Mục Lục

[Lời nói đầu 3](#_Toc29074439)

[CHƯƠNG 1: TỔNG QUAN HỆ ĐIỀU HÀNH ANDROID 4](#_Toc29074440)

[1.1 GIỚI THIỆU HỆ ĐIỀU HÀNH ANDROID 4](#_Toc29074441)

[1.2 LỊCH SỬ PHÁT TRIỂN 5](#_Toc29074442)

[1.4 ỨNG DỤNG ANDROID 7](#_Toc29074443)

[1.5 QUẢN LÝ BỘ NHỚ 8](#_Toc29074444)

[1.6 NHÂN LINUX 8](#_Toc29074445)

[1.7 LỊCH NÂNG CẤP 10](#_Toc29074446)

[1.8 CỘNG ĐỒNG MÃ NGUỒN MỞ 11](#_Toc29074447)

[1.9 BẢO MẬT VÀ TÍNH RIÊNG TƯ CỦA ANDROID 12](#_Toc29074448)

[1.10 GIẤY PHÉP PHÁT HÀNH 13](#_Toc29074449)

[1.11 ĐÓN NHẬN 14](#_Toc29074450)

[1.12 KIẾN TRÚC CỦA ANDROID 15](#_Toc29074451)

[1.12.1 Android platform 15](#_Toc29074452)

[1.12.2 Tầng linux kernel 15](#_Toc29074453)

[1.12.3 Native libraries 16](#_Toc29074454)

[1.12.4 Tầng runtime 16](#_Toc29074455)

[1.12.5 Tầng application framework 16](#_Toc29074456)

[CHƯƠNG 2: MÔI TRƯỜNG LẬP TRÌNH 18](#_Toc29074457)

[2.1 GIỚI THIỆU 18](#_Toc29074458)

[2.2 ANDROID SDK 18](#_Toc29074459)

[2.3 MÁY ẢO DALVIK 18](#_Toc29074460)

[2.4 CÁC GÓI JAVA CẦN THIẾT 19](#_Toc29074461)

[2.5 THÀNH PHẦN CHÍNH CỦA ECLIPSE KHI TẠO ỨNG DỤNG TRÊN ANDROID 20](#_Toc29074462)

[2.6 THÀNH PHẦN QUAN TRỌNG TRONG MỘT ANDROID PROJECT 20](#_Toc29074463)

[2.7 CHU KỲ SỐNG CỦA ỨNG DỤNG ANDROID 21](#_Toc29074464)

[2.7.1Chu kỳ sống thành phần 21](#_Toc29074465)

[2.7.2Activity stack 21](#_Toc29074466)

[2.7.3 Các trạng thái của chu kỳ sống 22](#_Toc29074467)

[2.7.4 Chu kỳ sống của ứng dụng 23](#_Toc29074468)

[2.7.5 Các sự kiện trong chu kỳ sống của ứng dụng 23](#_Toc29074469)

[2.7.6 Thời gian sống của ứng dụng 24](#_Toc29074470)

[2.7.7 Thời gian hiển thị của activity 24](#_Toc29074471)

[2.7.8 Các hàm thực thi của activity 24](#_Toc29074472)

[2.8.1 View 25](#_Toc29074473)

[2.8.2 ViewGroup 25](#_Toc29074474)

[2.9 CÁC CONTROL 26](#_Toc29074475)

[2.9.1 Các button 26](#_Toc29074476)

[2.9.2 Listview 26](#_Toc29074477)

[2.9.3 Edittext 27](#_Toc29074478)

[2.9.4 Textview 27](#_Toc29074479)

[2.9.5 Checkbox 27](#_Toc29074480)

[2.9.6 Menuoption 27](#_Toc29074481)

[2.9.7 Contextmenu 27](#_Toc29074482)

[2.10 TÙY BIẾN CONTROL (CUSTOM CONTROL) 28](#_Toc29074483)

[CHƯƠNG 3: KỸ THUẬT ĐỌC LẤY NỘI DUNG RSS 29](#_Toc29074484)

[3.1 TÌM HIỂU VỀ DỊCH VỤ RSS 29](#_Toc29074485)

[3.2 VÍ DỤ 29](#_Toc29074486)

[3.3 KỸ THUẬT ĐỌC NỘI DUNG CỦA TRANG WEBSITE VỀ ĐIỆN THOẠI 30](#_Toc29074487)

[CHƯƠNG 4: CHƯƠNG TRÌNH THỰC NGHIỆM 40](#_Toc29074488)

[4.1 PHÁT BIỂU BÀI TOÁN 40](#_Toc29074489)

[4.2 GIAO DIỆN ỨNG DỤNG 40](#_Toc29074490)

[KẾT LUẬN 44](#_Toc29074492)

[TÀI LIỆU THAM KHẢO 44](#_Toc29074493)

# Lời nói đầu

Mạng điện thoại di động xuất hiện tại Việt Nam từ đầu những năm 1990 và theo thời gian số lượng các thuê bao cũng như các nhà cung cấp các dịch vụ di động tại Việt Nam ngày càng tăng, do nhu cầu trao đổi thong tin ngày càng tăng và nhu cầu sử dụng sản phẩm công nghệ cao nhiều tính năng, cấu hình cao, chất lượng tốt, kiểu dáng mẫu mà đẹp, phong phú nên nhà cung cấp phải luôn luôn cải thiện, nâng cao những sản phẩm của mình. 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.

Cùng với sự phát triển của thị trường điện thoại di động là sự phát triển mạnh mẽ của xu hướng lập trình phần mềm ứng dụng cho các thiết bị di động. phần mềm, ứng dụng cho thiết bị di động hiện nay rất đa dạng và phong phú trên các hệ điều hành di động cũng phát triển mạnh mẽ và đang thay đổi từng ngày. Các hệ điều hành J2ME, Android, IOS, Hybrid, Web based Mobile Application đã có rất phát triển trên thị trường truyền thong di động.

Ngày nay, với sự phát triển nhanh chống của xã hội, nhu cầu cập nhật tin tức xã hội mọi lúc mọi nơi là rất cần thiết

# CHƯƠNG 1: TỔNG QUAN HỆ ĐIỀU HÀNH ANDROID

## 1.1 GIỚI THIỆU HỆ ĐIỀU HÀNH ANDROID

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Tổng công ty Android, với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005. Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào tháng 10 năm 2008.

Android có mã nguồn mở và Google phát hành mã nguồn theo Giấy phép Apache. Chính mã nguồn mở cùng với một giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên nhiệt huyết được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị, bằng một loại ngôn ngữ lập trình Java có sửa đổi. Vào tháng 10 năm 2012, có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play, cửa hàng ứng dụng chính của Android, ƣớc tính khoảng 25 tỷ lượt.

Những yếu tố này đã giúp Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới, vượt qua Symbian vào quý 4 năm 2010, và được các công ty công nghệ lựa chọn khi họ cần một hệ điều hành không nặng nề, có khả năng tinh chỉnh, và giá rẻ chạy trên các thiết bị công nghệ cao thay vì tạo dựng từ đầu. Kết quả là mặc dù được thiết kế để chạy trên điện thoại và máy tính bảng, Android đã xuất hiện trên TV, máy chơi game và các thiết bị điện tử khác. Bản chất mở của Android cũng khích lệ một đội ngũ đông đảo lập trình viên và những ngƣời đam mê sử dụng mã nguồn mở để tạo ra những dự án do cộng đồng quản lý. Những dự án này bổ sung các tính năng cao cấp cho những người dùng thích tìm tòi hoặc đưa Android vào các thiết bị ban đầu chạy hệ điều hành khác.

Android chiếm 75% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 3 năm 2012, với tổng cộng 500 triệu thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Sự thành công của hệ điều hành cũng khiến nó trở thành mục tiêu trong các vụ kiện liên quan đến bằng phát minh, góp mặt trong cái gọi là "cuộc chiến điện thoại thông minh" giữa các công ty công nghệ.

## 1.2 LỊCH SỬ PHÁT TRIỂN

Tổng công ty Android (Android, Inc.) được thành lập tại Palo Alto, California vào tháng 10 năm 2003 bởi Andy Rubin (đồng sáng lập công ty Danger),[20] Rich Miner (đồng sáng lập Tổng công ty Viễn thông Wildfire), Nick Sears (từng là Phó giám đốc T-Mobile), và Chris White (trưởng thiết kế và giao diện tại WebTV) để phát triển, theo lời của Rubin, "các thiết bị di động thông minh hơn có thể biết được vị trí và sở thích của người dùng". Dù những người thành lập và nhân viên đều là những người có tiếng tăm, Tổng công ty Android hoạt động một cách âm thầm, chỉ tiết lộ rằng họ đang làm phần mềm dành cho điện thoại di động. Trong năm đó, Rubin hết kinh phí. Steve Perlman, một người bạn thân của Rubin, mang cho ông 10.000 USD tiền mặt nhưng từ chối tham gia vào công ty.

Google mua lại Tổng công ty Android vào ngày 17 tháng 8 năm 2005, biến nó thành một bộ phận trực thuộc Google. Những nhân viên của chủ chốt của Tổng công ty Android, gồm Rubin, Miner và White, vẫn tiếp tục ở lại công ty làm việc sau thƣơng vụ này. Vào thời điểm đó không có nhiều thông tin về Tổng công ty, nhưng nhiều người đồn đoán rằng Google dự tính tham gia thị trường điện thoại di động sau bước đi này. Tại Google, nhóm do Rubin đứng đầu đã phát triển một nền tảng thiết bị di động phát triển trên nền nhân Linux. Google quảng bá nền tảng này cho các nhà sản xuất điện thoại và các nhà mạng với lời hứa sẽ cung cấp một hệ thống uyển chuyển và có khả năng nâng cấp. Google đã liên hệ với hàng loạt hãng phần cứng cũng nhƣ đối tác phần mềm, bắn tin cho các nhà mạng rằng họ sẵn sàng hợp tác với các cấp độ khác nhau.

