# Chương 1: Giới thiệu

# Chương 2: Nền tảng Android

## 2.1. Khái niệm



1. Điện thoại Android, HTC Dream (trái) và HTC Magic (phải)

Android được phát triển bởi tập đoàn Google, phiên bản đầu tiên ra đời năm 2008. Được xây dựng trên một nền tảng mở, và một bộ thư viện đa năng, mạnh mẽ với nguyên lý mở, Android đã nhanh chóng được cộng đồng lập trình viên di động hưởng ứng mạnh mẽ. Nền tảng Android tích hợp nhiều tính năng nổi bật:

- Android là một hệ điều hành nhân Linux, đảm bảo sự tương tác với các phần cứng, quản lý bộ nhớ, điều khiển các tiến trình tối ưu cho các thiết bị di động.

- Bộ ứng dụng khung cho phép sử dụng lại và thay thế các thành phần riêng lẻ.

- Máy ảo Dalvik được tối ưu cho các thiết bị di động, chạy các ứng dụng lập trình trên ngôn ngữ Java.

- Các thư viện cho phát triển ứng dụng mã nguồn mở bao gồm SQLite, WebKit, OpenGL và trình quản lý đa phương tiện.

- Hỗ trợ các chuẩn đa phương tiện phổ biến, thoại trên nền GSM, Bluetooth EDGE, 3G và Wifi

- Hỗ trợ Camera, GPS, la bàn, máy đo gia tốc…

- Bộ phát triển ứng dụng SDK đầy đủ gồm thiết bị giả lập, công cụ sửa lỗi, tích hợp với Eclipse SDK.

Android cung cấp một tập hợp đầy đủ các phần mềm cho thiết bị di động bao gồm: hệ điều hành, các khung ứng dụng và các ứng dụng cơ bản.

## 2.2. Đặc điểm

### 2.2.1. Tính mở

Android được xây dựng từ dưới đi lên cho phép người phát triển tạo các ứng dụng di động hấp dẫn với đầy đủ các điểm mạnh của các thiết bị cầm tay hiện có. Android hoàn toàn mở, một ứng dụng có thể gọi tới bất kể một chức năng lõi của điện thoại như tạo cuộc gọi, gửi tin nhắn hay sử dụng máy ảnh, cho phép người phát triển tạo phong phú hơn, liên kết hơn các tính năng cho người dùng. Android được xây dựng trên nhân Linux mở. Thêm nữa, nó sử dụng một máy ảo mà đã được tối ưu hóa bộ nhớ và phần cứng với môi trường di động. Android mà một mã nguồn mở, nó có thể được mở rộng để kết hợp tự do giữa các công nghệ nổi trội. Nền tảng này sẽ tiếp tục phát triển bởi cộng đồng phát triển để tạo ra các ứng dụng di động hoàn hảo.

### 2.2.2. Tính ngang bằng của các ứng dụng

Với Android, không có sự khác nhau giữa các ứng dụng điện thoại cơ bản với ứng dụng của bên thứ ba. Chúng được xây dựng để truy cập như nhau tới một loạt các ứng dụng và dịch vụ của điện thoại. Với các thiết bị được xây dựng trên nền tảng Android, người dùng có thể đáp ứng đầy đủ các nhu cầu mà họ thích. Chúng ta có thể đổi màn hình nền, kiểu gọi điện thoại, hay bất kể ứng dụng nào. Chúng ta thậm chí có thể hướng dẫn điện thoại chỉ xem những ảnh mình thích.

### 2.2.3. Phá vỡ rào cản phát triển ứng dụng

Android phá vỡ rào cản để tạo ứng dụng mới và cải tiến. Một người phát triển có thể kết hợp thông tin từ trang web với dữ liệu trên điện thoại cá nhân – chẳng hạn như danh bạ, lịch hay vị trí trên bản đồ – để cung cấp chính xác hơn cho người khác. Với Android, người phát triển có thể xây dựng một ứng dụng mà cho phép người dùng xem vị trí của những người bạn và thông báo khi họ đang ở vị trí lân cận. Tất cả được lập trình dễ dàng thông qua sự hỗ trợ của MapView và dịch vụ định vị toàn cầu GPS.

### 2.2.4. Dễ dàng và nhanh chóng xây dựng ứng dụng

Android cung cấp bộ thư viện giao diện lập trình ứng dụng đồ sộ và các công cụ để viết các ứng dụng phức tạp. Ví dụ, Android có thể cho phép người phát triển biết được vị trí của thiết bị và cho phép các thiết bị giao tiếp với nhau để có thể tạo nên mạng xã hội chia sẻ ngang hàng rộng khắp. Thêm nữa, Android còn bao gồm một bộ công cụ đầy đủ giúp cho việc phát triển trở nên dễ dàng.

## 2.3. Kiến trúc của nền tảng Android

### 2.3.1. Kiến trúc tổng quát

Android bao gồm bốn thành phần sau, theo [8]:

* + *Hệ điều hành*
  + *Thư viện và các giao diện lập trình ứng dụng*
  + *Khung ứng dụng*
  + *Ứng dụng*

### 2.3.2. Hệ điều hành



1. Nhân Linux trên Android

Android sử dụng nhân Linux 2.6 làm nhân cho các dịch vụ hệ thống như bảo mật, quản lý bộ nhớ, quản lý tiến trình (xử lý tiến trình, đa luồng), ngăn xếp mạng và trình điều khiển thiết bị (giao tiếp USB, giao tiếp hồng ngoại, không đây, v.v…). Nhân Linux này cũng có vai trò như một lớp trừu tượng giữa phần cứng và phần mềm.

### 2.3.3. Thư viện và các giao diện lập trình ứng dụng

Android có một thư viện khá phong phú [9] cung cấp sẵn để người lập trình có thể sử dụng. Hình 4 là sơ đồ tóm tắt về hệ thống thư viện này.



1. Thư viện lâp trình ứng dụng trên Android

#### Android runtime

Android có một tập các thư viện nòng cốt để cung cấp hầu hết các chức năng sẵn có trong thư viện cốt lõi của ngôn ngữ lập trình Java. Android Runtime: Bao gồm máy ảo Dalvik và các thư viện Android.

Các thư viện cơ bản: Các ứng dụng Android được phát triển trên môi trường Java, nhưng Dalvik lại không phải là một Java VM. Các thư viện cơ bản của Android cung cấp hầu hết các chức năng có trong thư viện cơ bản của Java cũng như là thư viện riêng của Android.

Máy ảo Dalvik: Dalvik là máy ảo để chạy các ứng dụng trên Android, đã được tối ưu để đảm bảo rằng một thiết bị có thể chạy được nhiều Instance một cách hiệu quả. Nó dựa vào nhân Linux để thực hiện đa luồng và quản lý bộ nhớ cấp thấp.

#### Bộ thư viện

Android cung cấp một số các APIs cho phát triển ứng dụng. Danh sách các API cơ bản sau được cung cấp bởi tất cả các thiết bị trên nền Android:

- android.util: Gói tiện ích cơ bản bao gồm nhiều lớp mức thấp như là các lớp quản lý (List, Stack…) lớp xử lý chuỗi, lớp xử lý XML

- android.os Gói hệ điều hành cung cấp truy cập đến các dịch vụ cơ bản như là chuyển tin nhắn, thông tin chéo, đồng hồ và gỡ lỗi.

- android.graphics Cung cấp các lớp đồ họa mức thấp thực hiện các chức năng đồ họa, màu, vẽ cơ bản.

- android.text Công cụ hiển thị và xử lý văn bản

- android.database Cung cấp các lớp mức thất bắt buộc cho việc điều khiển cursor khi làm việc với các cơ sở dữ liệu

- android.content Các giao tiếp lập trình nội dung được dùng để quản lý truy cập dữ liệu và xuất bản bằng cách cung cấp các dịch vụ thao tác với tài nguyên, Content Provider, và các gói

- android.view View là lớp giao diện người dùng cơ bản nhất. Tất cả giao diện người dùng được tạo ra đều phải sử dụng một tập các View để cung cấp cho các thành phần tương tác người dùng.

- android.widget Xây dựng dựa trên gói View. Những lớp Widget những thành phần giao diện được tạo sẵn được sử dụng để tạo nên giao diện người dùng. Các Widget bao gồm danh sách, nút bấm, hộp nhập, các kiểu trình bày(layout)

- com.google.android.maps bộ API mức cao cung cấp truy cập đến điều khiển bản đồ sẵn trong Androif từ ứng dụng được xây dựng. Bao gồm cả lớp MapView cũng như Overlay và MapController để tương tác với bản đồ bên trong ứng dụng.

- android.app Một gói thư viện bậc cao, cung cấp truy cập đến dữ liệu của ứng dụng. Gói ứng dụng cũng bao gồm lớp Activity và Service là thành phần cơ bản của mọi ứng dụng Android.

- Android.provider Để tạo thuận lợi cho người phát triển truy cập đến các Content Provider tiêu chuẩn(như là dữ liệu danh bạ), gói Cung cấp(Provider) bao gồm các lớp cho phép truy cập đến cơ sở dữ liệu chuẩn trong tất cả các bản phân phối Android.

- Android.telephony Các API điện đàm cung cấp khả năng tương tác trực tiếp với tầng điện thoại trong các thiết bị, cho phép tạo, nhận, theo dõi các cuộc gọi, tình trạng các cuộc gọi và tin nhắn SMS.

- android.webkit Gói WebKit cung cấp các API để làm việc với các nội dung Web-based bao gồm một lơp WebView để tạo ra giao diên web, nhúng trong ứng dụng và một trình quản lý cookie.

Cùng với các API của Android, còn có một tập các thư viện C/C++ như:

o OpenGL Thư viện dùng để tạo ra các đồ họa 3D dựa vào chuẩn OpenGLES 1.0 API

o FreeType Hỗ trợ xử lý bitmap và font vector

o GGL Thư viện cơ bản, dùng để cung cấp các engine đồ họa 2D

o Libc Thư viện C chuẩn, được tối ưu cho các thiết bị Linux-based

o SQLite Engine cơ sở dữ liệu quan hệ gọn nhẹ, dùng để lưu trữ dữ liệu của ứng dụng

o SSL Hỗ trợ sử dụng giao thức mã hóa Secure Sockets Layer trong bảo mật truyền thông Internet

- Ngoài các thư viện chuẩn của Android, để đáp ứng tiêu chí phù hợp với nhiều thiết bị khác nhau, Android còn có thể có các API phụ thuộc thiết bị như android.location, android.media, android.opengl, android.hardware, android.bluetooth, android.net.wifi, và android.telephony.

#### Khung ứng dụng trên Android



1. Khung ứng dụng

Kiến trúc của Android khuyến khích khái niệm Thành phần sử dụng lại, cho phép công bố và chia sẻ các Activity, Service, dữ liệu, với các ứng dụng khác với quyền truy cập được quản lý bởi khai báo.

Cơ chế đó cho phép người lập trình tạo ra một trình quản lý danh bạ hoặc trình quay số điện thoại mà có các thành phần người khác có thể tạo mới giao diện và mở rộng chức năng thay vì tạo lại chúng.

Những dịch vụ sau là những dịch vụ kiến trúc cơ bản nhất của tất cả các ứng dụng, cung cấp một framework cho mọi mọi phần mềm được xây dựng:

- Actitvity Manager: Điều khiển vòng đời của các Activity bao gồm cả quản lý các tầng Activity.

