

Tworzenie aplikacji dla systemu Windows 8

Zaprojektuj, napisz i opublikuj swoją aplikację



Aplikacje dla systemu Windows 8 — podejmij wyzwanie!

Jak pisać aplikacje dla systemu Windows 8 z wykorzystaniem API Windows Runtime?

Jak stworzyć pożądane opcje i nadać aplikacji wygląd zgodny
z zasadami nowoczesnego stylu projektu Microsoft?

Jak wykorzystać usługi online i opublikować aplikację w sieci?

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: Studio Gravite/Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

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/twapw8>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Materiały do książki można znaleźć pod adresem:
<ftp://ftp.helion.pl/przyklady/twapw8.zip>

ISBN: 978-83-246-7093-2

Copyright © Helion 2013

Printed in Poland.

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

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

Spis treści

- Rozdział 1. Przedmowa 7**
 - Dla kogo jest przeznaczona ta książka 9
 - Przygotowanie do pracy 10
 - Środowisko uruchomieniowe 10
 - Oprogramowanie 10
 - Uruchamianie po raz pierwszy przykładowego projektu 11
 - 1. Notifications Extensions 11
 - 2. Live SDK 11
 - 3. SQLite for Windows Runtime 12
 - 4. CharmFlyout 12
 - Specyfika programowania w Windows 8 13
- Rozdział 2. Platforma tworzenia aplikacji w systemie Windows 8 15**
 - Windows Runtime 16
 - Asynchroniczność 17
 - Asynchroniczność od zaplecza 18
 - Cztery cechy systemu okiem użytkownika i programisty 20
 - Specyfika aplikacji dla systemu Windows 8 21
- Rozdział 3. Wymagania certyfikacyjne aplikacji dla systemu Windows 8 23**
 - Wartość aplikacji 23
 - Reklamy 24
 - Przewidywalność 24
 - Aplikacje pod kontrolą 25
 - Treść aplikacji 26
 - Aplikacje łatwe do zrozumienia 27
- Rozdział 4. Tworzenie aplikacji dla systemu Windows 8 29**
 - Omówienie docelowej aplikacji 29
 - Rozpoczynamy pracę 30
 - Zaczynamy od Visual Studio 30
 - Środowisko pracy 32
 - Podstawowe elementy aplikacji 34
 - Ikony 34
 - Ekran powitalny 36
 - Testowanie aplikacji 37
 - Testowanie w obecnym systemie 37
 - Testowanie na symulatorze 37
 - Testowanie na zdalnym urządzeniu 39

Rozdział 5. Wygląd aplikacji	41
Projektowanie wyglądu aplikacji w Visual Studio	41
Składnia XAML	41
Projektowanie wyglądu aplikacji	44
Kontrolki do przechowywania zawartości	45
Poziomy układ aplikacji	47
Tworzenie listy przewijanej w poziomie i w pionie	47
Podpięcie danych do listy GridView	49
Nawigacja	52
Animacja	54
Pasek aplikacji i pasek nawigacji	56
Skalowanie i snapowanie	58
Widoki Snapped i Filled	58
Cykl życia aplikacji — stany: aktywna, wstrzymana, zamknięta	60
Stany w aplikacji	60
Symulowanie przechodzenia do innego stanu	62
Rozdział 6. Możliwości aplikacji dla systemu Windows 8	63
Dostęp do Charms	63
Wyszukiwanie w aplikacji	64
Udostępnianie danych dla innych aplikacji	67
Wykorzystanie kontraktu Ustawienia	69
Otwieranie zewnętrznego linku w przeglądarce	69
Przejsięcie do strony wewnątrz aplikacji	71
Rozwijany interfejs ustawień	71
Obsługa multimediiów	73
Obsługa urządzeń	75
Kamera	76
Geolokalizacja — moduł GPS	77
Rozdział 7. Zarządzanie danymi w aplikacji	81
Dane aplikacji	82
Ustawienia lokalne	82
Ustawienia roamingowe	83
Ustawienia tymczasowe	86
Zapisywanie multimediiów do plików aplikacji	87
Baza danych SQLite	87
Rozdział 8. Współpraca z usługami online	91
Usługi Live	91
Przygotowanie do pracy z usługami Live	91
Dostęp i zakres działania usług Live	94
Referencje REST	96
Logowanie	98
SkyDrive	100
Hotmail	102
RSS	103
Rozdział 9. Powiadomienia	107
Wyskakujące powiadomienia	107
Tworzenie szablonu powiadomienia z wykorzystaniem biblioteki NotificationsExtensions	109
Tworzenie szablonu powiadomienia z wykorzystaniem składni DOM	111
Sterowanie aplikacją za pomocą atrybutów	113
Usuwanie Schedule Toast Notifications	113

