



SMASHING
MAGAZINE



Juhani Lehtimäki

**PODRĘCZNIK
DLA PROJEKTANTÓW**

Android™ UI

SMASHING MAGAZINE

Klucz do sukcesu Twojej aplikacji!

Tytuł oryginału: Smashing Android UI: Responsive Android UI and Design Patterns for Phones and Tablets

Tłumaczenie: Mikołaj Szczepaniak

ISBN: 978-83-246-6859-5

This edition first published 2013

© 2013 John Wiley & Sons, Inc.

Translation copyright © 2014 by Helion S.A.

All Rights Reserved. Authorized translation from the English language edition published by John Wiley & Sons Limited. Responsibility for the accuracy of the translation rests solely with Helion S.A. and is not the responsibility of John Wiley & Sons Limited.

Wiley and the Wiley logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/ or its affiliates in the United States and/or other countries, and may not be used without written permission. All trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in the book.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

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.

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

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)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:

<ftp://ftp.helion.pl/przyklady/andrui.zip>

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/andrui>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

Spis treści

O autorze	9
Podziękowania autora	11
Wprowadzenie	13
CZĘŚĆ I WPROWADZENIE DO ARCHITEKTURY SYSTEMU ANDROID	17
Rozdział 1. Użyteczność i projekt interfejsu użytkownika — wprowadzenie	19
Technologia kontra projekt interfejsu	20
Zrozumieć model mentalny	21
Projektowanie dla użytkowników	26
Żadna aplikacja nie może robić wszystkiego — wybierz właściwe obszary działania	29
To Ty jesteś ekspertem; użytkownicy nie są projektantami	30
Poznaj swoich użytkowników — projektuj dla prawdziwych ludzi	30
Podsumowanie	33
Rozdział 2. Wstrzymaj się z kodowaniem	35
Budowa prototypów	36
Narzędzia projektowe	38
Testowanie przez użytkowników	42
Podsumowanie	46
Rozdział 3. Specyfika rozwiązań mobilnych i urządzeń z ekranami dotykowymi	47
Projektowanie dla urządzeń mobilnych	48
Projektowanie pod kątem interfejsów dotykowych	55
Podsumowanie	59
Rozdział 4. Wprowadzenie do platformy Android	61
Wyzwania stwarzane przez platformę Android	62
Wersje Androida	70
Dystrybucja aplikacji dla Androida	73
Co oznacza open source?	75
Podsumowanie	77

CZĘŚĆ II	FUNKCJE I KOMPONENTY INTERFEJSU UŻYTKOWNIKA PLATFORMY ANDROID	79
Rozdział 5.	Struktura aplikacji Androida i wskazówki dostępne w internecie	81
	Ogólna struktura aplikacji Androida	82
	Ogólna struktura projektu Androida	86
	Oficjalne wskazówki dla Androida	88
	Podsumowanie	89
Rozdział 6.	Intencje Androida	91
	Intencje umożliwiają aplikacjom wzajemną współpracę	92
	Korzystanie z serwisów społecznościowych i udostępnianie zasobów	93
	Praca z przeglądarkami	95
	Jak działają intencje Androida?	97
	Tworzenie własnych akcji	103
	Intencje są wszędzie	104
	Intencje kontra interfejsy API aplikacji	104
	Podsumowanie	105
Rozdział 7.	Struktura nawigacji w aplikacjach Androida	107
	Komponenty nawigacji w systemie Android, aktywności i intencje	108
	Kontrolki nawigacyjne Androida	109
	Podsumowanie	121
Rozdział 8.	Widżety aplikacji na ekranie domowym	123
	Zastosowania widżetów aplikacji ekranu domowego	124
	Aktualizowanie danych widżetu aplikacji	128
	Układ i funkcje widżetu aplikacji	130
	Implementacja widżetu aplikacji	135
	Podsumowanie	137
Rozdział 9.	Powiadamianie i informowanie użytkowników	139
	Metody powiadamiania użytkownika w systemie Android	140
	Kiedy powiadamiać użytkowników	143
	Kiedy nie powiadamiać użytkowników	148
	Unikanie wyskakujących okien	149
	Optymalne wykorzystywanie powiadomień na pasku statusu	150
	Podsumowanie	157
Rozdział 10.	Projektowanie obsługi przycisków sprzętowych, metod wprowadzania danych i czujników	159
	Projektowanie obsługi ekranu dotykowego	160
	Projektowanie obsługi przycisków sprzętowych	161
	Projektowanie obsługi klawiatury ekranowej	162
	Projektowanie obsługi klawiatur sprzętowych	170
	Projektowanie obsługi krzyżyków i gładzików	170
	Projektowanie obsługi rysika	171

Projektowanie głosowego sterowania aplikacją	172
Projektowanie obsługi zewnętrznych klawiatur, myszy i touchpadów	172
Projektowanie obsługi dołączonych inteligentnych akcesoriów	174
Projektowanie obsługi czujników	174
Projektowanie obsługi dodatkowego ekranu	174
Podsumowanie	175
Rozdział 11. Projektowanie komponentów interfejsu użytkownika platformy	177
Stosowanie widgetów interfejsu użytkownika	178
Modyfikowanie widgetów interfejsu użytkownika	189
Modyfikowanie typografii	191
Stosowanie ikon	197
Stosowanie animacji i efektów przejść	203
Podsumowanie	211
CZĘŚĆ III SKALOWALNY PROJEKT ANDROIDA	213
Rozdział 12. Zarządzanie zasobami Androida	215
Stosowanie zasobów Androida	216
Projektowanie zasobów pod kątem gęstości pikseli	218
Projektowanie pod kątem różnych wymiarów ekranu	223
Projektowanie interfejsu pod kątem różnych języków i regionów	225
Projektowanie obsługi kontrolki urządzenia	226
Projektowanie interfejsu dla różnych wersji platformy	226
Projektowanie interfejsu dla różnych trybów pracy urządzeń	227
Podsumowanie	227
Rozdział 13. Układy aplikacji Androida	229
Strategia układu systemu Android	230
Układy definiowane w plikach XML i w kodzie	232
Menedżery układów	232
Definiowanie wielkości układu	243
Przewijanie	244
Oś Z, porządek komponentów w ramach układu	245
Dopełnienia i marginesy	245
Importowanie i scalanie plików układu	246
Układy niestandardowe	247
Narzędzia do budowy interfejsu użytkownika dla systemu Android	247
Diagnostowanie układów	248
Podsumowanie	249
Rozdział 14. Skalowalna grafika	251
Format 9-patch	252
Obiekty drawable definiowane w plikach XML	257
Rysowanie z poziomu kodu aplikacji	265
Podsumowanie	269

Rozdział 15.	Skalowalność to nie wszystko — samodostosowujący się projekt	271
	Platforma Android to nie tylko telefony	272
	Samodostosowujący się projekt	274
	Typowe sposoby tworzenia samodostosowujących się interfejsów użytkownika	285
	Podsumowanie	288
Rozdział 16.	Implementowanie samodostosowujących się interfejsów użytkownika	289
	Wprowadzenie do fragmentów	290
	Architektura fragmentów i aktywności	292
	Migracja istniejących aplikacji	294
	Analiza przykładowej aplikacji	296
	Podsumowanie	309
CZĘŚĆ IV WZORCE PROJEKTOWE INTERFEJSU UŻYTKOWNIKA W SYSTEMIE ANDROID		311
Rozdział 17.	Wprowadzenie do wzorców projektowych interfejsu użytkownika	313
	Wzorce projektowe interfejsu użytkownika	314
	Zalety stosowania wzorców projektowych interfejsu użytkownika	315
	Wzorce projektowe we wskazówkach projektowych platformy Android	316
	Wzorce projektowe interfejsu użytkownika prezentowane w tej książce	317
	Podsumowanie	319
Rozdział 18.	Wzorce projektowe akcji użytkownika	321
	Stosowanie wzorca Action Bar	322
	Stosowanie wzorca projektowego Quick Actions	331
	Stosowanie wzorca projektowego szuflady akcji	337
	Stosowanie wzorca projektowego Pull-to-Refresh	339
	Stosowanie gestu Swipe-to-Dismiss	343
	Podsumowanie	345
Rozdział 19.	Wzorce projektowe nawigacji i układu	347
	Stosowanie wzorca projektowego Stacked Galleries	348
	Stosowanie kokpitu	350
	Stosowanie przestrzeni roboczych	354
	Stosowanie widoku dzielonego	358
	Stosowanie wzorca projektowego Expand-in-Context	360
	Stosowanie nawigacji bocznej	363
	Podsumowanie	366
Rozdział 20.	Wzorce projektowe danych	367
	Stosowanie dynamicznych list	368
	Stosowanie wzorca projektowego Image Placeholder	370
	Stosowanie wzorca projektowego Non-forced Login	372
	Stosowanie wzorca projektowego Drag-to-Reorder Handle	375
	Podsumowanie	377

Rozdział 21. Antywzorce projektowe interfejsu użytkownika	379
Unikaj ekranu ładowania	380
Unikaj ekranu poradnika	381
Unikaj stosowania okna potwierdzenia	383
Unikaj przycisku Cofnij widocznego na ekranie	384
Unikaj stosowania przycisku menu	385
Unikaj ukrywania paska statusu	386
Unikaj wyświetlania szybkich akcji gestem przewijania	387
Unikaj stosowania rozwiązań projektowych spoza Androida	388
Podsumowanie	389
Skorowidz	390



6

INTENCJE
ANDROIDA

SYSTEM INTENCJI ANDROIDA jest prawdopodobnie najważniejszym mechanizmem oferowanym przez tę platformę. Intencje umożliwiają wewnętrzne i zewnętrzne wiązanie aplikacji. System intencji umożliwia programistom wywoływanie zarówno funkcji platformy Android, jak i funkcji wszystkich pozostałych zainstalowanych aplikacji. Ten sam system umożliwia aplikacjom udostępnianie funkcji innym aplikacjom.

W tym rozdziale wyjaśnię, czym są intencje Androida i gdzie są stosowane.

Celem tego rozdziału jest szczegółowe opisanie wpływu mechanizmu intencji na sposób projektowania interfejsu użytkownika aplikacji budowanych dla systemu Android. Rozdział zawiera co prawda kilka przykładów, ale nie obejmuje pełnej specyfikacji intencji ani wszystkich przypadków użycia. Czytelników zainteresowanych szczegółami technicznymi zachęcam do zapoznania się z dokumentacją Androida (patrz strona <http://developer.android.com/reference/android/content/Intent.html>).

INTENCJE UMOŻLIWIAJĄ APLIKACJOM WZAJEMNĄ WSPÓŁPRACĘ

Intencja przypomina trochę technicznie i formalnie zdefiniowany komunikat wysyłany do komponentu aplikacji. Taki komunikat może być wysyłany albo wewnątrz aplikacji, albo przekazywany pomiędzy różnymi aplikacjami bądź nawet pomiędzy systemem operacyjnym a aplikacjami. Aplikacja może na przykład wysłać do systemu operacyjnego komunikat o potrzebie wybrania jakiegoś numeru telefonu.

Największą zaletą mechanizmu intencji jest możliwość współpracy różnych aplikacji i wzajemnego udostępniania funkcji w łatwy i bezproblemowy sposób. Każda aplikacja może zażądać od platformy identyfikacji pozostałych aplikacji udostępniających określone funkcje, po czym użyć jednej z tych aplikacji lub umożliwić użytkownikowi wybór właściwej.

Jednym z najbardziej popularnych przypadków użycia intencji jest udostępnianie przez aplikację jakiegoś zasobu, na przykład obrazu. Niezależnie od tego, czy chodzi o edytor zdjęć, aplikację obsługującą aparat fotograficzny, aplikację graficzną, czy dowolny inny program, istnieje możliwość poinformowania systemu Android o istnieniu obrazu do udostępnienia. System operacyjny „wie”, które spośród pozostałych zainstalowanych aplikacji mogą obsługiwać to żądanie.

Przeanalizujmy teraz konkretny przykład. Poniższa sekwencja obrazów ilustruje ciąg czynności wykonywanych przez użytkownika, który za pomocą czterech różnych aplikacji robi zdjęcie i publikuje je w serwisie społecznościowym. Na rysunku 6.1 pokazano pierwszy krok, polegający na zrobieniu zdjęcia za pomocą aplikacji aparatu dostępnej w Androidzie. Następnie użytkownik otwiera to zdjęcie w aplikacji edytora (Skitch), gdzie umieszcza na zdjęciu prosty napis (patrz rysunek 6.2). I wreszcie użytkownik udostępnia gotowe (przerobione) zdjęcie na Twitterze (patrz rysunek 6.3).



