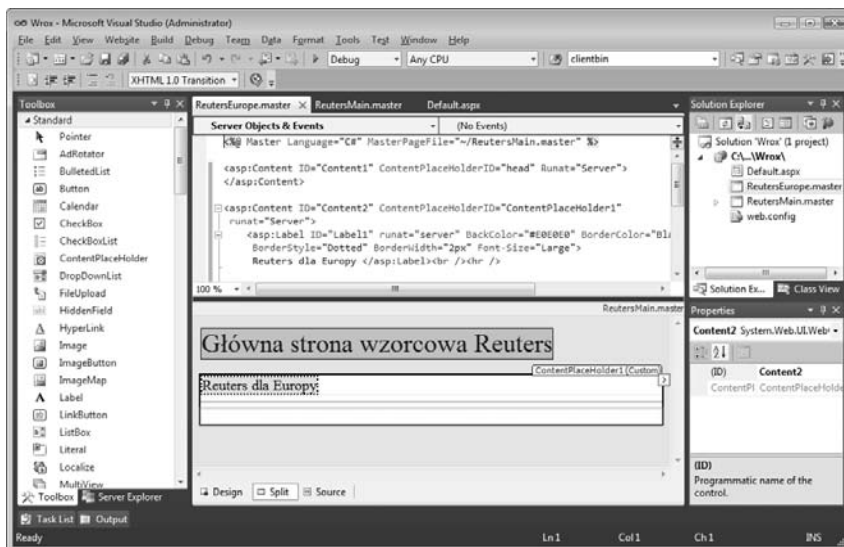
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
  runat="Server">
  <asp:Label ID="Label1" runat="server" BackColor="#E0E0E0" BorderColor="Black"
    BorderStyle="Dotted" BorderWidth="2px" Font-Size="Large">
    Reuters dla Europy </asp:Label><br /><hr />

    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
    </asp:ContentPlaceHolder>
</asp:Content>
```

[Plik ReutersEurope.master](#)

Patrząc na kod zaprezentowanej strony, można zauważyć, że nie różni się ona bardzo od typowej strony .aspx, która korzysta ze strony wzorcowej. Atrybut MasterPageFile jest ten sam. Umieszczony jest on jednak w dyrektywie @Master, podczas gdy zwykła strona używa dyrektywy @Page. Kontrolka Content2 także używa atrybutu ContentPlaceHolderID kontrolki Content. Atrybut łączy obszar zawartości z obszarem zawartości ContentPlaceHolder1, zdefiniowanym w głównej stronie wzorcowej.

Jedną z funkcji ASP.NET jest możliwość podglądu osadzonych stron wzorcowych bezpośrednio w widoku Design środowiska Visual Studio 2010. Wersje starsze od środowiska Visual Studio 2008 wyświetlały błąd, gdy próbowano pokazać osadzoną stronę wzorcową. Na rysunku 5.15 przedstawiono osadzoną stronę wzorcową w widoku Design środowiska Visual Studio 2010.



Rysunek 5.15

Wewnątrz podstrony wzorcowej pokazanej na listingu 5.18 można teraz używać tylu kontrolek serwerowych ContentPlaceHolder, ile tylko jest potrzebnych. Kontrolki te są dostępne dla stron z zawartością korzystających z tej strony wzorcowej. Na listingu 5.19 pokazano stronę z zawartością, która korzysta z podstrony wzorcowej ReutersEurope.master.

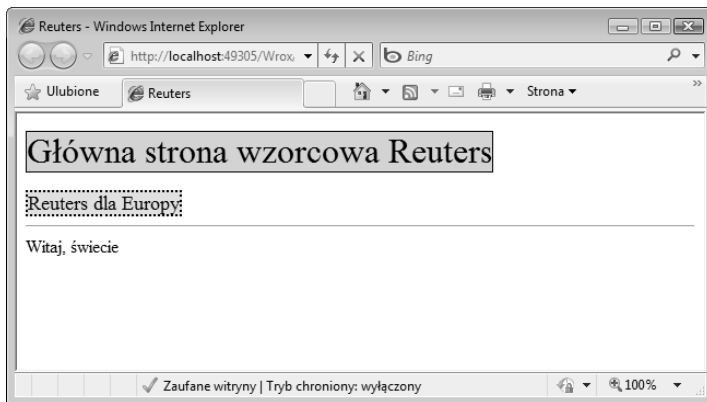
#### Listing 5.19. Strona z zawartością

##### Default.aspx