Kafelki	114
Aktualizacja treści kafelka	115
Kafelek podstawowy	116
Planowane powiadomienia	117
Okresowe powiadomienia	118
Wyświetlanie treści kafelka na ekranie blokowania	119
Kafelki pomocnicze	122
Okresowe i planowane powiadomienia dla kafelków pomocniczych	124
Wskaźniki	124
Okresowe powiadomienia dla wskaźników	126
Powiadomienia push	126
Uwierzytelnienie aplikacji	128
Deklaracja w manifeście	128
Utworzenie kanału powiadomień	129
Przekazanie URI kanału do serwera	129
Przechwycenie powiadomienia	131
Windows Azure Mobile Services	132
Rozdział 10. Zadania w tle	133
Wyzwalacze	134
Kanał kontroli	134
System	135
Powiadomienia Push	136
Czas	136
Utrzymanie	136
Warunki wystąpienia	136
Implementacja zadań w tle	137
Stworzenie projektu obsługującego zadanie w tle	137
Obsługa zadania	137
Dostosowanie manifestu	138
Zarejestrowanie zadania	139
Odwołanie wyzwalacza	139
Debugowanie wyzwalaczy	140
Rozdział 11. Sklep Windows	143
O czym należy pamiętać przy publikowaniu aplikacji	143
Lokalizacja aplikacji	143
Przedział wiekowy	144
Polityka prywatności	144
Funkcjonalność zgodna z opisem	144
Dane do testowania	145
Proces publikowania aplikacji	145
Jak zarabiać	147
Reklamy	148
Płatność z góry	149
Trial czasowy albo trial funkcjonalny	149
Zakup w aplikacji	151
Własne płatności	153
Oceny i komentarze	153
Statystyki	153
Wypłata	155
Podsumowanie	157
Skorowidz	158

Rozdział 2.

Platforma tworzenia aplikacji w systemie Windows 8

W poprzednich wersjach systemu Windows tworzyliśmy i instalowaliśmy standardowe aplikacje okienkowe działające w trybie pulpitu i wykorzystywaliśmy praktycznie dowolny język programowania, taki jak C#, Visual Basic (platforma .NET) czy C, C++ (aplikacje Win32). Ten typ aplikacji nie będzie działał na komputerach z systemem Windows 8 RT (dla procesorów ARM) ze względu na inną architekturę procesora.

Mieliśmy też (i dalej oczywiście mamy) aplikacje działające w przeglądarce, uruchamiane po przejściu na konkretną stronę internetową. Ten typ aplikacji tworzony jest za pomocą HTML i JavaScript. Takie aplikacje nadal możemy programować i będą one działać w systemie Windows 8.

Jednak najciekawszym typem aplikacji, o których będziemy pisać w tej książce, są tzw. aplikacje ze Sklepu Windows (ang. *Windows Store App*) — nowoczesne aplikacje dla systemu Windows 8. Kluczem w tworzeniu aplikacji jest API Windows Runtime (WinRT). To właśnie dzięki temu API mamy dostęp do usług systemowych, elementów związanych z przesyłaniem danych, wyświetlaniem oraz przekazywaniem grafiki i multimediów czy też do sprzętu zarówno zintegrowanego z urządzeniem (np. akcelerometr, GPS), jak i zewnętrznego (telewizor, drukarka, konsola Xbox).

My programiści możemy wykorzystać aż trzy zupełnie różne technologie i języki programowania do tworzenia aplikacji z wykorzystaniem Windows Runtime:

C lub C++ oraz XAML

Języki C i C++ to jedne z najbardziej popularnych i najbardziej potężnych języków programowania; stosowane są do tworzenia oprogramowania na różnych poziomach — od systemów operacyjnych, przez urządzenia przemysłowe, aż po nowoczesne aplikacje mobilne.

Wybierając język C++, możemy tworzyć aplikacje w naturalny sposób, dogodny dla programistów na co dzień korzystających z tego języka.

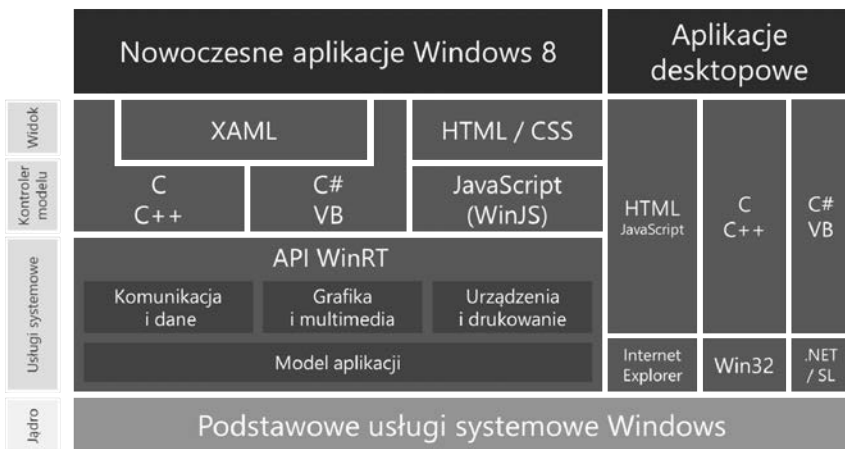
C# lub Visual Basic oraz XAML

C# i Visual Basic to języki obiektowe uruchamiane w środowisku .NET Framework. Z założenia ma to być język prosty i nowoczesny.

Tworzenie aplikacji w tych językach najbardziej przypomina tworzenie aplikacji na platformę Windows Phone lub w technologii WPF. Mamy do dyspozycji pliki stron tworzonych w XAML, z którymi łączą się tzw. pliki *code behind* zawierające logikę związaną z widokiem.

JavaScript oraz HTML i CSS

Umożliwienie pisania aplikacji dla systemu Windows 8 w języku JavaScript przy wykorzystaniu HTML i CSS do tworzenia wyglądu jest krokiem skierowanym głównie do twórców aplikacji sieci Web. Z poziomu JavaScript (rozszerzonego o bibliotekę WinJS) możemy uzyskać pełny dostęp do API i tworzyć kompletne aplikacje bez potrzeby nauki kolejnego języka. Schemat podziału języków w platformie Windows 8 został przedstawiony na rysunku 2.1.