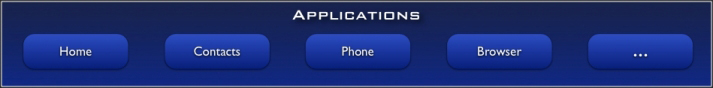
- Views: Được sử dụng để tạo lập các giao diện người dùng cho các Activity

- Notification Mamager: Cung cấp một cơ chế cố định và quy củ cho việc gửi các thông báo đến người dùng.

- Content Provider: Cho phép ứng dụng chia sẻ dữ liệu giữa các ứng dụng.

- Resource Manager: Hỗ trợ các thành phần không thuộc mã nguồn như là chuỗi ký tự, đồ họa được đặt bên ngoài.

#### Ứng dụng



1. Các ứng dụng trên Android

Đây là lớp trên cùng của kiến trúc nền tảng Android. Android sẽ hoạt động với một bộ các ứng dụng bao gồm ứng dụng thư điện tử, gửi tin nhắn, lịch, bản đồ, trình duyệt web, danh bạ v.v… Tất cả các ứng dụng được viết bằng ngôn ngữ Java. Các ứng dụng này có thể được cung cấp sẵn hoặc được phát triển bởi những lập trình viên.

## 2.4. Các thành phần của ứng dụng

Một ứng dụng trên Android được cấu thành từ bốn thành phần cơ bản sau:

* + *Activities (Hoạt động)*
  + *Services*
  + *Broadcast Receivers*
  + *Content Provider*

Các thành phần này không nhất thiết phải có mặt đầy đủ trong ứng dụng. Chúng ta có thể xem các thành phần nào được sử dụng trong ứng dụng bằng việc xem khai báo trong file AndroidManifest.xml.

### 2.4.1. Hoạt động (Activity)

#### Khái niệm

Một hoạt động là một giao diện người dùng trực quan mà người dùng có thể thực hiện trên đó mỗi khi được kích hoạt. Một ứng dụng có thể có nhiều hoạt động và chúng có thể gọi đến nhau chuyển giữa các hoạt động với nhau. Mỗi activity là một dẫn xuất của lớp android.app.Activity.

Mỗi hoạt động có một cửa sổ để vẽ lên. Thông thường cửa sổ này phủ đầy màn hình, ngoài ra nó cũng có thể có thêm các cửa sổ con khác như là hộp thoại…Nội dung của cửa sổ của hoạt động được cung cấp bởi một hệ thống cấp bậc các View (là đối tượng của lớp Views).

#### Vòng đời của hoạt động

Các hoạt động trong hệ thống được quản lý bởi một cấu trúc dữ liệu ngăn xếp. Khi có một hoạt động được khởi tạo, nó được đẩy vào trong ngăn xếp, chuyển sang trạng thái thực thi và hoạt trộng trước đó sẽ chuyển sang trạng thái chờ. Hoạt động này chỉ trở lại trang thái kích hoạt khi mà hoạt động vừa khởi tạo kết thúc việc thực thi.

Một hoạt động có ba trạng thái chủ yếu đó là:

Một activity có 3 trạng thái chính:

- Active hoặc running khi nó ở trên nhất màn hình và nhận tƣơng tác ngƣời dùng

- Paused khi Activity không còn là trọng tâm trên màn hình nhƣng vẫn hiện thị trƣớc ngƣời dùng.

- Stopped Khi một activity hoàn toàn bị che khuất, nó sẽ rơi vào trạng thái Stopped. Tuy nhiên, nó vẫn còn lƣu trữ toàn bộ thông tin trạng thái. Và nó thƣờng bị hệ thống đóng lại khi có tình trạng thiếu bộ nhớ.

Khi chuyển giữa các trạng thái, ứng dụng sẽ gọi các hàm callback ứng với các bƣớc chuyển:

void onCreate(Bundle savedInstanceState)

void onStart()

void onRestart()

void onResume()

void onPause()

void onStop()

void onDestroy()

Biểu đồ sau mô tả trạng thái trong vòng đời của một hoạt động. Hình chữ nhật viên còn thể hiện các phương thức Callback mà chúng ta có thể khai báo để gọi thực thi một số thao tác khi hoạt động chuyển sang trạng thái khác (phương thức Callback là phương thức được gọi lại bởi một phương thức khác khi có một sự kiện xảy ra). Các trạng thái chính của một hoạt động được thể hiện bởi các hình viên thuốc.



1. Vòng đời của một hoạt động

Vòng đời của một hoạt động có thể được thể hiện trong những quá trình sau:

Toàn bộ thời gian sống của một hoạt động bắt đầu từ lời gọi đầu tiên tới phương thức onCreate (Bundle) tới lời gọi phương thức onDestroy(). Trong quá trình này, một hoạt động sẽ khởi tạo lại tất cả các tài nguyên cần sử dụng trong phương thức onCreate() và giải phóng chúng khi phương thức onDestroy() được thực thi.

Thời gian sống có thể nhìn thấy của một hoạt động bắt đầu từ lời gọi tới phương thức onStart(), cho tới khi phương thức onStop() của nó được thực thi. Toàn bộ các tài nguyên đang được sử dụng bởi hoạt động vẫn tiếp tục được lưu giữ, người dùng có thể thấy giao diện nhưng không tương tác được với hoạt động do trong qua trình này hoạt động không ở trạng thái chạy tiền cảnh.

Thời gian sống tiền cảnh của một hoạt động là quá trình bắt dầu từ khi có lời gọi tới phương thức onResume() và kết thúc bằng lời gọi tới phương thức onPause(). Trong thời gian này, hoạt động chạy ở tiền cảnh và có thể tương tác với người dùng.

### 2.4.2. Service

#### Khái niệm

Một dịch vụ (service) là các đoạn mã được thực thi ngầm bởi hệ thống mà người sử dụng không thấy được. Mỗi Service đề được mở rộng từ lớp cơ sở là Service trong gói android.app. Có thể kết nối tới hoặc kích hoạt một Service thông qua interface mà Service đưa ra.Ví dụ như một chương trình chơi nhạc, sẽ có vài hoạt động cho phép người dùng duyệt danh sách các bài hát và lựa chọn bài nào để phát. Tuy nhiên, chức năng chơi nhạc không được thiết kế như một hoạt động bởi chúng ta sẽ muốn chuyển qua cửa sổ khác, như khi soạn tin nhắn thì bài nhạc vẫn tiếp tục được chơi. Trong trường hợp này, ứng dụng chơi nhạc sẽ khởi tạo một dịch vụ bằng cách sử dụng phương thức Context.startService().

Một ứng dụng có thể dễ dàng thực hiện liên kết tới một dịch vụ đang chạy (thậm chí khởi động nếu nó chưa thực thi) bằng phương thức Context.bindService(). Khi đó dịch vụ này sẽ cung cấp cho ứng dụng cơ chế để giao tiếp với chúng thông qua giao diện gọi là IBinder (đối với dịch vụ chơi nhạc có thể cho phép dừng hoặc chuyển qua bài nhạc kế tiếp).

#### Vòng đời của một dịch vụ

Vòng đời của một dịch vụ được hiểu là quá trình hoạt động từ khi nó được tạo ra cho tới khi bị loại khỏi hệ thống. Có hai cách thức để một dịch vụ có thể được chạy trong hệ thống:

Khi hệ thống có lời gọi tới phương thức Context.startService(). Trong trường hợp này, dịch vụ sẽ được thực hiện liên tục cho tới khi hệ thống gọi phương thức Context.stopService().

Khi các ứng dụng gọi phương thức Context.bindService() để tạo kết nối với dịch vụ (dịch vụ sẽ được khởi tạo nếu tại thời điểm đó nó đang không hoạt động). Ứng dụng sẽ nhận được một đối tượng IBinder do dịch vụ trả lại để có thể gọi các phương thức Callback phù hợp để truy cập tới các trạng thái của dịch vụ. Nếu do lời gọi Context.bindService() mà dịch vụ được khởi tạo thì nó sẽ được thực thi cho tới khi nào kết nối trên (tức là đối tượng IBinder) vẫn còn tồn tại.



Hình 7 Sơ đồ chuyển trạng thái của Service3

### 2.4.3. Bộ nhận quảng bá (Broadcast receivers)

#### Khái niệm

Bộ nhận quảng bá là một thành phần không làm gì cả nhưng nó nhận và phản hồi lại các thông báo quảng bá. Nhiều quảng bá có nguồn gốc từ mã hệ thống, ví dụ thông báo thay đổi múi giờ, pin yếu, ảnh đã chụp hay thay đổi ngôn ngữ. Các ứng dụng có thể khởi động quảng bá, ví dụ để các ứng dụng khác biết rằng dữ liệu đã được tải về xong trên thiết bị và sẵn sàng sử dụng.

Một ứng dụng có thể có bất kỳ số lượng bộ nhận quảng bá nào để nhận những thông báo quan trọng với nó. Tất cả các bộ nhận quảng bá được kế thừa từ lớp BroadcastReceiver.

Bộ nhận quảng bá không có giao diện. Tuy nhiên, chúng có thể khởi động một hoạt động để đáp lại thông tin mà nó nhận được, hay chúng có thể sử dụng NotificationManager để thông báo người dùng biết. Các thông báo có thể được sự chú ý của người dùng theo các cách các nhau như là sáng màn hình, rung thiết bị, bật âm thanh nào đấy… Thông thường, chúng đặt thông báo trên thanh trạng thái, nơi người dùng có thể nhận được thông báo.

### 2.4.4. Content provider

#### Khái niệm

Các ứng dụng có thể lưu trữ dữ liệu của mình trong các tập tin hoặc sử dụng cơ sở dữ liệu SQLite sẵn có v.v… Content Provider có chức năng cung cấp một tập hợp các phương thức cho phép một ứng dụng có thể lưu trữ và lấy dữ liệu được quản lý bởi content provider đó.

Content Provider là một đặc trưng riêng của Android, nhờ đó mà các ứng dụng có thể chia sẻ dữ liệu với nhau một cách dễ dàng

### 2.4.5. Các thành phần kích hoạt (các Intent)

Content provider được kích hoạt khi chúng được gọi từ một ContentResolver. Ba thành phần khác (hoạt động, dịch vụ và bộ nhận quảng bá) được kích hoạt bởi thông điệp không đồng bộ từ các intent. Một intent là một đối tượng có kiểu Intent chứa nội dung của thông điệp. Với các hoạt động và dịch vụ, nó gọi tên hành động được yêu cầu và xác định URI của dữ liệu tác động tới ở giữa. Ví dụ, nó có thể truyền tải một yêu cầu cho một hoạt động hiển thị một ảnh cho người dùng hay cho phép người dùng sửa văn bản. Với bộ nhận quảng bá, đối tượng Intent gọi tên của hành động được thông báo. Ví dụ, bộ nhận quản bá có thể thông báo các phần nó quan tâm là nút chụp ảnh đã được bấm.

*Có vài phương thức cho việc kích hoạt mỗi thành phần:*

Một hoạt động được khởi chạy thông qua một đối tượng Intent Context.startActivity() hay Activity.startActivityForResult(). Hoạt động đáp lại có thể theo dõi intent được tạo ra đó bằng phương thức getIntent() và cập nhật thông qua phương thức setIntent(Intent). Android gọi phương thức onNewIntent() để bỏ qua các intent đến trước nó.

Một hoạt động thường bắt đầu hoạt động khác. Nếu nó muốn trả lại kết quả hoạt động nó đã khởi chạy, nó sẽ gọi phương thức startActivityForResult() thay cho phương thức startActivity(). Ví dụ, nếu nó khởi chạy một hoạt động mà cho phép người dùng lấy một ảnh, nó có thể muốn lấy kết quả của ảnh được chọn. Kết quả được trả về trong một đối tượng Intent thông qua phương thức onActivityResult().

