

Zbigniew Fryźlewicz
Daniel Nikończuk



Windows Azure

Wprowadzenie
do programowania
w chmurze



Programuj w chmurze... i wznies się wysoko!

Autorstwo: Zbigniew Fryźlewicz (wstęp, rozdziały 1–7, dodatki),
Daniel Nikończuk (wstęp, rozdziały 1–7, dodatki).

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Ewelina Burska

Projekt okładki: Jan Paluch

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie?winazu>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody aplikacji opisanych w książce można znaleźć pod adresem:
<ftp://ftp.helion.pl/przyklady/winazu.zip>

ISBN: 978-83-246-3703-4

Copyright © Helion 2012

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wstęp	7
Ewolucja infrastruktury IT oraz sposobu wytwarzania i używania aplikacji	8
Podstawowe modele usług chmury	9
Wady i zalety chmury	12
Typowe scenariusze biznesowe	14
Odbiorcy książki	15
Rozdział 1. Platforma Windows Azure	17
1.1. Windows Azure	19
1.1.1. Kontroler zarządzania	19
1.1.2. Obliczenia	19
1.1.3. Dane	20
1.1.4. Sieć CDN	21
1.2. Windows Azure AppFabric	21
1.3. SQL Azure	21
1.4. Opłaty za korzystanie z platformy	22
Rozdział 2. Magazynowanie danych w Windows Azure	25
2.1. Azure Storage	25
2.1.1. Bezpieczeństwo danych w Azure Storage	26
2.1.2. Azure Tables	27
2.1.3. Azure Blobs	30
2.1.4. Azure Queues	33
2.2. SQL Azure	36
2.2.1. Architektura SQL Azure	37
2.2.2. Protokoły dostępu do SQL Azure	39
2.2.3. Ograniczenia SQL Azure	39
2.3. Dysk wirtualny w Azure Blobs	42

Rozdział 3. Przygotowania do utworzenia pierwszej aplikacji w Windows Azure ..	43
3.1. Zakładanie konta na witrynie Microsoft Online Services i subskrypcja usług	44
3.2. Założenie projektu i uruchamianie usług na portalu zarządzającym Windows Azure	49
3.2.1. Usługa Hosted Service	50
3.2.2. Konto w Azure Storage	52
3.3. Niezbędne i przydatne narzędzia do pracy z platformą Windows Azure	54
3.4. Role i ich instancje na platformie Windows Azure	56
3.5. Gwarancja jakości świadczonych usług (SLA)	56
Rozdział 4. Kalkulator w chmurze Azure	59
4.1. Przygotowanie projektu oraz wybór ról	60
4.2. Przegląd plików utworzonych ze wzorca projektu	60
4.3. Strona główna oraz obsługa kontrolek	65
4.4. Debugowanie lokalne	67
4.5. Wdrożenie aplikacji do chmury z wykorzystaniem Visual Studio i Azure SDK	69
4.6. Zatrzymanie i usunięcie aplikacji	74
Rozdział 5. Studium przypadku: serwis Moja-Muzyka	77
5.1. Biznesowa wizja systemu	78
5.2. Projektowa wizja systemu	78
5.3. Test przetwarzania plików wideo	79
5.4. Tworzenie projektu serwisu Moja-Muzyka	82
5.5. Budowanie warstwy prezentacji	83
5.6. Parsowanie strony filmu w serwisie YouTube	84
5.7. Komunikacja pomiędzy rolami (Azure Queues)	85
5.8. Logika roli ProcesorWideo	88
5.9. Lista wyników i lokalne testowanie aplikacji	92
5.10. Zmiany w projekcie przed wdrożeniem do chmury	101
5.11. Wdrożenie aplikacji do chmury Windows Azure	103
5.12. Dodanie nowych instancji ról	104
Rozdział 6. Studium przypadku: serwis Moje-Zdjęcia	107
6.1. Biznesowa wizja systemu	108
6.2. Projektowa wizja systemu	108
6.3. Moje-Zdjęcia — wersja lokalna	109
6.3.1. Tworzenie projektu	110
6.3.2. Dodanie modelu	110
6.3.3. Wygenerowanie kontrolera i widoków	113
6.3.4. Modyfikacja kontrolera	115
6.3.5. Modyfikacje widoków	121
6.3.6. Dodanie metody-akcji i widoku Search	129

6.4. Moje-Zdjęcia — wersja w chmurze	130
6.4.1. Zakładanie bazy SQL Azure	131
6.4.2. Modyfikacja plików konfiguracyjnych	133
6.4.3. Dodanie bibliotek MVC i silnika Razor	134
6.4.4. Wdrożenie serwisu do chmury	135
Rozdział 7. Podsumowanie	137
Dodatek A Literatura	139
A.1. Książki	139
A.2. Zasoby internetowe	139
Dodatek B Wdrożenie aplikacji do chmury	141
B.1. Utworzenie paczki wdrożeniowej w Visual Studio	141
B.2. Wysyłanie paczki i pliku konfiguracyjnego do chmury	142
B.3. Wykorzystanie plików wykonywalnych Windows Azure SDK	143
Dodatek C Windows Azure nie tylko dla środowiska .NET	145
C.1. PHP w Windows Azure	146
C.2. Java w Windows Azure	150
Dodatek D Diagnostyka w Windows Azure	157
Dodatek E Montowanie i używanie dysku wirtualnego Azure Drive	163
Dodatek F Migracja do SQL Azure	169
Dodatek G Dostawcy chmur	177
Skorowidz	179

Rozdział 4.

Kalkulator w chmurze Azure

Po uzyskaniu konta i subskrypcji usług na portalu Microsoft Online Services (<http://mop.microsoftonline.com>), a następnie założeniu projektu na portalu zarządzającym Windows Azure (<https://windows.azure.com>) możemy przejść do tworzenia aplikacji przeznaczonych dla chmury Azure. Naszą pierwszą aplikacją będzie kalkulator z operacjami dodawania, odejmowania, mnożenia i dzielenia. Ten prosty przykład z powodzeniem zilustruje pełny cykl budowania aplikacji z wykorzystaniem Visual Studio 2010, Azure Software Development Kit (SDK) oraz języka C#. Zakładamy, że czytelnik zna podstawy ASP.NET i potrafi w tej technologii stworzyć proste aplikacje webowe. Budowanie aplikacji dla chmury Azure wygląda w dużej mierze tak samo jak w klasycznej aplikacji ASP.NET.

Na poziomie „biznesowym” aplikację można scharakteryzować następująco: kalkulator ma realizować cztery podstawowe operacje arytmetyczne — dodawanie, odejmowanie, mnożenie i dzielenie. Ma być wyposażony w graficzny interfejs do wprowadzania argumentów i wyprowadzania wyników. Argumenty i wyniki to liczby zmiennoprzecinkowe.

Opis aplikacji na poziomie projektowym musi być bardziej szczegółowy. Argumenty operacji są wprowadzane za pomocą dwóch kontroltek typu `TextBox`. Wybór operacji i początek obliczeń jest wyzwalany kliknięciem jednej z czterech kontroltek typu `Button`. Zasadnicze obliczenia są realizowane w procedurze obsługi zdarzenia kliknięcia i przesyłania danych. W tej procedurze — metodzie zdarzeniowej — kolejno następują: pobranie i konwersja argumentów na wartości typu `double`, identyfikacja operacji i obliczenie wyniku. Wynik, po zamianie na typ `string`, wyświetla się w kontrolce `Label`.

Jak widać, projektowany kalkulator jest nieskomplikowaną aplikacją, ale powinien mieć graficzny interfejs użytkownika. Oznacza to, że w kalkulatorze istnieje konieczność utworzenia co najmniej jednej instancji *Web Role*.

4.1. Przygotowanie projektu oraz wybór ról

Po zainstalowaniu Azure Software Development Kit (podczas pisania tej książki była dostępna wersja 1.4) w Visual Studio pojawia się możliwość utworzenia projektu typu *Cloud Service*. Po wybraniu tej opcji wyświetli się okno dialogowe, w którym musimy zdecydować, jakie role chcemy dodać do projektu. Role i ich znaczenie opisano w rozdziale 3., „Przygotowania do utworzenia pierwszej aplikacji w Windows Azure”. Dokonane wybory spowodują wygenerowanie plików, które odróżnią projektowaną aplikację od typowego projektu ASP.NET. Dodatkowe pliki są powiązane z dodanymi rolami oraz właściwościami projektu, dostępnymi na najwyższej gałęzi drzewa plików projektu.