```
<%@ Page Language="VB" MasterPageFile="~/ReutersEurope.master" %>

<asp:Content ID="Content1" ContentPlaceHolderId="ContentPlaceHolder1"
  runat="server">
    Witaj, świecie
</asp:Content>
```

Jak można zauważyć, w pokazanej stronie wzorcowej wartością atrybutu MasterPageFile w dyrektywie Page jest utworzona podstrona wzorcowa. Dziedziczenie po stronie wzorcowej ReutersEurope powoduje, że wynikowa strona wzorcowa łączy w sobie obie strony (*ReutersMain.master* oraz *ReutersEurope.master*) w pojedynczą stronę wzorcową. Kontrolka Content strony z zawartością wskazuje na obszar zawartości również zdefiniowany przez podstronę wzorcową. Jak można zauważyć, w kodzie wykorzystano do tego atrybut ContentPlaceHolderId. W rezultacie otrzymujemy niezbyt ładną stronę końcową. Pokazano ją na rysunku 5.16.



Rysunek 5.16

Obserwując rezultat końcowy, można się przekonać, że podstrona wzorcowa działa całkiem nieźle.

## Strony wzorcowe dostosowane do przeglądarek

W wielu przypadkach programiści tworzą aplikacje, które będą wyświetlane w wielu różnych przeglądarkach. Niektórzy odbiorcy aplikacji mogą korzystać z przeglądarki Microsoft Internet Explorer, niektórzy mogą używać Firefoksa lub programu Google Chrome. Inni użytkownicy mogą przeglądać strony aplikacji na urządzeniu Pocket PC lub w telefonie komórkowym Nokia.

Z tego powodu ASP.NET pozwala umieścić na stronie z zawartością wiele stron wzorcowych. W zależności od przeglądarki użytej przez użytkownika końcowego silnik ASP.NET wstawi właściwy plik strony wzorcowej. Powstają zatem strony wzorcowe specyficzne dla danej przeglądarki i w ten sposób zapewniane są użytkownikom końcowym najlepsze możliwe doświadczenia pod-

czas oglądania stron. Można dzięki temu wykorzystać możliwości oferowane przez konkretne przeglądarki. Sposób wykorzystania wielu stron wzorcowych pokazany jest na listingu 5.20.

**Listing 5.20.** Strona z zawartością, która może pracować z więcej niż jedną stroną wzorcową