Rysunek 6.1. Użytkownik robi zdjęcie za pomocą aplikacji aparatu fotograficznego Androida
Źródło: Android



Rysunek 6.2. Użytkownik umieszcza na zdjęciu tekst w aplikacji Skitch

Źródło: Skitch, copyright 2012 Evernote Corporation



Rysunek 6.3. Użytkownik udostępnia gotowe zdjęcie na Twitterze

Źródło: Twitter

W opisanj sekwencji na szczególną uwagę zasługuje brak konieczności zapisywania zdjęcia w galerii czy systemie plików — użytkownik może osiągnąć swój cel bez zapisywania obrazu. Plik obrazu jest automatycznie przenoszony przez system operacyjny Android bez wiedzy i ingerencji użytkownika. Ostatni krok (udostępnienie zdjęcia na Twitterze) jest wykonywany przez aplikację Twitter, zatem użytkownik nie musi się dodatkowo logować w celu sprawdzenia, czy jego wpis ze zdjęciem rzeczywiście jest gotowy do publikacji.

KORZYSTANIE Z SERWISÓW SPOŁECZNOŚCIOWYCH I UDOSTĘPNIANIE ZASOBÓW

Aplikacje mobilne i serwisy społecznościowe wprost doskonale do siebie pasują. Ludzie kochają swoje telefony i uwielbiają się dzielić w wybranych serwisach społecznościowych dosłownie wszystkim, co widzą, słyszą i jedzą. Na niektórych platformach integracja z Facebookiem i Twitterem jest tylko jednym ze sloganów i narzędzi wykorzystywanych przez marketingowców. Urządzenia z systemem Android oferują możliwość udostępniania

zasobów ze wszystkich aplikacji we wszystkich serwisach społecznościowych, w tym na Google+, LinkedIn, Orkut i oczywiście na Facebooku i Twitterze. Ewentualne nowe serwisy społecznościowe muszą tylko przygotować wersje swoich aplikacji dla platformy Android, stosując odpowiednie filtry intencji (to zagadnienie zostanie wyjaśnione w dalszej części tego rozdziału). Po zainstalowaniu aplikacji użytkownicy mogą udostępniać w nowej sieci społecznościowej zasoby bezpośrednio z galerii, wszystkich odpowiednio zaimplementowanych aplikacji do robienia zdjęć, aplikacji pocztówkowych, edytorów graficznych, edytorów tekstu. Z tego samego powodu użytkownicy nie są ściśle przywiązani do oficjalnych aplikacji klienckich. Na rysunku 6.4 pokazano proces udostępniania obrazu przez użytkownika. Warto zwrócić uwagę na aplikacje Seismic, Plume, Tweet Lanes, TweetDeck i Twitter — wszystkie są klientami Twittera, a użytkownik może swobodnie wybrać tę, która posłuży do udostępnienia zdjęcia w tym serwisie.



Rysunek 6.4. Użytkownik wybrał opcję udostępnienia pliku z aplikacji galerii systemu Android. System operacyjny Android prosi użytkownika o wskazanie aplikacji, która ma być użyta do zakończenia tej operacji

Źródło: Android

Podczas budowy aplikacji dla systemu Android nie musimy tracić czasu na integrację z serwisami społecznościowymi — wszystkim zajmie się platforma. Nie musimy wybierać aplikacji, które naszym zdaniem powinny być obsługiwane; nie musimy też implementować żadnych funkcji odpowiedzialnych za udostępnianie zasobów w konkretnych serwisach. Nasze zadania ograniczają się do implementacji intencji udostępniania zasobów zgodnie ze specyfikacją. Warto pamiętać, że na przykład lista aplikacji widoczna na rysunku 6.4 jest generowana automatycznie przez system operacyjny, zatem nawet tego elementu nie

musimy samodzielnie implementować. Wszystkim zajmuje się system operacyjny. Warto też podkreślić, że lista widoczna na rysunku 6.4 obejmuje tylko te aplikacje, które oferują możliwość udostępniania obrazów. Użytkownik nigdy nie otrzymuje do wyboru aplikacji, które nie będą „wiedziały”, co zrobić z wybranym typem danych.

Przeanalizujmy teraz rysunek 6.5. Pokazano na nim efekt wywołania podobnej intencji udostępniania, jednak tym razem przedmiotem publikacji jest nie obraz, tylko adres URL skopiowany z przeglądarki. System interpretacji intencji Androida automatycznie określa, które aplikacje należy udostępnić na wyświetlanej liście. Do mechanizmu interpretacji intencji wróć w dalszej części tego rozdziału.



Rysunek 6.5. Użytkownik wybrał opcję udostępnienia adresu URL. System operacyjny Android nie wyświetla już aplikacji obsługujących obrazy, tylko aplikacje, które potrafią przetworzyć adres URL

Źródło: Android

PRACA Z PRZEGLĄDARKAMI

Przeglądarki należą do najważniejszych elementów smartfonów i tabletów. Są bodaj najczęściej używanymi aplikacjami na wszystkich urządzeniach z systemem Android. Okazuje się, że system intencji umożliwia wiązanie naszych aplikacji także z przeglądarkami. Przeglądarki dla Androida (przynajmniej te prawidłowo zaimplementowane) używają intencji do otwierania każdego linka dotkniętego przez użytkownika. Takie intencje

zwykle są używane przez samą przeglądarkę, jednak zdarza się, że zasoby wskazywane przez adres URL mogą być lepiej prezentowane przez inną aplikację. Aplikacja może zasygnalizować systemowi operacyjnemu możliwość obsługi adresów URL pasujących do pewnych wzorców, na przykład z określoną nazwą domeny. W momencie dotknięcia przez użytkownika linka pasującego do tego wzorca system operacyjny wyświetli listę z możliwością wyboru właściwej aplikacji. Na rysunku 6.6 pokazano przykład listy wyświetlonej po dotknięciu przez użytkownika zwykłego linka HTML wskazującego sklep Google Play. System operacyjny Android rozpoznaje ten link jako specjalny przypadek, który może być dodatkowo obsługiwany przez aplikację Google Play — w związku z tym system umożliwia użytkownikowi wybór aplikacji, która wyda mu się najwłaściwsza dla tego linka. Oprócz dwóch przeglądarek zainstalowanych przez samego użytkownika lista zawiera także aplikację Google Play jako jedną z możliwych opcji.



Rysunek 6.6. Użytkownik kliknął w przeglądarce Androida link wskazujący sklep Google Play. System operacyjny wykrył, że istnieje inna aplikacja, która może obsłużyć ten adres URL, i zaproponował użytkownikowi wybór kilku aplikacji, które mogą przetworzyć to żądanie

Źródło: Android

O sile opisanego mechanizmu decyduje między innymi brak wymagań dotyczących specjalnych konstrukcji składniowych po stronie serwisu internetowego. Strona internetowa zawiera standardowy link, który przeniósłby użytkownika na witrynę sklepu z aplikacjami Androida (gdyby ten użytkownik korzystał ze standardowej przeglądarki internetowej lub nie dysponował na swoim urządzeniu zainstalowaną aplikacją Google Play).

Wskazówka. Jeśli nasza aplikacja oferuje alternatywny sposób przeglądania treści dostępnej w internecie, koniecznie powinniśmy zadbać o subskrypcję adresów URL pasujących do odpowiedniego wzorca (na przykład domeny). Nie ma powodu, by nie skorzystać z tej możliwości. Tak zbudowana aplikacja będzie traktowana jako lepsza alternatywa dla przeglądarki podczas interakcji z wybranym rodzajem treści. Jeśli aplikacja nie jest taką alternatywą, powinniśmy albo przemyśleć swoją strategię, albo poprawić samą aplikację.

Warto pamiętać, że intencje związane z adresami URL obejmują pełne adresy, w tym wszystkie parametry. Oznacza to, że nasza aplikacja może bezpośrednio otwierać właściwą treść. W powyższym przykładzie wybór aplikacji Google Play spowoduje skierowanie użytkownika bezpośrednio do strony odpowiedniej aplikacji w sklepie Google Play. Podobnie, link do serwisu YouTube otwarty w aplikacji YouTube skieruje użytkownika bezpośrednio na stronę umożliwiającą odtworzenie odpowiedniego zapisu wideo.

JAK DZIAŁAJĄ INTENCJE ANDROIDA?

Czas zajrzeć do wnętrza systemu i sprawdzić, jak naprawdę działają intencje platformy Android. Ten podrozdział polecam także czytelnikom, których nie interesują szczegóły techniczne, ponieważ dobre rozumienie intencji może bardzo ułatwić ocenę, co jest, a co nie jest możliwe do osiągnięcia za pomocą tego mechanizmu. Warto też opanować kilka terminów związanych z intencjami. Przykłady prezentowane w tym rozdziale są bardzo proste, jednak czytelnicy niebędący programistami mogą je pominąć.

RODZAJE INTENCJI

Istnieją dwa rodzaje intencji: intencje aktywności (ang. *activity intents*) i intencje rozgłaszania (ang. *broadcast intents*).

- **Intencje aktywności.** Intencje aktywności zawsze cechują się jedną aplikacją nadawcy i jedną aplikacją obsługującą. Aplikacja obsługująca może mieć postać aktywności lub usługi. Intencje aktywności podzielono na dwie dodatkowe kategorie: intencje jawne i intencje niejawne.
 - **Intencje jawne.** Jeśli aplikacja została przygotowana z myślą o obsłudze danej intencji przez konkretną aktywność lub klasę usługi, warto wywołać intencję jawną. Taka intencja będzie obsługiwana bezpośrednio przez odpowiednią aktywność lub usługę. W ten sposób aplikacje zwykle obsługują komunikację wewnętrzną. Mimo że intencje jawne są bardzo ważnymi konstrukcjami, nie są zbyt interesujące w kontekście interfejsów użytkownika.
 - **Intencje niejawne.** Intencje niejawne są używane w sytuacji, gdy aplikacja wywołująca nie dysponuje z góry informacją, która aplikacja obsłuży to żądanie. Aplikacja wywołująca tworzy intencję opisującą rodzaj akcji, która ma być wykonana, po czym dołącza do tej intencji niezbędne dane i wysyła ją do systemu operacyjnego.

Mechanizm intencji niejawnych umożliwia tworzenie luźnych relacji pomiędzy aplikacjami wywołującymi a aplikacjami odpowiadającymi na te wywołania.

Interfejs pomiędzy tymi aplikacjami jest precyzyjnie zdefiniowany, ale żadna z nich nie dysponuje wiedzą o aplikacji po drugiej stronie. Luźne związki pomiędzy komponentami znacznie ułatwiają konserwację aplikacji, ponieważ zmiany w pozostałych komponentach lub aplikacjach nie utrudniają pracy innych aplikacji i komponentów (o ile nie naruszają wcześniejszego interfejsu). Niezależność od pozostałych aplikacji oznacza też, że aplikacje, o których być może nawet nie słyszeliśmy podczas prac nad własną aplikacją, mogą w przyszłości udostępniać jej przydatne funkcje.

Intencje niejawne są bardzo ciekawe i istotne z perspektywy projektanta interfejsu użytkownika. Zrozumienie ich działania jest absolutnie konieczne do budowania dobrych aplikacji dla Androida.

- **Intencje rozgłaszania.** Intencje rozgłaszania są (jak nietrudno się domyślić) wysyłane przez jedną aplikację, ale mogą być odbierane i obsługiwane przez wielu adresatów. Intencje aktywności zawsze są wysyłane przez jedną aplikację i obsługiwane przez jedną aplikację docelową. W niektórych przypadkach taka komunikacja jeden-jeden nie wystarczy. Niektóre zdarzenia, na przykład te dotyczące niskiego poziomu naładowania baterii, mogą zainteresować więcej aplikacji. W takich przypadkach konieczne jest zastosowanie techniki rozgłaszania. Mechanizm rozgłaszania wykorzystuje te same rozwiązania co intencje jawne, jednak tak wysyłane intencje nie są obsługiwane przez aktywności ani usługi, tylko przez odbiorców rozgłaszania.

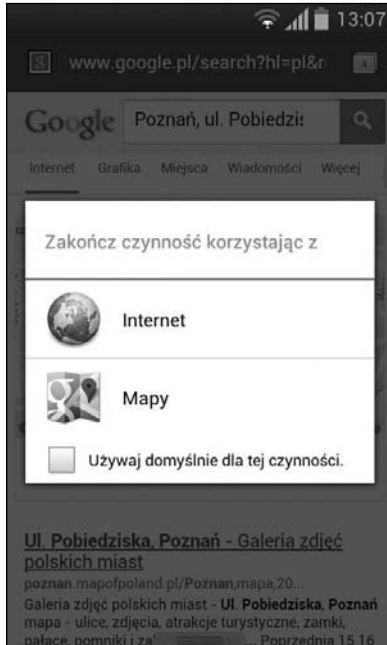
TECHNICZNY PRZYKŁAD WYSYŁANIA INTENCJI

W tym punkcie przeanalizujemy przykład kodu umożliwiającego użytkownikom aplikacji wykonywanie dodatkowych operacji na adresach pocztowych. Podobne rozwiązania są dość powszechne w takich domyślnych aplikacjach Androida jak Kalendarz czy Mapy Google. Dane adresowe dobrze ilustrują potencjał systemu intencji. Przypuśćmy, że nasza aplikacja dysponuje informacjami o adresie pocztowym. Warto w takim przypadku umożliwić użytkownikom przejście do widoku mapy lub nawet wybór opcji nawigowania do danego adresu. Warto pamiętać, że nie musimy wiedzieć, co użytkownicy zrobią z informacjami adresowymi. To do nich należy wybór aplikacji, której będą chcieli użyć.

