

NGÔ THỊ LAN (chủ biên)
NGÔ THỊ VINH
NGUYỄN THỊ THANH NHÀN
HÀ MẠNH HÙNG
NGUYỄN TUẤN ANH
NGUYỄN HẢI MINH

LẬP TRÌNH ỨNG DỤNG ANDROID



NHÀ XUẤT BẢN ĐHTN

Mục lục

Mục lục	i
Lời nói đầu	ix
1 Tổng quan về lập trình trên thiết bị di động	1
1.1 Sơ lược về hệ điều hành trên thiết bị di động	2
1.1.1 Hệ điều hành trên thiết bị di động	2
1.1.2 Lịch sử	3
1.2 Hệ điều hành Android	7
1.2.1 Lịch sử phát triển	8
1.2.2 Các tính năng	8
1.2.3 Các phiên bản	9
1.2.4 Ưu điểm	10
1.2.5 Nền tảng của hệ điều hành Android	13
1.2.6 Quá trình biên dịch và thực thi ứng dụng . .	16
1.2.7 Các thành phần chính của một ứng dụng An- droid	17
1.3 Môi trường phát triển ứng dụng Android	18
1.4 Bắt đầu với Android	20
1.4.1 Cài đặt Android studio	20
1.4.2 Tạo một project Android mới	26
1.4.3 Cấu trúc của một project	29
1.4.4 File manifest	32

1.4.5	File MainActivity.java	34
1.4.6	File strings.xml	35
1.4.7	Giao diện ứng dụng (Layout)	35
1.4.8	Sơ lược kiến trúc của một ứng dụng Android	36
1.4.9	Biên dịch và chạy thử ứng dụng Android . .	36
1.5	Các kiểu dữ liệu trong Android	38
2	Lập trình giao diện căn bản	45
2.1	Đối tượng View	45
2.2	Khai báo các đối tượng widget và layout trong file XML	47
2.3	Các đối tượng widget cơ bản	50
2.3.1	Đối tượng TextView	50
2.3.2	Đối tượng Button	52
2.3.3	Đối tượng EditText	57
2.3.4	Đối tượng CheckBox	60
2.3.5	Đối tượng Radio Button	62
2.3.6	Đối tượng Adapter	65
2.3.7	Đối tượng Spinner	66
2.3.8	Đối tượng AutoCompleteTextView	72
2.3.9	Đối tượng ListView	76
2.4	Các đối tượng Layout	83
2.4.1	Đối tượng FrameLayout	84
2.4.2	Đối tượng LinearLayout	85
2.4.3	Đối tượng RalativeLayout	88
2.4.4	Đối tượng TableLayout	90
2.5	Đối tượng menu	92
2.5.1	Các menu option	93
2.5.2	Các menu context	95

2.6	Đối tượng hộp thoại	100
2.7	Thông điệp cảnh báo dạng Toast	103
3	Lập trình với các thành phần cơ bản: Activity, Intent, Content Provider và Service	106
3.1	Chu trình của một ứng dụng Android	107
3.2	Đối tượng Intent và Intent Filter	112
3.2.1	Phân biệt Intent và Intent Filter	112
3.2.2	Gọi Activity sử dụng đối tượng Intent	116
3.2.3	Gọi Activity và truyền dữ liệu qua Intent	117
3.2.4	Nhận dữ liệu từ Activity qua Intent	117
3.3	Đối tượng quảng bá thông điệp Broadcast Reciever	121
3.4	Đối tượng Content Provider	126
3.5	Các dịch vụ (Services)	134
4	Lập trình mạng trong Android	141
4.1	Làm việc với HttpClient	142
4.2	Làm việc với UrlConnection	143
4.3	Xây dựng WebService đơn giản với PHP	143
4.4	Xử lý dữ liệu với Json trong Android	153
5	Lưu trữ dữ liệu trong Android	161
5.1	Lưu trữ dữ liệu với Shared preferences	161
5.1.1	Ghi dữ liệu vào Shared Preferences	162
5.1.2	Đọc dữ liệu từ Shared Preferences	163
5.1.3	Xoá dữ liệu SharedPreferences	163
5.2	Lưu trữ dữ liệu với hệ thống file trong Android	167
5.2.1	Làm việc với file trong bộ nhớ trong	168
5.2.2	Làm việc với file trong bộ nhớ ngoài	172
5.3	Lưu trữ dữ liệu với SQLite	173

6 Xuất bản và phân phối ứng dụng Android	189
6.1 Giới thiệu	189
6.2 Xuất file .apk lên Google Play	193
6.3 Quản lý phiên bản ứng dụng	197
Tài liệu tham khảo	199

Danh sách hình vẽ

1.1	Biểu tượng của một số hệ điều hành trên điện thoại di động.	2
1.2	Andy Rubin – người đã tạo ra hệ điều hành Android.	8
1.3	Biểu tượng của các phiên bản Android.	10
1.4	Ưu điểm của Android.	12
1.5	Kiến trúc của hệ điều hành Android.	14
1.6	Kiến trúc của hệ điều hành Android.	17
1.7	Tải công cụ phát triển Java.	22
1.8	Cài máy ảo chạy android trên Genymotion.	23
1.9	Xem xét giấy phép cài đặt của Android studio.	24
1.10	Lựa chọn nơi cài đặt Android studio.	25
1.11	Lựa chọn cài đặt android studio mới hay nâng cấp phiên bản android studio đã cài.	25
1.12	Màn hình tạo một dự án mới.	27
1.13	Chọn nền tảng của thiết bị chạy ứng dụng.	28
1.14	Chọn loại activity.	29
1.15	Đặt tên activity.	30
1.16	Giao diện Android studio của ứng dụng HelloWorld.	30
1.17	Cấu trúc project Android trong Android Studio.	31
1.18	Chạy ứng dụng Android với máy ảo Genyotion.	37
1.19	Giao diện ứng dụng HelloWorld.	38

2.1	Mô hình đối tượng View	46
2.2	Thí dụ về các đối tượng widget cơ bản	48
2.3	Thí dụ về đối tượng TextView	51
2.4	Thí dụ về đối tượng Button	56
2.5	Thí dụ về đối tượng EditText	59
2.6	Minh họa đối tượng CheckBox.	61
2.7	Thí dụ về đối tượng RadioButton.	64
2.8	Minh họa đối tượng Spinner	67
2.9	Kết quả thí dụ về Spinner	69
2.10	Thí dụ tổng hợp về đối tượng Checkbox, Radio Button và Spinner	70
2.11	Kết quả thí dụ 2-12.	73
2.12	Minh họa đối tượng AutoCompleteTextView.	74
2.13	Minh họa đối tượng ListView	76
2.14	Kết quả thí dụ 2-15	80
2.15	Kết quả thí dụ 2-16	83
2.16	Minh họa thuộc tính Padding của đối tượng widget .	86
2.17	Thí dụ đối tượng LinearLayout	87
2.18	Thí dụ đối tượng RalativeLayout	89
2.19	Thí dụ đối tượng TableLayout	91
2.20	Thí dụ về menu option	94
2.21	Minh họa về menu context	96
2.22	Kết quả thí dụ 2-21 về tạo menu context	101
2.23	Minh họa về hộp thoại cảnh báo	101
2.24	Kết quả thí dụ 2-22	102
2.25	Minh họa thông điệp Tost	103
2.26	Bài tập đối tượng CheckBox và Radio Button.	104
2.27	Bài tập đối tượng ListView	105
2.28	Bài tập đối tượng ListView	105

3.1	Thứ tự ưu tiên của các ứng dụng trên hệ điều hành Android.	108
3.2	Ngăn xếp các các Activities.	110
3.3	Sơ đồ chuyển trạng thái của một Activity.	111
3.4	. Hai loại Intent.	113
3.5	Kết quả thí dụ về Activity	122
3.6	Kết quả thí dụ 3-2	126
3.7	Mô hình chia sẻ dữ liệu qua ContentProvider	127
3.8	Thí dụ về Content Provider	129
3.9	Trình tự các trạng thái của Started Service và Bound Service	135
3.10	Giao diện lớp HelloActivity	137
4.1	Thí dụ đăng nhập hệ thống	146
4.2	Ứng dụng truy xuất file JSON	155
5.1	Giao diện ứng dụng sử dụng Shared Preferences	164
5.2	Xem file Shared Preferences trên thiết bị	167
5.3	Giao diện Thí dụ 5-2 (thao tác với File)	169
5.4	Giao diện Thí dụ 5-3.	176
6.1	Vòng đời phát triển của một ứng dụng Android.	190
6.2	Một số kho ứng dụng dành cho Android.	191
6.3	Đăng ký tài khoản developer của Google Store.	193
6.4	Nhập thông tin thanh toán phí đăng ký tài khoản.	194
6.5	Tạo keystore.	196
6.6	Tạo file apk trong android.	196

Danh sách bảng

1.1	Lịch sử các hệ điều hành trên thiết bị di động.	3
1.2	Lịch sử các phiên bản Android	11
3.1	Một số sự kiện hệ thống trong Android	123
3.2	Các thành phần của chuỗi URI	127
4.1	Ký hiệu các thành phần trong file JSON	154
4.2	Các phương thức của lớp JSONObject	155
5.1	Các phương thức mở một CSDL SQLite	174
5.2	Các phương thức của lớp Cursor	175

Lời nói đầu

Ngày nay, điện thoại thông minh và máy tính bảng đang rất phổ biến, số người sử dụng thiết bị cầm tay đang tăng lên với tốc độ nhanh chóng, nhanh hơn so với bất kỳ công nghệ nào khác trong lịch sử thế giới. Lượng người sử dụng di động tăng lên chóng mặt do tính di động của nó và một phần do sự bùng nổ của các ứng dụng cho điện thoại di động trên kho ứng dụng. Khó có thể nói chính xác về những gì xảy ra trong tương lai nhưng với xu thế hiện tại, ta có yên tâm mà nói rằng nếu ai đó muốn làm việc trong lĩnh vực công nghệ thông tin trong 5-10 năm tới, thì người đó nên biết về hệ thống thiết bị cầm tay và các ứng dụng chạy trên chúng. Vì vậy, môn học lập trình trên thiết bị di động là môn học rất cần thiết cho những người học trong lĩnh vực công nghệ thông tin.

Mục đích

Trong bối cảnh tràn ngập thông tin ngày nay, để học lập trình trên thiết bị di động, người học có thể dễ dàng tìm kiếm tài liệu học tập trên internet. Nhưng đối với người mới bắt đầu họ dễ dàng bị lạc trong kho thông tin khổng lồ ấy. Quyển sách “Lập trình ứng dụng android” này ra đời nhằm giúp các bạn sinh viên học môn “lập trình trên thiết bị di động” tốt hơn, có một tài liệu ngắn gọn, súc tích, giúp các bạn tiết kiệm được thời gian khi mới tiếp cận với lập trình trên thiết bị di động.

Độc giả của sách

Cuốn sách này phù hợp với những người mới bắt đầu tiếp cận với lập trình Android. Người học chỉ cần có các kiến thức cơ bản về lập trình và mới làm quen với nguyên lý lập trình hướng đối tượng. Cuốn sách đưa người đọc từ khi mới có một số kiến thức cơ sở về kỹ năng lập trình hướng đối tượng, chưa biết làm thế nào để cài đặt một công cụ phát triển phần mềm (Software Development Kit (SDK)) và môi trường phát triển tích hợp (IDE) cho ứng dụng Android đến có thể thể lập trình các ứng dụng Android sử dụng Java và XML. Sau khi hoàn thành cuốn sách này, bạn đọc sẽ thấy mình ở mức độ trung bình về chuyên môn trong lập trình Android và có khả năng tự tìm hiểu các cấp độ cao hơn.

Nội dung

Trong giáo trình này, độc giả sẽ được cung cấp tổng quan về nền tảng Android, chi tiết các bước để phát triển các ứng dụng Android cơ bản. Sở dĩ chúng tôi chọn phát triển ứng dụng trên android cho môn học lập trình trên thiết bị di động do android là nền tảng di động phổ biến nhất trên thế giới hiện nay. Nội dung chi tiết các chương của sách như sau:

Chương 1: cung cấp thông tin tổng quan về lập trình trên thiết bị di động và lập trình Android, giới thiệu các nền tảng Android và các công cụ phát triển của nó. Đồng thời trong phần này chúng tôi giải thích các khái niệm cơ bản mà bạn cần để tạo ra các ứng dụng Android đơn giản. Chúng tôi sẽ giới thiệu cách ứng dụng được tạo ra và chúng tôi sẽ đi qua các lớp activity, các lớp chính trong giao diện người dùng của ứng dụng. Cuối chương, chúng tôi hướng dẫn chi tiết các bước để độc giả có thể

dễ dàng bắt đầu tạo và chạy ứng dụng đầu tiên, “hello world”.

Chương 2: Chúng tôi giới thiệu vai trò quan trọng trong việc trình bày một giao diện người dùng của ứng dụng và cung cấp chi tiết các lớp và các ví dụ để người học có thể tạo ra các giao diện người dùng phức tạp.

Chương 3: cung cấp các kiến thức về truyền thông điệp, intent và lập trình dịch vụ cho ứng dụng Android.

Chương 4: Chương này trình bày cách thức để người học có thể tạo ra các ứng dụng có lưu trữ dữ liệu trên Android.

Chương 5: Cung cấp cho bạn cách thức để xuất bản ứng dụng của mình tới người dùng chung và đưa lên kho ứng dụng Google play nói riêng.

Giới hạn

Nội dung cuốn sách chỉ đề cập đến Android cơ bản và cho những người mới bắt đầu như trên đã nói ở trên nên phần nâng cao và chuyên sâu trong Android không được đề cập đến.

Các kí hiệu

Các kí hiệu sau được sử dụng trong sách:

- Làm nổi bật các thuật ngữ mới, các từ quan trọng trong giáo trình: in đậm nghiêng
- Các chỉ dẫn, chú ý (tips/warning): chữ in nghiêng
- Các URL tham khảo (Cross-reference): <https://developer.android.com>
- Mã nguồn được đặt trong khung

- Các bước thực hiện trong phần thực hành: Ví dụ, dãy các thao tác sau: *File* → *New* → *project* có nghĩa là kích chọn File, kích chuột trái vào New project.
- Các mã nguồn tham khảo trong đĩa CD kèm theo sách

Tập thể biên soạn dù đã cố gắng nhưng không thể tránh khỏi những thiếu sót, chúng tôi rất mong nhận được những ý kiến đóng góp của độc giả để giáo trình hoàn thiện hơn. Mọi ý kiến đóng góp gửi về địa chỉ email *ntlan@ictu.edu.vn*. Chúng tôi rất vui mừng và trân trọng những ý kiến đóng góp của bạn đọc.

Thái Nguyên, ngày 25 tháng 5 năm 2017

Nhóm tác giả

Chương 1

Tổng quan về lập trình trên thiết bị di động

1.1	Sơ lược về hệ điều hành trên thiết bị di động	2
1.2	Hệ điều hành Android	7
1.3	Môi trường phát triển ứng dụng Android	18
1.4	Bắt đầu với Android	20
1.5	Các kiểu dữ liệu trong Android	38

Chương 1 nhằm mục đích cung cấp các cơ sở ban đầu để người đọc có thể dễ dàng tiếp cận các kỹ năng chi tiết trong lập trình android trong các chương sau. Vì vậy chương này trình bày các kiến thức tổng quan về hệ điều hành trên thiết bị di động và hệ điều hành Android. Đồng thời, phần này cũng hướng dẫn chi tiết các bước từ cài đặt môi trường, viết một dự án Android (Android project) đến chạy thử nghiệm ứng dụng trên máy ảo cho người mới bắt đầu với lập trình android có thể thực hiện hoàn thiện một ứng dụng đơn giản.

1.1 Sơ lược về hệ điều hành trên thiết bị di động

1.1.1 Hệ điều hành trên thiết bị di động

Hệ điều hành trên thiết bị di động, thường gọi tắt là hệ điều hành di động (mobile OS), là một phần mềm chạy trên các thiết bị di động (điện thoại di động, điện thoại thông minh, máy tính bảng, thiết bị cầm tay PDA, thiết bị hỗ trợ đọc sách,...), có chức năng quản lý các tài nguyên phần mềm và thiết bị phần cứng trên thiết bị. Tương tự như các hệ điều hành chạy trên máy tính cá nhân (Windows, Linux, Mac OS, v.v..), hệ điều hành di động cũng đóng vai trò trung gian trong giao tiếp giữa người dùng và thiết bị. Bên cạnh đó, các hệ điều hành di động hiện đại hỗ trợ những tính năng đặc trưng của thiết bị di động thông minh: màn hình cảm ứng (touchscreen), mạng di động (cellular network), bluetooth, mạng không dây (Wi-Fi), định vị GPS (GPS navigation), camera, nhận dạng tiếng nói (speech recognition), ghi âm, chơi nhạc, truyền thông tầm ngắn (Near-Field Communications, NFC), v.v..

Rất nhiều người biết được sự tồn tại của hai hệ điều hành di động chính: iOS và Android. Nhưng đó không phải hệ điều hành di động duy nhất. Windows Phone, Blackberry, Nokia Asha và webOS (trước đây là Palm OS) là một số hệ điều hành di động khác đang được sử dụng.



Hình 1.1: Biểu tượng của một số hệ điều hành trên điện thoại di động.

1.1.2 Lịch sử

Cùng với sự phát triển mạnh mẽ của các thiết bị di động và đặc biệt là điện thoại di động, các hệ điều hành trên thiết bị di động cũng có lịch sử phát triển phong phú. Lịch sử phát triển ấy có thể tóm lược trong Bảng 1.1.

Bảng 1.1: Lịch sử các hệ điều hành trên thiết bị di động.

Năm	Tên HĐH	Mô tả
1974	DynaTAC 8000x	DynaTAC của Motorola được sản xuất từ năm 1974 là điện thoại di động đầu tiên. Nó được bán ra ngày 13/3/1984 với giá 3.995 USD
1982	Mobira Senator	Mobira Senator chiếc điện thoại di động đầu tiên của Nokia, nặng 9,8 kg
1992	Sony CM-H333	Sony Electronics mới cho ra mắt điện thoại đầu tiên Sony CM-H333
1994	Simon	Máy Simon của IBM có một màn hình cảm ứng hỗ trợ email và các tính năng PDA

Bảng 1.1 – Lịch sử mobile OS (tiếp)

Năm	Tên HĐH	Mô tả
1996	Palm Pilot 1000	Thiết bị hỗ trợ cá nhân kĩ thuật số (PDA) Palm Pilot 1000 ra đời với hệ điều hành Palm
1996	Microsoft Windows CE	Microsoft cho ra đời phiên bản hệ điều hành đầu tiên Windows CE 1.0
1998	Máy nhắn tin RIM 900 Interactive	BlackBerry ra mắt máy nhắn tin
1999	Nokia 7110	Symbian trở thành hệ điều hành điện thoại di động hiện đại đầu tiên trên điện thoại thông minh
2000	Ericsson R380	Symbian trở thành hệ điều hành điện thoại di động hiện đại đầu tiên trên điện thoại thông minh
2001	Kyocera 6035	Điện thoại thông minh đầu tiên sử dụng hệ điều hành Palm

Bảng 1.1 – Lịch sử mobile OS (tiếp)

Năm	Tên HĐH	Mô tả
2002	Nokia 7650 	smartphone đầu tiên của Nokia dùng hệ điều hành Symbian OS (6.1)
2005	N770 	Nokia giới thiệu hệ điều hành di động MeaGo trên máy tính bảng đầu tiên N770
2006		BlackBerry ra mắt điện thoại thông minh đầu tiên của mình
2007		Kỉ nguyên của iOS mở ra với sự ra đời của iPhone OS của Apple. iPhone OS 2.x ra mắt năm 2008, iPhone 3 (2009), iOS 4 (2010), iOS 5 (2011), iOS 6 (2012), iOS 8 (2014), iOS 10 cùng với iPhone 7 plus (10/2016)
2008	T-Mobile G1 	Android 1.0 với HTC Dream (T-Mobile G1) điện thoại thông minh đầu tiên sử dụng hệ điều hành Android

Bảng 1.1 – Lịch sử mobile OS (tiếp)

Năm	Tên HĐH	Mô tả
2009	Palm Pre	webOSvới Palm Pre năm 2012 WebOS chính thức khai tử
2009	Samsung S8500	Samsung công bố ra mắt hệ điều hành di động Bada với sự ra đời của Samsung S8500
2010		Windows Phoneđược phát hành
2011		Meego hệ điều hành dựa trên Linux là sự kết hợp giữa Maemo của Nokia và Moblin của Intel (sự kết hợp của nhà sản xuất điện thoại lớn nhất thế giới với một nhà sản xuất chip lớn nhất). Ngày 1.1.2014, Nokia chính thức khai tử hệ điều hành Symbian và MeaGo

Bảng 1.1 – Lịch sử mobile OS (tiếp)

Năm	Tên HĐH	Mô tả
2013	Ubuntu Touch 	Canonical ra mắt Ubuntu Touch phiên bản của Linux được thiết kế cho điện thoại thông minh. Các hệ điều hành được xây dựng trên nhân Linux Android sử dụng trình điều khiển Android nhưng không sử dụng bất kỳ mã Java nào như của Android
2013	BlackBerry 10 	BlackBerry phát hành hệ điều hành mới của họ cho điện thoại thông minh và máy tính bảng BlackBerry 10

1.2 Hệ điều hành Android

Android là hệ điều hành di động dựa trên một phiên bản sửa đổi của Linux. Android là một nền tảng mã nguồn mở bao gồm hệ điều hành, middleware và các ứng dụng chủ chốt dành riêng cho các thiết bị di động. Mục tiêu của nó là để điều hành, quản lý phần cứng và các tài nguyên phần mềm trên thiết bị. Nó đóng vai trò trung gian trong việc giao tiếp giữa người sử dụng và phần cứng của máy, cung cấp một môi trường cho phép người sử dụng phát triển và thực hiện các ứng dụng của họ một cách dễ dàng.

1.2.1 Lịch sử phát triển

Tổng công ty Android (Android, Inc.) được thành lập tại Palo Alto, California vào tháng 10 năm 2003 bởi Andy Rubin (Hình 1.2 (đồng sáng lập công ty Danger), Rich Miner (đồng sáng lập Tổng công ty Viễn thông Wildfire), Nick Sears (từng là Phó giám đốc T-Mobile), và Chris White (trưởng thiết kế và giao diện tại WebTV). Google mua lại Tổng công ty Android vào ngày 17 tháng 8 năm 2005. Tại Google, nhóm do Rubin đứng đầu đã phát triển một nền tảng thiết bị di động phát triển trên nền nhân Linux. Tôn chỉ của Rubin là: “Các thiết bị di động thông minh hơn có thể biết được vị trí và sở thích của người dùng”. Ngày 5 tháng 11 năm



Hình 1.2: Andy Rubin – người đã tạo ra hệ điều hành Android.

2007, Liên minh thiết bị cầm tay mở (Open Handset Alliance) được thành lập với mục đích phát triển các tiêu chuẩn mở cho thiết bị di động. Cùng ngày, Android cũng được ra mắt với vai trò là sản phẩm đầu tiên của Liên minh. Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước.

1.2.2 Các tính năng

Vì Android là mã nguồn mở và sẵn sàng tự do cho các nhà sản xuất để tuỳ chỉnh, không có cố định cấu hình phần cứng và phần

mềm. Các phiên bản sau này của Android có bổ sung thêm một số tính năng mới. Tuy nhiên, về cơ bản Android có hỗ trợ các tính năng chính sau:

- Lưu trữ - Sử dụng SQLite, một cơ sở dữ liệu, để lưu trữ dữ liệu
- Kết nối - Hỗ trợ GSM / EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth (bao gồm A2DP và AVRCP), WiFi, LTE và WiMAX
- Nhắn tin - Hỗ trợ cả SMS và MMS
- Trình duyệt Web - dựa trên WebKit mã nguồn mở, cùng với công cụ JavaScript của Chrome
- Hỗ trợ truyền thông - Bao gồm hỗ trợ các phương tiện sau: H.263, H.264 (3GP hoặc MP4), MPEG-4 SP, AMR, AMR-WB (trong bộ chứa 3GP), AAC, HE-AAC (ở định dạng MP4 hoặc 3GP container), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF và BMP
- Hỗ trợ phần cứng - Bộ cảm biến tốc độ, máy ảnh, la bàn kỹ thuật số, cảm biến, và GPS
- Đa cảm ứng - Hỗ trợ màn hình cảm ứng đa điểm
- Đa tác vụ - Hỗ trợ các ứng dụng đa tác vụ
- Hỗ trợ Flash
- Tethering - Hỗ trợ chia sẻ kết nối Internet dưới dạng điểm phát sóng có dây/không dây.

1.2.3 Các phiên bản

Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi

trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng.



Hình 1.3: Biểu tượng của các phiên bản Android.

Chi tiết lịch sử của các phiên bản Android được thể hiện trong Bảng 1.2.

1.2.4 Ưu điểm

Android là một hệ điều hành nguồn mở trong đó người dùng có thể sửa đổi, cải tiến, phát triển và nâng cấp theo một số nguyên tắc đã được quy định trước, nó cung cấp một cách tiếp cận thống nhất để phát triển ứng dụng. Phần mềm mã nguồn mở sẽ gần gũi với người dùng hơn bởi chính những người sử dụng là người tạo ra phần mềm đó.

Tính tuỳ biến: mã nguồn mở còn đa dạng trong tuỳ biến nguồn dữ liệu. Những đoạn mã trong chương trình được công khai, nên người dùng có thể thêm các chức năng mà người dùng muốn có.

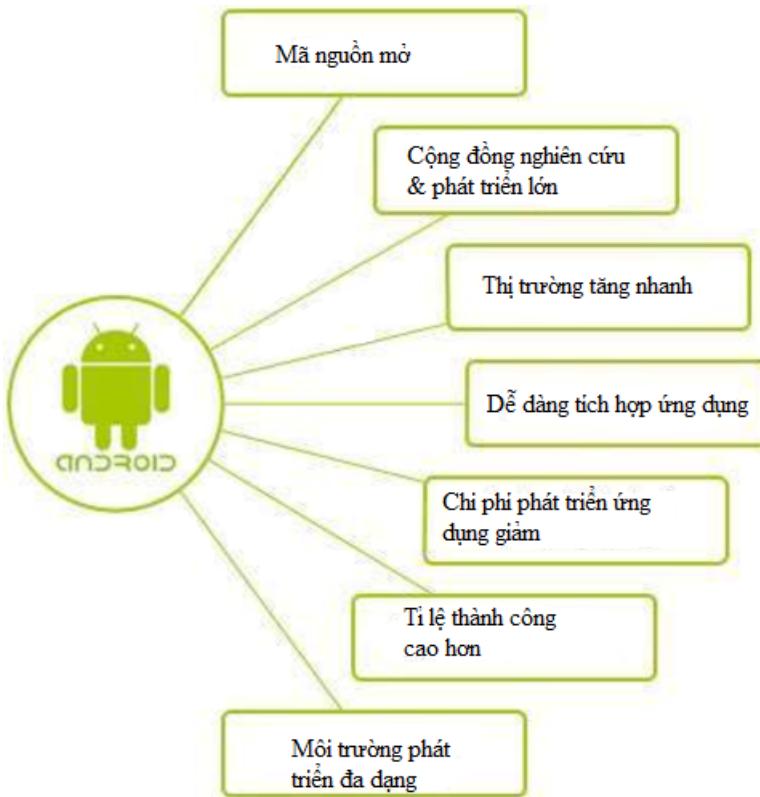
Android được phát triển và nâng cấp rất nhiều. Chính vì công

Bảng 1.2: Lịch sử các phiên bản Android

Phiên bản	Ngày xuất bản	Tên
1.1	9/2/2009	
1.5	30/4/2009	 Cupcake
1.6	15/9/2009	 Donut
2.0/2.1	26/10/2009	 Eclair
2.2 – 2.2.3	20/5/2010	 Froyo
2.3 – 2.3.7	6/12/ 2010	 Gingerbread
3.0 – 3.2.6	22/2/ 2011	 Honeycomb
4.0 - 4.0.4	18/10/2011	 Ice Cream Sandwich
4.1 – 4.3.1	9/7/2011	 Jelly Bean
4.4 – 4.4.4	30/10/2012	 Kitkat
5.0 -5.1.1	12/11/2014	 Lollipop
6.0 – 6.01	5/10/2015	 Marshmallow
70 – 7.1.1	22/8/2016	 Nougat

nghệ mở nên cộng đồng những nhà phát triển nền tảng và ứng dụng Android có quyền truy cập, điều chỉnh, nâng cấp và hoàn thiện hơn về tính năng và hiệu quả của nó.

Android miễn phí cho sử dụng thương mại. Android có một công ty lớn nhất, mạnh nhất và sáng tạo nhất đứng đằng sau nó: Google.Thêm vào đó, còn có các tập đoàn lớn khác trong hiệp hội OHA (Open Handset Alliance – Hiệp hội những nhà phát triển các thiết bị cầm tay mở) đứng sau đầu tư hàng tỷ đôla hỗ trợ cho việc nâng



Hình 1.4: Ưu điểm của Android.

cấp và phát triển nền tảng Android. Do đó, việc phát triển Android không những không làm tốn tiền mà còn mở ra rất nhiều cơ hội cho các nhà phát triển phần mềm di động.

Phát triển và quảng bá ứng dụng Android dễ dàng: Chỉ cần cài đặt môi trường Android và nắm vững công nghệ của nó, bạn có thể tạo ra những ứng dụng tối ưu cho cộng đồng, Android cung cấp “chợ ứng dụng” Android Market giúp cho việc quảng bá ứng dụng của nhà phát triển với cộng đồng người sử dụng hoàn toàn dễ dàng với chi phí thấp.

Đối với người sử dụng thiết bị, Android mang lại thuận lợi. Android có kho ứng dụng khổng lồ: Người dùng có thể tìm ra rất

nhiều ứng dụng miễn phí phù hợp với mục đích của mình. Android hỗ trợ đa nhiệm: Hệ điều hành Android trên Olive Pad hỗ trợ tính năng đa nhiệm, cho phép chạy nhiều ứng dụng cùng một lúc. Ví dụ, trong khi nghe nhạc, người dùng có thể lướt web, chơi game, đọc tài liệu ...

Bên cạnh đó Android cũng còn có một số hạn chế riêng so với những hệ điều hành di động khác như:

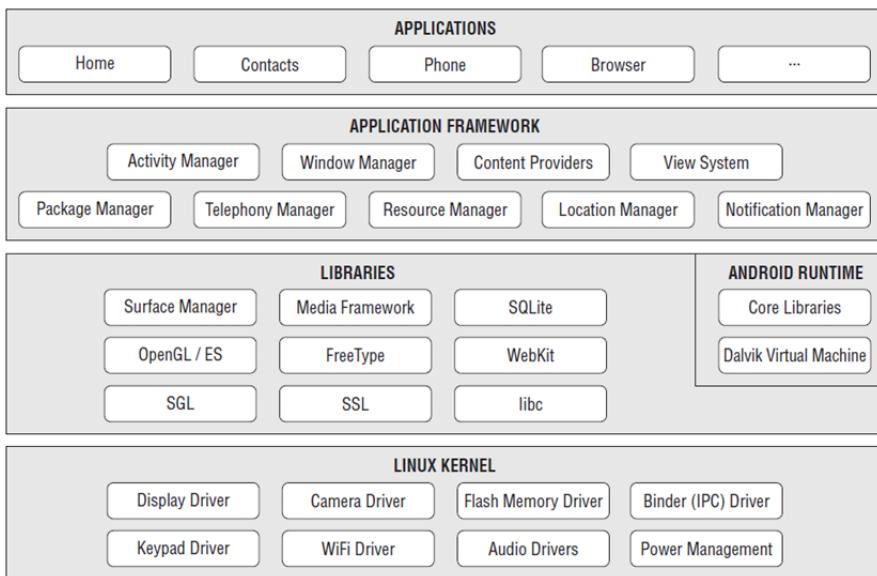
Bảo mật: Vì là hệ điều hành nguồn mở nên tất cả những thông tin về hệ thống bất kì ai cũng có thể nắm được. Đây là ưu điểm nhưng cũng là nhược điểm, bởi các hacker có thể tìm kiếm những lỗ hổng hệ thống và tìm cách tấn công các thiết bị.

Đột phá ý tưởng: Mặc dù, Android đang trên đà phát triển nhanh với nhiều cải tiến mới, hoàn thiện hơn. Nhưng ở góc độ nào đó Android vẫn là người chạy theo những ý tưởng của người khổng lồ iPhone. Những người phát triển ứng dụng cho Android hầu như đều lấy ý tưởng từ iPhone. Đây chính là thách thức cho những nhà phát triển hệ điều hành cũng như những phát triển ứng dụng cho Android.

1.2.5 Nền tảng của hệ điều hành Android

Android là một môi trường phân tầng, xây dựng trên nền của hệ điều hành nhân Linux, và nó bao hàm nhiều chức năng phong phú. Hệ thống giao diện người sử dụng gồm có: cửa sổ, khung nhìn, các tiểu trình để hiển thị các phần tử phổ biến như các hộp biên soạn, danh sách, danh sách thả xuống. Hình sau cho thấy một khung nhìn đơn giản hóa về các tầng phần mềm Android.

1. **Nhân Linux (Linux kernel)** là tầng thấp nhất, chính là phần lõi (core) của hệ điều hành Linux hoạt động trên bộ vi xử lý ARM (Acom RLSC Machine) hoặc Intel, sử dụng các trình điều



Hình 1.5: Kiến trúc của hệ điều hành Android.

khiển thiết bị để hoạt động với hệ thống phần cứng (màn hình, camera, bộ nhớ flash, card mạng, bàn phím, âm thanh, nguồn), quản lý bộ nhớ, điều khiển các tiến trình, hỗ trợ mạng v.v..

2. **Môi trường thực thi (Android runtime)** được xây dựng trên nhân Linux bao gồm hai thành phần: bộ thư viện lõi (core libraries), và máy ảo Dalvik.

Môi trường thực thi chịu trách nhiệm khởi tạo và thực thi ứng dụng Android. Mỗi chương trình ứng dụng Android chạy trong một tiến trình riêng với một máy ảo Dalvik dành riêng cho ứng dụng đó. Máy ảo Dalvik là một phiên bản của máy ảo Java do Google phát triển có nhiệm vụ đọc bytecode (loại mã trung gian khi biên dịch mã nguồn Java) của chương trình, lưu trên máy Android dưới định dạng DEX, và tương tác với bộ thư viện lõi. Bộ thư viện lõi là một bộ phận của bộ thư viện lớp trong Java SE và cung cấp những chức năng cơ bản cho các chương trình Java. Dalvik có cơ chế thu dọn dữ liệu rác

(Garbage Collector) cho phép tự động quản lý và thu hồi bộ nhớ từ các biến không sử dụng.

3. **Bộ thư viện hệ thống (System Libraries)** là bộ thư viện viết trên C/C++ và được tầng trên gọi thông qua giao diện Java. Bộ thư viện này chịu trách nhiệm cho những nhiệm vụ tính toán phức tạp (như đồ họa, phát âm thanh, truy cập cơ sở dữ liệu v.v..) và không phù hợp với thực thi trong máy ảo Dalvik. Phần này gồm thư viện Surface Manager (tạo các cửa sổ giao diện), OpenGL ES (hỗ trợ xây dựng ứng dụng đồ họa 2D và 3D), Media Framework (hỗ trợ xây dựng các ứng dụng về âm thanh, hình ảnh), SSL (cung cấp chức năng bảo mật thiết bị), SQLite (cơ sở dữ liệu quan hệ mã nguồn mở được nhúng trong thiết bị), Webkit (hỗ trợ hiển thị nội dung website), v.v.. Môi trường thực thi và bộ thư viện hệ thống có thể coi là hai tầng con của một tầng lớn gọi là Android Framework (để phân biệt với tầng Application trên cùng, tầng Application Framework, tầng Android Framework, và tầng lõi Linux).
4. **Khung ứng dụng (Application Framework)** chứa các thư viện Java hỗ trợ người dùng giao tiếp với tầng Android framework. Một phần của tầng này do Google cung cấp sẵn, một phần do người lập trình tạo ra. Tầng này cung cấp tất cả các loại dịch vụ sử dụng trong các ứng dụng ở tầng trên cùng (tầng Application). Quá trình phát triển ứng dụng cho Android luôn cần đến các dịch vụ của tầng này: Activity Manager (quản lý chu kỳ sống của các Activity trong ứng dụng Android), Telephony Manager (cung cấp thư viện để truy xuất đến các dịch vụ điện thoại cũng như là thông tin thuê bao), View system (xử lý giao diện trong ứng dụng Android), Location Manager

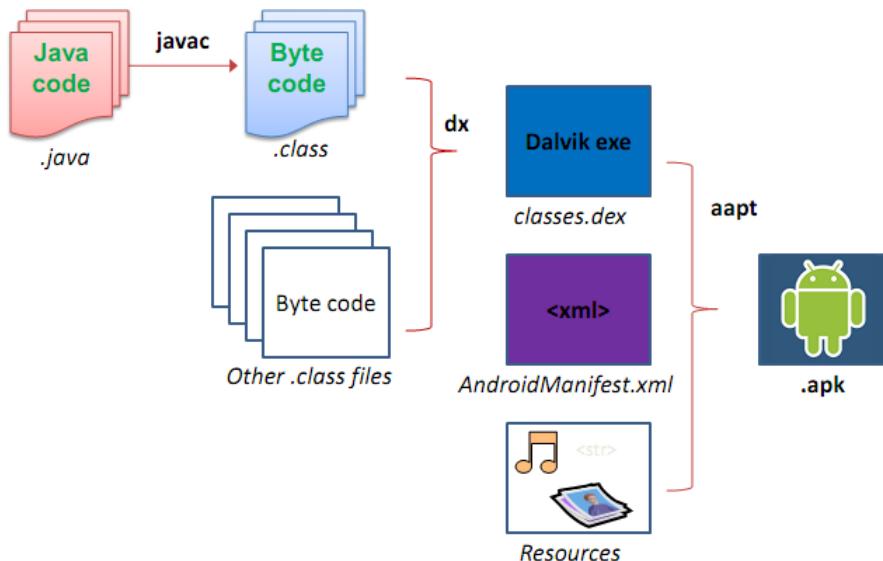
(cung cấp thư viện hỗ trợ người dùng định vị vị trí của thiết bị).

5. **Ứng dụng (Application)** Chúng ta sẽ tìm thấy tất cả ứng dụng Android ở lớp trên cùng. Chúng ta sẽ viết ứng dụng của bạn để được cài đặt trên lớp này chỉ. Ví dụ về các ứng dụng như Sách, Trình duyệt, Trò chơi, v.v.

1.2.6 Quá trình biên dịch và thực thi ứng dụng

Trong nền tảng Android, mã nguồn Java được biên dịch thành các tập tin .class. Bộ SDK của Android chứa một chương trình đặc biệt gọi là DX (viết tắt của Dalvik Executable) để chuyển các tập tin .class về định dạng DEX. Một tập tin .class chứa thông tin một lớp, còn một tập tin .dex chứa tổng hợp của nhiều lớp. Tập tin .dex này sẽ được dùng để chạy trên máy ảo Dalvik tương tự như file .exe trên Windows. Để phục vụ việc phân phối ứng dụng, các tập tin .dex một lần nữa được đóng gói vào một tập tin .apk nhờ một chương trình đặc biệt trong SDK gọi là AAPT. Tập tin .apk này có thể tải lên kho ứng dụng hoặc copy vào bộ nhớ của thiết bị Android. Trong quá trình cài đặt, các tập tin .dex và các tài nguyên của chương trình sẽ được giải nén khỏi file .apk. Quy trình biên dịch và đóng gói chương trình trong Android được trình bày trong Hình 1.6. Khi thực thi một ứng dụng, Android sẽ tạo ra một máy ảo Dalvik cho riêng ứng dụng đó và chạy trong một tiến trình riêng biệt. Mỗi tiến trình này có vai trò tương đương như một người dùng trong hệ thống Linux. Một ứng dụng mặc định không có bất cứ quyền hạn gì tác động đến hệ điều hành, người dùng hoặc ứng dụng khác. Đây là môi trường hoàn toàn cô lập bởi vì nó không có quyền xâm nhập vào ứng dụng hoặc tiến trình khác cũng như tiến trình khác cũng không có quyền xâm nhập vào nó. Do vậy, việc

cho phép trao đổi thông tin và tương tác qua lại giữa các tiến trình với ứng dụng trong Android phải được định nghĩa trước trong ứng dụng đó để khi cài đặt hệ điều hành Android sẽ nhận diện được.



Hình 1.6: Kiến trúc của hệ điều hành Android.

1.2.7 Các thành phần chính của một ứng dụng Android

Một ứng dụng Android bao gồm các thành phần chính sau đây:

- **Activities:** Là khôi cho phép xây dựng lên giao diện người dùng, tương tự như một cửa sổ hoặc một hộp thoại
- **Content providers:** Cho phép trùu tượng hóa dữ liệu được lưu trữ trên thiết bị khi được truy cập bởi nhiều ứng dụng.
- **Services:** các Activities và content providers có thời gian tồn tại ngắn (thời gian sống) và có thể bị tắt bất cứ lúc nào. Còn các service được thiết kế để chạy liên tục và độc lập với các Activity, các service có thể kiểm tra việc cập nhật các RSS, play các bản nhạc, ...