Ngày càng nhiều suy đoán rằng Google sẽ tham gia thị trường điện thoại di động xuất hiện trong tháng 12 năm 2006. Tin tức của BBC và Nhật báo phố Wall chú thích rằng Google muốn đƣa công nghệ tìm kiếm và các ứng dụng của họ vào điện thoại di động và họ đang nỗ lực làm việc để thực hiện điều này. Các phƣơng tiện truyền thông truyền thống lẫn online cũng viết về tin đồn rằng Google đang phát triển một thiết bị cầm tay mang thương hiệu Google. Một vài tờ báo còn nói rằng trong khi Google vẫn đang thực hiện những bản mô tả kỹ thuật chi tiết, họ đã trình diễn sản phẩm mẫu cho các nhà sản xuất điện thoại di động và nhà mạng. Tháng 9 năm 2007, InformationWeek đăng tải một nghiên cứu của Evalueserve cho biết Google đã nộp một số đơn xin cấp bằng sáng chế trong lĩnh vực điện thoại di động.

Ngày 5 tháng 11 năm 2007, Liên minh thiết bị cầm tay mở (Open Handset Alliance), một hiệp hội bao gồm nhiều công ty trong đó có Texas Instruments, Tập đoàn Broadcom, Google, HTC, Intel, LG, Tập đoàn Marvell Technology, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel và T-Mobile được thành lập với mục đích phát triển các tiêu chuẩn mở cho thiết bị di động. Cùng ngày, Android cũng được ra mắt với vai trò là sản phẩm đầu tiên của Liên minh, một nền tảng thiết bị di động được xây dựng trên nhân Linux phiên bản 2.6. Chiếc điện thoại chạy Android đầu tiên được bán ra là HTC Dream, phát hành ngày 22 tháng 10 năm 2008. Biểu trƣng của hệ điều hành Android mới là một con rôbốt màu xanh lá cây do hãng thiết kế Irina Blok tại California vẽ.

Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trƣớc. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng; ví dụ nhƣ phiên bản 1.5 Cupcake (bánh bông lan nhỏ có kem) tiếp nối bằng phiên bản 1.6 Donut (bánh vòng). Phiên bản mới nhất hiện nay là 5.0 Lollipop. Vào năm 2010, Google ra mắt loạt thiết bị Nexus—một dòng sản phẩm bao gồm điện thoại thông minh và máy tính bảng chạy hệ điều hành Android, do các đối tác phần cứng sản xuất. HTC đã hợp tác với Google trong chiếc điện thoại thông minh Nexus đầu tiên, Nexus One. Kể từ đó nhiều thiết bị mới hơn đã gia nhập vào dòng sản phẩm này, nhờ điện thoại Nexus 4 và máy tính bảng Nexus 10, lần lượt do LG và Samsung sản xuất. Google xem điện thoại và máy tính bảng Nexus là những thiết bị Android chủ lực của mình, với những tính năng phần cứng và phần mềm mới nhất của Android.

Năm 2014, Google công báo Android Wear, hệ điều hành dành cho các thiết bị đeo được.

1.3 GIAO DIỆN ANDROID

Giao diện ngƣời 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 vôlăng.

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 hính 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 theo 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 ỨNG DỤNG ANDROID

Android có lượng ứng dụng của bên thứ ba ngày càng nhiều, được chọn lọc và đặt trên một cửa hàng ứng dụng như Google Play hay Amazon Appstore để người dùng lấy về, hoặc bằng cách tải xuống rồi cài đặt tập tin APK từ trang web khác. Các ứng dụng trên Cửa hàng Play cho phép người dùng duyệt, tải về và cập nhật các ứng dụng do Google và các nhà phát triển thứ ba phát hành. Cửa hàng Play được cài đặt sẵn trên các thiết bị thỏa mãn điều kiện tương thích của Google. Ứng dụng sẽ tự động lọc ra một danh sách các ứng dụng tương thích với thiết bị của người dùng, và nhà phát triển có thể giới hạn ứng dụng của họ chỉ dành cho những nhà mạng cố định hoặc những quốc gia cố định vì lý do kinh doanh. Nếu người dùng mua một ứng dụng mà họ cảm thấy không thích, họ được hoàn trả tiền sau 15 phút kể từ lúc tải về, và một vài nhà mạng còn có khả năng mua giúp các ứng dụng trên Google Play, sau đó tính tiền vào trong hóa đơn sử dụng hàng tháng của người dùng. Đến tháng 9 năm 2012, có hơn 675.000 ứng dụng dành cho Android, và số lượng ứng dụng tải về từ Cửa hàng Play ước tính đạt 25 tỷ.

Các ứng dụng cho Android được phát triển bằng ngôn ngữ Java sử dụng Bộ phát triển phần mềm Android (SDK). SDK bao gồm một bộ đầy đủ các công cụ dùng để phát triển, gồm có công cụ gỡ lỗi, thư viện phần mềm, bộ giả lập điện thoại dựa trên QEMU, tài liệu hướng dẫn, mã nguồn mẫu, và hướng dẫn từng bước. Môi trường phát triển tích hợp (IDE) được hỗ trợ chính thức là Eclipse sử dụng phần bổ sung Android Development Tools (ADT). Các công cụ phát triển khác cũng có sẵn, gồm có Bộ phát triển gốc dành cho các ứng dụng hoặc phần mở rộng viết bằng C hoặc C++, Google App Inventor, một môi trường đồ họa cho những nhà lập trình mới bắt đầu, và nhiều nền tảng ứng dụng web di động đa nền tảng phong phú.

## 1.5 QUẢN LÝ BỘ NHỚ

Vì các thiết bị Android được thiết kế để quản lý bộ nhớ (RAM) để giảm tối đa mức tiêu thụ điện năng, trái với hệ điều hành máy tính để bàn luôn cho rằng máy tính sẽ có nguồn điện không giới hạn. Khi một ứng dụng Android không còn được sử dụng, hệ thống sẽ tự động ngưng nó trong bộ nhớ - trong khi ứng dụng về mặt kỹ thuật vẫn "mở", những ứng dụng này sẽ không tiêu thụ bất cứ tài nguyên nào (như năng lượng pin hay năng lượng xử lý) và nằm đó cho đến khi nó được cần đến. Cách làm như vậy có lợi kép là vừa làm tăng khả năng phản hồi nói chung của thiết bị Android, vì ứng dụng không nhất phải đóng rồi mở lại từ đầu, vừa đảm bảo các ứng dụng nền không làm tiêu hao năng lượng một cách không cần thiết.

Android quản lý các ứng dụng trong bộ nhớ một cách tự động: khi bộ nhớ thấp, hệ thống sẽ bắt đầu diệt ứng dụng và tiến trình không hoạt động được một thời gian, sắp theo thời điểm cuối mà chúng được sử dụng (tức là cũ nhất sẽ bị tắt trước). Tiến trình này được thiết kế ẩn đi với người dùng, để người dùng không cần phải quản lý bộ nhớ hoặc tự tay tắt các ứng dụng. Tuy nhiên, sự che giấu này của hệ thống quản lý bộ nhớ Android đã dẫn đến sự thịnh hành của các ứng dụng tắt chương trình của bên thứ ba trên cửa hàng Google Play; những ứng dụng kiểu như vậy được cho là có hại nhiều hơn có lợi.

## 1.6 NHÂN LINUX

Android có một hạt nhân dựa trên nhân Linux phiên bản 2.6, kể từ Android 4.0 Ice Cream Sandwich (bánh ngọt kẹp kem) trở về sau, là phiên bản 3.x, với middleware, thƣ viện và API viết bằng C, còn phần mềm ứng dụng chạy trên một nền tảng ứng dụng gồm các thư viện tương thích với Java dựa trên Apache Harmony. Android sử dụng máy ảo Dalvik với một trình biên dịch động để chạy 'mã dex' (Dalvik Executable) của Dalvik, thường được biên dịch sang Java bytecode. Nền tảng phần cứng chính của Android là kiến trúc ARM. Người ta cũng hỗ trợ x86 thông qua dự án Android x86, và Google TV cũng sử dụng một phiên bản x86 đặc biệt của Android.

Một số tính năng cũng được Google đóng góp ngược vào nhân Linux, đáng chú ý là tính năng quản lý nguồn điện có tên wakelock, nhưng bị những người lập trình chính cho nhân từ chối vì họ cảm thấy Google không có định sẽ tiếp tục bảo trì đoạn mã do họ viết. Google thông báo vào tháng 4 năm 2010 rằng họ sẽ thuê hai nhận viên để làm việc với cộng đồng nhân Linux, nhưng Greg KroahHartman, người bảo trì nhân Linux hiện tại của nhánh ổn định, đã nói vào tháng 12 năm 2010 rằng ông ta lo ngại rằng Google không còn muốn đưa những thay đổi của mình vào Linux dòng chính nữa. Một số lập trình viên Android của Google tỏ ý rằng "nhóm Android thấy chán với quy trình đó," vì nhóm họ không có nhiều người và có nhiều việc khẩn cấp cần làm với Android hơn.

Vào tháng 8 năm 2011, Linus Torvalds rằng "rốt cuộc thì Android và Linux cũng sẽ trở lại với một bộ nhân chung, nhưng điều đó có thể sẽ không xảy ra trong 4 hoặc 5 năm nữa". Vào tháng 12 năm 2011, Greg Kroah-Hartman thông báo kích hoạt Dự án Dòng chính Android, nhắm tới việc đưa một số driver, bản vá và tính năng của Android ngược vào nhân Linux, bắt đầu từ Linux 3.3. Linux cũng đưa tính năng autosleep (tự nghỉ hoạt động) và wakelocks vào nhân 3.5, sau nhiều nỗ lực phối trộn trước đó. Tương tác thì vẫn vậy nhưng bản hiện thực trên Linux dòng chính cho phép hai chế độ nghỉ: bộ nhớ (dạng nghỉ truyền thống mà Android sử dụng), và đĩa (là ngủ đông trên máy tính để bàn). Việc trộn sẽ hoàn tất kể từ nhân 3.8, Google đã công khai kho mã nguồn trong đó có những đoạn thử nghiệm đưa Android về lại nhân 3.8.

