BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Nguyễn Văn Cảnh

Giảng viên hướng dẫn: ThS. Phùng Anh Tuấn

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

XÂY DỰNG ỨNG DỤNG ANDROID LẤY DỮ LIỆU MỚI TRÊN HOSTING THEO THỜI GIAN TRỰC

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Nguyễn Văn Cảnh

Giảng viên hướng dẫn: ThS. Phùng Anh Tuấn

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: Nguyễn Văn Cảnh Mã SV: 1412101006

Lớp: CT1802 Ngành: Công nghệ thông tin

Tên đề tài: Xây dựng ứng dụng Android lấy dữ liệu mới trên Hosting theo thời gian trực

MỤC LỤC

LỜI CẢM ƠN

Đề tài "Xây dựng ứng dụng Android lấy dữ liệu trên hosting theo thời gian thực" là nội dung Em chọn để nghiên cứu và làm đồ án tốt nghiệp sau bốn năm học chương trình đại học ngành công nghệ thông tin tại trường Đại Học Dân Lập Hải Phòng.

Để hoàn thành quá trình nghiên cứu và hoàn thiện đồ án tốt nghiệp này, lời đầu tiên Em xin gửi lời cảm ơn chân thành cảm ơn tới toàn thể quý Thầy Cô, bạn bè của Trường Đại Học Dân Lập Hải Phòng.

Bày tỏ lòng biết ơn sâu sắc nhất thầy cô trong khoa công nghệ thông tin đã dìu dắt, chia sẻ những kiên thức quý báu trong suốt quá trình học tập tại trường. Đặc biệt là thầy ThS.Phùng Anh Tuấn cùng với tri thức và tâm huyết của Thầy đã tạo điều kiện em hoàn thành đồ án tốt nghiệp tại trường. Nếu không có Thầy đồ án tốt nghiệp của Em khó có thể hoàn thành được.

Cuối cùng, Em xin cảm ơn những người thân, bạn bè đã luôn bên Em, động viên, sẻ chia, giúp đỡ, cổ vũ tinh thần,…Đó là nguồn động lực giúp Em hoàn thành chương trình học và đồ án tốt nghiệp này.

Hải Phòng, ngày 03 tháng 08 năm 2018

Sinh viên

Nguyễn Văn Cảnh

LỜI MỞ ĐẦU

Hiện nay Công nghệ thông tin vô cùng phát triển thì mọi người đều sử dụng máy vi tính hoặc điện thoại di động để làm việc và giải trí. Do đó việc xây dựng các ứng dụng cho điện thoại di động đang là một ngành công nghiệp mới đầy tiềm năng và hứa hẹn nhiều sự phát triển vượt bậc của ngành khoa học kỹ thuật.

Phần mềm, ứng dụng cho điện thoại di động hiện nay rất đa dạng và phong phú trên các hệ điều hành di động. Các hệ điều hành J2ME, Adroid, IOS, Hybrid, Web bases Mobile Application đã rất phát triển trên thị trường truyền thông di động.

Trong vài năm trở lại đây, hệ điều hành Adroid ra đời với sự kế thừa những ưu việt của các hệ điều hành ra đời trước và sự kết hợp của nhiều công nghệ tiên tiến nhất hiện nay. Adroid đã nhanh chóng là đối thủ cạnh tranh mạnh mẽ với các hệ điều hành trước đó, hứa hẹn đây sẽ là hệ điều hành di động của tương lai và được rất nhiều người ưa thích chọn nó để sử dụng.

Cùng với sự phát triển nhanh chóng của xã hội, nhu cầu giải trí thông qua điện thoại di động ngày càng phổ biến. Vì vậy, Em đã chọn đề tài **"Xây dựng ứng dụng Android lấy dữ liệu mới trên hosting theo thời gian thực"** với mục đích nghiên cứu, tìm hiểu về ứng đụng và cũng để cập nhật thông tin mới nhất khi đó thể hiện thông báo mới khi có thông tin mới giúp người dùng có được những tin tức mới nhất.

CHƯƠNG 1: TỔNG QUAN HỆ ĐIỀU HÀNH ANDROID

1.1. Lịch sử hệ điều hành Android

Ban đầu, Android là hệ điều hành cho các thiết bị cầm tay dựa trên lõi Linux do công ty Android Inc.(California, Mỹ) thiết kế. Công ty này sau đó được Google mua lại vào năm 2005 và bắt đầu xây dựng Android Platform. Các thành viên chủ chốt tại Android Inc. gồm có: Andy Rubin, Rich Miner, Nick Sears, and Chris White.

Và sau tiếp, vào cuối năm 2007, thuộc về liên minh thiết bị cầm tay mã nguồn mở(Open Handset Alliance) gồm các thành viện nổi bật trong ngành viễn thông và thiết bị cầm tay như:

TexasInstruments, BroadcomCorporation, Google, HTC, Intel, LG, Marvell, Techn ologyGroup, Motorola, Nvidia, Qualcomm, SamsungElectronics, Sprint Nextel, TMobile, ARM Holdings, Atheros Communications, Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, ToshibaCorp, and Vodafone Group.

Mục tiêu của liên minh này là nhanh chóng đổi mới để đáp ứng tốt hơn cho nhu cầu người tiêu dùng và kết quả đầu tiên của nó chính là nền tảng Android. Android được thiết kế để phục vụ nhu cầu của các nhà sản xuất thiết, các nhà khai thác và các lập trình viên thiết bị cầm tay.

Phiên bản SDK lần đầu tiên phát hành vào tháng 11 năm 2007, hãng T-Mobile cũng công bố chiếc điện thoại Android đầu tiên đó là chiếc T-Mobile G1, chiếc smartphone đầu tiên dựa trên nền tảng Android. Một vài ngày sau đó, Google lại tiếp tục công bố sự ra mắt phiên bản Android SDK release Candidate 1.0.

Trong tháng 10 năm 2008, Google được cấp giấy phép mã nguồn mở cho Android Platform.

Khi Android được phát hành thì một trong số các mục tiêu trong kiến trúc của nó là cho phép các ứng dụng có thể tương tác được với nhau và có thể sử dụng lại các thành phần từ những ứng dụng khác.Việc tái sử dụng không chỉ được áp dụng cho cho các dịch vụ mà nó còn được áp dụng cho cả các thành phần dữ liệu và giao diện người dùng.

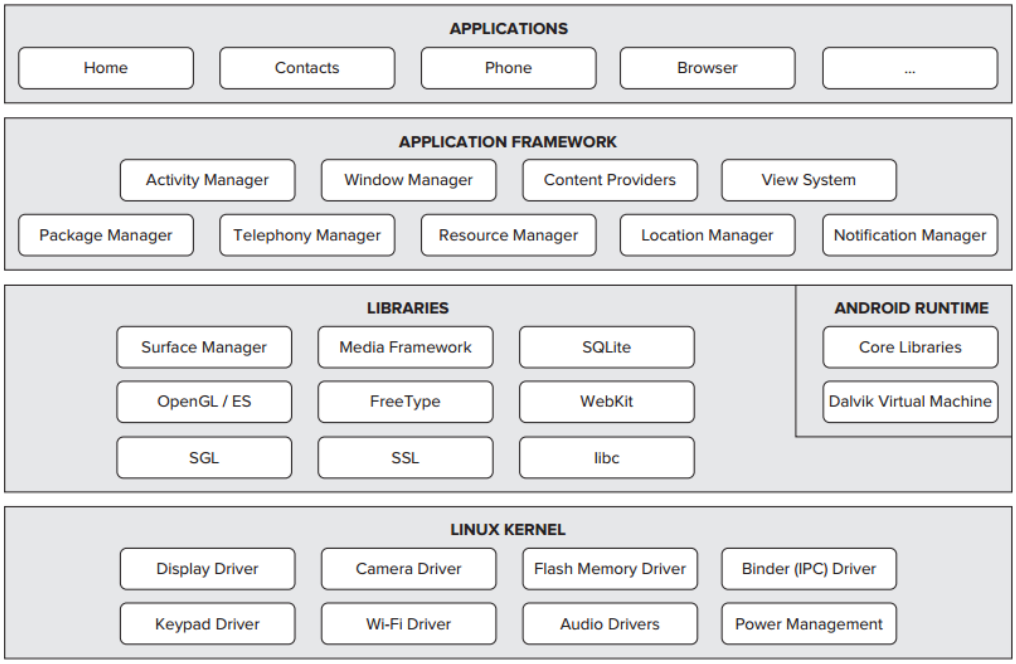
Vào cuối năm 2008, Google cho phát hành một thiết bị cầm tay được gọi là Android Dev Phone 1 có thể chạy được các ứng dụng Android mà không bị ràng buộc vào các nhà cung cấp mạng điện thoại di động.

Mục tiêu của thiết bị này là cho phép các nhà phát triển thực hiện các cuộc thí nghiệm trên một thiết bị thực có thể chạy hệ điều hành Android mà không phải ký một bản hợp đồng nào.Vào khoảng cùng thời gian đó thì Google cũng cho phát hành một phiên bản vá lỗi 1.1 của hệ điều hành này.

Ở cả hai phiên bản 1.0 và 1.1 Android chưa hỗ trợ soft-keyboard mà đòi hỏi các thiết bị phải sử dụng bàn phím vật lý. Android cố định vấn đề này bằng cách phát hành SDK 1.5 vào tháng tư năm 2009, cùng với một số tính năng khác. Chẳng hạn như nâng cao khả năng ghi âm truyền thông, vật dụng, và các live folder.

1.2. Kiến trúc của hệ điều hành Android

– Android gồm 5 phần chính sau được chứa trong 4 lớp:



Hình 1.2.1 Kiến trúc hệ điều hành Android

1.2.1. Nhân Linux

Android dựa trên Linux phiên bản 2.6 cho hệ thống dịch vụ cốt lõi như security[1], memory management, process management, network stack, and driver model. Kernel

[1] Bảo mật

Linux hoạt động như một lớp trừu tượng hóa giữa phần cứng và phần còn lại của phần mềm stack.

1.2.2. Thư viện

Android bao gồm một tập hợp các thư viện C/C++ được sử dụng bởi nhiều thành phần khác nhau trong hệ thống Android. Điều này được thể hiện thông qua nền tảng ứng dụng Android.

Một số các thư viện cơ bản được liệt kê dưới đây:

- Hệ thống thư viện C: một BSD có nguồn gốc từ hệ thống thư viện tiêu chuẩn C (libc), điều chỉnh để nhúng vào các thiết bị dựa trên Linux.
- Thư viện Media – dựa trên PacketVideo's OpenCORE; các thư viện hỗ trợ phát lại và ghi âm của âm thanh phổ biến và các định dạng video, cũng như các tập tin hình ảnh tĩnh,bao gồm cả MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG.
- Bề mặt quản lý – Quản lý việc truy xuất vào hệ thống hiển thị.
- LibWebCore - một công cụ trình duyệt web hiện đại mà quyền hạn cả hai trình duyệt web Android và xem web nhúng.
- SGL - Đồ họa 2D cơ bản của máy.
- Thư viện 3D – một thực hiện dựa vào OpenGL ES 1.0 APIs; các thư viện sử dụng phần cứng tăng tốc 3D (nếu có), tối ọu hóa cao rasterizer phần mềm 3D.
- FreeType- vẽ phông chữ bitmap và vector.

SQLite một công cụ cơ sở dữ liệu quan hệ mạnh mẽ và nhẹ có sẵn cho tất cả các ứng dụng.

1.2.3. Thực thi

Android bao gồm một tập hợp các thư viện cơ bản mà cung cấp hầu hết các chức năng có sẵn trong các thư viện lõi của ngôn ngữ lập trình Java. Tất cả các ứng dụng Android đều chạy trong tiến trình riêng. Máy ảo Dalvik đã được viết để cho một thiết bị có thể chạy nhiều máy ảo hiệu quả. Các VM Dalvik thực thi các tập tin thực thi Dalvik (dex). Định dạng được tối ưu hóa cho bộ nhớ tối thiểu.VM là dựa trên register-based, và chạy các lớp đã được biên dịch bởi một trình biên dịch Java để chuyển đổi thành các định dạng dex. Các VM Dalvik dựa vào nhân Linux cho các chức năng cơ bản như luồng và quản lý bộ nhớ thấp.

1.2.4. Nền tảng Android

Bằng cách cung cấp một nền tảng phát triển mở, Android cung cấp cho các nhà phát triển khả năng xây dựng các ứng dụng cực kỳ phong phú và sáng tạo. Nhà phát triển được tự do tận dụng các thiết bị phần cứng, thông tin địa điểm truy cập, các dịch vụ chạy nền, thiết lập hệ thống báo động, thêm các thông báo để các thanh trạng thái, và nhiều, nhiều hơn nữa. Nhà phát triển có thể truy cập vào các API cùng một khuôn khổ được sử dụng bởi các ứng dụng lõi. Các kiến trúc ứng dụng được thiết kế để đơn giản hóa việc sử dụng lại các thành phần; bất kỳ ứng dụng có thể xuất bản khả năng của và ứng dụng nào khác sau đó có thể sử dụng những khả năng (có thể hạn chế bảo mật được thực thi bởi khuôn khổ). Cơ chế này cho phép các thành phần tương tự sẽ được thay thế bởi người sử dụng.

– Cơ bản tất cả các ứng dụng là một bộ các dịch vụ và các hệ thống, bao gồm:

- Một tập hợp rất nhiều các View có khả năng kế thừa lẫn nhau dùng để thiết kế phần giao diện ứng dụng như: gridview, tableview, linearlayout,… .
- Một “Content Provider” cho phép các ứng dụng có thể truy xuất dữ liệu từ các ứng dụng khác (chẳng hạn như Contacts) hoặc là chia sẻ dữ liệu giữa các ứng dụng đó.
- Một “Resource Manager” cung cấp truy xuất tới các tài nguyên không phải là mã nguồn, chẳng hạn như: localized strings, graphics, and layout files.
- Một “Notifycation Manager” cho phép tất cả các ứng dụng hiển thị các custom alerts trong status bar. Activity Maanager được dùng để quản lý chu trình sống của ứng dụng và điều hướng các activity.

1.2.5. Tầng ứng dụng

Tầng ứng dụng (Application) là tầng giao tiếp với người dùng với các thiết bị Android như Danh bạ, tin nhắn, trò chơi, tiện ích tính toán, trình duyệt… Mọi ứng dụng viết đều nằm trên tầng này.

1.3. Giao diện hệ điều hành Android

Giao diện người dùng của Android dựa trên nguyên tắc tác động trực tiếp, sử dụng cảm ứng chạm tương tự như những động tác ngoài đời thực như vuốt, chạm, kéo giãn và thu lại để xử lý các đối tượng trên màn hình. Sự phản ứng với tác động của người dùng diễn ra gần như ngay lập tức, nhằm tạo ra giao diện cảm ứng mượt mà,

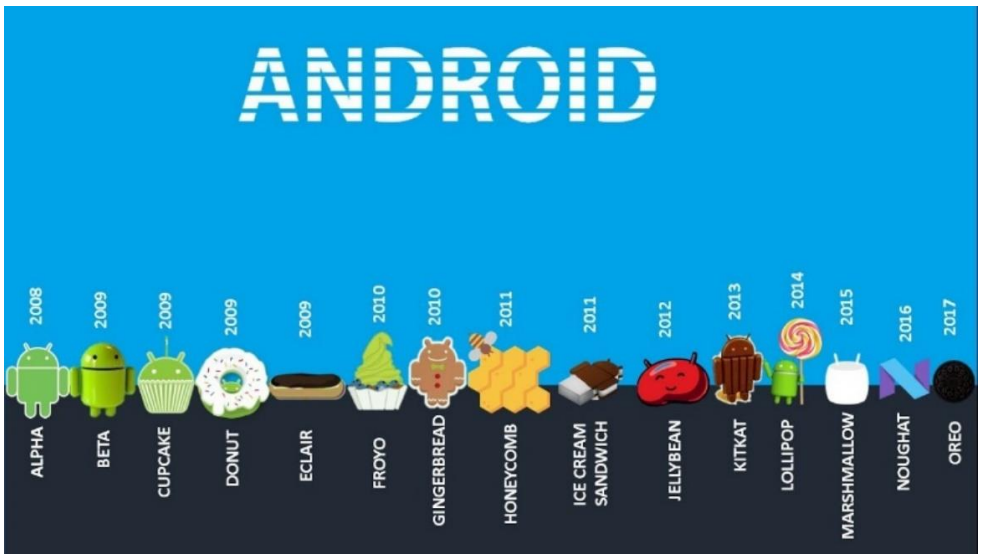
thường dùng tính năng rung của thiết bị để tạo phản hồi rung cho người dùng. Những thiết bị phần cứng bên trong như gia tốc kế, con quay hồi chuyển và cảm biến khoảng cách được một số ứng dụng sử dụng để phản hồi một số hành động khác của người dùng, ví dụ như điều chỉnh màn hình từ chế độ hiển thị dọc sang chế độ hiển thị ngang tùy theo vị trí của thiết bị, hoặc cho phép người dùng lái xe đua bằng xoay thiết bị, giống như đang điều khiển vô-lăng.

Các thiết bị Android sau khi khởi động sẽ hiển thị màn hình chính, điểm khởi đầu với các thông tin chính trên thiết bị, tương tự như khái niệm desktop (bàn làm việc) trên máy tính để bàn. Màn hình chính Android thường gồm nhiều biểu tượng (*icon*) và tiện ích (*widget*); biểu tượng ứng dụng sẽ mở ứng dụng tương ứng, còn tiện ích hiển thị những nội dung sống động, cập nhật tự động như dự báo thời tiết, hộp thư của người dùng, hoặc những mẩu tin thời sự ngay trên màn hình chính. Màn hình chính có thể gồm nhiều trang xem được bằng cách vuốt ra trước hoặc sau, mặc dù giao diện màn hình chính của Android có thể tùy chỉnh ở mức cao, cho phép người dùng tự do sắp đặt hình dáng cũng như hành vi của thiết bị theo sở thích. Những ứng dụng do các hãng thứ ba có trên Google Play và các kho ứng dụng khác còn cho phép người dùng thay đổi "chủ đề" của màn hình chính, thậm chí bắt chước hình dáng của hệ điều hành khác như Windows Phone chẳng hạn. Phần lớn những nhà sản xuất, và một số nhà mạng, thực hiện thay đổi hình dáng và hành vi của các thiết bị Android của họ để phân biệt với các hãng cạnh tranh.

Ở phía trên cùng màn hình là thanh trạng thái, hiển thị thông tin về thiết bị và tình trạng kết nối. Thanh trạng thái này có thể "kéo" xuống để xem màn hình thông báo gồm thông tin quan trọng hoặc cập nhật của các ứng dụng, như email hay tin nhắn SMS mới nhận, mà không làm gián đoạn hoặc khiến người dùng cảm thấy bất tiện. Trong các phiên bản đời đầu, người dùng có thể nhấn vào thông báo để mở ra ứng dụng tương ứng, về sau này các thông tin cập nhật được bổ sung thêm tính năng, như có khả năng lập tức gọi ngược lại khi có cuộc gọi nhỡ mà không cần phải mở ứng dụng gọi điện ra. Thông báo sẽ luôn nằm đó cho đến khi người dùng đã đọc hoặc xóa nó đi.

1.4. Quá trình phát triển phiên bản Android

Quá trình phát triển các phiên bản của hệ điều hành Android:



Hình 1.4.1 Các phiên bản Android đến nay

1.4.1. Phiên bản Android 1.5

Android Cupcake phiên bản Android thứ hai được phát triển bởi Google, là một phiên bản quan trọng được phát hành tới các thiết bị chạy Android vào tháng 5 năm 2009. Phiên bản này bao gồm các tính năng cho người dùng và nhà phát triển, cũng như các thay đổi trên API của nền tảng Android. Với lập trình viện, nền tảng Android 1.5 có thể được tải về trong bộ Android SDK[6].

1.4.2. Phiên bản Android 1.6

Android Donut là một phiên bản đã ngừng phát triển của hệ điều hành Android, được đặt tên mã theo chủ đề món tráng miệng. Được phát triển bởi Google, Donut ra mắt vào mùa thu năm 2009 cho một loạt các điện thoại thông minh, bổ sung thêm các tính năng mới như hỗ trợ điện thoại dùng mạng CDMA, hỗ trợ thêm cho các kích thước màn hình, và thêm tính năng chuyển đổi văn bản ra thoại[6].

