

Microsoft® **Visual Studio® 2010**

Mike Snell, Lars Powers

Poznaj środowisko Visual Studio 2010 Professional
i naucz się tworzyć mistrzowskie aplikacje

Jak wykorzystać technologię ASP.NET do tworzenia profesjonalnych witryn internetowych?

Jak tworzyć aplikacje biznesowe oparte na pakiecie Office?

Jak kompleksowo testować swój kod, zaprzegając do pracy Visual Studio?



Microsoft®
Visual Studio® 2010
KSIĘGA EKSPERTA

Hellon 

» Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

» 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 1991–2011

Microsoft Visual Studio 2010. Księga eksperta

Autorzy: Mike Snell, Lars Powers

Tłumaczenie: Tomasz Walczak

ISBN: 978-83-246-3029-5

Tytuł oryginału: [Microsoft Visual Studio 2010 Unleashed](#)

Format: 172×245, stron: 1264



Poznaj środowisko Visual Studio 2010 Professional i naucz się tworzyć mistrzowskie aplikacje

- Jak wykorzystać technologię ASP.NET do tworzenia profesjonalnych witryn internetowych?
- Jak tworzyć aplikacje biznesowe oparte na pakiecie Office?
- Jak kompleksowo testować swój kod, zaprzęgając do pracy Visual Studio?

Visual Studio 2010 to najnowsza wersja środowiska programistycznego firmy Microsoft. Każdy programista, który zdecyduje się wykorzystać bogaty zestaw zgromadzonych tu narzędzi, osiągnie maksymalną produktywność w pracy i będzie mógł tworzyć kod działający w systemie Windows oraz w sieci. Dodatkowo będzie mieć do dyspozycji technologię Silverlight i możliwość budowania aplikacji w chmurze, z użyciem platformy Microsoftu – Azure. Nie koniec na tym. W Visual Studio 2010 pojawiły się kolejne innowacje. Znajdziesz tu nowy edytor kodu, oparty na platformie WPF, rozszerzenia środowiska IDE związane z platformą MEF oraz możliwość pisania skryptów za pomocą technologii Silverlight. Ponadto języki platformy .NET wzbogacono o obsługę programowania dynamicznego. Pojawił się nowy język F#, służący do programowania funkcyjnego, oraz mechanizmy usprawniające szybkie pisanie kodu wyższej jakości.

„Microsoft Visual Studio 2010. Księga eksperta” zawiera kompletne omówienie środowiska Visual Studio 2010, a skoncentrowanie się na wersji Professional pozwoliło autorom na stworzenie opisu bardziej szczegółowego niż kiedykolwiek wcześniej. Dzięki temu podręcznikowi nauczysz się w pełni wykorzystywać możliwości platformy .NET, w tym technologii WPF (pozwalającej na tworzenie bogatych klientów), technologii WCF (stworzonej do budowania dynamicznych rozwiązań opartych na usługach) czy też technologii WF (umożliwiającej rozwijanie ustrukturyzowanych programów na podstawie procesów biznesowych). Znajdziesz tu także omówienie nowych narzędzi Microsoftu, przeznaczonych do testowania, instrumentacji aplikacji i analizowania kodu.

- Języki platformy .NET
- Przeglądarki i eksploratory
- Tworzenie projektów WPF
- Szablony XML
- Testowanie, refaktoryzacja i diagnozowanie kodu
- Platforma MEF
- Tworzenie aplikacji ASP.NET
- Bogate i inteligentne interfejsy użytkownika
- Złożone aplikacje internetowe
- Silverlight 4.0
- Aplikacje biznesowe oparte na pakiecie Office
- Technologia Windows Azure i aplikacje działające w chmurze

Wykorzystaj zdobytą wiedzę i zostań mistrzem programowania!

Spis treści

O autorach	21
Wprowadzenie	23
Dla kogo przeznaczona jest ta książka?	23
Koncentracja na wersji Visual Studio Professional	24
Materiały do pobrania	25
Jak zorganizowana jest książka?	25
Część I: Wprowadzenie do Visual Studio 2010	25
Część II: Szczegółowe omówienie środowiska IDE	25
Część III: Tworzenie kodu i zarządzanie nim	25
Część IV: Wzbogacanie środowiska Visual Studio	26
Część V: Tworzenie aplikacji dla przedsiębiorstw	26
Konwencje używane w książce	26
Część I Wprowadzenie do Visual Studio 2010	27
Rozdział 1. Krótki przegląd środowiska Visual Studio 2010	29
Produkty z rodziny Visual Studio	31
Wersje Express	32
Wersja Professional	33
Wersja Premium	34
Wersja Ultimate	34
MSDN	35
Powiązane narzędzia	36
Języki, platformy i szablony aplikacji	40
Wybieranie języków programowania	40
Platformy .NET	42
Wiele aspektów aplikacji dla platformy .NET	43
Tworzenie bogat(szy)ch interfejsów sieciowych	46
Formularze Windows	46
Platforma WPF	47
Rozwiązania oparte na pakiecie Office	50
Tworzenie klientów sieciowych	53
Budowanie witryn za pomocą technologii ASP.NET	54
Wzbogacanie aplikacji za pomocą AJAX-a	58
Programowanie za pomocą technologii Silverlight	59
Programowanie pod kątem chmury	63
Tworzenie aplikacji działających w chmurze	64
Uruchamianie aplikacji w chmurze	64
Publikowanie aplikacji w chmurze	66

Praca z danymi	67
Projektowanie danych	67
Oddzielanie projektu danych od modelu źródła danych	68
Tworzenie aplikacji okresowo nawiązujących połączenie	71
Tworzenie powiązanych rozwiązań opartych na usługach	71
Tworzenie aplikacji i procesów biznesowych	72
Tworzenie i konsumowanie usług	74
Podsumowanie	76
Rozdział 2. Środowisko IDE Visual Studio	77
Instalowanie środowiska Visual Studio	78
Wybór języka	78
Konfigurowanie środowiska programistycznego	79
Strona startowa	83
Opcje uruchomieniowe	84
Pierwszy projekt	85
Wybieranie docelowego środowiska	86
Poruszanie się po środowisku IDE	88
Menu	89
Liczne paski narzędzi	94
Dostosowywanie pasków narzędzi	95
Solution Explorer	97
Edytory tekstu	98
Graficzne okna projektowe	102
Okno narzędzi	103
Okno Properties	104
Zarządzanie wieloma oknami środowiska IDE	105
Przyczepianie	105
Dokowanie	106
Poruszanie się między oknami środowiska	108
Dostosowywanie czcionki	110
Podsumowanie	111
Rozdział 3. Języki platformy .NET	113
Wprowadzenie do języków	114
Programowanie obiektów	115
Typy, zmienne i stałe	129
Operatory	133
Podejmowanie decyzji i rozgałęzianie kodu	135
Pętle	138
Praca z grupami elementów	141
Programowanie z wykorzystaniem atrybutów	145
Tworzenie i zgłaszanie zdarzeń	147
Mechanizmy języka	150
Wykrywanie typu zmiennej na podstawie przypisania	151
Tworzenie obiektów i ustawianie ich właściwości w jednym wierszu kodu	153

Definiowanie kolekcji i inicjowanie ich wartości (nowość)	155
Tworzenie egzemplarzy nieistniejących klas	156
Dodawanie metod do istniejących klas	157
Dodawanie logiki biznesowej do wygenerowanego kodu	158
Dostęp do danych i pobieranie ich za pomocą języków .NET	160
Pisanie prostych funkcji anonimowych w kodzie	162
Dzielenie podzespółów na wiele plików	164
Bezpośrednie korzystanie z elementów XML (tylko w języku Visual Basic)	164
Usuwanie nieużywanych argumentów z metod obsługi zdarzeń (tylko w Visual Basic)	166
Automatyczne generowanie kodu do obsługi właściwości (nowość w Visual Basic)	166
Rezygnacja z podkreślenia przy kontynuowaniu wiersza w języku Visual Basic (nowość)	167
Korzystanie z dynamicznych obiektów i języków (nowość)	167
Kowariancja i kontrawariancja (nowość)	173
Platforma .NET	175
Podsumowanie	177

Część II Szczegółowe omówienie środowiska IDE 179

Rozdział 4. Rozwiązania i projekty 181

Wprowadzenie do rozwiązań	182
Tworzenie rozwiązania	183
Korzystanie z rozwiązań	188
Zapoznavanie się z projektami	194
Tworzenie projektu	194
Używanie plików definicji projektu	198
Praca z projektami	203
Podsumowanie	209

Rozdział 5. Przeglądarki i eksploratory 211

Okno Solution Explorer	212
Ikony i wskazówki graficzne	213
Zarządzanie rozwiązaniami	217
Zarządzanie projektami	218
Okno Class View	219
Pasek narzędzi	220
Pasek wyszukiwania	220
Panel obiektów	221
Panel składowych	222
Okno Server Explorer	223
Połączenia z danymi	224
Komponenty serwera	225
Okno Object Browser	229
Zmiana zasięgu	229
Przeglądanie obiektów	230

Okno Document Outline	232
Modyfikowanie elementów	233
Podsumowanie	234
Rozdział 6. Wprowadzenie do edytorów i okien projektowych	235
Podstawy	236
Edytor tekstu	237
Okna projektowe środowiska Visual Studio	240
Pisanie kodu w edytorze	240
Otwieranie edytora	241
Pisanie kodu	241
Budowa okna edytora kodu	243
Narzędzia do nawigowania po kodzie	246
Przeszukiwanie dokumentów	249
Diagnozowanie w edytorze kodu	258
Drukowanie kodu	262
Używanie okna Code Definition	262
Tworzenie i modyfikowanie dokumentów oraz szablonów XML	264
Generowanie szablonów	265
Projektowanie szablonów XML	265
Edycja arkuszy stylów XSLT	269
Używanie kaskadowych arkuszy stylów	270
Dodawanie zasad stylów	271
Definiowanie atrybutów arkuszy stylów	271
Tworzenie aplikacji klienckich dla systemu Windows	271
Tworzenie projektów aplikacji dla systemu Windows	272
Tworzenie projektów WPF	280
Tworzenie formularzy sieciowych	282
Projektowanie aplikacji opartych na formularzach sieciowych	283
Tworzenie komponentów i kontrolki	289
Tworzenie nowego komponentu lub kontrolki	290
Uwagi na temat pisania kodu komponentów	291
Tworzenie klas za pomocą okna Class Designer	293
Tworzenie diagramu klasy	293
Dodawanie elementów do diagramu	294
Definiowanie relacji między klasami	296
Definiowanie metod, właściwości, pól i zdarzeń	298
Podsumowanie	299
Rozdział 7. Społeczność .NET — interakcje w internecie	301
Możliwości Visual Studio związane ze społecznością	302
Strona startowa Visual Studio	303
Dostęp do systemu pomocy	312
Zarządzanie ustawieniami systemu pomocy	312
Zgłaszanie błędów i przysyłanie innych informacji zwrotnych	323
Przykłady	326

Wykrywanie i wykorzystanie współużytkowanych zasobów	328
Rodzaje współużytkowanych zasobów	328
Wyszukiwanie odpowiednich zasobów	328
Instalowanie i przechowywanie udostępnianych zasobów	330
Własny wkład w społeczność	332
Pakiety startowe a szablony	333
Tworzenie szablonów projektów	333
Tworzenie szablonów elementów	340
Tworzenie pakietów w celu ich rozpowszechniania	341
Podsumowanie	356

Część III Tworzenie kodu i zarządzanie nim 359

Rozdział 8. Korzystanie z narzędzi zwiększających produktywność 361

Podstawowe narzędzia pomocnicze edytorów tekstu	364
Śledzenie zmian	364
Wskazówki dotyczące problemów	364
Aktywne odnośniki	365
Kolorowanie składni	365
Schematy i nawigacja	367
Schematy kodu	367
Nawigowanie po znacznikach	370
Inteligentne znaczniki i operacje	372
Okno projektowe HTML-a	372
Okno projektowe formularzy Windows	373
Edytor kodu	373
Mechanizm IntelliSense	375
Uzupełnianie słów (Complete Word)	375
Okno z informacjami podręcznymi (Quick Info)	377
Okno z listą składowych (List Members)	378
Okno z informacjami o parametrach (Parameter Info)	379
Porządkowanie instrukcji Using	380
Fragmenty kodu i kod szablonowy	380
Dopasowywanie nawiasów	391
Dostosowywanie mechanizmu IntelliSense do własnych potrzeb	392
Okno Task List	393
Zadania związane z komentarzami	394
Zadania związane ze skrótami	395
Zadania użytkownika	396
Podsumowanie	396

Rozdział 9. Testowanie kodu 397

Podstawy testów jednostek	399
Tworzenie projektu testów	399
Generowanie testów na podstawie istniejącego kodu	402
Pisanie testów jednostek	405

Uruchamianie testów jednostek	407
Konfigurowanie opcji i ustawień testów	409
Platforma testów jednostek	413
Klasa TestContext	413
Klasy atrybutów testów	415
Operacje wykonywane przed testami jednostek i po nich	418
Klasy asercji	420
Testowanie wyjątków	422
Tworzenie testów jednostek zależnych od danych	423
Pisanie testów jednostek działających w ASP.NET	428
Definiowanie atrybutów środowiska ASP.NET	428
Generowanie testów jednostek ASP.NET	429
Konfigurowanie hosta projektu testów	430
Konfigurowanie atrybutów testów jednostek	431
Definiowanie połączenia za pomocą okna dialogowego Properties	431
Tworzenie testów uporządkowanych	433
Organizowanie testów	434
Okno Test View	434
Okno Test List Editor	435
Podsumowanie	438
Rozdział 10. Refaktoryzacja kodu	439
Podstawy refaktoryzacji w Visual Studio	441
Uruchamianie narzędzi do refaktoryzacji	441
Podgląd zmian	445
Zmienianie nazw	446
Uruchamianie operacji Rename	447
Używanie okna dialogowego Rename	449
Wyodrębnianie metod	450
Uruchamianie refaktoryzacji Extract Method	451
Wyodrębnianie metod	451
Generowanie szkieletu metody	457
Wyodrębnianie interfejsów	459
Uruchamianie refaktoryzacji Extract Interface	459
Wyodrębnianie interfejsów	459
Refaktoryzacja parametrów	462
Usuwanie parametrów	462
Zmiana kolejności parametrów	463
Hermetyzacja pól	465
Uruchamianie refaktoryzacji Encapsulate Field	465
Okno dialogowe Encapsulate Field	465
Podsumowanie	466
Rozdział 11. Diagnozowanie kodu	467
Podstawy diagnozowania	468
Scenariusz	469
Wiele etapów diagnozowania	469

Diagnozowanie aplikacji (samodzielne sprawdzanie)	470
Podsumowanie podstaw diagnozowania	481
Debugger środowiska Visual Studio	481
Menu i pasek narzędzi Debug	482
Opcje diagnozowania	487
Wkraczanie w kod, wychodzenie z niego i przeskakiwanie	488
Określanie warunków wstrzymania wykonywania kodu	494
Korzystanie z punktów śledzenia (opcja When Hit)	503
Podglądanie danych w debugerze	505
Korzystanie z funkcji „zmień i kontynuuj”	511
Diagnozowanie zaawansowane	513
Zdalne diagnozowanie	513
Diagnozowanie usług WCF	514
Diagnozowanie aplikacji wielowątkowych	515
Diagnozowanie aplikacji równoległych	520
Diagnozowanie skryptów działających po stronie klienta	526
Diagnozowanie informacji o awarii (plików zrzutów)	527
Podsumowanie	530
Rozdział 12. Wdrażanie kodu	533
Przegląd sposobów wdrażania rozwiązań po stronie klienta	534
Wprowadzenie do wdrażania metodą ClickOnce	534
Wprowadzenie do wdrażania za pomocą instalatora systemu Windows	535
Publikowanie projektów za pomocą technologii ClickOnce	536
Publikowanie projektów za pomocą instalatora systemu Windows	539
Edytor File System	542
Edytor Registry	543
Edytor File Types	544
Edytor User Interface	545
Edytor Custom Actions	546
Edytor Launch Conditions	546
Publikowanie witryn i aplikacji ASP.NET	549
Korzystanie z narzędzia Web Deployment Tool	550
Korzystanie z narzędzia Copy Web Site Tool	554
Podsumowanie	556
Część IV Wzbogacanie środowiska Visual Studio	557
Rozdział 13. Wprowadzenie do obiektowego modelu automatyzacji	559
Przegląd obiektowego modelu automatyzacji	561
Wersje modelu obiektowego	561
Kategorie automatyzacji	563
Obiekt główny DTE (DTE2)	564
Obiekty Solution i Project	565
Kontrolowanie projektów wchodzących w skład rozwiązania	568
Dostęp do kodu projektu	569

Okna	572
Dostęp do okien	572
Interakcja z oknami	573
Okna tekstowe i panele	576
Rodzaje okien narzędzi	578
Okna połączone	586
Paski poleceń	588
Dokumenty	592
Dokumenty tekstowe	593
Obiekty polecenia	604
Wykonywanie poleceń	606
Dodawanie klawiszy skrótów	607
Obiekty debugera	608
Zdarzenia automatyzacji	608
Podsumowanie	609
Rozdział 14. Tworzenie makr	611
Rejestrowanie makr	613
Korzystanie z okna Macro Explorer	614
Pisanie makr za pomocą środowiska IDE Macros	616
Projekty makr	617
Pisanie makr	620
Diagnozowanie	623
Obsługa zdarzeń	623
Wywoływanie makr	629
Podsumowanie	633
Rozdział 15. Tworzenie dodatków i kreatorów	635
Tworzenie pierwszego projektu dodatku	637
Ustawianie parametrów dodatku	637
Struktura dodatków	645
Cykl życia dodatków	645
Reagowanie na polecenia	651
Zarządzanie dodatkami	653
Przykładowy dodatek — paleta do wybierania kolorów	654
Początkowe operacje	655
Tworzenie klasy kontrolki użytkownika	655
Dopracowywanie klasy Connect	659
Udostępnianie ustawień dodatku	662
Tworzenie kreatorów dla środowiska Visual Studio	677
Analiza struktury kreatorów	677
Tworzenie kreatorów typu Add New Item	680
Podsumowanie	686
Rozdział 16. Rozbudowywanie edytora kodu przy użyciu platformy MEF	687
Problem z rozszerzaniem	688
Tworzenie dynamicznych aplikacji	688

Architektura platformy MEF	689
Reguły działania platformy MEF	689
Korzystanie z platformy MEF	690
Edytor środowiska Visual Studio i platforma MEF	691
Punkty dołączania rozszerzeń do edytora	692
Korzystanie z pakietu Visual Studio SDK	693
Korzystanie z menedżera rozszerzeń	699
Tworzenie rozszerzenia edytora za pomocą platformy MEF	701
Podsumowanie	709

Część V Tworzenie aplikacji dla przedsiębiorstw 711

Rozdział 17. Tworzenie aplikacji ASP.NET 713

Podstawy tworzenia witryn w ASP.NET	715
Tworzenie nowego projektu witryny lub aplikacji sieciowej	715
Kontrolowanie właściwości i opcji projektu	728
Tworzenie stron internetowych	736
Projektowanie interfejsu użytkownika	745
Określanie układu strony i położenia kontroltek	746
Tworzenie jednolitego wyglądu i zachowania	753
Tworzenie UI skonfigurowanego przez użytkownika	775
Praca z kontrolkami ASP.NET	787
Przegląd kontroltek ASP.NET	787
Standardowe kontrolki ASP.NET	790
Kontrolki do sprawdzania poprawności	792
Kontrolki logowania	794
Kontrolki nawigacyjne witryny	797
Kontrolki danych	798
Kontrolki użytkownika	800
Tworzenie aplikacji ASP.NET MVC	804
Wprowadzenie do aplikacji ASP.NET MVC	804
Tworzenie projektów MVC	808
Dodawanie mechanizmów za pomocą ASP.NET MVC	811
Podsumowanie	820

Rozdział 18. Tworzenie aplikacji opartych na formularzach Windows 821

Podstawy projektowania formularzy	822
Uwzględnianie użytkownika końcowego	823
Rola standardów UI	824
Planowanie interfejsu użytkownika	825
Tworzenie formularza	826
Typ projektu Windows Application	826
Właściwości i zdarzenia formularza	827
Dodawanie kontroltek i komponentów	830
Układ i pozycjonowanie kontroltek	831
Używanie kontenerów	836

Wygląd i zachowanie kontroltek	840
Praca z kontrolkami ToolStrip	841
Wyświetlanie danych	849
Tworzenie własnych kontroltek	853
Dziedziczenie po istniejącej kontrolce	853
Projektowanie kontrolki użytkownika	854
Tworzenie kontrolki niestandardowej	857
Podsumowanie	857
Rozdział 19. Tworzenie bogatszych i bardziej inteligentnych	
interfejsów użytkownika	859
Platforma Windows Presentation Foundation	860
Model programowania	863
Wprowadzenie do okna projektowego WPF	865
XAML i panele projektowe	866
Programowanie z wykorzystaniem WPF	871
Układ	871
Style i szablony	878
Wiązanie danych	881
Zdarzenia przekazywane	883
Tworzenie prostej przeglądarki obrazów	884
Rozpoczynanie tworzenia układu	885
Zapisywanie obrazów	891
Wiązanie rysunków	893
Metody do obsługi zdarzeń związanych	
z przyciskami i efekty do modyfikowania obrazu	894
Wybór katalogu przy użyciu standardowego okna dialogowego	895
Podsumowanie	901
Rozdział 20. Tworzenie bogatych aplikacji internetowych	903
Technologia ASP.NET Ajax i bogate interfejsy działające	
w różnych przeglądarkach	904
Ajaksowe kontrolki w ASP.NET	905
Tworzenie stron obsługujących częściową aktualizację	907
ASP.NET Ajax Library i Ajax Control Toolkit	915
Tworzenie wyjątkowych, bogatych interakcji opartych	
na przeglądarkach w systemie Windows	921
Niezależne aplikacje WPF a programy XBAP WPF	922
Tworzenie aplikacji WPF uruchamianych w przeglądarce	922
Zagadnienia związane z zabezpieczeniami	926
Instalowanie aplikacji XBAP	929
Udostępnianie interaktywnych aplikacji w różnych systemach	933
Wprowadzenie do Silverlight	933
Tworzenie aplikacji Silverlight	934
Silverlight 4.0	939
Podsumowanie	940

Rozdział 21. Praca z bazami danych	941
Tworzenie tabel i związków	942
Tworzenie nowej bazy danych SQL Server	943
Definiowanie tabel	945
Korzystanie z Database Diagram Designer	947
Praca z poleceniami w SQL-u	951
Pisanie zapytań	951
Tworzenie widoków	955
Tworzenie procedur składowanych	956
Tworzenie wyzwalaczy	960
Tworzenie funkcji definiowanych przez użytkownika	961
Używanie projektów baz danych	961
Tworzenie projektu bazy danych	962
Okno Schema View	967
Kompilowanie i wdrażanie	968
Uwagi na temat mechanizmu DAC	969
Tworzenie obiektów bazy danych w kodzie zarządzanym	970
Rozpoczynanie projektu SQL Server CLR	970
Tworzenie procedury składowanej w C#	971
Wiązanie kontroltek z danymi	974
Wprowadzenie do wiązania danych	974
Automatyczne generowanie związanych kontroltek Windows Forms	976
Modyfikowanie zbiorów danych o określonym typie	981
Ręczne wiązanie kontroltek formularzy Windows	982
Wiązanie danych w aplikacjach WPF	986
Wiązanie danych z kontrolkami sieciowymi	989
Odwzorowania obiektowo-relacyjne	993
Przegląd technologii LINQ	994
Odwzorowywanie przy użyciu narzędzia O/R Designer	996
Kod LINQ	998
Korzystanie z platformy Entity	1001
Kierowanie zapytań do modelu EDM	1006
Podsumowanie	1008
Rozdział 22. Aplikacje oparte na usługach	1011
Wprowadzenie do usług	1012
Dlaczego usługi sieciowe ASP.NET i WCF?	1014
Aplikacje oparte na usługach sieciowych ASP.NET	1015
Szablon projektu ASP.NET Web Service	1017
Tworzenie usługi sieciowej ASP.NET	1020
Konsumowanie usługi sieciowej ASP.NET	1033
Wyjątki w usługach sieciowych ASP.NET	1039
Aplikacje oparte na usługach WCF	1041
Szablon projektu WCF	1043
Tworzenie usług WCF	1045
Konfigurowanie usług WCF	1050

Konsumowanie usługi WCF	1055
Hosting i instalowanie usług WCF	1058
Podsumowanie	1060
Rozdział 23. Dodawanie procesów do aplikacji	1061
Podstawy technologii Windows Workflow	1063
Składniki procesu	1063
Szablony projektów typu Workflow	1065
Okno projektowe procesów	1067
Szablony elementów procesów	1068
Podstawy tworzenia procesów	1069
Korzystanie z wbudowanych czynności procesów	1082
Sterowanie przepływem	1083
Czynności Runtime i Primitives	1084
Czynności Error Handling	1086
Czynności Collection	1086
Obsługa transakcji	1087
Zarządzanie komunikatami	1089
Tworzenie procesów typu Flowchart	1094
Tworzenie aplikacji do zarządzania procesem	1096
Scenariusz — zgłoszenia dotyczące podróży	1096
Szablony projektów aplikacji	1097
Tworzenie bazy danych i biblioteki dostępu do danych	1099
Tworzenie usługi do obsługi rezerwacji w biurze podróży	1104
Tworzenie niestandardowej biblioteki czynności	1105
Projektowanie procesu zgłaszania podróży	1107
Tworzenie aplikacji klienckiej (formularze do zgłaszania i zatwierdzania podróży)	1119
Uruchamianie aplikacji do obsługi procesu zgłaszania podróży	1124
Podsumowanie	1127
Rozdział 24. Tworzenie aplikacji biznesowych opartych na pakiecie Office	1129
Przegląd rozszerzalnych funkcji pakietu Office	1131
Funkcje pakietu Office	1131
Typy projektów Office w Visual Studio	1135
Tworzenie dodatków dla pakietu Office	1136
Modyfikowanie wstążki	1137
Modyfikowanie panelu zadań	1141
Tworzenie regionów formularzy aplikacji Outlook	1144
Tworzenie rozszerzeń dokumentów Office	1147
Kontrolki kontenerowe	1147
Tworzenie paneli operacji	1149
Przechowywanie danych w pamięci podręcznej	1150
Implementowanie własnych tagów inteligentnych	1153
Podsumowanie	1156

Rozdział 25. Aplikacje działające w chmurze i technologia Windows Azure	1157
Podstawy platformy Azure	1158
Korzyści, jakie daje platforma Azure	1158
Konfigurowanie środowiska programistycznego	1160
Role w platformie Azure	1164
Szablony projektów na platformę Azure	1165
Sposoby przechowywania danych w platformie Azure	1167
Aplikacja Hello Cloud	1170
Rozwijanie i wdrażanie aplikacji na platformę Azure	1174
Scenariusz	1174
Rozwijanie aplikacji	1175
Subskrypcja usług platformy Azure (zakładanie konta)	1188
Tworzenie konta do przechowywania danych	1190
Tworzenie konta usług hosted service	1194
Przygotowywanie aplikacji do publikacji	1196
Publikowanie i wdrażanie aplikacji w chmurze	1199
Przenoszenie do środowiska produkcyjnego	1204
Następne kroki	1205
Podsumowanie	1207
Skorowidz	1209

Rozdział 9.

Testowanie kodu



W tym rozdziale:

- Podstawy testów jednostek
- Platforma testów jednostek
- Pisanie testów jednostek działających w ASP.NET
- Konfigurowanie atrybutów testów jednostek
- Tworzenie testów uporządkowanych
- Organizowanie testów

Programiści zawsze byli odpowiedzialni za testowanie kodu przed udostępnieniem go testerom lub użytkownikom. W przeszłości oznaczało to przejście przez każdy wiersz kodu w debugerze (włączając w to wszystkie warunki i błędy). W tym celu często trzeba było przygotować aplikacje testowe z operacjami potrzebnymi do wykonania kodu. Przejście przez cały kod pozwalało dojść do celu, ale nie zawsze programistom chciało się to robić (i bardzo trudno było to sprawdzić). W praktyce całą tę operację często pomijano w czasie wprowadzania zmian w kodzie i jego aktualizacji. Opisany proces utrudniał ponadto ustalenie, czy zmiany w kodzie wpływają na inne części systemu, a także powodował przesyłanie testerom i użytkownikom wersji o niższej jakości, co zwiększało liczbę usterek oraz wydłużało czas potrzebny na przesyłanie kodu tam i z powrotem między programistami a testerami.

Ten mechanizm (i efekty jego stosowania) wskazywał na potrzebę automatyzacji testowania jednostek. W efekcie powstały różne przeznaczone do tego platformy. Pierwszą taką platformą dla .NET była NUnit (*nunit.org*) — projekt o otwartym dostępie do kodu źródłowego działający dla większości języków .NET, który umożliwia pisanie kodu służącego do testowania innego kodu. Podobna platforma została wbudowana w Visual Studio 2005 i jest dostępna także w obecnej wersji.

Platforma testów jednostek w Visual Studio umożliwia kompilowanie testów wraz z aplikacją. Można też zdecydować się na zastosowanie programowania sterowanego testami i przygotować testy przed rozpoczęciem pisania kodu. W każdym przypadku ustrukturyzowane podejście do testów jednostek może doprowadzić do utworzenia pełnego zestawu testów współpracującego z aplikacją.

Pełny zestaw testów może składać się na test regresji dla większości komponentów, a nawet dla całego systemu. Daje to większą pewność przy wykonywaniu operacji, które wcześniej były bardzo ryzykowne, na przykład przy wprowadzaniu poprawek w ostatniej chwili, dokonywaniu refaktoryzacji czy dodawaniu elementów tuż przed zakończeniem pracy. Przy wykonywaniu tych działań można wykorzystać pełny zestaw testów jednostek do sprawdzenia, jakie elementy (i czy w ogóle) zostały zepsute w wyniku wprowadzenia zmian.

W tym rozdziale opisano wiele narzędzi, technologii i technik przeprowadzania testów przez programistów. Dzięki tym informacjom można zacząć czerpać liczne korzyści ze zautomatyzowanych testów dostępnych programistom. Prowadzą one do zmniejszenia liczby błędów, powstawania bardziej zrozumiałego kodu i większej pewności przy wprowadzaniu zmian w kodzie.

Podstawy testów jednostek

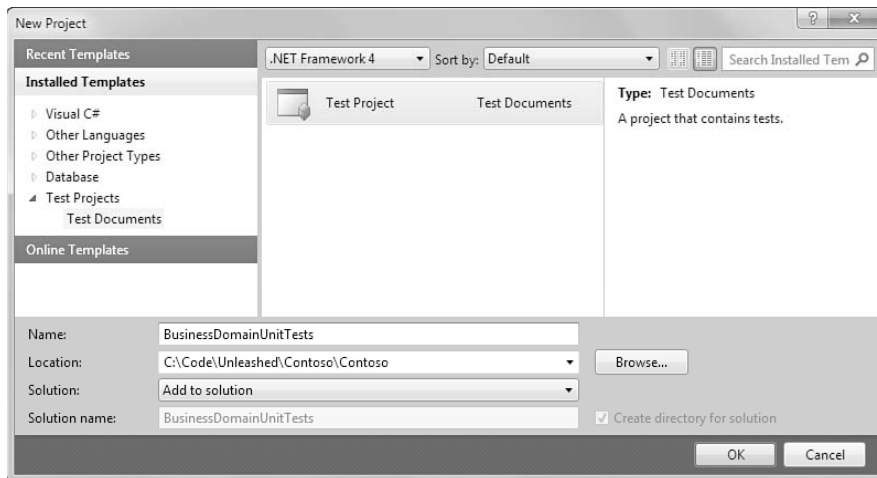
Testy jednostek w Visual Studio polegają na tworzeniu testów sprawdzających kod warstw aplikacji. Warstwy obejmują liczne klasy z obszaru logiki biznesowej i obsługi danych. Do sprawdzania interfejsu użytkownika służą zwykle inne metody automatycznego testowania. Są one przeznaczone głównie dla testerów. Potrzebne do tego funkcje i produkty znajdują się w wersji Premium środowiska Visual Studio, dlatego nie opisujemy ich w tym miejscu. W zamian koncentrujemy się na tworzeniu kodu do testowania innego kodu. Oczywiście, zakładamy przy tym, że programista w czasie rozwijania aplikacji stosuje prawidłowe techniki budowania architektury warstwowej. W tym podrozdziale omawiamy podstawy pisania testów jednostek. W dalszych punktach szczegółowo opisujemy przedstawione tu zagadnienia.

Tworzenie projektu testów

Testy programistów trzeba umieszczać w projektach testów. Projekt zawiera odpowiednie referencje do platformy testów jednostek i trzeba skonfigurować go tak, aby można go było uruchomić za pomocą narzędzi testowych wbudowanych w Visual Studio. Istnieją dwa główne sposoby tworzenia projektów testów jednostek. Można przygotować projekt testów w środowisku IDE lub automatycznie wygenerować testy jednostek dla istniejącego kodu w nowym projekcie testów. Zaczniemy od pierwszej z tych możliwości.

Szablon projektu testów

Nowy projekt testów w Visual Studio można utworzyć w oknie dialogowym *New Project* (*File/New/Project*). W oknie dialogowym należy przejść do węzła *Test Projects* w drzewie *Installed Templates*. Węzeł pokazano na rysunku 9.1. Warto zauważyć, że projekt testów można umieścić w nowym lub istniejącym rozwiązaniu. Zwykle testy projektów dodaje się do gotowych rozwiązań, ponieważ w testach trzeba umieścić referencje do projektów z rozwiązania.



Rysunek 9.1. W oknie dialogowym New Project można utworzyć nowy projekt testów dla rozwiązania

Dobre praktyki tworzenia projektów i klas testów

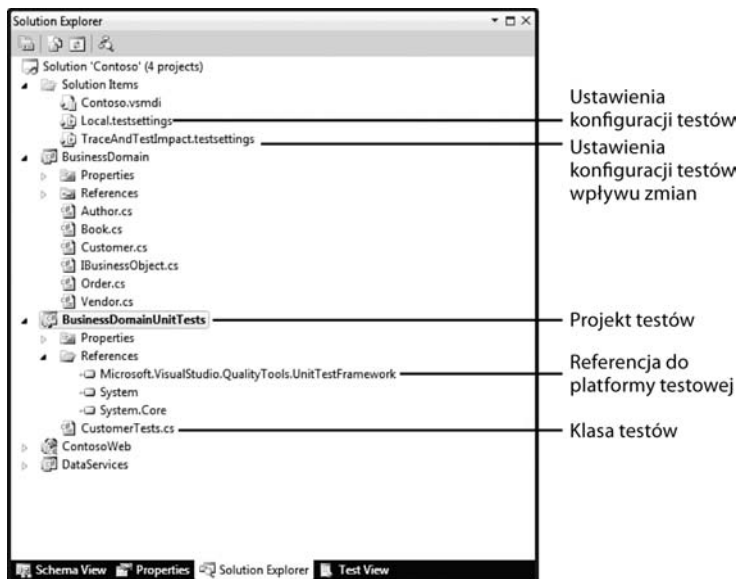
Należy ustalić reguły określające liczbę projektów testów tworzonych dla rozwiązania. Zwykle programiści tworzą jeden projekt testów dla każdego sprawdzanego projektu. Na przykład jeśli w jednym projekcie znajdują się obiekty biznesowe, a w innym usługi związane z danymi, można utworzyć dwa projekty testowe (po jednym dla każdego projektu głównego). Nie jest to jednak wymóg, a jedynie sposób na łatwe zrozumienie organizacji kodu.

Podobne zasady dotyczą klas testów. Należy utworzyć jedną klasę testów dla każdej klasy ze sprawdzanego docelowego projektu. Jeśli programista tworzy projekt testów dla kodu logiki biznesowej obejmującego obiekty `Customer` i `Order`, powinien przygotować klasy `CustomerTest` i `OrderTest`. Nie jest to konieczne, ale prowadzi do dobrego, logicznego uporządkowania kodu.

Projekt testów

W czasie tworzenia nowego projektu testów Visual Studio dodaje referencje do platformy testów jednostek (`Microsoft.VisualStudio.TestTools.UnitTestingFramework`). Ponadto dodaje do rozwiązania dwa pliki (w katalogu *Solution Items*). Są to pliki z konfiguracją (ustawieniami) testów. Ich nazwy to `Local.testsettings` i `TraceAndTestImpact.testsettings`. W projekcie testów tworzona jest też klasa testów używana do pisania testów jednostek. Na rysunku 9.2 pokazano każdy z tych elementów w oknie *Solution Explorer*. Poszczególne pliki omawiamy szczegółowo w dalszych punktach.

Rysunek 9.2.
Projekt testów
i powiązane elementy
w oknie Solution
Explorer

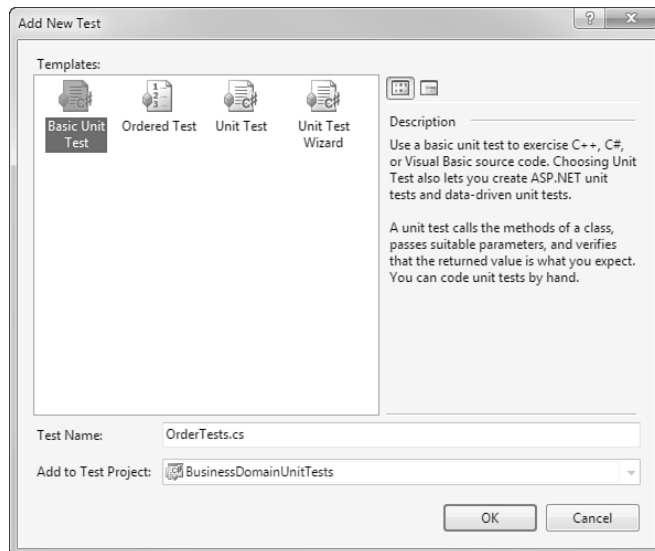


Do projektu można dodać nowe pliki testów. W tym celu należy kliknąć prawym przyciskiem myszy projekt testów i wybrać opcję *Add/New Test*. Ponadto można użyć menu *Project* i otworzyć okno dialogowe *Add New Test* widoczne na rysunku 9.3. W tym miejscu dostępne są następujące opcje:

- **Basic Unit Test.** Ten szablon tworzy prostą klasę testów z pustą metodą testową. Używa się go zwykle przy tworzeniu testów od podstaw (bez korzystania z funkcji automatycznego generowania testów).
- **Database Unit Test.** Ten szablon służy do tworzenia testów jednostek wykonywanych na procedurach składowanych z bazy danych (do stosowania tej funkcji potrzebna jest wersja Premium środowiska Visual Studio).
- **Ordered Test.** Ten szablon umożliwi utworzenie sekwencyjnej listy wspólnie wykonywanych testów (zobacz podrozdział „Tworzenie testów uporządkowanych” w dalszej części rozdziału).
- **Unit Test.** Ten szablon tworzy pustą klasę testów jednostek korzystającą z obiektu do śledzenia kontekstu testów. Tego typu klasy testów jednostek są generowane automatycznie (zobacz punkt „Generowanie testów na podstawie istniejącego kodu”).
- **Unit Test Wizard.** Ten szablon uruchamia kreator Unit Test Wizard służący do automatycznego generowania testów na podstawie istniejącego kodu rozwiązania (zobacz punkt „Generowanie testów na podstawie istniejącego kodu”).

Rysunek 9.3.