Bộ lưu trữ flash trên các thiết bị Android được chia thành nhiều phân vùng, như "system" dành cho hệ điều hành và "/data" dành cho dữ liệu người dùng và cài đặt ứng dụng. Khác với các bản phân phối Linux cho máy tính để bàn, người sở hữu thiết bị Android không được trao quyền truy cập root vào hệ điều hành và các phân vùng nhạy cảm như /system được thiết lập chỉ đọc. Tuy nhiên, quyền truy cập root có thể chiếm được bằng cách tận dụng những lỗ hổng bảo mật trong Android, điều mà cộng đồng mã nguồn mở thường xuyên sử dụng để nâng cao tính năng thiết bị của họ, kể cả bị những người ác ý sử dụng để cài virus và phần mềm ác ý.

Việc Android có được xem là một bản phân phối Linux hay không vẫn còn là vấn đề gây tranh cãi, tuy được Linux Foundation và Chris DiBona, trưởng nhóm mã nguồn mở Google, ủng hộ. Một số khác, như linux-magazine.com thì không đồng ý, do Android không không hỗ trợ nhiều công cụ GNU, trong đó có glibc.

## 1.7 LỊCH NÂNG CẤP

Google đưa ra các bản nâng cấp lớn cho Android theo chu kỳ từ 6 đến 9 tháng, mà phần lớn thiết bị đều có thể nhận được qua sóng không dây. Bản nâng cấp lớn mới nhất là Android 6.0 Marshmallow.

So với các hệ điều hành cạnh tranh khác, như iOS, các bản nâng cấp Android thường mất thời gian lâu hơn để đến với các thiết bị. Với những thiết bị không thuộc dòng Nexus, các bản nâng cấp thường đến sau vài tháng kể từ khi phiên bản được chính thức phát hành. Nguyên nhân của việc này một phần là do sự phong phú về phần cứng của các thiết bị Android, nên người ta phải mất thời gian điều chỉnh bản nâng cấp cho phù hợp, vì mã nguồn chính thức của Google chỉ chạy được trên những thiết bị Nexus chủ lực của họ. Chuyển Android sang những phần cứng cụ thể là một quy trình tốn thời gian và công sức của các nhà sản xuất thiết bị, những người luôn ưu tiên các thiết bị mới nhất và thường bỏ rơi các thiết bị cũ hơn. Do đó, những chiếc điện thoại thông minh thế hệ cũ thường không được nâng cấp nếu nhà sản xuất quyết định rằng nó không đáng để bỏ thời gian, bất kể chiếc điện thoại đó có khả năng chạy bản nâng cấp hay không. Vấn đề này còn trầm trọng hơn khi những nhà sản xuất điều chỉnh Android để đưa giao diện và ứng dụng của họ vào, những thứ này cũng sẽ phải làm lại cho mỗi bản nâng cấp. Sự chậm trễ còn được đóng góp bởi nhà mạng, sau khi nhận được bản nâng cấp từ nhà sản xuất, họ còn điều chỉnh thêm cho phù hợp với nhu cầu rồi thử nghiệm kỹ lưỡng trên hệ thống mạng của họ trước khi chuyển nó đến người dùng.

Việc thiếu các hỗ trợ hậu mãi của nhà sản xuất và nhà mạng đã bị những nhóm người dùng và các trang tin công nghệ chỉ trích rất nhiều. Một số người viết còn nói rằng giới công nghiệp do cái lợi về tài chính đã cố tình không nâng cấp thiết bị, vì nếu thiết bị hiện tại không cập nhật sẽ thúc đẩy việc mua thiết bị mới, một thái độ được coi là "xúc phạm". The Guardian đã than phiền rằng phương cách phân phối bản nâng cấp trở nên phức tạp chính vì những nhà sản xuất và nhà mạng đã cố tình làm nó như thế. Vào năm 2011, Google đã hợp tác cùng một số hãng công nghiệp và ra mắt "Liên minh nâng cấp Android", với lời hứa sẽ nâng cấp thường xuyên cho các thiết bị trong vòng 18 tháng sau khi ra mắt. Tính đến năm 2012, người ta không còn nghe nhắc đến liên minh này nữa.

## 1.8 CỘNG ĐỒNG MÃ NGUỒN MỞ

Android có một cộng đồng các lập trình viên và những người đam mê rất năng động. Họ sử dụng mã nguồn Android để phát triển và phân phối những phiên bản chỉnh sửa của hệ điều hành. Các bản Android do cộng đồng phát triển thường đem những tính năng và cập nhật mới vào nhanh hơn các kênh chính thức của nhà sản xuất/nhà mạng, tuy không được kiểm thử kỹ lưỡng cũng như không có đảm bảo chất lượng; cung cấp sự hỗ trợ liên tục cho các thiết bị cũ không còn nhận được bản cập nhật chính thức; hoặc mang Android vào những thiết bị ban đầu chạy một hệ điều hành khác, như HP Touchpad. Các bản Android của cộng đồng thường được root sẵn và có những điều chỉnh không phù hợp với những người dùng không rành rẽ, như khả năng ép xung hoặc tăng/giảm áp bộ xử lý của thiết bị. CyanogenMod là firmware của cộng đồng được sử dụng phổ biến nhất, và hoạt động như một tổ chức của số đông khác.

Trước đây, nhà sản xuất thiết bị và nhà mạng tỏ ra thiếu thiện chí với việc phát triển firmware của bên thứ ba. Những nhà sản xuất còn thể hiện lo ngại rằng các thiết bị chạy phần mềm không chính thức sẽ hoạt động không tốt và dẫn đến tốn tiền hỗ trợ. Hơn nữa, các firmware đã thay đổi như CyanogenMod đôi khi còn cung cấp những tính năng, như truyền tải mạng (tethering), mà người dùng bình thường phải trả tiền nhà mạng mới được sử dụng. Kết quả là nhiều thiết bị bắt đầu đặt ra hàng rào kỹ thuật như khóa bootloader hay hạn chế quyền truy cập root. Tuy nhiên, khi phần mềm do cộng đồng phát triển ngày càng trở nên phổ biến, và sau một thông cáo của Thư viện Quốc hội Hoa Kỳ cho phép "jailbreak" (vượt ngục) thiết bị di động, các nhà sản xuất và nhà mạng đã tỏ ra mềm mỏng hơn với các nhà phát triển thứ ba, thậm chí một số hãng như HTC, Motorola, Samsung và Sony, còn hỗ trợ và khuyến khích phát triển. Kết quả của việc này là dần dần nhu cầu tìm ra các hạn chế phần cứng để cài đặt được firmware không chính thức đã bớt đi do ngày càng nhiều thiết bị được phát hành với bootloader đã mở khóa sẵn hoặc có thể mở khóa, tương tự như điện thoại dòng Nexus, tuy rằng thông thường họ sẽ yêu cầu người dùng từ bỏ chế độ bảo hành nếu họ làm như vậy. Tuy nhiên, tuy được sự chấp thuận của nhà sản xuất, một số nhà mạng tại Mỹ vẫn bắt buộc điện thoại phải bị khóa.

Việc mở khóa và "hack" điện thoại thông minh và máy tính bảng vẫn còn là tác nhân gây căng thẳng giữa cộng đồng và công nghiệp. Cộng đồng luôn biện hộ rằng sự hỗ trợ không chính thức ngày càng trở nên quan trọng trước việc nền công nghiệp không cung cấp các bản cập nhật thường xuyên và/hoặc ngưng hỗ trợ cho chính các thiết bị của họ.

## 1.9 BẢO MẬT VÀ TÍNH RIÊNG TƯ CỦA ANDROID

Các ứng dụng Android chạy trong một "hộp cát", là một khu vực riêng rẽ với hệ thống và không được tiếp cận đến phần còn lại của tài nguyên hệ thống, trừ khi nó được người dùng trao quyền truy cập một cách công khai khi cài đặt. Trước khi cài đặt ứng dụng, Cửa hàng Play sẽ hiển thị tất cả các quyền mà ứng dụng đòi hỏi: ví dụ như một trò chơi cần phải kích hoạt bộ rung hoặc lưu dữ liệu vào thẻ nhớ SD, nhưng nó không nên cần quyền đọc tin nhắn SMS hoặc tiếp cận danh bạ điện thoại. Sau khi xem xét các quyền này, người dùng có thể chọn đồng ý hoặc từ chối chúng, ứng dụng chỉ được cài đặt khi người dùng đồng ý.

Hệ thống hộp cát và hỏi quyền làm giảm bớt ảnh hưởng của lỗi bảo mật hoặc lỗi chương trình có trong ứng dụng, nhưng sự bối rối của lập trình viên và tài liệu hướng dẫn còn hạn chế đã dẫn tới những ứng dụng hay đòi hỏi những quyền không cần thiết, do đó làm giảm đi hiệu quả của hệ thống này. Một số công ty bảo mật, như Lookout Mobile Security, AVG Technologies, và McAfee, đã phát hành những phần mềm diệt virus cho các thiết bị Android. Phần mềm này không có hiệu quả vì cơ chế hộp cát vẫn áp dụng vào các ứng dụng này, do vậy làm hạn chế khả năng quét sâu vào hệ thống để tìm nguy cơ.

Một nghiên cứu của công ty bảo mật Trend Micro đã liệt kê tình trạng lạm dụng dịch vụ trả tiền là hình thức phần mềm ác ý phổ biến nhất trên Android, trong đó tin nhắn SMS sẽ bị gửi đi từ điện thoại bị nhiễm đến một số điện thoại trả tiền mà người dùng không hề hay biết. Loại phần mềm ác ý khác hiển thị những quảng cáo không mong muốn và gây khó chịu trên thiết bị, hoặc gửi thông tin cá nhân đến bên thứ ba khi chưa được phép. Đe dọa bảo mật trên Android được cho là tăng rất nhanh theo cấp số mũ; tuy nhiên, các kỹ sư Google phản bác rằng hiểm họa từ phần mềm ác ý và virus đã bị thổi phồng bởi các công ty bảo mật nhằm mục đích thương mại, và buộc tội ngành công nghiệp bảo mật đang lợi dụng sự sợ hãi để bán phần mềm diệt virus cho người dùng. Google vẫn giữ quan điểm rằng phần mềm ác ý thật sự nguy hiểm là cực kỳ hiếm, và một cuộc điều tra do F-Secure thực hiện cho thấy chỉ có 0,5% số phần mềm ác ý Android là len vào được cửa hàng Google Play.