Một dịch vụ được bắt đầu thông qua một đối tượng Intent là Context.startService(). Android gọi phương thức onStart() của dịch vụ và thông qua đối tượng Intent của nó.

Tương tự, một intent có thể thông qua Context.bindService() để thiết lập một kết nối liên tục giữa các thành phần và dịch vụ đích. Dịch vụ nhận đối tượng Intent qua lời gọi onBind() (nếu dịch vụ chưa được chạy, bindService() có thể chọn bắt đầu nó). Cho ví dụ, một hoạt động có thể thiết lập kết nối với dịch vụ chơi nhạc đề cập ở phần trước để nó có thể cung cấp cho người dùng giao diện sử dụng để điều khiển chơi lại. Hoạt động sẽ gọi bindService để thiết lập kết nối và sau đó gọi phương thức đã định nghĩa bởi dịch vụ để áp dụng chơi lại ca khúc.

Một ứng dụng có thể khởi tạo một quảng bá thông qua đối tượng Intent bằng phương thức như Context.setBroadcast(), Context.setOrderedBroadcast() và Context.sendStickyBroadcast(). Android chuyển những intent tới tất cả các bộ nhận quảng bá nào quan tâm bằng việc gọi phương thức onReceive() của nó.

### 2.4.6. Ngắt một thành phần

Một hoạt động có thể bị ngắt thông qua việc gọi phương thức finish(). Một hoạt động có thể tắt một hoạt động khác (hoạt động đó được khởi động với lời gọi startActivityForResult()) bằng việc gọi finishActivity().

Để ngắt một dịch vụ chúng ta có thể sử dụng phương thức stopSelf(), hoặc bằng cách gọi Context.stopService().

Content provider được kích hoạt chỉ khi nó đáp lại yêu cầu từ một ContentResolver. Bộ nhận quảng bá được kích hoạt chỉ khi nó đáp lại một thông điệp quảng bá. Chúng ta không cần thiết phải tắt các thành phần này.

### 2.4.7. Tập tin khai báo (manifest)

Trước khi có thể khởi chạy một ứng dụng thành phần, nó phải xem ứng dụng bao gồm những thành phần nào. Thêm nữa, các ứng dụng khai báo các thành phần của nó trong một tập tin khai báo để đóng gói lại vào trong gói Android (tập tin .apk chứa các mã nguồn, tập tin và tài nguyên).

Tập tin này có cấu trúc của tập tin XML và luôn có tên là AndroidManifest.xml trong mọi ứng dụng. Nó làm một số thứ như thêm và khai báo các thành phần của ứng dụng, tên các thư viện ứng dụng cần liên kết tới (ngoài thư viện chuẩn của Android) và xác định các quyền cho ứng dụng.

Tuy nhiên, nhiệm vụ chính của tập tin khai báo là khai báo các thành phần của ứng dụng. Một hoạt động có thể được khai báo như sau:

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

<manifest . . . >

<application . . . >

<activity android:name="com.example.project.FreneticActivity"

android:icon="@drawable/small\_pic.png"

android:label="@string/freneticLabel"

. . . >

</activity>

. . .

</application>

</manifest>

Thuộc tính “name” của phần tử <activity> là tên các lớp con lớp Activity đã được cài đặt, thuộc tính “icon” và “label” trỏ đến tập tin tài nguyên chứa biểu tượng và nhãn được hiển thị cho người dùng.

Các thành phần khác được khai báo theo cách tương tự: <service> dịch vụ, <receiver> bộ nhận quảng bá và <provider> content provider. Các hoạt động, dịch vụ và content provider có thể cùng được khai báo trong tập tin khai báo hoặc có thể được tạo tự động trong mã (như đối tượng BroadcastReceiver) và được đăng ký với hệ thống bằng cách gọi Context.registerReceiver().

### 2.4.8. Bộ lọc Intent

Một đối tượng Intent có thể có tên rõ ràng trong thành phần đích. Nếu có, Android sẽ tìm thành phần đó (dựa trên khai báo trong tập tin khai báo) và kích hoạt nó. Nhưng nếu đích có tên không rõ ràng, Android phải xác định thành phần nào thích hợp nhất để đáp lại intent. Nó thực hiện so sánh đối tượng Intent với bộ lọc intent trong các đích có khả năng. Một thành phần của bộ lọc intent cho Android biết loại intent thành phần nào cần xử lý. Giống các thông tin cơ bản, chúng được khai trong tập tin khai báo. Đây là một ví dụ để thêm hai bộ lọc intent cho một hoạt động.

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

<manifest . . . >

<application . . . >

<activity android:name=

"com.example.project.FreneticActivity"

android:icon="@drawable/small\_pic.png"

android:label="@string/freneticLabel"

. . . >

<intent-filter . . . >

<action android:name=

"android.intent.action.MAIN" />

<category android:name=

"android.intent.category.LAUNCHER" />

</intent-filter>

<intent-filter . . . >

<action android:name=

"com.example.project.BOUNCE" />

<data android:mimeType="image/jpeg" />

<category android:name=

"android.intent.category.DEFAULT" />

</intent-filter>

</activity>

. . .

</application>

</manifest>

Bộ lọc đầu tiên – là sự kết hợp của hành động “android.intent.action.Main” và danh mục “android.intent.category.LAUNCHER” – là một ví dụ phổ biến. Nó đánh dấu lại một Intent và mô tả lại cho bộ khởi chạy ứng dụng, đặt vào trong danh mục các ứng dụng trên điện thoại. Nói theo cách khác, hoạt động làm lối vào cho ứng dụng sẽ được khởi chạy, người dùng sẽ thấy khi chọn khởi chạy ứng dụng.

Bộ lọc thứ hai được khai báo để hoạt động có thể thực thi trên một kiểu dữ liệu đặc biệt.

Một thành phần có thể có bao nhiêu số bộ lọc intent tùy thích, mỗi một cái lại được khai báo khác nhau cho các khả năng. Nếu nó không có một bộ lọc nào, nó có thể được được kích hoạt bởi các intent khác gọi đến.

Để tạo và đăng ký một bộ nhận quảng bá trong mã nguồn, bộ lọc intent tạo thẳng một đối tượng giống đối tượng IntentFilter. Tất cả các bộ lọc khác đều được cài đặt trong tập tin khai báo.

## 2.5. Công cụ hỗ trợ lập trình Android

Bộ công cụ phát triển phần mềm Android SDK gồm nhiều công cụ trợ giúp cho việc phát triển ứng dụng di động trên nền tảng Android. Thành phần quan trọng nhất của bộ công cụ này là trình giả lập Android và bộ plug-in phát triển ứng dụng Android trên Eclipse ADT, bên cạnh đó bộ SDK cũng bao gồm các công cụ khác cho việc gỡ rối, đóng gói và cài đặt ứng dụng trên trình giả lập và trên thiết bị.

**Trình giả lập Android**: Thực chất đây là một chương trình mô phỏng một thiết bị di động ảo trên máy tính. Chúng ta có thể sử dụng bộ giả lập này để thiết kế gỡ rối và kiểm tra ứng dụng của mình như trên thiết bị trước khi đưa chương trình vào thiết bị thật.

**Android Development Tools Plugin (cho Eclipse)** viết tắt là ADT, Plugin này hỗ trợ cho việc tạo và gỡ rối các ứng dụng và sử dụng môi trường phát triển tích hợp Eclipse trở nên dễ dàng hơn.

**Trình gỡ rối Dalvik Debug Monitor Service (DDMS)**: giúp quản lý các tiến trình trong bộ giả lập hoặc thiết bị và trợ giúp việc gỡ rối ứng dụng. Có thể sử dụng công cụ này để hủy các tiến trình, chọn một tiến trình cụ thể để gỡ rối, sinh các dữ liệu truy vết, xem bộ nhớ heap và thông tin về các luồng, chụp ảnh màn hình của trình giả lập.

**Android Debug Bridge (adb)**: Công cụ này cho phép cài đặt các tập tin .apk trên bộ giả lập hoặc thiết bị, đồng thời hỗ trợ truy cập hai đối tượng này từ cửa sổ dòng lệnh.

**Android Asset Packaging Tool (aapt)**: Công cụ để tạo tập tin .apk chứa các mã nhị phân và tài nguyên của một ứng dụng Android.

**sqlite3**: Đây là công cụ để truy cập các tập tin dữ liệu SQLite được tạo ra và sử dụng bởi một ứng dụng Android.

**dx**: Đây là công cụ biên dịch các tập tin .class thành mã bytecode chạy trên máy ảo Dalvik (lưu trữ dưới dạng tập tin .dex).

**mksdcard**: Giúp tạo một tập tin ảnh lưu trữ dữ liệu mà ta có thể sử dụng trên bộ giả lập, để mô phỏng sự có mặt của một thẻ nhớ ngoài (như thẻ SD).

**Android Virtual Devices (AVD)**: Cho phép chúng ta có thể tạo cấu hình thiết bị ảo, mô phỏng các đặc điểm của bộ giả lập Android. Với mỗi cấu hình, ta có thể xác định nền tảng cho nó chạy, tùy chỉnh phần cứng hay giao diện sử dụng. Mỗi tùy chỉnh của AVD sẽ như một thiết bị độc lập với dữ liệu lưu trữ của nó, thẻ nhớ SD, v.v…

Phần 3: Google Map API và các dịch vụ google map trên Android

Trong thế giới web ngày nay, các giải pháp bản đồ là một thành phần không thể thiếu. Chúng ta sử dụng chúng để thấy vị trí của mọi vật, để tìm vị trí của một địa chỉ, để lấy về thông tin dẫn đường, và làm nhiều thứ khác. Hầu hết thông tin có một vị trí, và nếu vật nào đó có một vị trí thì nó có thể hiển thị trên một bản đồ.

Có một số công ty cung cấp các giải pháp về bản đồ như Yahoo! Maps và Bing Maps, nhưng phổ biến nhất vẫn là google maps. Thực tế theo trang programmableweb.com, google maps là API phổ biến nhất trên internet. Theo thống kê của trang này tháng 5/2010, 43% là sử dụng Google Maps API.



Một trong hình ảnh nhìn từ vệ tinh trên Google Maps

Phần này sẽ trình bày chi tiết về API của google maps cũng như sử dụng và đăng ký dịch vụ này trên di động cụ thể là trên android.

3.1. Đăng ký API Key từ google maps

Để sử dụng được các dịch vụ và API của google maps bạn cần phải đăng ký với dịch vụ Google Maps và đồng ý với các điều khoản dịch vụ áp dụng trước khi bạn lấy dữ liệu từ Google Maps. Điều này sẽ được áp dụng khi bạn đang phát triển ứng dụng của bạn trên máy giả lập hoặc chuẩn bị đưa ứng dụng của bạn triển khai đến các thiết bị di động. Việc đăng ký một Maps API key rất đơn giản, miễn phí, bao gồm hai phần:

* Đăng ký mã MD5 mà bạn sẽ sử dụng trong ứng dụng của mình. Service sau đó sẽ cung cấp cho bạn một Maps API Key.
* Thêm một đoạn tham chiếu đến Maps API trong mỗi MapView. Bạn có thể sử dụng Maps API Key cho bất kỳ MapView trong bất kỳ ứng dụng Android, cung cấp ứng dụng được ký kết với giấy chứng nhận mã MD5 mà bạn đã đăng ký với service.