1.4.3. Phiên bản Android 2.0 - 2.1

Android Eclair là một phiên bản đã ngừng phát triển của hệ điều hành Android được phát triển bởi Google. Được giới thiệu vào ngày 26 tháng 10 năm 2009, Android 2.1 được xây dựng dựa trên những thay đổi quan trọng của phiên bản Android 1.6 "Donut". Hai bổ sung quan trọng trên Eclair là hỗ trợ NFC (sử dụng trong giải pháp thanh toán di động) và SIP (sử dụng trong điện thoại internet (VoIP) [6].

1.4.4. Phiên bản Android 2.2 - 2.2.3

Android Froyo là một phiên bản đã ngừng phát triển của hệ điều hành Android được Google phát triển, trải qua các phiên bản từ 2.2 tới 2.2.3. Nó được giới thiệu vào ngày 20 tháng 5 năm 2010 nhân hội nghị Google I/O 2010 [6].

Một trong những thay đổi nổi bật ở phiên bản Froyo là tính năng USB tethering và Wi-Fi hotspot. Các thay đổi khác là hỗ trợ dịch vụ Cloud to Device Messaging (C2DM), cho phép đẩy thông báo. Ngoài ra còn cải thiện tốc độ ứng dụng, được hiện thực thông qua biên dịch JIT [6].

1.4.5. Phiên bản Android 2.3 - 2.3.7

Android Gingerbread là một phiên bản đã ngừng phát triển của hệ điều hành Android vốn được phát triển bởi Google và ra mắt vào tháng 12 năm 2010. Phiên bản Gingerbread có các tính năng mới như hỗ trợ NFC (sử dụng trong các giải pháp thanh toán di động) và Session Initiation Protocol (SIP) (sử dụng trong điện thoại Internet VoIP).

Giao diện Gingerbread được tinh chỉnh rất nhiều, giúp người dùng làm chủ dễ dàng hơn, dễ sử dụng hơn, và tiết kiệm năng lượng hơn. Một bộ màu đơn giản với màu nền đen khiến cho thanh thông báo, menu và các thành phần giao diện khác nổi bật lên. Những cải thiện trong menu và thiết lập giúp người dùng điều hướng dễ dàng hơn [6].

Điện thoại thông minh Nexus S, ra mắt vào năm 2010, là chiếc điện thoại đầu tiên thuộc dòng Google Nexus chạy Gingerbread, và cũng là chiếc điện thoại đầu tiên trong dòng có tích hợp tính năng NFC.

Gingerbread sử dụng nhân Linux phiên bản 2.6.35.

1.4.6. Phiên bản Android 3.0 - 3.2.6

Android Honeycomb là một phiên bản hệ điều hành Android đã ngừng phát triển vốn được thiết kế dành riêng cho các thiết bị với màn hình lớn, đạc biệt là máy tính bảng. Được phát triển bởi Google, nó xuất hiện lần đầu tiên trên Motorola Xoom vào tháng 2 năm 2011. Ngoài các tính năng mới, Honeycomb còn giới thiệu một chủ đề giao diện người dùng mới với tên gọi "holographic" và một mô hình tương tác được xây dựng dựa trên các tính năng chính của Android, như đa nhiệm, thông báo và widget. [6]

1.4.7. Phiên bản Android 4.0 - 4.0.4

Android Ice Cream Sandwich là một phiên bản đã ngừng phát triển của hệ điều hành Android vốn được Google phát triển. Ra mắt vào ngày 19 tháng 10 năm 2011, Android 4.0 được xây dựng dựa trên những thay đổi quan trọng của phiên bản Android 3.0 "Honeycomb" (vốn chỉ dành cho máy tính bảng), trong một nỗ lực để tạo ra một nền tảng thống nhất dành cho cả điện thoại thông minh và máy tính bảng, trong khi đơn giản hóa hiện đại hóa các trải nghiệm Android xung quanh một tập hợp các hướng dẫn giao diện con người. Là một phần của nỗ lực này, Android 4.0 giới thiệu một bộ giao diện trực quan với tên mã "Holo", được xây dựng quanh một thiết kế sạch và đơn giản hơn, cùng một kiểu chữ mặc định mới với tên gọi Roboto. [6]

Android 4.0 cũng thiệu nhiều nhiều tính năng khác, bao gồm làm mới màn hình chủ, hỗ trợ NFC và cho phép "beam" nội dung sang thiết bị khác (mà sử dụng cùng công nghệ), và trình duyệt web được cập nhật, ứng dụng quản lý danh bạ tích hợp với mạng xã hội, khả năng truy cập máy ảnh và điều khiển âm nhạc từ màn hình khóa, hỗ trợ thư thoại trực quan, nhận diện khuôn mặt ("Face Unlock") để mở khóa thiết bị, cùng khả năng giám sát và giới hạn dữ liệu di động, và các cải tiến nội bộ khác. [6]

Android 4.0 nhận được nhiều đánh giá tích cực từ các nhà phê bình, họ khen ngợi giao diện hệ điều hành được tân trang, gọn gàng hơn so với các phiên bản trước, cùng với các cải thiện về hiệu suất và tính năng. Tuy nhiên, các nhà phê bình vẫn cảm thấy các ứng dụng gốc của Android 4.0 vẫn còn thua kém về chất lượng và tính năng so với các ứng dụng tương đương của bên thứ ba, và một số tính năng mới của hệ điều hành, đặc biệt là "mở khóa bằng khuôn mặt", không cần thiết lắm. [6]

1.4.8. Phiên bản Android 4.1 - 4.3.1

Android Jelly Bean là tên được đặt cho 3 phiên bản chính của hệ điều hành Android mobile operating system developed by Google, trải qua các phiên bản từ 4.1 đến 4.3.1.

Phiên bản đầu tiên, 4.1, được công bố tại hội nghị nhà phát triển Google I/O vào tháng 6 năm 2012, tập trung vào thiết kế cải thiện hiệu suất đẻ hệ điều hành chạy mượt và nhanh hơn, cải thiện hệ thống thông báo, cho phép thông báo có thể được “mở rộng” với các nút chức năng, và các thay nổi nội bộ khác. Hai phiên bản còn lại cũng có cùng tên Jelly Bean, được phát hành tương ứng vào tháng 10 năm 2012 và tháng 7

năm 2013, trong đó phiên bản 4.2 gồm tối ưu hóa, hỗ trợ nhiều người dùng cho máy tính bảng, widget cho màn hình khóa, tùy chỉnh nhanh, và screen saver, còn phiên bản 4.3 gồm các cải tiến và cập nhật nội bộ cho nền tảng Android. [6]

1.4.9. Phiên bản Android 4.4 - 4.4.4

Android KitKat là một phiên bản của hệ điều hành di động Android được phát triển bởi Google, kéo dài từ phiên bản 4.4 đến 4.4.4. Được tiết lộ vào ngày 3 tháng 9 năm 2013, KitKat được chủ yếu tập trung vào tối ưu hóa hệ điều hành để cải thiện hiệu năng cho các thiết bị giá rẻ với tài nguyên hạn chế. [6]

1.4.10. Phiên bản Android 5.0 - 5.1.1

Android Lollipop là một phiên bản của hệ điều hành di động Android phát triển bởi Google, mở rộng giữa 5.0 và 5.1.1. Ra mắt vào 25 tháng 6 năm 2014, trong suốt hội nghị Google I/O, nó có sẵn thông qua cập nhật OTA chính thức vào 12 tháng 11 năm 2014, cho các thiết bị chạy dịch vụ Android của Google (như thiết bị Nexus và Google Play edition). Mã nguồn của nó được đưa ra vào 3 tháng 11 năm 2014. [6]

Một trong những thay đổi lớn nhất trong Lollipop là được thiết kế lại giao diện người dùng được xây dựng dựa trên ngôn ngữ thiết kế gọi là Material Design. Những thay đổi khác bao gồm cải tiến thanh thông báo, có thể truy cập từ màn hình khóa và hiển thị trong ứng dụng như banner trên màn hình. Google còn thay đổi bên trong nền tảng này, với Android Runtime (ART) chính thức thay thế cho Dalvik để cải thiện hiệu năng ứng dụng, và tối ưu hóa việc sử dụng pin, gọi nội bộ là Project Volta. [6]

1.4.11. Phiên bản Android 6.0 - 6.0.1

Android Marshmallow (*có tên mã là M trong quá trình phát triển*) là phiên bản lớn thứ 6 của hệ điều hành Android. Được giới thiệu lần đầu vào tháng 5 năm 2015 tại Google I/O, nó đã được chính thức phát hành vào tháng 10 năm 2015.

Marshmallow chủ yếu tập trung vào cải thiện tổng thể trải nghiệm người dùng so với Lollipop, giới thiệu một cấu trúc quyền truy cập mới, các API mới cho các trợ lý ảo theo ngữ cảnh (nổi bật nhất qua chức năng "Now On Tap") một chức năng mới để tìm kiếm Google bằng cách quét các văn bản trong một ứng dụng và biến nó thành một giao diện đơn giản hơn bởi Google Now), một hệ thống quản lý điện năng mới sẽ giảm các hoạt động ngầm khi thiết bị đang không được sử dụng, hỗ trợ gốc cho nhận

dạng vân tay và cổng kết nối USB Type-C, khả năng chuyển dữ liệu và ứng dụng sang một thẻ microSD và dùng nó làm bộ nhớ chính, cùng nhiều thay đổi khác. [6]

1.4.12. Phiên bản Android 7.0 - 7.1.2

Android Nougat là một phiên bản phát hành của hệ điều hành Android. Lần đầu được phát hành dưới dạng một bản dựng beta vào ngày 9 tháng 3 năm 2016, nó đã được chính thức phát hành vào ngày 22 tháng 8 năm 2016, với các thiết bị Nexus được nhận bản cập nhật đầu tiên. [6]

Android 7.0 giới thiệu những thay đổi đáng chú ý tới hệ điều hành và nền tảng phát triển của nó, bao gồm khả năng hiển thị nhiều ứng dụng trên màn hình cùng một lúc bằng cách chia màn hình, hỗ trợ trả lời thông báo trực tiếp trong thẻ thông báo, cũng như một môi trường Java dựa trên OpenJDK và hỗ trợ hàm API vẽ đồ họa Vulkan, và cập nhật hệ thống "liên tục" trên các thiết bị được hỗ trợ.

1.4.13. Phiên bản Android 8.0 - 8.1

Android Oreo (tên mã phát triển là **Android O**) là phiên bản lớn thứ tám của hệ điều hành di động Android. Nó được phát hành lần đầu dưới dạng một phiên bản alpha xem trước cho nhà phát triển vào ngày 21 tháng 3 năm 2017. Bản xem trước thứ hai được phát hành ngày 17 tháng 5 năm 2017, được coi là phiên bản beta, và phiên bản xem trước thứ ba được phát hành ngày 8 tháng 6 năm 2017 với phần API được hoàn thiện. Vào ngày 24 tháng 7 năm 2017, một bản xem trước thứ tư được phát hành bao gồm những tính năng hệ thống cuối cùng cùng với những sửa lỗi và cải tiến mới nhất. Oreo được chính thức phát hành công khai vào ngày 21 tháng 8 năm 2017. [6]

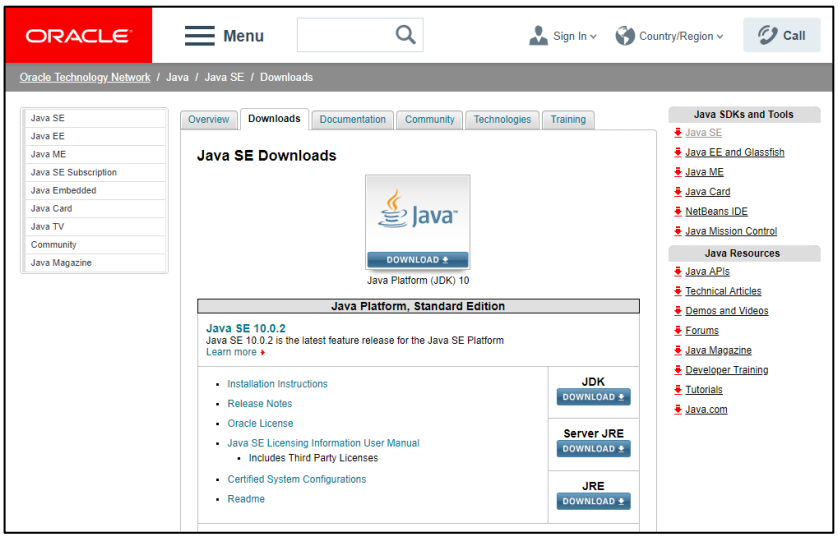
CHƯƠNG 2: TẠO ỨNG DỤNG VỚI ANDROID STUDIO

2.1. Cài đặt môi trường lập trình Android

2.1.1. Cài đặt JAVA JDK

– Bước 1: Tải file cài đặt từ đường dẫn

http://www.oracle.com/technetwork/java/javase/downloads/index.html



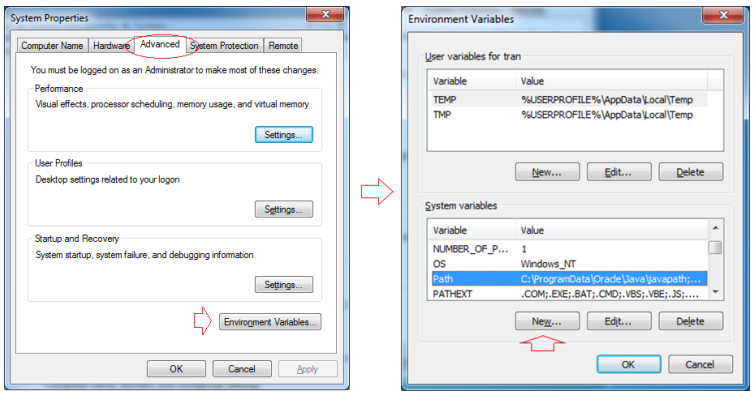
Lưu ý: Chọn phiên bản tương ứng với hệ diều hành đúng với máy đang sử dụng.

– Bước 2: Mở file cài đặt "jdk-*.exe" để tiến hành cài đặt

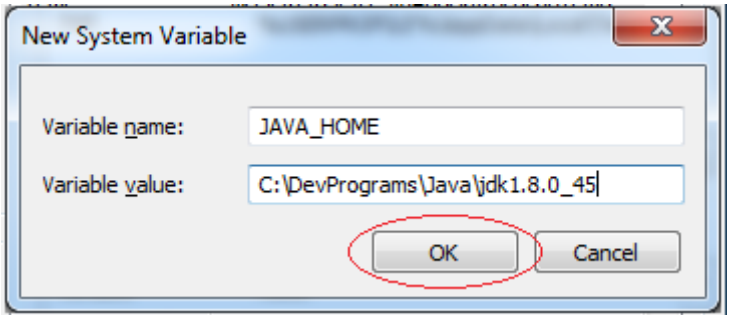
– Bước 3: Cấu hình biến môi trường cho Java

Việc này không bắt buộc, nhưng nếu trên máy tính của cài đặt nhiều phiên bản Java, thì việc cấu hình là cần thiết để xác định phiên bản java nào mặc định được sử dụng.

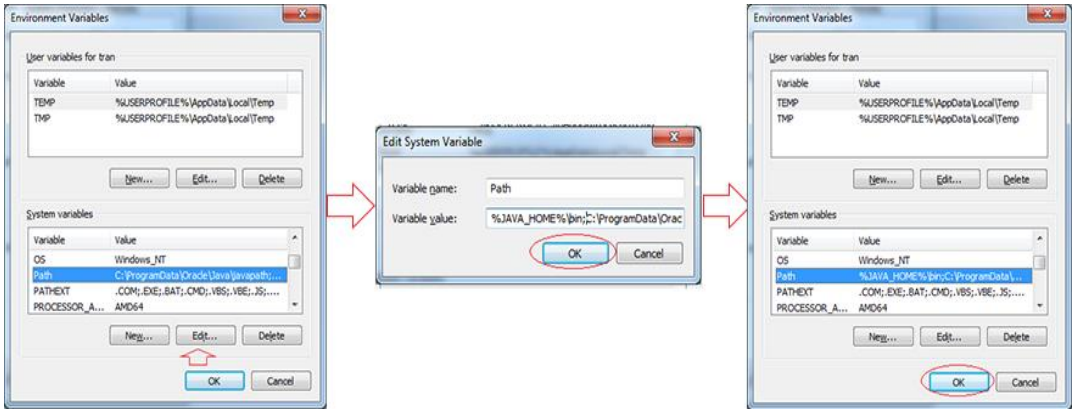
Trên Desktop, nhấn phải chuột vào Computer, chọn **Properties > Advanced system settings**.



– Nhập vào đường dẫn tới thư mục JDK:



– Tiếp theo đổi tên path là đã cài đặt xong Java JDK:



2.1.2. Cài đặt Android Studio

Android Studio là một nền tảng IDE [1](Integrated Development Environment) dùng để phát triển các ứng dụng android, được Google release vào khoảng đầu năm 2015 thay thế cho bản Eclipse cũ. Android Studio được phát triển dựa trên IntelliJ IDEA Community Edition - công cụ lập trình tốt nhất cho java, giúp cho các lập trình viên tạo ứng dụng, thực hiện các thay đổi một cách dễ dàng, bên cạnh đó có thể xem trước trong thời gian thực và thiết kế giao diện đẹp hơn trước. Tiếng Việt cũng đã được tích hợp trong Android Studio. Đặc biệt, Android Studio cho phép người dùng Import Project từ Eclipse sang và logic lập trình cũng tương tự.[6]

a. Cấu hình yêu cầu:

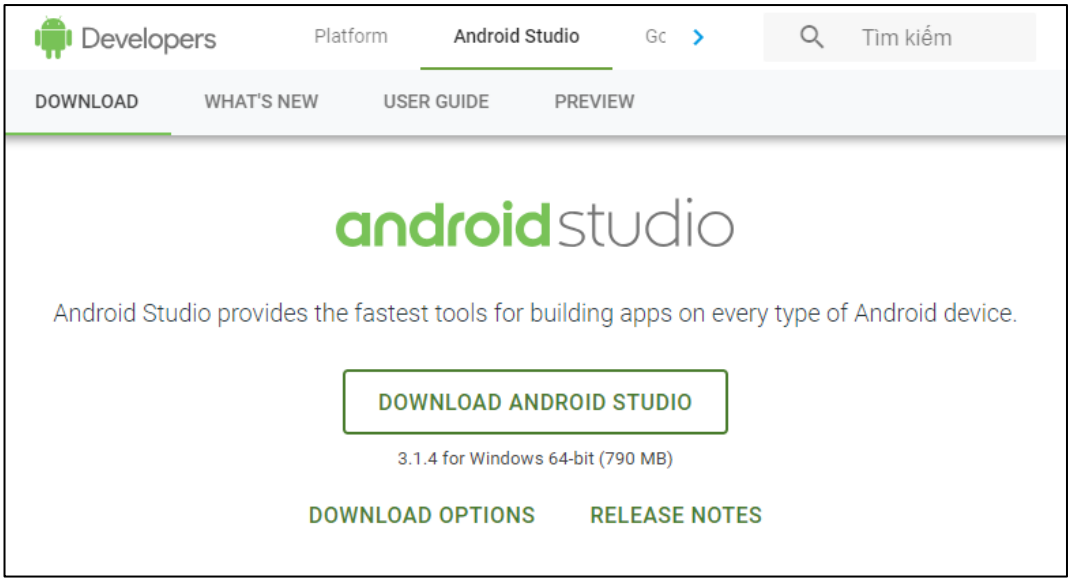
- Microsoft® Windows® 10/8/7/Vista/XP (32 or 64-bit)
- Tối thiểu 4 GB RAM,
- Ổ cứng trống ít nhất : 2 GB để cài đặt

[1] Môi tường lập trình phát triển tích hợp

- Độ phân giải tối thiểu 1280 x 800
- Java Development Kit (JDK) 7 trở lên
- Lựa chọn thêm cho accelerated emulator: Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality

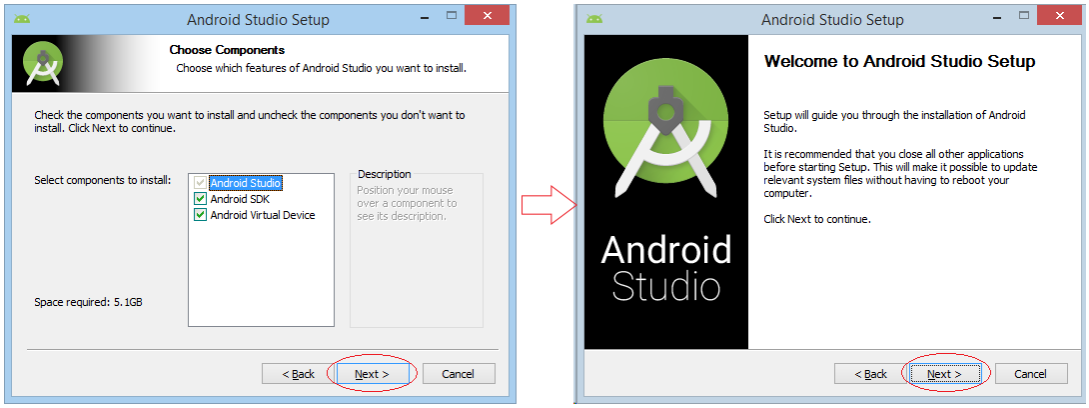
b. Tải file cài đặt

Dowload file cài đặt: *https://developer.android.com/studio/*

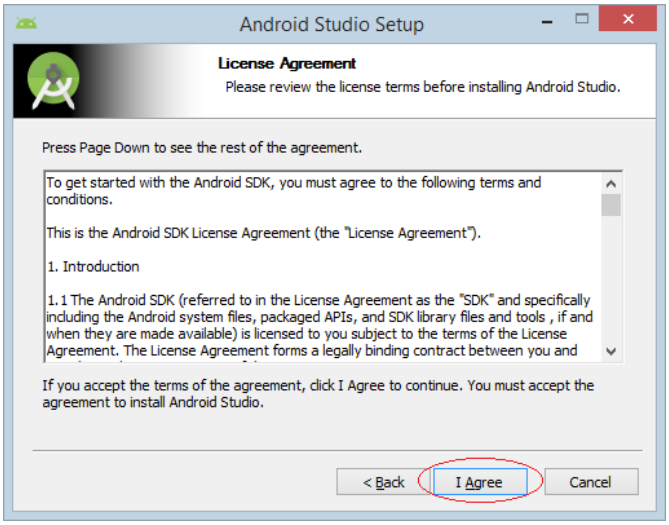


c. Cách cài đặt

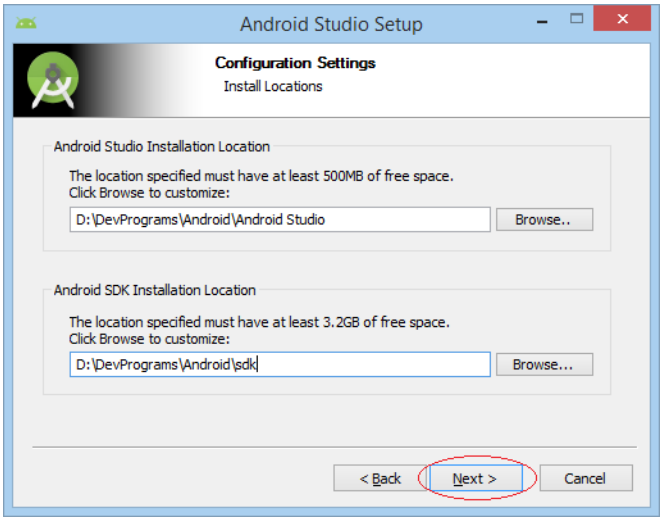
– Bước 1: Mở file cài đặt và ấn Next



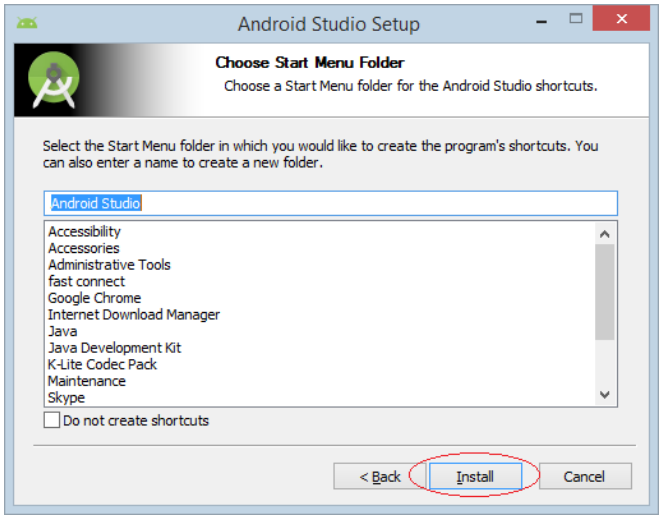
– Bước 2: Chọn “I agree” để xác nhận, đồng ý vợi các điều khoản và tiếp tục.



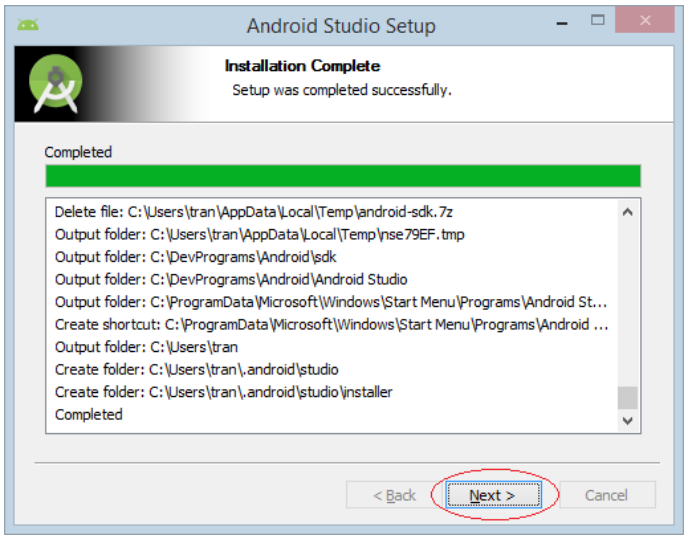
– Bước 3: Chọn nơi cài đặt Android Studio và Android SDK



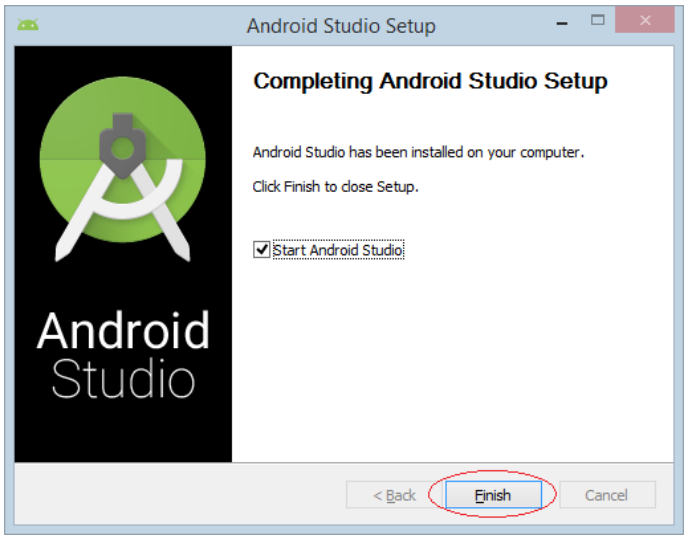
– Bước 4: Chọn Start Menu Folder. Ở bước này, có thể tích tạo short cut cho ứng dụng hoặc khôngBấm Install để bắt đầu cài đặt.



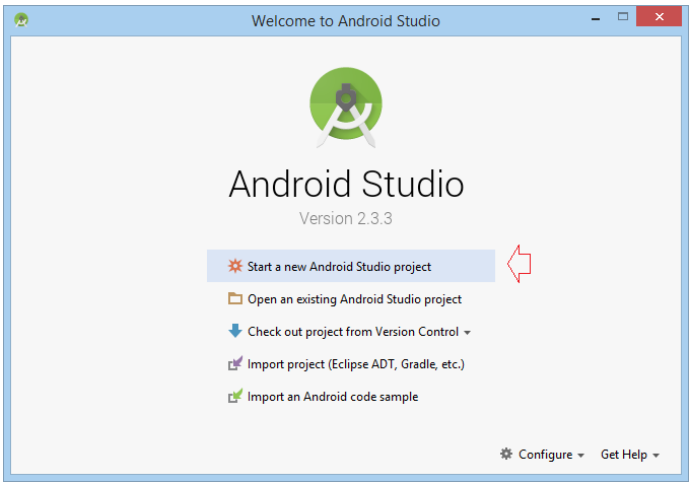
– Bước 5: Chờ hệ thống cài đặt các package cần thiết cho tới khi hoàn thành và bấm Next để tiếp tục.



– Bước 7: Bấm Finish và click Start Android Studio để khởi động phần mềm.



– Đã hoàn tất cài đặt Android Studio.



2.1.3. Máy ảo Android Genymotion

a. Yêu cầu

- Card đồ họa OpenGL 2.0.
- CPU hỗ trợ VT-x hoặc AMD-V và kích hoạt thiết lập BIOS.
- RAM: Tối thiểu 1 GB.
- Dung lượng trống của ổ cứng: Ít nhất là 2GB để cài đặt Genymotion và các máy ảo chạy Genymotion (đây chỉ là mức yêu cầu tối thiểu, bởi nếu sử dụng nhiều máy ảo cùng lúc và có nhiều ứng dụng, phần mềm cài đặt thì dung lượng trống có thể được yêu cầu nhiều hơn gấp 4 lần).
- Đảm bảo có kết nối Internet
- Độ phân giải màn hình: ít nhất 1366 x 768 pixel
- Oracle **VirtualBox 4.1** trở lên.
- Ngoài ra, người dùng cần có một tài khoản Genymotion để có thể sử dụng.

b. Tải file cài đặt:

Dowload: *https://www.genymotion.com/fun-zone/*

c. Cách cài đặt:

Bước 1: Tìm tới vị trí lưu file vừa tải về đầu tiên, sau đó click đúp chuột trái vào file đó để bắt đầu tiến hành. Giao diện đầu tiên sẽ cho lựa chọn ngôn ngữ để sử dụng.

Bước 2: Tùy theo ý muốn người dùng mà có thể **Next** ngay và cài đặt Genymotion ngay tại ví trí được gợi ý hoặc chọn **Browse** để tìm vị trí khác.

Bước 3: Tiếp tục **Next**.

Bước 4: Chọn mục **Create a desktop shortcut** để tạo biểu tượng cho phần mềm này trên máy tính, sau đó **Next** tiếp. Chọn vào **Install** để thực hiện cài đặt.

Bước 5: Rất có thể Genymotion sẽ tự động cài thêm một số phần mềm hỗ trợ, đồng thời hỏi có muốn cài đặt VirtualBox hay không? Nếu máy tính của đã có sẵn phần mềm VirtualBox mới nhất rồi thì thôi, còn nếu không, chọn **Yes** để cài đặt tiếp.

Bước 6: Tới đây, ngoài cửa sổ cài đặt Genymotion (sẽ tạm thời bị dừng lại), sẽ thấy có thêm một giao diện khác để tiến hành cài đặt VirtualBox.

Bước 7: Sẽ không có thêm bất cứ phần mềm hay thao tác thiết lập nào khác, nên các có thể yên tâm **Next** tới khi kết thúc > **Next** tiếp nào > Click chọn **Yes.**

Bước 8: Install.

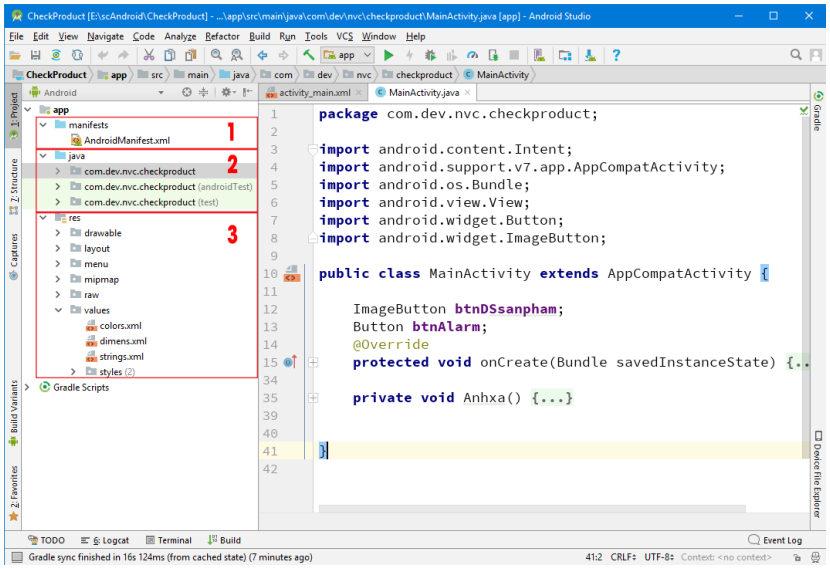
Bước 9: Bỏ dấu tick ở **Start Oracle WM VirtualBox** và chọn vào **Finish**.

Bước 10: Lúc này cửa sổ cài đặt cuối cùng của Genymotion xuất hiện, **Finish** để kết thúc quá trình.

Tài liệu tham khảo mục **2.1 cài đặt môi trường Android** [5]

2.2. Thành phần trong một dự án Android

Bao bọc một dự án Android[1] là thư mục app và phía dưới là ba thành phần chính của một project trong Android Studio:



Hình 2.2.1 Cấu trúc một project

[1] Android project

2.2.1. Tệp cấu hình Android[1]

[1] AndroidManifest.xml

a. Giới thiệu

Trong bất kì một dự án Android nào khi tạo ra đều có một tệp cấu hình *AndroidManifest.xml,* tệp này được dùng để định nghĩa các màn hình sử dụng, các quyền cũng như các giao diện cho ứng dụng. Đồng thời nó cũng chứa thông tin về phiên bản SDK cũng như main activity sẽ chạy đầu tiên. Tệp này được tự động sinh ra khi tạo một Android project. Trong tệp cấu hình bao giờ cũng có 3 thành phần chính đó là: application, permission và version. Hay nói cách khác đây là file dùng để cấu hình những thuộc tính cho ứng dụng của mà khi ứng dụng khởi chạy hệ điều hành có thể hiểu được và xử lí. [3][4]

b. Chức năng tệp cấu hình ứng dụng Android

– Đặt tên gói Java cho ứng dụng. Tên gói đóng vai trò như một mã nhận diện duy nhất cho ứng dụng.

– Nó mô tả các thành phần của ứng dụng - hoạt động, dịch vụ, hàm nhận quảng bá, và trình cung cấp nội dung mà ứng dụng được soạn bởi. Nó đặt tên các lớp triển khai từng thành phần và công bố các khả năng của chúng (ví dụ, những tin nhắn Intent mà chúng có thể xử lý). Những khai báo này cho phép hệ thống Android biết các thành phần là gì và chúng có thể được khởi chạy trong những điều kiện nào.

– Nó xác định những tiến trình nào sẽ lưu trữ các thành phần ứng dụng.

– Nó khai báo các quyền mà ứng dụng phải có để truy cập các phần được bảo vệ của API và tương tác với các ứng dụng khác.

– Nó cũng khai báo các quyền mà ứng dụng khác phải có để tương tác với các thành phần của ứng dụng.

– Nó liệt kê các lớp Instrumentation cung cấp tính năng tạo hồ sơ và các thông tin khác khi ứng dụng đang chạy. Những khai báo này chỉ xuất hiện trong bản kê khai khi ứng dụng đang được phát triển và thử nghiệm; chúng bị loại bỏ trước khi ứng dụng được công bố.

– Nó khai báo mức tối thiểu của API Android mà ứng dụng yêu cầu.

– Nó liệt kê các thư viện mà ứng dụng phải được liên kết với.