Google hiện đang sử dụng bộ quét phần mềm ác ý Google Bouncer để theo dõi và quét các ứng dụng trên Cửa hàng Google Play. Nó sẽ đánh dấu các phần mềm bị nghi ngờ và cảnh báo người dùng về những vấn đề có thể xảy ra trƣớc khi họ tải nó về máy. Android phiên bản 4.2 Jelly Bean được phát hành vào năm 2012 cùng với các tính năng bảo mật được cải thiện, bao gồm một bộ quét phần mềm ác ý được cài sẵn trong hệ thống, hoạt động cùng với Google Play nhưng cũng có thể quét các ứng dụng được cài đặt từ nguồn thứ ba, và một hệ thống cảnh báo sẽ thông báo cho người dùng khi một ứng dụng cố gắng gửi một tin nhắn vào số tính tiền, chặn tin nhắn đó lại trừ khi người dùng công khai cho phép nó.

Điện thoại thông minh Android có khả năng báo cáo vị trí của điểm truy cập Wi-Fi, phát hiện ra việc di chuyển của người dùng điện thoại, để xây dựng những cơ sở dữ liệu có chứa vị trí của hàng trăm triệu điểm truy cập. Những cơ sở dữ liệu này tạo nên một bản đồ điện tử để tìm vị trí điện thoại thông minh, cho phép chúng chạy các ứng dụng như Foursquare, Google Latitude, Facebook Places, và gửi những đoạn quảng cáo dựa trên vị trí. Phần mềm theo dõi của bên thứ ba như TaintDroid, một dự án nghiên cứu trong trường đại học, đôi khi có thể biết được khi nào thông tin cá nhân bị gửi đi từ ứng dụng đến các máy chủ đặt ở xa.

Bản chất mã nguồn mở của Android cho phép những nhà thầu bảo mật lấy những thiết bị sẵn có rồi điều chỉnh để sử dụng ở mức độ bảo mật cao hơn. Ví dụ như Samsung đã cộng tác với General Dynamics sau khi họ thâu tóm Open Kernel Labs để xây dựng lại Jellybean trên nền bộ vi kiểm soát dành cho dự án "Knox".

## 1.10 GIẤY PHÉP PHÁT HÀNH

Mã nguồn của Android được cấp phép theo các giấy phép phần mềm mã nguồn mở tự do. Google đưa phần lớn mã nguồn (bao gồm cả các lớp mạng và điện thoại) theo Giấy phép Apache phiên bản 2.0, và phần còn lại, các thay đổi đối với nhân Linux, theo Giấy phép Công cộng GNU phiên bản 2. Liên minh Thiết bị cầm tay mở đã thực hiện các thay đổi trên nhân Linux, với mã nguồn lúc nào cũng công khai. Phần còn lại của Android được Google phát triển một mình, và mã nguồn chỉ được công bố khi phát hành một phiên bản mới. Thông thường Google cộng tác với một nhà sản xuất phần cứng để cung cấp một thiết bị 'chủ lực' (thuộc dòng Google Nexus) với phiên bản mới nhất của Android, sau đó phát hành mã nguồn sau khi thiết bị này được bán ra.

Vào đầu năm 2011, Google quyết định tạm ngưng phát hành mã nguồn Android phiên bản 3.0 Honeycomb dành riêng cho máy tính bảng. Lý do, theo Andy Rubin trong một bài blog Android chính thức, là vì Honeycomb đã được làm gấp gáp để phục vụ cho Motorola Xoom, và họ không muốn các bên thứ ba tạo ra một "trải nghiệm người dùng cực kỳ tồi tệ" bằng cách cố gắng đưa vào điện thoại thông minh một phiên bản dành riêng cho máy tính bảng. Mã nguồn một lần nữa được xuất bản công khai vào tháng 11 năm 2011 với sự ra mắt của Android 4.0.

Mặc dù phần mềm là mã nguồn mở, các nhà sản xuất thiết bị không thể sử dụng thương hiệu Android của Google trừ khi Google chứng nhận rằng thiết bị của họ phù hợp với Tài liệu Định nghĩa Tương thích (Compatibility Definition Document - CDD). Các thiết bị cũng phải thỏa mãn định nghĩa này thì mới được cấp phép để cài các ứng dụng mã nguồn đóng của Google, gồm cả Google Play. Vì Android không hoàn toàn được phát hành theo giấy phép tương thích GPL, ví dụ như mã nguồn của Google là theo giấy phép Apache license, và cũng vì Google Play cho phép các phần mềm có bản quyền, Richard Stallman và Quỹ phần mềm tự do luôn chỉ trích Android và khuyên người dùng sử dụng hệ điều hành khác như Replicant.

## 1.11 ĐÓN NHẬN

Android được đón nhận bằng một thái độ thờ ơ khi ra mắt vào năm 2007. Mặc dù những nhà phân tích rất ấn tượng với việc các công ty công nghệ có tiếng tăm hợp tác cùng Google để tạo ra Liên minh thiết bị di động mở, người ta vẫn không rõ liệu các nhà sản xuất có sẵn sàng thay thế hệ điều hành mà họ đang dùng bằng Android hay không. Ý tưởng về một nền tảng phát triển mã nguồn mở dựa trên Linux đã thu hút sự quan tâm, nhưng cũng dấy lên những lo ngại rằng Android sẽ phải đối mặt với sự cạnh tranh mạnh mẽ từ những tay chơi có hạng trong thị trường điện thoại thông minh, như Nokia và Microsoft, và các hệ điều hành di động đối thủ cũng sử dụng Linux đang trong quá trình phát triển. Những công ty hàng đầu không giấu sự hoài nghi: Nokia được trích nói rằng "chúng tôi không xem đó là một sự đe dọa," và một thành viên của nhóm Windows Mobile của Microsoft nói rằng "tôi không hiểu rồi họ sẽ có tác động ra sao".

Kể từ đó Android đã phát triển để trở thành hệ điều hành dành cho điện thoại thông minh phổ biến nhất trên thế giới và là "một trong những trải nghiệm di động nhanh nhất hiện nay." Các nhà bình luận thì nhấn mạnh vào bản chất mã nguồn mở của hệ điều hành chính là một trong những yếu tố quyết định sức mạnh, cho phép các công ty như (Kindle Fire), Barnes & Noble (Nook), Ouya, Baidu, và những hãng khác đổi hướng phần mềm và phát hành những phần cứng chạy trên phiên bản Android đã thay đổi của riêng họ. Kết quả, nó được trang web công nghệ Ars Technica mô tả là "đương nhiên là hệ điều hành mặc định khi phát hành phần cứng mới" cho những công ty không có nền tảng di động riêng của họ. Chính sự mở và uyển chuyển này cũng hiện diện ở cấp độ ngƣời dùng cuối: Android cho phép người dùng điện thoại điều chỉnh thoải mái thiết bị của họ và ứng dụng thì có sẵn trên các cửa hàng ứng dụng và trang web không phải của Google. Những đặc điểm này được xem là đóng góp vào những thế mạnh chính của điện thoại Android so với các điện thoại khác.

Android cũng bị phê phán vì thiếu sự hỗ trợ hậu mãi từ nhà sản xuất và nhà mạng, nếu so sánh với iOS của Apple. Với những thiết bị không mang nhãn hiệu Nexus, nhà mạng luôn kiểm tra các tiêu chuẩn của họ rồi thực hiện thay đổi cho riêng từng thiết bị (bắt nguồn từ sự điều chỉnh của nhà sản xuất và sự đa dạng của thiết bị Android) được xem là tác nhân chính trì hoãn việc cập nhật. Những nhà bình luận cũng nói rằng ngành công nghiệp thiết bị di động vì lý do lợi nhuận đã cố tình không cập nhật thiết bị của họ, vì thiếu cập nhật trên thiết bị hiện tại sẽ thúc đẩy việc mua thiết bị mới.

## 1.12 KIẾN TRÚC CỦA ANDROID

"Understanding Android" là cách mà ta tiếp cận lập trình Android và thấu hiểu kiến trúc hệ thống của nó. Chúng ta có thể không cần hiểu rõ cấu trúc của một HDH nhưng Chúng ta vẫn có thể lập trình một ứng dụng trên HDH đó, đây là điều mà nhà sản xuất muốn khi release SDK với một framework có sẵn của họ. Như chúng ta biết điều này cũng có mặt tốt và xấu. Framework là một tầng cao cấp dành cho lập trình viên, nó đều có giới hạn của nó, chúng ta có thể chỉ có thể lập trình những ứng dụng phổ biến nhưng không nên tiến tới những ứng dụng cao cấp đi sâu vào hệ thống của HDH. Theo cách của mình, trước khi bắt đầu học Android, chúng ta nên nghiên cứu qua bản thân HDH Android, chúng ta không cần phải hiểu rõ nó như thể nào, mục đích quan trọng nhất của chúng ta là có cái nhìn chung và toàn diện nhất về Android.

### 1.12.1 Android platform

Bao gồm HDH Android đầy đủ tính năng, các ứng dụng và các tầng trung gian để developer có thể mở rộng, tùy chỉnh hoặc thêm vào các component của họ.

Có 4 tầng cơ bản trong HDH Android: Application Framework, Android Runtime, Native Libraries, Linux Kernel ... Mỗi tầng làm việc đều nhờ sự giúp đỡ của tầng bên dưới.

### 1.12.2 Tầng linux kernel

Đây là nhân của HDH Android, mọi xử lý của hệ thống đều phải thông qua tầng này. Linux Kernel cung cấp các trình điều khiển thiết bị phần cứng (driver) như: camera, USB, Wifi, Bluetooth, Display, Power Management ... Android dựa trên Linux phiên bản 2.6 lựa chọn các tính năng cốt lõi như bảo mật, quản lý bộ nhớ, quản lý tiến trình, mạng stack và các trình điều khiển phần cứng. Kernel hoạt động như một lớp trừu tượng giữa phần cứng và phần mềm còn lại của hệ thống.