3.1.1. Giới thiệu

Để đảm bảo rằng các ứng dụng sử dụng dữ liệu bản đồ một cách phù hợp, Google Maps service yêu cầu nguời phát triển ứng dụng đăng ký với service, đồng ý với điều khoản và cung cấp một mã MD5. Đối với mã đăng ký, service cung cấp cho chúng ta một Maps API Key - một chuỗi chữ số xác định duy nhất.

Google maps service cũng đòi hỏi mỗi MapView xác định tới service bằng các sử dụng Maps Api Key. Trước khi cung cấp map cho MapView, Service kiểm tra Maps API key được cung cấp bởi MapView Để đảm bảo rằng:

Để đăng ký một Maps API Key, bạn cần phải cung cấp dấu MD5 fingerprint mà bạn sẽ sử dụng để ký vào đơn xin của bạn.

Chạy Keytool với tham số -list. Bảng dưới đây liệt kê danh sách mà bạn có thể sử dụng:

|  |  |
| --- | --- |
| Keytool | Mô tả |
| -list | In ra mã MD5 |
| -keystore <keystore-name>.keystore | Tên của keystore chứa key đích |
| -storepass <password> | Mật khẩu cho keystore |
| -alias <alias\_name> | Các bí danh cho key để tạo ra các mã MD5. |
| -keypass <password> | Các mật khẩu cho key. |

3.1.2. Lấy mã MD5

Trước tiên cần phải xác định vị trí của file debug.keystore, đường dẫn thường được sử dụng là:

* Windows Vista & Windows 7: C:\Users\<user>\.android\debug.keystore
* Windows XP: C:\Documents and Settings\<user>\.android\debug.keystore
* OS X và Linux: ~/.android/debug.keystore

Nếu sử dụng Eclipse thì có thể vào Windows->Prefs->Android->Build để lấy đường dẫn.

Sau khi lấy được đường dẫn của debug.keystore. Bật cmd và chạy lệnh sau để lấy mã MD5 fingerprint:

keytool -list -alias androiddebugkey -keystore <path\_to\_debug\_keystore>.keystore -storepass android -keypass android

Kết quả của dòng lệnh trên như sau:

46:2C:DD:3F:5A:4E:97:6E:6E:7F:DD:A3:AD:90:FB:73

**3.1.3. Lấy key Google Maps API**

Chúng ta truy cập vào đường dẫn sau: <http://code.google.com/android/maps-api-signup.html> và làm theo các bước:

* Nếu chưa có tài khoản của google thì cần tạo một tài khoản mới
* Đồng ý các điều khoản từ Android maps API
* Chép mã MD5 đã tạo ở trên vào khung và chọn Generate API Key, trang web này sẽ tạo cho bạn một key.

Sau khi đã đăng ký với dịch vụ Google Maps và đã có được một Maps API Key, bạn phải bổ sung thêm cho các đối tượng MapView trong ứng dụng của bạn.

Đối với các yếu tố <MapView> khai báo trong các tập tin bố trí XML, thêm Maps API chính là giá trị của một thuộc tính đặc biệt - Android: apiKey.

<com.google.android.maps.MapView

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:enabled="true"

android:clickable="true"

android:apiKey="example\_Maps\_ApiKey\_String"

/>

**3.2. Các tham số truy cập trong Google Maps**

Để truy cập thông tin dự liệu từ Google Map chúng ta cần biết được tham số truy cập của url cũng như các quy luật truy cập. Dưới đây là danh sách một số tham số có thể truyền vào maps.google.com. Chúng ta có thể truyền nhiều tham số nếu muốn Google Maps hiển thị thông tin chỉ đường.

**3.2.1. Các tham số chung**

* Tham số q=

Query – bất cứ những gì truyền trong tham số này sẽ được đối xử như những gì được nhập vào khung truy vấn trên trang maps.google.com. Cụ thể:

* Một địa chỉ chính xác được tìm thấy sẽ được đánh dấu và một hộp thoại thông tin được hiển thị tại điểm cụ thể đó.
* Tên một thành phố, hay phân nửa chữ số đầu tiên của một mã thư tín được truyền vào thì khu vực này sẽ được đánh dấu và một hộp thoại thông ti.
* Một vị trí chính xác có thể được chỉ định bằng kinh độ, vĩ độ dạng thập phân (52.123N,2.456W hay 52.123,-2.456), độ, phút, giây (52 7 22.8N,2 27 21.6W hay 52 7 22.8,-2 27 21.6) hay độ và phút (52 7.38N,2 27.36W hay 52 7.38,-2 27.36).
* Tham số truyền vào ‘to’ và ‘from’ thì tính năng chỉ đường sẽ được kích hoạt.
* Tham số truyền vào ‘near’ hay ‘loc:’ thì tính năng tìm kiếm local sẽ được kích hoạt.Một vị trí cũng có thể được chỉ định sau chữ @ dạng thập phân (@52.123, -2.456). Ở dạng này kết quả trả về sẽ có mức độ phóng to/thu nhỏ thấp hơn so với dạng truy vấn bình thường không có @. Sử dụng dạng này sẽ gây khó khăn với các lựa chọn khác của tham số truy vấn. Ký hiệu @ thay vì “near” hay “loc:” thì sẽ dẫn đến chỉ một kết quả được hiển thị. Hãy so sánh câu truy vấn: “Church near Hallows Road Blackpool, Lancashire” sẽ trả về hơn 1000 kết quả trong khi với câu truy vấn “Church @ Hallows Road Blackpool, Lancashire” chỉ trả về 1 kết quả.
* Để đặt một hình đánh dấu tự chọn tại một vị trí bất kỳ, q=10+Downing+Str+is+not+here@51.50335,-0.227721

Đây không được là tên một quốc gia, nếu không thì không có maker nào được hiển thị; nhưng nó có thể rỗng khi tọa độ được sử dụng trong câu truy vấn.

* Việc thêm một nhãn “()” rất hữu ích để gợi ý cho Google Maps tọa độ cung cấp được sử dụng. Hãy xem xét một số câu truy vấn dưới đây:
* <http://maps.google.com/maps?q=24.197611,120.780512> :Rất không may là vị trí của chúng ta sẽ xuất hiện nằm gần một vài vị trí khác Misleading Distracting Known Point.
* <http://maps.google.com/maps?q=Dan@24.197611,120.780512> Nhãn được mở rộng đến một vài tên sân bay khác (Mặc dầu vị trí không vẫn không thay đổi)
* <http://maps.google.com/maps?q=24.197611,120.780512+(Dan)>  Dạng này có vẽ an toàn cho trong việc loại bỏ các kết quả mở rộng.
* <http://maps.google.com/maps?q=loc:24.197611,120.780512> Một dạng truy vấn khác nhưng cũng làm việc hiệu quả.
* Tham số near=

Có thể được dùng như một phần của một câu truy vấn thay vì đặt mọi thứ vào q=

* Tham số g=

Một địa chỉ/vị trí, cung cấp thêm ngữ cảnh phụ cho tham số q. Google Maps chính nó đã lưu thông tin tìm kiếm tại tham số này. Lưu ý rằng: nếu lần đầu tiên tìm kiếm nó có thể chứa vị trí bắt đầu.

3.2.2. Chế độ tìm kiếm

* mrt = chỉ rõ loại tìm kiếm:
* mrt= all: yêu cầu mọi thứ
* mrt=loc: tìm kiếm vị trí. Cần có tham số q=
* mrt=yp: tìm kiếm việc kinh doanh. Sẽ cần tham số q=.
* mrt=websearch: các trang web liên quan.
* mrt= realestate: tìm kiếm bất động sản.
* mrt =ds: các bản đồ liên quan – chỉ số Google của GeoRSS
* start= Bỏ qua thông tin đầu tiên được tìm thấy.
* num=Hiển thị số lượng kết quả được tìm thấy. Dãy hợp lệ từ 0-20

**3.2.3. Vị trí**

|  |  |
| --- | --- |
|  | |
| ll= | Kinh độ , vĩ độ trung tâm bản đồ. Chỉ chấp nhận tọa độ thập phân. Nếu sử dụng tham số này mà không có câu truy vấn, thì bản đồ được trỏ đến vị trí trung tâm tại điểm được cho nhưng không hiển thị maker hay cửa sổ thông tin. |
| sll= | Kinh độ, vĩ độ của điểm từ việc tìm kiếm kinh doanh được thực hiện. Bạn có thể sử dụng tham số này để thực hiện một tìm kiếm kinh doanh ngoài mặt phẳng dọc tâm. |
| Spn | Tính xấp xỉ lat/long. Mức phóng to/thu nhỏ sẽ được điều chỉnh cho vừa nếu không có tham số z=. Có thể sử dụng công cụ Mapki để tìm kinh độ và vĩ độ. |
| latlng= | Đây là một điều huỳnh bí. Nó nhận vào 3 số được cách bởi dấu phẩy. Hai số đầu tiên (biểu diễn vị độ và kinh độ được nhân lên gấp 1000000 lần) được bỏ qua. Số thứ ba xem như là số nội bộ Google “Company ID” cho doanh nghiệp đặc biệt. Ví dụ: latlng=latlng=0,0,14944637421527611642 biểu diễn cho Blackpool Community Church. |
| cid= | Giống như kinh độ và vĩ độ, nhưng tạo ra một kích thước bản đồ khác. Nó nhận vào 3 số cách nhau bởi dấu phẩy. |
| geocode= | Dãy các giá trị “geocode” được mã hóa cho các điểm đường đi được sử dụng trong dẫn đường. Xuất hiện khi các điểm dẫn đường trong “from: A to: B” không rõ rang. Ví dụ: &saddr=Were+St&daddr=Kings+Hwy+to:Princes+Hwy+to:Princes+Hwy+to:Monaro+Hwy+to:-35.43483,149.112175&hl=en&geocode=FbBP4\_0dSEXjCA%3BFWIj4\_0dpMDtCA%3BFe4g3\_0dunvzCA%3BFfBf0P0dEGzuCA%3BFSoL1\_0dtorjCA%3B là kết quả cuối cùng từ việc ghép các giá trị cho của mỗi điểm đường đi với %3B. |
| Radius | Giới hạn kết quả với bán kính xác định. Cần có tham số **sll** hay điểm trung tâm để có thể thực hiện. |

**3.2.4. Hiển thị bản đồ**

* t= loại bản đồ. Các lựa chọn gồm có: “m” (bản đồ-map), “k” (vệ tinh), “h” (hỗn tạp), “p” (địa hình vật chất), “e” (GoogleEarth).
* z= thiết lập chế độ phóng to thu nhỏ (Phạm vi từ 1 đến 20).
* Layer = kích hoạt overlay. Các lựa chọn sẵn có “t” (traffic – giao thông), “c” (street view).
* lci = kích hoạt các lớp vật bề mặt, cách nhau dấu phẩy
* com.panoramio.all: hình ảnh
* com.youtube.all: Video (từ YouTube)
* org.wiikipedia.en: Wikipedia (tiếng Anh)
* com.google.ugc.c752d13e87c4fbd7 : webcam
* bike: tuyến đường xe đạp
* view= có thể được sử dụng để chọn xem kiểu chữ (view=text) hay xem kiểu bản đồ (view=map).