```
<%@ Page Language="VB" MasterPageFile="~/Wrox.master"
    Mozilla:MasterPageFile="~/WroxMozilla.master"
    Opera:MasterPageFile="~/WroxOpera.master" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderId="ContentPlaceHolder1"
    runat="server">
    Witaj, świecie
</asp:Content>
```

Jak można zauważyć na pokazanym listingu strony z zawartością, może ona działać z trzema rodzajami stron wzorcowych. Pierwszy z nich korzysta z atrybutu `MasterPageFile`. To ustawienie domyślne, używane przez te przeglądarki, które nie spełniają kryteriów nałożonych przez kolejne opcje. Oznacza to, że jeżeli żądanie nie pochodzi od przeglądarki Opera lub Mozilla, wtedy używana jest domyślna strona wzorcowa `Wrox.master`. Jeżeli jednak żądanie pochodzi od przeglądarki Opera, wtedy używany jest plik `WroxOpera.master`. Pokazano to na rysunku 5.17.



**Rysunek 5.17**

Listę dostępnych przeglądarek można znaleźć w katalogu `C:\Windows\Microsoft.NET\Framework\v4.0.xxxx\CONFIG\Browsers` na serwerze, na którym aplikacje będą umieszczane. Niektóre z opcji wypisano poniżej:

- |            |             |            |             |
|------------|-------------|------------|-------------|
| ■ avantgo  | ■ generic   | ■ MME      | ■ palm      |
| ■ cassio   | ■ goAmerica | ■ mozilla  | ■ panasonic |
| ■ Default  | ■ ie        | ■ netscape | ■ pie       |
| ■ docomo   | ■ Jataayu   | ■ nokia    | ■ webtv     |
| ■ ericsson | ■ jphone    | ■ openwave | ■ winwap    |
| ■ EZWap    | ■ legend    | ■ opera    | ■ xiino     |
| ■ gateway  |             |            |             |

Można oczywiście dodać swoje własne pliki `.browser`, jeżeli zajdzie taka potrzeba.

## Porządek wywoływania zdarzeń

Podczas pracy ze stronami wzorcowymi i stronami z zawartością korzysta się z tych samych zdarzeń (na przykład ze zdarzenia Load). Należy wiedzieć, które ze zdarzeń występują przed innymi. Tworzy się bowiem jedną stronę z dwóch klas i wymagany jest przy tym odpowiedni porządek. Gdy użytkownik wysła z przeglądarki żądanie strony z zawartością, wtedy zdarzenia wywoływane są w następującej kolejności:

1. **Inicjalizacja kontrolek potomnych strony wzorcowej** — wszystkie kontrolki znajdujące się na stronie wzorcowej są inicjalizowane.
2. **Inicjalizacja kontrolek potomnych strony z zawartością** — wszystkie kontrolki znajdujące się na stronie z zawartością są inicjalizowane.
3. **Inicjalizacja strony wzorcowej** — inicjalizowana jest strona wzorcowa.
4. **Inicjalizacja strony z zawartością** — inicjalizowana jest strona z zawartością.
5. **Wczytywanie strony z zawartością** — wczytywana jest strona z zawartością (wywoływane jest zdarzenie Page\_Load, a następnie zdarzenie Page\_LoadComplete).
6. **Wczytywanie strony wzorcowej** — wczytywana jest strona wzorcowa (wywoływane jest zdarzenie Page\_Load, a następnie zdarzenie Page\_LoadComplete).
7. **Wczytywanie kontrolek potomnych strony wzorcowej** — na stronę wczytywane są kontrolki serwerowe strony wzorcowej.
8. **Wczytywanie kontrolek potomnych strony z zawartością** — na stronę wczytywane są kontrolki serwerowe strony z zawartością.

Podczas tworzenia aplikacji należy zwrócić uwagę na porządek występowania zdarzeń. Jeżeli trzeba użyć wartości kontrolek serwerowych umieszczonych na stronie wzorcowej w określonej stronie z zawartością, to nie można tego zrobić bezpośrednio z poziomu procedury obsługi zdarzenia Page\_Load strony z zawartością. Dzieje się tak, ponieważ zdarzenie wywoływane jest przed zakończeniem działania procedury obsługi zdarzenia Page\_Load strony wzorcowej. Opisany problem zmusił twórców ASP.NET do wprowadzenia nowego zdarzenia Page\_LoadComplete. Zdarzenie Page\_LoadComplete występuje zaraz po zdarzeniu Page\_Load. Można skorzystać z tego porządku w celu pobrania wartości ze strony wzorcowej, pomimo że zawartość strony po wywołaniu zdarzenia nie jest jeszcze uzupełniona.

## Buforowanie stron wzorcowych

Podczas pracy z typowymi stronami *.aspx* można ustawić sposób buforowania danych wyjściowych za pomocą następującej konstrukcji (lub jej odmiany):

