**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**TRẦN QUỐC THÁI  
NGUYỄN PHI VIỄN**

**ĐỒ ÁN CHUYÊN NGÀNH**

**ỨNG DỤNG CHĂM SÓC SỨC KHỎE TRÊN ANDROID**

**CHUYÊN NGÀNH KỸ THUẬT PHẦN MỀM**

**TP. HỒ CHÍ MINH, 2014**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**TRẦN QUỐC THÁI – 11520360**

**NGUYỄN PHI VIỄN – 11520687**

**ĐỒ ÁN CHUYÊN NGÀNH**

**ỨNG DỤNG CHĂM SÓC SỨC KHỎE TRÊN ANDROID**

**CHUYÊN NGÀNH KỸ THUẬT PHẦN MỀM**

**GIẢNG VIÊN HƯỚNG DẪN**

**TRẦN ANH DŨNG**

**TP. HỒ CHÍ MINH, 2014**

MỤC LỤC

[**DANH MỤC HÌNH VẼ** 6](#_Toc435041200)

[**DANH MỤC BẢNG** 8](#_Toc435041201)

[**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN** 9](#_Toc435041202)

[**Chương 1: MỞ ĐẦU.** 11](#_Toc435041203)

[**1.1** **Lý do chọn đề tài.** 11](#_Toc435041204)

[**1.1.1** **Hiện trạng** 11](#_Toc435041205)

[**1.1.2** **Các phần mềm liên quan và vấn đề cần giải quyết.** 12](#_Toc435041206)

[**Chương 2: CƠ SỞ LÝ THUYẾT** 18](#_Toc435041207)

[**1.1** **Lịch sử android.** 18](#_Toc435041208)

[**1.2** **Delving với máy ảo Dalvik.** 19](#_Toc435041209)

[**1.3** **Kiến trúc của android.** 20](#_Toc435041210)

[**1.3.1** **Tầng ứng dụng** 21](#_Toc435041211)

[**1.3.2** **Application framework** 21](#_Toc435041212)

[**1.3.3** **Library** 22](#_Toc435041213)

[**1.3.4** **Android Runtime** 23](#_Toc435041214)

[**1.3.5** **Linux kernel** 23](#_Toc435041215)

[**1.4** **Android Emulator.** 24](#_Toc435041216)

[**1.5** **Các thành phần trong một android project.** 25](#_Toc435041217)

[**1.5.1** **AndroidManifest.xml** 25](#_Toc435041218)

[**1.5.2** **File R.java** 28](#_Toc435041219)

[**1.6** **Chu kì một ứng dụng android.** 28](#_Toc435041220)

[**1.6.1** **Chu kỳ sống thành phần** 28](#_Toc435041221)

[**1.6.2** **Activity Stack** 29](#_Toc435041222)

[**1.6.3** **Các trạng thái của chu kỳ sống** 29](#_Toc435041223)

[**1.6.4** **Chu kỳ sống của ứng dụng** 31](#_Toc435041224)

[**1.6.5** **Các sự kiện trong chu kỳ sống của ứng dụng** 31](#_Toc435041225)

[**1.6.6** **Thời gian sống của ứng dụng** 32](#_Toc435041226)

[**1.6.7** **Thời gian hiển thị của Activity** 32](#_Toc435041227)

[**1.6.8** **Các phương thức của chu kỳ sống** 32](#_Toc435041228)

[**1.7** **Các thành phần giao diện trong android.** 34](#_Toc435041229)

[**1.7.1** **View** 34](#_Toc435041230)

[**1.7.2** **ViewGroup** 35](#_Toc435041231)

[**1.7.3** **LinearLayout** 35](#_Toc435041232)

[**1.7.4** **FrameLayout** 36](#_Toc435041233)

[**1.7.5** **AbsoluteLayout** 37](#_Toc435041234)

[**1.7.6** **RetaliveLayout** 37](#_Toc435041235)

[**1.7.7** **TableLayout** 38](#_Toc435041236)

[**1.7.8** **Button** 39](#_Toc435041237)

[**1.7.9** **ImageButton** 40](#_Toc435041238)

[**1.7.10** **ImageView** 41](#_Toc435041239)

[**1.7.11** **ListView** 41](#_Toc435041240)

[**1.7.12** **TextView** 43](#_Toc435041241)

[**1.7.13** **EditText** 43](#_Toc435041242)

[**1.7.14** **CheckBox** 44](#_Toc435041243)

[**1.7.15** **MenuOptions** 45](#_Toc435041244)

[**1.7.16** **ContextMenu** 47](#_Toc435041245)

[**1.7.17** **Quick Search Box** 47](#_Toc435041246)

[**1.7.18** **Activity & Intent** 48](#_Toc435041247)

[**1.8** **Content Provider và Uri.** 53](#_Toc435041248)

[**1.9** **Background Service.** 53](#_Toc435041249)

[**1.10** **Telephony.** 57](#_Toc435041250)

[**1.11** **SQLite.** 57](#_Toc435041251)

[**1.12** **Android & WebService.** 58](#_Toc435041252)

[**1.12.1** **Khái niệm Web service và SOAP** 58](#_Toc435041253)

[**Chương 3: XÂY DỰNG ỨNG DỤNG.** 60](#_Toc435041254)

[**3.1** **Mô tả chức năng.** 60](#_Toc435041255)

[**3.1.1** **Chức năng đo nhịp tim** 60](#_Toc435041256)

[**3.1.2** **Chức năng tính chỉ số BMI.** 60](#_Toc435041257)

[**3.1.3** **Chức năng đo chỉ số WHR.** 61](#_Toc435041258)

[**3.1.4** **Chức năng đo số bước chạy.** 61](#_Toc435041259)

[**3.1.5** **Chức năng xem tin tức online.** 62](#_Toc435041260)

[**3.1.6** **Chức năng video call** 62](#_Toc435041261)

[**3.1.7** **Chức năng chẩn đoán bệnh dựa theo triệu chứng** 62](#_Toc435041262)

[**3.1.8** **Chức năng tra cứu thuốc** 62](#_Toc435041263)

[**3.2** **Thiết kế ứng dụng.** 62](#_Toc435041264)

[**3.2.1** **Use case Diagram** 62](#_Toc435041265)

[**3.2.2** **Package Diagram** 63](#_Toc435041266)

[**3.2.3** **Activity Diagram** 64](#_Toc435041267)

[**3.2.4** **Sequence Diagram** 67](#_Toc435041268)

[**3.2.5** **Thiết kế thuật toán** 72](#_Toc435041269)

[**3.2.6** **Màn hình ứng dụng.** 76](#_Toc435041270)

[**Chương 4: KẾT QUẢ** 82](#_Toc435041271)

[**4.1** **Giao diện ứng dụng.** 82](#_Toc435041272)

[**4.1.1** **Chức năng đo nhịp tim.** 82](#_Toc435041273)

[**4.1.2** **Chức năng tính BMI.** 84](#_Toc435041274)

[**4.1.3** **Chức năng tính WHR.** 84](#_Toc435041275)

[**4.1.4** **Chức năng đo bước chạy.** 85](#_Toc435041276)

[**4.1.5** **Chức năng lập lịch uống thuốc.** 85](#_Toc435041277)

[**4.1.6** **Màn hình thông tin phần mềm.** 86](#_Toc435041278)

[**4.2** **Đánh giá độ chính xác và các tác nhân ảnh hưởng.** 86](#_Toc435041279)

[**4.3** **Hạn chế** 88](#_Toc435041280)

[**4.4** **Hướng phát triển** 89](#_Toc435041281)

[**4.5** **Kết luận** 89](#_Toc435041282)

# **DANH MỤC HÌNH VẼ**

[Hình 1. 1: Màn hình đo nhịp tim của ứng dụng Runtastic Heart Rate. 13](#_Toc409169726)

[Hình 1. 2:. Màn hình đo nhịp tim của ứng dụng Heart Beat Rate 13](#_Toc409169727)

[Hình 1. 3: Màn hình tính chỉ số BMI. 15](#_Toc409169728)

[Hình 2. 1: Android timeline. 16](#_Toc409169757)

[Hình 2. 2: Cấu trúc stack hệ thống Android. 19](#_Toc409169758)

[Hình 2. 3: Android emulator. 23](#_Toc409169759)

[Hình 2. 4: Activity stack. 27](#_Toc409169760)

[Hình 2. 5: Chu kỳ sống của Activity. 28](#_Toc409169761)

[Hình 2. 6: Các phương thức trong một Activity. 29](#_Toc409169762)

[Hình 2. 7: Cấu trúc một giao diện ứng dụng Android. 33](#_Toc409169763)

[Hình 2. 8: Bố trí các widget sử dụng LinearLayout. 34](#_Toc409169764)

[Hình 2. 9: Bố trí các widget trong FrameLayout. 35](#_Toc409169765)

[Hình 2. 10: Bố trí widget trong RetaliveLayout. 36](#_Toc409169766)

[Hình 2. 11: Bố trí widget trong TableLayout. 37](#_Toc409169767)

[Hình 2. 12: ImageButon. 38](#_Toc409169768)

[Hình 2. 13: ImageView và ImageButton. 39](#_Toc409169769)

[Hình 2. 14: Minh hoạ cho một ListView. 41](#_Toc409169770)

[Hình 2. 15: Minh hoạ option menu. 45](#_Toc409169771)

[Hình 2. 16: Minh hoạ Quick Search Box. 46](#_Toc409169772)

[Hình 2. 17: Truyền dữ liệu giữa 2 Activity. 49](#_Toc409169773)

[Hình 2. 18: Các thuộc tính của Intent. 50](#_Toc409169774)

[Hình 2. 19: Các Action đã được định nghĩa sẵn trong Intent. 51](#_Toc409169775)

[Hình 2. 20: Chu trình sống của một Service. 54](#_Toc409169776)

[Hình 2. 21: SQLite Manager. 56](#_Toc409169777)

[Hình 3. 1: Sơ đồ cơ sở dữ liệu ứng dụng. 61](#_Toc409169778)

[Hình 3. 2: Sơ đồ Use case tổng thể. 66](#_Toc409169779)

[Hình 3. 3: Sơ đồ package ứng dụng. 67](#_Toc409169780)

