

Spis treści (skrótowy)

	Wprowadzenie	xxiii
1	Zaczynamy. <i>Skok na głęboką wodę</i>	1
2	Tworzenie interaktywnych aplikacji. <i>Aplikacje, które coś robią</i>	39
3	Wiele aktywności i intencji. <i>Jakie są Twoje intencje?</i>	73
4	Cykl życia aktywności. <i>Była sobie aktywność</i>	115
5	Interfejs użytkownika. <i>Podziwiał widoki</i>	163
6	Widoki list i adaptery. <i>Zorganizuj się</i>	227
7	Fragmety. <i>Zadbaj o modularyzację</i>	269
8	Fragmety zagnieżdżone. <i>Zadbaj o potomstwo</i>	325
9	Paski akcji. <i>Na skróty</i>	365
10	Szuflady nawigacyjne. <i>Z miejsca na miejsce</i>	397
11	Bazy danych SQLite. <i>Odpal bazę danych</i>	437
12	Kursory i zadania asynchroniczne. <i>Nawiązywanie połączenia z bazą danych</i>	471
13	Usługi. <i>Do usług</i>	541
14	Material Design. <i>W materialistycznym świecie</i>	597
A	ART. <i>Środowisko uruchomieniowe Androida</i>	649
B	ADB. <i>Android Debug Bridge</i>	653
C	Emulator. <i>Emulator Androida</i>	659
D	Pozostałości. <i>Dziesięć najważniejszych zagadnień (których nie opisaliśmy)</i>	663

Spis treści (z prawdziwego zdarzenia)

W

Wprowadzenie

Twój mózg jest nastawiony na Androida. Jesteś tu po to, by się czegoś nauczyć, natomiast Twój mózg robi Ci przysługę, upewniając się, że to, czego się nauczyłeś, szybko wyleci z pamięci. Twój mózg myśli sobie: „Lepiej zostawić miejsce na coś ważnego, na przykład: których dzikich zwierząt lepiej unikać albo czy jeżdżenie nago na snowboardzie to dobry pomysł”. A zatem, w jaki sposób możesz skłonić swój mózg, by myślał, że Twoje życie zależy od umiejętności pisania aplikacji na Androida?

	Dla kogo jest przeznaczona ta książka?	xxiv
	Wiemy, co sobie myślisz	xxv
	Wiemy, co sobie myśli Twój mózg	xxv
	Metapoznanie — myślenie o myśleniu	xxvii
	Oto co MY zrobiliśmy	xxviii
	Przeczytaj to	xxx
	Zespół recenzentów technicznych	xxxii
	Podziękowania	xxxiii

Zaczynamy

1

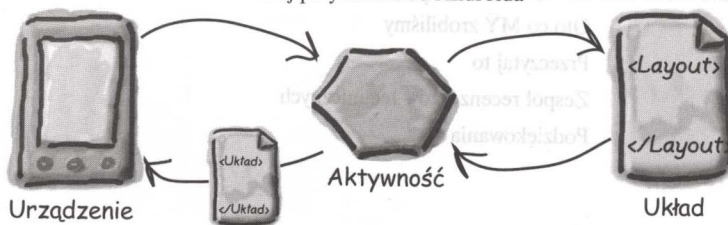
Skok na głęboką wodę

Android błyskawicznie podbił świat.

Każdy chce mieć smartfon lub tablet, a urządzenia z Androidem są niezwykle popularne. W tej książce nauczymy Cię, jak **pisać własne aplikacje**, a zaczniemy od pokazania procesu przygotowania bardzo prostej aplikacji i uruchomienia jej na wirtualnym urządzeniu z Androidem. W trakcie tych prac poznasz także kilka podstawowych komponentów wszystkich aplikacji na Androida, takich jak **aktywności i układy**. **Jedyną rzeczą, której będziesz do tego potrzebować, jest znajomość Javy, choć wcale nie musisz być w niej mistrzem...**



Witamy w Androidowie	2
Platforma Android w szczegółach	3
Środowisko programistyczne	5
Zainstaluj Javę	6
Stwórzmy prostą aplikację	8
Aktywności z wysokości 15 tysięcy metrów	12
Tworzenie aplikacji (ciąg dalszy)	13
Tworzenie aplikacji (ciąg dalszy)	14
Właśnie utworzyłeś swoją pierwszą aplikację na Androida	15
Android Studio utworzy pełną strukturę katalogów aplikacji	16
Przydatne pliki projektu	17
Edycja kodu z użyciem edytorów Android Studio	18
Uruchamianie aplikacji w emulatorze Androida	23
Tworzenie wirtualnego urządzenia z Androidem	24
Uruchomienie aplikacji w emulatorze	27
Postępy możesz obserwować w konsoli	28
Jazda próbna	29
Ale co się właściwie stało?	30
Usprawnianie aplikacji	31
Czym jest układ?	32
Plik activity_main.xml zawiera dwa elementy	33
Plik układu zawiera odwołanie do łańcucha, a nie sam łańcuch znaków	34
Zajrzyjmy do pliku strings.xml	35
Weź swoją aplikację na jazdę próbna	37
Twój przyborek do Androida	38



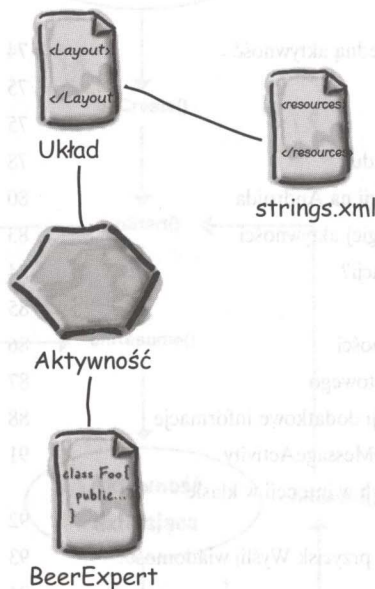
Tworzenie interaktywnych aplikacji

Aplikacje, które coś robią

2

Większość aplikacji musi w jakiś sposób reagować na poczynania użytkowników.