Rysunek 2.1. Platforma tworzenia aplikacji dla systemu Windows 8

W tej książce skupimy się na tworzeniu aplikacji z wykorzystaniem C# oraz XAML.

Windows Runtime

Windows Runtime to zestaw API, który zawiera następujące cechy:

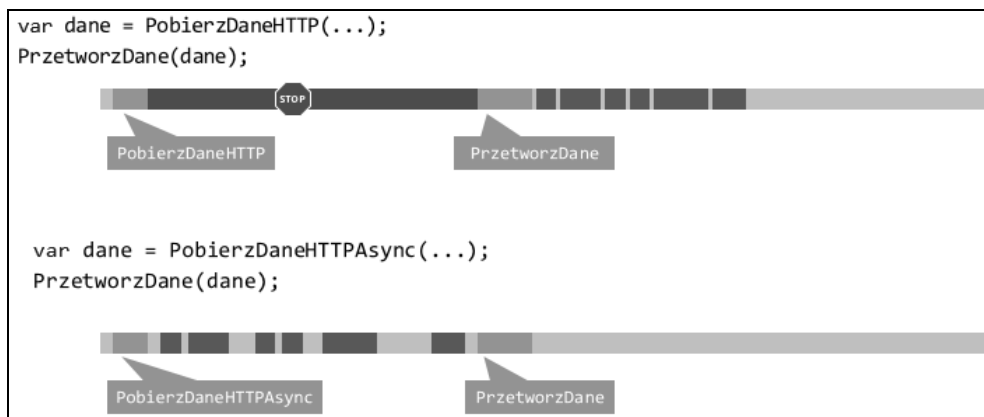
- ♦ implementację nowego stylu nazwanego Modern UI;
- ♦ prosty interfejs modelu programowania dla deweloperów systemu Windows (nie ma potrzeby uczenia się Win32 czy tego, czym są HDC, WndProc lub LPARAM);

- ♦ zarządzanie wyglądem poprzez pliki XAML (podobne do Silverlight oraz WPF);
- ♦ zaprojektowanie na działanie asynchroniczne;
- ♦ chronione API (sandbox). Mamy możliwość tworzenia aplikacji gotowych do wystawienia na Windows Store, jednak nie stworzymy w ten sposób aplikacji w standardzie COM (niskopoziomowe API).

W programowaniu wyglądu aplikacji WinRT umożliwia pisanie kodu zarówno w HTML z użyciem stylów CSS, jak i w formacie XAML. W tej książce opisujemy język C# do programowania aplikacji i XAML do definiowania jej wyglądu.

Asynchroniczność

Jak wiemy, programowanie synchroniczne jest mało praktyczne, aplikacje pisane w taki sposób są mniej responsywne i wydajne z punktu widzenia użytkownika. Dlatego właśnie programowanie asynchroniczne staje się dziś normą. Dzięki takiemu podejściu nie wstrzymujemy pracy np. w przypadku pobierania danych z serwera czy wykonywania wielu dłuższych operacji. Rysunek 2.2 pokazuje porównanie uruchomienia metody synchronicznie i asynchronicznie.



Rysunek 2.2. Porównanie wywołania synchronicznego i asynchronicznego

Pisanie aplikacji dla systemu Windows 8 stało się jeszcze prostsze i przyjaźniejsze zarówno dla programisty, jak i dla użytkownika dzięki modyfikatorowi metody `async` i słowu kluczowemu `await`. Opiszemy je pokrótce.

`async`

Oznacza, że metoda jest asynchroniczna. To także obliguje, by w ciele metody zawrzeć referencję `await`, którą opiszemy w dalszej części tego podrozdziału. Metody asynchroniczne mogą zwracać trzy typy danych:

1. `Task<TResult>` — jeśli metoda posiada zwracany wynik, którego argument jest typu `TResult`.
2. `Task` — jeśli metoda nie posiada parametru zwracanego lub zwracany parametr nie ma żadnego argumentu.
3. `Void` — jeśli metoda asynchroniczna ma być uchwytym zdarzenia.

Zwracany typ `Task` można porównać do obietnicy, jaką składa ta metoda — że w tym miejscu pojawi się obiekt typu `TResult`, gdy tylko operacja asynchroniczna dobiegnie końca.

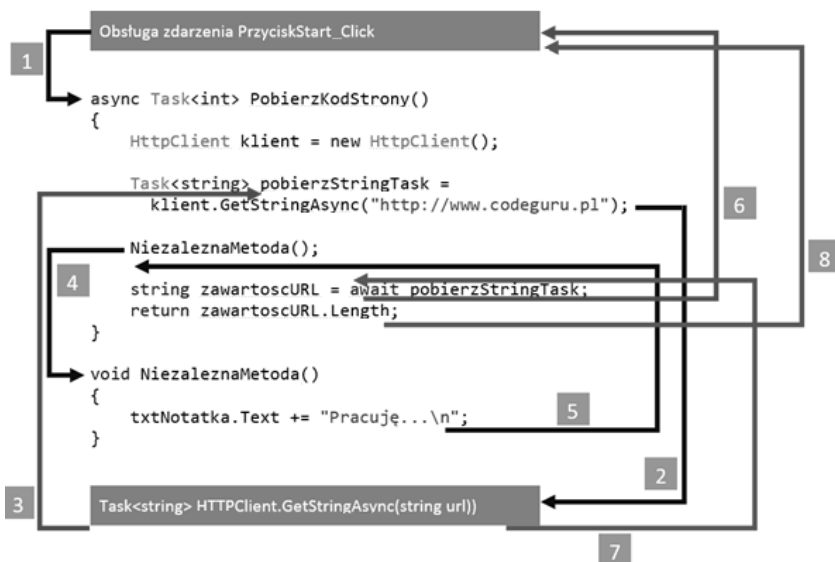
`await`

Występuje zawsze w ciele metody asynchronicznej i oznacza miejsce, w którym dana metoda nie może kontynuować pracy, dopóki oczekująca operacja asynchroniczna nie zostanie zakończona. W tym czasie metoda jest wstrzymywana, a sterowanie przekazywane z powrotem do wywołania tej metody asynchronicznej.