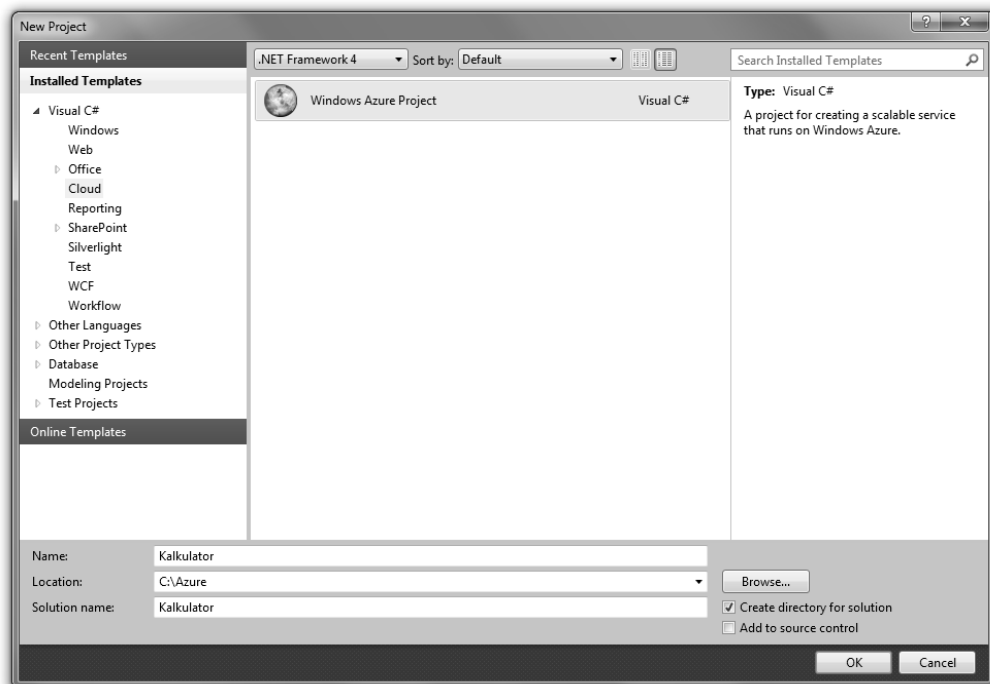
Przystępując do tworzenia projektu, należy uruchomić Visual Studio 2010 jako administrator. Jeśli tego nie zrobimy, to istnieje możliwość, że w późniejszym kroku nie uruchomi się Development Fabric, czyli lokalny emulator chmury. Oto wymagane działania:

1. Kliknij prawym przyciskiem myszki ikonkę Visual Studio 2010 (VS) i z menu kontekstowego wybierz *Uruchom jako administrator*.
2. Utwórz nowy projekt, wybierając z menu *File/New/Project*.
3. W nowym oknie dialogowym, w gałęzi języka *Visual C#*, wybierz opcję typu projektu *Cloud*. W środkowej części okna pojawi się tylko jeden rodzaj projektu: *Windows Azure Project*.
4. Podaj lokalizację i nazwę projektu; dla uproszczenia opisu wprowadź *C:\Azure* oraz *Kalkulator*. Rysunek 4.1 przedstawia okno dialogowe po tej operacji. Kliknij przycisk *OK*.
5. Po kliknięciu *OK* pojawi się okno dialogowe doboru ról do projektu. Wybierz *ASP.NET Web Role*, a następnie kliknij przycisk ze znakiem *>*. Po prawej stronie w oknie powinna się pojawić wybrana rola — rysunek 4.2. Taki sposób postępowania pozwala dodać większą liczbę ról do projektu. Na potrzeby projektowanego kalkulatora dodaj tylko *ASP.NET Web Role* i potwierdź wybór kliknięciem przycisku *OK*.

Po tych czynnościach Visual Studio wygeneruje podstawowe pliki projektu. Można je zobaczyć w drzewie plików projektu, w prawej górnej części ekranu.

4.2. Przegląd plików utworzonych ze wzorca projektu

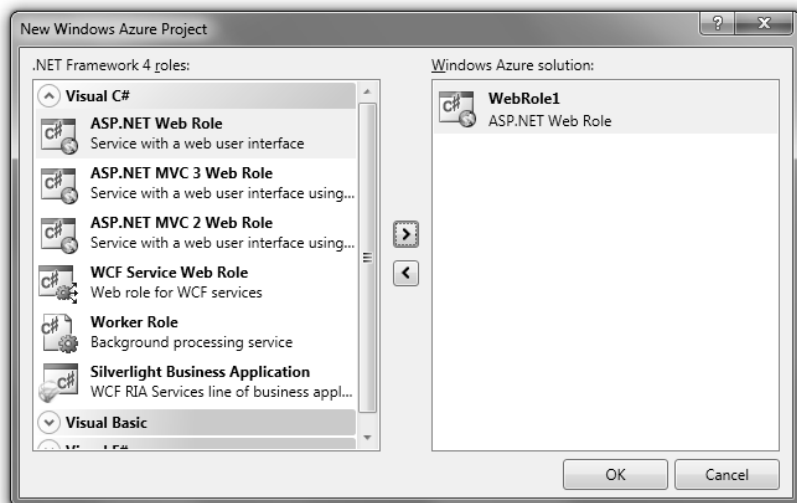
Środowisko Visual Studio automatycznie wygenerowało podstawowe pliki projektu, tak samo jak w przypadku klasycznego projektu ASP.NET. Jednak w tym przypadku pojawi się kilka nowych plików, które wcześniej nie występowały. Przyjrzyjmy się im.



Rysunek 4.1. Okno dialogowe wyboru typu projektu, jego lokalizacji i nazwy

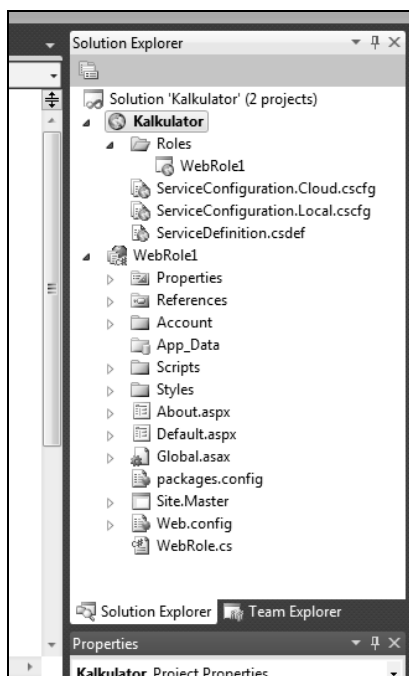
Rysunek 4.2.

Dodanie do nowego projektu roli serwera webowego ASP.NET Web Role



Zaraz po utworzeniu projektu drzewo powinno wyglądać tak jak na rysunku 4.3. Załóżmy, że tak naprawdę środowisko VS utworzyło dwa projekty — projekt *Kalkulator* oraz projekt *WebRole1*. Pierwszy jest projektem typu *CloudService*, drugi — projektem ASP.NET z dodatkowym plikiem o nazwie *WebRole.cs*.

Rysunek 4.3.
*Drzewo plików
projektu*

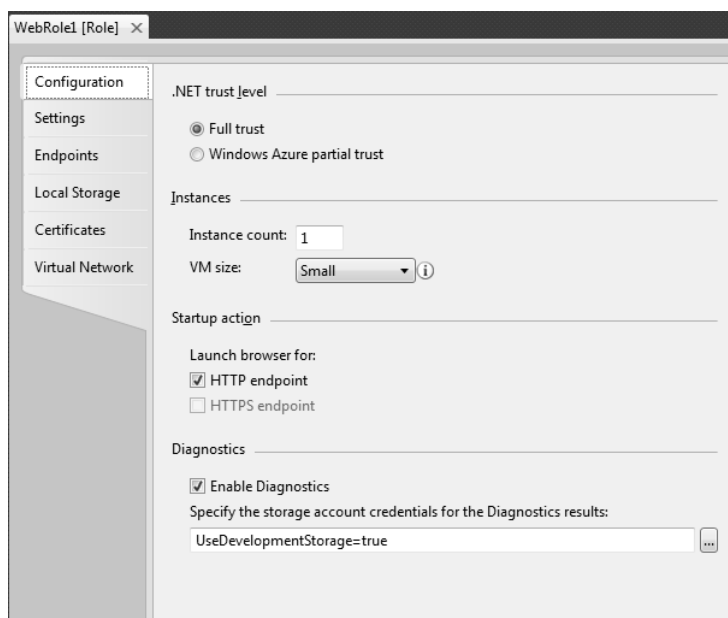


Projekt *Kalkulator* jest odpowiedzialny za konfigurację ról (gałąź *Roles*) i ogólnych ustawień projektu Azure. Gdybyśmy dodali więcej ról do projektu niż tylko *WebRole1*, to pojawiłoby się odpowiednio więcej gałęzi w katalogu *Roles* projektu *Kalkulator*.