Cấu trúc tệp *AndroidManifest.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<manifest>
    <uses-permission /> <permission /> <permission-tree />
<permission-group />
    <instrumentation />
    <uses-sdk /> <uses-configuration /> <uses-feature />
    <supports-screens />
    <compatible-screens />
    <supports-gl-texture />
    <application>
        <activity>
            <intent-filter> <action /> <category /> <data />
</intent-filter>
            <meta-data />
        </activity>
        <activity-alias>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </activity-alias>
        <service>
            <intent-filter> . . . </intent-filter>
            <meta-data/>
        </service>
        <receiver>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </receiver>
        <provider>
            <grant-uri-permission />
            <meta-data />
            <path-permission />
        </provider>
        <uses-library />
    </application>
</manifest>
```

c. Một số thẻ trong AndroidManifest.xml

`<uses-permission>…<permission/>`: cú pháp lệnh để xin cấp một quyền gì đó khi ứng dụng tương tác với dữ liệu nào đó mà google không cho phép dùng tùy ý.

`<activity>…</activity>`: thẻ để khai báo các activity trong ứng dụng và các thuộc tính của activity đó, nếu như không khai bao thì khi khởi chạy ứng dụng sẽ lỗi.

`<service>…</service>`: một dịch vụ chạy ngầm trên ứng dụng của kể cả khi ứng dụng đã tắt đi và khi sử dụng phải khai báo chúng trong AndroidMainifest.

`<receiver>…</receiver>`: khi báo khi sử dụng Broadcast Reciver[1], cái này là để lắng nghe các sự kiện thay đổi của hệ thống như khi bất wifi, khi sạc pin…

`<uses-sdk>…</ uses-sdk >`: Thẻ xác định phiên bản SDK.

2.2.2. Thư mục Java

Đây chính là nơi chứa các nơi chứa các tệp tin, thư mục[2] của dự án, có thể tạo các gói ở đây và bên trong là các lớp.

2.2.3. Thư mục Res

Gói Drawable: Đây là gói chứa các file hình ảnh, config xml… trong dự án android.Ví dụ như muốn ứng dụng sử dụng một hình ảnh nào đó là backgroud thì ảnh đó sẽ bỏ vào thư mục này.Hoặc một muốn điều chỉnh một nút button khi click vào màu xanh còn khi không click vào màu trắng thì sẽ config trong file xml và lưu vào trong này.[2][3]

Gói Layout: Đây là gói lưu các file xml về giao diện của các màn hình ứng dụng của. Ở trên phần số một có các package lưu các class, các class này sẽ kết nối với các file xml trong thư mục layout nào để tạo nên một màn hình có giao diện cho người dùng thao tác.[1]

Gói Mipmap: Đây là gói sẽ chứa ảnh logo ứng dụng chúng ta, lúc này có nói là các file hình ảnh sẽ được chứa trong thư mục drawable nhưng ngoại lệ thằng ảnh logo thì chứa trong thư mục ***mipmap*** này cho chuẩn.[2]

Gói Values: Sẽ có rất nhiều file ở bên trong như sau:

- **color.xml**: đây là file định nghĩa các mã màu trong dự án android của ban, khi sử dụng màu nào chỉ cần gọi tên tên mã màu đã định nghĩa trong đây ra là xong.
- **dimens.xml**: đây là file mà sẽ định nghĩa ra các kích thước như cỡ chữ, chiều cao, chiều rộng các view. Tại sao không set cứng mà phải định nghĩa ở

[1] Một trong các lớp của android
[2] Package

đây? Như các biết thì thiết bị Android có cả ngàn chiếc và 1 đống nhà sản xuất nó khác IOS hoàn toàn thế nên sẽ có rất nhiều kích thước màn hình khác nhau loại 4 inch, 5inch, 5.5 inch… chính vì thế khi set một đoạn văn bản nào đó kích thước là 10sp thì màn hình 4 inch hiển thị rất đẹp chứ qua 5.5 là nó rất nhỏ. Đây là lí do có file dimens.xml nhé.

- **strings.xml**: đây là file định nghĩa các đoạn văn bản trong ứng dụng Android của ví dụ như có một đoạn văn bản mà sử dụng đi sử dụng lại trong các màn hình khác nhau, khi set cứng ở nhiều nơi thì khi cần chỉnh sửa thì phải tìm hết tất cả và sửa lại.Bây giờ định nghĩa đoạn văn bản đó trong đây và khi dùng thì gọi ra sử dụng và sau này chỉnh sữa chỉ cần sửa trong đây là xong, nó sẽ apply tất cả mọi nơi.[3]
- **styles.xml**: đây chính là nơi định nghĩa các giao diện của các file layout trong thư mục layout đã nói phía trên. Kiểu như thế này nhé, muốn chỉnh một nút Button chiều cao 10dp, chiều rộng 10dp, màu xanh… và lại sử dụng kiểu thiết kế này ở 5 màn hình khác nhau. không thể mỗi màn hình lại định nghĩa lại như thế sẽ làm duplicate code (lặp lại) và sẽ không tối ưu tí nào cả.Thay vào đó chỉ cần định nghĩa một file giao diện như trên và ở mỗi màn hình chỉ cần gọi là xong.

2.2.4. Tệp Build.grade

Build.grade là file để thực thi, xây dựng, đóng gói cho dự án Android ấn định phiên bản SDK, Vesion ứng dụng, package, thêm thư viện ngoài…[4]

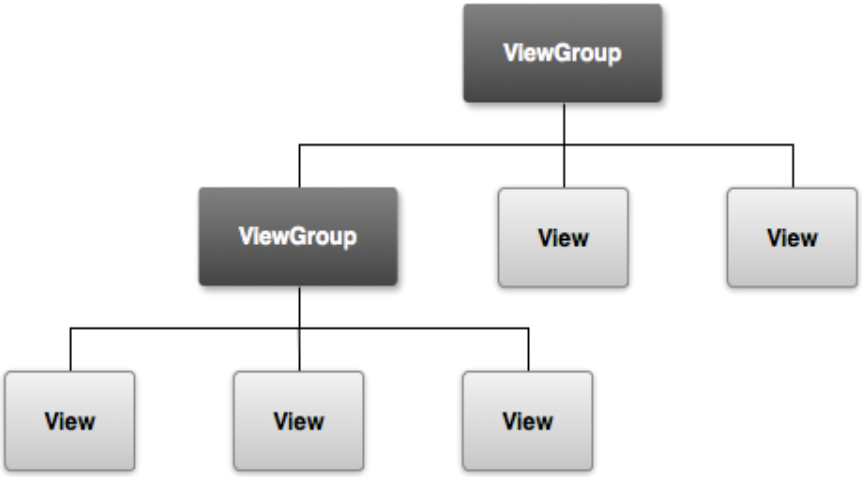
2.3. Thành phần giao diện

2.3.1. Nhóm hiển thị

a. Khái niệm

Nhóm hiển thị[1] là một lớp được sử dụng để chứa các thành phần hiển thị[2] và nhóm hiển thị khác để tổ chức và kiểm soát layout của một màn hình. Các đối tượng ViewGroup được sử dụng cho việc tạo ra một hệ thống phân cấp của các đối tượng View do đó có thể tạo các layout phức tạp hơn.

[1] ViewGroup
[2] View



Hình 2.3.1.1. Sơ đồ phân cấp các thành phần giao diện

b. Một số viewgroup cơ bản

LinearLayout: Tồn tại để hiển thị các phần tử theo một thứ tự xếp chồng lên nhau theo chiều ngang hoặc chiều dọc. LinearLayout cũng có thể được sử dụng để gán weight cho các phần tử `View` con để các phần tử được cách khoảng trên màn hình theo tỉ lệ tương ứng với nhau.

RelativeLayout: Lớp con này của `ViewGroup` cho phép hiển thị các phần tử trên màn hình tương đối với nhau, cung cấp nhiều tính linh hoạt hơn và tự do trong cách layout của xuất hiện so với `LinearLayout`.

FrameLayout: Được thiết kế để hiển thị một View con tại một thời điểm, FrameLayout vẽ các phần tử trong một ngăn xếp và cung cấp một cách đơn giản để hiển thị một phần tử trên các kích cỡ màn hình khác nhau.

ScrollView: Một lớp mở rộng của `FrameLayout`, lớp `ScrollView` xử lý việc cuộn các đối tượng con của nó trên màn hình.

RecyclerView:Lớp `RecyclerView` là một lớp con của ViewGroup, nó liên quan đến các lớp `ListView` và `GridView` và nó được cung cấp bởi Google thông qua thư viện hỗ trợ RecyclerView cho các phiên bản Android cũ hơn. Lớp `RecyclerView` đòi hỏi việc sử dụng các mẫu thiết kế view holder để tái sử dụng phần tử một cách có hiệu quả và nó hỗ trợ việc sử dụng một `LayoutManager`, một thành phần trang trí, và một phần tử động để làm cho thành phần này vô cùng linh hoạt và đơn giản.

CoordinatorLayout: Được thêm gần đây vào thư viện hỗ trợ thiết kế, lớp CoordinatorLayout sử dụng một đối tượng Behavior để xác định cách các phần tử *View* con sẽ được sắp xếp và di chuyển khi người dùng tương tác với ứng dụng của .

Tài liệu tham khảo mục 2.3.1.b [4]

2.3.2. Thành phần hiển thị

a. Khái niệm

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kiểu View và ViewGroup. Mỗi một kiểu là được kế thừa từ lớp View và tất cả các kiểu đó được gọi là các Widget. Tất cả mọi Widget đều có chung các thuộc tính cơ bản như là cách trình bày vị trí, background, kích thước, lề,… Tất cả những thuộc tính chung này được thể hiện hết ở trong đối tượng View. Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc phân. Một screen là một tập hợp các Layout và các widget được bố trí có thứ tự. Để thể hiện một screen thì trong hàm onCreate() của mỗi activity cần phải được gọi một hàm setContentView(R.layout.main); hàm này sẽ load giao diện từ file XML lên để phân tích thành mã bytecode.

b. Một số view cơ bản trong Android

Tên view	Chức năng
TextView	Cho phép người dùng hiển thị một đoạn văn bản lên màn hình mà không cho phép người dùng sửa nó.
EditText	Cho phép người dùng nhập, xóa, sửa một đoạn văn bản vào trong đó. Có nhiều dạng EditText khác nhau như: Plaintext, Person Name, Password, Email, Phone, ….
Button	Dùng để thiết lập các sự kiện khi người dùng thao tác với nó.
ImageButton	ImageButton: là dạng button nhưng có thể chèn thêm hình ảnh vào để giao diện thêm sinh động, trực quan hơn.
CheckBox	Một dạng button đặc biệt chỉ có hai trạng thái là check và uncheck.
ToggleButton	Có thể xem nó như một Checkbox và có kèm theo hiệu ứng ánh sáng bật/tắt thể hiện trạng thái Check/Uncheck.
RadioButton	Chọn một trong các Radio.
Spinner	Là view thể hiện quá trình xử lý diễn ra bên dưới ứng

Tên view	Chức năng
	dụng, tạo cảm giác thực cho người dùng.Có 2 dạng progressBar là Large và Horizontal.
ListView	Chỉ cho phép chọn một trong nhóm lựa chọn.
GridView	Là view được hiển thị dưới dạng lưới gồm nhiều item con bên trong và ta có thể tùy chỉnh nội dung cũng như các đối tượng nằm bên trong item một cách tùy ý.
ViewFlipper	Cho phép định nghĩa một tập hợp nhiều View nhưng chỉ có một View hiển thị tại một thời điểm và hỗ trợ hiệu ứng chuyển đổi giữa các View.
QuickContactBage	Hiển thị hình ảnh gắn liền với một đối tượng bao gồm số điện thoại, tên, email đồng thời hỗ trợ việc gọi, nhắn tin sms, email hay tin nhắn tức thời (IM – instant message).
ImageView	Hiển thị hình ảnh.
Switch	Tồn tại hai trặng thái đóng mở.
ProgressBar	Thể hiện tiến trình, mức độ.

c. Thuộc tích cơ bản widget

Tên thuộc tính	Công dụng
layout_width	Chiều rộng của control (Tính theo trục Ox từ góc trên bên trái qua phải).Có 3 thuộc tính: **wrap_content** (Co dãn theo nội dung control), **match_parent** (kích thươc chiều ngang bằng kích thước đối tượng chứa nó).
layout_heigth	Chiều cao của control (Tính theo trục Oy từ góc trên bên trái xuống dưới). Có 3 thuộc tính như ***layout:width***
text	Nội dung hiển thị.
texcolor	Màu chữ.
background	Màu nền của view.
id	Định danh của view.
gravity	Căn lề cho nọi dung của view.
textStyle	Thiết lập dáng chữ: đậm, kiểu in nghiêng, gạch chân.
textSize	Thiết lập cỡ chữ.
fontFamily	Thiết lập họ kiểu chữ.
inputType	Thiết lập kiểu nhập dữ liệu.
hint	Đoạn văn bản gợi ý cho người dùng biết về chức năng hay ràng buộc gì đó của view.

2.4. Vòng đời ứng dụng Android

– Các Activity[1] trong hệ thống được quản lý như một ngăn xếp activity[2]. Khi một activity mới bắt đầu nó được đặt lên đầu của ngăn xếp và trở thành Running Activity[3] đồng thời activity trước đó sẽ nằm ngay phía dưới trong ngăn xếp đó và sẽ bị ẩn đi cho đến khi activity ở trên thoát ra khỏi ngăn xếp.

– Một Activity gồm 4 trạng thái chính:

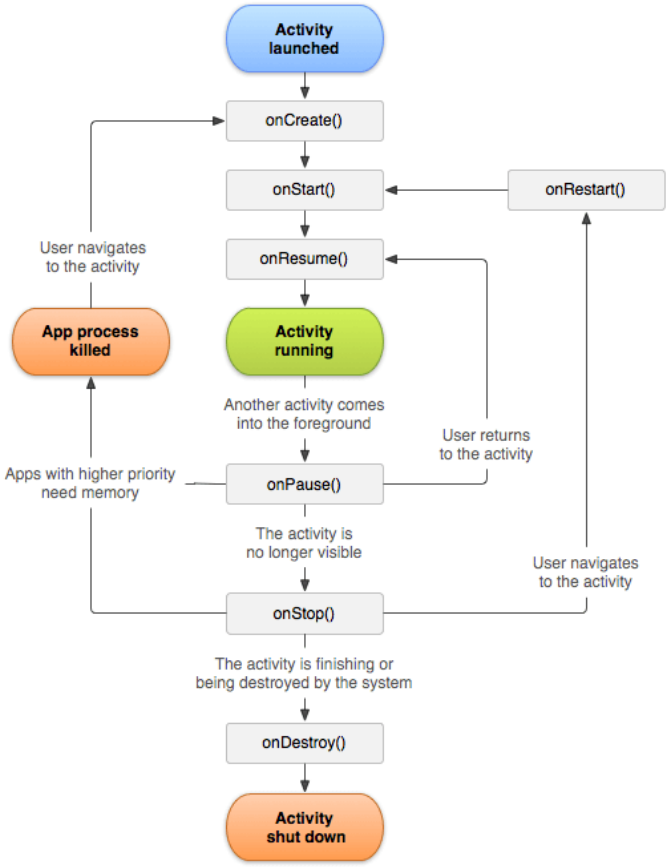
- Nếu activity ở phía trên của màn hình (hay ở trên cùng của ngăn xếp), thì nó đang ở trạng thái active (hoạt động) / running (đang chạy). Ví dụ khi ta cần gọi điện thì activity bấm số đó đang ở trạng thái active.

[1] Lớp được người lập trình xây dựng kế thừa từ lớp Activity
[2] Activity stack
[3] Activity đang chạy

• Nếu activity không thể tương tác nhưng vẫn nhìn thấy (khi mà bị che bởi một activity khác nhưng người dùng vẫn có thể nhìn thấy nó ở phía sau) thì activity này đang ở trạng thái paused[1]. Khi ở trạng thái này activity có thể bị xóa bỏ bởi hệ thống khi thiết bị thiếu bộ nhớ. Ví dụ khi có một activity khác dạng dialog hiện lên chỉ che đi một phần của activity hiện tại thì activity vào trạng thái paused.

• Nếu activity hoàn toàn bị che khuất bởi activity khác thì nó đang ở trạng thái stopped[2]. Activity này vẫn giữ được tất cả trạng thái và thông tin, nhưng không còn hiển thị với người dùng và thường xuyên bị xóa bỏ bởi hệ thống khi thiếu bộ nhớ. Ví dụ khi ta tắt màn hình thì khi đó activity vào trạng thái stopped.

• Nếu activity ở trạng thái paused hay stopped, hệ thống có thể xóa bỏ activity đó khỏi bộ nhớ bằng cách yêu cầu nó tự kết thúc hoặc xóa bỏ tiến trình của nó. Khi acitivty đó hiển thị lại với người dùng thì sẽ được khởi tạo lại và khôi phục lại trạng thái trước đó.



[1] Tạm dừng
[2] Đã dừng

Hình ảnh 2.4.1 Vòng đời của một Activity

2.5. Lớp Intent

a. Khái niệm

Intent là một thành phần quan trọng trong Android. **Intent** cho phép các thành phần ứng dụng có thể yêu cầu các hàm từ các thành phần ứng dụng android khác.

Intent là đối tượng của lớp `android.content.Intent`. Mã của có thể gửi **Intent** vào hệ thống **Android** với chỉ định thành phần mục tiêu gửi đến.

Một **Intent** có thể chứa dữ liệu thông qua một `Bundle`. Dữ liệu này có thể được sử dụng bởi các thành phần tiếp nhận.

b. Các loại Intent

Android hỗ trợ hai loại Intent là Intent tường minh (explicit) và Intent không tường minh (implicit).

Một ứng dụng có thể xác định thành phần mục tiêu một cách trực tiếp vào Intent (mục tiêu được yêu cầu là rõ ràng) hoặc yêu cầu hệ thống Android đánh giá các thành phần đã đăng ký trên dữ liệu đích để chọn ra một cái để gửi yêu cầu đến (Intent không tường minh)[2].

Intents tường minh:

Intent tường minh (Explicit intents): Là những ý định (intent) chỉ định rõ ràng tên của các thành phần mục tiêu để xử lý; trong đó, trường mục tiêu (tùy chọn) được sét một giá trị cụ thể thông qua các phương thức `setComponent()` hoặc `setClass()`.

Ví dụ tạo Intents tường minh:

```
//Màn hình 1:
//cách 1:
//Sử dụng Bundle
Intent intent1 = new Intent(this, ManHinh2Activity.class);
Bundle bundle = intent1.getExtras();
bundle.putString("khoa1","Giá trị 1");
bundle.putString("khoa1","Giá trị 2");
//cách 2:
//Sử dụng Bundle
Intent intent2 = new Intent(this, ManHinh2Activity.class);
Bundle bundle2 = new Bundle();
bundle2.putString("khoa1","Giá trị 1");
bundle2.putString("khoa2","Giá trị 2");
```

```
intent2.putExtra(bundle2);
//cách 3:
//Sử dụng putExtra()
Intent intent3 = new Intent(this, ManHinh2Activity.class);
intent3.putExtra("Khoa1","Giá trị 1");
intent3.putExtra("Khoa2","Giá trị 1");
//----------------------------------------
//màn hình 2:
Intent intent = this.getIntent();
String str_giatri1 = intent.getStringExtra("khoa1");
String str_giatri2 = intent.getStringExtra("khoa2");
```

– Intents không tường minh

Intent không tường minh (Implicit Intents): Là những ý định (intent) không chỉ định rõ một mục tiêu thành phần, nhưng bao gồm đầy đủ thông tin cho hệ thống để xác định các thành phần có sẵn là tốt nhất để chạy cho mục đích đó. Hãy xem xét một ứng dụng liệt kê các nhà hàng có sẵn ở gần . Khi bấm vào một tùy chọn nhà hàng cụ thể, ứng dụng sẽ hỏi một ứng dụng khác để hiển thị các tuyến đường đến nhà hàng đó. Để đạt được điều này, nó có thể gửi một ý định rõ ràng trực tiếp đến các ứng dụng **Google Maps**, hoặc gửi ý định ngầm, ý định sẽ được chuyển giao cho bất kỳ ứng dụng nào cung cấp các tính năng bản đồ (map) (chẳng hạn, **Yahoo Maps**).

Ví dụ Intents không tường minh:

```
//Một Intent không tường minh, yêu cầu xem một URL
//Một đường dẫn URL vốn là một thứ đuược xem trên trình duyệt
Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse("//google.com"));
startActivity(intent);
```

c. Truyền dữ liệu giữa các activities

– Truyền dữ liệu đến thành phần đích

Để truyền dữ liệu cho intent, ta dùng phương thức **putExtras().** Extras là **một cặp key/value**. key luôn luôn là kiểu string. value có thể sử dụng kiểu dữ liệu nguyên thủy hoặc đối tượng của String, Bundle, ...

– Thành phần tiếp nhận có thể lấy lại được đối tượng intent thông qua hàm **getIntent()**. Để lấy ra được dữ liệu, tùy thuộc vào kiểu dữ liệu truyền đi, sử dụng các phương thức **getStringExtra(), getIntExtra()**;

Ví dụ:

```
Intent intent = new Intent(MainActivity.this,
Main2Activity.class);
Intent.putExtra("name","Tên tôi là");
Intent intent = getIntent();
//-----------------------
String hoTen = intent.getStringExtra("name";
```

Ta có thể sử dụng đối tượng **Bundle**. Đóng gói tất cả dữ liệu vào trong Bundle xong truyền Bundle cho Intent.

Ví dụ:

```
Intent mIntent = new Intent(MainActivity.this,
Main2Activity.class);
Bundle mBundle = new Bundle();
mBundle.putString(key, value);
mIntent.putExtras(mBundle);
-------------------------------------------------------------
String value = getIntent().getExtras().getString(key);
```

– Sử dụng intent chia sẻ

Có rất nhiều ứng dụng android cho phép chia sẻ dữ liệu với những người khác, ví dụ: facebook, G+, Gmail... có thể gửi dữ liệu tới một vài thành phần nào đó. ví dụ dưới đây sẽ mô tả việc sử dụng intent như vậy

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/plain");
intent.putExtra(android.content.Intent.EXTRA_TEXT,"News for you!");
startActivity(intent);
```

– Lấy lại kết quả sữ liệu từ sub-activity

Một activity có thể được đóng lại thông qua button back trên điện thoại. Trong trường hợp đó, hàm **finish()** sẽ được thực thi. Nếu activity đã được khởi chạy cùng với hàm **startActivity(intent)**, người gọi không cần thiết phải có kết quả hoặc phản hồi từ activity mà có thể close ngay lập tức.

Nếu start một activity cùng với hàm `startActivityForResult()`, như vậy là mong muốn có phản hồi từ **sub-activity**. Khi một sub-activity kết thúc, hàm `onActivityResult()` trên sub-activity sẽ được gọi và có thể thực hiện hành động dựa trên kết quả.

Trong hàm `startActivityForResult()`, có thể xác định hành động sẽ khởi chạy. Hành động này sẽ mong muốn nhận về kết quả. Một hành động đã được khởi

chạy cũng có thể thiết lập một đoạn mã code mà người gọi có thể sử dụng để xác định hành động sẽ được cancel hoặc not.

Sub-activity sử dụng hàm `finish()` để khởi tạo một intent mới và truyền dữ liệu cho nó. Nó cũng thiết lập kết quả thông qua hàm `setResult()`

```
public void onClick(View view) {
   Intent i = new Intent(this,ActivityMot.class);
   i.putExtra("giati1","giá trị 1 ActivityMot");
   i.putExtra("giati2", "giá tị 2 ActivityMot");
   startActivityForResult(i, REQUEST_CODE);
}
```

Nếu sử dụng hàm `startActivityForResult()`, thì sau khi start activity sẽ gọi đến sub-activity.

Nếu hàm sub-activity kết thúc, nó sẽ gửi dữ liệu đến cái đã gọi nó thông qua intent. điều này được xử lý trên hàm `finish()`.

```
@Override
public void finish() {
Intent data = new Intent();
data.putExtra("returnKey1", "Swinging on a star. ");
data.putExtra("returnKey2", "You could be better then you are.
");
setResult(RESULT_OK, data);
super.finish();
}
```

Một sub-activity kết thúc, hàm onActivityResult() trong activity cha sẽ được gọi.

```
@Override
protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
if (data.hasExtra("returnKey1")) {
Toast.makeText(this, data.getExtras().getString("returnKey1"),
Toast.LENGTH_SHORT).show();
            }
     }
}
```

2.6. Shared prefenrences

2.6.1. Khái niệm

Đây là một **Interface** cho phép lưu trữ và đọc dữ liệu với bằng các cặp key và value và nó được lưu dưới dạng một file xml, dữ liệu nó có thể lưu là ở dạng nguyên thuỷ như: int, float, string, boolean, long. Dữ liệu của Shared Preferences sẽ được lưu

ở trong ứng dụng android luôn chính vì thế nếu các xoá ứng dụng đi hoặc là xoá dữ liệu app thì dữ liệu này sẽ hoàn toàn bị biến mất.

2.6.2. Cách sử dụng

a. Lưu dữ liệu xuống Shared Prenfernces

– Đầu tiên, để lưu dữ liệu xuống thì sẽ khởi tạo một đối tượng (biến) thuộc kiểu **Shared Preferences**, với cú pháp như sau:

```
SharedPreferences sharedPreferences = getSharedPreferences
(SHARED_PREFERENCES_NAME, Context.MODE_PRIVATE );
```

Trong đó:

- `SHARED_PREFERENCES_NAME`: là tên của file **Shared Preferences**.
- `Context.MODE_PRIVATE`: đây là một chế độ bảo mật dữ liêu trong android, khi để như vậy có nghĩa là chỉ cho ứng dụng hiện tại truy cập vào file Shared Preferences này thôi mà không một ứng dụng nào có quyền truy cập vào được.