Za pomocą okna dialogowego *Add New Test* można dodawać nowe testy do projektu testów



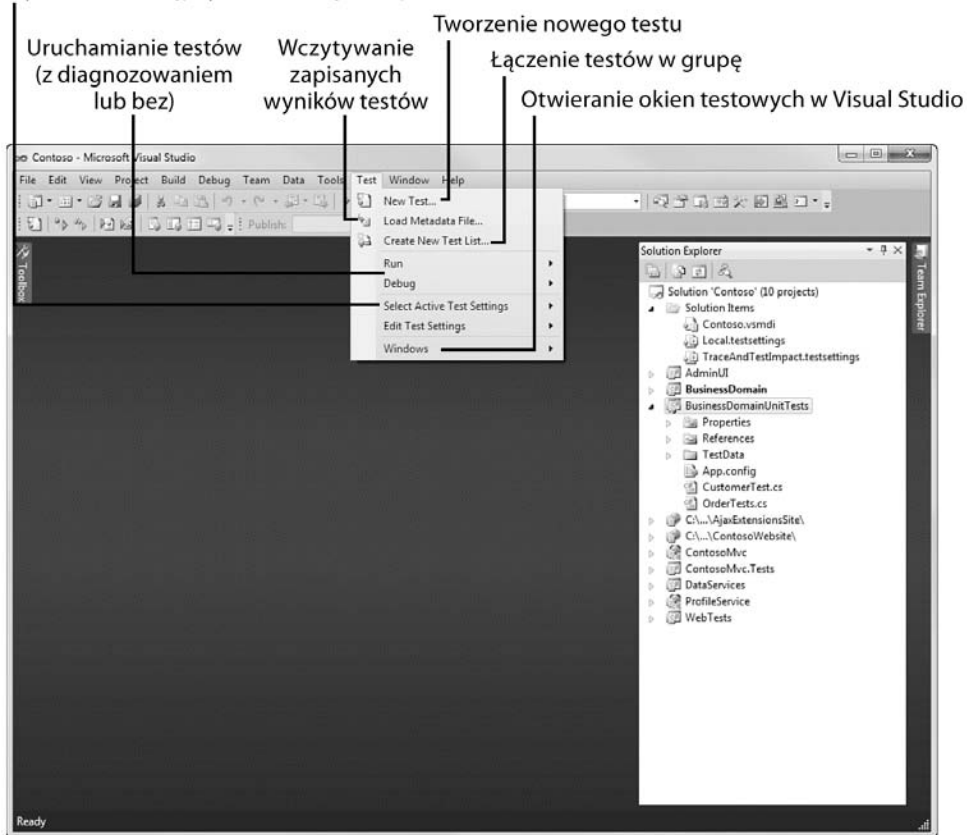
Menu Test

Menu *Test* można wykorzystać do uzyskania dostępu do wszystkich funkcji środowiska Visual Studio związanych z testami przeprowadzanymi przez programistów. Menu pozwala między innymi dodawać klasy testów jednostek do projektu testów (*Test/New Test*). Można go ponadto

używać do wczytywania informacji o wcześniejszych testach (*Load Metadata File*), uruchamiania testów (z diagnozowaniem lub bez), wybierania ustawień stosowanych w następnym przebiegu testów, a także edytowania takich ustawień. Można też uzyskać dostęp do wielu okien służących do wyświetlania i organizowania testów w aplikacji. Na rysunku 9.4 pokazano przykład ilustrujący wiele funkcji omawianego menu.

Wskazywanie aktywnych plików z ustawieniami testów.

Edytowanie dostępnych w rozwiązaniu plików z ustawieniami



Rysunek 9.4. Menu Test środowiska Visual Studio pozwala uzyskać dostęp do funkcji środowiska związanych z testami przeprowadzanymi przez programistów

Generowanie testów na podstawie istniejącego kodu

Visual Studio umożliwia też automatyzację procesu tworzenia testów jednostek. Można kliknąć prawym przyciskiem myszy istniejącą klasę i wybrać z menu podręcznego opcję *Create Unit Tests*. Inna możliwość to wybranie szablonu elementu *Unit Test Wizard*. Oba rozwiązania powodują utworzenie przez Visual Studio zestawu testów jednostek na podstawie kodu klasy.

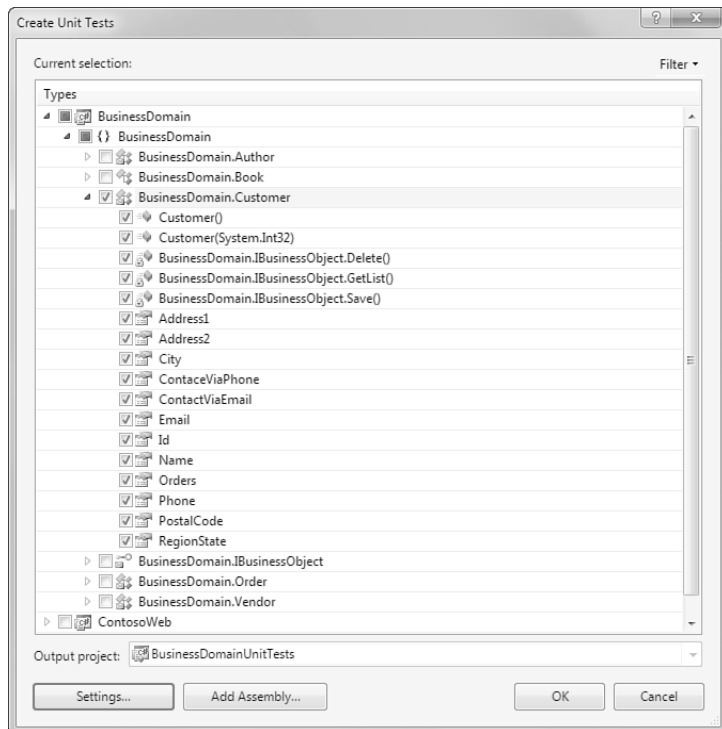
Wygenerowany kod to dobry punkt wyjścia do rozwijania testów jednostek. Visual Studio analizuje metody i właściwości klasy, na podstawie której programista chce wygenerować testy, oraz tworzy prawdziwe, możliwe do uruchomienia testy. Oczywiście, trzeba dokończyć każdy test przez dodanie odpowiednich wartości zmiennych i asercji, jednak kod wygenerowany przez narzędzia to dobry punkt wyjścia.

Przyjrzyjmy się przykładowi. Załóżmy, że programista używa obiektu `Customer` zawierającego standardowe właściwości, takie jak `Name`, `Address`, `Phone` i `Email`. Może także zawierać inne metody, takie jak `Save`, `Update` czy `Delete`. Środowiska Visual Studio można użyć do wygenerowania nowych testów jednostek dla klasy. W tym celu należy albo wybrać opcję *Unit Test Wizard* w oknie dialogowym *Add New Test*, albo kliknąć prawym przyciskiem myszy plik klasy i wybrać opcję *Create Unit Tests*.

W obu scenariuszach Visual Studio wyświetli okno dialogowe *Create Unit Tests* przedstawione na rysunku 9.5. Widoczne są w nim typy dostępne w rozwiązaniu. W powyższym oknie dialogowym można wybrać składowe, dla których trzeba wygenerować testy. Można też użyć ustawienia *Output project* do określenia projektu testów, w którym mają znaleźć się wygenerowane testy. Polega to na utworzeniu nowego projektu testów (w języku C# lub Visual Basic) albo na umieszczeniu kodu w istniejącym projekcie testów.

Rysunek 9.5.

Można zezwolić środowisku Visual Studio na wygenerowanie testów jednostek na podstawie istniejącego kodu



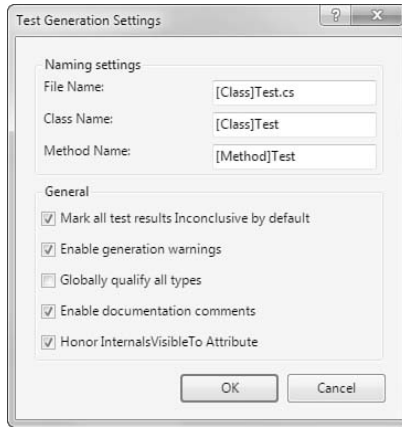
Przycisk *Add Assembly* w dolnej części okna dialogowego umożliwia wybranie podzespołu (pliku *.dll*) spoza projektu. Testy można wygenerować dla dowolnego podzespołu .NET.

Zarządzanie ustawieniami procesu generowania testów

Przycisk *Settings* pozwala otworzyć okno dialogowe, które służy do określania różnych ustawień związanych z generowaniem testów jednostek. Rysunek 9.6 przedstawia przykładowe informacje w tym oknie. Warto zauważyć, że można użyć tekstu makr ([File], [Class] i [Method]) do poinformowania Visual Studio o tym, jak ma tworzyć nazwy generowanych plików, klas i metod testów.

Rysunek 9.6.

Można kontrolować sposób generowania testów jednostek przez Visual Studio



Warto zwrócić uwagę na dodatkowe ustawienia w grupie *General*. Umożliwiają one ustawienie każdego generowanego testu jako niejednoznacznego, umieszczenie komentarzy w wygenerowanym kodzie i tak dalej.

Uwaga



Kreator Create Unit Tests umożliwia wygenerowanie testów jednostek dla wszystkich projektów w rozwiązaniu. Zaznaczenie klasy i wygenerowanie testów powoduje dodanie w projekcie testów referencji do projektu z testowanym kodem.

Wygenerowany kod testów

Visual Studio generuje testy dla wszystkich metod i właściwości obiektu zaznaczonych w oknie dialogowym *Create Unit Tests*. Zastanówmy się na przykład nad właściwością *Name*. Listing 9.1 przedstawia kod wygenerowany jako test jednostek tej właściwości (kod w języku Visual Basic wygląda podobnie). Test tworzy nowy obiekt typu *Customer*, a następnie próbuje ustawić wartość właściwości *Name* docelowej zmiennej. Potem określany jest oczekiwany efekt. Tu właściwość *Name* jest typu *string*, dlatego kod testu tworzy zmienną o nazwie *expected* tego typu. Definiuje też zmienną typu *string* o nazwie *actual*. Następnie przypisuje do właściwości *Customer.Name* wartość zmiennej *expected* i — po ustawieniu właściwości — wczytuje wartość zmiennej *actual*. Na zakończenie za pomocą asercji *Assert.AreEqual* sprawdza,

czy właściwość została poprawnie ustawiona. Jest to prawidłowy test właściwości. Programista musi jedynie podać poprawną wartość dla zmiennej `expected` (punkt `TODO`) i usunąć wywołanie `Assert.Inconclusive`.

Listing 9.1. Przykładowy automatycznie wygenerowany test jednostek

```
[TestMethod()]
public void NameTest() {
    Customer target = new Customer(); // TODO: Initialize to an appropriate value
    string expected = string.Empty; // TODO: Initialize to an appropriate value
    string actual;
    target.Name = expected;
    actual = target.Name;
    Assert.AreEqual(expected, actual);
    Assert.Inconclusive("Verify the correctness of this test method.");
}
```

Testowanie metod jest trochę bardziej pracochłonne. Jeśli metoda zwraca wartość, trzeba ją sprawdzić, aby się upewnić, że jest zgodna z oczekiwaniami. Jeżeli metoda nie zwraca wartości, trzeba napisać dodatkowy kod do sprawdzenia, czy działa prawidłowo. Na listingu 9.2 pokazano automatycznie wygenerowany test metody, która nie zwraca wartości. Oczywiście, jeśli wywołanie metody spowoduje błąd, test zakończy się niepowodzeniem. Można jednak napisać dodatkowe testy i asercje.

Listing 9.2. Przykładowy automatycznie wygenerowany test jednostki

```
[TestMethod()]
public void SaveTest() {
    Customer target = new Customer(); // TODO: Initialize to an appropriate value
    target.Save();
    Assert.Inconclusive("A method that does not return a value cannot be
    ↪verified.");
}
```

Uwaga



W czasie generowania kodu do testowania konstruktorów Visual Studio dodaje tekst `ConstructorTest` do nazwy klasy. Jeśli istnieje kilka konstruktorów, Visual Studio doda do nazwy liczbę, na przykład `CustomerConstructorTest1`.

Pisanie testów jednostek

Możliwe, że Czytelnicy zaczynają się domyślać, jak wykorzystać automatycznie wygenerowane testy jednostek. Przyjrzyjmy się pisaniu testów jednostek, aby lepiej je zrozumieć. Warto pamiętać, że test jednostek to po prostu kod testu, który ma wywoływać kod aplikacji. W takim kodzie testu znajdują się założenia dotyczące tego, czy określone warunki są spełnione, czy

nie po wykonaniu kodu aplikacji. Test może powieść się lub nie w zależności od wyników tych założeń. Jeśli programista oczekuje, że warunek będzie spełniony, a tak się nie stanie, wtedy test kończy się niepowodzeniem.

Kiedy projekt i klasa testu są już gotowe, trzeba wykonać trzy operacje, aby utworzyć typowy test jednostki:

1. Dodać atrybut `TestMethod` do kodu testu.
2. Uruchomić badany kod, przekazując znane wartości w taki sposób, jak parametry.
3. Zastosować asercje do oceny wyników testów.

Przyjrzyjmy się bliżej praktycznemu przykładowi. Załóżmy, że programista korzysta z usługi sieciowej zwracającej profil użytkownika z bazy danych. Ta usługa sieciowa przyjmuje jako parametr identyfikator klienta. Sygnatura funkcji może wyglądać tak:

```
public Customer GetCustomerProfile(int customerId)
```

Można napisać prosty test wywołujący tę usługę i przekazujący znany identyfikator z bazy danych. Dobrym sposobem na testy jednostek baz danych jest sprawdzanie znanego stanu bazy. W teście można następnie sprawdzić nie tylko to, czy zwracane dane działają, ale także czy są poprawne. Na listingu 9.3 znajduje się przykładowy test.

Listing 9.3. *Przykładowy test jednostek*

```
[TestMethod()]
public void GetCustomerProfileTest() {
    CustomerProfile cutProfileService = new CustomerProfile();
    int customerId = 1234;
    Customer customer = cutProfileService.GetCustomerProfile(customerId);
    Assert.AreEqual(customer.Id, 1234);
}
```

Warto zauważyć, że powyższy kod jest podobny do dowolnego kodu w języku C# (i Visual Basic). Aby oznaczyć metodę jako test, należy dodać atrybut `TestMethod`. W kodzie trzeba utworzyć egzemplarz testowanego obiektu i wywołać sprawdzaną metodę. Jeśli wywołanie się nie powiedzie (lub zostanie zgłoszony wyjątek), test zakończy się niepowodzeniem. Następnie w teście znajduje się asercja, która pozwala sprawdzić, czy zwrócony obiekt odpowiada oczekiwanemu. Jeśli ta asercja okaże się fałszywa (przekazane do niej wartości nie są sobie równe), test zakończy się niepowodzeniem. Jeśli asercja będzie prawdziwa, test się powiedzie. Można dodać kilka innych asercji, aby uzupełnić powyższy test (przez sprawdzenie innych znanych danych na temat klienta). Ponadto można utworzyć pewne dodatkowe testy sprawdzanej metody, na przykład określające, co się stanie przy próbie pobrania danych nieistniejącego klienta.

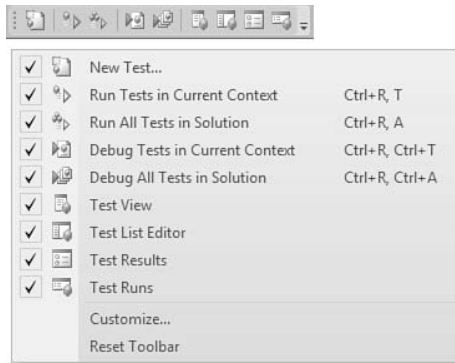
Powyższy kod powinien pozwolić zrozumieć podstawy działania testów jednostek. W dalszej części rozdziału zajmiemy się nimi nieco bardziej szczegółowo.

Uruchamianie testów jednostek

Testy jednostek można uruchamiać w kilku miejscach środowiska Visual Studio. Dwa z nich to pasek narzędzi *Test Tools* i menu *Test*. Obie możliwości związane są z podobnym zestawem opcji pokazanych na rysunku 9.7. W górnej części rysunku znajduje się pasek narzędzi *Test Tools*. W dolnej części pokazano zestaw przycisków opcji, które można dodać do paska lub usunąć z niego. Lista ta pomaga zrozumieć znaczenie ikon paska narzędzi i poznać klawisze skrótu służące do uruchamiania testów.

Rysunek 9.7.

Menu Test umożliwia uruchamianie testów z debuggerem lub bez niego



Testy można uruchamiać z debuggerem lub bez. Pierwsza z tych opcji umożliwia włączenie debugera, jeśli test zakończy się niepowodzeniem. Ta właściwość jest przydatna do używania testów do rozwiązywania problemów z kodem. Częściej stosuje się jednak drugie rozwiązanie, które pozwala po prostu uruchomić zestaw testów jednostek i sprawdzić wyniki. Testy można uruchamiać wtedy w bieżącym kontekście lub w kontekście całego rozwiązania. Bieżący kontekst pozwala uruchomić dowolne wybrane teksty, przeprowadzić testy od lokalizacji kursora lub włączyć wszystkie testy danej klasy (jeżeli wyświetlany jest kod klasy, a nie testu). Czasem trudno określić, co jest bieżącym kontekstem dla środowiska Visual Studio. Wtedy można albo uruchomić wszystkie testy z rozwiązania, albo przeprowadzić testy z poziomu okna *Test View* (zobacz podrozdział „Organizowanie testów”).

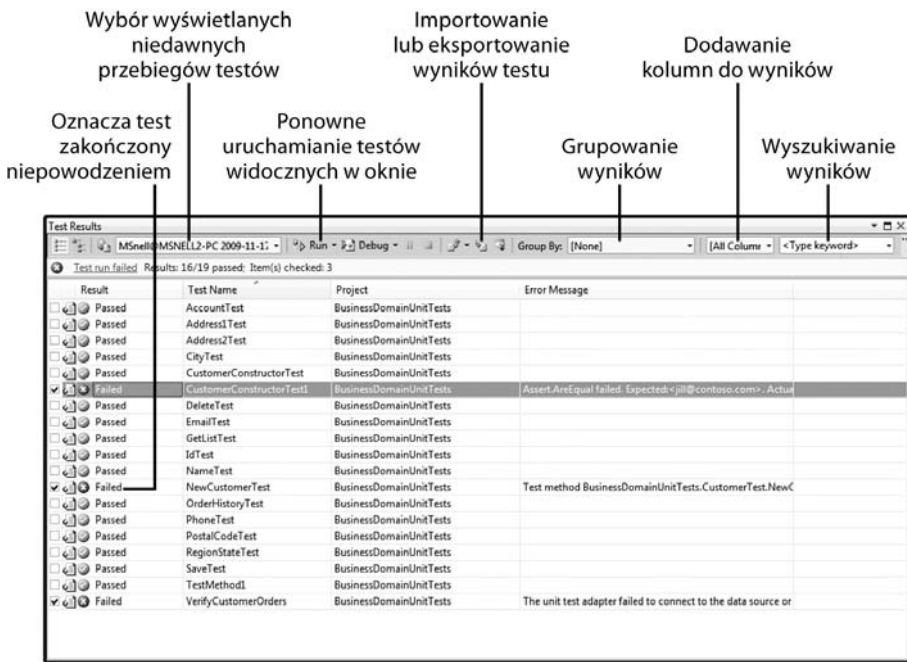
Uwaga



Przy uruchamianiu projektu testów wszystkie projekty, do których dodano referencje, są rekompilowane razem z projektem testów.

Przeglądanie wyników testów

Po uruchomieniu testów wyniki można zobaczyć w oknie *Test Results* (zwykle zadokowane jest w dolnej części środowiska IDE). Okno umożliwia sprawdzenie, które testy zakończyły się powodzeniem, a które nie. Rysunek 9.8 przedstawia liczne funkcje dostępne w tym oknie. Warto

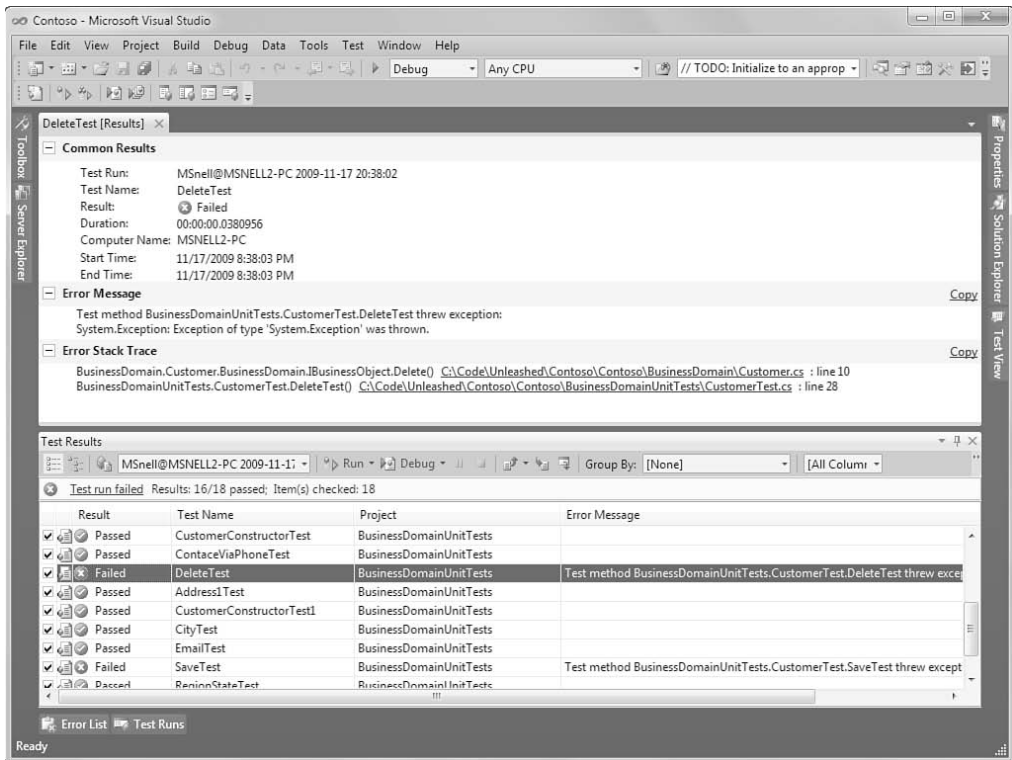


Rysunek 9.8. Okno Test Results umożliwia wyświetlenie wyników przebiegów testów i ponowne uruchomienie wybranych testów

zwrócić uwagę na to, że dany przebieg testu kończy się niepowodzeniem, jeśli nie powiedzie się przynajmniej jeden test. W tym przypadku powiodło się 16 z 18 testów. Jednak niepowodzenie dwóch testów spowodowało, że cały test zakończył się porażką.

Okno *Test Results* udostępnia pasek narzędzi z wieloma opcjami i kontrolkami. Jedną z opcji umożliwia wyświetlenie wyników wszystkich testów przeprowadzonych w czasie sesji środowiska Visual Studio. Pozwala to na wyświetlenie jednocześnie wyników wszystkich testów. Inna opcja umożliwia łatwe ponowne uruchomienie testów z danego przebiegu. Jeszcze inna służy do eksportowania wyników do pliku i udostępniania danych innym osobom (które mogą zaimportować je za pomocą tego samego paska narzędzi). Można też dodać nowe kolumny do wyników testu, aby wyświetlić więcej danych i lepiej je uporządkować (dostępnych jest ponad 20 kolumn).

Dwukrotne kliknięcie testu zakończonego powodzeniem powoduje wyświetlenie szczegółowych informacji na jego temat. Strona ze szczegółami obejmuje nazwę testu, wyniki, czas trwania testu, a także godzinę jego rozpoczęcia i zakończenia. Po dwukrotnym kliknięciu testu zakończonego niepowodzeniem programista zobaczy wiersz kodu sprawdzanej klasy, w którym wystąpił błąd. Aby zobaczyć szczegółowe informacje o takim teście, należy kliknąć go na liście wyników prawym przyciskiem myszy i wybrać opcję *View Test Results Details*. Na rysunku 9.9 przedstawiono stronę ze szczegółowymi danymi na temat nieudanego testu. Warto zauważyć, że obejmują one komunikat o błędzie, ślad stosu i odnośnik do kodu, który spowodował błąd.



Rysunek 9.9. Można wyświetlić szczegółowe informacje o wynikach dowolnego testu (udanego lub nieudanego)

Wyświetlanie przebiegów testów

Visual Studio udostępnia też okno dialogowe *Test Runs*, służące do przeglądania przebiegów testów z bieżącej sesji (i zaimportowanych). Okno można otworzyć za pomocą paska narzędzi *Test Tools* lub z menu *Test/Windows/Test Runs*. Na rysunku 9.10 pokazano przykładową zawartość omawianego okna dialogowego. Można dwukrotnie kliknąć dany przebieg, aby wczytać szczegółowe informacje o wynikach do okna *Test Results*.

Konfigurowanie opcji i ustawień testów

Visual Studio umożliwia kontrolowanie i dostosowywanie środowiska testowego. Służy do tego okno dialogowe *Options*. Ponadto w plikach konfiguracyjnych można kontrolować ustawienia testów specyficznych dla rozwiązania. W tym punkcie opisujemy, jak stosować obie techniki konfigurowania do lepszego zarządzania środowiskiem testowym.

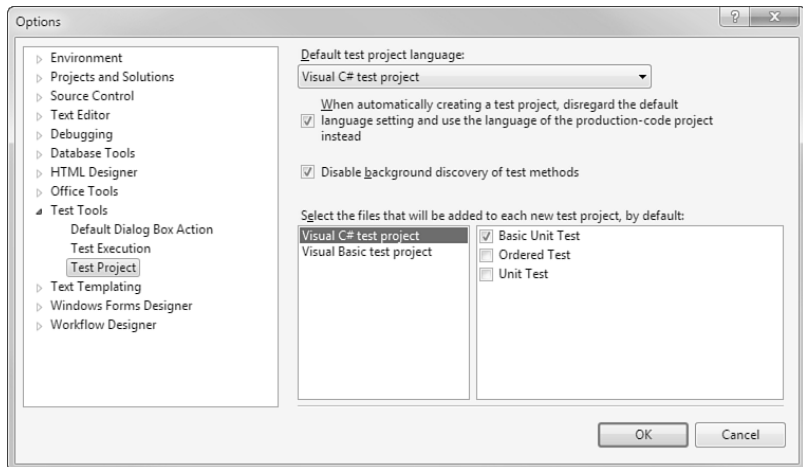
Test Run Name	Status	Owner
Completed Runs (16) (C:\Code\Unleashed\Contoso\Contoso\TestResults)		
MSnell@MSNELL2-PC 2009-11-17 20:38:02	16/18 passed, 2 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:36:33	16/18 passed, 2 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:31:31	15/18 passed, 3 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:30:59	16/18 passed, 2 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:29:41	1/18 passed, 17 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:29:19	0/18 passed, 1 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:28:32	1/18 passed, 17 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:27:56	1/18 passed, 17 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:26:16	0/18 passed, 10 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:25:40	17/18 passed, 1 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:24:41	16/18 passed, 2 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:24:20	16/18 passed, 2 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:23:38	14/18 passed, 4 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:17:41	15/18 passed, 3 failed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 20:17:21	1/1 passed	CEI.DOMAIN\MSnell
MSnell@MSNELL2-PC 2009-11-17 19:51:58	1/1 passed	CEI.DOMAIN\MSnell

Rysunek 9.10. Okno dialogowe Test Runs umożliwia wyświetlenie wyników wszystkich przebiegów testów z bieżącej sesji środowiska Visual Studio

Opcje projektów testów

Można kontrolować sposób generowania projektów testów przez Visual Studio. Należy to zrobić w węzle *Test Tools/Test Project* w oknie dialogowym *Options (Tools/Options)*. Na rysunku 9.11 pokazano przykładową zawartość tego okna. Można tu określić język domyślny (Visual Basic lub C#) dla projektu testów stosowany przez Visual Studio przy tworzeniu lub generowaniu nowego projektu tego typu. Warto zauważyć, że można zmienić domyślny język ustawiony w środowisku IDE dla projektów testów.

Rysunek 9.11. Okno dialogowe Options umożliwia dostosowanie domyślnych ustawień szablonów projektów testów w Visual Studio

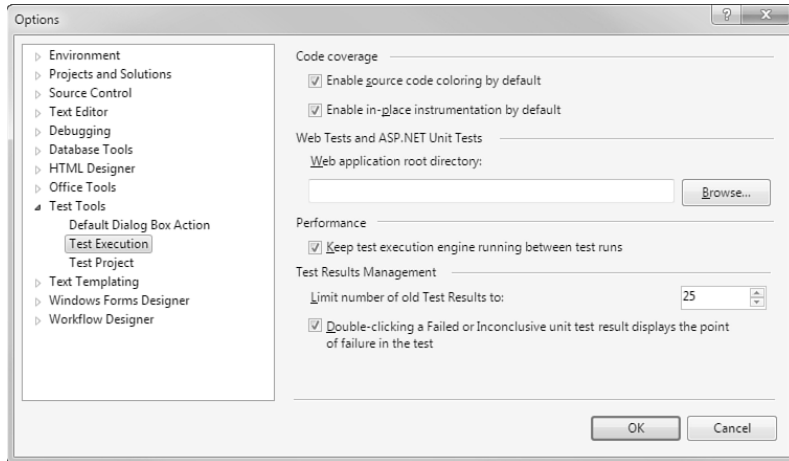


Opcje w węźle *Test Project* pozwalają też określić, które pliki należy domyślnie umieścić w projekcie testów w czasie jego tworzenia. To, jakie opcje są dostępne, zależy od języka. Można określić pliki podstawowych testów jednostek, testów jednostek baz danych, testów uporządkowanych lub testów jednostek (uruchamiany jest wtedy kreator Unit Test Wizard).

Zarządzanie wynikami testów

Visual Studio domyślnie przechowuje wyniki ostatnich 25 przebiegów testów. Po przekroczeniu tego limitu środowisko przed zapisaniem wyników nowego przebiegu usunie informacje o najstarszym. W oknie dialogowym *Options (Tools/Options)* można zwiększyć (lub zmniejszyć) tę wartość. Należy wybrać węzeł podrzędny *Test Execution* w węźle *Test Tools*, aby wyświetlić opcje widoczne na rysunku 9.12. Warto zwrócić uwagę na opcję *Limit Number of Old Test Runs To* (czyli ogranicz liczbę dawnych przebiegów testów do).

Rysunek 9.12.
W oknie dialogowym *Options* można określić liczbę przechowywanych przebiegów testów



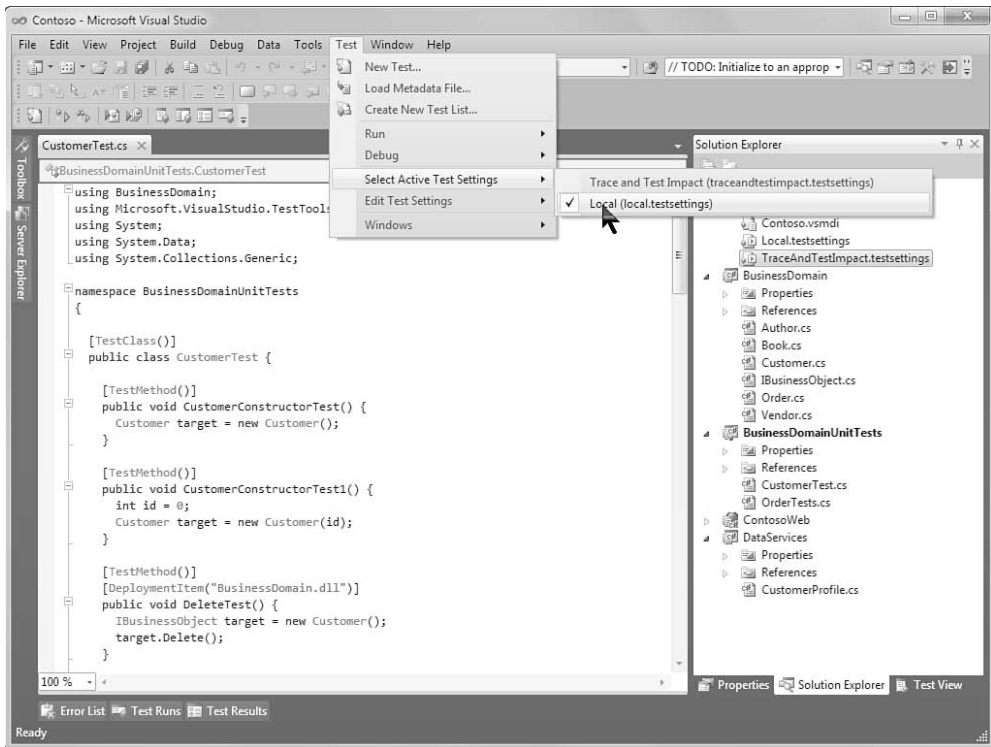
Kontrolowanie ustawień testów

Visual Studio udostępnia plik ustawień umożliwiający kontrolowanie wykonywania testów. W większości sytuacji można pozostawić domyślną zawartość pliku ustawień. Jednak nieraz warto zmodyfikować niektóre opcje, dotyczące na przykład uruchamiania skryptów instalacyjnych w ramach testów, limitu czasu i tak dalej.

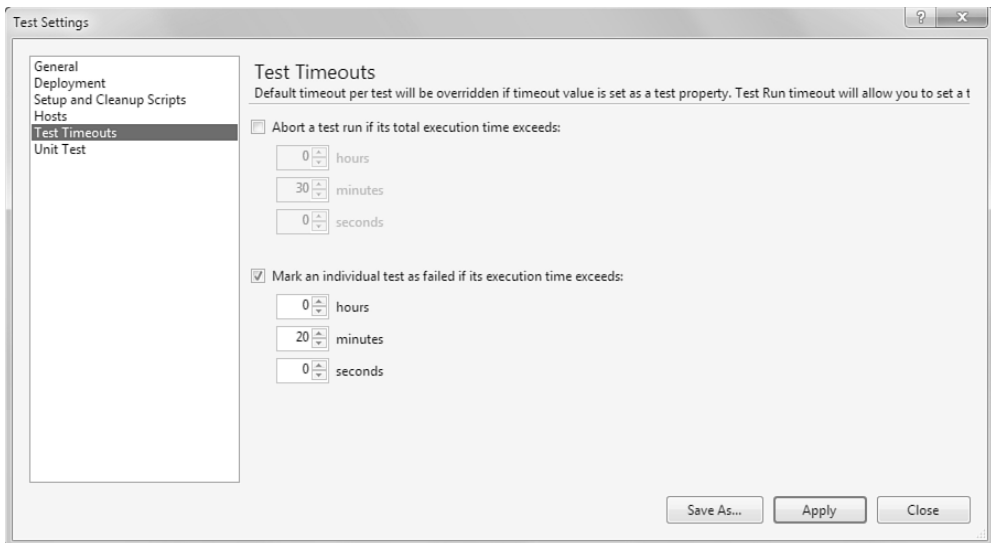
Pliki z ustawieniami testów można znaleźć na poziomie rozwiązania w oknie *Solution Explorer* (zobacz rysunek 9.2). Każdy plik ustawień testów ma rozszerzenie *.testsettings*. W dowolnym momencie za pomocą opcji *Test/Select Active Test Settings* można określić, który plik *.testsettings* ma być aktywny. Na rysunku 9.13 pokazano, jak wybrać aktywny plik ustawień.

Pliki *.testsettings* w Visual Studio to zwykłe pliki XML z konfiguracją. Kod w języku XML można bezpośrednio edytować. Visual Studio udostępnia jednak okno dialogowe do łatwiejszego modyfikowania ustawień. Okno dialogowe *Test Settings* można uruchomić przez dwukrotne kliknięcie pliku *.testsettings*. Na rysunku 9.14 pokazano to okno z wybranym węzłem *Test Timeouts*.

Część III Tworzenie kodu i zarządzanie nim



Rysunek 9.13. Za pomocą menu Test można wybrać aktywny plik z ustawieniami testów



Rysunek 9.14. Okno dialogowe Test Settings umożliwia zarządzanie ustawieniami związanymi z przeprowadzaniem testów (w tym limitem czasu)

Zaznaczenie węzła po lewej stronie okna dialogowego *Test Settings* pozwala skonfigurować różne ustawienia. Poniżej znajduje się przegląd opcji, którymi można zarządzać w każdym węzle okna *Test Settings*:

- **General.** Zawiera opcje do określania nazwy pliku ustawień, udostępniania opisu, generowania nazw dla poszczególnych przebiegów testów (na rysunku 9.10 pokazano nazwy takich przebiegów).
- **Deployment.** Umożliwia określenie dodatkowych plików i katalogów instalowanych w ramach przeprowadzania testów. Domyślnie testowane podzespoły są automatycznie instalowane w katalogu *bin* testu. Pozwala to przed przeprowadzeniem testów zainstalować nowe elementy (na przykład bazy danych oparte na plikach).
- **Setup and Cleanup Scripts.** Umożliwia określenie skryptu uruchamianego przed uruchomieniem testów lub po ich zakończeniu (albo w obu sytuacjach). Różni się to od uruchamiania kodu przed testami klas. Tu można zdefiniować kod wykonywany przed dowolną grupą testów (lub po niej).
- **Hosts.** Umożliwia określenie hosta, na którym można przeprowadzić testy. Zwykle jest to host domyślny (środowisko Visual Studio). Ponadto można ustawić testy tak, aby działały na serwerze środowiska ASP.NET.
- **Test Timeouts.** Umożliwia określenie czasu oczekiwania środowiska Visual Studio (lub innego hosta) przed porzuceniem długo działających testów. Ponadto można oznaczyć zbyt długo trwające testy jako nieudane. Na rysunku 9.12 przedstawiono przykładowe ustawienia w tym oknie dialogowym.
- **Unit Test.** Udostępnia zaawansowane opcje wczytywania dodatkowych podzespołów potrzebnych do przeprowadzenia testów.

Platforma testów jednostek

Platforma testów jednostek to część środowiska Visual Studio, a nie platformy .NET. System testowania jednostek w Visual Studio obejmuje: zestaw klas platformy, narzędzia i host, gdzie wykonywane są testy. Przestrzeń nazw zawierająca klasy platformy testów jednostek to `Microsoft.VisualStudio.TestTools.UnitTesting`. Dla programistów najważniejsze w tej przestrzeni nazw są klasy atrybutów i klasa `Assert`. W tym podrozdziale omawiamy podstawowe scenariusze korzystania z klas atrybutów i asercji (a także innych klas).

Klasa `TestContext`

Platforma testów jednostek obejmuje klasę służącą do przechowywania informacji związanych z wykonywaniem testów. Jest to klasa `TestContext`. Właściwości tej klasy służą do pobierania danych o działających testach. Informacje obejmują: ścieżkę do katalogu z testem, adres URL uruchamianego testu (jeśli są to testy jednostek ASP.NET) i informacje o wiązaniu danych (na

przykład połączenie z danymi lub obecny wiersz danych w wykonywanym teście). Wiadomości na temat kontekstu testu są przechowywane we właściwościach omawianej klasy. Kluczowe właściwości i ich zastosowania opisano w tabeli 9.1.

Tabela 9.1. Kluczowe właściwości klasy *TestContext*

Właściwość	Opis
CurrentTestOutcome	Umożliwia określenie wyniku ostatniego przeprowadzonego testu. Jest to przydatne w metodach oznaczonych jako <code>TestCleanup</code> (zobacz punkt „Klasy atrybutów testów”).
DataConnection	Umożliwia wyświetlenie łańcucha połączenia używanego w testach jednostek sterowanych danymi (zobacz punkt „Tworzenie testów jednostek zależnych od danych”).
DataRow	Umożliwia dostęp do kolumn w wierszu z danymi w czasie pracy z testami jednostek zależnych od danych (zobacz punkt „Tworzenie testów jednostek zależnych od danych”).
RequestedPage	Zapewnia dostęp do obiektu <code>Page</code> reprezentującego żadaną stronę w testach jednostek ASP.NET (zobacz podrozdział „Pisanie testów jednostek działających w ASP.NET”).
TestDeploymentDir	Umożliwia wyświetlenie ścieżki do katalogu, w którym testy są instalowane i uruchamiane. Jest to przydatne, jeśli trzeba zapisywać lub wczytywać dane z pliku z tego katalogu.
TestLogsDir	Umożliwia wyświetlenie ścieżki do katalogu, w którym zapisywane są wyniki testów.
TestName	Umożliwia określenie nazwy obecnie przeprowadzanego testu.

Klasa `TestContext` domyślnie jest niedostępna w kodzie. Aby uzyskać do niej dostęp, najpierw trzeba zdefiniować właściwość o tej samej nazwie i typie w klasie testu (potrzebna właściwość jest automatycznie tworzona w czasie generowania testów jednostek na podstawie istniejącego kodu). Platforma testów jednostek w momencie uruchamiania testów automatycznie tworzy egzemplarz klasy `TestContext`, a następnie sprawdza, czy w kodzie źródłowym znajduje się właściwość o tej nazwie. Jeśli ją znajdzie, przypisuje egzemplarz klasy `TestContext` do właściwości. Od tego momentu można korzystać z właściwości do pobierania informacji o kontekście przeprowadzanego testu. Poniższy kod pokazuje, jak zdefiniować właściwość `TestContext` w teście jednostek w języku C#:

```
public TestContext TestContext { get; set; }
```

Środowisko Visual Studio w czasie tworzenia obiektu typu `TestContext` generuje kompletną właściwość. Poniżej pokazano kod właściwości wygenerowany przez Visual Studio:

```
private TestContext testContextInstance;

public TestContext TestContext {
    get {
        return testContextInstance;
    }
    set {
        testContextInstance = value;
    }
}
```

Uwaga

Niektóre klasy atrybutów wymagają zdefiniowania parametru typu `TestContext` dla metod opatrzonych danym atrybutem. Dotyczy to na przykład atrybutu `ClassInitialize` (zobacz dalej). W takim scenariuszu platforma testów jednostek automatycznie przekazuje egzemplarz klasy `TestContext` do metody w momencie jej uruchamiania.

Klasy atrybutów testów

Testy jednostek są uruchamiane przez Visual Studio za pomocą hosta do przeprowadzania testów jednostek. Host musi zbadać kod, znaleźć w nim testy jednostek i uruchomić je w odpowiedni sposób. Wykorzystuje do tego atrybuty. Przypomnijmy, że atrybuty służą do udostępniania metadanych na temat kodu. Inny kod (na przykład host testów jednostek) może wykorzystać mechanizm refleksji do określenia różnych informacji na temat uruchamianego kodu.