W projekcie *Kalkulator* można znaleźć trzy dodatkowe pliki: *ServiceDefinition.csdef*, *ServiceConfiguration.Local.cscfg* oraz *ServiceConfiguration.Cloud.cscfg*. Pierwszy opisuje, z jakiego typu kodu składa się budowana aplikacja. Plik *ServiceDefinition.csdef* powstaje przy tworzeniu projektu i nie można go zmieniać w czasie działania aplikacji (po wdrożeniu). Pozostałe pliki (*ServiceConfiguration.Local.cscfg* i *ServiceConfiguration.Cloud.cscfg*) zawierają właściwości ról i dodatkowe ustawienia, które można edytować. Tworzone są dwa pliki konfiguracji — jeden jest używany przy lokalnym wytwarzaniu aplikacji (Local), a drugi przy wdrożeniu do chmury. To właśnie w pliku *ServiceConfiguration.Cloud.cscfg* ustawiamy liczbę instancji roli do uruchomienia. Plik konfiguracyjny można edytować w dowolnej chwili działania aplikacji. Przykładowo, jeśli w pewnym momencie zdecydujemy, że utworzona aplikacja wymaga istnienia większej liczby instancji ról, to powinniśmy edytować lub zmieniać właśnie ten plik. Wymiana i edytowanie są możliwe także z portalu zarządzającego, w którym zakładaliśmy projekt — zob. rozdział 3. Oba wymienione pliki można oczywiście edytować z poziomu środowiska Visual Studio i, jak większość plików konfiguracyjnych, mają one format dokumentów XML.

Warto wspomnieć, że Visual Studio dostarcza interfejsowy konfigurator dla każdej roli i automatycznie generuje pliki konfiguracyjne na podstawie wprowadzonych danych — rysunek 4.4. Po dwukrotnym kliknięciu na gałąź *WebRole1* można zmieniać właściwości roli, między innymi takie jak:

Rysunek 4.4.
*Konfigurator
 właściwości roli
 — generuje
 część pliku
 konfiguracyjnego
 odpowiedzialnego
 za daną rolę*



- ♦ liczba instancji roli;
- ♦ poziom zaufania roli w środowisku .NET (ang. *.NET trust level*);
- ♦ wielkość maszyny wirtualnej, na której ma działać aplikacja (ang. *VM Size*);
- ♦ punkty końcowe (ang. *Endpoints*), którymi będzie można się połączyć z rolą (HTTP, HTTPS).

Po utworzeniu projektu pliki konfiguracyjne mają domyślną zawartość — listingi 4.1 i 4.2.

Listing 4.1. Domyślna zawartość pliku *ServiceDefinition.csdef*

```
<?xml version="1.0" encoding="utf-8"?>
<ServiceDefinition name="Kalkulator"
  xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefinition">
  <WebRole name="WebRole1">
    <Sites>
      <Site name="Web">
        <Bindings>
          <Binding name="Endpoint1" endpointName="Endpoint1" />
        </Bindings>
      </Site>
    </Sites>
    <Endpoints>
      <InputEndpoint name="Endpoint1" protocol="http" port="80" />
    </Endpoints>
    <Imports>
      <Import moduleName="Diagnostics" />
    </Imports>
  </WebRole>
</ServiceDefinition>
```

Listing 4.2. Domyślna zawartość pliku *ServiceConfiguration.Cloud.cscfg* (lub *Local*)

```

<?xml version="1.0" encoding="utf-8"?>
<ServiceConfiguration serviceName="Kalkulator"
xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceConfiguration"
osFamily="1" osVersion="*">
  <Role name="WebRole1">
    <Instances count="1" />
    <ConfigurationSettings>
      <Setting name="Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString"
        value="UseDevelopmentStorage=true" />
    </ConfigurationSettings>
  </Role>
</ServiceConfiguration>

```

Należy również poświęcić uwagę plikowi *WebRole.cs* i podobnym, np. *WorkerRole.cs*. Zawierają one klasy o nazwach *WebRole*, *WorkerRole* etc., które dziedziczą po klasie *RoleEntryPoint*. Klasa *RoleEntryPoint* zawiera metody, które można przesłonić i w ten sposób zdefiniować reakcje danej instancji roli na zmiany środowiskowe. Główne metody klasy *RoleEntryPoint* do przesłonięcia to:

- ◆ *OnStart* — wywoływana, gdy instancja jest uruchamiana po raz pierwszy;
- ◆ *OnStop* — wywoływana, gdy instancja jest właśnie zatrzymywana;
- ◆ *Run* — wywoływana, gdy instancja roli zostanie zainicjalizowana (*OnStart*); w metodzie tej implementowane powinny być długotrwałe zadania obliczeniowe, np. pobieranie asynchroniczne zadań z kolejki.

Ostatni przypadek, czyli metoda *Run*, jest opisana dokładniej w rozdziale 5. (serwis Moja-Muzyka), przy okazji omawiania roli *Worker Role*. System zarządzający aplikacjami w Azure (*AppFabric*) wywołuje przy uruchamianiu instancji roli metodę *OnStart* z klasy *WebRole*. Gdy ta metoda zwróci wartość *true*, system wywołuje metodę *Application_Start* (znaną z pliku *Global.asax* w klasycznym ASP.NET). Z chwilą gdy instancja roli kończy działanie, najpierw wywoływana jest metoda *Application_End* — znana z pliku *Global.asax* — a na koniec metoda *OnStop* z klasy *WebRole*. Listing 4.3 zawiera klasę *WebRole* wygenerowaną przez Visual Studio.

Listing 4.3. Klasa *WebRole* domyślnie wygenerowana przez Visual Studio

```

public class WebRole : RoleEntryPoint
{
    public override bool OnStart()
    {
        // For information on handling configuration changes
        // see the MSDN topic at http://go.microsoft.com/fwlink/?LinkId=166357.

        return base.OnStart();
    }
}

```

Na potrzeby opisywanego przykładu nie musimy niczego w niej zmieniać. Do działania aplikacji konieczne jest zwrócenie wartości *true* przez metodę *OnStart*, inaczej rola w ogóle się nie uruchomi.

4.3. Strona główna oraz obsługa kontrolek

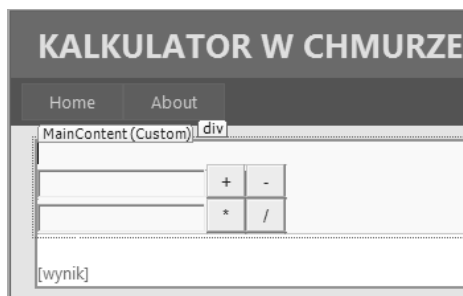
Poznaliśmy już nowy rodzaj projektu i dodatkowy plik roli (*WebRole.cs*). Najwyższy czas zająć się zasadniczą częścią rozwoju aplikacji *Kalkulator*, czyli rozmieszczeniem kontrolek na stronie oraz implementacją ich obsługi.

Na stronie *Default.aspx* — zgodnie z opisem projektowym aplikacji — umieścimy kilka kontrolek: dwie kontrolki typu `TextBox` do wprowadzenia argumentów operacji, cztery kontrolki typu `Button` do wskazywania wymaganej operacji i jedną kontrolkę typu `Label` do wyświetlania wyniku. Oto wymagane czynności:

1. Kliknij dwukrotnie plik *Default.aspx* w gałęzi plików projektu.
2. Gdy do środkowej części ekranu załaduje się kod strony, ustaw ją w trybie *Design* (kliknij przycisk *Design* w lewym dolnym rogu środkowego panelu).
3. Usuń ze strony tekst powitalny.
4. Zmień tytułowy napis na stronie z *My ASP.NET Application* na *Kalkulator w chmurze*. W tym celu otwórz i poddaj edytowaniu w trybie *Design* plik *Site.Master*. Po zmianie tekstu zapisz i zamknij plik *Site.Master*. Wróć do edytowania pliku *Default.aspx*.
5. W przyborniku — panel o nazwie *Toolbox* — wyszukaj potrzebne kontrolki i po prostu przeciągnij je na stronę *Default.aspx*. Na stronie umieść, jedna pod drugą, dwie kontrolki `TextBox`. Następnie wstaw cztery kontrolki `Button`, po dwie przy każdej kontrolce `TextBox`. Na koniec, poniżej kontrolek `TextBox`, wstaw kontrolkę `Label`. Kontrolkom `Button` zmień właściwość *Text* na znaki odpowiadające operacjom arytmetycznym — czynności te wykonaj za pomocą panelu *Properties*. Po tych operacjach strona *Default.aspx* powinna mieć wygląd taki jak na rysunku 4.5. Dla przejrzystości kodowania warto też nazwać wszystkie kontrolki według jakiejś konwencji, przypisując wartości atrybutom ID.

Rysunek 4.5.

Ekran główny kalkulatora po wstawieniu kontrolek



6. Każdej kontrolce `Button`, dalej nazywanej przyciskiem, ustaw właściwość *CommandName* na nazwę tekstową danej operacji arytmetycznej: dla + niech to będzie *dodaj*, dla - niech to będzie *odejmij* etc.