```
<%@ OutputCache Duration="10" Varybyparam="None" %>
```

Dzięki temu strona będzie przechowywana w pamięci podręcznej serwera przez 10 sekund. Wielu programistów korzysta z techniki buforowania danych wyjściowych w celu zwiększenia wydajności swoich stron ASP.NET. Takie rozwiązanie ma także sens w przypadku stron z danymi, które nie tracą swojej ważności zbyt szybko.

W jaki sposób można zastosować buforowanie danych wyjściowych stron ASP.NET w przypadku stron wzorcowych? Po pierwsze, nie można zastosować buforowania wyłącznie w odniesieniu do stron wzorcowych. Nie można umieścić dyrektywy OutputCache na samej

stronie wzorcowej. Jeżeli spróbujemy coś takiego zrobić, wtedy podczas drugiego pobrania strony pojawi się błąd. Aplikacja nie będzie mogła odnaleźć strony w pamięci podręcznej.

Aby możliwa była współpraca mechanizmu buforowania ze stronami wzorcowymi, należy umieścić dyrektywę `OutputCache` na stronie z zawartością. W ten sposób buforowana będzie treść zarówno strony z zawartością, jak i strony wzorcowej (należy pamiętać, że w tym miejscu jest to już jedna strona). Dyrektywa `OutputCache` umieszczona na stronie wzorcowej nie spowoduje, że pojawi się błąd. Strona nie będzie jednak buforowana. Dyrektywa działa tylko ze stronami z zawartością.

Inną nową i ciekawą funkcją środowiska ASP.NET 4 w obszarze buforowania jest możliwość sprawdzania stanu na poziomie kontrolki. Choć natychmiast przychodzi na myśl możliwość kontrolowania w ten sposób stanu takich kontrolki jak `GridView` i innych obejmujących rozbudowany stan, technikę tę można zastosować także do kontrolki `ContentPlaceholder`.

Można na przykład napisać kod podobny do poniższego:

```
<asp:ContentPlaceHolder ID="ContentPlaceholder1" runat="server"
  ViewStateMode="Disabled">
```

```
</asp:ContentPlaceHolder>
```

Przy takich ustawieniach kontrolka `ContentPlaceholder1` nie będzie korzystać ze stanu, nawet jeśli będą go używać pozostałe elementy strony. Opcja `ViewStateMode` przyjmuje wartości `Disabled`, `Enabled` i `Inherit`. Ustawienie `Disabled` powoduje wyłączenie obsługi stanu kontrolki, opcja `Enabled` włącza tę funkcję, a opcja `Inherit` spowoduje zastosowanie wartości określonej w dyrektywie `@Page`. Wyłączenie obsługi stanu prowadzi do poprawy wydajności stron.

## ASP.NET AJAX i strony wzorcowe

Wiele większych aplikacji ASP.NET w dzisiejszych czasach korzysta ze stron wzorcowych oraz możliwości, jakie daje ta technologia. Pozwala ona tworzyć portale na podstawie określonego wzorca. ASP.NET 4 obejmuje instalowaną domyślnie technologię ASP.NET AJAX. Okazuje się wkrótce, że strony wzorcowe współpracują z technologią AJAX całkiem niezłe.



### UWAGA

Technologia ASP.NET AJAX omówiona jest w rozdziale 18. tej książki.

Każda strona, która będzie korzystała z technologii AJAX, musi obejmować kontrolkę `ScriptManager`. Jeżeli tworzona strona z zawartością korzystająca z AJAX-a dziedziczy po stronie wzorcowej, wtedy kontrolka `ScriptManager` musi być umieszczona na stronie wzorcowej.



### UWAGA

Należy pamiętać, że na stronie może znajdować się tylko jedna kontrolka `ScriptManager`.

Skonfigurowanie strony wzorcowej pod kątem obsługi technologii AJAX nie jest trudne. Wystarczy dodać do strony wzorcowej kontrolkę ScriptManager. Przykład takiego rozwiązania przedstawia listing 5.21.

---

**Listing 5.21.** Strona wzorcową z obsługą technologii AJAX

---