[Hình 3. 4: Activity Diagram chức năng đo nhịp tim 68](#_Toc409169781)

[Hình 3. 5: Activity Diagram chức năng đo chỉ số BMI 69](#_Toc409169782)

[Hình 3. 6: Activity Diagram chức năng đo chỉ số WHR 69](#_Toc409169783)

[Hình 3. 7: Activity Diagram chức năng đo số bước chạy 70](#_Toc409169784)

[Hình 3. 8: Activity Diagram chức năng lập lịch uống thuốc 70](#_Toc409169785)

[Hình 3. 9: Sequence Diagram chức năng đo nhịp tim 71](#_Toc409169786)

[Hình 3. 10: Sequence Diagram chức năng tính BMI 72](#_Toc409169787)

[Hình 3. 11: Sequence Diagram chức năng tính WHR 73](#_Toc409169788)

[Hình 3. 12: Sequence Diagram chức năng lập lịch uống thuốc 74](#_Toc409169789)

[Hình 3. 13: Sequence Diagram chức năng đo số bước đi 75](#_Toc409169790)

[Hình 3. 14: Dạng tín hiệu nhịp tim. 75](#_Toc409169791)

[Hình 3. 15: Sự hấp thụ ánh sáng của động mạch khi truyền qua ngón tay. 76](#_Toc409169792)

[Hình 3. 16: Mô tả hoạt động thuật toán đo nhịp tim. 77](#_Toc409169793)

[Hình 3. 17: Màn hình đo nhịp tim. 77](#_Toc409169794)

[Hình 3. 18: Màn hình đo nhịp tim (kết quả) 78](#_Toc409169795)

[Hình 3. 19: Màn hình đo nhịp tim (Hướng dẫn) 78](#_Toc409169796)

[Hình 3. 20: Màn hình tính BMI 79](#_Toc409169797)

[Hình 3. 21: Màn hình tính WHR 80](#_Toc409169798)

[Hình 3. 22: Màn hình lập lịch uống thuốc 81](#_Toc409169799)

[Hình 3. 23: Màn hình đo số bước chạy 82](#_Toc409169800)

[Hình 3. 24: Màn hình thông tin thêm về phần mềm. 82](#_Toc409169801)

[Hình 4. 1: Màn hình đo nhịp tim. 83](#_Toc409169802)

[Hình 4. 2: Màn hình đo nhịp tim (kết quả). 84](#_Toc409169803)

[Hình 4. 3: Màn hình đo nhịp tim (Hướng dẫn). 84](#_Toc409169804)

[Hình 4. 4: Màn hình tính BMI. 85](#_Toc409169805)

[Hình 4. 5: Màn hình tính WHR. 85](#_Toc409169806)

[Hình 4. 6: Màn hình đo số bước chạy. 86](#_Toc409169807)

[Hình 4. 7: Màn hình lập lịch uống thuốc. 86](#_Toc409169808)

[Hình 4. 8: Màn hình thông tin phần mềm. 87](#_Toc409169809)

**DANH MỤC BẢNG**

[Bảng 3. 1: Mô tả bảng dữ liệu User 60](#_Toc409094889)

[Bảng 3. 2: Mô tả bảng dữ liệu RatioBMI 61](#_Toc409094890)

[Bảng 3. 3: Mô tả bảng dữ liệu RatioWHR 62](#_Toc409094891)

[Bảng 3. 4: Mô tả bảng dữ liệu StepRun 62](#_Toc409094892)

[Bảng 3. 5: Mô tả bảng dữ liệu TimeTableTake 63](#_Toc409094893)

[Bảng 4. 1: Các chỉ số BMI. 82](#_Toc409099794)

[Bảng 4. 2: Chỉ số WHR. 82](#_Toc409099795)

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

**MỞ ĐẦU**

Ngành công nghiệp di động phát triển nhanh chóng đã biến smartphone trở thành trung tâm của thời đại số. Sau nhiều cuộc đua từ cấu hình, tính năng, cho đến những trải nghiệm người dùng và ứng dụng của smartphone vào nhiều lĩnh vực trong cuộc sống, giờ đây các nhà sản xuất đang tập trung phát triển tính năng theo dõi và chăm sóc sức khỏe cho smartphone kết hợp cùng các thiết bị thông minh đeo trên người.

Thực tế là lĩnh vực chăm sóc sức khỏe cá nhân đã bắt đầu hưởng lợi từ cuộc cách mạng smartphone và các thiết bị đeo thông minh, như vòng đeo tay sức khỏe, smartwatch, hay những thiết bị khác tích hợp cảm biến có khả năng theo dõi tình trạng sức khỏe người dùng. Các thiết bị đeo trên người này hoạt động như những máy đo nhịp tim, huyết áp, thân nhiệt và nhiều dấu hiệu khác của cơ thể sẵn sàng phát tín hiệu cảnh báo khi có dấu hiệu bất thường. Và điều quan trọng là chúng có khả năng kết nối và gửi thông tin thu thập được cho ứng dụng chạy trên smartphone phân tích, xử lý và cho ra ngay những biểu đồ sức khỏe để người dùng tham khảo, điều chỉnh hành vi sinh hoạt, ăn uống, luyện tập có lợi cho sức khỏe. Thông tin còn có thể được ứng dụng trên smartphone cập nhật vào hồ sơ y tế điện tử cá nhân trên mây, hay gửi ngay đến cho bác sỹ điều trị để có những phân tích sâu hơn.

Nhóm sinh viên Trần Quốc Thái – Nguyễn Phi Viễn thực hiện phát triển ứng dụng chăm sóc sức khỏe trên android nhằm tìm hiểu hơn về cách áp dụng các kỹ thuật y khoa trên adroid. Đây là đồ án đầu tiên về lĩnh vực y tế nên sẽ có nhiều sai sót, mong nhận được nhận xét để nhóm có thể hoàn thiện hơn.

Sau cùng xin cảm ơn Thầy Trần Anh Dũng đã tận tình hướng dẫn chúng em để có thể hoàn thành đồ án này.

Nhóm sinh viên thực hiện

Trần Quốc Thái 11520360

Nguyễn Phi Viễn 11520687

# **Chương 1: MỞ ĐẦU.**

* 1. **Lý do chọn đề tài.**
     1. **Hiện trạng**

Nhịp tim phản ánh tình trạng sức khỏe cũng như cảm xúc của con người. Nếu nhịp tim không ổn định đó là dấu hiệu bệnh lý, tùy mức độ mà có thể là những bệnh lý khác nhau. Đối với người bệnh tim mạch cần theo dõi nhịp tim thường xuyên và cần hạn chế sự không ổn định đó.

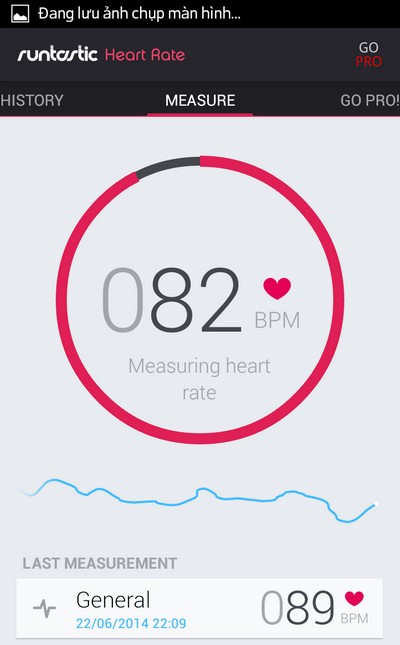
Trong sinh hoạt hằng ngày người dùng đôi khi có nhu cầu theo dõi tình hình thể trạng của mình như BMI – chỉ số khối cơ thể (tỉ số giữa cân nặng và chiều cao) hoặc chỉ số WHR – tỉ số vòng eo và vòng mông. Chỉ số BMI giúp người dùng xác định người dùng đang bị bệnh béo phì hay bị bệnh suy dinh dưỡng. Chỉ số WHR giúp người dùng xác định tỉ lệ phối mỡ trên cơ thể (bổ sung phần còn thiếu cho chỉ số BMI), khi người dùng có một chỉ số WHR lý tưởng thì ngoài việc tăng sức hấp dẫn còn giảm nguy cơ mắc các bệnh tiểu đường, rối loại tim mạch, ung thư buồng trứng (đối với nữ) và ung thư tuyến tiền liệt, ung thư tinh hoàn (đối với nam).

Sau công việc mệt mỏi người dùng cần tập thể dục để giảm stress cũng như có được một sức khỏe tốt hơn. Loại hình thường thấy ở mọi người là đi bộ hoặc chạy bộ. Phương pháp tập thể dục này rất hiệu quả, khi người dùng đi bộ đúng phương pháp sẽ góp phần làm cho hệ cơ bắp thêm dẻo dai, các khớp vận động tốt, hệ tuần hoàn lưu thông một cách thông suốt và mạnh hơn, đi bộ có tác dụng tốt cho những người bị bệnh lý về tim mạch, đi bộ còn giúp làm mạnh mẽ và an tĩnh thần kinh, phòng tránh nhức đầu, mất ngủ, và chứng trầm cảm, ngoài ra đi bộ còn giúp hạ huyết áp, phòng ngừa cảm cúm và là liệu pháp giảm béo tốt nhất và tạo cho con người có eo thon thả và dáng người gọn gàng, đẹp. Đi bộ giúp tâm trạng thư thái, thoải mái vì do tập trung tâm trí vào thời điểm hiện tại và kết nối với thiên nhiên.

Hiện nay điện thoại smartphone rất phổ biến, với số tiền không quá cao họ có thể mua cho mình một chiếc điện thoại với đầy đủ các tính năng. Nhận thấy tình hình này nhóm em chọn đề tài phát triển ứng dụng chăm sóc sức khỏe trên android nhằm giúp người dùng theo dõi tình trạng sức khỏe mà không cần thêm các thiết bị chuyên dụng khác. Phần mềm sẽ cung cấp một số chức năng như:

* Đo nhịp tim.
* Tính chỉ số BMI.
* Tính chỉ số WHR.
* Đo số bước chạy, tính số calo tiêu thụ trong này, tính số quãng đường đi được trong ngày.
* Tra cứu thông tin thuốc
* Giao tiếp thông qua video call
* Chẩn đoan bệnh dựa theo triệu chứng.
  + 1. **Các phần mềm liên quan và vấn đề cần giải quyết.**
* Runtastic Heart Rate (tương thích Android 2.2 và iOS 6.0 trở lên)

Đây là ứng dụng của Runtastic, hãng nổi tiếng với các ứng dụng và thiết bị theo dõi sức khỏe trên người.