Z tego rozdziału dowiesz się, co zrobić, aby Twoje aplikacje były **nieco bardziej interaktywne**. Przekonasz się, jak zmusić aplikację, by coś **zrobiła** w odpowiedzi na działania użytkownika, oraz jak sprawić, by **aktywności i układy porozumiewały się ze sobą** jak starzy kumple. Przy okazji pokażemy Ci **nieco dokładniej, jak naprawdę działa Android** — poznasz plik **R**, czyli ukryty klejnot, który spaja pozostałe elementy aplikacji.



W tym rozdziale napiszemy aplikację Doradca piwny	40
Utworzenie projektu	42
Utworzyliśmy domyślną aktywność i układ	43
Dodawanie komponentów w edytorze projektu	44
Plik activity_find_beer.xml zawiera nowy przycisk	45
Zmiany w kodzie XML układu...	48
...są uwzględniane w edytorze projektu	49
Stosuj zasoby łańcuchowe, a nie łańcuchy podawane w kodzie	50
Zmiana układu i zastosowanie w nim zasobów łańcuchowych	51
Weź swoją aplikację na jazdę próbną	52
Dodanie wartości do komponentu Spinner	53
Dodanie do komponentu Spinner odwołania do string-array	54
Jazda próbna komponentu Spinner	54
Musimy zadbać o to, by przycisk coś robił	55
Niech przycisk wywoła metodę	56
Jak wygląda kod aktywności?	57
Dodaj do aktywności metodę onClickFindBeer()	58
Metoda onClickFindBeer() musi coś robić	59
Dysponując obiektem View, można odwoływać się do jego metod	60
Aktualizacja kodu aktywności	61
Pierwsza wersja aktywności	63
Jazda próbna — test modyfikacji	65
Tworzenie własnej klasy Javy	66
Dodaj do aktywności wywołanie metody naszej klasy, aby była wyświetlana FAKTYCZNA porada	67
Kod aktywności, wersja 2	69
Co się dzieje podczas wykonywania tego kodu?	70
Jazda próbna — test aplikacji	71
Twój przyborek do Androida	72

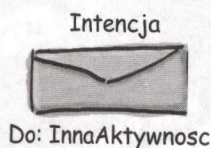
Wiele aktywności i intencji

3

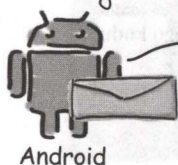
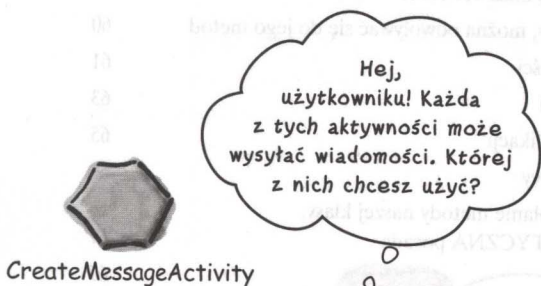
Jakie są Twoje intencje?

Większość aplikacji potrzebuje więcej niż jednej aktywności.

Dotychczas mieliśmy do czynienia z aplikacjami składającymi się tylko z jednej aktywności. Kiedy jednak sprawy się komplikują, jedna aktywność zwyczajnie nie wystarczy. Dlatego w tym rozdziale pokażemy Ci, jak **tworzyć aplikacje składające się z wielu aktywności** i jak nasze aplikacje mogą porozumiewać się z innymi, wykorzystując w tym celu **intencje**. Pokażemy także, jak można używać intencji, by **wykraczać poza granice naszych aplikacji**, i jak wykorzystywać **aktywności należące do innych aplikacji dostępnych w urządzeniu do wykonywania akcji**. To wszystko zapewni nam znacznie większe możliwości.



Aplikacja może zawierać więcej niż jedną aktywność	74
Oto struktura naszej aplikacji	75
Utworzenie projektu	75
Utworzenie drugiej aktywności i układu	78
Przedstawiamy plik manifestu aplikacji na Androida	80
Użycie intencji do uruchomienia drugiej aktywności	83
Co się dzieje po uruchomieniu aplikacji?	84
Jazda próbna aplikacji	85
Przekazanie tekstu do drugiej aktywności	86
Aktualizacja właściwości widoku tekstowego	87
Metoda putExtra() zapisuje w intencji dodatkowe informacje	88
Aktualizacja kodu aktywności CreateMessageActivity	91
Zastosowanie informacji przekazanych w intencji w klasie ReceiveMessageActivity	92
Co się dzieje, gdy użytkownik kliknie przycisk Wyślij wiadomość?	93
Jazda próbna aplikacji	94
Jak działają aplikacje na Androida?	95
Co się dzieje podczas działania kodu?	99
Jak Android korzysta z filtrów intencji?	102
Musisz uruchomić aplikację na PRAWDZIWYM urządzeniu	105
Jazda próbna aplikacji	107
Zmień kod, aby wyświetlać okno dialogowe	111
Jazda próbna aplikacji	112
Twój przybornik do Androida	114



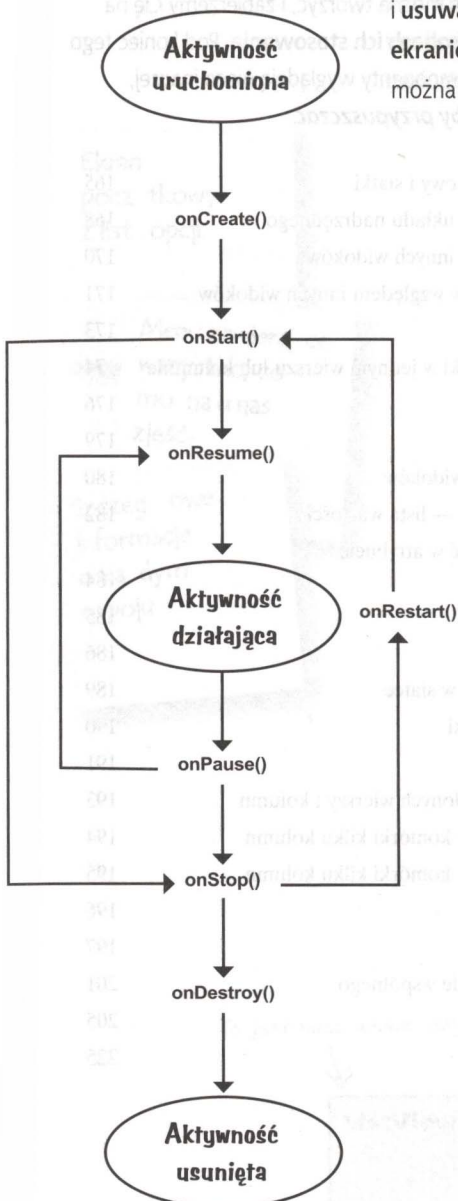
Cykl życia aktywności

Była sobie aktywność

4

Aktywności stanowią podstawę wszystkich aplikacji na Androida.