Tạo đối tượng Editer từ biến sharedPreferences đã tạo ở trên, mục đích của thằng này là để có thể mở file **shared_preferences_name.xml** ra và đưa dữ liệu vào, cú pháp như sau:

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

Đưa dữ liệu vào:

```
editor.putX(String key,value);
```

Sau khi đã put dữ liệu xong thì sẽ gọi hàm **commit()** hoặc là **apply()** để xác nhận những thay đổi[4].

b. Cách lấy dữ liệu từ Shared Prenfernces

Đây là cách lấy ra tất cả các dữ liệu mà ở trên đã lưu:

```
SharedPreferences sharedPreferences =
getSharedPreferences(SHARED_PREFERENCES_NAME,
Context.MODE_PRIVATE);
boolean isFirtsLauncher =
sharedPreferences.getBoolean("IS_FIRTS_LAUNCHER",true);
int highScore = sharedPreferences.getInt("HIGH_SCORE",0);
float mark = sharedPreferences.getFloat("MARK",0.0f);
long downloadProgress =
sharedPreferences.getLong("DOWNLOAD_PROGRESS",0);
String name = sharedPreferences.getString("NAME","");
```

c. Cách xóa dữ liệu đã lưu xuống Shared Prenfernces

– Xoá đi dữ liệu nào đó đã lưu trong Shared Preferences thì chỉ cần gọi lệnh:

```
editor.remove(String key);
```

– Xoá đi hết tất cả giá trị đã lưu thì dùng cú pháp:

```
editor.clear();
```

– Gọi editor.apply() hoặc commit() để lưu thay đổi

2.7. BroadcastReceiver

2.7.1. Khái niệm

Broadcast Receiver là một trong bốn thành phần chính trong Android, với mục đích là lắng nghe các sự kiện, trang thái của hệ thông phát ra thông qua Intent nhờ đó mà các lập trình viên có thể xử lí được các sự kiện hệ thống ở bên trong ứng dụng[1].

Android BroadcastReceiver là một thành phần nơi có thể đăng ký sự kiện của hệ thống[1] hay ứng dụng[2] . Sẽ nhận được thông báo về các sự kiện một khi đã đăng ký. Việc phát tin (Broadcast) bắt nguồn từ hệ thống cũng như từ các applications[1]. Ví dụ về việc phát tin bắt nguồn từ hệ thống có thể kể ra như thông báo pin yếu. Với cấp độ ứng dụng có thể kể ra như khi download một file nhạc. Ứng dụng nhạc này sẽ nhận được thông báo về việc download, sau khi kết thúc sẽ đưa bài hát này vào danh sách để bật. Để thực hiện được việc này, ứng dụng nhạc cần thiết phải đăng ký cho sự kiện. Một ví dụ đơn giản khác, người dùng sẽ nhận thông báo khi pin của máy đã xuống dưới một mức độ nào đấy.

2.7.2. Chức năng chính

Broadcast Receiver có thể hoạt động được cả khi ứng dụng bị tắt đi, nghĩa là ở background chính vì vậy nó thường được sử dụng với service.

Ứng dụng cần xử lí một công việc nào đó mà phải phuộc thuộc vào hệ thống, chỉ khi hệ thông phát ra một sự kiện, hành động nào đó thì công việc này mới được thực hiện.Những trường hợp được ứng dụng nhiều nhất khi sử dụng Broadcast Receiver đó là: lắng nghe sự kiện thay đổi mạng, lắng nghe sự kiện pin yếu, lăng nghe tin nhắn, cuộc gọi đến…

Một công dụng tuyệt vời nữa đó là dùng để start Service của ứng dụng trong Broadcast Receiver.

---

[1] System
[2] Application

2.7.3. Cách sử dụng

a. Đăng ký một sự kiện với Broadcast Receiver

Chúng ta sẽ có hai cách để đăng ký Android Broadcast Receiver:

- **Một là phương thức tĩnh** khi đăng ký broadcast receiver trong ứng dụng thông qua file AndroidManifest.xml.
- **Hai là phương thức động** để đăng ký broadcast receiver thông qua việc sử dụng method Context.registerReceiver(). Các broadcast receiver được đăng ký động có thể được gỡ bởi method Context.unregisterReceiver().

Lưu ý: chúng ta phải hết sức cẩn thận khi tạo ra các broadcast receiver động. Nếu quên việc gỡ đăng ký broadcast receiver, hệ thống sẽ thông báo lỗi lộ (leaked) broadcast receiver.

b. Tạo Broadcast Receiver và sử dụng method onReceive()

Một BroadcastReceiver sẽ mở rộng (extends) class trừu tượng (abstract) BroadcastReceiver. Điều này có ý nghĩa phải sử dụng method onReceive() của class chính (base) này. Mỗi khi có sự kiện xảy ra, Android sẽ gọi method onReceive() ở broadcast receiver đã được đăng ký. Ví dụ, nếu đăng ký cho sự kiện ACTION_POWER_CONNECTED, mỗi khi di động được kết nối với nguồn điện, method onReceive() broadcast receiver của sẽ được gọi.

– Method **onReceive()** có 2 đối số (argument)

```
onReceive(Context context,Intent intent)
```

Context (phạm vi) có thể được sử dụng để chạy service hoặc activity và intent là object (đối tượng) cùng với action (hành động) đã sử dụng để đăng ký receiver của. Nó có thể chứa các thông tin thêm mà có thể dùng để implementation (thi hành).

Ví dụ:

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.widget.Toast;
public class NetworkChangeReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Network is turned ON/OF",
Toast.LENGTH_SHORT).show();
    }
}
```

c. Đăng ký Broadcast Receiver tĩnh

Giống như đã nói ở trên, broadcast receiver có thể đăng ký tĩnh thông qua file AndroidManifest.xml. Element (yếu tố) được sử dụng để định rõ sự kiện (event) mà receiver nên phản ứng lại. Chính vì thế mỗi lần sự kiện này (MyBroadcast) xảy ra, method onReceive() của MyBroadcastReceiver sẽ được gọi.

Khi method onReceive() kết thúc, BroadcastReceiver của sẽ bị hủy.

Một vài sự kiện cần có PERMISSIONS[1]

Ví dụ:

```
<action android:name="android.intent.action.PHONE_STATE" />
```

Chúng ta cần cho thêm:

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

d. Đăng ký broadcast receiver động

```
BroadcastReceiver mReceiver = new MyBroadcastReceiver();
registerReceiver(this.myReceiver,new
IntentFilter("MyBroadcast"));
```

Chúng ta cần tránh những task chạy lâu trong các Broadcast receiver. Đối với các task lâu chúng ta có thể chạy service trong receiver. Các receiver được đăng ký động sẽ được gọi trong UI[2] thread. Các receiver được đăng ký động sẽ hạn chế bất cứ thao tác UI, vì vậy phương thức `onReceive()` sẽ được chạy ở tốc độ cao nhất. Application có thể trở nên chậm chạp và phát sinh lỗi tồi tệ "Application Not Responding".

2.7.4. Các Intent trong Broadcast receiver

Các Intent được sử dụng trong broadcast receiver khác với Intent sử dụng để chạy activity. Intent dùng để broadcasting sẽ chạy ở background và người dùng sẽ không nhận thấy intent này. Tuy nhiên intent được sử dụng để chạy activity người dùng có thể nhìn thấy.

[1] Cho phép
[2] Viết tắt của User interface giao diện người dùng

2.7.5. Security issues with Android Broadcast receivers

Từ Android 3.1, hệ thống android sẽ không nhận được intent ngoài (external). Nếu sử dụng các phiên bản trước, hãy cân nhắc các điều sau:

- Hãy cho broadcast receiver không dùng được với các application ngoài (external) sử dụng android:export = false. Nếu không thì các application khác có thể lạm dụng chúng.
- Hãy chắc chắn rằng các broadcast mà gửi không nhận được từ các application khác. Hãy chỉ định hạn chế (limitation) để broadcast chỉ có thể nhận trong application.
- Cũng giống như thế khi đăng ký một receiver, hãy chắc chắn rằng không nhận các broadcast của các applications khác.

2.8. Thành phần dịch vụ trong Android

Một dịch vụ[1] là một thành phần chạy ngầm trên hệ điều hành để thực hiện các hoạt động dài hạn mà không cần phải tương tác với người sử dụng và nó hoạt động ngay cả khi ứng dụng bị phá hủy[4].

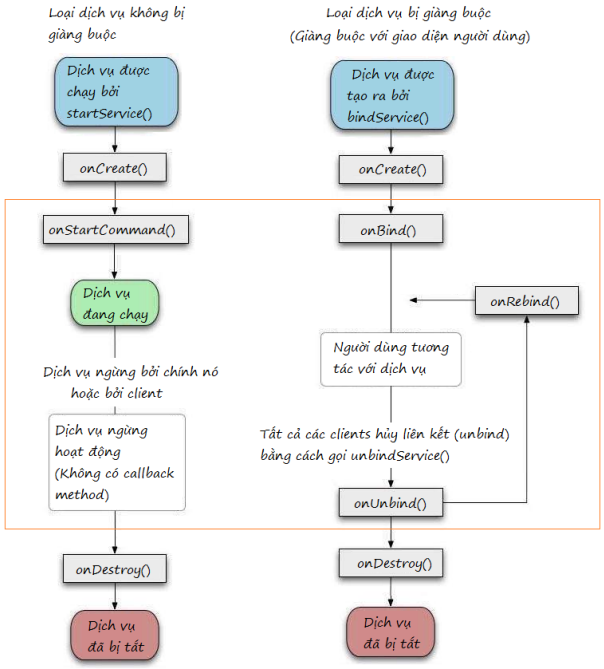
Một dịch vụ cơ bản có thể có hai loại.

Trạng thái	**Mô tả**
Started (Được khởi động)	Một dịch vụ được gọi là started (được khởi động) khi một thành phần ứng dụng, chẳng hạn như Activity khởi động nó bằng cách gọi startService(). Một khi được gọi, dịch vụ này có thể chạy ở chế độ nền vô thời hạn, thậm chí cả khi thành phần đã khởi động nó bị phá hủy. Dịch vụ này còn được gọi là dịch vụ không bị giàng buộc (Un Bounded Service).
Bound (Giàng buộc)	Một dịch vụ được giàng buộc (bound) khi một thành phần ứng dụng giàng buộc nó bằng cách gọi bindService(). Một dịch vụ ràng buộc cung cấp một giao diện client-server cho phép các thành phần tương tác với dịch vụ, gửi các yêu cầu, nhận kết quả, và thậm chí làm như vậy xuyên qua nhiều tiến trình với Interprocess communication (IPC)

[1] Service trong Android

Trạng thái	Mô tả
	(Truyền thông nhiều tiến trình).

Một dịch vụ có phương thức gọi lại chu kỳ vòng đời của nó (life cycle callback methods) mà có thực hiện (implement) để theo dõi những thay đổi trong trạng thái của dịch vụ và có thể thực hiện công việc ở giai đoạn thích hợp. Sơ đồ dưới đây về bên trái cho thấy vòng đời khi dịch vụ được tạo ra với ***startService()***, sơ đồ bên phải cho thấy vòng đời của dịch vụ được tạo ra bởi ***bindService()***.



Hình 2.8.1 Service trong Android

Để tạo ra một dịch vụ, tạo một lớp Java mở rộng lớp **Service** hoặc một trong các lớp con của nó. Lớp **Service** định nghĩa các phương thức callback khác nhau và quan trọng nhất được đưa ra dưới đây. Không cần phải thực hiện (implements) tất cả các phương thức callbacks.

Ngoài hai loại dịch vụ trên, có một dịch vụ khác gọi là **IntentService**. **IntentService** được sử dụng để thực hiện các nhiệm vụ một lần duy nhất, nghĩa là khi nhiệm vụ hoàn thành dịch vụ tự hủy.

So sánh các loại dịch vụ:

Unbound Service (Không giàng buộc)	**Bound Service (Giàng buộc)**	**Intent Service**
Unbounded Service được sử dụng để thực hiện nhiệm vụ lâu dài và lặp đi lặp lại.	**Bounded Service** được sử dụng để thực hiện nhiệm vụ ở nền (background) và giàng buộc với thành phần giao diện.	**Intent Service** được sử dụng để thực hiện các nhiệm vụ một lần duy nhất, nghĩa là khi nhiệm vụ hoàn thành dịch vụ tự hủy.
Unbound Service được khởi động bởi gọi startService().	**Bounded Service** được khởi động bởi gọi ***bindService()***.	**Intent Service** được khởi động bởi gọi ***startService()***.
Unbound Service bị dừng lại hoặc bị hủy bởi gọi một cách tường minh phương thức ***stopService()***.	**Bounded Service** bị gỡ giàng buộc hoặc bị hủy bởi gọi ***unbindService().***	**IntentService** gọi một cách không tường minh phương thức ***stopself()*** để hủy
Unbound Service độc lập với thành phần đã khởi động nó.	**Bound Service** phụ thuộc vào thành phần giao diện đã khởi động nó.	**Intent Service** độc lập với thành phần đã khởi động nó.

Các phương thức callback và mô tả:

Callback	**Description**
onStartCommand()	Hệ thống gọi phương thức này khi một thành phần khác, chẳng hạn như một **Activity**, yêu cầu khởi động dịch vụ, bằng cách gọi ***startService()***. Nếu thực thi phương pháp này, trách nhiệm của là ngừng dịch vụ khi nó hoàn thành công việc, bằng cách gọi phương thức **stopSelf()** hoặc ***stopService()***.
onBind()	Hệ thống gọi phương thức này khi thành phần khác muốn liên kết với các dịch vụ bằng cách gọi ***bindService()***. Nếu thi hành phương pháp này, phải cung cấp một giao diện (Giao diện ứng dụng) mà khách hàng sử dụng để giao tiếp với các

Callback	Description
	dịch vụ, bằng cách trả lại một đối tượng ***IBinder***. Phải luôn luôn thi hành phương thức này, nhưng nếu không muốn cho phép ràng buộc, có thể trả về null.
onUnbind()	Hệ thống gọi phương thức này khi tất cả các clients đã bị ngắt kết nối từ một giao diện cụ thể được công bố bởi các dịch vụ.
onRebind()	Hệ thống gọi phương thức này khi khách hàng mới đã kết nối với dịch vụ, sau khi trước đó đã được thông báo rằng tất cả đã bị ngắt kết nối trong ***onUnbind(Intent)***.
onCreate()	Hệ thống gọi phương thức này khi dịch vụ được tạo ra sử dụng đầu tiên ***onStartCommand()*** hoặc ***onBind()***. Gọi một lần tại thời điểm thiết lập.
onDestroy()	Hệ thống gọi phương thức này khi dịch vụ không còn được sử dụng và đang bị hủy (destroy). Dịch vụ của nên thi hành điều này để dọn dẹp các dữ liệu rác...

2.9. Animation trong android

2.9.1. Hiệu ứng cơ bản

– Fade in: Là hiệu ứng làm mờ đối tượng.

– Fade out: Là hiệu ứng làm mờ dần đối tượng.

– Blink: Là hiệu ứng làm nhấp nháy đối tượng.

– Zoom in: Là hiệu ứng phóng to đối tượng.

– Zoom out: Là hiệu ứng thu nhỏ đối tượng.

– Rotate: Là hiệu ứng xoay đối tượng.

– Move: Là hiệu ứng dịch chuyển đối tượng.

– Side up: Là hiệu ứng trượt đối tượng lên trên.

– Side down: Là hiệu ứng trượt đối tượng xuống dưới.

– Sequential animation: Lặp lại một trong các hiệu ứng trên.

– Together animition: Xảy ra đồng thời hai hoặc nhiều trong các hiệu ứng trên.

2.9.2. Cách sử dụng

Bước 1: Tạo tập tin xml định nghĩa animation

(Tập tin này được đặt trong thư mục **anim** dưới thư mục **res** (res ⇒ anim ⇒ xml))

Bước 2: Nạp animation

Tại activity tạo một đối tượng của lớp Animation và nạp tập tin xml sử dụng phương thức **loadAnimation()** của lớp **AnimationUtils**

Bước 3: Thiết lập sự kiện (Tuỳ chọn)

Nếu muốn xử lý các sự kiện như start, end và repeat, phải cài đặt giao diện AnimationListener cho activity của. Lưu ý bước này là tuỳ chọn, có thể bỏ qua nếu không cần thiết.

Bước 4: Bắt đầu animation

Chúng ta có thể bắt đầu một animation bất cứ khi nào mình muốn bằng cách gọi phương thức **startAnimation()** trên bất kỳ thành phần UI nào mà chúng ta muốn thiết lập animation.

2.10. Hệ quản trị cơ sở dữu liệu SQLite

2.10.1. Khái niệm

SQLite là một cơ sở dữ liệu SQL mã nguồn mở, nó lưu trữ dữ liệu vào một tập tin văn bản trên một thiết bị. Nó mặc định đã được tích hợp trên thiết bị Android. Để truy cập dữ liệu này, không cần phải thiết lập bất kỳ loại kết nối nào cho nó như JDBC, ODBC,... SQLite được Richard Hipp viết dưới dạng thư viện bằng ngôn ngữ C [1][2].

2.10.2. Ưu điểm

- Tin cậy: các hoạt động transaction (chuyển giao) nội trong cơ sở dữ liệu được thực hiện trọn vẹn, không gây lỗi khi xảy ra sự cố phần cứng
- Tuân theo chuẩn SQL92 (chỉ có một vài đặc điểm không hỗ trợ)
- Không cần cài đặt cấu hình
- Kích thước chương trình gọn nhẹ, với cấu hình đầy đủ chỉ không đầy 300 kB
- Thực hiện các thao tác đơn giản nhanh hơn các hệ thống cơ sở dữ liệu khách/chủ khác
- Không cần phần mềm phụ trợ

- Phần mềm tự do với mã nguồn mở, được chú thích rõ ràng

2.10.3. Sử dụng SQLite trong Android

– Bước 1: Tạo một đối tượng với các phương thức khởi tạo, getter, setter

– Bước 2: Tạo SQLite Database Handler class

Tạo class để xử lý các thao tác CRUD(create, read, update, delete) đối với database.

1. Đầu tiên, tạo một Android project (File => New Android Project)
2. Tạo một class `DatabaseHandler.java` (src/package => New => Class)
3. Class `DatabaseHandler` sẽ kế thừa class `SQLiteOpenHelper` (Đây là một class mà Android cho phép xử lý các thao tác đối với database của SQLite, vì vậy có thể tạo một class khác thừa kế nó và tùy chỉnh việc điều khiển database theo ý mình):
4. Sau khi kế thừa từ class `SQLiteOpenHelper`, việc chúng ta cần làm tiếp theo đó là override lại 2 phương thức `onCreate()` và `onUpgrade()`

- `onCreate()`: Đây là nơi để chúng ta viết những câu lệnh tạo bảng. Nó được gọi khi database đã được tạo.
- `onUpgrade()`: Nó được gọi khi database được nâng cấp, ví dụ như chỉnh sửa cấu trúc các bảng, thêm những thay đổi cho database,..

– Bước 3: Truy vấn

Phương thức `getStudent()` sẽ đọc một record student trong bảng, nhận student id là tham số.

Phương thức `getAllStudents()` sẽ trả về tất cả student có trong bảng dưới dạng một array list của Student.

Thêm một bản ghi:

Tạo một phương thức là `addObj()`nhận một object như là một tham số. `ContentValues` được sử dụng để lưu các giá trị tương ứng với các trường trong bảng.

Cập nhật dữ liệu trong bảng:

```
public int update(String table, ContentValues values, String
whereClause, String[] whereArgs)
```

- Đối số 1: tên bảng

- Đối số 2: đối tượng muốn chỉnh sửa (với giá trị mới)
- Đối số 3: tập các điều kiện lọc (dùng dấu ? để tạo điều kiện lọc)
- Đối số 4: tập các giá trị ủa điều kiện lọc (lấy theo đúng thứ tự)

Xóa một bản ghi:

```
public int delete(String table, String whereClause, String[]
whereArgs)
```

- Đối số 1: tên bảng
- Đối số 2: tập điều kiện lọc
- Đối số 3: tập các giá trị của điều kiện lọc

CHƯƠNG 3: TỔNG QUAN VỀ WEBSERVICE

3.1. Hosting

3.1.1. Giới thiệu

Hosting là không gian lưu trữ trên máy chủ có cài đặt các dịch vụ Internet như world wide web (www), truyền file(FTP), Mail…, có thể chứa nội dung trang web hay dữ liệu trên không gian đó. Hosting đồng thời cũng là nơi diễn ra tất cả các hoạt động giao dịch, trao đổi thông tin giữa website với người sử dụng Internet và hỗ trợ các phần mềm Internet hoạt động. DN có thể chọn thuê web hosting của nhà cung cấp dịch vụ (ISP) có dung lượng phù hợp với dung lượng website. Với bất kỳ hình thức nào (tự trang bị máy chủ hay thuê máy chủ) thì DN cũng nên có các hiểu biết cần thiết về Hosting và máy chủ Web.

Lý do phải thuê Web Hosting để chứa nội dung trang web, dịch vụ mail, ftp, vì những máy tính đó luôn có một địa chỉ cố định khi kết nối vào Internet (đó là địa chỉ IP), còn như nếu truy cập vào internet như thông thường hiện nay thông qua các IPS (Internet Service Provider - Nhà cung cấp dịch vụ Internet) thì địa chỉ IP trên máy luôn bị thay đổi, do đó dữ liệu trên máy của không thể truy cập được từ những máy khác trên Internet.

3.1.2. Yêu cầu và tính năng cần thiết của hosting

Đầu tiên phải nói đến về vấn đề tốc độ. Máy chủ chạy dịch vụ Web phải có cấu hình đủ lớn để đảm bảo xử lý thông suốt, phục vụ cho số lượng lớn người truy cập. Phải có đường truyền kết nối tốc độ cao để đảm bảo không bị nghẽn mạch dữ liệu.

Máy chủ phải được người quản trị hệ thống chăm sóc, cập nhật, bảo dưỡng thường xuyên nhằm tránh các rủi ro về mặt kỹ thuật cũng như bảo mật.

Web Hosting phải có một dung lượng đủ lớn (tính theo MBytes) để lưu giữ được đầy đủ các thông tin, dữ liệu, hình ảnh,... của Website

Phải có bandwidth (băng thông) đủ lớn để phục vụ các hoạt động giao dịch, trao đổi thông tin của Website

Phải hỗ trợ truy xuất máy chủ bằng giao thức FTP để cập nhật thông tin.

Hỗ trợ các các ngôn ngữ lập trình cũng như cơ sở dữ liệu để thực thi các phần mềm trên Internet hoặc các công cụ viết sẵn để phục vụ các hoạt động giao dịch trên Website như gửi mail, upload qua trang Web, quản lý sản phẩm, tin tức...

Hỗ trợ đầy đủ các dịch vụ E-mail như POP3 E-mail, E-mail Forwarding, DNS...

Có giao diện quản lý Web Hosting để dễ dàng quản lý website, các tài khoản FTP, Email...

Không bị chèn các banner quảng cáo của nhà cung cấp.

3.1.3. Dung lượng hosting

Dung lượng của web hosting là khoảng không gian được phép lưu trữ dữ liệu của mình trên ổ cứng của máy chủ. Như đã nói ở trên, thuê một web hosting cũng giống như thuê văn phòng trong một nhà cao ốc. Vậy ở đây, dung lượng của web hosting cũng giống như diện tích văn phòng của.

3.1.4. Lưu lượng hosting

Lưu lượng của web hosting là lượng dữ liệu (tính bằng MB) trao đổi giữa website của với người sử dụng trong một tháng. Ví dụ nếu tải lên website của mình một tệp tài liệu có kích thước là 1MB và có 100 khách hàng tải tệp tài liệu đó về thì đã tiêu tốn tổng cộng 101MB lưu lượng.

3.2. Cơ sở dữ liệu trên web

3.2.1. Khái niệm

Cơ sở dữ liệu[1] hiểu một cách cơ bản là một hệ thống lưu trữ các thông tin một cách có cấu trúc giúp tìm kiếm thông tin hay hiển thị thông tin một cách thông minh giúp người dung khai thác sức mạnh của thông tin hay bằng cách sử dụng các ứng dụng công nghệ trong cơ sở dữ liệu được lưu trữ tạo ra. Cơ sở dữ liệu gồm các các table, row cột là những thuộc tính phần tử trong một cơ sở dữ liệu tùy theo hệ quản trị cơ sở dữ liệu khác nhau mà kiểu dữ liệu sẽ có giá trị khác nhau.

Hệ cơ sở dữ liệu là một ngôn ngữ được tạo ra dùng để xử lý cơ sở dữ liệu đồng thời giúp làm việc và lưu trữ cơ sở dữ liệu một cách khoa học nhất. Khi tìm hiểu hệ quản trị cơ sở dữ liệu sẽ phải học về các phần cấu trúc dữ liệu cú pháp thêm xóa sửa, import, export dữ liệu trên .

3.2.2. Đặc điểm của database

Đặc điểm của database chính là có thể truy xuất thông tin, dữ liệu theo nhiều cách khác nhau, thông tin từ cơ sở dữ liệu được đảm bảo nhất và toàn vẹn dữ liệu, không hề có sự trùng lặp thông tin, nếu có thì tỷ lệ rất thấp. Một cơ sở dữ liệu database có thể có nhiều người sử dụng cùng một lúc.

---

[1] Database

Như vậy từ đặc điểm của cơ sở dữ liệu database ta có thể thấy nó có nhiều ưu điểm và hạn chế nhất định:

Về mặt ưu điểm của cơ sở dữ liệu database đó chính là nhờ vào việc thông tin lưu trữ không bị trùng lặp giúp đảm bảo tính thông nhất cũng như toàn vẹn của dữ liệu. Nhờ việc không bị trùng lặp giúp giảm thời gian xử lý dữ liệu, cũng như tránh khỏi những sai sót trong quá trình kiểm tra cơ sở dữ liệu database. Ngoài ra, nhờ vào việc có thể truy xuất từ các cách khác nhau nên nhiều người có thể sử dụng cơ sở dữ liệu cùng một lúc mà không phải qua các khâu rườm rà phức tạp. Từ đó tạo điều kiện thuận lợi cho việc sử dụng, quản lý, truy cập dữ liệu,… Bên cạnh đó cơ sở dữ liệu database cũng có thể được lưu trữ dưới nhiều dạng khác nhau như ổ cứng, usd hay đĩa CD.

Tuy vậy, cơ sở dữ liệu database cũng có những hạn chế nhất định,vì nhiều người chung quyền sử dụng, khai thác cơ sở dữ liệu, nên chủ quyền của người người sử dụng khác nhau có thể bị xâm phạm, ngoài ra vấn đề bảo mật cũng thực sự đáng quan tâm khi mà ai cũng có thể xâm nhập vào cơ sở dữ liệu database, từ đó dẫn đến nguy cơ bị tấn công, đánh cắp dữ liệu. Điều này thường gặp nhiều nhất ở các công ty cung cấp hosting dùng chung, sau khi lập trình web, thiết kế website xong thì người ta thường hay đặt website của mình lên các hosting này vì giá thành rẻ, nhưng đó lại là điểm yếu, chỉ cần một cơ sở dữ liệu database của một website bị tấn công sẽ làm liên lụy đến những trang khác. Đó còn là chưa kể đến những ảnh hưởng từ việc thiết bị lưu trữ cơ sở dữ liệu bị hư hỏng, làm mất toàn bộ dữ liệu của người dùng. Do đó khi sử dụng cơ sở dữ liệu database thì việc cần làm đó là phải backup dữ liệu một cách thường xuyên, đừng trông chờ vào các nhà cung cấp như công ty hosting. Đây cũng là lời khuyên của những chuyên gia lập trình database.

3.2.3. Các loại Database thường dùng

a. Database bán cấu trúc

Cơ sở dữ liệu database bán cấu trúc có thể lưu trữ được nhiều loại dữ liệu khác nhau, nó được lưu lại dưới định dạng XML, các thông tin mô tả dữ liệu, đối tượng được trình bày trong các thẻ tag. Các chuyên gia về lập trình database dự đoán database bán cấu trúc sẽ là hướng đi mới trong nghiên cứu và ứng dụng về cơ sở dữ liệu.

b. Cơ sở dữ liệu dạng file

Cơ sở dữ liệu databse dạng file thường gặp nhất đó chính là *.mdb Foxpro, ngoài ra cũng có một số định dạng file khác có thể kể đến như dạng file text, dạng file ascii, dạng file *.dbf…

c. Database hướng đối tượng

Một dạng cơ sở dữ liệu database khác đó chính là cơ sở dữ liệu database hướng đối tượng. Những hệ quản trị cơ sở dữ liệu hỗ trợ cơ sở dữ liệu database hướng đối tượng đó chính là hệ quản trị cơ sở dữ liệu MS SQL Server, Postgres, Oracle. Về cơ sở dữ liệu hướng đối tượng này thì nó cũng là một dạng bảng dữ liệu thuần, nhưng trong đó có bổ sung thêm các trường hướng đối tượng khác như hành vi đối tượng. Phân cấp của nó cũng rất rõ ràng, đối tượng chính được thể hiện bằng dòng dữ liệu, tập hợp các đối tượng trong một bảng và ta gọi đây là lớp dữ liệu.

d. Database quan hệ

Cơ sở dữ liệu database cuối cùng mà datadesignsb muốn chia sẻ đến các đó chính là database quan hệ. Có nhiều thực thể khác nhau (dữ liệu khác nhau) được lưu trữ trong bảng dữ liệu, và vừa chúng có mối liên hệ với nhau. Từ đó người ta gọi nó là cơ sở dữ liệu database quan hệ. Một vài hệ quản trị cơ sở dữ liệu có hỗ trợ cơ sở dữ liệu database quan hệ như Oracle, MS SQL Server, MySQL,… Đây đều là những hệ quản trị cơ sở dữ liệu rất nổi tiếng.

3.2.4. Các hệ quản trị cơ sở dữ liệu phổ biến

a. Hệ quản trị CSDL Mysql

Bất kỳ một lập trình viên đang học thiết kế website thì hệ quản tri CSDL[1] phổ biến nhất là mysql , Mysql là một hệ quản trị CSDL dành cho ngôn ngữ lập trình web php được phát triển bởi Swedish là mã nguồn mở. hệ quản trị MYsql có thể chạy trên nhiều nền tảng dành cho các hệ điều hành Microsoft windows, linux, unix, mac osx, tuy nhiên Mysql là mã nguồn mở được dùng miễn phí có thể dùng bản thương mại phải trả phí. Mysql có thể chạy nhanh nếu được tối ưu trên server chạy đa luồng.

b. Hệ quản trị CSDL SQLite

Đây là một hệ quản trị mới phát triển về sau không cần phải lưu dạng cấu trúc dùng các phần mềm để lưu trữ mà đươc lưu trữ trực tiếp trên ổ đĩa hay điện thoại của

[1] Cơ sở dữ liệu

người dùng.Nếu đang phát triển các ứng dụng mobile di động thì đây là một lựa chọn tuyệt vời cho.

c. Hệ quản trị CSDL MS Access

Đây là một hệ quản trị CSDL được phát triển từ hang Microsoft dưới dạng phân mềm.Microsoft Access là phần mềm quản lý cơ sở dữ liệu phổ biến cho các công ty vừa và nhỏ không cần mở rộng database chạy nhanh và ổn định với số lượng data không hiều.MS access tích hợp với MS office package có giao diện dễ sử dụng.

d. Hệ quản trị CSDL MS SQL Server

Nếu như đang có định hướng phát triển phần mêm bán hàng, thiết kế website bán hàng hay quản lý trên nền tảng C# thì chắc chắn phải học qua MS SQL Server. Đây là hệ CSDL được phát triển bởi hang phần mềm Microsoft Inc.

3.3. Dịch vụ web

3.3.1. Khái niệm

Là những thành phần dùng để chuyển đổi một ứng dụng thông thường sang một ứng dụng web. Đồng thời nó cũng xuất bản các chức năng của mình để mọi người dùng internet trên thế giới đều có thể sử dụng thông qua nền tảng web. Web[1]service truyền thông bằng cách sử dụng các giao thức mở, tài nguyên phần mềm có thể xác định bằng địa chỉ URL, thực hiện các chức năng và đưa ra các thông tin người dùng yêu cầu, các ứng dụng độ lập và tự mô tả chính nó. Nó bao gồm các modun độc lập cho hoạt động của khác hàng và doanh nghiệp và bản thân nó được thực thi trên server. Nền tảng cơ bản của WS là XML + HTTP. Bất cứ một ứng dụng nào cũng đều có thể có một thành phần WS. WS có thể được tạo ra bằng bất kỳ một ngôn ngữ lập trình nào.

3.3.2. Đặc điểm

Cho phép client và server tương tác ngay cả trong môi trường khác nhau. (Ví dụ server chạy Linux, client chạy Windows).