### 1.12.3 Native libraries

* System C library - có nguồn gốc từ hệ thống thư viện chuẩn C (libc), điều chỉnh các thiết bị nhúng trên Linux.
* Media Libraries - mở rộng từ PacketVideo's OpenCORE; thư viện hỗ trợ playback và recording của nhiều định dạng video và image phổ biến: MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
* Surface Manager - quản lý việc hiển thị và kết hợp đồ họa 2D và 3D.
* LibWebCore - Android dùng lại webkit engine cho việc render trình duyệt mặc định của HDH Android browser và cho dạng web nhúng (như HTML nhúng)
* SGL - 2D engine
* 3D libraries - Thư viện 3D dựa trên OpenGL ES 1.0 API, có nâng cấp tăng tốc "hardware 3D acceleration"
* FreeType - render bitmap và vector font.
* SQLite - quản lý database của ứng dụng.

### 1.12.4 Tầng runtime

Mỗi ứng dụng Android chạy trên một proccess riêng của Dalvik VM (máy ảo). Dalvik được viết để chạy nhiều máy ảo cùng một lúc một cách hiệu quả trên cùng một thiết bị.

Máy ảo Dalvik thực thi các file mang định dạng .dex (Dalvik Excutable), định dạng này là định dạng đã được tối ưu hóa để chỉ chiếm một vùng nhớ vừa đủ xài và nhỏ nhất có thể. VM chạy các class (đã được compile trước đó bởi 1trình biên dịch ngôn ngữ Java), sở dĩ VM chạy được các class này là nhờ chương trình DX tool đã convert các class sang định dạng .dex.

### 1.12.5 Tầng application framework

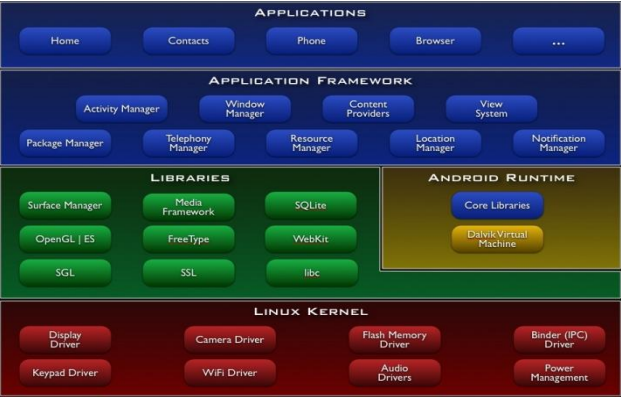
Đây là tầng mà Google xây dựng cho các developer để phát triển các ứng dụng của họ trên Android, chỉ bằng cách gọi các API có sẵn mà Google đã viết để sử dụng các tính năng của phần cứng mà không cần hiểu cấu trúc bên dưới.

Bằng cách cung cấp một nền tảng phát triển mở, Android 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 thức, thêm các thông báo để các thanh trạng thái, và nhiều, nhiều hơn nữa.

Tất cả các ứng dụng thường gồm một bộ các dịch vụ và hệ thống cơ bản sau:

* View UI dùng để xây dựng layout của ứng dụng bao gồm: list view, text field, button, dialog, form ...
* Content Providers cho phép các ứng dụng có thể truy cập dữ liệu từ các ứng dụng khác (như ứng dụng của ta có thể lấy thông tin Contacts của điện thoại Android), hoặc để chia sẻ dữ liệu của riêng ứng dụng
* Resource Manager cung cấp cách thức truy cập đến non-code resources như các asset, graphic, image, music, video ...
* Notification Manager cho phép tất cả các ứng dụng hiển thị thông báo của mình trên HDH.
* Activity Manager quản lý vòng đời của các ứng dụng.

Ở góc nhìn của người dùng ta có thêm tầng application (là ứng dụng do chúng ta viết), sau đây là sơ đồ tổng quát như hình 1.1:



*Hình 1.1 Kiến trúc android*

# CHƯƠNG 2: MÔI TRƯỜNG LẬP TRÌNH

## 2.1 GIỚI THIỆU

Trong chương này sẽ giới thiệu các công cụ lập trình cho Android (Android Development Tools). Chúng ta sẽ dần làm quen với Eclipse và Android Development Tool plug-in, thông qua Android SDK và công cụ của nó, chạy một ứng dụng Android trên Emulator(tạm dịch là trình giả lập giao diện của Android trên PC). Với những kĩ năng đó trong tay, chúng ta sẽ tìm hiểu các gói Java packages được cung cấp trong SDK giúp nâng cao khả năng lập trình trên Android.

## 2.2 ANDROID SDK

Android SDK, viết tắt của Android Software Development Kit, có thể gọi là “công cụ phát triển phần mềm” mà cụ thể ở đây là phát triển ứng dụng cho Android OS. Đến đây thì chắc các chúng ta cũng đã hiểu đơn giản Google Adroid SDK là bộ công cụ được chính Google xây dựng và phát hành miễn phí đến giới Developer để họ dễ dàng xây dựng và phát triển các ứng dụng chạy được trên Android OS.

Android SDK có các đặc tính sau:

* Được phát triển và cung cấp miễn phí
* Truy cập đến phần cứng Wi-Fi
* GSM, EDGE và tính năng mạng 3G cho phép việc chuyển dữ liệu, gọi điện hay gửi tin SMS trong mạng di động.
* Gói API toàn diện cho các dịch vụ nền tảng ví dụ như GPS
* Truy cập toàn bộ phần cứng điều khiển đa phương tiện như chơi nhạc, ghi âm hay sử dụng microphone và camera
* Chia sẻ dữ liệu trong kho dữ liệu.
* Tích hợp trình duyệt dựa trên bộ WebKit (mã nguồn mở)
* P2P hổ trợ sử dung Google Talk.
* Hỗ trợ tốt đồ họa 3D nhờ sử dụng OpenGL ES.

Android SDK cần phải được cài đặt để xây dựng các ứng dụng Android. Android SDK bao gồm tài liệu tích hợp HTML\_base lõi chính của Javadoc để thể hiện các gói và lớp trong Java

## 2.3 MÁY ẢO DALVIK

Một trong những thành phần quan trọng của Android là máy ảo Dalvik. Thay vì sử dụng máy ảo Java như trước kia Android sử dùng máy ảo của riêng nó được thiết kế để bảo đảm rằng đa ứng dụng có thể chạy mượt mà trên một thiết bị di động.

Máy ảo Dalvik sử dụng lõi Linux để xử lý các chức năng ở mức thấp bao gồm bảo mật, các tiến trình, các luồng và quản lý vùng nhớ. Nó cũng có thể viết bằng ứng dụng C/C++ để chạy trực tiếp lên lõi Linux bên dưới. Giữa phần cứng và các dịch vụ hệ thống được quản lý bởi máy ảo Dalvik, nó là một thành phần ở giữa. Bằng cách sử dụng máy ảo này để chạy ứng dụng, các nhà phát triển hoàn toàn không phải bận tâm gì về các phần cứng bên dưới. Khi máy ảo này chạy nó sẽ tạo ra tập tin có đuôi là .dex, tập tin này được SDK tạo ra bởi sự chuyển đổi từ các lớp biên dịch ngôn ngữ Java

## 2.4 CÁC GÓI JAVA CẦN THIẾT

- java.lang - gói chứa các lớp lõi của Java

- java.io - xuất nhập.

- java.net - kết nối mạng.

- java.util - chứa các lớp tiện ích. Bao gồm: Log thường được dùng đề viết LogCat.

- java.text - tiện ích sử lý văn bản

- java.math - các lớp toán học.

- javax.net - các lớp mạng.

- javax.security - Security-related classes.

- javax.xml - các lớp liên quan đến DOM-based XML.

- org.apache.\* - các lớp liên quan đến HTTP-related

- org.xml - các lớp liên quan đến SAX-based XML

Các gói dùng cho lập trình Android:

* Android.app - Android application model access
* Android.content - truy xuất dữ liệu trong Android
* Android.net - bao gồm Uri class dùng cho việc truy xuất các nội dung khác
* Android.graphics - đồ họa
* Android.opengl - các lớp OpenGL
* Android.os - truy cập ở mức độ hệ thông đến môi trường Android
* Android.provider - các lớp liên quan đến ContentProvider
* Android.telephony - khả năng truy cập Telephony
* Android.text - Text layout
* Android.util - tập các tiện thao tác trên văn bản,bao gồm XML
* Android.view - thành phần UI (giao diện người dùng)
* Android.webkit - Browser functionality
* Android.widget - thành phần UI mức độ cao hơn

Trong đó các gói gần như là lõi của mọi ứng dụng trên Android đó là Android.app, Android.view và Android.content.

## 2.5 THÀNH PHẦN CHÍNH CỦA ECLIPSE KHI TẠO ỨNG DỤNG TRÊN ANDROID

Môi trường phát triển Rich Java bao gồm trình biên dịch, class autocompletion, và tích hợp Javadoc

Bẫy lỗi code

Quản lý và chạy Android Emulator profile

The Dalvik Debug Monitoring Service (DDMS)

Khung nhìn các tiến trình (Thread and heap views)

Quản lý Hệ thống tập tin giao diện (Emulator filesystem management)

Điều khiển dữ liệu và hội thoại (Data and voice network control)

Điều khiển giao diện (Emulator control)

Ghi lỗi hệ thống và ứng dụng (System and application logging)

## 2.6 THÀNH PHẦN QUAN TRỌNG TRONG MỘT ANDROID PROJECT

Activity (Android.app.Activity): đây là lớp khởi tạo giao diện ứng dụng nội bộ trên Android tương tư như MIDlet trong J2ME.

Service (Android.app.Service): cung cấp các dịch vụ liên quan đến client/service. Một Service sẽ chạy ngầm bên dưới, sau đó các client (Activity) sẽ kết nối và truy xuất các hàm trên dịch thông qua Interface class.

Broadcast receiver (Android.content.BroadcastReceiver): đây là một ứng dụng chạy ngầm dùng để đọc và cập nhật thông tin trên UI, ví dụ như cập nhật sự thay đỗi giờ, pin...

Content Provider: cung cấp chức năng truy vấn dữ liệu giữa các ứng dụng của Android.