7. Otwórz plik *Default.aspx.cs* i do klasy `_Default` dodaj metodę wykonaj z listingu 4.4.

Listing 4.4. Zawartość metody *wykonaj*, odpowiedzialnej za wykonywanie obliczeń kalkulatora

```
protected void wykonaj(object sender, CommandEventArgs e)
{
    double dArg1 = 0.0;
    double dArg2 = 0.0;
    // zamiana danych użytkownika na wartości double
    bool wart10K = Double.TryParse(arg1.Text, out dArg1);
    bool wart20K = Double.TryParse(arg2.Text, out dArg2);
    if (wart10K && wart20K)
    {
        double liczba = 0;
        string op = "";
        switch (e.CommandName)
        {
            case "dodaj":
                op = "+";
                liczba = dArg1 + dArg2;
                break;
            case "odejmij":
                op = "-";
                liczba = dArg1 - dArg2;
                break;
            case "pomnoz":
                op = "*";
                liczba = dArg1 * dArg2;
                break;
            case "podziel":
                op = "/";
                liczba = dArg1 / dArg2;
                break;
        }
        wynik.Text = arg1.Text + op + arg2.Text + " = " + liczba;
    }
    else
    {
        wynik.Text = "Nie można obliczyć wyniku na podstawie wprowadzonych wartości";
    }
}
```

8. Kolejnym krokiem jest przypisanie do obsługi zdarzenia `Command` wszystkich czterech przycisków pojedynczej metody `wykonaj`. W tym celu w panelu właściwości (ang. *Properties*) kliknij symbol żółtej błyskawicy i przejdź do listy zdarzeń danego obiektu. Dla każdego przycisku znajdź pozycję *Command* i powiąż z nią metodę `wykonaj`.

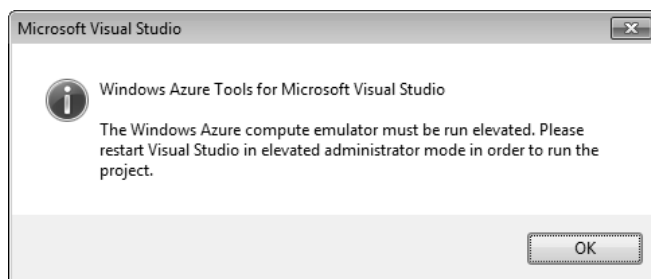
9. Zapisz wszystkie zmiany dokonane w projekcie.

Teraz możemy przystąpić do przetestowania działania zbudowanej aplikacji. Wciśnięciem klawisza *F5* uruchamiamy kompilację i zaczynamy debugowanie lokalne. Działa ono podobnie jak w okienkowych aplikacjach Windows. Po naciśnięciu *F5* zostanie

uruchomiona domyślna przeglądarka internetowa, a w niej aplikacja *Kalkulator*. Jeśli wcześniej nie uruchomiliśmy Visual Studio jako administrator, może pojawić się błąd jak na rysunku 4.6. W takim przypadku należy zamknąć projekt oraz Visual Studio, a następnie ponownie uruchomić środowisko, ale tym razem z uprawnieniami administratora. Ponadto może się pojawić monit systemu o konieczności dodania wyjątku bezpieczeństwa dla emulatora Windows Azure. Należy wtedy potwierdzić zgłaszany wyjątek.

Rysunek 4.6.

Błąd, który pojawia się w przypadku gdy nie uruchomiliśmy VS jako administrator

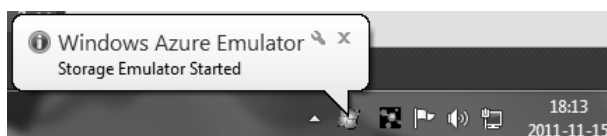


4.4. Debugowanie lokalne

Z chwilą gdy w trybie administratora uruchomimy Visual Studio i otworzymy nowo utworzony projekt *Kalkulator*, uruchomi się emulator Development Fabric. Emulator ten symuluje zarówno obliczenia w chmurze, jak i magazynowanie danych w chmurze, czyli Azure Storage. Uruchomienie emulatora jest sygnalizowane błękitnym logo na pasku zadań — obszar obok systemowego zegara i datownika — rysunek 4.7.

Rysunek 4.7.

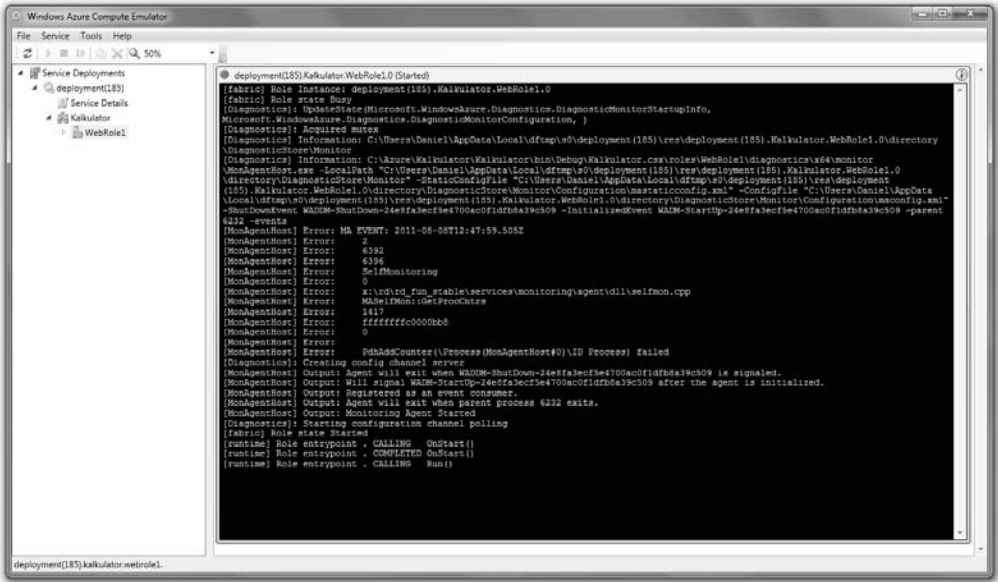
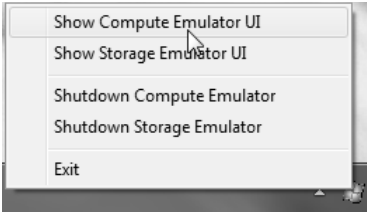
Logo Development Fabric w pasku ikon powiadomień



Jeśli po tej czynności logo jest niewidoczne, to trzeba kliknąć biały trójkącik na pasku zadań i w oknie dialogowym wybrać ikonę *Windows Azure Simulation Monitor*, a jako zachowanie wskazać *Pokaż ikony i powiadomienia*. Zaletą Development Fabric jest również możliwość podglądania na bieżąco zachowań emulatora chmury. Wystarczy kliknąć prawym przyciskiem myszki na błękitne logo i w ten sposób uruchomić menu kontekstowe. Z menu kontekstowego (widocznego na rysunku 4.8) można wybrać różne opcje, między innymi *Show Compute Emulator UI* oraz *Show Storage Emulator UI*. Dodatkowo w dowolnej chwili można wyłączyć poszczególne części Development Fabric przez wybranie opcji *Shutdown Compute Emulator* lub *Shutdown Storage Emulator*. Jeśli wybierzemy *Exit*, to natychmiast zamkniemy środowisko emulujące.

Kliknijmy opcję *Show Compute Emulator UI* i zobaczymy, jak pracuje jedyna instancja roli *WebRole1*, która uruchamia się przy starcie kalkulatora. Zaraz po uruchomieniu aplikacji możemy wyświetlić konsolę naszej roli, wybierając ją z lewego panelu — rysunek 4.9.

Rysunek 4.8.
*Menu kontekstowe
Development Fabric*

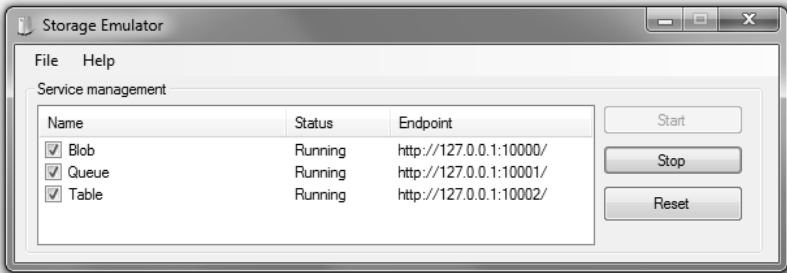


Rysunek 4.9. *Panel administracyjny emulatora Windows Azure*

Łatwo zauważyć, że emulator Development Fabric najpierw wywołał metodę `OnStart`, a następnie metodę `Run`, dokładnie tak, jak zostało to wcześniej opisane.

Jeżeli chodzi o lokalny emulator składowania danych — również możemy otworzyć jego interfejs użytkownika, klikając w menu kontekstowym Development Fabric opcję *Show Storage Emulator UI*. Można w nim zobaczyć, pod jakimi portami figurują odpowiednie komponenty *Development Storage*. Możemy je również zresetować lub zatrzymać — rysunek 4.10.

Rysunek 4.10.
*Interfejs
użytkownika
Development
Storage*



Na koniec warto dodać, że dzięki Development Fabric możemy dokładnie tak samo jak w klasycznym ASP.NET debugować aplikacje. Jeśli jednak istnieje więcej niż jedna instancja danej roli, to pułapki będą reagowały tak jak w aplikacjach współbieżnych. Z tego powodu warto najpierw przetestować aplikację przy jednej instancji, a dopiero potem zwiększać ich liczbę.