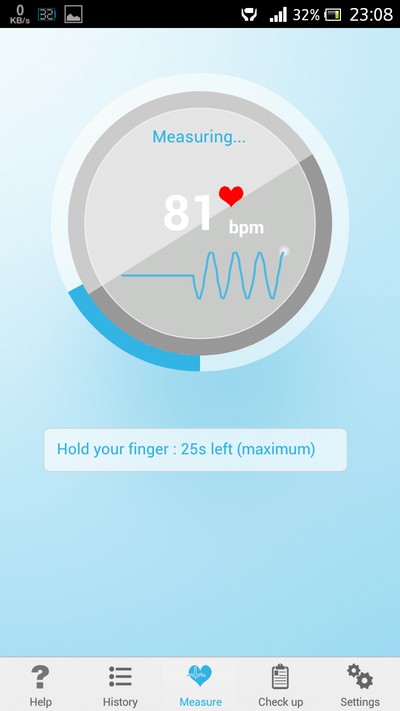


Hình 1. 1: Màn hình đo nhịp tim của ứng dụng Runtastic Heart Rate.

Cách thức sử dụng ứng dụng này tương tự như cảm biến nhịp tim trên Galaxy S5, người dùng sẽ đặt ngón tay của mình giữa camera và đèn flash của smartphone, thuật toán trên ứng dụng sẽ đo sự thay đổi độ sáng của đèn flash, từ đó tính toán ra sự thay đổi của mao mạch ngón tay và đưa ra kết quả về nhịp tim của người dùng.

Runtastic Heart Rate được đánh giá là ứng dụng đo nhịp tim tốt nhất hiện nay trên nền tảng Android. Tuy nhiên nhược điểm của ứng dụng đó là đối với phiên bản miễn phí chỉ cho phép người dùng đo nhịp tim tối đa 3 lần trong mỗi ngày. Nếu muốn có số lần đo nhiều hơn người dùng cần phải bỏ tiền ra để mua phiên bản có phí của ứng dụng.

* Heart Beat Rate (tương thích Android 2.2 trở lên và iOS 4.3 trở lên)



Hình 1. 2:. Màn hình đo nhịp tim của ứng dụng Heart Beat Rate

Cách thức hoạt động và cơ chế đo nhịp tim của ứng dụng Heart Beat Rate cũng tương tự như ứng dụng kể trên. Sau khi cài đặt ứng dụng, người dùng đặt ngón tay vào camera của smartphone, sau đó nhấn nút Start trên giao diện chính của ứng dụng để bắt đầu quá trình đo nhịp tim.

Sau khi quá trình đo nhịp tim hoàn tất, ứng dụng sẽ cho phép người dùng lựa chọn trạng thái hiện tại của người dùng là đang nghỉ ngơi (Resting) hay đang vận động, tập luyện (Training).

Từ kết quả đo, kết hợp với trạng thái hiện tại của người dùng, ứng dụng sẽ đưa ra lời khuyên cho biết nhịp tim của người dùng ở mức bình thường (Normal), cao (High) hoặc quá chậm (Low). Đồng thời ứng dụng cũng đưa ra những lời khuyên cho người dùng thường xuyên kiểm tra nhịp tim, đặc biệt vào thời điểm buổi sáng khi mới thức dậy, vào thời điểm sau khi vận động hoặc sau khi ăn xong...

Ứng dụng này không hạn chế số lần đo nhịp tim trong ngày của người dùng nhưng bản miễn phí của nó lại không có chức năng lưu lại lịch sử kết quả đo.

* Ứng dụng Chỉ số sức khỏe BMI



Hình 1. 3: Màn hình tính chỉ số BMI.

Đây là ứng dụng đo có chức đo chỉ số BMI của người dùng. Ứng dụng cho phép nhập vào thông tin chiều cao, cân nặng và chọn giới tính sau đó xuất ra kết quả cho người dùng, phần mềm chp phép lưu lại lịch sử đo đạt. Tuy nhiên phần mềm này chưa hoàn thiện vì chỉ số khối con người chỉ nói lên được sự liên quan giữa chiều cao cân nặng chứ về mức độ phân phối mỡ thì vẫn chưa đánh giá được.

* Ứng dụng Google Fit



Hình 1. 4: Màn hình ứng dụng Google Fit

Ứng dụng có chức năng đo số bước đi trong ngày, tính số calo tiêu thụ trong ngày, tính số quãng đường đi được và tính được số giờ tập thể dục trong ngày.

* Ứng dụng MedScape



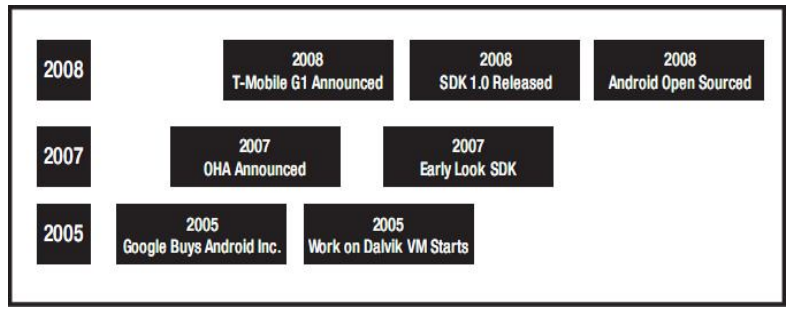
Hình 1. 5: Màn hình ứng dụng MedScape

Trên thực tế vẫn còn nhiều ứng dụng đáp ứng được những như cầu này tuy nhiên đa số các chức năng đều rời rạc hoặc phải trả phí cho một ứng dụng hoàn thiện, người dùng muốn sử dụng thì phải cài nhiều ứng dụng hoặc phải trả phí. Vấn đề cần giải quyết là cần tích hợp lại chức năng của các phần mềm và cải tiến sao cho nó phù hợp với người dùng và phát hành miễn phí.

# **Chương 2: CƠ SỞ LÝ THUYẾT**

* 1. **Lịch sử android.**

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



Hình 2. 1: Android timeline.

Và sau tiếp, vào cuối năm 2007, thuộc về Liên minh Thiết bị Cầm tay Mã Nguồn mở (Open Handset Alliance) gồm các thành viên nổi bật trong ngành viễn thông và thiết bị cầm tay như:

Texas Instruments, Broadcom Corporation, Google, HTC, Intel, LG, Marvell Technology Group, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel, T-Mobile, ARM Holdings, Atheros Communications, Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, and Vodafone Group,…

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

Phiên bản SDK lần đầu tiên phát hành vào tháng 11 năm 2007, hãng T-Mobile cũng công bố chiếc điện thoại Android đầu tiên đó là chiếc T-Mobile G1, chiếc smartphone đầu tiên dựa trên nền tảng Android. Một vài ngày sau đó, Google lại tiếp tục công bố sự ra mắt phiên bản Android SDK release Candidate 1.0. Trong tháng 10 năm 2008, Google được cấp giấy phép mã nguồn mở cho Android Platform.

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

Vào cuối năm 2008, Google cho phát hành một thiết bị cầm tay được gọi là Android Dev Phone có thể chạy được các ứng dụng Android mà không bị ràng buộc vào các nhà cung cấp mạng điện thoại di động. Mục tiêu của thiết bị này là cho phép các nhà phát triển thực hiện các cuộc thí nghiệm trên một thiết bị thực có thể chạy hệ điều hành Android mà không phải ký một bản hợp đồng nào. Vào khoảng cùng thời gian đó thì Google cũng cho phát hành một phiên vản vá lỗi 1.1 của hệ điều hành này. Ở cả hai phiên bản 1.0 và 1.1 Android chưa hỗ trợ soft-keyboard mà đòi hỏi các thiết bị phải sử dụng bàn phím vật lý. Android cố định vấn đề này bằng cách phát hành SDK 1.5 vào tháng Tư năm 2009, cùng với một số tính năng khác. Chẳng hạn như nâng cao khả năng ghi âm truyền thông, vật dụng, và các live folder.

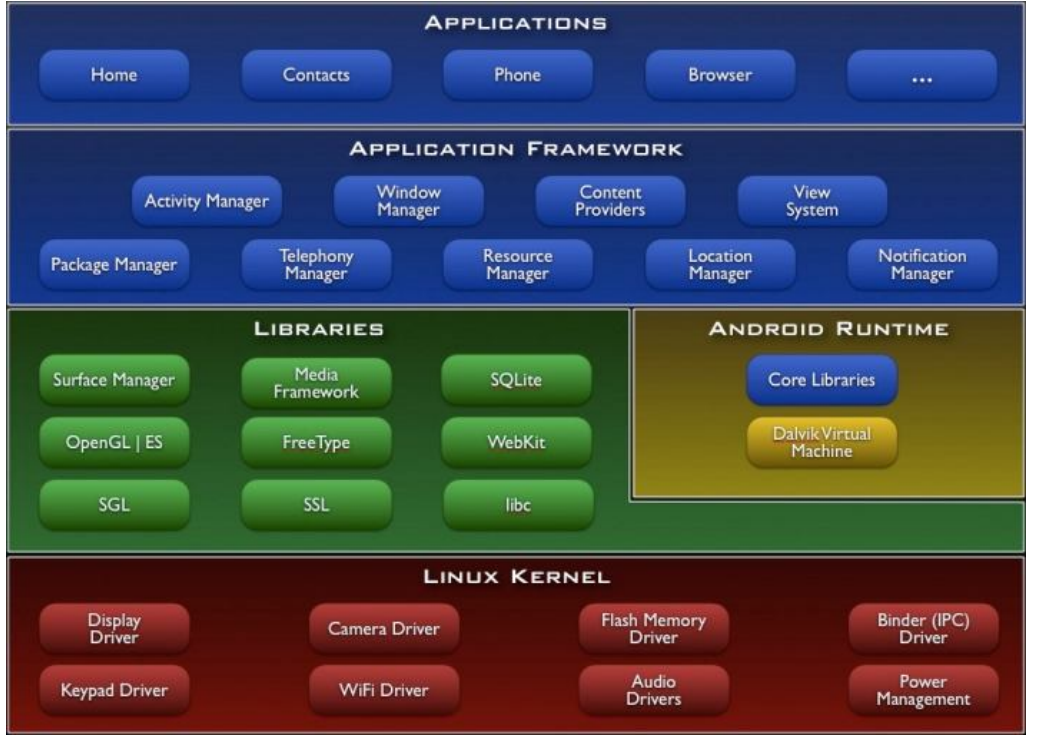
* 1. **Delving với máy ảo Dalvik.**

Dalvik là máy ảo giúp các ứng dụng java chạy được trên các thiết bị động Android. Nó chạy các ứng dụng đã được chuyển đổi thành một file thực thi Dalvik (dex). Định dạng phù hợp cho các hệ thống mà thường bị hạn chế về bộ nhớ và tốc độ xử lý. Dalvik đã được thiết kế và viết bởi Dan Bornstein, người đã đặt tên cho nó sau khi đến thăm một ngôi làng đánh cá nhỏ có tên là Dalvík, nơi mà một số tổ tiên của ông sinh sống.

