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 : Mai Minh Việt

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 ÔN LUYỆN TRẮC NGHIỆM TIẾNG ANH

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Mai Minh Việt

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: Mai Minh Việt Mã SV: 1613101002

Lớp: CTL1001 Ngành: Công nghệ thông tin

Tên đề tài: Xây dựng ứng dụng Android ôn luyện trắc nghiệm tiếng Anh

Mục Lục

LỜI CẢM ƠN

Để đồ án này đạt kết quả tốt đẹp, em đã nhận được sự hỗ trợ, giúp đỡ của nhiều cơ quan, tổ chức, cá nhân. Với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến tất cả các cá nhân và cơ quan đã tạo điều kiện giúp đỡ trong quá trình học tập và nghiên cứu làm đồ án. Trước hết em xin gởi tới các thầy cô khoa Công nghệ Thông tin trường Đại học Dân Lập Hải Phòng lời chào trân trọng, lời chúc sức khỏe và lời cảm ơn sâu sắc. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của thầy cô, đến nay em đã có thể hoàn thành đồ án: "Xây dựng ứng dụng android ôn luyện trắc nghiệm tiếng anh". Đặc biệt em xin gửi lời cảm ơn chân thành nhất tới thầy giáo - Ths. Phùng Anh Tuấn đã quan tâm giúp đỡ, hướng dẫn em hoàn thành tốt đồ án này trong thời gian qua. Em xin bày tỏ lòng biết ơn đến ban lãnh đạo Trường Đại học Dân Lập Hải Phòng, Phòng Đào Tạo, các Khoa Phòng ban chức năng đã trực tiếp và gián tiếp tạo mọi điều kiện trong suốt quá trình học tập tại trường. Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế của một sinh viên, đồ án này không thể tránh được những thiếu sót. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy cô để em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công việc thực tế sau này.

Xin chân thành cảm ơn!

Sinh viên

Chương 1: GIỚI THIỆU 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 là 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, ước 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 người 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.

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 từchối 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 thương 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 như 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 đưa 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 phương 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 trưng 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 trước. 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 như 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, như 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 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 theoe 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 Android

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, thư 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 Kroah- Hartman, 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

Trong phiên bản 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 8.0 Oreo.

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 hưa 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 hỗtrợ 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 vàtài 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 trước 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 cóthể 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 người 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.

Chương 2: MÔI TRƯỜNG PHÁT TRIỂN ỨNG DỤNG ANDROID STUDIO

2.1. Sơ lược về Android Studio

Google cung cấp một công cụ phát triển ứng dụng Android trên Website chính thức dựa trên nền tảng IntelliJ IDEA gọi là Android Studio. Android studio dựa vào IntelliJ IDEA, là một IDE tốt cho nhất Java hiện nay. Do đó Android Studio sẽ là môi trường phát triển ứng dụng tốt nhất cho Android.

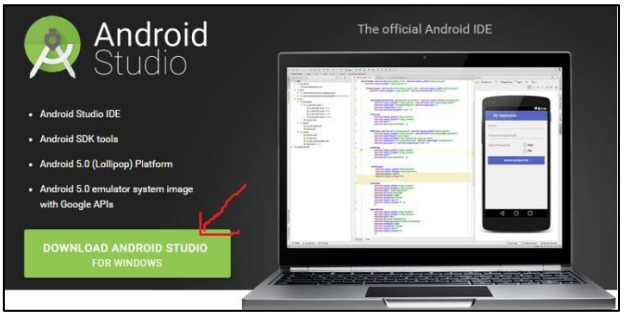
2.1.1. Cài đặt Android Studio

a. Yêu cầu phần cứng máy tính

- Microsoft® Windows® 10/8/7 (32 or 64-bit)
- 4 GB RAM. (Khuyến cáo là 8GB)
- Chip core I3 trở lên
- 400 MB hard disk space + ít nhất 1GB cho Android SDK, emulator
- Độ phân giải tối thiểu 1366 x 768

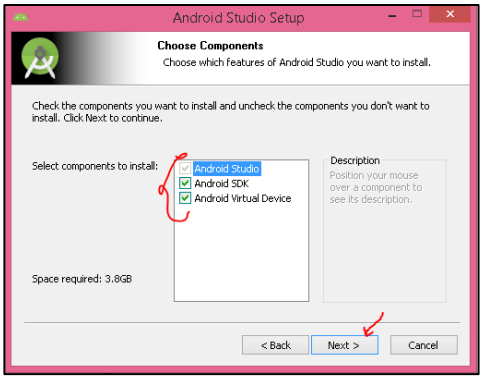
b. Phần mềm Android Studio

- Vào đường link: *http://developer.android.com/sdk/index.html*
- Để download bản mới nhất và tiến hành cài đặt click như hình:



Hình 2.1.1.1. Trang web tải Android studio

- Khi cài đặt chú ý chọn cả SDK và trình giả lập thiết bị ảo android như hình:



Hình 2.1.1.2 Giao diện cài đặt

- Tiếp tục chọn next và agree cho đến khi hoàn tất.

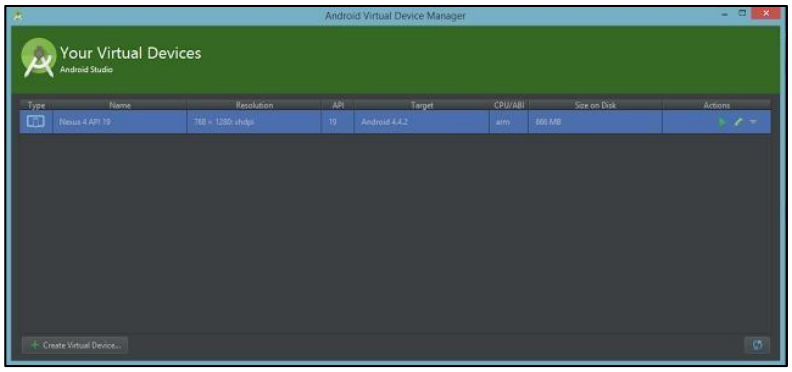
- Đây là màn hình khởi động.

Hình 2.1.1.3 Màn hình khởi động

- Các bước cài đặt thiết bị ảo trong android studio

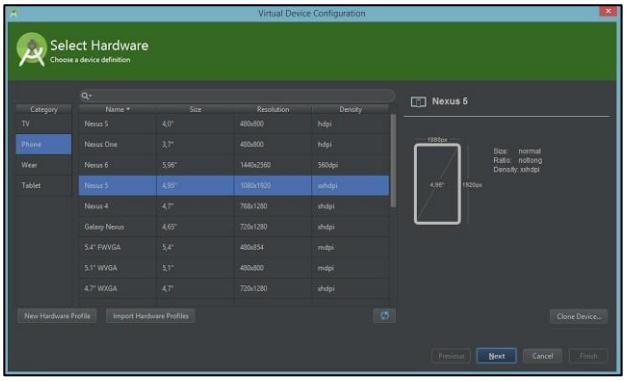
Máy ảo Android là một phần không thể thiếu khi chúng ta lập trình ứng dụng cho hệ điều hành Android, nó giúp chúng ta chạy thử ứng dụng ngay trên máy tính. Trong Android Studio có cung cấp cho chúng ta một máy ảo Android mặc định là Android Virtual Device viết tắt là AVD.

Để cài đặt máy ảo mở Android Studio lên và click vào nút **AVD Manager**. Cửa sổ AVD Manager sẽ xuất hiện:



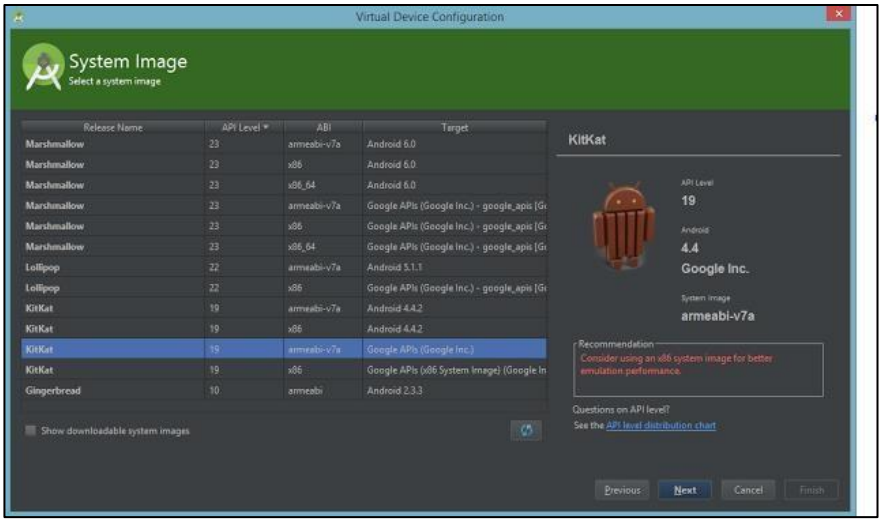
Hình 2.1.1.4. Giao diện cài máy ảo

Ở đây ta đang có sẵn một cái máy ảo, ta sẽ tạo thêm một cái máy ảo nữa bằng cách ấn vào nút **Create Virtual Device** ở góc dưới cùng bên trái. Sau đó sẽ hiển thị ra một cửa sổ nữa:



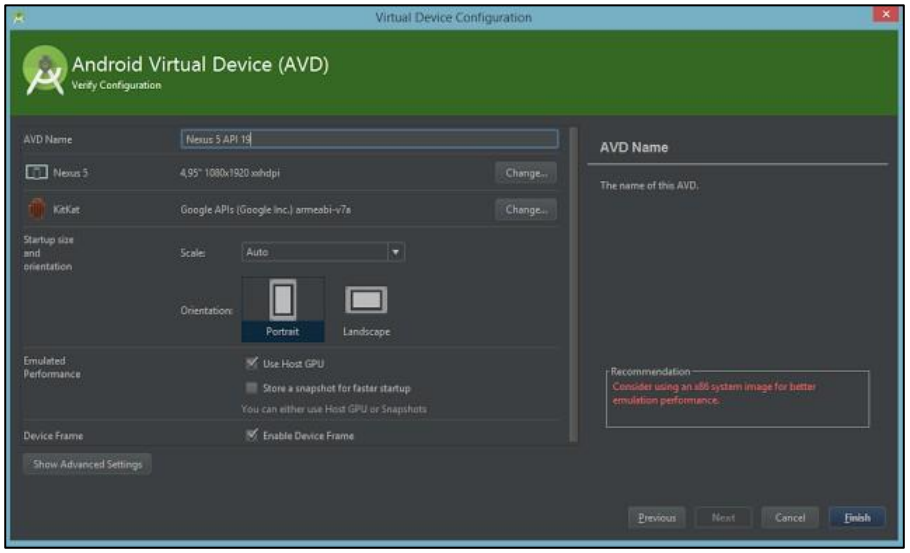
Hình 2.1.1.5. Giao diện chọn máy ảo

Ở đây có sẵn các mẫu điện thoại, ta chọn mẫu mà mình muốn, ví dụ chọn Nexus 5. Chọn xong bấm "Next" để tiếp tục:



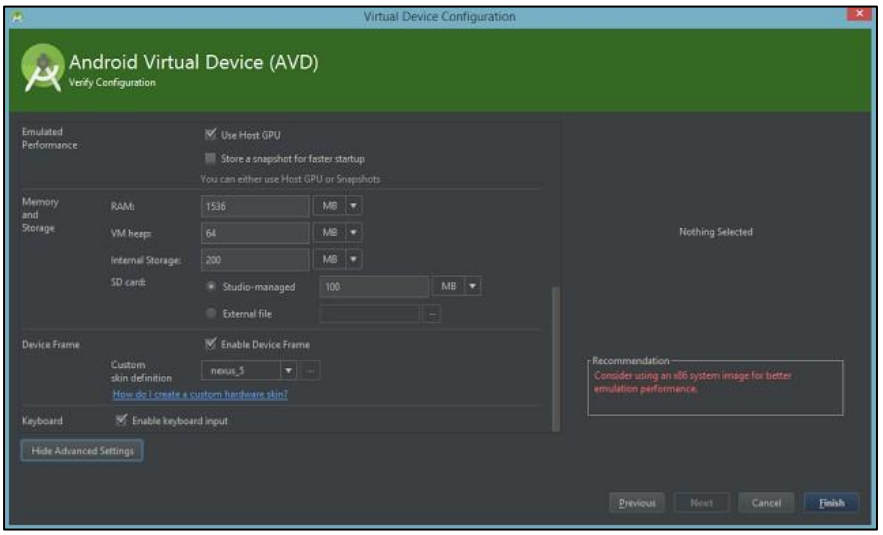
Hình 2.1.1.6. Giao diện chọn máy ảo

Ở đây các ta chọn phiên bản hệ điều hành muốn cài sau đó ấn Next để tiếp tục:



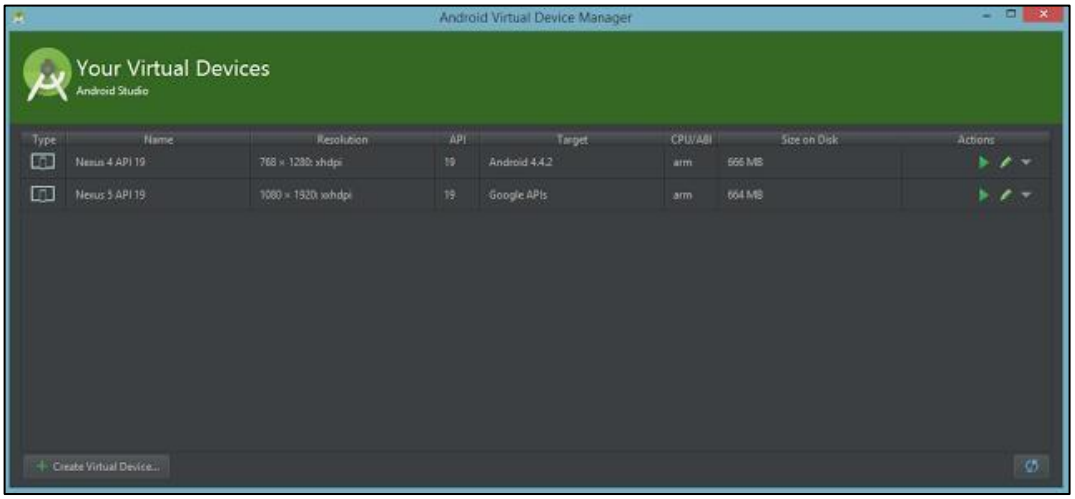
Hình 2.1.1.7. Giao diện cài đặt máy ảo

Ở cửa sổ tiếp theo này ta có thể tùy chỉnh cấu hình máy ảo, ví dụ như độ phân giải, CPU, màn hình nằm ngang hay dọc... Nếu muốn tùy chỉnh nhiều hơn nữa ta bấm vào nút **Show Advanced Setting** ở phía dưới cùng bên trái:



Hình 2.1.1.8. Giao diện cài đặt máy ảo

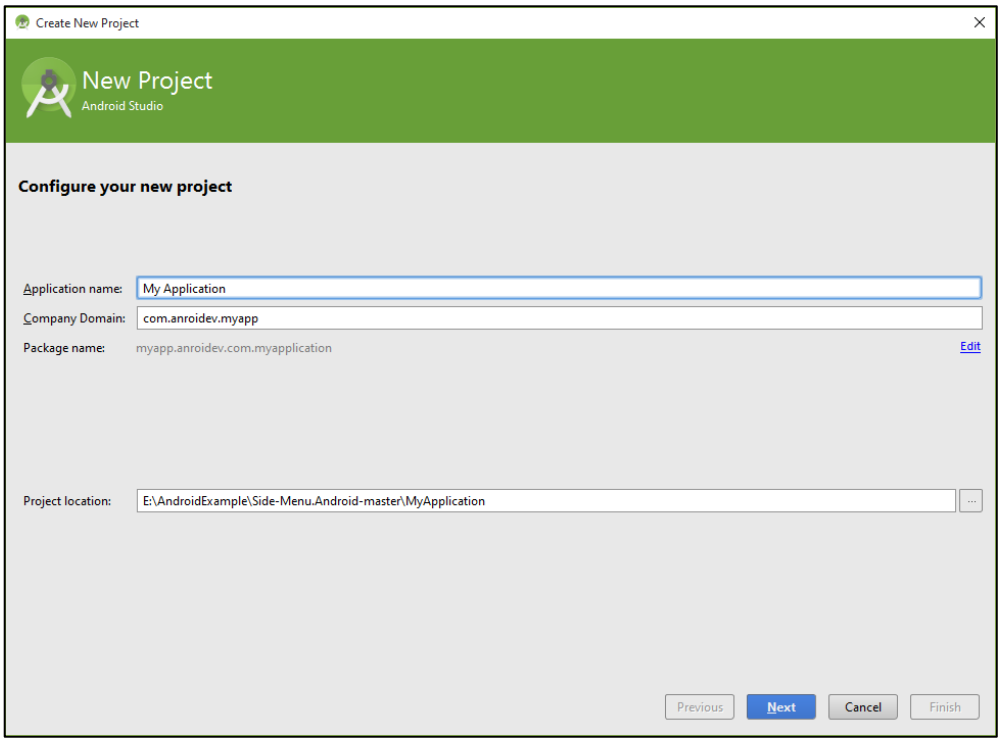
Ta kéo xuống phía dưới sẽ thấy thêm phần tùy chỉnh RAM, bộ nhớ trong, thẻ nhớ... Sau khi tùy chỉnh theo ý muốn ấn "Finish" để tiến hành tạo máy ảo và chờ một lúc để Android Studio lưu thông tin máy ảo. Sau khi lưu xong ta sẽ thấy trong cửa sổ **AVD Manager** có thêm một cái máy ảo nữa:



Hình 2.1.1.9. Giao diện cài đặt máy ảo

2.1.2. Cấu trúc dự án android trong ANDROID STDUIO.

a. Tạo mới một project.

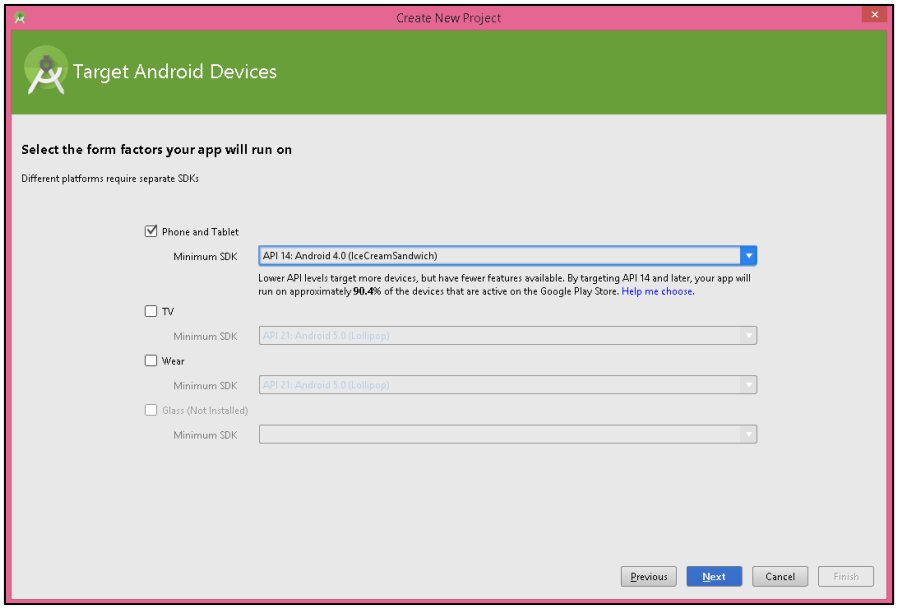


Hình 2.1.2.1. Tạo project

- **Application Name:** Tên ứng dụng muốn đặt
- **Company Domain:** Tên domain công ty, thường được dùng để kết hợp với tên Application để tạo thành Package (chú ý viết thường hết và có ít nhất 1 dấu chấm).
- **Package name:** Nó sẽ tự động nối ngược Company Domain với Application name.
- **Project location:** Là nơi lưu trữ ứng dụng.

Sau đó nhấp **Next** để tiếp tục.

b. Cài đặt một project.



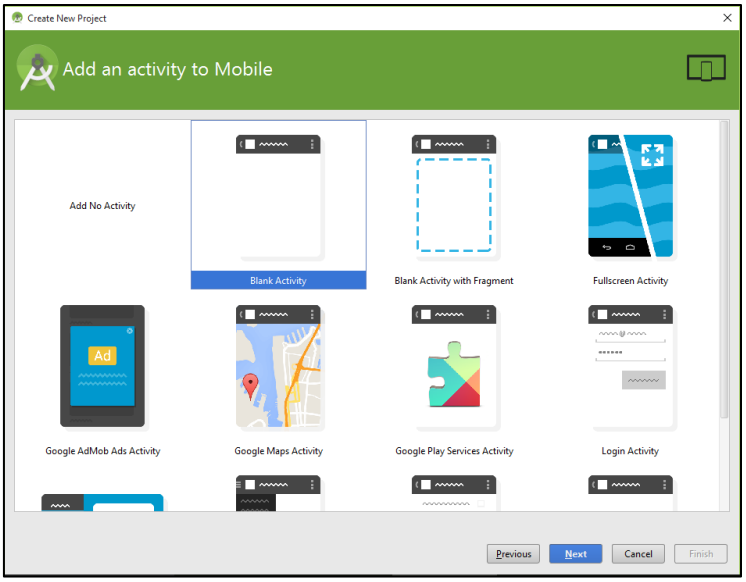
Hình 2.1.2.2. Giao diện lựa chọn thiết bị

Ở hộp thoại trên cho phép ta lựa chọn là ứng dụng sẽ được viết cho những thiết bị nào (Phone and Tablet, TV, Wear).

Ở mục Minium SDK, quy định phiên bản android tối thiểu để chạy ứng dụng.

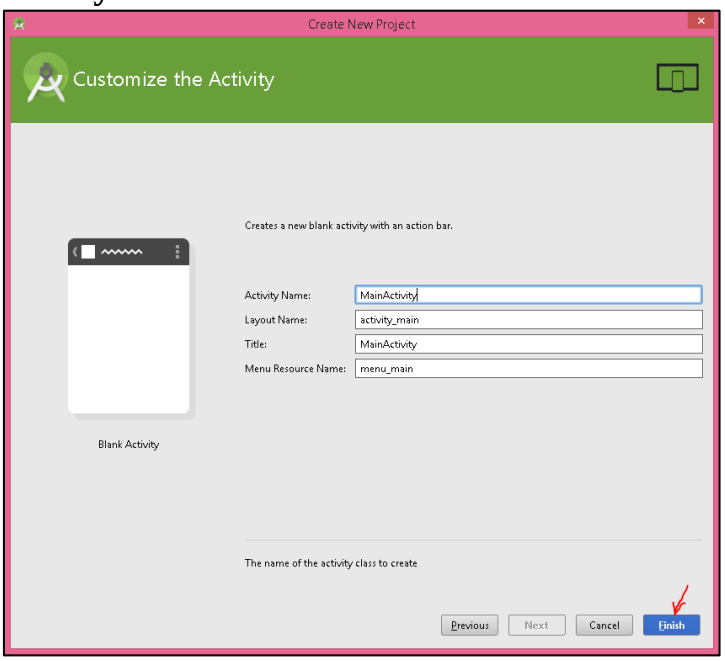
Hiện nay bản **API14 Android 4.0 (IceCreamSandwich)** vẫn đứng đầu về số lượng thiết bị sử dụng chiếm tới hơn 90%) nên thường lựa chọn.

Màn hình này hiển thị cho phép chọn loại Activity cần sử dụng.



Hình 2.1.2.3. Chọn kiểu giao diện cho ứng dụng

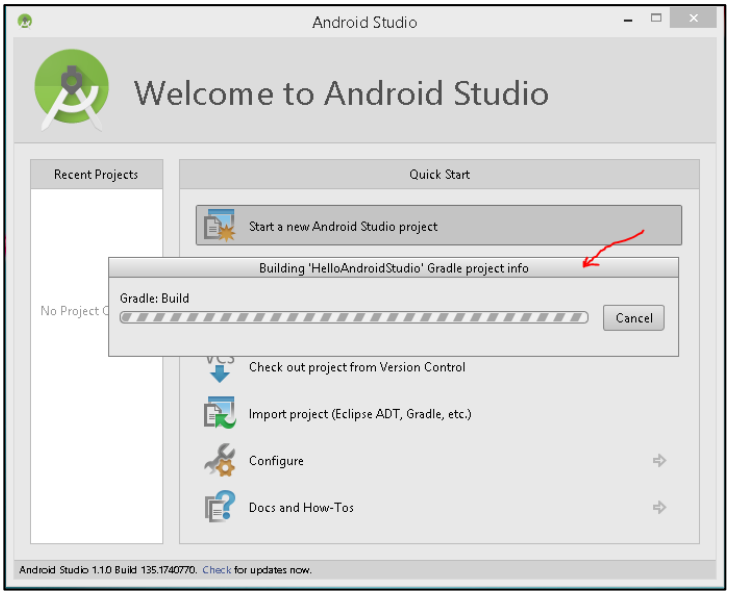
Chon **Blank Activity** rồi bấm Next:



Hình 2.1.2.4. Tuỳ chỉnh Activity

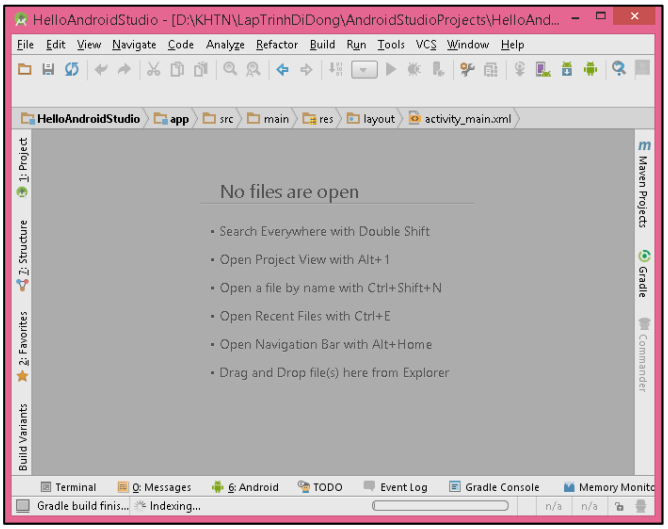
- **Activity Name:** Tên class Activity (java) để ta viết mã lệnh
- **Layout Name:** Tên file XML làm giao diện cho Activity Name.
- **Title:** Tiêu đề hiển thị khi kích hoạt Activity trên thiết bị.
- **Menu Resource Name:** Tên file xml để tạo menu cho phần mềm.