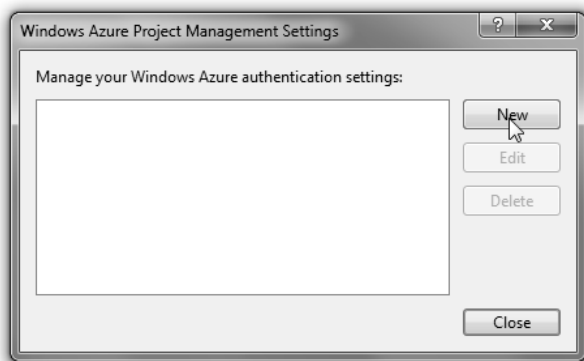
4.5. Wdrożenie aplikacji do chmury z wykorzystaniem Visual Studio i Azure SDK

Nareszcie nadszedł czas, gdy po przetestowaniu aplikacji lokalnie chcemy ją pokazać całemu światu, innymi słowy: wysłać ją do chmury Azure. Można to zrobić dwojako. Sposób pierwszy to przygotowanie specjalnej paczki wdrożeniowej i załadowanie jej do chmury razem z plikiem konfiguracyjnym poprzez portal <https://windows.azure.com>. Sposób drugi to skorzystanie z nowych komend środowiska Visual Studio, które pojawiają się w nim po zainstalowaniu Windows Azure Tools for Visual Studio oraz Windows Azure SDK. Preferowany jest drugi sposób, bo zdecydowanie ułatwia wdrożenie aplikacji w chmurze Azure. Do sukcesu wdrożenia z Visual Studio niezbędne jest zainstalowanie w systemie odpowiedniego **certyfikatu X.509**. Jeśli zdecydujemy się na opcję drugą, to utworzenie i zainstalowanie certyfikatu za pomocą Visual Studio będą dość proste.

Jeżeli dzięki wiadomościom zawartym w poprzednim rozdziale udało nam się utworzyć projekt na portalu <https://windows.azure.com>, wykorzystamy go teraz do wdrożenia zbudowanego kalkulatora. Pamiętajmy, że musimy mieć zarówno uruchomioną usługę *Hosted Service*, jak i założone konto *Storage Account* — zob. rozdział 3. Oto konieczne czynności:

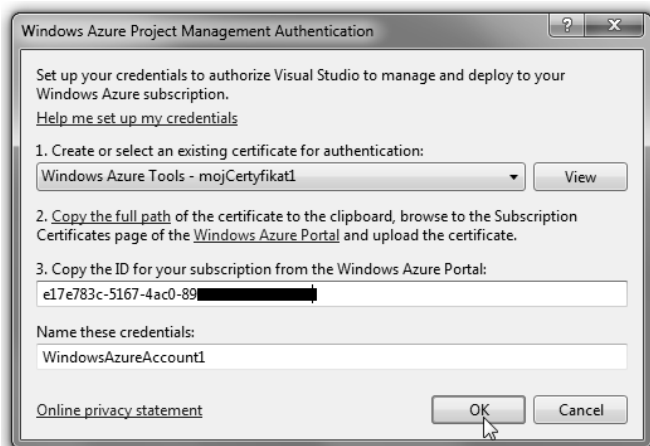
1. Kliknij prawym przyciskiem myszki na ikonkę projektu *CloudService* (w naszym przypadku *Kalkulator*) i wybierz opcję *Publish*.
2. W nowym oknie dialogowym pojawi się trzykrokový kreator. W pierwszym kroku *Sign in* (domyślnie otwartym) z menu rozwijalnego *Choose your subscription*: wybierz *<Manage...>*. Pojawi się nowe okno (rysunek 4.11), w którym należy dodać swoje dane identyfikacyjne. Możemy to zrobić, klikając przycisk *New*. W kolejno otwartym oknie pozwolimy programowi Visual Studio na wygenerowanie certyfikatu, który później — z portalu zarządzającego — należy dodać do zaufanych certyfikatów.
3. W menu rozwijalnym *Create or select an existing certificate for authentication* wybierz *<Create...>*. W nowym oknie nadaj przyjazną nazwę swojemu nowemu certyfikatowi i kliknij *OK*. Po kliknięciu tego przycisku certyfikat zostaje utworzony i zainstalowany w systemie. Teraz należy go wysłać do portalu zarządzającego.

Rysunek 4.11.
Okno zarządzania
poświadczeniami
do Windows Azure



4. Nie zamykając poprzedniego okna, zaloguj się do portalu <https://windows.azure.com>.
5. Po zalogowaniu, korzystając z menu *Hosted Services, Storage Accounts & CDN*, wybierz odsyłacz *Management Certificates*; wtedy w górnej części okna dialogowego pojawi się przycisk *Add certificate*. Po jego kliknięciu wyświetli się nowe okno dialogowe, w którym — po kliknięciu przycisku *Browse* — możesz wskazać plik certyfikatu na dysku swojego lokalnego komputera. Najłatwiej w tym miejscu będzie wrócić do poprzedniego okna w Visual Studio i kliknąć link, od którego zaczyna się drugi punkt, czyli *Copy the full path*. Następnie w oknie dialogowym wyboru pliku certyfikatu po prostu wklej skopiowaną ścieżkę. Kliknij przycisk *Otwórz*, a następnie *OK*. Po chwili certyfikat zostanie załadowany.
6. Wróć do okna Visual Studio. Trzeci punkt to prośba o wprowadzenie *Subscription ID*. Wartość tę również można znaleźć na portalu zarządzającym. W lewym panelu wybierz menu *Hosted Services*, zaznacz węzeł swojego projektu, a po prawej stronie, w sekcji *Subscription ID*, zobaczysz właściwe ID. Identyfikator ten należy skopiować do pola oznaczonego etykietą *Copy the ID for your subscription from the Windows Azure Portal*. W ostatnim polu należy wprowadzić dowolną nazwę, którą w przyszłości będziemy identyfikować teraz wprowadzone dane. Okno po tych operacjach powinno wyglądać tak jak na rysunku 4.12. Kliknij przycisk *OK* — w tym momencie Visual Studio łączy się z serwerami Windows Azure i sprawdza, czy wprowadzone dane są poprawne.
7. Po kliknięciu *OK* nastąpi powrót do poprzedniego okna, w którym pojawi się na liście konto Twoich poświadczeń do Azure. Kliknij *Close*.
8. W oknie, do którego powróciłeś (rysunek 4.13), w menu rozwijalnym *Choose your subscription*: powinny być już wybrane nowo dodane poświadczenia. Kliknij *Next >*.
9. W kolejnym kroku pojawi się okno dodawania nowego Hosted Service. Ponieważ wykonaliśmy już to poprzez portal zarządzający, pomijamy ten krok, klikając w oknie *Create Windows Azure Services* przycisk *Cancel*. Widzimy w tym momencie okno drugiego etapu wdrażania aplikacji.

Rysunek 4.12.
*Okno wprowadzania
danych do
uwierzytelnienia
komputera*



Rysunek 4.13.
*Pierwszy krok
wdrażania
aplikacji
do chmury*



10. W oknie tym, widocznym na rysunku 4.14, z listy rozwijalnej *Hosted service*: wybierz założony w poprzednim rozdziale projekt *AzureTest*. Z listy *Environment*: wybierz *Staging*. Środowisko *Staging* jest środowiskiem testowym, a *Production* — środowiskiem finalnym, przeznaczonym dla końcowego użytkownika. Zawsze warto wysłać aplikację najpierw do środowiska *Staging*, aby po przetestowaniu, tym razem w prawdziwej chmurze, przełączyć ją później do środowiska *Production*. Przełączenie do środowiska *Production* wymaga jednego kliknięcia na portalu zarządzającym.
11. Z listy *Build configuration*: wybierz *Release*, a z *Service configuration*: wybierz *Cloud*. W zakładce *Advanced Settings* (rysunek 4.15) należy upewnić się, że w polu *Storage account*: jest wybrane Twoje konto *Storage*. Zatwierdź ustawienia kliknięciem *Next >*.

Rysunek 4.14.