Từ góc nhìn của một nhà phát triển, Dalvik trông giống như máy ảo Java (Java Virtual Machine) nhưng thực tế thì hoàn toàn khác. Khi nhà phát triển viết một ứng dụng dành cho Android, anh ta thực hiện các đoạn mã trong môi trường Java. Sau đó, nó sẽ được biên dịch sang các bytecode của Java, tuy nhiên để thực thi được ứng dụng này trên Android thì nhà phát triển phải thực thi một công cụ có tên là dx. Đây là công cụ dùng để chuyển đổi bytecode sang một dạng gọi là dex bytecode. "Dex" là từ viết tắt của "Dalvik executable" đóng vai trò như cơ chế ảo thực thi các ứng dụng Android.

* 1. **Kiến trúc của android.**

Mô hình sau thể hiện một cách tổng quát các thành phần của hệ điều hành Android. Mỗi một phần sẽ được đặc tả một cách chi tiết dưới đây.



Hình 2. 2: Cấu trúc stack hệ thống Android.

* + 1. **Tầng ứng dụng**

Android được tích hợp sẵn một số ứng dụng cần thiết cơ bản như: contacts, browser, camera, Phone,… Tất cả các ứng dụng chạy trên hệ điều hành Android đều được viết bằng Java.

* + 1. **Application framework**

Bằng cách cung cấp một nền tảng phát triển mở, Android cung cấp cho các nhà phát triển khả năng xây dựng các ứng dụng cực kỳ phong phú và sáng tạo. Nhà phát triển được tự do tận dụng các thiết bị phần cứng, thông tin địa điểm truy cập, các dịch vụ chạy nền, thiết lập hệ thống báo động, thêm các thông báo để các thanh trạng thái, và nhiều, nhiều hơn nữa.

Nhà phát triển có thể truy cập vào các API cùng một khuôn khổ được sử dụng bởi các ứng dụng lõi. Các kiến trúc ứng dụng được thiết kế để đơn giản hóa việc sử dụng lại các thành phần; bất kỳ ứng dụng có thể xuất bản khả năng của mình và ứng dụng nào khác sau đó có thể sử dụng những khả năng (có thể hạn chế bảo mật được thực thi bởi khuôn khổ). Cơ chế này cho phép các thành phần tương tự sẽ được thay thế bởi người sử dụng.

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

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

Activity Maanager được dùng để quản lý chu trình sống của ứng dụng và điều hướng các activity.

* + 1. **Library**

Android bao gồm một tập hợp các thư viên C/C++ được sử dụng bởi nhiều thành phần khác nhau trong hệ thống Android. Điều này được thể hiện thông qua nền tảng ứng dụng Android. Một số các thư viện cơ bản được liệt kê dưới đây:

* **System C library:** a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices.
* **Media Libraries** **–** based on PacketVideo's OpenCORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
* **Surface Manager –** Quản lý việc truy xuất vào hệ thống hiển thị
* **LibWebCore –** a modern web browser engine which powers both the Android browser and an embeddable web view.
* **SGL –** the underlying 2D graphics engine.
* **3D libraries –** an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer.
* **FreeType** **–** bitmap and vector font rendering.

**SQLite** – a powerful and lightweight relational database engine available to all applications.

* + 1. **Android Runtime**

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

* + 1. **Linux kernel**

Android dựa trên Linux phiên bản 2.6 cho hệ thống dịch vụ cốt lõi như security, memory management, process management, network stack, and driver model. Kernel Linux hoạt động như một lớp trừu tượng hóa giữa phần cứng và phần còn lại của phần mềm stack.

* 1. **Android Emulator.**

Android SDK và Plugin Eclipse được gọi là một Android Deverloper Tool (ADT). Các Android coder sẽ cần phải sử dụng công cụ IDE (Integrated Development Enveronment) này để phát triển, debugging và testing cho ứng dụng. Tuy nhiên, các coder cũng có thể không cần phải sử dụng IDE mà thay vào đó là sử dụng command line để biên dịch và tất nhiên là vẫn có Emulator như thường.

Android Emulator được trang bị đầy đủ hầu hết các tính năng của một thiết bị thật. Tuy nhiên, một số đã bị giới hạn như là kết nối qua cổng USB, camera và video, nghe phone, nguồn điện giả lập và bluetooth.

Android Emulator thực hiện các công việc thông qua một bộ xử lý mã nguồn mở, công nghệ này được gọi là QEMU (http://bellard.org/qemu/) được phát triển bởi Fabrice Bellard.



Hình 2. 3: Android emulator.

* 1. **Các thành phần trong một android project.**
     1. **AndroidManifest.xml**

Trong bất kì một project Android nào khi tạo ra đều có một file AndroidManifest.xml, file này được dùng để định nghĩa các screen sử dụng, các permission cũng như các theme cho ứng dụng. Đồng thời nó cũng chứa thông tin về phiên bản SDK cũng như main activity sẽ chạy đầu tiên.

File này được tự động sinh ra khi tạo một Android project. Trong file manifest bao giờ cũng có 3 thành phần chính đó là: application, permission và version.

Dưới đây là nội dung của một file AndroidManifest.xml :

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

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

**package**=*"app.healthcare"*

**android:versionCode**=*"1"*

**android:versionName**=*"1.0"* >

<uses-sdk

**android:minSdkVersion**=*"14"*

**android:targetSdkVersion**=*"21"* />

<uses-feature **android:name**=*"android.hardware.camera"* />

<uses-feature **android:name**=*"android.hardware.camera.flash"* />

<uses-feature **android:name**=*"android.hardware.camera.autofocus"* />

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

<uses-permission **android:name**=*"android.permission.WAKE\_LOCK"* />

<application

**android:allowBackup**=*"true"*

**android:icon**=*"@drawable/ic\_launcher"*

**android:label**=*"@string/app\_name"*

**android:theme**=*"@style/AppTheme"* >

<activity

**android:name**=*".MainActivity"*

**android:label**=*"@string/app\_name"* >

<intent-filter>

<action **android:name**=*"android.intent.action.MAIN"* />

<category **android:name**=*"android.intent.category.LAUNCHER"* />

</intent-filter>

</activity>

<!-- Entry for RegisterActivity.class -->

<service

**android:name**=*".StepRunServices"*

**android:exported**=*"false"*/>

</application>

</manifest>

* **Application**

Thẻ <application>, bên trong thẻ này chứa các thuộc tính được định nghĩa cho ứng dụng Android như:

* **android:icon =** “drawable resource” 🡪 Ở đây đặt đường dẫn đến file icon của ứng dụng khi cài đặt. VD: android:icon = “@drawable/icon”.
* **android:name =** “string” 🡪 thuộc tính này để đặt tên cho ứng dụng Android. Tên này sẽ được hiển thị lên màn hình sau khi cài đặt ứng dụng.
* **android:theme =** “drawable theme” 🡪 thuộc tính này để đặt theme cho ứng dụng. Các theme là các cách để hiển thị giao diện ứng dụng.

Ngoài ra còn nhiều thuộc tính khác…

* **Permission**

Bao gồm các thuộc tính chỉ định quyền truy xuất và sử dụng tài nguyên của ứng dụng. Khi cần sử dụng một loại tài nguyên nào đó thì trong file manifest của ứng dụng cần phải khai báo các quy ền truy xuất như sau:

<uses-feature **android:name**=*"android.hardware.camera"* />

<uses-feature **android:name**=*"android.hardware.camera.flash"* />

<uses-feature **android:name**=*"android.hardware.camera.autofocus"* />

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

<uses-permission **android:name**=*"android.permission.WAKE\_LOCK"* />

SDK version

Thẻ xác định phiên bản SDK được khai báo như sau:

<uses-sdk

**android:minSdkVersion**=*"14"*

**android:targetSdkVersion**=*"21"* />

Ở đây chỉ ra phiên bản SDK nhỏ nhất mà ứng dụng hiện đang sử dụng.

* + 1. **File R.java**

File R.java là một file tự động sinh ra ngay khi tạo ứng dụng, file này được sử dụng để quản lý các thuộc tính được khai báo trong file XML của ứng dụng và các tài nguyên hình ảnh. Mã nguồn của file R.java được tự động sinh khi có bất kì một sự kiện nào xảy xa làm thay đổi các thuộc tính trong ứng dụng. Chẳng hạn như, kéo và thả một file hình ảnh từ bên ngoài vào project thì ngay lập tức thuộc tính đường dẫn đến file đó cũng sẽ được hình thành trong file R.java hoặc xoá một file hình ảnh thì đường dẫn tương ứng đến hình ảnh đó cũng tự động bị xoá.

Có thể nói file R.java hoàn toàn không cần phải đụng chạm gì đến trong cả quá trình xây dựng ứng dụng.