W dotychczasowych krótkich przykładach pokazaliśmy, że testy jednostek oznaczane są za pomocą klas atrybutów zdefiniowanych w przestrzeni nazw testów. Przykładowo, klasa testu ma atrybut `TestClass`, a metodę testową należy oznaczyć atrybutem `TestMethod`. W tabeli 9.2 pokazano listę najczęściej stosowanych klas atrybutów z przestrzeni nazw testów jednostek.

Tabela 9.2. Klasy atrybutów testów dostępne w Visual Studio

Test	Opis
<code>AssemblyCleanup</code>	Służy do definiowania metod, które należy uruchomić po wykonaniu przez platformę wszystkich testów z danego podzespołu. Jest to przydatne, jeśli trzeba uporządkować zasoby po wykonaniu wszystkich testów. Warto zauważyć, że taki atrybut można przypisać tylko do jednej metody w podzespołe.
<code>AssemblyInitialize</code>	Służy do definiowania metod, które należy uruchomić przed uruchomieniem przez platformę testów z danego podzespołu. Jest to przydatne, jeśli trzeba zainicjować zasoby dla wszystkich testów z podzespołu. Warto zauważyć, że taki atrybut można przypisać tylko do jednej metody w podzespołe.
<code>ClassCleanup</code>	Służy do określania metody, którą należy uruchomić jeden raz po wykonaniu wszystkich testów z danej klasy testów. Jest to przydatne, jeśli trzeba zresetować stan systemu (na przykład w bazie danych) po zakończeniu testów.
<code>ClassInitialize</code>	Służy do określania metody, którą host powinien uruchomić jeden raz przed uruchomieniem testów z danej klasy testów. Jest to przydatne, jeśli trzeba zresetować bazę danych lub wykonać kod przygotowujący środowisko testowe. Metoda musi przyjmować parametr typu <code>TestContext</code> .
<code>DataSource</code>	Służy do udostępniania informacji o połączeniu w testach jednostek zależnych od danych („Tworzenie testów jednostek zależnych od danych”).

Tabela 9.2. Klasy atrybutów testów dostępne w Visual Studio — ciąg dalszy

Test	Opis
DeploymentItem	Służy do określania dodatkowych plików (.dll, .txt i innych), które trzeba zainstalować w katalogu, gdzie przeprowadzane są testy.
ExpectedException	Służy do określania, że zgodnie z oczekiwaniami kod metody testowej powinien zgłosić wyjątek podanego typu. Działanie metod testowych tego rodzaju jest uważane za prawidłowe, jeśli w czasie wykonywania testu pojawi się wyjątek odpowiedniego typu (w przeciwnym razie test kończy się niepowodzeniem). Mechanizm ten jest przydatny do testowania oczekiwanych warunków wystąpienia błędów w kodzie (zobacz punkt „Testowanie wyjątków”).
HostType	Używany do zastępowania domyślnego hosta testów (środowiska Visual Studio). Zwykle nie trzeba stosować tego atrybutu. Jeśli jednak programista pisze testy uruchamiane w procesie innego hosta (na przykład w środowisku ASP.NET), może wykorzystać ten atrybut (wraz z atrybutami UriToTest i AspNetDevelopmentServerHost).
Ignore	Dodawany do metody testowej w celu określenia, że daną metodę należy pominąć, jeśli stosowany jest host do przeprowadzania testów.
TestClass	Służy do określania, że dana klasa jest klasą testów zawierającą jeden lub kilka testów jednostek (metod testowych).
TestCleanup	Służy do wskazywania metody, którą należy uruchomić jednokrotnie po wykonaniu każdej metody testowej. Wskazaną metodę można wykorzystać do uruchomienia operacji porządkujących po wykonaniu każdej metody testowej w danej klasie testów. Porządkowanie na poziomie klasy można przeprowadzić za pomocą atrybutu ClassCleanup.
TestInitialize	Służy do określania, że daną metodę należy uruchomić jednokrotnie przed wykonaniem każdej metody testowej. Jest to przydatne, jeśli trzeba zresetować stan systemu przed wykonaniem każdej metody testowej w danej klasie testów. Jeśli programista chce tylko zainicjować stan dla wszystkich metod z danej klasy testów, powinien użyć atrybutu ClassInitialize.
TestProperty	Służy do definiowania właściwości (pary nazwa – wartość) metody testowej. Informacje zapisane we właściwości można wczytać w kodzie metody testowej.
TestMethod	Służy do oznaczania metod w klasie testów jako testów jednostek. Metody testowe nie mogą zwracać wartości (zwracają typ void). Powodzenie lub niepowodzenie zależy od warunków błędów oraz asercji. Metody testowe nie mogą przyjmować parametrów, ponieważ host nie potrafi przekazywać parametrów do metod. Istnieją jednak sposoby na zasymulowanie przekazywania parametrów. Więcej informacji na ten temat zawiera punkt „Tworzenie testów jednostek zależnych od danych”.
Timeout	Służy do określania limitu czasu (w milisekundach) dla danej metody testowej. Jeśli test będzie trwał dłużej, środowisko zatrzyma go i uzna za zakończony niepowodzeniem.

W tabeli 9.2 pokazano, że istnieje wiele klas atrybutów. Zapewniają one kontrolę nad testami jednostek. Jeśli programista pisze klasę testów dla obiektu `Customer`, może zdefiniować klasę `CustomerTest`. Na listingu 9.4 pokazano szkielet kilku testów z tej klasy. Zademontrowano też liczne atrybuty stosowane w typowych klasach testów. Warto zwrócić uwagę na metody do inicjowania klasy testów i porządkowania zasobów po przeprowadzeniu testu.

Listing 9.4. *Przykładowa klasa testów*

```
using System;
using BusinessDomain;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System.Data;
using System.Collections.Generic;

namespace BusinessDomainUnitTests {

    [TestClass()]
    public class CustomerTest {

        public TestContext TestContext { get; set; }

        [ClassInitialize()]
        public static void InitTests(TestContext testContext) {
            // Wywołanie kodu resetującego testową bazę danych (Utilities.ResetTestDb();).
        }

        [ClassCleanup()]
        public static void CleanupPostTests() {
            // Wywołanie kodu resetującego testową bazę danych (Utilities.ResetTestDb();).
        }

        [TestMethod()]
        public void CustomerConstructorTest() {
            // Testuje konstruktor domyślny.
        }

        [TestMethod()]
        public void CustomerConstructorTest1() {
            // Testuje konstruktor przyjmujący identyfikator klienta.
        }

        [TestMethod()]
        [DeploymentItem("BusinessDomain.dll")]
        public void DeleteTest() {

        }

        [TestMethod()]
        [DeploymentItem("BusinessDomain.dll")]
```

```
public void GetListTest() {  
  
}  
  
[TestMethod()]  
[DeploymentItem("BusinessDomain.dll")]  
[Timeout(5000)]  
  
public void SaveTest() {  
  
}  
  
}  
  
}
```

Następny krok polega na napisaniu kodu w każdej metodzie testowej. Kod powinien kierować wywołania do obiektu `Customer` i sprawdzać asercje związane z wynikami wywołań. Do tworzenia asercji służy opisana dalej klasa `Assert`.

Operacje wykonywane przed testami jednostek i po nich

Dobłą praktyką przy tworzeniu testów jednostek jest pisanie ich dla znanego stanu systemu. Dotyczy to bazy danych, plików i innych elementów składających się na system. W ten sposób programista może mieć pewność, że potrzebne elementy będą dostępne przy przeprowadzaniu testów. Oczywiście, same testy często naruszają stan. Testy usuwają dane, modyfikują je, dodają nowe rekordy i wykonują podobne operacje. Wtedy potrzebna jest możliwość przywrócenia stanu systemu przed wykonaniem testów (i [lub] po ich zakończeniu), aby zagwarantować stały stan na czas testów i umożliwić programistom uruchamianie ich za pomocą jednego kliknięcia (jest to następna dobra praktyka z obszaru testów jednostek).

Zwykle trzeba napisać kod utrzymujący stały stan systemu. Kod może kopiować znaną i poprawną testową bazę danych do katalogu testów (można wykorzystać do tego także atrybut `DeploymentItem`), resetować bazę danych za pomocą instrukcji w SQL-u, realizować plan generowania danych w celu utworzenia bazy, kopiować pliki lub sprawdzać inne instalowane elementy.

Kod resetujący system jest specyficzny dla środowiska. Jednak aby zagwarantować wywołanie potrzebnego kodu w czasie przeprowadzania testów, można zastosować kilka klas atrybutów: `ClassInitialize` i `ClassCleanup` lub `TestInitialize` i `TestCleanup`. Pierwsza para powoduje wykonanie kodu na początku (i na końcu) przebiegu testów dla całej klasy testów jednostek. Druga para pozwala uruchomić kod przed wykonaniem każdego testu w danej klasie testów i po ich przeprowadzeniu.

Zwykle operacje inicjujące i porządkujące wywoływane są na poziomie klasy. Jeśli programista używa klasy `Utilities` z metodą resetującą bazę danych, może zagwarantować wywołanie metody, oznaczając ją atrybutem `ClassInitialize`. Warto zauważyć, że metoda przyjmuje

obiekt `TestContext` przekazywany przez platformę testów jednostek. Dobrą praktyką jest ponowne resetowanie systemu po wykonaniu testów jednostek. W poniższym kodzie pokazano dwie metody testów. Metody przygotowują testy i porządkują po nich.

```
[ClassInitialize()]
public static void InitTests(TestContext testContext) {
    Utilities.ResetTestDb();
}

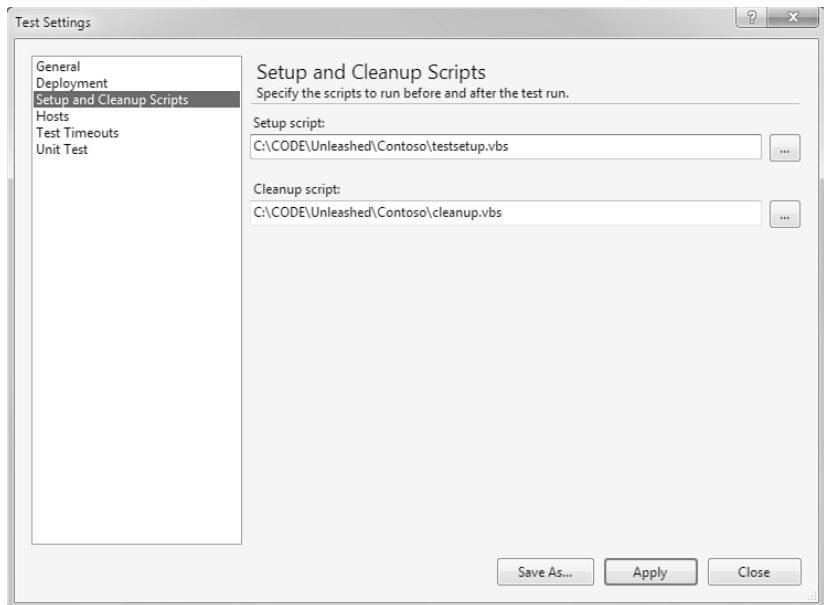
[ClassCleanup()]
public static void CleanupPostTests() {
    Utilities.ResetTestDb();
}
```

Kontrolowanie operacji inicjujących i porządkujących na poziomie wykonywania testów

Zdarza się, że operacje inicjujące i porządkujące trzeba uruchomić na poziomie wyższym niż klasa testów. Czasem przed uruchomieniem klas testów trzeba wywołać skrypty. Wtedy można wykorzystać plik `.testsettings` (zobacz wcześniejszy punkt „Kontrolowanie ustawień testów”) do zdefiniowania skryptu wywoływanego przed uruchomieniem testów i po ich zakończeniu.

Na rysunku 9.15 pokazano węzeł *Setup and Cleanup Scripts* w oknie dialogowym *Test Settings*. W tym miejscu można zdefiniować skrypty inicjujące i porządkujące wywoływane przy uruchamianiu dowolnych testów z rozwiązania.

Rysunek 9.15.
Definiowanie skryptów inicjujących i porządkujących wywoływanych przy wykonywaniu dowolnych testów z rozwiązania



Klasy asercji

Przestrzeń nazw `UnitTesting` zawiera także typ statyczny `Assert`. Ten typ udostępnia liczne metody umożliwiające sprawdzenie, czy wynik testów jest zgodny z oczekiwaniami. Należy wywołać statyczne metody i sprawdzić wartość warunku (`true` lub `false`). Jeśli warunek logiczny nie jest spełniony, działanie asercji kończy się niepowodzeniem. Same asercje nie zwracają wyników. Zamiast tego w czasie wykonywania programu automatycznie powiadamiają platformę testów jednostek o sukcesie lub niepowodzeniu.

Można na przykład napisać test jednostki wczytujący znany rekord z bazy danych. Następnie można dodać asercję dotyczącą tego rekordu, aby udowodnić, że program potrafi pobrać dane z bazy, prawidłowo wywołać odpowiednie zbiory danych i dotrzeć do konkretnego obiektu. Poniżej pokazano prostą asercję do sprawdzania, czy dwie zmienne mają tę samą wartość. Jeśli wartości są sobie równe (`AreEqual`), działanie asercji kończy się powodzeniem. Jeżeli wartości są różne, asercja zgłasza niepowodzenie, a platforma testów jednostek oznacza test jako nieudany.

```
Assert.AreEqual(cust.Id, customerId);
```

`AreEqual` to tylko jedna z wielu metod asercji dostępnych w klasie `Assert`. Większość metod asercji dotyczy jednego zagadnienia — porównywania dwóch wartości. W tabeli 9.3 przedstawiono bardziej kompletną listę metod.

Tabela 9.3. Asercje testów

Test	Opis
<code>AreEqual</code> i <code>AreNotEqual</code>	Służy do sprawdzania, czy dwie wartości są sobie równe, czy nie.
<code>AreSame</code> i <code>AreNotSame</code>	Pozwala sprawdzić, czy dwa obiekty są tym samym obiektem, czy nie.
<code>Fail</code>	Metoda <code>Assert.Fail</code> oznacza warunek jako niespełniony bez jego sprawdzania. Może to być przydatne przy wartościowaniu warunku za pomocą operacji logicznych i dojściu do miejsca w kodzie, gdzie operacje reprezentują test zakończony niepowodzeniem.
<code>Inconclusive</code>	Służy do oznaczania, że test nie zakończył się ani niepowodzeniem, ani powodzeniem (wynik jest niejednoznaczny).
<code>IsInstanceOfType</code> i <code>IsNotInstanceOfType</code>	Pozwala sprawdzić, czy obiekt jest określonego typu, czy nie.
<code>IsNull</code> i <code>IsNotNull</code>	Pozwala określić, czy obiekt zawiera referencję <code>null</code> , czy nie.
<code>IsTrue</code> i <code>IsFalse</code>	Służy do sprawdzania, czy warunek jest spełniony, czy nie.

Każda metoda asercji ma kilka przeciążonych wersji. Umożliwiają one porównywanie różnych typów danych — łańcuchów znaków, liczb, obiektów i kolekcji ogólnych. Ponadto dostępne są wersje, które pozwalają po prostu zastosować asercję lub dodatkowo wyświetlić komunikat w sytuacji, kiedy asercja nie jest spełniona. Na przykład poniżej zapisano tę samą asercję, co wcześniej, jednak ta wersja wyświetla komunikat o błędzie, jeśli warunek nie jest spełniony.

```
Assert.AreEqual(cust.Id, customerId, "Identyfikatory klienta są różne.");
```

Przyjrzyjmy się kilku przykładowym metodom, których szkielet pokazano na wcześniejszym listingu (zobacz listing 9.4). Należy uzupełnić te metody przez napisanie kodu korzystającego z klasy `Customer`, a następnie poczynić asercje na temat wyników. Listing 9.5 zawiera przykładowy test konstruktora klasy `Customer`. Warto zauważyć, że testujemy znany rekord danych przez wczytanie egzemplarza klasy `Customer` na podstawie identyfikatora rekordu. Następnie sprawdzamy wyniki w klasie `Assert`.

Listing 9.5. Przykładowy test jednostek ilustrujący korzystanie z klasy `Assert`

```
[TestMethod()]
public void CustomerConstructorTest1() {

    // Testuje konstruktor przyjmujący identyfikator klienta.

    int customerId = 1; // Testowanie danych klienta.
    Customer cust = new Customer(customerId);

    // Sprawdzanie, czy dla testowego klienta zwrócono poprawne dane.
    // Sprawdzanie operacji ustawiania i wczytywania wartości właściwości.
    Assert.AreEqual(cust.Id, customerId,
        "Identyfikatory klienta są różne.");

    Assert.AreEqual(cust.Email, "jdevelop@contoso.com");
    Assert.AreEqual(cust.Name, "Jan Programista");
    Assert.AreEqual(cust.City, "Brzeg");
    Assert.AreEqual(cust.PostalCode, "00000");
    Assert.AreEqual(cust.RegionState, "DL");
    Assert.AreEqual(cust.Address1, "ul. Microsoftu 100");
    Assert.IsTrue(cust.Address2 == "");
    Assert.IsTrue(cust.Phone == "");
}
```

Klasa `Assert` zawiera też wersje metod `AreEqual` i `AreNotEqual` dla ogólnych typów danych. Wersje te umożliwiają sprawdzanie, czy dwa typy ogólne są takie same. Programista musi określić typ ogólny za pomocą standardowej składni, `<T>` [lub `(of T)` w języku Visual Basic], i przekazać dwie porównywane zmienne. Poniżej pokazano przykładowy kod:

```
Assert.AreEqual<Customer>(cust1, cust2);
```

Sprawdzanie kolekcji obiektów

Przestrzeń nazw `UnitTesting` obejmuje też klasę asercji `CollectionAssert`. Przy jej użyciu można sprawdzić zawartość klas kolekcji. Można na przykład wywołać metodę `Contains`, aby ustalić, czy dana kolekcja zawiera konkretny element (lub go nie zawiera). Metoda `AllItems` ↪ `AreInstancesOfType` służy do sprawdzania, czy kolekcja zawiera egzemplarze tego samego typu. Można też sprawdzić, czy dwie kolekcje są takie same (`AreEqual` i `AreNotEqual`) lub czy zawierają identyczne elementy, ale niekoniecznie w tej samej kolejności (`AreEquivalent` i `AreNotEquivalent`).

Sprawdzanie łańcuchów znaków

Klasa `StringAssert` zawiera metody służące do sprawdzania łańcuchów znaków i ich fragmentów. Metoda `Contains` pozwala ustalić, czy łańcuch obejmuje określony podłańcuch. Metoda `StartsWith` umożliwia określenie, czy łańcuch rozpoczyna się od podanych znaków, a metoda `EndsWith` sprawdza końcową część łańcucha. Metody `Matches` i `DoesNotMatch` umożliwiają ustalenie, czy łańcuch znaków pasuje do zdefiniowanego wyrażenia.

Testowanie wyjątków

Testy jednostek należy pisać w taki sposób, aby ustalić, czy kod działa w oczekiwany sposób zarówno w korzystnych, jak i niekorzystnych warunkach. Korzystne warunki można sprawdzić w opisany wcześniej sposób za pomocą metod klasy `Assert`. Jednak często trzeba określić, czy kod zgłasza odpowiedni wyjątek w odpowiedzi na specyficzne wywołanie. Można wtedy zastosować atrybut `ExpectedException` do przetestowania określonych warunków wystąpienia błędów.

Atrybut `ExpectedException` należy dodać do metody testowej. Atrybut przyjmuje jako parametr typ oczekiwanego wyjątku. Jeśli wywołanie metody testowej spowoduje zgłoszenie wyjątku odpowiedniego typu, test zakończy się powodzeniem. Jeżeli wyjątek nie wystąpi lub będzie miał niewłaściwy typ, test zakończy się niepowodzeniem.

Załóżmy, że programista chce sprawdzić, co stanie się przy próbie utworzenia nowego klienta przy użyciu niepełnych danych. Może w tym celu napisać kod zgłaszający niestandardowy wyjątek typu `InvalidCustomerException` i dodać do metody testowej następujący atrybut:

```
[TestMethod()]
[ExpectedException(typeof(InvalidCustomerException),
    "Nie zgłoszono wyjątku InvalidCustomerException.")]
public void NewCustomerTest() {

    // Tworzenie nieprawidłowego nowego egzemplarza klasy Customer.
}
```

Warto zauważyć, że jeśli powyższy kod nie zgłosi wyjątku, jako wynik testu przekazany zostanie komunikat o błędzie ("Nie zgłoszono wyjątku `InvalidCustomerException`"). Komunikat to opcjonalny parametr atrybutu `ExpectedException`.

Można łączyć asercje z atrybutem `ExpectedException`. Aby metoda testowa w takich warunkach przeszła test, asercje muszą być spełnione, a metoda musi zgłosić wyjątek.

Uwaga



Zgłoszony wyjątek musi mieć dokładnie taki sam typ, jak oczekiwany. Wyjątek nie może na przykład dziedziczyć po oczekiwanym. Wtedy test zakończy się niepowodzeniem.

Tworzenie testów jednostek zależnych od danych

Często lepiej jest napisać jeden test jednostek i uruchomić go wielokrotnie dla różnych wartości parametrów, niż przygotowywać wiele podobnych testów. Cechą dobrych testów jednostek jest pokrycie różnych warunków. Oczywiście, testy jednostek nie przyjmują parametrów. Utrudnia to przekazywanie danych do testów. Dlatego trzeba powiązać testy jednostek z danymi. Następnie można nakazać platformie testów jednostek uruchomienie testu dla każdego wiersza danych. Można też wykorzystać powiązane dane testowe w samym teście jednostek i wywołać kod na wiele sposobów.

Łączenie się z danymi

Do wiązania testów jednostek z danymi testowymi służy klasa atrybutu `DataSource` z platformy testów jednostek. Klasa umożliwia bezpośrednie podanie łańcucha połączenia z danymi testowymi lub określenie nazwy łańcucha połączenia zapisanego w pliku konfiguracji projektu. Obie techniki pozwalają nawiązać połączenie z bazą danych (na przykład SQL Server lub Access), plikiem `.csv` (zwykle tworzonym w Notatniku lub Excelu) lub plikiem `.xml`.

Klasa atrybutu `DataSource` ma trzy wersje. Pierwsza przyjmuje jeden parametr, `dataSource` ↪ `SettingName`. Wymaga to przekazania nazwy ustawień źródła danych zdefiniowanych w pliku konfiguracji. Druga wersja przyjmuje parametry `connectionString` i `tableName`. Należy wtedy przekazać poprawny łańcuch połączenia ze źródłem danych i określić nazwę tabeli, którą system ma powiązać z testem jednostek. Ostatnia wersja przyjmuje parametry `providerInvariantName`, `connectionString`, `tableName` i `dataAccessMethod`. Pierwszy służy do określania typu dostawcy, na przykład dostawcy plików CSV, bazy SQL Server i tak dalej. Drugi, łańcuch połączenia, zależy od wybranego dostawcy i określa sposób dostępu do danych. Trzeci parametr to nazwa tabeli (lub pliku) z danymi. Czwarty, metoda dostępu do danych, określa sposób powiązania danych z testem jednostek — sekwencyjny lub swobodny.

Załóżmy, że programista pisze test jednostek do sprawdzenia informacji o zamówieniach i koncie klienta. Może wczytać dane klienta, pobrać złożone zamówienia, określić przyznany kredyt i sprawdzić obecny stan rachunku. Wszystkie te operacje dla danego klienta można wykonać w jednym teście jednostek. Test można powiązać z wierszami danych testowych klienta, aby sprawdzić poprawność kodu w różnych scenariuszach. Przyjmijmy, że dane testowe znajdują się w pliku `.csv` w aplikacji testowej. Wtedy atrybut `DataSource` można dodać do metody testowej w następujący sposób:

```
[TestMethod(),
DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",
"|DataDirectory|\OrderTestData.csv", "OrderTestData#csv",
DataAccessMethod.Sequential),
DeploymentItem("BusinessDomainUnitTests\OrderTestData.csv")]
public void VerifyCustomerOrders() {
```

```
    // Kod sprawdzający informacje o zamówieniach i koncie klienta.
```

```
}
```

Warto zauważyć, że w przykładowym kodzie pierwszy parametr atrybutu `DataSource` określa dostawcę plików `.csv`. Następny parametr to łańcuch połączenia z plikiem danych. Trzeci parametr (`OrderTestData#csv`) informuje, że nazwa tabeli nie istnieje (użyto nazwy pliku). Ostatni parametr, wartość wyliczeniowa `DataAccessMethod.Sequential`, określa, że każdy wiersz należy powiązać z testem jednostek w sposób sekwencyjny. Zwróćmy też uwagę na to, że dodano atrybut `DeploymentItem`, aby zagwarantować, że plik z danymi testowymi zostanie zainstalowany wraz z testem jednostek.

Nawiązywanie połączenia przy użyciu pliku konfiguracyjnego

W innym scenariuszu można przechowywać informacje o połączeniu w pliku konfiguracyjnym. Jest to przydatne, jeśli programista nie chce na trwałe zapisywać informacji w kodzie testów jednostek. Możliwe, że ta sama testowa baza danych jest wykorzystywana w wielu testach. W środowisku programistycznym i testowym połączenie z bazą może wyglądać inaczej. Wtedy umieszczenie informacji o połączeniu w pliku konfiguracyjnym ułatwia zarządzanie nimi.

Pierwszy krok w tym podejściu polega na utworzeniu pliku konfiguracyjnego. W tym celu wystarczy dodać plik `app.config` do projektu testów jednostek (należy kliknąć plik projektu prawym przyciskiem myszy i wybrać opcję *Add/New Item*). Jedyną trudność to prawidłowe określenie formatu pliku konfiguracyjnego. Trzeba dodać sekcję konfiguracyjną specyficzną dla narzędzi testowych środowiska Visual Studio. Warto zauważyć, że w sekcji należy wskazać wersję 10. platformy testów jednostek. Oto przykładowy kod:

```
<configSections>
  <section name="microsoft.visualstudio.testtools"

  type="Microsoft.VisualStudio.TestTools.UnitTesting.TestConfigurationSection,
  Microsoft.VisualStudio.TestTools.UnitTesting.Framework, Version=10.0.0.0,
  ↪Culture=
  neutral, PublicKeyToken=b03f5f7f11d50a3a" />
</configSections>
```

Następnie należy dodać sekcję z łańcuchem połączenia. Trzeba określić nazwę łańcucha i podać informacje o połączeniu oraz dostawcy danych. Tekst łańcucha zależy oczywiście od dostawcy i rodzaju danych. Dla danych przechowywanych w plikach `.xlsx`, `.csv` i `.xml` oraz bazach Access, Oracle i SQL potrzebne są specyficzne elementy w łańcuchu połączenia. W poniższym przykładzie wykorzystano dostawcę `OleDb` do utworzenia łańcucha połączenia dla pliku `.csv`:

```
<connectionStrings>
  <add name="OrderTestDataCnn"
    connectionString="Provider=Microsoft.Jet.OLEDB.4.0; Data Source=
  |DataDirectory|TestData\; Extended Properties='text;HDR=Yes;FMT=Delimited'"
    providerName="System.Data.OleDb" />
</connectionStrings>
```

Ostatni krok polega na utworzeniu sekcji źródła danych zagnieżdżonej w sekcji narzędzi testowych. Należy określić nazwę testowego źródła danych i wskazać łańcuch połączenia. Ponadto można podać metodę dostępu do danych (dostęp sekwencyjny lub swobodny) i nazwę tabeli (jeśli jest potrzebna). Ostatecznie plik konfiguracyjny powinien wyglądać tak jak na listingu 9.6.

Listing 9.6. Przykładowy plik konfiguracyjny testu jednostek

```

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="microsoft.visualstudio.testtools"

type="Microsoft.VisualStudio.TestTools.UnitTesting.TestConfigurationSection,
Microsoft.VisualStudio.TestTools.UnitTesting, Version=10.0.0.0,
↪Culture=
neutral, PublicKeyToken=b03f5f7f11d50a3a" />
  </configSections>
  <connectionStrings>
    <add name="OrderTestDataCnn"
      connectionString="Provider=Microsoft.Jet.OLEDB.4.0; Data Source=
|DataDirectory|TestData\; Extended Properties='text;HDR=Yes;FMT=Delimited' "
      providerName="System.Data.OleDb" />
  </connectionStrings>
  <microsoft.visualstudio.testtools>
    <dataSources>
      <add name="OrderTestDataSource"
        connectionString="OrderTestDataCnn" dataTableName="OrderTestData#csv"
        dataAccessMethod="Sequential"/>
    </dataSources>
  </microsoft.visualstudio.testtools>
</configuration>

```

Uwaga

Warto zauważyć, że w jednym pliku konfiguracyjnym można zdefiniować wiele łańcuchów połączeń (każdy z nich musi mieć inną nazwę).

Następnie można wykorzystać skonfigurowane testowe źródło danych w metodzie testowej. W tym celu wystarczy podać nazwę źródła danych w atrybucie `DataSource` metody testowej. Prowadzi to do zwiększenia przejrzystości kodu, ułatwia wielokrotne wykorzystanie łańcuchów połączeń i umożliwia centralne zarządzanie połączeniem, jeśli jest ono inne na poszczególnych komputerach. Poniżej pokazano, jak wykorzystać źródło danych:

```

[TestMethod(),
DataSource("OrderTestDataSource")]
public void VerifyCustomerOrders() {

    // Kod sprawdzający informacje o zamówieniach i koncie klienta.

}

```

Warto zauważyć, że w tym przykładzie dane testowe znajdują się w pliku `.csv` o nazwie `Order-TestData.csv`. Plik dodano do katalogu `TestData` w projekcie testów. Katalog i plik są następnie przenoszone do katalogu `bin`, w którym środowisko uruchamia testy. Dzieje się tak, ponieważ

łańcuch połączenia obejmuje fragment `|DataDirectory|TestData\`. Można też bezpośrednio wymusić instalowanie plików, dodając atrybut `DeploymentItem` do metody testowej.

Wskazówka



Łańcuchy połączeń dla testów jednostek można skonfigurować za pomocą okna dialogowego *Properties*. Zobacz na przykład punkt „Konfigurowanie atrybutów testów jednostek” w dalszej części rozdziału.

Stosowanie danych testowych w asercjach

Dostęp do danych testowych powiązanych z metodami testowymi można uzyskać za pomocą obiektu kontekstu testu. Przypomnijmy, że jeśli programista ustawi właściwość typu `TestContext` (zobacz punkt „Klasa `TestContext`”), obiekt kontekstu zostanie udostępniony przez platformę. Metoda `DataRow` klasy `TestContext` pozwala pobrać wiersz powiązany z wykonywanym testem. Do metody można przekazać nazwę pola lub indeks, aby pobrać dane z odpowiedniej kolumny. Zwykle należy ustawić zmienne dla wszystkich pól danych testowych. Następnie można wykorzystać zmienne w asercjach w testach jednostek.

Na listingu 9.7 pokazano przykład zastosowania danych testowych (układ danych testowych przedstawiono na rysunku 9.18 w dalszej części rozdziału). Warto zauważyć, że każdą kolumnę trzeba rzutować lub przekształcić na odpowiedni typ danych. Następnie można napisać właściwe asercje.

Listing 9.7. Przykładowy test jednostek, w którym wykorzystano powiązane dane

```
[TestMethod(),
DataSource("OrderTestDataSource")]
public void VerifyCustomerOrders() {

    // Pobieranie wartości z wierszy danych.
    string email = (string)TestContext.DataRow["email"];
    int numOrders = (int)TestContext.DataRow["num_orders_todate"];
    double creditLimit = Convert.ToDouble(TestContext.DataRow["credit_limit"]);
    double accountBalance =
        ↳Convert.ToDouble(TestContext.DataRow["account_balance"]);

    // Tworzenie obiektu Customer na podstawie adresu e-mail.
    Customer cust = new Customer(email);

    // Sprawdzanie informacji o zamówieniach i koncie klienta.
    Assert.IsTrue(cust.OrderHistory.Count == numOrders,
        "Liczba zamówień jest nieprawidłowa.");
    Assert.IsTrue(cust.Account.Balance == accountBalance,
        "Stan rachunku jest nieprawidłowy.");
    Assert.IsTrue(cust.Account.CreditLimit == creditLimit,
        "Limit debetu jest nieprawidłowy.");
}
```

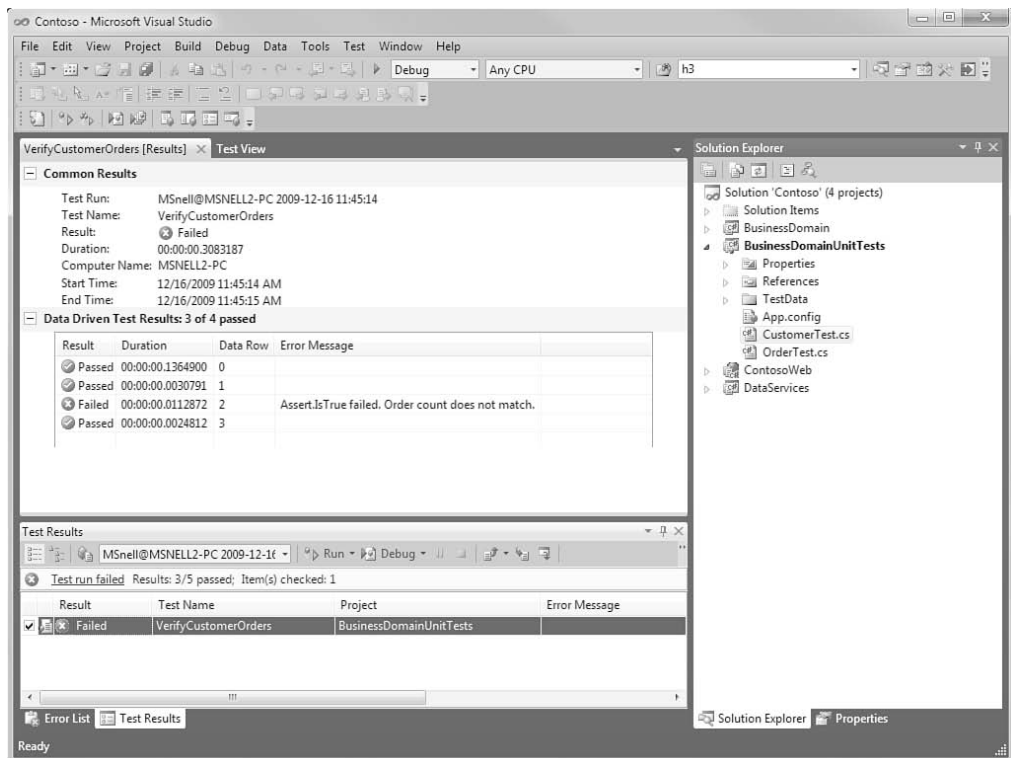
```

Assert.IsTrue(cust.Account.CreditLimit > cust.Account.Balance,
    "Limit debetu musi być mniejszy niż stan rachunku.");
}

```

Uruchamianie testów zależnych od danych i wyświetlanie ich wyników

Testy jednostek zależne od danych można uruchamiać w taki sam sposób jak inne testy jednostek. Główna różnica polega na tym, że cały test jednostek kończy się niepowodzeniem, jeśli test zakończy się w ten sposób dla choćby jednego wiersza danych. Można dwukrotnie kliknąć wynik nieudanego testu, aby wyświetlić szczegółowe informacje na jego temat. Wyniki testów zależnych od danych pojawiają się w odrębnym okienku pod standardowymi wynikami testu jednostek. Przykładowe dane pokazano na rysunku 9.16.



Rysunek 9.16. Wyniki testów jednostek zależnych od danych wyświetlane są pod standardowymi wynikami całego testu jednostek

Warto zauważyć, że dla trzech z czterech wierszy testy jednostek zakończyły się powodzeniem. Można zobaczyć komunikat o błędzie (z asercji) dla wiersza, który spowodował błąd. Dwukrotne kliknięcie tego wiersza spowoduje wyświetlenie szczegółowych informacji o niepowodzeniu, w tym śladu stosu błędów i odnośnika do wiersza kodu, gdzie wystąpił problem.

Pisanie testów jednostek działających w ASP.NET

Można wybrać, czy testy jednostek mają być przechowywane i uruchamiane w środowisku ASP.NET zamiast w domyślnym procesie hosta VSTest. Jest to przydatne przy pisaniu testów współdziałających z obiektami ASP.NET, takimi jak Page i Session. Czasem programista chce przetestować usługę sieciową lub inny kod aplikacji związany z interfejsem sieciowym. Można wtedy umieścić testy jednostek dla aplikacji ASP.NET na serwerze IIS lub na serwerze środowiska Visual Studio. Takie rozwiązanie gwarantuje, że testy będą miały pełny dostęp do zmiennych środowiskowych ASP.NET.

Definiowanie atrybutów środowiska ASP.NET

Aby określić, że test jednostek ma działać w środowisku ASP.NET, należy dodać odpowiednie atrybuty do metody testowej. Liczne atrybuty znajdują się w przestrzeni nazw `UnitTesting.Web`. Dlatego należy najpierw dodać odpowiednią instrukcję `using` (imports w języku Visual Basic) w górnej części pliku klasy testów, które mają działać w ASP.NET. Instrukcja powinna wyglądać tak:

```
using Microsoft.VisualStudio.TestTools.UnitTesting.Web;
```

Ponadto w aplikacji z testem jednostek należy dodać referencję do witryny. Umożliwi to dostęp do stron witryny i innych klas, które mogą znajdować się w katalogu `App_Code` oraz innych miejscach.

Przy tworzeniu testów jednostek w ASP.NET używane są trzy główne atrybuty: `UrlToTest`, `HostType` i `AspNetDevelopmentServerHost`. Atrybut `UrlToTest` umożliwia określenie strony, którą należy wywołać w czasie uruchamiania danego testu jednostek. Strona jest wywoływana przez platformę testów, a dla testu jednostek dostępny jest kontekst żądania sieciowego (poprzez obiekt Page). W kodzie testu jednostek można korzystać ze środowiska ASP.NET w taki sam sposób, jak w pliku z kodem ukrytym strony internetowej.

Atrybut `HostType` umożliwia zmianę typu hosta na ASP.NET. W tym celu należy przekazać do atrybutu łańcuch znaków `ASP.NET`.

Przy korzystaniu z hosta w postaci serwera IIS wystarczy ustawić atrybuty `UrlToTest` i `HostType`. Jeśli jednak programista używa serwera ASP.NET Development (współdziałającego z Visual Studio), musi ponadto dodać atrybut `AspNetDevelopmentServerHost`. Jako parametr atrybutu należy przekazać ścieżkę do aplikacji sieciowej. Trzeba też podać nazwę głównego katalogu aplikacji sieciowej.

Na listingu 9.8 pokazano przykład zastosowania wszystkich trzech wymienionych atrybutów do zdefiniowania testu jednostek ASP.NET działającego na lokalnym serwerze rozwojowym.

Listing 9.8. *Test jednostek ASP.NET*

```
[TestMethod()]
[HostType("ASP.NET")]
[AspNetDevelopmentServerHost("C:\\CODE\\Unleashed\\Contoso\\ContosoWeb", "/")]
[UrlToTest("http://localhost:55136/ShoppingCart.aspx")]
public void AddShoppingCartItemTest() {

    // Pobieranie żądanej strony.
    Page reqPage = TestContext.RequestedPage;
    Assert.IsTrue(reqPage.Title == "Koszyk zakupów", "Tytuł strony jest
    ↪niewłaściwy.");

    // Rzutowanie na typ strony.
    ShoppingCartPage actualPage = (ShoppingCartPage)reqPage;
    Assert.IsNotNull(actualPage.Cart, "Na stronie nie ma koszyka.");

    // Sprawdzanie poprawności kodu do obsługi koszyka zakupów.
    actualPage.Cart.AddItem("Produkt 1", 1, 12.75);
    actualPage.Cart.AddItem("Produkt 2", 2, 26.95);
    Assert.IsTrue(actualPage.Cart.Items.Count == 2, "Liczba elementów jest
    ↪niewłaściwa.");
    Assert.AreEqual(actualPage.Cart.CalculateCartTotal(), 66.65);

}
```

Warto zauważyć, że referencje do obiektów ASP.NET pochodzą z właściwości `RequestedPage` obiektu `TestContext`. Właściwości tej można użyć do rzutowania danych bezpośrednio na typ żądanej strony (tu jest to typ `ShoppingCartPage`). Oczywiście, właściwość `RequestedPage` ma typ `System.Web.UI.Page`, dlatego zapewnia dostęp do obiektów `Server`, `Session`, `Request`, `Response` i podobnych.