- **Intents:** là các message (thông điệp) hệ thống, chạy liên tục bên trong thiết bị để cảnh báo các ứng dụng về sự thay đổi các sự kiện hoặc trạng thái các thiết bị phần cứng (thẻ nhớ, ...), dữ liệu tới, các sự kiện ứng dụng khác. Người lập trình không thể phản hồi lại các intents nhưng có thể tạo ra các intents cho ứng dụng của mình.

1.3 Môi trường phát triển ứng dụng Android

Để thiết lập môi trường phát triển ứng dụng Android, bạn cần bộ công cụ phát triển phần mềm (SDK - Software Development Kit), một môi trường phát triển tích hợp (IDE-Integrated Development Environment), công cụ phát triển Java (JDK - Java Software Development Kit) và một thiết bị ảo để kiểm thử chương trình.

Một số IDE để phát triển ứng dụng Android:

- **Eclipse:** (<https://eclipse.org/downloads>) Là môi trường phát triển tích hợp cho Java, được phát triển bởi IBM trước đây, hiện nay nó được phát triển bởi Eclipse. Ngoài Java nó còn hỗ trợ nhiều ngôn ngữ khác như PHP, C, C#, XML ...
- **Android Studio:**
(<https://developer.android.com/studio/index.html>) Google đã đưa ra phiên bản Android studio để thay thế cho Eclipse năm 2015. So với Eclipse, Android Studio vẫn cho phép tổ chức ứng dụng gần tương tự như Eclipse, đồng thời vẫn cho phép import các ứng dụng được viết bằng Eclipse trước đây vào trong nó. Android Studio cung cấp một môi trường lập trình tương đối dễ sử dụng cho người lập trình.
- **Basic4Android:** (<https://www.b4x.com/b4a.html>) (còn được gọi là B4A) là một công cụ phát triển ứng dụng nhanh

chóng cho các ứng dụng Android. B4A là một sự thay thế cho lập trình với Java và SDK Android. B4A cho phép thiết kế trực quan giúp đơn giản hóa quá trình xây dựng các giao diện người dùng trên điện thoại và máy tính bảng với các kích thước màn hình khác nhau. Ngôn ngữ lập trình của B4A tương tự như Visual Basic.

Một số IDE phát triển ứng dụng mobile đa nền tảng (phát triển các ứng dụng và chạy được trên các hệ điều hành android, iOS, winphone):

- **Phone Gap:** (<http://phonegap.com>) là một framework mã nguồn mở, tạo ứng dụng di động đa nền tảng sử dụng HTML5, CSS,... do Notobi tạo ra và bán lại cho Adobe năm 2011.
- **Xamarin:** (<https://www.xamarin.com/studio>) là công cụ giúp sử dụng ngôn ngữ C Sharp để lập trình các ứng dụng trên Android, iOS, Window phone, được Microsoft cung cấp miễn phí.
- **Unity:** (<https://unity3d.com/>) là nền tảng phát triển game 3D trên điện thoại di động.
- **Corona SDK:** (<http://coronalabs.com/>) là nền tảng phát triển game 2D trên điện thoại di động.
- **Komodo Edit:** (<http://www.activestate.com/komodo-ide/downloads/ide>) là phần mềm nguồn mở, một sản phẩm của ActiveState, được dựa trên Mozilla, sử dụng cùng framework của Firefox, điều này có nghĩa là Komodo hỗ trợ đa hệ điều hành mà không cần Java, và có các tính năng quen thuộc cũng như hệ thống addon đầy đủ trong Firefox.

- **IntelliJ IDEA** (<http://www.jetbrains.com/idea/index.html>) là IDE có hỗ trợ Java EE, Spring và cả lập trình Android, hỗ trợ bổ sung các plugins để mở rộng tính năng, hỗ trợ script tự động biên dịch, đóng gói Gradle, hỗ trợ Play Framework, Scala... do JetBrains, một công ty phần mềm có những công cụ rất hữu ích tiện dụng cho lập trình viên.

1.4 Bắt đầu với Android

Bây giờ, bạn đọc đã có ý tưởng về một ứng dụng trên điện thoại di động và đã sẵn sàng để biến nó thành hiện thực, triển khai lên chợ ứng dụng. Bạn đọc đang nóng lòng muốn bắt đầu nhận được những lượt tải đầu tiên của bạn, đánh giá và lợi nhuận ... Nhưng chúng ta sẽ bắt đầu như thế nào và bắt đầu từ đâu? Phần này sẽ hướng dẫn các bước sau để độc giả có thể bắt đầu phát triển ứng dụng của mình với Android:

- 1) Cài đặt môi trường phát triển ứng dụng Android
- 2) Tạo một project mới
- 3) Viết mã lệnh
- 4) Biên dịch ứng dụng và chạy thử

Dể tiện trình bày, trong phần này, chúng ta sẽ thực hiện các bước tạo ứng dụng HelloWorld (in ra màn hình dòng chữ "Hello World").

1.4.1 Cài đặt Android studio

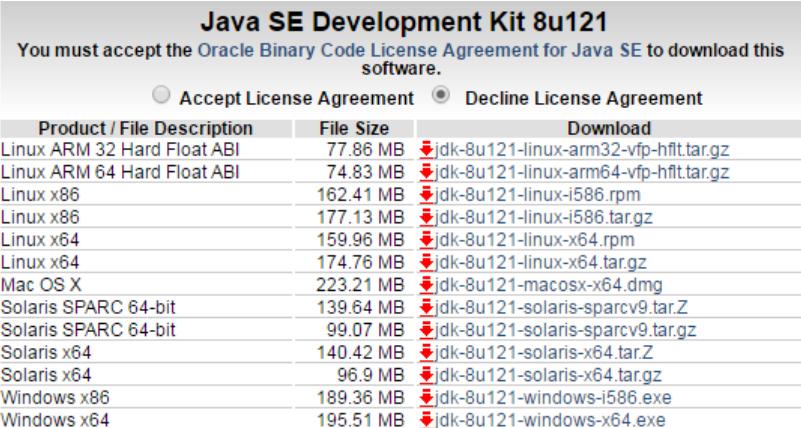
Giáo trình này sẽ hướng dẫn cài đặt Android Studio trên hệ điều hành Window, vì những lý do sau đây:

- Android Studio cung cấp hệ thống xây dựng dựa trên gradles linh hoạt. Mẫu, trình biên tập, ProGuard và khả năng liên kết ứng dụng...
- Android Studio là bộ công cụ sử dụng để xây dựng, thử nghiệm, chạy và đóng gói ứng dụng. Hệ thống này xây dựng thay thế hệ thống Ant được sử dụng với Eclipse ADT. Ta có thể tùy chỉnh, cấu hình quá trình xây dựng. Tạo nhiều APK, sử dụng lại mã và các nguồn thông qua các tập nguồn. Tính linh hoạt của hệ thống xây dựng Android cho phép bạn đạt được tất cả những điều này mà không cần sửa đổi các tệp nguồn cốt lõi của ứng dụng.
- Cấu trúc của dự án và tệp: Chế độ xem dự án Android hiển thị các thư mục nguồn quan trọng ở cấp cao nhất của phân cấp mô đun, nhóm các tệp tin xây dựng cho tất cả các mô-đun trong cùng một thư mục, hiển thị các tệp nguồn, tệp tài nguyên khác nhau...
- Gõ lỗi và Hiệu năng: Android Studio cung cấp chế độ xem bộ nhớ và chế độ xem CPU để bạn có thể dễ dàng theo dõi hiệu suất và mức sử dụng bộ nhớ của mình để theo dõi việc sử dụng CPU, tìm các đối tượng không được phân bổ, xác định vị trí rò rỉ bộ nhớ và theo dõi lượng bộ nhớ mà thiết bị kết nối đang sử dụng. Android Studio cho phép bạn theo dõi sự phân bổ bộ nhớ vì nó giám sát việc sử dụng bộ nhớ. Theo dõi phân bổ bộ nhớ cho phép bạn giám sát nơi các đối tượng đang được phân bổ khi bạn thực hiện các hành động nhất định. Android Studio hỗ trợ chú thích cho các biến, tham số và giá trị trả lại để giúp bạn bắt lỗi, chẳng hạn như trường hợp ngoại lệ con trỏ null và xung đột loại tài nguyên.

- Tính ổn định IDE: Android Studio cho phép trải nghiệm cảm thấy nhanh hơn và mạnh mẽ hơn.

Để cài đặt được Android Studio, yêu cầu cấu hình của máy Windows là: Hệ điều hành Microsoft Windows 10/8/7/Vista/2003 (32 hoặc 64-bit). Các phần mềm cần trước khi bắt đầu lập trình ứng dụng Android là: Java JDK5 trở lên, Java Runtime Environment (JRE) 6 và Android Studio.

Bước 1: Trước khi cài đặt Android, ta cần cài đặt Java. Nếu máy tính người dùng đã được cài đặt Java JDK thì bước này được bỏ qua. Chúng ta có thể tìm thấy bộ công cụ phát triển Java trong trang web: <http://www.oracle.com/technetwork/java/javase/downloads>. Sau khi chọn **Accept License Agreement** (Hình 1.7), ta cần chọn phiên bản phù hợp với hệ điều hành trên máy tính đang dùng và tải về. Sau đó tiến hành cài đặt Java JDK vừa được tải.



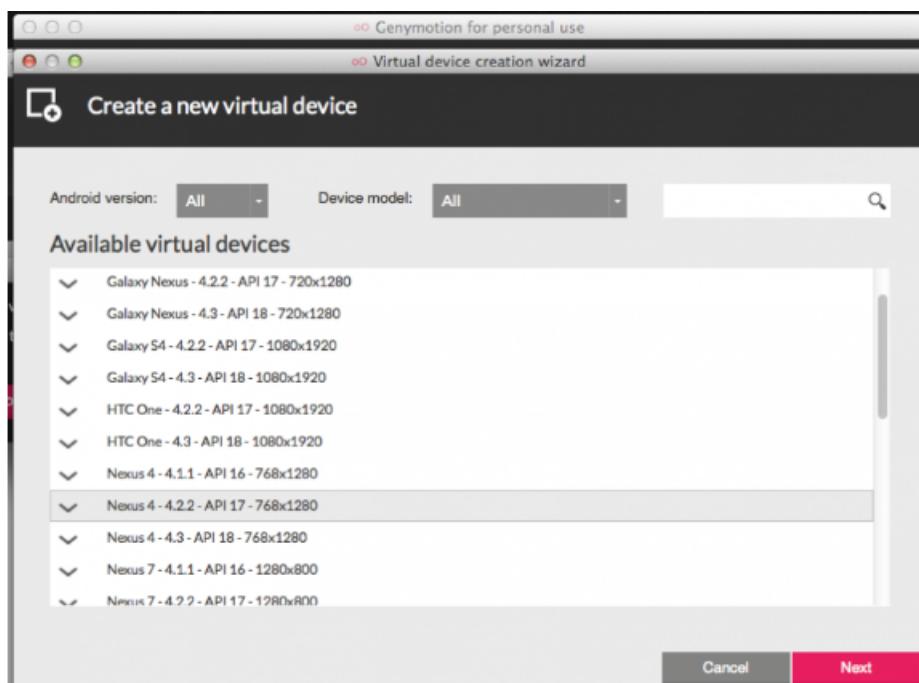
Java SE Development Kit 8u121		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
<input type="radio"/> Accept License Agreement	<input checked="" type="radio"/> Decline License Agreement	
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.86 MB	jdk-8u121-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.83 MB	jdk-8u121-linux-arm64-vfp-hflt.tar.gz
Linux x86	162.41 MB	jdk-8u121-linux-i586.rpm
Linux x86	177.13 MB	jdk-8u121-linux-i586.tar.gz
Linux x64	159.96 MB	jdk-8u121-linux-x64.rpm
Linux x64	174.76 MB	jdk-8u121-linux-x64.tar.gz
Mac OS X	223.21 MB	jdk-8u121-macosx-x64.dmg
Solaris SPARC 64-bit	139.64 MB	jdk-8u121-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.07 MB	jdk-8u121-solaris-sparcv9.tar.gz
Solaris x64	140.42 MB	jdk-8u121-solaris-x64.tar.Z
Solaris x64	96.9 MB	jdk-8u121-solaris-x64.tar.gz
Windows x86	189.36 MB	jdk-8u121-windows-i586.exe
Windows x64	195.51 MB	jdk-8u121-windows-x64.exe

Hình 1.7: Tải công cụ phát triển Java.

Bước 2: Cài đặt máy ảo Geny motion (Hình 1.8).

Mặc dù bộ SDK của Android đã được tích hợp Emulator cho việc chạy thử các ứng dụng nhưng nó có hạn chế là tốc độ

chậm. Ta có thể sử dụng máy ảo Android là Genymotion thay thế. Để có thể tải bộ cài đặt về ta truy cập trang web <http://www.genymotion.com> và đăng kí một tài khoản. Sau khi đăng kí hoàn tất, ta tải bản phù hợp với hệ điều hành trên máy và tiến hành cài đặt.



Hình 1.8: Cài máy ảo chạy android trên Genymotion.

Lưu ý: là phải đăng nhập vào trang web thì mới tải được. Khuyến cáo nên tải phiên bản Genymotion kèm với VirtualBox (một máy ảo tương tự như VMWare nhưng miễn phí và nguồn mở).

Tạo máy ảo: sau khi cài đặt xong genymotion, ta chạy Genymotion (chứ không phải Genymotion shell), chọn thiết bị muốn cài đặt máy ảo để tiến hành download và cài đặt.

Bước 3: Tải Android studio tại trang web: <https://developer.android.com/studio>

[//developer.android.com/studio/index.html](http://developer.android.com/studio/index.html). Lưu ý tải phiên bản Android Studio 32 bit hoặc 64 bit phụ thuộc vào hệ điều hành. Khi tải Android Studio về thì nó sẽ bao gồm các thành phần: Android Studio IDE, Android SDK tools, Android 5.0 (Lollipop) Platform và Android 5.0 emulator system image with Google APIs. khi cài đặt Andoid studio, lúc màn hình giao diện đầu tiên hiện ra chào mừng (Hình 1.9), nhấn vào Next để cài đặt, và chọn I Agree để chấp nhận cài đặt. Lựa chọn nơi cài đặt trong Hình 1.10.

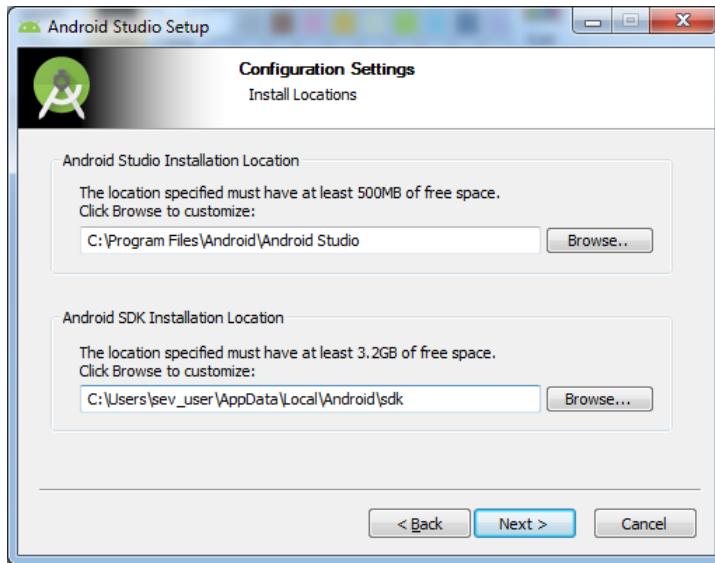


Hình 1.9: Xem xét giấy phép cài đặt của Android studio.

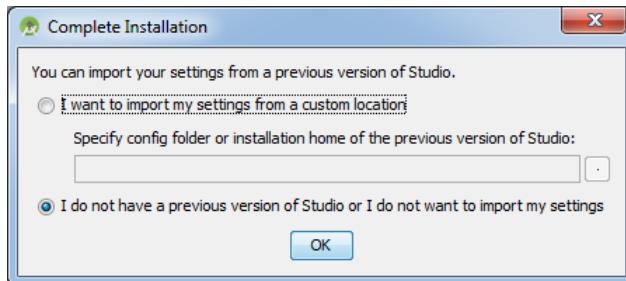
Chọn nút finish để kết hoàn tất cài đặt. Màn hình hiển thị (Hình 1.11) cho phép lựa chọn cập nhập bản android studio đã có trong máy hay cài mới.

Lưu ý:

Nếu chưa cài Java JDK như trong bước 1 thì sau cài Android studio, tạo một project đầu tiên, Android studio sẽ tự động cài đặt java JDK khi máy tính có kết nối internet.



Hình 1.10: Lựa chọn nơi cài đặt Android studio.



Hình 1.11: Lựa chọn cài đặt android studio mới hay nâng cấp phiên bản android studio đã cài.

Nếu lỗi đường dẫn: Nếu sau khi cài đặt đủ 3 bước, khởi động Android studio nếu gặp lỗi không tìm được đường dẫn của java path thì cần tạo biến môi trường để tạo đường dẫn java home.

Nếu không muốn dùng máy ảo Genymotion:, chúng ta bỏ qua bước 2 và cài thiết bị ảo Android trong Android Studio như sau: Khởi chạy Trình quản lý AVD của Android bằng cách kích chuột vào biểu tượng **AVD Manager**. Một màn hình hiển thị thiết bị ảo mặc định xuất hiện. Chúng ta có thể chọn thiết bị ảo mặc định này hoặc kích vào nút tạo thiết bị ảo mới **Create new Virtual**

device để tạo thiết bị ảo khác.

Khi thiết bị ảo được tạo thành công thành công nó có nghĩa là môi trường phát triển Android đã sẵn sàng. Chúng ta có thể bắt tay vào phát triển ứng dụng Android đầu tiên cho mình.

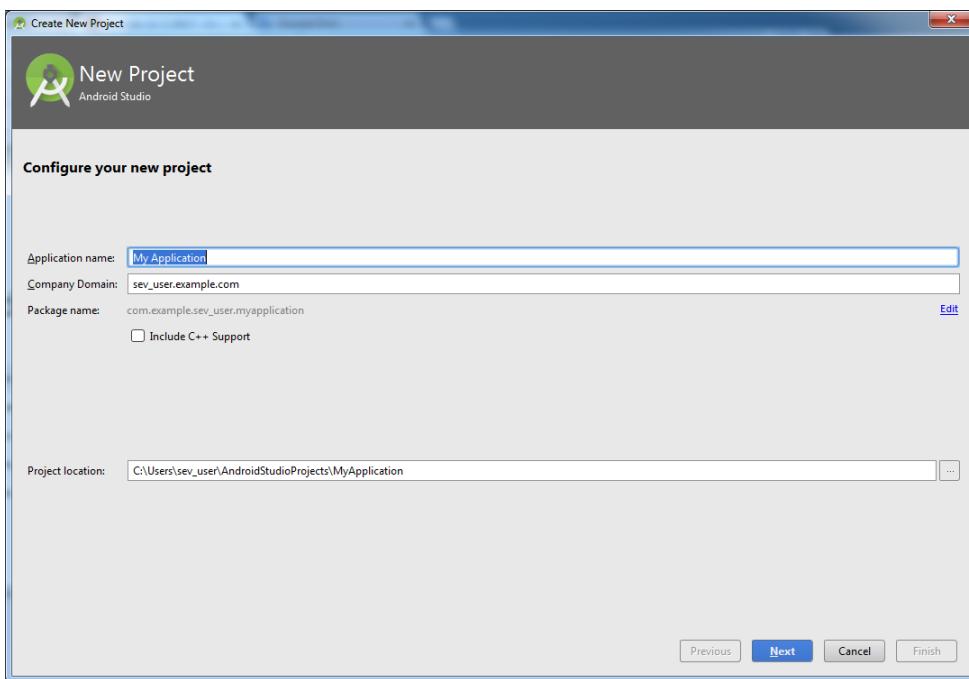
Nếu không dùng máy ảo, chúng ta có thể sử dụng thiết bị thật như sau:

- Kết nối thiết bị thật máy tính.
- **Bật USB debugging** trên thiết bị nếu nó chưa được bật bằng cách vào *Settings* → *Developer options*. (Trong Android, Developer options được mặc định là ẩn. Để hiển thị nó, ta vào *Settings* → *About phone* → *Build*.
- Chạy ứng dụng trong Android Studio: kích vào **app module** trong cửa sổ Project và chọn *Run* → *Run* hoặc kích biểu tượng **Run** trên thanh công cụ (toolbar). Trong cửa sổ **Deployment Target**, chọn thiết bị thật đang kết nối, kích **OK**. Android Studio sẽ cài đặt ứng dụng trên thiết bị đang kết nối và khởi chạy nó.

1.4.2 Tạo một project Android mới

Trong phần này, chúng ta sẽ thực hiện các bước để tạo một ứng dụng trong Android. Ta thực hiện các bước để tạo một project mới trong Android studio và quan sát các thành phần trong project này.

1. **Bước 1:** Thực hiện theo dây tùy chọn *File* → *New project* → *Configure your new project* → *select factor your application is run on* → *add activity* → *Customise your activity* và cuối cùng chọn *finish*. Bây giờ, đặt tên cho ứng dụng là *HelloWorld*, như Hình 1.12.

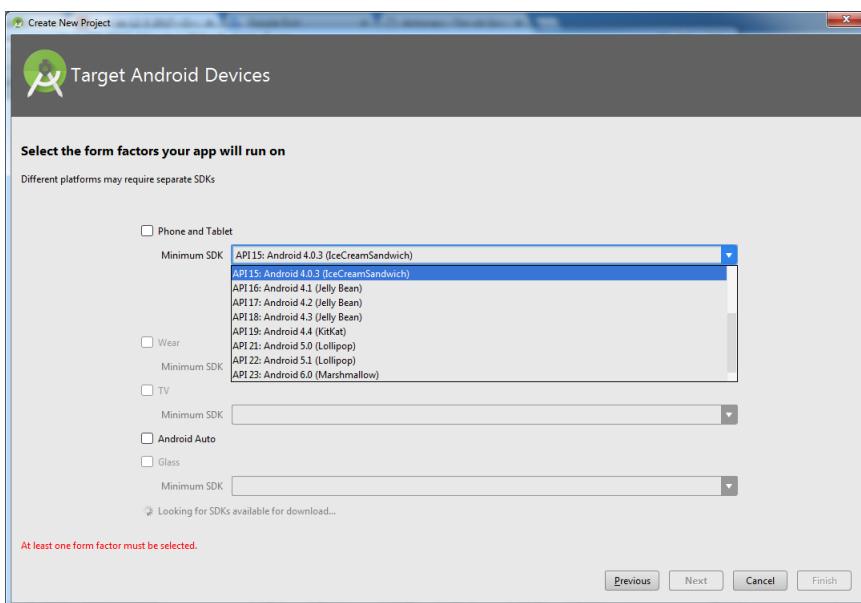


Hình 1.12: Màn hình tạo một dự án mới.

- Tên Project: là tên của dự án. *Tên nên được đặt theo quy tắc viết liền không dấu và viết hoa các chữ cái đầu từ.*
- Application name: tên của ứng dụng cần tạo.
- Company Domain: tên miền của công ty hoặc có thể đặt theo ý thích, khi đặt tên ở company domain nó sẽ tự sinh ra tên gói (package name).
- Package name: Đây là mã của ứng dụng, mỗi ứng dụng có một mã duy nhất và không thể trùng nhau. Khi đăng tải lên Google Play thì nó không được bắt đầu bằng example. Bình thường nó sẽ lấy tên miền nối với tên project nhưng ta có thể sửa bằng cách ấn nút Edit ở bên phải. *Tên gói nên viết thường và phải ít nhất có 1 dấu chấm ngăn cách, nên tránh đặt tên package có chữ Android trong đó vì khi tải lên Google play sẽ không chấp*

nhận.

- Project location: chọn thư mục lưu project.

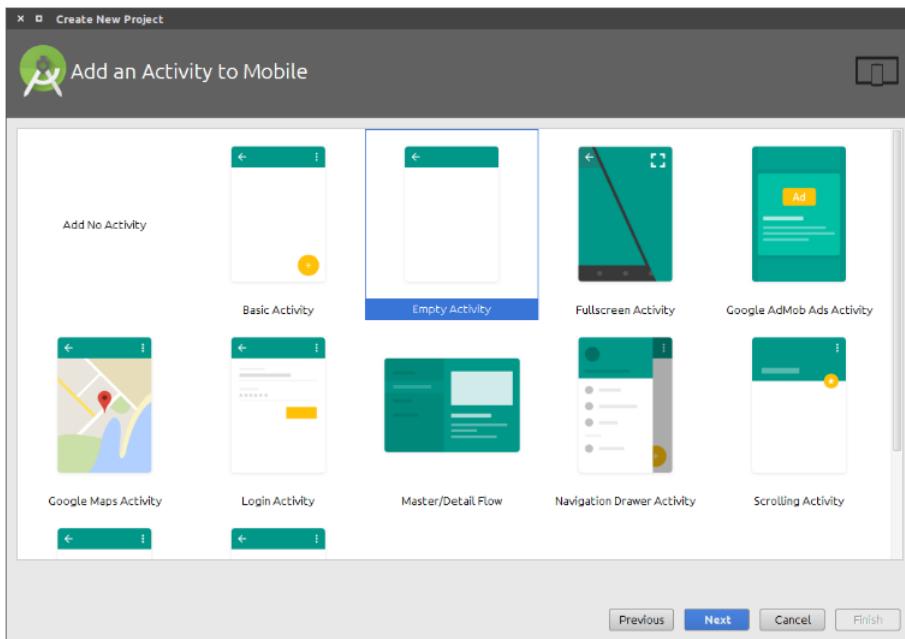


Hình 1.13: Chọn nền tảng của thiết bị chạy ứng dụng.

2. **Bước 2:** Sau khi nhập đầy đủ thông tin trên, bấm Next để tiếp tục. Màn hình yêu cầu chọn nền tảng của thiết bị (platform device) sẽ chạy ứng dụng xuất hiện như Hình 1.13. Mặc định Android Studio sẽ chọn là Phone and Tablet – ứng dụng chạy trên điện thoại và máy tính bảng. Chúng ta ấn Next để tiếp tục. *Lưu ý: minimum SDK trong hình là chọn API tương đương với các điện thoại chạy Android 4.0.3 trở lên sẽ chạy được ứng dụng của mình. Trong trường hợp điện thoại Android thấp hơn thì chắc chắn sẽ không chạy.*

3. **Bước 3:** Màn hình cho phép chọn các activity xuất hiện (Hình 1.14). Ở màn hình này có khá nhiều activity mỗi activity có 1 chức năng riêng, chúng ta sẽ tìm hiểu dần các loại activity trong các chương sau. Ở đây, ta giải sử chọn một activity trống

→ *EmptyActivity* → *Next*. Đặt tên cho Activity và cho layout trong Hình 1.15 và bấm Next.

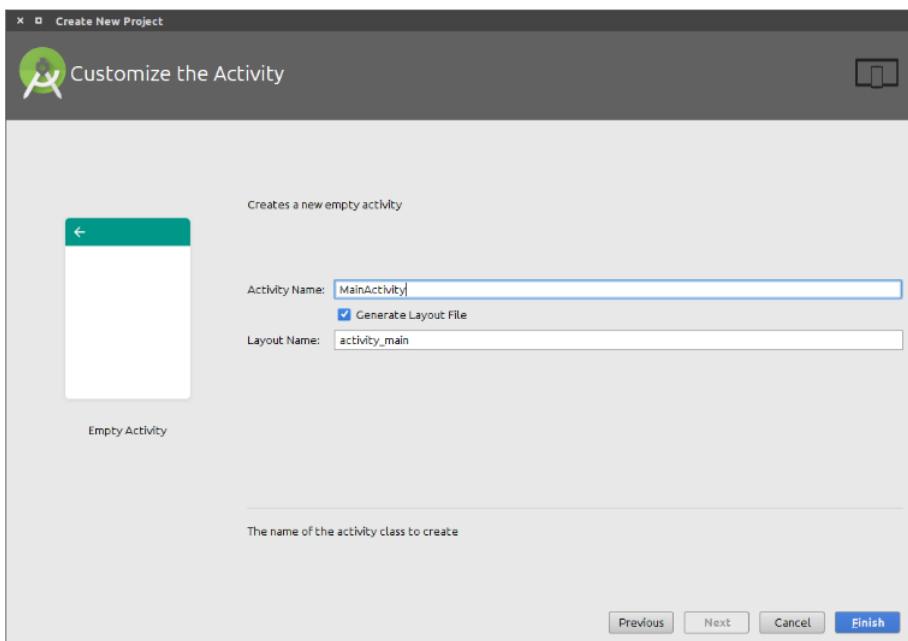


Hình 1.14: Chọn loại activity.

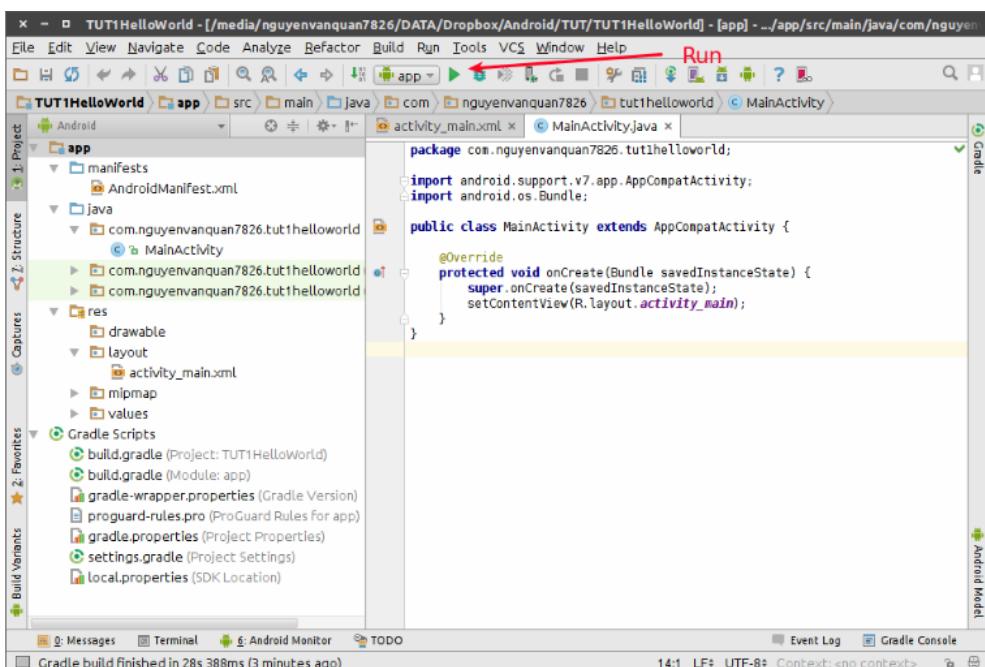
4. Bước 4: Bấm Finish, đợi Android studio build. Màn hình cho phép chúng ta gõ mã lệnh (code) xuất hiện (Hình 1.16). Bên tay trái là khung cấu trúc Project của chúng ta, bên tay phải là trình soạn thảo code. Hiện tại đang có 2 file được mở là `MainActivity.java` và `activity_main.xml`. File Java để viết code sự kiện, dữ liệu, file xml để thiết kế giao diện như mình nói ở trên. Trước khi bắt tay vào gõ lệnh của ứng dụng, chúng ta hãy xem xét cấu trúc của ứng dụng trong Android Studio và các file chính của ứng dụng.

1.4.3 Cấu trúc của một project

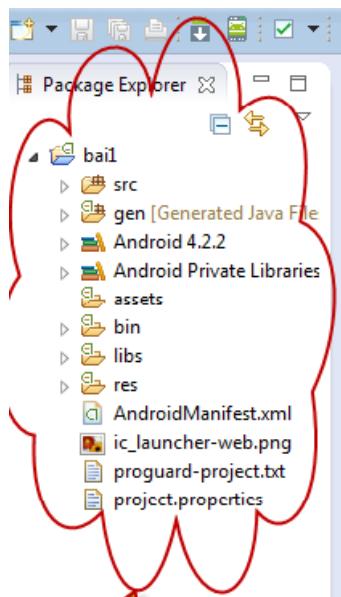
Một project Android được tổ chức theo cấu trúc cây thư mục (Hình 1.17).



Hình 1.15: Đặt tên activity.



Hình 1.16: Giao diện Android studio của ứng dụng HelloWorld.



Hình 1.17: Cấu trúc project Android trong Android Studio.

- androidmanifest.xml: Là file mô tả các thành phần ứng dụng như các services và các activities, ...
- build.xml: Là một file kịch bản phục vụ cho việc biên dịch và cài đặt ứng dụng trên thiết bị thật.
- project.properties: Được sử dụng bởi file kịch bản Ant.
- assets/: Là thư mục chứa các file tĩnh mà người lập trình muốn đóng gói để triển khai trên thiết bị.
- bin/: Chứa các file của ứng dụng khi nó đã được biên dịch
- gen/: Chứa các mã nguồn được sinh ra khi build các tool của Android.
- lib/: Chứa đựng các file thư viện mà ứng dụng sử dụng
- src/: Chứa các file mã nguồn java
- res/: Chứa các tài nguyên như các icon và các GUI layout.

- res/drawable: Chứa các ảnh (jpeg, png, ...)
- res/menu: Mô tả XML cho các menu cụ thể
- res/layout: Mô tả XML cho các giao diện người dùng cụ thể
- res/raw: Các file dùng cho mục đích chung như (như file CSV chứa thông tin tài khoản).
- res/value: Chứa các string hoặc các dimensions
- res/xml: Chứa các file xml dùng chung.

1.4.4 File manifest

Mỗi một project Android có một phần từ khóa là file manifest (Androidmanifest.xml), file này chứa bảng nội dung của project, liệt kê tất cả các thành phần chính của ứng dụng, quyền, ... File này được sử dụng bởi Android runtime để ràng buộc ứng dụng vào hệ điều hành. File Androidmanifest.xml sẽ tự động được sinh ra khi ứng dụng mới được tạo. *File manifest của các ứng dụng nhỏ, đơn giản thì làm việc rất tốt. Một file manifest phù hợp chỉ khoảng hơn 1000 dòng, trong trường hợp kích thước file manifest quá lớn thì ứng dụng có thể bị đơ khi triển khai.*

Nội dung của file manifest được đặt trong cặp thẻ manifest:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.commonsware.android.search">
...
</manifest>
//xmlns:android: chỉ namespace
//package: chỉ gói java mà ứng dụng sử dụng.
```

Các thẻ Permissions, Instrumentations, and Applications:

- Thẻ uses-permission: Xác định quyền cho các chức năng trong ứng dụng.

- Thẻ permission: xác định quyền các các activities và các services được yêu cầu bởi các ứng dụng khác khi truy cập dữ liệu.
- Thẻ Instrumentation: Xác định phần code sẽ được thực thi khi các sự kiện phím hệ thống được nhấn phục vụ cho mục đích giám sát.
- Thẻ uses-library: là thẻ tùy chọn, phục vụ cho mục đích mapping các services.
- Thẻ uses-sdk: Xác định phiên bản SDK của Android sẽ được sử dụng để biên dịch.
- Thẻ Application: Định nghĩa nội dung chính cho ứng dụng.

Thí dụ file Androidmanifest.xml có nội dung như sau:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.commonsware.android">
    <uses-permission android:name="android.permission.ACCESS_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_GPS" />
    <uses-permission android:name="android.permission.ACCESS_ASSISTED_GPS" />
    <uses-permission android:name="android.permission.ACCESS_CELL_ID" />
    <application>
        ...
    </application>
</manifest>
```

Phần tử application là nội dung chính của file manifest, khi tạo ứng dụng Android thì thẻ này có nội dung mặc định như sau:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.commonsware.android.skeleton">
    <application>
        <activity android:name=".Now" android:label="Now">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

```
</manifest>
```

Thẻ uses-sdk: sẽ cho phép chỉ định phiên bản SDK Android cũ nhất sẽ được sử dụng cho ứng dụng. Thí dụ:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.commonsware.android.search">
    <uses-sdk minSdkVersion="2" />
    ...
</manifest>
//Không cần chỉ định phiên bản sdk sau cùng.
```

1.4.5 File MainActivity.java

Main Activity code là một Java file với tên MainActivity.java. Đây là một file ứng dụng thực sự mà cuối cùng được chuyển đổi thành một Dalvik có thể thực thi và chạy ứng dụng của bạn. Sau đây là phần code mặc định được tạo cho ứng dụng HelloWorld (hiển thị dòng chữ “HelloWorld” ra màn hình). Ở đây, R.layout.activity_main tham chiếu tới file activity_main.xml được đặt trong thư mục res/layout. Phương thức onCreate() là một trong nhiều phương thức mà được tính toán khi một Activity được tải.

Dưới đây là mã mặc định được tạo cho ứng dụng Hello World.

```
package com.example.helloworld;
import android.os.Bundle;
import android.app.Activity;
public class MainActivity extends Activity {
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

1.4.6 File strings.xml

File strings.xml nằm trong thư mục res / values và nó chứa tất cả các văn bản mà ứng dụng của bạn sử dụng. Ví dụ, tên của button, nhãn, văn bản mặc định, tên của ứng dụng ...

```
<resources>
    <!--Tên của ứng dụng khi hiển thị trong điện thoại của bạn-->
    <string name="app_name">Ứng dụng HelloWorld</string>
    <!--Chuỗi văn bản hiển thị trên màn hình khi ứng dụng của bạn được mở-->
    <string name="hello_world">Hello World!\nĐây là ứng dụng Android đầu tiên
        của tôi</string>
</resources>
```

1.4.7 Giao diện ứng dụng (Layout)

Layout (Giao diện ứng dụng): ở đây là một ví dụ đơn giản RelativeLayout (chúng ta sẽ nghiên cứu trong các chương 2). Các TextView là một thành phần giúp hiển thị các chuỗi trong ứng dụng Android và nó có các thuộc tính như android: layout_width (dùng để thiết lập chiều rộng của TextView), android: layout_height (dùng để thiết lập chiều cao), string (dùng để gọi đến file strings.xml nằm trong thư mục res / values). Do đó, string/hello_world gọi đến chuỗi hello_world định nghĩa trong file strings.xml để hiển thị. Nội dung file activity_main.xml ví dụ HelloWorld như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="dev4u.com.helloworld.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```

    android:layout_centerInParent="true"
    android:text="@string/hello_world" />

    <!--Dòng android:layout_centerInParent="true" dùng để căn giữa chiều cao và
        chiều rộng màn hình, nhưng nó chỉ áp dụng cho RelativeLayout-->
</RelativeLayout>

```

1.4.8 Sơ lược kiến trúc của một ứng dụng Android

Kiến trúc của một ứng dụng Android có các thành phần cơ bản sau:

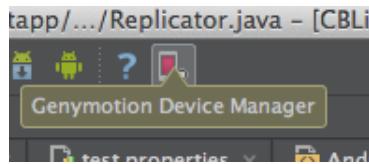
- Services: Thực hiện chức năng nền tảng
- Intent: Thực hiện kết nối giữa các Activity và cơ chế truyền dữ liệu
- Resource Externalization: Các tài nguyên mở rộng như chuỗi, đồ họa
- Notification: Ánh sáng, âm thanh, biểu tượng, thông báo, hộp thoại và toast
- Content Providers: Chia sẻ dữ liệu giữa các ứng dụng.

Trong các chương sau, chúng ta sẽ tìm hiểu sâu hơn về các thành phần trên.

1.4.9 Biên dịch và chạy thử ứng dụng Android

Sau khi hoàn thiện mã lệnh, ta có thể chọn nút Run để chạy ứng dụng. Sau khi chạy ứng dụng, chương trình sẽ hỏi chọn máy điện thoại thật (nếu có kết nối) hoặc chọn mở máy ảo. Nếu chạy máy ảo với Genymotion, ta có thể thực hiện các bước sau:

1. Kết nối với Android Studio:



Hình 1.18: Chạy ứng dụng Android với máy ảo Genymotion.

- Từ Android Studio, ta chọn Preferences, bên trái chọn Plugins, bên phải kích nút Browse repositories..., gõ vào ô tìm kiếm Genymotion, kích chuột phải lên Genymotion → Download and Install.
- Sau khi cài đặt xong, Ok và khởi động lại Android Studio, ta sẽ thấy xuất hiện biểu tượng của Genymotion.
- Kích chọn biểu tượng đó sẽ mở ra cửa sổ Preferences ở phần cấu hình của Genymotion, chọn đường dẫn chính xác của phần mềm Genymotion đã cài vào máy.
- Bây giờ kích thêm lần nữa vào biểu tượng của Genymotion, ta sẽ thấy được danh sách các thiết bị đã cài đặt, hãy chọn 1 thiết bị và bấm chọn Start... là máy ảo sẽ lập tức được chạy lên.
Trường hợp Genymotion hỏi nơi cài đặt SDK thì ta cần tiến hành cài đặt đúng đường dẫn của SDK là được.

2. Sử dụng Genymotion để test ứng dụng trong Android Studio

- Mở project trong Android Studio.
- Click vào biểu tượng Genymotion trong Android Studio, chọn máy ảo và click Start...
- Quay trở lại Android Studio, cho Run ứng dụng, ở cửa sổ hiện ra, chọn máy ảo của Genymotion trong danh sách là xong.

Sau khi chạy lên, chúng ta sẽ thấy điện thoại (thật hoặc ảo) chạy ứng dụng, màn hình có chữ Hello World (Hình 1.19).



Hình 1.19: Giao diện ứng dụng HelloWorld.

Như vậy đến đây, bạn đọc có thể hình dung được các bước thao tác chính để viết một ứng dụng trong Android và sẵn sàng nghiên cứu các kỹ thuật chi tiết trong lập trình Android và tạo ra các ứng dụng phức tạp hơn trong các phần sau của giáo trình.

1.5 Các kiểu dữ liệu trong Android

Thư mục res/values có thể chứa một hoặc nhiều file XML định nghĩa các kiểu dữ liệu khác nhau, bao gồm:

- Kiểu Integer Array (mảng số nguyên) chứa dữ liệu kiểu mảng, được truy xuất file XML chứa layout sử dụng từ khoá @array và từ ứng dụng Android sử dụng lớp R.array. Thí dụ: File res/values/integers.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <integer-array name="bits">
        <item>4</item>
        <item>8</item>
        <item>16</item>
        <item>32</item>
    </integer-array>
</resources>
```

Ứng dụng Android truy cập truy xuất kiểu mảng:

```
Resources res = getResources();
int[] bits = res.getIntArray(R.array.bits);
```

- Kiểu integer là dữ liệu kiểu nguyên, được truy xuất file XML chứa layout sử dụng từ khoá @integer và từ ứng dụng Android sử dụng lớp R.integer.