Sau khi cấu hình xong, bấm Finish, Màn hình Build Gradle project hiển thị:



Hình 2.1.2.5. Giao diện khởi tạo ứng dụng

Khi build xong mặc định có màn hình dưới đây:



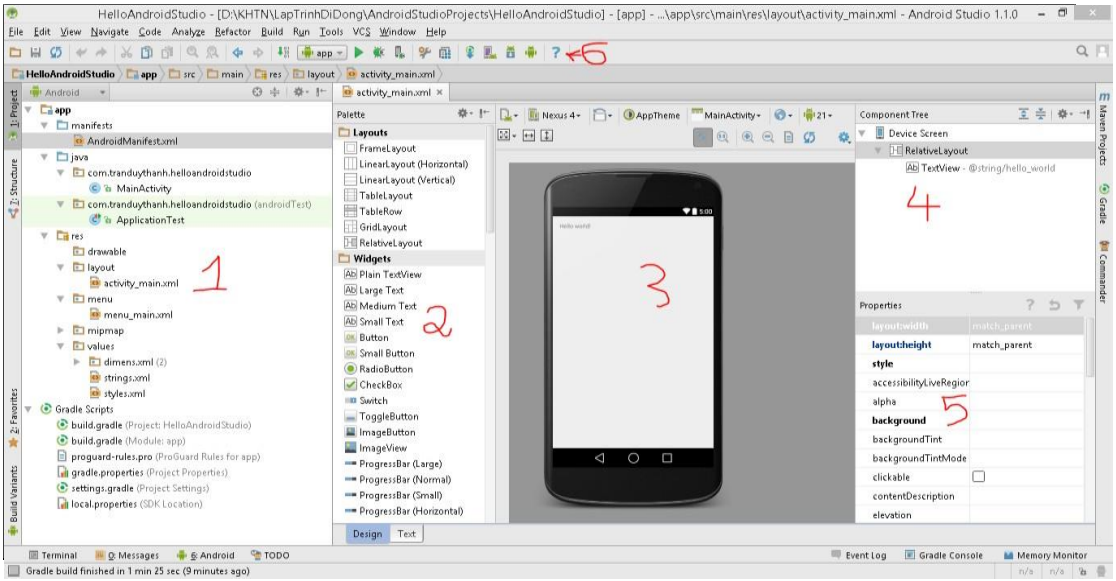
Hình 2.1.2.6. Giao diện khi khởi tạo project xong.

c. Màn hình làm việc của dự án android studio

Theo mặc định Android Studio hiển thị các files trong project theo góc nhìn Android. Góc nhìn này Android Studio sẽ tổ chức các files theo 3 module

- **manifests**: chứa file AndroidManifest.xml.
- **java**: chứa các file mã nguồn Java.
- **res**: chứa tất cả các file layout, xml, giao diện người dùng (UI), ảnh

Mở Project mặc định **activity_main.xml** sẽ được chọn ta có màn hình như sau:

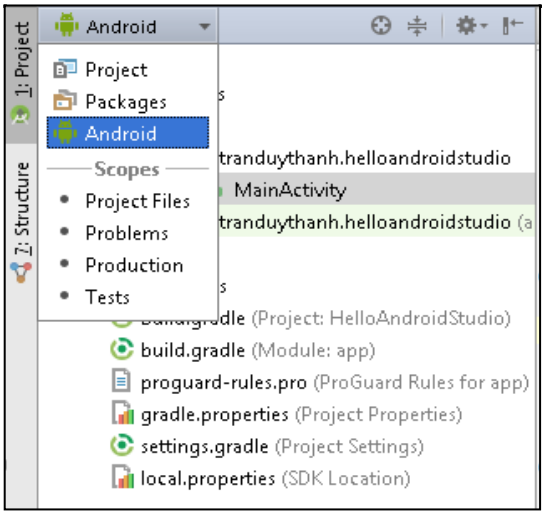


Hình 2.1.2.7. Giao diện Android Studio

Ở trên tạm thời chia làm 6 vùng làm việc mà lập trình viên thường tương tác.

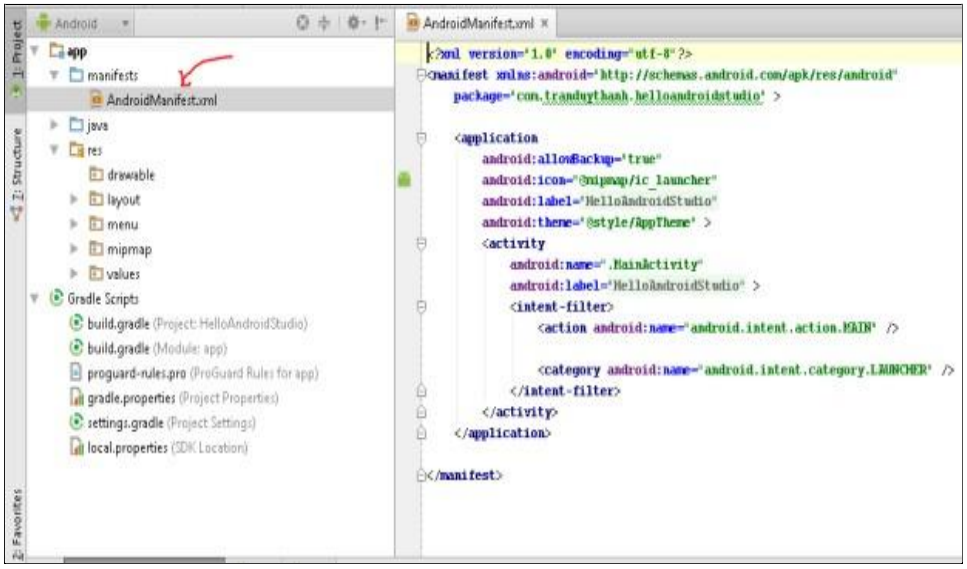
- Vùng 1.

Là nơi cấu trúc hệ thống thông tin của Ứng dụng, Ta có thể thay đổi cấu trúc hiển thị (thường để mặc định là **Android**).



Hình 2.1.2.8. Lựa chọn khung nhìn kiểu Android của ứng dụng

+Màn hình ở chế độ Android:



Hình 2.1.2.9. Màn hình ở ché độ Android

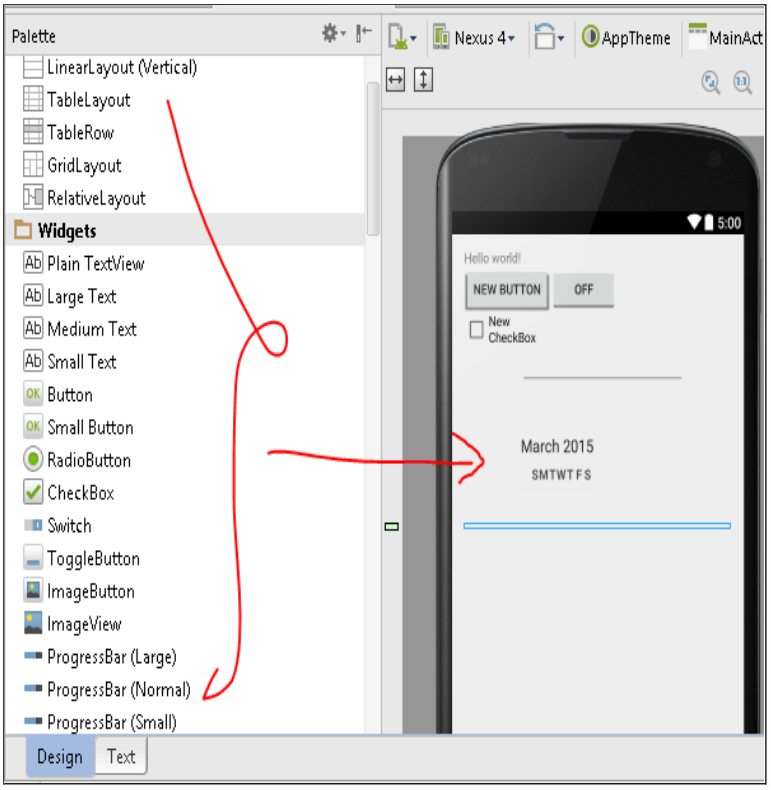
Ta có thể thấy **AndroidManifest.xml** nằm ở đây. File này vô cùng quan trọng trong việc cấu hình ứng dụng.

Các thư mục:

- **drawable**: chứa các file hình ảnh và xml trong ứng dụng.
- **layout**: chứa các giao diện màn hình đoợc thiết kế doới dạng xml.
- **values**: chứa các file lưu giá trị màu sắc, kích thước, chuỗi....

- Vùng 2.

Là vùng khá quan trọng cho những bạn mới bắt đầu lập trình, nó là nơi hiển thị các View mà Android hỗ trợ, cho phép bạn kéo thả trực tiếp vào vùng 3 (Giao Diện Thiết Bị) để thiết kế.



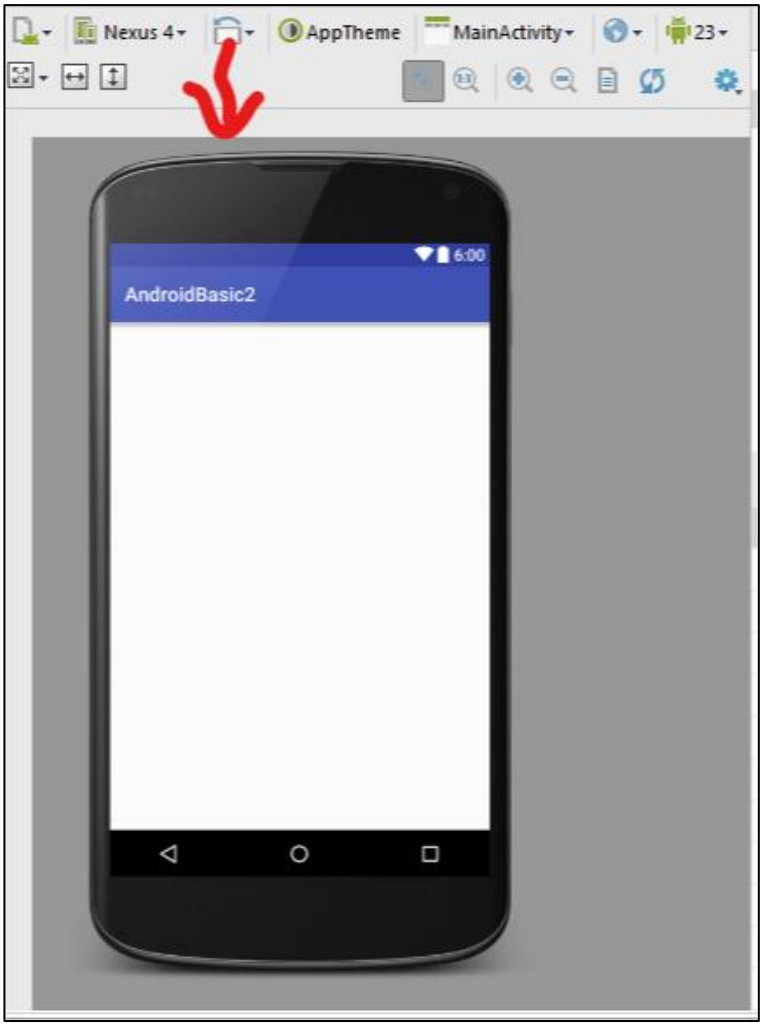
Hình 2.1.2.10. Vùng hiển thị các View mà Android hỗ trợ

Ở vùng số 2 này nó có 2 tab: **Design** và **Text** ở góc trái dưới cùng.

- Tab **Design** là tab mà ta đang nhìn và thao tác với nó (cho phép thiết kế giao diện bằng cách kéo thả).
- Tab **Text** là tab cho phép ta thiết kế giao diện bằng viết thẻ XML

- **Vùng 3.**

Là vùng giao diện thiết bị, cho phép các kéo thả View vào đây và đồng thời cho ta hiểu chính View.

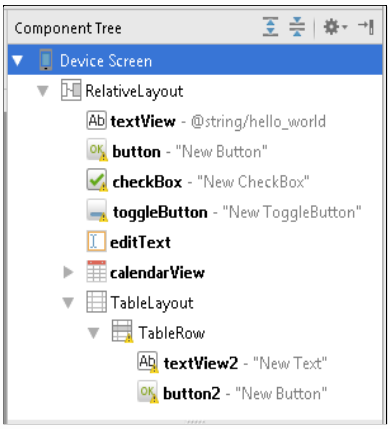


Hình 2.1.2.11 Vùng cho phép kéo thả View

Vùng 3 ta có thể chọn cách hiển thị theo nằm ngang nằm đứng, phóng to thu nhỏ, căn chỉ control, lựa chọn loại thiết bị hiển thị…

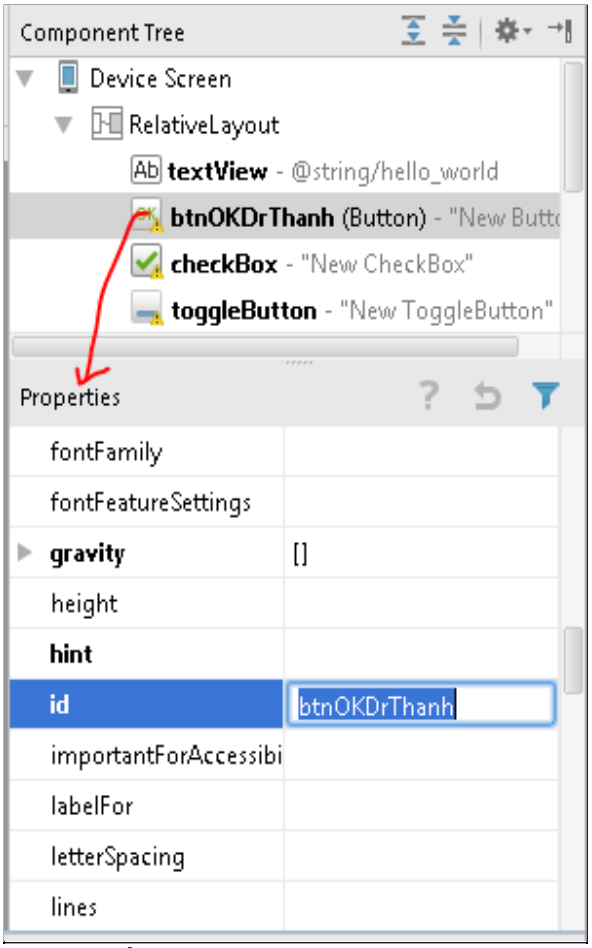
- Vùng 4.

Khi màn hình có nhiều View thì vùng 4 này trở lên hữu ích, nó cho phép hiển thị giao diện theo dạng cấu trúc cây, nên ta dễ dàng quan sát và lựa View bị chồng lập trên giao diện (vùng 3).



- Vùng 5.

Vùng này rất quan trọng, đây là vùng cho phép thiết lập trạng thái hay thuộc tính cho các View trên giao diện.