```
<%@ Master Language="VB" %>

<script runat="server">

</script>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
  <asp:ContentPlaceHolder id="head" runat="server">
    <asp:ContentPlaceHolder>
  </head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:ScriptManager ID="ScriptManager1" runat="server" />
      <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

        <asp:ContentPlaceHolder>
      </div>
    </form>
  </body>
</html>
```

Jak można zauważyć na listingu 5.21, jedyną różnicą pomiędzy stroną wzorcową w technologii AJAX a standardową stroną wzorcową jest obecność w kodzie kontrolki serwerowej ScriptManager. Technikę tę należy stosować wtedy, gdy strona wzorcową korzysta z pewnych udogodnień technologii AJAX, a nawet wtedy, gdy strony z zawartością w ogóle nie korzystają z AJAX-a.

Kontrolka ScriptManager na stronie wzorcowej jest przydatna także wtedy, gdy istnieją pewne wspólne fragmenty kodu w języku JavaScript, wykorzystywane na wszystkich stronach aplikacji. Na listingu 5.22 pokazano, w jaki sposób można łatwo umieścić kod w języku JavaScript na wszystkich stronach korzystających ze strony wzorcowej.

---

**Listing 5.22.** Włączanie skryptów poprzez stronę wzorcową

---

```
<%@ Master Language="VB" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
  <asp:ContentPlaceHolder id="head" runat="server">
    <asp:ContentPlaceHolder>
  </head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:ScriptManager ID="ScriptManager1" runat="server">
        <Scripts>
      </div>
    </form>
  </body>
</html>
```

```

        <asp:ScriptReference Path="myScript.js" />
    </Scripts>
</asp:ScriptManager>
<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

    </asp:ContentPlaceHolder>
</div>
</form>
</body>
</html>

```

Dzięki zastosowaniu rozwiązania pokazanego w przykładzie plik *myScript.js* będzie umieszczany na każdej stronie z zawartością, która korzysta ze strony wzorcowej z obsługą AJAX-a. Jeżeli strona z zawartością także musi korzystać z technologii AJAX, wtedy nie można już tak zwyczajnie dodać do niej kontrolki `ScriptManager`. Strona z zawartością musi w takim przypadku korzystać z kontrolki `ScriptManager`, która jest już obecna na stronie wzorcowej.

W związku z powyższym, jeżeli strona z zawartością musi dołączyć do kontrolki `ScriptManager` dodatkowe składniki, wtedy może skorzystać z kontrolki serwerowej `ScriptManagerProxy` i za jej pomocą dostać się do kontrolki ze strony wzorcowej. Korzystanie z kontrolki `ScriptManagerProxy` pozwala dodać do kontrolki `ScriptManager` elementy, które są specyficzne dla danego egzemplarza strony z zawartością.

Na listingu 5.23 pokazano, w jaki sposób strona z zawartością może dodać kolejne skrypty do strony poprzez kontrolkę `ScriptManagerProxy`.

---

#### Listing 5.23. Wstawianie dodatkowych elementów za pomocą kontrolki `ScriptManagerProxy`

---

```

<%@ Page Language="VB" MasterPageFile="~/AjaxMaster.master" %>

<asp:Content ID="Content1" ContentPlaceHolderId="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderId="ContentPlaceHolder1"
runat="server">