Thí dụ, file res/values/integers.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <integer name="max_speed">75</integer>
    <integer name="min_speed">5</integer>
</resources>
```

Ứng dụng Android truy cập truy xuất kiểu integer:

```
Resources res = getResources();
int maxSpeed = res.getInteger(R.integer.max_speed);
```

- Kiểu bool chứa dữ liệu kiểu logic, được truy xuất file xml chứa layout sử dụng từ khoá @bool và từ ứng dụng Android sử dụng lớp R.bool. Kiểu dữ liệu này chỉ nhận một trong hai giá trị true hoặc false.

Thí dụ, file res/values/bools.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <bool name="yes">true</bool>
    <bool name="no ">false</bool>
</resources>
```

Ứng dụng Android truy cập truy xuất kiểu bool:

```
Resources res =getResources();
boolean screenIsSmall = res.getBoolean(R.bool.yes);
```

File XML layout truy cập kiểu bool:

```
<ImageView
    android:layout_height="fill_parent"
    android:layout_width="fill_parent"
    android:src="@drawable/logo"
    android:adjustViewBounds="@bool/yes" />
```

- Kiểu color chứa dữ liệu kiểu color (màu sắc), được truy xuất từ file xml chứa layout sử dụng từ khoá color và từ ứng dụng Android sử dụng lớp R.color. Các giá trị màu có thể nhận các hằng tên màu sắc như green, red, blue,... hoặc các hằng số bắt đầu bởi ký tự #.

Thí dụ, file File res/values/colorsxml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="opaque_red">#f00</color>
    <color name="translucent_red">#80ff0000</color>
</resources>
```

Ứng dụng Android truy cập truy xuất kiểu color:

```
Resources res = getResources();
int color = res.getColor(R.color.opaque_red);
```

File XML layout truy cập kiểu color:

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textColor="@color/translucent_red"
    android:text="Hello"/>
```

- Kiểu dimen chứa dữ liệu kiểu dimen (chiều), được truy xuất từ file xml chứa layout sử dụng từ khoá @dimen và từ ứng dụng Android sử dụng lớp R.dimen. Một dimen là một giá trị số được theo sau bởi một đơn vị đo. Android hỗ trợ các đơn vị đo như dp (Density-independent Pixel, 1dp tương ứng với 1 pixel), sp (Scale-independent Pixel), pt(Point, bằng 1/72 inch), px (Pixel), mm (Millimeter), in (Inche).

Thí dụ, file res/values/dimens.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="textview_height">25dp</dimen>
    <dimen name="textview_width">150dp</dimen>
    <dimen name="ball_radius">30dp</dimen>
    <dimen name="font_size">16sp</dimen>
</resources>
```

Ứng dụng Android truy cập truy xuất kiểu dimen:

```
Resources res = getResources();
float fontSize = res.getDimension(R.dimen.font_size);
```

File XML layout truy cập kiểu dimen:

```
<TextView
    android:layout_height="@dimen/textview_height"
    android:layout_width="@dimen/textview_width"
    android:textSize="@dimen/font_size"/>
```

- Kiểu ID chứa định danh các đối tượng, được truy xuất được truy từ file XML chứa layout sử dụng từ khoá @id và từ ứng dụng Android sử dụng lớp R.id. Kiểu này định nghĩa một ID duy nhất trong file XML sử dụng thẻ item.

Thí dụ, file res/values/ids.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <item type="id" name="button_ok" />
    <item type="id" name="dialog_exit" />
</resources>
```

File XML layout truy cập kiểu ID:

<Button android:id="id/button_ok"/>

Ứng dụng Android truy cập truy xuất kiểu ID:

showDialog(R.id.dialog_exit);

Kiểu String chứa giá trị kiểu xâu kí tự, được truy từ file xml chứa layout sử dụng từ khoá @string và từ ứng dụng Android sử dụng lớp R.string. Thí dụ: File res/values/strings.xml có nội dung:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello, World!</string>
</resources>
```

File XML layout truy cập kiểu string:

```
<EditText
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="@string/hello" />
```

Ứng dụng Android truy cập truy xuất kiểu string:

```
String string = getString(R.string.hello);
```

- Kiểu style định nghĩa các khuôn dạng và kiểu dáng của một đối tượng giao diện, được truy xuất từ file XML layout sử dụng từ khoá @style và từ ứng dụng Android sử dụng lớp R.style.

Thí dụ, file res/values/styles.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="CustomText" parent="@style/Text">
        <item name="android:textSize">20sp</item>
        <item name="android:textColor">#008</item>
    </style>
</resources>
```

File XML layout truy cập kiểu styles:

```
<EditText
    style="@style/CustomText"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Hello, World!" />
```

Câu hỏi và bài tập

1. Thị phần của hệ điều hành Android trên các thiết bị di động như thế nào?
2. Hệ điều hành di động là gì? Kể tên một số hệ điều hành thông dụng.
3. Hệ điều hành Android là gì?
4. Có những IDE nào để phát triển ứng dụng Android? Phân tích ưu nhược điểm của các IDE đó.
5. Cài đặt Android Studio trong máy tính.
6. Mô tả kiến trúc của một ứng dụng Android.
7. Một file AndroidManifest là gì?
8. Viết chương trình hiển thị lên màn hình điện thoại dòng chữ "Xin chào bạn".

Chương 2

Lập trình giao diện căn bản

2.1	Đối tượng View	45
2.2	Khai báo các đối tượng widget và layout trong file XML	47
2.3	Các đối tượng widget cơ bản	50
2.4	Các đối tượng Layout	83
2.5	Đối tượng menu	92
2.6	Đối tượng hộp thoại	100
2.7	Thông điệp cảnh báo dạng Toast	103

Chương này cung cấp cho người đọc những kiến thức về nguyên lý hoạt động và cách sử dụng của các thành phần điều khiển (còn gọi là các widgets hay controls) cơ bản trên giao diện ứng dụng Android. Các đối tượng widgets là một trong những thành phần chủ yếu để xây dựng giao diện đồ họa của một ứng dụng Android.

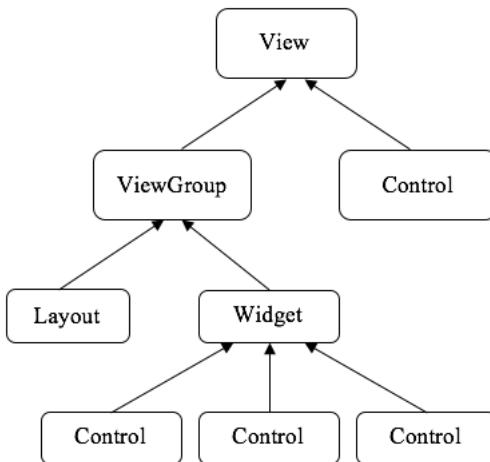
2.1 Đối tượng View

View là lớp giao diện người dùng cơ bản nhất chứa các lớp giao diện như widget hay các lớp tổ chức giao diện (gọi là các layout). ViewGroup là lớp mở rộng của đối tượng View, một ViewGroup có thể chứa nhiều đối tượng View. Các đối tượng widgets trong một ViewGroup kết nối với nó thông qua các View con. Các ViewGroup

cũng được mở rộng để chứa các thành phần layout của ứng dụng.

Các đối tượng Activities giống như một cửa sổ (window) hay một Form chứa các thành phần điều khiển trên giao diện cho phép người dùng tương tác với ứng dụng Android. Mỗi Activity đều gắn liền với một đối tượng View mặc định. Android cho phép xây dựng giao diện ứng dụng bằng cách khai báo và khởi tạo các widgets và các layout phù hợp, nhưng thông thường các nhà phát triển ứng dụng thường định nghĩa giao diện ứng dụng trong một file XML. Điều này cho phép tách biệt riêng giữa phần giao diện và phần code tạo nên sự mềm dẻo và linh hoạt khi phát triển ứng dụng.

Các thành phần giao diện trong file XML được hiển thị trên giao diện người dùng thông qua đối tượng View. Để thiết lập nội dung



Hình 2.1: Mô hình đối tượng View

cho đối tượng View, người lập trình cần sử dụng phương thức:

```
setContentView(R.layout.tên_file_xml);
```

Giả sử file XML chứa các thành phần giao diện được hiển thị trên một View là main_activity.xml trong thư mục layout của ứng dụng thì câu lệnh trên viết như sau:

```
        setContentView(R.layout.main_activity);
```

Các đối tượng widget trên giao diện người dùng trong một Activity được truy cập thông qua View bằng cách sử dụng phương thức findViewById(). Chẳng hạn, người lập trình lấy đối tượng EditText được định nghĩa trong file XML như sau:

```
@Override  
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main_activity);  
    EditText name=(EditText)findViewById(R.id.name);  
}
```

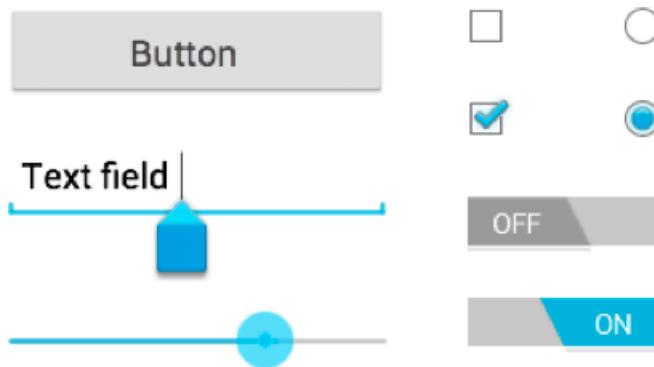
Khi muốn kiểm tra xem đối tượng nào trên giao diện nào được chọn (click) thường sử dụng phương thức getId(). Thí dụ, với mỗi nút nhấn (Button) được chọn thì thực hiện các hành động tương ứng như sau:

```
@Override  
public void onClick(View v) {  
    switch (v.getId()) {  
        case R.id.btn_play: //button thứ nhất  
            play(); break;  
        case R.id.btn_pause: //button thứ hai  
            pause(); break;  
        case R.id.btn_playpre://button thứ ba  
            playPre(); break;  
        case R.id.btn_playnext: //button thứ tư  
            playNext(); break;  
        case R.id.btn_playlist: //button thứ năm  
            showplaylist(mSeekBar);  
    }  
}
```

2.2 Khai báo các đối tượng widget và layout trong file XML

Android cũng giống như các ngôn ngữ lập trình hướng đối tượng khác bao gồm các lớp đối tượng widgets phổ biến trên giao diện

như TextView, EditText, ListView, Spinner, Button, CheckBox và RadioButton, RadioGroup, ImageButton, ... như Hình 2.2. Các đối



Hình 2.2: Thí dụ về các đối tượng widget cơ bản

tượng layout cho phép tổ chức các đối tượng con bên trong nó theo một cách thức nào đó. Trong Android có bốn đối tượng layout (được trình bày trong mục 2.4) bao gồm FrameLayout, LinearLayout, RelativeLayout, TableLayout và AbsoluteLayout. Các đối tượng widgets thường được khai báo bên trong một layout và mỗi file XML (ứng với một cửa sổ giao diện) thường có ít nhất một đối tượng layout ngoài cùng chứa tất cả các đối tượng giao diện khác bên trong nó. Chẳng hạn như đoạn lệnh sau khai báo một đối tượng TextView bên trong một LinearLayout:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/text_id"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a TextView" />
    <!-- khai báo các đối tượng khác -->
</LinearLayout>
```

Thuộc tính của một đối tượng widget hoặc layout được khai báo

theo cú pháp:

android:tên_thuộc_tính = giá trị

Các widgets và layouts cùng kế thừa từ đối tượng View nên chúng có một số thuộc tính chung như sau:

id là định danh của đối tượng, mỗi đối tượng trong một ứng dụng Android phải có một định danh duy nhất. Tên định danh phải được đặt sau từ khoá `@+id/`. Thí dụ, id của đối tượng TextView trong đoạn mã trên là `text_id`. Giá trị id cho phép truy cập đến đối tượng từ mã nguồn java theo cú pháp `R.id.Tên_định_danh`

layout_height xác định chiều cao của đối tượng, thông thường nó có thể nhận một trong các giá trị hằng số như `fill_parent`, `match_parent` và `wrap_content`. `fill_parent` cho phép đối tượng chiếm toàn bộ không gian của đối tượng cha, `match_parent` cho phép đối tượng chiếm toàn bộ không gian còn trống của đối tượng cha, `wrap_content` cho phép đối tượng có kích thước vừa đủ với nội dung hiển thị trong nó. Ngoài ra, người lập trình có thể chỉ định kích thước cụ thể cho chiều cao, chẳng hạn chiều cao là `125dip` hoặc `125dp` như sau:

android:layout_height="125dip"

layout_width thuộc tính này nhận giá trị tương tự như thuộc tính `layout_height` nhưng xác định chiều cao cho đối tượng.

layout_weight cho phép xác định mức độ quan trọng của các đối tượng widgets khi chia sẻ cùng một không gian trống trên giao diện. Thuộc tính này nhận một giá trị nằm trong khoảng từ 0 đến 1, với mức 0 là mức cao nhất. Đối tượng nào có thuộc

tính `layout_weight` càng gần 0 thì càng được cấp nhiều không gian trống khi phóng to hoặc thu nhỏ giao diện ứng dụng.

`gravity` cho phép căn nội dung của đối tượng theo lề trên, dưới, trái hoặc phải tương ứng với các hằng giá trị là top, bottom, right, center, left,....

`padding` cho phép xác định khoảng cách từ biên của đối tượng tới nội dung bên trong nó.

2.3 Các đối tượng widget cơ bản

2.3.1 Đối tượng TextView

Đối tượng TextView thường được sử dụng để miêu tả hoạt động trên giao diện hoặc hướng dẫn người dùng nhập liệu.

Các TextView được khai báo trong file XML cùng với một đối tượng layout như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <TextView>
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="You were expecting something profound?" />
</LinearLayout>
```

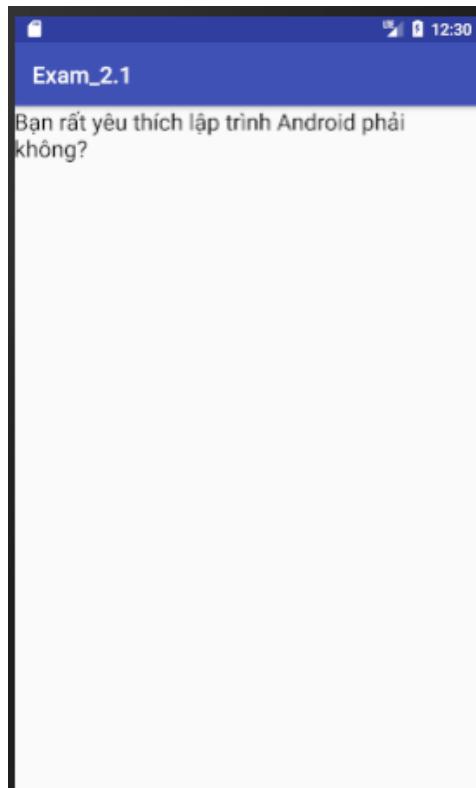
Thuộc tính **`android:text`** chứa giá trị người lập trình muốn hiển thị trên TextView. Các thuộc tính khác của đối tượng TextView thường được sử dụng là:

`android:typeface` thiết lập bìa mặt chữ (các giá trị như normal, sans, serif, monospace).

android:textStyle xác định kiểu chữ như đậm (bold), nghiêng (italic), hoặc vừa nghiêng vừa đậm (bold_italic).

android:textColor thiết lập màu chữ, có thể nhận các giá trị ở hệ hexa từ #000000 đến #FFFFFF hoặc viết ở dạng các hằng xâu chỉ màu sắc như "red", "green", ...

Thí dụ 2.1. Thiết kế giao diện cho ứng dụng Android gồm một TextView hiển thị dòng chữ như Hình 2.3.



Hình 2.3: Thí dụ về đối tượng TextView

File activity_main.xml có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
```

```

    android:layout_height="fill_parent" >
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="20dp"
        android:text="Bạn rất yêu thích lập trình phải không?" />
</LinearLayout>

```

2.3.2 Đối tượng Button

Đối tượng Button (nút nhấn) cho phép người dùng nhấn chọn trên giao diện để thực hiện một công việc cụ thể theo yêu cầu của bài toán. Trên cùng một giao diện ứng dụng có thể có nhiều Button khác nhau. Khai báo đối tượng Button trong file XML như sau:

```

<Button android:id="@+id/start_button"
        android:text="@string/Start"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

```

Đối tượng Button trên có định danh (id) là *start_button* và có nhãn hiện thị với người dùng là biến xâu có id là *Start*. Chiều rộng và chiều cao của Button sẽ tự động tăng lên hoặc giảm xuống theo kích thước của nhãn hiển thị trên nó. Để xử lý sự kiện trên các Button, từ chương trình java cần lấy về định danh của chúng sử dụng phương thức *findViewById()*. Các Button có thể nhận biết được sự kiện click chuột trên giao diện từ người dùng thông qua một bộ lắng nghe sự kiện *OnClickListener* được gắn với chúng. Cách sử dụng đối tượng Button được minh họa qua đoạn code sau:

```

public class Main extends Activity implements OnClickListener {
    public void onCreate() {
        startButton = findViewById(R.id.start_button); //lấy ID
        // gắn bộ lắng nghe cho Button
        startButton.setOnClickListener(this);
        ...
    }
    @Override

```

```

public void onClick(View v) {
    switch (v.getId()) {
        case R.id.start_button:
            // thực hiện công việc 1; break;
            ...
        case R.id.some_other_button:
            // thực hiện công việc n
    }
}

```

Trong thực tế, khi viết phần mềm hiếm khi chỉ sử dụng một cách bắt sự kiện của của đối tượng Button mà có thể kết hợp sử dụng một trong năm cách dưới đây:

Cách 1. Sử dụng lớp thành viên: Cách này rất hữu ích khi chương trình có nhiều đối tượng Button nhưng mỗi đối tượng phải lắng nghe các sự kiện từ các bộ lắng nghe khác nhau. Trong đoạn chương trình mẫu dưới đây, lớp HandleClick là lớp thành viên của lớp Main.

Thí dụ 2-2. Sử dụng lớp trong

```

public class Main extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //gắn bộ lắng nghe cho Button là một thành viên của lớp HandleClick
        findViewById(R.id.button1).setOnClickListener(new HandleClick());
    }
    private class HandleClick implements OnClickListener{
        public void onClick(View arg0) {
            Button btn = (Button)arg0; //ép kiểu đối tượng view về đối tượng Button
            // Tham chiếu tới đối tượng TextView trong file xml
            TextView tv = (TextView) findViewById(R.id.textview1);
            // cập nhật thuộc tính text cho TextView
            tv.setText("Bạn vừa nhấn Button " + btn.getText());
        }
    }
}

```

Cách 2. Sử dụng giao diện (Interface): khai báo một bộ lắng nghe OnClickListener và khởi tạo bằng cách sử dụng cú pháp new

OnClickListener()....

Thí dụ 2-3. Sử dụng giao diện là một bộ lắng nghe

```
public class Main extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //sử dụng handleClick gắn với một bộ lắng nghe sự kiện
        (R.id.button1).setOnClickListener(handleClick);
    }
    private OnClickListener handleClick = new OnClickListener(){
        public void onClick(View arg0) {
            Button btn = (Button)arg0;
            TextView tv = (TextView) findViewById(R.id.textview1);
            tv.setText("Bạn vừa nhấn Button " + btn.getText());
        }
    }
}
```

Cách 3. Sử dụng lớp trong nắc danh (giấu tên): khai báo một giao diện OnClickListener bên trong lời gọi phương thức setOnClickListener(). Đây là cách được sử dụng phổ biến và rất hữu ích khi mỗi bộ lắng nghe có những chức năng riêng mà không thể chia sẻ với các bộ lắng nghe khác. Tuy nhiên, cách này làm cho những người mới bắt đầu lập trình thấy khó hiểu khi đọc mã chương trình.

Thí dụ 2-4. Lớp trong giấu tên

```
public class Main extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        findViewById(R.id.button1).setOnClickListener(new OnClickListener(){
            public void onClick(View arg0) {
                Button btn = (Button)arg0;
                TextView tv = (TextView) findViewById(R.id.textview1);
                tv.setText("Bạn vừa nhấn Button " + btn.getText());}
        });
    }
}
```

Cách 4. Hiện thực hoá giao diện OnClickListener từ lớp Activity: với cách này, chính lớp Activity sẽ trở thành một bộ lắng nghe. Khi

đối tượng Activity được thực thi, phương pháp này sẽ tiết kiệm được một không gian bộ nhớ bởi không đòi hỏi mỗi đối tượng con của lớp phải lưu trữ một phương thức onClick. Lớp Activity có thể hiện thực hóa nhiều giao diện bằng cách sử dụng dấu phẩy (,) phân cách các giao diện.

Thí dụ 2-5. Hiện thực hóa trong lớp Activity

```
public class main extends Activity implements OnClickListener{
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        findViewById(R.id.button1).setOnClickListener(this);
    }
    public void onClick(View arg0) {
        Button btn = (Button)arg0;
        TextView tv = (TextView) findViewById(R.id.textview1);
        tv.setText("You pressed " + btn.getText());
    }
}
```

Cách 5: Sử dụng thuộc tính android:onClick trong file layout.

Thí dụ 2-6. Sử dụng thuộc tính textitandroid:onClick

```
public class Main extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    public void HandleClick(View arg0) {
        Button btn = (Button)arg0;
        TextView tv = (TextView) findViewById(R.id.textview1);
        tv.setText("You pressed " + btn.getText());
    }
}
```

Trong file xml đối tượng Button khai báo thuộc tính *android:onClick* là tên phương thức được thực thi khi click chuột lên Button.

```
<Button android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button 1"
        android:onClick="HandleClick"/>
```

Bốn cách sử dụng đầu tiên có thể sử dụng cho các sự kiện khác nhau như onLong Click, onKey, onTouch, onCreateContextMenu, onFocusChange, nhưng cách thứ năm chỉ áp dụng cho sự kiện onClick. Với cách thứ năm thì trong phần mã chương trình java không cần sử dụng lệnh findViewById và setOnClickListener cho từng nút nhấn.

Thí dụ 2-7. Thiết kế giao diện gồm một TextView và ba Button như hình 2.4, khi nhấn vào mỗi Button thì hiển thị nhãn của Button vừa chọn như dòng “Bạn vừa chọn ‘nhãn Button được chọn’”:



Hình 2.4: Thí dụ về đối tượng Button

File activity_main.xml có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView android:id="@+id/textview1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Click a button."
        android:textSize="20dp"/>
    <LinearLayout android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <Button android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Button 1"
```

```

        android:onClick="HandleClick"/>
<Button android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button 2"
        android:onClick="HandleClick"/>
<Button android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button 3"
        android:onClick="HandleClick"/>
    </LinearLayout>
</LinearLayout>
```

File MainActivity.java có nội dung như sau:

```

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void HandleClick(View arg0) {
        Button btn = (Button)arg0;
        TextView tv = (TextView) findViewById(R.id.textview1);
        tv.setText("Bạn vừa chọn " + btn.getText());
    }
}
```

2.3.3 Đối tượng EditText

Đối tượng TextView chỉ cho phép người dùng đọc dữ liệu trên giao diện, còn đối tượng EditText cho phép người dùng vừa dùng vừa đọc, vừa ghi dữ liệu lên nó. Đối tượng EditText được khai báo trong file xml như sau:

```
<EditText
    android:id="@+id/field"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:singleLine="false" />
```

Các thuộc tính thường sử dụng của đối tượng EditText bao gồm:

android:autoText: cho phép tự động kiểm tra lỗi chính tả khi nhận giá trị true.

android:capitalize: tự động chuyển chữ cái đầu tiên thành chữ in hoa khi nhận giá trị true.

android:digits: chỉ cho phép nhập ký tự số khi nhận giá trị true.

android:singleLine: chỉ cho phép nhập ký tự trên một dòng khi nhận giá trị false.

android:maxLength: nhận giá trị là một số cụ thể đặt trong cặp "" để thiết lập số lượng kí tự tối đa nhập vào.

android:inputType: nhận các giá trị "number" hoặc "text" cho phép chỉ định kiểu dữ liệu được nhập vào.

android:password: nhận giá trị true cho phép người dùng nhập dữ liệu dưới dạng mật khẩu (password).

android:imeOptions: cho phép người dùng nhấn Enter tương ứng với nhấn nút Next trên bàn phím mềm để chuyển đến đối tượng nhập liệu tiếp theo trên giao diện.

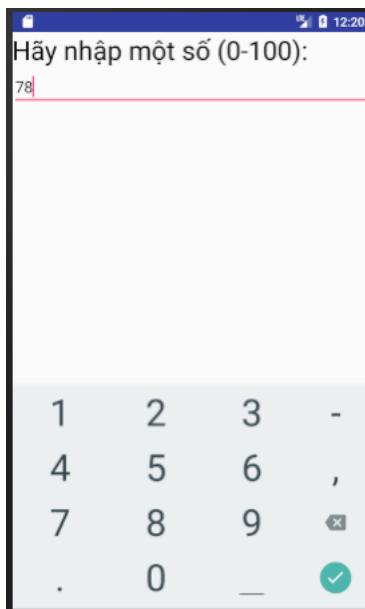
Để lắng nghe các sự kiện nhập liệu trên đối tượng EditText người lập trình phải hiện thực hóa giao diện lắng nghe TextWatcher đồng thời viết đè một trong các sự kiện before *TextChanged()*, *onTextChanged()* và *afterTextChanged()*.

Một bộ lắng nghe sự kiện được gắn liền với một EditText qua phương thức *addTextChangedListener()*.

Thí dụ 2-8. Thiết kế giao diện như hình 2.5. Viết chương trình kiểm tra dữ liệu nhập vào đối tượng EditText có phải là số lớn hơn 100 hay không, nếu số nhập vào lớn hơn 100 thì thay thế số đó là 100. Ngoại lệ sẽ sinh ra nếu dữ liệu nhập vào không phải là số.

File main.xml chứa giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
```



Hình 2.5: Thí dụ về đối tượng EditText

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="Hãy nhập một số (0-100):"/>
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/percent"
        android:text="0"
        android:maxLength="3"
        android:inputType="number"/>
</LinearLayout>
```

File MainActivity.java chứa nội dung như sau:

```
public class MainActivity extends Activity {
    private EditText edit;
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        edit=(EditText) findViewById(R.id.percent);
```

```

//gán bộ lắng nghe cho EditText
edit.addTextChangedListener(percentWatcher);
}
private final TextWatcher percentWatcher = new TextWatcher() {
@Override
/*phương thức được thực thi sau khi kết thúc nhập dữ liệu*/
public void afterTextChanged(Editable s){
try{
Log.d("Percentage", "input: " + s);//in giá trị vừa nhập ra nhật ký
if(Integer.parseInt(s.toString())>100)//Chuyển dữ liệu nhập vào thành dạng
    số
    s.replace(0, s.length(), "100");//thay thế dữ liệu bằng giá trị 100
}
catch(NumberFormatException nfe){
/*Ngoại lệ xảy ra nếu dữ liệu nhập vào không phải là số*/
Log.d("Lỗi:","Dữ liệu nhập không phải số");
}
}
public void beforeTextChanged(CharSequence s, int start, int count, int
    after) {
/* phương thức này thường không được sử dụng phổ biến*/
}
public void onTextChanged(CharSequence s, int start, int before, int count) {
/*phương thức này thường không được sử dụng phổ biến*/
}
};
}

```

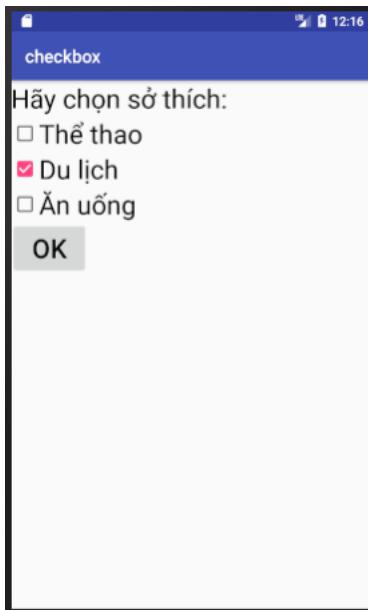
2.3.4 Đối tượng CheckBox

Khi người lập trình muốn thiết kế giao diện cho phép người sử dụng lựa chọn các giá trị cố định trên giao diện thì có thể sử dụng một trong các đối tượng như CheckBox hoặc Radio Button hay Spinner. Đối tượng Radio Button và Spiner sẽ được trình bày ngay trong các mục tiếp theo.

Các đối tượng CheckBox cho phép người dùng lựa chọn một hoặc nhiều lựa chọn tùy ý. Chẳng hạn, trên giao diện liệt kê các sở thích "Thể thao", "Du lịch", "Ăn uống", ... người dùng có thể lựa chọn nhiều sở thích bằng cách chọn nhiều CheckBox trên giao diện.

Khai báo đối tượng CheckBox trong file xml như sau:

```
<CheckBox android:id="@+id/cbxBox1"
```



Hình 2.6: Minh họa đối tượng CheckBox.

```
    android:layout_width="20dp"
    android:layout_height="20dp"
    android:text="CheckBox" //nhan miêu tả
    android:checked="false"/>
```

Thuộc tính *android:checked* cho phép thiết lập trạng thái của CheckBox là true (checked) hoặc false (unchecked).

Các thuộc tính thường dùng của CheckBox trong chương trình java là: *.isChecked()*: trả về giá trị 'true' nếu CheckBox ở trạng thái được chọn, ngược lại trả về giá trị false *.setChecked()*: thiết lập trạng thái checked hoặc unchecked cho CheckBox

Để lắng nghe sự kiện chọn CheckBox thì người lập trình phải gắn bộ lắng nghe sự kiện CheckBox.OnClickListener lên CheckBox và viết đè phương thức onClick(). Ngoài ra, CheckBox có thể được gắn bộ lắng nghe CompoundButton.OnCheckedChangeListener và viết đè phương thức onCheckedChanged().

Thí dụ 2-9. Đối tượng CheckBox

```

public class CheckBoxDemo extends Activity implements
    CompoundButton.OnCheckedChangeListener{
    CheckBox cb; //Khai báo biến
    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        setContentView(R.layout.main);
        cb=(CheckBox)findViewById (R.id.check);
        cb.setOnCheckedChangeListener (this);
    }
    //Viết đè phương thức bắt buộc onCheckedChanged
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
        if (isChecked) {
            //Nếu CheckBox ở trạng thái được chọn thì thực hiện các công việc ở đây
            cb.setText("This checkbox is: checked");
        }
        else {
            //Nếu CheckBox không được chọn thì thực hiện các công việc ở đây
            cb.setText("This checkbox is: unchecked");
        }
    }
}

```

Sự kiện onClick() cho các CheckBox sẽ được trình bày rõ hơn trong thí dụ về các đối tượng lựa chọn 2-12.

2.3.5 Đối tượng Radio Button

Các đối tượng RadioButton thông thường được nhóm theo cùng nhóm để cho phép người dùng chỉ được chọn một lựa chọn trong mỗi nhóm. Chẳng hạn, khi chọn giới tính thì thông thường người dùng được phép chọn một trong hai giá trị 'Nam' hoặc 'Nữ'.

Đối tượng RadioGroup nhóm các đối tượng RadioButton theo tường nhóm. Khai báo đối tượng RadioButton trong file XML như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView

```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="Hãy chọn một hệ điều hành:" />
<RadioGroup
    android:id="@+id/group"
    android:orientation="vertical">//sắp xếp các con theo chiều dọc
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <RadioButton android:id="@+id/radio1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="Android" />
    <RadioButton android:id="@+id/radio2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="iOS" />
    <RadioButton android:id="@+id/radio3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="WindowsPhone" />
</RadioGroup>
<TextView
    android:id="@+id/tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="30dip"
    android:text="" />
</LinearLayout>
```

Để lắng nghe sự kiện chọn trên các RadioButton thì chúng phải được gắn các bộ lắng nghe sự kiện. Mỗi RadioButton có thể ngắn bộ lắng nghe RadioGroup.OnClickListener và viết đè phương thức onClick(). Ngoài ra, người lập trình có thể xử lý sự kiện lựa chọn trên các đối tượng RadioButton thông qua đối tượng cha của nó là RadioGroup. Trong trường hợp này các RadioGroup phải được khai báo định danh trong file XML.

Các phương thức thường dùng của đối tượng RadioGroup trong chương trình java bao gồm:

.check(): thiết lập trạng thái checked cho Radio Button thông



Hình 2.7: Thí dụ về đối tượng RadioButton.

qua id (thí dụ: group.check(R.id.radio1))

`.clearCheck()`: hủy chọn tất cả các Radio trong nhóm

`.getCheckedRadioButtonId()`: lấy về id của Radio Button đang được chọn (nếu không có Radio nào được chọn thì phương thức trả về giá trị -1).

Thí dụ 2-10. Tạo giao diện gồm một TextView và một RadioGroup chứa ba Radio Button như hình 2.7. RadioGroup được gắn bộ lắng nghe sự kiện OnCheckedChangeListener để mỗi khi người dùng chọn một Radio Button trên giao diện thì hiển thị nhãn tương ứng trên đối tượng TextView.

```

public class RadioDemo extends Activity implements
    RadioGroup.OnCheckedChangeListener{
    RadioGroup rg; TextView tv;
    protected void onCreate(Bundle c){
        super.onCreate(c);
        setContentView(R.layout.activity_radio_demo);
        rg=(RadioGroup)findViewById(R.id.group); //lấy định danh RadioGroup
        rg.check(R.id.radio2); //Radio có nhãn Scissors được chọn đầu tiên
        rg.setOnCheckedChangeListener(this); //Gắn bộ lắng nghe cho RadioGroup
    }
}

```

```

tv=new TextView(this); //khai báo đối tượng TextView
tv=(TextView)findViewById(R.id.tv);
}
//Viết đè sự kiện bắt buộc onCheckedChanged, tham số id chứa định danh của Radio
Button được chọn và được so sánh lần lượt với các id của từng Radio Button để
éxác định đối tượng nào được chọn
public void onCheckedChanged(RadioGroup rg, int id) {
    switch(id){
        case R.id.radio1:
            tv.setText("Bạn vừa chọn HDH Android"); break;
        case R.id.radio2:
            tv.setText("Bạn vừa chọn HDH iOS"); break;
        case R.id.radio3:
            tv.setText("Bạn vừa chọn HDH WindowsPhone"); break;
    }
}
}

```

2.3.6 Đối tượng Adapter

Đối tượng Adapter thường được sử dụng kết hợp với các đối tượng AutoCompleteTextView, Spinner và Listview để cung cấp dữ liệu cho các đối tượng này. Trong các mục tiếp theo, đối tượng Adapter sẽ được trình bày cụ thể với từng trường hợp nêu trên.

Đối tượng Adapter được sử dụng nhiều nhất là ArrayAdapter. Mỗi đối tượng ArrayAdapter khi được khởi tạo cần ba tham số. Đó là ngữ cảnh (context), kiểu phần tử và danh sách các phần tử. Thông thường ngữ cảnh là chỉ chính lớp chứa nó, kiểu phần tử thường được chọn là một trong các kiểu có sẵn của Android theo cú pháp `android.R.layout.tên định danh`, chẳng hạn `android.R.layout.simple_list_item_1` hoặc `android.R.layout.simple_list_item_2,...`. Danh sách các phần tử thường là một danh sách các hằng số được định nghĩa theo một số cách khác nhau. Thí dụ:

Cách 1: `String[] items={"this", "is", "a", "really", "silly", "list"};`

Cách 2: `List<String> items = new ArrayList<String>();`

sau đó thêm các phần tử bằng phương thức add(), thí dụ:

```
    items.add("Athos");
    items.add("Porthos");
    items.add("Aramis");
```

Đối tượng ArrayAdapter sẽ được khởi tạo như sau:

```
ArrayAdapter<String> adt = new ArrayAdapter<String>(this,
    android.R.layout.simple_list_item_1, items);
```

Ngoài ra, người lập trình có thể đọc thêm về hai đối tượng Adapter khác là CursorAdapter để chuyển đổi con trỏ và SimpleAdapter để chuyển đổi dữ liệu từ các file XML.

2.3.7 Đối tượng Spinner

Đối tượng Spinner có chức năng giống như đối tượng ComboBox trong lập trình GUI nói chung.

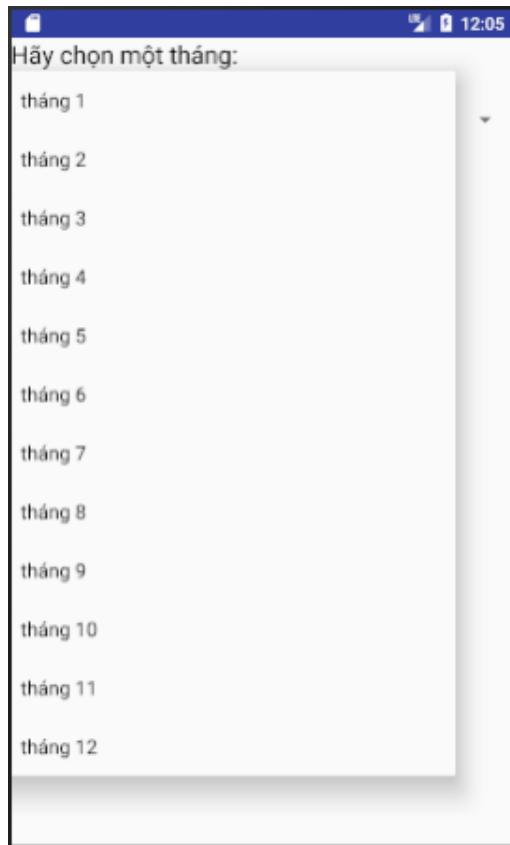
Đối tượng Spinner được định nghĩa trong file XML như sau:

```
<Spinner android:id="@+id/spinner"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
```

Dữ liệu của mỗi đối tượng Spinner thường được lấy từ một đối tượng ArrayAdapter sử dụng các phương thức sau với adt là một Spinner:

```
adt.setDropDownViewResource(android.R.layout.
    simple_spinner_dropdown_item);
spin.setAdapter(aa); //thiết lập Adapter cho Spinner
```

Để lắng nghe sự kiện lựa chọn của người dùng thì mỗi Spinner phải được gắn một bộ lắng nghe sự kiện AdapterView.OnItemSelectedListener và viết đè một trong các phương thức bắt buộc là onItemSelected() hoặc onNothingSelected(). Phương thức onItemSelected() được thực thi khi người dùng lựa chọn một



Hình 2.8: Minh họa đối tượng Spinner

phần tử (gọi là item) từ danh sách của Spinner. Phương thức onNothingSelected() được thực thi khi người dùng không chọn item nào từ danh sách.

Thí dụ 2-11: Tạo giao diện gồm hai đối tượng TextView và một đối tượng Spinner như hình 2.9. Đối tượng Spinner chứa danh sách các hằng số lựa chọn về tháng trong năm. Chương trình java thực hiện xử lý khi có một item trong Spinner được chọn thì hiển thị item đó lên TextView, nếu không có Item nào được chọn thì giá trị hiển thị trên TextView là trống.