Drugi krok wdrażania aplikacji do chmury: wybieranie projektu, środowisk i konfiguracji

**Rysunek 4.15.**

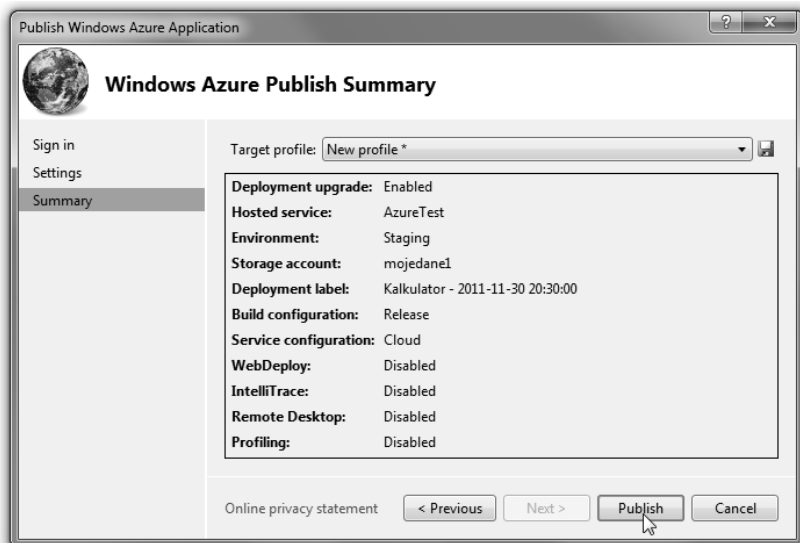
Zakładka Advanced Settings z miejscem wyboru konta Storage



- 12.** Kolejne okno przed potwierdzeniem powinno wyglądać tak jak na rysunku 4.16. Aby rozpocząć wdrażanie do chmury, kliknij przycisk *Publish*. Uruchomi to proces wysyłania paczki aplikacji do chmury. Proces ten może trwać kilka minut. W Visual Studio wszelkie komunikaty można obserwować w małym okienku w dolnym panelu ekranu — rysunek 4.17.
- 13.** Gdy w oknie komunikatów pojawi się komunikat *Complete*, możesz sprawdzić działanie aplikacji w środowisku testowym — *Staging*. W tym celu kliknij link oznaczony *Website URL* — rysunek 4.17. Jeżeli stwierdzisz, że aplikacja działa poprawnie, to możesz ją przełączyć do środowiska *Production*.

Rysunek 4.16.

*Okno
podsumowania
przed
ostatecznym
wdrożeniem
do chmury*

**Rysunek 4.17.**

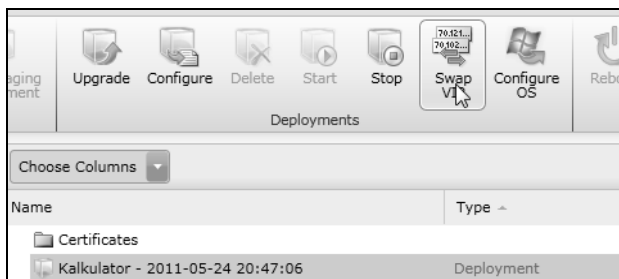
*Panel podglądu
postępu
ładowania
aplikacji
do chmury*



- 14.** Na portalu zarządzającym wybierz *Hosted Services*, następnie węzeł z subskrypcją, a potem właściwe wdrożenie (w tym przypadku jego nazwa to „Kalkulator — data wysłania”). Na górze ekranu pojawi się przycisk *Swap VIP* — rysunek 4.18. Po jego kliknięciu pojawi się okno potwierdzenia — rysunek 4.19. Kliknij przycisk *OK*. Z chwilą potwierdzenia następuje zamiana adresów wewnętrznych środowiska *Staging* na *Production*. Wtedy w sekcji *DNS Name* pojawi się końcowy adres URL utworzonej aplikacji — rysunek 4.20.

Rysunek 4.18.

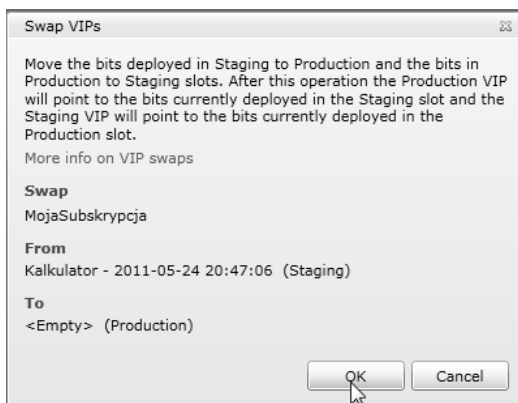
*Ekran pokazujący
lokalizację przycisku
Swap VIP*



Jeśli dotarłeś do tego miejsca opisu, to prawdopodobnie udało Ci się zbudować, załadować i uruchomić swoją pierwszą aplikację w chmurze Windows Azure — przyjmij gratulacje!

Rysunek 4.19.

*Okno potwierdzenia
zmiany środowiska
Staging na Production*

**Rysunek 4.20.**

*Panel boczny,
w którym pojawi
się końcowy
adres URL
zbudowanej
aplikacji*

Name	Type	Status	Environment	Created
mojaSubskrypcja	Subscription	Active		2011-09-02 08:29:10 UTC
test-kalkulator	Hosted Service	Created		
Certificates				
Kalkulator - 2011-09-02	Deployment	Ready	Production	
WebRole1	Role	Ready	Production	
WebRole1_IN_0	Instance	Ready	Production	

Created
2011-09-02 08:29:10 UTC
Cores used
1
DNS name
http://test-kalkulator.cloudapp.net
Environment
Production

Jednak z chwilą uruchomienia aplikacji w chmurze Azure rozpoczyna się naliczanie opłat związanych z czasem działania i używanymi zasobami. Dlatego od razu warto wiedzieć, co należy zrobić, aby zatrzymać to naliczanie.

4.6. Zatrzymanie i usunięcie aplikacji

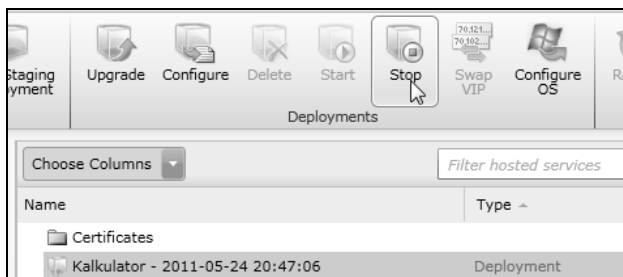
Jedynym sposobem na zatrzymanie licznika płatności Azure jest zatrzymanie aplikacji (Hosted Service) i usunięcie konta Azure Storage. Ponieważ w aplikacji *Kalkulator* nie magazynujemy w chmurze żadnych danych, konta Storage właściwie nie musimy usuwać. Jednak warto poznać sam mechanizm usuwania i mieć pewność, że żadne opłaty za nieużywane konto nie będą naliczane i pobierane.

Aby usunąć aplikację z chmury, trzeba ją najpierw zatrzymać. Oto wszystkie wymagane działania:

1. Zaloguj się na swoje konto na portalu zarządzającym Windows Azure (<https://windows.azure.com>).
2. W lewym menu wybierz *Hosted Services*.
3. W środkowej części ekranu powinny się pojawić węzły wszystkich aplikacji umieszczonych w chmurze. Wybierz węzeł z aplikacją kalkulatora oznaczony „Kalkulator — data i godzina”.

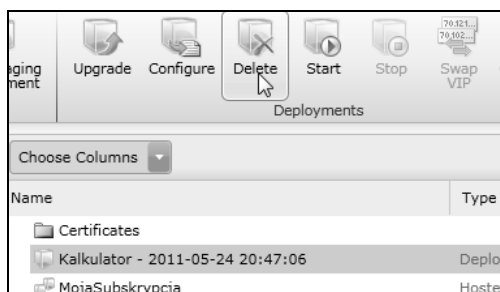
4. W menu w górnej części ekranu powinien się uaktywnić przycisk *Stop* — rysunek 4.21; kliknij go.

Rysunek 4.21.
Zatrzymywanie aplikacji



5. Poczekaj do momentu aż kolumna *Status* przy aplikacji kalkulatora zmieni się na *Stopped*. Czas oczekiwania może się wydłużyć nawet do kilku minut.
6. W menu w górnej części ekranu powinien się teraz uaktywnić przycisk *Delete* — rysunek 4.22; kliknij go.

Rysunek 4.22.
Usuwanie aplikacji z chmury



7. Poczekaj, aż aplikacja kalkulatora całkowicie zniknie z drzewa wraz z węzłem roli i jej instancji. Od tego momentu nie będą już naliczane żadne opłaty za usługę *Hosted Services* (przez aplikację kalkulatora).

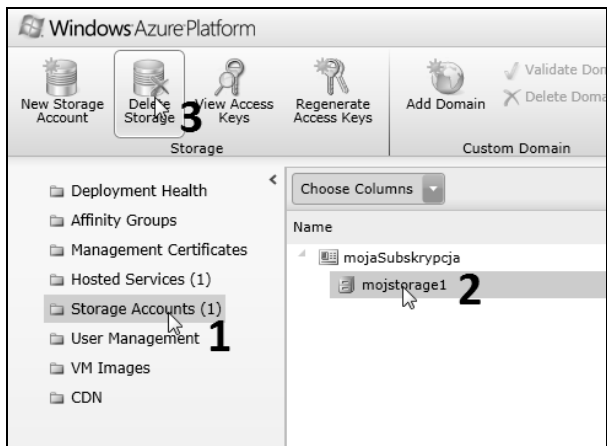
Do usunięcia konta Storage niezbędne są następujące działania:

1. Zaloguj się na swoje konto na portalu zarządzającym Windows Azure (<http://windows.azure.com>).
2. W lewym menu wybierz *Storage Accounts* — rysunek 4.23, punkt 1.
3. W środkowej części ekranu pojawiają się węzły powiązane z kontem i aplikacjami; zaznacz właściwy *mojstorage1* — rysunek 4.23, punkt 2.
4. W menu w górnej części ekranu powinien się uaktywnić przycisk *Delete Storage* — rysunek 4.23, punkt 3.; kliknij go. Pojawi się monit potwierdzenia usunięcia konta; kliknij przycisk *Yes*.