**3.2.5. Dẫn đường**

* saddr= địa chỉ nguồn. Sử dụng tham số này khi yêu cầu dẫn đường lái xe. Khi phát triển trên nền tảng di động, để sử dụng MyLocatiion (tọa độ GPS) ta thiết lập saddr trống “” đối với Android, đối với iPhone, thiết lập saddr là “Current%20Location”.
* daddr= các địa chỉ đích. Sử dụng khi yêu cầu thông tin dẫn đường.
* dirflg: loại đường
* dirflg=h: chuyển sang chế độ tìm kiếm “Tránh đường cao tốc” (Avoid Highway).
* dirflg=t: Chuyển sang chế độ tìm kiếm “Tránh trạm thu phí” (Avoid Tolls)
* dirflg=r: chuyển sang chế độ tìm kiếm “Vận tải công cộng” (Public Transit), chỉ làm thực hiện được tại một số khu vực.
* dirflg=w: dẫn đường cho người đi bộ
* dirflg=b: dẫn đường cho xe đạp – chỉ một số khi vực.
* doflg = Đơn vị khoảng cách. (mặc định đơn vị đang lưu hành của nước hiện tại)
* doflg=ptk: kilomet
* doflg=ptm: dặm (mile)

**3.2.6. Quản lý thông tin đầu ra**

* hl = ngôn ngữ của host. Chỉ một số ngôn ngữ được hỗ trợ. Ví dụ: để sử dụng tiếng việt ta chọn hl=vi.
* ie= có thể được sử dụng để chỉ định rõ tập hợp mã hóa ký tự đầu vào. Ví dụ: ie=UTF8.
* ie = có thể được sử dụng để chỉ định rõ tập hợp mã hóa ký tự đầu vào.
* output = định dạng dữ liệu xuất.
* output=html: sử dụng định dạng trang Google Local phong cách cũ trước khi được hợp nhất với Google Maps.
* output =kml: xuất ra một file KML chứa thông tin biểu diễn bản đồ hiện tại.
* output=embed: xuất ra file HTML phù hợp cho việc tích hợp vào các site cung cấp thứ ba.
* Output = dragdir trả về một đối tượng JSON chứa geocode được chuyển đổi và một tập nhiều đường thẳng sử dụng trong tham số saddr (địa chỉ bắt đầu của tuyến đường) và daddr (điểm cuối của tuyến đường).
* v = khi oupt=kml được chọn thì tham số này chỉ rõ phiên bản của file kml. Mặc định từ phiên bản 2.0 đến 2.2.

**3.3. File thông tin KML**

KML là một định dạng file để hiển thị dữ liệu địa lý trong một trình duyệt Earth như Google Earth, Google Maps, Google Maps cho di động. KML sử dụng một cấu trúc dựa trên tag với các thành phần lồng nhau và các thuộc tính và được dựa trên chuẩn XML. Để lấy được dữ liệu từ google map trả về chúng ta có thể phân tích và đọc ra các thông tin cần thiết.

**3.3.1. Placemark**

Một placemark là một trong những tính năng được sử dụng phổ biến trong Google Earth. Nó đánh dấu một vị trí trên bề mặt của Earth, sử dụng một icon hình cái đinh màu vàng. Một Placemark đơn giản nhất chỉ gồm một thành phần <Point>, chỉ rõ vị trí của Placemark. Chúng ta có thể chỉ rõ tên và một icon túy ý cho Placemark, và cũng có thể thêm các thành phần hình học vào nó. Dưới đây là một file code placemark đơn giản :

<?xml version="1.0" encoding="UTF-8"?>

<kml xmlns="http://www.opengis.net/kml/2.2">

<Placemark>

<name>Simple placemark</name>

<description>Attached to the ground. Intelligently places itself

at the height of the underlying terrain.</description>

<Point>

<coordinates>-122.0822035425683,37.42228990140251,0</coordinates>

</Point>

</Placemark>

</kml>

Cấu trúc file được chia như sau:

* Phần đầu là thông tin của file XML. Đây là phần nằm ở dòng thứ nhất của mỗi file kml. Không có khoảng trắng hay ký tự nào xuất hiện trước dòng này.
* Một phần khai báo về namespace KML. Đây là dòng thứ 2 trong mỗi file KML 2.2.
* Một đối tượng Placemark chứa các thành phần sau:
* Một tag *name* được sử dụng như một tên cho Placemark.
* Một tag *description* xuất hiện trong “balloon” đính kèm vào Placemark

Một tag *Point* chỉ ra vị trí của một Placemark trên bề mặt của Earth (*longitude*, *latitude*, và một thuộc tính tùy chọn *altitude*).

**3.3.2. Đường đi (Path)**

Nhiều loại đưởng đi khác nhau có thể được tạo ra trong Google Earth. Trong KML, một đường đi được tạo ra bởi thành phần <LineString>. Hãy xem một đọa code trong trong file KML đưới đây bạn sẽ thấy đường đi được tại ra như thế nào:

<?xml version="1.0" encoding="UTF-8"?>

<kml xmlns="http://www.opengis.net/kml/2.2">

<Document>

<name>Paths</name>

<description>Examples of paths. Note that the tessellate tag is by default

set to 0. If you want to create tessellated lines, they must be authored

(or edited) directly in KML.</description>

<Style id="yellowLineGreenPoly">

<LineStyle>

<color>7f00ffff</color>

<width>4</width>

</LineStyle>

<PolyStyle>

<color>7f00ff00</color>

</PolyStyle>

</Style>

<Placemark>

<name>Absolute Extruded</name>

<description>Transparent green wall with yellow outlines</description>

<styleUrl>#yellowLineGreenPoly</styleUrl>

<LineString>

<extrude>1</extrude>

<tessellate>1</tessellate>

<altitudeMode>absolute</altitudeMode>

<coordinates> -112.2550785337791,36.07954952145647,2357

-112.2549277039738,36.08117083492122,2357

-112.2552505069063,36.08260761307279,2357

-112.2564540158376,36.08395660588506,2357

-112.2580238976449,36.08511401044813,2357

-112.2595218489022,36.08584355239394,2357

-112.2608216347552,36.08612634548589,2357

-112.262073428656,36.08626019085147,2357

-112.2633204928495,36.08621519860091,2357

-112.2644963846444,36.08627897945274,2357

-112.2656969554589,36.08649599090644,2357

</coordinates>

</LineString>

</Placemark>

</Document>

</kml>

3.3.3. Truy xuất thông tin dữ liệu từ file KML

Để truy xuất thông tin từ file KML mà google map trả về ta sử dụng XML DOM. Đây là mô hình đối tượng và giao diện lập trình chuẩn cho văn bản XML. XML DOM định nghĩa các đối tượng và thuộc tính của tất cả thành phần XML, và các phương thức (interface) để truy cập chúng.

* Truy xuất thông tin địa điểm (Places)

Ta hãy xem xét đoạn code dưới để lấy về thông tin đia điểm

ArrayList<PlaceModel> lstPlaceMode = new ArrayList<PlaceModel>();

//Build url

StringBuilder urlString = new StringBuilder();

urlString.append("http://maps.google.com/maps?q=");

urlString.append(keyword); //nội dung tìm kiếm

urlString.append(Double.toString(lat));

urlString.append(",");

urlString.append(Double.toString(lng));

urlString.append("&num=");

urlString.append(Integer.toString(num));

urlString.append("&ie=UTF8&output=kml");

// get the kml (XML) doc. And parse it. Lất về file kml (xml). Và đôc thông tin

Document doc = null;

HttpURLConnection urlConnection= null;

URL url = null;

try

{

//Declare connection

url = new URL(urlString.toString());

urlConnection=(HttpURLConnection)url.openConnection();

urlConnection.setRequestMethod("GET");

urlConnection.setDoOutput(true);

urlConnection.setDoInput(true);

urlConnection.connect();

DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

DocumentBuilder db = dbf.newDocumentBuilder();

doc = db.parse(urlConnection.getInputStream());

if(doc.getElementsByTagName("Placemark").getLength()>0)

{

for(int i=0;i<doc.getElementsByTagName("Placemark").getLength();i++)

{

PlaceModel placeModel = new PlaceModel();

//Get ATM name

String name = doc.getElementsByTagName("Placemark")

.item(i).getChildNodes().item(0)

.getFirstChild().getNodeValue();

//Get the address of ATM

String address = doc.getElementsByTagName("Placemark")

.item(i).getChildNodes().item(2)

.getFirstChild().getNodeValue();

placeModel.setName(name);

placeModel.setAddress(address);

lstPlaceMode.add(placeModel);

}

}

}

catch (MalformedURLException e)

{

e.printStackTrace();

}

catch (IOException e)

{

e.printStackTrace();

}

catch (ParserConfigurationException e)

{

e.printStackTrace();

}

catch (SAXException e)

{

e.printStackTrace();

}

Đầu tiên chúng ta để ý đến đối tượng PlaceModel đây là đối tượng để chứa các thông tin trả về gồm: tên địa điểm (name), địa chỉ của địa điểm (addresss) và tọa độ trên bản đồ (lat & lng).

Đoạn code:

StringBuilder urlString = new StringBuilder();

urlString.append("http://maps.google.com/maps?q=");

urlString.append(keyword); //nội dung tìm kiếm

urlString.append(Double.toString(lat));

urlString.append(",");

urlString.append(Double.toString(lng));

urlString.append("&num=");

urlString.append(Integer.toString(num));

urlString.append("&ie=UTF8&output=kml");

đây là nơi chúng ta sẽ xây dựng url cho việc lấy thông tin từ google map trong đó q=keyword là địa điểm chúng ta sẽ tìm kiếm, num= số lượng địa điểm tối đa được trả về, output=kml định dạng file trả về(ở đây chúng ta sẽ lấy về file KML để xử lý).

Tiếp theo:

String name = doc.getElementsByTagName("Placemark").item(i).getChildNodes().item(0) .getFirstChild().getNodeValue();

Đoạn code này sẽ đọc thông tin từ tag <name> trong file, và tương tự:

//Get the address of ATM

String address = doc.getElementsByTagName("Placemark")

.item(i).getChildNodes().item(2) .getFirstChild().getNodeValue();

Sẽ đọc thông tin từ <address> trong file KML.