    <asp:ScriptManagerProxy ID="ScriptManagerProxy1" runat="server">
        <Scripts>
            <asp:ScriptReference Path="myOtherScript.js" />
        </Scripts>
    </asp:ScriptManagerProxy>
</asp:Content>

```

W tym przypadku strona z zawartością korzysta z kontrolki `ScriptManagerProxy` w celu dodania na stronę kolejnych skryptów. Kontrolka `ScriptManagerProxy` działa dokładnie tak samo jak główna kontrolka `ScriptManager`. Różnica polega na tym, że kontrolka `ScriptManagerProxy` przewidziana jest do pracy ze stronami z zawartością dziedziczącymi po stronach wzorcowych. Kontrolka `ScriptManagerProxy` będzie współpracować z kontrolką `ScriptManager` i wykona wszystkie wymagane operacje.



## Podsumowanie

W czasie tworzenia aplikacji, w której prawie każda strona korzysta ze wspólnego nagłówka, stopki lub sekcji nawigacyjnej, dobrym rozwiązaniem jest zastosowanie stron wzorcowych. Strony wzorcowe są łatwe w implementacji i pozwalają wprowadzić zmiany do każdej strony aplikacji poprzez modyfikację pojedynczego pliku. Wystarczy sobie wyobrazić, o ile łatwiejsze jest takie zadanie przy aplikacji, która zawiera tysiące stron.

W niniejszym rozdziale opisano strony wzorcowe w ASP.NET i wyjaśniono sposób tworzenia stron wzorcowych w aplikacjach sieciowych. Oprócz podstawowych zagadnień poruszono także temat porządku wywoływania zdarzeń, buforowanie oraz dostosowanie stron wzorcowych do konkretnych przeglądarek. Na koniec mała rada. Jeżeli w aplikacji występuje pewien szablon, naturalnym odruchem powinno być wykorzystanie stron wzorcowych — technika ta daje olbrzymie możliwości.

Zaawansowane programowanie

# ASP.NET 4

z wykorzystaniem C# i VB

Platforma ASP.NET to główny konkurent języka Java w zakresie tworzenia aplikacji internetowych oraz dynamicznych stron internetowych. Każda jej wersja dostarcza wiele interesujących ulepszeń, a wśród nich te najważniejsze – pozwalające na zdjęcie z programisty obowiązku pisania dużych ilości nudnego kodu, bez którego jeszcze niedawno aplikacja nie mogłaby istnieć.

Niniejsza książka została napisana przez grupę wyjątkowych autorów. Bill Evjen to najaktywniejszy promotor technologii .NET, Scott Hanselman to główny menedżer w jednym z działów firmy Microsoft, prowadzący szkolenia dotyczące ASP.NET na całym świecie, a Devin Rader to pracownik firmy Infragistics. Ta doborowa trójka stworzyła świetny podręcznik, w całości poświęcony ASP.NET. Znajdziesz w nim informacje na temat stosowania języków VisualBasic oraz C# do tworzenia dynamicznych stron. Dowiesz się, jak wykorzystać kontrolki serwerowe, budować aplikacje z wykorzystaniem wzorca MVC oraz tchnąć życie w strony za pomocą technologii AJAX. Te i wiele innych bezcennych informacji, porad i wskazówek odkryjesz dzięki tej wyjątkowej książce!

**Sprawdź, jaka moc drzemie w platformie ASP.NET!**

▼ **Produktywność programistów**

▼ **Infrastruktura i tworzenie aplikacji ASP.NET**

▼ **Środowisko ADO.NET**

▼ **Kompilacja aplikacji ASP.NET**

▼ **Monitorowanie stanu aplikacji**

▼ **Kontrolki serwerowe**

▼ **Wykorzystanie stron wzorcowych**

▼ **Kompozycje i skórki**

▼ **Wykonywanie zapytań z wykorzystaniem LINQ**

▼ **Wykorzystanie formatu XML**

▼ **Bezpieczeństwo aplikacji ASP.NET**

▼ **Instrumentacja**

Nr katalogowy: 5767



Księgarnia internetowa:  
<http://helion.pl>



Zamówienia telefoniczne:  
**0 801 339900**



**0 601 339900**



**Helion**

Sprawdź najnowsze promocje:

🔴 <http://helion.pl/promocje>

🔴 Książki najchętniej czytane:

🔴 <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

🔴 <http://helion.pl/newosci>

**Helion SA**

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: [helion@helion.pl](mailto:helion@helion.pl)

<http://helion.pl>

**helion.pl**  
księgarnia  
internetowa

**Cena 199,00 zł**

ISBN 978-83-246-2846-9



9 788324 628469

Informatyka w najlepszym wydaniu