Generowanie testów jednostek ASP.NET

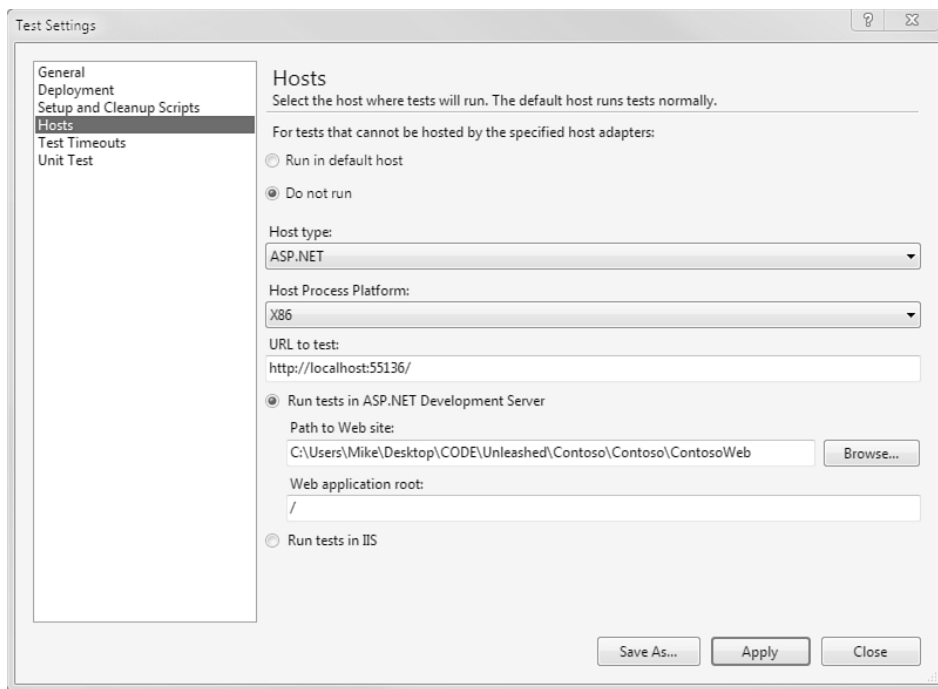
Za pomocą Visual Studio można automatycznie wygenerować testy jednostek ASP.NET. Często jest to zalecana metoda tworzenia testów, ponieważ przygotowuje projekt testów do pracy ze środowiskiem ASP.NET i automatycznie konfiguruje atrybuty.

Testy jednostek ASP.NET generuje się w taki sam sposób, jak wszystkie inne testy jednostek. Najpierw należy otworzyć kod (plik z kodem ukrytym strony lub inny plik klasy). Następnie trzeba kliknąć prawym przyciskiem myszy kod i wybrać opcję *Create Unit Tests*. Spowoduje to uruchomienie kreatora *Create Unit Tests* (zobacz rysunek 9.5). W tym miejscu można wybrać metody, dla których kreator ma wygenerować testy. Obsługiwane są zdarzenia związane ze stroną, kliknięcie przycisków i inne zdarzenia kontrolki. Można też napisać test jednostek wyszukujący kontrolki na stronie i sprawdzający ich zawartość.

Konfigurowanie hosta projektu testów

Programiści często tworzą dla witryny odrębny projekt testów jednostek. Projekt może dotyczyć testów powiązanych z funkcjami witryny. Można też utworzyć dodatkowe projekty testów do sprawdzania innych bibliotek kodu. Tego typu podział ułatwia pisanie i przeprowadzanie testów jednostek.

Ponadto można skonfigurować ustawienia projektu testów specyficznie pod kątem hosta. Dzięki temu nie trzeba definiować licznych atrybutów ASP.NET w każdym teście jednostek. Zamiast tego można skonfigurować plik *.testsettings* w taki sposób, aby współdziałał ze specyficznym hostem. W tym celu należy kliknąć dwukrotnie plik *.testsettings* (*Solution Items/Local.testsettings*), co spowoduje wyświetlenie okna dialogowego *Test Settings*, widocznego na rysunku 9.17.



Rysunek 9.17. Plik *.testsettings* można wykorzystać do skonfigurowania całego projektu testów, tak aby działał dla witryny ASP.NET

Wskazówka



Jeśli rozwiązanie zawiera wiele projektów testów jednostek, warto utworzyć kilka wersji pliku *.testsettings*. Następnie można łatwo przełączać się między konfiguracjami dla hostów ASP.NET i standardowego (VSTest).

W oknie dialogowym należy otworzyć widoczny po lewej stronie węzeł *Hosts*, a następnie ustawić wartość *ASP.NET* dla opcji *Host type*. Spowoduje to udostępnienie kilku dodatkowych opcji: *URL to Test*, *Path to Web Site* i *Web Application Root*. Każdą z tych opcji można ustawić na wartości domyślne dla całego projektu testów jednostek.

Konfigurowanie atrybutów testów jednostek

Pokazaliśmy, że istnieje wiele atrybutów, które można zastosować do testów jednostek. Oczywiście, możliwe jest skonfigurowanie każdego z nich w edytorze kodu. Jednak Visual Studio udostępnia pewną pomoc w oknie dialogowym *Properties*. Można wybrać dany test jednostek, wyświetlić okno *Properties*, a następnie skonfigurować atrybuty stosowane do metody testowej. Visual Studio wygeneruje kod atrybutów dla metody testowej. Przyjrzyjmy się przykładowi.

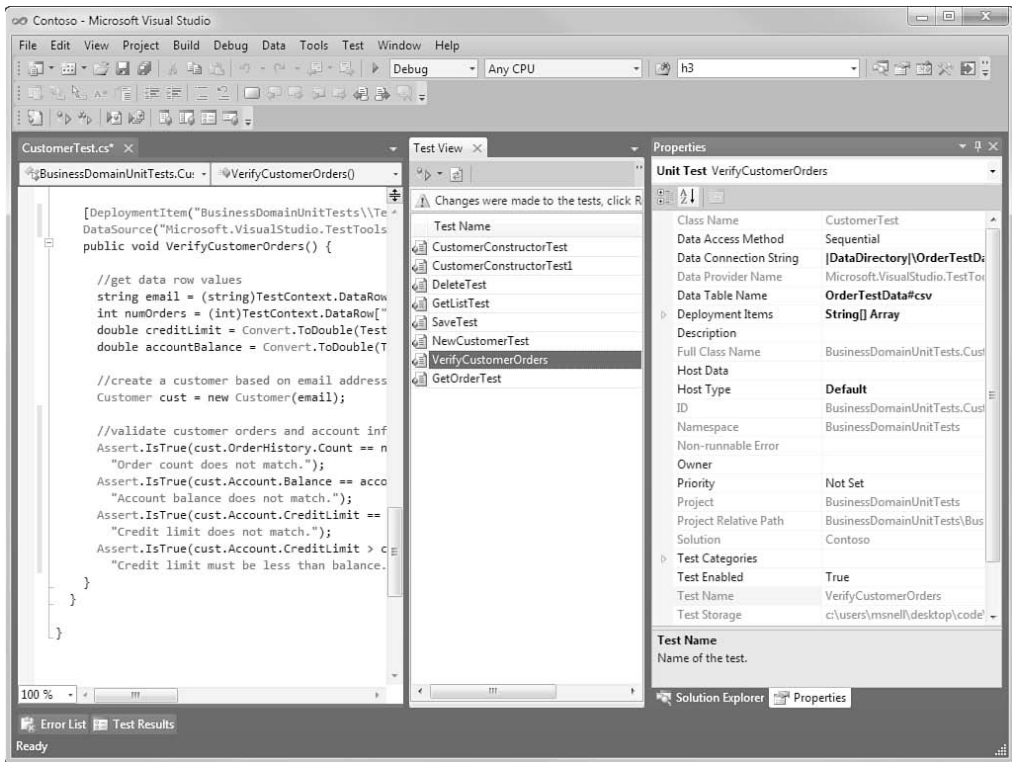
Definiowanie połączenia za pomocą okna dialogowego *Properties*

Pierwszym krokiem przy stosowaniu okna *Properties* do konfigurowania testów jednostek jest wybranie testu. Należy to zrobić w oknie *Test View* (*Test/Windows/Test View*). Następnie można kliknąć prawym przyciskiem myszy test na liście i wybrać opcję *Properties*, aby wyświetlić okno *Properties* dla danego testu. Na rysunku 9.18 pokazano przykładowe dane.

Warto zauważyć, że w oknie *Properties* znajdują się właściwości wielu atrybutów, które można zastosować do metody testowej. Zmiana wartości jednej z właściwości spowoduje wygenerowanie przez środowisko Visual Studio atrybutu i kodu w pliku z kodem (między kodem i omawianym oknem następuje dwustronna synchronizacja).

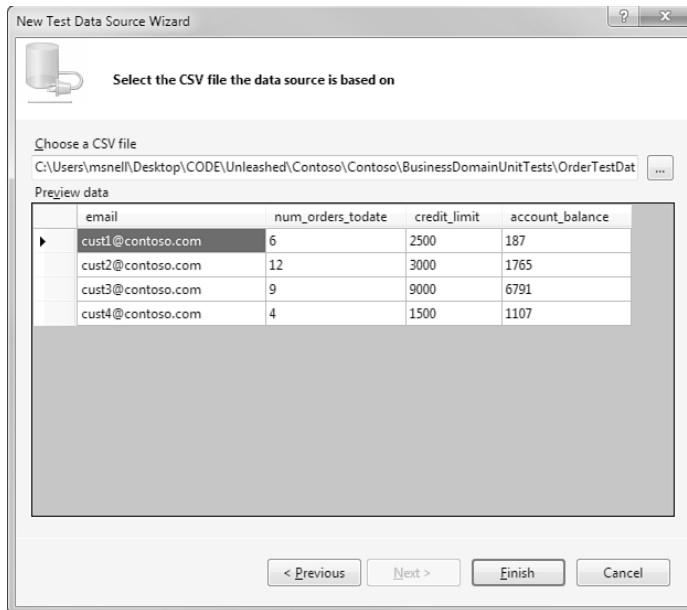
Przykładowo, właściwość *Data Connection String* ułatwia zdefiniowanie atrybutu *DataSource*. Kliknięcie przycisku z wielokropkiem przy właściwości powoduje uruchomienia kreatora, dzięki któremu można zdefiniować łańcuch połączenia z bazą danych SQL, plikiem CSV lub plikiem XML (oczywiście, można nawiązać połączenie także z innymi źródłami danych, ale wymaga to napisania własnego łańcucha połączenia). We wcześniejszym przykładzie używaliśmy pliku CSV. Po zaznaczeniu opcji związanej z tym formatem można przejść do pliku CSV, a nawet podejrzec zapisane w nim dane. Na rysunku 9.19 pokazano przykładowy ekran kreatora. Po ustawieniu wszystkich opcji kreator zaktualizuje kod, dodając atrybuty *DataSource* i *DeploymentItem*. Poniżej znajduje się uzyskany kod konfiguracyjny:

```
[TestMethod(),
    DeploymentItem("BusinessDomainUnitTests\TestData\OrderTestData.csv"),
    DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",
        "|DataDirectory|OrderTestData.csv", "OrderTestData#csv",
        DataAccessMethod.Sequential)]
public void VerifyCustomerOrders() { ...
```



Rysunek 9.18. Można zaznaczyć test w oknie Test View, a następnie skonfigurować jego atrybuty w oknie Properties

Rysunek 9.19. Testy jednostek można powiązać z testowymi danymi, w tym z prostym plikiem .csv

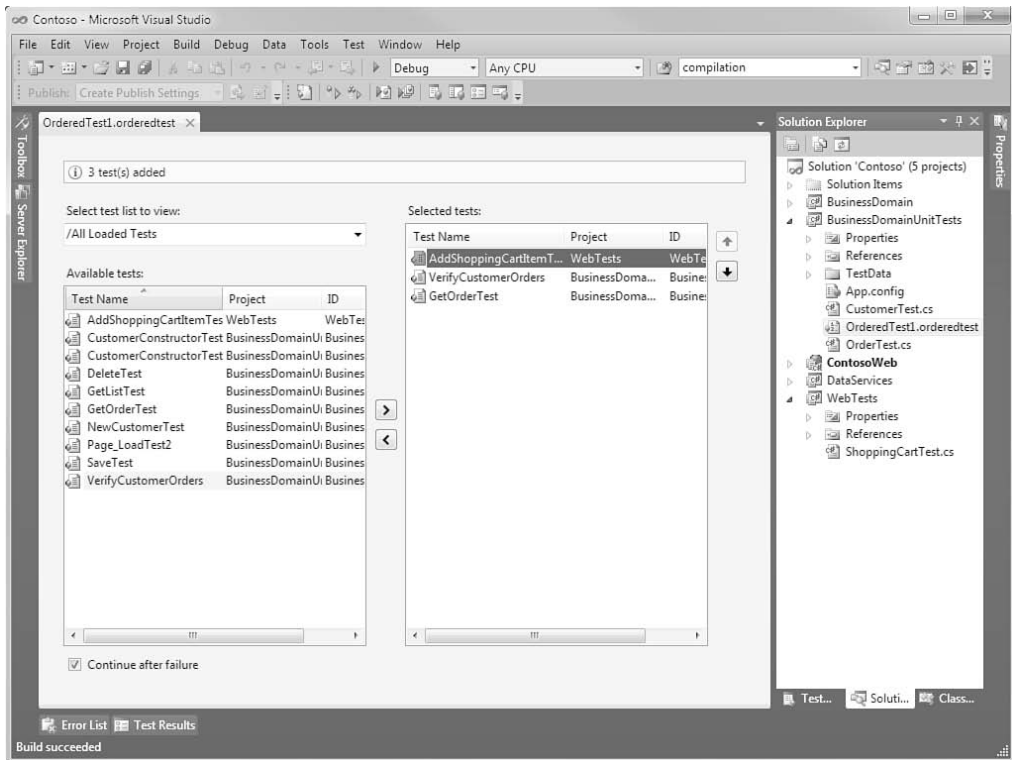


Tworzenie testów uporządkowanych

Visual Studio umożliwia łączenie grup testów jednostek, określanie kolejności ich wykonywania i traktowanie wyników w taki sposób, jakby uzyskano je dla pojedynczego testu. Może to być przydatne, jeśli programista chce napisać testy jednostek zależne od siebie. Można na przykład wstawić rekord w jednym teście i oczekiwać, że ten sam rekord będzie dostępny w dalszym teście. Oczywiście, jest to niezgodne z dobrymi praktykami testów jednostek — możliwe powinno być przeprowadzenie każdego testu niezależnie od pozostałych. Na szczęście można utworzyć test uporządkowany, który łączy poszczególne testy jednostek w nowy, niezależny test.

Test uporządkowany można dodać do projektu testów, klikając projekt prawym przyciskiem myszy i wybierając opcję *Add/Ordered Test*. Można też wybrać szablon *Ordered Test* w oknie dialogowym *Add New Test* (zobacz rysunek 9.3).

Test uporządkowany to po prostu plik XML oparty na szablonie *Ordered Test*. Nie trzeba jednak ręcznie edytować kodu w języku XML. Środowisko Visual Studio udostępnia jako pomoc okno projektowe dla testów uporządkowanych. Przykładową zawartość takiego okna pokazano na rysunku 9.20.



Rysunek 9.20. Do testu uporządkowanego można dodać istniejący test jednostek, aby utworzyć nowy test, w którym poszczególne testy uruchamiane są w określonej kolejności

Po lewej stronie okna dialogowego znajdują się wszystkie testy z rozwiązania. Można wyświetlić testy z wybranej listy (zobacz podrozdział „Organizowanie testów”). Po lewej stronie należy zaznaczyć poszczególne testy i użyć strzałki (>) w celu ich dodania do testu uporządkowanego. Skierowane w górę i w dół strzałki widoczne po prawej stronie służą do zmiany kolejności przeprowadzania testów.

Przy wykonywaniu testów uporządkowanych środowisko Visual Studio uruchamia każdy test jednostek w podanej kolejności. Jeśli którykolwiek test się nie powiedzie, cały test uporządkowany zakończy się niepowodzeniem. Oczywiście, można wyświetlić szczegółowe informacje na temat testu uporządkowanego, aby zobaczyć, które testy zakończyły się sukcesem, a które — porażką.

Organizowanie testów

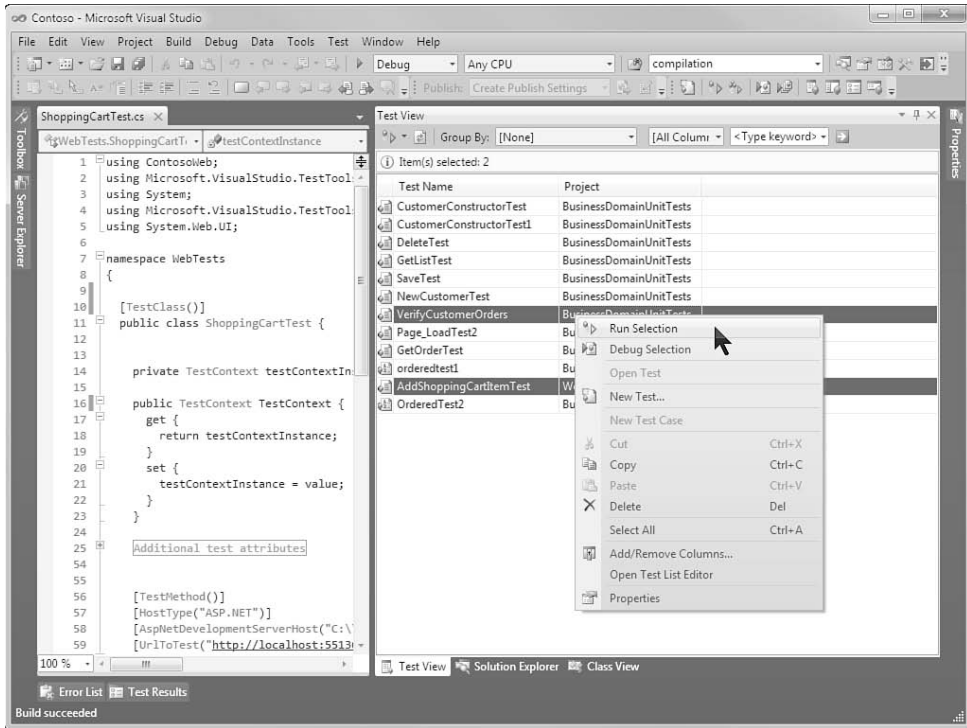
Większość projektów testów jednostek składa się z licznych testów — nieraz są ich setki. Ponadto często w jednym rozwiązaniu znajduje się wiele projektów testów jednostek. Zarządzanie wszystkimi testami i uruchamianie tylko tych potrzebnych w danym momencie bywa trudne. Visual Studio udostępnia pewną pomoc w tym zakresie — okna *Test View* i *Test List Editor*.

Okno Test View

Okno *Test View* umożliwia wyświetlenie wszystkich testów w otwartym rozwiązaniu, zaznaczenie tych, którymi programista jest zainteresowany, i albo uruchomienie ich, albo zarządzanie właściwościami ich atrybutów.

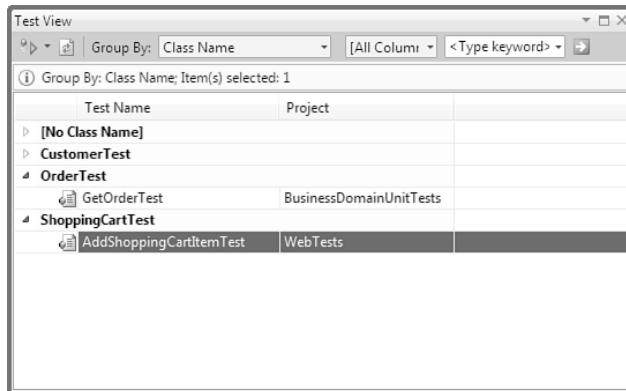
Dostęp do okna *Test View* można uzyskać z menu *Test (Test/Windows/Test View)*. Na rysunku 9.21 pokazano przykładową zawartość okna. Warto zauważyć, że można zaznaczyć jeden lub kilka testów, kliknąć prawym przyciskiem myszy i uruchomić wybrane testy w jednym przebiegu. Dzięki temu nie trzeba uruchamiać wszystkich testów jednostek, jeśli ważna jest tylko jedna metoda testowa (lub ich mała grupa).

W górnej części okna *Test View* znajduje się pasek narzędzi. Umożliwia on wyszukiwanie testów na podstawie słów kluczowych i modyfikowanie sposobów grupowania lub filtrowania testów. Sprawia to, że znalezienie testów jest łatwiejsze. Można na przykład zmienić wartość opcji *Group By* na *Class Name*, aby wyświetlić wszystkie testy dla określonej klasy. Przykładowe ustawienia pokazano na rysunku 9.22.



Rysunek 9.21. Okno Test View umożliwia łatwe zaznaczenie zbioru testów do uruchomienia

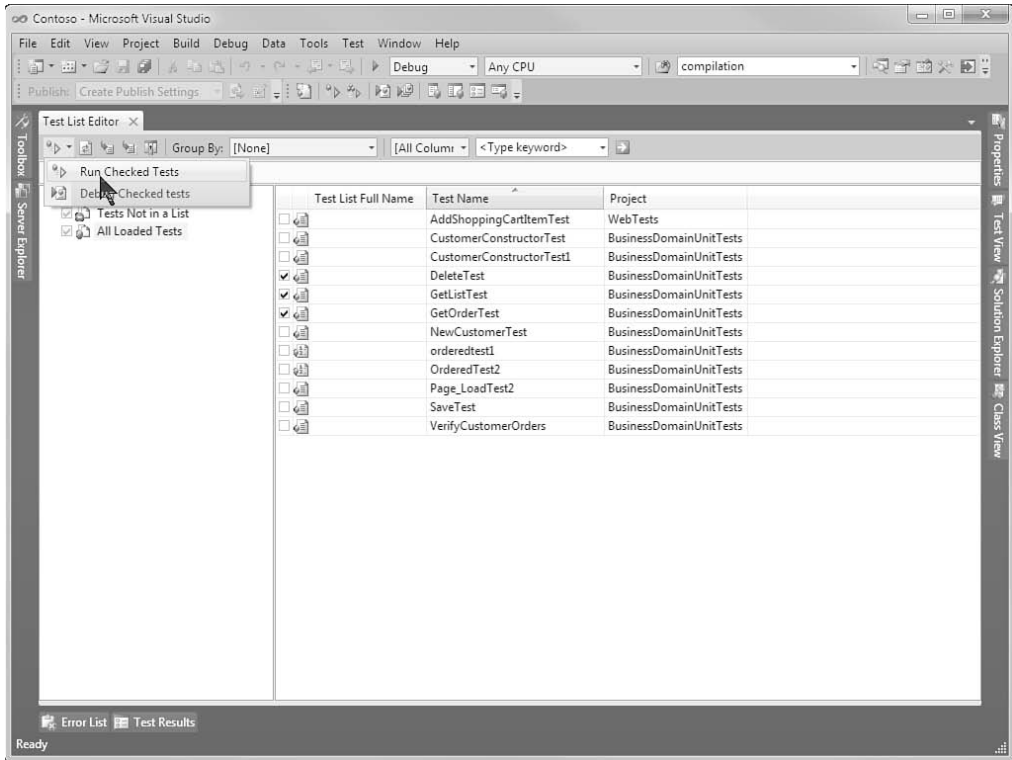
Rysunek 9.22. Można użyć opcji Group By, aby ułatwić sobie znalezienie testów jednostek



Okno Test List Editor

Visual Studio udostępnia też okno *Test List Editor*. Posiada ono podobne funkcje jak okno *Test View*. Umożliwia wyszukiwanie testów, grupowanie ich i uruchamianie zbiorów testów. Jednak w odróżnieniu od okna *Test View* pozwala także zapisywać zdefiniowane listy testów. W ten sposób można utworzyć i ponownie wykorzystać listy, kiedy zajdzie potrzeba uruchomienia zbioru testów.

Dostęp do okna *Test List Editor* można uzyskać z menu *Test (Test/Windows/Test List Editor)*. Pojawi się okno podobne do tego z rysunku 9.23. Po lewej stronie widnieją listy testów. Domyślnie dostępne są opcje *Lists of Tests*, *Tests Not in a List* i *All Loaded Tests*. Jeśli programista wybierze listę z testami (na przykład *All Loaded Tests*), może albo uruchomić całą listę, albo użyć pól wyboru widocznych po prawej stronie okna do wybrania z listy testów do przeprowadzenia.

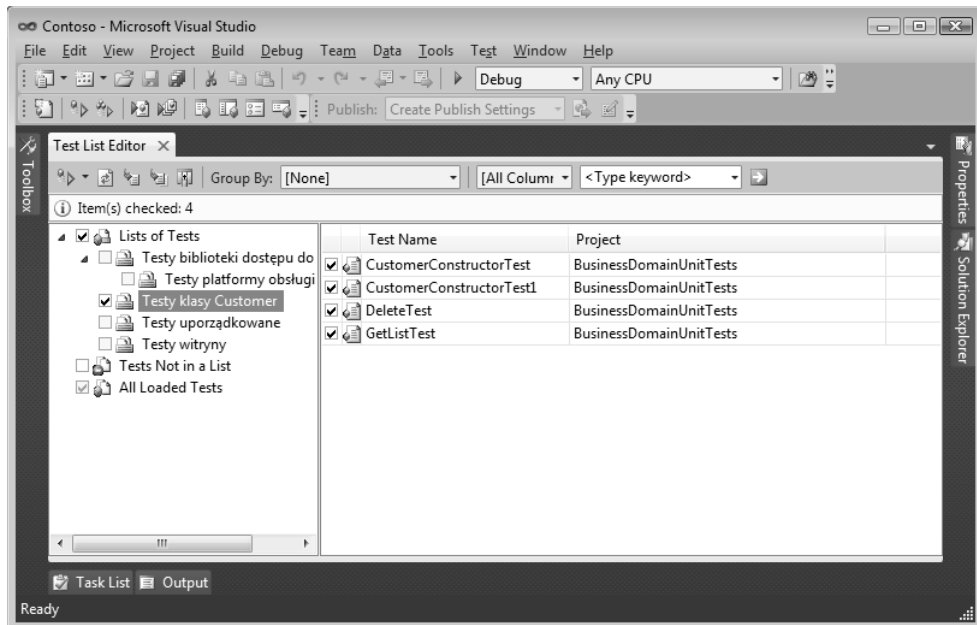
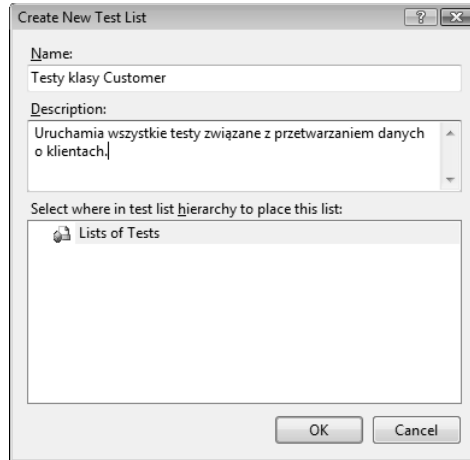


Rysunek 9.23. Można zaznaczyć testy w oknie Test List Editor i uruchomić je jako grupę

Okna *Test List Editor* można używać do tworzenia nowych list testów w celu zapisania testów regularnie wykonywanych w jednym zestawie. Aby utworzyć listę, należy kliknąć prawym przyciskiem myszy pozycję *Lists of Tests* w oknie *Test List Editor*, a następnie wybrać opcję *New Test List*. Pojawi się okno dialogowe *Create New Test List*. W oknie można podać nazwę listy testów i określić jej pozycję wśród innych list (zobacz rysunek 9.24).

Aby dodać test jednostek do nowej listy, można zastosować technikę przeciągania. Najpierw należy wybrać listę *All Loaded Tests* w celu wyświetlenia wszystkich wczytanych testów. Następnie trzeba zaznaczyć wszystkie testy, które mają znaleźć się na nowej liście. Wybrane testy wystarczy przeciągnąć na nazwę niestandardowej listy widoczną po lewej stronie okna *Test List Editor*. W ten sposób testy zostaną powiązane z nową listą. Na rysunku 9.25 pokazano kilka przykładowych niestandardowych list i testy z jednej z nich.

Rysunek 9.24.
Można definiować nowe listy testów do wielokrotnego użytku



Rysunek 9.25. Można definiować niestandardowe listy testów i dodawać do nich testy jednostek

Po utworzeniu listy testów do przeprowadzenia można uruchomić je za pomocą paska narzędzia okna *Test List Editor*. W tym celu należy wybrać opcję *Run Checked Tests* (zobacz rysunek 9.23).

Podsumowanie

W tym rozdziale pokazaliśmy, jak używać platformy testów jednostek do definiowania projektów testów i jak za pomocą atrybutu `TestClass` tworzyć pliki klas testów. Każda metoda testowa powinna mieć atrybut `TestMethod`. Przy użyciu klasy atrybutu `DataSource` można tworzyć testy jednostek powiązane z danymi.

Visual Studio umożliwia generowanie testów jednostek na podstawie istniejącego kodu. Udostępnia też w narzędziu *Test View* okno dialogowe *Properties*, aby ułatwić generowanie kodu atrybutów testów.

Można tworzyć testy jednostek obsługiwane przez ASP.NET (na serwerze IIS lub serwerze rozwojowym środowiska Visual Studio). W testach jednostek ASP.NET wykorzystywany jest obiekt `TestContext` do uzyskania dostępu do informacji o środowisku ASP.NET, w tym o przetwarzanej stronie, sesji i zmiennych serwera.

Pisanie testów jednostek może prowadzić do zmniejszenia liczby problemów w kodzie produkcyjnym. Przygotowanie zestawu testów kodu ułatwia jego zrozumienie i zwiększa niezawodność. Ponadto programista może z większą pewnością wprowadzać zmiany, ponieważ potrafi określić, w jakim fragmencie kodu w wyniku modyfikacji pojawiły się usterki.

Skorowidz

!, 136
!=", 135
#End Region, 368
#endregion, 368
#region, 368
#Region, 368
\$CALLER, 504
\$FUNCTION, 504
&, 135, 136
&&, 136
.NET, 23, 175
.NET Framework 4, 74
.NET Framework Components, 831
//HACK, 394
//TODO, 394
@Page, 769
@Register, 804
@WebService, 1018, 1019
^, 136
|, 136
||, 136
+, 135
<%@ Page StylesheetTheme="" %>, 774
<%@ Page Theme="" %>, 773
<%@ Register %>, 804
<%= %>, 165
<>, 144
<asp:Label>, 782
<asp:WebPartZone>, 782
<Author>, 386
<Code>, 386, 388
<CodeSnippet>, 386
<CodeSnippets>, 386
<Declarations>, 386
<Default>, 386
<Description>, 386
<Function>, 386
<Header>, 386
<ID>, 386
<Literal>, 386
<Shortcut>, 386
<Snippet>, 386
<SnippetType>, 386

<SnippetTypes>, 386
<Title>, 386
<ToolTip>, 386
<ZoneTemplate>, 782
=, 135
==, 135

A

Abstract, 121
Access, 993
Access board section 508, 734
AccessDataSource, 800, 993
Accessibility Validation, 289
Accordion, 917
Action, 175, 745, 1154
ActionsPane.StackOrder, 1150
ActionsPaneControl1, 1150
Activate, 574
Active Template Library, 41
ActivePoint, 601
ActiveWindow.Object.GetItem, 620
Activity, 1068, 1077
Activity Designer, 1068, 1069
Activity Designer Library, 1066
Activity Library, 1065
ActivityXamlServices.Load(), 1080
Add ASP.NET Folder, 725, 771
Add Connection, 944
Add Controller, 814
Add Correlation Initializers, 1112
Add Database Reference, 970
Add New Data Source, 976
Add New Item, 56, 189, 340, 680, 728, 737, 738, 758, 765, 827, 1018
 niestandardowe szablony, 341
Add New Project, 809, 1017, 1043
Add New Stored Procedure, 956
Add New Test, 403, 433
Add New Trigger, 960
Add New User Control, 290
Add Reference, 86, 730, 731
Add Related Tables, 948

- Add Service Reference, 730, 1033, 1034, 1036
- Add Stored Procedure, 971
- Add Style Rule, 271, 758, 759, 760
- Add Task List Shortcut, 395
- Add User Control, 655
- Add View, 818
- Add Web Reference, 730, 1035
- Add-in, 51
- AddIn, 562
- Add-in Manager, 639, 653
- Add-in Wizard, 677
- AddNamedCommand2, 651
- AddToCollection<T>, 1087
- ADO.NET, 175
- ADO.NET Data Service, 728
- ADO.NET Entity Data Model, 69, 812, 1001
- adres URL, 807, 1016
- Advanced compile options, 207
- AggregateCatalog, 691
- AJAX, 53, 58, 904
 - kontrolki, 58
 - UpdatePanel, 58
 - UpdateProgress, 58
- AJAX Client Behavior, 728
- AJAX Client Library, 728
- AJAX Control Library, 728
- Ajax Control Toolkit, 915
 - kontrolki, 917
 - stosowanie kontroltek, 919
- Ajax Extensions, 905, 907
- AJAX Master Page, 727
- AJAX-enabled WCF Service, 728
- aktualizacja częściowa, 907
- aktualizacja danych w modelu EDM, 1007
- aktualizacja lokalnego systemu pomocy, 313
- aktualizacja schematu bazy danych, 968
- aktywne odnośniki, 365
- AlwaysVisibleControlExtender, 917
- analiza kodu, 470
- Anchor, 834, 835
- anchoring, 833
- And, 136
- AndAlso, 136
- animacja, 862
- aplikacje, 43
 - aplikacje .NET, 43
 - aplikacje AJAX, 58
 - aplikacje Azure, 1170
 - aplikacje biznesowe, 68, 71, 72
 - aplikacje ClickOnce, 535
 - aplikacje klienckie, 1064
 - aplikacje konsolowe, 242
 - aplikacje LINQ to SQL, 996
 - aplikacje MEF, 690
 - aplikacje nadrzędne, 689
 - aplikacje okresowo nawiązujące połączenie, 71
 - aplikacje oparte na usługach sieciowych
 - ASP.NET, 1015
 - aplikacje oparte na usługach WCF, 1041
 - aplikacje rozproszone, 63
 - aplikacje równoległe, 520
 - aplikacje SharePoint, 45, 52
 - aplikacje Web 2.0, 904
 - aplikacje Web Form, 808, 809
 - aplikacje wielowątkowe, 176, 515
 - aplikacje WinForm, 44
 - aplikacje XAML dla przeglądarek, 48, 922
- aplikacje ASP.NET, 44, 54, 63, 714
- definiowanie sposobu kompilacji, 731
- diagnostyka, 736
- docelowa wersja platformy .NET, 733
- dodawanie referencji, 730
- dodawanie stron internetowych, 737
- dostępność, 733
- interfejs użytkownika, 746
- katalogi, 726
- kompilacja, 731
- kompozycje, 770
- opcje konfiguracyjne, 729
- pliki, 727
- położenie kontroltek, 747
- prekompilacja, 736
- projekt, 715
- projektowanie interfejsu użytkownika, 746
- referencje, 729
- serwer, 735
- Silverlight, 736
- skórki, 770
- strony wzorcowe, 765
- układ strony, 747
- wdrażanie, 549
- web.config, 55
- właściwości, 729
- zachowanie podczas ładowania strony, 735
- zgodność ze standardem dostępności stron
 - HTML, 735
- aplikacje ASP.NET MVC, 805, 917
 - AcceptVerbs, 816
 - ActionResult, 815
 - adres URL, 807

- Application_Start, 819
- Content, 812
- Controllers, 811
- Create, 815
- dane, 805, 806
- definiowanie przekierowań adresów URL
 - do kontrolera, 819
- dodawanie mechanizmów, 812
- dodawanie rekordów do modelu, 815
- kodu interfejsu użytkownika, 806
- kontroler, 806, 807
- model, 805, 806
- model przetwarzania żądań, 807
- Models, 811
- projekt, 809
- przekierowanie adresów URL, 819
- RegisterRoutes, 819
- Scripts, 812
- struktura katalogów, 810
- szablon aplikacji, 805
- testowanie, 809
- tworzenie kontrolera, 814
- tworzenie modelu, 812
- tworzenie projektów, 809
- tworzenie widoku, 817
- UrlRoutingModule, 807
- uruchamianie aplikacji, 820
- View, 815
- ViewResult, 815
- Views, 811
- widok, 805, 806, 807
- zalety platformy ASP.NET MVC, 808
- żądania HTTP, 807
- aplikacje do zarządzania procesem, 1062, 1096
 - baza danych, 1099
 - biblioteka dostępu do danych, 1100
 - dodawanie usługi do utrwalania procesu, 1116
 - dostęp do danych, 1108
 - komunikacja z pamięcią trwałą, 1118
 - pamięć trwała, 1117
 - projekt, 1097
 - projektowanie procesu, 1107, 1110
 - tworzenie, 1096
 - tworzenie aplikacji klienckiej, 1119
 - tworzenie biblioteki czynności, 1105
 - tworzenie niestandardowej czynności Code Activity, 1106
 - tworzenie niestandardowej czynności złożonej, 1105
 - tworzenie pamięci trwałej, 1117
 - tworzenie projektów, 1098
 - uruchamianie aplikacji do obsługi procesu, 1124
 - usługa, 1104
 - utrwalanie procesów, 1116
- aplikacje działające w chmurze, 64, 1158
 - instalacja usługi w chmurze, 66
 - konfiguracja aplikacji o roli Web, 1186
 - projekt, 64, 1175
 - przenoszenie do środowiska produkcyjnego, 1204
 - przygotowanie aplikacji do publikacji, 1196
 - publikowanie aplikacji, 66, 1199
 - rozwijanie aplikacji, 1175
 - tworzenie, 1160, 1170
 - tworzenie interfejsu o roli Web, 1180
 - tworzenie modelu danych opartego na tabeli, 1175
 - uruchamianie aplikacji, 64
 - wdrażanie, 1188, 1199
 - wdrażanie w środowisku testowym, 1200
- aplikacje Office, 1135
 - projekt, 1135
- aplikacje sieciowe, 44, 54, 282, 715
 - diagnozowanie, 470
 - projekt, 715
 - wiązanie danych, 989
- aplikacje Silverlight, 44, 59, 736, 934
 - interfejs użytkownika, 935
 - kontrolki, 936
 - MainPage.xaml, 935
 - pokaz slajdów, 936
 - projekt, 59, 934
 - strona XAML, 935
 - tworzenie, 60, 934, 935
 - wybór rodzaju klienta, 61
- aplikacje Windows, 46, 271
 - dodawanie kontrolki do formularza, 273
 - dostosowywanie wyglądu formularza, 273
 - formularze, 273
 - kontrolki, 273
 - pisanie kodu, 277
 - projekt, 46, 272, 826
 - rozmieszczanie kontrolki, 275
 - układ formularza, 46
 - układ tabelaryczny, 275
 - zdarzenia, 47
- aplikacje WPF, 44, 865, 904, 922
 - aplikacje przeznaczone do uruchamiania w przeglądarkach, 922
 - DataTemplate, 893
 - dodawanie kontrolki, 282
 - formularze, 281
 - górne menu aplikacji, 887

- aplikacje WPF
 - komponenty formularzy Windows, 861
 - lista obrazów, 886
 - ListBox, 886
 - manipulowanie obrazami, 895
 - metody do obsługi zdarzeń związanych, 894
 - nawigowanie po znacznikach, 371
 - pliki XAML, 280
 - projekt, 280, 866
 - przeglądarka obrazów, 888
 - przyciski, 894
 - rozmiar tabeli, 889
 - tworzenie, 871, 884
 - tworzenie układu, 885
 - układ, 871, 885
 - wiązanie danych, 986
 - wiązanie rysunków, 893
 - zapisywanie obrazów, 891
- aplikacje XBAP, 922
 - app.xaml, 924
 - bezpieczeństwo, 926
 - ClickOnce, 929
 - diagnozowanie, 925
 - formularze, 924
 - instalacja, 929
 - nawigowanie po witrynie, 928
 - PageX.xaml, 924
 - projekt, 923
 - strony, 924
 - tworzenie, 922
 - typy MIME, 931
 - uruchamianie, 925
 - WindowX.xaml, 924
 - zarządzanie zabezpieczeniami, 928
- app.config, 1103
- App_Browsers, 727
- App_Code, 428, 717, 726, 1018
- App_Data, 716, 717, 726
- App_GlobalResources, 726
- App_LocalResources, 726
- App_Themes, 727, 771
- App_WebReferences, 726
- AppFabric, 1158, 1160, 1164
- Application, 863, 1017
- application fabric, 1158
- Application Type, 206
- Application.EnableVisualStyles, 840, 841
- Application_Start, 819
- Apply Style, 763
- Apply Styles, 287, 763
- architektura ASP.NET MVC, 806
- architektura MVC, 714
- architektura platformy MEF, 689
- architektura platformy WPF, 861
- AreEqual, 420
- AreNotEqual, 420
- AreNotSame, 420
- AreSame, 420
- argumenty procesu, 1070, 1071
- arkusze stylów, 756, 758
- arkusze stylów XSLT, 264, 269
- Array, 142
- ArrayList, 143
- asercje, 404, 420
 - AreEqual, 420
 - AreNotEqual, 420
 - AreNotSame, 420
 - AreSame, 420
 - CollectionAssert, 421
 - Fail, 420
 - Inconclusive, 420
 - IsFalse, 420
 - IsInstanceOfType, 420
 - IsNotInstanceOfType, 420
 - IsNotNull, 420
 - IsNull, 420
 - IsTrue, 420
 - sprawdzanie kolekcji obiektów, 421
 - sprawdzanie łańcuchów znaków, 422
 - stosowanie danych testowych, 426
 - StringAssert, 422
- asocjacja, 297
- ASP, 165
- ASP.NET, 44, 52, 54, 177, 282, 714, 922
 - cykl życia strony, 741
 - członkostwo, 795
 - interfejs konfigurowany przez użytkownika, 776
 - interfejs użytkownika, 746
 - katalogi, 725, 726
 - kompozycje, 770
 - kontrolki, 788
 - kontrolki danych, 799
 - kontrolki logowania, 795
 - kontrolki nawigacyjne witryny, 798
 - kontrolki użytkownika, 801
 - kontrolki walidacyjne, 793, 794
 - kontrolki Web Part, 776
 - logowanie, 795
 - mapa witryny, 798
 - model zdarzeń, 741