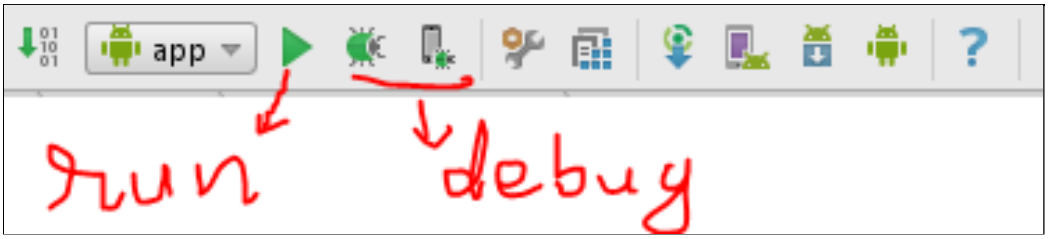


Hình 2.1.2.12 Thiết lập trạng thái hay thuộc tính giao diện

- Vùng 6

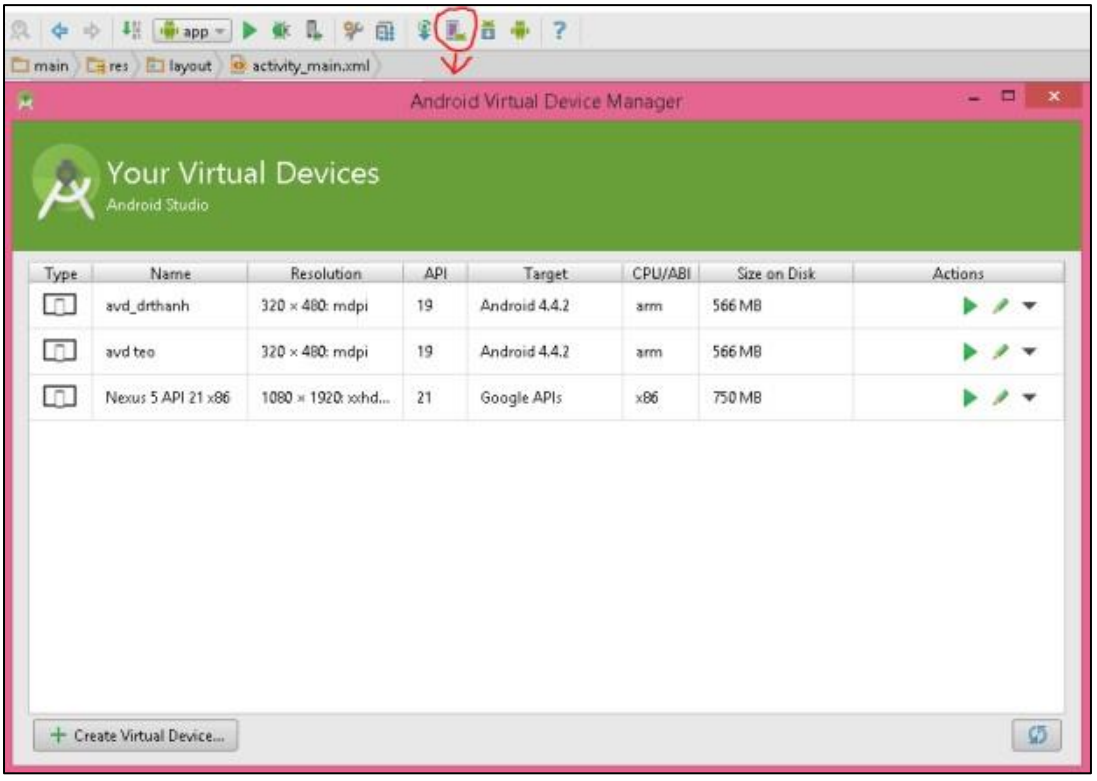
Là vùng các chức năng quan trọng thường dùng trong Android Studio.

+ Chạy ứng dụng và Debug ứng dụng:



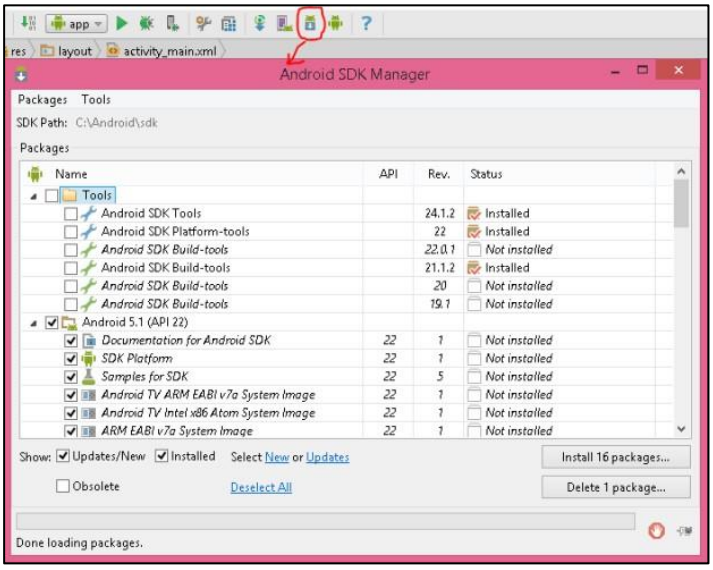
Hình 2.1.2.13. Chạy ứng dụng

+ Quản lý máy ảo (AVD Manager)



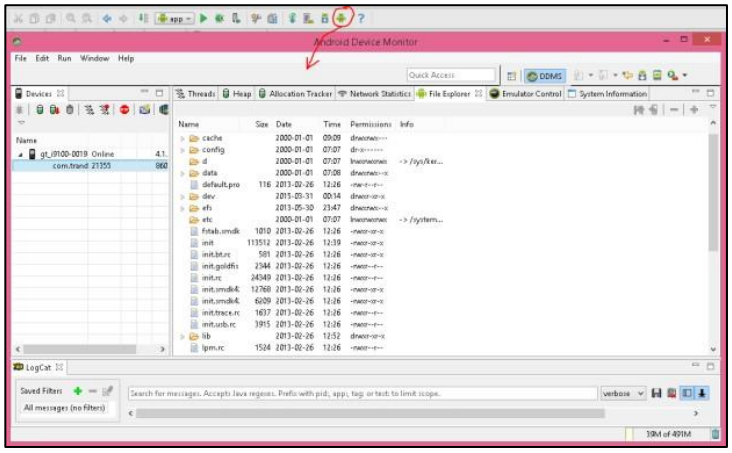
Hình 2.1.2.14 Quản lý máy ảo

+ Quản lý Android SDK Manager (thường dùng để cập nhật**).**



2.1.2.15 Quản lý máy ảo

+ Quản lý Android Device Manager



Hình 2.1.2.16. Quản lý máy ảo

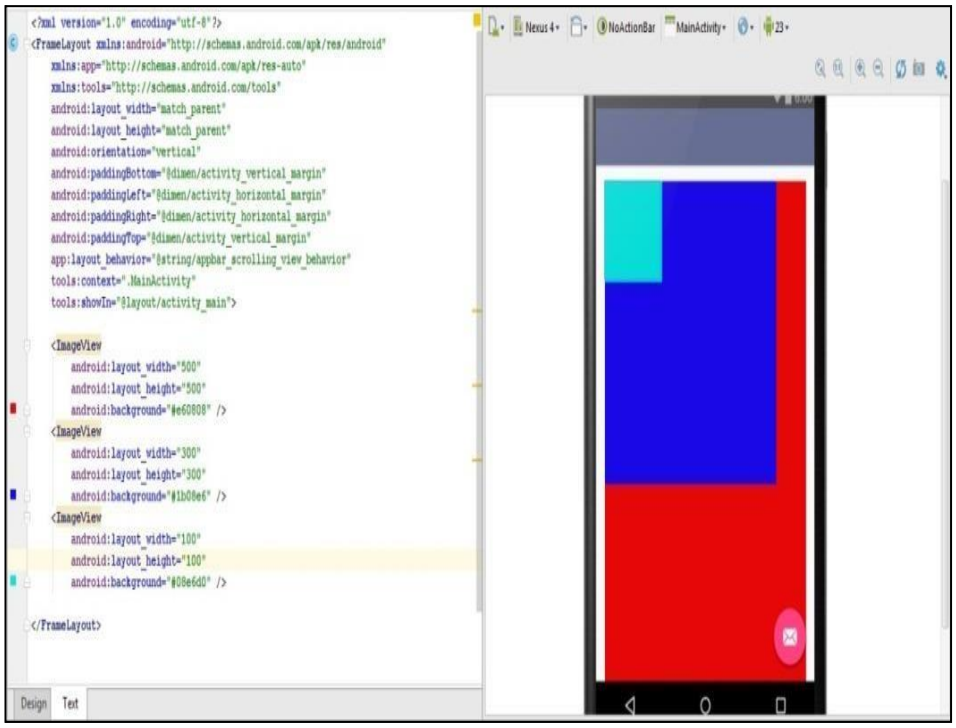
2.1.3. Xây dựng chương trình trong Android Studio

a. Giới thiệu android Layout

Layout là nơi chứa các View trên giao diện và mỗi layout có một cách sắp xếp các control khác nhau, vì vậy với mỗi cấu trúc giao diện khác nhau ta nên chọn layout cho phù hợp. Sau đây là một số layout cơ bản cho để ta thiết kế giao diện.

- FrameLayout.

Là loại layout cơ bản nhất, nó sẽ được dùng nhiều khi ta sử dụng vẽ giao diện nâng cao sau này. Khi ta kéo các control vào thì mặc định các control sẽ nằm ở vị trí trên cùng bên trái. Các control khi được kéo vào framelayout sẽ bị đè lên nhau, control sau sẽ đè lên control trước. Cách duy nhất để căn các control vào giữa là sử dụng thuộc tính android: **layout gravity**="center". Ta có thể tham khảo đoạn XML sau để hiểu thêm về framelayout.



Hình 2.1.3.1. Thiết kế Giao diện

- LinearLayout.

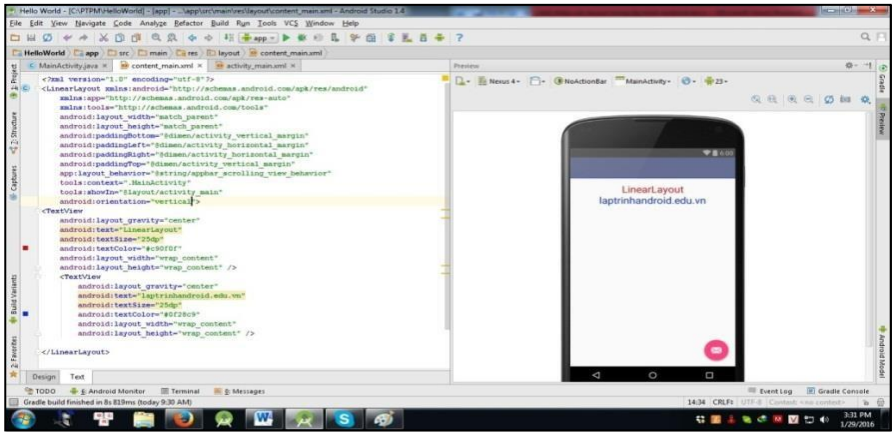
Layout này cho phép ta vẽ giao diện theo 2 hướng, từ trái qua phải hoặc từ trên xuống dưới. Để xét chiều cho các control trong layout ta sử dụng thuộc tính **orientation.**

- Android: orientation="horizontal": Xếp các control từ trái sang phải (theo cột).
- Android: orientation="vertical": Xếp các control từ trên xuống dưới (theo hàng).

Với những giao diện có độ phức tạp vừa phải thì dùng LinearLayout là rất hiệu quả, rất thuận tiện trong thiết kế và đi bảo trì ứng dụng sau này.

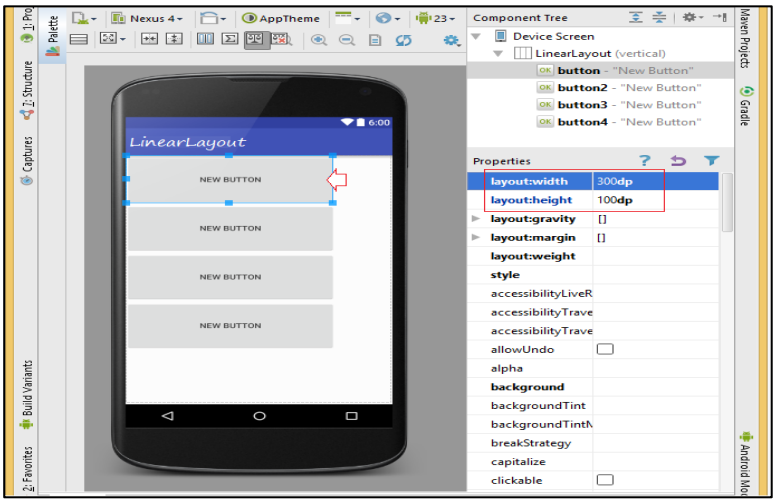
Sau đây là đoạn XML demo cách sử dụng layout này:

+ Theo chiều ngang



Hình 2.1.3.2. Giao diện theo chiều ngang

+ Theo chiều dọc



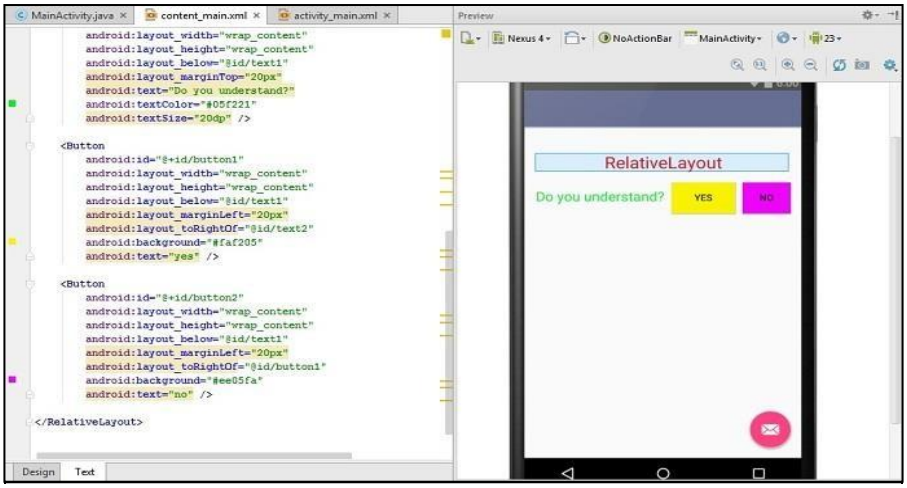
Hình 2.1.3.4. Giao diện theo chiều dọc

- RelativeLayout.

Layout này cho phép ta sắp xếp các control theo vị trí tương đối giữa các control khác kể cả control chứa nó. Khi gặp những layout có độ phức tạp cao, có nhiều giao diện nhỏ thì sử dụng RelativeLayout là lựa chọn tốt nhất. Một vài chú ý khi sử dụng layout này:

- Các control đều có id riêng, việc đặt tên id phải rõ rang dễ hiểu.
- Các control được sắp xếp dựa vào id của các control khác.
- Các control có sự ràng buộc và tương tác với nhau nên khi thay đổi một control sẽ làm thay đổi vị trí của mọi control khác. Vì vậy rất khó trong việc bảo trì nếu giao diện quá phức tạp.