Jedną z najważniejszych zalet stosowania intencji jest brak konieczności pisania jakiegokolwiek kodu map czy nawigacji na poziomie naszej aplikacji — możemy po prostu przekazać dane adresowe do przetworzenia przez inne aplikacje. Wysyłanie intencji jest bardzo proste. Warto przeanalizować poniższy przykład kodu. Kod w tej formie mógłby występować dosłownie wszędzie, jednak na potrzeby tego przykładu zostanie umieszczony w klasie aktywności. Intencja jest wywoływana w momencie dotknięcia przez użytkownika przycisku interfejsu.

```
sendIntent.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Uri geoUri = Uri.parse("geo:0,0?q="+ addressField.getText().toString());
        Intent mapCall = new Intent(Intent.ACTION_VIEW, geoUri);
        startActivity(mapCall);
    }
});
```


Po wysłaniu intencji na ekranie zostanie wyświetlone doskonale znane okno dialogowe wyboru aplikacji (patrz rysunek 6.7). Jeśli na przykład użytkownik wybierze Mapy Google, aplikacja otworzy widok mapy, od razu wskazując prawidłowy adres (patrz rysunek 6.8).



Rysunek 6.7. Okno dialogowe wyboru aplikacji po wysłaniu intencji przez aplikację

Źródło: Android



Rysunek 6.8. Mapy Google otwierają wysłany adres

Źródło: Google Inc.

FILTRY INTENCJI: AKCJE, DANE I KATEGORIE

Skąd system operacyjny „wie”, która aktywność, usługa lub który odbiorca rozgłaszania ma otrzymać daną intencję? Skąd pewność, że wysyłana przez naszą aplikację intencja będzie obsługiwana tylko przez aktywności wykonujące operacje, na których nam zależy? Przekazujemy sterowanie poza własną aplikację do jakiejś innej aplikacji. W tej kwestii musimy się zdać na system operacyjny, który zadba o kierowanie użytkowników do właściwych aplikacji.

Warto w tym kontekście zajrzeć do wnętrza systemu i zrozumieć, jak działa mechanizm Androida odpowiedzialny za interpretację intencji. W systemie Android zastosowano dwa główne komponenty. Dla każdej aktywności, usługi lub każdego odbiorcy rozgłaszania można zdefiniować zbiór powiązanych filtrów intencji (zbiór intencji możemy zdefiniować albo w pliku manifestu aplikacji, albo dynamicznie, w czasie wykonywania kodu). Intencja zawiera definicję akcji i pole danych, a często obejmuje także kategorie i pewne dane dodatkowe. Po otrzymaniu intencji system operacyjny porównuje akcję, dane i kategorie z filtrami intencji wszystkich aplikacji i wybiera tylko pasujące aplikacje.

Akcje i kategorie to po prostu nazwy. Nie ma w tym mechanizmie niczego skomplikowanego. Nieco trudniejsza jest obsługa właściwych danych i ew. danych dodatkowych. Dane są definiowane albo w formie identyfikatora URI, albo jako typ MIME. Identyfikator URI składa się z dwóch części oddzielonych dwukropkiem. Pierwsza część definiuje typ danych lub schemat. Druga część identyfikuje same dane. Na przykład identyfikator URI *tel:123456789* oznacza, że typem danych jest *tel*, natomiast dane mają postać *123456789*. Z perspektywy mechanizmu interpretacji intencji zasadnicze znaczenie ma typ danych.

Interfejsy API systemu Android definiują wiele standardowych akcji, kategorii i kluczy dodatkowych danych. Te standardowe definicje intencji są stosowane niemal we wszystkich elementach platformy Android. Niektóre są wywoływane przez sam system operacyjny, inne są używane przez aplikacje domyślne dostarczane wraz z tym systemem. Standardowe akcje obejmują wysyłanie (udostępnianie), wybieranie numeru, dzwonicie, wyświetlanie i wiele innych zadań.

W tabeli 6.1 opisano standardowe akcje aktywności, natomiast w tabeli 6.2 wymieniono standardowe akcje rozgłaszania. Kompletną listę akcji używanych w pakiecie Android SDK można znaleźć w dokumentacji intencji na stronie <http://developer.android.com/reference/android/content/Intent.html>.

Tabela 6.1. Standardowe akcje aktywności Androida

Nazwa akcji	Opis akcji
<code>ACTION_ANSWER</code>	Żąda obsłużenia przychodzącego połączenia telefonicznego.
<code>ACTION_ATTACH_DATA</code>	Określa, że pewna część danych powinna zostać dołączona w jakimś innym miejscu.
<code>ACTION_CALL</code>	Żąda połączenia telefonicznego z osobą wskazaną w danych.
<code>ACTION_CHOOSER</code>	Żąda wyświetlenia okna wyboru aktywności, w którym użytkownik będzie mógł wskazać dalsze działania.
<code>ACTION_DELETE</code>	Żąda usunięcia przekazanych danych z kontenera.
<code>ACTION_DIAL</code>	Żąda wybrania numeru telefonicznego wskazanego w danych.
<code>ACTION_EDIT</code>	Żąda bezpośredniego dostępu do edycji przekazanych danych.
<code>ACTION_FACTORY_TEST</code>	Główny punkt wejściowy na potrzeby testów fabrycznych.
<code>ACTION_GET_CONTENT</code>	Żąda wyświetlenia okna, w którym użytkownik będzie mógł wybrać konkretny rodzaj danych, oraz zwrócenia wybranego typu.
<code>ACTION_INSERT</code>	Żąda umieszczenia pustego elementu w danym kontenerze.
<code>ACTION_MAIN</code>	Żąda uruchomienia jako głównego punktu wejściowego (nie oczekuje żadnych danych).
<code>ACTION_PICK</code>	Żąda wybrania elementu z przekazanych danych i zwrócenia wybranego elementu.
<code>ACTION_PICK_ACTIVITY</code>	Żąda wybrania aktywności dla danej intencji i zwrócenia odpowiedniej klasy.

Nazwa akcji	Opis akcji
ACTION_RUN	Żąda uruchomienia przekazanych danych (cokolwiek to znaczy).
ACTION_SEARCH	Żąda wykonania operacji wyszukiwania.
ACTION_SEND	Żąda dostarczenia pewnych danych do kogoś innego.
ACTION_SENDTO	Żąda wysłania wiadomości do adresata wskazanego w danych.
ACTION_SYNC	Żąda synchronizacji danych.
ACTION_VIEW	Żąda wyświetlenia danych na ekranie urządzenia.

Tabela 6.2. Standardowe akcje rozgłaszania Androida

Nazwa akcji	Opis akcji
ACTION_BATTERY_CHANGED	Ta akcja jest regularnie rozgłaszana w celu przekazania stanu ładowania, poziomu naładowania i innych informacji na temat baterii urządzenia.
ACTION_BOOT_COMPLETED	Ta akcja jest rozgłaszana tylko raz, bezpośrednio po uruchomieniu systemu.
ACTION_PACKAGE_ADDED	Akcja informuje o zainstalowaniu na urządzeniu nowego pakietu aplikacji.
ACTION_PACKAGE_DATA_CLEARED	Akcja informuje o usunięciu przez użytkownika danych jakiegoś pakietu.
ACTION_PACKAGE_REMOVED	Akcja informuje o usunięciu z urządzenia jakiegoś istniejącego pakietu aplikacji.
ACTION_PACKAGE_RESTARTED	Akcja informuje o ponownym uruchomieniu (przez użytkownika) jakiegoś pakietu i o zabiciu wszystkich procesów tego pakietu.
ACTION_POWER_CONNECTED	Akcja informuje o podłączeniu urządzenia do zewnętrznego źródła zasilania.
ACTION_POWER_DISCONNECTED	Akcja informuje o odłączeniu urządzenia od zewnętrznego źródła zasilania.
ACTION_SHUTDOWN	Akcja informuje o wyłączeniu urządzenia.
ACTION_TIME_CHANGED	Akcja informuje o zmianie strefy czasowej.
ACTION_TIMEZONE_CHANGED	Akcja informuje o ustawieniu godziny.
ACTION_TIME_TICK	Akcja informuje o zmianie bieżącej godziny.
ACTION_UID_REMOVED	Akcja informuje o usunięciu z systemu identyfikatora użytkownika.

Oprócz typu danych i akcji system operacyjny uwzględnia kategorię i filtr intencji. W większości przypadków jedyną pasującą kategorią jest kategoria **domyślna**. Za każdym razem, gdy nasz kod wysyła intencję, system operacyjny automatycznie dodaje do niej kategorię domyślną. Właśnie dlatego zawsze należy dodawać tę kategorię do definiowanych filtrów intencji.

Kategorie intencji są istotne tylko wtedy, gdy chcemy zastąpić aktywności ekranu domowego lub stacji dokującej. Wyjątkiem od tej zasady jest kategoria programu startowego. Wszystkie aktywności z filtrem intencji obejmującym tę kategorię będą wyświetlane w oknie programu startowego aplikacji.

Intencje mogą oczywiście wysyłać więcej danych (nie tylko identyfikatory URI). Każda intencja może obejmować dodatkowe pola danych, które nie podlegają formalnej specyfikacji i nie są uwzględniane przez mechanizm interpretacji intencji. Dodatkowe pola są powiązane z różnymi akcjami. Aktywności obsługujące określone typy akcji oczekują dodatkowych danych z konkretnymi kluczami. Typowymi przykładami takich dodatkowych kluczy są wiadomości poczty elektronicznej, tytuły, teksty, tematy i dane strumieniowe (na przykład podczas udostępniania obrazów). Kompletną listę takich standardowych dodatków można znaleźć w dokumentacji platformy Android na stronie <http://developer.android.com/reference/android/content/Intent.html>.

TECHNICZNY PRZYKŁAD OTRZYMYWANIA INTENCJI

Z technicznego punktu widzenia odbieranie intencji nie jest bardziej złożone niż ich wysyłanie. W tym punkcie wykorzystamy ten sam przykład, tyle że zacierpnięty ze strony odbierającej intencje. Wyobraźmy sobie, że nasza aplikacja może zaoferować użytkownikom pewną ciekawą usługę w momencie, w którym chcą wyświetlić adres pocztowy. Usługa może na przykład wyświetlić instrukcje nawigacyjne dla rowerzystów lub pasażerów transportu publicznego bądź tekstowy opis miejsca, w którym znajduje się dany adres.

W procesie odbierania intencji potrzebujemy dwóch komponentów. Po pierwsze, musimy dodać aktywność do pliku manifestu. W elemencie `activity` musimy zdefiniować filtr intencji, aby system Android mógł określić, które rodzaje intencji mogą być obsługiwane przez daną aktywność. W poniższym przykładzie kodu widać możliwy sposób definiowania filtra intencji z myślą o obsłudze adresów URI ze schematem `geo`. Schemat `geo` identyfikatora URI jest formalną specyfikacją opisywania geolokalizacji.

```
<activity
  android:name=".intents.ReceiveIntentExampleActivity"
  android:label="Smashing Android UI" >
  <intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:scheme="geo" />
  </intent-filter>
</activity>
```



Zachęcam do zeskanowania tego kodu QR za pomocą telefonu z systemem Android — w ten sposób można otworzyć aplikację i sprawdzić działanie tego przykładu. Warunkiem działania tego rozwiązania jest oczywiście wcześniejsza instalacja aplikacji dołączonej do tej książki. Informacje na ten temat można znaleźć we „Wprowadzeniu”.