Wiesz już, jak tworzyć aktywności i jak sprawić, by jedna aktywność uruchomiła drugą, używając do tego celu intencji. Ale **co tak naprawdę dzieje się za kulisami?** W tym rozdziale nieco dokładniej poznamy **cykl życia aktywności**. Co się dzieje, kiedy aktywność **jest tworzona i usuwana**? Jakie metody są wywoływane, gdy aktywność jest **wyświetlana i pojawia się na ekranie**, a jakie gdy aktywność **traci miejsce wprowadzania i jest ukrywana**? W jaki sposób można **zapisywać i odtwarzać stan aktywności**?



Jak właściwie działają aktywności?	116
Aplikacja stopera	118
Kod układu aplikacji stopera	119
Dodanie kodu obsługującego przyciski	122
Metoda runTimer()	123
Obiekty Handler umożliwiają planowanie wykonania kodu	124
Pełny kod metody runTimer()	125
Kompletny kod aktywności StopwatchActivity	126
Obrót ekranu zmienia konfigurację urządzenia	132
Od narodzin do śmierci: stany aktywności	133
Cykl życia aktywności: od utworzenia do usunięcia	134
W jaki sposób radzić sobie ze zmianami konfiguracji?	136
Co się stanie po uruchomieniu aplikacji?	139
Tworzenie i usuwanie to nie cały cykl życia aktywności	142
Cykl życia aktywności: widzialny czas życia	143
Zaktualizowany kod aktywności StopwatchActivity	147
Co się dzieje podczas działania aplikacji?	148
Jazda próbna aplikacji	149
A co się dzieje, jeśli aplikacja jest tylko częściowo widoczna?	150
Cykl życia aktywności: życie na pierwszym planie	151
Zatrzymanie stopera w razie wstrzymania aktywności	154
Kompletny kod aktywności	157
Wygodny przewodnik po metodach cyklu życia aktywności	161
Twój przyborek do Androida	162

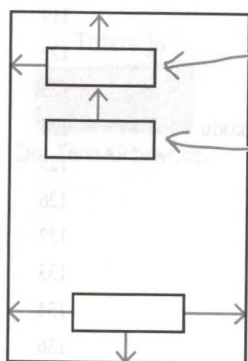
Interfejs użytkownika

5

Podziwiają widoki

Nie masz innego wyjścia, musisz tworzyć szalowe układy.

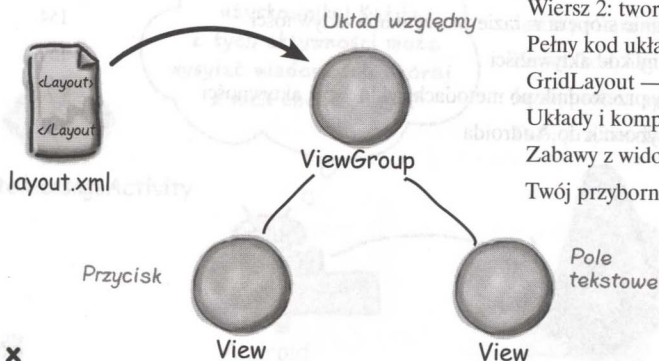
Jeśli chcesz pisać aplikacje, których inni będą **używać**, musisz zadbać o to, by **wyglądały one dokładnie tak, jak sobie tego życzysz**. Zagadnienie tworzenia układów potraktowaliśmy dotychczas bardzo powierzchownie, najwyższy zatem czas, by **przyjrzeć mu się dokładniej**. W tym rozdziale pokażemy Ci różne **typy układów**, które można tworzyć, i zabierzemy Cię na wycieczkę po najważniejszych **komponentach GUI** i **sposobach ich stosowania**. Pod koniec tego rozdziału przekonasz się, że choć wszystkie te układy i komponenty wyglądają nieco inaczej, to jednak mają ze sobą **więcej wspólnego, niż można by przypuszczać**.



Widoki mogą być rozmieszczane względem układu nadrzędnego...

... bądź względem innych widoków.

Trzy kluczowe układy: względny, liniowy i siatki	165
Rozmieszczanie widoków względem układu nadrzędnego	168
Rozmieszczanie widoków względem innych widoków	170
Atrybuty do rozmieszczania widoków względem innych widoków	171
RelativeLayout — podsumowanie	173
Układ LinearLayout wyświetla widoki w jednym wierszu lub kolumnie	174
Zmieńmy nieco prosty układ liniowy	176
Dodawanie wagi do widoków	179
Dodawanie wagi do większej liczby widoków	180
Stosowanie atrybutu android:gravity — lista wartości	182
Inne wartości, których można używać w atrybucie android:layout_gravity	184
Kompletny układ liniowy	185
LinearLayout — podsumowanie	186
Układ GridLayout wyświetla widoki w siatce	189
Dodawanie widoków do układu siatki	190
Utwórzmy nowy układ siatki	191
Wiersz 0: dodajemy widoki do określonych wierszy i kolumn	193
Wiersz 1: tworzymy widok zajmujący komórki kilku kolumn	194
Wiersz 2: tworzymy widok zajmujący komórki kilku kolumn	195
Pełny kod układu siatki	196
GridLayout — podsumowanie	197
Układy i komponenty GUI mają wiele wspólnego	201
Zabawy z widokami	205
Twój przybornik do Androida	225



Widoki list i adaptery

Zorganizuj się

6

Chcesz wiedzieć, jaki jest najlepszy sposób na określenie struktury aplikacji?

Znasz już podstawowe elementy konstrukcyjne używane do tworzenia aplikacji, więc teraz nadszedł czas, **żebyś się lepiej zorganizował**. W tym rozdziale pokażemy Ci, jak możesz **przekształcić** zbiór pomysłów **w niesamowitą aplikację**. Zobaczysz, że **listy danych** mogą stać się kluczowym elementem projektu aplikacji i że **łączenie ich** może prowadzić do powstania **aplikacji łatwej w użyciu i zapewniającej ogromne możliwości**. Przy okazji zapoznasz się z **obiektami nasłuchującymi i adapterami**, dzięki którym Twoja aplikacja stanie się bardziej dynamiczna.