Warto zauważyć, że metody asynchroniczne nie są wywoływane w oddzielnym wątku, więc aby móc wykorzystywać mechanizm asynchroniczności, nie jest wymagana wielowątkowość.

Asynchroniczność od zaplecza

Przedstawimy teraz przykład, dzięki któremu zrozumiesz przebieg procesu asynchroniczności w typowej sytuacji. Zaczniij od analizy rysunku 2.3, który przedstawia metody związane z pobraniem i przetworzeniem kodu strony internetowej.



Rysunek 2.3. Przykład przebiegu procesu asynchroniczności

Asynchroniczność najłatwiej będzie wytłumaczyć na przykładzie prostego przepływu informacji pomiędzy metodami.

1. Uchwyty zdarzenia wywołuje metodę `PobierzKodStrony` i asynchronicznie oczekuje na jej zakończenie.
2. Metoda tworzy instancję klasy `HttpClient` i wywołuje asynchroniczną metodę `GetStringAsync` w celu pobrania zawartości strony jako `string`.
3. W metodzie `GetStringAsync` następuje wstrzymanie jej wykonania, np. z powodu trwającego procesu pobierania danych ze strony internetowej. By nie blokować zasobów, metoda przekazuje sterowanie do jej wyzwalacza, czyli do `PobierzKodStrony`. Jednocześnie metoda zwraca obiekt `Task<TResult>`, który jest obietnicą, że w tym miejscu pojawi się wynik działania metody po jej wykonaniu. Ponieważ `pobierzStringTask` jeszcze się nie zakończyło, to `PobierzKodStrony` może kontynuować inne zadania, które nie zależą od wyniku `GetStringAsync`. Operacje te są reprezentowane przez metodę `NiezaleznaMetoda`.
4. `NiezaleznaMetoda` jest metodą synchroniczną, która wykonuje swoją pracę i zwraca sterowanie do wyzwalacza.
5. `PobierzKodStrony` wykonało wszystkie zadania niezależne od `GetStringAsync`. Ponieważ `GetStringAsync` nie zakończyło swojej pracy, to `PobierzKodStrony` używa operatora `await` do wstrzymania postępu swojej pracy i przekazania sterowania do swojego wyzwalacza. W związku z tym zwraca `Task<int>`. `Task` jest swego rodzaju obietnicą zwrócenia wartości typu `integer` zawierającej długość pobranego `stringa`.

Warto zauważyć, że w sytuacji, gdy `GetStringAsync` (a tym samym `pobierzStringTask`) zakończyłyby pracę przed tym, jak `PobierzKodStrony` wstrzymałby swój proces, oczekując na wynik, sterowanie pozostałoby w metodzie `PobierzKodStrony`. Oznaczałoby to stratę kosztu wstrzymania i powrotu do metody `PobierzKodStrony`.

Wewnątrz wyzwalacza metody `PobierzKodStrony` cały proces jest powtarzany. Wyzwalacz może wykonywać inne metody, które nie zależą od `PobierzKodStrony`, a w przypadku zakończenia lub braku takich zadań przechodzi w stan oczekiwania na wynik (ang. *awaiting*). Kiedy metoda oczekuje na wynik, aplikacja kończy metodę wstrzymującą, czyli `GetStringAsync`. Uchwyty zdarzenia czeka na wynik `PobierzKodStrony`, a `PobierzKodStrony` czeka na wynik `GetStringAsync`.

6. `GetStringAsync` kończy swoją pracę i przygotowuje wynik (typu `string`) do zwrócenia. Jednak zwrócenie wyniku nie odbywa się w tradycyjny sposób (pamiętajmy, że metoda już zwróciła element typu `Task` w kroku 3.). Wynik jest przechowywany w obiekcie, który reprezentuje ukończenie metody, czyli w `pobierzStringTask`. Operator `await` odbiera rezultat z `pobierzStringTask`. Następuje skojarzenie wyniku ze zmienną `zawartoscURL`.
7. Kiedy `PobierzKodStrony` ma już przekazany wynik, oblicza jego długość. Wtedy metoda `PobierzKodStrony` również jest ukończona i oczekujące zdarzenie może kontynuować swoje działanie.

Podsumowując: metoda synchroniczna zwraca natychmiast wynik swojego działania i żadne inne prace nie mogą być wykonywane w tym samym czasie. Natomiast metody asynchroniczne (kroki 3. i 6.) zwracają numer zadania, kiedy ich praca jest wstrzymywana (w wyniku oczekiwania na ukończenie innej operacji). Kiedy metoda asynchroniczna kończy swoją pracę, zadanie jest oznaczane jako wykonane i jego ewentualny rezultat jest przechowywany w zadaniu.