Intent: nền tảng để truyền tải các thông báo. Intent được sử dụng để gửi các thông báo đi nhằm khởi tạo 1 Activity hay Service để thực hiện công việc mà chúng ta mong muốn

## 2.7 CHU KỲ SỐNG CỦA ỨNG DỤNG ANDROID

Một tiến trình Linux gói gọn một ứng dụng Android đã được tạo ra cho ứng dụng khi codes cần được xây dựng, khởi chạy và sẽ còn chạy cho đến khi:

* Nó không phụ thuộc.
* Hệ thống cần lấy lại bộ nhớ mà nó chiếm giữ cho các ứng dụng khác.

Một sự khác thường và đặc tính cơ bản của Android là thời gian sống của tiến trình ứng dụng không được điều khiển trực tiếp bởi chính nó. Thay vào đó, nó được xác định bởi hệ thống qua một kết hợp của:

* Những phần của ứng dụng mà hệ thống biết đang chạy
* Những phần đó quan trọng như thế nào đối với người dung
* Bao nhiêu vùng nhớ chiếm lĩnh trong hệ thống.

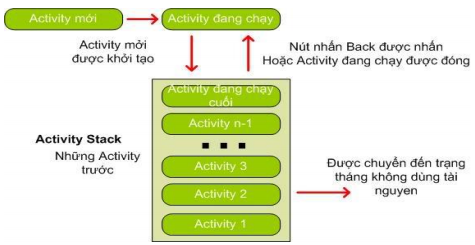
### 2.7.1Chu kỳ sống thành phần

Các thành phần ứng dụng có một chu kỳ sống, tức là mỗi thành phần từ lúc bắt đầu khởi tạo và đến thời điểm kết thúc, đôi lúc chúng có thể là active (visible hoặc invisible) hoặc inactive.

### 2.7.2Activity stack

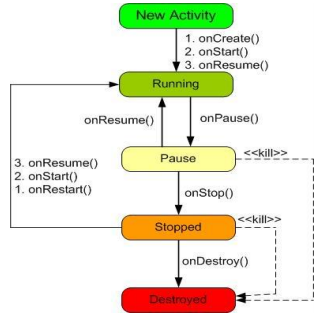
Bên trong hệ thống các activity được quản lý như một activity stack. Khi một Activity mới được start, nó được đặt ở đỉnh của stack và trở thành activity đang chạy activity trước sẽ ở bên dưới activity mới và sẽ không thấy trong suốt quá trình activity mới tồn tại.

Nếu người dùng nhấn nút Back thì activity kết tiếp của stack sẽ di duyển lên và trở thành active.



*Hình 2.1 Activity stack*

### 2.7.3 Các trạng thái của chu kỳ sống



*Hình 2.2 Chu kỳ sống của Activity*

Một Activity chủ yếu có 4 chu kỳ chính sau:

* Active hoặc running: Khi Active là được chạy trên màn hình. Activity này tập trung vào những thao tác của người dùng trên ứng dụng.
* Paused: Activity là được tạm dừng (paused) khi mất focus nhưng người dùng vẫn trông thấy. Có nghĩa là một Activity mới ở trên nó nhưng không bao phủ đầy màn hình. Một Activity tạm dừng là còn sống nhưng có thể bị kết thúc bởi hệ thống trong trường hợp thiếu vùng nhớ.
* Stopped: Nếu nó hoàn toàn bao phủ bởi Activity khác. Nó vẫn còn trạng thái và thông tin thành viên trong nó. Người dùng không thấy nó và thường bị loại bỏ trong trường hợp hệ thống cần vùng nhớ cho tác vụ khác.
* Killed: Khi hệ thống bị thiếu bộ nhớ, nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái stop hoặc paused cũng có thể bị giải phóng và khi nó được hiển thị lại thì các Activity này phải khởi động lại hoàn toàn và phục hồi lại trạng thái trước đó.

### 2.7.4 Chu kỳ sống của ứng dụng

Trong một ứng dụng Android có chứa nhiều thành phần và mỗi thành phần đều có một chu trình sống riêng. Và ứng dụng chỉ được gọi là kết thúc khi tất cả các thành phần trong ứng dụng kết thúc. Activity là một thành phần cho phép người dùng giao tiếp với ứng dụng. Tuy nhiên, khi tất cả các Activity kết thúc và người dùng không còn giao tiếp được với ứng dụng nữa nhưng không có nghĩa là ứng dụng đã kết thúc. Bởi vì ngoài Activity là thành phần có khả năng tương tác người dùng thì còn có các thành phần không có khả năng tương tác với người dùng như là Service, Broadcast receiver. Có nghĩa là những thành phần không tương tác người dùng có thể chạy background dưới sự giám sát của hệ điều hành cho đến khi người dùng tự tắt chúng.

### 2.7.5 Các sự kiện trong chu kỳ sống của ứng dụng

Nếu một Activity được tạm dừng hoặc dừng hẳn, hệ thống có thể bỏ thông tin khác của nó từ vùng nhớ bởi việc gọi hàm finish() của nó, hoặc đơn giản giết tiến trình của nó. Khi nó được hiển thị lần nữa với người dùng, nó phải được hoàn toàn restart và phục hồi lại trạng thái trước. Khi một Activity chuyển qua chuyển lại giữa các trạng thái, nó phải báo việc chuyển của nó bằng việc gọi hàm transition.



*Hình 2.3*

Tất cả các phương thức là những móc nối mà chúng ta có thể override để làm tương thich công việc trong ứng dụng khi thay đổi trạng thái. Tất cả các Activity bắt buộc phải có onCreate() để khởi tạo ứng dụng. Nhiều Activity sẽ cũng hiện thực onPause() để xác nhận việc thay đổi dữ liệu và mặt khác chuẩn bị dừng hoạt động với người dung.

### 2.7.6 Thời gian sống của ứng dụng

Bảy phương thức chuyển tiếp định nghĩa trong chu kỳ sống của một Activity. Thời gian sống của một Activity diễn ra giữa lần đầu tiên gọi onCreate() đến trạng thái cuối cùng gọi onDestroy(). Một Activity khởi tạo toàn bộ trạng thái toàn cục trong onCreate(), và giải phóng các tài nguyên đang tồn tại trong onDestroy().

### 2.7.7 Thời gian hiển thị của activity

Visible lifetime của một activity diễn ra giữa lần gọi một onStart() cho đến khi gọi onStop(). Trong suốt khoảng thời gian này người dùng có thể thấy activity trên màn hình, có nghĩa là nó không bị foreground hoặc đang tương tác với người dùng. Giữa 2 phương thức người dùng có thể duy trì tài nguyên để hiển thị activity đến người dùng.

### 2.7.8 Các hàm thực thi của activity

- OnCreate(...): hàm này được gọi khi lớp Activity được khởi tạo, dùng để thiết lập giao diện ứng dụng và thực thi những thao tác cơ bản.

- onStart(): hàm này được gọi khi lớp ứng dụng xuất hiện trên màn hình.

- onResume(): hàm được gọi ngay sau OnStart hoặc khi người dùng focus ứng dụng, hàm này sẽ đưa ứng dụng lên top màn hình.

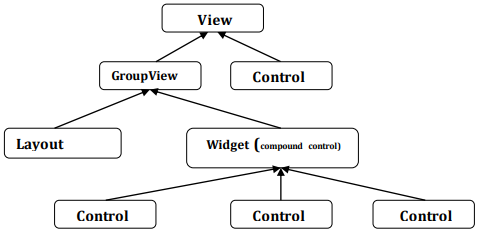
- onPause(): hàm được gọi khi hệ thống đang focus đến 1 activity trước đó.

- onStop(): hàm được gọi khi một activity khác được khởi động và focus.

- onRestart(): đưọc gọi khi ứng dụng chuyển sang onStop(), nhưng muốn khởi động lại bằng onStart().

2.8 THÀNH PHẦN GIAO DIỆN ANDROID

### 2.8.1 View



*Hình 2.4*

Trong Android giao diện người dùng được xây dựng từ các đối tượng View. Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc phân cấp như hình trên. 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 là setContentView (R.layout.main); hàm này sẽ load giao diện từ file MAIN.XML lên để phân tích thành mã bytecode

### 2.8.2 ViewGroup

ViewGroup là sự mở rộng của class View hay nói cách khác ViewGroup chính là các WidgetLayout được dùng để bố trí các đối tượng khác trong một screen.Có các loại ViewGroup như sau:

* Linear Layout

LinearLayout được dùng để bố trí các thành phần giao diện theo chiều ngang hoặc chiều dọc nhưng trên một line duy nhất mà không có xuống dòng.

* Frame layout

FrameLayout được dùng để bố trí các đối tượng theo kiểu giống như làcác Layer trong Photoshop. Những đối tượng nào thuộc Layer bên dưới thì sẽ bị che khuất bởi các đối tượng thuộc Layer nằm trên. FrameLayer thường được sử dụng khi muốn tạo ra các đối tượng có khung hình bên ngoài chẳng hạn như contact image button.

* Table layout

Layout này được sử dụng khi cần thiết kế một table chứa dữ liệu hoặc cần bố trí các widget theo các row và column. Chẳng hạn như, giao diện của một chiếc máy tính đơn giản hoặc một danh sách dữ liệu.

* AbsoluteLayout

Layout này được sử dụng để bố trí các widget vào một vị trí bất kì trong layout dựa vào 2 thuộc tính toạ độ x, y. Tuy nhiên, kiểu layout này rất ít khi được dùng bởi vì toạ độ của các đối tượng luôn cố định và sẽ không tự điều chỉnh được tỷ lệ khoảng cách giữa các đối tượng. Khi chuyển ứng dụng sang một màn hình có kích thước với màn hình thiết kế ban đầu thì vị trí của các đối tượng sẽ không còn được chính xác như ban đầu.

* RELATIVE LAYOUT

Layout này cho phép bố trí các widget theo một trục đối xứng ngang hoặc dọc. Để đặt được đúng vị trí thì các widget cần được xác định một mối ràng buộc nào đó với các widget khác. Các ràng buộc này là các ràng buộc trái, phải, trên, dưới so với một widget hoặc so với layout parent. Dựa vào những mối ràng buộc đó mà RetaliveLayout cũng không phụ thuộc vào kích thước của screen thiết bị. Ngoài ra, nó còn có ưu điểm là giúp tiết kiệm layout sử dụng nhằm mục đích giảm lượng tài nguyên sử dụng khi load đồng thời đẩy nhanh quá trình xử lý.