Ekran
pocztkowy
z list opcji

Menu zawieraj
ce wszystko,
co ma na
nas zjeść

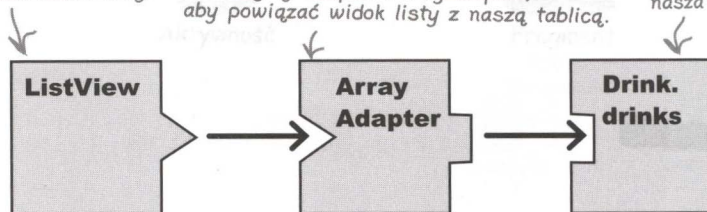
Szczeg
owe
informacje
o kawy
napoju

Każda aplikacja zaczyna się od pomysłu	228
Skategoryzuj swoje pomysły — aktywności: poziom główny, kategoria i szczegóły/edycja	229
Nawigowanie po aktywnościach	230
Użyj ListView do nawigowania po danych	231
Napiszemy aplikację kafeтерии Coffeina	232
Aktywność szczegółów napoju	233
Struktura aplikacji dla kafeтерии Coffeina	234
Układ aktywności głównego poziomu składa się z obrazka i listy	238
Kompletny kod układu aktywności głównego poziomu	240
Zapewnianie reakcji ListView na kliknięcia za pomocą obiektu nasłuchującego	241
Kompletny kod aktywności TopLevelActivity	243
Jak utworzyć aktywność listy?	249
Łączenie widoków list z tablicami za pomocą adaptera ArrayAdapter	251
Dodanie adaptera ArrayAdapter do aktywności DrinkCategoryActivity	252
Co się stanie po wykonaniu kodu?	253
Jak obsługiwaliśmy kliknięcia w aktywności TopLevelActivity?	256
Kompletny kod aktywności DrinkCategoryActivity	258
Aktywność szczegółów wyświetla informacje o jednym rekordzie	259
Wypełnienie widoków danymi	261
Kod aktywności DrinkActivity	263
Jazda próbna aplikacji	266
Twój przyborek do Androida	268

To jest nasz widok listy.

Utworzymy adapter ArrayAdapter,
aby powiązać widok listy z naszą tablicą.

To jest
nasza tablica.



Fragmenty

7

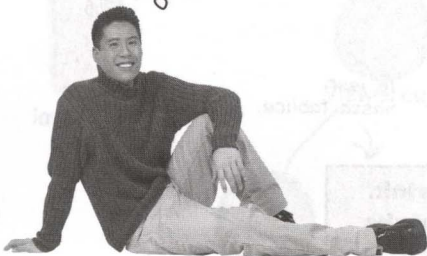
Zadbaj o modularyzację

Wiesz już, jak tworzyć aplikacje, które działają tak samo niezależnie od tego, na jakim urządzeniu zostały uruchomione...

...ale co zrobić w przypadku, kiedy akurat chcesz, by aplikacja **wyglądała i działała inaczej** w zależności od tego, czy zostanie uruchomiona **na telefonie, czy na tablecie**? W tym rozdziale pokażemy Ci, co zrobić, aby aplikacja wybierała **układ, który najlepiej pasuje do wielkości ekranu urządzenia**. Oprócz tego przedstawimy **fragmenty**, czyli **modularne komponenty kodu**, które mogą być **wielokrotnie używane przez różne aktywności**.

Struktura aplikacji Trener	273
Klasa Workout	275
Jak dodać fragment do projektu?	276
Jak wygląda kod fragmentu?	278
Przypomnienie stanów aktywności	282
Cykl życia fragmentów	283
Nasze fragmenty dziedziczą metody cyklu życia	284
Jazda próbna aplikacji	286
Jak utworzyć fragment typu ListFragment?	290
Zaktualizowany kod klasy WorkoutListFragment	292
Jazda próbna aplikacji	294
Powiązanie listy z widokiem szczegółów	295
Stosowanie transakcji fragmentu	301
Zaktualizowany kod aktywności MainActivity	302
Jazda próbna aplikacji	303
Kod fragmentu WorkoutDetailFragment	305
Struktury aplikacji na tablety i telefony	307
Różne opcje katalogów	309
Układ MainActivity dla telefonów	315
Kompletny kod aktywności DetailActivity	319
Zmodyfikowany kod aktywności MainActivity	321
Jazda próbna aplikacji	322

A zatem fragment będzie zawierał jedną listę. Kiedy chcieliśmy użyć aktywności zawierającej pojedynczą listę, zastosowaliśmy aktywność typu ListActivity. Zastanawiam się, czy przypadkiem nie istnieje jakiś typ fragmentu stanowiący odpowiednik tej klasy.



Fragменты zagnieżdżone

Zadbaj o potomstwo

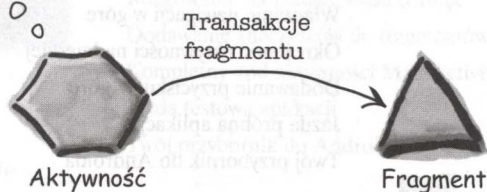
18

Wiesz już, że stosowanie fragmentów w aktywnościach pozwala na wielokrotne wykorzystywanie kodu i zwiększa elastyczność aplikacji.

W tym rozdziale mamy zamiar pokazać Ci, jak zagnieżdżać fragmenty w innych fragmentach. Dowiesz się, jak używać menedżera fragmentów podrzędnych, by poskromić niesforne transakcje. Ponadto dowiesz się, dlaczego tak ważna jest **znajomość różnic między aktywnościami i fragmentami**.