Dưới đây là node thông tin trích ra từ file KML trả về:

<Placemark>

<name>HSBC</name>

<Snippet><![CDATA[No. 6 Nhà Th?, Hoan Kiem, Hanoi, Vietnam<br/>]]></Snippet>

<address>No. 6 Nhà Th?, Hoan Kiem&lt;br/&gt;Hanoi, Vietnam</address>

<StyleMap>

<Pair>

<key>normal</key>

<Style>

<IconStyle>

<Icon>

<href>http://maps.gstatic.com/intl/en\_ALL/mapfiles/kml/paddle/A.png</href>

</Icon>

<hotSpot x="0.500000" y="0.000000" xunits="fraction" yunits="fraction" />

</IconStyle>

<ListStyle>

<ItemIcon>

<href>http://maps.gstatic.com/intl/en\_ALL/mapfiles/kml/paddle/A-lv.png</href>

</ItemIcon>

</ListStyle>

</Style>

</Pair>

<Pair>

<key>highlight</key>

<Style>

<IconStyle>

<scale>1.300000</scale>

<Icon>

<href>http://maps.gstatic.com/intl/en\_ALL/mapfiles/kml/paddle/A.png</href>

</Icon>

<hotSpot x="0.500000" y="0.000000" xunits="fraction" yunits="fraction" />

</IconStyle>

<ListStyle>

<ItemIcon>

<href>http://maps.gstatic.com/intl/en\_ALL/mapfiles/kml/paddle/A-lv.png</href>

</ItemIcon>

</ListStyle>

</Style>

</Pair>

</StyleMap>

<Point>

<coordinates>105.850385,21.029302,0</coordinates>

</Point>

</Placemark>

* Truy xuất thông tin đường đi (driving direction)

Tương tự đọa code dưới đây sẽ đọc thông tin tập hợp các điểm của một đường đi từ file KML mà google map trả về. Trong đó ta thấy ta đã sử dụng từ khóa saddr và daddr là địa điểm nguồn và địa điểm đích của đoạn đường

// Khai báo thông tin url đến google map

StringBuilder urlString = new StringBuilder();

urlString.append("http://maps.google.com/maps?f=d&hl=en");

urlString.append("&saddr=");//from

urlString.append( Double.toString((double)src.getLatitudeE6()/1.0E6 ));

urlString.append(",");

urlString.append( Double.toString((double)src.getLongitudeE6()/1.0E6 ));

urlString.append("&daddr=");//to

urlString.append( Double.toString((double)dest.getLatitudeE6()/1.0E6 ));

urlString.append(",");

urlString.append( Double.toString((double)dest.getLongitudeE6()/1.0E6 ));

urlString.append("&ie=UTF8&0&om=0&output=kml");

Log.d("xxx","URL="+urlString.toString());

//Lấy về file kml và đọc thông tin tọa độ

Document doc = null;

HttpURLConnection urlConnection= null;

URL url = null;

try

{

url = new URL(urlString.toString());

urlConnection=(HttpURLConnection)url.openConnection();

urlConnection.setRequestMethod("GET");

urlConnection.setDoOutput(true);

urlConnection.setDoInput(true);

urlConnection.connect();

DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

DocumentBuilder db = dbf.newDocumentBuilder();

doc = db.parse(urlConnection.getInputStream());

if(doc.getElementsByTagName("GeometryCollection").getLength()>0)

{

String path = doc.getElementsByTagName("GeometryCollection").item(0).getFirstChild().getFirstChild().getFirstChild().getNodeValue() ;

String [] pairs = path.split(" ");

String[] lngLat = pairs[0].split(","); // lngLat[0]=longitude lngLat[1]=latitude lngLat[2]=height

// src

GeoPoint startGP = new GeoPoint((int)(Double.parseDouble(lngLat[1])\*1E6),(int)(Double.parseDouble(lngLat[0])\*1E6));

mMapView01.getOverlays().add(new MyOverLay(startGP,startGP,1));

GeoPoint gp1;

GeoPoint gp2 = startGP;

for(int i=1;i<pairs.length;i++)

{

lngLat = pairs[i].split(",");

gp1 = gp2;

gp2 = new GeoPoint((int)(Double.parseDouble(lngLat[1])\*1E6),(int)(Double.parseDouble(lngLat[0])\*1E6));

mMapView01.getOverlays().add(new MyOverLay(gp1,gp2,2,color));

}

mMapView01.getOverlays().add(new MyOverLay(dest,dest, 3));

}

}

catch (MalformedURLException e)

{

e.printStackTrace();

}

catch (IOException e)

{

e.printStackTrace();

}

catch (ParserConfigurationException e)

{

e.printStackTrace();

}

catch (SAXException e)

{

e.printStackTrace();

}

Đưới đây là thông tin các tọa độ của một tuyến đường mà ta phải lọc ra để vẽ lại trên bản đồ:

<GeometryCollection><LineString><coordinates>106.702430,10.771050,0.000000 106.701460,10.773190,0.000000 106.700420,10.775660,0.000000 106.699900,10.776730,0.000000 106.695820,10.780500,0.000000 106.695820,10.780500,0.000000 106.698820,10.783700,0.000000 106.701610,10.786540,0.000000 106.705650,10.790870,0.000000 106.705650,10.790870,0.000000 106.707400,10.792750,0.000000 106.707400,10.792750,0.000000 106.709900,10.795390,0.000000 106.710040,10.795630,0.000000 106.710710,10.797360,0.000000 106.710930,10.798130,0.000000 106.711030,10.798640,0.000000 106.711270,10.801140,0.000000 106.711270,10.801140,0.000000 106.711470,10.801160,0.000000 106.711620,10.801270,0.000000 106.712340,10.801300,0.000000 106.713260,10.801220,0.000000 106.714010,10.801050,0.000000 106.716180,10.800350,0.000000 106.717940,10.799420,0.000000 106.717940,10.799420,0.000000 106.717860,10.799750,0.000000 106.716830,10.800260,0.000000 </coordinates></LineString></GeometryCollection>

**3.4. Một số API phổ biến của google maps trên android**

3.4.1. Lớp GeoPoint

Đây là một lớp không thay đổi tượng trưng cho một cặp kinh độ và vĩ độ, được lưu ở kiểu số nguyên. Để vẽ được các điểm trên bản đồ ta cần phải sử dụng lớp này để chuyển đổi các tọa độ điểm.

* Constructor

public GeoPoint(int latitudeE6, int longitudeE6): Khởi tạo một đối tượng GeoPoint với kinh độ và vĩ độ được cho, đơn vị microdegree (degrees\*1E6)

* Các phương thức

|  |  |
| --- | --- |
| Tên phương thức | Mô tả |
| public int **getLatitudeE6**() | Trả về vĩ độ của đối tượng GeoPoint ở đơn vị microdegrees |
| public int **getLongitudeE6**() | Trả về kinh độ của đối tượng GeoPoint ở đơn vị microdegrees |
| public java.lang.String **toString**() | Overrides phương thức toString()của lớp java.lang.Object |
| public boolean **equals**(java.lang.Object object) | Overrides phương thức equals của lớp java.lang.Object |
| public int **hashCode**() | Override phương thức hashCode() của lớp java.lang.Object |

* Ví dụ:

float lat = -23.4456f;

float lng = 45.44334f;

GeoPoint gp = new GeoPoint((int)(lat \* 1E6), (int)(lng \* 1E6));

Đoạn code trên sẽ chuyển tọa độ lat, lng thành tọa độ đối tượng GeoPoint để có thể vẽ được trên bản đồ.

3.4.2. OverlayItem & ItemizedOverlay

OverlayItems được sử dụng để cung cấp tính năng đánh dấu bằng hình ảnh vào Map Views thông qua lớp ItemizedOverlay.

ItemizedOverlays cung cấp phương thức tiện lợi cho việc thêm cá hình ảnh đánh dấu vào một bản đồ, nó cho phép gán một hình ảnh đánh đấu và thông tin liên quan (kiểu text) vào một vị trí địa lý cụ thể. Thực thể ItemizedOverlay điều khiển công việc vẽ, sắp xếp, sự kiện click, và tối ưu hóa việc sắp xếp mỗi hình OverlayItem.

Để thêm một hình ảnh đánh dấu ItemizedOverlay vào bản đồ ta tạo một lớp mới kế thừa thừa lớp ItemizedOverlay<OverlayItem>.

Cần phải gọi hàm populate để rang buộc việc tạo mỗi OverlayItem; phương thức populate() phải được gọi bất cứ nơi nào mà dữ liệu của OverlayItem thay đổi.

Ví dụ:

import android.graphics.drawable.Drawable;

import com.google.android.maps.GeoPoint;

import com.google.android.maps.ItemizedOverlay;

import com.google.android.maps.OverlayItem;

public class MyItemizedOverlay extends ItemizedOverlay<OverlayItem> {

public MyItemizedOverlay(Drawable defaultMarker) {

**super(boundCenterBottom(defaultMarker));**

**populate();**

}

@Override

protected OverlayItem createItem(int index) {

switch (index) {

case 1:

Double lat = 37.422006\*1E6;

Double lng = -122.084095\*1E6;

GeoPoint point = new GeoPoint(lat.intValue(), lng.intValue());

OverlayItem oi;

oi = new OverlayItem(point, "Marker", "Marker Text");

return oi;

}

return null;

}

@Override

public int size() {

// Return the number of markers in the collection

return 1;

}

}

Để thêm một ItemizedOverlay vào bản đồ, tạo một thực thể mới (truyền hình ảnh đánh đấu Drawable để sử dụng cho mỗi vị trí dánh dấu) và thêm vào danh sách Overlay của bản đồ.

List<Overlay> overlays = mapView.getOverlays();

MyItemizedOverlay markers = new

MyItemizedOverlay(r.getDrawable(R.drawable.marker));

overlays.add(markers);

* Lớp OverlayItem

Các thành phần:

|  |  |
| --- | --- |
| Tên biến | Mô tả |
| Static int [ITEM\_STATE\_FOCUSED\_MASK](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/OverlayItem.html#ITEM_STATE_FOCUSED_MASK) | Chỉ ra trạng thái focus của overlayitem |
| Static int [ITEM\_STATE\_PRESSED\_MASK](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/OverlayItem.html#ITEM_STATE_PRESSED_MASK) | Chỉ ra overlayitem trong trạng thái được nhấn |
| Static int [ITEM\_STATE\_SELECTED\_MASK](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/OverlayItem.html#ITEM_STATE_SELECTED_MASK) | Chỉ ra overlayitem trong trạng thái được chọn |
| protected  android.graphics.drawable.Drawable mMaker |  |
| protected  [GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) mPoint | Vị trí của item |
| protected  java.lang.String mSnippet | Thông tin trích gọn của item |
| protected  java.lang.String mTitle | Tiêu đề item |

Các phương thức:

|  |  |
| --- | --- |
| Tên phương thức | Mô tả |
| public void **setMarker**  (android.graphics.drawable.Drawable marker) | Thiết lập marker được sử dụng khi vẽ một item trên bản đồ. Nếu maker thiết lập null thì maker mặc định sẽ được vẽ. Maker có thể được vẽ bằng các kết hợp giá trị null, các thuộc tính R.attr.state\_pressed, R.attr.state\_seleted và R.attr.state\_focused |
| public android.graphics.drawable.Drawable **getMarker**(int stateBitset) | Trả về maker được sử dụng khi vẽ item này trên bản đồ. Khi giá trị trả về là null điều này có nghĩa là maker mặc định được sử dụng. Các maker khác nhau được trả về tùy thuộc vào các tình trạng khác nhau của maker.  Parameter:  stateBitset – tình trạng hiện tại |
| public static void **setState**  (android.graphics.drawable.Drawable drawable,  int stateBitset) | Thiết lập một drawable tương ứng trạng thái được cho.  Parameter:  Drawable –drawble cần chỉnh sửa  stateBitset – trạng thái để thiết lập |
| public java.lang.String **getTitle**() | Trả về tiêu đề của overlay |
| public java.lang.String **getSnippet**() | Trả về thông tin mô tả của overlay |
| public [GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) **getPoint**() | Trả về đối tượng GeoPoint của overlay |
| public java.lang.String **routableAddress**() | Trả về vị trí của item ở định dạng map-routable |

* Lớp ItemizedOverlay<Item extends OverlayItem>

Là một lớp cơ sở cho một Overlay mà chứa một danh sách OverlayItems. Điều khiển việc sắp xếp bắc-đến-Nam cho việc vẽ trên bản đồ, tạo ra một span, vẽ một marker cho mỗi điểm.