Ta có thể tham khảo đoạn XML demo sau để hình dung dễ hơn:



Hình 2.1.3.5. Thiết kế giao diện

b. Giới thiệu một số android View cơ bản

- **TextView**: là view sử dụng để hiển thị text màn hình. TextView được định nghĩa bởi thẻ <TextView> trong xml.
- **EditText:** là view dùng để lấy giá trị từ người dùng nhập vào. EditText được định nghĩa bởi thẻ <EditText>trong xml.
- **ImageView:** là một view sử dụng rất nhiều trong ứng dụng android, ImageView sử dụng để hiển thị hình ảnh.
- **Button:** là view được sử dụng khá nhiều trong android, hầu như sử dụng ở mọi nơi cùng với EditText, TextView. Button có chức năng là làm nhiệm vụ nào đó khi mà người dùng click trong phương thức onClick.
- **ListView**: được tạo từ một danh sách các ListItem. ListItem là một dòng (row) riêng lẻ trong listview nơi mà dữ liệu sẽ được hiển thị. Bất kỳ dữ liệu nào trong listview chỉ được hiển thị thông qua listItem. Có thể coi listview như là một nhóm cuộn của các ListItem.

c. Bắt và xử lý sự kiện trên giao diện.

Sự kiện là một cách hữu ích để thu thập dữ liệu về sự tương tác của người dùng với các thành phần tương tác của ứng dụng. Giống như bấm vào một nút hoặc chạm vào màn hình cảm ứng, vv. Ta có thể nắm bắt những sự kiện trong chương trình và có những xử lý thích hợp theo yêu cầu. Có hai khái niệm liên quan đến quản lý sự kiện Android:

- **Event Listeners** là một interface. Event Listeners được sử dụng để đăng ký sự kiện cho các thành phần trong UI. (Đăng ký sự kiện). Trong các giao tiếp event listener có những phương thức sau đây:

- **onClick():** Thuộc View.OnClickListener. Nó được gọi khi người dùng hoặc chạm vào item (khi ở chế độ cảm ứng), hoặc lựa chọn vào item với các phím điều hướng và nhấn nút "enter" phù hợp.
- **onLongClick():** Thuộc View.OnLongClickListener. Nó được gọi khi người dùng chạm và giữ item (khi ở chế độ cảm ứng), hoặc lựa chọn vào item với các phím điều hướng sau đó nhấn và giữ phím "enter".
- **onFocusChange():** Thuộc View.OnFocusChangeListener. Nó được gọi khi người dùng điều hướng ra khỏi item, bằng cách sử dụng phím điều hướng.
- **onKey():** Thuộc View.OnKeyListener. Nó được gọi khi người dùng lựa chọn và nhấn lên item.
- **onTouch():** Thuộc View.OnTouchListener. Nó được gọi khi người dùng thực hiện một hành động xác định đủ điều kiện như là một sự kiện cảm ứng, bao gồm việc nhấn, thoát ra, hoặc bất kỳ cử chỉ chuyển động vẽ trên màn hình (bên trong phạm vi của item).
- **onCreateContextMenu():**
  Thuộc View.OnCreateContextMenuListener. Nó được gọi khi một menu ngữ cảnh (Context Menu) đang được xây dựng (là kết quả của một "long click"). Xem thêm thông tin về context menus trong hướng dẫn phát triển Menus.

Ví dụ dưới đây cho thấy làm thế nào để đăng ký một bộ bắt sự kiện khi nhấp chuột vào một Button.

```
// Create an anonymous implementation ò OnClickListener
Private OnclickListener mCorKyListener = new OnClickListener(){
Public void onCLick(View V/){
//do something when the button í cliicked
}
};

Protected void onCreate(Bundle savedValues){
…

// Caoture our button from layout
Button button = (Button) findViewById(R.id.corky);
// Register the onCLick listener with the implementation above
Buton.setOnClickListener(mCorkyListener);
…

}
```

Ta cũng có thể tìm thấy cách thuận tiện hơn để bổ sung OnClickListener như một phần Activity. Ví dụ:

```
Public class ExmapleActivity extends Activity implements OnClickLiistener
{
…
Button button = (Button)findViewById(R.id.corky;
Button.setOnClickListener(this);
}
//Implement the OnClickListener callback
Public void onClick(View v){
//do something when the bitton is clicked
}
…
}
```

Chú ý rằng lời gọi **onClick()** trong ví dụ trên không trả về giá trị, nhưng các phương thức của bộ nghe sự kiện khác phải trả lại một biến kiểu boolean. Lý do phụ thuộc vào sự kiện này. Đây là một vài lý do:

- **onLongClick()** - Trả về một giá trị kiểu boolean để cho biết ta đã dùng sự kiện này và nó không cần thực hiện "long click") thêm nữa. Trả về giá trị *TRUE* để chỉ ra rằng ta đã xử lý sự kiện này và nó nên dừng lại ở đây; trả về *FALSE* nếu ta không xử lý nó và / hoặc sự kiện nên chuyển tới bất kỳ bộ nghe sự kiện **on-click** nào khác.
- **onKey()** - Trả về một giá trị kiểu boolean để cho biết ta đã dùng sự kiện này và nó không cần được thực hiện thêm. Trả về giá trị *TRUE* để chỉ ra rằng ta đã xử lý sự kiện này và nó nên dừng lại ở đây; trả về *FALSE* nếu ta không xử lý nó và / hoặc sự kiện nên chuyển tới bất kỳ bộ nghe sự kiện **on-key** nào khác.
- **onTouch()** - Trả về một giá trị kiểu boolean để cho biết: liệu bộ nghe của ta đã dùng sự kiện này hay chưa. Điều quan trọng là sự kiện này có thể có nhiều hành động nối tiếp nhau. Vì vậy, nếu trả về *FALSE*, ta biết rằng ta đã không sử dụng và cũng không quan tâm đến hành động tiếp theo từ sự kiện này. Như vậy, ta không được gọi tới bất kỳ thao tác nào khác bên trong sự kiện này.

- **Event Handlers** – Là phương thức xử lý khi phát sinh sự kiện. (Xử lý sự kiện)

Nếu ta đang xây dựng một thành phần tùy chỉnh từ View, ta sẽ phải định nghĩa một số phương thức sử dụng như của xử lý sự kiện mặc định. Trong tài

liệu về Custom Components, ta sẽ tìm hiểu một số callbacks thường được sử dụng để xử lý sự kiện, bao gồm:

- onKeyDown(int, KeyEvent) - Được gọi khi một sự kiện nhấn phìm mới xảy ra.
- onKeyUp(int, KeyEvent) - Được gọi khi mọt sự kiện thả phím xảy ra.
- onTrackballEvent(MotionEvent) - Được gọi khi một sự kiện chuyển động trackball xảy ra.
- onTouchEvent(MotionEvent) - Được gọi khi một sự kiện chuyển động màn hình cảm ứng xảy ra.
- onFocusChanged(boolean, int, Rect) - Được gọi khi view được chọn (focus) hoặc bỏ chọn.

Có một số phương thức khác mà ta nên biết, chúng không phải là một phần của lớp View, nhưng có thể trực tiếp tác động đến cách bạn có thể xử lý các sự kiện. Vì vậy, khi quản lý sự kiện phức tạp hơn bên trong một layout, ta nên xem xét các phương pháp sau:

- Activity.dispatchTouchEvent(MotionEvent) - Điều này cho phép Activity bắt tất cả các sự kiện chạm màn hình trước khi chúng được gửi đến cửa sổ.
- ViewGroup.onInterceptTouchEvent(MotionEvent) - Điều này cho phép ViewGroup xem các sự kiện như chúng được gửi đến các View con.
- ViewParent.requestDisallowInterceptTouchEvent(boolean) - Gọi điều này trên View cha để xác định rằng nó không nên bắt các sự kiện chạm màn hình với onInterceptTouchEvent(MotionEvent)

Chương 3: KỸ THUẬT XÂY DỰNG BỘ ĐỀ TRẮC NGHIỆM TIẾNG ANH

3.1. Xây dựng bộ đề trắc nghiệm tiếng Anh

a. SQLite

SQLite là phần mềm quản lý cơ sở dữ liệu tương tự Mysql, PostgreSQL…Đặc điểm của SQLite là gọn, nhẹ, đơn giản. Chương trình gồm 1 file duy nhất vỏn vẹn chưa đến 400kB, không cần cài đặt, không cần[3]cấu hình hay khởi động mà có thể sử dụng ngay. Dữ liệu Database cũng được lưu ở một file duy nhất. Không có khái niệm user, password hay quyền hạn trong SQLite Database.

SQLite không thích hợp với hệ thống lớn nhưng ở quy mô vừa tầm thì SQLite phát huy uy lực và không hề yếu kém về mặt chức năng hay tốc độ. Với các đặt điểm trên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm … và là sự lựa chọn phù hợp cho những người bắt đầu học Database.

SQLite Engine không là một Standalone Process giống như các cơ sở dữ liệu khác, bạn có thể liên kết nó cách tĩnh hoặc một cách động tuỳ theo yêu cầu ứng dụng của bạn. SQLite truy cập các file lưu giữ của nó một cách trực tiếp.

b. Xây dựng bộ đề trắc nghiệm tiếng Anh

- Bước 1:

Ta sử dụng Excel để xây dựng bộ đề trắc nghiệm tiếng Anh và lưu file dưới đuôi *csv

Trong cấu trúc bộ đề gồm có 10 cột, mỗi cột chứa một thông tin khác nhau:

+ id: Định danh cho câu hỏi

+question: Nội dung câu hỏi

+ans_a; ans_b; ans_c; ans_d: Thứ tự các đáp án A,B,C,D

+result: Đáp án đúng

+num_exam: Mã đề

+image: Ảnh

+subject: Môn học

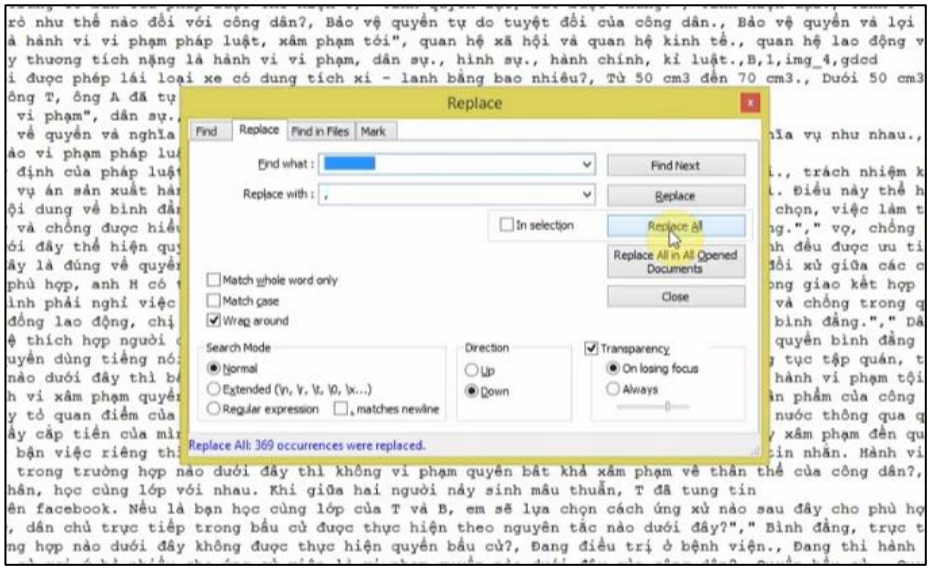
_id	question	ans_a	ans_b	ans_c	ans_d	result	num_exar	image	subject
1	I'm ..... An	A. in	B. a	C. from	D. too	C	1		TA
2	There .... t	A. is	B. many	C. are	D. isn't	C	1		TA
3	Hi, my	A. name is	B. names'	C. names	D. is name	A	1		TA
4	Jimmy is h	A. her	B. she	C. his	D. he	C	1		TA
5	I .... two c	A.has	B. haven't	C. hasn't	D. have	D	1		TA
6	Your cat ..	A. is	B. are	C. an	D. a	A	1		TA
7	Mary stud	A. to/at	B. from/in	C. from/to	D. on/to	C	1		TA
8	Are they s	A. they	B. they do	C. are the	D. are the	A	1		TA
9	Choose th	A. May	B. April	C. June	D. Peter	D	1		TA
10	Can you sv	A. No, I ca	B. No, tha	C. Yes, I ca	D. Yes, I ca	D	1		TA
11	Would yo	A. a	B. an	C. some	D. the	B	1		TA
12	Goodbye,	A. Later se	B. See you	C. See late	D.You see	B	1		TA
13	Can you sp	A. Yes, I ca	B. No, I ca	C.No, I car	D.Yes,I do	C	1		TA
14	I can spea	A. VietNai	B. EngLanc	C. Americ	D. English	D	1		TA
15	..... you lik	A. Can	B. Are	C. Is	D. Would	D	1		TA
16	..... are yo	A. What	B. How	C. Where	D.When	C	1		TA
17	What doe	A. from	B. and	C. during	D. end	C	1		TA
18	His book a	A. at	B.in	C. down	D. on	D	1		TA
19	How	A. much	B. very	C. many	D. on	C	1		TA
20	Where is s	A. She's at	B. She's fr	C. She's ni	D. She's M	A	1		TA
21	..... per	A. How	B. When	C. How	D. How m	D	1		TA
22	They live .	A. on	B. in	C. under	D. for	B	1		TA

Tracnghiem

Hình 3.1.1. Cấu trúc bộ đề tiếng anh file Excel

- Bước 2:

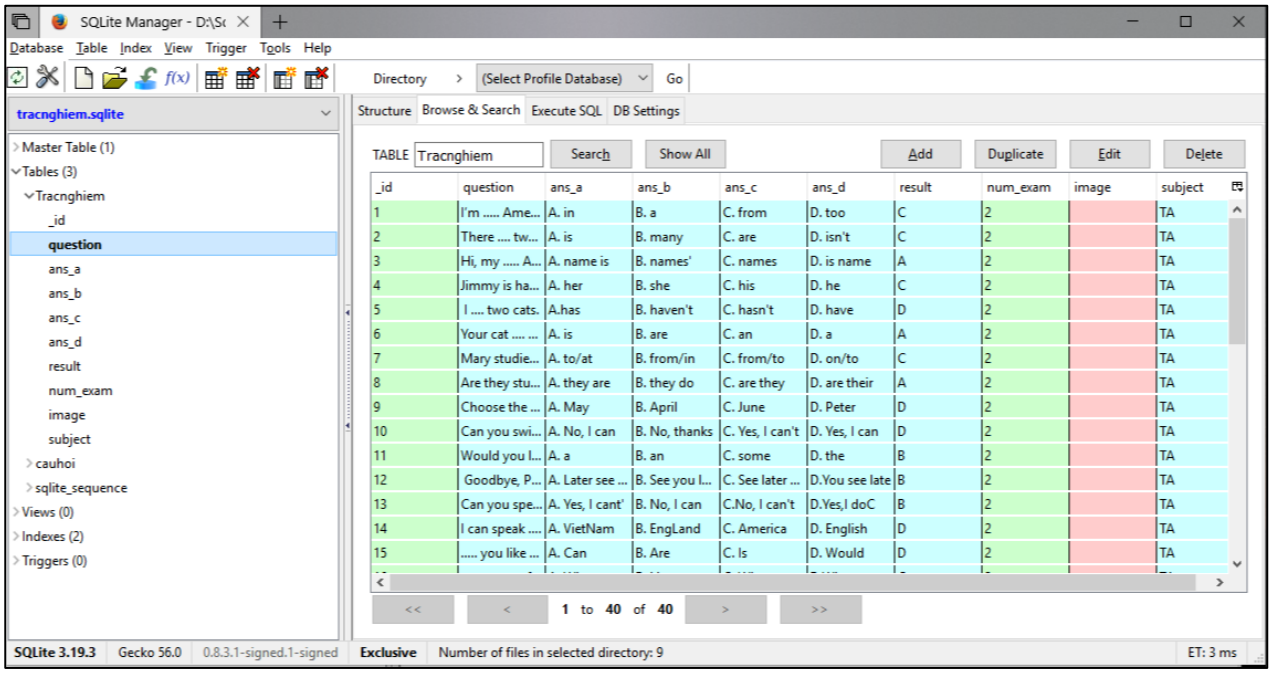
Sau đó ta mở file Excel bằng notepad và dùng Replace sửa hết dấu cách thành dấu phẩy và lưu encoding thành UTF-8



Hình 3.1.2. Tìm và thay thế

- Bước 3:

Ta thêm bộ đề vào SQLite



Hình 3.1.2 Dữ liệu sau khi thêm vào SQLite

3.2. Kỹ thuật lật trang câu hỏi

ViewPage trong Android

-ViewPage là một đối tượng khá giống như Slide trình diễn của PowerPoint.

-ViewPage có thể trượt chuyển đổi giữa[1]các giao diện một cách nhẹ nhàng và khá mượt, thay vì chuyển đổi màn hình qua một sự kiện chớp đen như trên ti-vi. Màn hình hiển thị trước nó hoặc sau nó sẽ được hiển thị ra ngay tức thì liền với nó.

- Bước 1:

Ta code giao diện cho MainActivity và một ViewPage để chứ nội dung

- Bước 2:

Ta tạo giao diện fragment chứa nội dung bên trong

```
        <!-- fragment_screen_slide_page.xml -->
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/content"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent" >

    <TextView style="?android:textAppearanceMedium"
        android:padding="16dp"
        android:lineSpacingMultiplier="1.2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/lorem_ipsum" />
</ScrollView>
```

- **Bước 3:**

Chúng ta tạo các file java cho các fragment tương ứng

```
import android.support.v4.app.Fragment;
...
public class ScreenSlidePageFragment extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {
        ViewGroup rootView = (ViewGroup) inflater.inflate(
                R.layout.fragment_screen_slide_page, container, false);

        return rootView;
    }
}
```

- **Bước 4:**

Tạo file giao diện cho ViewPager

```
<!-- activity_screen_slide.xml -->
<android.support.v4.view.ViewPager
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

- **Bước 5:**

Tạo một file PagerAdapter.java để cấu hình cho ViewPage hiển thị các fragment được tạo ở trên màn hình và hiển thị

```
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
...
public class ScreenSlidePagerActivity extends FragmentActivity {
    /**
     * The number of pages (wizard steps) to show in this demo.
```

```
     */
    private static final int NUM_PAGES = 5;
    /**    private ViewPager mPager;
    /**
     * The pager adapter, which provides the pages to the view pager widget.
     */
    private PagerAdapter mPagerAdapter;

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

        // Instantiate a ViewPager and a PagerAdapter.
        mPager = (ViewPager) findViewById(R.id.pager);
        mPagerAdapter = new
ScreenSlidePagerAdapter(getSupportFragmentManager());
        mPager.setAdapter(mPagerAdapter);
    }
    @Override
    public void onBackPressed() {
        if (mPager.getCurrentItem() == 0) {
            // If the user is currently looking at the first step, allow the
system to handle the
            // Back button. This calls finish() on this activity and pops the
back stack.
            super.onBackPressed();
        } else {
            // Otherwise, select the previous step.
            mPager.setCurrentItem(mPager.getCurrentItem() - 1);
        }
    }

       private class ScreenSlidePagerAdapter extends
FragmentStatePagerAdapter {
        public ScreenSlidePagerAdapter(FragmentManager fm) {
            super(fm);
        }
        @Override
        public Fragment getItem(int position) {
            return new ScreenSlidePageFragment();
        }
        @Override
        public int getCount() {
            return NUM_PAGES;
        }
```

```
    }
}
```

3.3 Kỹ thuật tính thời gian trắc nghiệm

Để sát hạch trình độ học viên một cách khách quan và chính xac ta thực hiện có tính toán về mặt thời gian. Trong mỗi phần thì hệ thống[2]ấn định cho mỗi phần thi là 15 phút cho 20 câu hỏi trắc nghiệm. Ta sử dụng CountDownTimer

CountDownTimer là class cung cấp chức năng giúp người lập trình tạo bộ đếm ngược trong Android.

- Ví dụ:

```
totalTimer=15;
timer=new CounterClass(totalTimer*60*1000,1000);
questionController= new QuestionController(this);
arr_Ques= new ArrayList<Question>();
arr_Ques=questionController.getQuestion(1,"TA");

tvKiemtra=(TextView)findViewById(R.id.tvKiemTra);
tvTimer=(TextView)findViewById(R.id.tvTimer);
tvXemDiem=(TextView)findViewById(R.id.tvScore);

tvKiemtra.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        checkAnawer();

    }
});

tvTimer.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

    }
});
tvXemDiem.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish();
        Intent intent1= new
Intent(ScreenSlideActivity.this,TestDoneActivity.class);
        intent1.putExtra("arr_Ques",arr_Ques);
        startActivity(intent1);
    }
});
  timer.star();
```

Đoạn code trên ta hiểu rằng, một đối tượng có tên totalTimer được tạo ra ấn định là 15 phút kèm theo hai đối tượng số 60*1000 và 1000 có nghĩa là cứ 1000 mi-li giây thì phương thức onTick sẽ được chạy một lần, cứ như thế

trong suốt 60000 mi-li giây. 60000 mi-li giây trôi qua, thì phương thức onFinish() sẽ chạy.

Phướng thức onTick, kèm theo một đối số millisUntilFinished, đây là đối số cho ta biết sô mi-li còn lại của bộ đếm

Phương thức onFinish, phương thức này được tự động gọi khi 60000 mi-li giây của bộ đếm trôi qua, đồng nghĩa hết "Hết Giờ".

3.4 Kỹ thuật tính điểm trắc nghiệm

Khi hoàn xong đề trắc nghiệm tiếng Anh học viên muốn biết được tổng bao nhiêu điểm.

- Bước 1:

Ta sử dụng đoạn code bắt sự kiện đáp án đúng dựa vào cơ sở dữ liệu trong SQLite

```
radioGroup.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup radioGroup, int
checkId) {
                getItem(mPageNumber).choiceID= checkId;

getItem(mPageNumber).setTraloi(geChoiceFromID(checkId));
            //    Toast.makeText(getActivity(),"Đây là đáp
án"+checkId, Toast.LENGTH_SHORT).show();
            }
        });
```

-Bước 2:

Tổng điểm là 100 và có 20 câu hỏi ta lấy câu trả lời đúng nhân với 5

```
public class TestDoneActivity extends AppCompatActivity {

    ArrayList<Question> arr_QuesBegin= new ArrayList<Question>();
    int numNoAns=0;
    int numTrue=0;
    int numFalse=0;
    int totalScore=0;

    TextView tvTrue, tvFalse, tvNotAns,tvTotalScore;
    Button btnExit;