Tworzenie zagnieżdżonych fragmentów	326
Kod fragmentu StopwatchFragment	332
Układ fragmentu StopwatchFragment	335
Metoda getFragmentManager() tworzy transakcje na poziomie aktywności	340
Zagnieżdżone fragmenty wymagają zagnieżdżonych transakcji	341
Kompletny kod fragmentu WorkoutDetailFragment	343
Jazda próbna aplikacji	344
Dlaczego kliknięcie przycisku powoduje awarię aplikacji?	345
Przyjrzyjmy się kodowi układu StopwatchFragment	346
Zaimplementuj we fragmencie interfejs OnClickListener	349
Powiązanie obiektu nasłuchującego OnClickListener z przyciskami	351
Kod fragmentu StopwatchFragment	352
Jazda próbna aplikacji	354
Kod fragmentu WorkoutDetailFragment	358
Jazda próbna aplikacji	359
Twój przyborek do Androida	364

Jak widzę, pojawiły się jakieś transakcje fragmentu. Muszę je natychmiast zastosować.



Paski akcji

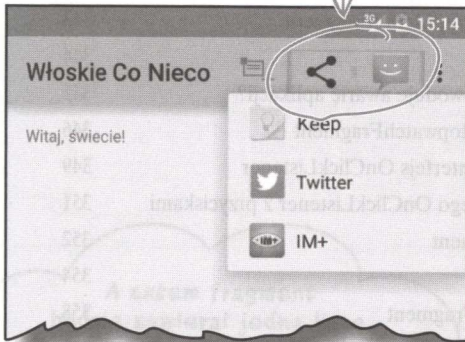
Na skróty

9

Każdy lubi chodzić na skróty.

Z tego rozdziału dowiesz się, jak korzystając z pasków akcji, wzbogacić aplikację o możliwość chodzenia na skróty. Pokażemy Ci, jak uruchamiać inne aplikacje za pomocą *elementów akcji* dodawanych do pasków akcji, jak udostępniać treści innym aplikacjom, używając *dostawcy akcji współdzielenia*, oraz jak poruszać się w górę hierarchii aplikacji za pomocą przycisku *W górę* umieszczonego na *pasku akcji*. Dowiesz się też, jak można nadawać tworzonym aplikacjom spójny wygląd i sposób działania, korzystając z **motywów**, i poznasz *pakiet biblioteki wsparcia systemu Android*.

Właśnie tak wygląda akcja udostępniania wyświetlona na pasku akcji. Po jej kliknięciu wyświetlana jest lista aplikacji, którym możemy udostępnić treści.

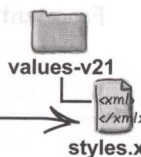


Świetne aplikacje mają przejrzystą strukturę	366
Różne typy nawigacji	367
Zacznijmy od paska akcji	368
Pakiet bibliotek Support Libraries	369
Twój projekt może już używać bibliotek wsparcia	370
Zadbamy, by aplikacja używała aktualnych motywów	371
Zastosowanie motywu w pliku AndroidManifest.xml	372
Definiowanie stylów w pliku zasobów stylów	373
Określenie domyślnego motywu w pliku styles.xml	374
Co się dzieje podczas działania aplikacji?	375
Dodawanie elementów do paska akcji	376
Plik zasobów menu	377
Atrybut showAsAction menu	378
Dodawanie nowego elementu akcji	379
Utworzenie aktywności OrderActivity	382
Uruchomienie aktywności OrderActivity po kliknięciu przycisku Złóż zamówienie	383
Kompletny kod aktywności MainActivity	384
Dzielenie się treściami z poziomu paska akcji	386
Określanie treści za pomocą intencji	388
Kompletny kod aktywności MainActivity	389
Włączanie nawigacji w górę	391
Określanie aktywności nadrzędnej	392
Dodawanie przycisku W górę	393
Jazda próbna aplikacji	394
Twój przyborek do Androida	395

API 21?
Idealnie pasuje.



Android



Name: AppTheme
Parent: Theme.Material.Light

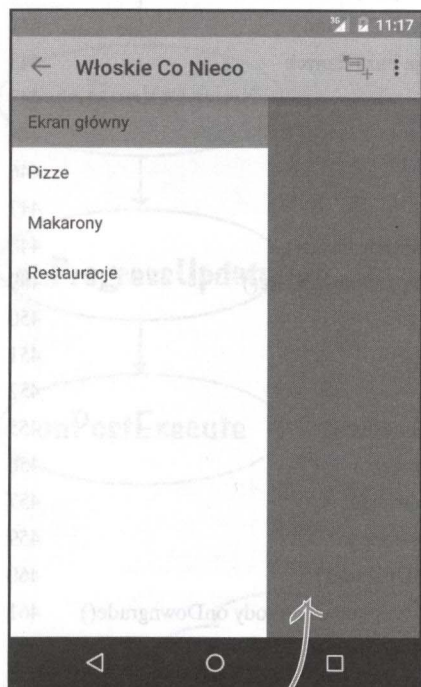
10

Szufłady nawigacyjne

Z miejsca na miejsce

Aplikacje są nieporównanie lepsze, gdy można się po nich łatwo poruszać.

W tym rozdziale przedstawimy Ci **szufładę nawigacyjną** — wysuwany panel, który jest wyświetlany na ekranie po przesunięciu palcem lub kliknięciu ikony umieszczonej na pasku akcji. Pokażemy Ci, jak można wyświetlać w niej **listę odnośników** umożliwiających przechodzenie do **kluczowych węzłów aplikacji**. Oprócz tego przekonasz się, że **przełączanie fragmentów** pozwala **łatwo docierać do tych węzłów** i je **szybko wyświetlać**.



Zawartość ekranu jest wyświetlana w układzie `FrameLayout`. Chcemy, by wypełniła ona cały dostępny obszar ekranu. W tym przykładzie jest ona częściowo przesłonięta przez szufładę nawigacyjną.