Cztery cechy systemu okiem użytkownika i programisty

System, a w szczególności platforma aplikacyjna, różni się od poprzednich wersji systemu Windows na wielu płaszczyznach. Oto cztery najistotniejsze cechy, które sprawiają, że system jest inaczej postrzegany nie tylko przez programistów, ale też przez użytkowników.

- 1. Nowy interfejs.** Tak zwany Modern UI lub styl projektu Microsoft to tak naprawdę zbiór wytycznych, które umożliwiają stworzenie pięknej i spójnej aplikacji. Wytyczne zawierają nie tylko informacje o układzie treści na stronie. To także opis zachowania się aplikacji w typowych sytuacjach, np. wykorzystanie animacji w sposób zwiększający intuicyjność oraz odpowiednie zapisywanie danych, aby użytkownik miał do nich dostęp w miarę możliwości na wszystkich swoich urządzeniach połączonych jednym kontem Microsoft.
- 2. Integracja z systemem.** Standardowe polecenia wykonywane w aplikacji są wykonywane z jednego, wspólnego miejsca dla systemu i innych aplikacji — prawego paska *Charms*. Znajduje się tam pięć podstawowych ikon, które dają dostęp m.in. do wyszukiwania, współdzielenia treści oraz ustawień aplikacji. Z jednej strony użytkownik wie, że tego typu opcje znajdzie właśnie w tym miejscu. Z drugiej strony, programista może przeznaczyć maksymalnie dużo miejsca w aplikacji na zasadniczą treść, a pozostałe elementy interfejsu, zwykle te, które są globalne dla całej aplikacji, zamieścić właśnie w pasku *Charms*.
- 3. Wszystkie aplikacje w jednym miejscu.** Sklep Windows umożliwia znalezienie aplikacji w nowym stylu w jednym, centralnym miejscu. Jeśli aplikacja jest płatna, użytkownik może także za nią zapłacić. Ułatwia to znacznie poszukiwanie i instalowanie aplikacji.
- 4. Responsywność aplikacji.** Ta cecha powoduje, że aplikacje muszą być zaprojektowane tak, aby użytkownik jak najmniej odczuł opóźnienia wynikające np. z ładowania się aplikacji, oczekiwania na pobranie danych itp. Wiąże się to z dwiema rzeczami. Po pierwsze: wykorzystaniem asynchroniczności w miejscach, gdzie kod wykonuje się dłużej. To powoduje, że aplikacja nie zawiesza swojego działania. Po drugie: aplikacja musi sprawiać wrażenie aktywnej i odpowiadającej na interakcję użytkownika. Służą do tego m.in. animacje oraz indykatory postępu.

Specyfika aplikacji dla systemu Windows 8

W tradycyjnych aplikacjach dużą rolę odgrywała otoczka, która często przykrywała treść, jaką miała nieść aplikacja — pasek tytułowy aplikacji, menu, paski przewijania, a także komponenty nawigacyjne, jak np. wypukłe przyciski, które często bywały zbędne i nieprzemyślane. Dodatkowo dużo miejsca na ekranie z aplikacją zajmowały elementy systemowe, np. pasek startu.

W aplikacjach dla systemu Windows 8 stawia się na przejrzystość aplikacji i ma to odzwierciedlenie w projekcie. Pierwszą zmianą jest to, że aplikacje są pełnoekranowe. Pozwala to użytkownikowi skupić się na jednej aplikacji. Jednak dla wygody użytkowników przyzwyczajonych do pracy z dwoma aplikacjami na ekranie wprowadzony został tryb *snapped*, który umożliwia zadokowanie aplikacji z lewej lub prawej strony ekranu.

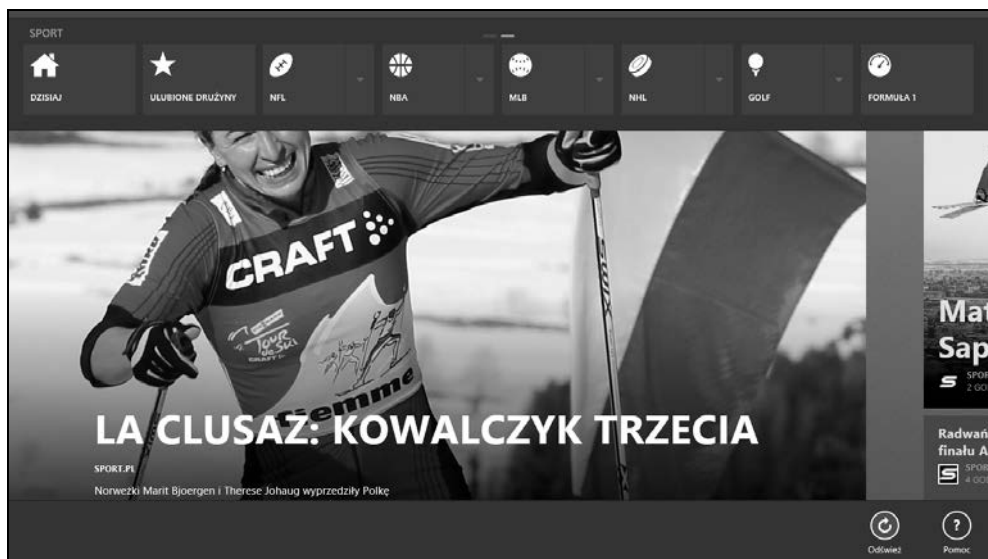
Aplikacje dla systemu Windows 8 w wielu aspektach różnią się od tradycyjnych aplikacji:

- ♦ mają mieć szybki i płynny interfejs użytkownika,
- ♦ są dostosowane do obsługi dotykiem,
- ♦ są pełnoekranowe,
- ♦ integrują się z systemem oraz łączą się z innymi aplikacjami dzięki kontraktom,
- ♦ działają nie tylko na procesorach x86 i x64, ale także na ARM.