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

        Intent intent= getIntent();
        arr_QuesBegin= (ArrayList<Question>)
intent.getExtras().getSerializable("arr_Ques");
        begin();
        checkResult();
        totalScore =numTrue*5;
        tvNotAns.setText(""+numNoAns);
```

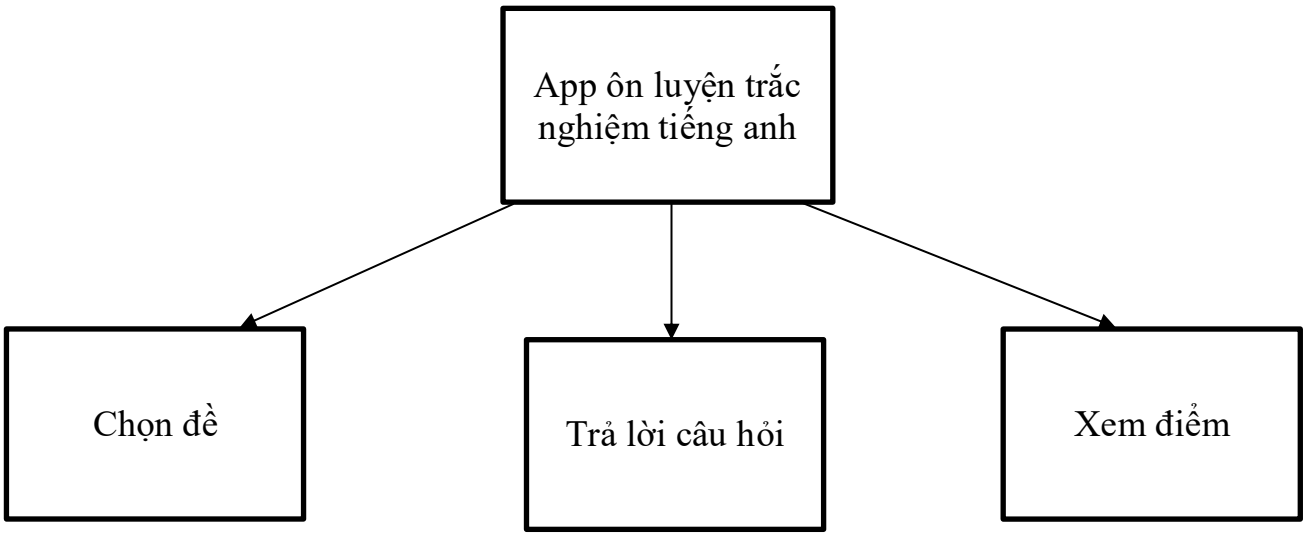
```
            tvFalse.setText(""+numFalse);
            tvTrue.setText(""+numTrue);
            tvTotalScore.setText(""+totalScore);
```

Chương 4: CHƯƠNG TRÌNH THỰC NGHIỆM

4.1. Phát biểu bài toán

Hiện nay tiếng là một ngôn ngữ rất cần thiết cho nhiều đối tượng và các lĩnh vực khác nhau trong xã hội. Do chúng ta đang sống trong môi trường phi bản ngữ nên việc thực hành và ôn luyện tiếng Anh còn gặp nhiều khó khăn. Xuất phát từ nhu cầu thực tế, học sinh sinh viên và những người muốn học tiếng Anh nhưng không có thời gian nhiều để đi đến các trường học hay trung tâm hoặc muốn ôn muốn ôn luyện lại những kiến thức đã học ở mọi nơi, tận dụng thời gian rảnh để ôn luyện như ở bến đợi xe hay lúc nghỉ trưa … mà không cần mang nhiều sách vở.Xuất phát từ nhu cầu thực tế việc xây dựng ứng dụng ôn luyện trắc nghiệm tiếng Anh là cần thiết, chương trình này gồm các đề tiếng Anh có tính thời gian và tính điểm sẽ giúp học từ mới cách sử dụng và ôn luyện ngữ pháp.

4.2. Mô hình chức năng



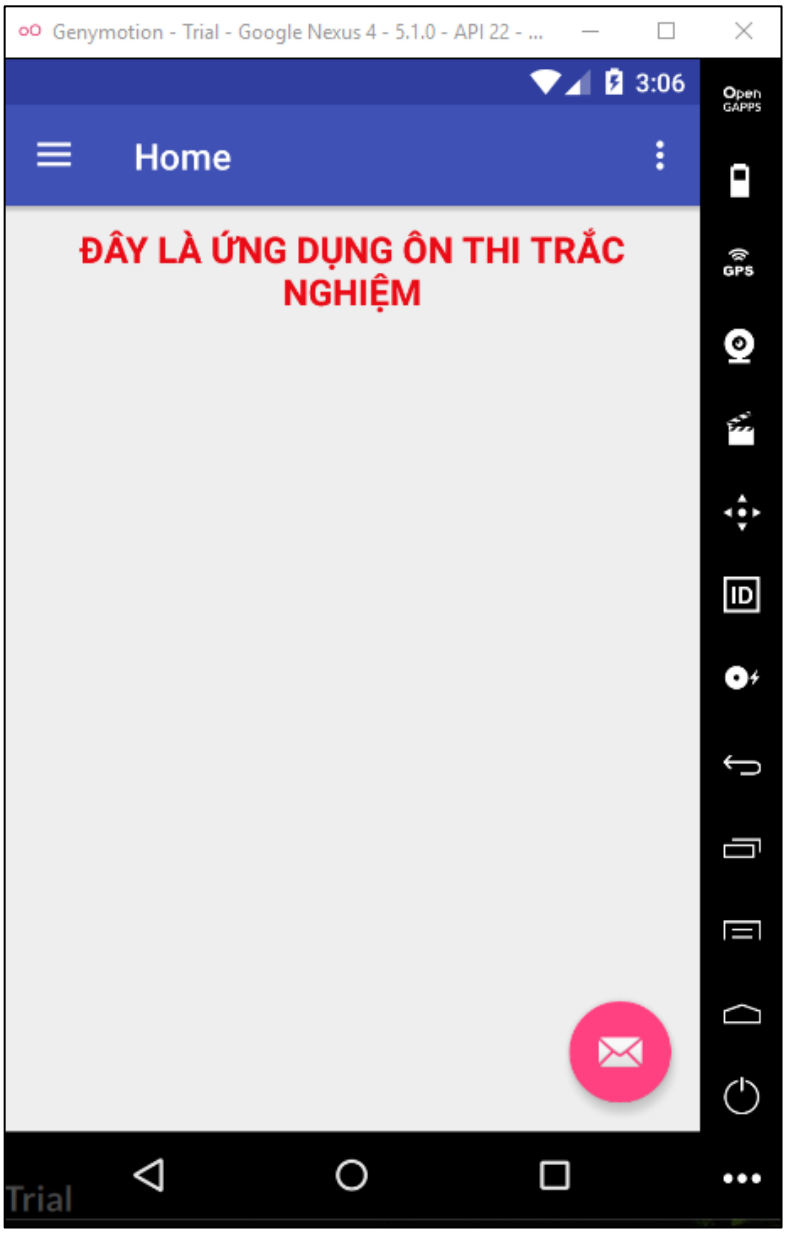
- Chọn đề: Chọn các mã đề thi trắc nghiệm tiếng Anh.

- Trả lời câu hỏi: Chọn đáp án đúng có thời gian đếm ngược và xem danh sách các câu đã trả lời.

- Xem điểm: Hiển thị các đáp án câu đúng, câu sai, câu đã trả lời, tổng điểm.

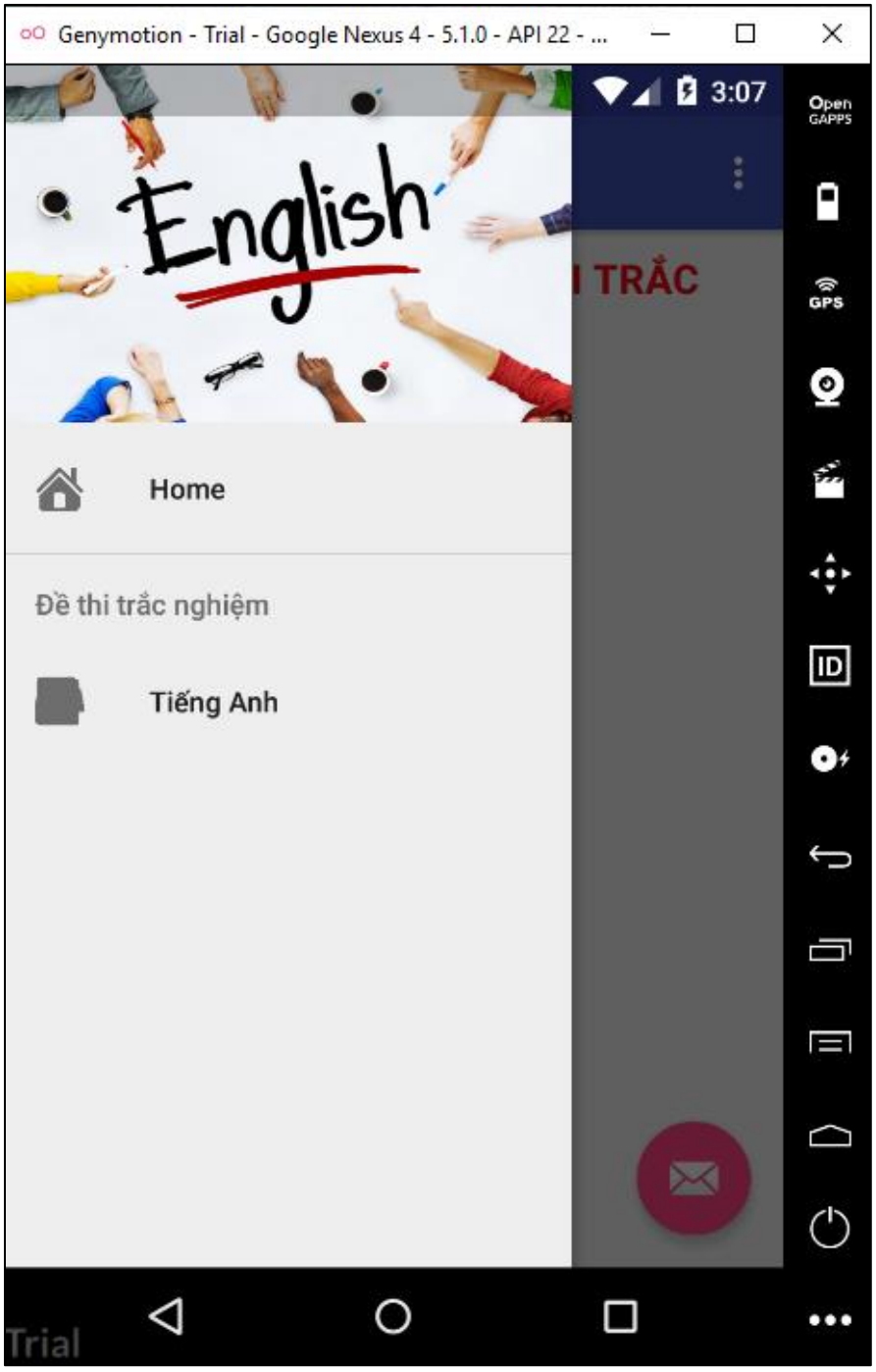
4.3. Giao diện chương trình

- Giao diện màn hình chính



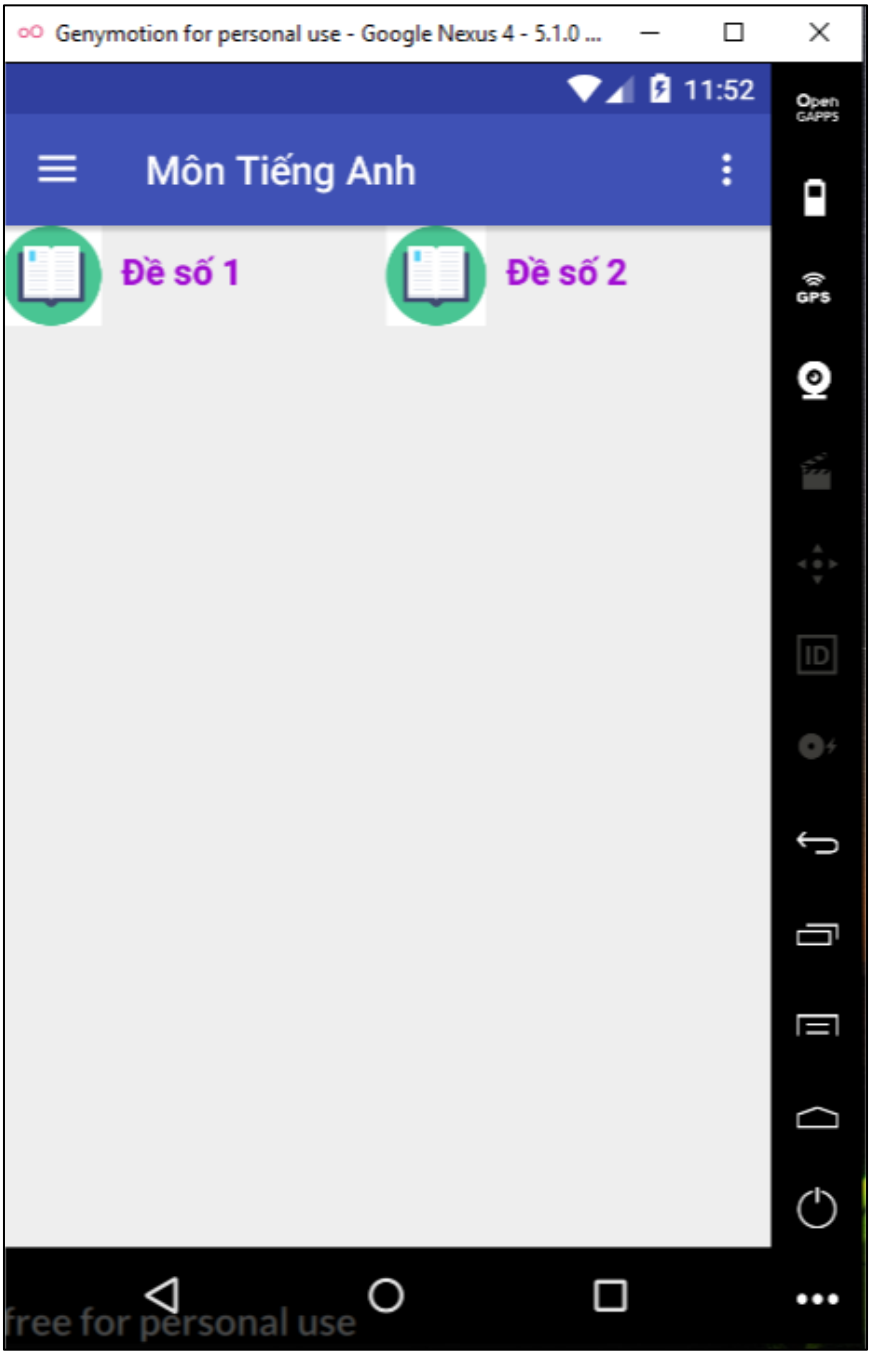
Hình 4.2.1. Giao diện màn hình chính

- Giao diện menu:



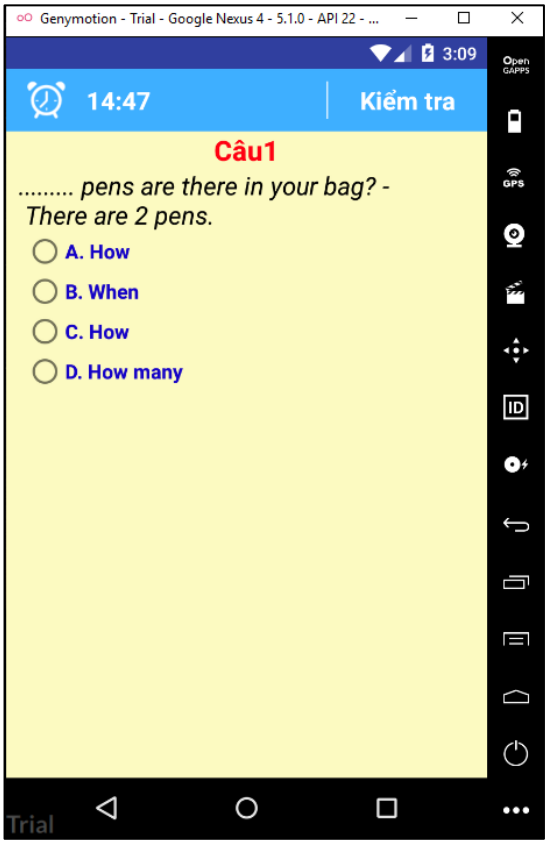