File SpinnerDemo.java chứa nội dung như sau:

```
//Lớp chính hiện thực hoá bộ lắng nghe sự kiện AdapterView.OnItemSelectedListener
```

```

public class SpinnerDemo extends Activity implements
    AdapterView.OnItemSelectedListener{
    TextView selection;
    String[] items= { "January", "February",
        "March", "April", "May", "June", "July", "August", "September",
        "October", "November", "December" };
    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        setContentView(R.layout.main);
        selection=(TextView)findViewById (R.id.selection); //đối tượng TextView
        Spinner spin=(Spinner)findViewById (R.id.spinner); //đối tượng Spinner
        spin.setOnItemSelectedListener(this); // gắn bộ lắng nghe cho spinner
        //Tạo đối tượng Adapter chứa dữ liệu cho spinner
        ArrayAdapter<String> adt=new ArrayAdapter<String>(this,
            android.R.layout.simple_spinner_item, items);
        adt.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spin.setAdapter (adt); //gắn dữ liệu vào spinner
    }
    //Viết đè phương thức onItemSelected thực thi khi có một item từ danh sách được
    //chọn
    public void onItemSelected(AdapterView<?> parent, View v, int position, long
        id) {
        selection.setText("Bạn vừa chọn tháng: "+items[position]); //hiển thị giá
        trị item được chọn lên TextView
    }
    //Viết đè phương thức onNothingSelected thực thi khi không có item nào được chọn
    public void onNothingSelected (AdapterView<?> parent) {
        selection.setText(""); //Xoá giá trị trên TextView
    }
}

```

File main.xml chứa nội dung như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30dip"
        android:text="Hãy chọn một Item:" />
    <Spinner
        android:id="@+id/spinner"
        android:layout_width="fill_parent"
        android:layout_height="80dip" />
    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"

```

```
    android:layout_height="wrap_content"
    android:textSize="30dip"
    android:text="" />
</LinearLayout>
```

Kết quả thí dụ 2-11 như hình 2.9.

Thí dụ 2-12. Thiết kế giao diện như hình 2.10. Chương trình java

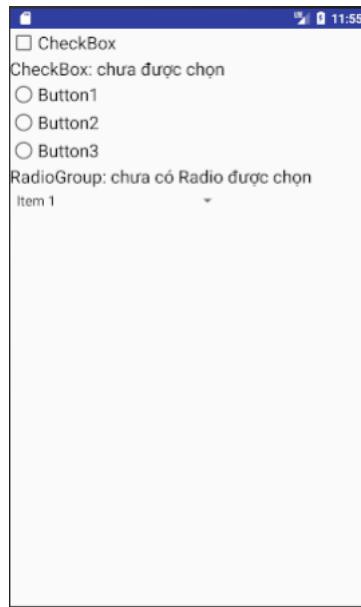


Hình 2.9: Kết quả thí dụ về Spinner

xử lý sự kiện: khi người dùng chọn CheckBox thì hiển thị trạng thái của CheckBox lên TextView thứ nhất, khi người dùng chọn một trong các Radio Button thì hiển thị nhãn của Radio được chọn lên TextView thứ 2. CheckBox và các Radio được gắn bộ lắng nghe sự kiện OnClickListerner, Spinner được gắn bộ lắng nghe sự kiện OnItemSelectedListener.

File main.xml chứa nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
```



Hình 2.10: Thí dụ tổng hợp về đối tượng Checkbox, Radio Button và Spinner

```
    android:layout_height="fill_parent" >
<CheckBox
    android:id="@+id/cbxBox1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="20dp"
    android:text="CheckBox"
    android:checked="false" />
<TextView
    android:id="@+id/txtCheckBox"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textSize="20dp"
    android:text="CheckBox: chưa được chọn" />
<RadioGroup
    android:id="@+id/rgGroup1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton
        android:id="@+id/RB1"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:textSize="20dp"
        android:text="Button1"/>
    <RadioButton
        android:id="@+id/RB2"
        android:layout_height="wrap_content"
```

```
        android:layout_width="wrap_content"
        android:textSize="20dp"
        android:text="Button2" />
    <RadioButton
        android:id="@+id/RB3"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:textSize="20dp"
        android:text="Button3" />
</RadioGroup>
<TextView
    android:id="@+id/txtRadio"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textSize="20dp"
    android:text="RadioGroup: chưa có Radio được chọn"/>
<Spinner
    android:id="@+id/spin"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:textSize="30dp"
    android:layout_marginTop="2dp"/>
</LinearLayout>
```

File SelectExample.java chứa nội dung như sau:

```
public class SelectExample extends Activity {
    private CheckBox checkBox;
    private TextView txtCheckBox, txtRadio;
    private RadioButton rb1, rb2, rb3;
    private Spinner spin;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        checkBox = (CheckBox) findViewById(R.id.cbxBox1);
        txtCheckBox = (TextView) findViewById(R.id.txtCheckBox);
        txtRadio = (TextView) findViewById(R.id.txtRadio);
        rb1 = (RadioButton) findViewById(R.id.RB1);
        rb2 = (RadioButton) findViewById(R.id.RB2);
        rb3 = (RadioButton) findViewById(R.id.RB3);
        spin = (Spinner) findViewById(R.id.spin);
        checkBox.setOnClickListener(new CheckBox.OnClickListener() {
            public void onClick(View v){
                if (checkBox.isChecked()) {
                    txtCheckBox.setText("CheckBox: được chọn");
                }
                else
                {
                    txtCheckBox.setText("CheckBox: chưa được chọn");
                }
            }
        });
    }
}
```

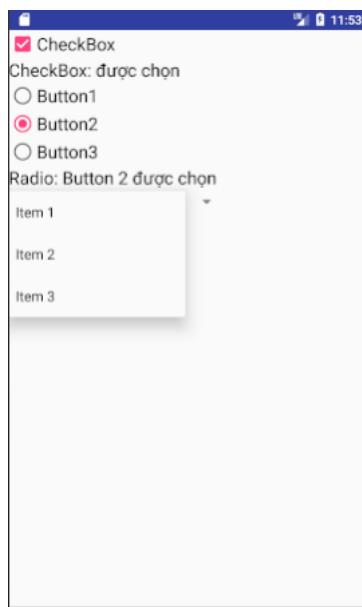
```
        }
    });
rb1.setOnClickListener(new RadioGroup.OnClickListener() {
    public void onClick(View v){
        txtRadio.setText("Radio: Button 1 được chọn");
    }
});
rb2.setOnClickListener(new RadioGroup.OnClickListener() {
    public void onClick(View v){
        txtRadio.setText("Radio: Button 2 được chọn");
    }
});
rb3.setOnClickListener(new RadioGroup.OnClickListener() {
    public void onClick(View v){
        txtRadio.setText("Radio: Button 3 được chọn");
    }
});
List<String> lsMusketeers = new ArrayList<String>();
lsMusketeers.add("Item 1");
lsMusketeers.add("Item 2");
lsMusketeers.add("Item 3");
ArrayAdapter<String> aspnMusketeers =new ArrayAdapter<String>(this,
    android.R.layout.simple_spinner_item,lsMusketeers);
aspnMusketeers.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spin.setAdapter(aspnMusketeers);
spin.setOnItemSelectedListener(new OnItemSelectedListener() {
    public void onNothingSelected(AdapterView<?> arg0) { }
    public void onItemSelected(AdapterView<?> parent, View v,int
        position, long id) {
    }
});
}
}
```

Kết quả thí dụ 2-12 như hình 2.11.

2.3.8 Đối tượng AutoCompleteTextView

Dối tượng AutoCompleteTextView cho phép người dùng nhập dữ liệu nhanh và thuận tiện hơn đối tượng EditText, nó tự động hiển thị danh sách các từ gợi ý để người dùng lựa chọn dựa trên các ký tự đầu tiên được nhập bởi người dùng. Khai báo đối tượng AutoCompleteTextView trong file xml như sau:

<AutoCompleteTextView



Hình 2.11: Kết quả thí dụ 2-12.

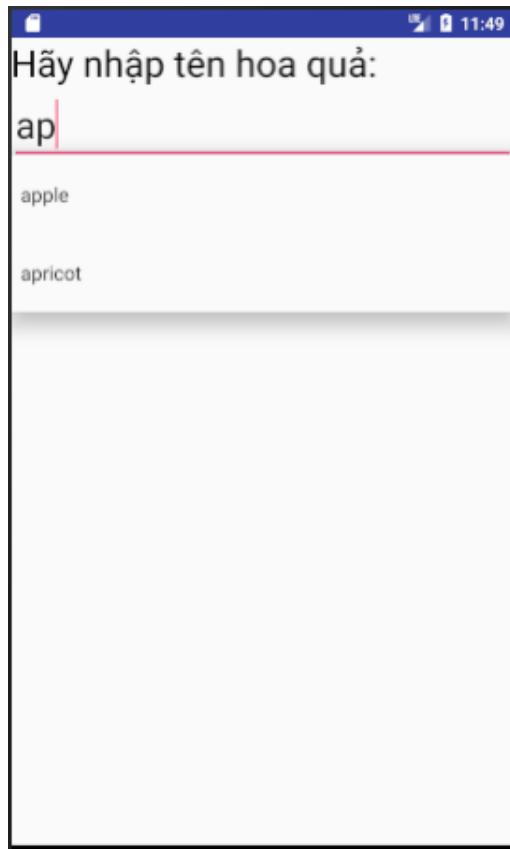
```
    android:id="@+id/autocomplete"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:completionThreshold="2" />
```

Thuộc tính completionThreshold cho phép đặt số lượng các ký tự đầu tiên được so khớp giữa các từ trong thư viện được với từ đang được nhập trong AutoCompleteTextView. Để lắng nghe sự kiện nhập liệu trên đối tượng AutoCompleteTextView, người lập trình phải hiện thực hóa giao diện TextWatcher và viết đè một trong các phương thức onTextChanged(), beforeTextChanged(), afterTextChanged() tương tự như đối tượng EditText.

Phương thức addTextChangedListener() cho phép gắn bộ lắng nghe sự kiện vào đối tượng AutoCompleteTextView.

```
    autocomplete = (AutoCompleteTextView) findViewById(R.id.autocomplete)
    autocomplete.addTextChangedListener(this);
```

Danh sách các từ gợi ý khi nhập dữ liệu cho đối tượng AutoComplete-



Hình 2.12: Minh họa đối tượng AutoCompleteTextView.

TextField được lưu trong một đối tượng Adapter, sau đó được gắn vào đối tượng này sử dụng phương thức setAdapter(), cụ thể như sau:

```
String autocompleteItems [] = {"apple", "banana", "mango", "pineapple", "apricot",
    "orange", "pear", "grapes"};
autocomplete.setAdapter(new ArrayAdapter<String>(this,
    android.R.layout.simple_dropdown_item_1line, autocompleteItems)); }
```

Thí dụ 2-13. Minh họa về hoạt động của đối tượng AutoCompleteTextView như hình 2.12. Đối tượng AutoCompleteTextView cho phép hiện các từ gợi ý khi người dùng nhập tên loại hoa quả, sau khi kết thúc nhập một TextView sẽ hiển thị giá trị vừa chọn trên

AutoCompleteTextView.

File main.xml chứa nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="30dp"
        android:text="Hãy nhập tên hoa quả:" />
    <AutoCompleteTextView
        android:id="@+id/autocomplete"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="30dp"
        android:completionThreshold="2"/>
    <TextView
        android:id="@+id/field"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="30dp"
        android:text="" />
</LinearLayout>
```

File AutoComplete.java chứa nội dung như sau:

```
public class AutoComplete extends Activity implements TextWatcher {
    private TextView field;
    private AutoCompleteTextView autocomplete;
    String autocompleteItems [] = {"apple", "banana", "mango",
        "pineapple", "apricot",
        "orange", "pear", "grapes"};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        field = (TextView) findViewById(R.id.field);
        autocomplete = (AutoCompleteTextView)findViewById(R.id.autocomplete);
        autocomplete.addTextChangedListener(this);
        autocomplete.setAdapter(new ArrayAdapter<String>(this,
            android.R.layout.simple_dropdown_item_1line, autocompleteItems));
    }
    @Override
    public void onTextChanged(CharSequence arg0, int arg1, int arg2, int arg3) {
        field.setText(autocomplete.getText());
    }
}
```

```

public void beforeTextChanged(CharSequence s, int start,int count, int
    after) { }
public void afterTextChanged(Editable s) { }
}

```

2.3.9 Đối tượng ListView

Đây là một trong những thành phần quan trọng nhất và được sử dụng đến 80 % trong các ứng dụng Android.



Hình 2.13: Minh họa đối tượng ListView

Đối tượng ListView cho phép thiết kế ứng dụng với danh sách cuộn các phần tử (item). Đối tượng có chức năng tương tự ListView là đối tượng ListActivity. Tuỳ theo sở thích, người lập trình có thể sử dụng một trong hai đối tượng trên cho việc xây dựng ứng dụng của mình.

Đối tượng ListView được khai báo trong file xml dạng như sau:

```

<ListView
    android:id="@+id/list"

```

```
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"/> >
```

Khi sử dụng đối tượng ListActivity người lập trình thường kế thừa lớp cha ListActivity thay cho Activity, đồng thời viết đè phương thức onListItemClick. Phương thức này được thực thi khi người dùng lựa chọn một item trên ListView. Dữ liệu của ListView là một danh sách các item. Thông thường danh sách các phần tử được tạo từ một đối tượng ArrayAdapter, sau đó được gắn vào ListActivity thông qua phương thức setListAdapter.

Thí dụ 2-14. Sử dụng đối tượng ListActivity File main.xml khai báo giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <TextView
        android:id="@+id/selection"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>
    <ListView
        android:id="@+android:id/list" //trong một số trường hợp dùng cách khai báo
        ID này
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:drawSelectorOnTop="false" />
</LinearLayout>
```

File ListViewDemo.java được viết như sau:

```
public class ListViewDemo extends ListActivity {
    TextView selection;
    String[] items={"Vật lý", "Hoá học", "Toán Cao cấp", "Anh văn chuyên
    ngành", "Nhập môn lập trình", "Lập trình nâng cao", "Lập trình Web", "Thiết
    kế đồ họa", "Lập trình mạng", "Trí tuệ nhân tạo", "Khai phá dữ liệu", "Lập
    trình ứng dụng Android", "Mạng máy tính", "Kỹ thuật truyền tin", "Kiến trúc
    máy tính", "Cơ sở dữ liệu", "Công nghệ và thiết bị mạng", "Virus máy
    tính"};
    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
```

```

        setContentView(R.layout.main);
    //lấy dữ liệu cho ListActivity thông qua đối tượng ArrayAdapter
        setListAdapter (new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1, items));
        selection=(TextView)findViewById (R.id.selection);
    }
//viết đè phương thức onListItemClick
public void onListItemClick (ListView parent, View v, int position, long id) {
//hiển thị item được chọn lên TextView
    selection.setText(items[position]);
}
}

```

Kết quả thí dụ 2-14 như hình 2.13.

Người lập trình có thể tạo một đối tượng ListView với mỗi dòng chứa các đối tượng khác nhau, chẳng hạn như ảnh hoặc các xâu kí tự để tạo nên một giao diện đẹp và linh hoạt thông qua việc sử dụng một trong các kiểu của đối tượng ListAdapter là BaseAdapter (android.widget.BaseAdapter).

Thí dụ 2-15. Thí dụ này sẽ trình bày cách sử dụng đối tượng ListView cơ bản nhất.

File main.xml chứa đối tượng ListView được định nghĩa như sau:

```

<ListView
    android:id="@+id/list"
    android:layout_width="wrap_content"
    android:layout_height="fill_parent" />

```

Phương thức MainActivity.onCreate() sẽ chứa đoạn mã như sau:

```

ListView listView = (ListView) findViewById(R.id.list);
listView.setAdapter(new BaseAdapter(){
    public int getCount() {
        return 0;
    }
    public Object getItem(int position) {
        return null;
    }
    public long getItemId(int position) {
        return 0;
    }
    public View getView(int position, View convertView, ViewGroup parent) {

```

```
        return null;
    });
}
```

Phương thức getCount() trả về số lượng các item có trong ListView. Phương thức getItem() và getItemId() trả về giá trị và định danh (id) của một item đang được chọn trong ListView trong trường hợp mỗi item chỉ chứa một đối tượng duy nhất.

Trong thí dụ này mỗi item của ListView được định nghĩa trong một file XML có tên list_row.xml gồm một TextView như sau:

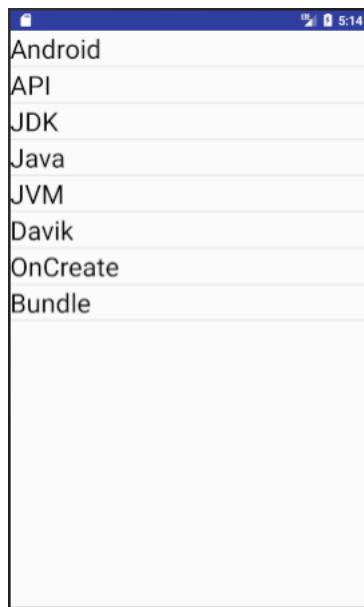
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView android:text="@+id/TextView01"
        android:id="@+id/TextView01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Trong lớp MainActivity khai báo mảng hằng xâu ký tự cung cấp dữ liệu cho ListView: String[] words = "Android", "API", "JDK";

Với các ListView có chứa nhiều đối tượng trong một dòng thì mỗi dòng sẽ tương ứng với một View, đối tượng ListView sẽ được khởi tạo như sau:

```
listView.setAdapter(new BaseAdapter(){
    public int getCount() {
        return words.length;
    }
    public Object getItem(int position) {
        return words[position];
    }
    public long getItemId(int position) {
        return position;
    }
    public View getView(int position, View convertView, ViewGroup parent) {
        LayoutInflator inflater = (LayoutInflator)
            getSystemService(LAYOUT_INFLATER_SERVICE);
        View view = inflater.inflate(R.layout.list_row, null);
    }
});
```

```
TextView textView = (TextView) view.findViewById(R.id.TextView01);
textView.setText(words[position]);
return view;
});
```



Hình 2.14: Kết quả thí dụ 2-15

Kết quả thí dụ 2-15 như hình 2.14.

Trong trường hợp mỗi item của ListView chứa nhiều đối tượng được định nghĩa trong một file XML thì phương thức `getView()` cho phép lấy về giá trị của mỗi item và đặt trong một đối tượng View.

Các phương thức `getCount()`, `getItem()`, `getItemId()` được tính toán các giá trị thông qua thành phần lưu trữ dữ liệu của ListView. Phương thức `getView()` sử dụng đối tượng `LayoutInflater` để lấy đối tượng View trong mỗi dòng của ListView.

Thí dụ 2-16 .Thí dụ về ListView với mỗi phần tử của ListView là một đối tượng View chứa hai đối tượng con có kiểu Image và TextView.

```

public class AdvancedListViewActivity extends ListActivity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //lấy ngữ cảnh (context) ứng dụng
        Context ctx = getApplicationContext();
        //lấy về đối tượng thư mục chứa các tài nguyên ứng dụng
        Resources res = ctx.getResources();
        String[] options = res.getStringArray(R.array.country_names);
        TypedArray icons = res.obtainTypedArray(R.array.country_icons);
        //Tạo nội dung cho ListView, lớp ImageAndListAdapter được định nghĩa trong
        //file riêng
        setListAdapter(new ImageAndListAdapter(ctx, R.layout.main_list_item, options,
            icons));
    }
}

```

Phương thức onCreate() ở trên khai báo hai mảng, mảng options chứa danh sách các hằng xâu kí tự là tên các quốc gia, mảng icons chứa danh sách các icon là biểu tượng lá cờ của mỗi quốc gia. Các hằng xâu và các icon này được định nghĩa trong file countries.xml như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="country_names">
        <item>Bhutan</item>
        <item>Colombia</item>
        <item>Italy</item>
        <item>Jamaica</item>
        <item>Kazakhstan</item>
        <item>Kenya</item>
    </string-array>
    <array name="country_icons">
        <item>@drawable/bhutan</item>
        <item>@drawable/colombia</item>
        <item>@drawable/italy</item>
        <item>@drawable/jamaica</item>
        <item>@drawable/kazakhstan</item>
        <item>@drawable/kenya</item>
    </array>
</resources>

```

Danh sách các ảnh làm icon phải được đặt trong thư mục res/draw-

able. Một đối tượng Adapter sẽ thực hiện chức năng cầu nối giữa một đối tượng AdapterView và dữ liệu nhìn thấy trong ListView, đồng thời nó cung cấp cách thức truy cập tới đối tượng View ứng với mỗi item trong ListView này.

ArrayAdapter là một lớp con của lớp Adapter, đối tượng này sẽ được sử dụng để truy cập đến đối tượng View trong ListView.

Tạo tệp ImageAndTextAdapter.java với nội dung như sau:

```
public class ImageAndTextAdapter extends ArrayAdapter<String> {
    private LayoutInflater mInflater;
    private String[] mStrings; // Lưu trữ mảng tên các quốc gia
    private TypedArray mIcons; // Lưu trữ tên các ảnh icon
    // mViewResourceId chứa ID của của đối tượng View được chọn trong ListView
    private int mViewResourceId;
    public ImageAndTextAdapter(Context ctx, int viewResourceId, String[] strings,
        TypedArray icons) {
        super(ctx, viewResourceId, strings);
        mInflater = (LayoutInflater)ctx.getSystemService(
            Context.LAYOUT_INFLATER_SERVICE);
        mStrings = strings;
        mIcons = icons;
        mViewResourceId = viewResourceId;
    }
    @Override
    public int getCount() {
        return mStrings.length;
    }
    @Override
    public String getItem(int position) {
        return mStrings[position];
    }
    @Override
    public long getItemId(int position) {
        return 0;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        convertView = mInflater.inflate(mViewResourceId, null);
        // Lấy về đối tượng ImageView trong dòng ListView
        ImageView iv = (ImageView)convertView.findViewById(R.id.option_icon);
        iv.setImageDrawable(mIcons.getDrawable(position));
        // Lấy về đối tượng TextView trong dòng ListView
        TextView tv = (TextView)convertView.findViewById(R.id.option_text);
        tv.setText(mStrings[position]);
        return convertView;
    }
}
```

Kết quả thí dụ 2-16 như hình 2.15.



Hình 2.15: Kết quả thí dụ 2-16

2.4 Các đối tượng Layout

Các đối tượng Layout được sử dụng để tổ chức các đối tượng widget trong giao diện theo các cách khác nhau (thường gọi là các containers). Một file XML layout thường sử dụng một trong các đối tượng Layout làm đối tượng cha của các đối tượng widget và layout khác trong giao diện. Android có năm đối tượng layout là FrameLayout, LinearLayout, RelativeLayout, TableLayout và AbsoluteLayout.

Đối tượng FrameLayout cho phép giới hạn vị trí một đối tượng trên màn hình làm cho giao diện ứng dụng vẫn giữ được tổ chức khi kích thước màn hình thay đổi. Đối tượng LinearLayout cho phép tổ chức các đối tượng widget con của nó lần lượt theo chiều dọc hoặc theo chiều ngang, đồng thời chúng được căn theo lề trái, lề phải

hoặc căn giữa,... Đối tượng RelativeLayout cho phép cho phép tổ chức các đối tượng con dựa trên mối quan hệ của chúng trong giao diện. Đối tượng TableLayout cho phép tổ chức các đối tượng con theo dạng bảng, mỗi đối tượng widget sẽ được đặt trong một ô. Đối tượng AbsoluteLayout cho phép cố định vị trí các widget trên giao diện. Phần dưới đây sẽ trình bày cách sử dụng một số đối tượng layout phổ biến trong Android.

2.4.1 Đối tượng FrameLayout

Như đã giới thiệu ở phần trên, đối tượng FrameLayout giới hạn phạm vi của một đối tượng trên giao diện khi kích thước màn hình thay đổi. Một số thuộc tính cơ bản đối tượng FrameLayout gồm:

id: định danh của đối tượng *foreground*: màu sắc của đối tượng khi đưa con trỏ qua *foregroundGravity*: căn các đối tượng con, có thể nhận các giá trị như top, bottom, left, right, center, cent_vertical, center_horizontal,... *Thí dụ*: Tạo một file XML layout activity_main.xml với đối tượng FrameLayout như sau:

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <ImageView
        android:src="@drawable/ic_launcher"
        android:scaleType="fitCenter"
        android:layout_height="250px"
        android:layout_width="250px"/>
    <TextView
        android:text="Frame Demo"
        android:textSize="30px"
        android:textStyle="bold"
        android:layout_height="fill_parent"
        android:layout_width="fill_parent"
        android:gravity="center"/>
</FrameLayout>
```

File MainActivity.java có nội dung:

```
import android.os.Bundle;
import android.app.Activity;
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Trong file res/values/strings.xml định nghĩa tên ứng dụng:

```
<string name="app_name">FrameLayout Demo</string>
```

2.4.2 Đối tượng LinearLayout

LinearLayout là một mô hình cho phép sắp xếp các đối tượng widget hoặc các container con theo dòng hoặc theo cột. Dưới đây sẽ trình bày các khái niệm cơ bản và thuộc tính của đối tượng LinearLayout.

Orientation: xác định các đối tượng con trong LinearLayout được trình bày theo dòng hoặc theo cột và được chỉ định qua thuộc tính android:orientation của nó trong file XML. Trong chương trình java có thể thay đổi giá trị thuộc tính này thông qua phương thức setOrientation() với 2 giá trị là HORIZONTAL và VERTICAL.

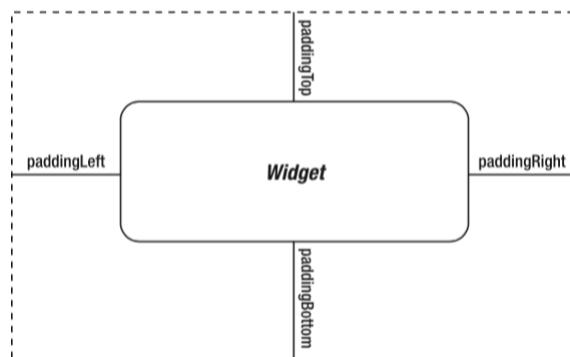
Fill Model: Tất cả các đối tượng widget đều phải được cung cấp thuộc tính chiều rộng (android:layout_width) và chiều cao (android:layout_height), chúng có thể nhận một trong các giá trị sau:

- Chỉ định giá trị cụ thể theo pixel (ví dụ: 125px)
- wrap_content: đối tượng sẽ điền đầy không gian tự nhiên còn trống trên màn hình.
- fill_parent: đối tượng sẽ được điền kín trong không gian của đối tượng cha (một container) chứa nó.

Weight: chỉ khoảng cách giữa các đối tượng widget con và được thiết lập thông qua thuộc tính android:layout_weight.

Gravity: cách căn các đối tượng con theo lề trái, lề phải, ở giữa hay kết hợp giữa các kiểu, nó được xác định qua thuộc tính android:layout_gravity hoặc sử dụng phương thức setGravity(). Khi căn một cột nó thường nhận một trong các giá trị: left, center_horizontal , right, khi căn một dòng nó thường nhận giá trị center_vertical.

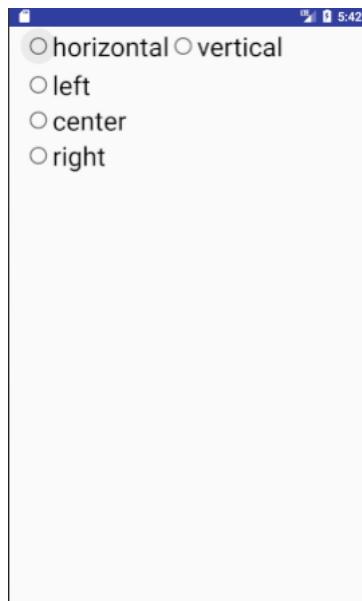
Padding: Cho phép thiết lập khoảng trống giữa các đối tượng widget qua thuộc tính android:padding hoặc sử dụng phương thức setPadding(). Thuộc tính này cho phép thiết lập khoảng trống cả bốn phía, để thiết lập riêng theo từng hướng có thể sử dụng các thuộc tính android:paddingLeft, android:paddingRight, android:paddingTop và android:paddingBottom.



Hình 2.16: Mô hình minh họa thuộc tính Padding của đối tượng widget

Thí dụ 2-17. Thiết kế giao diện sử dụng đối tượng LinearLayout chứa năm đối tượng Radio Button được nhóm thành hai nhóm như hình 2.17.

RadioGroup thứ nhất chứa hai Radio Button có nhãn horizontal và vertical, RadioGroup thứ hai chứa ba RadioButton là left, center và right. Khi chọn Radio Button có nhãn nào thì cho phép sắp xếp



Hình 2.17: Thí dụ đối tượng LinearLayout

hoặc căn các đối tượng widget trong nhóm theo chức năng tương ứng.

Phương thức onCreate() chứa đoạn mã như sau:

```

orientation=(RadioGroup)findViewById (R.id.orientation);
orientation.setOnCheckedChangeListener (this);
gravity=(RadioGroup)findViewById (R.id.gravity);
gravity.setOnCheckedChangeListener (this);

Phương thức viết đònCheckedChanged chứa nội dung như sau:
switch (checkedId) {
    case R.id.horizontal: //sắp xếp các con theo chiều ngang
        orientation.setOrientation(LinearLayout.HORIZONTAL); break;
    case R.id.vertical: //sắp xếp các con theo chiều dọc
        orientation.setOrientation(LinearLayout.VERTICAL); break;
    case R.id.left: //căn các con theo lề trái của cột
    gravity.setGravity (Gravity.LEFT); break;
    case R.id.center: //căn các con giữa cột
    gravity.setGravity (Gravity.CENTER_HORIZONTAL); break;
    case R.id.right: //căn các con theo lề phải của cột
    gravity.setGravity (Gravity.RIGHT); break;
}

```

2.4.3 Đối tượng RalativeLayout

Các đối tượng con được sắp đặt trong mối quan hệ của nó với đối tượng cha và các đối tượng con khác trong một RalativeLayout.

Các thuộc tính chỉ định mối quan hệ của đối tượng con với đối tượng container chứa nó.

android:layout_alignParentTop: căn chỉnh lề trên của các đối tượng widget với lề trên của đối tượng cha.

android:layout_alignParentBottom: căn chỉnh lề dưới của các đối tượng widget với lề dưới của đối tượng cha.

android:layout_alignParentLeft: căn chỉnh lề trái của các đối tượng widget với lề trái của đối tượng cha.

android:layout_alignParentRight: căn chỉnh lề phải của các đối tượng widget với lề phải của đối tượng cha.

android:layout_centerHorizontal: vị trí của đối tượng widget theo chiều ngang nằm ở trung tâm của đối tượng cha.

android:layout_centerVertical: vị trí của đối tượng widget theo chiều dọc nằm ở trung tâm của đối tượng cha.

android:layout_centerInParent: vị trí của đối tượng widget theo chiều ngang và chiều dọc nằm ở trung tâm của đối tượng cha.

Năm thuộc tính sau cho phép căn một đối tượng widget với các đối tượng widget khác trong cùng một container:

android:layout_alignTop: căn lề trên của đối tượng widget với lề trên của đối tượng được chỉ định trong thuộc tính.

android:layout_alignBottom: căn lề dưới của đối tượng widget với lề dưới của đối tượng được chỉ định trong thuộc tính.

android:layout_alignLeft: căn lề trái của đối tượng widget với lề trái của đối tượng được chỉ định trong thuộc tính.

android:layout_alignRight: căn lề phải của đối tượng widget

với lề phải của đối tượng được chỉ định trong thuộc tính.

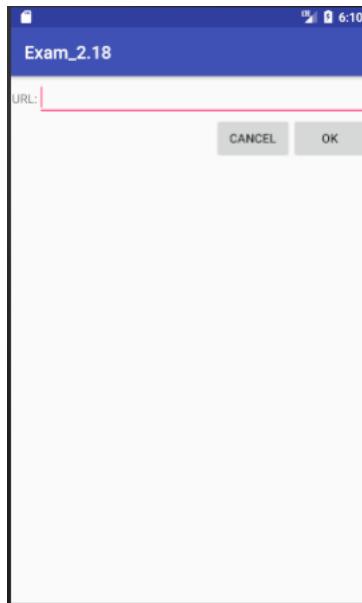
android:layout_alignBaseline: xác định đường căn cơ bản cho hai đối tượng widget được căn (các đường này không nhìn thấy bằng mắt). Thuộc tính này thường dùng để căn các nhãn và các đối tượng nhập liệu thẳng hàng. Nếu muốn một đối tượng widget này ở bên phải đối tượng widget kia thì sử dụng cặp thuộc tính như sau:

```
    android:layout_toRightOf = "@+id/label"
```

```
    android:layout_alignBaseline = "@+id/label"
```

Định danh của đối tượng widget được chỉ định trong thuộc tính sử dụng cú pháp "@+id/tên_ID".

Thí dụ 2-18. Thiết kế file xml để tạo giao diện như hình 2.18. Nội



Hình 2.18: Thí dụ đối tượng RalativeLayout

dụng file xml được thiết kế như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
```

```

    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="5px">
        <TextView android:id="@+id/label"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="URL:"
            android:layout_alignBaseline="@+id/entry"
            android:layout_alignParentLeft="true"/>
        <EditText
            android:id="@+id/entry"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@+id/label"
            android:layout_alignParentTop="true"/>
        <Button
            android:id="@+id	ok"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/entry"
            android:layout_alignRight="@+id/entry"
            android:text="OK" />
        <Button
            android:id="@+id/cancel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toLeftOf="@+id/ok"
            android:layout_alignTop="@+id/ok"
            android:text="Cancel" />
    </RelativeLayout>

```

2.4.4 Đối tượng TableLayout

Đối tượng TableLayout cho phép tổ chức các đối tượng widget theo dạng bảng chứa các dòng và các cột. Một số thuộc tính của thường quan tâm của đối tượng này như sau:

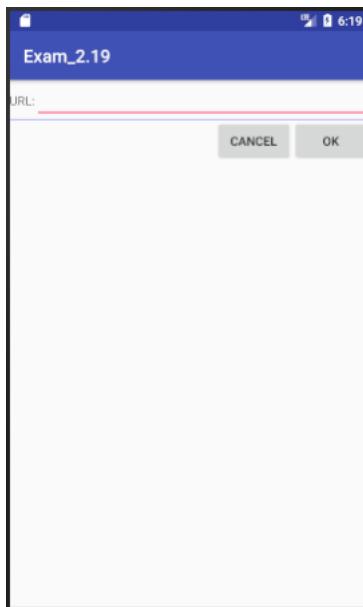
android:stretchColumns: chỉ định độ giãn giữa các cột trong bảng, có thể là một giá trị số nguyên hoặc danh sách các cột được phân cách nhau bởi dấu phẩy. Thuộc tính này giúp căn các cột trong bảng hết không gian còn trống của dòng giúp cho nội dung trong bảng đẹp hơn.

android:shrinkColumns: nhận giá trị là một số nguyên hoặc danh

sách các cột được phân cách nhau bởi dấu phẩy. Các cột được liệt kê trong thuộc tính này có độ rộng vừa với kích thước của nội dung bên trong. Thuộc tính này có thể làm cho một số cột bị đẩy ra ngoài lề bên phải màn hình.

android:collapseColumns: Cho phép ẩn thông tin một số cột trong bảng, nhận giá trị là một số nguyên hoặc danh sách các cột được phân cách nhau bởi dấu phẩy

Thí dụ 2-19. Thiết kế file xml để tạo giao diện như hình 2.19: Nội dung file xml được thiết kế như sau:



Hình 2.19: Thí dụ đối tượng TableLayout

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="1">
    <TableRow>
        <TextView
            android:text="URL:" />
        <EditText android:id="@+id/entry" />
```

```

        android:layout_span="3"/>
    
```

```

</TableRow>
<View
    android:layout_height="2px"
    android:background="#0000FF" />

```

```

<TableRow>
    <Button android:id="@+id/cancel"
        android:layout_column="2"
        android:text="Cancel" />
    <Button android:id="@+id/ok"
        android:text="OK" />

```

```

</TableRow>
</TableLayout>

```

2.5 Đối tượng menu

Trong các ứng dụng Android thường sử dụng hai loại menu là menu tùy chọn (menu option) và menu ngữ cảnh (menu context).

Các menu option được hiện lên khi người dùng nhấn nút menu trên thiết bị. Các menu ngữ cảnh được hiện lên khi người dùng nhấn và giữ trên một đối tượng widget cụ thể.

Các menu được định nghĩa trong các file XML được đặt trong thư mục res/menu. Cách định nghĩa hai loại menu trên trong file XML là giống nhau và tuân theo cú pháp sau:

```

<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/icon1"
        android:title="One"
        android:icon="@drawable/first" />

```

Mỗi menu thường gồm ba thuộc tính định danh (ID), tiêu đề (title) và biểu tượng (Icon). Các icon là các file ảnh được đặt trong thư mục res/drawable.

Một menu có thể chứa các menu con bên trong nó và được định nghĩa trong file XML theo cú pháp bên dưới. Menu có tiêu đề Submenu chứa hai menu con là Add, Delete.