## 2.9 CÁC CONTROL

### 2.9.1 Các button

Sở dĩ widget button được giới thiệu đầu tiên trong số các widget khác là vì đây là đối tượng có thể nói là được dùng nhiều nhất trong hầu hết các ứng dụng Android.

### 2.9.2 Listview

Được sử dụng để thể hiện một danh sách các thông tin theo từng cell. Mỗi cell thông thường được load lên từ một file XML đã được cố định trên đó số lượng thông tin và loại thông tin cần được thể hiện.

Để thể hiện được một list thông tin lên một screen thì cần phải có 3 yếu tố chính:

* Data Source: Data Source có thể là một ArrayList, HashMap hoặc bất kỳ một cấu trúc dữ liệu kiểu danh sách nào.
* Adapter: Adapter là một class trung gian giúp ánh xạ dữ liệu trong Data Source vào đúng vị trí hiển thị trong ListView. Chẳng hạn, trong Data Source có một trường name và trong ListView cũng có một TextView để thể hiện trường name này. Tuy nhiên, ListView sẽ không thể hiển thị dữ liệu trong Data Source lên được nếu như Adapter không gán dữ liệu vào cho đối tượng hiển thị.
* ListView: ListView là đối tượng để hiển thị các thông tin trong Data Source ra một cách trực quan và người dùng có thể thao tác trực tiếp trên đó.

### 2.9.3 Edittext

Trong Android đối tượng EditText được sử dụng như một TextField hoặc một TextBox. Giá trị Android:singleLine bằng false, edittext sẽ là 1 Texbox, ngược lại nó là 1 Textfield.

### 2.9.4 Textview

TextView có tác dụng là để hiển thị văn bản.

### 2.9.5 Checkbox

Nhận hai giá trị true hoặc false. Đối tượng CheckBox cho phép chọn nhiều item cùng một lúc.

### 2.9.6 Menuoption

Có 2 cách tạo một MenuOptions:

* Tạo bằng code và tập tin XML
* Khai báo bằng code XML

### 2.9.7 Contextmenu

ContextMenu được sử dụng để hiển thị các tuỳ chọn khi người dùng nhấn dài vào một Button trên màn hình. Để tạo một ContextMenu ta cũng có 2 cách giống như tạo MenuOptions ở trên chỉ khác tên phương thức:

* ContextMenu: đối tượng để add các context menu item
* View: Đối tượng nơi mà xảy ra sự kiện
* ContextMenuInfo: Cho biết vị trí xảy ra sự kiện trong ListView.

## 2.10 TÙY BIẾN CONTROL (CUSTOM CONTROL)

Chúng ta đã biết Android đã cung cấp cho chúng ta rất nhiều Control hữu ích.Tuy nhiên ứng dụng của chúng ta đôi khi cũng cần những control do chính chúng ta tạo ra để đáp ứng các nhu cầu của ứng dụng mà việc sử dụng các control có sẵn rất khó khăn.

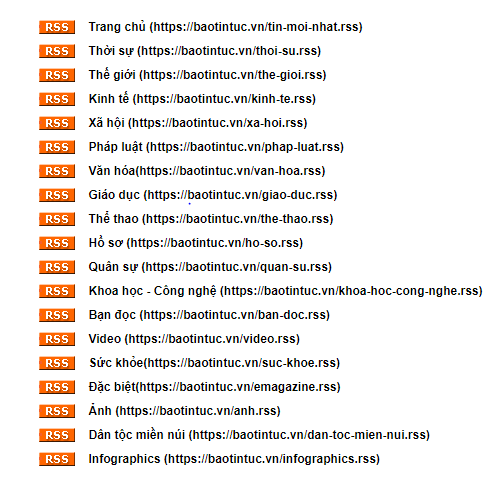
# CHƯƠNG 3: KỸ THUẬT ĐỌC LẤY NỘI DUNG RSS

## 3.1 TÌM HIỂU VỀ DỊCH VỤ RSS

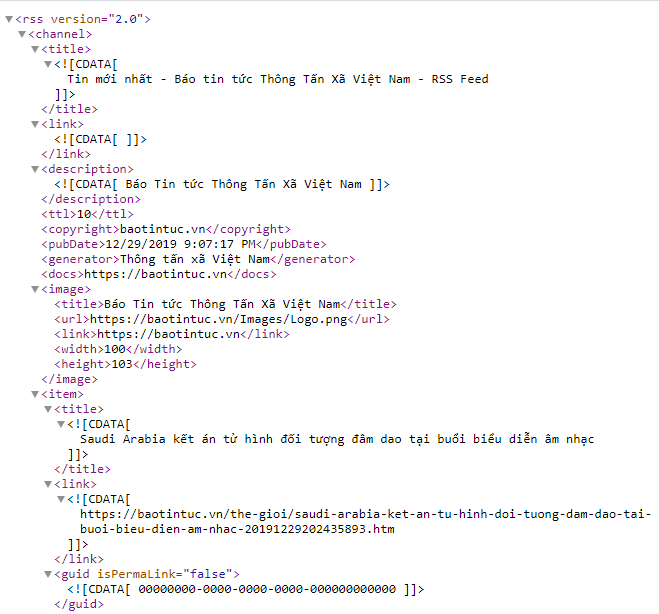
RSS ( Really Simple Syndication) là dịch vụ cung cấp thông tin cực kì đơn giản, giúp người dung có thể đọc được những tin tức cập nhật mới từ một hoặc nhiều website cùng lúc mà không cần vào trực tiếp trang web đó. Theo đó RSS không những đem lại lợi ích thiết thực cho người dùng mà còn là phương tiện quảng bá website tiết kiệm nhất, không cần đến những chiến lược quảng cáo rầm rộ tốn kém.

## 3.2 VÍ DỤ

Trang web Baotintuc.vn hiện tại cung cấp các nguồn kênh dữ liệu dưới đây theo định dạng chuẩn mới nhất RSS 2.0. Các nguồn kênh tin này là miễn phí cho việc sử dụng dưới mục đích cá nhân và phi lợi nhuận. Bạn chỉ việc copy và dán các địa chỉ URL này vào code của bạn để xử lý lấy dữ liệu tin tức là được.



Đây là ví dụ về định dạng RSS 2.0



## 3.3 KỸ THUẬT ĐỌC NỘI DUNG CỦA TRANG WEBSITE VỀ ĐIỆN THOẠI

Quy trình thực hiện:

* Đầu tiên tiến hành thêm thư viện dependence vào trong file build.gradle.

implementation 'com.squareup.picasso:picasso:2.5.2'

* Picaso: là thư viện Andriod mã nguồn mở rất phổ biến, dùng để load hoặc hiển thị hình ảnh trong ứng dụng Android.
* Tiếp theo là quyền cho phép ứng dụng lấy dữ liệu từ Internet trong AndroidManifest.

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

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

* Tiếp theo là tạo giao diện để hiển thị nội dung.
* Ở tl\_trangchu.xml tạo 1 listview
* Tạo 1activity\_listview\_nd.xml cho mỗi item hiển thị.

tl\_trangchu.xml

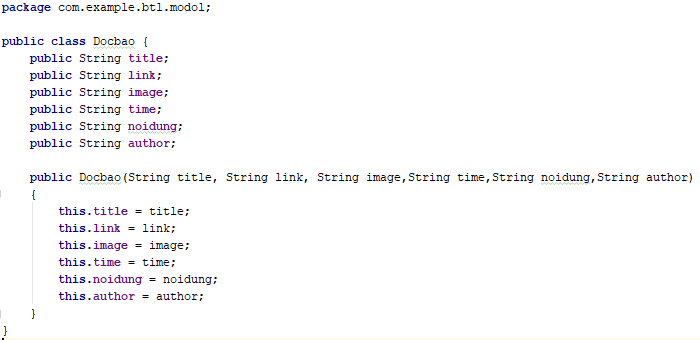


activity\_listview\_nd.xml



* Tiếp theo tạo 1 class Docbao để tạo ra đối tượng lưu dữ liệu sau khi bóc tách từng thành phần của XML.

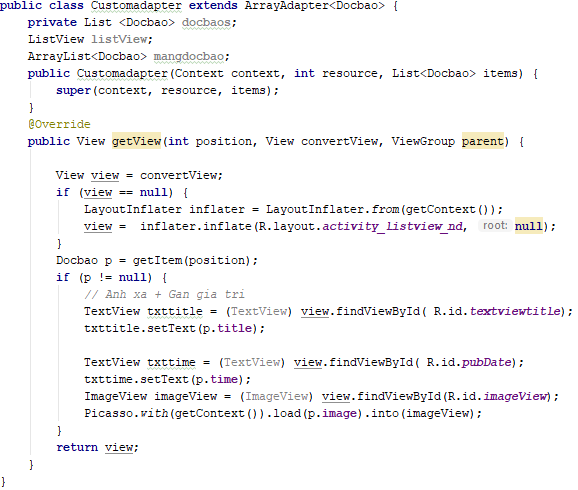
Docbao.java



Các thuộc tính của class gồm:

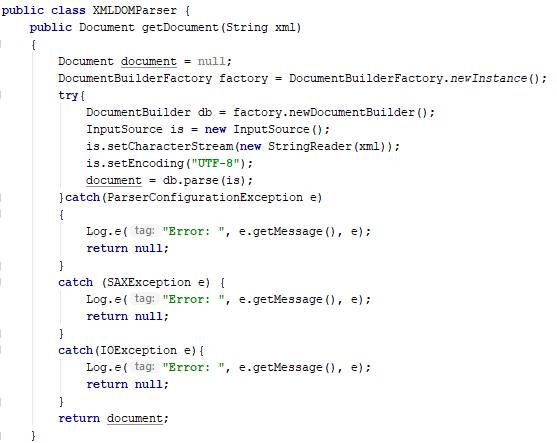
* Title : tiêu đề tin tức
* Link: đường dẫn trang chi tiết tin tức.
* Image: ảnh tin tức
* Time: ngày đăng
* Noidung: nội dung tóm tắt của tin
* Author: tên đầu báo
* Tạo class CustomAdapter để hiển thị dữ liệu
* Phương thức getView(), đây là phương thức để khởi tạo view để add nó vào Listview, ở đây nó sẽ tạo các row trên listview và set dữ liệu các thuộc tính trong Docbao.java lên các row đó.