* 1. **Chu kì một ứng dụng android.**

Một tiến trình Linux gói gọn một ứng dụng Android đã được tạo ra cho ứng dụng khi codes cần được run 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 bới chính nó. Thay vào đó, nó được xác định bởi hệ thống qua một kết hợp của:

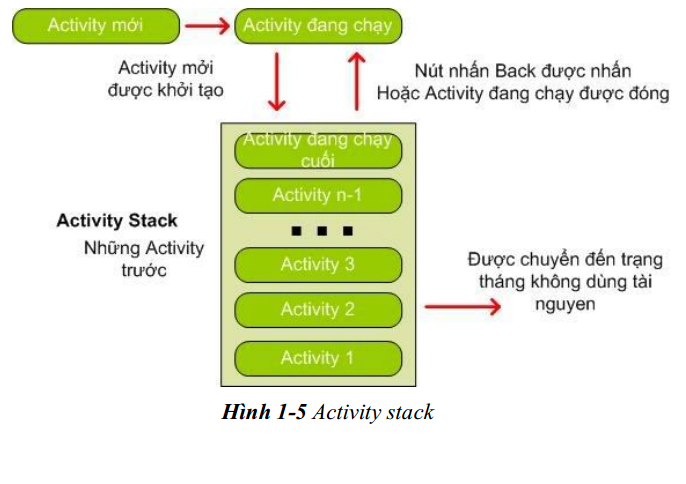
* Những phần của ứng dụng mà hệ thống biết đang chạy
* Những phần quan trọng như thế nào đối với người dùng
* Bao nhiêu vùng nhớ chiếm lĩnh trong hệ thống.
  + 1. **Chu kỳ sống thành phần**

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. Giữa đó, đôi lúc chúng có thể là active hoặc inactive, hoặc là trong trường hợp activies nó có thể visible hoặc invisible.

* + 1. **Activity Stack**

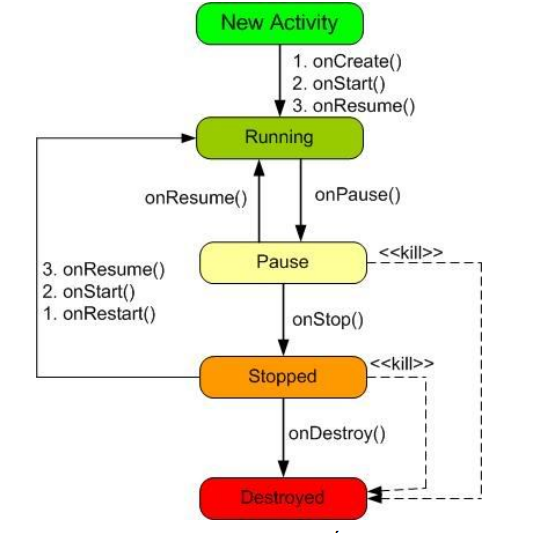
Bên trong hệ thống các activity được quản lý như một activity stack. Khi một Activity mới được start, nó được đặt ở đỉnh của stack và trở thành activity đang chạy activity trước sẽ ở bên dưới activity mới và sẽ không thấy trong suốt quá trình activity mới tồn tại.

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



Hình 2. 4: Activity stack.

* + 1. **Các trạng thái của chu kỳ sống**



Hình 2. 5: Chu kỳ sống của Activity.

Một Activity chủ yếu có 3 chu kỳ chính sau:

* Active hoặc running: Khi Activity 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.

* + 1. **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.

* + 1. **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 finish() (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. 6: Các phương thức trong một Activity.

Tất cả các phương thức là những móc nối mà có thể override để làm tương thích 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 dùng.

* + 1. **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().

* + 1. **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.

* + 1. **Các phương thức của chu kỳ sống**

**Phương thức**: onCreate()

* Được gọi khi activity lần đầu tiên được tạo
* Ở đây làm tất cả các cài đặt tĩnh, tạo các view, kết nối dữ liệu đến list và .v.v
* Phương thức này gửi qua một đối tượngBundle chứa đựng từ trạng thái trược của Activity
* Luôn theo sau bởi onStart()

**Phương thức:** onRestart()

* Được gọi sau khi activity đã được dừng, chỉ một khoảng đang khởi động lần nữa (stared again)
* Luôn theo sau bởi onStart()

**Phương thức:** onStart()

* Được gọi trước khi một activity visible với người dùng.
* Theo sau bởi onResume() nếu activity đến trạng thái foreground hoặc onStop() nếu nó trở nên ẩn.

**Phương thức:** onResume()

* Được gọi trước khi activity bắt đầu tương tác với người dùng
* Tại thời điểm này activity ở trên đỉnh của stack activity.
* Luôn theo sau bởi onPause()

**Phương thức**: onPause()

* Được gọi khi hệ thống đang resuming activity khác.
* Phương thức này là điển hình việc giữ lại không đổi dữ liệu.
* Nó nên được diễn ra một cách nhanh chóng bởi vì activity kế tiếp sẽ không được resumed ngay cho đến khi nó trở lại.
* Theo sau bởi onResume nếu activity trở về từ ở trước, hoặc bởi onStop nếu nó trở nên visible với người dùng.
* Trạng thái của activity có thể bị giết bởi hệ thống.

**Phương thức:** onStop()

* Được gọi khi activity không thuộc tầm nhìn của người dùng.
* Nó có thể diễn ra bởi vì nó đang bị hủy, hoặc bởi vì activity khác vữa được resumed và bao phủ nó.
* Được theo sau bởi onRestart() nếu activity đang đở lại để tương tác với người dùng, hoặc onDestroy() nếu activity đang bỏ.
* Trạng thái của activity có thể bị giết bởi hệ thống.

**Phương thức:** onDestroy()

* Được gọi trước khi activity bị hủy.
* Đó là lần gọi cuối cùng mà activity này được nhận.
* Nó được gọi khác bởi vì activity đang hoàn thành, hoặc bởi vì hệ thống tạm thởi bị hủy diệt để tiết kiệm vùng nhớ.
* Có thể phân biệt giữa 2 kịch bản với phương isFinshing().

Trạng thái của activity có thể được giết bởi hệ thống.

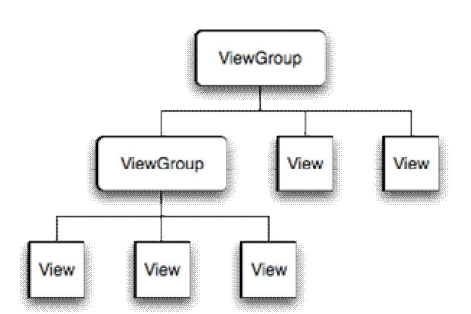
* 1. **Các thành phần giao diện trong android.**
     1. **View**

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kiểu View và ViewGroup. Mỗi một kiểu là một hậu duệ của class View và tất cả các kiểu đó được gọi là các Widget.

Tất cả mọi widget đều có chung các thuộc tính cơ bản như là cách trình bày vị trí, background, kích thước, lề,… Tất cả những thuộc tính chung này được thể hiện hết ởtrong đối tượng View.

Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc phân cấp như hình dưới. 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

XML lên để phân tích thành mã bytecode.



Hình 2. 7: Cấu trúc một giao diện ứng dụng Android.

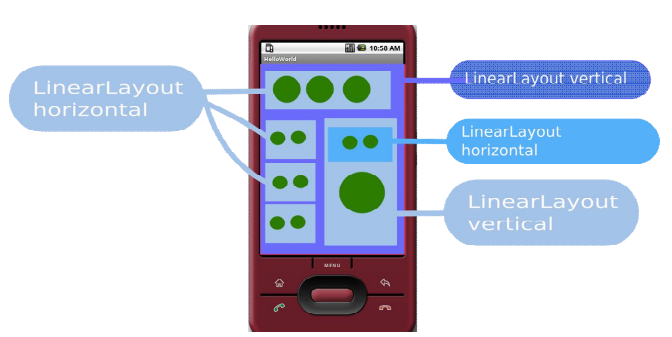
* + 1. **ViewGroup**

ViewGroup thực ra chính là View hay nói đúng hơn thì ViewGroup chính là các widget Layout được dùng để bố trí các đối tượng khác trong một screen. Có một sốloại ViewGroup như sau:

* + 1. **LinearLayout**

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.

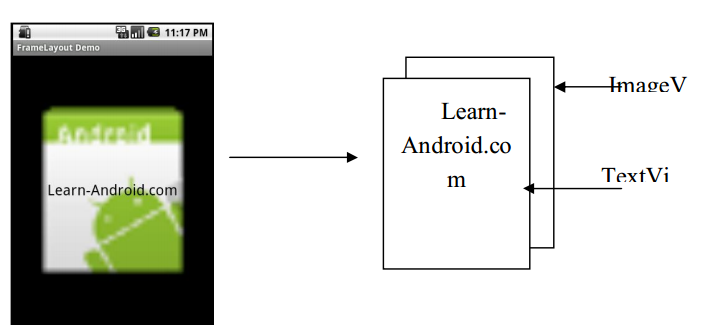
LinearLayout làm cho các thành phần trong nó không bị phụ thuộc vào kích thước của màn hình. Các thành phần trong LinearLayout được dàn theo những tỷ lệ cân xứng dựa vào các ràng buộc giữa các thành phần.



Hình 2. 8: Bố trí các widget sử dụng LinearLayout.

* + 1. **FrameLayout**

FrameLayout được dùng để bố trí các đối tượng theo kiểu giống như là các Layer trong Photoshop. Những đối tư ợng 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.



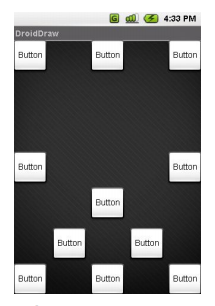
Hình 2. 9: Bố trí các widget trong FrameLayout.

* + 1. **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 th ước với màn hình thiết kế ban đầu thì vị trí của các đối tượng sẽ không còn được chính xác như ban đầu.

* + 1. **RetaliveLayout**

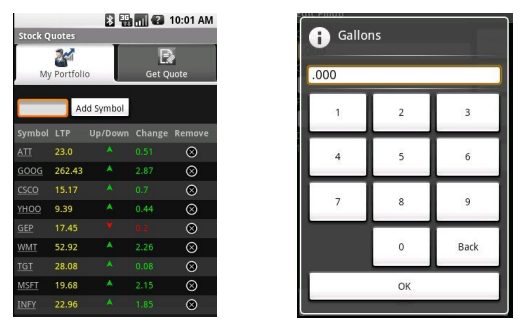
Layout này cho phép bố trí các widget theo một trục đối xứng ngang hoặc dọc. Để đặt được đúng vị trí thì các widget cần được xác định một mối ràng buộc nào đó với các widget khác. Các ràng buộc này là các ràng buộc trái, phải, trên, dưới so với một widget hoặc so với layout parent. Dựa vào những mối 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.