W kodzie aktywności możemy odczytać geolokalizację dołączoną do intencji. W poniższym przykładzie identyfikator URI geolokalizacji jest odczytywany z danych intencji i wyświetlany w niezmienionej formie. W prawdziwej aplikacji należałoby przetworzyć identyfikator URI, aby na jego podstawie zdecydować o dalszych działaniach.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setTitle("Receive Intent Example");
    setContentView(R.layout.receive_intent_example);
    TextView addressText = (TextView) findViewById(R.id.example_address_field);
    addressText.setText("" + getIntent().getData());
}
```

TWORZENIE WŁASNYCH AKCJI

Nie musimy się ograniczać do standardowych, predefiniowanych akcji. Nic nie stoi na przeszkodzie, abyśmy tworzyli własne. Być może nasza aplikacja oferuje usługę, która będzie przydatna dla pozostałych programistów; a może sami budujemy wiele aplikacji, ale chcemy uniknąć ścisłych związków pomiędzy nimi.

Nasze niestandardowe akcje to tak naprawdę nazwy dla zdefiniowanych akcji. Zaleca się poprzedzanie akcji nazwą pakietu, aby uniknąć ich mylenia z akcjami definiowanymi przez innych producentów oprogramowania i programistów.

Poniższy kod definiuje filtr intencji dla zdefiniowanej przez mnie niestandardowej akcji. Gdybym teraz opublikował nazwę tej akcji (`com.androiduipatterns.smashingandroidui.examples.EXAMPLE_ACTION`) na stronie internetowej mojej aplikacji, pozostali programiści mogliby jej użyć do zintegrowania swoich aplikacji z tą konkretną aktywnością.

```
<activity
    android:name=".intents.ReceiveCustomIntentExampleActivity"
    android:label="Smashing Android UI" >
    <intent-filter>
        <action android:name="com.androiduipatterns
            .smashingandroidui.examples.EXAMPLE_ACTION" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

INTENCJE SĄ WSZĘDZIE

W systemie Android niemal wszystko jest wywoływane przy użyciu intencji. Za każdym razem, gdy uruchamiamy aplikację z poziomu programu startowego, w rzeczywistości używamy intencji wywołanej przez ten program startowy lub ekran domowy. Co więcej, nawet ekran domowy uruchamiamy, wywołując intencję. Jeśli więc chcemy zastąpić domyślny ekran domowy innym ekranem, musimy doprowadzić do sytuacji, w której inna aplikacja będzie obsługiwała intencję ekranu domowego. Ekran domowy Androida i zasobnik aplikacji to także aplikacja, tyle że z odpowiednim filtrem intencji.

Nawet domyślna aplikacja telefonu jest uruchamiana za pośrednictwem intencji. Także wybór telefonu i nawiązanie połączenia odbywają się przy użyciu intencji. Obsługę tych intencji możemy zaimplementować też we własnych aplikacjach. Jeśli postanowimy napisać aplikację w miejsce domyślnego programu do dzwonienia, będziemy mogli to zrobić.

W systemie Android nie istnieje **pojęcie** przeglądarki Androida. Mówiąc precyzyjnie, istnieje aplikacja nazwana Android Browser, jednak jej istnienie nie wyklucza możliwości stosowania wielu innych przeglądarek na tej platformie. Oznacza to, że nie możemy z góry zakładać, że użytkownik naszej aplikacji używa tej czy innej przeglądarki. Każda aplikacja obsługująca adresy URL może zyskać status domyślnej przeglądarki użytkownika — wystarczy, że implementuje aktywność z odpowiednim filtrem intencji.

Ta elastyczność platformy Android stwarza mnóstwo możliwości także dla naszych aplikacji. Warto przy tym pamiętać o wyzwaniach — nie możemy zakładać, że użytkownik korzysta na przykład z domyślnej książki adresowej czy przeglądarki bądź domyślnego ekranu domowego lub programu do dzwonienia.

INTENCJE KONTRA INTERFEJSY API APLIKACJI

Intencje nie są jedynym sposobem udostępniania funkcji przez nasze aplikacje. Wiele serwisów społecznościowych i innych usług udostępnia interfejsy API, które umożliwiają integrowanie ich funkcji bezpośrednio z budowanymi aplikacjami. Mimo że w pewnych przypadkach takie rozwiązanie jest uzasadnione, warto dokładnie przeanalizować wady i zalety tego modelu. W wielu sytuacjach będziemy musieli sami zaimplementować mnóstwo funkcji, które w normalnych okolicznościach moglibyśmy czerpać z innych aplikacji za pomocą intencji. Co więcej, każda rozbudowa aplikacji i serwisu społecznościowego o nową funkcję będzie dezaktualizowała naszą aplikację do czasu uzupełnienia implementacji o obsługę tej funkcji. W przypadku zastosowania intencji nowe funkcje otrzymamy automatycznie przy okazji aktualizacji aplikacji klienckiej.

Inną poważną wadą ścisłej integracji jest konieczność implementacji w kodzie aplikacji mechanizmów uwierzytelniania użytkownika (jeśli wymaga tego stosowany interfejs API). W takim przypadku musimy uzyskać od użytkownika dane uwierzytelniające lub otworzyć wbudowany widok odpowiedzialny za uwierzytelnianie. Niezależnie od wybranego modelu użytkownik będzie musiał zaufać naszej aplikacji — albo wpisać swoje dane uwierzytelniające w oknie naszej aplikacji, albo autoryzować wykonywanie operacji przez naszą aplikację w jego imieniu. Kto z nas byłby skłonny udostępnić swoje dane dostępu do

Facebooka przypadkowej aplikacji? Czy dalibyśmy nieznannej aplikacji prawo publikowania wpisów na naszej tablicy na Facebooku?

PODSUMOWANIE

Mam nadzieję, że udało mi się dowieść potencjału i elastyczności mechanizmu intencji. Podczas projektowania aplikacji warto poświęcić chwilę na analizę możliwości sprawienia, że nasz produkt będzie integralną częścią urządzenia użytkownika. Czy nasza aplikacja dysponuje funkcjami, które mogłaby udostępniać pozostałym aplikacjom w formie intencji? Czy może wykorzystać istniejące funkcje udostępniane przez inne aplikacje, tak abyśmy nie musieli sami pisać odpowiedniego kodu?

Ten rozdział nie miał na celu prezentacji wszystkich technicznych aspektów intencji. Czytelnikom zainteresowanym szczegółowymi informacjami na ten temat polecam książkę Reto Meiera zatytułowaną *Professional Android 4 Application Development* (Wiley, 2012) oraz dokumentację dla programistów opublikowaną na stronach firmy Google (patrz adres <http://developer.android.com/reference/android/content/Intent.html>).

Informacje na temat intencji udostępnianych przez inne aplikacje można znaleźć na stronie internetowej Open Intents pod adresem www.openintents.org (szczególnie cenny jest utrzymywany i udostępniany w tym serwisie rejestr intencji).

SKOROWIDZ

1

12key, 226

A

Able Remote, 174

AccelerateDecelerateInterpolator, 209

AccelerateInterpolator, 209

Action Bar, 111, 162, 322

ActionBarSherlock, 87, 330, 336, 358

actionDone, 168

actionGo, 168

actionNext, 168

actionNone, 168

actionPrevious, 168

actionSearch, 168

actionSend, 168

actionUnspecified, 168

activity intents, 97

AdapterViewFlipper, 133

adjustPan, 163

adjustResize, 163

ADW Launcher, 69

akcje

kategorie, 330

aktualizacje starszych urządzeń, 72

aktualizowanie aplikacji, 295

aktywność, 83, 108

architektura, 292

cykl życia, 114

definiowanie tybu wprowadzania

danych, 163

fragmenty, 290

inicjowanie, 108

kontrola przepływu i układu, 293

przykładowa aplikacja, 298, 305

dodatkowa aktywność, 307

tryb adjustPan, 164

tryb adjustResize, 164

uruchamianie flagi intencji, 108

wyzwalanie, 85

zabijanie, 114

zakończenie, 115

Alديو, 351

Amazon Android App Store, 74

analiza przykładowej aplikacji, 296

AnalogClock, 133

Android

Action Bar, 24

biblioteki tworzone przez

społeczność, 76

cykl aktualizacji, 70

diagram struktury, 82

dokumentacja referencyjna, 88

domyślna czcionka, 192

domyślne komponenty tekstowe, 178

działanie menu systemu, 24

ekran stanu baterii, 50

ekrany domowe

widżety aplikacji, 123

element sterujący

krzyżyk, 171

elementy składowe aplikacji, 82

fragmentacja, 62, 77

framework

Android, 76

układy, 83

gotowe animacje, 204

Holo

aplikacja Tasks, 67

ikony startowe, 200

interfejs API, 66

definiowanie akcji, kategorii

i kluczy, 100

mechanizm animacji

właściwości, 206

zapisywanie stanu aktywności, 114

interpretacja

intencji, 99

stanów, 1911

jako open source, 75

jądro Linuxa, 75

klawiatura ekranowa, 162

kolory domyślne, 195

komponenty, 75

nawigacji, 108

platformy, 82

licencja open source, 63

łączenie z dodatkowymi

ekranami, 174

Market, 67

modyfikacja CyanogenMod, 69

na tabletach, 71

nieograniczoność instalowania, 73

obrazy 9-patch, 253

obsługa

gestów, 55

urządzeń zewnętrznych, 172

oficjalne wskazówki, 88

otwartość na modyfikacje, 62

pakiet Android SDK, 86

platforma nowych możliwości,

61

powiadomienia, 140

nadużywanie, 142

poziomy API, 70

praca nad bazą kodu, 76

predefiniowane kwalifikatory,

216

projektowanie aplikacji, 48

projekty bibliotek, 87

różnorodność

urządzeń, 62, 63

zastosowań, 62

skalowanie grafiki, 184

skórki OEM, 64

społeczność programistów, 76

standardowe interpolatory, 209

sterowanie urządzeniami

różnorodność sposobów, 159

stosowanie animacji przejść, 120

struktura, 82

projektu, 86

system intencji, 91

szablony

dla Pencil, 41

projektowe, 38

szybkie wyczerpywanie baterii,

49

tablety a telefony, 64

tryby kafelkowania, 261

udostępnianie zasobów aplikacji,

93

warunek zgodności, 66

wersje, 70

a producenci urządzeń, 71

statystyki popularności, 70

wygląd systemu w zależności od

producenta, 64

zakładki, 24, 355

Android ADT, 357

Android Asset Studio, 42

Android Browser, 104

Android Compatibility Program, 63

Android Design, 89

Android Design Preview, 42

Android Developers, 88

Android Development Tools, 247

Android SDK

emulatory, 285

kontekstowy pasek akcji, 336

obsługa wzorca projektowego

Action Bar, 330

piksele niezależne od gęstości,

222

Android-PullToRefresh, 341

animacje

czas odtwarzania, 208

dla widoków i ich treści, 205

efekt spowalniania, 208

interpolatory, 208

klatki kluczowe, 208

nadpisywanie, 204

nadużywanie, 203

obiektów, 207

pokłatkowe, 206

przejść, 120

stosowanie, 203

transformacji, 205

układu, 208

właściwości, 207

na starszych urządzeniach, 207

odtworzenie, 208

tworzenie, 207

zastępowania fragmentów, 292

animatory, 207

AnticipateOvershootInterpolator, 210
 antywzorzec, 318, 379
 ekran ładowania, 380
 ekran poradnika, 381
 okno potwierdzenia, 383
 przycisk Cofnij, 384
 przycisk Menu, 385
 rozwiązania projektowe spoza Androida, 388
 Swipe Overlay Quick Actions, 332
 szybkie akcje
 gest przewijania, 387
 ukrywanie paska statusu, 386
 APK, 73
 aplikacja
 a serwisy społecznościowe, 93
 aktualizowanie danych, 128
 alternatywne sposoby przeglądania treści, 97
 analiza zadań, 49
 autonomiczna, 54
 a internetowa, 54
 bezpieczeństwo, 74
 definiowanie person, 31
 dodawanie alternatywnych zasobów, 216
 dopracowanie projektu, 38
 ekran początkowy, 115
 funkcje, 28
 a cel użytkownika, 28
 hybrydowa, 54
 implementowanie żądanych funkcji, 30
 informująca o błędach, 147
 instalowanie
 bezpośrednie, 73
 czcionek, 192
 warunek, 73
 intencje, 92
 interfejsy API, 104
 internetowa, 54
 ograniczenia, 54
 intuicyjność, 22
 jakość, 19
 Javy, 40
 klienta poczty elektronicznej, 126
 kod
 rysowanie, 265
 logowanie
 błędy, 146
 nieobowiązkowe, 372
 ładowanie, 380
 łatwość użycia, 22
 migracja, 294
 modułowość, 294
 obsługa komunikacji wewnętrznej, 97
 odkrywanie problemów, 43
 oficjalna dokumentacja, 88
 plik manifestu, 136
 pobieranie, 74
 pomoc dla użytkownika, 382