- MVC, 56, 57
- personalizacja interfejsów użytkownika, 776
- pliki, 727
- procedury obsługi zdarzeń, 744
- skórki, 770
- strony internetowe, 737
- strony wzorcowe, 765
- style, 754
- testy jednostek, 428
- uwierzytelnianie użytkowników, 796
- Web Part, 776
- wykresy, 800, 801
- zdarzenia, 741
- ASP.NET Ajax, 904
 - aktualizacja częściowa, 907
 - ASP.NET MVC, 917
 - kontrolki, 905
 - odświeżanie częściowe, 912
 - ScriptManager, 906
 - ScriptManagerProxy, 906
 - stronicowanie, 912
 - Timer, 906
 - UpdatePanel, 906, 908
 - UpdateProgress, 906, 913
 - wyświetlanie informacji o postępie pracy serwera, 913
 - wyświetlanie powiadomień, 913
- ASP.NET Ajax Control Toolkit, 916
- ASP.NET Ajax Library, 915
 - kontrolki, 915
- ASP.NET Development Server, 719
- ASP.NET Empty Web Site, 1017
- ASP.NET Membership, 795
- ASP.NET MVC, 805
 - ASP.NET Ajax, 917
- ASP.NET MVC 2 Web Application, 809
- ASP.NET MVC 2 Web Role, 1167
- ASP.NET MVC Web Application, 57
- ASP.NET Non-Visual Controls, 779
- ASP.NET Reports Web Site, 717
- ASP.NET Web Application, 54
- ASP.NET Web Role, 64, 1167, 1170, 1175
- ASP.NET Web Service, 716
- ASP.NET Web Site, 283, 716
- ASPNETDB.MDF, 797
- AspNetDevelopmentServerHost, 428
- Assembly Name, 206
- AssemblyCatalog, 691
- AssemblyCleanup, 415
- AssemblyInitialize, 415
- Assert, 420, 421
 - AreEqual, 404, 420, 421
 - AreNotEqual, 421
 - Inconclusive, 405
- Assign, 1085
- AssociatedUpdatePanelID, 913
- Asynchronous JavaScript and XML, 53, 904
- ATL, 41
- atrybuty, 145
- atrybuty deklaratywne, 145
- atrybuty testów jednostek, 431
- Attach Style Sheet, 761, 762
- Attach to Process, 483, 485, 514, 515
- Authenticode, 349
- AutoCompleteExtender, 917
- AutoGenerateDeleteButton, 992
- AutoGenerateEditButton, 992
- automatyczne formatowanie kodu, 101
- automatyczne generowanie kodu do obsługi właściwości, 166
- automatyzacja, 560
 - kategorie automatyzacji, 563
- automatyzacja procesu kompilacji, 37
- automatyzacja procesu tworzenia testów jednostek, 402
- automatyzacja testowania jednostek, 398
- Autos, 485, 506
- AutoScaleDimensions, 835
- AutoScaleMode, 835, 836
- Azure, 65, 66, 1158
 - ADO.NET, 1177
 - aplikacje, 1170
 - aplikacje o roli Web, 1164, 1168
 - aplikacje o roli Worker, 1164
 - AppFabric, 1158, 1160, 1164
 - ASP.NET, 1163
 - DevFabric, 1159, 1161, 1170
 - hosting, 1159
 - instalacja pakietu SDK, 1160
 - interfejs o roli Web, 1180
 - klasy, 1178
 - kolejki, 1168
 - konfiguracja aplikacji o roli Web, 1186
 - konfiguracja środowiska programistycznego, 1160
 - obiekt źródła danych, 1183
 - obiekty blob, 1168
 - platforma programowania, 1159
 - projekt, 1165, 1167
 - przechowywanie danych, 1159, 1167
 - przygotowanie aplikacji do publikacji, 1196
 - publikowanie aplikacji, 1199

Azure

- relacyjna baza danych, 1169
- role, 1164
- rozwijanie aplikacji, 1174, 1175
- serwer IIS, 1163
- SQL Azure, 1169
- subskrypcja usług, 1188
- tabele, 1168
- tworzenie interfejsu o roli Web, 1180
- tworzenie konta do przechowywania danych, 1190
- tworzenie konta usług hosted service, 1194
- tworzenie modelu danych opartego na tabeli, 1175
- tworzenie projektów, 1175
- wdrażanie aplikacji, 1174, 1188
- zakładanie konta, 1188
- zalety stosowania, 1158

Azure SDK, 1160

B

BackColor, 828, 840

base, 122

Basic Unit Test, 401

baza danych, 67, 942, 1099

- aktualizowanie schematu, 968
- budowanie związków tabel, 948
- definiowanie tabel, 945
- diagram, 947
- funkcje definiowane przez użytkownika, 961
- indeksy, 946
- klucze obce, 946
- kompilowanie, 968
- LINQ, 994
- łańcuch połączenia, 973
- obiekty bazy danych, 970
- odwzorowanie obiektowo-relacyjne, 993
- ograniczenia, 946
- procedury składowane, 956
- projekt, 961
- Schema View, 967
- skrypty SQL, 962
- SQL, 951
- SQL Server, 942
- tabele, 942
- tworzenie, 943
- tworzenie projektu, 962
- wdrażanie, 968
- wiązanie danych, 974
- widoki, 955
- wyzwalacze, 960

związki jeden do jednego, 950

związki tabel, 948

związki wiele do wielu, 950

związki zwrotne, 950

bezpieczeństwo aplikacji XBAP, 926

bezpośrednie korzystanie z elementów XML, 164

BI, 1169

biblioteka dostępu do danych, 1100

biblioteka MSDN, 314

BigInteger, 42

Bin, 726

Binding, 881

Bindings, 607

BitmapSource, 891, 893

Blank Solution, 184, 1098

błędy, 475

bogate aplikacje internetowe, 904

bogate interakcje oparte na przeglądawkach
w systemie Windows, 921

bogaty interfejs użytkownika, 61

Bookmark, 1149

Bookmarks, 247

Bookmarks Window, 247

bool, 130

Boolean, 130

Bottom, 833

break, 137, 138, 140

Break All, 485, 491

Break at Function, 483

Breakpoint, 478, 495, 562

Breakpoint Condition, 500, 501

Breakpoint Hit Count, 503

Breakpoint2, 562

Breakpoints, 480, 482, 484, 496

- dodawanie etykiet punktów przerwania, 499

- dostęp do punktu przerwania, 498

- Go To Source Code, 498

- pasek narzędzi, 496, 497

- przerywanie działania na podstawie warunków, 500

- ustawianie warunków punktu przerwania, 500

- zarządzanie etykietami punktów, 499

- zarządzanie punktami przerwania, 496, 498

Bring to Front, 752, 846

bubbling event, 883

budowanie luźno powiązanych aplikacji sieciowych, 56

budowanie stylu CSS, 759

budowanie związków tabel, 948

Build, 91

Build Configuration, 193

Build events, 207

BuildDependencies, 562
 BuildDependency, 562
 BuildEvents, 562, 609
 BuildingBlockGalleryContentControl, 1149
 Business Intelligence, 1169
 BusinessEntities, 1025, 1045, 1047
 Button, 274, 792, 863
 byte, 130
 Byte, 130

C

C Runtime Library, 41
 C#, 115
 CA, 349
 CAB, 540
 Cached, 1152
 CachedAttribute, 1151
 CachedDataHostItem, 1153
 CachedDataItem, 1153
 CAL, 38
 Calendar, 792
 CalendarExtender, 918, 919
 Call Hierarchy, 248
 Call Stack, 476, 485, 519
 CallBase(), 388
 CancellationScope, 1089
 Canvas, 696, 871
 CascadingDropDown, 918
 case, 137, 138
 Case, 137, 138
 Case Else, 138
 CatalogZone, 779
 Catch, 146, 147
 CDbf, 132
 CE, 71
 CenterParent, 828
 CenterScreen, 828
 Cert2spc.exe, 349
 Certificate Creation Tool, 349
 certyfikaty Authenticode, 349
 CGI Web Role, 1167
 ChangePassword, 796
 CharLeft, 600
 CharRight, 600
 Chart, 800, 801
 CheckBox, 792
 chmurki, 841
 chmury obliczeniowe, 63, 1158
 Choose Data Source, 909
 Choose Location, 719
 CInt, 132
 class, 115, 121, 122, 126
 Class, 115, 123
 Class Designer, 32, 94, 293
 asocjacja, 297
 definiowanie metod, 298
 definiowanie pól, 298
 definiowanie zdarzeń, 298
 dziedziczenie, 296
 Inheritance, 296, 297
 interfejs, 296
 refaktoryzacja, 444
 tworzenie diagramu klasy, 293
 tworzenie klas, 293
 View Class Diagram, 294
 wyświetlanie składowych, 294
 Class Details, 298, 299
 Class Name, 244
 Class View, 219, 448
 Class View New Folder, 220
 Class View Settings, 220
 filtrowanie, 220
 panel obiektów, 221
 panel składowych, 222
 pasek narzędzi, 220
 pasek wyszukiwania, 220
 przyciski paska narzędzi, 220
 View Class Diagram, 220
 ClassCleanup, 415, 418
 ClassInitialize, 415, 418
 ClassName(), 388
 Clear All DataTips, 483, 486
 Clear All DataTips Pinned to, 486
 ClearBookmarks, 594
 ClearCollection<T>, 1087
 ClickOnce, 208, 534, 929
 lokalizacja katalogu instalacji, 538
 lokalizacja katalogu publikacji, 538
 publikowanie projektów, 536
 Publish Wizard, 536, 538
 sposoby wdrażania aplikacji, 535
 tworzenie pakietu instalacyjnego, 536
 wdrażanie za pomocą płyty CD, 535
 wdrażanie za pomocą serwera WWW, 535
 właściwości publikacji pakietu, 537
 wykonywanie za pomocą serwera WWW, 535
 Client Access License, 38
 Close, 574
 cloud computing, 63

- Cloud Service, 64
- CLR, 40
- clrversion, 335
- CLS, 40
- CMMI, 37
- Code Activity, 1068, 1069, 1075
- Code Definition, 262, 263
- Code DOM, 682
- Code Snippets Manager, 389
- CodeActivity, 1063, 1075, 1077
- CodeActivityContext, 1076
- CodeClass, 569
- CodeElement, 569
- CodeElement.Kind, 569
- CodeElements, 569, 570
- CodeModel, 569
- Collapse to Definitions, 369
- CollapseAll, 621
- CollapsiblePanelExtender, 917
- Collection, 1086
- CollectionAssert, 421
- CollectionBase, 143, 144
- Column.Width, 889
- COM, 640
- COM Components, 831
- ComboBoxContentControl, 1149
- Command, 562, 605
 - Bindings, 607
 - składowe, 605
- Command Window, 578, 583
 - uruchamianie makra, 632
- CommandBar, 588, 589, 590, 591
 - składowe, 590
- CommandBarControl, 589
- CommandBars, 591
- Commands, 562, 605, 651
 - AddNamedCommand, 607
 - Raise, 606
- Commands2, 562, 651
- CommandWindow, 562, 578, 583
- Common Language Runtime, 40
- Common Language Specification, 40
- Common Type System, 40
- CompareValidator, 794
- CompensableActivity, 1087, 1088, 1089
- Compensate, 1089
- CompensationHandler, 1088
- Compile options, 207
- Compiler conditions, 207
- Complete Word, 375
- Component Class, 290
- Component Designer, 744
- component tray, 830
- CompositionContainer, 691
- Configuration, 562
- Configure Data Synchronization, 71
- Confirm, 1089
- ConfirmButtonExtender, 918, 921
- Connect, 640, 645, 659
- Console.ReadLine, 378
- Console.WriteLine, 242, 244
- const, 133
- Const, 133
- consume first, 377
- ContainerControl, 835
- ContentPlaceHolder, 766, 767, 768
- ContentResult, 808
- Context, 1017
- ContextParams, 678
- ContextToken, 231
- Continue, 485, 487, 493
- ContosoCSR, 539
- Control, 290, 830, 863
- Control Flow, 1083
- ControlTemplate, 879
- ControlToValidate, 793
- Convert, 133
- Copy, 94
- Copy to Output Directory, 352
- Copy Web Project, 216
- Copy Web Site Tool, 549, 554
 - połączenie z witryną, 555
 - Remote Site, 556
- CorrelationScope, 1090
- Create a New Binding, 1054
- Create a New Service Endpoint, 1051
- Create GUID, 661
- Create New SQL Server Database, 943
- Create New Test List, 436
- Create Schema, 265
- Create Transparent Windows Form, 382
- Create Unit Test Project, 809
- Create Unit Tests, 402, 403, 404, 429
- Create User Task, 396
- CreateEditPoint, 599
- CreateToolWindow2, 659, 660, 661
- CreateUserWizard, 796
- CRM, 1013
- CRT, 41

- CSS, 270, 757
 - Apply Styles, 763
 - arkusze stylów, 756
 - budowanie stylu, 759
 - definiowanie stylów, 757
 - dodawanie zasady stylu, 759
 - hierarchia stylów, 757
 - Manage Styles, 760
 - modyfikacja stylów, 764
 - narzędzia do tworzenia stylów, 756
 - stosowanie stylów, 763
 - style elementów, 755
 - style wewnętrzzwerszowe, 755
 - style z poziomu strony, 755
 - tworzenie arkusza stylów, 758
 - właściwości stylu, 764
 - zarządzanie stylami, 760
 - zasady stylów, 755
 - CSS Outline, 758
 - CSS Properties, 287, 764
 - CssClass, 755, 763
 - CTS, 40
 - Current Page, 761
 - CurrentSettings.vsssettings, 82
 - Custom Actions, 546
 - custom control, 853
 - Customer Feedback Options, 324
 - Customize, 95
 - Customize the Start Page, 84
 - CustomValidator, 794
 - Cut, 94
 - Cut Line, 192
 - cykl życia dodatków, 645
 - cykl życia formularza Windows, 829
 - cykl życia strony ASP.NET, 741, 742
 - czcionki, 110
 - części, 690, 695
 - członkostwo ASP.NET, 795
 - czynności procesów, 72, 1082
 - AddToCollection<T>, 1087
 - Assign, 1085
 - CancellationScope, 1089
 - ClearCollection<T>, 1087
 - CompensableActivity, 1089
 - Compensate, 1089
 - Confirm, 1089
 - CorrelationScope, 1090
 - Delay, 1085
 - DoWhile, 1084
 - ExistsInCollection<T>, 1087
 - Flowchart, 1095, 1114
 - FlowDecision, 1095
 - FlowSwitch<T>, 1095
 - ForEach, 1084
 - If, 1084
 - InitializeCorrelation, 1090
 - InvokeMethod, 1085
 - Parallel, 1084
 - ParallelForEach, 1084
 - Persist, 1085
 - Pick, 1084
 - Receive, 1090, 1115
 - ReceiveAndSendReply, 1090
 - RemoveFromCollection<T>, 1087
 - Rethrow, 1086
 - Send, 1090
 - SendAndReceiveReply, 1090
 - Sequence, 1084
 - Switch, 1084
 - TerminateWorkflow, 1085
 - Throw, 1086
 - TransactedReceiveScope, 1090
 - TransactionScope, 1089
 - TryCatch, 1086
 - While, 1084
 - WriteLine, 1085
- ## D
- DAC, 969
 - dane, 67
 - dane hierarchiczne, 849
 - dane tabelaryczne, 852
 - dane XML, 164, 165
 - Data, 92
 - Data Source Configuration Wizard, 976, 977
 - Data Sources, 978
 - Data Tier Applications, 969
 - Data/New Query, 952
 - Database, 961
 - Database Diagram Designer, 947, 949
 - Database Unit Test, 401
 - DataContext, 882
 - DataGridView, 852, 978, 980, 990
 - dostosowanie edycji komórek, 983
 - edycja komórek, 983
 - modyfikacja zapytania dla źródła danych, 985
 - typy komórek, 852
 - wiązanie danych, 980, 983
 - zmiana typu kolumny, 984
 - źródła danych, 852

- DataGridViewButtonColumn, 983
- DataGridViewColumn, 983
- DataGridViewComboBoxColumn, 983, 984
- DataList, 800, 808, 990
- DataPager, 800
- DataServiceContext, 1177
- DataSet, 161, 1151
- DataSet Designer, 981, 982, 984
- DataSource, 415, 423, 431
- DataTable, 161
- DataTemplate, 893
- DataTip, 508, 509
- DatePickerContentControl, 1149
- DateTimePicker, 980, 983
- Debug, 91, 470, 482
 - tryb diagnostyczny, 483
 - tryb spoczynku, 482
- DEBUG, 206
- Debug Location, 517
- Debug Source Files, 192
- Debug with Mixed, 528
- debuger, 207, 477, 481
 - aplikacje sieciowe, 489
 - Break All, 491
 - Breakpoints, 496
 - Continue, 493
 - Detach All, 493
 - dołączanie do działającego procesu, 477
 - kontynuowanie wykonywania kodu, 493
 - kończenie sesji diagnostycznej, 493
 - menu Debug w czasie sesji diagnostycznej, 484
 - menu Debug w stanie spoczynku, 482
 - obiektowy model automatyzacji, 608
 - opcje diagnozowania, 487
 - pasek narzędzi Debug, 486
 - podglądanie danych, 505
 - podglądanie zmiennych, 505
 - podpowiedzi DataTip, 508
 - przechodzenie przez kod, 491
 - punkt śledzenia, 503
 - rozpoczynanie sesji diagnostycznej, 489, 491
 - Run To Cursor, 490
 - sesja diagnostyczna, 483
 - Start Debugging, 491
 - Step Into, 489, 491
 - Step Out, 493
 - Step Over, 489, 491
 - Terminate All, 493
 - uruchamianie sesji diagnostycznych, 482
 - ustawianie punktów przerwania, 494
 - ustawianie punktów przerwania funkcji, 495
 - wartości zmiennych, 505
 - warunki wstrzymania wykonywania kodu, 494
 - wkraczanie w kod, 488, 489
 - zarządzanie warunkami wstrzymania działania programu, 496
 - zmień i kontynuuj, 511
- Debugger, 260, 562, 608
- Debugger.Break, 260
- Debugger2, 562
- Debugger3, 562
- Debugger4, 562
- Debugger5, 562
- DebuggerEvents, 562
- decimal, 130
- Decimal, 130
- default, 137
- Default namespace, 206
- Default.aspx, 716, 908, 931, 1170
- default.htm, 931
- default.html, 931
- definiowanie
 - aplikacje WinForm, 47
 - atrybuty arkuszy stylów, 271
 - atrybuty środowiska ASP.NET, 428
 - interfejsy, 124
 - klasy, 115
 - kolekcje, 155
 - metody, 117, 298
 - plik VSContent, 345, 346
 - pola, 298
 - procedury obsługi zdarzenia, 829
 - referencje sieciowe, 1033
 - relacje między klasami, 296
 - stałe, 133
 - struktury, 126
 - tabele, 945
 - układ strony Web Part, 780
 - właściwości, 298
 - zdarzenia, 148, 298
- deklaracja
 - przestrzenie nazw, 127
 - skórki, 772
 - typy anonimowe, 156
 - zmiennie, 130
- dekoracje, 693, 697
 - projekt, 702
- Delay, 1085
- Delay signing, 209
- delegaty, 147

- DELETE, 955
- Delete All Breakpoints, 483
- DeleteQuery, 992
- dependency injection, 689
- Deploy SQL, 552
- DeploymentItem, 416, 431
- deserializacja, 1153
- Design, 55
- design schema, 68
- Detach All, 485, 493
- DetailsView, 800, 989
- development fabric, 1159
- Development Fabric, 64, 65, 1173
- Development Storage Service, 1173
- Development Tools Environment, 561
- DevFabric, 1159, 1161, 1170
- diagnozowanie, 258, 468, 481
 - analiza kodu, 470
 - aplikacje sieciowe, 470
 - aplikacje XBAP, 925
 - Breakpoints, 480
 - Call Stack, 476
 - diagnozowanie błędów, 475
 - diagnozowanie innych procesów, 476
 - etapy, 469
 - Exception Assistant, 476
 - Exceptions, 474
 - Immediate, 476
 - kodowanie, 469
 - kompilacja, 469
 - kontynuowanie diagnozowania, 478
 - kroczenie przez kod, 480
 - Locals, 476
 - makra, 623
 - podglądanie zmiennych, 505
 - procedury składowane, 958
 - przegląd kodu, 470
 - przerywanie w momencie wystąpienia błędu, 474
 - rozwiązywanie problemów, 470
 - Run To Cursor, 472
 - samodzielne sprawdzanie, 470
 - scenariusz, 469
 - Show Next Statement, 478
 - skrypty działające po stronie klienta, 526
 - Start Without Debugging, 472
 - Step Into, 480
 - stos wywołań, 476
 - testy jednostek, 470
 - Thrown, 475
 - uruchamianie aplikacji w trybie diagnozowania, 472
 - ustawianie punktów przerwania, 478
 - wartości zmiennych, 476
 - włączanie diagnozowania stron internetowych, 470
 - zarządzanie zbiorem wyjątków, 474
 - zdalne diagnozowanie, 513
- diagnozowanie aplikacji równoległych, 520
 - Parallel Stacks, 521
 - Parallel Tasks, 525
- diagnozowanie aplikacji wielowątkowych, 515
 - Call Stack, 519
 - Debug Location, 517
 - oznaczanie wątków, 515
 - przerywanie działania kodu po wejściu do konkretnego wątku, 519
 - Show Threads in Source, 516
 - sprawdzanie poszczególnych wątków, 518
 - Threads, 517
 - wizualizacja wątków, 515
 - wykrywanie wątków, 515
 - zarządzanie diagnozowanymi procesami i wątkami, 517
- diagnozowanie informacji o awarii, 527
 - Cache Symbols in This Directory, 530
 - Debug with Mixed, 528, 530
 - diagnozowanie z wykorzystaniem pliku zrzutu, 528
 - pliki symboli, 529
 - pliki zrzutów, 527
 - Save Dump As, 527
 - Set Symbol Paths, 528
 - tworzenie zrzutów informacji diagnostycznych, 527
- diagnozowanie usług WCF, 514
 - dołączanie debugera do usług WCF, 515
 - wkraczanie w kod usługi WCF, 514
- diagram bazy danych, 947
- Diagram Designer, 950
- diagram klasy, 293
 - dodawanie elementów, 294
 - wyświetlanie składowych, 294
- DialogResult, 681
- Dim, 131
- direct event, 883
- DirectoryCatalog, 691
- DirectoryImageList, 896
- Disable All Breakpoints, 483, 486
- Disassembly, 485
- DISCO, 1016
- Disco.exe, 1030
- Discovery Document, 1016
- DispatcherObject, 863
- DisplayAfter, 913

- DivideByZeroException, 146
- do...while, 140
- Do...While, 140
- Dock, 835
- DockPanel, 871, 872
- Document, 51, 562, 592, 594
 - składowe, 592
- Document Outline, 232, 371, 372
 - Collapse All, 234
 - Expand All, 234
 - modyfikacja elementów, 233
 - Move Down in Container, 234
 - Move into Next Container, 234
 - Move Out of Current Container, 234
 - Move Up in Container, 234
 - pasek narzędzi, 234
 - Type Name Display Style, 234
- Document Type Definition, 264
- DocumentReader, 862
- Documents, 562, 592
- dodatki, 51, 636
 - Add-in Manager, 653
 - aplikacja nadrzędna, 638
 - COM, 640
 - Connect, 640, 645, 659
 - CreateToolWindow2, 659
 - cykl życia, 645
 - DTE.AddIns, 653
 - IDTCommandTarget, 645
 - IDTExtensibility2, 645, 646
 - IDTToolsOptionsPage, 662, 663
 - język, 638
 - kod wygenerowany przez kreator dodatków, 640
 - kreator dodatków, 637
 - menedżer dodatków, 653
 - obiektywny model automatyzacji, 653
 - obsługa zdarzeń, 645
 - okno About, 639
 - OnAddInsUpdate, 647
 - OnBeginShutdown, 647
 - OnConnection, 648
 - OnDisconnection, 650
 - OnStartupComplete, 651
 - opcje, 639
 - opisywanie dodatków, 639
 - paleta do wybierania kolorów, 654
 - parametry dodatku, 637
 - programowe zarządzanie dodatkami, 653
 - projekt, 637
 - przechwytywanie zdarzeń kontrolki użytkownika, 661
 - reagowanie na polecenia, 651
 - rejestrwanie strony Options, 665
 - sekwencja zdarzeń w czasie usuwania, 646
 - sekwencja zdarzeń w czasie wczytywania, 646
 - strona Options, 662
 - struktura dodatków, 645
 - tworzenie interfejsu użytkownika, 662
 - udostępnianie ustawień dodatku, 662
 - wyświetlanie kontrolki użytkownika, 660
 - wyświetlanie okna narzędzi, 660
 - zarządzanie dodatkami, 653
- dodatki dla pakietu Office, 1136
 - modyfikacja wstążki, 1137
- dodawanie
 - baza danych, 944
 - dane do pamięci podręcznej, 1151
 - elementy do diagramu klasy, 294
 - elementy do wstążki, 1139
 - fragmenty kodu do środowiska Visual Studio, 388
 - klawisze skrótu, 607
 - kontrolki, 273, 282, 284
 - kontrolki do strony internetowej, 738
 - kontrolki użytkownika do strony, 803
 - logika biznesowa do wygenerowanego kodu, 158
 - materiały do lokalnego systemu pomocy, 313
 - metody do istniejących klas, 157
 - projekt do rozwiązania, 194
 - referencje, 730
 - strony internetowe do witryny, 737
 - tekst, 599
 - zakładki, 247
 - zależności, 689
- dokowanie, 106, 108, 834
- dokumentacja środowiska Visual Studio, 314
- dokumenty, 51, 592
 - CSS, 270
 - DTD, 264
 - HTML, 288
 - obiektywny model automatyzacji, 592
 - WSDL, 1029
 - XDR, 264
 - XSD, 265
 - XSLT, 269
- dokumenty tekstowe, 593
 - dodawanie tekstu, 599
 - modyfikacja, 595, 600
 - punkt edycji, 599
 - wstawianie komentarzy, 601
 - zmiana pozycji obiektu EditPoint, 600

- dokumenty XML, 165, 264
 - edycja, 264
 - generowanie szablonów, 265
 - uruchamianie szablonu XSLT, 269
 - dołączanie debugera do usług WCF, 515
 - dopasowywanie nawiasów, 391
 - dopasowywanie strukturalne, 690
 - dopracowywanie poleceń SQL, 953
 - dostawcy obrazów dla kontroltek, 850
 - dostawcy rozszerzeń, 837
 - dostęp do danych, 67, 160, 909
 - dostęp do kodu projektu, 569
 - dostęp do okien, 572
 - dostęp do pamięci podręcznej danych, 1152
 - dostęp do paneli okna tekstu, 577
 - dostępność, 733
 - dostępność składowych, 118
 - dostosowywanie
 - czcionki, 110
 - edytory, 100
 - mechanizm IntelliSense, 392
 - odwzorowanie źródła danych, 980
 - paski narzędzi, 95
 - wstążka, 53
 - wygląd kontroltek WPF, 878
 - double, 130, 132, 133
 - Double, 130
 - DoWhile, 1083, 1084
 - DropDownList, 792
 - DropDownListContentControl, 1149
 - drukowanie kodu, 262
 - czcionki, 262
 - kolory, 262
 - Print, 262
 - DTD, 265
 - DTE, 561, 562, 564
 - ActiveWindow, 573
 - AddIns, 653
 - Commands, 605
 - Debugger, 608
 - Documents, 592
 - Events, 608
 - ExecuteCommand, 606
 - MainWindow, 572
 - ToolWindows, 573, 660
 - Windows, 573
 - DTE2, 561, 564
 - ActiveDocument, 564
 - ActiveSolutionProjects, 564
 - ActiveWindow, 564
 - CommandBars, 564
 - Commands, 564
 - Debugger, 564
 - Documents, 564
 - Events, 564
 - ExecuteCommand, 565
 - LaunchWizard, 565
 - MainWindow, 565
 - Quit, 565
 - Solution, 564
 - StatusBar, 565
 - ToolWindows, 565
 - WindowConfigurations, 565
 - DTEEvents, 628
 - OnMacrosRuntimeReset, 628
 - OnStartupComplete, 628
 - duck typing, 690
 - dynamic, 167
 - Dynamic Data Entities Web Site, 717
 - Dynamic Data Field, 727
 - Dynamic Data Linq to SQL, 717
 - Dynamic Data Linq to SQL Web Site, 717
 - DynamicObject, 169
 - TryCreateInstance, 169
 - TryGetMember, 169
 - TryInvokeMember, 169
 - TrySetMember, 169
 - dynamiczne aplikacje, 688
 - dynamiczne obiekty, 167
 - dynamiczne typy danych, 167
 - dziedziczenie, 121, 296
 - przesłanie operacji, 122
 - składowe wirtualne, 122
 - wywołanie metody z klasy bazowej, 122
 - dzielenie podzespółów na wiele plików, 164
- ## E
- ECMA, 41
 - ECMAScript, 905
 - Edit, 89
 - Edit and Continue, 511
 - Edit breakpoint labels, 499
 - Edit DataSet with Designer, 984
 - Editor Classifier, 695
 - Editor Margin, 696
 - Editor Text Adornment, 697
 - Editor Viewport Adornment, 698
 - EditorClassifier1, 695
 - EditorClassifier1Format, 695

- EditorMargin1, 696
- EditorZone, 779, 786
- EditPoint, 562, 596, 597, 599, 600, 601
 - InsertFromFile, 600
- EditPoint2, 597
 - składowe, 597
- EDM, 68, 69, 977, 1001
- edycja
 - arkusze stylów XSLT, 269
 - dokumenty XML, 264
 - model EDM, 1003
- edycje środowiska Visual Studio 2010, 24, 31
- edytor arkuszy CSS, 270
 - definiowanie atrybutów arkuszy stylów, 271
 - dodawanie zasad stylów, 271
- edytor File System, 542
- edytor formularzy, 830, 831
 - paleta komponentów, 831
- edytor HTML Source Editor, 285
 - opcje formatowania kodu, 286
 - tabele HTML, 286
- edytor manifestów, 352
- edytor Registry, 543
- edytor środowiska Visual Studio, 692
 - dekoracje, 693
 - formatowanie, 692
 - IntelliSense, 693
 - marginesy, 692
 - mechanizmy obsługi myszy, 693
 - mechanizmy obsługi upuszczania, 693
 - opcje, 693
 - rodzaje zawartości, 692
 - suwaki, 692
 - tagi, 693
 - typy klasyfikacyjne, 692
- edytor XML, 264
- edytory kodu, 98
 - Code Definition, 262
 - edytor języka C#, 98
 - edytor języka SQL, 956, 957
 - edytor języka Visual Basic, 99
 - inteligentne znaczniki, 373
- edytory tekstu, 98, 236
 - aktywne odnośniki, 365
 - diagnozowanie, 258
 - dostosowywanie, 100
 - drukowanie kodu, 262
 - elementy okna, 243
 - IntelliSense, 363, 375
 - kolorowanie składni, 365
 - konfiguracja punktów przerwania, 259
 - kontrola wykonywania kodu, 260
 - margines wyboru, 245
 - margines ze wskazówkami, 244
 - narzędzia pomocnicze, 364
 - nawigowanie po kodzie, 246
 - numerowanie wierszy, 246
 - obszar wirtualny, 237
 - Options, 238
 - otwieranie edytora, 241
 - panel kodu, 244
 - pisanie kodu, 240, 241
 - przeszukiwanie dokumentów, 249
 - schematy kodu, 367
 - śledzenie zmian, 364
 - ustawianie punktów przerwania, 258
 - wirtualne odstępki, 239
 - wskazówki dotyczące problemów, 364
 - wyszukiwanie przyrostowe, 257
 - zakładki, 247
 - zawijanie wierszy, 237, 239
 - zaznaczanie tekstu, 237
- EF, 1001
- egzemplarz klasy, 131
- eksplorator szablonów, 266
- eksportowanie
 - moduł makra, 618
 - szablony, 333
 - ustawienia, 82
- ElementHost, 1143
- ElementHost.Child, 1143
- elementy projektów, 203
- elementy rozwiązania, 188
- elementy XML, 164
- Else, 136
- Empty Web Site, 717
- EmptyResult, 808
- Enable Event Handling Code, 629
- Enable Virtual Space, 238
- Enable Windows Azure Tools, 1161
- EnableClientScript, 793
- Encapsulate Field, 441, 465
- End If, 136
- End Sub, 117
- Enforce Foreign Key Constraint, 950
- Entity, 68
- Entity Data Model, 68, 69
- Entity Data Model Designer, 1002

- Entity Framework, 1001
 - aktualizacja danych w modelu EDM, 1007
 - graficzny widok modelu pojęciowego, 1003
 - kierowanie zapytań do modelu EDM, 1006
 - konfiguracja modelu EDM, 1002
 - Mapping Details, 1005
 - Model Browser, 1004
 - model EDM, 1001
 - model fizyczny, 1004
 - model pojęciowy, 1003
 - modyfikacja modelu EDM, 1002
 - odwzorowanie między modelami, 1005
 - odwzorowanie obiektów na niestandardowe funkcje, 1005
 - okno projektowe modelu EDM, 1003
 - projekt, 1001
 - tabele, 1001
 - zapytania, 1007
 - zapytania LINQ, 1006
 - EntityDataSource, 800
 - EntityServices, 1025, 1045
 - enum, 120
 - Enum, 121
 - EnvDTE, 561
 - EnvDTE.IDTWizard, 677
 - EnvDTE100, 561
 - EnvDTE80, 561
 - EnvDTE90, 561
 - EnvDTE90a, 561
 - EnvironmentEvents, 623, 625, 627
 - Error Handling, 1086
 - Error List, 623
 - etapy diagnozowania, 469
 - etykiety, 274
 - etykiety punktów przerwania, 499
 - event, 147
 - EventHandler, 147, 149
 - Events, 562, 608
 - Excel, 1147
 - Excel 2007 Workbook, 1136, 1147
 - Excel 2010, 51
 - Exception Assistant, 476
 - ExceptionGroups, 562
 - Exceptions, 474, 483, 486
 - Execute, 681, 1077
 - ExecuteCommand, 606
 - ExecuteOption, 652
 - Existing Style Sheet, 761
 - ExistsInCollection<T>, 1087
 - ExpandAll, 621
 - ExpandCollapseNodes, 621
 - ExpectedException, 416, 422
 - Explicit naming, 373
 - Export DataTips, 483, 486
 - Export File, 618
 - Expression Blend, 40, 48, 61, 62
 - Expression Design, 40
 - Expression Encoder, 40
 - Expression Media, 40
 - Expression Studio, 40
 - Expression Web, 40
 - ext_cm_AfterStartup, 648
 - ext_cm_CommandLine, 648
 - ext_cm_External, 648
 - ext_cm_Solution, 648
 - ext_cm_Startup, 648
 - ext_cm_UISetup, 648
 - ext_ConnectMode, 648
 - ext_DisconnectMode, 650
 - ext_dm_HostShutdown, 650
 - ext_dm_SolutionClosed, 650
 - ext_dm_UISetupComplete, 650
 - ext_dm_UserClosed, 650
 - extender providers, 837
 - Extender Wizard, 920
 - Extensible Application Markup Language, 53
 - Extensible Markup Language, 1016
 - Extension Manager, 341, 699
 - tryb działania, 700
 - extension method, 157
 - ExtensionAttribute, 157
 - Extract Interface, 441, 459
 - wyodrębnianie interfejsów, 459
 - Extract Method, 441, 451
- ## F
- F#, 114
 - Fail, 420
 - false, 138
 - FastCGI, 1167
 - File, 85, 89
 - File Breakpoint, 501, 502
 - File System, 542
 - File Transfer Protocol, 722
 - File Types, 544
 - FileResult, 808
 - FileUpload, 792
 - FilteredTextBoxExtender, 918
 - filtry ISAPI, 721

- filtry punktów przerwania, 502
- Finally, 146, 147
- Find, 562
- Find and Replace, 249, 253
 - Choose Search Folders, 253
 - Find In Files, 253
 - Find Results, 254
 - Find Symbol, 256
 - Find Symbol Results, 256
 - podmianianie tekstu w plikach, 253, 255
 - Replace in Files, 255
 - tworzenie grup przeszukiwanych katalogów, 253
 - wyniki wyszukiwania, 254
 - wyszukiwanie symboli, 256
 - wyszukiwanie tekstu w plikach, 253
- Find Results, 254
- Find Symbol, 249
- FK, 950
- float, 130, 132
- Floating, 108
- Flowchart, 73, 1095, 1113, 1114
- FlowDecision, 1095
- FlowDocumentReader, 862
- FlowLayoutPanel, 836, 837
- FlowSwitch<T>, 1095
- FolderBrowserDialog, 895
- Font, 840
- for, 139
- For...Each, 139
- For...Next, 139
- fora społeczności MSDN, 315
 - dyskusja, 319
 - ikony wątków, 322
 - nawigowanie po forach i wątkach, 320
 - otwieranie nowego wątku, 318
 - powiadomienia dotyczące własnych wątków, 321
 - Windows Live, 318
 - wyszukiwanie odpowiedzi na pytania, 316
- foreach, 139
- ForEach, 1083, 1084
- ForeColor, 828, 840
- foreign key, 950
- Foreign Key Relationship, 948, 950
- Form, 823, 835
 - BackgroundImage, 828
- formalny opis usługi sieciowej, 1029
- Format, 92
- formatowanie kodu, 101
- Forms Designer, 830
- FormStartPosition, 827
- formularz startowy, 826
- formularze sieciowe, 282, 737, 808, 809
 - Apply Styles, 287
 - dodawanie kontroltek, 284
 - Manage Styles, 287
 - modyfikacja znaczników, 285
 - projekt, 283
 - projektowanie aplikacji, 283
 - rozmieszczanie kontroltek, 284
 - sposób wyświetlania, 741
 - sprawdzanie poprawności kodu, 288
 - tworzenie, 285
 - wygląd strony w przeglądarce, 288
 - zarządzanie stylami, 287
 - zgodność ze standardami, 289
- formularze Windows, 44, 46, 62, 271, 274, 822
 - BackColor, 828
 - cykl życia, 829
 - dodawanie kontroltek, 273, 274, 830
 - dostosowywanie wyglądu formularza, 273
 - dziedziczenie wyglądu innego formularza, 827
 - ForeColor, 828
 - Forms Designer, 830
 - formularz startowy, 826
 - głębokość kontroltek, 846
 - klasa formularza, 826
 - kolejność przechodzenia po kontrolkach, 840
 - kolory, 828
 - komponenty, 830
 - kontenery, 836
 - kontrola automatycznego skalowania, 835
 - kontrolki, 830
 - kontrolki kontenerowe, 835
 - kultura, 823
 - lokalizacja, 823
 - menu, 842
 - okno projektowe, 373
 - Opacity, 828
 - pasek narzędzi, 844
 - pasek statusu, 847
 - pisanie kodu, 277
 - położenie początkowe, 827
 - pozycjonowanie kontroltek, 831
 - projekt, 826
 - projektowanie, 822
 - przezroczystość, 828
 - rozmieszczanie kontroltek, 275
 - standardy UI, 824
 - StartPosition, 827

- tworzenie, 826
- tworzenie procedury obsługi zdarzeń, 829
- układ kontroltek, 831
- użytkownik końcowy, 823
- wiązanie danych, 982
- właściwości, 827
- wygląd, 828
- wyświetlanie danych, 849
- zdarzenia, 827, 828, 829
- formularze WinForm, 44
- formularze WPF, 48, 49, 102, 281, 310
 - dodawanie kontroltek, 282
 - projektowanie, 62
- FormView, 800, 989
- FPSE, 553
- fragmenty kodu, 380
 - Code Snippets Manager, 389
 - dodawanie fragmentów kodu do środowiska Visual Studio, 388
 - format XML fragmentów kodu, 385
 - funkcje, 388
 - okno narzędzi, 390
 - otaczanie kodu, 383
 - Surround With, 383
 - tworzenie własnych fragmentów kodu, 385
 - udostępnianie przez sieć, 390
 - węzły XML, 386
 - wstawianie, 381
- FrameworkElement, 882
- FrontPage Server Extensions, 553
- FTP, 722
- Func, 175
- Function, 117, 134, 162
- funkcje anonimowe, 162
- funkcje definiowane przez użytkownika, 961
- funkcje pakietu Office, 1131
- funkcje wewnętrzzwerszowe, 163