Hình 2. 10: Bố trí widget trong RetaliveLayout.

* + 1. **TableLayout**

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.



Hình 2. 11: Bố trí widget trong TableLayout.

* + 1. **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.

Để thiết kế giao diện với một button ta có 2 cách như sau:

Thiết kế bằng XML

<uses-feature **android:name**=*"android.hardware.camera"* />

<uses-feature **android:name**=*"android.hardware.camera.flash"* />

<Button

**android:layout\_width=***"wrap\_content"*

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

**android:id=***"@+id/cmdButton1"*

**android:text=***"Touch me!"*

**android:onClick=***"touchMe"*/>

Thuộc tính android:onClick="touchMe" được dùng để nắm bắt sự kiện click vào button. Khi sự kiện click button xảy ra thì phương thức “touchMe” được khai báo trong thẻ thuộc tính sẽ được gọi. Nếu trường hợp phương thức “touchMe” chưa được khai báo trong file mã nguồn tương ứng thì sẽ phát sinh một exception. Ngược lại, phương thức “touchMe” sẽ nhận được một đối tham biến là đối tượng View nơi đã phát sinh ra sự kiện. Đối tượng View này có thể ép kiểu trực tiếp sang kiểu Button vì thực chất nó là một button.

VD: trong file mã nguồn khai báo một hàm như sau:

**public void** touchMe(**View** v){

**Button** *me* =(**Button**) v;

*me*.setText(“Touched”);

}

* Thiết kế bằng code

Thực ra mà nói thì nếu không phải đòi hỏi phải custom lại một widget thì không cần phải sử dụng tới code. Trong một số trường hợp bắt buộc chúng ta phải custom các widget đ ể cho phù hợp với hoàn cảnh. Chẳng hạn như trong game, các menu hay các nút điều khiển,…

* + 1. **ImageButton**

Cũng tương tự như Button, ImageButton chỉ có thêm một thuộc tính

**android:src** *= “@drawable/icon”*

để thêm hình ảnh vào và không có thẻ text

<ImageButton

**android:layout\_width***="wrap\_content"*

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

**android:id***="@+id/cmdButton1"*

**android:src***="@drawable/icon"*

**android***:onClick="touchMe"*/>



Hình 2. 12: ImageButon.

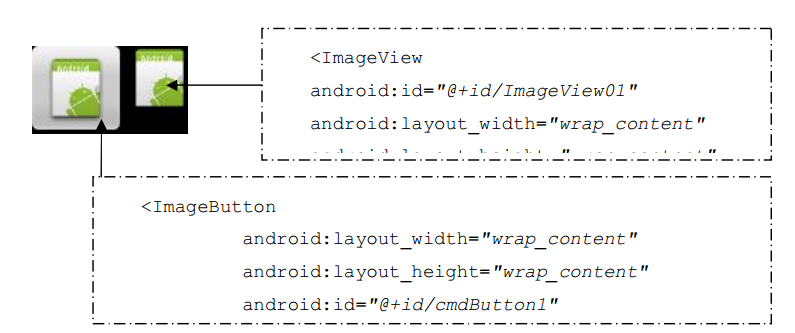
* + 1. **ImageView**

Được dùng để th ể hiện một hình ảnh. Nó cũng giống như ImageButton, chỉ khác là không có hình dáng của một cái button.

Code:

**ImageView** *iv* = new **ImageView**(**this**);

*iv*.setImageResource(R.drawable.*icon*);



Hình 2. 13: ImageView và ImageButton.

* + 1. **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 đ ã 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.

<ImageView

**android:id***="@+id/ImageView01"*

**android**:**layout\_width***="wrap\_content"*

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

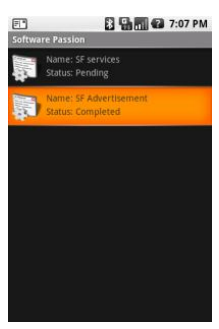
<ImageButton

**android:layout\_width***="wrap\_content"*

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

**android:id***="@+id/cmdButton1"*

* **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.



Hình 2. 14: Minh hoạ cho một ListView.

* + 1. **TextView**

TextView ngoài tác dụng là để hiển thị văn bản thì nó còn cho phép định dạng nội dung bằng thẻ html.

VD:

**TextView** *textView* =   
 (**TextView**)findViewById(R.id.*textView*);

**CharSequence** *styledText* =

**Html**.fromHtml("<i>This</i> is some <b>styled</b> <s>text</s>*"*);

*textView*.setText(*styledText*);

Nội dung TextView cũng có thể được định dạng bằng thẻ html ngay trong XML.

* + 1. **EditText**

Trong Android đối tượng EditText được sử dụng như một TextField hoặc một TextBox.

<EditText

**android:id***="@+id/EditText01"*

**android:layout\_width***="wrap\_content"*

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

**android:textStyle***="bold"*

**android:textSize***="20dip"*

**android:textColor***="#000000"*

**android:text***="Hello Android!"*

**android:singleLine***="true"*

**android:inputType***="textCapWords*"/>

Các thuộc tính cần chú ý sử dụng EditText đó là:

**android:inputType** *= “…”* sử dụng để xác định phương thức nhập cho EditText. Chẳng hạn như khi muốn một ô để nhập password hay một ô để nhập Email thì thuộc tính này sẽ làm điều đó.

**android:singleLine** *= “true”* EditText của sẽ trở thành một TextField, ngược lại sẽ là TextBox.

* + 1. **CheckBox**

Nhận 2 giá trị true hoặc false. Đối tượng CheckBox cho phép chọn nhiều item cùng một lúc.

Khai báo: **CheckBox** *cb* = new **CheckBox**(Context …);

XML:

<CheckBox

**android:id***="@+id/CheckBox01"*

**android:layout\_width***="wrap\_content"*

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

**android:text***="Check me"*

**android:checked***="true"/>*

* + 1. **MenuOptions**

Có 2 cách tạo một MenuOptions:

Tạo bằng code:

**public** **class** Main **extends** Activity {

**private** **int** *searchBtnId* = Menu.*FIRST*;

**private** **int** *scheduleBtnId* = Menu.*FIRST* + 1;

**private** **int** *playBtnId* = Menu.*FIRST* + 2;

**private** **int** *stopBtnId* = Menu.*FIRST* + 3;

**private** **int** *group1Id* = 1;

**private** **int** *group2Id* = 2;

@Override

**public** **void** onCreate(**Bundle** savedInstanceState) {

**super**.onCreate(savedInstanceState);

**this**.setContentView(R.layout.*main*);

}

@Override

**public** **boolean** onCreateOptionsMenu(**Menu** menu) {

menu.add(group1Id,searchBtnId,searchBtnId,"Search");

menu.add(group2Id,scheduleBtnId,scheduleBtnId,R.string.*schedule*);

menu.add(*group2Id*,*playBtnId* ,*playBtnId***,"**Play**"**);

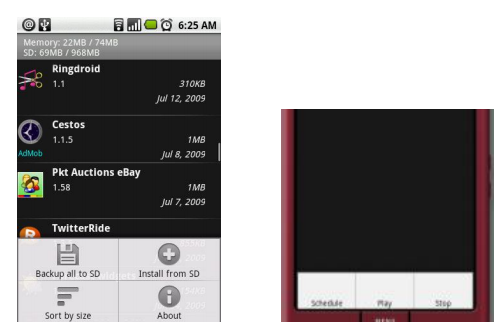
menu.add(*group2Id*,*stopBtnId* ,*stopBtnId*,R.string.*stop*);

menu.setGroupVisible(1, **false**);

return **super**.onCreateOptionsMenu(menu);

}

}



Hình 2. 15: Minh hoạ option menu.

* + 1. **ContextMenu**

ContextMenu được sử dụng để hiển thị các tuỳ chọn khi người dùng nh ấn dài vào một cell nào đó trong ListView. Để tạo một ContextMenu ta cũng có 2 cách giống như tạo MenuOptions ở trên chỉ khác tên phương thức.

Khi nhấn dài vào một cell trong ListView thì phương thức:

**public** **void** onCreateContextMenu(**ContextMenu** menu, **View** v, **ContextMenuInfo** menuInfo)

sẽ được gọi và truyền vào 3 tham số là:

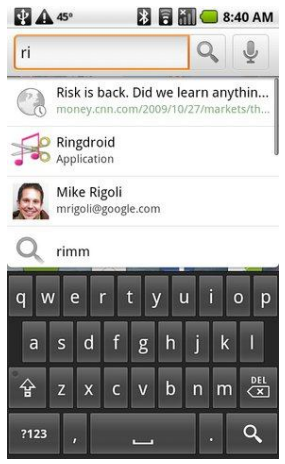
ContextMenu: đối tượng để add các context menu item

View: Đối tượng nơi mà xảy ra sự kiện

ContextMenuInfo: Cho biết vị trí xảy ra sự kiện trong ListView.

* + 1. **Quick Search Box**

Một trong những tính năng mới trong phiên bản Android 1.6 đó là Quick Search Box. Đây là khuôn khổ tìm kiếm mới trên toàn hệ thống Android, điều này làm cho người dùng có thể nhanh chóng tìm kiếm bất c ứ thứ gì có trên chiếc điện thoại Android của họ và cả các tài nguyên trên web khi họ đang online. Nó tìm kiếm và hiển thị kết quả tìm kiếm ngay khi đang gõ. Nó cũng cung cấp các kết quả từ các gợi ý tìm kiếm web, danh sách doanh nghiệp địa phương, và thông tin khác từ Google, chẳng hạn như báo giá cổ phiếu, thời tiết, và tình trạng chuyến bay. Tất cả điều này có sẵn ngay từ màn hình chủ, bằng cách khai thác trên Quick Search Box (QSB).



Hình 2. 16: Minh hoạ Quick Search Box.