powiadamy o zmianach w środowisku, 85
 pozycja w sklepie, 75
 priorytety powiadomień, 156
 problemy w użytkowaniu, 22
 projekt ekranu, 231
 projektowanie dla Androida, 48
 prototyp, 36
 przełączanie, 113
 przygotowanie projektu, 32
 samodoskonalący się projekt, 276
 statystyki rozkładu wersji, 72
 sterowanie głosowe, 172
 struktura, 36
 testowanie
 rzeczywiste dane, 44
 transfer danych, 52
 połączenia z siecią WiFi, 53
 statystyki, 52
 tworzenie
 wybór funkcji, 72
 wybór wersji, 72
 udostępnianie funkcji, 91
 umieszczanie w sklepach, 75
 uruchamianie, 114
 użyteczność, 19
 wersje zastępujące domyślną, 65
 widget, 124
 wielozadaniowość, 48
 wybór obszaru działania, 29
 sklepu, 74
 wywołująca, 97
 z perspektywy użytkownika, 21
 zachowanie spójności z platformą, 24
 zamieszczanie w sklepach, 74
 zapamiętywanie stanu, 48
 zastąpienie ekranu domowego, 67
 programu uruchamiania, 67
 zerwanie połączenia, 51
 zmiana sieci, 51
 zużycie energii, 49
 atrybut
 android:animateLayoutChanges, 208
 android:fillAfter, 206
 android:fillViewport, 245
 android:layout_gravity, 240
 android:layout_height, 244
 android:layout_width, 244
 android:orientation, 237
 android:shape, 258
 android:typeface, 193
 background, 190
 grawitacji, 240
 imeOptions, 168, 169
 nextFocusDown/Up/Left/Right, 171
 onEditorActionListener, 169
 parentActivityName, 113
 previewImage, 134

scaleType, 185
 SoftInputMode, 163
 textSize, 193
 weight, 237

B

back stack, 83
 Balsamiq, 40
 Beautiful Widgets, 124
 bezpieczeństwo aplikacji, 74
 biblioteka
 animacje właściwości, 207
 serwis GitHub, 76
 zastosowanie we wzorcach projektowych, 316
 bitmapy, 261
 kafelkowanie
 morror, 261
 repeat, 261
 obiekty drawable, 261
 blog twórców Androida, 88
 błędy, 146
 procesów synchronizacji, 148
 wymagające obsługi, 146
 zadania wykonywane w tle, 148
 związane z zadaniami pierwszoplanowymi, 146
 BounceInterpolator, 210
 Bouncer, 74
 broadcast intents, 97
 budowa
 aplikacji hybrydowych, 54
 prototypów, 36
 Button, 133

C

car, 227
 Catch notes, 373
 cele użytkownika, 26, 27
 a funkcje aplikacji, 28
 identyfikacja, 27
 lista, 28
 Chronometer, 133
 companion widget, 124
 Contextual Action Bar, 332
 Cupcake, 70
 cwac-endless, 370
 CyanogenMod, 77
 CycleInterpolator, 210
 czcionki, 192
 dodawanie własnych, 192
 instalowanie, 192
 niestandardowe, 192
 czujniki, 174
 położenia, 174
 ruchu, 174
 środowiskowe, 174

D

date, 167
 datetime, 167
 DecelerateInterpolator, 211

definiowanie
 elementu fragment, 291
 filtru intencji, 102
 gradientów, 260
 kolorów, 189
 minimalnej wielkości widgetu, 131
 obiektów drawable
 wielowastwowych, 262
 obiektu koloru, 260
 person, 30
 prostokąta, 258
 skalowalnej wersji obrazu, 254
 własnego stylu tekstu, 197
 density-independent pixels, 222
 design pattern, 313
 desk, 227
 diagnozowanie układów, 248
 diagram architektury informacji, 277
 Gmail, 277
 dip, 222
 długie naciśnięcie, 334
 dodawanie
 niestandardowych elementów graficznych do komponentu, 190
 własnych czcionek, 192
 dokumentacja
 Androida, 91
 dla programistów, 88
 referencyjna, 88
 dołączona aplikacja, 14
 kod źródłowy, 16
 zgodność, 16
 Donut, 70
 dopełnienia, 245, 259
 dostęp do grafiki platformy, 203
 dostępność, 195
 dostosowywanie
 komponentów
 automatyczne, 288
 w trybie jeden do jednego, 287
 ruchomych ekranów, 286
 wyglądu tekstu, 191
 dots per inch, 219
 dowód słuszności, 38
 dół rodzica, 233
 dp, 222
 dpad, 226
 D-pady, 170
 DPI, 219
 DragSortListView, 376
 Draw 9-patch, 256
 Droid, 192
 dynamiczna lista, 368
 dostosowywanie do dużego ekranu, 369
 odmiany, 369
 rozwiązywane problemy, 368
 skutki stosowania, 368
 dynamiczny charakter treści
 pomieszczenie na ekranie, 361

dystrybucja
 aplikacji, 73
 OEM, 64

E

Eclair, 70
 Eclipse, 40
 ekran
 bezdotykowy, 161
 Dodaj nowy element, 280
 domowy, 65
 alternatywa, 67, 69
 dodatkowe interfejsy API, 69
 siatka, 131
 widgety aplikacji, 86
 dotykowy
 elastyczny, 161
 pojemnościowy, 160
 przewijanie, 244
 przyszłość, 161
 rezystancyjny, 160
 rodzaje, 160
 rysik, 171
 dzielony
 odmiany, 359
 kokpitu, 350
 koloru i informacji o kolorze
 przykładowa aplikacja, 300
 komunikatu, 370
 listy elementów, 278
 ładowania, 380
 alternatywa, 381
 stosowanie, 381
 wyświetlanie, 380
 łączenie, 285
 na pierwszym planie, 286
 opcjonalna treść, 287
 początkowy, 350
 poleceń głosowych, 172
 poradnika, 381
 alternatywa, 382
 wady, 381
 ruchomy, 286
 szczegółów elementu, 279
 technika ruchomych ekranów na pierwszym planie, 286
 w kolumnach, 285
 wyboru koloru
 przykładowa aplikacja, 298
 elastyczność, 283
 element
 drawable, 179
 include, 246
 kształtu, 258
 layer-list, 262
 merge, 246
 potomny, 262
 RadioGroup, 181
 Evernote
 ekran poradnika, 382

F

fill_parent, 244
 FLAG_ACTIVITY_CLEAR_TOP, 108, 118
 FLAG_ACTIVITY_NEW_TASK, 108
 FLAG_ACTIVITY_SINGLE_TOP, 108
 foldery struktury projektu, 86
 Format 9-patch, 252
 formatowanie
 na podstawie kodu języka HTML, 196
 fragmentacja, 62
 fragmenty, 83, 277, 290
 a wersje platformy Android, 294
 architektura, 292
 dodawanie do układów, 291
 gwarancja elastyczności, 292
 implementowanie, 290
 izolowanie, 294
 metody cyklu życia, 290
 przykładowej aplikacji, 299
 stopy tylne, 292
 transakcje, 291, 292
 tworzenie, 290
 wywoływanie macierzystej aktywności, 293
 zastępowanie, 291
 animacje, 292
 FrameLayout, 132
 framework
 wieloplatformowy, 54
 zarządzania zasobami systemu Android, 215
 Froyo, 70
 funkcja
 Cofnij, 383
 opcjonalna i wymagana, 72

G

galeria, 189
 generalized densities, 220
 generowanie ikon, 42, 198
 gesty, 55
 długie naciśnięcie, 334
 dotykowe
 obsługiwane w systemie Android, 56
 ekran rezystancyjny, 160
 ekrany pojemnościowe, 160
 obsługa
 funkcja opcjonalna, 72
 przez widget, 130
 odkrywanie, 57
 rzucanie, 344, 354
 krytyczne spojrzenie, 357
 wady, 387
 szybkość reakcji aplikacji, 85
 wielodotkowe, 55
 wykrywanie, 58
 GetJar, 74

- gęstość
 - ekranu, 218
 - kwalifikatory, 220
 - w ustawieniach Androida, 219
 - nieskalowalne obrazy, 220
 - niska, 220
 - piksele niezależne, 221
 - pikseli, 221
 - foldery kwalifikatorów, 221
 - kategorie, 220
 - superwysoka, 220
 - średnia, 220
 - TV, 220
 - uogólniona, 220
 - Gigbeat, 22
 - Gingerbread, 70
 - gładzik, 170
 - Gmail, 130
 - przycisk Cofnij, 383
 - Go Launcher, 69
 - Google I/O 2012, 360
 - Google Now, 343
 - Google Play, 74
 - fragmenty, 283
 - profil programisty, 89
 - rozszerzanie kontekstowe, 361
 - wskazówki udostępniania aplikacji, 89
 - zamieszczanie aplikacji, 74
 - złośliwe oprogramowanie, 74, 75
 - Google Play Music Player, 126, 333
 - Google RSS Reader, 359
 - Google TV, 272
 - góra rodzica, 233
 - gradienty, 260
 - definiowanie
 - kolorów, 260
 - liniowy, 260
 - radialny, 252, 260
 - rodzaje, 260
 - sweep, 260
 - GridLayout, 132
 - GridView, 133
 - H**
 - hdpi, 220, 222
 - Hierarchy Viewer, 248
 - Holo, 66
 - ikony paska akcji, 201
 - Holo light
 - ikony paska akcji, 201
 - Honeycomb, 70, 71
 - animacje właściwości, 207
 - fragmenty, 294
 - kwalifikatory wielkości ekranu, 224
 - hosty widgetów, 123
 - I**
 - Ice Cream Sandwich, 70
 - IDE, 40
 - identyfikacja
 - elementów listy
 - ikony, 202
 - zasobów konkretnych urządzeń, 216
 - identyfikator URI, 100
 - schemat geo, 102
 - ikony, 197
 - a tło, 200
 - aplikacji, 323
 - automatyczne skalowanie
 - obrazu, 198
 - bezpośrednie referencje, 202
 - dostęp do grafiki, 203
 - generowanie, 198
 - gotowe, 332
 - grupy, 198
 - kopie w strukturze aplikacji, 202
 - menu, 199
 - na ekranie kokpitu, 350
 - na liście, 202
 - okien dialogowych, 202
 - paleta Google, 200
 - paska akcji, 200
 - paska stanu, 201
 - źródła, 201
 - platformy Android, 202
 - reprezentujące akcje, 198
 - startowe, 198
 - domyślnych aplikacji systemu Android, 200
 - typy, 198
 - zakładek, 201
 - źródła, 201
 - zestawy, 203
 - znaczenie, 198
- ImageButton, 133
 - ImageView, 133
 - implementacja
 - animacji poklatkowej, 206
 - fragmentów, 283, 290
 - gestów, 55
 - grafiki, 265
 - intencji udostępniania zasobów, 94
 - komponentów tekstowych
 - użycie miar skalowalnych, 212
 - krzyżyków i gładzików, 170
 - list
 - źródła, 188
 - powiadomień na pasku statusu, 154
 - referencyjna, 71
 - samodostosowujących się projektów, 277
 - widgetu
 - aplikacji, 135
 - logika i funkcja, 136
 - wskazówki, 126
 - wzorca
 - Action Bar, 330
 - Action Drawer, 339
 - Dashboard, 353
 - Drag-to-Reorder Handle, 376
 - Dynamic Lists, 369
 - Expand-in-context, 363
 - Image Placeholder, 372
 - Non-forced Login, 374
 - Pull-to-Refresh, 341
 - Quick Actions, 336
 - Side Navigation, 366
 - Split View, 360
 - Stacked Galleries, 350
 - Swipe-to-Dismiss, 345
 - Workspaces, 357
 - importowanie
 - plików układu, 246
 - instalowanie aplikacji, 73
 - integracja z serwisami społecznościowymi, 94
 - inteligentne akcesoria, 174
 - intencje, 85, 91
 - a interfejsy API aplikacji, 104
 - adres URL, 95, 97
 - aktywności, 97, 100
 - Androida
 - zasada działania, 97
 - definiowanie
 - kategoria domyślna, 101
 - zbioru, 99
 - filtry, 99
 - akcje, 99, 100
 - dane, 100
 - kategorie, 99, 102
 - flagi kontrolne, 108, 117
 - jawne, 85, 97
 - lista wyboru aplikacji, 96
 - mechanizm interpretacji, 95
 - nawigacja w systemie, 108
 - niejawne, 85, 97
 - nowe zadania, 116
 - oczekujące, 154
 - otrzymywanie, 102
 - otwieranie
 - ekranu domowego, 109
 - linków dotkniętych, 95
 - platformy Android, 62
 - powiadomienia, 108
 - przykład wykorzystania, 92
 - rozgłaszania, 97, 98, 101
 - w systemie Android, 104
 - wiązanie aplikacji
 - z przeglądarkami, 95
 - widgetów, 129
 - współpraca aplikacji, 92
 - wysyłanie, 98
 - wywoływanie, 98
 - zalety stosowania, 98
 - zintegrowanie powiadomień, 154
 - źródła, 105
 - Intent.FLAG_ACTIVITY_NO_ ANIMATION, 204
 - interfejs naturalny, 55
 - interfejs użytkownika, 19
 - antyzworce projektowe, 379
 - bezwzględne położenie elementów, 230
 - budowa przy użyciu fragmentów, 298
 - definiowanie na tematach domyślnych, 66
 - dodawanie fragmentów, 291
 - dotykowy, 55, 57, 160