```

<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id_submenu"
        android:orderInCategory="0" //là một số nguyên chỉ thứ tự trong mục menu
        android:title="Submenu">
        <menu>
            <item android:id="@+id/add"
                android:title="Add"
                android:visible="true"
                android:alphabeticShortcut="n"/> //phím tắt truy cập menu
            <item android:id="@+id/delete"
                android:title="Delete"
                android:visible="false" //ẩn menu với người dùng
                android:alphabeticShortcut="g" />
        </menu>
    </item>
</menu>

```

Ngoài ra, các menu có thể được nhóm theo từng nhóm sử dụng cặp thẻ `<group>...</group>`.

2.5.1 Các menu option

Khi người dùng nhấn nút menu trên thiết bị thì các menu hiển thị chế độ icon, nếu có nhiều menu thì nút "more" sẽ được hiển thị để cho phép chuyển đến menu tiếp theo.

Các menu được gắn vào ứng dụng Android thông qua việc viết đè phương thức `onCreateOptionsMenu()` trong chương trình java và chỉ định file XML chứa khai báo về menu theo cú pháp sau:

```

\hspace*{1cm}@Override \\
\hspace*{1cm}public boolean onCreateOptionsMenu(Menu menu) { \\
\hspace*{1.5cm}MenuInflater inflater = getMenuInflater(); \\
\hspace*{1.5cm}inflater.inflate(R.menu.menumain, menu); //tham số menumain là tên \\
    file menu.xml\\ \\
\hspace*{1.5cm}return true; \\
}

```

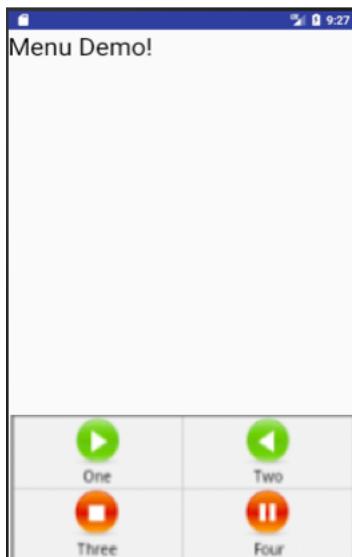
Để bắt sự kiện khi người dùng chọn một trong các menu option thì phải viết đè phương thức `onOptionsItemSelected()` bên dưới và sử dụng câu lệnh switch để xác định menu được chọn như sau:

```

@Override
public boolean onOptionsItemSelected(MenuItem item) {
//Nếu menu được chọn thì hiển thị thông điệp cảnh báo dạng TOAST lên màn hình
switch (item.getItemId()) {
case R.id.icon1:
    Toast.makeText(this, "Menu 1 được chọn", Toast.LENGTH_LONG).show(); break;
case R.id.icon2:
    Toast.makeText(this, "Menu 2 được chọn!", Toast.LENGTH_LONG).show(); break;
...
}
return true;

```

Thí dụ 2-20. Thiết kế giao diện gồm bốn menu như hình 2.20. Mỗi khi menu được chọn thì hiển thị thông điệp dạng TOAST tương ứng. File *res/menu.xml* chứa các menu như sau:



Hình 2.20: Thí dụ về menu option

```

<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/icon1"
          android:title="One"
          android:icon="@drawable/first" />
    <item android:id="@+id/icon2"
          android:title="Two"
          android:icon="@drawable/second" />
    <item android:id="@+id/icon3"
          android:title="Three"
          android:icon="@drawable/third" />
    <item android:id="@+id/icon4"
          android:title="Four"
          android:icon="@drawable/fourth" />
</menu>

```

```
    android:icon="@drawable/three" />
<item android:id="@+id/icon4"
    android:title="Four"
    android:icon="@drawable/four" />
</menu>
```

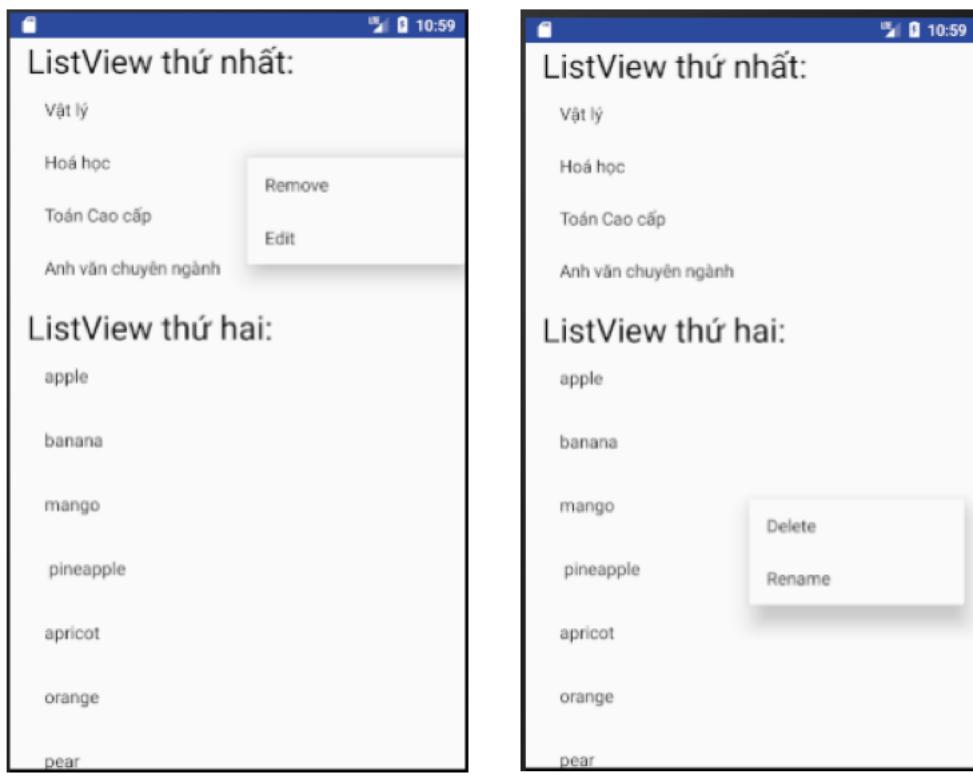
File MenuDemo.java chứa mã chương trình như sau:

```
public class MenuDemo extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.icon1:
                Toast.makeText(this, "Menu one", Toast.LENGTH_LONG).show();
                break;
            case R.id.icon2:
                Toast.makeText(this, "Menu two", Toast.LENGTH_LONG).show();
                break;
            case R.id.icon3:
                Toast.makeText(this, "Menu three", Toast.LENGTH_LONG).show();
                break;
            case R.id.icon4 :
                Toast.makeText(this, "Menu four", Toast.LENGTH_LONG).show();
                break;
        }
        return true;
    }
}
```

2.5.2 Các menu context

Các menu ngữ cảnh (menu context) được nới lên khi người dùng nhấn và giữ trên trên đối tượng widget đăng ký sử dụng menu context. Phương thức onCreateOptionsMenu() được gọi mỗi khi một context menu được yêu cầu. Không giống như các menu tùy

chọn, các menu context sẽ bị hủy bỏ ngay sau khi chúng được sử dụng xong.



Hình 2.21: Minh họa về menu context

Không như các menu option, các menu context được gắn vào từng đối tượng widget cụ thể bằng cách gọi phương thức registerForContextMenu() trong hàm onCreate() của lớp chính. Chẳng hạn, người lập trình đăng ký menu context cho đối tượng ListView như sau:

```
mListView1 = (ListView) findViewById(R.id.list1);
registerForContextMenu(mListView1);
```

Để gắn các menu context được định nghĩa trong một file XML vào một ứng dụng Android, người lập trình phải viết đè phương

thức onContextMenu() theo cú pháp như sau:

```
@Override
    public void onContextMenu(ContextMenu menu, View v,
    ContextMenu.ContextMenuItemInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        if (v.getId()==R.id.listView1){//nhấn và giữ trên ListView 1
            MenuInflater inflater = getMenuInflater();
            //lấy danh sách menu được định nghĩa trong file menu_listview_1.xml trong
            //thư mục res/menu
            inflater.inflate(R.menu.menu_listview_1, menu);
        }
    }
```

Mỗi đối tượng widget có thể được gắn các menu context khác nhau và chúng được định nghĩa trong các file XML riêng biệt. Thông qua đối tượng View là tham số của hàm giúp người lập trình xác định được đối tượng widget nào được chọn qua ID của nó. Khi một menu context được chọn thì phương thức onContextItemSelected() sẽ được thực thi. Người lập trình phải viết để phương thức này theo cú pháp như sau:

```
public boolean onContextItemSelected(MenuItem item) {
    switch(item.getItemId()){ //lấy id của đối tượng widget được chọn
        case R.id.remove: //nếu đối tượng widget có id là remove được chọn
            //thực hiện công việc x
            return true;
        case R.id.edit:
            //thực hiện công việc y
            return true;
        ...
        default:
            return super.onContextItemSelected(item);
    }
}
```

Thí dụ 2-21. Tạo hai menu context khác nhau cho hai đối tượng ListView. File *activity_main.xml* chứa giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```

    android:layout_height="fill_parent"
    xmlns:tools="http://schemas.android.com/tools"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical"
    tools:context="com.example.admin.exam_221.MainActivity">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:textSize="30dp"
        android:text="ListView thứ nhất:"/>
    <ListView
        android:id="@+id/list1"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        android:divider="@android:color/transparent"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:textSize="30dp"
        android:text="ListView thứ hai:"/>
    <ListView
        android:id="@+id/list2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/list1"
        android:dividerHeight="10dp"
        android:divider="@android:color/transparent"/>
</LinearLayout>

```

File *menu_listview_1.xml* chứa hai menu context Remove và Edit cho Listview thứ nhất:

```

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/remove"
          android:title="Remove" />
    <item android:id="@+id/edit"
          android:title="Edit" />
</menu>

```

File *menu_listview_2.xml* chứa hai menu context Delete và Rename cho Listview thứ hai:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/delete"
        android:title="Delete"
        app:showAsAction="ifRoom|withText"/>
    <item android:id="@+id/rename"
        android:title="Rename"
        app:showAsAction="ifRoom|withText"/>
</menu>
```

File *MainActivity.java* chứa mã chương trình như sau:

```
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ListView mListview1=(ListView)findViewById(R.id.list1);
        ListView mListview2=(ListView)findViewById(R.id.list2);
        String[] items1={"Vật lý", "Hoá học", "Toán Cao cấp", "Anh văn chuyên
                        ngành", "Nhập môn lập trình"};
        String[] items2={"apple", "banana", "mango", "pineapple", "apricot",
                        "orange", "pear", "grapes"};
        mListview1.setAdapter(new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1,items1));
        mListview2.setAdapter(new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1,items2));
        //Đăng ký menu context cho hai ListView
        registerForContextMenu(mListview1);
        registerForContextMenu(mListview2);
    }
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenu.ContextMenuItemInfo menuInfo){
        super.onCreateContextMenu(menu,v,menuInfo);
        //Gắn menu context cho ListView thứ nhất
        if(v.getId()==R.id.list1){
            MenuInflater inflater=getMenuInflater();
            inflater.inflate(R.menu.menu_listview_1,menu);
        }
        //Gắn menu context cho ListView thứ hai
        if(v.getId()==R.id.list2){//For second listview
            MenuInflater inflater=getMenuInflater();
            inflater.inflate(R.menu.menu_listview_2,menu);
        }
    }
    //Xử lý sự kiện tương ứng với từng menu context mỗi khi chúng được chọn
    @Override
    public boolean onContextItemSelected(MenuItem item){
```

```

        AdapterView.AdapterContextMenuInfo
        info=(AdapterView.AdapterContextMenuInfo)item.getMenuInfo();
        int index=info.position;
        View view=info.targetView;
        switch(item.getItemId()){
            case R.id.remove:
                Toast.makeText(this, "Loại bỏ item", Toast.LENGTH_LONG).show();
                return true;
            case R.id.edit:
                Toast.makeText(this, "Sửa item", Toast.LENGTH_LONG).show();
                return true;
            case R.id.delete:
                Toast.makeText(this, "Xoá item", Toast.LENGTH_LONG).show();
                return true;
            case R.id.rename:
                Toast.makeText(this, "Đổi tên item", Toast.LENGTH_LONG).show();
                return true;
            default:
                return super.onContextItemSelected(item);
        }
    }
}

```

Kết quả thí dụ 2-12 như hình 2.22 (a) và 2.22 (b)

2.6 Đối tượng hộp thoại

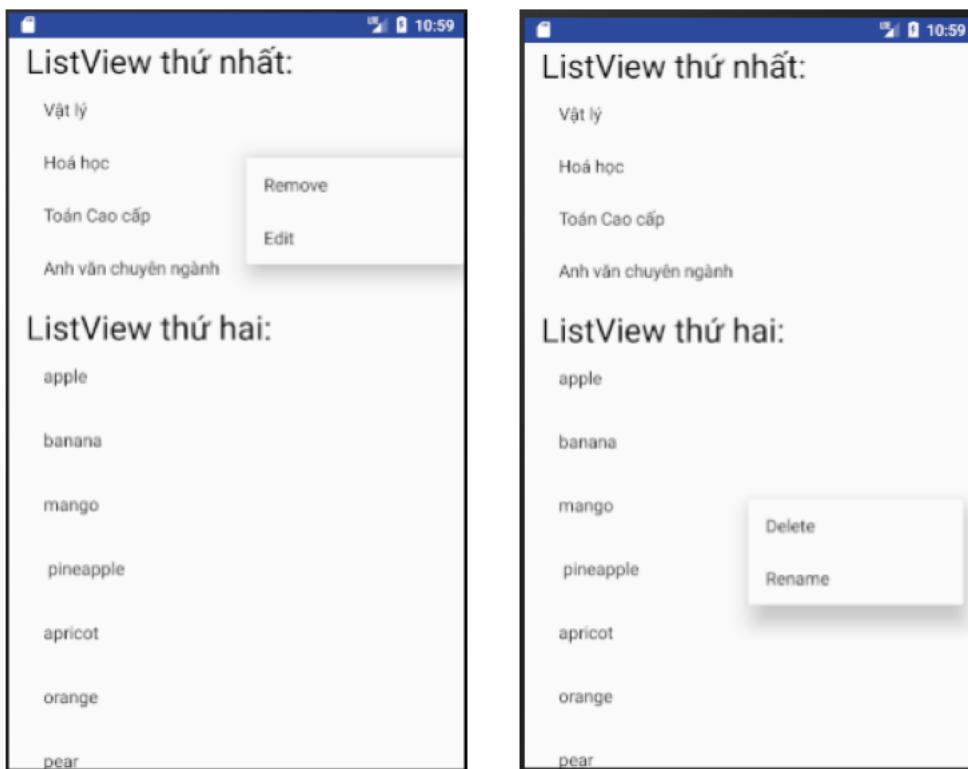
Giả sử người dùng đang thực hiện cập nhật dữ liệu trên giao diện ứng dụng thì tắt Activity, thông thường sẽ có một hộp thoại cảnh báo hiện ra để nhắc nhớ họ. Người dùng sẽ lựa chọn một trong các hành động trên hộp thoại là đồng ý(save/ok), không đồng ý(discard) hoặc tắt hộp thoại (cancel). Lớp AlertDialog cung cấp cho người dùng ba hành động tương ứng với các tùy chọn trên là Positive, Neutral và Cancel.

Để tạo hộp thoại người lập trình sử dụng cú pháp như sau:

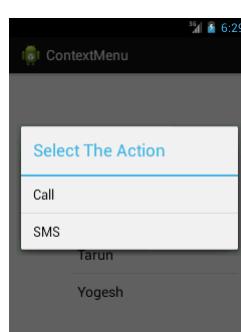
```

AlertDialog = new AlertDialog.Builder(this)
    .setTitle("Tiêu đề hộp thoại").setMessage("Thông điệp chuyển đến người dùng ")
    //nhan thường là save hoặc OK...
    setPositiveButton("Nhấn", new AlertDialog.OnClickListener(){

```



Hình 2.22: Kết quả thí dụ 2-21 về tạo menu context



Hình 2.23: Minh họa về hộp thoại cảnh báo

```

public void onClick(DialogInterface dialog, int which) {
    //Các lệnh thực hiện khi người dùng chọn nút Save hoặc OK, ...
}
//nhan thường là Discard hoặc Tù bỏ, ...
.setNeutralButton("Nhân", new AlertDialog.OnClickListener() {
    public void onClick(DialogInterface dialog, int which) {

```

```

    // các lệnh thực hiện khi người dùng chọn nút từ bỏ
}}
//nhan thường là Cancel, No,..
.setNegativeButton("Nhấn", new AlertDialog.OnClickListener() {
    public void onClick(DialogInterface dialog, int which) {
        alertDialog.cancel(); //tắt hộp thoại
}}).create();

```

Thí dụ 2-22. Trình bày thí dụ về hộp thoại cảnh báo dạng Alert như hình 2.23. File *MainActivity.java* chứa mã như sau:



Hình 2.24: Kết quả thí dụ 2-22

```

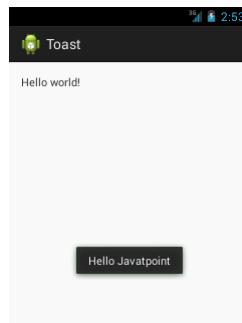
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setMessage("Bạn chắc chắn muốn đóng ứng dụng?")
            .setCancelable(false)
            ..setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    finish();
                }
            })
            .setNegativeButton("No", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dialog.cancel();
                }
            });
    }
}

```

```
        }
    });
    AlertDialog alert = builder.create();
    alert.setTitle("AlertDialogExample");
    alert.show();
    setContentView(R.layout.activity_main);
}
}
```

2.7 Thông điệp cảnh báo dạng Toast

Thông điệp Toast là loại thông điệp có thể xuất hiện đột ngột hoặc biến mất mà không cần sự tương tác của người dùng. Thí dụ thông điệp Toast cảnh báo pin ở mức thấp, cảnh báo một nhiệm vụ nào đó đã hoàn thành,



Hình 2.25: Minh họa thông điệp Tost

Dễ tạo ra một thông điệp Toast rất đơn giản vì lớp Toast cung cấp phương thức `makeText()` với tham số String hoặc ID và trả về một thẻ hiện (instance) của lớp Toast. Phương thức `makeText()` cung cấp độ trễ thông điệp được hiển thị trên giao diện ứng dụng bằng hai hằng số `LENGTH_SHORT` hoặc `LENGTH_LONG`.
Thí dụ, tạo một thông điệp Toast như sau:

```
Toast.makeText(getApplicationContext(),"Hello      Android",
Toast.LENGTH_SHORT).show();
```

Câu hỏi và bài tập

1. Vai trò của đối tượng View trong Android?
2. Có bao nhiêu cách cài đặt sự kiện onClick cho một đối tượng Button trong Android và sự khác nhau giữa các cách cài đặt đó?
3. Nêu sự khác biệt giữa việc sử dụng các đối tượng CheckBox và Radio Button?
4. Sự khác biệt giữa đối tượng EditText và đối tượng AutoCompleteTextView?
5. Cho biết sự khác biệt giữa menu option và menu context?
6. Nêu sự khác biệt của các cách tổ chức giao diện sử dụng các đối tượng Layout trong Android?
7. Thiết kế ứng dụng như Hình 2.26. Lập trình để khi nhấn nút OK thì hiển thị các giá trị đã được chọn của Radio và các CheckBox lên TextView.



Hình 2.26: Bài tập đối tượng CheckBox và Radio Button.

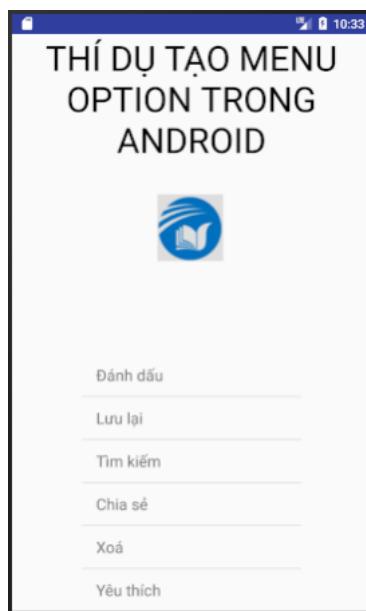
8. Thiết kế giao diện và lập trình để tạo danh sách tên các quốc

gia và từ viết tắt của chúng như Hình 2.27.

Australia	au
Austria	at
Belgium	be
Brazil	br
Canada	ca
China	cn
Denmark	dk
France	fr
Germany	de
Hong Kong	hk
India	in
Indonesia	id
Italy	it
Korea	kr
Netherlands	nl
Norway	no

Hình 2.27: Bài tập đối tượng ListView

9.Thiết kế giao diện gồm các menu option như Hình 2.28. Lập trình để mỗi khi chọn một menu option màn hình sẽ hiện thông điệp dạng Toast chứa tên menu được chọn.



Hình 2.28: Bài tập đối tượng ListView

Chương 3

Lập trình với các thành phần cơ bản: Activity, Intent, Content Provider và Service

3.1	Chu trình của một ứng dụng Android	107
3.2	Đối tượng Intent và Intent Filter	112
3.3	Đối tượng quảng bá thông điệp Broadcast Reciever	121
3.4	Đối tượng Content Provider	126
3.5	Các dịch vụ (Services)	134

Chương này nhằm cung cấp cho người đọc những kiến thức và kỹ năng lập trình về:

- Đối tượng Activity và chu trình của một ứng dụng Android
- Đối tượng Intent và Intent Filter, truyền thông điệp giữa các Activity sử dụng với đối tượng Intent
- Đối tượng quảng bá thông điệp Broadcast Reciever
- Đối tượng chia sẻ dữ liệu Content Provider
- Các dịch vụ trong Android

3.1 Chu trình của một ứng dụng Android

Một ứng dụng Android được xây dựng từ một hoặc nhiều thành phần trong số các thành phần chính là các Activity, Service, Content Provider, Intent như đã nói ở phần đầu. Ngoài ra còn có hai thành phần nữa là Broadcast Receiver và Notification. Broadcast Receiver là một dạng Intent được gửi đến các ứng dụng Android và các ứng dụng có thể lắng nghe và phản hồi lại các thông điệp với các sự kiện phù hợp. Các Notification thường được sử dụng để phát tín hiệu cảnh báo đến người dùng và ngắt trạng thái của Activity hiện hành, chúng được phát ra từ một Service hoặc một Broadcast Receiver trong hệ thống. Thí dụ, khi có tin nhắn hoặc cuộc gọi đến thiết bị thì đèn flash sáng, có chuông reo, hiển thị icon hoặc các hộp thoại.

Các thành phần trên khi được sử dụng trong ứng dụng Android phải được khai báo trong thẻ file manifest. Một ứng dụng Android chỉ chứa duy nhất một thẻ application và nó là thẻ cha chứa tất cả các thẻ con khác như Activity, Service, Content Provider và Broadcast Receiver.

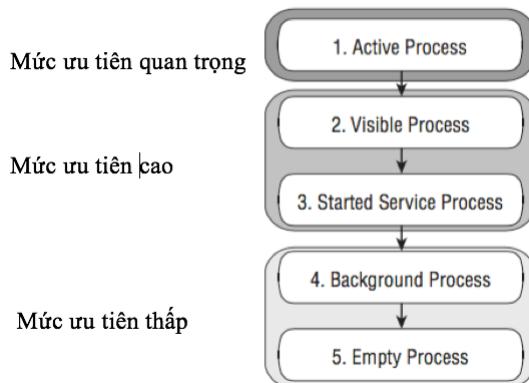
Cú pháp khai báo thẻ activity trong thẻ application như sau:

```
<application android:icon="@drawable/icon" android:theme="@style/my_theme">
    <activity android:name=".MyActivity"    android:label="@string/app_name">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    [ ... các thẻ khác... ]
</application>
```

Một ứng dụng Android có thể gồm nhiều Activity, mỗi Activity tương ứng với một cửa sổ hoặc một hộp thoại muốn được hiển thị tới người dùng. Mỗi Activity chứa một thẻ intent-filter để xác

định các Intent khởi chạy Activity.

Một thiết bị chạy hệ điều hành Android thường có nhiều ứng dụng chạy đồng thời, chúng được gán một độ ưu tiên nhất định. Khi hai tiến trình có cùng mức ưu tiên thì tiến trình nào đã từng ở mức ưu tiên thấp lâu hơn sẽ bị huỷ. Các tiến trình ở trạng thái kích hoạt (Active process) có mức ưu tiên cao nhất gọi là các mức ưu tiên quan trọng, các tiến trình ở trạng thái không nhìn thấy (Visible process) và tiến trình khởi tạo các dịch vụ (Started Service Process) ở mức ưu tiên cao, các tiến trình chạy ở trạng thái ẩn (Background Process) và tiến trình rỗng có mức ưu tiên thấp nhất (Empty Process).



Hình 3.1: Thứ tự ưu tiên của các ứng dụng trên hệ điều hành Android.

Các tiến trình Active thường gọi là các tiến trình ở trạng thái foreground, ở trạng thái này người dùng có thể tương tác với ứng dụng và chúng sẽ bị tắt sau khi sử dụng xong.

Các tiến trình Active bao gồm:

- Các Activity chạy ở trạng thái active để phản hồi lại các sự kiện của người dùng.
- Các Activity, Service, hoặc các Broadcast Receiver đang thực thi sự kiện onReceive()

- Các Services đang thực thi sự kiện onStart(), onCreate(), hoặc onDestroy().

Các tiến trình Visible thường là các Activity ở trạng thái bị che khuất bởi các Activity khác hoặc bị thu nhỏ, chúng được nhìn thấy nhưng không tương tác với người dùng.

Các tiến trình Started Service là các tiến trình khởi tạo các dịch vụ được xử lý liên tiếp, chúng không tương tác trực tiếp với người dùng và không có giao diện. Tiến trình vẫn được xem là ở trạng thái foreground và sẽ không bị huỷ cho đến khi các nguồn tài nguyên vẫn cần thiết cho các tiến trình Active và Visible.

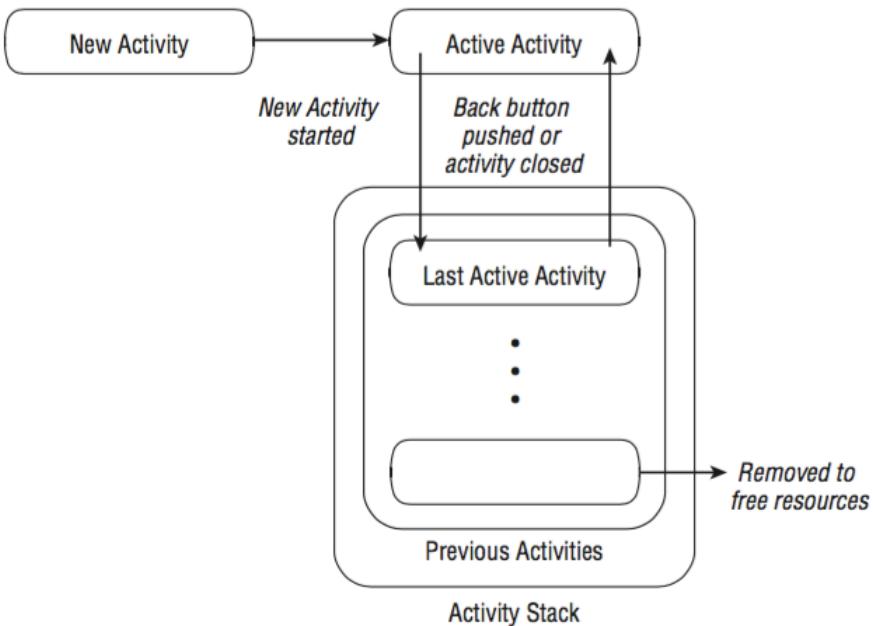
Các tiến trình Background là các Activities không được nhìn thấy và không có bất kỳ dịch vụ nào, chúng được khởi tạo là các tiến trình chạy ẩn bên trong hệ thống. Android sẽ huỷ các tiến trình này để thu hồi tài nguyên hệ thống cho các tiến trình foreground sử dụng.

Các tiến trình Empty Android vẫn lưu trữ các ứng dụng khi chúng đã kết thúc chu kỳ thời gian sống trong bộ nhớ cache để khởi chạy lại ở các lần gọi tiếp theo. Các tiến trình này gọi là tiến trình rỗng và sẽ bị huỷ khi được yêu cầu.

Các Activity trong hệ thống tồn tại ở các trạng thái khác nhau và chúng được xếp hàng trong ngăn xếp theo cơ chế vào sau ra trước (last in first out). Khi có một Activity mới được tạo thì nó được di chuyển đến đỉnh của ngăn xếp, nếu người dùng chuyển nó về trạng thái foreground thì Activity tiếp theo sẽ được chuyển lên và trở thành tiến trình Active.

Các trạng thái của một Activity bao gồm:

Trạng thái Active: là trạng thái Activity nhìn thấy được và đang tương tác với người dùng. Android cố gắng đáp ứng mọi yêu cầu cần thiết cho nó, thậm chí huỷ các Activity ở đáy ngăn xếp để



Hình 3.2: Ngăn xếp các các Activities.

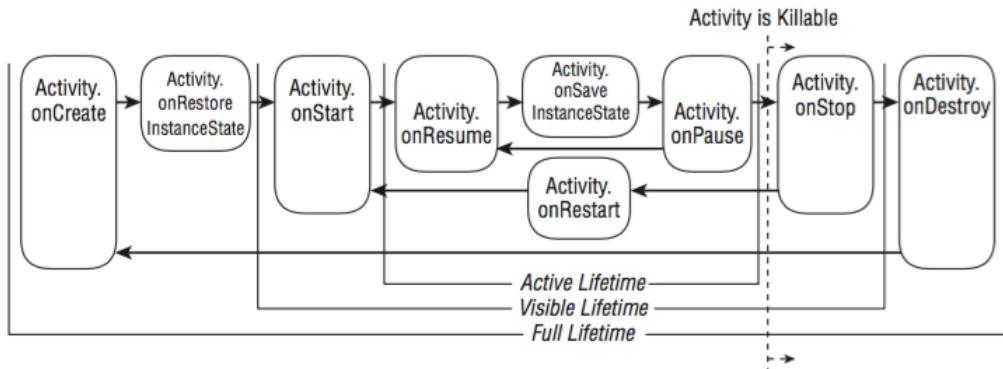
đáp ứng tài nguyên cho Activity này hoạt động. Khi Activity khác trở chuyển thành trạng thái Active thì Activity hiện hành chuyển sang trạng thái Pause.

Trạng thái Pause: ở trạng thái này thì Activity được nhìn thấy nhưng không được chọn và không thể tương tác với người dùng. Trong những trường hợp rất khẩn thiết thì Android có thể huỷ các Activity này để nhường tài nguyên cho Activity đang ở trạng thái Active.

Trạng thái Stopped: khi một Activity không thể được nhìn thấy thì nó ở trạng thái kết thúc. Activity sẽ lưu lại các tất cả các trạng thái và các thông tin liên quan, tuy nhiên nó là ứng cử viên đầu tiên sẽ bị giải phóng nếu hệ thống cần sử dụng tài nguyên. Khi một Activity ở trạng thái Stopped thì nó không thể hoạt động được.

Trạng thái inactive: sau khi Activity bị huỷ, trước khi nó được

khởi chạy lại nó sẽ ở trạng thái inactive. Các phương thức xử lý sự



Hình 3.3: Sơ đồ chuyển trạng thái của một Activity.

kiện tương ứng với từng trạng thái của một Activity như sau:

```
public class MyActivity extends Activity
    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        // Khởi tạo Activity
    }
    // Trạng thái RestoreInstanceState được gọi sau khi phương thức onCreate đã kết thúc
    @Override
    public void onRestoreInstanceState(Bundle savedInstanceState) {
        // Khôi phục lại giao diện người dùng
        super.onRestoreInstanceState(savedInstanceState);
    }
    // Phương thức khởi động lại Activity
    @Override
    public void onRestart(){
        super.onRestart();
    }
    // Trạng thái Start được gọi khi bắt đầu nhìn thấy (visible)
    @Override
    public void onStart(){
        super.onStart();
    }
    // Trạng thái Resume được gọi khi bắt đầu chu kỳ active, Activity tiếp tục hoạt động
    // sau khi đã ở trạng thái inactive
    @Override
    public void onResume(){
        super.onResume();
    }
    // Trạng thái onSaveInstanceState được gọi khi có sự thay đổi trên giao diện (UI) ở
    // cuối chu kỳ active
    @Override
```

```

public void onSaveInstanceState(Bundle savedInstanceState) {
    //Cập nhật các thay đổi giao diện (UI)
    super.onSaveInstanceState(savedInstanceState);
}
// Trạng thái Pause được gọi ở cuối chu kỳ active
@Override public void onPause(){
    super.onPause();
}
// Trạng thái stopped được gọi ở cuối chu kỳ nhìn thấy (visible).
@Override
public void onStop(){
    super.onStop();
}
// Trạng thái Destroy được gọi khi kết thúc ứng dụng
@Override
public void onDestroy(){
    // Giải phóng tài nguyên và ngắt kết nối đến cơ sở dữ liệu
    super.onDestroy();
}
}

```

Một Activity có thể tồn tại ở một trong ba chu kỳ sống là chu kỳ đầy đủ (Full Lifetime) bắt đầu phương thức onCreate() cho tới phương thức onDestroy(), chu kỳ nhìn thấy (Visible Lifetime) bắt đầu từ phương thức onStart() cho tới phương thức onStop() và chu kỳ kích hoạt (Active Lifetime) bắt đầu từ phương thức onResume() cho tới phương thức onPause().

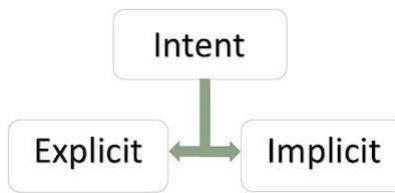
3.2 Đối tượng Intent và Intent Filter

3.2.1 Phân biệt Intent và Intent Filter

Các intent trong Android là một đối tượng thông điệp cho phép các thành phần trong cùng một ứng dụng hoặc thuộc các ứng dụng khác nhau tương tác với nhau.

Intent là đối tượng của class android.content.Intent. Có hai loại Intent là Intent tường minh và Intent không tường minh.

Intent tường minh (Explicit intents) là những intent có các thành phần được xác định rõ ràng. Chẳng hạn, gọi một Activity có tên ActivityTwo sử dụng Intent như sau:



Hình 3.4: . Hai loại Intent.

```

Intent i = new Intent(getApplicationContext(), ActivityTwo.class);
startActivity(i);

```

Intent không tường minh (Implicit Intent) là những intents có các thành phần không được xác định rõ ràng, trong trường hợp này Intent cung cấp các thông tin về hành động (action) sẽ được thực thi hoặc báo cáo và các dữ liệu (data) tương ứng sẽ được xử lý. Thí dụ, tạo một Intent không tường minh truy cập đến một trang Web như sau:

```

Intent intent=new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.javatpoint.com"));
startActivity(intent);

```

Các Intent Filter trong Android giống như một bộ lọc sự kiện, nó được sử dụng để đăng ký cho các thành phần Activities, Services và Broadcast Receivers sẽ thực hiện các hành động trên các dữ liệu tương ứng của các intent không tường minh. Một Intent Filter được định nghĩa bên trong một thành phần Activity, Service hoặc Broadcast Receiver để lọc sự kiện cho nó sử dụng thẻ <intent-filter> với ba thuộc tính action, category và data trong file manifest như sau:

Thuộc tính action là hành động sẽ được thực thi và được chỉ định bởi thuộc tính *android:name*. Mỗi Intent Filter có thể gồm một hoặc nhiều action trong số các action dưới đây:

là một hành động chính, khởi tạo đối tượng mà không nhận dữ liệu từ Intent. Action này kết hợp với category LAUNCHER sẽ cho phép ứng dụng xuất hiện trong danh sách ứng dụng trên màn hình của thiết bị.

ACTION_ANSWER cho phép mở một Activity và trả lời một cuộc gọi đến.

ACTION_CALL cho phép hiển thị ứng dụng gọi điện và thực hiện gọi đến số được cung cấp trong URI của Intent.

ACTION_DELETE cho phép chạy một Activity và xoá các mục trống được lưu trữ trong vị trí được chỉ định bởi URI.

ACTION_DIAL cho phép hiển thị ứng dụng gọi điện với số được cung cấp trong URI của .

ACTION_EDIT yêu cầu Activity soạn thảo dữ liệu trong URI.

ACTION_INSERT cho phép mở một Activity và chèn một Item mới vào vị trí được chỉ định trong dữ liệu. Khi gọi một Activity, nó sẽ trả về một URI tham chiếu tới vị trí item mới được chèn.

ACTION_PICK cho phép chạy một Activity con và chọn một item từ dữ liệu URI. Khi đóng Activity sẽ trả về một URI tham chiếu tới item vừa được chọn.

ACTION_SEARCH cho phép khởi chạy giao diện người dùng và thực hiện tìm kiếm với từ khoá được đưa ra trong dữ liệu của Intent sử dụng truy vấn SearchManager.QUERY.

ACTION_SENDTO cho phép chạy một Activity để gửi một tin nhắn tới một địa chỉ (contact) cụ thể được chỉ định trong dữ liệu của URI.

ACTION_SEND cho phép chạy một Activity để gửi dữ liệu cụ thể.

ACTION_VIEW là loại action phổ biến nhất, cho phép hiển

thị dữ liệu được cung cấp qua URI của Intent.

ACTION_WEB_SEARCH cho phép mở một Activity thực hiện tìm kiếm trang Web có địa chỉ được cung cấp qua URI của Intent.

Thuộc tính category chỉ định những trường hợp mà các action cần phục vụ, được khai báo qua thuộc tính *android:category*. Mỗi Intent Filter có thể có một hoặc nhiều thẻ category với các giá trị dưới đây:

ALTERNATIVE chỉ hành động này có thể được thực hiện tùy ý với một menu context nào đó của đối tượng.

SELECTED_ALTERNATIVE tương tự như ALTERNATIVE nhưng nó sử dụng một danh sách các khả năng có thể cho trước.

BROWSABLE chỉ định một hành động có thể trong trình duyệt.

DEFAULT chỉ hành động mặc định trên dữ liệu được định nghĩa của Intent Filter. Điều này rất cần thiết cho các Activity được gọi bởi một Intent tường minh.

GADGET chỉ định một Activity có thể chạy nhúng bên trong một Activity khác.

HOME chỉ định đây là Activity đầu tiên được hiển thị khi khởi tạo thiết bị.

LAUNCHER chỉ định một Activity xuất hiện trong danh sách các ứng dụng.

Thuộc tính data cho phép xác định các kiểu dữ liệu tương ứng với các action mà các thành phần sẽ xử lý. Dữ liệu được chỉ định thông qua các thuộc tính sau:

android:host: chỉ định một tên host (tên miền) hợp lệ (thí dụ com.google).

android:mimetype: chỉ định loại dữ liệu xử lý như dạng văn bản (text/plain), ảnh (image/*), video (video/*),....

android:path: chỉ định một đường dẫn hợp lệ trong URI (thí dụ /transport/boats/)

android:port: chỉ định chỉ số cổng hợp lệ trên host

android:scheme: yêu cầu các nội dung cụ thể (thí dụ content or http).

Thí dụ, các Activity được định nghĩa trong file manifest như sau:

```

<activity android:name="MainActivity">
    <!-- Activity này sẽ xuất hiện trong danh sách ứng dụng trên màn hình -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/> <category
            android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
<activity android:name="ShareActivity">
    <!-- Activity này thực hiện hành động "SEND" với dữ liệu text -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
</activity>
<activity android:name="MultiSendActivity">
    <!-- Activity này thực hiện hành động "SEND" và "SEND_MULTIPLE" với dữ liệu media
        -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/><action
            android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>

```

3.2.2 Gọi Activity sử dụng đối tượng Intent

Giả sử Activity A có tên MyActivity triệu gọi Activity B có tên MyOtherActivity như sau:

intent = new Intent(MyActivity.this, MyOtherActivity.class);

```
startActivity(intent);
```

Sau khi thực hiện câu lệnh trên Activity A đang ở trạng thái kích hoạt (active) sẽ chuyển sang trạng thái visible và Activity B (MyOtherActivity) sẽ trở thành Activity ở trạng thái active.

3.2.3 Gọi Activity và truyền dữ liệu qua Intent

Khi Activity A triệu gọi đến một Activity B đồng thời truyền dữ liệu cho nó theo cú pháp như sau:

```
//Tạo Activity B từ lớp DisplayMessageActivity
Intent intent = new Intent(this, DisplayMessageActivity.class);
//Giả sử dữ liệu được lấy từ một đối tượng EditText trong Activity A
EditText editText = (EditText) findViewById(R.id.editText);
String message = editText.getText().toString();
//Truyền dữ liệu cho Intent qua biến message và khởi chạy nó
intent.putExtra(EXTRA_MESSAGE, message);
startActivity(intent);
```

3.2.4 Nhận dữ liệu từ Activity qua Intent

Một Activity có thể được khởi khay và truyền dữ liệu cho nó đồng thời chờ nhận kết quả trả về theo cú pháp sau:

```
//Các lệnh trong thân hàm onCreate() của Activity A
EditText text = (EditText) findViewById(R.id.inputforintent);
String string = text.getText().toString();
//Mở Activity B có tên ResultActivity
Intent i = new Intent(this, ResultActivity.class);
//Truyền cho Activity B biến yourkey có giá trị nằm trong biến string.
i.putExtra("yourkey", string);
//Khởi tạo Activity B và chờ nhận kết quả trả về
startActivityForResult(i, REQUEST_CODE);
```

Kiểu dữ liệu truyền giữa các Activity ở dạng xâu kí tự, nếu là các kiểu dữ liệu khác như float, int,... người lập trình có thể chuyển đổi về kiểu xâu và ngược lại.

Khi Activity B nhận được dữ liệu từ Activity A, nó sẽ lấy các dữ liệu về sử dụng các câu lệnh sau:

```
extras = getIntent().getExtras();
String inputString = extras.getString("yourkey");
```

Activity B muốn gửi dữ liệu ngược lại trả lại Activity A đã gọi nó thì người lập trình phải viết đè phương thức onFinish() của nó như sau:

```
@Override
public void finish() {
    Intent intent = new Intent(); //Tạo một Intent mới
    //Lấy dữ liệu từ EditText trên giao diện
    EditText editText= (EditText) findViewById(R.id.returnValue);
    String string = editText.getText().toString();
    //Gửi dữ liệu về Activity A gọi nó qua biến returnkey với giá trị nằm trong biến
    string
    intent.putExtra("returnkey", string);
    //Trả về mã kết thúc với hằng RESULT_OK và truyền thông điệp qua Intent
    setResult(RESULT_OK, intent);
    super.finish(); //Gọi phương thức finish() của lớp cha
}
```

Activity A thực hiện nhận dữ liệu truyền về từ Activity B bằng cách viết đè phương thức onActivityResult() theo cú pháp sau:

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
        if (data.getStringExtra("returnkey")) {
            String result = data.getExtras().getString("returnkey");
            if (result != null && result.length() > 0) {
                // Hiển thị giá trị nhận được
                Toast.makeText(this, result, Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

Hằng số RESULT_OK và REQUEST_CODE cho biết việc truyền dữ liệu giữa Activity B cho Activity A có thành công hay không.
Thí dụ 3-1. Tạo hai giao diện Activity A có tên MainActivity.java và Activity B có tên ResultActivity.java.

File activity_main.xml chứa giao diện Activity A gồm một EditText và một Button như sau:

t

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
    <EditText
        android:id="@+id/inputforintent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:minHeight="60dp"
        android:text="Activity A"
        android:textSize="20sp" >
    </EditText>
    <Button
        android:id="@+id/startintent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/inputforintent"
        android:layout_below="@+id/inputforintent"
        android:onClick="onClick"
        android:text="Gọi Activity B" />
</RelativeLayout>
```

File activity_result.xml chứa giao diện của Activity B có tên Activity ResultActivity gồm một TextView để hiển thị dữ liệu từ Activity A gửi sang và một EditText để nhập dữ liệu gửi về Activity A.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/displayintentextra"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Input" />
    <EditText
        android:id="@+id/returnValue"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <requestFocus />
    </EditText>
```

```
</LinearLayout>
```

Trong file manifest cần khai báo thêm activity ResultActivity như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.vogella.android.first"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="14" />
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:label="@string/app_name"
            android:name=".MainActivity" >
            <intent-filter >
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:label="Result Activity"
            android:name=".ResultActivity" >
        </activity>
    </application>
</manifest>
```

File MainActivity.java có nội dung như sau:

```
package ...;
public class MainActivity extends Activity {
    // Hằng số quyết định giá trị trả về từ Activity B
    private static final int REQUEST_CODE = 10;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    // Phương thức được thực thi khi click lên nút nhấn và gọi Activity B qua Intent
    public void onClick(View view) {
        EditText text = (EditText) findViewById(R.id.inputforintent);
        String string = text.getText().toString();
        Intent i = new Intent(this, ResultActivity.class);
        // Truyền dữ liệu cho Activity B
        i.putExtra("yourkey", string);
        startActivityForResult(i, REQUEST_CODE);
    }
}
```

```

    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data){
        if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
            //nhận dữ liệu trả về từ Activity B
            if (data.hasExtra("returnkey")) {
                String result = data.getExtras().getString("returnkey");
                if (result != null && result.length() > 0) {
                    Toast.makeText(this, result, Toast.LENGTH_SHORT).show();
                }
            }
        }
    }
}

```

File `ActivityResult.java` chứa nội dung như sau:

```

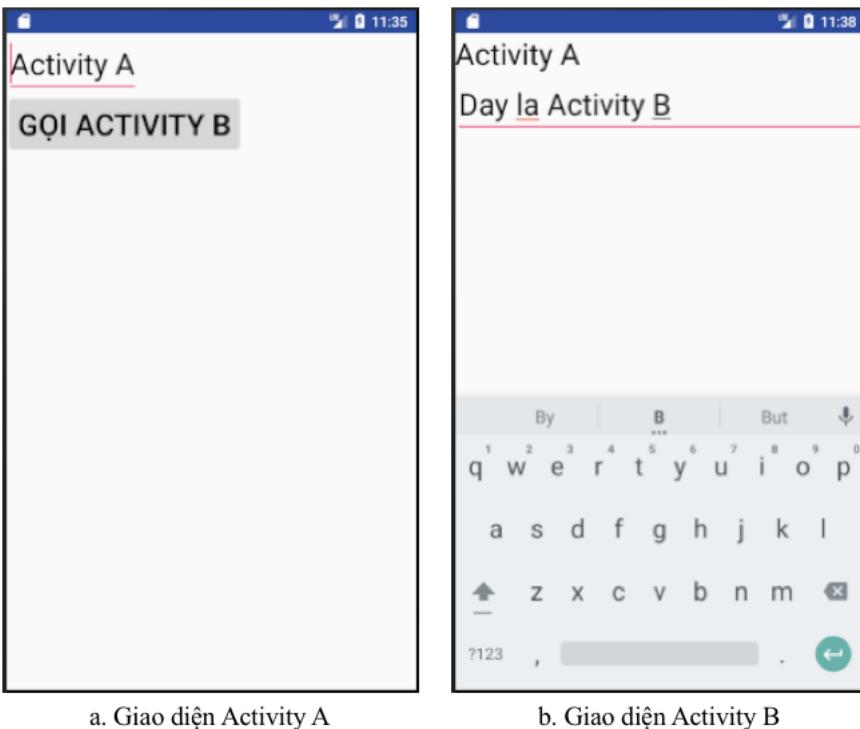
public class ResultActivity extends Activity {
    @Override
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_result);
        //Lấy dữ liệu từ Activity A và hiển thị lên TextView
        Bundle extras = getIntent().getExtras();
        String inputString = extras.getString("yourkey");
        TextView view = (TextView) findViewById(R.id.displayintextextra);
        view.setText(inputString);
    }
    //Khi đóng Activity B thì truyền dữ liệu về Activity A
    @Override
    public void finish() {
        Intent intent = new Intent();
        EditText editText= (EditText) findViewById(R.id.returnValue);
        String string = editText.getText().toString();
        intent.putExtra("returnkey", string);
        setResult(RESULT_OK, intent);
        super.finish();
    }
}

```

Kết quả thí dụ 3-1 như hình 3.5.