Przede wszystkim wiąże się to ze zmianą podejścia do projektowania aplikacji. Dobra aplikacja jest nie tylko intuicyjna, ale także zapewnia bardzo dobry tzw. *user experience*, dzięki czemu jest łatwa w użyciu.

Zmienione są też zasady umieszczania elementów nawigacji. Gdzie tylko jest to możliwe, nawigację powinno integrować się z zawartością, a pozostałe elementy nawigacji powinny znajdować się w górnym pasku poleceń (ang. *navigation bar*). Są to paski wysuwane gestem przeciągnięcia z górnej lub dolnej krawędzi ekranu lub poprzez kliknięcie prawym przyciskiem myszy w dowolnym miejscu na ekranie. Na rysunku 2.4 przedstawiona jest przykładowa aplikacja, która zawiera górny pasek nawigacji i dolny pasek poleceń aplikacji.

Dodatkowo istotnym elementem jest pasek *Charms*, który wysuwa się z prawej krawędzi ekranu (wyświetlany jest przez gest najechnania myszą w prawy górny lub dolny róg ekranu i przejechanie odpowiednio w dół lub w górę). Zawiera on przyciski odnoszące się do najczęściej wykonywanych funkcji w systemie, ale też w aplikacjach! Wszystkie przyciski (oprócz przycisku *Start*) zmieniają swój kontekst w zależności od aplikacji, w której został otwarty pasek *Charms*. Rysunek 2.5 przedstawia uruchomioną aplikację, dla której został wysunięty pasek *Charms*.



Rysunek 2.4. Przykład aplikacji zawierającej górny i dolny pasek aplikacji



Rysunek 2.5. Aplikacja z wysuniętym paskiem Charms

Skorowidz

A

- adres URI, 129
- aktualizacja
 - treści kafelka, 115
 - współrzędnych, 79
- animacja, 54
- API Live Connect, 98
- API Windows Runtime, 15
- aplikacja Moje Spotkania, 29
- aplikacje ze Sklepu Windows, 15
- asynchroniczność, 19
- atrybut Launch, 113

B

- baza danych SQLite, 87
- biblioteka
 - CharmFlyoutLibrary, 72
 - NotificationsExtensions, 11, 109, 125
 - SQLite, 87
 - Unit Test Library, 32
 - WinJS, 16
- błąd powiadomienia, 109

C

- CharmFlyout, 12
- cykl życia aplikacji, 61

D

- dane aplikacji, 82
- debugowanie wyzwalaczy, 140
- designer, 33, 36
- dodawanie
 - animacji, 55
 - elementów do kontraktu, 70, 73
 - kafelka pomocniczego, 123
 - kontraktu do aplikacji, 64
 - przestrzeni nazw, 70
 - spotkania, 89
 - strony wyszukiwania, 65
 - wskaźnika, 125

- wydarzenia do kalendarza, 102
- wyzwalacza czasowego, 139
- zdjęcia na SkyDrive, 100
- dokument testowej licencji, 150
- dostęp do
 - bibliotek multimedialnych, 73
 - biblioteki użytkownika, 81
 - Charms, 63
 - ekranu blokowania, 121, 122
 - usług Live, 95
- dysk SkyDrive, 77
- działanie powiadomień, 126

E

- edycja pliku manifestu, 36
- ekran
 - blokowania, 121
 - powitalny, Splash Screen, 34, 37
- element listy, 48
- elementy Flyout, 72

F

- FIFO, First In First Out, 117
- folder
 - Assets, 33
 - Common, 33
- formularz
 - w języku C#, 42
 - w języku XAML, 42

G

- glify, 125
- GPS, 77

H

- Hotmail, 102

I

- identyfikator
 - klienta, 92
 - pakietu aplikacji, 92

- SID, 128
- ikony, 34
- ikony wskaźników, badge, 13
- implementacja kamery, 76
- zadań w tle, 137
- interfejs
 - Modern UI, 20
 - użytkownika, 42

J

- język
 - C#, 16
 - C++, 16
 - JavaScript, 16
 - LINQ, 87
 - Visual Basic, 16
 - XAML, 16, 41

K

- kafelek, tile, 35, 107, 114
 - podstawowy, primary tile, 115
 - pomocniczy, secondary tile, 122
- kanał
 - informacyjny, 103
 - kontroli, 134
 - powiadomień, 128
 - RSS, 103
- klasa
 - ApplicationData, 84
 - FileOpenPicker, 75
 - KanalyRSS, 104
 - ObservableCollection<T>, 105
 - Spotkanie, 49, 88
 - SystemCondition, 136
 - SystemTrigger, 135
 - ToastNotifier, 114
 - Windows.Web.Syndication, 103
- kolekcja, 50
- konfiguracja aplikacji, 92
- konto
 - deweloperskie, 11, 91
 - Microsoft Account, 32

kontrakt, 63
 Udostępnianie danych, 67
 Ustawienia, 69
 Wyszukiwanie, 64, 67

kontrolka
 AppBar, 57
 Grid, 45, 49
 GridView, 48
 ListView, 47
 PopUp, 55
 StackPanel, 45
 TextBlock, 43
 typu Image, 77
 korzyści, 155