* + 1. **Activity & Intent**
       1. **Activity**

Activity là một thành chính của một ứng dụng Android, được dùng để hiển thị một màn hình và nắm bắt các hoạt động xảy ra trên màn hình đó. Khi làm việc với Activity cần nắm bắt được một số kiến thức cơ bản như sau:

* Chu kỳ sống của một Activity

(Xem chu kỳ ứng dụng của Android mục 1.6)

* Tạo menu và dialog
* Khởi động một Activity

Để khởi động một Activity ta sử dụng Intent sẽ tìm hiểu kỹ hơn ở phần b. Tuy nhiên, trong phần này tôi sẽ hướng dẫn cách chuyển giữa các Intent theo 2 loại:

* Khai báo không tường minh:

Cung cấp chính xác thông tin của activity cần gọi b ằng cách truyền vào tên class của Activity đó

VD: Từ Activity A muốn chuyển qua Activity B ta khai báo một Intent trong

Activity A:

**void** onCreateContextMenu

**Intent** *intent* = **new** **Intent**(**this**, B.**class**);

startActivity(*intent*);

* Khai báo không tường minh

Cung cấp các thao tác cần làm gì với loại dữ liệu nào, hệ thông sẽ tìm đến activity tương ứng để khởi động.

VD: Để xem thông tin một contact nào đó trong Activity của ứng dụng Contact trong Android ta chỉ đến dữ liệu contact và chỉ đến Activity View contact như sau:

**Intent** *i =* **new Intent**();

*i*.setAction(**Intent***.ACTION\_VIEW*);

*i*.setData(**Uri**.withAppendedPath(

android.provider.**Contacts**.**People**.*CONTENT\_URI*, 1));

startActivity(*i*);

* Tính liên lạc giữa 2 activity

Khi chuyển sang một Activity khác ta có thể gửi kèm dữ liệu trong Intent đó như sau:

intent.putExtra(*“key1”,* *“value1”*);

intent.putExtra(*“key2”,* 23);

Bên phía Activity được khởi động hay được chuyển đế n, có thể lấy dữ liệu được gửi như sau:

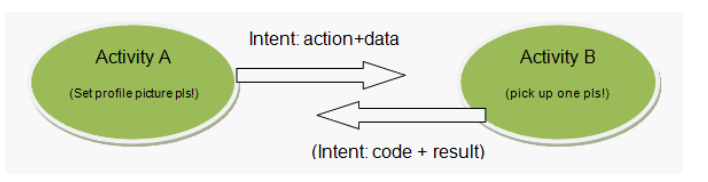
getIntent().getExtra().getString(“*key1*”);

getIntent().getExtra().getInt(“*key2*”);

* Task

Android là một hệ điều hành đa tiến trình. Khi lập trình trên nền tảng Android thì tiến trình là một vấn đề cần phải được chú ý nhiều nhất. Mặc dù Android hỗ trợ đa tiến trình nhưng trên một thiết bị di động với c ấu hình thấp mà chúng ta quá lạm dụng tiến trình thì sẽ rất tốn bộ xử lý điều này cũng đồng nghĩa với việc đang biến ứng dụng của trở thành một thứ phần mềm tiêu thụ điện năng.

* + - 1. **Intent**
* Khái niệm Intent:
  + Là một cấu trúc dữ liệu mô tả cách thức, đối tượng thực hiện của một Activity
  + Là cầu nối giữa các Activity: ứng dụng Android thường bao gồm nhiều Activity, mỗi Activity ho ạt động độc lập với nhau và thực hiện những công việc khác nhau. Intent chính là người đưa thư, giúp các Activity có thể triệu gọi cũng như truyền các dữ liệu cần thiết tới một Activity khác. Điều này cũng giống như việc di chuyển qua lại giữa các Forms trong lập trình Windows Form.



Hình 2. 17: Truyền dữ liệu giữa 2 Activity.

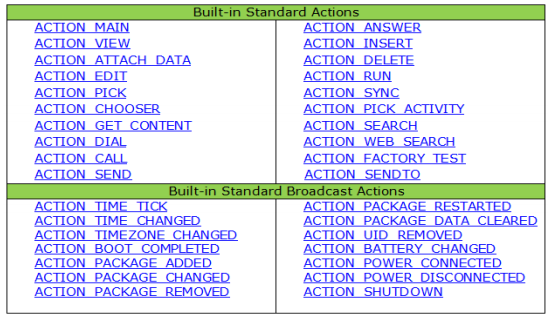
* Dữ liệu của Intent:
  + Intent về cơ bản là một cấu trúc dữ liệu, được mô tả trong lớp android.content.Intent
  + Các thuộc tính của một đối tượng Intent:



Hình 2. 18: Các thuộc tính của Intent.

* Các Action được định nghĩa sẵn:

Dưới đây là những hằng String đã được định nghĩa sẵn trong lớp Intent. Đi kèm với nó là các Activity hay Application được xây dựng sẵn sẽ được triệu gọi mỗi khi Intent tương ứng được gửi (tất nhiên khi được cung cấp đúng data).



Hình 2. 19: Các Action đã được định nghĩa sẵn trong Intent.

* 1. **Content Provider và Uri.**

Trong hệ thống Android tất cả các tài nguyên ngư Contact, SMS,… đều được lưu trữ vào CSDL SQLite của hệ thống. Cũng như các CSDL khác, CSDL mà hệ thống Android sử dụng để lưu trữ thông tin cũng cho phép chúng ta truy vấn dữ liệu như một CSDL MSSQL thông thường. Tuy nhiên, trong hệ thống đó chúng ta không cần phải thao tác bằng lệnh SQL nhiều để truy xuất dữ liệu mà thay vào đó Android đã được trang bị một API cho phép người lập trình có thể dễ dàng truy xuất dữ liệu. Đó gọi là ContentProvider. ContentProvider cung cấp cho chúng ta một đối tượng con trỏ giúp chúng ta có thể dễ dàng lấy được bất cứ dữ liệu lưu trữ nào chỉ cần cung cấp một đường dẫn đúng đến dữ liệu đó. Đường dẫn này còn được gọi là Uri.

Cấu trúc gồm có 4 phần chính như sau:

Phần A: Đây là tiền tố chỉ ra dữ liệu được điều khiển bởi Content Provider và nó không bao giờ thay đổi.

Phần B: Phần này chỉ đến nơi lưu trữ dữ liệu. Cũng giống như cấu trúc của một số điện thoại thì cái này có thể hình dung nó như là mã quốc gia hoặc cũng có thể coi nó như là tên của CSDL.

Phần C: Phần này chỉ ra loại dữ liệu. Chẳng hạn như, dữ liệu contact, dữ liệu SMS,… Phần này có thể coi nó như là tên của một table

Phần D: Phần này chỉ đến đúng vị trí của dữ liệu, có thể coi phần này như là ID của row trong table hoặc một dữ liệu nào đó dùng để truy vấn.

* 1. **Background Service.**

Service là 1 trong 4 thành phần chính trong 1 ứng dụng Android (Activity, Service, BroadcastReceiver, ContentProvider) thành phần này chạy trong hậu trường và làm những công việc không cần tới giao diện như chơi nhạc, download, xử lí tính toán…

*Một Service có th ể được sử dụng theo 2 cách:*

* Nó có thể được bắt đ ầu và được cho phép hoạt động cho đến khi một người nào đó dừng nó lại hoặc nó tự ngắt. Ở chế độ này, nó được bắt đầu bằng cách gọi Context.startService() và dừng bằng lệnh Context.stopService(). Nó có thể tự ngắt bằng lệnh Service.stopSelf() hoặc Service.stopSelfResult(). Chỉ cần một lệnh stopService() để ngừng Service lại cho dù lệnh startService() được gọi ra bao nhiêu lần.
* Service có thể được vận hành theo như đã được lập trình việc sử dụng một Interface mà nó đ ịnh nghĩa. Các người dùng thiết lập một đường truyền tới đối tượng Service và sử dụng đường kết nói đó để thâm nhập vào Service. Kết nối này được thiết lập bằng cách gọi lệnh Context.bindService() và được đóng lại bằng cách gọi lệnh Context.unbindService(). Nhiều người dùng có thể kết nối tới cùng một thiết bị. Nếu Service vẫn chưa được khởi chạy, lệnh bindService() có thể tùy ý khởi chạy nó. Hai chế độ này thì không tách biệt toàn bộ. có thể kết nối với một Service mà nó đ ã được bắt đ ầu với lệnh startService(). Ví dụ, một Service nghe nhạc ở chế độ nền có thể được bắt đ ầu bằng cách gọi lệnh startService() cùng với một đối tượng Intent mà định dạng được âm nhạc để chơi. Chỉ sau đó, có thể là khi người sử dụng muốn kiểm soát trình chơi nhạc hoặc biết thêm thông tin về bài hát hiện tại đang chơi, th ì sẽ có một Activity tạo lập một đường truyền tới Service bằng cách gọi bindService(). Trong trường hợp như thế này, stopService() sẽ không thực sự ngừng Service cho đến khi liên kết cuối cùng được đóng lại. Giống như một Activity, một Service cũng có các phương thức chu kỳ thời gian mà có thể cài đặt để kiểm soát những sự thay đổi trong trạng thái của nó. Service chỉ có 3 phương thức được gọi đến trong chu trình sống là:

void onCreate()

void onStart(Intent intent)

void onDestroy()

Bằng việc thực hiện những phương thức này, có thể giám sát 2 vòng lặp của chu kỳ thời gian của mỗi Service Entire lifetime của một Service diễn ra giữa thời gian onCreate() được gọi ra và thời gian mà onDestroy() trả lại. Giống như một Activity, một Service lại tiết hành cài đặt ban đầu ở onCreate(), và giải phóng tất cả các tài nguyên còn lại ở onDestroy() Ví dụ, một Service phát lại nhạc có thể tạo ra một luồng và bắt đầu chơi nhạc onCreate(),và sau đó luồng chơi nhạc sẽ dừng lại ở onCreate(), Active lifetime của một Service bắt đầu bằng một lệnh tới onStart(). Đâylà phương thức được chuyển giao đối tượng Intent mà đã được thông qua để tới startService() Service âm nhạc sẽ mở đối tượng Intent để quyết định xem sẽ chơi loại nhạc nào và bắt đầu phát nhạc. Không có callback tương đương nào cho thời điểm Service ngừng lại – không có phương thức onStop(). Các phương thức onCreate() và onDestroy() được gọi cho tất cả các Service dù chúng có được bắt đầu bằng Context.startService() hoặc Context.bindService() hay không. Tuy nhiên, onStart() chỉ được gọi ra đối với các Service bắt đầu bằng startService(). Nếu một Service cho phép những Service khác kết nối với nó thì sẽ có thêm các phương thức callback dành cho Service đó để thực hiện.