3.3 Đối tượng quảng bá thông điệp Broadcast Reciever

Đối tượng Broadcast Reciever cho phép một Activity nhận các thông điệp quảng bá từ các Activity khác. Chúng thường được sử



Hình 3.5: Kết quả thí dụ về Activity

dụng khi muốn nhắc người dùng cần chú ý đến một sự kiện nào nó đã xảy ra trong hệ thống.

Để sử dụng đối tượng Broadcast Reciever, người lập trình cần đăng ký sử dụng nó trong file manifest theo cú pháp như sau:

```
<receiver android:name="ServiceManager">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
    </intent-filter>
</receiver>
```

Trong đó, thuộc tính *android:name* chỉ định tên lớp nhận thông điệp quảng bá và thẻ *action* xác định tên sự kiện xảy ra trong hệ thống hoặc tên lớp thực hiện quảng bá. Thí dụ trong trường hợp

này là lớp SeviceManager sẽ nhận thông điệp quảng bá khi quá trình khởi động hệ thống hoàn thành.

Bảng 3.1 dưới đây liệt kê một số sự kiện hệ thống như sau:

Tên sự kiện	Mô tả
Intent.ACTION_BOOT_COMPLETED	Khởi động hệ thống hoàn thành. Yêu cầu cấp quyền truy cập đến thiết bị android.permission.RECEIVE_BOOT_COMPLETED
Intent.ACTION_POWER_CONNECTED	Khi sạc nguồn được kết nối
Intent.ACTION_BATTERY_LOW	Cảnh báo khi pin ở mức thấp
Intent.ACTION_BATTERY_OKAY	Trạng thái pin tốt trở lại

Bảng 3.1: Một số sự kiện hệ thống trong Android

Để tạo một thông điệp broadcast từ một Activity người lập trình cần sử dụng cú pháp như sau:

```
Intent intent = new Intent();
//Tên lớp quảng bá thông điệp
intent.setAction("com.example.Broadcast");
//Gửi dữ liệu qua Intent
intent.putExtra("HighScore", 1000);
//Quảng bá Intent
sendBroadcast(intent);
```

Để nhận một thông điệp BroadcastReceive người lập trình phải xây dựng lớp con kế thừa lớp BroadcastReceive và viết đè phương thức onReceive() theo cú pháp sau:

```
package ...;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
public class MyReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        //các lệnh cần thực hiện khi nhận được thông điệp quảng bá
    }
}
```

Ngoài ra, người lập trình có thể thực hiện đăng ký nhận thông điệp quảng bá sử dụng cú pháp sau:

```
filter = new IntentFilter("com.example.Broadcast");
MyReceiver receiver = new MyReceiver();
registerReceiver(receiver, filter);
```

Nếu muốn huỷ đăng ký nhận thông điệp quảng bá sử dụng cú pháp lệnh:

```
unregisterReceiver(receiver);
```

Thí dụ 3-2: Tạo hai ứng dụng, một ứng dụng gồm một Activity gửi có tên MyActivity gửi một thông điệp quảng bá là “Đây là thông điệp quảng bá”, ứng dụng thứ hai chứa một Activity có tên MyBroadcastReceive đăng ký nhận thông điệp quảng bá từ ứng dụng thứ nhất và hiển thị thông điệp nhận được lên màn hình qua thông điệp Toast.

File activity_main.xml chứa giao diện như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >
    <EditText android:id="@+id/extraintent"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="30dp"
        android:textColor="#000000"
        android:hint="Nhập thông điệp cần quảng bá" />
    <Button
        android:id="@+id/btnStartBroadcast"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/extraintent"
        android:onClick="broadcastIntent"
        android:text="Quảng bá" />
</RelativeLayout>
```

File manifest chứa khai báo đối tượng receiver như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.admin.exam_32">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name="MyBroadcastReceiver">
        </receiver>
    </application>
</manifest>
```

File MainActivity.java chứa mã như sau:

```
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void broadcastIntent(View view)
    {
        Intent intent = new Intent(this,MyBroadcastReceiver.class);
        EditText et = (EditText)findViewById(R.id.extraIntent);
        // gửi dữ liệu Intent
        intent.putExtra("message", (CharSequence)et.getText().toString());
        intent.setAction("MainActivity");
        sendBroadcast(intent);
    }
}
```

File MyBroadcastReceiver.java chứa mã như sau:

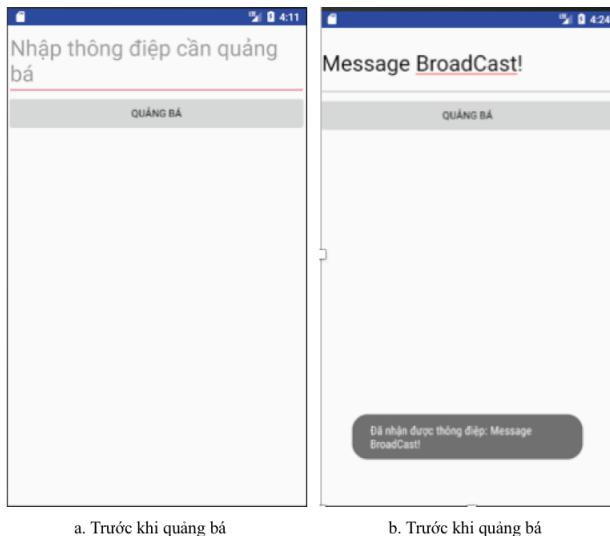
```
public class MyBroadcastReceiver extends BroadcastReceiver{
    @Override
    public void onReceive(Context context, Intent intent) {
        CharSequence intentData = intent.getCharSequenceExtra("message");
        Toast.makeText(context, "Đã nhận được thông điệp: "+intentData,
        Toast.LENGTH_LONG).show();
```

```

    }
}

```

Kết quả thí dụ 3-2 như hình 3.6



Hình 3.6: Kết quả thí dụ 3-2

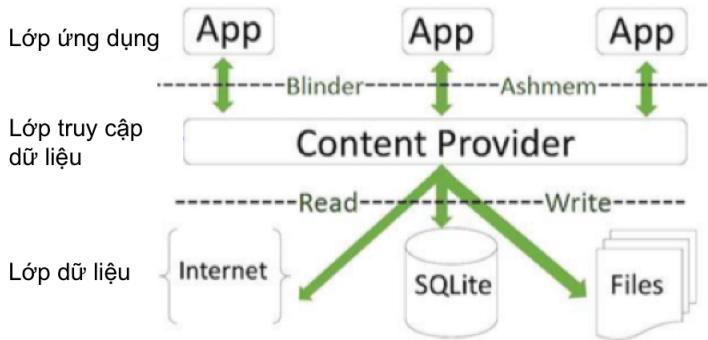
3.4 Đối tượng Content Provider

Đối tượng Content Provider cho phép một ứng dụng chia sẻ dữ liệu của nó với các ứng dụng khác trong hệ thống. Chẳng hạn như danh bạ (contact) trên thiết bị được sử dụng bởi nhiều ứng dụng như tin nhắn, cuộc gọi, email,..

Các dạng dữ liệu được dùng chung bởi nhiều ứng dụng thường bao gồm: các cặp giá trị dùng chung (preferences), các file dữ liệu được lưu trữ trong bộ nhớ, các cơ sở dữ liệu SQLite trong hệ thống và các dữ liệu chia sẻ qua mạng.

Khai báo lớp ContentProvider như sau:

```
class MyApplication extends ContentProvider { }
```



Hình 3.7: Mô hình chia sẻ dữ liệu qua ContentProvider

Để truy vấn một Content Provider, người lập trình phải xác định các thành phần trong chuỗi URI theo định dạng:

<prefix>://<authority>/<data_type>/<id>

Các thành phần trong truy chuỗi URI được mô tả như bảng 3.2.

Phần	Mô tả
<prefix>	Luôn được thiết lập là content://
<authority>	Chỉ định tên cụ thể của Content Provider (ví dụ: contacts, browser,...). Riêng đối với các ứng dụng không phải của hệ thống thì bạn phải chỉ định tên đầy đủ tới package của ứng dụng, ví dụ: com.ictu.androidebook)
<data_type>	Chỉ rõ kiểu dữ liệu (ví dụ: Để lấy tất cả các liên hệ trong Contacts Content Provider thì kiểu dữ liệu là people và URI sẽ là: content://contacts/people)
<id>	Chỉ định rõ một record (ví dụ: Nếu muốn lấy địa chỉ liên lạc thứ 5 trong Contacts Content Provider thì URI sẽ là: content://contacts/people/5)

Bảng 3.2: Các thành phần của chuỗi URI

Các bước để tạo một đối tượng ContentProvider bao gồm:

Bước 1: Tạo ra một lớp con kế thừa lớp ContentProvider

Bước 2: Xác định giá trị của chuỗi URI được sử dụng để truy

cập đến nội dung

Bước 3: Tạo cơ sở dữ liệu SQL, trong Android sử dụng SQLite

Bước 4: Viết đè các phương thức truy vấn đến cơ sở dữ liệu

Bước 5: Đăng ký sử dụng đối tượng Content Provider cho Activity trong file manifest sử dụng thẻ <provider>.

Các phương thức cần viết đè trong quá trình làm việc với đối tượng ContentProvider bao gồm:

onCreate(): được gọi khi khởi động Content Provider

query(): nhận một yêu cầu từ phía truy xuất và trả về một đối tượng con trả Cursor.

insert(): thêm một bảng ghi mới vào content provider.

delete(): xoá một bản ghi đang tồn tại trong content provider.

update(): cập nhật một bản ghi đang tồn tại trong content provider.

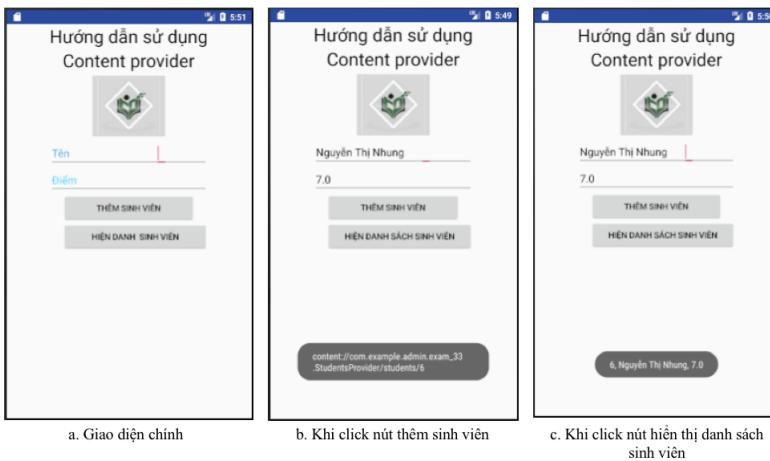
getType(): trả về kiểu MIME (Multipurpose Internet Mail Extensions) của dữ liệu trong chuỗi URI được đưa ra.

Thí dụ 3-3 sẽ giải thích chi tiết về các bước tạo một đối tượng Content Provider làm việc với cơ sở dữ liệu SQLite.

Thí dụ 3-3: Tạo cơ sở dữ liệu College gồm bảng students sau đó thiết kế giao diện như hình 3.8 cho phép thêm các bảng ghi dữ liệu về sinh viên gồm trường Tên và Điểm bằng cách click vào nút Thêm sinh viên. Mỗi khi thêm một bản ghi thì hiển thị nội dung của chuỗi URI ra màn hình như hình (b). Khi người dùng click chọn nút Hiển thị danh sách sinh viên thì hiển thị thông tin từng sinh viên trong danh sách ra màn hình như hình c. Tạo một project mới có tên MyApplication và thực hiện các bước như sau:

Bước 1: File MainActivity.java có nội dung như sau:

```
package ...;
import ...;
public class MainActivity extends Activity {
```



Hình 3.8: Thí dụ về Content Provider

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}

// phương thức cho phép thêm một bản ghi sinh viên vào cơ sở dữ liệu (CSDL)
public void onClickAddName(View view) {
    ContentValues values = new ContentValues();
    // lấy giá trị tên sinh viên
    values.put(StudentsProvider.NAME,
        ((EditText)findViewById(R.id.editText2)).getText().toString());
    // lấy giá trị điểm sinh viên
    values.put(StudentsProvider.GRADE,
        ((EditText)findViewById(R.id.editText3)).getText().toString());
    thực hiện câu truy vấn insert vào CSDL
    Uri uri = getContentResolver().insert(StudentsProvider.CONTENT_URI, values);
    Toast.makeText(getApplicationContext(),
        uri.toString(), Toast.LENGTH_LONG).show();
}

// Phương thức lấy thông tin các sinh viên trong CSDL
public void onClickRetrieveStudents(View view) {
    String URL = "content://com.example.MyApplication.StudentsProvider";
    Uri students = Uri.parse(URL);
    Cursor c = managedQuery(students, null, null, null, "name");
    // di chuyển con trỏ đến bản ghi đầu tiên
    if (c.moveToFirst()) {
        do{
    // hiển thị ID, tên và điểm của sinh viên qua một thông điệp Toat
        Toast.makeText(this,
            c.getString(c.getColumnIndex(StudentsProvider._ID)) +
            ", " + c.getString(c.getColumnIndex( StudentsProvider.NAME)) +
            ", " + c.getString(c.getColumnIndex( StudentsProvider.GRADE)),
            Toast.LENGTH_SHORT).show();
    }
}

```

```

        } while (c.moveToNext());
    }
}

```

Bước 2: Tạo file StudentsProvider.java và soạn thảo nội dung như sau:

```

package...;
import...;
public class StudentsProvider extends ContentProvider {
    static final String PROVIDER_NAME = "com.example.MyApplication.StudentsProvider";
    static final String URL = "content://" + PROVIDER_NAME + "/students";
    static final Uri CONTENT_URI = Uri.parse(URL);
    // khai báo các hằng xâu tương ứng với tên các cột của bảng
    static final String _ID = "_id"; // chứa khóa
    static final String NAME = "name"; // chứa tên sinh viên
    static final String GRADE = "grade"; // chứa điểm sinh viên
    // khai báo biến kiểu một cặp giá trị HashMap
    private static HashMap<String, String> STUDENTS_PROJECTION_MAP;

    static final int STUDENTS = 1; // lựa chọn tất cả các sinh viên
    static final int STUDENT_ID = 2; // lựa chọn một sinh viên
    // tạo một URI cho phép so sánh giá trị được chọn
    static final UriMatcher uriMatcher;
    static{
        uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
        uriMatcher.addURI(PROVIDER_NAME, "students", STUDENTS);
        uriMatcher.addURI(PROVIDER_NAME, "students/#", STUDENT_ID);
    }
    // Định nghĩa CSDL
    private SQLiteDatabase db; // khai báo biến kiểu CSDL SQLite
    static final String DATABASE_NAME = "College"; // tên CSDL là College
    static final String STUDENTS_TABLE_NAME = "students"; // tên bảng dữ liệu là
    students
    static final int DATABASE_VERSION = 1; // phiên bản CSDL
    // câu truy vấn thực hiện tạo bảng students trong CSDL
    static final String CREATE_DB_TABLE =
        " CREATE TABLE " + STUDENTS_TABLE_NAME +
        " (_id INTEGER PRIMARY KEY AUTOINCREMENT, " +
        " name TEXT NOT NULL, " +
        " grade FLOAT NOT NULL);";
    // Lớp Helper là lớp thực sự tạo và quản lý kho dữ liệu của provider
    private static class DatabaseHelper extends SQLiteOpenHelper {
        // toán tử tạo lập
        DatabaseHelper(Context context){
            super(context, DATABASE_NAME, null, DATABASE_VERSION);
        }
        // phương thức thực thi câu truy vấn tạo bảng students trong CSDL
        @Override
        public void onCreate(SQLiteDatabase db) {
            db.execSQL(CREATE_DB_TABLE);
        }
    }
}

```

```
    }
    // phương thức thực thi câu truy vấn xoá bảng students trong CSDL khi nó đã tồn
    // tại
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + STUDENTS_TABLE_NAME);
        onCreate(db);
    }
}
// phương thức thực hiện tạo CSDL cho phép ghi dữ liệu nếu nó chưa tồn tại
@Override
public boolean onCreate() {
    Context context = getContext();
    DatabaseHelper dbHelper = new DatabaseHelper(context);
    db = dbHelper.getWritableDatabase();
    return (db == null)? false:true;
}
// phương thức cho phép thêm một bản ghi mới vào CSDL
@Override
public Uri insert(Uri uri, ContentValues values) {
    long rowID = db.insert(STUDENTS_TABLE_NAME, "", values);
// nếu thực hiện thêm bản ghi thành công thì trả về một chuỗi uri được nối thêm ID
// của dòng mới thêm
    if (rowID > 0) {
        Uri _uri = ContentUris.withAppendedId(CONTENT_URI, rowID);
        getContext().getContentResolver().notifyChange(_uri, null);
        return _uri;
    }
    throw new SQLException("Failed to add a record into " + uri);
}
// thực hiện câu truy vấn tới bảng students
@Override
public Cursor query(Uri uri, String[] projection, String selection,
String[] selectionArgs, String sortOrder) {
    SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
    qb.setTables(STUDENTS_TABLE_NAME);
    switch (uriMatcher.match(uri)) {
        case STUDENTS:
            // thiết lập tham chiếu cho câu truy vấn
            qb.setProjectionMap(STUDENTS_PROJECTION_MAP);
            break;
        // nối thêm một xâu vào mệnh đề WHERE của câu truy vấn
        case STUDENT_ID:
            qb.appendWhere( _ID + "=" + uri.getPathSegments().get(1));
            break;
        default:
    }
    if (sortOrder == null || sortOrder == ""){
        // mặc định sắp xếp danh sách theo tên sinh viên
        sortOrder = NAME;
    }
    // thực hiện câu truy vấn
```

```

        Cursor c = qb.query(db, projection, selection, selectionArgs,null, null,
        sortOrder);
        // xem nội dung của URI thay đổi
        c.setNotificationUri(getContext().getContentResolver(), uri);
        return c;
    }
    // câu truy vấn xoá dữ liệu trong bảng sinh viên
    @Override
    public int delete(Uri uri, String selection, String[] selectionArgs) {
        int count = 0;
        switch (uriMatcher.match(uri)){
            case STUDENTS:
                //nếu không có điều kiện xoá tất cả các sinh viên
                count = db.delete(STUDENTS_TABLE_NAME, selection, selectionArgs);
                break;
            //xoá sinh viên với ID xác định
            case STUDENT_ID:
                String id = uri.getPathSegments().get(1);
                count = db.delete( STUDENTS_TABLE_NAME, _ID + " = " + id +
                    (!TextUtils.isEmpty(selection) ? "
                    AND (" + selection + ')' : ""), selectionArgs);
                break;
            default:
                throw new IllegalArgumentException("Unknown URI " + uri);
        }
        getContext().getContentResolver().notifyChange(uri, null);
        return count;
    }
    // phương thức lấy dữ liệu sinh viên
    @Override
    public String getType(Uri uri) {
        switch (uriMatcher.match(uri)){
            // lấy tất cả các sinh viên
            case STUDENTS:
                return "vnd.android.cursor.dir/vnd.example.students";
            // lấy một sinh viên cụ thể
            case STUDENT_ID:
                return "vnd.android.cursor.item/vnd.example.students";
            default:
                throw new IllegalArgumentException("Unsupported URI: " + uri);
        }
    }
}

```

Bước 3: Tạo file manifest có nội dung như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.MyApplication">
    <application
        android:allowBackup="true"

```

```
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <provider android:name="StudentsProvider"
            android:authorities="com.example.MyApplication.StudentsProvider"/>
    </application>
</manifest>
%
```

Bước 4: Tạo file activity_main.xml có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.admin.exam_33.MainActivity">
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hướng dẫn sử dụng"
        android:textSize="30dp" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Content provider"
        android:textSize="30dp"
        android:layout_centerHorizontal="true" />
    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageButton"
        android:src="@drawable/abc"
        android:layout_centerHorizontal="true" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button2"
        android:text="Thêm sinh viên"
        android:onClick="onClickAddName" />
    <EditText
```

```

    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/editText" />
<EditText
    android:id="@+id/editText2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="Tên" />
<EditText
    android:id="@+id/editText3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="Điểm" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hiển danh sinh viên"
    android:id="@+id/button"
    android:onClick="onClickRetrieveStudents"/>
</RelativeLayout>

```

3.5 Các dịch vụ (Services)

Một dịch vụ (Service) là một thành phần được chạy bên trong hệ thống để xử lý các công việc trong thời gian dài. Thí dụ như dịch vụ nghe nhạc, tải dữ liệu từ mạng Internet vẫn được thực hiện bên trong hệ thống trong khi người dùng vẫn có thể tương tác với giao diện của các ứng dụng khác đang chạy.

Sử dụng các dịch vụ có các lợi ích là: (1) khi một tiến trình (process) chạy sinh ra các tiểu trình (threads) và khi tiến trình kết thúc thì các tiểu trình của nó sẽ kết thúc, còn các dịch vụ có thể được hệ thống khởi động lại và kết thúc tùy theo mức độ quan trọng của nó; (2) các Activity và Broadcast Received độc lập với các tiểu trình mà nó sinh ra còn các Service thì gắn kết chặt chẽ với các tiểu trình nó sinh ra và có thể kết thúc các tiểu trình khi không cần thiết bằng cách sử dụng phương thức `stopSelf()`.

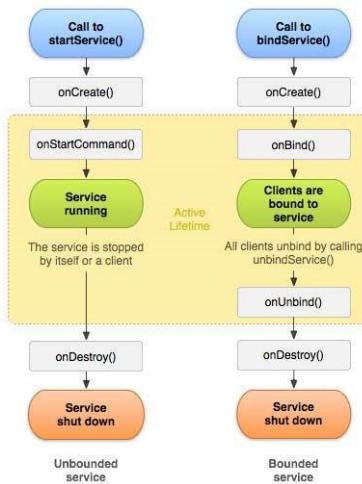
Có hai loại dịch vụ:

Started Service: một service thuộc loại started được triệu gọi khi một thành phần của ứng dụng khởi động nó bằng cách gọi phương thức `startService()`. Mỗi lần được khởi động, các dịch vụ chạy trên hệ thống là vô thời hạn.

Bound Service: các Bound Services được gọi khi một thành phần ứng dụng liên kết với nó bằng cách gọi phương thức `bindService()`. Một Bound Service cung cấp một giao diện client-server cho phép các thành phần tương tác với Service gửi yêu cầu và nhận kết quả từ nó.

Mỗi Service có tập các phương thức cho phép người lập trình hiện thực hóa để giám sát sự thay đổi các trạng thái của nó và thực thi các công việc phù hợp ở mỗi trạng thái.

Sơ đồ mô tả vòng đời của các phương thức tương ứng với các trạng thái của từng loại Service từ khi được tạo cho tới khi bị huỷ.



Hình 3.9: Trình tự các trạng thái của Started Service và Bound Service

Ý nghĩa của từng phương thức như sau:

`onStartCommand()`: phương thức này được gọi khi có một thành phần nào đó (các activities) đã yêu cầu khởi tạo Service bằng cách gọi phương thức `startService()`. Dịch vụ sẽ bị huỷ nếu người dùng

gọi phương thức stopSelf() hoặc stopService() khi không có nhu cầu sử dụng dịch vụ.

onBind(): phương thức này được gọi khi có một thành phần khác gửi một liên kết tới Service thông qua việc gọi phương thức bindService(). Trong trường hợp này người lập trình phải cung cấp một giao diện để các client giao tiếp với Service và trả về một đối tượng IBinder.

onUnbind(): phương thức này được gọi khi tất cả các client đã bị ngắt kết nối tới giao diện được tạo bởi các Service.

onRebind(): phương thức này được gọi khi có một client mới kết nối tới Service mà trước đó tất cả các kết nối đã bị ngắt bởi phương thức *onUnbind()*.

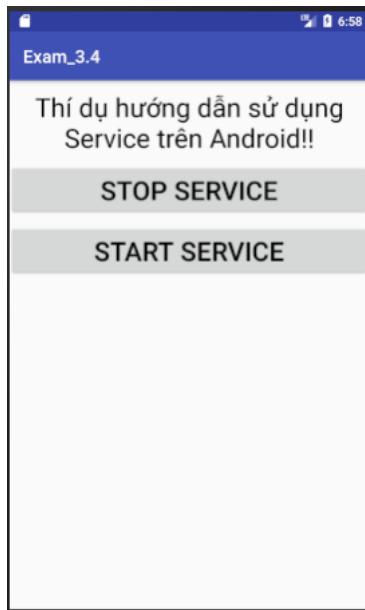
onCreate(): phương thức này được gọi khi các Services được tạo lần đầu tiên bởi gọi phương thức *onStartCommand()* hoặc *onBind()*.

onDestroy(): phương thức này được gọi khi các Services đã không còn được sử dụng trong thời gian dài. Người lập trình nên gọi phương thức này để giải phóng các tài nguyên hệ thống.

Để sử dụng các Service cần khai báo trong file manifest như sau:

```
<manifest ... >
    ...
    <application ... >
        <service android:name=".ExampleService" />
        ...
    </application>
</manifest>
```

Thí dụ 3-4. Tạo một lớp dịch vụ có tên HelloService liên tục in ra trong cửa sổ nhật ký Log dòng “Service running” 5 lần sau đó kết thúc dịch vụ. Lớp HelloActivity chứa hai nút nhấn “Start Service” và “Stop Service” cho phép khởi động và dừng Service HelloService như hình 3.10.



Hình 3.10: Giao diện lớp HelloActivity

File HelloService.java định nghĩa dịch vụ như sau:

```
public class HelloService extends Service {
    private static final String TAG = "HelloService";
    private boolean isRunning = false;
    // phương thức khởi tạo Service
    @Override
    public void onCreate() {
        Log.i(TAG, "Service onCreate");
        isRunning = true;
    }
    //phương thức thực thi khi Service được khởi tạo
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Log.i(TAG, "Service onStartCommand");
        //tạo ra một tiểu trình mới cho Service,
        //luôn nhớ rằng mỗi thread sẽ thực hiện các công việc khác nhau.
        new Thread(new Runnable() {
            @Override
            public void run() {
                //tạo vòng lặp đợi 1000 milli giây sau mỗi lần lặp
                for (int i = 0; i < 5; i++) {
                    try {
                        Thread.sleep(1000);
                    } catch (Exception e) {
                    }
                }
                //nếu dịch vụ đang chạy thì in ra dòng "Hello Service, Service running"
            }
        }).start();
    }
}
```

```

        if(isRunning){
            Log.i(TAG, "Service running");
        }
    }
    // Dừng Service khi kết thúc công việc
    stopSelf();
}
}).start();
//hàng START_STICKY được sử dụng cho các dịch vụ bắt đầu và kết thúc khi cần
thiết
return Service.START_STICKY;
}
//phương thức cho biết Service đang được kết nối
@Override
public IBinder onBind(Intent arg0) {
    Log.i(TAG, "Service onBind");
    return null;
}
//phương thức huỷ dịch vụ
@Override
public void onDestroy() {
    isRunning = false;
    Log.i(TAG, "Service onDestroy");
}
}
}

```

File HelloActivity.java có nội dung như sau:

```

public class HelloActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_hello);
        // Khởi động dịch vụ khi người dùng click lên nút "Start Service"
        findViewById(R.id.start_service).setOnClickListener(new
            View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent intent = new Intent(HelloActivity.this, HelloService.class);
                    startService(intent);
                }
            });
        // Huỷ dịch vụ khi người dùng click lên nút Stop Service
        findViewById(R.id.stop_Service).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(HelloActivity.this, HelloService.class);
                stopService(intent);
            }
        });
    }
}

```

File activity_hello.xml có nội dung như sau:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".HelloActivity">
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/textView"
        android:text="Thí dụ hướng dẫn sử dụng Service trong Android!!!"
        android:layout_marginTop="10dp"
        android:textAppearance="?android:attr/textAppearanceMedium"/>
    <Button
        android:id="@+id/start_service"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView2"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:text="Start Service" />
    <Button
        android:id="@+id/stop_Service"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/start_service"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:text="Stop Service" />
</RelativeLayout>
```

File AndroidManifest.xml định nghĩa Service như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.javatechig.serviceexample" >
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
```

```

    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
<activity
    android:name=".HelloActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<!-- khai báo dịch vụ -->
<service android:name=".HelloService"
    android:exported="false"/>
</application>
</manifest>

```

Kết quả thí dụ 3-4 như hình 3.10.

Câu hỏi và bài tập

1. Vòng đời một Activity có thể ở vào những trạng thái nào?
2. Các Activity ở vào trạng thái nào có thể bị huỷ để giải phóng tài nguyên hệ thống?
3. Thứ tự ưu tiên của các thành ứng dụng Android được sắp xếp như thế nào?
4. Cách truyền tham số từ một Activity này sang Activity khác?
5. Phân biệt giữa đối tượng Intent và Intent Filter?
6. Vai trò của các thông điệp quảng bá trong hệ thống?
7. Nêu lợi ích của việc sử dụng các dịch vụ (Service) trong Android?
8. Vai trò của đối tượng Content Provider trong Android?
9. Nêu định dạng của chuỗi URI được sử dụng cho đối tượng Content Provider?
10. Nêu các bước tạo một đối tượng Content Provider?

Chương 4

Lập trình mạng trong Android

4.1	Làm việc với HttpClient	142
4.2	Làm việc với UrlConnection	143
4.3	Xây dựng WebService đơn giản với PHP	143
4.4	Xử lý dữ liệu với Json trong Android	153

Android hỗ trợ các hàm API cho phép kết nối với các mạng truyền thông phổ biến như Wifi, bluetooth, NFC,... Chương này sẽ trình bày các hàm API hỗ trợ lập trình kết nối ứng dụng Android với mạng Internet.

Các ứng dụng Android có nhu cầu truy xuất chủ yếu đến các dịch vụ web (web services) thông qua giao thức HTTP. Các dịch vụ web chia làm hai dạng là XML/SOAP và RESTful. Các dịch vụ web dạng XML/SOAP có tính năng đầy đủ hơn nhưng yêu cầu dung lượng lớn hơn, còn các dịch vụ dạng RESTful thì khá nhẹ nhưng các tính năng được cắt giảm để đơn giản hơn và không sử dụng định dạng dữ liệu XML. RESTful sử dụng phương pháp JSON (JavaScript Object Notation) và các định dạng dữ liệu khác cho WEB.

Phần sau đây sẽ trình bày cách truy xuất dịch vụ web dạng RESTful. Android cung cấp thư viện Apache HttpClient hỗ trợ lập trình với giao thức HTTP ở mức cao và hỗ trợ sử dụng phương hai

phương thức GET and POST.

Trước hết, để ứng dụng Android có thể truy cập vào mạng người lập trình cần cấp quyền truy cập trong file manifest như sau:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Quyền kiểm tra trạng thái kết nối mạng của thiết bị được kiểm soát bởi thẻ:

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

4.1 Làm việc với HttpClient

Lớp HttpClient cung cấp phương thức GET hỗ trợ kết nối đến dịch vụ web. Việc truy xuất đến dịch vụ Web sử dụng lớp HttpClient theo cú pháp:

```
url = new URL(link);
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet();
request.setURI(new URI(link));
```

Tham số link là đường dẫn đến địa chỉ của Server, các thí dụ phần tiếp theo sẽ giải thích rõ hơn về cách sử dụng lớp HttpClient.

Để sử dụng các lớp hỗ trợ lập trình PHP trong ứng dụng Android người lập trình phải thêm đoạn lệnh sau vào file build.gradle:

```
android {
    useLibrary 'org.apache.http.legacy'
}
```

4.2 Làm việc với *URLConnection*

Lớp *URLConnection* kết hợp với lớp *URLEncoder* cung cấp các phương thức kết nối đến dịch vụ web qua phương thức POST và cho phép mã hoá các thông tin như sau:

```
url = new URL(link);
String data = URLEncoder.encode("username", "UTF-8")
+ "=" + URLEncoder.encode(username, "UTF-8");
data += "&" + URLEncoder.encode("password", "UTF-
8") + "=" + URLEncoder.encode(password, "UTF-8");
URLConnection conn = url.openConnection();
```

4.3 Xây dựng WebService đơn giản với PHP

Phần này sẽ trình bày cách tích hợp PHP và MySQL vào trong ứng dụng Android. Kỹ thuật này được sử dụng khi một ứng dụng Android muốn truy cập vào một dịch vụ web chạy trên hệ thống của chính nó.

CSDL MySQL được tạo sử dụng một kịch bản đơn giản như sau:

```
<?php
$con=mysqli_connect("example.com", "username", "password");
$sql="CREATE DATABASE my_db"; // Tạo CSDL có tên my_db
if (mysqli_query($con,$sql)) {
    echo "CSDL my_db đã được tạo thành công!";
}
?>
```

Người lập trình cần tạo các bảng trong CSDL:

```
<?php
$con=mysqli_connect("example.com", "username", "password", "my_db");
//Tạo bảng table1 với 3 cột
$sql="CREATE TABLE table1(Username CHAR(30),
Password CHAR(30),Role CHAR(30))";
if (mysqli_query($con,$sql)) {
```

```

        echo "Bảng đã được tạo thành công";
    }
?>

```

Các lệnh chèn một bản ghi dữ liệu vào bảng trong CSDL:

```

<?php
$con=mysqli_connect("example.com","username","password","my_db");
$sql="INSERT INTO table1 (FirstName, LastName, Age) VALUES ('admin',
'admin','adminstrator')";
if (mysqli_query($con,$sql)) {
    echo "Các giá trị đã được chèn thông công vào CSDL";
}
?>

```

Để đọc các bản ghi trong CSDL cần sử dụng phương thức GET hoặc phương thức POST với cú pháp lệnh theo trình tự lệnh như sau:

```

<?php
$con=mysqli_connect("example.com","username","password","databasename");
if (mysqli_connect_errno($con)) {
// Nếu kết nối không thành công
    echo "Lỗi kết nối tới CSDL MySQL: " . mysqli_connect_error();
}
$username = $_GET['username']; // hoặc $username = $_POST['username'];
$password = $_GET['password']; //hoặc $password = $_POST['password'];
$result = mysqli_query($con, "SELECT Role FROM table1 where
Username='".$username' and Password='".$password"");
$row = mysqli_fetch_array($result);
$data = $row[0];
// Hiển thị dữ liệu
if($data){
    echo $data;
}
//Đóng CSDL
mysqli_close($con);
?>

```

Có hai cách kết nối trang web PHP tới CSDL MySQL:

Cách 1: Sử dụng phương thức GET kết hợp với hai lớp HttpURLConnection và HttpClient theo cú pháp:

url = new URL(link);

```
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet();
request.setURI(new URI(link));
```

Sau đó sử dụng phương thức execute của lớp HttpClient để trả về một đối tượng HttpResponse và nhận dữ liệu như sau:

```
HttpResponse response = client.execute(request);
```

```
BufferedReader in = new BufferedReader (new InputStreamReader(response.getEntity().getContent()));
```

Cách 2: Sử dụng phương thức POST kết hợp với hai lớp URLEncoder, URLConnection theo cú pháp:

```
URL url = new URL(link);
String data = URLEncoder.encode("username", "UTF-8")
+ "=" + URLEncoder.encode(username, "UTF-8");
data += "&" + URLEncoder.encode("password", "UTF-8")
+ "=" + URLEncoder.encode(password, "UTF-8");
URLConnection conn = url.openConnection();
```

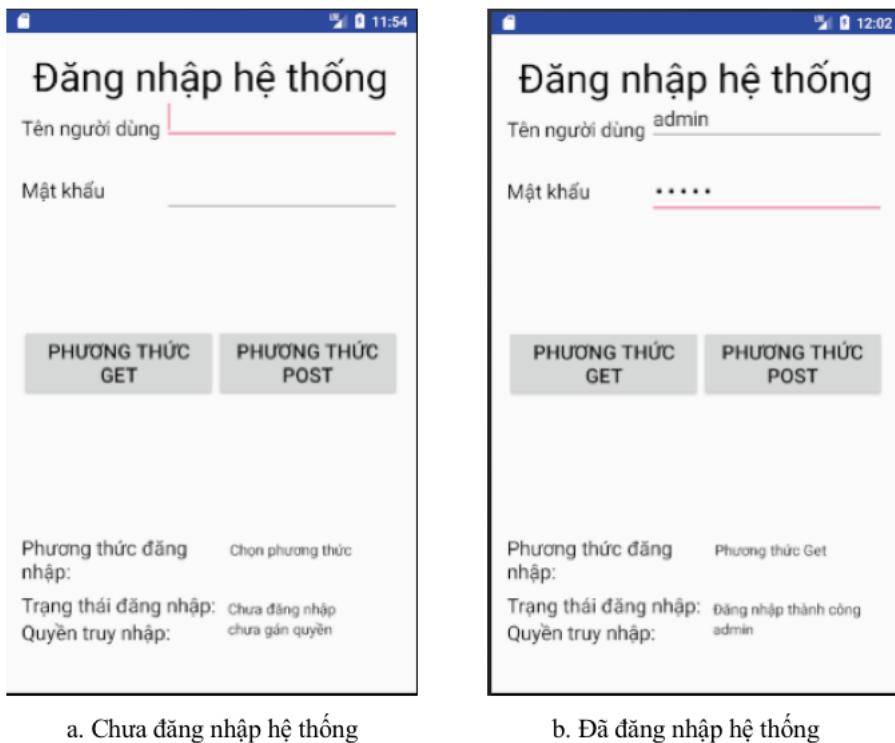
Sau đó thực hiện ghi dữ liệu và mở luồng để nhận dữ liệu phản hồi:

```
OutputStreamWriter wr = new OutputStreamWriter(conn.getOutputStream());
wr.write( data );
BufferedReader reader = new BufferedReader(new
InputStreamReader( conn.getInputStream() ));
```

Thí dụ 3-5: Thí dụ này minh họa cách kết nối ứng dụng Android đến CSDL 000webhost.com chứa một bảng có tên table1, bảng này gồm 3 cột (Username, Password, Role) và chứa một bản ghi đã được tạo sẵn.

Trang web php chứa các lệnh như sau:

```
<?php
$con=mysqli_connect("mysql10.000webhost.com", "username", "password", "db_name");
if (mysqli_connect_errno($con)) {
echo "Failed to connect to MySQL: " . mysqli_connect_error();
```



Hình 4.1: Thí dụ đăng nhập hệ thống