Po krótkim czasie konto powinno zniknąć z listy. Od tej pory nie płacimy już za korzystanie z konta Storage, trzeba jednak pamiętać, że usunięcie konta Storage wiąże się z bezpowrotną utratą przechowywanych w nim danych.

Rysunek 4.23.

Trzy kroki
do usunięcia
konta Storage



Skorowidz

A

- aplikacja jako usługa, SaaS, 10
- aplikacja Javy w chmurze, 150, 155
- aplikacja webowa, 109
- aplikacje hostowane w chmurze, 18
 - ASP.NET, 20
 - ASP.NET MVC, 20
 - PHP, 20
 - WCF, 20
- architektura SQL Azure, 37
- archiwum aplikacji webowej, 152
- ASP.NET, 20
- ASP.NET MVC, 20
- ASP.NET MVC 3, 54
- Azure Blobs, 16, 20, 26, 30, 42
 - prawa dostępu do kontenera, 33
 - struktura, 31, 32
- Azure Queues, 16, 20, 26, 33
 - struktura, 34
- Azure Service Management API, 106
- Azure Storage, 25
 - Azure Blobs, 26, 30
 - Azure Queues, 26, 33
 - Azure Tables, 26
 - bezpieczeństwo danych, 26
 - skalowalność, 26
- Azure Tables, 16, 20, 26, 27
 - encje, entities, 27
 - kommunikacja, 29
 - partycjonowanie poziome, 30
 - podział składników, 28
 - tabele, tables, 27
 - właściwości, properties, 27

B

- baza danych SQL Azure, 130, 131
- bezpieczeństwo danych, 26
- biblioteka Microsoft.WindowsAzure.CloudDrive, 163
- biblioteki MVC, 134
- billing i zużycie, 39
- blob, 30, 31
- blob podzielony na fragmenty, Block Blob, 31
- blob stronicowany, Page Blob, 31, 32, 163
- Block Blob, 31
- bloki, blocks, 31
- BPOS, Business Productivity Online Suite, 12

C

- chmura, 7
 - IaaS, infrastruktura jako usługa, 10
 - modele usług, 9
 - PaaS, platforma jako usługa, 10
 - rodzaje chmur, 10
 - SaaS, aplikacja jako usługa, 10
 - wady i zalety, 12
- chmura Azure, 15
- chmura obliczeniowa, 7, 137
- chmura prywatna, 9
- chmura publiczna, 9
- chmura Windows Azure, 103
- Cloud Computing, 7, 106, 137
- cmd, 143
- Code-First, 108
- Content Delivery Network, 21
- CRUD, Create, Read, Update, Delete, 113

cspack, 143
 csrun, 144
 cykl życia komunikatu, 35

D

debuger Visual Studio, 100
 definicja chmury, 7
 definiowanie dla roli pamięci lokalnej, 90, 163
 dekompilowanie kodu, 33
 diagnostyka, 160
 diagnostyka w Windows Azure, 157
 diagnozowanie aplikacji, 162
 diagram przypadków użycia, 109
 diagram sekwencji, 79
 dodanie
 instancji ról, 104
 klasy modelu, 110
 kontrolera ZdjecieController, 114
 metody-akcji, 129
 silnika Razor, 134
 widoku Search, 129
 dodatki, 55
 dodawanie
 poświadczeń, 102
 bibliotek MVC, 134
 widoków, 128
 dostawcy chmur, 177
 dysk wirtualny, 42, 163
 dzielenie przez zero, 158

E

emulator Azure Storage, 77
 encja, entity, 27, 28
 Entity Framework, 107
 Entity Framework 4.1, 16, 108
 etapowe uruchamianie aplikacji, 77
 ewolucja infrastruktury IT, 8

F

FIFO, 36
 folder bin, 143
 format VHD, 42
 framework ASP.NET MVC 3, 16, 107

G

generowanie paczki wdrożeniowej, 142
 gwarancja jakości, 56
 gwarancja SLA, 57

I

IaaS, Infrastructure as a Service, 10
 infrastruktura jako usługa, IaaS, 10
 inicjalizacja pamięci cache, 164
 instalacja
 ASP.NET MVC 3, 54
 Visual Studio 2010, 54
 Windows Azure SDK, 54
 Windows Azure Tools for Microsoft
 Visual Studio, 54
 instalowanie Windows Azure dla Eclipse, 146
 IntelliTrace, 157, 159, 162
 interfejs serwisu Moja-Muzyka, 83

J

Java w Windows Azure, 150
 język C#, 17
 język VB, 17
 JSP, Java Server Pages, 152

K

klasa
 _Default, 86, 165
 definiująca model danych, 93
 LocalResource, 164
 WorkerRole, 88, 89
 Zdjecie, 108
 ZdjecieController, 129
 klient Azure Blobs, 88, 164
 klient Azure Storage, 164
 klucz do konta, 102
 klucz główny, 28
 kod strony index.jsp, 152
 kodek FFMPEG, 79
 kolejka inputqueue, 78
 kolejki, 34
 komenda
 cmd, 143
 cspack, 143
 csrun, 144
 komunikacja pomiędzy rolami, 85
 komunikat, 34
 MessageID, 35
 MessageTTL, 35
 PopReceipt, 35
 VisibilityTimeout, 35
 konfiguracje maszyn wirtualnych, 105
 konta w serwisie MSN Hotmail, 45
 kontener, 30, 31
 kontener dysków wirtualnych, 164
 konto MOCP, 46

konto w Azure Storage, 52
kontroler, 115
kontroler dostępu, Access Control, 17, 21
kontroler zarządzania, Fabric Controller, 19
kontroler ZdjecieController, 113
kontrolka lastBulletedList, 94

L

linia komend, 143
LINQ, Language Integrated Query, 29
lista wyników, 92
logi IntelliTrace, 160
logi IntelliTrace pobrane z chmury, 161
lokalne środowisko uruchomieniowe, 77
lokalny emulator chmury, 144

M

magazynowanie danych, 25
 Azure Storage, 25
 SQL Azure, 36
maszyna wirtualna, 20
mechanizm Azure Drive, 163
mechanizm IntelliTrace, 157
metoda
 Create, 165
 Delete, 120
 Details, 120
 HttpNotFound, 120
 Mount, 165
 OnStart, 86, 88, 89, 96, 164
 Page_Load, 86, 94, 164
 parsująca, 84
 Run, 89, 90, 97
 startButton_Click, 94
 updateList, 94
 wykonaj zmodyfikowana, 157
metody-akcje
 Create, 116
 Delete, 119
 DeleteConfirmed, 119
 Edit, 117
 Search, 129
migracja bazy, 175
migracja do SQL Azure, 169
model usługi, service model, 19
modyfikacja kontrolera, 115
modyfikacja plików konfiguracyjnych, 133
modyfikacje widoków, 121
montowanie Azure Drive, 163
MP3, 79
MP4, 79
MSMQ, Microsoft Message Queueing, 34
MVC, 16

N

narzędzie
 Azure Storage Explorer, 100
 do przeglądania baz SQL Azure, 136
 SQL Azure Migration Wizard, 169

O

obiekt
 CloudStorageAccount, 86, 101
 POCO, 110
 Regex, 84
 StreamWriter, 165
 ViewResult, 120
obiekty bazy, 172
obliczenia, Compute, 19
obsługa zdarzenia kliknięcia, 87
Office 365, 12
opcja CDN, 21
opłaty, 22