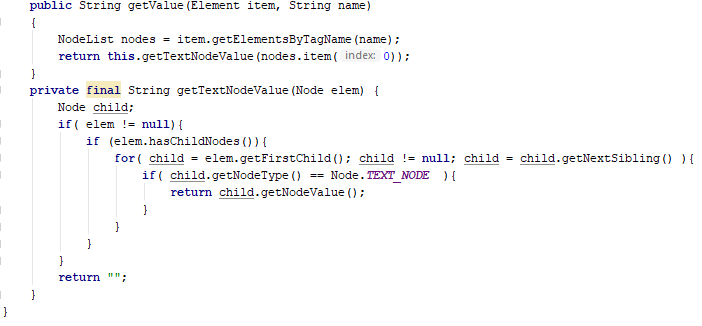
Customadapter.Java



* Vì khi gọi class Customadapter ở các class khác ta sẽ truyền cho nó 1 list Docbao nên dòng code trên sẽ có tác dụng là lấy ra từng đối tượng mới tương ứng với từng view, khi lấy được thuộc tính nó sẽ có được dữ liệu của 1 Docbao như : title, link,… và dữ liệu sẽ set lên row của listview.
* Tạo class XMLDOMParser.java
* Class này xử lý các hàm mặc định để đọc RSS

XMLDOMParser.java





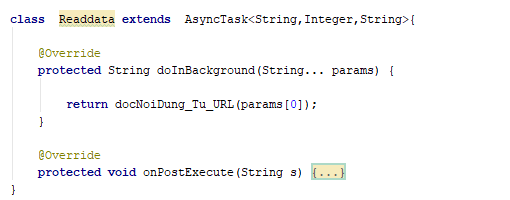
* Sử lý code để trong class chính để lấy dữ liệu.
* Code AsyncTask để lấy dữ liệu trên Internet
* Parse dữ liệu dạng Document sang Arraylist
* Set A dapter cho Listview
* Khởi tạo lớp kế thừa từ AsyncTaskvà chuyển đường dẫn cung cấp qua dịch vụ RSS để hiển thị lên giao diện.

Hàm onCreateView



* Tạo class Readdata Đọc và hiển thị nội dung RSS.

Class Readdata

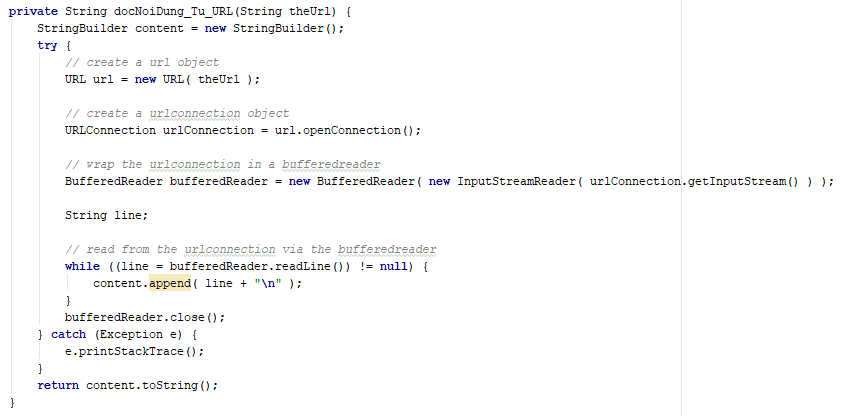


* Hàm onPostExcute trong class Readdata



* Hiển thị nội dung RSS
* Document là một biết thuộc class Document để lưu trữ những tài liệu đã được lấy về.Chúng ta có thể tự do trích xuất thông tin.

Hàm docNoiDung\_Tu\_URL



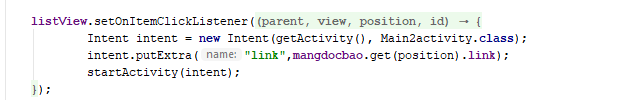
* Đọc nội dung từ nguồn RSS
* Kết quả sau khi thực hiện



*Hình 3.1*

* Để xem được chi tiết tin tức mà người dùng ấn vào ta sẽ phải làm các công việc sau
* Tạo một màn hình hiển thị trang chi tiết tin tức.
* Bắt sự kiện khi người người dùng ấn vào một dòng bất kỳ trên listview
* Sử dụng Intent để truyền dữ liệu từ màn hình này sang màn hình khác.
* Hiển thị dữ liệu trang web bằng webview

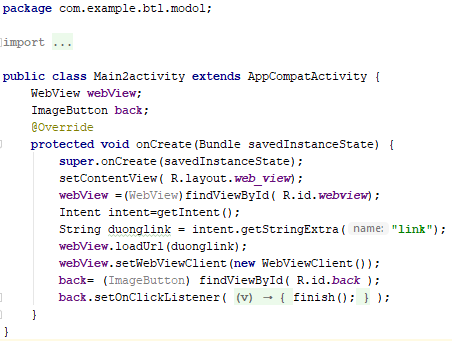
Hàm bắt sự kiện khi click để xem chi tiết.



Web\_view.xml



Main2activity.java(xử lý webview)



* Để hiển thị trang webview chỉ cần việc gọi hàm LoadUrl truyền vào link mà ta nhân được là xong.
* Cuối cùng là ta gọi hàm setWebViewClient (new WebviewClient()) để khi load trang web tin tức thì nó vẫn nằm gọn trong ứng dụng của ta chứ không nó sẽ bật trình duyệt để hiển thị.



*Hình 3.2*

# CHƯƠNG 4: CHƯƠNG TRÌNH THỰC NGHIỆM

## 4.1 PHÁT BIỂU BÀI TOÁN

Ngày nay, vai trò của các ứng dụng đọc tin tức trên thiết bị di động ngày càng trở nên quan trọng. khác với báo trí truyền thông có giới hạn cập nhật tin tức, các tờ báo trực tuyến đã cung cấp được sự tiện lợi trong việc cập nhật và phát triển thông tin. Về phía người dùng, họ xem thông tin mọi lúc mọi nơi với thiết bị di động của mình. Về phía người cung cấp thông tin, họ có thể dễ dàng cập nhật tin tức mới nhất, thời sự nhất thông qua dicchj vụ cung cấp thông tin cực kì đơn giản là RSS (Readlly Simple Syndication). Do đó việc sử dụng các ứng dụng đọc tin tức luôn là điều cần thiết nhất hiện nay, nhằm đáp ứng nhu cầu cập nhật thông tin của mỗi người.

## 4.2 GIAO DIỆN ỨNG DỤNG

* Giao diện màn hình chờ khi khởi động



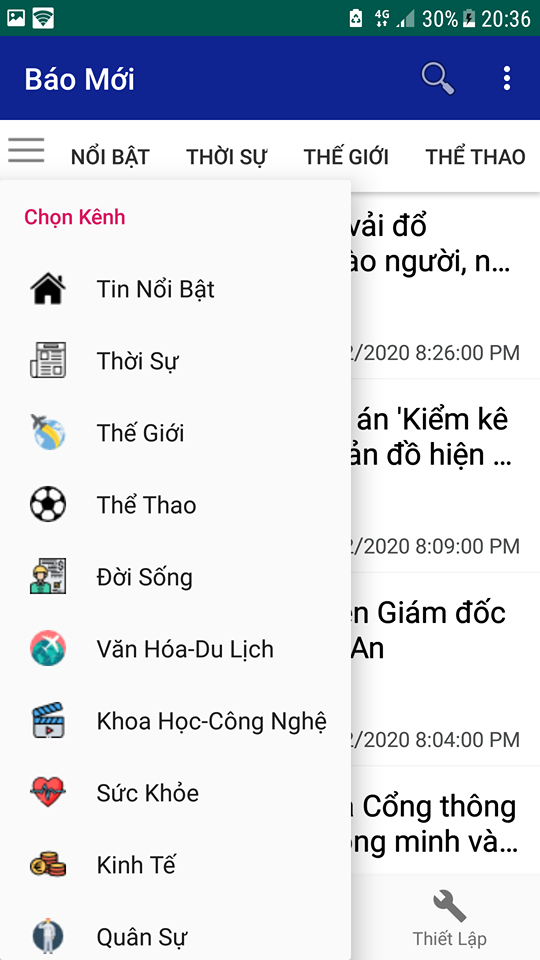
Hình 4. 1 Giao diện màn hình chờ

* Giao diện màn hình Chính



*Hình 4. 2 Giao diện màn hình chính*

* Giao diện màn hình menu kênh



Hình 4. 3 Giao diện MENU kênh

* Phần kiểm tra kết nối Internet



Hình 4.4 Giao diện khi không có internet

* Màn hình tìm kiếm tin tức



Hình 4.5 Giao diện tìm kiếm

* Giao diện hiển thị nội dung tin tức

|  |  |
| --- | --- |
| Hình 4. 6 Giao diện nội dung tin tức | Hình 4. 7 Giao diện nội dung tin tức |

# KẾT LUẬN

Trên đây em đã khảo sát trên mặt lý thuyết đối với xây dựng ứng dụng Android. Đồ án hướng tới mục tiêu xây dựng ứng dụng Android truy xuất cơ sở dữ liệu trên Internet. Trong khoảng thời gian nhất định dành cho việc thực hiện đề tài, nên một số vấn đề vẫn chưa được hoàn chỉnh. Tuy nhiên, đồ án đã đạt được một số kết quả.

Do thời gian hạn chế, nên hiện tại đồ án mới chỉ dừng lại ở thao tác với dữ liệu khi kết nối Internet ổn định còn ứng dụng chưa có chức năng lưu trữ dữ liệu offline cho phép người dùng xem lại dữ liệu khi ứng dụng mất kết nối Internet truy xuất được CSDL trên Internet. Ứng dụng chưa có chức năng upload ảnh cục bộ lên CSDL trên Internet. Trong tương lai em sẽ tiếp tục phát triển ứng dụng hoàn chỉnh hơn.

# TÀI LIỆU THAM KHẢO

Tài liệu tham khảo trực tuyến:

[1].[Online] http://o7planning.org/

[2].[Online] ttps://www.simplifiedcoding.net