- dotykowy,
 - nawigacja, 22
 - rozszerzanie o nowe gesty, 58
 - efektywność, 20
 - elastyczność, 292
 - głosowy, 172
 - gruntowna przebudowa, 295
 - klawiatura, 163
 - kolekcje widgetów, 83
 - komponenty, 85, 177
 - gęstość ekranu, 219
 - kontekst używania aplikacji, 48
 - logiczność, 22
 - łączenie obszarów stałych
 - i skalowalnych, 230
 - na różnych platformach, 55
 - obsługa i prezentowanie danych, 367
 - opracowanie, 19
 - powiadomienia, 140
 - projekt
 - wskazówki, 89
 - projektanci a programiści, 20
 - projektowanie, 20
 - Android SDK, 41
 - Balsamiq, 40
 - model mentalny, 22
 - narzędzia, 38, 42
 - OmniGraffle, 38
 - WireframeSketcher, 39
 - znaczenie dla użytkownika, 48
 - przyciski, 179
 - reagowanie na gesty, 58
 - różnorodność urządzeń, 273
 - samodostosowujący się
 - tworzenie, 285
 - schematy, 36, 297
 - sekcje, 283
 - skalowalność, 64
 - stosowane konwencje, 24
 - struktury większe niż ekran, 244
 - system Android
 - definiowanie, 230
 - uchwyt przeciągania, 375
 - widgety, 178
 - wizualne wskazówki, 22
 - wzorce projektowe, 314
 - zakładki, 355
 - automatyczne
 - dostosowywanie, 288
 - stosowanie ikon, 201
 - zamiennik obrazu, 371
 - interpolator, 208
- J**
- Jelly Bean, 70, 71
 - animacje przejść, 121
 - powiadomienia na pasku statusu, 155
- K**
- kafelkowanie, 261
 - liniowe, 261
 - radialne, 261
 - sweep, 261
 - tryby, 261
 - kanwa, 265
 - klasa
 - ActionMode, 336
 - Android.R.anim, 204
 - Canvas, 265
 - Context, 290
 - dziedziczka
 - po klasie View, 265
 - Fragment, 290
 - FragmentActivity, 294
 - FragmentManager, 291
 - FragmentTransaction, 291
 - Notification, 156
 - Notification.Builder, 154, 156
 - NotificationManager, 154
 - ObjectAnimator, 207
 - PendingIntent, 154
 - ResponsiveExampleActivity, 299
 - układu, 247
 - ViewGroup
 - dziedziczenie, 247
 - ViewPager, 358
 - klatki kluczowe, 208
 - klawiatura ekranowa, 163
 - element ScrollView, 163
 - metody wprowadzania danych, 163
 - niestandardowa, 170
 - numeryczna, 166
 - przycisk akcji, 167
 - rodzaje, 68
 - sposób wyświetlania, 163
 - typ danych wejściowych, 166
 - klawiatura sprzętowa, 170
 - kod
 - moduły fragmentów, 292
 - stosowanie wielokrotne, 246
 - kod QR
 - analiza aplikacji, 296
 - animacje poklatkowe, 206
 - animacje układu, 208
 - animacje właściwości, 207
 - dopełnienia i marginesy, 245
 - gradienty, 260
 - grawitacja układu, 241
 - importowanie plików układu, 247
 - instalacja skanera, 15
 - kafelkowanie, 262
 - kokpit, 353
 - kontekstowy pasek akcji, 332
 - kształty, 258
 - modyfikowanie typografii, 191
 - nawigacja w systemie Android, 109
 - obraccanie obiektu drawable, 264
 - obrazy 9-patch, 253
 - otrzymywanie intencji, 102
 - powiadomienia, 156
 - przycisk akcji i opcje IME, 167
 - rozszerzanie kontekstowe, 363
 - rysowanie kształtu, 269
 - transformacje, 205
 - tryb edycji pola tekstowego, 169
 - typ danych wejściowych, 166
 - układ liniowy, 239
 - układ siatki, 243
 - układ względny, 233
 - uruchamianie, 15
 - widgety interfejsu użytkownika, 178
 - wielowarstwowy obiekt drawable, 263

kokpit, 350

 - dostosowywanie do dużego ekranu, 352
 - krytyczne spojrzenie, 352
 - miejsce powiadomień, 351
 - odmiany, 352
 - skutki stosowania, 351

kolory, 195, 260

 - cieniowanie tekstu, 195
 - definiowanie, 189
 - zasobów, 189
 - domyślne, 195
 - konstrukcje składniowe kodu, 189
 - stosowanie plików kolorów, 190

komponenty, 82

 - definiowanie wielkości, 244
 - dostosowywanie
 - automatyczne, 288
 - jeden do jednego, 287
 - EditText, 178, 179
 - foldery, 86
 - fragmenty, 283
 - galerii, 189
 - graficzne
 - skalowanie, 251
 - ImageView, 184, 256
 - wartości atrybutu scaleType, 185
 - interfejsu użytkownika
 - automatyczne
 - dostosowywanie, 277
 - kolejność na osi Z, 245
 - konieczność ustawienia wymiarów, 243
 - kontrolki przybliżania, 185
 - listy rozwijanej, 337
 - modyfikowanie wyglądu i sposobu obsługi, 189
 - nadpisywanie oryginalnych funkcji, 192
 - odległości, 245
 - parametry stanu, 190
 - paska akcji, 323
 - pola wyboru, 180
 - porządek w ramach układu, 245
 - przełączników, 179
 - selektory, 190
 - tekstowe, 191
 - znaczniki języka HTML, 196
 - TextView, 178, 179
 - VideoView, 186
 - w układzie
 - liniowym, 237
 - ramkowym, 239

siatki, 242
 względny, 233
 włączników, 180
 wielokrotnego użytku, 37, 283
 wizualne prezentacje
 narzędzia Androida, 265
 wyboru, 181
 wysuwanej szuflady, 187
 względem komponentu
 macierzystego, 234
 z możliwością klikania, 188
 zastępowanie, 288
 komunikacja
 między aplikacjami i systemem,
 92
 między fragmentem
 a aktywnością, 293
 komunikaty
 aplikacji Evernote, 140
 bez możliwości interakcji, 141
 o błędach, 146
 o krytycznych zdarzeniach, 142
 o nowych wiadomościach poczty,
 141
 o zadaniach w tle, 143
 pasek statusu, 142
 serwisów społecznościowych,
 149
 tosty, 141
 zadania wykorzystujące
 połączenie sieciowe, 143
 zdarzenia, 145
 konstruktory
 tworzenie obiektu paint, 267
 kontener przewijania, 182, 244
 obszary skalowalne, 230
 kontrelement kontrolny, 239
 kontroler mediów, 186
 kontrolka
 przybliżania, 185
 nawigacyjna, 325
 widoku, 323
 konwersja
 wielkości komponentu, 223
 korzystanie
 z serwisów społecznościowych,
 93
 krój pisma, 193
 krzyżyk, 170
 definiowanie kolejności
 aktywowania, 171
 kształty, 257
 dodawanie dopełnienia, 259
 kwalifikatory
 dostępnej szerokości i wysokości,
 225
 języka i regionu, 225
 konfiguracyjne, 216
 łączenie, 217
 najmniejszej szerokości, 224
 orientacji ekranu, 225
 poziomy API, 217
 proporcji ekranu, 225
 stacje dokujące, 227
 sterowanie urządzeniem, 226

tryb pracy urządzenia, 227
 wersji systemu operacyjnego, 227
 wielkość ekranu, 224
 zasobów
 kategorie gęstości pikseli, 220
 kraj i region, 225

L

land, 225
 large, 224, 309
 layout, 83
 ldpi, 220, 222
 Lehtimäki Juhani, 9
 licencja
 Apache, 76
 GPL, 75
 LinearInterpolator, 211
 LinearLayout, 132
 Lint, 294
 lista, 187
 elementy, 188, 277
 opcji
 komponent wyboru, 181
 przewijane w poziomie, 189
 rozwijana, 181
 wydajność, 188
 zastępowanie widokiem siatki,
 288
 ListView, 133
 long, 225

Ł

łączenie powiadomień, 153

M

marginesy, 245
 match_parent, 244
 mdpi, 220, 222
 mechanizm
 animacji właściwości, 206
 automatycznego pobierania
 dodatkowych elementów, 368
 nasłuchiwania zdarzeń, 300
 podziału widoku na strony, 357,
 358
 przewijania, 244
 rozwłaszania, 98
 slektora, 189
 sterowania
 ekran dotykowy, 160
 menedżery
 układów, 232, 249, 277
 zasobów, 216, 277
 menu rozwijane, 333
 metoda
 addToBackStack(), 292
 commit, 291
 drawArc, 265
 drawBitmap, 265
 drawCircle, 265
 drawColor, 265
 drawLines, 265
 drawRect, 265

drawText, 265
 getActivity, 290
 getFragmentManager(), 291
 invalidate(), 266
 onActivityCreated(), 290
 onCreateView(), 290
 onDraw, 266
 onDraw(Canvas), 265
 OnScrollListener.onScroll, 370
 overridePendingTransition(), 204
 setColor, 306
 setCustomAnimations(), 292
 startActionMode, 337
 super, 265
 View.registerForContextMenu(),
 337

metody wprowadzania danych
 tryby, 163
 model mentalny, 21
 czytnika e-booków, 22
 kształtowanie, 22
 oczekiwania użytkowników, 24
 moduł rozszerzeń, 40
 modyfikowanie
 typografii, 191

N

nad, 234
 nakładka, 187
 niestandardowa, 333
 navexposed, 226
 navhidden, 226
 nawigacja, 107
 boczna, 314, 363
 dodatkowe funkcje, 364
 dostosowywanie do dużego
 ekranu, 364
 krytyczne spojrzenie, 365
 skutki stosowania, 364
 kontrolki, 109
 nie związana z aktywnościami,
 118
 obsługa krzyżyka, 170
 udoskonalenie mechanizmów,
 113
 widgety, 126
 New Quick Actions 3D, 337
 Nexus, 71
 nger, 226
 nieobowiązkowe logowanie, 372
 dostosowywanie do dużego
 ekranu, 374
 konto demonstracyjne, 374
 krytyczne spojrzenie, 374
 skutki stosowania, 374
 nieodwracalne operacje
 okno potwierdzenia, 383
 niestandardowy ROM, 76
 night, 227
 nodpi, 220, 221
 nokeys, 226
 nonav, 226
 normal, 168, 224
 NotificationCompat2, 157

notlong, 225
 notnight, 227
 notouch, 226
 number, 167

O

oating screen, 286
 obiekt
 Canvas, 265
 drawable
 definiowany w plikach XML, 257
 podzielony na warstwy, 259
 złożony, 262
 kafelkowanie, 261
 koloru, 260
 OnScrollListener, 370
 Paint, 266
 obrazy
 9-patch, 252
 generowane przez narzędzia graficzne, 256
 jako elementy potomne, 262
 narzędzie SDK, 256
 struktura, 253
 w kodzie aplikacji, 256
 właściwa interpretacja, 253
 skalowanie i obracanie, 252, 263
 obsługa
 marginesów, 133
 różnych gęstości pikseli, 198
 obszar
 łączenie rodzajów, 230
 skalowalny, 230
 stały, 230
 odbieranie intencji, 102
 oddalanie obrazu, 57
 odkrywanie gestów, 57
 odświeżanie, 340
 okna dialogowe, 142
 okno potwierdzenia, 383
 alternatywa, 383
 stosowanie, 383
 wady, 383
 określanie
 docelowej grupy użytkowników, 30
 OmniGraffle, 38
 opcjonalna treść, 287
 open source, 63, 75
 rozpowszechnianie
 oprogramowania, 63
 środowisko, 62
 umowy licencyjne, 63
 opisywanie geolokalizacji, 102
 orientacja ekranu, 225
 original equipment manufacturer, 64
 oś Z, 245
 otwarte interfejsy API, 66
 OvershootInterpolator, 211