L

licencja deweloperska, 13, 32
 LINQ, Language Integrated Query, 87
 lista GridView, 51
 listy, 48
 Live SDK, 11
 Live SDK 5.3, 91
 logowanie do aplikacji, 98

Ł

ładowanie treści do klas, 105

M

mechanizm powiadomień, 109
 metoda
 GetAccessStatus, 121
 LoadState, 65
 RemoveAccess, 122
 SaveState, 60
 metody asynchroniczne, 17
 model biznesowy Freemium, 151
 modele zarabiania aplikacji, 148
 moduł lokalizacji, 79
 modyfikator sealed, 137

N

należności, 155
 narzędzie
 Event Viewer, 140
 Remote Debugger, 40
 nawigacja
 hierarchiczna, 52
 płaska, 52
 pomiędzy stronami, 54
 Notifications Extensions, 11

O

obiekt
 typu MediaElement, 74
 typu DataPackage, 67

obsługa
 bazy danych, 89
 języka LINQ, 87
 kamery, 76
 kliknięcia, 66
 lokalizacji, 78
 multimediiów, 73
 testowej licencji, 150
 wyskakujących powiadomień, 108
 zadania w tle, 137
 zakupu wewnątrz aplikacji, 152
 zdarzenia, 51, 70
 oceny i komentarze aplikacji, 153
 odstępy pomiędzy elementami, 44
 odwołanie wyzwalacza, 139
 ograniczenia zadań, 133
 okna wysuwane, 71
 okno
 FlyOut, 55, 71
 Manage NuGet Packages, 72
 Properties, 43
 Quick Launch, 30
 statystyk, 154
 opcja Location, 78
 operacje CRUD, 132
 operator await, 129
 otwieranie zewnętrznego linku, 69

P

paczka z aplikacją, 33
 panel Solution Explorer, 33, 34
 pasek
 aplikacji, 25, 56
 Charms, 20–22, 63
 nawigacji, 56
 poleceń górny, 21
 platforma tworzenia aplikacji, 16
 plik
 App.xaml, 33
 App.xaml.cs, 33, 98
 licencja.xml, 149
 MainPage.xaml, 33, 57, 72
 MainPage.xaml.cs, 33, 49, 70, 99
 manifestu
 opcja Toast capable, 108
 zakładka Declarations, 120, 138
 Package.appxmanifest, 33
 Roaming.cs, 85
 StandardStyles.xaml, 57
 pliki
 .mp4, 75
 .pfx, 33
 .wmv, 75
 code behind, 16
 płatności, 153
 pobieranie
 danych RSS, 105
 listy spotkań, 89
 współrzędnych, 79
 pole RoamingStorageQuota, 83

powiadomienia
 okresowe, 118
 okresowe dla wskaźników, 126
 planowane, 117
 push, 126, 131
 wyskakujące, 107
 wyskakujące długoterminowe, 109
 powiązanie, binding, 50, 109
 procesory ARM, 15
 programowanie
 asynchroniczne, 17
 dla Windows 8, 13
 synchroniczne, 17
 projekt WRC, 137
 przechwytywanie
 argumentów, 113
 powiadomień, 131
 zdjęć, 77
 przekazywanie URI kanału, 129
 przestrzeń na dane, 81
 przysyłanie powiadomień, 129
 publikowanie aplikacji, 143, 145
 punkty końcowe, endpoints, 127

R

referencje, 33
 do biblioteki, 72
 REST, 96
 rejestracja
 kontraktu udostępniania, 68
 wyzwalacza, 139
 zdarzenia, 70
 rejestracje wielokrotne, 70
 rejestrowanie
 aplikacji, 93
 konta, 91
 reklamy, 24, 148
 responsywność aplikacji, 20
 REST, 96
 roaming danych, 83
 RSS, 103

S

serializacja, 84
 serwer WNS, 128
 serwisy
 mobilny, 132
 w chmurze, 127
 SID, Package Security Identifier, 127
 sideload, 13
 skalowanie, 58
 Sklep Windows, 7, 20, 143
 SkyDrive, 100
 słowo kluczowe
 async, 120
 await, 18
 snapowanie, 58
 specyfika
 aplikacji, 21
 programowania, 13

SQLite for Windows Runtime, 12
 SSO, 98
 standard PCI DSS, 26
 stany aplikacji, 60, 62
 status weryfikacji aplikacji, 148
 statystyki, 153
 sterowanie aplikacją, 113
 strona SpotkaniaStrona.xaml, 52, 76
 styl elementu, 49
 style przycisków, 57
 symulator testowania aplikacji, 39
 szablon

aplikacji
 Blank App, 31
 Grid App, 31
 Split App, 31
 powiadomienia, 109
 strony, 45, 46