Zmiany w aplikacji dla restauracji Włoskie Co Nieco	398
Szufłady nawigacyjne bez tajemnic	399
Struktura aplikacji dla restauracji Włoskie Co Nieco	400
Utworzenie fragmentu <code>TopFragment</code>	401
Utworzenie fragmentu <code>PizzaFragment</code>	402
Utworzenie fragmentu <code>PastaFragment</code>	403
Utworzenie fragmentu <code>StoresFragment</code>	404
Dodanie układu <code>DrawerLayout</code>	405
Kompletna zawartość pliku <code>activity_main.xml</code>	406
Inicjalizacja listy szufłady nawigacyjnej	407
Zmiana tytułu paska akcji	412
Zamykanie szufłady nawigacyjnej	413
Zaktualizowany kod pliku <code>MainActivity.java</code>	414
Stosowanie <code>ActionBarDrawerToggle</code>	417
Modyfikowanie elementów paska akcji w trakcie działania aplikacji	418
Zaktualizowany kod aktywności <code>MainActivity</code>	419
Włączenie możliwości otwierania i zamykania szufłady nawigacyjnej	420
Synchronizacja stanu przycisku <code>ActionBarDrawerToggle</code>	421
Zaktualizowany kod aktywności <code>MainActivity</code>	422
Obsługa zmian konfiguracji	425
Reagowanie na zmiany stosu cofnięć	426
Dodawanie znaczników do fragmentów	427
Kompletny kod aktywności <code>MainActivity</code>	429
Jazda testowa aplikacji	435
Twój przybornik do Androida	436

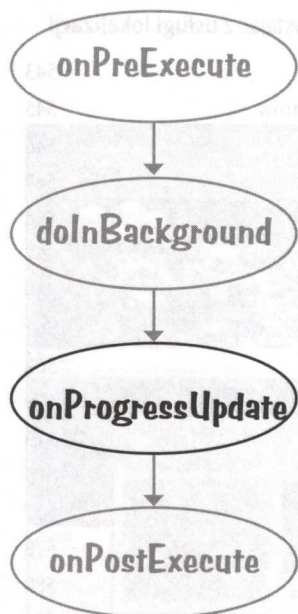
Kursory i zadania asynchroniczne

Nawiązywanie połączenia z bazą danych

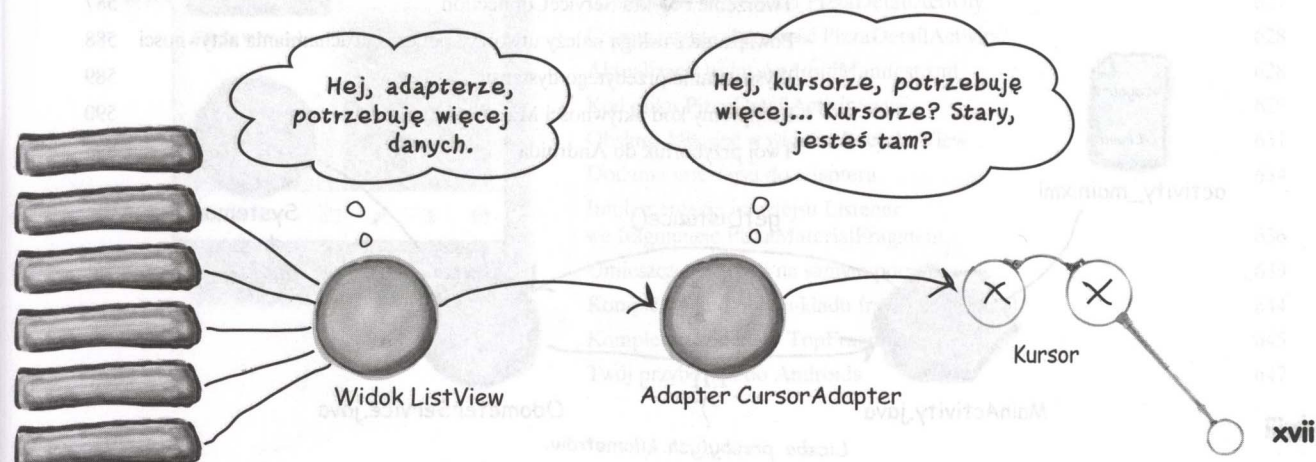
12

Jak łączysz swoje aplikacje z bazami danych SQLite?

Dotychczas dowiedziałeś się, jak tworzyć bazy danych, używając pomocnika SQLite. Kolejnym krokiem będzie uzyskanie dostępu do tych baz danych w aktywnościach. Z tego rozdziału dowiesz się, jak **tworzyć kursory**, **by pobierać dane z bazy danych**, **jak poruszać się po kursorach** oraz **jak pobierać z nich dane**. Oprócz tego dowiesz się, jak używać **adapterów kursorów**, by łączyć kursory z widokami list. A na koniec przekonasz się, że pisanie wydajnego **kodu wielowątkowego** korzystającego z klasy **AsyncTask** może zagwarantować wysoką wydajność działania aplikacji.



Aktualny kod aktywności DrinkActivity	474
Określanie tabeli i kolumn	478
Zapytania z wieloma warunkami	479
Stosowanie funkcji SQL w zapytaniach	481
Poruszanie się po kursorze	488
Pobieranie wartości z kursora	489
Kod aktywności DrinkActivity	490
Dodanie ulubionych napojów do aktywności DrinkActivity	508
Kod aktywności DrinkActivity	513
Nowy kod aktywności głównego poziomu	518
Zmodyfikowany kod aktywności TopLevelActivity	524
Metoda onPreExecute()	531
Metoda doInBackground()	532
Metoda onProgressUpdate()	533
Metoda onPostExecute()	534
Klasa AsyncTask	535
Kod aktywności DrinkActivity	537
Twój przybornik do Androida	540



Bazy danych SQLite

11

Odpal bazę danych

Jeśli rejestrujesz najlepsze wyniki lub przesyłane komunikaty, to Twoja aplikacja będzie musiała przechowywać dane.

A w Androidzie dane są zazwyczaj bezpiecznie i trwale przechowywane w **bazach danych SQLite**. W tym rozdziale pokażemy Ci, jak **utworzyć bazę danych, dodawać do niej tabele, wypełnić ją wstępnie danymi**, a wszystko to za pomocą **pomocnika SQLite**. Dowiesz się też, w jaki sposób można bezproblemowo przeprowadzać **aktualizacje** struktury bazy danych i jak w razie konieczności wycofania zmian wrócić do jej **wcześniejszych wersji**.