P

pakiet
 aplikacji, 73
 obsługi przykładowej aplikacji, 308
 parametry stanu
 specyfikacja, 191
 źródła, 191
 pasek akcji, 162
 akcje kontekstowe, 322
 dodatkowe funkcje, 324
 dostępność, 327
 dostosowywanie do dużego ekranu, 326
 dzielenie, 328
 ikony, 200
 interpretacja układu, 322
 intuicyjność ikon akcji, 327
 komponenty, 323
 kontekstowy, 332
 alternatywa, 333
 dostosowywanie do dużego ekranu, 335
 Gmail, 332
 implementacja, 336
 tryb, 329
 źródła, 332
 logo firmy, 322
 marka producenta, 323
 podzielony, 328, 329
 samodostosowujące się akcje, 330
 skutki stosowania, 324
 tworzenie motywów, 331
 wybór akcji, 329
 wysokość, 322
 wzorzec projektowy, 322
 zajmowana przestrzeń, 327
 pasek postępu, 144, 182, 184
 oceny, 184
 poziomy, 184
 rodzaje prezentowanych procesów, 184
 wirujący okrąg, 184
 wyszukiwania, 184
 pasek stanu
 ikony, 201
 pasek statusu
 ikony powiadomień, 151
 komponenty powiadomień, 152
 ukrywanie
 antywzorzec, 386
 lepsze rozwiązanie, 386
 stosowanie, 386
 wady, 386
 pasek tytułów, 354, 356
 Pencil, 41
 persona, 30
 definiowanie, 31
 test użytkownika, 43
 phone, 167
 PhoneGap, 54
 piksele
 niezależne od gęstości, 221
 przykład konwersji, 222
 w kodzie aplikacji, 222
 wymiary komponentów, 244
 niezależnie od skali, 223
 skalujące, 255
 pixels per inch, 219
 platforma Android, 177
 komponenty, 177
 wykorzystanie przez różne urządzenia, 272
 Play Books, 22
 plik konfiguracyjny widgetu, 135
 plik manifestu widgetu, 136
 plik zasobu, 86
 Plume, 94
 pod, 234
 podgląd widgetu, 134
 pole tekstowe, 168, 179
 przycisk akcji, 167
 pole wyboru, 180
 daty, 182
 liczby, 182
 port, 225
 porządkowanie elementów, 239
 ręczne, 375
 powiadomienia
 anulowanie, 153
 automatycznie ukrywane, 141
 dla aktualnie realizowanych zadań, 153
 dotknięcie, 154
 flaga trwających zadań, 153
 ikony, 152, 201
 inwazyjność, 140
 korygowanie, 153
 łączenie, 153
 na czas, 153
 na pasku statusu, 142, 144
 elementy, 152
 implementacja, 154
 kryteria, 142
 optymalne wykorzystanie, 150
 treść, 151
 o błędach, 146
 kryteria, 146
 podczas logowania, 146
 utrata połączenia sieciowego, 150
 zadań w tle, 148
 o zdarzeniach, 145
 okna dialogowe, 142
 unikanie, 149
 osadzone, 140
 priorytety, 156
 definiowanie, 156
 prostota komunikatu, 146
 przechodzenie do aplikacji, 117
 różnorodność metod, 139
 techniki, 140
 tosty, 141
 mała inwazyjność, 148
 umiarkowanie, 149
 umożliwienie reakcji, 144
 wbudowane, 140, 145
 zalety, 148

- wersja Jelly Bean, 155
 - zasadność, 143, 148
 - zgodność wsteczna, 157
 - poziomy API, 70
 - wersje, nazwy i kody, 70
 - PPI, 219
 - prezentowanie
 - strumienia lub kategorii treści, 348
 - priorytety powiadomień, 156
 - Priming, 363
 - procesy zatwierdzania, 74
 - program zgodności, 63
 - ProgressBar, 133
 - projekt
 - a rozwój urządzeń mobilnych, 51
 - antywzorzec, 388
 - aplikacji, 296
 - a strony internetowej, 274
 - badanie opinii, 43
 - bibliotek, 87, 217
 - dostosowujący się, 64
 - ekranu aplikacji, 231
 - modyfikacji, 76
 - strony internetowej, 274
 - struktury, 86
 - folderów, 86
 - komponentów, 86
 - testowanie przez użytkowników, 42
 - wprowadzanie zmian, 295
 - wykorzystanie bibliotek, 87
 - zamienniki dla elementów systemu, 67
 - projektowanie
 - aktywności instalacyjnej, 130
 - animacje przejść, 121
 - aplikacji
 - obsługa tabletów, 63
 - architektura informacyjna, 32
 - budowa prototypów, 36
 - buforowanie i zapisywanie danych, 52
 - dla różnych urządzeń, 272
 - dla urządzeń mobilnych, 48
 - dla użytkowników, 26
 - głosowego sterowania aplikacją, 172
 - identyfikacja miejsca przebywania, 116
 - ikony startowej, 199
 - interfejsu
 - graficznego, 247
 - różne języki i regiony, 225
 - różne wersje platformy, 226
 - samodostosowujący się projekt, 276
 - tryby pracy urządzeń, 227
 - użytkownika, 20
 - koncentracja
 - na najważniejszych zagadnieniach, 29
 - na potrzebach, 27
 - koszty przesyłania danych, 52
 - nawigacji, 108
 - nowe koncepcje, 32
 - obsługi
 - czujników, 174
 - dotykowego ekranu, 174
 - ekranu dotykowego, 160
 - inteligentnych akcesoriów, 174
 - klawiatur sprzętowych, 170
 - klawiatury ekranowej, 162
 - kontrolki urządzenia, 226
 - krzyżyków i gładzików, 170
 - przycisków sprzętowych, 161
 - rysika, 171
 - zewnętrznych klawiatur, myszy, touchpadów, 172
 - oficjalne zalecenia projektowe, 88
 - określenie grupy odbiorców, 30
 - optymalizacja energii baterii, 49
 - pasje użytkowników, 53
 - pod kątem interfejsów
 - dotykowych, 55
 - powiadomień, 140
 - projekt wizualny, 32
 - projektu, 314
 - przycisku
 - Cofnij, 110, 118
 - Up, 113
 - skórki OEM, 64
 - struktura nawigacji, 107
 - widgetów, 125
 - obsługa gestów, 130
 - wielkość, 131
 - zalecenia, 126
 - zużycie baterii, 128
 - wielkość ekranu, 223
 - wykrywanie gestów, 58
 - wymagania sprzętowe, 49
 - zapisywanie koncepcji, 36
 - zasobów
 - gęstość pikseli, 218
 - prototyp, 36
 - a implementowanie, 36
 - dopracowanie, 38
 - papierowy, 36
 - test użytkownika, 45
 - przygotowanie
 - Android SDK, 41
 - szczegółowy, 37
 - przeciągnięcie, 56, 58
 - przeciągnij, aby odświeżyć, 339
 - dostosowywanie do dużego ekranu, 341
 - krytyczne spojrzenie, 341
 - skutki stosowania, 341
 - przegląd plus szczegółły, 358
 - przeglądarki
 - dla Androida, 95
 - przejścia
 - do innych aplikacji, 116
 - między aktywnościami, 120, 204
 - nadpisywanie animacji, 204
 - między ekranami, 354, 358
 - między widokami, 348
 - rezygnacja, 204
 - stosowanie, 203
 - przestrzenie robocze, 354
 - dostosowywanie do dużego ekranu, 356
 - krytyczne spojrzenie, 357
 - skutki stosowania, 356
 - przetwarzanie wielozadaniowe, 113
 - przewijanie, 244
 - widoku, 244
 - przycisk, 179
 - akcji, 167, 323
 - Dalej, 167
 - Gotowe, 167
 - opcje, 168
 - Cofnij, 110, 115, 204
 - a Up, 112
 - alternatywa, 110, 385
 - antywzorzec, 384
 - przejście do poprzedniego ekranu, 119
 - przerywanie operacji, 119
 - wady, 384
 - wskaźnik postępu, 119
 - graficzny, 179
 - Home, 109, 115
 - Menu, 161
 - alternatywa, 385
 - antywzorzec, 385
 - wady, 385
 - opcji, 181
 - przełącznika, 180
 - sprzętowy, 161
 - Up, 111, 325
 - reguły, 113
 - przypadek użycia, 26
 - przystosowanie aplikacji do większego ekranu, 296
 - publikowanie adresu URL, 95
 - obrazów, 92
 - puknięcie, 56
 - i przytrzymanie, 57
 - podwójne, 56
 - Pulse, 348
- ## Q
- qwerty, 226
- ## R
- reakcja na gesty, 58
 - RelativeLayout, 132
 - responsive design, 64, 271
 - Roboto, 192
 - rozdzaje intencji, 97
 - ROM, 69
 - rozciąganie, 56
 - rozwłaszanie intencji, 85
 - rozkład wersji systemu Android
 - statystyki, 70
 - rozszerzanie
 - kontekstowe, 360
 - dotatkowe funkcje, 362
 - dostosowywanie do dużego ekranu, 362
 - skutki stosowania, 361

rozszerzanie sekcji treści, 360

rozwiązania projektowe spoza Androida, 388

stosowanie, 389

wady, 388

rysyk, 171

sterowanie urządzeniem, 172

rysowanie kształtu

- przykład, 266
- na kanwie, 265
- z poziomu kodu aplikacji, 265

rzucane widoki, 355

rzucanie, 58

S

samodostosowujący się projekt, 271, 274

3D, 286

ogólny proces budowy, 277

przygotowywanie dla aplikacji Androida, 276

przykładowa aplikacja, 302

smartfony i tablety, 282

szacowanie stosunku kosztów do korzyści, 285

telefony, 277

wzorze projektowe, 317

zmiana układu komponentów, 275

scalanie plików układu, 246

ScrollView, 244

Seismic, 94

selektory, 190

- dla obiektów drawables, 264
- tło dla komponentów, 190

siatka ekranu domowego, 131

- wielkość widgetów, 132

side loading, 73

Side Navigation, 314

skalowalność

- wymiary minimalne, 275

skalowanie

- automatyczne, 220
- a odrębne zasoby, 220
- grafiki, 251
- niewystarczające, 272
- obrazów, 252, 263
- układy, 230
- zapobieganie, 221

skanowanie kodów QR

- instalacja skanera, 15

skórki OEM, 64

sliding drawer, 187

SlidingMenu, 366

small, 224

so input mode, 163

Songkick, 351

sp, 223

spinner, 181

społeczność programistów Androida, 76

sprzęt, 62

StackView, 133

state_checkable, 191

state_checked, 191

state_enabled, 191

state_focused, 191

state_pressed, 191

state_selected, 191

state_window_focused, 191

sterowanie głosowe, 172

stos

- galerii, 348
- dodatkowe funkcje, 348
- dostosowywanie do dużego ekranu, 348
- skutki stosowania, 348

tylny, 83

- a nawigacja boczna, 363
- modyfikowanie działania, 108
- transakcje fragmentów, 292

stosowanie animacji, 203

efektów przejść, 203

fragmentów na starszych urządzeniach, 294

ikon, 197

obrazów 9-patch, 255

- w kodzie, 256

zasobów Androida, 216

strona rodzica

- lewa, 233
- prawa, 233

struktura nawigacji, 107

statystyki ekranów, 221

styl tekstu, 193

- przykłady, 194

stylus, 226

swipe views, 355

SwipeToDismiss-NOA, 345

system operacyjny

- interpretacja intencji, 99
- mechanizm wyszukiwania zasobów, 217
- ograniczenia widgetów, 129
- piksele niezależne od gęstości, 222
- sposób wyświetlania powiadomień, 156
- środowisko wielozadaniowe, 114
- właściwa interpretacja obrazów 9-patch, 253
- wymuszenie obsługi stosu aktywności, 108

szczypanie, 56

szuflada akcji, 337, 338

- dostosowywanie do dużego ekranu, 339
- krytyczne spojrzenie, 339
- skutki stosowania, 338

szybkie akcje, 331

- długie naciśnięcie, 334
- dostosowywanie do dużego ekranu, 335
- gest przewijania alternatywny, 387

antywzorzec, 387

wady, 387

kontekstowy pasek akcji, 332

krytyczne spojrzenie, 335

menu rozwijane, 333, 337

niestandardowa nakładka, 333, 337

skutki stosowania, 334

T

tapping, 55

Tasks, 375

techniczna wykonalność projektu, 38

tekst

- cieniowanie, 196
- dostępność, 195
- kolor, 195
- krój pisma, 193
- przezroczystość, 195
- skalowalność, 193
- styl, 193
- definiowanie dla całej aplikacji, 197
- wielkość, 193
- domyślna, 194
- niezależna od gęstości, 223
- rekomendowana, 194

temat

- Holo, 66
- OEM, 64

test użytkownika, 43

- analiza wskazanych problemów, 44
- dobór uczestników, 43
- liczebność uczestników, 45
- neutralność organizatora, 44
- planowanie, 43
- prototyp papierowy, 45
- reakcja na wykryte problemy, 46
- scenariusze, 43
- testowania funkcji, 44
- znaczenie kontekstu korzystania, 45

testowanie aplikacji, 42

- prototypów, 41

text, 167

textAutoComplete, 167

textAutoCorrect, 167

textCapCharacters, 167

textEmailAddress, 167

textMultiLine, 167

textNoSuggestions, 167

textPassword, 167

textPersonName, 167

textShortMessage, 167

textUri, 167

TextView, 133

theming, 331

time, 167

Titanium Appcelerator, 54

title strip, 354

toast, 141

TouristEye, 334

trackball, 226
 transakcja, 291
 transformacje, 205
 a animacje właściwości, 207
 problemy, 205
 tryb
 akcji, 329
 kontekstowego paska akcji, 329
 wprowadzania danych
 klawiatura sprzętowa, 170
 tvdpi, 220
 Tweet Lanes, 94
 TweetDeck, 94
 Twitter, 368
 aplikacja, 94
 tworzenie
 animacji właściwości, 207
 fragmentów, 290
 grafiki
 łączenie obiektów w warstwę,
 262
 listy celów użytkownika, 28
 obiektu paint, 266
 samodostosowujących się
 interfejsów, 285
 treści niezależnej od gęstości
 pikseli, 221
 układów, 247
 własnej klasy, 192
 własnych akcji, 103
 zasobów graficznych
 sposoby konwersji, 222
 typ danych wejściowych, 166
 typografia, 191