P

PaaS, Platform as a Service, 10
paczka wdrożeniowa, 141
Page Blob, 31, 32, 163
pamięć lokalna, 77
parametry połączenia
 do bazy SQL Azure, 174
 do bazy SQL Server, 171
 do magazynów danych, 162
parsowanie strony filmu, 84
PartitionKey, 28, 30
PHP w Windows Azure, 146
platforma jako usługa, PaaS, 10
platforma Windows Azure, 7, 16, 17, 137
 komponenty, 18
 opłaty, 22
 podział platformy, 18
 SQL Azure, 17, 21
 Windows Azure, 17, 19
 Windows Azure AppFabric, 17, 21
plik
 _Layout.cshtml, 121
 cmd.exe, 143
 Create.cshtml, 114, 122
 cspack.exe, 143
 csharp.exe, 144
 Default.aspx, 83, 167
 Default.aspx.cs, 80, 83, 165
 Delete.cshtml, 114, 126
 Details.cshtml, 114

plik

Edit.cshtml, 114, 124
 ffmpeg.exe, 89
 Index.cshtml, 121
 index.php, 147
 konfiguracyjny, 106
 MojeZdjecia.cshtml, 128
 MvcWebRole1, 133
 NotSupportedByAzureFile.Config, 169
 OutputqueueMP3.cs, 83
 OutputqueueMP3DataServiceContext.cs, 83
 ServiceConfiguration.Cloud.cscfg, 100
 ServiceConfiguration.cscfg, 96, 133, 141, 155
 ServiceConfiguration.Local.cscfg, 100, 144
 ServiceDefinition.csdef, 89, 164
 Site.Master, 83
 Tomcat.cspkg, 155
 Tomcat.sln, 150
 VHD, 163
 WAR, 152
 Web.config, 112, 114, 133
 WebRole.cs, 82
 WorkerRole.cs, 82, 103
 ZdjecieController.cs, 114

pliki

.cshtml, 121
 cmd Buildme.cmd, 154
 cmd Packme.cmd, 154
 cmd Runme.cmd, 154
 konfiguracyjne, 133
 wideo, 79
 wykonywalne narzędzia SDK, 143

pobieranie klucza, 102

POCO, 16

pojemniki w chmurze, 162

połączenie z bazą danych, 133

poświadczenie konta, 77, 101

prawa dostępu do kontenera, 33

ProcesorWideo, 78

projekt

JSP, 152
 Kalkulator, 137, 157
 Moja-Muzyka, 78
 Moje-Zdjęcia, 107
 PHPAzure1, 147
 Windows Cloud Project, 82

protokół

REST, 17
 SOAP, 18
 XML, 18

provisioning, 39

przechowywanie danych, 27

przekazywanie komunikatów, 78

przestrzeń dla magazynu danych, Azure Storage, 19

przestrzeń nazw System.IO, 165

przetwarzanie plików wideo, 79

przycisk Start, 87

R

relacyjna baza danych, 17

relacyjna baza danych w chmurze, 37

rodzaje chmur, 10

rola, roles, 20

Web Role, 20, 78

Worker Role, 20, 78

rola Aplikacja, 82, 92

rola ProcesorWideo, 82, 88

routing, 39

RowKey, 28, 30

S

SaaS, Software as a Service, 10

scenariusz biznesowy, 14

schemat współdziałania ról, 79

serwer

bazy danych, 132

SQL Azure, 131

Tomcat, 151

wirtualny, 145

serwis

Moja-Muzyka, 77, 137

Moje-Zdjęcia, 107, 138

Shared Access Signatures, 33

sieć CDN, Content Delivery Network, 19

skalowalność aplikacji, 104

składowanie danych, 25, 30

skrypty migracyjne, 173

SLA, Service Level Agreement, 56

SQL Azure, 21, 36, 37, 107, 131, 170

architektura, 37

ograniczenia, 39

protokoły dostępu, 39

sposoby dostępu, 40

warstwa infrastruktury, 38

warstwa kliencka, 38

warstwa platformy, 38

warstwa usług, 38

subskrypcja usług platformy, 44

system kolejowania, 34

system operacyjny chmury, 17

system zarządzania komunikatami, 33

szyna danych, Service Bus, 17, 21

Ś

ścieżka do pamięci lokalnej, 164
środowisko
 .NET, 145
 Development Fabric, 153
 Eclipse, 17, 145
 Hyper-V, 19
 Visual Studio, 147
 Visual Studio 2010, 15, 54

T

tabela, table, 27
TDS, Tabular Data Stream, 39
technologia
 Ajax, 17
 ASP.NET, 15
 MS SQL Server, 36, 37
 Silverlight, 17
test kodeka, 81
testowanie aplikacji, 148
TimeStamp, 28
T-SQL, 40
tworzenie
 aplikacji webowej, 150
 bazy danych, 174
 paczki wdrożeniowej, 143
 pierwszej aplikacji, 43
 projektu Moje-Zdjęcia, 110
 projektu serwisu, 82
 wirtualnego dysku w chmurze, 163
 Worker Role, 154
typ danych
 Binary, 29
 Boolean, 29
 DateTime, 29
 Double, 29
 Guid, 29
 Int32, 29
 Int64, 29
 String, 29
typ danych SQL Azure, 40

U

usługa
 Azure Storage, 20
 Hosted Service, 50
 hostingu, 56
 „na żądanie”, 7
 WCF, 20
użytkownik anonimowy, 108
użytkownik zarejestrowany, 108

V

VHD, Virtual Hard Drive, 42, 163
Visual Studio, 147
Visual Studio 2010, 15, 54

W

warstwa
 infrastruktury, 38, 39
 kliencka, 38
 platformy, 38, 39
 prezentacji, 82
 usług, 38, 39
 billing i zużycie, 39
 provisioning, 39
 routing, 39
WCF Data Services, 20, 29
wdrażanie aplikacji do chmury, 103, 130, 141
wdrażanie aplikacji PHP do chmury, 149
wdrożenie serwisu do chmury, 135
Web Role, 20
węzeł, 30
widok
 Delete, 126
 Details, 126
 Edit, 124
 Index, 129
 widok MojeZdjecia, 128
Windows Azure, 19, 25, 43
 dane, 20
 Java, 150
 kontroler zarządzania, 19
 obliczenia, 19
 PHP, 146
 sieć CDN, 21
Windows Azure Activity Log, 104
Windows Azure AppFabric, 21
Windows Azure SDK, 15, 54, 141
Windows Azure SDK dla Eclipse, 145
Windows Azure Tomcat Solution Accelerator,
 151
Windows Azure Tools for Microsoft Visual
 Studio, 54
Windows Azure Tools for Visual Studio, 86, 141
Windows Intune, 12
Windows Live SkyDrive, 12
wirtualne środowisko hostingowe, 11
wizja systemu, 78, 108
właściwości komunikatu, 35
właściwość, properties, 27, 29
 nazwa, name, 29
 typ, type, 29
 wartość, value, 29

właściwość Clean on role recycle, 90
Worker Role, 20
współdziałanie ról, 79
współdzielone sygnatury dostępowe, 33
wtyczka diagnostyczna, 159
wtyczka Silverlight, 33
wybór ról, 82
wybór wzorca projektu, 111
wygenerowanie kontrolera, 113

wygenerowanie widoków, 115
wysyłanie paczki do chmury, 142
wywołanie błędu, 160

Z

zakładanie bazy SQL Azure, 131
zakładanie konta, 44
założenie projektu, 49
zmienna środowiskowa Path, 143
zwiększenie instancji roli, 105

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

W 2008 roku na konferencji w Los Angeles firma Microsoft po raz pierwszy zaprezentowała publicznie swoje niezwykle interesujące dzieło — platformę Windows Azure. Jednak dopiero dwa lata później poprawiona, ustandaryzowana i ulepszona platforma podbiła serca wielu właścicieli firm. Dlaczego tak się stało? Otóż Azure pozwala projektować i uruchamiać oprogramowanie w zupełnie nowatorski sposób. Jej architektura umożliwia firmie pozbycie się własnej, często kłopotliwej infrastruktury IT oraz stworzenie skalowalnych i niezawodnych narzędzi, idealnie dopasowanych do specyfiki działalności, a także gwarantuje opłacalność ekonomiczną całego przedsięwzięcia. Z tych powodów programiści „działający w chmurze” są dziś najbardziej poszukiwanymi pracownikami na rynku!

Windows Azure. Wprowadzenie do programowania w chmurze to jedna z pierwszych książek o tej nowoczesnej technologii napisanych w języku polskim. Przeznaczona dla osób zajmujących się technologiami webowymi oraz studentów kierunków informatycznych, zawiera opis wszystkich najważniejszych składników „chmury Azure”. Dowiesz się z niej, jak działa chmura i jak stworzyć oraz uruchomić aplikację na tej platformie. Będziesz mógł przeanalizować trzy kompletne, coraz bardziej skomplikowane przykłady budowania aplikacji z wykorzystaniem frameworka NET 4.0 i Windows Azure SDK. Znajdziesz tu także aż siedem dodatków, między innymi na temat polecanej literatury, wdrożenia aplikacji, dysku wirtualnego Azure Drive i diagnostyki — wszystkie napisane z myślą o praktycznym wykorzystaniu możliwości Windows Azure.

- Platforma Windows Azure
- Pierwsza aplikacja w Windows Azure
- Studia przypadku — serwisy Moja-Muzyka i Moje-Zdjęcia
- Wdrożenie aplikacji do chmury
- Diagnostyka w Windows Azure
- Dostawcy chmur

**Nie przegap kolejnego kroku w ewolucji technologicznej
— zdobądź wiedzę o Windows Azure!**

Nr katalogowy: 7235



Księgarnia internetowa
<http://helion.pl>



Zamówienia telefoniczne:
0 801 339900



0 601 339900



Helion

Sprawdź najnowsze promocje:

• <http://helion.pl/promocje>

Książki najchętniej czytane:

• <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

• <http://helion.pl/novosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

helion.pl
księgarnia
internetowa

Cena 39,00 zł

ISBN 978-83-246-3703-4



9 788324 637034

Informatyka w najlepszym wydaniu