Constructor:

|  |  |
| --- | --- |
|  |  |
| public **ItemizedOverlay**(Drawable   defaultMarker) | Tạo một ItemizedOverlay mới  **Parameters: Tham số**  defaultMarker - Một Drawable được vẽ trên bản đồ cho mỗi item trong overlay |

Các phương thức:

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| protected static Drawable **boundCenterBottom**(Drawable balloon) | Điều chỉnh phạm vi của một drawable theo đó (0,0) là pixel ở vị trí trung tâm của hàng cuối của drawable. |
| protected static Drawable **boundCenter**(Drawable balloon) | Điều chỉnh phạm vi của một drawable theo đó (0,0) là pixel ở vị trí trung tâm của drawable. |
| protected abstract [Item](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/ItemizedOverlay.html) **createItem**(int i) | Phương thức này được định nghĩa bởi lớp con để tạo các item thực sự. Hàm này chỉ được gọi từ populate() |
| public abstract int **size**() | Số lượng item trong overlay. |
| public [GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) **getCenter**() |  |
| protected int **getIndexToDraw**(int drawingOrder) | Trả về bậc của item với index được cho sẵn. Mặc định, các item được xếp loại bởi vĩ độ. Lớp kế thừa có thể override phương thức này để thay đổi thứ tự vẽ ra. |
| public void d**raw**(Canvas canvas,  [MapView](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html) mapView, boolean shadow) | Vẽ một maker trên mỗi item. Phương thức populate() phải được hủy bỏ trước. |
| protected final void **populate**() | Phương thức này dùng thực hiện tất cả xử lý trên một ItemizedOverlay mới. Các lớp kế thừa cung cấp các item thông qua phương thức createItem(int). Lớp kế thừa cần gọi phương thức này khi nó có dữ liệu. |
| public [Item](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/ItemizedOverlay.html) **getFocus**() | Trả về item hiện tại được focus, hay giá trị null nếu không có item nào được focus |
| public final [Item](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/ItemizedOverlay.html) **getItem**(int position) | Trả về item từ index |
| public boolean **onTap**([GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) p,  [MapView](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html) mapView) | Điều khiển một sự kiện tap. |

3.4.3. Lớp MapController

MapController được sử dụng để điều khiển bản đồ, cho phép thiết lập vị trí giữa của bản đồ và các mức phóng to/thu nhỏ. Có một số phương thức được địn nghĩa cho việc phóng to, thu nhỏ thông quan lớp này. Android định nghĩa 21 mức phóng to/thu nhỏ cho bản đồ. Ở mức 1, xích đạo của trái đất dài 256 pixels. Mỗi lần phóng to thì con số này tang gấp 2 lần.

Có thể tham chiếu đến controller của MapView bằng cách sử dụng phương thức getController.

MapController mapController = myMapView.getController();

Các vị trí các điểm bản đồ của các lớp mapping trên Android biểu diễn bằng các đối tượng GeoPoint, chứa kinh độ và vĩ độ được đo ở đơn vị microdegree. Để chuyển độ thành microdegree ta nhân số này với 1E6 (1.000.000).

Trước khi có thể sử dụng tọa độ lưu trong đối tượng Location, cần chuyển tọa độ này thành microdegree và lưu lại thành đối tượng GeoPoint.

Double lat = 37.422006\*1E6;

Double lng = -122.084095\*1E6;

GeoPoint point = new GeoPoint(lat.intValue(), lng.intValue());

Để định vị vị trí giữa và điều chỉnh kích thướt bản đồ chúng ta sử dụng phương thức setCenter và setZoom có trong đối tượng MapController của MapView.

mapController.setCenter(point);

mapController.setZoom(1);

Phương thức setCenter sẽ nhảy đến một vị trí mới. Để chuyển động này một cách mượt mà chung ta sử dụng thêm phương thức animateTo().

mapController.animateTo(point);

Dưới đây là đoạn code mô tả cách sử dụng lớp này MapController

1. Đầu tiên chúng ta thiết kế giao diện cho ứng dụng như sau:

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

<LinearLayoutxmlns: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/myLocationText"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="@string/hello"

/>

<com.google.android.maps.MapView

android:id="@+id/myMapView"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:enabled="true"

android:clickable="true"

android:apiKey="myMapKey"

/>

</LinearLayout>

1. Cấu hình MapView và lưu một tham chiếu đến MapController của nó.

**MapController mapController;**

@Override

public void onCreate(Bundle savedInstance

State) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

// Get a reference to the MapView

MapView myMapView = (MapView)findViewById(R.id.myMapView);

**// Get the Map View’s controller**

**mapController = myMapView.getController();**

// Configure the map display options

myMapView.setSatellite(true);

myMapView.setStreetView(true);

myMapView.displayZoomControls(false);

// Zoom in

**mapController.setZoom(17);**

}

Các phương thức trong lớp MapController

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| public void **stopPanning**() | Đặt lại tình trạng của pan để bản đồ đứng yên một chỗ. Điều này cần thiết nếu chúng ta nhận một sự kiện keydown nhưng không nhận sự kiện key-up tương ứng |
| public boolean **onKey**(android.view.View v,  int keyCode,  android.view.KeyEvent event) | Xử lý các sự kiện nhấn phím và chuyển các sự kiện này thành “pan” phù hợp với map. Được định nghĩa trong lớ[ View.OnKeyListener. |
| public void **animateTo**([GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) point) | Chuyển bản đồ về phía điểm được chọn |
| public void **animateTo**([GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) point,android.os.Message message) | Chuyển bản đồ về điểm được chọn. Nếu và khi sự chuyển động đến điểm cuối cùng, một thông điệp sẽ được gửi đi (nếu không phải null). |
| public void **animateTo**([GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) point,  java.lang.Runnable runnable) |  |
| public void **scrollBy**(int x,  int y) | Cuộn bản đồ theo lượng pixel được cho sẵn.  Parameters  x- cuộng theo chiều ngang  y – cuộn theo chiều dọc |
| public void **setCenter**([GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) point) | Đặt góc nhìn bản đồ đến điểm được cho. Sẽ không có bất kỳ chuyển động nào được thực hiện |
| public void **stopAnimation**(boolean jumpToFinish) | Dừng bất kỳ một chuyển động nào đang được thực hiện. |
| public int **setZoom**(int zoomLevel) | Sets the zoomlevel of the map. The value will be clamped to be between 1 and 21 inclusive, though not all areas have tiles at higher zoom levels. This just sets the level of the zoom directly; for a step-by-step zoom with fancy interstitial animations, use [zoomIn()](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapController.html#zoomIn()) or [zoomOut()](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapController.html#zoomOut()).  Thiết lập mức độ phóng to thu nhỏ của bản đồ. Tất cả giá trị sẽ được nằm giữa 1 và 21, mặc dầu không phải tất cả khu vực đều có các khối vật ở mức cao hơn các mức zoom. Hàm này chỉ thiết lập mức zoom một cách trực tiếp.  Sử dụng zoomIn() hay ZoomOut() để thay đổi mức zoom của bản đồ. |
| public void **zoomToSpan**(int latSpanE6,  int lonSpanE6) | Điều chỉnh độ phóng to thu nhỏ của bản đồ vì vậy khoảng cách được trao của kinh độ và vĩ độ sẽ được hiển thị. Bời vì zoom chỉ có thể lấy về các mức riên lẻ, và bởi vì tỉ lệ kích cỡ màn hình của bản đồ có thể không khớp với tỉ lệ được cho, nên chất lượng khớp sẽ có thể biến đổi. |
| public boolean **zoomIn**() | Phóng to một đơn vị. |
| public boolean **zoomOut**() | Thu nhó đơn vị |
| public boolean **zoomInFixing**(int xPixel,  int yPixel) | Zoom in by one zoom level. This begins an animated zoom step. If you have added the ZoomControls View (from [MapView.getZoomControls()](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html#getZoomControls())), then this will also call through to [MapView.displayZoomControls(boolean)](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html#displayZoomControls(boolean)) causing the ZoomControls to appear briefly, but they will not take focus.  This zoom will also pan the map to keep a single point fixed on the screen. Specify the point you want fixed by giving its pixel coordinates.  **Parameters:**  xPixel - offset, in pixels from the left of the map, where the fixed point of our zoom will be.  yPixel - offset, in pixels from the top of the map, where the fixed point of our zoom will be.  **Returns:**  true if we managed to zoom in, false if we hit the limit. |
| public boolean **zoomOutFixing**(int xPixel,  int yPixel) | Zoom out by one zoom level. This begins an animated zoom step. If you have added the ZoomControls View (from [MapView.getZoomControls()](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html#getZoomControls())), then this will also call through to [MapView.displayZoomControls(boolean)](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html#displayZoomControls(boolean)) causing the ZoomControls to appear briefly, but they will not take focus.  This zoom will also pan the map to keep a single point fixed on the screen. Specify the point you want fixed by giving its pixel coordinates.  **Parameters:**  xPixel - offset, in pixels from the left of the map, where the fixed point of our zoom will be.  yPixel - offset, in pixels from thetop of the map, where the fixed point of our zoom will be.  **Returns:**  true if we managed to zoom out, false if we hit the limit. |

3.4.4. MapView & MapActivity

Nhiều công nghệ bản đồ trong Android dựa trên control MapView và phần mở rộng android.app.Activity được gọi là MapActivity. Lớp MapView và MapActivity là thành phần cốt lõi của bản đồ trên Android nó cho phép hiển thị và thao tác một bản đồ trên Android. Một trong những điều phải nhớ về 2 lớp này là chúng làm việc cùng nhau. Đặc biệt để sử dụng MapView, bạn cần khai báo bên trong một MapActivity. Thêm vào đó để sử dụng được MapView bạn cần cung cấp api-key, việc lấy api-key đã được đề cập ở trên. Ngoài ra ứng dụng của bạn cần truy cập được internet.



Hình trên cho thấy một ứng dụng hiển thị một bản đồ ở chế độ street-view. Ứng dụng cũng cho thấy rõ phóng to, thu nhỏ và thay đổi chế độ nhìn của bản đồ như thế nào. Layout của ứng dụng được thiết kế như bên dưới:

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical" android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent">

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="horizontal" android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content">

<Button android:id="@+id/zoomin" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:text="+"/>

<Button android:id="@+id/zoomout" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:text="-"/>

<Button android:id="@+id/sat" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:text="Satellite"/>

<Button android:id="@+id/street" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:text="Street"/>

<Button android:id="@+id/traffic" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:text="Traffic"/>

</LinearLayout>

<com.google.android.maps.MapView android:id="@+id/mapview"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:apiKey="07vhL0usFXryRakmo2A4t8aKViWwKyGJGEDqpdg"

/>

</LinearLayout>

Dưới đây là một đoạn code mẫu cho ứng dụng bản đồ trên (Listing 7-13)

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import com.google.android.maps.MapActivity;

import com.google.android.maps.MapView;

public class MapViewDemoActivity extends MapActivity

{

private MapView mapView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.mapview);

mapView = (MapView)findViewById(R.id.mapview);

Button zoominBtn = (Button)findViewById(R.id.zoomin);

Button zoomoutBtn = (Button)findViewById(R.id.zoomout);

Button satBtn = (Button)findViewById(R.id.sat);

Button streetBtn = (Button)findViewById(R.id.street);

Button trafficBtn = (Button)findViewById(R.id.traffic);

// zoomin

zoominBtn.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View view)

{

mapView.getController().zoomIn();

}});

// zoom out

zoomoutBtn.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View view)

{

mapView.getController().zoomOut();

}});

// satellite

satBtn.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View view)

{

mapView.setStreetView(false);

mapView.setTraffic(false);

mapView.setSatellite(true);

}});

// street

streetBtn.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View view)

{

mapView.setTraffic(false);

mapView.setSatellite(false);

mapView.setStreetView(true);

}});

// traffic

trafficBtn.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View view)

{

mapView.setSatellite(false);

mapView.setStreetView(false);

mapView.setTraffic(true);

}});

}

@Override

protected boolean isRouteDisplayed() {

return false;

}

}

Như hiển thị trong Listing 7-13, hiển thị MapView cũng sử dụng onCreate() không khác so với hiển thị các control khác. Để phóng to, thu nhỏ bạn sử dụng lớp MapController của MapView. Để làm được điều này bằng cách mapView.getController và sau đó gọi phương thức thích hợp zoomIn() hay zoomOut().