Jaśnie panie,
oto pańska baza
danych. Czy mogę
jeszcze czymś służyć?

onCreate()

Pomocnik SQLite

Nazwa: Coffeina
Wersja: 1

DRINK

Baza danych SQLite

Znowu w kafeterii Coffeina	438
Android trwale przechowuje dane, używając baz danych SQLite	439
Android udostępnia kilka klas związanych z SQLite	440
Obecna struktura aplikacji kafeterii Coffeina	441
Pomocnik SQLite zarządza Twoją bazą danych	443
Pomocnik SQLite	443
Tworzenie pomocnika SQLite	444
Wnętrze bazy danych SQLite	446
Tabele tworzymy w języku SQL	447
Wstawianie danych za pomocą metody insert()	448
Aktualizacja rekordów za pomocą metody update()	449
Określanie wielu warunków	450
Kod klasy CoffeinaDatabaseHelper	451
Co robi kod pomocnika SQLite?	452
Co zrobić, gdy trzeba będzie zmienić bazę?	455
Bazy danych SQLite mają numer wersji	456
Aktualizacja bazy danych — omówienie	457
Jak pomocnik SQLite podejmuje decyzje?	459
Aktualizacja bazy w metodzie onUpgrade()	460
Przywracanie starszej wersji bazy za pomocą metody onDowngrade()	461
Zaktualizujmy bazę danych	462
Aktualizacja istniejącej bazy danych	465
Zmiana nazwy tabeli	466
Pełny kod pomocnika SQLite	467
Kod pomocnika SQLite (ciąg dalszy)	468
Co się dzieje podczas działania kodu?	469
Twój przyborek do Androida	470

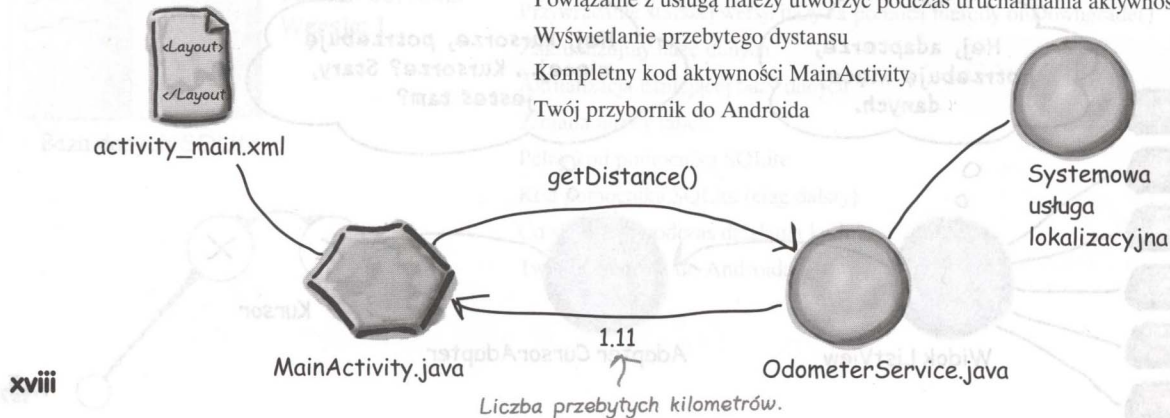
13

Usługi Do usług

Są operacje, które będziemy chcieli realizować niezależnie od tego, która aplikacja jest widoczna na ekranie.

Na przykład kiedy rozpoczniesz odtwarzanie pliku w aplikacji muzycznej, to najprawdopodobniej będziesz oczekiwać, że będzie ona kontynuowała odtwarzanie nawet wówczas, gdy przejdziesz do innej aplikacji. Z tego rozdziału dowiesz się, jak używać **usług**, by radzić sobie w sytuacjach takich jak ta. Przy okazji nauczysz się korzystać z kilku **wbudowanych usług systemu Android**. Dowiesz się, jak za pomocą **usługi powiadomień** sprawić, by użytkownik zawsze był dobrze poinformowany, i jak poznać aktualne położenie geograficzne, korzystając z **usługi lokalizacji**.

Aplikacja z usługą uruchomioną	543
Usługa IntentService z wysokości 15 tysięcy metrów	545
Jak rejestrować komunikaty?	546
Kompletny kod usługi DelayedMessageService	547
Kompletny kod usługi DelayedMessageService	554
Jak używać usługi powiadomień?	557
Uruchamianie intencji przez powiadomienie	559
Wysyłanie powiadomień za pomocą usługi systemowej	561
Kompletny kod usługi DelayedMessageService	562
Etapy tworzenia usługi Odometer	570
Zdefiniowanie obiektu Binder	573
Klasa Service ma cztery kluczowe metody	575
Dodanie obiektu LocationListener do usługi	577
Rejestracja obiektu LocationListener	578
Kompletny kod usługi OdometerService	580
Aktualizacja pliku AndroidManifest.xml	582
Aktualizacja układu aktywności MainActivity	586
Tworzenie obiektu ServiceConnection	587
Powiązanie z usługą należy utworzyć podczas uruchamiania aktywności	588
Wyświetlanie przebytego dystansu	589
Kompletny kod aktywności MainActivity	590
Twój przybornik do Androida	595



Material Design

14

W materialistycznym świecie

W API poziomie 21 Google wprowadził Material Design.

Z tego rozdziału dowiesz się, czym **jest Material Design** i jak sprawić, by nasze aplikacje do niego pasowały. Zaczniemy od przedstawienia **widoków kart** (ang. *card views*), których możemy wielokrotnie używać w całej aplikacji, zapewniając jej **spójny wygląd i sposób obsługi**. Następnie przedstawimy widok `RecyclerView` — elastycznego przyjaciela widoków list. Przy okazji dowiesz się także, jak **tworzyć własne adaptery** i jak całkowicie zmienić wygląd widoku `RecyclerView`, używając **zaledwie dwóch wierszy kodu**.