```

}
$username = $_POST['username'];
$password = $_POST['password'];
$result = mysqli_query($con, "SELECT Role FROM table1 where
Username='".$username' and Password='".$password"'");
$row = mysqli_fetch_array($result);
$data = $row[0];
if($data){
    echo $data;
}
mysqli_close($con);
?>

```

Giao diện chương trình chính ứng dụng Android cho phép kết nối tới CSDL sử dụng phương thức GET hoặc phương thức POST như hình 4.1:

Tạo project Android và thực hiện các bước như sau:

Bước 1: File activity_main.xml định nghĩa giao diện của ứng

dụng có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/editText1"
        android:layout_below="@+id/editText1"
        android:layout_marginTop="25dp"
        android:textColor="#000000"
        android:textSize="20dp"
        android:ems="10"
        android:inputType="textPassword" >
    </EditText>
    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="44dp"
        android:textColor="#000000"
        android:textSize="20dp"
        android:ems="10" >
        <requestFocus android:layout_width="wrap_content" />
    </EditText>
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/editText1"
        android:layout_alignParentLeft="true"
        android:textColor="#000000"
        android:textSize="20dp"
        android:text="Tên người dùng" />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textColor="#000000"
```

```
    android:textSize="40dp"
    android:text="Đăng nhập hệ thống"
    android:textAppearance="?android:attr/textAppearanceLarge" />
<TextView
    android:id="@+id/textView7"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="chưa gán quyền"
    android:textColor="#000000"
    android:textSize="15dp"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:layout_below="@+id/textView4"
    android:layout_toRightOf="@+id/button1"
    android:layout_toEndOf="@+id/button1"
    android:layout_marginLeft="12dp"
    android:layout_marginStart="12dp" />
<TextView
    android:id="@+id/textView5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:textSize="20dp"
    android:text="Quyền truy nhập:"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginBottom="31dp" />
<TextView
    android:id="@+id/textView8"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Phương thức đăng nhập:"
    android:textColor="#000000"
    android:textSize="20dp"
    android:layout_marginBottom="34dp"
    android:layout_above="@+id/textView5"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_toStartOf="@+id/textView7"
    android:layout_toLeftOf="@+id/textView7" />
<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:textSize="20dp"
    android:text="Trạng thái đăng nhập:"
    android:layout_above="@+id/textView5"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />
<TextView
    android:id="@+id/textView6"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Chưa đăng nhập"
        android:textColor="#000000"
        android:textSize="15dp"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:layout_above="@+id/textView7"
        android:layout_alignLeft="@+id/textView7"
        android:layout_alignStart="@+id/textView7" />

<TextView
    android:id="@+id/textView9"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:textSize="15dp"
    android:text="Chọn phương thức"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:layout_alignBaseline="@+id/textView8"
    android:layout_alignBottom="@+id/textView8"
    android:layout_toRightOf="@+id/textView8"
    android:layout_toEndOf="@+id/textView8"
    android:layout_marginLeft="14dp"
    android:layout_marginStart="14dp" />
<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="loginPost"
    android:textColor="#000000"
    android:textSize="20dp"
    android:text="Phương thức Post"
    android:layout_centerVertical="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_toRightOf="@+id/textView4"
    android:layout_toEndOf="@+id/textView4" />
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:textSize="20dp"
    android:onClick="login"
    android:text="Phương thức Get"
    android:layout_centerVertical="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_toLeftOf="@+id/button2"
    android:layout_toStartOf="@+id/button2" />
<TextView
    android:id="@+id/textView2"
```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/editText2"
        android:layout_alignBottom="@+id/editText2"
        android:layout_alignParentLeft="true"
        android:textColor="#000000"
        android:textSize="20dp"
        android:text="Mật khẩu" />
</RelativeLayout>

```

Bước 2: File AndroidManifest.xml chứa quyền truy nhập đến mạng Internet như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.phpmysql" >
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.phpmysql.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

```

Bước 3: Tạo lớp xử lý đăng nhập trên giao diện SigninActivity.java có nội dung:

```

public class SigninActivity extends AsyncTask{
    private TextView statusField,roleField;
    private Context context;
    private int byGetOrPost = 0;
    // flag =0 sử dụng phương thức GET, flag =1 sử dụng phương thức POST, mặc định là GET

    public SigninActivity(Context context,TextView statusField,TextView roleField,int flag) {
        this.context = context;
    }
}

```

```
this.statusField = statusField; //TextView hiển thị trạng thái đăng nhập
this.roleField = roleField; //TextView hiển thị quyền Admin
byGetOrPost = flag;
}

protected void onPreExecute(){ // không dùng đến
}

//Phương thức kết nối đến dịch vụ web và nhận phản hồi từ nó trả về một xâu kí tự
@Override
protected String doInBackground(String... arg0) {
    if(byGetOrPost == 0){ // sử dụng phương thức Get
        try{
            String username = (String)arg0[0];
            String password = (String)arg0[1];
            String link =
                "http://myphpmysqlweb.hostei.com/login.php?username="+username+"&
                password="+password;
            URL url = new URL(link);
            HttpClient client = new DefaultHttpClient();
            HttpGet request = new HttpGet();
            request.setURI(new URI(link));
            HttpResponse response = client.execute(request);
            BufferedReader in = new BufferedReader(new
                InputStreamReader(response.getEntity().getContent()));
            //Tạo vùng đệm lưu dữ liệu phản hồi
            StringBuffer sb = new StringBuffer("");
            String line="";
            // Đọc phản hồi từ Server
            while ((line = in.readLine()) != null) {
                sb.append(line);
                break;
            }
            // Đóng luồng dữ liệu
            in.close();
            return sb.toString();
        } catch(Exception e){
            return new String("Exception: " + e.getMessage());
        }
    } else{ // ngược lại sử dụng phương thức POST
        try{
            String username = (String)arg0[0];
            String password = (String)arg0[1];
            String link="http://myphpmysqlweb.hostei.com/loginpost.php";
            String data = URLEncoder.encode("username", "UTF-8") + "=" +
                URLEncoder.encode(username, "UTF-8");
            data += "&" + URLEncoder.encode("password", "UTF-8") + "=" +
                URLEncoder.encode(password, "UTF-8");
            URL url = new URL(link);
            URLConnection conn = url.openConnection();
            conn.setDoOutput(true);
```

```

// tạo đối tượng ghi dữ liệu
OutputStreamWriter wr = new OutputStreamWriter(conn.getOutputStream());
wr.write( data );
wr.flush();
BufferedReader reader = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
StringBuilder sb = new StringBuilder();
String line = null;

// Đọc phản hồi từ Server
while((line = reader.readLine()) != null) {
    sb.append(line);
    break;
}
return sb.toString();
} catch(Exception e){
    return new String("Exception: " + e.getMessage());
}
}

// Sau khi thực thi phương thức kết nối đến Server thì hiển thị kết quả lên các
TextView
@Override
protected void onPostExecute(String result){
    this.statusField.setText("Đăng nhập thành công");
    this.roleField.setText(result);
}
}

```

Bước 4: File MainActivity.java có nội dung như sau:

```

public class MainActivity extends Activity {
    private EditText usernameField,passwordField;
    private TextView status,role,method;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        usernameField = (EditText)findViewById(R.id.editText1);
        passwordField = (EditText)findViewById(R.id.editText2);
        status = (TextView)findViewById(R.id.textView6);
        role = (TextView)findViewById(R.id.textView7);
        method = (TextView)findViewById(R.id.textView9);
    }
    // Được gọi khi Click lên Button có nhãn Login - Get
    public void login(View view){
        String username = usernameField.getText().toString();
        String password = passwordField.getText().toString();
        method.setText("Get Method");
        new SigninActivity(this,status,role,0).execute(username,password);
    }
}

```

```
        }
    // Được gọi khi Click lên Button có nhan Login Post
    public void loginPost(View view){
        String username = usernameField.getText().toString();
        String password = passwordField.getText().toString();
        method.setText("Post Method");
        new SigninActivity(this,status,role,1).execute(username,password);
    }
}
```

4.4 Xử lý dữ liệu với Json trong Android

JSON (JavaScript Object Notation) là định dạng dữ liệu độc lập và là sự chuyển đổi tốt nhất của định dạng XML. Phần này sẽ trình bày cú pháp của file JSON và cách trích xuất các thông tin cần thiết từ nó.

Android cung cấp các lớp khác nhau để làm việc trên dữ liệu JSON là SONArray, JSONObject, JSONStringer và JSONTokenizer.

Một file JSON có định dạng gồm các trường thông tin, mỗi trường gồm các thuộc tính khác nhau, chẳng hạn file JSON gồm 3 trường như sau:

```
{
    "sys": {
        "country": "GB",
        "sunrise": 1381107633,
        "sunset": 1381149604
    },
    "weather": [
        {
            "id": 711,
            "main": "Smoke",
            "description": "smoke",
            "icon": "50n"
        }
    ],
    "main": {
        "temp": 28.5
    }
}
```

```

    "temp":304.15,
    "pressure":1009,
}
}

```

Một file JSON gồm nhiều thành phần, bảng sau đây định nghĩa các thành phần được ký hiệu như sau:

Các thành phần	Mô tả
[Biểu diễn một mảng
{	Biểu diễn một đối tượng
key/value	Dại diện cho một cặp thuộc tính và giá trị của đối tượng; các giá trị có thể nhận kiểu khác nhau như String, int, float,...

Bảng 4.1: Ký hiệu các thành phần trong file JSON

Khởi tạo đối tượng file JSON và chỉ định đối tượng nhận dữ liệu từ nó:

```

String in;
JSONObject reader = new JSONObject(in);

```

Đọc dữ liệu từ các đối tượng trong file JSON:

```

// Đọc đối tượng có tên là sys
JSONObject sys = reader.getJSONObject("sys");
// Lấy giá trị của thuộc tính country
country = sys.getString("country");
// Đọc đối tượng có tên là main
JSONObject main = reader.getJSONObject("main");
// Lấy giá trị của thuộc tính temp
temperature = main.getString("temp");

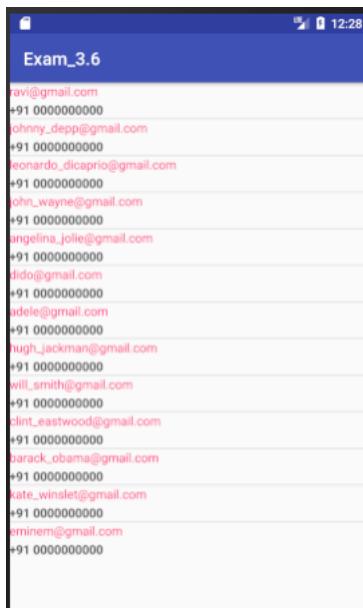
```

Một số phương thức của lớp JSONObject và ý nghĩa của chúng được liệt kê trong bảng 4.2

Thí dụ 3.6. Thiết kế ứng dụng Android cho phép truy thông tin về các contact chứa trong một file JSON từ Server trên mạng tại địa chỉ <http://api.androidhive.info/contacts/> như sau:

Phương thức	Mô tả
get(String name)	Trả về giá trị của thuộc tính được chỉ định
getBoolean(String name)	Trả về giá trị kiểu Boolean của thuộc tính được chỉ định
getDouble(String name)	Trả về giá trị kiểu Double của thuộc tính được chỉ định
getInt(String name)	Trả về giá trị kiểu Int của thuộc tính được chỉ định
getLong(String name)	Trả về giá trị kiểu Long của thuộc tính được chỉ định
length()	Trả về số lượng cặp thuộc tính/giá trị của đối tượng
names()	Trả về một mảng chứa tên các thuộc tính của đối tượng

Bảng 4.2: Các phương thức của lớp JSONObject



Hình 4.2: Ứng dụng truy xuất file JSON

Cấu trúc file JSON chứa một mảng các contact, mỗi contact gồm các thông tin có dạng như sau:

```
{
  "contacts": [
    {
      "id": "c200",
      "name": "Ravi Tamada",
      "email": "ravi@gmail.com",
      "phone": "+91 0000000000"
    }
  ]
}
```

```

    "address": "xx-xx-xxxx,x - street, x - country",
    "gender" : "male",
    "phone": {
        "mobile": "+91 0000000000",
        "home": "00 000000",
        "office": "00 000000"
    }
},
{
    "id": "c201",
    "name": "Johnny Depp",
    "email": "johnny_depp@gmail.com",
    "address": "xx-xx-xxxx,x - street, x - country",
    "gender" : "male",
    "phone": {
        "mobile": "+91 0000000000",
        "home": "00 000000",
        "office": "00 000000"
    }
},
...
]
}

```

Tạo project Android và thực hiện các bước sau:

Bước 1: Tạo file res/layout/list_item.xml chứa thông tin về một contact gồm email và số điện thoại:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="@dimen/activity_horizontal_margin">
    <TextView
        android:id="@+id/email"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="2dip"
        android:textColor="@color/colorAccent" />
    <TextView
        android:id="@+id/mobile"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#5d5d5d"
        android:textStyle="bold" />
</LinearLayout>

```

Bước 2: Tạo file res/layout/activity_main.xml định nghĩa giao diện chính của ứng dụng:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.tutorialspoint7.myapplication.MainActivity">
    <ListView
        android:id="@+id/list"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
</RelativeLayout>
```

Bước 3: Tạo file AndroidManifest.xml cho phép kết nối Internet có nội dung như sau:

```
<uses-permission android:name="android.permission.INTERNET" />
```

Bước 4: File MainActivity.java chứa chương trình chính có nội dung:

```
public class MainActivity extends AppCompatActivity {
    private String TAG = MainActivity.class.getSimpleName();
    private ListView lv;
    //Mảng chứa danh sách contact, mỗi phần tử là một cặp giá trị kiểu xâu
    ArrayList<HashMap<String, String>> contactList;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //Nội dung của ListView trên giao diện chính là danh sách các contact
        contactList = new ArrayList<>();
        lv = (ListView) findViewById(R.id.list);
        //Triệu gọi lớp GetContacts để lấy danh sách Contact từ máy chủ trên mạng
        new GetContacts().execute();
    }
    // Khai báo lớp GetContact
    private class GetContacts extends AsyncTask<Void, Void, Void> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            Toast.makeText(MainActivity.this, "Json Data is
                downloading", Toast.LENGTH_LONG).show();
        }
        @Override
        protected Void doInBackground(Void... arg0) {
```

```

//Khởi tạo đối tượng từ lớp HttpHandler để kết nối đến Server
HttpHandler sh = new HttpHandler();
// Gửi yêu cầu đến Server với url và nhận phản hồi
String url = "http://api.androidhive.info/contacts/";
String jsonStr = sh.makeServiceCall(url);
Log.e(TAG, "Response from url: " + jsonStr);
if (jsonStr != null) {
    try {
        JSONObject jsonObj = new JSONObject(jsonStr);
        // Lấy về mảng các node (mỗi node là một object) trong file JSON
        JSONArray contacts = jsonObj.getJSONArray("contacts");
        //Duyệt từng Contact trong file
        for (int i = 0; i < contacts.length(); i++) {
            JSONObject c = contacts.getJSONObject(i);
            String id = c.getString("id");
            String name = c.getString("name");
            String email = c.getString("email");
            String address = c.getString("address");
            String gender = c.getString("gender");
            // Đọc node Phone của một contact trong mảng
            JSONObject phone = c.getJSONObject("phone");
            String mobile = phone.getString("mobile");
            String home = phone.getString("home");
            String office = phone.getString("office");
            // Lưu dữ liệu cho một contact trong mảng
            HashMap<String, String> contact = new HashMap<>();
            // Thêm các cặp thuộc tính của từng node con vào mảng HashMap
            contact.put("id", id);
            contact.put("name", name);
            contact.put("email", email);
            contact.put("mobile", mobile);
            // Thêm contact vào danh sách
            contactList.add(contact);
        }
    } catch (final JSONException e) {
        // Nếu có lỗi xảy ra thì tạo ra một Thread thông báo lỗi
        Log.e(TAG, "Json parsing error: " + e.getMessage());
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                Toast.makeText(getApplicationContext(),
                    "Json parsing error: " + e.getMessage(),
                    Toast.LENGTH_LONG).show();
            }
        });
    }
}
// Nếu kết nối đến Server không thành công thì tạo ra một Thread thông báo lỗi
} else {
    Log.e(TAG, "Couldn't get json from server.");
    runOnUiThread(new Runnable() {
        @Override
        public void run() {

```

```
        Toast.makeText(getApplicationContext(),
                "Couldn't get json from server. Check LogCat for possible
                errors!",
                Toast.LENGTH_LONG).show();
    }
}
);
}
return null;
}
@Override
protected void onPostExecute(Void result) {
    super.onPostExecute(result);
//Hiển thị danh sách contact lên ListView
    ListAdapter adapter = new SimpleAdapter(MainActivity.this, contactList,
        R.layout.list_item, new String[]{"email", "mobile"},
        new int[]{R.id.email, R.id.mobile});
    lv.setAdapter(adapter);
}
}
```

Bước 5: Xây dựng lớp `HttpHandler.java` kết nối đến Server chứa file JSON như sau:

```
public class HttpHandler {  
    private static final String TAG = HttpHandler.class.getSimpleName();  
    public HttpHandler() {  
    }  
    //Thực hiện kết nối đến Server tại địa chỉ được truyền vào  
    public String makeServiceCall(String reqUrl) {  
        String response = null;  
        try {  
            URL url = new URL(reqUrl);  
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();  
            conn.setRequestMethod("GET");  
            //Đọc phản hồi từ Server  
            InputStream in = new BufferedInputStream(conn.getInputStream());  
            response = convertStreamToString(in);  
        } catch (MalformedURLException e) {  
            Log.e(TAG, "MalformedURLException: " + e.getMessage());  
        } catch (ProtocolException e) {  
            Log.e(TAG, "ProtocolException: " + e.getMessage());  
        } catch (IOException e) {  
            Log.e(TAG, "IOException: " + e.getMessage());  
        } catch (Exception e) {  
            Log.e(TAG, "Exception: " + e.getMessage());  
        }  
        return response;  
    }  
    // Hàm chuyển dữ liệu phản hồi từ Server thành một xâu kí tự
```

```
private String convertStreamToString(InputStream is) {
    BufferedReader reader = new BufferedReader(new InputStreamReader(is));
    StringBuilder sb = new StringBuilder();
    String line;
    try {
        while ((line = reader.readLine()) != null) {
            sb.append(line).append('\n');
        }
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        try {
            is.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    return sb.toString();
}
```

Câu hỏi và bài tập

1. Nêu cách tích hợp mã nguồn PHP vào ứng dụng Android và cho biết ý nghĩa của việc tích hợp?
2. Phân biệt phương thức GET và phương thức POST khi kết nối trang web PHP tới CSDL MySQL?
3. Định dạng dữ liệu JSON có ưu, nhược điểm gì so với dữ liệu dạng XML?
4. Cho biết cấu trúc của một file dữ liệu JSON?
5. Khi kết nối Internet cần khai báo quyền truy cập trong file.

Chương 5

Lưu trữ dữ liệu trong Android

5.1	Lưu trữ dữ liệu với Shared preferences	161
5.2	Lưu trữ dữ liệu với hệ thống file trong Android	167
5.3	Lưu trữ dữ liệu với SQLite	173

Các ứng dụng Android trong quá trình thực thi đòi hỏi phải truy xuất và cập nhật đến các dữ liệu trong hệ thống. Các dữ liệu này phải được lưu trữ theo cách nào đó để thuận tiện cho việc sử dụng lại. Chẳng hạn, thông tin về một địa chỉ liên lạc (contact) gồm tên, địa chỉ, số điện thoại, v.v... Chương này sẽ trình bày một số kỹ thuật lưu trữ dữ liệu mà Android sử dụng là Shared preferences, lưu trữ trên file văn bản và cơ sở dữ liệu SQL.

5.1 Lưu trữ dữ liệu với Shared preferences

Việc lưu trữ dữ liệu sử dụng *Shared preferences* có thể được mô tả minh họa qua ví dụ sau: giả sử người dùng đang chơi một trò chơi trên Android, trước khi chơi trò chơi người chơi đã lựa chọn các thông số của trò chơi như độ sáng trong trò chơi, mức độ âm lượng và độ khó. Sau khi chơi xong người chơi đã tắt trò chơi và có thể tiếp tục chơi vào ngày hôm sau. Shared Preferences cho phép lưu lại các thông số người dùng đã thiết lập trước đó, khi người

dùng chơi lại các thiết lập trước đó có thể tự động được cập nhật mà không cần phải thiết lập lại.

Shared Preferences bản chất là một file xml của ứng dụng. Nó lưu các dữ liệu thô dưới dạng các cặp khóa và giá trị (key-value) vào các tập tin của ứng dụng. Người lập trình cũng có thể chọn một chế độ lưu trữ riêng tư (PRIVATE) mà các ứng dụng khác không thể truy cập vào các tập tin này, do đó cách lưu trữ này là khá an toàn.

5.1.1 Ghi dữ liệu vào Shared Preferences

Để lưu ghi dữ liệu vào file lưu trữ của ứng dụng, người lập trình cần thực hiện lấy về đối tượng SharedPreferences bằng cách gọi tới phương thức *getSharedPreferences()*. Phương thức này gồm có 2 đối số là tên file XML và chế độ lưu trữ (thường sử dụng PRIVATE)

```
SharedPreferences shPre =  
    this.getSharedPreferences("file_name",  
        Context.MODE_PRIVATE);
```

Tạo đối tượng Editor để sửa đổi dữ liệu như sau:

```
SharedPreferences.Editor editor = shPre.edit();
```

Sau khi khởi tạo được đối tượng Editor người lập trình có thể sử dụng các phương thức tương ứng với kiểu dữ liệu muốn ghi vào file như:

- *putString(String key, string value); // ghi dữ liệu kiểu xâu*
- *.putInt(String key, int value); // ghi dữ liệu kiểu int*
- *putBoolean(String key, boolean value); // ghi dữ liệu kiểu boolean*
- *putFloat(String key, float value) // ghi dữ liệu kiểu float*
- *putLong(String key, long value) // ghi dữ liệu kiểu long*

Tham số thứ nhất luôn là kiểu String.

Thí dụ người lập trình thực hiện ghi dữ liệu như sau:

```
editor.putString("name","Nguyen Van A");
editor.putBoolean("isNam",true);
```

Để thực hiện cập nhật dữ liệu vào file sau khi thực hiện các phương thức put thì phải gọi phương thức apply() hoặc commit() của đối tượng Editor để ghi dữ liệu vào file:

```
editor.apply(); hoặc editor.commit();
```

5.1.2 Đọc dữ liệu từ Shared Preferences

Để đọc được dữ liệu từ SharedPreferences người lập trình tạo đối tượng SharedPreferences giống như ghi dữ liệu:

```
SharedPreferences shPre =
this.getSharedPreferences("file_name",
Context.MODE_PRIVATE);
```

```
SharedPreferences.Editor editor = shPre.edit();
```

Sau đó sử dụng các phương thức đọc tương ứng với kiểu dữ liệu của đối tượng Editor:

- int getInt(String key, int defaultValue);
- String getString(String key, String defaultValue);
- boolean getBoolean(String key, boolean defaultValue);
- ...

5.1.3 Xoá dữ liệu SharedPreferences

Phương remove() của đối tượng Editor được sử dụng để xoá từng cặp dữ liệu trong SharedPreferences.

Thí dụ:

```
editor.remove("name"); // xoá cặp có key là name
```

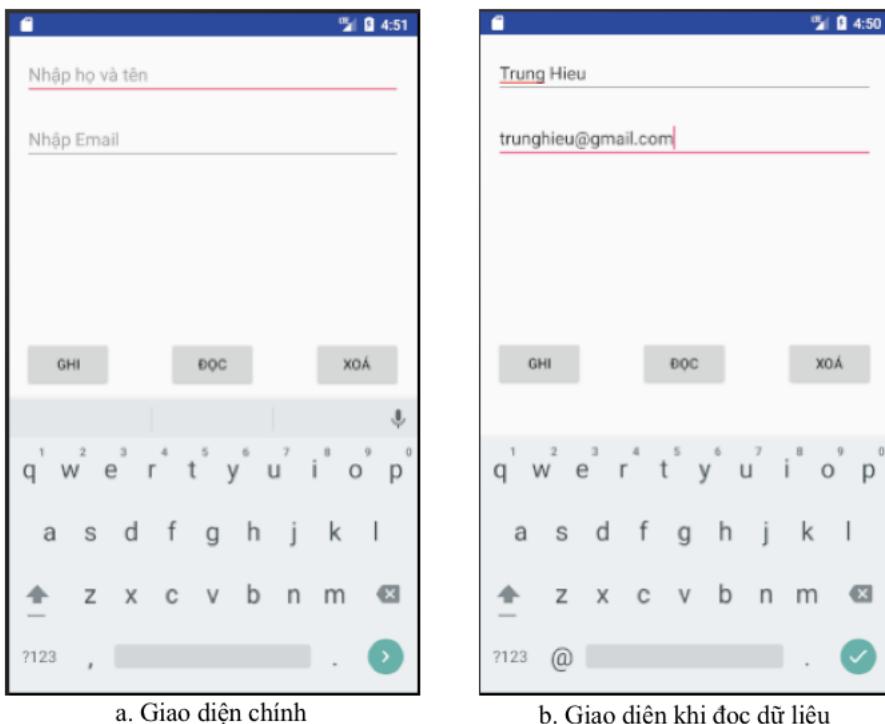
```
editor.remove("email"); // xoá cặp có key là email
editor.commit(); // thực hiện xoá
```

Phương thức *clear()* của đối tượng Editor cho phép xoá toàn bộ dữ liệu trong SharedPreferences.

Thí dụ:

```
editor.clear(); editor.commit(); // thực hiện xoá
```

Thí dụ 4-1. Thiết kế giao diện như hình 5.1 (a). Xây dựng chương



Hình 5.1: Giao diện ứng dụng sử dụng Shared Preferences

trình để khi người dùng nhập dữ liệu vào 2 đối tượng EditText rồi nhấn nút Ghi thì dữ liệu được ghi vào file Myprefs.xml trên hệ thống, khi nhấn nút Đọc thì đọc dữ liệu từ file Myprefs.xml hiển thị lên các EditText tương ứng, khi nhấn nút Xoá thì xoá dữ liệu trên các EditText. Nếu người dùng tắt ứng dụng hoàn toàn và mở lại nó thì giá trị vừa nhập sẽ lưu lại trên các EditText.

Tạo project mới và soạn thảo nội dung các file như sau:

Bước 1. Soạn thảo file activiy_main.xml có nội dung như sau:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin" >
    <Button
        android:id="@+id/btnSave"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:onClick="Save"
        android:text="Ghi" />
    <Button
        android:id="@+id/btnRetr"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:onClick="Get"
        android:text="Đọc" />
    <Button
        android:id="@+id/btnClear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/etEmail"
        android:layout_centerVertical="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:onClick="clear"
        android:text="Xoá" />
    <EditText
        android:id="@+id/etEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Nhập Email"
        android:inputType="textEmailAddress"
        android:layout_below="@+id/etName"
        android:layout_marginTop="20dp"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />
    <EditText
        android:id="@+id/etName"
        android:layout_width="match_parent"
```

```

    android:layout_height="wrap_content"
    android:ems="10"
    android:hint="Nhập họ và tên"
    android:inputType="text"
    android:layout_alignParentTop="true"
    android:layout_alignLeft="@+id/etEmail"
    android:layout_alignStart="@+id/etEmail" />
</RelativeLayout>

```

Bước 2: Soạn thảo file MainActivity.java có nội dung như sau:

```

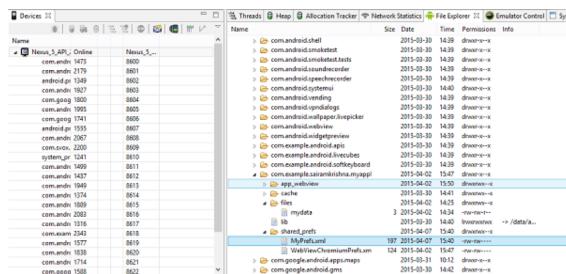
public class MainActivity extends Activity {
    SharedPreferences sharedpreferences;
    TextView name;
    TextView email;
    public static final String mypreference = "mypref";
    public static final String Name = "nameKey";
    public static final String Email = "emailKey";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        name = (TextView) findViewById(R.id.etName);
        email = (TextView) findViewById(R.id.etEmail);
        sharedpreferences = getSharedPreferences(mypreference,
            Context.MODE_PRIVATE);
        //Nếu dữ liệu đã tồn tại trước đó thì hiển thị lên các EditText
        if (sharedpreferences.contains(Name)) {
            name.setText(sharedpreferences.getString(Name, ""));
        }
        if (sharedpreferences.contains(Email)) {
            email.setText(sharedpreferences.getString(Email, ""));
        }
    }
    //Phương thức ghi dữ liệu vào Shared preferences
    public void Save(View view) {
        String n = name.getText().toString();
        String e = email.getText().toString();
        SharedPreferences.Editor editor = sharedpreferences.edit();
        editor.putString(Name, n);
        editor.putString(Email, e);
        editor.commit();
    }
    //Phương thức xoá dữ liệu trên các EditText
    public void clear(View view) {
        name = (TextView) findViewById(R.id.etName);
        email = (TextView) findViewById(R.id.etEmail);
        name.setText("");
        email.setText("");
    }
}

```

```
//Phương thức đọc dữ liệu từ Shared preferences
public void Get(View view) {
    name = (TextView) findViewById(R.id.etName);
    email = (TextView) findViewById(R.id.etEmail);
    sharedpreferences = getSharedPreferences(mypreference,
        Context.MODE_PRIVATE);
    if (sharedpreferences.contains(Name)) {
        name.setText(sharedpreferences.getString(Name, ""));
    }
    if (sharedpreferences.contains>Email)) {
        email.setText(sharedpreferences.getString>Email, ""));
    }
}
```

Bước 3: Chạy ứng dụng và kiểm tra kết quả trong hình 5.2.



Hình 5.2: Xem file Shared Preferences trên thiết bị

5.2 Lưu trữ dữ liệu với hệ thống file trong Android

Trong trường hợp cần lưu lại dữ liệu phức tạp hơn (khó có thể lưu lại dạng key-value như trong shared preference), người lập trình có thể sử dụng các file có định dạng văn bản như .txt để lưu dữ liệu. Trong Android, để làm việc (nhập/xuất) với file người lập trình có thể dùng các lớp của gói java.io. Phần sau đây sẽ trình bày cách làm việc với file nằm trong bộ nhớ (cả bộ nhớ trong và bộ nhớ ngoài).

5.2.1 Làm việc với file trong bộ nhớ trong

Lớp FileOutputStream cho phép truy xuất đến hệ thống file của Android. Lớp OutputStreamWriter cung cấp các phương thức cho việc ghi dữ liệu vào file. Việc ghi dữ liệu vào file được thực hiện theo trình tự các lệnh như sau:

```
// Tạo đối tượng từ lớp FileOutputStream
FileOutputStream fOut =
openFileOutput("textfile.txt", MODE_PRIVATE);
// Tạo đối tượng từ lớp OutputStreamWriter
OutputStreamWriter osw = new OutputStreamWriter(fOut);
//--- Ghi dữ liệu dạng String vào file ---
osw.write("String write to file");
// Đẩy hết dữ liệu từ bộ đệm vào file sau mỗi lần ghi
osw.flush();
// Đóng file
osw.close();
```

Sau khi file textfile.txt được tạo ra, nó sẽ được lưu vào thư mục dành riêng cho ứng dụng tại đường dẫn `/data/data/package-name/files`.

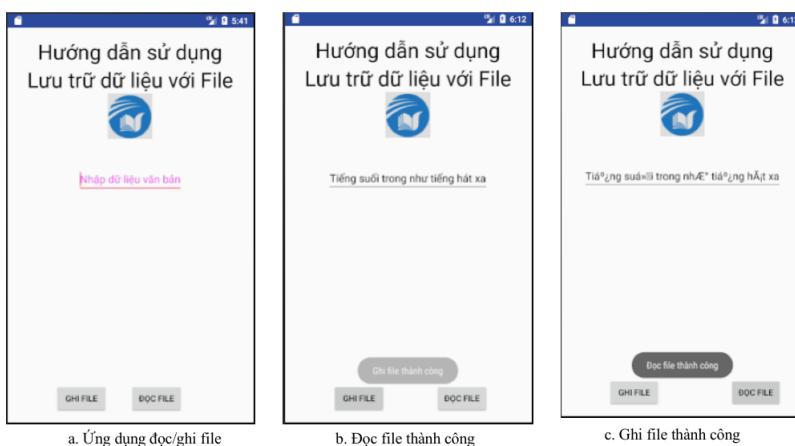
Việc đọc dữ liệu từ file cũng thao tác tương tự như việc ghi dữ liệu nhưng cần tạo đối tượng InputStreamReader từ luồng nhập liệu (FileInputStream) và tiến hành đọc dữ liệu bằng phương thức `read()`. Mỗi lần đọc hệ thống sẽ đọc số byte là `READ_BLOCK_SIZE` và lưu vào bộ đệm (mảng byte). Dữ liệu đọc được từ bộ đệm sẽ được chuyển thành dạng xâu và nối thêm vào biến `s`. Quá trình được lặp lại cho đến khi hết nội dung của file:

```
FileInputStream fIn = openFileInput("textfile.txt");
InputStreamReader isr = new InputStreamReader(fIn);
char[] inputBuffer = new char[READ_BLOCK_SIZE];
String s = "";
int charRead;
while ((charRead = isr.read(inputBuffer)) > 0)
{
    String readString = String.valueOf(inputBuffer, 0, charRead);
    s += readString;
    inputBuffer = new char[READ_BLOCK_SIZE];
}
```

```
textBox.setText(s);
```

Ngoài ra người lập trình có thể sử dụng trực tiếp phương thức write() của lớp FileOutputStream và phương thức read của lớp FileInputStream để thao tác với file. Điều này sẽ được minh họa qua thí dụ 5-2 sau đây.

Thí dụ 5-2. Thiết kế giao diện như hình 5.3 (a). Lập trình để khi



Hình 5.3: Giao diện Thí dụ 5-2 (thao tác với File)

người dùng nhấn vào nút Ghi File thì sẽ ghi giá trị được nhập trong EditText vào file văn bản có tên mydata, nếu việc ghi thành công thì hiện thông điệp Toast có nội dung “Ghi dữ liệu thành công” như hình 5.3 (b), khi người dùng nhấn nút Đọc File thì đọc nội dung của file hiển thị lên EditText, đồng thời hiển thị thông điệp Toast có nội dung “Đọc file thành công” như hình 5-3 (c).

Tạo project mới và thực hiện các bước sau:

Bước 1. File layout activity_main.xml có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
```

```
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">
<TextView android:text="Hướng dẫn sử dụng" android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/textview"
    android:textSize="35dp"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Lưu trữ dữ liệu với File"
    android:id="@+id/textView"
    android:layout_below="@+id/textview"
    android:layout_centerHorizontal="true"
    android:textColor="#ff7aff24"
    android:textSize="35dp" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Ghi File"
    android:id="@+id/button"
    android:layout_alignParentBottom="true"
    android:layout_alignLeft="@+id/textView"
    android:layout_alignStart="@+id/textView" />
<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/editText"
    android:hint="Nhập dữ liệu văn bản"
    android:focusable="true"
    android:textColorHighlight="#ff7eff15"
    android:textColorHint="#ffff25e6"
    android:layout_below="@+id/imageView"
    android:layout_alignRight="@+id/textView"
    android:layout_alignEnd="@+id/textView"
    android:layout_marginTop="42dp"
    android:layout_alignLeft="@+id/imageView"
    android:layout_alignStart="@+id/imageView" />
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/imageView"
    android:src="@drawable/abc"
    android:layout_below="@+id/textView"
    android:layout_centerHorizontal="true" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Đọc File"
```

```
    android:id="@+id/button2"
    android:layout_alignTop="@+id/button"
    android:layout_alignRight="@+id/editText"
    android:layout_alignEnd="@+id/editText" />
</RelativeLayout>
```

Bước 2: File MainActivity.java có nội dung như sau:

```
public class MainActivity extends Activity {
    Button b1,b2;
    TextView tv;
    EditText ed1;
    String data;
    private String file = "mydata"; //tên file
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        b1=(Button)findViewById(R.id.button);
        b2=(Button)findViewById(R.id.button2);
        ed1=(EditText)findViewById(R.id.editText);
        tv=(TextView)findViewById(R.id.textView2);
    // sự kiện khi nhấn nút Ghi File đểghi file
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                data=ed1.getText().toString();
                try {
                    FileOutputStream fOut = openFileOutput(file,MODE_WORLD_READABLE);
                    fOut.write(data.getBytes());
                    fOut.close();
                    Toast.makeText(getApplicationContext(),"file
                    saved",Toast.LENGTH_SHORT).show();
                }
                catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    // sự kiện khi nhấn nút Đọc File đểđọc file
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    FileInputStream fin = openFileInput(file);
                    int c;
                    String temp="";
                    while( (c = fin.read()) != -1){
                        temp = temp + Character.toString((char)c);
                    }
                    ed1.setText(temp);
                }
            }
        });
    }
}
```

```

        Toast.makeText(getApplicationContext(),"file
            read",Toast.LENGTH_SHORT).show();
    }
    catch(Exception e){
    }
}
});
}}
```

5.2.2 Làm việc với file trong bộ nhớ ngoài

Các file lưu trữ trong bộ nhớ trong chỉ có thể được truy cập bởi ứng dụng tạo ra nó. Bên cạnh đó dung lượng lưu trữ của bộ nhớ trong thường hạn chế hơn bộ nhớ ngoài (thường dùng thẻ nhớ - SDCard). Vì vậy, trong trường hợp người lập trình muốn chia sẻ thông tin lưu trữ với các ứng dụng khác thì có thể sử dụng bộ nhớ ngoài. Cách thao tác với file trong bộ nhớ ngoài hoàn toàn tương tự với file trong bộ nhớ trong nhưng chỉ khác phần phân định vị trí file trong đối tượng của lớp FileInputStream và FileOutputStream.

Để truy cập đến bộ nhớ ngoài người lập trình phải khai báo quyền truy cập trong file manifest như sau:

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

Các bước thực hiện ghi file như sau:

```

// Lấy thư mục lưu trữ của bộ nhớ ngoài
File sdCard = Environment.getExternalStorageDirectory();
// Lấy đường dẫn và tạo thư mục MyFiles trong bộ nhớ ngoài
File directory =
new File(sdCard.getAbsolutePath() + "/MyFiles");
directory.mkdirs();
// Tạo file
File file = new File(directory, "textfile.txt");
// Khai báo đối tượng FileOutputStream
FileOutputStream fOut = new FileOutputStream(file);
```

Các bước thực hiện đọc file tương tự như sau:

```
File sdCard = Environment.getExternalStorageDirectory();
```

```
File directory = new File (sdCard.getAbsolutePath() + "/MyFiles");
File file = new File(directory, "textfile.txt");
FileInputStream fIn = new FileInputStream(file);
InputStreamReader isr = new InputStreamReader(fIn);
```

Lúc này file textfile.txt sẽ được lưu trữ tại đường dẫn /MyFiles/-textfile.txt

5.3 Lưu trữ dữ liệu với SQLite

SQLite là cơ sở dữ liệu (CSDL) quan hệ mã nguồn mở được tích hợp sẵn trên Android, người lập trình có thể sử dụng mà không cần sử dụng trình điều khiển kết nối ứng dụng (driver) đến nó.

Lớp android.database.sqlite cung cấp các lớp quản lý CSDL SQLite. Để tạo CSDL sử dụng cú pháp sau:

```
SQLiteDatabase mydatabase =
    openOrCreateDatabase("database name", MODE_PRIVATE, null);
```

Các phương thức phổ biến của lớp SQLiteDatabase như bảng 5.1.

Để thực thi một câu truy vấn SQL sử dụng phương thức execSQL() theo cú pháp:

execSQL(String sql, Object[] bindArgs)

Trong đó, tham số sql là câu truy vấn, bindArgs là các giá trị tham số trong trường hợp muốn thực hiện câu truy vấn select, delete hay update bản ghi với điều kiện nào đó. Thí dụ, người lập trình thực hiện câu lệnh truy vấn như sau:

```
mydatabase.execSQL("CREATE TABLE IF NOT EXISTS User(id integer primary key
    auto_increment, Username VARCHAR, Password VARCHAR);");
mydatabase.execSQL("INSERT INTO User VALUES('admin', 'admin');");
```

Đọc các bản ghi từ một bảng CSDL cần sử dụng lớp Cursor với cú pháp như sau:

Phương thức	Mô tả
openDatabase(String path, SQLiteDatabase.CursorFactory factory, int flags, DatabaseErrorHandler errorHandler)	Mở một CSDL đã tồn tại với giá trị cờ phù hợp. Trong đó, factory cho phép trả về một lớp con của lớp Cursor khi thực thi câu truy vấn, errorHandler là một trình bắt lỗi mở CSDL, flag là cờ cho biết chế độ truy cập đến file, thông thường giá trị cờ là OPEN_READWRITE hoặc OPEN_READONLY
openDatabase(String path, SQLiteDatabase.CursorFactory factory, int flags)	Tương tự như phương thức mở CSDL trên nhưng không sử dụng trình điều khiển lỗi.
openOrCreateDatabase(String path, SQLiteDatabase.CursorFactory factory)	Mở và tạo mới CSDL nếu nó chưa tồn tại
openOrCreateDatabase(File file, SQLiteDatabase.CursorFactory factory)	Mở CSDL với đường dẫn đến vị trí file CSDL.