U

UCD, *Patrz* user centered design
 uchwyt przeciągania
 dostosowywanie do dużego
 ekranu, 376
 odmiany, 376
 skutki stosowania, 375
 w celu ponownego
 uporządkowania, 375
 udostępnianie zasobów, 92, 93
 układy, 83
 definiowanie
 obszarów, 230
 w kodzie, 232
 w plikach XML, 232
 wielkości, 243
 diagnozowanie, 248
 dodawanie fragmentów, 291
 grawitacja, 240
 importowanie plików, 246
 liniowe, 237
 kierunek, 237
 menadżery, 232
 niestandardowe, 247
 przykładowa aplikacja, 302
 ramkowe, 239
 plik układu, 303
 siatki
 i tabel, 242
 kokpit, 353

systemu Android, 229
 w ramach układów, 232
 wielopanelowe, 359
 względne, 232
 punkty zaczeplenia, 233, 234
 ukrywanie akcji, 337
 UriImageViewHelper, 372
 uruchamianie
 kodów QR, 15
 przypadki użycia ikony startowej,
 115
 urządzenia mobilne
 gęstości ekranów, 219
 kontekst używania, 48
 modyfikacja domyślnych
 kolorów systemu, 64
 obsługa sieci bezprzewodowych,
 53
 ograniczenia, 48, 49
 połączenie z internetem, 51
 rodzaje ekranów, 221
 rozwój, 51
 specyfika, 47
 tablety, 63
 z systemem Android, 272
 zastosowanie widgetów, 128
 wersje aplikacji domyślnych, 65
 urządzenia Nexus, 71
 user centered design, 26
 usługi, 85
 usuwanie
 pojedynczych elementów, 343
 wersje Androida, 343
 użytkownik
 a powiadomienia, 142
 atrakcyjność widgetów, 137
 cele, 26
 grupy odbiorców, 30
 model mentalny, 21
 ocena animacji, 203
 ocena aplikacji, 26
 oczekiwania, 24
 odkrywanie gestów, 57
 pasja i zaangażowanie, 53
 potrzeby, 26
 przewidywanie skutków
 aplikacji, 21
 przycisk Up, 112
 przyzwyczajenia, 24
 reakcja na błędy, 146
 stosowanie jednego komponentu
 w aplikacjach, 324
 symulowanie zachowań aplikacji,
 21
 żądania funkcji, 30

V

view pager, 357
 ViewFlipper, 133
 ViewPagerIndicator, 358
 ViewStub, 133

W

walled garden, 73
 wersje kolorystyczne, 190
 wheel, 226
 widget
 a zarządzanie pamięcią, 128
 aktualizacje na żądanie, 129
 aktualizowanie danych, 128
 aplikacji, 86, 123, 124
 na ekranach domowych, 123
 Offi, 125
 bezpośrednie funkcje aplikacji,
 126
 definiowanie minimalnej
 wielkości, 131
 dodawanie przez użytkowników,
 130
 dostawca, 136
 godziny i daty, 182
 implementacja, 135
 interfejsu użytkownika, 85, 178
 modyfikowanie, 189
 jako element wspomagający
 nawigację, 126
 komponenty, 133
 marginesy, 133
 mediów, 184
 obsługa gestów i interakcji, 130
 ograniczenia, 126, 129
 plik konfiguracyjny, 135
 podgląd, 134
 przestrzeń na siatce ekranu, 132
 skalowalny, 133
 skróty
 przykłady, 127
 tablety, 128
 tekstowy, 178
 towarzyszący, 124
 układy, 135
 i funkcje, 130
 i komponenty, 132
 wielkość, 131
 zastosowanie marketingowe, 124
 automatyzowane aktualizacje,
 129
 zbiór szablonów, 133
 widok
 adaptera, 187
 dzielony, 358
 dostosowywanie do małego
 ekranu, 359
 skutki stosowania, 359
 kalendarza, 182
 podział na strony, 357
 wielkość
 ekranu, 223
 dostępna szerokość
 i wysokość, 225
 kategorie, 223
 kwalifikatory, 224
 najmniejsza szerokość, 224
 podział szczegółowy, 224
 proporcje, 225
 standardy rozdzielczości, 225

- wielkość
 uogólniona, 223
 tekstu, 193
 etykiety, 194
- wielozadaniowość środowiska
 mobilnego, 48
- WiFi, 53
- winamp, 338
- WireframeSketcher, 39
- włącznik, 180
- workspaces, 354
- wprowadzanie
 danych tekstowych, 179
- wrap_content, 244
- wskazówki
 dla akcji, 327
 dla wydawców aplikacji, 89
 projektowe, 89, 316
 wzorzec Action Bar, 324
- wskaźnik ładowania danych, 240
- wtyczka, 40
- wyбір rodzaju aplikacji, 54
- wyrównanie
 dołu, 235
 góry, 235
 lewej strony, 235
 linii bazowej, 235
- wyrzucić, aby usunąć, 343
- dotychczasowe funkcje, 344
- dostosowywanie do dużego
 ekranu, 344
- krytyczne spojrzenie, 345
- skutki stosowania, 344
- wysuwana szuflada, 187, 337
- wyśrodkowanie, 233
- w pionie, 233
- w poziomie, 233
- wywoływanie
 funkcji platformy i aplikacji, 91
- wyznaczanie
 minimalnej i maksymalnej
 wielkości, 283
- wzorzec projektowy, 313
- Action Bar, 322, 385
- dostosowywanie do dużego
 ekranu, 326
- odmiany, 328
- rozwiązywane problemy, 322
- techniczna implementacja, 330
- wady, 327
- wskazówki projektowe, 324
- Action Drawer, 337
- krytyczne spojrzenie, 339
- rozwiązywane problemy, 337
- techniczna implementacja, 339
- akcji użytkownika, 321
- Contextual Action Bar, 332
- danych, 367
- Dashboard, 350
- odmiany, 352
- rozwiązywane problemy, 350
- techniczna implementacja, 353
- dostępność bibliotek, 316
- Drag-to-Reorder Handle, 375
- rozwiązywane problemy, 375
- techniczna implementacja, 376
- Dynamic Lists, 368
- techniczna implementacja, 369
- Expand-in-context, 360
- rozwiązywane problemy, 361
- techniczna implementacja, 363
- Image Placeholder, 370, 381
- rozwiązywane problemy, 370
- techniczna implementacja, 372
- interfejsu użytkownika, 314
- kategorie, 318
- nawigacji i układu, 347
- nazewnictwo, 318
- Non-forced Login, 372
- rozwiązywane problemy, 372
- techniczna implementacja, 374
- Overview Besides Details, 358
- przykładowe
 aplikacje, 317
- kody, 318
- Pull-to-Refresh, 339, 348
- dostosowywanie do dużego
 ekranu, 341
- krytyczne spojrzenie, 341
- rozwiązywane problemy, 340
- skutki stosowania, 341
- techniczna implementacja, 341
- Quick Actions, 331
- rozwiązywane problemy, 331
- techniczna implementacja, 336
- Side Navigation, 363
- odmiany, 366
- rozwiązywane problemy, 363
- techniczna implementacja, 366
- solidne podstawy, 315
- Split View, 358
- rozwiązywane problemy, 358
- techniczna implementacja, 360
- spójność platformy, 316
- Stacked Galleries, 348
- rozwiązywane problemy, 348
- techniczna implementacja, 350
- stosowanie, 314
- Swipe-to-Dismiss, 343
- dotychczasowe funkcje, 344
- rozwiązywane problemy, 343
- skutki stosowania, 344
- techniczna implementacja, 345
- Workspaces, 354
- rozwiązywane problemy, 354
- techniczna implementacja, 357
- wskazówki projektowe
 źródła, 316
- zalety, 315
- instalacja aplikacji
 błędy, 148
- pierwszoplanowe
 błędy, 146
- przechodzenie do nowej
 aplikacji, 116
- realizowane w tle, 143
- sposoby powiadomień, 150
- ustawienie flagi dla
 powiadomienia, 153
- wykonywane w tle
 błędy, 148
- zakładki, 243, 354
- aktywności, 357
- zamiennik
 dla elementów systemu, 66
- klawiatury, 66
- obrazu, 370
- dostosowywanie do dużego
 ekranu, 371
- skutki stosowania, 371
- zamknięty ogród, 73
- zapobieganie skalowaniu, 221
- zarządzanie
 przestrzenią dzielącą
 komponenty, 245
- zasoby, 216
- algorytm wyboru, 217
- dla różnych gęstości ekranów,
 219
- dla różnych rodzajów środowisk,
 216
- graficzne, 221
- nadpisywanie, 217
- odpowiednie foldery zasobów,
 216
- procedura wyboru, 217
- projektowanie, 218
- zatrzymywanie działających
 procesów, 119
- zbliżanie obrazu
 gesty, 57
- zdarzenia, 145
- aktualnie wyświetlanego ekranu,
 145
- wielokrotne
 powiadomienia, 153
- znacznik czasowy
 powiadomienia, 153
- zestawy ikon firmy Google, 203
- zgodność wsteczna powiadomień,
 157
- zmiana
 ekranu domowego
 skutki, 67
- układu
 w zależności od urządzenia,
 275
- zużycie baterii
 operacje, 50
- zwężanie, 56
- zwijanie rozszerzonej treści, 362

X

XDA Developers, 76
 xhdpi, 220, 222
 xlarge, 224, 309

Z

z lewej strony, 234
 z prawej strony, 234
 zadania, 109



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**

NIENAGANNY INTERFEJS UŻYTKOWNIKA TO WARUNEK KONIECZNY DO ODNIESIENIA SUKCESU!

Smashing Magazine to jeden z najpopularniejszych serwisów poświęconych profesjonalnemu tworzeniu stron WWW i interfejsów użytkownika. Korzystają z niego przede wszystkim profesjonalni projektanci i deweloperzy. Ta książka to kompilacja wiedzy najwybitniejszych autorów i ekspertów z zespołu Smashing Magazine. Sięgnij po nią już teraz i uświadom sobie, że nawet najlepsza aplikacja nie sprzeda się, jeżeli nie będzie atrakcyjna wizualnie i łatwa w zastosowaniu. Zanim napiszesz pierwszą linię kodu, przygotuj prototyp i przetestuj go na potencjalnych użytkownikach.

Dzięki lekturze tej książki poznasz struktury aplikacji, mechanizm intencji oraz najlepsze techniki nawigowania wśród dostępnych opcji. Kolejne rozdziały poprowadzą Cię przez szczegóły projektowania komponentów interfejsu, stosowania ikon oraz używania efektów. Część trzecia kompendium jest poświęcona zarządzaniu zasobami Androida, skalowaniu oraz układom interfejsu. Na końcu będziesz miał szansę zapoznać się z najlepszymi wzorcami tworzenia aplikacji na tę platformę. Książka ta jest nieocenionym źródłem informacji na temat tworzenia przyjaznych i wydajnych aplikacji. Musisz ją przeczytać!

Dzięki tej książce:

- poznasz najlepsze wzorce projektowe nawigacji i układu
- wykorzystasz w najlepszy sposób zasoby Androida
- zaprojektujesz funkcjonalny interfejs użytkownika
- odniesiesz sukces na platformie Android!

helion.pl
księgarnia
internetowa

Nr katalogowy: 14167



Księgarnia internetowa:

<http://helion.pl>



Zamówienia telefoniczne:

0 801 339900



0 601 339900

 **WILEY**



Helion

Sprawdź najnowsze promocje:

• <http://helion.pl/promocje>

Książki najchętniej czytane:

• <http://helion.pl/bestsellery>

Zamów informacje o nowościach

• <http://helion.pl/nowosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

sięgnij po **WIECEJ**



KOD KORZYŚCI

ISBN 978-83-246-6859-5



Cena 69,00 zł

Informatyka w najlepszym wydaniu