Przedstawiamy Material Design	598
Struktura aplikacji dla restauracji Włoskie Co Nieco	600
Utworzenie widoku <code>CardView</code>	603
Kompletny kod pliku <code>card_captioned_image.xml</code>	604
Utworzenie prostego adaptera	606
Zdefiniowanie obiektu <code>ViewHolder</code> na potrzeby adaptera	607
Utworzenie obiektów <code>ViewHolder</code>	608
Każdy widok <code>CardView</code> wyświetla zdjęcie i podpis	609
Dodanie danych do widoków <code>CardView</code>	610
Kompletny kod pliku <code>CaptionedImagesAdapter.java</code>	611
Utworzenie widoku <code>RecyclerView</code>	612
Dodanie widoku <code>RecyclerView</code> do układu	613
Kod klasy <code>PizzaMaterialFragment</code>	614
Do rozmieszczania zawartości <code>RecyclerView</code> używa menedżera układu	615
Określanie menedżera układu	616
Kompletny kod klasy <code>PizzaMaterialFragment</code>	617
Zastosowanie fragmentu <code>PizzaMaterialFragment</code> w aktywności <code>MainActivity</code>	618
Co się stanie po uruchomieniu kodu?	619
Utworzenie aktywności <code>PizzaDetailActivity</code>	627
Co musi robić aktywność <code>PizzaDetailActivity</code> ?	628
Aktualizacja pliku <code>AndroidManifest.xml</code>	628
Kod pliku <code>PizzaDetailActivity.java</code>	629
Obsługa kliknięć w widoku <code>RecyclerView</code>	631
Dodanie interfejsu do adaptera	634
Implementacja interfejsu <code>Listener</code> we fragmencie <code>PizzaMaterialFragment</code>	636
Umieszczenie treści na samym początku	639
Kompletny kod pliku układu <code>fragment_top.xml</code>	644
Kompletny kod klasy <code>TopFragment</code>	645
Twój przybornik do Androida	647

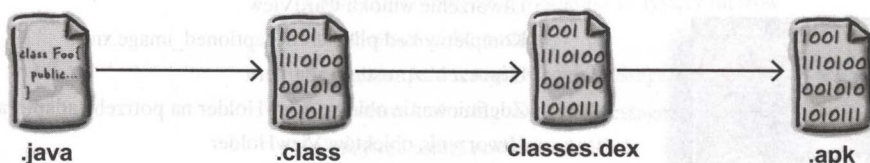
ART

Środowisko uruchomieniowe Androida

A

Aplikacje na Androida muszą działać na urządzeniach wyposażonych w słabe procesory i bardzo małą pamięć.

Aplikacje pisane w Javie mogą potrzebować sporo pamięci, a ponieważ działają wewnątrz własnej wirtualnej maszyny Javy (JVM), ich uruchomienie na komputerze o niewielkiej mocy obliczeniowej może trwać dosyć długo. Android rozwiązuje ten problem, nie używając JVM do uruchamiania aplikacji. Zamiast JVM używa całkowicie odmiennej wirtualnej maszyny, nazywanej **środowiskiem uruchomieniowym Androida** (ang. *Android Runtime*, w skrócie **ART**). W tym dodatku zobaczysz, jak to się dzieje, że ART umożliwia dobre działanie aplikacji napisanych w Javie nawet na małych komputerach o niewielkiej mocy obliczeniowej.



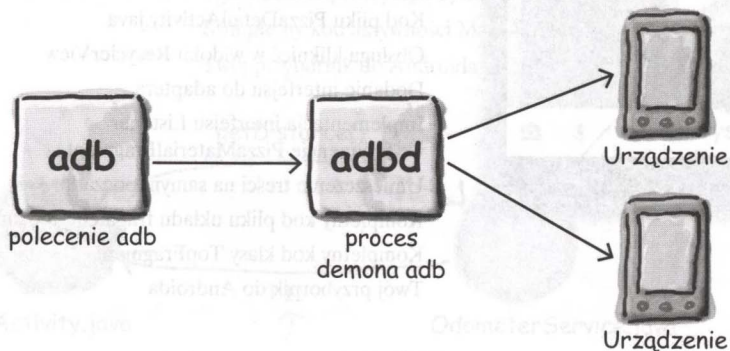
ADB

Android Debug Bridge

B

W tej książce skoncentrowaliśmy się na zaspokajaniu wszystkich potrzeb związanych z pisaniem aplikacji na Androida z wykorzystaniem IDE.

Zdarzają się jednak sytuacje, w których zastosowanie narzędzi obsługiwanych z poziomu wiersza poleceń jest po prostu przydatne. Mamy tu na myśli na przykład przypadki, gdy Android Studio nie jest w stanie zauważyć urządzenia z Androidem, choć my wiemy, że ono *istnieje*. W tym rozdziale przedstawimy **Android Debug Bridge** (w skrócie **ADB**) — obsługiwany z poziomu wiersza poleceń program narzędziowy, którego można używać do komunikacji z emulatorem lub z rzeczywistymi urządzeniami zaopatrzonymi w Androida.



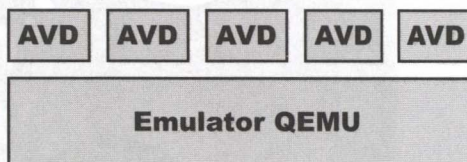
Emulator

Emulator Androida

Czy miałeś kiedyś wrażenie, że cały swój czas spędzasz, czekając na emulator?

Nie ma najmniejszych wątpliwości co do tego, że emulator Androida jest bardzo przydatny. Dzięki niemu możemy się przekonać, jak nasza aplikacja będzie działała na urządzeniach innych niż te, do których mamy fizyczny dostęp. Niekiedy jednak można odnieść wrażenie, że emulator działa... wolno. W tym dodatku wyjaśnimy, dlaczego tak się dzieje. Ale to nie wszystko, damy Ci bowiem także kilka wskazówek, jak przyspieszyć jego działanie.

Wszystkie wirtualne urządzenia z Androidem (AVD) są wykonywane przez emulator QEMU.



Pozostałości

Dziesięć najważniejszych zagadnień (których nie opisaliśmy)

Nawet po tym wszystkim, co opisaliśmy w tej książce, wciąż pozostaje wiele innych interesujących zagadnień.

Jest jeszcze kilka dodatkowych spraw, o których musisz się dowiedzieć. Czuliśmy się nie w porządku, gdybyśmy je pominęli, a jednocześnie chcieliśmy oddać w Twoje ręce książkę, którą dasz radę podnieść bez intensywnego treningu na siłowni. Dlatego zanim odłożysz tę książkę, przeczytaj kilka dodatkowych zagadnień opisanych w tym dodatku.

Bateria już jest prawie rozładowana... jeśli to kogoś interesuje.



Android

1. Rozpowszechnianie aplikacji	664
2. Dostawy treści	665
3. Klasa WebView	666
4. Animacje	667
5. Mapy	668
5. Mapy (ciąg dalszy)	669
6. Klasa CursorLoader	669
7. Odbiorcy komunikatów	670
8. Widżety aplikacji	671
9. Grafika 9-patch	672
10. Testowanie	673

Skorowidz

674