Phần lớn được xây dựng dựa trên mã nguồn mở và phát triển các chuẩn đã được công nhận. (Ví dụ XML).

Nó có thể triển khai bởi 1 phần mềm ứng dụng phía server (Ví dụ : PHP, Oracle Application server, Microsoft .NET)

---

[1] Dịch vụ web

3.3.3. Cách thức hoạt động

Nền tảng cơ bản của WS là XML và HTTP.

XML cung cấp một ngôn ngữ mà có thể được sử dụng giữa ngôn ngữ lập trình và các nền tảng khác. Đồng thời, nó còn có thể được dùng để mô tả những thông điệp và chức năng phức tạp. Do web service là sự kết hợp của nhiều thành phần khác nhau, do đó web serices sử dụng các tính năng và đặc trưng ủa các thành phần này để giao tiếp với nhau. Vì vậy XML là một công cụ chính yếu để giải quyết vấn đề này. Web service tận dụng khả năng giải quyết vấn đề của các ứng dụng lớn trên các hệ điều hành khác nhau cho chúng giao tiếp với nhau. Yêu cầu này được đáp ứng với lập trình Java, một ngôn ngữ việt một lần sử dụng mọi nơi là một lựa chọn thích hợp cho phát triển webservice.

Giao thức HTTP là giao thức được sử dụng nhiều nhất trong các giao thúc trên internet.

Nền tảng của WS bao gồm các thành phần:

- SOAP (Sinple Object Access Protocol): Giao thức truy cập đối tượng đơn giản.
- UDDI (Universal Description, Discovered and Integrated).
- WSDL (Web Services Desription Language): Ngôn ngữ mô tả WS.

3.3.4. Nền tảng Webservice

– Webservice bao gồm 3 thành phần cơ bản, đó là: SOAP, WSDL, và UDDI.

a. Giao thức SOAP

– SOAP[1] là một giao thức dựa trên nền XML cho phéo ứng dụng trao đổi các thông tin thông qua HTTP. Nói cụ thể hơn thì SOAP là một giao thức dùng để truy cập các dịch vụ web.

– SOAP là viết tắt của Simple Object Access Protocol.

– SOAP là một giao thức truyền thông.

– SOAP là một định dạng dùng để gửi đi các thông điệp.

– SOAP được thiết kế để giao tiếp thông qua internet.

– SOAP là một nền tảng XML.

– SOAP rất đơn giản và có thể mở rộng.

[1] viết tắt từ tiếng Anh *Simple Object Access Protocol*

– SOAP cho phép chúng ta vượt qua bức tường lửa (firewall)

– SOAP là một tiêu chuẩn W3C.

– SOAP là giao thức sử dụng XML để định nghĩa dữ liệu dạng thuần văn bản (plaint text) thông qua HTTP. SOAP là cách mà web service sử dụng để truyền tải dữ liệu. Vì dựa trên XML nên SOAP là một giao thức không phụ thuộc platform cũng như bất kì ngôn ngữ lập trình nào.

– Một thông điệp SOAP được chia thành hai phần là header và body. Phần header chỉ ra địa chỉ web service, host, Content-Type, Content-Length tương tự như một thông điệp HTTP.

b. WSDL

WSDL là một ngôn ngữ dựa trên nền XML dùng để định vị và mô tả WS.

- WSDL là viết tắt của Web services Description language.
- WSDL dựa trên nền tảng XML.
- WSDL được dùng để mô tả WS.
- WSDL được dùng để định vị WS.
- WSDL là một tiêu chuẩn W3C.

WSDL định nghĩa cách mô tả web service theo cú pháp tổng quát XML, bao gồm các thông tin:

- Tên service
- Giao thức và kiểu mã hóa sẽ được sử dụng khi gọi các hàm của web service.
- Loại thông tin: những thao tác, những tham số, những kiểu dữ liệu gồm có giao diện của web service, công với tên cho giao diện này.
- Phần giao diện mô tả giao diện và giao thức kết nối, phần thi hành mô tả thông tin để truy xuất service.

c. UDDI

UDDI là một thư mục dịch vụ, nơi mà chúng ta có thể đăng ký và tìm kiếm các dịch vụ web.

- UDDI là viết tắt của Universal Description, Discovery and integration.
- UDDI là một thư mục dùng để lưu trữ thông tin về các dịch vụ web.

- UDDI là một thư mục các giao diện dịch vụ web được mô tả bởi WSDL.
- UDDI giao tiếp thông qua SOAP.
- UDDI được xây dựng để góp phần vào nền tảng Microsoft .NET.

Để có thể sử dụng các dịch vụ, trước tiên client phải tìm dịch vụ, ghi nhận thông tin về cách sử dụng dịch vụ và biết được đối tượng cung cấp dịch vụ. UDDI định nghĩa một số thành phần cho biết trước các thông tin này để cho phép các client truy tìm và nhận lại những thông tin yêu cầu sử dụng web services.

Cấu trúc UDDI gồm các thành phần:

- White pages: chứ thông tin liên hệ và các định dạng chính yếu của web services (tên giao dịch, địa chỉ...) Những thông tin này cho phép các đối tượng khác xác định được service.
- Yellow pages: chứa thông tin mô tả web service theo những chủng loại khác nhau. Những thông tin này cho phép các đối tượng thấy web service theo từng chủng loại của nó.
- Green pages: chứa thông tin kỹ thuật mô tả các hành vi và các chức năng của web service để tìm kiếm.

3.4. RESTful Web Service

3.4.1. Khái niệm

RESTful web service là các dịch vụ web được xây dựng dựa trên cấu trúc REST (REpresentational State Transfer). Tức là nó giống như một kiến trúc, nguyên tắc cần tuân theo để thiết kế, xây dựng một web service.

Trong kiến trúc REST mọi thứ đều được coi là tài nguyên, chúng có thể là: tệp văn bản, ảnh, trang html, video, hoặc dữ liệu động… REST server cung cấp quyền truy cập vào các tài nguyên, REST client truy cập và thay đổi các tài nguyên đó. Ở đây các tài nguyên được định danh dựa vào URI, REST sử dụng một vài đại diện để biểu diễn các tài nguyên như văn bản, JSON, XML.

Kiến trúc REST bao gồm bốn nguyên tắc cơ bản: GET, POST, PUT, DELETE

3.4.2. Cách sử dụng phương thức HTTP

Như chúng ta đã biết, HTTP cung cấp các phương thức dùng để lấy dữ liệu, trèn dữ liệu, cập nhập dữ liệu hoặc xóa dữ liệu. Và khi sử dụng những phương thức này,

chúng ta cần xác định rõ ràng mục đích sử dụng mỗi khi gọi tới một phương thức. Và gợi ý cụ thể cho các phương thức như sau:

- GET: dùng để truy xuất một tài nguyên (phương thức này gần như là phổ biến nhất)
- POST: dùng để tạo một tài nguyên trên máy chủ (*VD như đăng kí tài khoản, sau khi điền form thông tin, dùng phương thức POST để gửi dữ liệu lên máy chủ*)
- PUT: dùng để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó.
- DELETE: dùng để huỷ bỏ hoặc xoá một tài nguyên.

3.4.3. Phi trạng thái (stateless)

Phi trạng thái có nghĩa là máy chủ sẽ không lưu giữ thông tin của client mà nó giao tiếp, thông tin hoặc được giữ trên client hoặc được chuyển thành trạng thái của tài nguyên. Mỗi request lên server thì client phải đóng gói thông tin đầy đủ để thằng server hiểu được.

Điều này đem lại hai lợi ích:

- Giúp tách biệt client ra khỏi sự thay đổi của server.
- Giúp hệ thống của dễ phát triển,bảo trì, mở rộng vì không cần tốn công CRUD trạng thái của client.

3.4.4. Hiển thị cấu trúc thư mục như URI

REST đưa ra một cấu trúc để người dùng có thể truy cập vào tài nguyên của nó thông qua các URL

Các địa chỉ REST service cần phải thật trực quan đến mức đơn giản, có thể dự đoán, và dễ hiểu. Ví dụ: chỉ cần nhìn vào thanh địa chỉ URL ta có thể đoán rằng nó đang trỏ tới cái gì và cung cấp tài nguyên gì.

Và để tạo ra đáp ứng yêu cầu trên thì ta nên định nghĩa URI có câu trúc giống thư mục. Loại URI này có phân cấp, có gốc là một đường dẫn đơn, các nhánh từ gốc là các đường dẫn phụ dẫn đến các các vùng service chính.

Cấu trúc này giúp cho nhà phát triển dễ dàng trong việc cài đặt service của mình hướng vào một loại tài nguyên cụ thể nào đó.

Ngoài ra còn có một số quy tắc bổ sung trong khi nói về cấu trúc địa chỉ của RESTful Web service:

- Giấu các đuôi tài liệu mở rộng của bản gốc trong máy chủ (.jsp, .php, .asp), nếu có, vì vậy có thể giấu một số thứ mà không cần thay đổi địa chỉ Urls.
- Để mọi thứ là chữ thường.
- Thay thế các khoảng trống bằng gạch chân hoặc hoặc gạch nối (một trong hai loại).
- Thay vì sử dụng mã (404 Not Found) khi yêu cầu địa chỉ cho một phần đường dẫn, luôn luôn cung cấp một trang mặc định hoặc tài nguyên như một phản hồi.

3.4.5. Định dạng dữ liệu (html, json, text, xml…)

Khi Client gửi một yêu cầu tới web service, nó thường được truyền tải dưới dạng dữ liệu mà máy tính hiểu được (XML hoặc JSON) và thông thường nhận về với hình thức tương tự từ máy chủ.

Tuy nhiên Client cũng có thể chỉ định kiểu dữ liệu nhận về mà nó mong muốn (JSON, hoặc XML,..), các chỉ định này được gọi là các kiểu MIME, nó được gửi kèm trên phần HEADER của request.

CHƯƠNG 4: ỨNG DỤNG THÔNG BÁO SẢN PHẨM

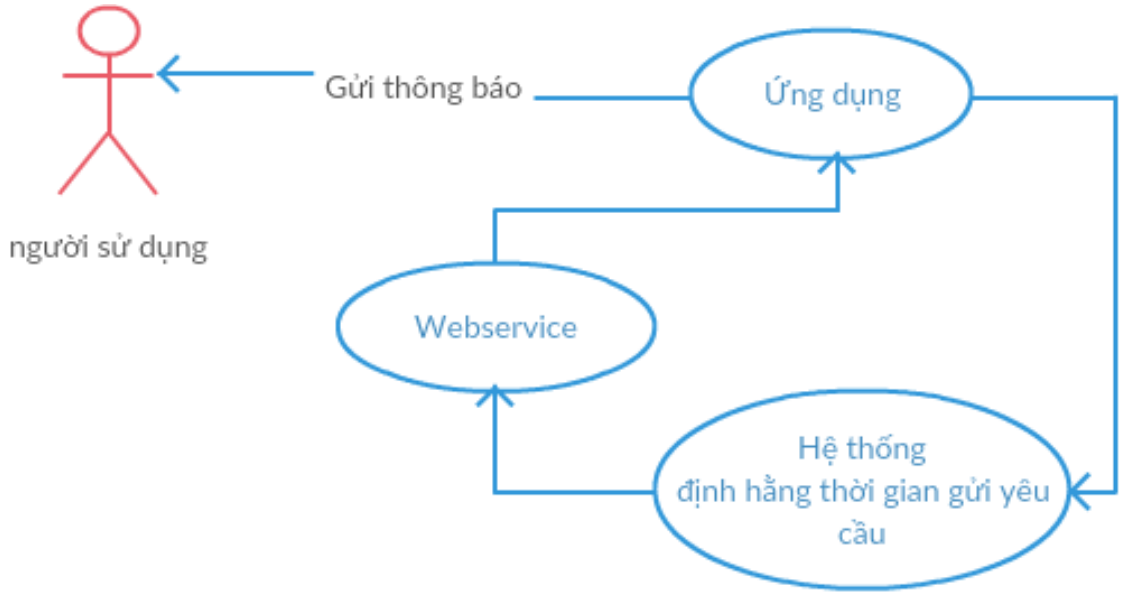
4.1. Phân tích ứng dụng

4.1.1. Phái biểu bài toán

Nhận thông tin mới cần quan tâm từ hosting theo thời gian thực là một nhu cầu cần thiết trong thực tế. Một giải pháp thương dùng là gửi tin nhắn từ hosting đến các thiết bị di dộng. khi có dữ liệu mới. Tuy nhiên, hiện nay các web. Đề tài nghiên cứu một giải pháp khác là xây dựng một ứng dụng cho thiết bị di động Android, tự động kiểm tra và lấy dữ liệu mới từ hosting về thiết bị theo một định hằng thời gian. Cũng xuất phát từ mong muốn của những doanh nghiệp thường xuyên có những cập nhật về thông tin sản phẩm, họ mong muốn thông báo thông tin khuyến mãi, tình trạng sản phẩm, những sản phẩm mới họ sắp cung ứng ra thị trường một cách nhanh nhất tới nhân viên và những khách hàng của họ.

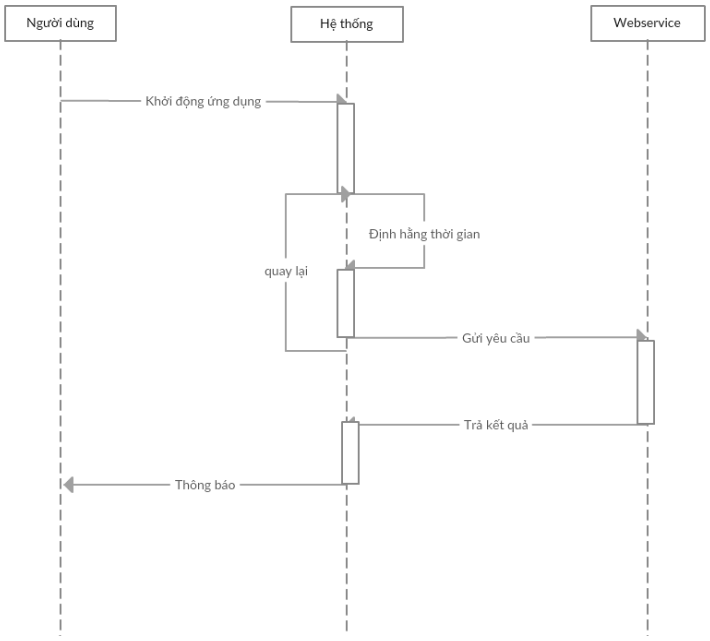
Xuất phát từ những nhu cầu trên, vậy nên một ứng dụng thông báo sản phẩm là một điều hết sức cần thiết.

4.1.2. Biểu đồ quy trình nghiệp vụ



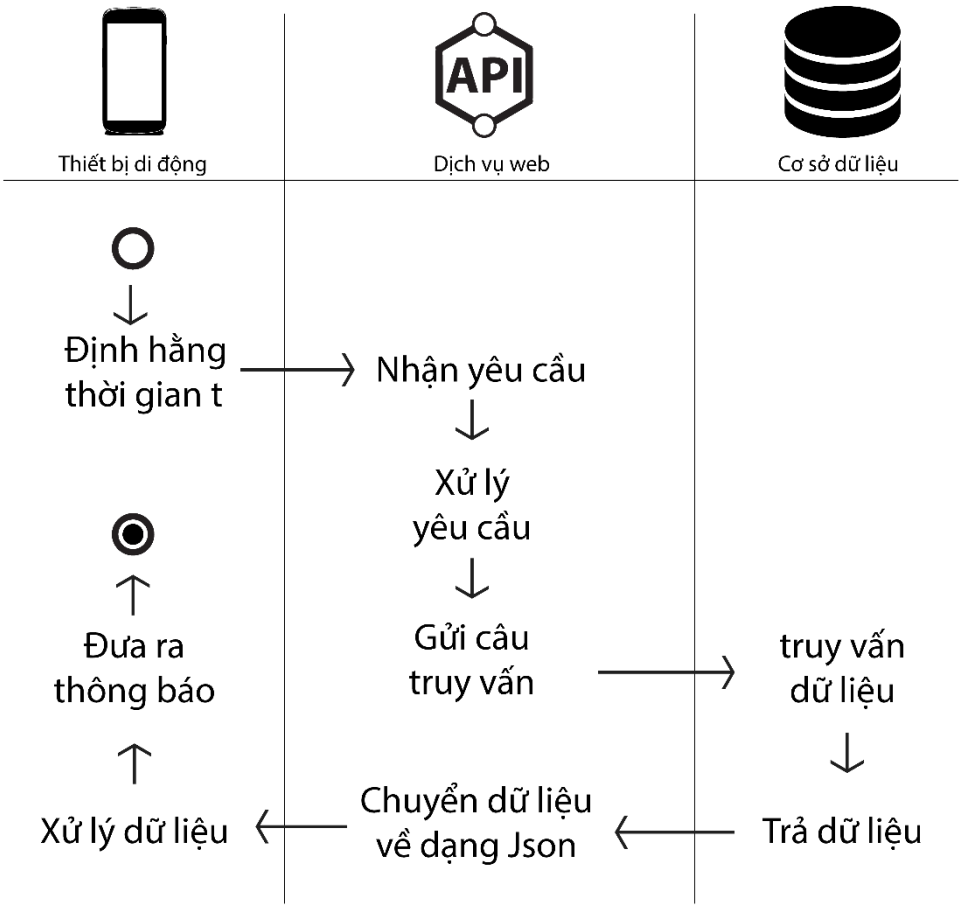
Hình 4.1.2.1 Quy trình nghiệp vụ nhận thông báo

4.1.3. Biểu đồ tuần tự



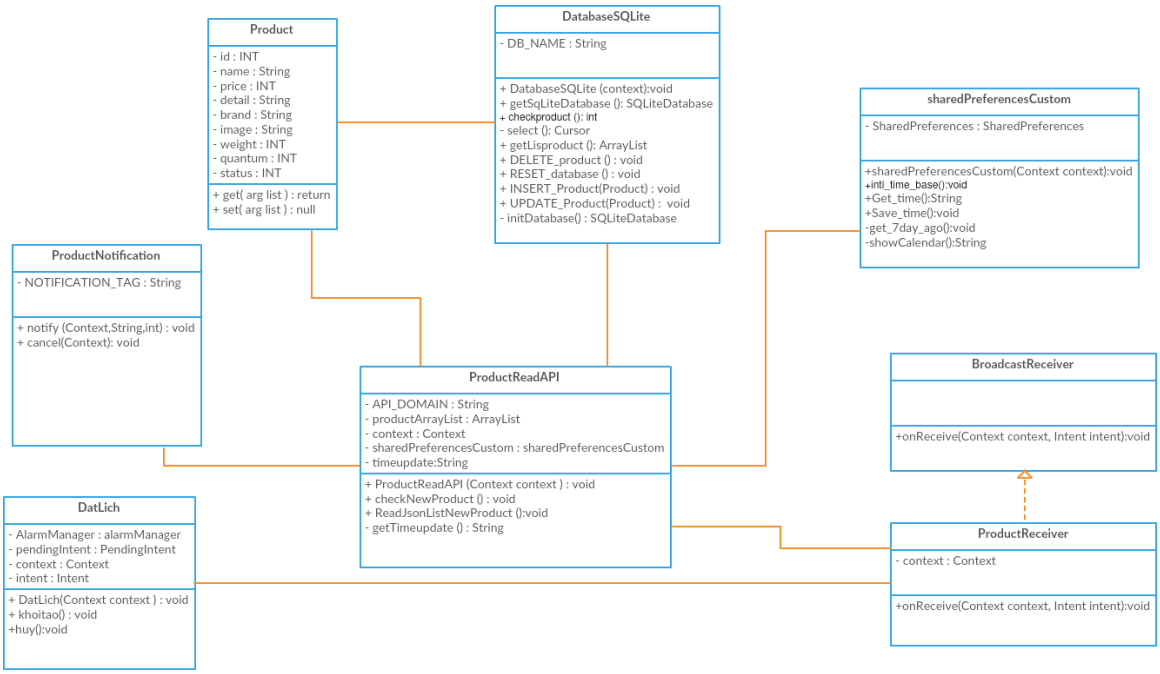
Hình 4.1.3.1 Quá trình thực thi nhận thông báo

4.1.4. Biểu đồ hoạt động

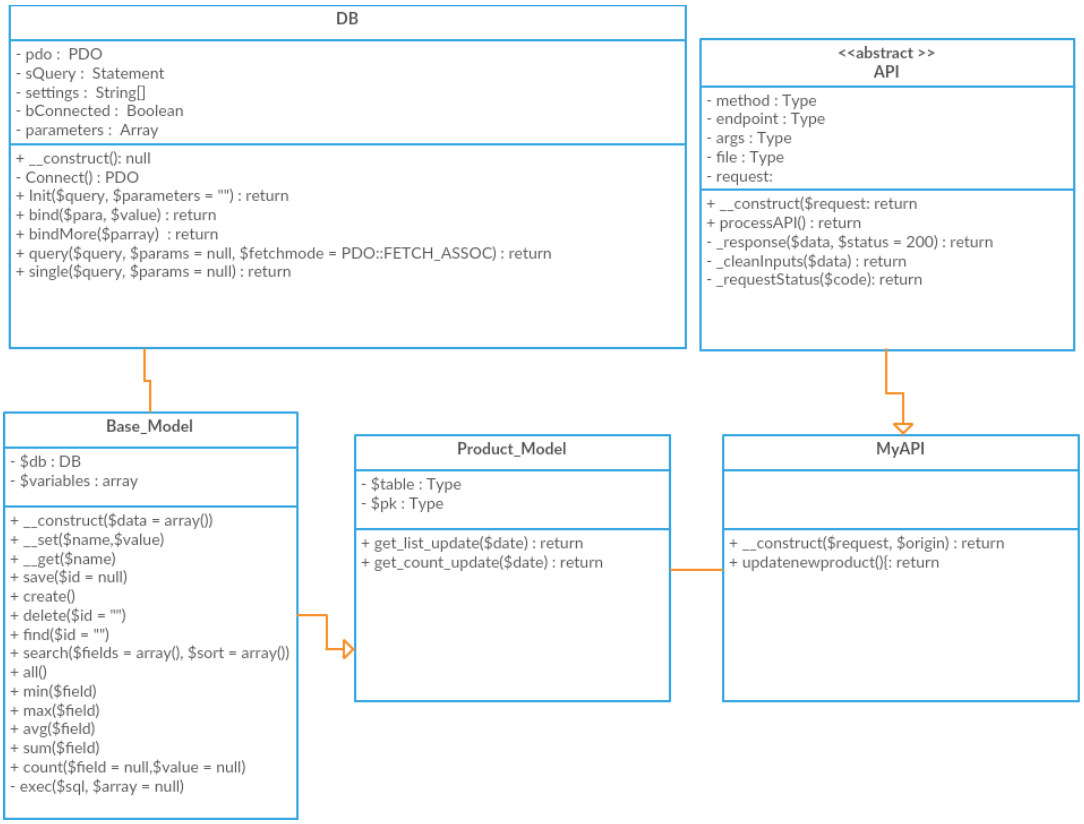


Hình 4.1.4.1. Mô hình hoạt động ứng dụng định hằng thời gian thực

4.1.5. Biểu đồ lớp



Hình 4.1.5.1 Các lớp trong ứng dụng Android

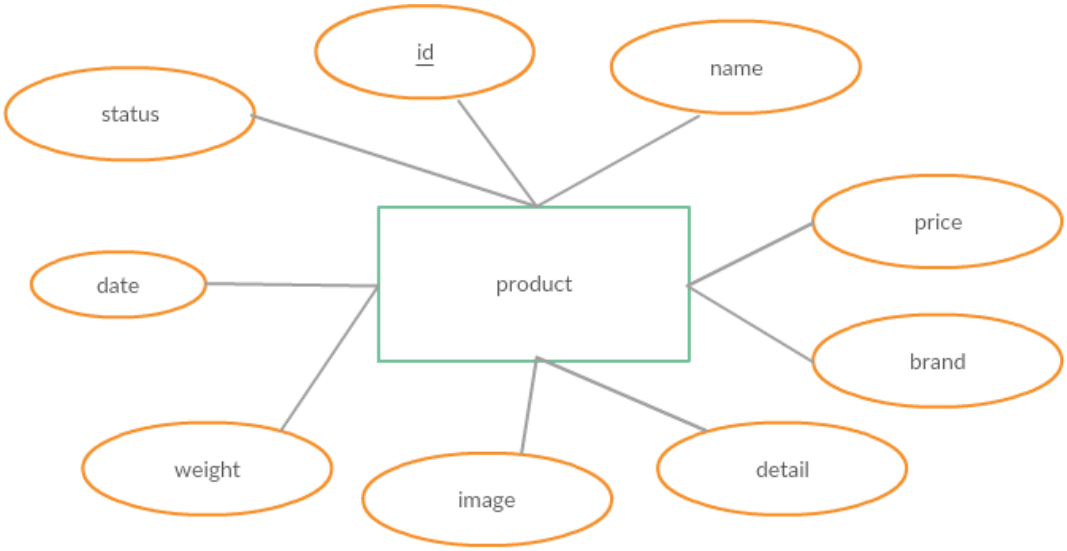


Hình 4.1.5.2 Các lớp xây dựng webservice

4.2. Xây dựng cơ sở dữ liệu

Đối tượng: product

Thuộc tính: id, name, brand, detail, image, weight, quantum, date, status



Hình 4.2.1. Biểu đồ quan hệ

#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm
1	**id**	int(11)			Không	*Không*		AUTO_INCREMENT
2	**name**	varchar(500)	utf8mb4_unicode_ci		Không	*Không*		
3	**price**	bigint(20)			Không	0		
4	**detail**	longtext	utf8mb4_unicode_ci		Có	*NULL*		
5	**brand**	varchar(255)	utf8mb4_unicode_ci		Không	đang cập nhật		
6	**image**	varchar(500)	utf8mb4_unicode_ci		Không	/no-image.jpg		
7	**weight**	int(11)			Không	0		
8	**quantum**	int(11)			Không	0		
9	**date**	timestamp			Không	CURRENT_TIMESTAMP		
10	**status**	int(11)			Không	1		

Hình 4.2.2. Bảng product trong hệ quản trị CSDL MySQL

4.3. Xây dựng dịch vụ web

– Cấu trúc yêu cầu đến dịch vụ web:

```
http://{domain}/api/{endpoint}/({args..}? {Parameter=…})
```

- {domain}: tên miền.
- {endpoint}: tên phương thức xử lý yêu cầu trong lớp xử lý yêu cầu.
- {args..}: danh sách các biến mỗi biến cách nhau bởi dấu "/"
- Parameter: là các tham số truyền vào theo phương thức GET.

4.3.2. Lớp truy vấn cơ sở dữ liệu

a. Tổng quan về lớp truy vấn CSDL

– Lớp truy vấn này để trực tiếp làm nhiệu vụ với CSDL từ kết nối, đến truy vấn dữ liệu.

```
class DB
{
//Thhộc tính lớp
    private $pdo;
    private $sQuery;
    private $settings;
    private $bConnected = false;
    private $log;
    private $parameters;
//Phương thức của lớp
    public function __construct(){…}
    private function Connect(){…}
    public function CloseConnection(){…}
    private function Init($query, $parameters = ""){…}
    public function bind($para, $value) {…}
    public function bindMore($parray) {…}
    public function query($query, $params = null, $fetchmode =
PDO::FETCH_ASSOC) {…}
    public function single($query, $params = null) {…}
}
```

b. Thuộc tính

```
private $pdo;
private $sQuery;
private $settings;
private $bConnected = false;
private $parameters;
```

$pdo: chứa đối tượng của lớp PDO trong PHP.

$sQuery: câu lênh MySQL tham chiếu.

$settings: mảng chứ cách tham số kết nối MySQL.

$bConnected: biến kiểm tra kết nối.

$parameters: mảng chứa cách tham số truyền vào câu lệnh $sQuery.

c. Phương thức bind($para, $value) và bindMore($parray)

– Hai phương thức này dùng để thêm pham số vào mảng $parameters

```
public function bind($para, $value)
{
    $this->parameters[sizeof($this->parameters)] = [":" .
$para , $value];
}
public function bindMore($parray)
{
    if (empty($this->parameters) && is_array($parray)) {
        $columns = array_keys($parray);
        foreach ($columns as $i => &$column) {
            $this->bind($column, $parray[$column]);
        }
    }
}
```

d. Phương thức connect()

Dùng để kết nối tới hệ quản trị cơ sở dữ liệu MySQL:

```
private function Connect()
{
    $this->settings =
parse_ini_file(PATH_SYSTEM."/settings.ini.php");
    $dsn            = 'mysql:dbname=' . $this-
>settings["dbname"] . ';host=' . $this->settings["host"] . '';
    try {
        $this->pdo = new PDO($dsn, $this->settings["user"],
$this->settings["password"], array(
            PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"
        ));

        $this->pdo->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
        $this->pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES,
false);
        $this->bConnected = true;
    }
    catch (PDOException $e) {
        $this->ExceptionLog($e->getMessage());
        die();
    }
}
```

e. Phương thức Init($query, $parameters = "")

– Khởi tạo câu truy vấn:

- Kết nối CSDL
- Kiểm tra dữ liệu đầu vào
- Thêm tham số vào mảng $parameters

```
private function Init($query, $parameters = "")
{
    if (!$this->bConnected) {
        $this->Connect();
    }
    try {
        $this->sQuery = $this->pdo->prepare($query);
        $this->bindMore($parameters);
        if (!empty($this->parameters)) {
            foreach ($this->parameters as $param => $value) {
                if(is_int($value[1])) {
                    $type = PDO::PARAM_INT;
                } else if(is_bool($value[1])) {
                    $type = PDO::PARAM_BOOL;
                } else if(is_null($value[1])) {
                    $type = PDO::PARAM_NULL;
                } else {
                    $type = PDO::PARAM_STR;
                }
          $this->sQuery->bindValue($value[0],$value[1],$type);
            }
        }
        $this->sQuery->execute();
    }
    catch (PDOException $e) {die();}
    $this->parameters = array();
}
```

f. Phương thức query()

– Thực thi câu lệnh Mysql và trả về kết quả

```
public function query($query, $params = null, $fetchmode =
PDO::FETCH_ASSOC){
    $query = trim(str_replace("\r", " ", $query));
    $this->Init($query, $params);
    $rawStatement = explode(" ", preg_replace("/\s+|\t+|\n+/",
" ",$query));
    $statement = strtolower($rawStatement[0]);
```

```
    if ($statement === 'select' || $statement === 'show') {
        return $this->sQuery->fetchAll($fetchmode);
    } elseif ($statement === 'insert' || $statement ===
'update' || $statement === 'delete') {
        return $this->sQuery->rowCount();
    } else {
        return NULL;
    }
}
```

4.3.3. Lớp nhận yêu cầu

a. Cấu trúc lớp

```
abstract class API
{
    protected $method = '';
    protected $endpoint = '';
    protected $args = array();
    protected $file = null;
    protected $request;
    public function __construct($request) {...}
    public function processAPI() {...}
    private function _response($data, $status = 200) {...}
    private function _cleanInputs($data) {... }
    private function _requestStatus($code) {...}
}
```

b. Thuộc tính

```
protected $method = '';
protected $endpoint = '';
protected $args = array();
protected $file = null;
protected $request;
```

- $method: chứa phương thức truy vấn (POST, GET, PUT, DELETE).
- $endpoint: chứa tên yêu cầu (tên hàm để thực hiệu các yêu cầu).
- $args: mảng chứa các tham số trong yêu cầu.
- $file: chứa tên file được truyền lên.
- $request: yêu cầu.

c. Phương khởi tạo

– Ở phương thức này sẽ đổ dữ liệu vào các thuộc tính: method, endpoint, args

```
public function __construct($request) {
        header("Content-Type: application/json");
```

```
        $this->args = explode('/', rtrim($request, '/'));
        $this->endpoint = array_shift($this->args);
        $this->method = $_SERVER['REQUEST_METHOD'];
        if ($this->method == 'POST' &&
array_key_exists('HTTP_X_HTTP_METHOD', $_SERVER)) {
            if ($_SERVER['HTTP_X_HTTP_METHOD'] == 'DELETE') {
                $this->method = 'DELETE';
            } else if ($_SERVER['HTTP_X_HTTP_METHOD'] ==
'PUT') {
                $this->method = 'PUT';
            } else {
                throw new Exception("Unexpected Header");
            }
        }
    }
```

d. Phương thức _cleanInputs($data)

– Phương thức này trả về mảng dữ liệu của các tham số.

```
private function _cleanInputs($data) {
        $clean_input = Array();
        if (is_array($data)) {
            foreach ($data as $k => $v) {
                $clean_input[$k] = $this->_cleanInputs($v);
            }
        }
        else {
            $clean_input = trim(strip_tags($data));
        }
        return $clean_input;
}
```

e. Phương thức _ requestStatus($code)

Trả về trạng thái server theo theo tham số $code được truyền vào.

```
private function _requestStatus($code) {
        $status = array(
            200 => 'OK',
            404 => 'Not Found',
            405 => 'Method Not Allowed',
            500 => 'Internal Server Error',);
        return ($status[$code]) ? $status[$code] :
$status[500]; }
```

f. Phương thức _ response($data, $status = 200)

– Tham số truyền vào và trạng thái.

– Chuyển đổi dữ liệu thành dạng Json.

– Trả về dữ liệu của yêu cầu và trạng thái server.

```
private function _response($data, $status = 200) {
        header("HTTP/1.1 " . $status . " " . $this-
>_requestStatus($status));
        return json_encode($data, JSON_PRETTY_PRINT);
    }
```

g. Phương thức processAPI()

– Thực thi yêu cầu:

- Đầu tiên sẽ kiểm tra phương xem là loại phương thức gì được yêu cầu gửi tới.
- Nhận các tham số được yêu cầu gửi đến theo loại phương thức.
- Kiểm tra xem endpoint[1] có tồn tại không nếu có thì thự thi phương thức phương thức đó và trả về kết quả.
- Còn không thì trả về lỗi 404[2].

– Mã nguồn phương thức:

```
public function processAPI() {
        switch($this->method) {
            case 'DELETE':
            case 'POST':
                $this->request = $this->_cleanInputs($_POST);
                break;
            case 'GET':
                array_shift($_GET);
                $this->request = $this->_cleanInputs($_GET);
                break;
            case 'PUT':
                $this->request = $this->_cleanInputs($_GET);
                $this->file =
file_get_contents("php://input");
                break;
            default:
```

[1] Phương thức yêu cầu thực thi
[2] Lỗi không tim thấy

```
                return $this->_response('Invalid Method',
405);
                break;
        }
        if ((int)method_exists($this, $this->endpoint) > 0) {
            return $this->_response($this->{$this-
>endpoint}($this->args));
        }
        return $this->_response(array("No Endpoint"=>$this-
>endpoint), 404);
    }
```

4.3.4. Lớp xử lý yêu cầu

– Lớp này được kết thừa từ lớp nhận yêu cầu.

– Chứa các phương thức endpoint

– Mỗi phương thức endpoint này sẽ thực thi các yêu cầu khác nhau theo yêu cầu từ mỗi thiết bị khác.

– Ví dụ: một phương thức để nhận thông tin từ sản phẩm mới.

```
class MyAPI extends API
{
    public function __construct($request, $origin) {
        try {
            parent::__construct($request);
        } catch (Exception $e) {
        }
    }
protected function updatenewproduct(){
    $Product = new Product_Model();
    switch ($this->method){
        case "GET":
            if(isset($this->args[0])){
                if (isset($this->request['timeupdate'])){
                    if($this->args[0]==="count"){
                        return array("data"=>$Product-
>get_count_update($this-
>request['timeupdate']),"request"=>$this->request);
                    }
                    if($this->args[0]==="lists"){
                        return array("data"=>$Product-
>get_list_update($this-
>request['timeupdate']),"request"=>$this->request);
                    }
```

```
                }
            }
            return array();
            break;
        case "POST":
            return array(
                "status" => "success",
                "endpoint" => $this->endpoint,
                "method" => $this->method,
                "args" => $this->args,
                "request" => $this->request);
            break;
        default: return array("error"=>"Can not determine the
method");
    }
}
}
```

4.4. Xây dựng ứng dụng Android

4.4.1. Gửi yêu cầu và nhận dữ liệu

a. Thông tin sản phẩm mới

– Để nhận thông tin có sản phẩm mới hay chưa:

Xây dựng chuỗi yêu cầu:

```
http://{API_DOMAIN}/api/updatenewproduct/count?timeupdate={tim
eupdate}
```

Trong đó {API_DOMAIN} tên miền webservice {timeupdate} thời cập nhật từ lần trước thười gian này được lưu bởi lớp `SharedPreferences`

Khai báo một đối tượng thuộc lớp `JsonObjectRequest` để gửi yêu cầu vời năm tham số truyền vào lần lượt là:

- Phương thức truyền (GET, POST, DELETE, PUT) ở đây ta chuyền phương thức GET
- Chuỗi yêu cầu:
- Tham số đối tượng gửi đi: null
- Phương thức `onResponse()` nhận kết quả trả về `count_new_product = response.getInt("data");` nếu `count_new_product` lớn hớn 0 thì tức là có sản phẩm mới ta thực hiên lưu thời gian cập nhật mới gửi đến điện

thoại một thông báo là có sản phẩm mới và đọc dữ liệu mới và thêm vào trong SQLite, gửi một thông báo tới màn hình cho người dùng biết.

• Phương thức `Response.ErrorListener()` thực hiện khi gửi yêu cầu sảy ra lỗi.

```
JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(
Request.Method.GET,
url_Request, null,
new Response.Listener<JSONObject>() {
@Override
public void onResponse(JSONObject response) {
try {
  count_new_product = response.getInt("data");
  if (count_new_product > 0){
  sharedPreferencesCustom.Save_time();
  ProductNotification.notify(context,"có "+count_new_product
+" sản phẩm mới",1);
  ReadJsonListNewProduct();
    }
    } catch (JSONException e) {
     e.printStackTrace();
    }
  }
},
new Response.ErrorListener() {
@Override
public void onErrorResponse(VolleyError error) {
      Log.d("error",error.toString());
       }
  }
);
```

Khai báo đối tượng thộc lớp RequestQueue của thư viện Volley rùi thêm yêu cầu jsonObjectRequest vào hàng đợi để thực hiện

```
RequestQueue requestQueue = Volley.newRequestQueue(context);
requestQueue.add(jsonObjectRequest);
```

b. Danh sách sản phẩm mới

– Tương tự như kiểm tra có sản phẩm mới ta cugnx sử dụng thư viện Volley để đọc thông tin sản phẩm mới.

– Chuỗi yêu cầu:

```
http://{API_DOMAIN}/api/updatenewproduct/lists?timeupdate={tim
eupdate}
```

Cách đọc sản phẩn trong phương thức `onResponse()` đọc thông tin mảng danh sách sản phẩm mới ở dạng Json, lấy thông tin cả mảng sau đó đọc thông tin từng phần tử bằng vòng for và khởi tạo đối tượng product thêm thông tin đọc gán thông tin đọc được vào đối tượng đó. Sau đó kiểm tra đối tượng product nếu có trong SQLite rùi thì thực hiện cập nhật sản phẩm đó còn chưa thì thực hiện thêm mới sản phẩm.

```
JSONArray data = response.getJSONArray("data");
     for (int i=0 ; i< data.length() ; i++){
         JSONObject object = data.getJSONObject(i);
         Product product = new Product(
object.getInt("id"),
object.getString("name"),
object.getInt("price"),
object.getString("detail"),
object.getString("brand"),
object.getString("image"),
object.getInt("weight"),
object.getInt("quantum"),
object.getString("date"),
object.getInt("status")
);
    productArrayList.add(product);
    DatabaseSQLite sqLite = new DatabaseSQLite(context);
    if (sqLite.checkproduct(product.getId()) > 0){
        sqLite.UPDATE_Product(product);
    }else
    {
         sqLite.INSERT_Product(product);
    }
```

4.4.2. Lớp ProductReceiver

Lớp là một trong những lớp được kế thừa từ lớp BroadcastReceiver và nó để lắng nghe sự kiện khi lớp định hằng thời gian gọi nó qua phương thức onReceive().

Trong phương thức `onReceive()` sẽ kiểm tra thiết bị có kết nối internet hay không nếu có thì khởi tạo đối tượng thộc lớp ProductReadAPI[1] để gọi phương thức kiểm tra sản phẩm:

– Mã nguồn lớp `ProductReceiver`:

[1] Lớp gửi yêu cầu và đọc dữ liệu

```
public class ProductReceiver extends BroadcastReceiver {
    Context context;
    @Override
    public void onReceive(Context context, Intent intent) {
        this.context = context;
        if (MyFunction.isConnected(context)){
            ProductReadAPI  productReadAPI = new
ProductReadAPI(context);
           productReadAPI.checkNewProduct();
        }else Log.d("Error","Không có kết nối internet");
    }
}
```

4.4.3. Lớp định hằng thời gian

a. Mô tả

Đây là một lớp dùng để định một khoảng thời gian t nào đó để thực hiện một chức năng nào đó. Ví dụ đến một khoảng thời gian t thì chương trình sẽ kiểm tra và cập nhật sản phẩm mới được cập nhật trên hosting.

Xây dựng lớp này ta định một khoảng thời gian t được tính bằng mili giây ví dụ định thời 5 phút ta lấy 5*60*1000. Sử dụng phương thức `setRepeating()` của lớp `AlarmManager`[1] để định hằng một khoảng thời gian t sẽ thực hiện chức năng cập nhật sản phẩm mới.

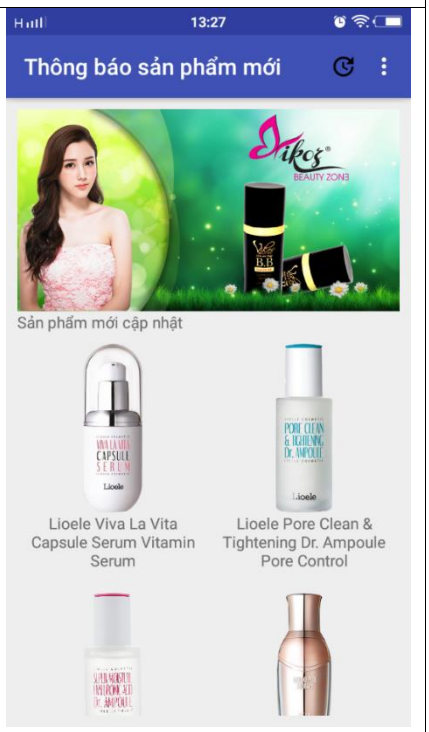
b. Mã nguồn

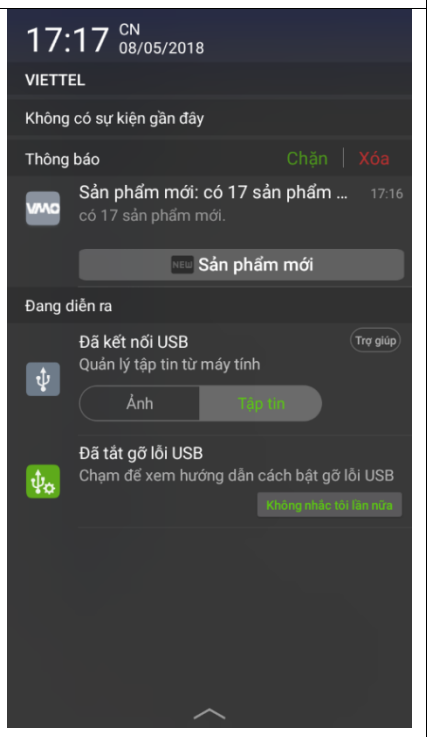
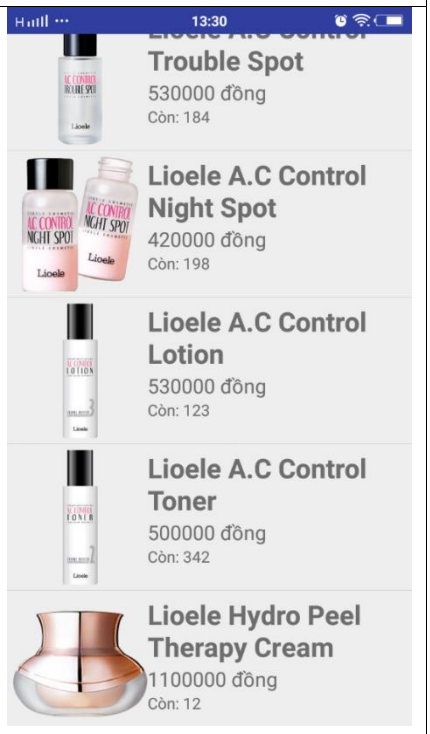
```
public class DatLich {
    private AlarmManager alarmManager;
    private PendingIntent pendingIntent;
    private Context context;
    private Intent intent;
    public DatLich( Context context) {
        this.context = context;
        this.alarmManager = (AlarmManager)
context.getSystemService(ALARM_SERVICE);
        intent = new Intent(context, ProductReceiver.class);
    }

    public void khoitao(){
        pendingIntent = PendingIntent.getBroadcast(
                context,
```

[1] Lớp cung cấp các phương thức để truy cập dịch vụ báo thức của hệ thống

```
                0,
                intent,
                PendingIntent.FLAG_UPDATE_CURRENT
        );
        long interval = 60*5*1000;
        alarmManager.setRepeating(
                AlarmManager.ELAPSED_REALTIME,
                SystemClock.elapsedRealtime()+interval,
                interval,
                pendingIntent);
    }

    public void huy(){
        pendingIntent.cancel();
        alarmManager.cancel(pendingIntent);
    }
}
```

4.4.4. Một số giao diện chương trình

Mô tả	Hình ảnh
Màn hình chính của chương trình: – Phần khu vực hình ảnh khi ấn sẽ hiể thị ra toàn bộ danh sách sản phẩm. • Nếu thiết bị đang được kết nối internet thì dữ liệu sẽ lấy từ trực tiếp từ hosting • Còn không thì sẽ lấy dữ liệu ở SQLite để hiển thị. – Phần kế tiếp là phền hiển thị những sản phẩm mới cập nhật trong 7 ngày gấn nhất.	

Mô tả	Hình ảnh
Đây là phần thông báo của thiết bị khi có sản phẩm mới được cập nhật, phần thông báo gồm có: – Icon của ứng dụng – Tiêu đề thông báo – Nội dung thông báo ấn vào để hiển thị màn hình chính của chương trình – Nút ấn để di chuyển tới màn hình xem sản phẩm mới được cập nhật.	
Màn hình này hiển thị danh sách các sản phẩm theo dạng lisview. Gồm có các thông tin: – Hình ảnh sản phẩm – Tên sản phẩm – Giá sản phẩm – Số lượng trong kho	

Mô tả	Hình ảnh
Màn hình này hiển thị thông tin chi tiết của sản phẩm: – Tên sản phẩm – Hình ảnh sản phẩm – Thương hiệu của sản phẩm – Giá sản phẩm – Số lượng sản phẩm trong kho – Chi tiết về sản phẩm	

TỔNG KẾT

Trong khoảng thời gian 3 tháng cũng không phải là một thời gian dài Em đã học hỏi được rất nhiều thứ từ nghiên cứu học, triển khai làm đồ án tốt nghiệp, Em đã viết được chương trình chạy trên thiết bị Android, tạo được cơ sở dữ liệu trên hosting, xây dựng được webservice để truy xuất dữ liệu trên hosting, định hằng thời gian cập nhật được dữ liệu mới từ hosting cho ứng dụng Android.

Song song với đó, bước đầu xây dựng thành công một ứng dụng Android cập nhật dữ liệu mới theo định hằng thời gian thực. Tuy nhiên, do thời gian và khả năng có hạn, nên Em chưa đi sâu tìm hiểu được thêm về ứng dụng, về giao diện ứng dụng vẫn còn sơ sài và không được trau chuốt, ứng dụng vẫn còn ít chức năng mới chỉ dừng lại ở cập nhật thông tin mới, hiển thị thông báo khi có thông tin mới và hiển thị thông tin lưu thông tin về máy.

Trong thời gian tới Em sẽ phát triển ứng dụng này thêm nhiều chức năng mới như cho người dùng theo dõi sản phẩm, đánh giá sản phẩm, thêm chỉ điểm bán sản phẩm, kết nối tài khoản mạng xã hội của mỗi người để có thể chia sẻ thông tin sản phẩm.

TÀI LIỆU THAM KHẢO

❖ **Tài liệu**

[1].Giáo trình Android của đại học Khoa Học Tự Nhiên

[2].Tài liệu lập trình Android Full – FPT Software

[3].Beginning Android 4 Application Development.

❖ **Tham khảo trực tuyến**

[4].https://developer.android.com/guide

[5].https://www.youtube.com/user/khoazend

[6].https://vi.wikipedia.org/