G

- GAC, 729
- galeria środowiska Visual Studio, 331
- GDI+, 176
- Generate From Usage, 373, 374
- GenerateSwitchCases(), 388
- Generic Handler, 727
- generowanie
 - kod, 158
 - schematy kodu, 367
 - szablony XML, 265

- szkielety metod, 457
- testy jednostek ASP.NET, 429
- testy na podstawie istniejącego kodu, 402
- związane kontrolki Windows Forms, 976
- GetClassificationSpans, 695
- GetElement, 169
- Global.asax, 716, 744, 819
- głębokość kontroltek na formularzu, 846
- główną przestrzeń nazw, 127
- graficzne okna projektowe, 102
- graficzne wyróżniki tekstu, 697
- Grid, 702, 871, 874, 885, 888
 - okna dialogowe, 875
- Grid.ColumnDefinitions, 874
- GridSize, 275
- GridView, 801, 808, 911, 989, 990
 - AutoGenerateDeleteButton, 992
 - AutoGenerateEditButton, 992
 - uaktualnianie danych, 992
 - źródło danych, 990
- GUID, 188, 335, 661

H

- HACK, 394
- Hashtable, 143, 170
- Help, 93, 312
- Help Library Manager, 312
- hermetyzacja pól, 465
- Hit Count, 502
- HorizontalAlignment, 886
- host, 1042, 1049
- host procesu, 1064
- hosting, 1158
- hosting usług WCF, 1058
 - hosting samodzielny, 1058
 - IIS, 1059
 - usługi systemu Windows, 1059
 - WAS, 1059
- HostType, 416, 428
- HoverMenuExtender, 918
- HTML Designer, 751
- HTML Page, 728
- HTML Source Editor, 285
- HTMLWindow, 562
- HTMLWindow3, 562
- HTTP, 74, 75, 1016, 1031
- Hypertext Transfer Protocol, 1016

I

- IClassifier, 695
- IComponent, 289
- Icon, 206
- Icons Gallery Add-In, 1140
- ICustomerProfile, 1045
- IDE, 88, 969
- IDE Macros, 616
- identyfikator GUID, 661
- identyfikator skórki, 775
- IDictionary, 1074, 1081
- IDTCommandTarget, 645
- IDTCommandTarget.Exec, 651
- IDTExtensibility2, 636, 645, 646
 - OnAddInsUpdate, 646, 647
 - OnBeginShutdown, 646, 647
 - OnConnection, 646, 648
 - OnDisconnection, 646, 650
 - OnStartupComplete, 646, 651
- IDTTToolsOptionsPage, 662, 683
 - składowe, 663
- IDTWizard, 677, 679, 680
- IEnumerable, 161
- IEnumerable<T>, 174
- IEnumerator<T>, 174
- if, 136
- If...Then...Else, 136
- Ignore, 416
- IIS, 718, 720, 1059
- ikona podzespołu, 206
- ikony punktów przzerwania, 496
- Image, 705, 792, 888
- ImageList, 47, 850
- imageMso, 1140
- Images Collection Editor, 851
- Immediate, 476, 482, 485
- implementacja interfejsu, 125, 297
 - inteligentne znaczniki, 374
- Implements, 125
- Implicit naming, 373
- implicit typing, 151
- Import and Export Settings, 80, 81, 82, 303
 - wybór ustawień, 81
- Import DataTips, 483, 486
- importowanie przestrzeni nazw, 128
- importowanie ustawień, 82
- imports, 128
- in, 174
- Inconclusive, 420
- Incremental Search, 257
- indeksy, 946
- Index Columns, 946
- Indexes/Keys, 946
- informacje o parametrach, 379
- Inherited Form, 827
- Inherits, 121
- inicjator obiektów, 153
- inicjowanie kolekcji, 155
- inicjowanie obiektów zdarzeń, 628
- Init, 743
- InitComplete, 743
- InitializeCorrelation, 1090
- Inline Function, 961
- INSERT, 954, 955
- Insert Breakpoint, 478, 495
- Insert Snippet, 381
- Insert Tracepoint, 504
- instalacja
 - aplikacje XBAP, 929
 - ClickOnce, 208
 - pakiet SDK, 349
 - szablony Visual Studio, 331, 339
 - udostępniane zasoby, 330
 - usługi w chmurze technologii Azure, 66
 - usługi WCF, 1058
- instalacja środowiska Visual Studio 2010, 78
 - wybór języka, 78
- instalator systemu Windows, 535, 539
 - Custom Actions, 546
 - dołączanie plików do pakietu instalacyjnego, 541
 - dostosowanie wyglądu kreatora instalacji, 545
 - File System, 542
 - File Types, 544
 - Launch Conditions, 546
 - powiązania typów plików, 544
 - projekt, 540
 - publikowanie projektów, 539
 - Registry, 543
 - Setup Wizard, 540
 - User Interface, 545
 - warunki wstępne, 546
 - wpisy w rejestrze, 543
 - wyjatkowe wymagania związane z instalacją, 546
- instrukcje SLQ, 909
- instrukcje Using, 380
- int, 130
- Integer, 130
- inteligentne operacje, 372
- inteligentne znaczniki, 372, 373
 - Generate From Usage, 373, 374
- generowanie, 373

- implementacja interfejsu, 374
 - refaktoryzacja, 442
 - IntelliSense, 285, 299, 363, 372, 375, 693
 - Add New Line on Enter at End of Fully Typed Word, 393
 - Committed by Pressing the Space Bar, 393
 - Committed by Typing the Following Characters, 393
 - Complete Word, 375
 - dopasowywanie nawiasów, 391
 - dostosowywanie mechanizmu, 392
 - fragmenty kodu, 380
 - kod szablonowy, 380
 - List Members, 378
 - lista uzupełniania, 375
 - okno z informacjami o parametrach, 379
 - okno z informacjami podręcznymi, 377
 - okno z listą składowych, 378
 - otaczające fragmenty kodu, 383
 - otaczanie kodu fragmentami, 383
 - Parameter Info, 379
 - Place Code Snippets in Completion Lists, 393
 - Place Keywords in Completion Lists, 393
 - porządkowanie instrukcji Using, 380
 - Pre-select Most Recently Used Members, 393
 - Quick Info, 377
 - Remove and Sort, 380
 - Remove Unused Usings, 380
 - Show Completion List After a Character Is Typed, 393
 - Sort Usings, 380
 - tryb pierwszeństwa tekstu, 377
 - tryb standardowy, 377
 - uzupełnianie słów, 375
 - interakcja z oknami, 573
 - interface, 125
 - Interface, 125
 - interfejs użytkownika, 746, 822
 - kultura, 823
 - lokalizacja, 823
 - niepisane standardy, 824
 - planowanie, 825
 - początkowy plan układu, 825
 - projektowanie, 822
 - standardy, 824
 - użytkownik końcowy, 823
 - wstępny projekt, 825
 - interfejsy, 124, 296
 - definiowanie, 125
 - implementacja, 125, 297
 - refaktoryzacja, 459
 - Internal, 119
 - InternalsVisibleToAttribute, 164
 - InvokeMethod, 1085
 - IronPython, 171
 - IronRuby, 171
 - Is, 135
 - ISAPI, 721
 - IsCached(), 1152
 - IService1.cs, 1044
 - IsFalse, 420
 - IsInstanceOfType, 420
 - IsNot, 135
 - IsNotInstanceOfType, 420
 - IsNotNull, 420
 - IsNull, 420
 - IsTrue, 420
 - Item Collection Editor, 848
 - itemName, 335
 - iteratory, 139
- J**
- JavaScript, 59, 526, 793, 905
 - jednostki eksportowane, 690
 - język programowania, 40, 114
 - ASP, 165
 - CSS, 757
 - F#, 114
 - JavaScript, 59
 - język skryptowy, 41
 - T-SQL, 956
 - UML, 297
 - VBA, 1130
 - WSDL, 1012, 1029
 - XAML, 44, 48, 53, 61, 863
 - XHTML, 55
 - JOIN, 954
 - jQuery, 53, 716
 - JScript 8.0, 41
 - JScript File, 728
 - JsonResult, 808
 - Just-In-Time, 487
- K**
- kanaly RSS, 306, 308
 - kaskadowe arkusze stylów, 270
 - katalog, 691
 - katalog folderów, 691
 - katalog podzespółów, 691
 - katalog typów, 691

- katalogi ASP.NET, 726
- katalogi projektu, 203
- katalogi rozwiązania, 190
- katalogi zakładek, 248
- Key, 156
- kierowanie zapytań do modelu EDM, 1006
- klasy, 115
 - Action, 1154
 - Activity, 1077
 - Application, 863
 - Array, 142
 - ArrayList, 143
 - asocjacja, 297
 - AssemblyCatalog, 691
 - Assert, 420
 - atrybuty, 145
 - Binding, 881
 - BitmapSource, 891
 - CachedDataHostItem, 1153
 - CachedDataItem, 1153
 - CodeActivity, 1063, 1075, 1077
 - CollectionAssert, 421
 - CollectionBase, 143, 144
 - Connect, 645, 659
 - ContainerControl, 835
 - Control, 290, 830, 863
 - ControlTemplate, 879
 - Convert, 133
 - DataContext, 882
 - DataGridViewColumn, 983
 - DataServiceContext, 1177
 - DataTemplate, 893
 - Debugger, 260
 - definiowanie, 115
 - definiowanie relacji między klasami, 296
 - diagram, 293
 - DirectoryCatalog, 691
 - DispatcherObject, 863
 - DocumentReader, 862
 - dostępność składowych, 118
 - DTEEvents, 628
 - DynamicObject, 169
 - dziedziczenie, 121, 296
 - EventHandler, 149
 - FlowDocumentReader, 862
 - FlowLayoutPanel, 836, 837
 - FolderBrowserDialog, 895
 - Form, 823
 - implementacja interfejsu, 125
 - interfejsy, 296
 - klasy anonimowe, 156
 - klasy bazowe, 121
 - klasy częściowe, 158, 738
 - klasy kolekcji, 143
 - klasy LINQ, 1000
 - klasy LINQ to SQL, 996
 - klasy ogólne, 174
 - klasy statyczne, 120
 - konstruktory, 118, 154
 - kontrola dostępu do danych, 116
 - MessageMapper, 258
 - metody, 117
 - NativeActivity, 1063, 1077
 - ObjectQuery, 1007
 - Panel, 836
 - pola, 115
 - poziom dostępu do klasy, 118
 - przeciążanie składowych, 123
 - przesyłanie operacji, 122
 - ServerDocument, 1152, 1153
 - składowe statyczne, 120
 - składowe wirtualne, 122
 - SmartTag, 1153
 - SplitContainer, 837
 - StringAssert, 422
 - Style, 878
 - TableLayoutPanel, 836
 - TableServiceContext, 1177
 - TestContext, 413
 - ToolStrip, 844
 - ToolStripContainer, 838
 - ToolStripItem, 841, 844
 - ToolTip, 841
 - Tuple, 144
 - tworzenie, 293
 - TypeCatalog, 691
 - ukrywanie składowych, 123
 - UrlRoutingModule, 807
 - Visual, 863, 879
 - WebService, 1027
 - WebServiceAttribute, 1027
 - WizardDialog, 681
 - właściwości, 115
 - WorkflowApplication, 1074
 - WorkflowInvoker, 1074
 - WorkflowServiceHost, 1075
 - wyświetlanie składowych, 294
- klasy atrybutów testów, 415
- klasy testów, 400, 406
- klawisze skrótów, 607

- klient sieciowy, 53
- klient SOAP, 1037
- klient WCF, 1042
- klucz główny, 945, 950
- klucze obce, 946, 948, 950
- kod HTML, 285
- kod LINQ, 998
- kod SQL, 953
- kod szablonowy, 380
- kod XAML, 48, 280
- kod XML, 165
- kodowanie, 469
- kolejki, 1168
- kolejkowanie komunikatów, 228
- kolekcje, 141, 143
 - ArrayList, 143
 - CollectionBase, 143
 - definiowanie, 155
 - Hashtable, 143
 - inicjowanie, 155
 - Queue, 143
 - SortedList, 143
 - Stack, 143
- kolekcje ogólne, 144
 - wariancja, 174
- kolorowanie składni, 365
- kompilacja, 469
 - aplikacje ASP.NET, 731
 - procedury składowane, 974
- kompilator, 364
- komponenty, 289, 830
 - pisanie kodu, 291
 - tworzenie, 290
- komponenty COM, 661
- kompozycje, 770
 - App_Themes, 771
 - stosowanie, 773
 - tworzenie, 771
- komunikacja P2P, 176
- komunikacja z pamięcią trwałą, 1118
- komunikaty SOAP, 1031
- komunikaty XML, 165
- konfiguracja atrybutów testów jednostek, 431
 - DataSource, 431
 - definiowanie połączenia, 431
 - DeploymentItem, 431
- konfiguracja chmury, 65
- konfiguracja kompilacji, 193, 206
- konfiguracja punktów przerwania, 259
- konfiguracja środowiska programistycznego, 79
- konfiguracja usług WCF, 76, 1050
- konfiguracja uwierzytelniania użytkowników, 796
- konflikty nazw, 128
- konstruktory, 118, 154
- konsumowanie usługi sieciowej ASP.NET, 1033
- konsumowanie usługi WCF, 74, 76, 1055
- kontekst danych, 882
- kontener łączący, 690
- kontenery, 836
 - FlowLayoutPanel, 836, 837
 - SplitContainer, 836, 837
 - StatusStrip, 847
 - TableLayoutPanel, 836
 - ToolStripContainer, 838
- kontrakt usługi, 1042
- kontrakty, 690
- kontrawariancja, 173
- kontrola dostępu do danych, 116
- kontrola wersji, 37
- kontrola wykonywania kodu, 260
- kontroler, 806, 807
- kontrolki, 273, 282, 289, 738, 823, 830
 - Canvas, 871
 - chmurki, 841
 - ContentPlaceHolder, 766
 - DataGridView, 852, 980
 - DataList, 990
 - DateTimePicker, 980, 983
 - DetailsView, 989
 - DockPanel, 872
 - dokowanie, 834
 - dziedziczenie po istniejącej kontrolce, 853
 - formularze Windows, 1147
 - FormView, 989
 - Grid, 874
 - GridView, 989, 990
 - Image, 888
 - ImageList, 850
 - kolejność przechodzenia, 840
 - kontenery, 836
 - kontrola automatycznego skalowania, 835
 - kontrolki ASP.NET Ajax Library, 915
 - kontrolki danych, 989
 - kontrolki kontenerowe, 835
 - kontrolki sieciowe, 989
 - kontrolki Silverlight, 936
 - kontrolki WPF, 705
 - kontrolki XAML, 49
 - kontrolki złożone, 290
 - kotwiczenie, 833

kontrolki

- ListBox, 886
- menu, 842
- MenuStrip, 46, 841, 842, 843
- modyfikacja atrybutów, 277
- Panel, 46
- pasek narzędzi, 844
- pasek statusu, 847
- pozycjonowanie, 749, 831
- Repeater, 989
- rozmieszczanie, 275
- SplitContainer, 46, 849
- Splitter, 837
- StackPanel, 876
- StatusBar, 847
- StatusStrip, 46, 841
- style wizualne, 840
- Timer, 274
- ToolStrip, 841, 844, 847
- ToolStripItem, 841, 844
- TreeView, 849
- tworzenie, 290
- tworzenie kontroltek, 853
- układ, 831
- UpdatePanel, 58
- UpdateProgress, 58
- wiązanie danych, 974
- wskazówki, 841
- wygląd, 840
- zachowanie, 840
- zmiana rozmiaru, 277

kontrolki Ajax Control Toolkit, 917

- Accordion, 917
- AlwaysVisibleControlExtender, 917
- AutoCompleteExtender, 917
- CalendarExtender, 918, 919
- CascadingDropDown, 918
- CollapsiblePanelExtender, 917
- ConfirmButtonExtender, 918, 921
- FilteredTextBoxExtender, 918
- HoverMenuExtender, 918
- ListSearchExtender, 918
- MaskedEditExtender, 918
- ModalPopupExtender, 918
- PasswordStrength, 918
- PopupControlExtender, 918
- ReorderList, 918
- SliderExtender, 918
- SlideShowExtender, 919
- stosowanie kontroltek, 919

- TabContainer, 919

- TextBoxWatermarkExtender, 919

kontrolki ASP.NET, 788

- AccessDataSource, 800
- adaptacyjne generowanie, 789
- Button, 792
- Calendar, 792
- ChangePassword, 796
- Chart, 800, 801
- CheckBox, 792
- CompareValidator, 794
- CreateUserWizard, 796
- CustomValidator, 794
- DataList, 800
- DataPager, 800
- DetailsView, 800
- DropDownList, 792
- EntityDataSource, 800
- FileUpload, 792
- FormView, 800
- GridView, 801
- Image, 792
- kontrolki danych, 799
- kontrolki do sprawdzania poprawności, 793
- kontrolki logowania, 795
- kontrolki nawigacyjne witryny, 798
- kontrolki standardowe, 792
- kontrolki użytkownika, 801
- kontrolki walidacyjne, 793, 794
- Label, 792
- LinqDataSource, 800
- ListBox, 792
- ListView, 801
- Literal, 792
- Login, 795
- LoginName, 796
- LoginStatus, 796
- LoginView, 795
- model obsługi zdarzeń, 789
- MultiView, 792
- ObjectDataSource, 800
- Panel, 792
- PasswordRecovery, 795
- QueryExtender, 801
- RadioButton, 792
- RangeValidator, 794
- RegularExpressionValidator, 793, 794
- Repeater, 801
- RequiredFieldValidator, 794
- SiteMapDataSource, 800

- SiteMapPath, 798
 - skórki, 790
 - sprawdzanie poprawności danych, 790, 793
 - SqlDataSource, 800
 - style, 790
 - Table, 792
 - TextBox, 792
 - układ oparty na szablonach, 790
 - ValidationSummary, 793, 794
 - wiązanie danych, 790
 - Wizard, 793
 - wprowadzanie danych, 790
 - wykrywanie przeglądarek, 789
 - XML, 793
 - XmlDataSource, 800
 - zachowywanie stanu, 789
 - zgodność ze standardem XHTML, 789
 - kontrolki ASP.NET Ajax, 905
 - ScriptManager, 905, 906
 - ScriptManagerProxy, 906
 - Timer, 906
 - UpdatePanel, 906
 - UpdateProgress, 906, 913
 - kontrolki niestandardowe, 290, 805, 853, 857
 - kontrolki „rysowane przez programistę”, 857
 - tworzenie, 857
 - kontrolki użytkownika, 290, 853, 854
 - dodawanie kontroltek, 855
 - projekt, 854
 - projektowanie, 854
 - przechwytywanie zdarzeń, 661
 - tworzenie, 655
 - zagnieżdżanie kontroltek użytkownika, 855
 - kontrolki użytkownika ASP.NET, 801
 - dodawanie kontrolki do strony, 803
 - projekt, 802
 - rejestrwanie kontrolki na stronie, 804
 - tworzenie, 802
 - Web User Control, 802
 - kontrolki Web Part, 776
 - CatalogZone, 779
 - EditorZone, 779, 786
 - LayoutEditorPart, 786
 - strefy strony, 777
 - style, 778
 - ustawienia kontroltek, 788
 - WebPartManager, 777, 779
 - WebPartZone, 777, 779
 - zarządzanie układem strony, 777
 - kontrolki źródła danych, 993
 - AccessDataSource, 993
 - ObjectDataSource, 993
 - SiteMapDataSource, 993
 - SqlDataSource, 993
 - XMLDataSource, 993
 - kontrolki-hosty, 1148
 - kontynuowanie wykonywania kodu, 493
 - konwersja typów, 131
 - kończenie sesji diagnostycznej, 493
 - kotwiczenie, 833
 - kowariancja, 173
 - kreator dodatków, 637, 677
 - kreator eksportowania szablonów, 337
 - kreator modelu EDM, 69
 - kreatory, 677
 - Add New Item, 680
 - ContextParams, 678
 - Execute, 681
 - formularz, 682
 - identyfikator kreatora, 679
 - IDTToolsOptionsPage, 683
 - IDTWizard, 677, 679, 680
 - numer wersji, 679
 - plik szablonu, 682
 - pliki .vsdir, 679, 680
 - pliki .vsz, 679
 - struktura kreatorów, 677
 - tworzenie, 680
 - tworzenie okna dialogowego, 681
 - tworzenie plików .vsz i .vsdir, 685
 - uruchamianie kreatora, 683
 - WizardDialog, 681
 - wizardResult, 679
 - kroczenie przez kod, 480
 - krotki, 144
 - dodawanie elementów, 144
 - tworzenie, 144
 - kultura, 823
- L**
- Label, 274, 755, 792
 - LabelInstructions, 768
 - Language Integrated Query, 160
 - Launch Conditions, 546
 - Layout, 102
 - LayoutEditorPart, 786
 - LayoutMode, 275
 - Left, 833

liczba dojść do punktu przerwania, 502
 liczniki wydajności, 228
 LIKE, 909
 LineDown, 601
 LineUp, 601
 LinkedWindows, 587
 LINQ, 156, 160, 161, 176, 816, 994, 998
 atrybuty, 999
 klasy, 1000
 odzworowanie O/R, 995
 podsystemy, 995
 stosowanie obiektów LINQ, 1000
 wyrażenia lambda, 162, 163
 zapytania, 994
 LINQ to Objects, 995
 LINQ to SQL, 717, 728, 995, 996
 LINQ to SQL Classes, 728, 996
 LINQ to XML, 995
 LinqDataSource, 800
 List Members, 377, 378
 lista składowych, 378
 lista zadań do wykonania, 393
 ListBox, 792, 885, 886
 ListObject, 1149
 ListSearchExtender, 918
 ListView, 47, 801, 850
 Literal, 792
 Load, 743
 Load Macro Project, 619, 629
 Load Metadata File, 402
 Load Test Virtual User Pack, 39
 LoadComplete, 743
 local type inference, 151
 Local.testsettings, 400
 Locals, 476, 485, 505, 506
 logika biznesowa, 158
 Login, 795
 Login.aspx, 796
 LoginName, 796
 LoginStatus, 796
 LoginView, 795
 logowanie, 795
 uwierzytelnianie użytkowników, 796
 lokalizacja, 823
 lokalizacja kodu źródłowego na potrzeby
 diagnozowania, 192
 lokalna pamięć podręczna dla bazy danych, 71
 long, 130
 Long, 130

Ł

łańcuch połączenia, 973
 łańcuchy znaków, 130
 łączenie łańcuchów znaków, 135
 łączenie okien narzędzi, 587

M

machinename, 335
 Macro Explorer, 614, 617, 619
 Macros, 616, 619
 diagnozowanie makra, 623
 dodawanie projektów, 619
 obsługa zdarzeń, 623
 pisanie makr, 620
 wywoływanie makra, 629
 Main, 242, 367
 MakeCert.exe, 349
 MakeZipExe.exe, 349
 makra, 345, 612
 diagnozowanie, 623
 dodawanie deklaracji nowych zdarzeń, 627
 dodawanie projektów, 619
 dostęp do okna, 620
 dostęp do zdarzenia, 627
 DTEEvents, 628
 DTEEvents.OnMacrosRuntimeReset, 628
 DTEEvents.OnStartupComplete, 628
 edytor kodu, 621
 eksportowanie modułu, 618
 EnvironmentEvents, 623, 625
 format przechowywania, 618
 inicjowanie obiektów zdarzeń, 628
 kod makra, 615
 Macro Explorer, 614
 Macros, 616
 makra tymczasowe, 614, 617
 metody obsługi zdarzeń, 626
 modyfikacja kodu, 621
 obsługa zdarzeń, 623
 parametry, 632
 pisanie makr, 616, 620
 pliki .vsmacros, 619
 projekt, 617
 rejestrwanie, 613
 skrótów klawiaturowe, 631
 tworzenie, 613
 uruchamianie, 613, 614, 629
 WindowHiding, 627

- współużytkowanie makr, 618
- wstrzymanie rejestrowania, 613
- wyświetlanie komunikatów, 615
- zarządzanie makrami, 614
- zmiana formatu przechowywania projektu
 - makr, 619
 - zmiana rozmiaru IDE, 632
- Manage Styles, 287, 288, 758, 760, 761, 762
- Managed Execution Framework, 23
- Managed Extensibility Framework, 688
- manifest VSIX, 352, 353
- manipulowanie obrazami, 895
- manipulowanie zakładkami w dokumencie, 594
- Manual, 828
- mapa witryny, 728, 798, 993
- Mapping Details, 69, 1002
- marginesy, 696
- MaskedEditExtender, 918
- Master, 768
- Master Page, 56, 727
- master pages, 765
- mechanizm DAC, 969
- mechanizm generowania kodu, 158
- mechanizm IntelliSense, 363, 375
- mechanizm nawigowania po znacznikach, 370
- mechanizm obsługi korelowania, 1112
- mechanizm szeregowania zadań, 523
- mechanizm wstawiania fragmentów kodu, 381
- mechanizm wyszukiwania przyrostowego, 577
- mechanizmy języka, 150
- mechanizmy obsługi myszy, 693
- mechanizmy obsługi upuszczania, 693
- MEF, 23, 688
 - AggregateCatalog, 691
 - aktywowanie, 690
 - aplikacje, 690
 - aplikacje nadrzędne, 689
 - architektura platformy, 689
 - AssemblyCatalog, 691
 - CompositionContainer, 691
 - części, 690, 695
 - dekoracje, 697, 698
 - DirectoryCatalog, 691
 - dodawanie zależności, 689
 - dopasowywanie strukturalne, 690
 - Editor Margin, 696
 - Editor Text Adornment, 697
 - Editor Viewport Adornment, 698
 - edytor środowiska Visual Studio, 691, 692
 - Extension Manager, 699
 - jednostki eksportowane, 690, 692
 - jednostki importowane, 690
 - katalog, 691
 - katalog folderów, 691
 - katalog podzespołów, 691
 - katalog typów, 691
 - kontener łączący, 690
 - kontrakty, 690
 - menedżer rozszerzeń, 699
 - model programowania, 689
 - określanie nazw, 690
 - projekt dekoracji, 702
 - punkty dołączania rozszerzeń do edytora, 692
 - reguły działania, 689
 - rozszerzalne platformy, 689
 - rozszerzenia, 689
 - stosowanie, 690
 - tworzenie rozszerzenia edytora, 701
 - TypeCatalog, 691
- Memory, 485
- menedżer dodatków, 653
- menu, 89, 842
 - Build, 91
 - Data, 92
 - Debug, 91, 482
 - Edit, 89
 - elementy menu, 844
 - File, 85, 89
 - Format, 92
 - główne, 842
 - Help, 93, 312
 - MenuStrip, 842, 843
 - Project, 89, 91
 - Refactor, 89, 90, 442
 - Table, 92
 - Test, 93, 401
 - Tools, 92
 - View, 90
 - Website, 90
 - Window, 93
- MenuItem, 888
- MenuStrip, 46, 841, 842, 843
- MessageMapper, 258
- Method Name, 244
- metody, 117
 - definiowanie, 117, 298
 - dodawanie metod do istniejących klas, 157
 - metody częściowe, 158
 - przeciążanie, 123
 - przesłanianie, 122
 - refaktoryzacja, 450

metody dodatkowe, 157
 użycie, 158
 metody sieciowe, 1015
 wywoływanie, 1032
 metody usługi sieciowej, 1015
 MFC, 41
 Microsoft Excel, 1132
 Microsoft Expression, 39
 Microsoft Foundation Class, 41
 Microsoft InfoPath, 1132
 Microsoft Office, 63, 1130
 Microsoft Outlook, 1132
 Microsoft Patterns and Practices, 306
 Microsoft PowerPoint, 1132
 Microsoft Report Viewer, 717
 Microsoft Silverlight, 59
 Microsoft Solutions Framework (MSF)
 for CMMI 5.0, 36
 Microsoft Word, 1130, 1132
 Microsoft.Office.Tools.Excel.SmartTag, 1153
 Microsoft.Office.Tools.Excel.Worksheet, 1147
 Microsoft.Office.Tools.SmartTagBase, 1153
 Microsoft.Office.Tools.Word.Document, 1147
 Microsoft.Office.Tools.Word.SmartTag, 1153
 Microsoft.VisualStudio.CoreUtility, 351
 Microsoft.VisualStudio.QualityTools.
 ↳ UnitTestFramework, 400
 Microsoft.VisualStudio.TestTools.UnitTesting, 413
 Microsoft.VisualStudio.Tools.Applications, 1152
 MIME, 931
 ModalPopupExtender, 918
 model, 806
 Model Browser, 1002, 1004
 model EDM, 68, 69, 977, 1001
 zapytania, 1006
 model formularzy sieciowych, 809
 model MVC, 56
 model procesu, 72
 model projektowy, 68
 model przetwarzania żądań w aplikacjach ASP.NET
 MVC, 807
 model źródła danych, 68
 Model-View-Controller, 56, 714
 model-widok-kontroler, 56
 Modify Style, 758, 759
 Modules, 485
 modyfikacja
 dokumenty tekstowe, 595
 kod makra, 621
 manifest VSIX, 352

model EDM, 1002
 panel zadań pakietu Office, 1141
 plik .vstemplate, 338
 strona startowa, 310
 style CSS, 287, 764
 tekst, 600
 wstążka, 1137
 zapytania dla źródła danych, 985
 znaczniki HTML, 285
 monitor zdalnego diagnozowania, 513
 MOSS, 776
 MSDN, 35, 314
 MSDN Forums, 315
 MSF for Agile 5.0, 36
 MSI, 341
 MSMQ, 1014
 msvsmon.exe, 513
 MultiView, 792
 MustInherit, 121
 MVC, 56, 714, 805

N

NamedRange, 1149
 namespace, 127
 Namespace, 127
 narzędzia, 36
 narzędzia pomocnicze edytorów tekstu, 364
 nawigowanie po kodzie, 246
 refaktoryzacja, 441
 tworzenie stylów, 756
 VSTO, 1130
 NativeActivity, 1063, 1077
 nawiasy, 391
 nawigacja, 367
 nawigacja po środowisku IDE, 88
 nawigowanie po kodzie, 246
 Call Hierarchy, 248
 numerowanie wierszy, 246
 zakładki, 247
 nawigowanie po znacznikach, 370
 Document Outline, 371
 nazwy, 127
 nazywanie elementów, 373
 NetCF, 45
 new, 123, 141
 New, 156
 New Breakpoint, 483, 486, 495
 New Cloud Service Project, 64, 1166, 1170, 1175
 New Data Breakpoint, 483

- New File, 241
 - New Line, 192
 - New Macro Project, 619
 - New Project, 43, 50, 54, 79, 84, 86, 183, 184, 194, 272, 330, 334, 962, 1135
 - Online Templates, 328
 - New Query, 952
 - New Service Endpoint Element Wizard, 1051, 1052
 - New Style, 761
 - New Style Sheet, 761
 - New Test, 401
 - New Test List, 436
 - New Web Site, 197, 283, 715
 - niejawne określanie typu zmiennych, 151
 - zmiennie lokalne, 153
 - niestandardowa strona startowa, 310
 - Not, 136
 - NotInheritable, 121
 - numerowanie wierszy, 246
 - NUnit, 398
- O**
- O/R Designer, 996
 - dodawanie jednostek bazy danych, 996
 - klasy danych, 997
 - klasy LINQ to SQL, 996
 - kod LINQ, 998
 - nazwy jednostek, 999
 - obszar projektowy, 997
 - tabele, 997
 - objektowy model automatyzacji, 560
 - BuildEvents, 609
 - CodeClass, 569
 - CodeElement, 569
 - CodeElements, 569, 570
 - Command, 605
 - Command Window, 583
 - CommandBar, 588, 589, 590
 - CommandBarControl, 589
 - Commands, 605
 - CommandWindow, 578, 583
 - debuger, 608
 - Debugger, 608
 - Document, 592
 - Documents, 592
 - dodatki, 653
 - dodawanie tekstu, 599
 - dokumenty, 592
 - dokumenty tekstowe, 593
 - dostęp do kodu projektu, 569
 - dostęp do okien, 572
 - dostęp do paneli okna tekstu, 577
 - DTE, 561, 564
 - DTE2, 561, 564
 - EditPoint, 597, 599, 600
 - EditPoint2, 597
 - Events, 608
 - hierarchia modelu obiektowego kodu, 570
 - hierarchia rozwiązań i projektów, 566
 - interakcja z oknami, 573
 - kategorie automatyzacji, 563
 - klasy, 570
 - kontrola projektów wchodzących w skład rozwiązania, 568
 - manipulowanie tekstem, 599
 - modyfikacja dokumentów tekstowych, 595
 - modyfikacja tekstu, 600
 - obiekty debugera, 608
 - obiekty polecenia, 604
 - okna, 572
 - okna narzędzi, 578
 - okna połączone, 586
 - okna tekstowe, 576
 - okno poleceń, 583
 - okno Toolbox, 581
 - Output, 584
 - OutputWindow, 578, 584
 - OutputWindowPane, 584, 585
 - OutputWindowPanes, 584
 - panele, 576, 577
 - pasek poleceń, 588
 - podzespoły, 561
 - polecenia, 604
 - Project, 565, 568
 - projekty, 565
 - przełączanie stanu ukończenia zadań, 580
 - przestrzenie nazw, 570
 - rozwiązania, 565
 - Solution, 565, 567
 - sprawdzanie kolekcji okien, 575
 - Task List, 579
 - TaskItem, 580
 - TaskList, 578, 579, 580
 - TextDocument, 593
 - TextPane, 577
 - TextSelection, 596
 - TextWindow, 576
 - ToolBox, 578, 581
 - ToolBoxItem, 581

- objektowy model automatyzacji
 - ToolStripItems, 581
 - ToolStripTabs, 581
 - typy modelu automatyzacji, 562
 - ustawianie aktywnego okna, 575
 - VirtualPoint, 597
 - vsCMElement, 571
 - wersje modelu obiektowego, 561
 - Window, 572, 573
 - Windows, 588
 - wykonywanie poleceń, 606
 - wykonywanie poleceń w oknie poleceń, 583
 - wyszukiwanie przyrostowe, 577
 - zadania, 579
 - zdarzenia automatyzacji, 608
- objekty, 115, 130, 131, 153
 - inicjator obiektów, 153
 - konstruktory, 154
 - tworzenie, 153
- objekty bazy danych, 970
 - łańcuch połączenia, 973
 - procedury składowane, 971
- objekty blob, 1168
- objekty dynamiczne, 169
 - DynamicObject, 169
 - stosowanie, 171
 - tworzenie, 169
- objekty polecenia, 604
- objekty statyczne, 120
- objekty współużytkowane, 120
- object, 167
- Object, 152
- Object Browser, 229
 - Class View, 230
 - Edit Custom Component Set, 230
 - modyfikacja niestandardowego zbioru komponentów, 229
 - opcje zasięgu, 229
 - panel z opisem, 231
 - przeglądanie obiektów, 230
 - Rename, 447
 - zmiana nazw, 447
 - zmiana zasięgu, 229
- object initializer, 153
- ObjectDataSource, 800, 993
- ObjectQuery, 1007
- obrazy, 850
- obsługa danych, 67
- obsługa dodatków, 175
- obsługa korelowania, 1112
- obsługa okien środowiska IDE, 109
- obsługa ruchów kursora nad paletą, 655
- obsługa transakcji, 1087
- obsługa wariacji, 174
- obsługa właściwości, 166
- obsługa wyjątków, 146
- obsługa zapytań LINQ, 162
- obsługa zbiorów danych, 67
- obsługa zdarzeń, 149, 623
 - dotatki, 645
 - kontrolki wstążki, 1141
 - usuwanie nieużywanych argumentów z metod obsługi zdarzeń, 166
- obszary kodu, 368
- ODBC, 993
- oddzielanie projektu danych od modelu źródła danych, 68
- odpowiedź SOAP, 1032
- odświeżanie częściowe, 912
- odwzorowanie O/R, 994
- odwzorowanie obiektowo-relacyjne, 993
 - LINQ, 994
 - O/R Designer, 996
- odwzorowywanie obiektów na modele bazy danych, 68
- odwzorowywanie źródeł danych na kontrolki, 978
- Office, 44, 50, 63, 1130
- ograniczenia, 946
- okna, 572
 - dostęp do okien, 572
 - interakcja z oknami, 573
- okna dialogowe, 681, 875, 895
- okna narzędzi, 103, 578
- okna połączone, 586
- okna projektowe, 102, 240
 - okno projektowe aplikacji sieciowych, 283
 - okno projektowe formularzy Windows, 373
 - okno projektowe HTML, 283, 372
 - okno projektowe klas, 295
- okna tekstowe, 576
- okno edytora kodu, 243
- okno projektowe procesów, 1067
 - powierzchnia projektowa procesu, 1067
- okno projektowe szablonów XML, 267
 - Content Model View, 268
 - Graph View, 267, 268
 - Start View, 267
- okno projektowe WPF, 280, 282, 865
 - kontrola rozmiaru, 868
 - kontrolka przybliżania, 869
 - okno właściwości, 868

- określanie pozycji, 868
 - panele projektowe, 866
 - Properties, 868
 - widok podzielony, 281
 - XAML, 866
 - zarządzanie panelami, 867
 - określanie nazw, 690
 - OLE DB, 993
 - OnAddInsUpdate, 646, 647
 - OnAfterCreated, 683
 - OnBeginShutdown, 646, 647
 - OnConnection, 646, 648
 - OnDisconnection, 646, 650
 - One-Click Publishing, 554
 - Online Templates, 328
 - OnMacrosRuntimeReset, 629
 - OnPaint, 857
 - OnSizeChange, 705
 - OnStartupComplete, 629, 646, 651
 - Opacity, 828
 - opcje diagnozowania, 487
 - opcje uruchomieniowe, 84
 - Open Folder in Windows Explorer, 1082
 - Open Project, 84
 - operacje inteligentne, 372
 - OperationContract, 1045
 - operator, 133
 - łączenie, 135
 - operatory arytmetyczne, 134
 - operatory logiczne, 135, 136
 - operatory warunkowe, 135, 136
 - operatory zapytań, 161
 - porównywanie, 134
 - przypisanie, 133
 - opis usługi sieciowej, 1029
 - opisywanie dodatków, 639
 - opóźnione podpisywanie, 209
 - Option Infer Off, 153
 - Option Strict, 153
 - Options, 84, 100, 101
 - Options and Settings, 483, 486
 - Or, 136
 - Oracle, 993
 - Order by, 163
 - Ordered Test, 433
 - OrElse, 136
 - organizowanie testów, 434
 - lista testów, 436
 - Test List Editor, 435
 - Test View, 434
 - uruchamianie testów z listy testów, 437
 - wyszukiwanie testów, 434, 435
 - wyświetlanie wszystkich testów, 434
 - otaczające fragmenty kodu, 383
 - otaczanie kodu fragmentami, 383
 - otwarte aplikacje .NET, 688
 - otwarte dokumenty, 592
 - otwieranie edytora, 241
 - otwieranie starszej aplikacji, 88
 - out, 174
 - outlining, 367
 - Outlining, 369
 - Outlook, 1144
 - tworzenie regionów formularzy, 1144
 - wiązanie regionu z klasą komunikatu, 1146
 - Outlook Form Region, 1144
 - Output, 482, 484, 494, 578, 584
 - wyświetlanie informacji, 586
 - Output project, 403
 - Output type, 206
 - OutputWindow, 562, 578, 584
 - OutputWindowPane, 584, 585
 - składowe, 585
 - OutputWindowPanes, 584
 - Overridable, 122
 - override, 122, 123
 - Overrides, 122, 123
- ## P
- P2P, 176
 - Page, 428
 - Page_Event, 744
 - pakiet Office, 50
 - pakiet Visual Studio SDK, 693
 - pakiety, 341
 - plik VSContent, 342
 - VSI, 342
 - VSIX, 349
 - pakiety startowe, 333
 - pakowanie rozszerzeń do plików VSIX, 349
 - paleta do wybierania kolorów, 654
 - paleta komponentów, 830
 - PaletteControl, 661
 - pamięć podręczna danych, 1133, 1150
 - CachedAttribute, 1151
 - dodawanie danych, 1151
 - dostęp do pamięci podręcznej danych, 1152
 - wyspy danych, 1150, 1151