MapView hỗ trợ một số chế độ hiển thị: map, street, satellite và traffic. Map là chế độc mặc định. Chết độ hiển thị Street chứa thông tin đường đi như tên đường. Chế độ Satelite hiển thị bản đồ ở chế độ vệ tinh. Chế độ Traffic hiển thị thông tin giao thông trên bản đồ. Lưu ý rằng chế độ Traffic chỉ được hỗ trợ trên một số tuyến đường cao tốc chính. Để thay đổi chế độ hiển thị, bạn phải gọi các phương thức thích hợp được thiết lập là true và các chế độ khác là false.

* Lớp MapView

Một View hiển thị một bản đồ (dữ liệu được lấy về từ dịch vụ Google Maps). Khi được focus, nó sẽ bắt lấy các hành động nhấn phím và chạm vào màn hình để dịch chuyển và thay đổi kích thướt bản đồ. Nó có thể được điều khiển một cách theo cách lập trình (getController()) và có thể vẽ một các Overlay trên bản đồ (getOverlays()).

MapView cũng là một ViewGroup với LayoutParameters cho phép gán các chế độ hiển thị, tại các offset pixel xác định hay các tọa độ xác định.

Một MapView có thể được khởi tạo bằng một MapActivity do nó phụ thuộc vào các tiến trình truy cập vào mạng và các file hệ thống bên dưới; những tiến trình này phải tuân theo lifecycle trong MapActivity. Dưới đây là thông tin chi tiết về lớp MapView:

Constructor

|  |
| --- |
| **MapView**(android.content.Context context,android.util.AttributeSet attrs)           Khởi tạo một đối tượng MapView |
| **MapView**(android.content.Context context, android.util.AttributeSet attrs, int defStyle) |
| **MapView**(android.content.Context context, java.lang.String apiKey) |

Phương thức:

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| protected void **onSizeChanged**(int w, int h, int oldw,int oldh) | Điều chỉnh lại kích thướt của đối tượng map và chuyển lại vị trí các đường kẻ |
| public void **computeScroll**() | Bắt lấy các sự kiện cuộn màn hình và điều chỉnh lại camera bản đồ |
| protected final void **onDraw**(android.graphics.Canvas canvas) | Đây là phương thức vẽ chính. Vẽ nền của bản đồ, các vật phủ trên bản đồ (overlay), các đường lưới và logo của Google |
| public boolean **onKeyDown**(int keyCode,  android.view.KeyEvent event) | Truyến các phím nhấn đến các đối tượng overlay trước. Nếu các overlay này không xử lý thì sẽ được chuyển qua sự kiện dịch chuyển bản đồ |
| public boolean **onKeyUp**(int keyCode,  android.view.KeyEvent event) | Giống như onKetDown |
| public void **displayZoomControls**(boolean takeFocus) | Hiển thị các control phóng to/thu nhỏ |
| public int **getZoomLevel**() | Trả về mức phóng to/thu nhỏ của bản đồ |
| public void **setSatellite**(boolean on) | Thiết lập bản đồ ờ chế độ vệ tinh |
| public boolean **isSatellite**() | Kiểm tra xem bản đồ có ở chế độ hiển thị vệ tinh hay không |
| public void **setStreetView**(boolean on) |  |
| public boolean **isStreetView**() |  |
| public GeoPoint **getMapCenter**() | Trả về vị trí trung tâm hiện tại của bản đồ |
| public MapController **getController**() | Trả về đối tượng MapController cho bản đồ, có thể sử dụng đối tượng này để điều khiển bản đồ |
| public final java.util.List<Overlay> **getOverlays**() | Truy cập danh sách các overlay |
| public int **getLatitudeSpan**() | Lấy vĩ độ hiện tại của span (từ cạnh trên cùng đến cạnh cuối cùng của bản đồ) ở độ thập phân được nhân lên một triệu lần |
| public int **getLongitudeSpan**() | Lấy kinh độ hiện tại của span (từ cạnh trên cùng đến cạnh cuối cùng của bản đồ) ở độ thập phân được nhân lên một triệu lần |
| public void **setReticleDrawMode**  (MapView.ReticleDrawMode mode) | Chỉ rõ các đường kẻ được vẽ như thế nào khi thiết bị không ở chế độ cảm ứng.  Tham số:  Mode – chế độ lưới được vẽ. Các giá trị hợp lệ: MapView.ReticleDrawMode.DRAW\_RETICLE\_OVER – đây là chế độ mặc định và ở chế độ này các đường lưới sẽ được vẽ lên các overlay vì vậy nó sẽ luôn thấy được. [MapView.ReticleDrawMode.DRAW\_RETICLE\_UNDER](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.ReticleDrawMode.html#DRAW_RETICLE_UNDER) các đường lưới sẽ được vẽ trước khi tất cả các overlay được vẽ. [MapView.ReticleDrawMode.DRAW\_RETICLE\_NEVER](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.ReticleDrawMode.html#DRAW_RETICLE_NEVER) – các đường lưới không được vẽ, |
| public int **getMaxZoomLevel**() | Trả về chế độ zoom tối đa |
| public void **onSaveInstanceState**(android.os.Bundle state) | Lưu tình trạng hiện tại của MapView vào một Bundle bao gồm mức độ zoom, và tình trạng của hộp thoại zoom |
| public void **onRestoreInstanceState**  (android.os.Bundle state) | Phục hồi lại tình trạng của MapView vào một Bundle |
| public android.view.View **getZoomControls**() | Trả về một View chứa một widget ZoomControls (ví dụ nút +/- ). |
| public void **setBuiltInZoomControls**(boolean on) | Bật các controls phóng to/thu nhỏ được dụng sẵn. Nếu được bật, MapView sẽ hiển thị các controls zoom một các tự động. |
| public Projection **getProjection**() | Lấy về một phép chiếu cho việc chuyển tọa độ pixel và tọa độ theo kinh vĩ độ. |

* Lớp MapActivity

MapActivity là lớp cơ sở mà bạn kế thừa để tạo các Activity mới có thể bao gồm một MapView. Lớp MapActivity điều khiển vòng đời ứng dụng và quản lý các dịch vụ bên dưới cần cho việc hiển thị bản đồ.

Một lớp kế thừa phải tạo một MapView riêng của mình trong hàm onCreate(android.os.Bundle), hoặc với constructor MapView (sau đó thêm nó vào một layout View với ViewGroup.addView(View)) hay khởi tạo từ layout XML.

Đây không phải là gói chuẩn trong thư viện Android. Để sử dụng nó, cần phải thêm một tag sau trong file XML trong file AndroidManifest.xml:

<uses-library android:name="com.google.android.maps" />

Các phương thức

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| protected void **onCreate**(android.os.Bundle icicle) | Trong phương thức này chúng ta khởi tạo các thành phần tỉnh, tạo đối tượng map và dịch vụ traffic, nhưng không nên bắt đầu bất kỳ tiến trình nào tại đây. Lớp thừa kế nên tạo một đối tượng MapView sau khi gọi đến super.onCreate() |
| public void **onNewIntent**(android.content.Intent newIntent) |  |
| protected void **onResume**() | Hàm này cho biết map được vẽ lại, |
| protected void **onPause**() | Tạm dừng các tiến trình như dịch vụ traffic bởi vì chúng ta sẽ không muốn chúng chạy mãi mãi bên dưới. |
| protected void **onDestroy**() | Dừng tất cả tiến trình và giải phóng tất cả tài nguyên. |
| protected abstract boolean **isRouteDisplayed**() | Hàm này sử dụng cho mục đích tính toán, máy chủ cần biết liệu bạn có đang hiển thị bất kỳ thông tin đường đi nào không, như hướng dẫn lái xe. Lớp kế thừa phải hiện thực phương thức này để báo cáo thông tin hay sẽ bị vi phạm điều kiện sử dụng. |
| protected boolean **isLocationDisplayed**() | Phương thức này được sử dụng cho mục đích tính toán, máy chủ cần biết liệu là bạn có đang sử dụn thiết bị định vị nào không. Mặc định phương thức này trả về true nếu một lớp kế thừa của MyLocationOverlay được thêm vào thông qua MapView.getOverlays() và sự định vị được bật. |

3.4.5. Lớp MyLocationOverlay

Lớp MyLocationOverlay là một Overlay đặc biệt được thiết kế để hiển thị vị trí hiện tại và hướng trên bản đồ.

Để sử dụng được lớp này chúng ta cần tạo một thực thể mới, truyền vào trong ứng dụng Context, MapView đích và thêm thực thể này vào danh sách Overlay của MapView như đoạn code dưới đây:

List<Overlay> overlays =

mapView.getOverlays();

MyLocationOverlay myLocationOverlay =

new MyLocationOverlay(this, mapView);

overlays.add(myLocationOverlay);

Chúng ta có thể sử dụng MyLocationOverlay để hiển thị cả vị trí hiện tại và hướng đi hiện tại của chúng ta.

Đoạn code mẫu dưới đây chỉ cách bật tính năng la bàn và đánh đấu, trong thực thể này MapController của MapView cũng được truyền vào, cho phép Overlay tự động cuộn bản đồ nếu marker duy chuyển ra màn hình.

myLocationOverlay.enableCompass();

myLocationOverlay.enableMyLocation(mapView.getMapController());

|  |  |
| --- | --- |
| Tên phương thức | Mô tả |
| public **MyLocationOverlay**  (Context context,[MapView](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/MapView.html) mapView) | Khởi dụng một MyLocationOverlay mới |
| public boolean **enableCompass**() | Bật chế độ cập nhật từ thiết bị cảm ứng la bàn. |
| public void **disableCompass**() | Bật chế độ cập nhật từ thiết bị cảm ứng la bàn. |
| public boolean **isCompassEnabled**() | Kiểm tra la bàn có được hiển thị hay không |
| public void **disableMyLocation**() | Dừng cập nhật vị trí |
| public void **onSensorChanged**  (int sensor,float[] values) | Được gọi bởi SensorManager khi giá trị la bàn thay đồi |
| public void **onLocationChanged**(Location location) | Bộ lắng nghe để lấy vị trí cập nhật từ LocationManagerService |
| protected void **drawCompass**(Canvas canvas,  float bearing) | Vẽ la bàn. Mặc định, vẽ một la bàn và một mũi tên ở góc trên bên trái |
| public [GeoPoint](http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/GeoPoint.html) **getMyLocation**() | Trả về một GeoPoint tương ứng với vị trí hiện tại của người dung. |
|  |  |

**Phần 4:Xây dựng phần mềm hỗ trợ tìm kiếm và dẫn đường MapPro**

4.1. Giới thiệu phần mềm MapPro

4.2. Cấu trúc phần mềm

4.3. Kiến trúc hệ thống

4.4. Sơ đồ Use-Case

4.4.1. Sơ đồ tổng quát

* Sơ đồ
* Danh sách Actor
* Danh sách use-case
* Đặc tả use-case

4.4.2. Sơ đồ phân rã

4.5. Thiết kế giao diện

4.5.1. Danh sách màn hình

4.5.2. Mô tả chi tiết từng màn hình

Chương 5: Đánh giá kết quả và kết luận

5.1. Đánh giá luận văn

5.2. Đánh giá ứng dụng

5.2.1. Kết quả đạt được

5.2.2. Hạn chế

5.3. Hướng phát triển

5.4 Kết luận