Ś

ścieżki dostępu, 96–98
 środowisko
 programistyczne, 10
 uruchomieniowe, 10

T

tabele, 88
 tajny klucz klienta, 92
 technologia WCF, 126
 testowanie aplikacji, 62
 na symulatorze, 38
 na zdalnym urządzeniu, 40
 trial
 czasowy, 149
 funkcjonalny, 149
 tryb
 Filled, 59
 Snapped, 13, 21, 41
 tworzenie
 aplikacji
 dostęp do usług, 94
 korzyści, 155
 model biznesowy, 147
 ograniczenia, 13
 oprogramowanie, 10
 podstawowe elementy, 34
 projektowanie wyglądu, 41
 wymagania, 10
 wymagania certyfikacyjne, 23
 zakres działania, 94
 zewewnętrzne komponenty, 11
 dokumentów XML, 109
 dynamicznych powiadomień, 114
 ekranu powitalnego, 37

interfejsu użytkownika, 41
 kanału powiadomień, 129
 listy, 50
 listy przewijanej, 47
 nowego projektu, 31
 powiadomienia okresowego, 118
 szablonu powiadomienia
 DOM, 111, 112
 NotificationsExtensions, 109
 szablonu strony, 45, 46
 wskaźnika, 125
 zaplanowanego powiadomienia,
 117

typy
 powiadomień, 120
 zapisywanych danych, 81

U

udostępnianie
 danych, 67
 treści, 68
 układ
 aplikacji, 47
 strony, 46
 uruchamianie projektu, 11
 usługi Live, 91
 ustawienia
 lokalne, 82
 roamingowe, 83, 84
 tymczasowe, 86
 ustawienie
 adresu zewnętrznego, 71
 HighPriority, 84
 strony, 71
 usuwanie
 kafelka, 124
 obiektu ze SkyDrive, 101
 powiadomienia, 113
 spotkania, 90
 wydarzenia, 103
 uwierzytelnienie aplikacji, 92, 128

V

Visual Studio 2012, 30

W

wersja trial, 23, 149
 widok Snapped, 58
 Windows
 Azure Mobile Services, 132
 Notification Service, WNS, 127
 Push Notification Service, 26
 Runtime, 16, 31

wirtualna maszyna, 30
 wskaźnik dla kafelka
 pomocniczego, 125
 wskaźniki, Badge, 107, 124
 wybór
 typu powiadomień, 120
 zadań, 121
 wygląd aplikacji, 44
 wyjątek LiveConnectException, 98
 wymagania certyfikacyjne
 aplikacji, 143
 funkcjonalność, 23, 26
 polityka prywatności, 25
 przewidywalność, 24
 reklamy, 24
 wysuwany interfejs ustawień, 73
 wyszukiwanie danych, 66
 wyświetlanie
 pliku multimedialnego, 75
 treści kafelka, 119
 współrzędnych, 78
 wyzwalacz, Trigger, 133
 ControlChannelTrigger, 134
 MaintenanceTrigger, 136
 PushNotificationTrigger, 136
 SystemTrigger, 135
 TimeTrigger, 136

Z

zablokowany ekran, 119
 zadanie w tle, Background Tasks
 implementacja, 137
 ograniczenia, 133
 warunki wystąpienia, 136
 zakres
 dostępu do usług, 95
 podstawowy, 96
 programisty, 96
 rozszerzony, 96
 zapisywanie
 multimediów, 87
 na dysku SkyDrive, 77
 ustawień roamingowych, 84
 zdarzenie
 handler, 68
 CommandRequested, 69
 Tapped, 51
 zmiany danych, 85
 zmiana stanu aplikacji, 61, 62
 zrzutowanie wartości, 83

Ż

żądanie HTTP, 130

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**

Tworzenie aplikacji dla systemu Windows 8

Zaprojektuj, napisz i opublikuj swoją aplikację

Tworzenie aplikacji dla nowego systemu operacyjnego firmy Microsoft to całkiem ciekawa (a czasem także intratna) perspektywa dla programistów chcących dotrzymać kroku zmieniającym się technologiom i wyjść naprzeciw potrzebom użytkowników. Platforma Windows 8, z zupełnie odmiennym interfejsem, oferuje nowe możliwości, m.in. w zakresie wykorzystania ekranów dotykowych czy integracji między aplikacją i systemem, ale jednocześnie wymaga od projektantów dostosowania się do nowych reguł gry, by pisane przez nich aplikacje nie odbiegały jakością, wyglądem i sposobem działania od reszty elementów charakterystycznych dla tej wersji systemu Windows.

Jeśli chcesz pisać takie programy i umieszczać je w Sklepie Windows, koniecznie musisz zacząć od tej książki. Dzięki niej zbudujesz od podstaw własną aplikację! Znajdziesz tu informacje na temat platformy tworzenia aplikacji dla systemu Windows 8, nauczysz się budować układ aplikacji, projektować jej wygląd i zapewniać funkcjonalność. Zrozumiesz, jak ważne są kwestie zarządzania danymi i ich wymiany, jak wykorzystać potęgę internetu do wspomaganie pracy Twojej aplikacji, jak działają zadania w tle. Dowiesz się także, jak opublikować gotową aplikację w Sklepie Windows. Prawda, że warto?

- Platforma tworzenia aplikacji w systemie Windows 8
- Wymagania certyfikacyjne aplikacji dla systemu Windows 8
- Tworzenie aplikacji dla systemu Windows 8
- Wygląd aplikacji
- Możliwości aplikacji dla systemu Windows 8
- Zarządzanie danymi i synchronizacja z chmurą
- Współpraca z usługami online
- Powiadomienia
- Zadania w tle
- Publikacja w Sklepie Windows

Bądź na bieżąco, twórz aplikacje dla systemu Windows 8!

helion.pl
księgarnia internetowa

Nr katalogowy: 13844



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>

sięgnij po **WIĘCEJ**



KOD KORZYŚCI

ISBN 978-83-246-7093-2



9 788324 670932

Cena: 37,00 zł

Informatyka w najlepszym wydaniu