Hình 4.2.2. Giao diện Menu

- Giao diện đề:

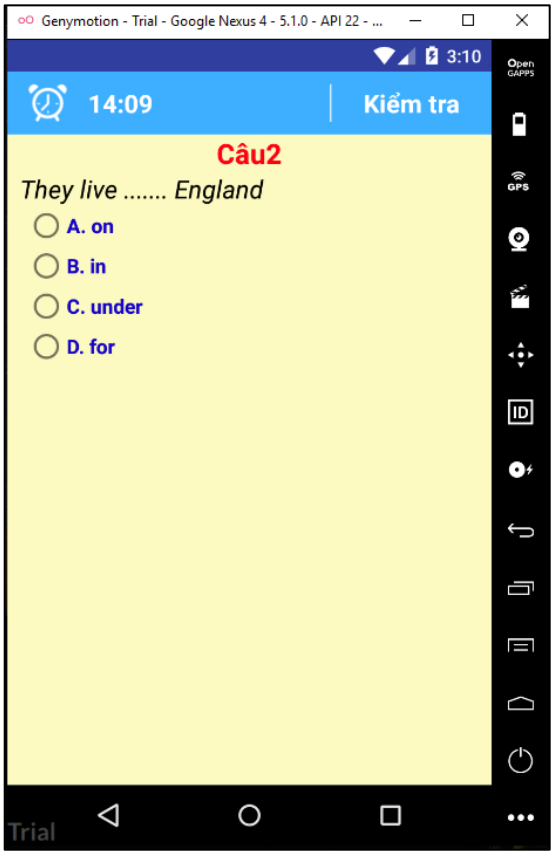


Hình 4.2.3. Giao diện chọn đề

- Giao diện trả lời:

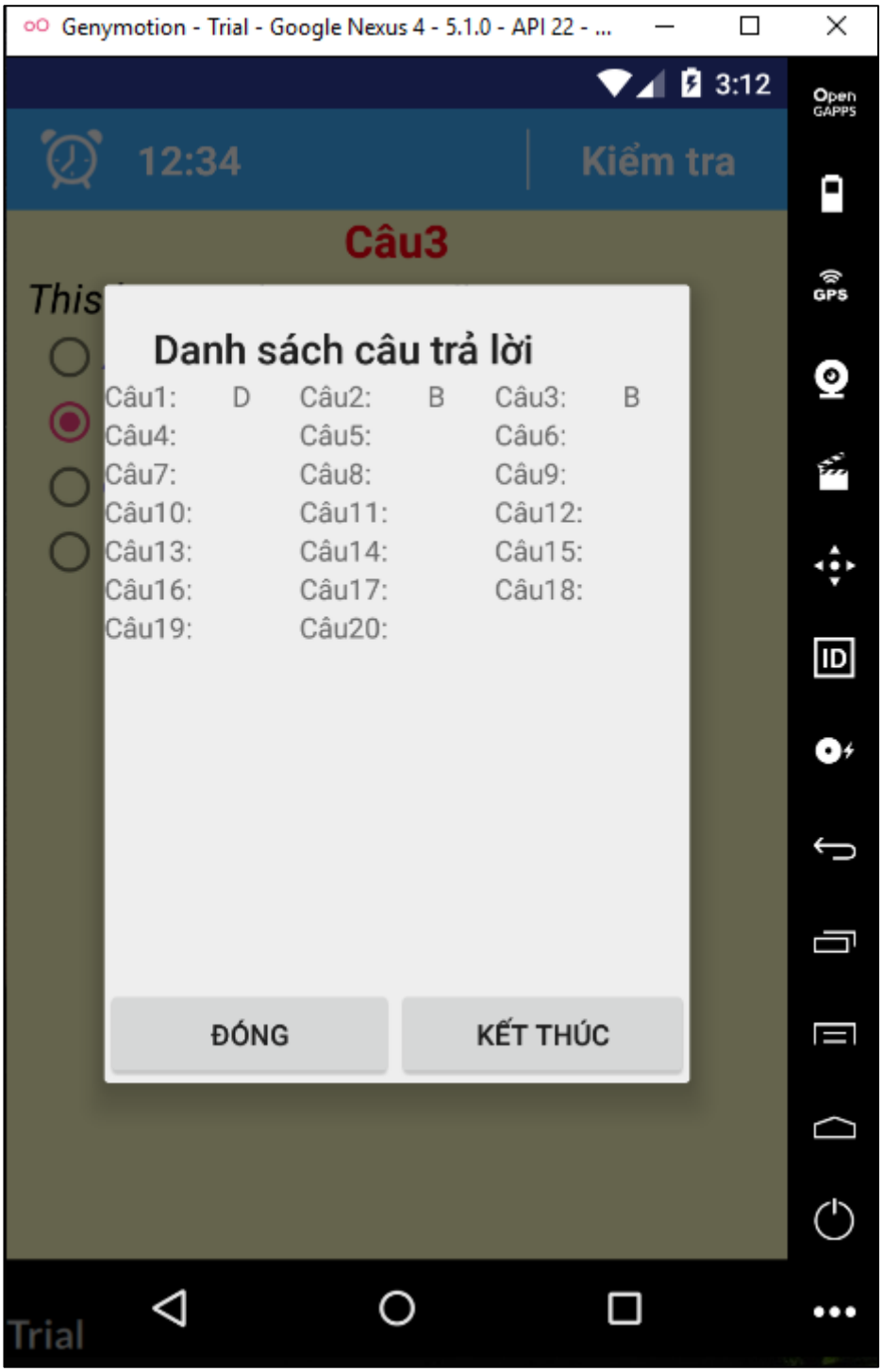


Hình 4.2.4. Giao diện trả lời 1



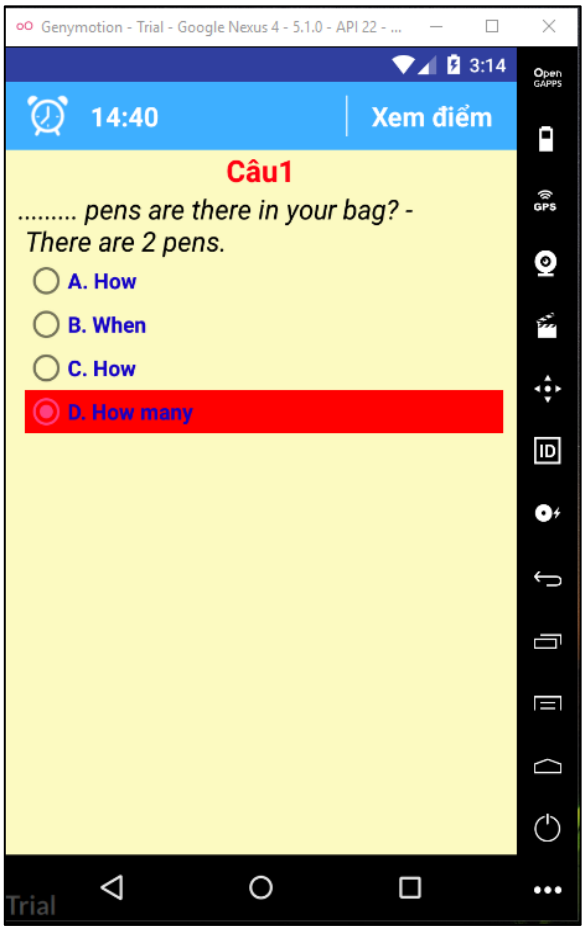
Hình 4.2.5. Giao diện trả lời 2

- Giao diện tổng kết câu đã trả lời

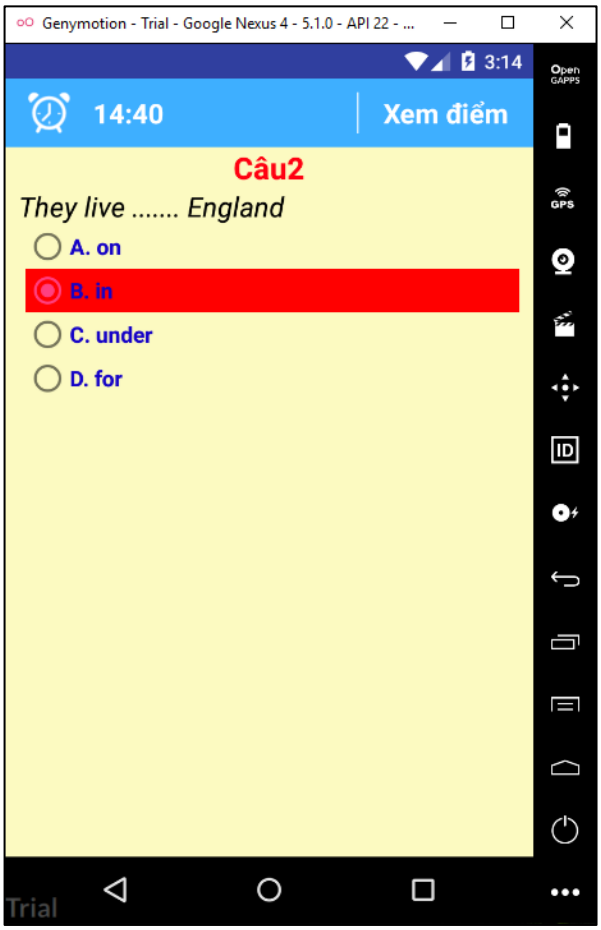


Hình 4.2.6. Giao diện xem câu trả lời

- Giao diện đáp án

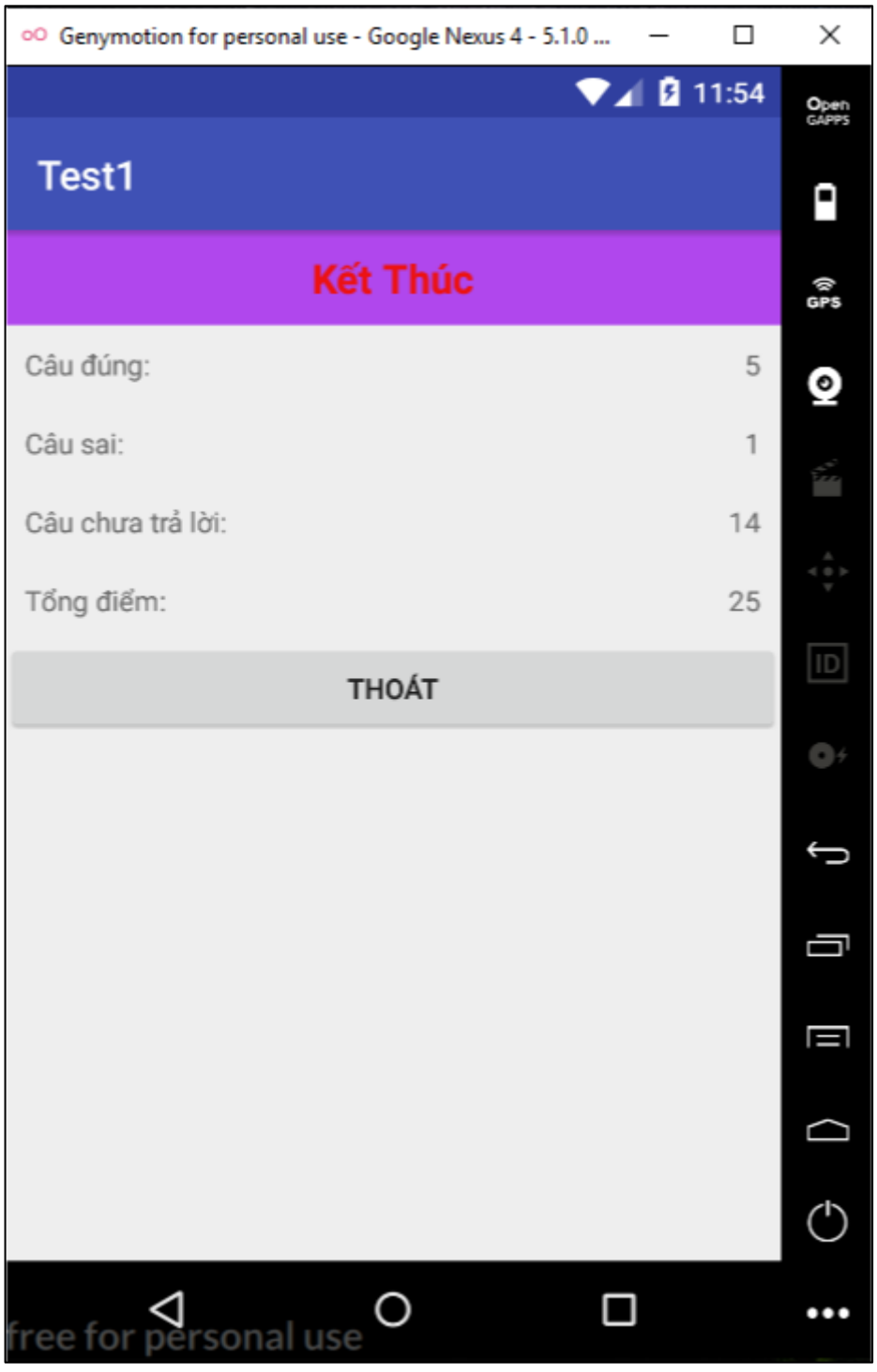


Hình 4.2.7. Giao diện đáp án 1



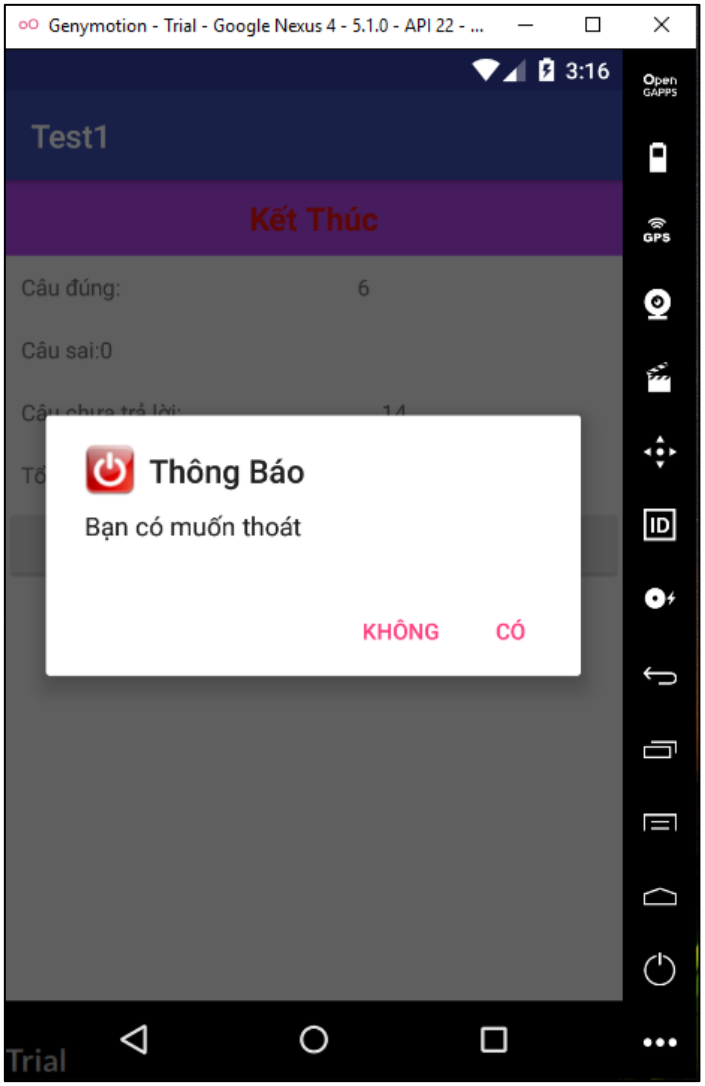
Hình 4.2.8. Giao diện đáp án 2

- Giao diện xem điểm:



Hình 4.2.9. Giao diện xem câu trả lời và xem điểm

- Xác nhận thoát chương trình:



Hình 4.2.10. Giao diện thoát về màn hình chọn đề

KẾT LUẬN

Sau bao nỗ lực và cố gắng cùng với sự hỗ trợ tích cực từ giáo viên hướng dẫn và nhà trường phần mền đồ án xây dựng ôn luyện trắc nghiệm tiếng Anh đã cơ bản hoàn thiện. Trong khoảng thời gian nhất định dành cho việc thực hiện đề tài, nên một số vấn đề và trình độ nên vẫn chưa được hoàn chỉnh, bộ đề còn ít. Tuy nhiên, đồ án đã đạt được một số kết quả:

-Về lý thuyết: Tìm hiểu, nghiên cứu được cách tạo cơ sở dữ liệu, các kỹ thuật lập trình với cơ sở dữ liệu để Xây dựng ứng dụng android ôn luyện trắc nghiệm tiếng anh như: xây dựng ứng dụng trắc nghiệm, lấy dữ liệu từ SQL, tìm kiếm dữ liệu, kỹ thuật lật trang, kỹ thuật đếm ngược thời gian từ Android.

-Về thực nghiệm: Bước đầu xây dựng thành công trong xây dựng bộ đề trắc nghiệm tiếng Anh trong Android.

Trong tương lai em sẽ tìm hiểu thêm và phát triển ứng dụng có thêm nhiều bộ đề, tải bộ đề được trên internet…

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo trực tuyến

[1]. https://developer.android.com/training/animation/screen-slide
[2]. https://hocweb.com.vn/lap-trinh-di-dong/
[3]. http://it.die.vn/s/sqlite/