Bảng 5.1: Các phương thức mở một CSDL SQLite

```
// Chọn tất cả các bảng ghi từ bản CSDL
Cursor resultSet = mydatabase.rawQuery("Select * from User", null);
// Dưa con trả về đầu bảng
resultSet.moveToFirst();
// Duyệt từng bản ghi
while(resultSet.isAfterLast() == false){
    // Lấy giá trị cột Username của bản ghi hiện tại
    String username = resultSet.getColumnIndex("Username")
    // Lấy giá trị cột Password của bản ghi hiện tại
    String password = resultSet.getColumnIndex("Password")
    // Chuyển đến bản ghi tiếp theo
    resultSet.next();
}
```

Chèn một bản ghi mới vào CSDL với tham số truyền vào:

```
SQLiteDatabase db = this.getWritableDatabase();
ContentValues contentValues = new ContentValues();
contentValues.put("Username", "superadmin");
contentValues.put("Password", "123");
// Thực hiện truy vấn
db.insert("contacts", null, contentValues);
```

Cập nhật bản ghi từ cơ sở dữ với tham số truyền vào:

```

Integer id =5;
SQLiteDatabase db = this.getWritableDatabase();
ContentValues contentValues = new ContentValues();
contentValues.put("Username", "superadmin");
contentValues.put("Password", "12345");
//Thực hiện câu truy vấn
db.update("contacts", contentValues, "id = ? ", new String[] {
    Integer.toString(id) } );

```

Các phương thức phổ biến của lớp Cursor như bảng 5.2.

Phương thức	Mô tả
getColumnCount()	Trả về một số nguyên là tổng số cột của bảng
getColumnIndex(String columnName)	Trả về một số nguyên là địa chỉ của cột có tên được chỉ định trong bảng
getColumnName(int columnIndex)	Lấy về tên cột trong bảng có địa chỉ columnIndex
getColumnNames()	Trả về một mảng gồm tên các cột trong bảng
getCount()	Trả về tổng số dòng trong bảng
getPosition()	Trả về vị trí hiện hành của con trỏ trong bảng
isClosed()	Trả về giá trị true nếu con trỏ đã đóng và fasle nếu ngược lại

Bảng 5.2: Các phương thức của lớp Cursor

Để quản lý mọi hoạt động của CSDL lớp SQLiteOpenHelper sẽ tự động quản lý việc tạo và cập nhật CSDL.

```
public class DBHelper extends SQLiteOpenHelper {
```

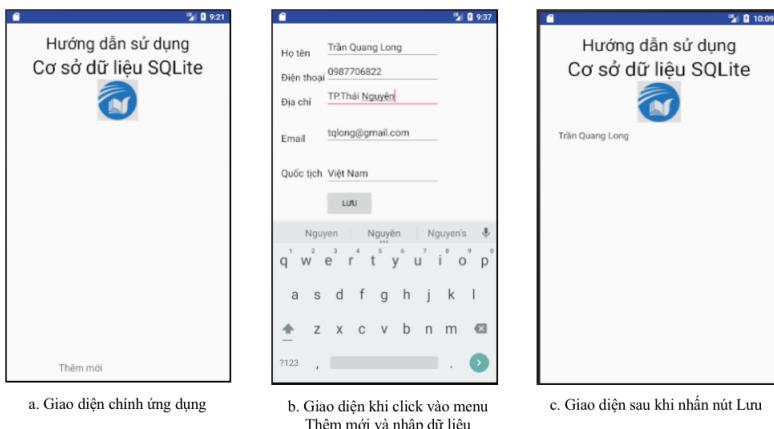
```

public DBHelper(){
    super(context,DATABASE_NAME,null,1);
}
public void onCreate(SQLiteDatabase db) {}
public void onUpgrade(SQLiteDatabase database, int oldVersion, int newVersion) {}
}

```

Thí dụ 5-3 dưới đây sẽ minh họa các thao tác với CSDL SQLite.

Thí dụ 5-3. Thiết giao diện ứng dụng gồm giao diện chính như hình 5.4 (a) gồm một menu option có nhãn “Thêm mới”. Khi người dùng chọn menu thì hiển thị giao diện cho phép nhập dữ liệu về một contact như hình 5.4. (b), giao diện này chứa 2 menu option là Edit _ Contact và Delete _ Contact, khi người dùng nhập các thông tin về một contact và nhấn nút Lưu thì lưu các thông tin về contact vào CSDL đồng thời trả lại màn hình chính và danh sách các contact đã nhập sẽ hiển thị trên màn hình chính như hình 5.4 (c)



Hình 5.4: Giao diện Thí dụ 5-3.

Tạo project mới và thực hiện các bước sau:

Bước 1: Tạo file layout activity_display_contact.xml định nghĩa giao diện nhập liệu cho contact như sau:

```

<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"

```

```
    android:id="@+id/scrollView1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    tools:context=".DisplayContact" >
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="370dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin">
<EditText
    android:id="@+id/editTextName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="82dp"
    android:ems="10"
    android:inputType="text" >
</EditText>
<EditText
    android:id="@+id/editTextEmail"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editTextStreet"
    android:layout_below="@+id/editTextStreet"
    android:layout_marginTop="22dp"
    android:ems="10"
    android:inputType="textEmailAddress" />
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/editTextName"
    android:layout_alignParentLeft="true"
    android:text="Họ tên"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editTextCity"
    android:layout_alignParentBottom="true"
    android:layout_marginBottom="28dp"
    android:onClick="run"
    android:text="Lưu" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/editTextEmail"
```

```
    android:layout_alignLeft="@+id/textView1"
    android:text="Email"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<TextView
    android:id="@+id/textView5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/editTextPhone"
    android:layout_alignLeft="@+id/textView1"
    android:text="Điện thoại"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/editTextEmail"
    android:layout_alignLeft="@+id/textView5"
    android:text="Địa chỉ"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<EditText
    android:id="@+id/editTextCity"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignRight="@+id/editTextName"
    android:layout_below="@+id/editTextEmail"
    android:layout_marginTop="30dp"
    android:ems="10"
    android:inputType="text" />
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/editTextCity"
    android:layout_alignBottom="@+id/editTextCity"
    android:layout_alignParentLeft="true"
    android:layout_toLeftOf="@+id/editTextEmail"
    android:text="Quốc tịch"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<EditText
    android:id="@+id/editTextStreet"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editTextName"
    android:layout_below="@+id/editTextPhone"
    android:ems="10"
    android:inputType="text" >
    <requestFocus />
</EditText>
<EditText
    android:id="@+id/editTextPhone"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
```

```
        android:layout_alignLeft="@+id/editTextStreet"
        android:layout_below="@+id/editTextName"
        android:ems="10"
        android:inputType="phone|text" />
    </RelativeLayout>
</ScrollView>
```

Bước 2: File layout activity_main.xml chứa giao diện chính có nội dung như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textSize="30dp"
        android:text="Hướng dẫn sử dụng" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Cơ sở dữ liệu SQLite"
        android:id="@+id/textView2"
        android:layout_below="@+id/textView"
        android:layout_centerHorizontal="true"
        android:textSize="35dp"
        android:textColor="#ff16ff01" />
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView"
        android:layout_below="@+id/textView2"
        android:layout_centerHorizontal="true"
        android:src="@drawable/logo"/>
    <ScrollView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/scrollView"
        android:layout_below="@+id/imageView"
        android:layout_alignParentLeft="true"
```

```

        android:layout_alignParentStart="true"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true">
    <ListView
        android:id="@+id/listView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true" >
    </ListView>
</ScrollView>
</RelativeLayout>

```

Bước 3: Tạo file res/menu/main_menu.xml định nghĩa menu option cho giao diện chính của ứng dụng:

```

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:id="@+id/item1"
        android:icon="@drawable/add"
        android:title="Thêm mới" >
    </item>
</menu>

```

Bước 4: Tạo file res/menu/display_contact.xml định nghĩa các menu option cho giao diện hiển thị và nhập dữ liệu cho một contact (hình 4.4. (b)):

```

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:id="@+id/Edit_Contact"
        android:orderInCategory="100"
        android:title="Sửa"/>
    <item
        android:id="@+id/Delete_Contact"
        android:orderInCategory="100"
        android:title="Xoá"/>
</menu>

```

Bước 5: File AndroidManifest.xml khai báo Activity DisplayContact:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sairamkrishna.myapplication" >
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".DisplayContact"/>
    </application>
</manifest>
```

Bước 6: Tạo lớp truy cập và quản lý CSDL DBHelper.java có nội dung như sau:

```
public class DBHelper extends SQLiteOpenHelper {
    public static final String DATABASE_NAME = "MyDBName.db";
    public static final String CONTACTS_TABLE_NAME = "contacts";
    public static final String CONTACTS_COLUMN_ID = "id";
    public static final String CONTACTS_COLUMN_NAME = "name";
    public static final String CONTACTS_COLUMN_EMAIL = "email";
    public static final String CONTACTS_COLUMN_STREET = "street";
    public static final String CONTACTS_COLUMN_CITY = "place";
    public static final String CONTACTS_COLUMN_PHONE = "phone";
    private HashMap hp;
    // Tạo CSDL
    public DBHelper(Context context) {
        super(context, DATABASE_NAME , null, 1);
    }
    // Tạo bảng CSDL
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table contacts " + "(id integer primary key, name text, phone
        text, email text, street text, place text)");
    }
    // Tạo mới CSDL nếu nó đã tồn tại
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // TODO Auto-generated method stub
        db.execSQL("DROP TABLE IF EXISTS contacts");
        onCreate(db);
    }
}
```

```

    }
// Phương thức chèn một bản ghi vào CSDL
public boolean insertContact (String name, String phone, String email,
String street, String place) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("name", name);
    contentValues.put("phone", phone);
    contentValues.put("email", email);
    contentValues.put("street", street);
    contentValues.put("place", place);
    db.insert("contacts", null, contentValues);
    return true;
}
// Chọn một bản ghi từ bảng contacts với id chỉ định
public Cursor getData(int id) {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res = db.rawQuery( "select * from contacts where id='"+id+"'", null );
    return res;
}
// Lấy số lượng dòng của bảng contacts
public int numberOfRows(){
    SQLiteDatabase db = this.getReadableDatabase();
    int numRows = (int) DatabaseUtils.queryNumEntries(db, CONTACTS_TABLE_NAME);
    return numRows;
}
// Phương thức cập nhật dữ liệu cho một bản ghi trong bảng contacts có id chỉ định
public boolean updateContact (Integer id, String name, String phone, String
email, String street, String place) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("name", name);
    contentValues.put("phone", phone);
    contentValues.put("email", email);
    contentValues.put("street", street);
    contentValues.put("place", place);
    db.update("contacts", contentValues, "id = ? ", new String[] {
        Integer.toString(id) } );
    return true;
}
// Phương thức xoá một bản ghi trong bảng contacts có id chỉ định
public Integer deleteContact (Integer id) {
    SQLiteDatabase db = this.getWritableDatabase();
    return db.delete("contacts",
    "id = ? ",
    new String[] { Integer.toString(id) });
}
// Phương thức lấy thông tin về tất cả các contact
public ArrayList<String> getAllCotacts() {
    ArrayList<String> array_list = new ArrayList<String>();
    SQLiteDatabase db = this.getReadableDatabase();

```

```

        Cursor res = db.rawQuery( "select * from contacts", null );
        res.moveToFirst();
        // nếu con trỏ chưa ở cuối file thì tiếp tục đọc bản ghi
        while(res.isAfterLast() == false){
            array_list.add(res.getString(res.getColumnIndex(CONTACTS_COLUMN_NAME)));
            res.moveToNext();
        }
        return array_list;
    }
}

```

Bước 7: Tạo lớp DisplayContact.java xử lý dữ liệu trên giao diện nhập liệu cho các contact (hình 5.4. (b)):

```

public class DisplayContact extends Activity {
    int from_Where_I_Am_Coming = 0;
    private DBHelper mydb ;
    TextView name ;
    TextView phone;
    TextView email;
    TextView street;
    TextView place;
    int id_To_Update = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_display_contact);
        name = (TextView) findViewById(R.id.editTextName);
        phone = (TextView) findViewById(R.id.editTextPhone);
        email = (TextView) findViewById(R.id.editTextStreet);
        street = (TextView) findViewById(R.id.editTextEmail);
        place = (TextView) findViewById(R.id.editTextCity);
    }
    // Tạo đối tượng truy cập CSDL
    mydb = new DBHelper(this);
    // Đọc giá trị biến id contact được chọn từ Activity MainActivity gửi sang
    Bundle extras = getIntent().getExtras();
    if(extras !=null) {
        int Value = extras.getInt("id");
        if(Value>0){
            // Chọn bản ghi có id vừa nhận được
            Cursor rs = mydb.getData(Value);
            id_To_Update = Value; // đánh dấu id được chọn
            rs.moveToFirst();
            // Lấy thông tin của bản ghi có id được chọn
            String nam = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS_COLUMN_NAME));
            String phon = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS_COLUMN_PHONE));
            String emai = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS_COLUMN_EMAIL));
            String stree = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS_COLUMN_STREET));
            String plac = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS_COLUMN_CITY));
            // Đóng con trỏ
        }
    }
}

```

```

        if (!rs.isClosed()) {
            rs.close();
        }
    // Ẩn button có nhãn Lưu
    Button b = (Button) findViewById(R.id.button1);
    b.setVisibility(View.INVISIBLE);
    // Đoạn mã sau hiển thị thông tin lấy được của contact đã chọn lên các EditText
    // tương ứng
    name.setText((CharSequence)nam);
    name.setFocusable(false);
    name.setClickable(false);
    phone.setText((CharSequence)phon);
    phone.setFocusable(false);
    phone.setClickable(false);
    email.setText((CharSequence)emai);
    email.setFocusable(false);
    email.setClickable(false);
    street.setText((CharSequence)stree);
    street.setFocusable(false);
    street.setClickable(false);
    place.setText((CharSequence)plac);
    place.setFocusable(false);
    place.setClickable(false);
}
}
}
}
// Tạo menu option cho phép chỉnh sửa và xoá một contact được chọn
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Lấy id của contact từ Activity cha của nó
    Bundle extras = getIntent().getExtras();
    if(extras !=null) {
        int Value = extras.getInt("id");
        if(Value>0){
    // nếu có contact được chọn thì hiển thị 2 menu cho phép Sửa hoặc Xoá contact
            getMenuInflater().inflate(R.menu.display_contact, menu);
        } else{
    // nếu không có contact nào được chọn thì hiển thị menu thêm mới contact
            getMenuInflater().inflate(R.menu.menu_main, menu);
        }
    }
    return true;
}
// Xử lý sự kiện khi các menu option được chọn
public boolean onOptionsItemSelected(MenuItem item) {
    super.onOptionsItemSelected(item);
    switch(item.getItemId()){
        case R.id.Edit_Contact:
    // Nếu chọn menu Sửa Contact thì cập nhật giá trị của bản ghi và hiển thị nút Lưu
            Button b = (Button) findViewById(R.id.button1);
            b.setVisibility(View.VISIBLE);
            name.setEnabled(true);
            name.setFocusableInTouchMode(true);
    }
}

```

```
name.setClickable(true);
phone.setEnabled(true);
phone.setFocusableInTouchMode(true);
phone.setClickable(true);
email.setEnabled(true);
email.setFocusableInTouchMode(true);
email.setClickable(true);
street.setEnabled(true);
street.setFocusableInTouchMode(true);
street.setClickable(true);
place.setEnabled(true);
place.setFocusableInTouchMode(true);
place.setClickable(true);
return true;
case R.id.Delete_Contact:
// Nếu chọn menu Xoá Contact thì hiển thị hộp thoại xác nhận
// có chắc chắn xoá contact không
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setMessage("Bạn có chắc chắn muốn xoá contact?")
        .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
// Nếu chọn OK thì xoá contact
            public void onClick(DialogInterface dialog, int id) {
                mydb.deleteContact(id_To_Update);
                Toast.makeText(getApplicationContext(), "Xoá thành công",
                    Toast.LENGTH_SHORT).show();
                Intent intent = new
                    Intent(getApplicationContext(),MainActivity.class);
                startActivity(intent);
            }
        })
// Nếu chọn Cancel thì đóng hộp thoại
        .setNegativeButton("No", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
            }
        });
    AlertDialog d = builder.create();
    d.setTitle("Are you sure");
    d.show();
    return true;
default:
    return super.onOptionsItemSelected(item);
}
}
// Phương thức này được thực thi khi Activity MainActivity (hình 5.4. (a)) gọi
// phương thức startActivityForResult(intent) để khởi động lớp này.
public void run(View view) {
    Bundle extras = getIntent().getExtras();
    if(extras !=null) {
        int Value = extras.getInt("id");
        if(Value>0){ // nếu có một contact trong Listview Activity cha được chọn
            // thì cập nhật contact và trả về Activity cha
            if(mydb.updateContact(id_To_Update,name.getText().toString(),
                phone.getText().toString(),
                email.getText().toString(),
                street.getText().toString(),
                place.getText().toString(),
                value.getText().toString(),
                photo.getBitmap())>0)
                Toast.makeText(getApplicationContext(), "Cập nhật thành công",
                    Toast.LENGTH_SHORT).show();
        }
    }
}
```

```
        phone.getText().toString(), email.getText().toString(),
        street.getText().toString(), place.getText().toString())){
    Toast.makeText(getApplicationContext(), "Cập nhật thành công",
    Toast.LENGTH_SHORT).show();
}

// Trở về màn hình chính
Intent intent = new
Intent(getApplicationContext(),MainActivity.class);
startActivity(intent);

} else{
    Toast.makeText(getApplicationContext(), "Cập nhật không thành công",
    Toast.LENGTH_SHORT).show();
}
} else{
}

// Nếu không có contact nào được chọn thì chèn một bản ghi mới vào bảng
// và trả về Activity cha (màn hình chính)
if(mydb.insertContact(name.getText().toString(),
    phone.getText().toString(),
    email.getText().toString(), street.getText().toString(),
    place.getText().toString())){
    Toast.makeText(getApplicationContext(), "Đã chèn một bản ghi mới",
    Toast.LENGTH_SHORT).show();
}
else{
    Toast.makeText(getApplicationContext(), "Chèn bản ghi không thành
    công", Toast.LENGTH_SHORT).show();
}

// Trở về màn hình chính
Intent intent = new Intent(getApplicationContext(),MainActivity.class);
startActivity(intent);
}

}
```

Bước 8: Soạn thảo file chương trình MainActivity.java chính như sau:

```
public class MainActivity extends ActionBarActivity {  
    public final static String EXTRA_MESSAGE = "MESSAGE";  
    private ListView obj;  
    DBHelper mydb;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        // Tạo CSDL  
        mydb = new DBHelper(this);  
        // Lấy danh sách các contact và hiển thị họ tên lên ListView  
        ArrayList array_list = mydb.getAllCotacts();  
        ArrayAdapter arrayAdapter=new  
            ArrayAdapter(this,android.R.layout.simple_list_item_1, array_list);
```

```
obj = (ListView)findViewById(R.id.listView1);
obj.setAdapter(arrayAdapter);
// lắng nghe sự kiện chọn các item trong danh sách ListView
obj.setOnItemClickListener(new OnItemClickListener(){
    @Override
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long
        arg3) {
        int id_To_Search = arg2 + 1; // vị trí contact trong ListView được chọn
        // Tạo đối tượng Bundle để truyền id cho Activity con
        Bundle dataBundle = new Bundle();
        dataBundle.putInt("id", id_To_Search);
        // Triệu gọi Activity DisplayContact sử dụng Intent
        Intent intent = new Intent(getApplicationContext(),DisplayContact.class);
        intent.putExtras(dataBundle);
        startActivity(intent);
    }
});
// Tạo menu Thêm mới cho ứng dụng
@Override
public boolean onCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item){
    super.onOptionsItemSelected(item);
    // Nếu chọn menu Thêm mới thì truyền giá trị id cho Activity là 0
    // và gọi Activity DisplayContact con qua Intent
    switch(item.getItemId()) {
        case R.id.item1:
            Bundle dataBundle = new Bundle();
            dataBundle.putInt("id", 0);
            Intent intent = new Intent(getApplicationContext(),DisplayContact.class);
            intent.putExtras(dataBundle);
            startActivity(intent);
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
}
```

Câu hỏi và bài tập

1. Cho biết ưu điểm của việc lưu trữ dữ liệu với Shared preferences?
2. Định dạng của dữ liệu Shared preferences có đặc điểm gì?
3. Nêu trình tự các bước đọc/ghi dữ liệu trên Shared preferences trong Android?
4. Các bước xoá dữ liệu từ Shared preferences trong Android?
5. Lưu trữ dữ liệu trong file .txt của Android có ưu, nhược điểm gì?
6. Để truy nhập đến bộ nhớ ngoài (SDCard) của thiết chạy hệ điều hành Android cần khai báo quyền truy nhập trong file manifest như thế nào?
7. Nêu ưu điểm của việc sử dụng CSDL SQLite trong Android so với file .txt và Shared preferences trong việc lưu trữ dữ liệu?
8. Trình bày các bước tạo và mở một CSDL SQLite trong Android?
9. Nêu các bước đọc các bản ghi từ CSDL SQLite và hiển thị các thông tin của từng bản ghi?
10. Trình bày các bước thêm mới một bản ghi vào CSDL SQLite?
11. Trình bày các bước cập nhật thông tin một bản ghi vào CSDL SQLite?

Chương 6

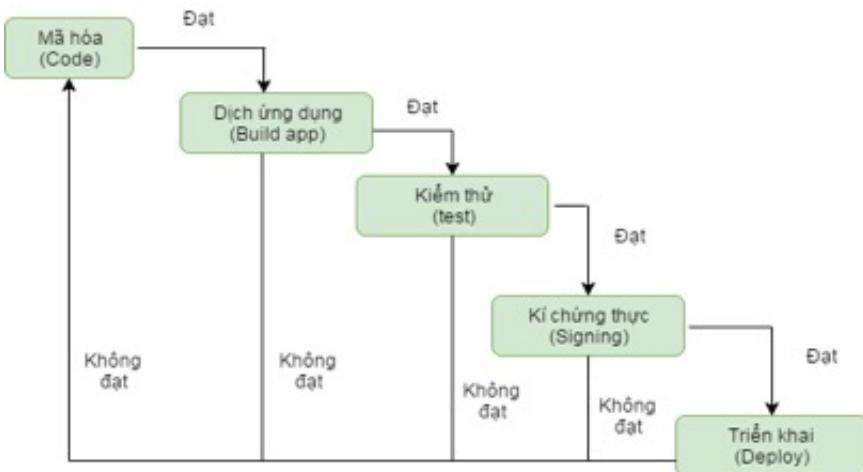
Xuất bản và phân phối ứng dụng Android

6.1	Giới thiệu	189
6.2	Xuất file .apk lên Google Play	193
6.3	Quản lý phiên bản ứng dụng	197
Tài liệu tham khảo		199

Chương 6 trình bày các bước cơ bản để người lập trình có thể triển khai ứng dụng của mình đến người dùng cuối. Có thể triển khai ứng dụng trên các kho ứng dụng khác nhau, nhưng cuốn sách này sẽ tập trung hướng dẫn cách triển khai ứng dụng lên kho ứng dụng Google Play.

6.1 Giới thiệu

Xuất bản ứng dụng Android là quá trình đưa ứng dụng Android của chúng ta đến người dùng. Xuất bản là giai đoạn cuối của quá trình phát triển ứng dụng Android. Sau khi phát triển, các ứng dụng Android có thể được đóng gói dễ dàng và bán ra thông qua cửa hàng như Google



Hình 6.1: Vòng đời phát triển của một ứng dụng Android.

Play (<https://play.google.com>), SlideME (<http://slideme.org>), Opera Mobile Store (<http://html5.oms.apps.opera.com>), Mobango (www.mobango.com/), F-droid (<https://f-droid.org>) và Amazon Appstore (<https://www.amazon.com/mobile-apps>). Chúng ta cũng có thể phát hành các ứng dụng của mình bằng cách gửi trực tiếp cho người dùng hoặc cho phép người dùng tải xuống từ trang web của riêng chúng ta. Tuy nhiên để có thể cài đặt ứng dụng từ nguồn không phải từ Google Play Store, thiết bị cài đặt ứng dụng của chúng ta cần bật tùy chọn “Unknown source” trong phần Settings > Security. (Giáo trình này hướng dẫn người dùng xuất bản ứng dụng của mình lên Google Play).

Các bước để triển khai ứng dụng Android tới người dùng như sau:

- Bước 1: Thủ nghiệm trên các thiết bị khác nhau (Regression Testing): Trước khi chúng ta xuất bản ứng dụng, chúng ta cần đảm bảo ứng dụng của chúng ta sử dụng tốt trên các dòng máy Android khác nhau. Vì vậy, hãy thực hiện các thử nghiệm trên các thiết bị khác nhau bao gồm điện thoại và máy tính bảng.



Hình 6.2: Một số kho ứng dụng dành cho Android.

- Bước 2: Đánh giá ứng dụng (Application Rating): Khi chúng ta xuất bản ứng dụng của mình tại Google Play, chúng ta sẽ phải chỉ định xếp hạng nội dung cho ứng dụng, thông báo cho người dùng Google Play về mức trưởng thành. (A) Tất cả mọi người (b) Độ trưởng thành thấp (c) Độ trưởng thành trung bình (d) Độ trưởng thành cao.
- Bước 3: Khu vực mục tiêu (Targeted Regions) Google Play cho phép chúng ta kiểm soát những quốc gia và vùng lãnh thổ nào ứng dụng của chúng ta sẽ được bán. Theo đó, chúng ta phải quan tâm đến việc thiết lập múi giờ, nội địa hóa hoặc bất kỳ yêu cầu cụ thể nào khác theo khu vực mục tiêu.
- Bước 4: Kích thước ứng dụng (Application Size) hiện tại, kích thước tối đa cho một APK được xuất bản trên Google Play là 50 MB. Nếu ứng dụng của chúng ta vượt quá kích thước đó hoặc nếu chúng ta muốn cung cấp tải xuống phụ, chúng ta có thể sử dụng tệp mở rộng APK mà Google Play sẽ lưu trữ miễn phí trên cơ sở hạ tầng máy chủ của nó và tự động xử lý việc tải xuống thiết bị.
- Bước 5: Khả năng tương thích SDK và màn hình (SDK and

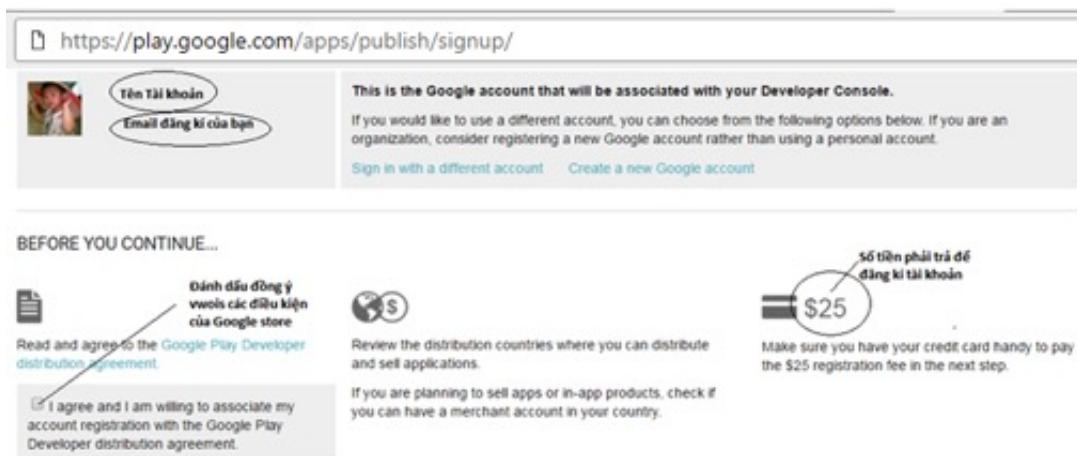
Screen Compatibility) Điều quan trọng là đảm bảo rằng ứng dụng của chúng ta được thiết kế để chạy đúng trên các phiên bản hệ điều hành Android và kích thước màn hình thiết bị mà chúng ta muốn nhắm mục tiêu.

- Bước 6: Giá ứng dụng (Application Pricing) Quyết định xem ứng dụng của chúng ta sẽ miễn phí hay trả phí là quan trọng vì trên Google Play. Nếu chúng ta muốn bán ứng dụng của mình thì chúng ta sẽ phải chỉ định giá của nó bằng các đơn vị tiền tệ khác nhau.
- Bước 7: Nội dung khuyến mại (Promotional Content) đây là phương pháp tiếp thị tốt để cung cấp nhiều nội dung đồ họa chất lượng cao để giới thiệu ứng dụng hoặc thương hiệu của chúng ta. Sau khi xuất bản, chúng xuất hiện trên trang chi tiết sản phẩm của chúng ta, trong danh sách cửa hàng và kết quả tìm kiếm và ở nơi khác.
- Bước 8: Xây dựng và sẵn sàng tải APK (Build and Upload release-ready APK): tạo phiên bản của ứng dụng và sẵn sàng cho phát hành. Chi tiết các bước xây dựng và tải bản ứng dụng lên Google Play được trình bày chi tiết trong Phần 5.2.
- Bước 9: Hoàn thành chi tiết về ứng dụng (Finalize Application Detail) Google Play cung cấp cho bạn một số cách để quảng cáo ứng dụng của bạn và tương tác với người dùng trên trang chi tiết sản phẩm của chúng ta, từ các đồ họa đầy màu sắc, ảnh chụp màn hình và video đến các mô tả được bản địa hóa, chi tiết phát hành và các liên kết đến các ứng dụng khác của chúng ta. Vì vậy, ta có thể trang trí trang ứng dụng của mình và cung cấp càng nhiều chi tiết rõ nét mà ta có thể có.

6.2 Xuất file .apk lên Google Play

Để xuất file .apk lên Google Play cần thực hiện các bước sau:

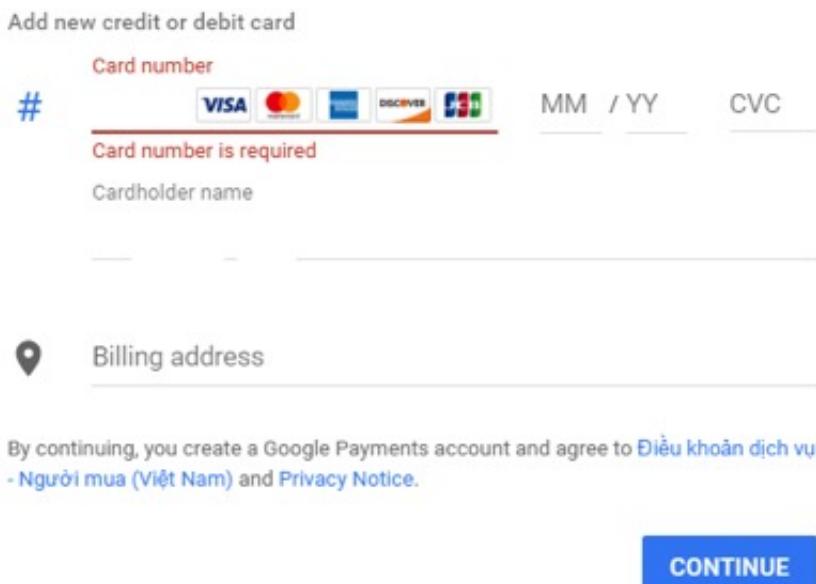
- Bước 1: Trước khi xuất bản ứng dụng lên Google play chúng ta cần phải có tài khoản developer trên Website <https://play.google.com/apps/publish/>. Chúng ta truy cập vào đường dẫn trên và tạo tài khoản mới hoặc đăng ký tài khoản bằng tài khoản gmail của mình. Để đăng ký, người dùng cần



Hình 6.3: Đăng kí tài khoản developer của Google Store.

chuẩn bị một thẻ có thể thanh toán online (visa, master card, ...). Nhập các thông tin về tài khoản thanh toán tiền và nhấn nút continue để hoàn tất thủ tục đăng kí tài khoản. Nếu thẻ của chúng ta được chấp nhận, Google sẽ đưa tài khoản của chúng ta vào trạng thái chờ kiểm tra, sau khoảng vài ngày sẽ có phản hồi từ Google chấp nhận tài khoản của chúng ta hay không. Trong trường hợp không được chấp nhận, Google sẽ yêu cầu chúng ta gửi thêm tài liệu chứng thực thông tin cá nhân và tài khoản ngân hàng của chúng ta. (Trong thời gian chờ đợi

Google kiểm tra trạng thái thanh toán, chúng ta vẫn có thể đưa ứng dụng lên server của Google, tuy nhiên chưa thể phân phối đến người dùng).



Hình 6.4: Nhập thông tin thanh toán phí đăng ký tài khoản.

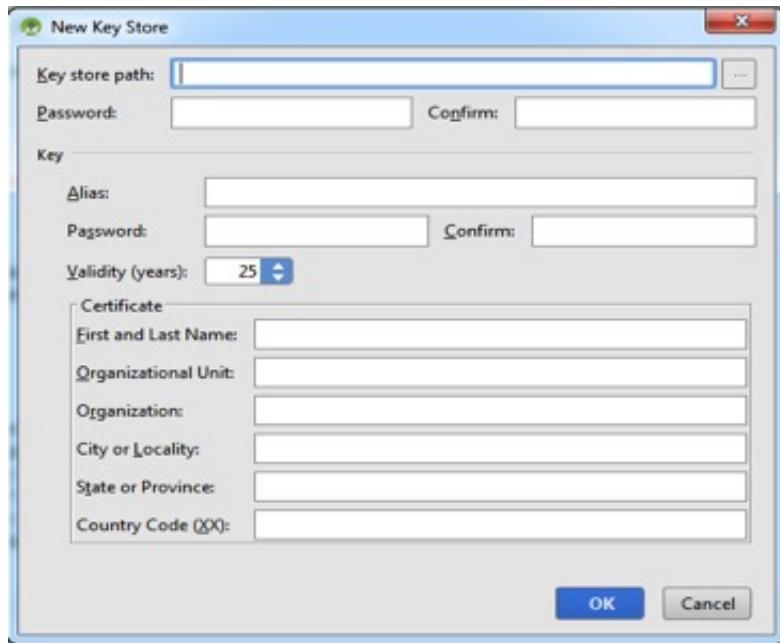
- Bước 2: Tạo file signed apk trong Android studio. Do múi giờ của Việt Nam và múi giờ của trụ sở Google store chênh lệch nhau nên để ứng dụng của chúng ta lên được website chúng ta cần chỉnh đồng hồ của hệ thống máy tính lùi lại ngày hiện tại 2 ngày (ví dụ hiện tại là ngày 10 thì chúng ta chỉnh lại ngày 8), trước khi tạo file signeg apk. Tạo chứng thực số (keystore): Tất cả ứng dụng Android đều phải được ký theo một chứng thực số trước khi có thể cài đặt lên thiết bị hoặc Android Emulator. Keystore giống như một dạng chữ kí để đánh dấu bản quyền của chúng ta lên ứng dụng. Để tạo keystore trong android chúng ta mở ứng dụng trong Android Studio và chọn Build, chọn Generate Signed APK, chọn Create new keystore(

xuất hiện màn hình như Hình 6.5). Điền đầy đủ các thông tin sau:

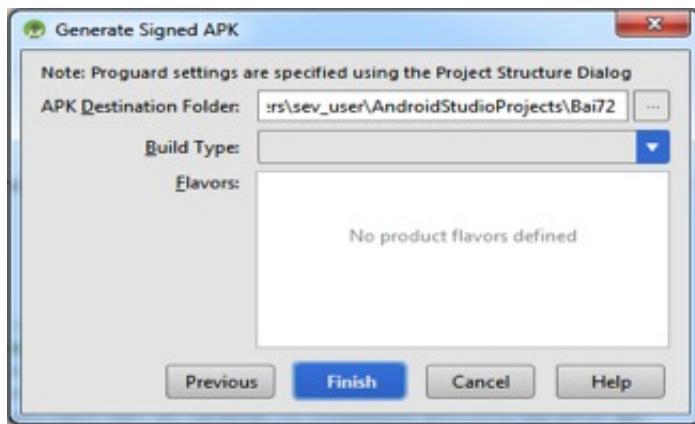
- Key store path: nhập đường dẫn lưu keystore.
- Password, confirm: Nhập mật khẩu và xác nhận lại mật khẩu.
- Alias: tên bí danh (nên đặt giống tên keystore)
- Nhập lại mật khẩu và xác nhận mật khẩu trong phần Key
- Nhập các thông tin của chúng ta: họ tên (First and last name), đơn vị tổ chức (Organizational Unit), thành phố (City or Locality), tỉnh thành (State or Province), mã nước (Country code) (ví dụ 25000). Phần thông tin cá nhân có thể bỏ trống nhưng chúng ta nên đặt đầy đủ.

Nhập xong chọn nút ok. Android sẽ tạo ra keystore cho chúng ta. Ta tiến hành ký lên ứng dụng (file apk) trên theo chứng thực mới được tạo ra. Trong mục Build Type, chọn release thì ứng dụng mới triển khai trên Google store được. Nhấn Finish để kết thúc việc tạo file signed apk. *Chú ý là cần phải lưu lại keystore và mật khẩu tương ứng với ứng dụng để có thể cung cấp ứng dụng của chúng ta khi Google store sau này.*

- Bước 3: Tải ứng dụng lên Google Play. Đăng nhập vào tài khoản developer của mình trên website <https://play.google.com/apps/publish/>, chọn Add new application để tải ứng dụng mới lên. Ta cần cung cấp đầy đủ thông tin yêu cầu về ứng dụng trước khi có thể phân phối đến người dùng như: tải file ứng dụng (apk) của mình lên hệ thống, mô tả ứng dụng và chính sách giá, ngôn ngữ... Cung cấp tiêu đề (tên ứng dụng), mô tả, tải lên file icon lớn (512x512px) và



Hình 6.5: Tạo keystore.



Hình 6.6: Tạo file apk trong android.

ít nhất 02 hình chụp màn hình của ứng dụng, chọn loại ứng dụng, cấu hình website và email hỗ trợ...: Đặt giá bán ứng dụng (hiện tại ở Việt Nam ta chỉ được phép đăng ứng dụng miễn phí) và các quốc gia muốn phân phối ứng dụng: Cuối cùng bấm “Save and publish”. Ứng dụng sẽ bắt đầu được phân

phối trên hệ thống Google Play. *Chú ý: cần phải mất từ vài giờ đến vài ngày để ứng dụng có thể vượt qua được hệ thống kiểm tra của Google và phân phối khắp mạng CDN của Google.* Sau khi hoàn tất các bước như trên, ta đã hoàn thành việc phát triển một ứng dụng cho hệ điều hành Android và phân phối đến người dùng. Ta có thể thường xuyên ghé thăm trang dành cho developer này để xem các thông kê khác nhau liên quan đến việc cài đặt và sử dụng ứng dụng của mình như: số lượt cài đặt/gỡ bỏ theo ngày, tỉ lệ các phiên bản Android đang dùng, các lỗi crash ứng dụng, đánh giá, phản hồi của người dùng...

6.3 Quản lý phiên bản ứng dụng

Đánh số phiên bản phần mềm: Phiên bản phần mềm được đánh số trong file AndroidManifest.xml, dưới hai thuộc tính là android:versionCode và android:versionName:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="vn.edu.humg.android_course.JSON"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="13" />
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        \dot
    </application>
</manifest>
```

Trong đó versionCode là số hiệu phiên bản, có kiểu số nguyên, dùng cho hệ thống để phân biệt các phiên bản của ứng dụng được cài đặt. Mỗi khi nâng cấp phiên bản của ứng dụng, ta cần thay đổi (tăng) số này trước khi phân phối. Tham số versionName là tên

của phiên bản. Tham số này không được dùng bởi hệ thống, mà chỉ là tên phiên bản người dùng sẽ nhìn thấy khi xuất bản lên các chợ ứng dụng. VersionName có kiểu chuỗi và có thể đặt tùy ý, tuy nhiên định dạng thường được dùng là <major>.<minor>.<point>. Ở đó <major> là số hiệu phiên bản chính, <minor> là phiên bản phụ, <point> là số hiệu cập nhật nhỏ trong phiên bản phụ, ví dụ “1.0.1”, “2.1.0”... Ngoài ra, để ứng dụng có thể được phân phối trên chợ ứng dụng, ta cần chỉ ra icon và tiêu đề của ứng dụng để hiển thị trong danh sách ứng dụng. Để làm điều này, ta cần đặt giá trị cho tham số android:icon và android:label của thẻ <application>.

Câu hỏi và bài tập

1. Có các chợ ứng dụng nào mà người phát triển ứng dụng Android có thể triển khai ứng dụng của mình?
2. Các thao tác cần thiết để triển khai ứng dụng lên Google Play?

Tài liệu tham khảo

- [1] Darwin, Ian, *Android cookbook*, O'Reilly Media, Inc.", 2012.
- [2] Morris, Jason, *Android User Interface Development: Beginner's Guide*, PACKT publishing Ltd, 2011.
- [3] Murphrey. M, *Beginning Android 2*, Apress, 2010.
- [4] Jackson, Wallace, and Kunal Mittal, *Android apps for absolute beginners*, Apress, 2011.
- [5] <https://vi.wikipedia.org/wiki/Android>
- [6] <https://www.tutorialspoint.com/android/>
- [7] https://www.tutorialspoint.com/android/android_sqlite_database.htm
- [8] <https://developer.android.com/training/index.html>