- pamięć trwała, 1117
- Panel, 46, 792, 836
- panele, 576, 577, 838
- panele operacji pakietu Office, 1132, 1149
 - ActionsPane.StackOrder, 1150
 - kontrola wyświetlania, 1150
 - StackStyle, 1150
 - tworzenie, 1149
- panele WPF, 871
- panele zadań pakietu Office, 1132
 - dodawanie kontrolek, 1142
 - modyfikacja, 1141
- Parallel, 1084
- Parallel LINQ, 995
- Parallel Stacks, 484, 521, 523
 - pasek narzędzi, 522
 - wątki, 521
 - widok zadań, 523
 - zadania, 523
- Parallel Tasks, 484, 523, 525
 - Location, 525
 - lokalizacja zadania, 525
- ParallelForEach, 1084
- Parameter Info, 379
- parametry, 379
 - refaktoryzacja, 462
 - usuwanie parametrów, 462
 - zmiana kolejności parametrów, 463
- parametry dodatku, 637
- parametry projektu, 334
- parametry szablonów, 335
- Parse, 132
- Partial, 159
- partial class, 738
- pasek menu, 842, 843
- pasek narzędzi, 94, 844
 - Breakpoints, 480
 - Debug, 482, 486
 - dostosowywanie, 95
 - ToolStrip, 844
 - ToolStripItem, 844
- pasek poleceń, 588
 - kontrolki, 590
- pasek statusu, 847
 - edycja elementów StatusStrip, 848
 - StatusBar, 847
 - StatusStrip, 847
- PasswordRecovery, 795
- PasswordStrength, 918
- Paste, 94
- Persist, 1084, 1085
- pętle, 138
 - do...while, 140
 - Do...While, 140
 - for, 139
 - For...Each, 139
 - For...Next, 139
 - foreach, 139
 - until, 140
 - Until, 140
 - while, 140
- Pick, 1084
- PictureBox, 47
- PictureContentControl, 1149
- pinning, 105
- pisanie kodu, 240, 241, 242, 277
 - kod komponentów, 291
 - kod makr, 616, 620
- pisanie testów jednostek, 405
- PK, 950
- PlainTextContentControl, 1149
- planowanie interfejsu użytkownika, 825
- platforma .NET, 23, 42, 175
- platforma .NET Compact Framework 3.5, 45
- platforma Azure, 1158
- platforma Entity, 68, 1001
- platforma MEF, 23, 688
- platforma testów jednostek, 398, 413
- platforma WCF, 23
- platforma WF, 23
- platforma Windows Presentation Foundation, 860
- platforma WPF, 23, 47
- Play, 94
- plik definicji projektu, 198
 - projekt Visual C#, 198
 - projekt Visual C++, 200
- plik definicji rozwiązania, 185
- pliki
 - addin, 654, 665
 - asax, 727
 - ascx, 727
 - ashx, 727
 - asmx, 75, 727, 1018
 - ASP.NET, 727
 - aspx, 727
 - cab, 540
 - config, 264, 728
 - cs, 727
 - cspkg, 66, 1199
 - csproj, 198

- css, 728
- dacpac, 969
- dbml, 728
- DISCO, 1016
- edmx, 69
- htm, 728
- js, 728
- master, 727
- MSI, 341
- pdb, 529, 530
- rdlc, 728
- resx, 728
- sitemap, 728, 798
- skin, 728
- svc, 728
- sync, 71
- testsettings, 411
- vb, 727
- vbproj, 198
- VSCContent, 342
- vmdir, 679
- VSI, 342
- VSIX, 310, 341, 349
- vsmacros, 619
- vssettings, 82
- vstemplate, 336, 338
- vsz, 679
- XAML, 280
- xml, 264
- xsd, 67, 265
- xsl, 264
- XSLT, 269
- pliki rzzutów, 527
- PLINQ, 176, 995
- pobieranie danych, 160
- początkowy formularz, 827
- podejmowanie decyzji, 135
- podgląd zmian, 445
- podglądanie danych w debugerze, 505
 - podpowiedzi DataTip, 508
 - wizualizacja danych, 510
- podglądanie zmiennych, 505
 - Autos, 506
 - Locals, 505
 - QuickWatch, 508
 - Watch, 507
- podmienianie tekstu w plikach, 253, 255
- podpisywanie plików, 349
- podpisywanie podzespołu, 209
- podpowiedzi DataTip, 483, 508
 - przyczepianie, 509
 - podzespoły, 164
 - podzespoły zaprzyjaźnione, 164
 - stosowanie, 164
 - pola, 115
 - definiowanie, 298
 - pola prywatne, 116
 - pola publiczne, 116
 - pole tekstowe, 274
 - polecenia, 604
 - argumenty, 606
 - Command, 605
 - dodatki, 651
 - klawisze skrótu, 607
 - obiektowy model automatyzacji, 604
 - polecenia nazwane, 651
 - wykonywanie, 606
 - polecenia SQL, 951
 - połączenie kontrolki ze źródłem danych, 881
 - połączenie z bazą danych, 943
 - połączenie z danymi, 224
 - pomoc, 312
 - aktualizacja lokalnego systemu pomocy, 313
 - Customer Feedback Options, 324
 - dodawanie materiałów do lokalnego systemu pomocy, 313
 - dokumentacja środowiska Visual Studio, 314
 - Find Content Online, 313
 - fora społeczności MSDN, 315
 - Help Library Manager, 312
 - menu Help, 312
 - MSDN, 314
 - MSDN Forums, 315
 - przesyłanie informacji zwrotnych, 323
 - przykłady, 326
 - Report a Bug, 324
 - Samples, 326
 - zarządzanie ustawieniami systemu pomocy, 312
 - zgłaszanie błędów, 323, 324
 - pomoc internetowa, 312
 - pomoc w usprawnianiu środowiska Visual Studio, 324
 - PopupControlExtender, 918
 - porównywanie, 134
 - poruszanie się po środowisku IDE, 88, 108
 - porządkowanie instrukcji Using, 380
 - Position, 750
 - powiązanie arkusza XSLT z dokumentem XML, 270
 - poziomy dostępu do składowych, 118, 119
 - pozycjonowanie bezwzględne, 748
 - pozycjonowanie kontrolki, 749, 831
 - pozycjonowanie względne, 748
 - pozycjonowanie za pomocą stylów CSS, 748

- PreInit, 743
- PreLoad, 743
- PreRender, 743
- PreRenderComplete, 743
- PresentationCore.dll, 860
- PresentationFramework.dll, 860
- Preview Changes, 445, 446
- primary key, 950
- Primitives, 1084
- Print, 262
- Private, 116, 119
- procedury obsługi zdarzeń, 829
 - kontrolki, 745
 - strony ASP.NET, 744
- procedury składowane, 956
 - C#, 971
 - diagnozowanie, 958
 - kompilacja, 974
 - polecenia SQL, 957
 - wdrażanie, 974
- proces, 1062, 1063, 1065
 - Activity, 1069, 1077
 - Activity Designer, 1069
 - Activity Library, 1065
 - argumenty, 1070
 - Code Activity, 1069, 1075
 - CodeActivity, 1075
 - CompensableActivity, 1087, 1088
 - Control Flow, 1084
 - czynności, 1082, 1083
 - czynności Collection, 1086
 - czynności Control Flow, 1083
 - czynności Error Handling, 1086
 - czynności Flowchart, 1094
 - czynności Messaging, 1089
 - czynności Primitives, 1084
 - czynności Runtime, 1084
 - czynności Transaction, 1087
 - DoWhile, 1083
 - elementy, 1068
 - Execute, 1077
 - ForEach, 1083
 - kolekcje, 1086
 - NativeActivity, 1077
 - obsługa kolekcji, 1086
 - obsługa transakcji, 1087
 - okno projektowe, 1067
 - Persist, 1084
 - projekt, 1065
 - projektowanie, 1107, 1110
 - przechowywanie, 1074
 - Sequence, 1083
 - składniki, 1063
 - sterowanie przepływem, 1083
 - szablony elementów, 1068
 - TransactionScope, 1087
 - tworzenie, 1069
 - tworzenie procesu opartego na komunikatach, 1090
 - tworzenie procesu opartego na XAML, 1069
 - tworzenie za pomocą szablonu Code Activity, 1075
 - utrwalanie, 1116
 - WCF Workflow Service, 1069
 - WCF Workflow Service Application, 1065
 - While, 1083
 - WorkflowApplication, 1074
 - WorkflowInvoker, 1074
 - WorkflowServiceHost, 1075
 - wywoływanie, 1074
 - XAML, 1069
 - zapisywanie procesu, 1077
 - zarządzanie komunikatami, 1089
- proces dynamiczny, 1080
 - konfiguracja, 1080
 - sposoby wywoływania, 1081
- proces hosta, 1063
- proces tworzenia oprogramowania, 182
- proces wiązania danych, 975
- Processes, 485
- procesy biznesowe, 72
- Program, 562
- programiści, 61
- programowanie obiektowe, 114, 115
- programowanie pod kątem chmury, 63
- programowanie z wykorzystaniem atrybutów, 145
- programowanie z wykorzystaniem WPF, 871
- programowanie za pomocą okna Server Explorer, 228
- programowe zarządzanie dodatkami, 653
- Project, 89, 91, 562, 565, 568, 575
 - CodeModel, 569
 - składowe, 568
- Project Dependencies, 192
- Project Designer, 204, 205
- Project Explorer, 618
- Project Properties, 827
- Project.ProjectItems, 565
- ProjectItem, 562, 575
- ProjectItems, 562
- projectname, 335

- projekt, 43, 85, 182, 194, 203
 - Activity, 1069
 - Activity Designer, 1069
 - Activity Designer Library, 1066
 - Activity Library, 1065
 - ADO.NET Entity Data Model, 812, 1001
 - aplikacje ASP.NET, 715
 - aplikacje ASP.NET MVC, 809
 - aplikacje do zarządzania procesem, 1097
 - aplikacje działające w chmurze, 64
 - aplikacje Office, 1135
 - aplikacje SharePoint, 52
 - aplikacje sieciowe, 283
 - aplikacje Silverlight, 59, 934
 - aplikacje Windows, 46, 272
 - aplikacje WPF, 48, 280, 866
 - aplikacje XBAP, 923
 - ASP.NET Empty Web Site, 1017
 - ASP.NET MVC 2 Web Application, 809
 - ASP.NET MVC 2 Web Role, 1167
 - ASP.NET MVC Web Application, 57
 - ASP.NET Web Application, 54
 - ASP.NET Web Role, 1167
 - ASP.NET Web Service, 1017
 - Azure, 1165
 - Blank Solution, 184
 - CGI Web Role, 1167
 - Cloud Service, 64
 - Code Activity, 1069, 1075
 - Database, 961
 - definiowanie niestandardowych parametrów, 336
 - dekoracje, 702
 - dodatki, 637
 - Editor Classifier, 695
 - Editor Margin, 696
 - Editor Text Adornment, 697
 - Editor Viewport Adornment, 698
 - eksportowanie do szablonu, 334, 336
 - elementy projektów, 203
 - Enable Windows Azure Tools, 1161
 - Excel 2007 Workbook, 1136
 - Inherited Form, 827
 - katalogi, 203
 - kontrolki użytkownika ASP.NET, 802
 - LINQ to SQL Classes, 996
 - makra, 617
 - parametry, 334
 - plik definicji projektu, 198
 - proces, 1065
 - rozwiązania oparte na pakiecie Office, 50
 - Setup Wizard, 540
 - Shared Add-in, 637
 - Silverlight Application, 59, 934
 - Silverlight Class Library, 934
 - Silverlight Project, 934
 - SQL Server CLR, 970
 - SQL Server Project, 970
 - strona startowa, 303
 - struktura projektu, 204
 - tworzenie, 85, 194, 333
 - tworzenie szablonów, 333
 - typy projektów, 195
 - usługi WCF, 1043
 - Visual Studio Add-in, 637
 - Visual Studio SDK, 694
 - VSIX, 350
 - WCF, 75
 - WCF Service Application, 75, 1043, 1049
 - WCF Service Library, 1049
 - WCF Service Web Role, 1167
 - WCF Workflow Service, 1069
 - WCF Workflow Service Application, 1043, 1065
 - Windows Application, 203, 826
 - Windows Azure Cloud Service, 1165
 - Windows Control Library, 854
 - Windows Form Application, 46
 - WinForms, 290
 - właściwości, 204
 - Word 2007 Document, 1136
 - Worker Role, 1167
 - Workflow Console Application, 1066
 - WPF Application, 48, 280, 866
 - WPF Browser Application, 48, 923
 - wybór docelowego środowiska, 86
 - zależności, 191
 - zarządzanie, 218
- projekt bazy danych, 961
 - DAC, 969
 - Database Options, 964
 - Import Database Scheme, 966
 - importowanie schematu, 966
 - opcje wdrażania, 969
 - Project Properties, 964
 - Schema View, 967
 - SQL Server 2008 Wizard, 963
 - struktura projektu, 964
 - tworzenie, 962
 - właściwości, 964
 - zarządzanie projektami, 969

- projekt startowy, 191
- projekt testów jednostek, 399, 400
 - opcje, 410
 - szablony, 399
 - tworzenie, 399
- projekt witryny, 196, 715
 - tworzenie, 197, 715
- projektanci, 61
- projektowanie aplikacji opartych na formularzach sieciowych, 283
- projektowanie danych, 67
- projektowanie formularzy, 822
- projektowanie formularzy WPF, 49, 62
- projektowanie interfejsu użytkownika, 746, 830
- projektowanie klas do obsługi danych, 68
- projektowanie kontrolki użytkownika, 854
- projektowanie procesu, 73, 1107, 1110
- projektowanie strony internetowej, 283
- projektowanie szablonów XML, 265
- Properties, 46, 47, 102, 104, 127, 431, 745, 868
- Property, 563
- Property Pages, 190, 729
- Protected, 119
- Protected Internal, 119
- przechodzenie przez kod, 491
- przechowywanie danych w platformie Azure, 1167
 - kolejki, 1168
 - obiekty blob, 1168
 - SQL Azure, 1169
 - tabele, 1167
- przechowywanie procesów, 1074
- przechowywanie udostępnianych zasobów, 330
- przechwytywanie zdarzeń kontrolki użytkownika, 661
- przeciążanie metody, 123
- przeciążanie składowych, 123
- przegląd kodu, 470
- przeglądanie obiektów, 230
- przeglądarka obrazów, 884
- przełączanie stanu ukończenia zadań, 580
- przeniesienie aplikacji na inną wersję platformy .NET, 87
- przerywanie działania na podstawie warunków, 500
- przerywanie działania programu w danej lokalizacji, 501
- przesłanie operacji, 122
- przestrzenie nazw, 127
- przesyłanie informacji zwrotnych, 323
- przeszukiwanie dokumentów, 249
 - Find and Replace, 249
 - Find Symbol, 249
 - Quick Find, 249, 250
 - Quick Replace, 250
 - Search In Files, 249
- przetwarzanie kolekcji, 139
- przezroczystość, 828
- przyciąganie do linii, 276
- przyciski, 274
- przyczepianie okna, 105
- przyczepianie podpowiedzi DataTip, 509
- przykłady, 326
- przypadki użycia, 823
- przypisanie, 133
- public, 117, 119
- Public, 119
- publikowanie aplikacji ASP.NET, 549
 - Copy Web Site Tool, 549, 554
 - Web Deployment Tool, 549, 550
- publikowanie aplikacji działającej w chmurze, 66, 1199
- publikowanie projektów, 536
 - instalator systemu Windows, 539
- publikowanie witryn, 549
- Publish, 929
- Publish Cloud Service, 1199, 1202
- Publish Web, 554
- Publish Wizard, 536, 538, 929, 930
- punkt edycji, 599
- punkt końcowy usługi, 1042
- punkt przerwania, 245, 476, 480, 494
 - Condition, 500
 - etykiety, 499
 - Filter, 500
 - filtry, 502
 - funkcje, 495
 - Has Changed, 500, 501
 - Hit Count, 500, 502
 - ikony, 496
 - Is True, 500
 - konfiguracja, 259
 - kontrola wykonywania programu, 261
 - liczba dojść do punktu przerwania, 502
 - Location, 500
 - przerywanie działania programu w danej lokalizacji, 501
 - punkt śledzenia, 503
 - ustawianie, 258, 478, 494
 - ustawianie warunków, 500
 - ustawienia licznika, 503
 - When Hit, 500
 - zarządzanie, 498
- punkt śledzenia, 503
 - Continue execution, 504
 - ustawianie, 504
 - When Breakpoint Is Hit, 504
 - When Hit, 503
- punkty dołączania rozszerzeń do edytora, 692

Q

Query Builder, 909
 Query Designer, 94, 952
 Query/View Designer, 951, 956, 957
 QueryExtender, 801
 Queue, 143
 Quick Find, 249, 250

- All Open Documents, 250
- Current block, 250
- Current Document, 250
- doprecyzowanie wyszukiwania, 251
- Entire Solution, 250
- Find Next, 251
- Look In, 250, 251
- podmianianie tekstu, 252
- przeglądanie wyników wyszukiwania, 251
- Quick Replace, 252
- Selection, 250
- zasięg wyszukiwania, 250

 Quick Info, 377
 Quick Replace, 250, 252
 QuickWatch, 486, 508

R

RAD, 822
 RadioButton, 792
 RangeValidator, 794
 Rapid Application Development, 822
 raporty sieciowe, 717
 rdbsetup.exe, 513
 ReadOnly, 117
 Receive, 1090, 1115
 ReceiveAndSendReply, 1090, 1091
 Record TemporaryMacro, 613
 RedirectResult, 808
 RedirectToRouteResult, 808
 Refactor, 89, 90, 442
 refaktoryzacja, 440

- Class Designer, 444
- Encapsulate Field, 441, 465
- Extract Interface, 441, 459
- Extract Method, 441, 451
- generowanie szkieletu metody, 457
- hermetyzacja pól, 465
- inteligentne znaczniki, 442
- narzędzia, 441
- parametry, 462
- podgląd zmian, 445

Preview Changes, 445
 Refactor, 442
 Remove Parameters, 441, 462
 Rename, 441, 446
 Reorder Parameters, 441, 463
 uruchamianie narzędzi, 441
 usuwanie parametrów, 462
 wyodrębnianie interfejsów, 459
 wyodrębnianie metod, 450
 wyodrębnianie pojedynczych wierszy kodu, 456
 zmiana kolejności parametrów, 463
 zmiana nazw, 446
 referencje, 729
 referencje sieciowe, 1033

- wyświetlanie, 1036

 refleksja, 145
 region, 368
 regiony formularzy aplikacji Outlook, 1144
 registeredorganization, 335
 RegisterRoutes, 819
 Registers, 485
 RegularExpressionValidator, 793, 794
 rejestrowanie kontrolki na stronie, 804
 rejestrowanie makr, 613
 relacje między klasami, 296

- asocjacja, 297
- dziedziczenie, 296
- interfejs, 296

 relacyjne bazy danych, 993
 Remote Debugging Monitor, 513
 Remoting, 1014
 Remove and Sort, 380
 Remove Parameters, 441, 462
 Remove Unused Usings, 380
 RemoveFromCollection<T>, 1087
 Rename, 441, 446, 449

- opcje, 449
- Preview Reference Changes, 449
- Search in Comments, 450
- Search in Strings, 450
- zmiana nazw, 449

 Reorder Parameters, 441, 463
 ReorderList, 918
 Repeater, 801, 808, 989
 Report a Bug, 324
 Report Wizard, 728
 Reporting Services, 1169
 RequestedPage, 429
 RequiredFieldValidator, 794
 Resource File, 206, 728

- Restart, 485
- Rethrow, 1086
- return, 118
- Return, 117
- REVIEW, 395
- ręczne wiązanie kontrolek formularzy Windows, 982
- RIA, 904, 940
- Ribbon (Visual Designer), 1137
- Ribbon (XML), 1137
- RibbonControlSizeLarge, 1139
- Rich Internet Applications, 940
- RichTextContentControl, 1149
- Right, 833
- RightToLeft, 823
- RightToLeftLayout, 823
- role, 1164
 - Web, 1164
 - Worker, 1164
- Root namespace, 206
- rootnamespace, 335
- routed events, 883
- Row.Height, 889
- rozbudowywanie edytora kodu, 688
- rozgałęzianie kodu, 135
- rozłączanie okien narzędzi, 587
- rozmieszczanie kontrolek, 275, 284
 - przyciąganie do linii, 276
 - układ tabelaryczny, 275
- rozpoczynanie sesji diagnostycznej, 489
- rozpowszechnianie pików VSIX, 354
- rozszerzalne funkcje pakietu Office, 1131
 - pamięć podręczna danych, 1133
 - panele operacji, 1132
 - panele zadań, 1132
 - tagi inteligentne, 1134
 - wstążka, 1133
- rozszerzenia, 688, 689
- rozszerzenia dokumentów Office, 1147
 - kontrolki formularzy Windows, 1147
 - kontrolki kontenerowe, 1147
 - kontrolki-hosty, 1148
 - pamięć podręczna danych, 1150
 - tagi inteligentne, 1153
 - tworzenie paneli operacji, 1149
 - wyspy danych, 1150
- rozszerzenia edytora, 701
- rozwiązania, 182
 - dodawanie projektu, 194
 - elementy, 188
 - hierarchia plików, 185
 - katalogi, 190
 - kontrola projektów, 568
 - lokalizacja kodu źródłowego na potrzeby diagnozowania, 192
 - plik definicji rozwiązania, 185
 - praca programistów, 182
 - projekt, 184
 - projekt startowy, 191
 - referencje do projektów, 188
 - stosowanie, 188
 - tworzenie, 183
 - właściwości, 190
 - właściwości konfiguracji kompilacji, 193
 - zależności projektu, 191
 - zarządzanie, 217
- RS, 1169
- RSS, 306
- Run, 94
- Run TemporaryMacro, 613
- Run To Cursor, 472, 490
- Runtime, 1084
- rzutowanie, 132
- rzutowanie w dół, 123

S

- SafeArray, 607
- safeitemname, 335
- safeprojectname, 335
- samodzielne sprawdzanie, 470
- Save Dump As, 486, 527
- SaveStateComplete, 743
- sbyte, 130
- scalanie łańcuchów znaków, 135
- Scalar-valued Function, 961
- ScaleTransform, 895
- scenariusz diagnozowania, 469
- Schema View, 967
- schematy kodu, 367
 - menu Outlining, 369
 - polecenia, 369
 - tworzenie, 367
- ScriptManager, 905, 906, 907
- ScriptManagerProxy, 906
- Scripts, 716
- SDK, 349, 693
- Sealed, 121
- Search In Files, 249
- Select, 163
- SELECT, 953, 955

- Select Active Test Settings, 411
- SELECT DISTINCT, 984
- Select Master Page, 738
- Select Style Sheet, 762
- Select...Case, 137
- SelectedItem, 563
- Send, 1090
- Send to Back, 752, 846
- SendAndReceiveReply, 1090
- Sequence, 1083, 1084
- SequentialWorkflow, 73
- SequentialWorkflowActivity, 1062
- serializacja, 1151
- Server Explorer, 67, 223, 942
 - Add Data Connection, 224
 - Add Server, 224
 - budowanie związków tabel, 948
 - Data Connections, 224, 943
 - definiowanie kolumn, 945
 - definiowanie tabel, 945
 - Event Logs, 226
 - indeksy, 946
 - klucze obce, 946
 - komponenty serwera, 225
 - Management Classes, 226
 - Management Events, 227
 - Message Queues, 228
 - modyfikacja definicji tabel, 948
 - ograniczenia, 946
 - określanie klucza głównego, 945
 - Performance Counters, 228
 - połączenia z danymi, 224
 - połączenie z bazą danych, 943
 - programowanie, 228
 - przeciąganie, 228
 - Services, 228
 - Step Into Stored Procedure, 958
 - tworzenie diagramu bazy danych, 947
 - wyzwalacze, 960
- ServerDocument, 1152, 1153
- Service Configuration Editor, 1051
- Service Reference Settings, 1034, 1035
- Service.asmx, 716
- Service.svc, 717
- Service1.svc, 1044
- Service1.svc.cs, 1044
- ServiceConfiguration.csfg, 1171
- serwer IIS, 718, 720
- serwer TFS, 36
- Session, 428, 1017
- Set as Recording Project, 617
- Set as Startup Project, 1049
- Set Next Statement, 260
- Set Primary Key, 945
- Set Symbol Paths, 528
- SetFocus, 575
- SetSelectionContainer, 574, 575
- SetTabPicture, 574, 575
- Setup Wizard, 540
- Shadows, 123
- Shared, 120
- Shared Add-in, 637
- SharePoint, 38, 45, 52
- short, 130
- Short, 130
- Show Next Statement, 478
- Show Threads in Source, 516
- Sign Tool, 349
- SignTool.exe, 349
- silnik uruchomieniowy procesu, 1064
- Silverlight, 44, 53, 59, 62, 736, 904, 922, 933
 - tworzenie aplikacji, 934
- Silverlight 4.0, 939
- Silverlight Application, 59, 934
- Silverlight Class Library, 934
- Silverlight Project, 934
- Simple Object Access Protocol, 1016
- SimpleTypeName(), 388
- Single, 130
- Site Map, 728
- Site.css, 716
- Site.master, 716, 765, 803
- SiteMapDataSource, 800, 993
- SiteMapPath, 798
- SketchFlow, 40
- Skin File, 728
- SkinId, 772
- składniki procesu, 1063
- składowe, 118
 - składowe statyczne, 120
 - składowe wirtualne, 122
- skoroszyt aplikacji Excel, 1147
- skórki, 770
 - identyfikator, 775
 - SkinId, 772
 - skórki nazwane, 772
 - skórki nienazwane, 772
 - tworzenie pliku skórki, 772
 - typy definicji, 772

- skróty klawiaturowe
 - makra, 631
 - polecenia, 607
- skrypty działające po stronie klienta, 526
- skrypty SQL, 962
- SliderExtender, 918
- SlideShowExtender, 919
- SLQ, 909
- słowa kluczowe
 - class, 115
 - Class, 115
 - Function, 117
 - Private, 116
 - ReadOnly, 117
 - return, 118
 - Return, 117
 - Sub, 117
- SmartTag, 1153
- SmartTagBase, 1153
- SnapLines, 275
- SnapToGrid, 275
- snippets, 380
- SOAP, 75, 1012, 1016, 1031
 - wyjątki, 1041
- SoapException, 1040, 1041
- Software Publisher Certificate Test Tool, 349
- solution, 182
- Solution, 563, 565, 567
 - składowe, 567
- Solution Explorer, 88, 97, 205, 212, 241, 1049
 - Add New Solution Folder, 217
 - ASP.NET Configuration, 217
 - Copy Web Project, 216
 - Copy Web Site, 217
 - ikony, 213
 - ikony sygnalizacyjne, 216
 - Nest Related Files, 217
 - Properties, 217
 - przyciski paska narzędzi, 216
 - Refresh, 217
 - Unhide All, 217
 - View Class Diagram, 217
 - View Code, 217
 - View Designer, 217
 - wskazówki graficzne, 213
 - zarządzanie projektami, 218
 - zarządzanie rozwiązaniami, 217
 - zmiana nazw, 447
- Solution Property Pages, 190
- Solution.Projects, 565, 566
- Solution2, 563, 567
- Solution3, 563, 567
- Solution4, 563
- SolutionEvents, 626
- Sort Usings, 380
- SortedList, 143
- sortowanie instrukcji Using, 380
- Source, 55
- SourceControl, 563
- SourceControl2, 563
- Split, 55, 56
- SplitButton, 847
- SplitContainer, 46, 836, 837, 849
- Splitter, 837
- społeczność, 302
 - strona startowa Visual Studio, 303
 - wkład w społeczność, 332
- sprawdzanie kolekcji okien, 575
- sprawdzanie poprawności danych, 793
- sprawdzanie poprawności kodu HTML, 288
- sprawdzanie usługi sieciowej, 1031
- SQL, 951
 - funkcje definiowane przez użytkownika, 961
 - procedury składowane, 956
- SQL Azure, 1169
- SQL Debugger, 959
 - parametry, 959
- SQL Server, 942, 993
- SQL Server 2008 Wizard, 963
- SQL Server CLR, 970
- SQL Server Compact Edition, 71
- SQL Server Project, 970
- SqlCommand, 971
- SqlConnection, 971
- SqlDataSource, 800, 993
- SqlDataSource.UpdateQuery, 992
- SquiggleTag, 693
- SSL, 722, 797
- Stack, 143
- StackPanel, 871, 876, 885, 888
- StackStyle, 1150
- stałe, 133
- standardowe okna dialogowe, 895
- standardowy pasek narzędzi, 94
- standardy UI, 824
- Start, 191
- Start Automatic Outlining, 369
- Start Debugging, 483, 487, 491
- Start Page, 303
- Start Page news channel, 84
- Start Without Debugging, 191, 472, 483
- StartCaching(), 1152

- Starter Kit, 717
- StartPage.xaml, 310
- StartPosition, 827
- Startup object, 206
- Startup Project, 191
- StateMachineActivity, 1062
- StateMachineWorkflow, 73
- statement, 128
- Static, 120
- Static Code Analyzer, 470
- StatusBar, 847
- StatusStrip, 46, 841, 847
 - edycja elementów, 848
- Step Into, 480, 483, 486, 489, 491
- Step Into Stored Procedure, 958
- Step Out, 486, 493
- Step Over, 483, 486, 489, 491
- sterowanie przepływem w procesie, 1083
- Stop, 260
- Stop Debugging, 485
- Stop Hiding Current, 369
- Stop Outlining, 369
- Storage Connection String, 1186, 1197
- stos wywołań, 476
- stosowanie kompozycji, 773
- stosowanie obiektów dynamicznych, 171
- stosowanie stylów CSS, 763
- strefy strony Web Part, 777
- string, 151
- String, 151
- StringAssert, 422
- strona startowa, 83, 303, 304
 - Get Started, 305
 - Getting Started, 303
 - Guidance and Resources, 303, 306
 - kanały RSS, 306, 307
 - Latest News, 303, 306
 - Microsoft Patterns and Practices, 306
 - modyfikacja, 310
 - obszar projektów, 303
 - operacje wykonywane przy uruchamianiu, 308
 - StartPage.xaml, 310
 - tworzenie niestandardowej strony startowej, 310
 - ustawianie niestandardowych kanałów z wiadomościami, 307
 - wiadomości związane ze środowiskiem, 306
- stronicowanie, 912
- strony ASP.NET, 55, 737
 - arkusze stylów, 754, 756
 - cykl życia, 741
 - dodawanie kontrolki, 738
 - dodawanie procedur obsługi zdarzeń, 744
 - dodawanie procedur obsługi zdarzeń kontrolki, 745
 - dodawanie zasady stylu, 759
 - generowanie strony, 742
 - inicjowanie strony, 742
 - Init, 743
 - InitComplete, 743
 - kod źródłowy, 739
 - Load, 743
 - LoadComplete, 743
 - model zdarzeń, 741
 - obsługa zdarzeń związanych z odesłaniem strony, 742
 - odpowiedzi na zdarzenia, 741
 - odpowiedź dla użytkownika, 742
 - opcje pozycjonowania kontrolki, 749
 - położenie kontrolki, 747
 - PreInit, 743
 - PreLoad, 743
 - PreRender, 743
 - PreRenderComplete, 743
 - programowanie, 55
 - projekt, 55
 - projektowanie, 55, 738
 - rozpoczęcie przetwarzania, 741
 - SaveStateComplete, 743
 - sprawdzanie poprawności strony, 742
 - strony wzorcowe, 56, 765
 - style, 754
 - style elementów, 755
 - style z poziomu strony, 755
 - synchronizacja między kodem źródłowym a oknem projektowym, 55
 - tworzenie, 737
 - tworzenie jednolitego wyglądu i zachowania, 754
 - układ strony, 747
 - Unload, 744
 - ustawianie tego samego rozmiaru kontrolki, 752
 - ViewState, 742
 - wczytanie strony, 742
 - widok podzielony, 739
 - widok projektowy, 738
 - widok Split, 56
 - wyrównywanie elementów, 752
 - zarządzanie nawigacją i projektem, 56
 - zdarzenia, 743
 - żądanie użytkownika, 741
- strony obsługujące częściową aktualizację, 907
- strony Web Part, 777
 - chromowanie elementów, 781
 - definiowanie stref, 779
 - dodawanie elementów Web Part do stref, 781

- strony Web Part
 - edycja układu strony, 786
 - LayoutEditorPart, 786
 - strefy, 777, 780
 - tworzenie, 778
 - układ strony, 780
 - umożliwianie użytkownikom konfiguracji strony, 783
 - ustawienia kontroltek, 788
- strony wzorcowe, 56, 727, 765
 - @Page, 769
 - ContentPlaceHolder, 766, 768
 - dodawanie treści do elementu <head/>, 768
 - Master, 768
 - obszary z treścią, 768
 - plik kodu, 766
 - pozycjonowanie, 768
 - Site.master, 765
 - tworzenie, 765
 - tworzenie strony z treścią, 767
 - tytuł strony opartej na stronie wzorcowej, 769
 - układ kontroltek, 768
 - zagnieżdżanie stron wzorcowych, 769
- struct, 126
- structural matching, 690
- structure, 126
- Structure, 126
- struktura kreatorów, 677
- struktury, 125, 130
 - definiowanie, 126
- studia ergonomii, 823
- style, 754, 755
- Style, 878
- style elementów, 755
- Style Sheet, 728, 758
- style wewnątrzwerszowe, 755
- style z poziomu strony, 755
- Styles, 758
- Stylesheet, 270
- StylesheetTheme, 774
- Sub, 117
- subskrypcja usług platformy Azure, 1188
- subskrypcja zdarzeń, 149
- SvcConfigEditor.exe, 1051
- switch, 137
 - case, 137
- Switch, 1084
- synchronizacja danych, 71
- system pomocy, 312
- System.Activities, 177
- System.Activities.XamlIntegration, 1081
- System.AddIn, 175
- System.Array, 142
- System.Boolean, 130
- System.Byte, 130
- System.CodeDom, 175
- System.Collections, 143, 175
- System.Collections.Generic, 144
- System.ComponentModel, 175
- System.ComponentModel.Composition, 690
- System.ComponentModel.Composition.
 - ↳ExportAttribute, 690
- System.ComponentModel.Composition.Hosting, 690
- System.ComponentModel.Composition.Import
 - ↳Attribute, 690
- System.ComponentModel.IComponent, 289
- System.Configuration, 175
- System.Data, 175
- System.Decimal, 130
- System.Diagnostics, 175, 260
- System.Diagnostics.Contracts, 176
- System.Diagnostics.EventLog, 228
- System.Double, 130
- System.Drawing, 176
- System.Dynamic, 170, 176
- System.EnterpriseServices, 176
- System.Globalization, 176
- System.Int16, 130
- System.Int32, 130
- System.Int64, 130
- System.IO, 176
- System.Linq, 176
- System.Media, 176
- System.Messaging, 176
- System.Net, 176
- System.Net.PeerToPeer, 176
- System.NotImplementedException, 683
- System.Object, 130
- System.Runtime.CompilerServices, 157
- System.Runtime.Serialization, 1047
- System.Security, 176
- System.ServiceModel, 176
- System.Single, 130
- System.String, 130, 152
- System.Threading, 176
- System.Threading.Tasks, 176, 523
- System.Tuple, 144
- System.Web, 177
- System.Web.Mvc, 57, 807
- System.Web.Services.Protocols, 1041

System.Web.UI.WebControls, 989
 System.Web.UI.WebControls.WebParts, 776
 System.Windows, 177, 878
 System.Windows.Controls.Page, 924
 System.Windows.Data.Binding, 881
 System.Windows.Forms, 203, 895
 System.Windows.Forms.Control, 290
 System.Windows.Forms.Integration, 1143
 System.Windows.Forms.UserControl, 290
 System.Windows.Media.Imaging, 891
 System.Windows.Window, 924
 System.Workflow.Activities, 177
 System.Xml, 177
 System.Xml.Linq.XDocument, 165
 System.Xml.Linq.XElement, 165
 szablony, 44, 51, 329, 333
 instalacja, 339
 modyfikacja pliku .vstemplate, 338
 opcje eksportowania, 337
 parametry, 335
 tworzenie, 333
 szablony elementów, 340
 tworzenie, 340
 szablony internetowe, 330
 szablony procesu, 36
 szablony projektu testów, 399
 szablony witryny, 54, 716
 szablony XML, 264
 generowanie, 265
 okno projektowe, 267
 projektowanie, 265
 widoki szablonu, 266
 XML Schema Explorer, 266
 szablony XSD, 264
 szeregowanie zadań, 523

Ś

śledzenie stanu projektu, 38
 śledzenie zmian, 364
 środowisko IDE, 30, 88, 89
 Build, 91
 Class Designer, 94
 Customize, 95
 czcionki, 110
 Data, 92
 Debug, 91
 dokowanie okna, 106
 dostosowywanie edytorów, 100
 dostosowywanie pasków narzędzi, 95

Edit, 89
 edytor kodu języka C#, 98
 edytor kodu języka Visual Basic, 99
 edytory kodu, 98
 edytory tekstu, 98
 File, 89
 Format, 92
 graficzne okna projektowe, 102
 Help, 93
 konfiguracja, 79
 menu, 89
 nawigacja, 88, 94
 okna projektowe, 102
 okno narzędzi, 103
 opcje uruchomieniowe, 84
 otwieranie starszej aplikacji, 88
 pasek menu, 89
 paski narzędzi, 94
 poruszanie się między oknami, 108
 Project, 89, 91
 projekt, 85
 Properties, 104
 Query Designer, 94
 Refactor, 89, 90
 Solution Explorer, 97
 standardowy pasek narzędzi, 94
 strona startowa, 83
 Table, 92
 Test, 93
 Toolbox, 103
 Tools, 92
 View, 90
 Website, 90
 Window, 93
 zarządzanie ustawieniami, 80
 zarządzanie wieloma oknami, 105
 środowisko IDE Macros, 616

T

Tab Order, 841
 TabContainer, 919
 TabControl, 373, 855
 tabele, 942
 budowanie związków tabel, 948
 definiowanie kolumn, 945
 definiowanie struktury, 942
 edycja struktury, 942
 klucz główny, 945
 modyfikacja definicji, 948

- tabele
 - określanie klucza głównego, 945
 - wartości null, 945
 - wirtualne tabele, 955
 - związki tabel, 942
- tabele HTML, 286
- TabIndex, 753
- Table, 92, 792, 1167
- Table Designer, 946
- TableLayoutPanel, 836, 837, 855
- Tables and Columns, 949
- TableServiceContext, 1177
- Table-valued Function, 961
- tablice, 141
 - Array, 142
 - dostęp do elementów, 142
 - indeks, 141
 - inicjacja wartości elementów, 142
 - liczba elementów, 142
 - rozmiar, 141
 - tablice wielowymiarowe, 142
- tagi, 693
- tagi inteligentne, 1134, 1153
 - implementacja, 1153
 - SmartTagBase, 1153
- Target Framework, 87
- Task, 523
- Task List, 393, 578
 - obiektowy model automatyzacji, 579
 - zadania użytkownika, 396
 - zadania związane z komentarzami, 394
 - zadania związane ze skrótami, 395
- TaskItem, 563, 580
 - Delete, 579
 - składowe, 579
- TaskItems, 563, 580
 - Add, 579
- TaskItems2, 563
- TaskList, 563, 578, 579, 580
- technologia ASP.NET Ajax, 904
- technologia ClickOnce, 534
- technologia LINQ, 994
- technologia PLINQ, 995
- technologia Silverlight, 933
- technologia Windows Workflow, 1063
- Template, 51, 563
- Terminate All, 485, 493
- TerminateWorkflow, 1085
- Test, 93, 401
- Test List Editor, 435, 436
- Test Results, 407
- Test Runs, 409
- Test Settings, 413
- Test View, 407, 431, 432, 434
- TestClass, 415, 416
- TestCleanup, 416, 418
- TestContext, 413, 429
 - CurrentTestOutcome, 414
 - DataConnection, 414
 - DataRow, 414
 - RequestedPage, 414
 - TestDeploymentDir, 414
 - TestLogsDir, 414
 - TestName, 414
- TestInitialize, 416, 418
- TestMethod, 406, 416, 438
- testowanie, 398
- testowanie jednostek, 398
- testowanie usług WCF, 1049
- testowanie wyjątków, 422
 - ExpectedException, 422
- TestProperty, 416
- testy jednostek, 398, 399, 470
 - asercje, 420
 - AssemblyCleanup, 415
 - AssemblyInitialize, 415
 - Assert, 420
 - automatyzacja procesu tworzenia testów
 - jednostek, 402
 - Basic Unit Test, 401
 - ClassCleanup, 415, 418
 - ClassInitialize, 415, 418
 - Create Unit Tests, 403
 - Database Unit Test, 401
 - DataSource, 415
 - DeploymentItem, 416
 - ExpectedException, 416, 422
 - generowanie testów na podstawie istniejącego
 - końca, 402
 - HostType, 416
 - Ignore, 416
 - klasy asercji, 420
 - klasy atrybutów testów, 415
 - klasy testów, 400, 406
 - konfiguracja atrybutów, 431
 - konfiguracja opcji, 409
 - kontrola operacji inicjujących i porządkujących
 - na poziomie wykonywania testów, 419
 - kontrola ustawień testów, 411
 - Local.testsettings, 400

- menu Test, 401
- opcje projektów testów, 410
- operacje wykonywane po testach, 418
- operacje wykonywane przed testami, 418
- Ordered Test, 401
- organizowanie testów, 434
- piki .testsettings, 411
- pisanie testów jednostek, 405
- platforma, 413
- pliki z konfiguracją, 400
- projekt, 399, 400
- przeglądanie wyników testów, 407
- szablony projektów, 399, 401
- Test Results, 407
- Test Runs, 409
- Test Settings, 413
- Test View, 407, 434
- TestClass, 415, 416
- TestCleanup, 416, 418
- TestContext, 413
- TestInitialize, 416, 418
- TestMethod, 406, 416
- TestProperty, 416
- Timeout, 416
- TraceAndTestImpact.testsettings, 400
- tworzenie projektu, 399
- Unit Test, 401
- Unit Test Wizard, 401, 402, 411
- uruchamianie testów, 402
- uruchamianie testów jednostek, 407
- wczytywanie informacji o wcześniejszych testach, 402
- wygenerowany kod testów, 404
- wyniki testów, 407
- wyświetlanie przebiegów testów, 409
- zarządzanie ustawieniami procesu generowania testów, 404
- zarządzanie wynikami testów, 411
- testy jednostek ASP.NET, 428
 - AspNetDevelopmentServerHost, 428
 - atrybuty środowiska ASP.NET, 428
 - generowanie testów, 429
 - HostType, 428
 - konfiguracja hosta projektu testów, 430
 - UrlToTest, 428
- testy jednostek zależne od danych, 423
 - DataSource, 423, 425
 - łańcuch połączenia, 423
 - nawiązywanie połączenia przy użyciu pliku konfiguracyjnego, 424
 - połączenie z danymi, 423
 - stosowanie danych testowych w asercjach, 426
 - uruchamianie testów, 427
 - wyniki testów, 427
 - źródło danych, 425
- testy obciążenia, 39
- testy sieciowe, 39
- testy uporządkowane, 433
 - Ordered Test, 433
- testy wytrzymałości, 39
- TextBlock, 880
- TextBox, 274, 751, 792
- TextBoxWatermarkExtender, 919
- TextDocument, 563, 593, 594
 - CreateEditPoint, 599
 - składowe, 594
- TextPane, 563, 577
 - TryToShow, 577
- TextPane2, 563
- TextPoint, 596, 599
 - CreateInstance, 599
- TextSelection, 596
- TextWindow, 563, 576
 - ActivePane, 577
 - Panes, 577
- TFS, 36
 - automatyzacja procesu kompilacji, 37
 - dostęp do sieci WWW, 38
 - kontrola wersji, 37
 - raporty, 38
 - raporty ogólne, 38
 - szablon procesu, 36
 - Web Access, 38
 - zarządzanie projektem, 37
 - zarządzanie przypadkami testowymi, 37
 - zarządzanie wymaganiami, 37
- Then, 136
- this, 157
- ThisAddIn, 1142
- ThisDocument, 1150
- ThisWorkbook, 1150
- ThisWorkbook_Startup, 1150
- Threads, 485, 517
- Throw, 1086
- Thrown, 475
- time, 335
- Timeout, 416
- Timer, 274, 906
- TODO, 394
- Toggle All Outlining, 369

- Toggle Breakpoint, 483, 486
- Toggle Outlining Expansion, 369
- Toolbox, 67, 103, 578, 581, 830, 831
 - dodawanie elementów, 582
 - usuwanie elementów, 582
- ToolBox, 563, 578, 581
- ToolBoxItem, 563, 581
- ToolBoxItem2, 563
- ToolBoxItems, 581
 - Add, 581
- ToolBoxTab, 563
- ToolBoxTab2, 563
- ToolBoxTab3, 563
- ToolBoxTabs, 581
- ToolkitScriptManager, 919
- Tools, 92
- ToolStrip, 841, 844, 847, 850
- ToolStripContainer, 838, 847
- ToolStripItem, 841, 844
- ToolTip, 841
- ToolWindows, 660
- Top, 833
- TRACE, 206
- TraceAndTestImpact.testsettings, 400
- TransactedReceiveScope, 1090
- Transaction, 1087
- TransactionScope, 1087, 1089
- Transact-SQL, 956
- transakcje, 1087
- transformacje obrazów, 895
- TravelDataLib, 1098
- TravelManagerWeb, 1098
- TravelRequestService, 1098
- TreeNode Editor, 850
- TreeView, 849, 850, 905
 - edytor węzłów, 850
 - obrazy, 850
- Triggers, 912
- true, 138
- Try...Catch...Finally, 146
- tryb nazywania elementów, 373
- try-catch, 376
- TryCatch, 1086
- TryCreateInstance, 169
- TryGetMember, 169, 172
- TryInvokeMember, 169
- TrySetMember, 169
- T-SQL, 956, 961
- tunneling event, 883
- Tuple, 42, 144
 - Create, 144
- tworzenie
 - aplikacje ASP.NET MVC, 805
 - aplikacje biznesowe, 72
 - aplikacje do zarządzania procesem, 1096
 - aplikacje działające w chmurze, 64, 1170
 - aplikacje LINQ to SQL, 996
 - aplikacje okresowo nawiązujące połączenie, 71
 - aplikacje Silverlight, 59, 934
 - aplikacje Windows, 46
 - aplikacje WPF uruchamiane w przeglądarce, 922
 - aplikacje XBAP, 922
 - arkusze stylów, 758
 - baza danych, 943, 1099
 - biblioteka dostępu do danych, 1100
 - czynności Code Activity, 1106
 - czynności złożone, 1105
 - diagram bazy danych, 947
 - diagram klasy, 293
 - dodatki dla pakietu Office, 1136
 - dokumenty XSD, 265
 - dynamiczne aplikacje, 688
 - egzemplarz klasy, 131
 - egzemplarz nieistniejącej klasy, 156
 - elementy Web Part, 782
 - formularze sieciowe, 282, 285
 - formularze Windows, 826
 - fragmenty kodu, 385
 - interfejs sieciowy, 46
 - interfejs użytkownika, 822
 - katalogi zakładdek, 248
 - klasy, 293
 - klasy kontrolki użytkownika, 655
 - klient sieciowy, 53
 - klucze obce, 948
 - komponenty, 290
 - kompozycje, 771
 - kontroler, 814
 - kontrolki, 290, 853
 - kontrolki niestandardowe, 857
 - kontrolki użytkownika ASP.NET, 802
 - kreatory, 677
 - kreatory Add New Item, 680
 - lista testów, 436
 - makra, 345, 613
 - menu, 842
 - metody, 299
 - model, 812
 - model EDM, 69
 - model procesu, 72
 - niestandardowa wstążka pakietu Office, 1140
 - niestandardowe strona startowa, 310

- obiekty, 119, 153
 - obiekty bazy danych w kodzie zarządzanym, 970
 - obiekty dynamiczne, 169
 - okna dialogowe, 681
 - pakiet instalacyjny ClickOnce, 536
 - pakiety, 341
 - pakiety za pomocą narzędzia VSI, 342
 - panele operacji pakietu Office, 1149
 - pasek narzędzi, 844
 - pasek statusu, 847
 - połączenie z bazą danych, 943
 - procedury obsługi zdarzeń, 829
 - procedury składowane, 956
 - procedury składowane w C#, 971
 - proces, 1069
 - proces dynamiczny, 1080
 - proces oparty na komunikatach, 1090
 - proces oparty na XAML, 1069
 - proces typu Flowchart, 1094
 - proces za pomocą szablonu Code Activity, 1075
 - projekt, 85, 194, 333
 - projekt aplikacji sieciowej, 715
 - projekt aplikacji Windows, 272
 - projekt bazy danych, 962
 - projekt dekoracji, 702
 - projekt dodatku, 637
 - projekt instalacyjny aplikacji dla systemu Windows, 541
 - projekt testów, 399, 400
 - projekt VSIX, 350
 - projekt witryny internetowej, 197
 - projekt WPF, 280
 - punkt edycji, 599
 - regiony, 368
 - regiony formularzy aplikacji Outlook, 1144
 - rozszerzenia dokumentów Office, 1147
 - rozszerzenia edytora, 701
 - rozwiązania, 183
 - rozwiązania WPF, 48
 - schematy kodu, 367
 - skórki, 772
 - strony ASP.NET, 737
 - strony obsługujące częściową aktualizację, 907
 - strony oparte na stronie wzorcowej, 57
 - strony Web Part, 778
 - strony wzorcowe, 56, 765
 - struktury, 125
 - style, 756
 - szablony elementów, 340
 - szablony projektów, 333
 - szablony XML, 265
 - tabele, 942
 - testy jednostek, 406
 - testy jednostek zależne od danych, 423
 - testy uporządkowane, 433
 - usługi sieciowe ASP.NET, 1020
 - usługi WCF, 74, 75, 1045, 1104
 - widok, 817
 - widoki (baza danych), 955
 - witryny, 907
 - witryny ASP.NET, 715
 - wyjątki SOAP, 1041
 - wyjątki usługi sieciowej, 1040
 - wyliczenia, 120
 - wyzwalacze, 960
 - zapytania SQL, 951
 - zapytania typu Management Event, 227
 - zbiory danych, 67
 - zdarzenia, 147
 - zrzut informacji diagnostycznych, 527
 - TypeCatalog, 691
 - typed dataset, 67
 - typy danych, 130
 - dynamic, 167
 - dynamiczne typy danych, 167
 - konwersja, 131
 - łańcuchy znaków, 130
 - obiekty, 130
 - struktury, 130
 - typy anonimowe, 156
 - typy ogólne, 143, 144, 161
 - typy proste, 130
 - typy MIME, 931
 - typy projektów, 195
 - typy złączeń, 954
- ## U
- UDDI, 1016
 - udostępnianie interaktywnych aplikacji w różnych systemach, 933
 - udostępnianie ustawień dodatku, 662
 - UIHierarchyItem, 620, 621
 - układ kontrolek, 831
 - układ strony ASP.NET, 747
 - opcje pozycjonowania kontrolki, 749
 - pozycjonowanie bezwzględne, 748, 751
 - pozycjonowanie względne, 748
 - pozycjonowanie za pomocą stylów CSS, 748
 - układ płynny, 748, 753

- układ tabelaryczny, 275
- ukrywanie składowych, 123
- UML, 297
- Undo, 94
- UNDONE, 394
- Uniform Resource Identifier, 1016
- uniform resource locator, 1016
- Unit Test, 401
- Unit Test Wizard, 401, 402, 411
- UnitTesting.Web, 428
- Universal Description, Discovery, and Integration, 1016
- Unload, 744
- Unload Macro Project, 629
- until, 140
- Until, 140
- UPDATE, 955, 992
- Update Service Reference, 1037
- UpdatePanel, 58, 906, 908
- UpdatePanelTrigger Collection Editor, 912
- UpdateProgress, 58, 906, 913
- UpdateQuery, 992
- uproszczone deklarowanie właściwości, 166
- URI, 1016, 1031, 1042
- UriSource.LocalPath, 893
- URL, 1016
- UrlRoutingModule, 807
- UrlToTest, 428
- uruchamianie
 - aplikacje MVC, 820
 - aplikacje w chmurze, 64
 - aplikacje w trybie diagnozowania, 470, 471, 472
 - aplikacje XBAP, 925
 - kod, 260
 - kreator, 683
 - narzędzie do refaktoryzacji, 441
 - sesje diagnostyczne, 482
 - szablon XSLT dla dokumentu XML, 269
 - testy jednostek, 407
 - testy zależne od danych, 427
 - usługi WCF, 1049
 - zdalne diagnozowanie, 513
- uruchamianie makra, 613, 614, 629
 - makra z parametrami, 632
 - menu, 630
 - okno poleceń, 632
 - paski narzędzi, 630
 - skrótów klawiaturowe, 631
- urządzenia mobilne, 45
- user control, 853
- UserControl, 835
- userdomain, 335
- UserLookupActivityLib, 1098
- username, 335
- using, 128, 367
- Using, 153, 380
- usługa WAS, 1118
- usługi, 71, 74
- usługi sieciowe, 75, 514, 1012
 - format danych, 1012
 - odkrywanie, 1013
 - SOAP, 75, 1012
 - wywoływanie, 1037
 - XML, 1012
- usługi sieciowe ASP.NET, 1012, 1014, 1015
 - atrybut WebMethod, 1027
 - atrybut WebService, 1027
 - definiowanie referencji sieciowej, 1033
 - DISCO, 1016
 - formalny opis usługi sieciowej, 1029
 - HTTP, 1016, 1031
 - interfejsy, 1026
 - kod metod sieciowych, 1025
 - konsumowanie, 1033
 - metody, 1015
 - odpowiedź SOAP, 1032
 - opis, 1029
 - pliki usługi sieciowej, 1018
 - pośrednik usługi, 1036
 - projekt, 1017
 - SOAP, 1016, 1031
 - SoapException, 1040
 - sprawdzanie usługi sieciowej, 1031
 - tworzenie, 1020
 - tworzenie wyjątku usługi sieciowej, 1040
 - UDDI, 1016
 - URI, 1016
 - web.config, 1018
 - WebMethod, 1027
 - WebService, 1027
 - WebServiceAttribute, 1027
 - WSDL, 1016, 1029
 - WSE, 1016
 - wyjątki, 1039
 - wyświetlanie metody sieciowej, 1031
 - wywoływanie metody sieciowej, 1032
 - wywoływanie usługi sieciowej, 1037
 - XML, 1016
 - XSD, 1016
 - żądanie SOAP, 1031

usługi uruchomieniowe procesu, 1064
 usługi WAS, 76
 usługi WCF, 74, 75, 717, 1012, 1014, 1041, 1104

- adres, 1042
- adres punktu końcowego, 1053
- diagnozowanie, 514
- działania, 1042
- elementy wiązania, 1042
- host, 1042, 1049
- hosting, 1058
- kanały, 1042
- klient, 1042
- konfiguracja, 76, 1050
- konsumowanie, 74, 76, 1055
- kontrakt, 1042
- pliki usług, 1044
- projekt, 1043
- punkt końcowy, 1042
- testowanie, 1049
- tworzenie, 74, 75, 1045
- URI, 1042
- uruchamianie, 1049
- wiązanie, 1042, 1054

 ustawianie

- aktywne okno, 575
- filtry punktów przerwania, 502
- punkty przerwania, 258, 478, 494
- punkty śledzenia, 504
- warunki punktu przerwania, 500

 usuwanie

- nieużywane argumenty z metod obsługi zdarzeń, 166
- nieużywane instrukcje Using, 380
- zadania, 579

 utrwalanie procesu, 1116
 uwierzytelnianie użytkowników, 796
 uzupełnianie słów, 375

V

Validate, 742
 ValidationSummary, 793, 794
 Variant, 152
 VBA, 1130
 VBScript, 526
 VendorReservationService, 1098
 VerticalAlignment, 886
 View, 90
 View Call Hierarchy, 248
 View Class Diagram, 293
 ViewportAdornment1, 698
 ViewportAdornment1Factory, 698
 ViewResult, 808
 ViewState, 742, 809
 virtual, 122
 VirtualPoint, 596, 601
 Visual, 863, 879
 Visual Basic, 41

- automatyczne generowanie kodu do obsługi właściwości, 166
- kontynuowanie wiersza, 167
- usuwanie nieużywanych argumentów z metod obsługi zdarzeń, 166

 Visual Basic for Applications, 1130
 Visual C#, 41
 Visual C++, 41
 Visual Database Tools, 942
 Visual F#, 41
 Visual Studio 2010, 23, 30

- edycje, 31
- strona startowa, 83
- subskrypcja pakietu MSDN, 35

 Visual Studio 2010 Express, 32
 Visual Studio 2010 Image Library, 1140
 Visual Studio 2010 Premium, 31, 34
 Visual Studio 2010 Professional, 24, 31, 33
 Visual Studio 2010 SDK, 349
 Visual Studio 2010 Ultimate, 31, 34
 Visual Studio Add-in, 637
 Visual Studio Agents 2010, 39
 Visual Studio Code Gallery, 310
 Visual Studio Content Installer, 347
 Visual Studio Debugger, 472
 Visual Studio Export Template Wizard, 333
 Visual Studio Extensibility, 693
 Visual Studio Extension, 341
 Visual Studio Extension Manager, 341
 Visual Studio Gallery, 342
 Visual Studio Installer, 342, 540
 Visual Studio SDK, 693

- Editor Classifier, 695
- szablony projektu, 694

 Visual Studio Team Explorer 2010, 38
 Visual Studio Test Professional 2010, 38
 Visual Studio Tools for Office, 1130
 Visual styles, 206
 Visual Web Developer 2010 Express, 33
 void, 117
 VScloudService.exe, 1161, 1164
 vsCMElement, 571

vsCommandExecOption, 652
vsCommandExecOptionDoDefault, 652
vsCommandExecOptionDoPromptUser, 652
vsCommandExecOptionPromptUser, 652
vsCommandExecOptionShowHelp, 652
VSContent, 342
 atrybuty, 344
 Attribute, 344
 Attributes, 344
 ContentVersion, 344
 definiowanie pliku, 345, 346
 Description, 343
 DisplayName, 343
 FileContentType, 343
 FileName, 343
 węzły, 343
VSContentInstaller.exe, 347
VSI, 342
 plik VSContent, 342
 tworzenie pakietów, 342
VSIX, 341, 349
 Copy to Output Directory, 351
 dodawanie rozszerzenia, 351
 edytor manifestów, 352
 kompilacja projektu, 353
 manifest, 352
 projekt, 350
 rozpowszechnianie, 354
 szablon projektu, 351
 testowanie procesu instalacji, 354
vsPaneShowHow, 577
vstemplate, 336, 338, 344
VSTest, 428
VSTO, 1130
 dodatki dla pakietu Office, 1136
 niestandardowe panele zadań, 1143
 obsługa zdarzeń kontrolki, 1141
 pamięć podręczna danych, 1152
vsToolBoxItemFormat, 581
vsToolBoxItemFormatDotNETComponent, 582
VSTS Database Edition, 962

W

W3C's Web Content Accessibility Guidelines, 734
wariancja, 174
wariancja delegatów, 174
warstwa prezentacji, 860
wartości null, 945
wartości zmiennych, 505
warunki punktu przerwania, 500

WAS, 76, 1059, 1118
Watch, 484, 507
wątki, 515, 523
WCAG, 734
WCF, 23, 75, 176, 1012, 1014, 1041
 projekt usługi, 75
WCF Service, 717, 728
WCF Service Application, 75, 1043, 1044, 1049, 1050
WCF Service Host, 1049
WCF Service Library, 1049, 1050, 1051, 1055
WCF Service Web Role, 1167
WCF Workflow Service, 1068, 1069
WCF Workflow Service Application, 1043, 1065
wczytywanie danych, 160
wdrażanie aplikacji działających w chmurze, 1199
 Publish Cloud Service, 1202
 Visual Studio, 1202
 wdrażanie w środowisku testowym, 1200
wdrażanie aplikacji na platformę Azure, 1174
wdrażanie procedur składawanych, 974
wdrażanie rozwiązań po stronie klienta, 534
 ClickOnce, 534
 instalator systemu Windows, 535
Web, 1164
Web 2.0, 904
Web Configuration File, 728
Web Deployment Tool, 549, 550
 Deploy SQL, 551
 Include All Databases Configured in Deploy SQL
 Tab, 551
 One-Click Publishing, 553, 554
 opcje wdrażania baz SQL Server, 551
 opcje wdrażania rozwiązań sieciowych, 551
 Package/Publish, 550
 Publish Web, 554
 skrypty SQL, 552
 uruchamianie procesu publikowania, 553
Web Form, 727
web forms, 737
Web Part, 776
Web Platform Installer, 33
Web Reference Name, 1035
Web Role, 64
Web Service, 727
Web Service Description Language, 1016
Web Service Enhancements, 1016
Web Site, 907, 1017
Web Site Administration Tool, 796
Web User Control, 727, 802
web.config, 55, 716, 717, 1018, 1050
WebForm, 44

- WebMethod, 1027
- webnamespace, 335
- WebPartManager, 777, 779
- WebPartManagerMain.Personalization.
 - ↳ ResetPersonalizationState(), 788
- WebPartZone, 777, 779
- WebRole.cs, 1170
- WebService, 1020, 1027
- WebServiceAttribute, 1027
- Website, 90
- wersje Visual Studio 2010, 31
- WF, 23, 72, 177, 1062, 1063
- When Breakpoint Is Hit, 504
- When Hit, 503, 504
- Where, 163
- WHERE, 909
- while, 140
- While, 140, 1083, 1084
- wiadomości związane ze środowiskiem, 306
- wiązanie danych, 861, 909, 974
 - aplikacje sieciowe, 989
 - aplikacje WPF, 986
 - automatyczne generowanie związanych
 - kontrolki Windows Forms, 976
 - DataGridView, 983
 - DataList, 990
 - DetailsView, 989
 - dostosowywanie odwzorowania źródła danych, 980
 - FormView, 989
 - GridView, 989, 990
 - kontrolki sieciowe, 989
 - kontrolki źródła danych, 993
 - modyfikacja zapytania dla źródła danych, 985
 - modyfikacja zbiorów danych o określonym
 - typie, 981
 - odwzorowywanie źródeł danych na kontrolki, 978
 - proces wiązania danych, 975
 - Repeater, 989
 - ręczne wiązanie kontrolki formularzy
 - Windows, 982
 - wiązanie kontrolki, 986
 - wybór obiektów źródła danych, 978
 - wybór źródła danych, 976
 - XAML, 987
 - złożone wiązanie danych, 975
 - zmiana odwzorowywania danych tabeli, 979
- wiązanie danych w WPF, 881
 - DataContext, 882
 - kontekst danych, 882
 - System.Windows.Data.Binding, 881
- widok, 806, 807
- widok Split, 56
- widoki (baza danych), 955
- wielozadaniowość, 523
- Win32_ComputerSystem, 226
- Win32_Desktop, 226
- Win32_LogicalDisk, 226
- Win32_NetworkAdapter, 226
- Win32_NetworkConnection, 226
- Win32_NTEventLogFile, 226
- Win32_OperatingSystem, 226
- Win32_Printer, 226
- Win32_Process, 226
- Win32_Processor, 226
- Win32_Product, 226
- Win32_Service, 226
- Win32_Share, 226
- Win32_SystemAccount, 226
- Win32_Thread, 226
- Window, 93, 563, 572, 573, 592
 - CommandBars, 590
 - LinkedWindows, 586
 - składowe, 573
 - WindowFrame, 586
- Window2, 563
- WindowHiding, 627
- Windows, 563, 588
- Windows Application, 203, 826, 976
- Windows Azure, 63, 1158
- Windows Azure Activity Log, 1204
- Windows Azure Cloud Service, 1165, 1166, 1170
- Windows Communication Foundation, 23, 74
- Windows Control Library, 854
- Windows Form Application, 46
- Windows Forms, 272, 822
- Windows Forms Designer, 275
- Windows Forms Designer Options, 275
- Windows Forms User Control, 1141
- Windows Live, 318
- Windows Management Instrumentation, 226
- Windows Presentation Foundation, 23, 44, 271, 860
- Windows Process Activation Service, 76, 1059, 1118
- Windows Workflow, 72, 1062, 1063
 - aplikacje klienckie, 1064
 - CodeActivity, 1063
 - czynności Collection, 1086
 - czynności Control Flow, 1083
 - czynności Error Handling, 1086
 - czynności Flowchart, 1094
 - czynności Messaging, 1089

- Windows Workflow
 - czynności Primitives, 1084
 - czynności procesów, 1082
 - czynności Runtime, 1084
 - czynności związane z komunikatami, 1089
 - komunikaty, 73
 - NativeActivity, 1063
 - obsługa transakcji, 1087
 - okno projektowe procesów, 1067
 - pamięć trwała, 1117
 - proces, 72, 1063
 - proces hosta, 1063
 - projektowanie procesu, 73
 - projekty, 1065
 - silnik uruchomieniowy procesu, 1064
 - składniki procesu, 1063
 - sposoby komunikowania się z pamięcią trwałą, 1118
 - stan, 73
 - sterowanie przepływem, 1083
 - szablony elementów procesów, 1068
 - szablony projektów, 1065
 - tworzenia procesów, 1069
 - usługi uruchomieniowe procesu, 1064
 - zarządzanie komunikatami, 1089
 - zbiory działań, 72
- Windows Workflow Foundation, 23, 1062, 1063
- Windows2, 563, 659
 - CreateLinkedWindowFrame, 586
- WindowsBase.dll, 860
- WindowsDefaultBounds, 828
- WindowsDefaultLocation, 828
- WindowVisibilityEvents, 627, 628
- WinForm, 44
- WinForms, 271
- With, 153
- witryny ASP.NET, 54, 714, 715
 - ASP.NET Development Server, 719
 - ASP.NET Reports Web Site, 717
 - ASP.NET Web Service, 716
 - ASP.NET Web Site, 716
 - dodawanie plików, 728
 - dodawanie stron internetowych, 737
 - Dynamic Data Entities Web Site, 717
 - Dynamic Data Linq to SQL Web Site, 717
 - Empty Web Site, 717
 - IIS, 718
 - język programowania, 724
 - katalogi, 725
 - kompozycje, 770
 - komunikacja z serwerem sieciowym przy użyciu protokołu HTTP, 723
 - lokalny egzemplarz IIS, 720
 - mapa witryny, 728
 - opcje projektu, 729
 - pliki, 727
 - położenie witryny, 717
 - serwer roboczy, 719, 720
 - skórki, 728, 770
 - SSL, 722
 - Starter Kit, 717
 - struktura witryny, 725
 - szablony, 716
 - tworzenie, 715
 - tworzenie jednolitego wyglądu i zachowania, 754
 - udostępnianie witryny przy użyciu protokołu FTP, 722
 - WCF Service, 717
 - właściwości, 729
- Wizard, 793
- WizardDialog, 681
 - Category, 681
 - ClassName, 681
- wizardResult, 679
- wizardResultBackOut, 679
- wizardResultCancel, 679
- wizardResultFailure, 679
- wizardResultSuccess, 679
- wizualizacja danych, 510
- wkład w społeczność, 332
 - pakiety startowe, 333
 - szablony, 333
 - tworzenie szablonów projektów, 333
- wkraczanie w kod, 488, 489
- właściwości, 115, 166
 - definiowanie, 298
- właściwości formularzy Windows, 827
- właściwości konfiguracji kompilacji, 193
- właściwości projektu, 204
 - Application, 205
 - Build, 206
 - Build Events, 206
 - Compile, 207
 - Debug, 207
 - Publish, 208
 - Reference Paths, 208
 - References, 208
 - Resources, 208
 - Security, 208
 - Settings, 208
 - Signing, 209

- właściwości rozwiązania, 190
- włączanie diagnozowania stron internetowych, 470
- WMI, 226, 227
- Word 2007 Document, 1136
- Word 2010, 51
- Workbook, 51
- Worker, 1164
- Worker Role, 1167
- workflow application, 1062
- Workflow Console Application, 1065, 1066, 1069
- WorkflowApplication, 1074
- WorkflowInvoker, 1074
- WorkflowInvoker.Invoke, 1074, 1080, 1081
- WorkflowServiceHost, 1075
- WPF, 23, 44, 47, 61, 62, 177, 271, 280, 860, 923
 - animacja, 862
 - Application, 863
 - architektura platformy, 861
 - biblioteka klas, 863
 - Button, 863
 - Canvas, 871
 - Control, 863
 - ControlTemplate, 879
 - DispatcherObject, 863
 - DockPanel, 872
 - DocumentReader, 862
 - dostosowywanie wyglądu kontroltek, 878
 - FlowDocumentReader, 862
 - formularze, 49, 874
 - Grid, 874
 - Grid.ColumnDefinitions, 874
 - język XAML, 863
 - kontrolki kontenerowe, 871
 - media, 861
 - model programowania, 863
 - obsługa czcionek, 862
 - obsługa dokumentów, 862
 - okna dialogowe, 874, 875
 - okno projektowe, 865
 - panele, 871
 - połączenie kontrolki ze źródłem danych, 881
 - projekt, 48
 - przypisywanie stylu, 879
 - StackPanel, 876
 - style, 878
 - Style, 878
 - szablony, 878, 879
 - TextBlock, 880
 - tworzenie aplikacji, 871
 - tworzenie kodu, 48
 - tworzenie układu, 871
 - Visual, 863, 879
 - wiązanie danych, 861, 881
 - WrapPanel, 877
 - współdziałanie z formularzami Windows, 861
 - wygląd kontroltek, 862
 - wypełnienie gradientem, 879
 - XAML, 48
 - zdarzenia przekazywane, 883
 - zmiana szablonu przycisku, 880
 - znaczniki, 48
- WPF Application, 48, 280, 866
- WPF Browser Application, 48, 923
- WPF Custom Control Library, 49
- WPF User Controls, 49
- wpisywanie kodu, 363
- WrapPanel, 871, 877, 885
- WriteLine, 1085
- WS-*, 1016
- WSAT, 796
- WSDL, 1012, 1016, 1029
- WSDL.exe, 1030
- WSE, 1016
- wskazówki, 841
- wskazówki dotyczące problemów, 364
- wspólna specyfikacja języka, 40
- wspólne środowisko uruchomieniowe, 40
- wspólny system typów, 40
- współużytkowane zasoby, 328
 - instalacja udostępnianych zasobów, 330
 - przechowywanie udostępnianych zasobów, 330
 - rodzaje współużytkowanych danych, 329
 - wyszukiwanie szablonów według technologii, 329
 - wyszukiwanie zasobów, 328
- współużytkowanie makr, 618
- wstawianie fragmentów kodu, 381
- wstawianie komentarzy do okna z tekstem, 601
- wstążka, 1133
 - dodawanie elementów, 1139
 - elementy, 1139
 - grupy, 1139
 - modyfikacja, 1137
 - obsługa zdarzeń kontroltek, 1141
 - powierzchnia projektowa, 1138
 - Ribbon (Visual Designer), 1137
 - Ribbon (XML), 1137
 - zakładki, 1139
- wstrzymywanie kodu, 260
- wybór docelowego środowiska, 86
- wybór katalogu, 895

wybór źródła danych, 976, 977, 990
 wygląd kontrolkek, 840
 wygląd strony w przeglądarce, 288
 wyjątki, 146, 422, 474
 usługi sieciowe, 1039
 wykonywanie poleceń, 606
 obiektowy model automatyzacji, 583
 wykresy, 800, 801
 wykrywanie typu zmiennej na podstawie przypisania, 151
 wyliczenia, 120
 wyniki testów jednostek, 407
 testy jednostek zależne od danych, 427
 wyniki zapytania SQL, 953
 wyodrębnianie interfejsów, 459
 Extract Interface, 459
 wyodrębnianie metod, 450, 451
 Extract Method, 451
 generowanie szkieletu metody, 457
 wyodrębnianie pojedynczych wierszy kodu, 456
 wyrażenia lambda, 162, 163
 WYSIWYG, 55
 wyspy danych, 1150, 1151
 wyszukiwanie przyrostowe, 257, 577
 wyszukiwanie symboli, 256
 wyszukiwanie szablonów według technologii, 329
 wyszukiwanie tekstu w plikach, 253
 wyszukiwanie testów, 435
 wyświetlanie danych, 849
 dane hierarchiczne, 849
 dane tabelaryczne, 852
 DataGridView, 852
 ImageList, 850
 TreeView, 849
 wyświetlanie
 dokumenty tekstowe, 596
 kontrolki użytkownika, 660
 metody sieciowe, 1031
 okna narzędzi, 660
 podpowiedzi DataTip, 508
 referencje sieciowe, 1036
 wywoływanie
 makra, 629
 metody sieciowe, 1032
 metody z klasy bazowej, 122
 procesy, 1074
 usługi sieciowe, 1037
 wyzwalacze, 960
 tworzenie, 960

wzbogacanie pakietu Microsoft Office, 1149
 wzbogacanie środowiska, 688
 wzorzec MVC, 805

X

XAML, 44, 48, 53, 61, 271, 280, 863, 866
 procesy, 1069
 składnia, 864
 wiązanie danych, 987
 współpraca między członkami zespołu projektowego, 865
 XAML Browser Application, 48, 922
 XBAP, 48, 904, 922, 923
 Xbox, 45
 XDocument, 165
 XDR, 264, 265
 XElement, 165
 XHTML, 55, 288
 XML, 164, 165, 993, 1016
 XML (kontrolka ASP.NET), 793
 XML Schema Document, 1016
 XML Schema Explorer, 265, 266
 XML Schemat Document, 265
 XML-Data Reduced, 264
 XmlDataSource, 800
 XMLDataSource, 993
 XmlSerializer, 1151
 xml-stylesheet, 270
 XNA, 45
 Xor, 136
 XSD, 264, 265, 343, 1016
 XSLT, 269

Y

year, 335

Z

zachowanie kontrolkek, 840
 zadania, 523, 579
 przełączanie stanu ukończenia zadań, 580
 zadania użytkownika, 396
 zadania związane z komentarzami, 394
 HACK, 394
 niestandardowe znaczniki w komentarzach, 395
 TODO, 394
 UNDONE, 394

- zadania związane ze skrótami, 395
- zagnieżdżanie kontrolek użytkownika, 855
- zagnieżdżanie stron wzorcowych, 769
- zakładki, 247, 594
 - dodawanie, 247
 - katalogi, 248
- zależności projektu, 191
- zapisywanie
 - komunikat XML, 165
 - obrazy, 891
 - proces, 1077
- zapytania LINQ, 156, 160, 994
 - Entity Framework, 1006
 - funkcje anonimowe, 162
 - operatory, 161
 - Order by, 163
 - Select, 163
 - Where, 163
 - wyrażenia lambda, 162, 163
- zapytania SQL, 951
 - DELETE, 955
 - diagram, 953
 - dopracowywanie zapytań, 953
 - INSERT, 954, 955
 - JOIN, 954
 - kod SQL, 953
 - kryteria, 953
 - SELECT, 953, 955
 - tworzenie, 951
 - tworzenie tabel, 955
 - UPDATE, 955
 - wyniki, 953
 - złączenia, 954
- zarządzanie aplikacją, 226
- zarządzanie arkuszami stylów, 287
- zarządzanie cyklem życia aplikacji, 36
- zarządzanie dodatkami, 653
- zarządzanie etykietami punktów, 499
- zarządzanie grupami elementów, 141
- zarządzanie komunikatami, 1089
- zarządzanie makrami, 614
- zarządzanie projektami, 37, 218
- zarządzanie projektami baz danych, 969
- zarządzanie przypadkami testowymi, 37
- zarządzanie punktami przerwania, 496, 498
- zarządzanie rozwiązaniami, 217
- zarządzanie stylami CSS, 287, 760
- zarządzanie ustawieniami systemu pomocy, 312
- zarządzanie ustawieniami środowiska IDE, 80
- zarządzanie wieloma oknami środowiska IDE, 105
- zarządzanie wymaganiami, 37
- zarządzanie wynikami testów, 411
- zarządzanie zabezpieczeniami, 226
- zarządzanie zdalnymi usługami, 225
- zasady stylów, 755
- zasoby współużytkowane, 328
- zatrzymywanie kodu, 260
- zaznaczanie tekstu, 237
- zbiory działań, 72
- zbiór danych, 67
 - dostęp do danych, 67
 - tworzenie, 67
- zbiór danych o określonym typie, 981
- zdalne diagnozowanie, 513
 - monitor zdalnego diagnozowania, 513
 - msvsmon.exe, 513
 - rdbsetup.exe, 513
 - Remote Debugging Monitor, 513
 - uprawnienia, 513
- zdarzenia, 147, 227
 - aplikacje Windows, 47
 - ASP.NET, 741
 - definiowanie, 148, 298
 - DTEEvents, 628
 - EventArgs, 148
 - EventHandler, 147, 149
 - formularze Windows, 827, 828
 - makra, 623
 - obiektowy model automatyzacji, 608
 - obsługa, 149
 - OnPaint, 857
 - strony ASP.NET, 743
 - subskrybowanie, 149
 - tworzenie, 147
 - zgłaszanie, 149
- zdarzenia przekazywane, 883
 - zdarzenia bezpośrednie, 883
 - zdarzenia przesyłane w dół, 883
 - zdarzenia przesyłane w górę, 883
- zegar, 274
- zestaw testów, 398
- zgłaszanie błędów, 323, 324
- zgłaszanie zdarzeń, 147, 149
- Zip to Exe Conversion Tool, 349
- złączenia, 954
- złożone wiązanie danych, 975
- zmiana
 - format przechowywania projektu makr, 619
 - kolejność parametrów, 463
 - rozmiar kontrolek, 277
 - wersja platformy .NET, 87

zmiana nazw, 446

Object Browser, 447

Rename, 447, 449

Solution Explorer, 447

zmienne, 130

deklaracja, 131

niejawne określanie typu, 151

rzutowanie, 132

wykrywanie typu zmiennej na podstawie
przypisania, 151

zmień i kontynuuj, 511

znaczniki HTML, 285

zrzut informacji diagnostycznych, 527

Zune, 71

związki tabel, 948

jeden do jednego, 950

wiele do wielu, 950

związki zwrotne, 950

Ż

źródło danych, 852, 909, 976

AccessDataSource, 993

Database, 977

dostosowywanie odwzorowania, 980

kontrolki, 993

ObjectDataSource, 993

SiteMapDataSource, 993

SqlDataSource, 993

wybór obiektów, 978

wybór źródła danych, 976

XMLDataSource, 993

Ż

żądania HTTP, 807

żądania SOAP, 1031

Microsoft Visual Studio 2010

Visual Studio 2010 to najnowsza wersja środowiska programistycznego firmy Microsoft. Każdy programista, który zdecydował się wykorzystać bogaty zestaw zgromadzonych tu narzędzi, osiągnie maksymalną produktywność w pracy i będzie mógł tworzyć kod działający w systemie Windows oraz w sieci. Dodatkowo będzie mieć do dyspozycji technologię Silverlight i możliwość budowania aplikacji w chmurze z użyciem platformy Microsoftu — Azure. Na tym jednak nie koniec. W Visual Studio 2010 pojawiły się kolejne innowacje. Znajdziesz tu nowy edytor kodu, oparty na platformie WPF, rozszerzenia środowiska IDE związane z platformą MEF oraz możliwość pisania skryptów za pomocą technologii Silverlight. Ponadto języki platformy .NET wzbogacono o obsługę programowania dynamicznego. Pojawił się nowy język F#, służący do programowania funkcyjnego, oraz mechanizmy upraszczające szybkie pisanie kodu wyższej jakości.

„Microsoft Visual Studio 2010. Księga eksperta” zawiera kompletne omówienie środowiska Visual Studio 2010, a skoncentrowanie się na wersji Professional pozwoliło autorom na stworzenie opisu bardziej szczegółowego niż kiedykolwiek wcześniej. Dzięki temu podręcznikowi nauczysz się w pełni wykorzystywać możliwości platformy .NET, w tym technologii WPF (pozwalającej na tworzenie bogatych klientów), technologii WCF (stworzonej do budowania dynamicznych rozwiązań opartych na usługach) czy też technologii WF (umożliwiającej rozwijanie strukturyzowanych programów na podbitanie procesów biznesowych). Znajdziesz tu także omówienie nowych narzędzi Microsoftu, przeznaczonych do testowania, instrumentacji aplikacji i analizowania kodu.

- Języki platformy .NET
- Przeglądarki i eksploratory
- Tworzenie projektów WPF
- Scałbony XML
- Testowanie, refaktoryzacja i diagnozowanie kodu
- Platforma MEF
- Tworzenie aplikacji ASP.NET
- Bogate i inteligentne interfejsy użytkownika
- Dołączone aplikacje internetowe
- Silverlight 4.0
- Aplikacje biznesowe oparte na pakiecie Office
- Technologia Windows Azure i aplikacje działające w chmurze

Wykorzystaj zdobytą wiedzę i zostań mistrzem programowania!

Mike Snell zawodowo pomaga zespołom budować oprogramowanie dla przedsiębiorstw i tworzyć aplikacje komercyjne. Kieruje działem Solutions w CII (www.cniamerica.com). Wraz z zespołem, którym zarządza, oferuje klientom usługi z zakresu konsultacji, mentoringu i rozwijania architektury. Mike jest też dyrektorem regionalnym Microsoftu (www.TheRegion.com).

Lars Powers jest głównym doradcą technicznym do spraw ISV w zespole Developer and Platform Evangelism Microsoftu. Współpracuje z najważniejszymi partnerami tej korporacji z zakresu ISV i pomaga im tworzyć rozwiązania przy użyciu technologii nowej generacji. Przed związaniem się z Microsoftem Lars był niezależnym konsultantem do spraw szkoleń i doradztwa na temat platformy .NET.

W katalogu: 5849



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



Zamówienia telefoniczne:
0 801 339900



0 601 339900



Helion

Sprawdź najnowsze promocje:
@ facebook.com/helion
Książki najchętniej czytane:
@ twitter.com/helionbestseller
Zamów informacje o nowościach:
@ <http://helion.pl/newsletter>

Helion Sp. z o.o.
ul. Rakowiecka 1c, 44-100 Gliwice
tel.: 22 230 18 43
e-mail: helion@helion.pl
<http://helion.pl>

hellon.pl
Księgarnia
Internetowa

Cena: 199,00 zł

ISBN 978-83-246-3029-5



9 788324 630295

Informatyka w najlepszym wydaniu