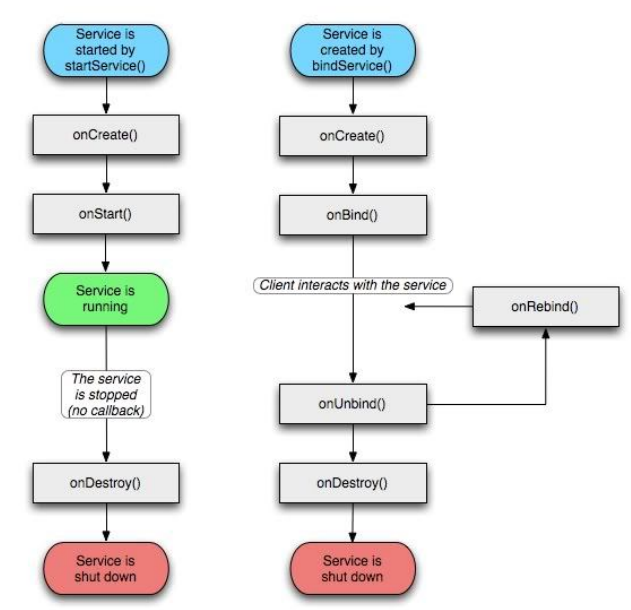
IBinder onBind(Intent intent)

boolean onUnbind(Intent intent)

void onRebind(Intent intent)

Hàm callback onBind() thông qua đối tượng Intent đã đựoc truy ền đến bindService và onUnbind() được chuyển giao đối tượng mà đã được chuyển đến. Nếu Service đang được chỉ định (binding), onBind() quay trở lại kênh thông tin mà người dùng sử dụng để tương tác với Service. Phương thức onUnbind() có thể yêu cầu onRebind() được gọi nếu một người dùng kết nối với Service.

Biểu đồ dưới đây minh họa cho các phương thức callback giành cho một Service.



Hình 2. 20: Chu trình sống của một Service.

Mặc dù, nó phân tách các Service được tạo ra thông qua startService với các Service mà được tạo ra bằng bindService(). Hãy nhớ rằng bất kì Service nào, cho dù nó được khởi tạo như thế nào thì nó vẫn có thể cho phép các người dùng kết nối tới nó một cách hiệu quả nhất, cho nên bất kì Service nào cũng có thể được chỉ định thông qua các các phương thức onBind()và onUnbind().

* 1. **Telephony.**

Telephony là một trong 4 thành phần chính của một hệ thống Android. Nó cho phép người lập trình có thể lấy các thông tin của hệ thống như thông tin SIM, thông tin thiết bị, thông tin m ạng,… Ngoài ra, chúng ta cũng có thể cài đặt các thông s ố cho thiết bị nếu các thông số đó có thể thay đổi được. Tất cả những điều đó được quản lý bởi một class TelephonyManager trong Android.

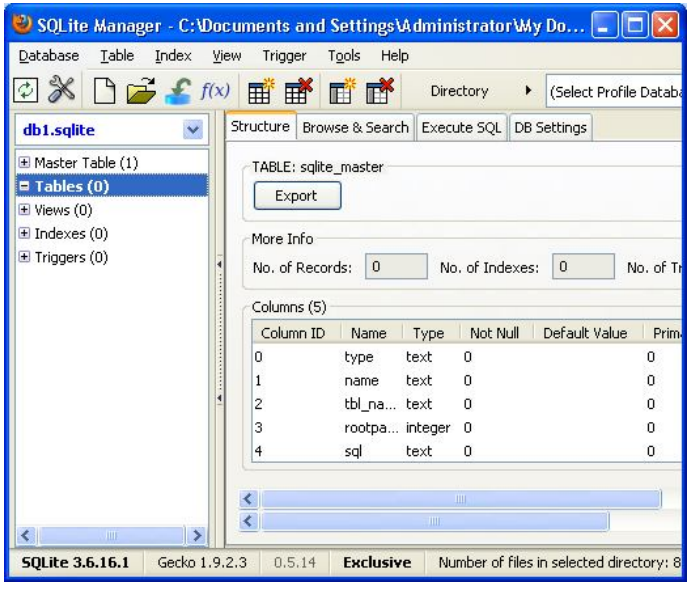
* 1. **SQLite.**

SQLite là một dạng CSDL tương tự như Mysql, PostgreSQL... Đặc điểm của SQLite là gọn, nhẹ, đơn giản. Chương trình gồm 1 file duy nhất vỏn vẹn chưa đến 500kB, không cần cài đặt, không cần 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 những 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 đặc điểm trên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm v..v.. và là sự lưa chọn phù hợp cho những người bắt đầu học database. Hiện nay thì SQLite đã được ứng dụng vào smartphone như iPhone và Android để lưu trữ dữ liệu.Để có thể dễ dàng thao tác với SQLite chúng ta có th ể sử dụng trình duyệt FireFox và tải về plugin SQLite tại link sau:

http://code.google.com/p/sqlite-manager/

Sau khi tải về file xpi, kéo file này vào cửa sổ firefox để cài đặt plugin.

Sau khi cài đặt plugin xong thì vào Menu\_tools trong firefox sẽ có chức năng SQLite Manager. Giao diện của SQLite manager trong firefox như sau:



Hình 2. 21: SQLite Manager.

* 1. **Android & WebService.**
     1. **Khái niệm Web service và SOAP**

Webservice là một d ịch vụ cung cấp cơ chế triệu gọi các đối tư ợng từ xa thông qua giao thức HTTP cùng với cơ chế truy ền tải định dạng đối tượng theo công nghệ XML. Chính vì sử dụng giao thức HTTP của Web nên giờ đây các lời gọi trở nên đơn giản và thông qua được các rào cản về tường lửa. Để đảm bảo điều này, một giao thức mới là SOAP (Simple Object Access Protocol) ra đời để hỗ trợ cho Web services. SOAP được định nghĩa dựa trên giao thức chuẩn HTTP, SOAP cho phép dữ liệu chuyển đi bằng HTTP và định dạng theo chuẩn XML. Các lời gọi hàm tham số truy ền hàm, dữ liệu trả về từ hàm, tất cả đều được chuyển sang dạng XML và có thể dễ dàng xử lý bởi tất cả các ngôn ngữ. Một thế mạnh khác đó là nếu các đối tượng phân tán xây dựng trên mô hình Web services sẽ có thể triệu gọi lẫn nhau, bất ch ấp đối tượng đó được viết trên ngôn ngữ Java của Sun hay .NET của Microsoft. Hiện tại, SOAP được coi là một sự thay đổi lớn kể từ khi COM, RMI, CORBA ra đời.

**Chương 3: XÂY DỰNG ỨNG DỤNG.**

* 1. **Mô tả chức năng.**
     1. **Chức năng đo nhịp tim**

Khi người dùng chọn chức năng đo nhịp tim điện thoại sẽ bật camera và đèn flash, người dùng sẽ đặt ngón tay vào camera, toàn bộ phần nửa màn hình dưới của điện thoại sẽ có một màu đỏ (đây là hình ngón tay được đèn flash chiếu vào). Người dùng đặt ngón tay sát vừa phải vào camera, chú ý giữa ngón tay và camera không có khoảng cách và cũng không được đặt quá chặt vào vì nếu đặt quá xa camera sẽ không nhận được hình ảnh của mao mạch trên đầu ngón tay, nếu đặt quá sát thì quá trình lưu thông máu trong mao mạch sẽ bị tắt ngẽn, phần mềm không đo được nhịp tim. Sau một vài giây đầu tiên (ít hơn 10 giây) người dùng sẽ nhận thấy hình ảnh trái tim phía trên màn hình nhấp nháy (nghĩa là phần mềm đã nhận được sự thay đổi của nhịp tim). Phần mềm sẽ mất 10 giây để tính ra nhịp tim của người dùng. Để có một nhịp tim gần chính xác nhất cần khoảng 30 giây để phân tích dữ liệu trung bình.

* + 1. **Chức năng tính chỉ số BMI.**

Khi người dùng chọn chức năng tính chỉ số BMI phần mềm sẽ hiện lên màn hình cho người dùng xem thông dữ liệu mà phần mềm đã ghi nhận được trong thời gian sử dụng cùng với các màn hình nhập thông tin chiều cao (tính bằng mét) và cân nặng (tính bằng kilogram).Sau khi nhập thông tin và chọn tính chỉ số phần mềm sẽ xuất ra chỉ số BMI cho người dùng kèm theo đánh giá về chỉ số này như béo phì hay thân hình bình thường. Sau khi phần mềm tính toán xong sẽ lưu kết quả vào cơ sở dữ liệu bao gồm thời điểm sử dụng chức năng này, chỉ số vừa tính ra và thể trạng của người dùng. Việc lưu lại này giúp người dùng theo dõi quá trình thay đổi thể trạng của mình (chỉ số càng tăng hoặc càng giảm, tốt nhất là tiến tới khoảng từ 18,5 đến 24,9) nếu không được như mong muốn người dùng có thể dựa theo sự thay đổi (thuộc phân khúc nào) để điều chỉnh chế độ cũng như nhận định về ăn uống, sinh hoạt.

* + 1. **Chức năng đo chỉ số WHR.**

Khi người dùng chọn chức năng tính chỉ số WHR phần mềm sẽ hiện lên màn hình cho người dùng xem thông dữ liệu mà phần mềm đã ghi nhận được trong thời gian sử dụng cùng với các màn hình nhập thông tin chu vi vòng eo (tính bằng centimet) và chu vi vòng mông (tính bằng centimet) và chọn tiêu chuẩn (có hai tiêu chuẩn là WHO và IDI & WPRO). Sau khi nhập thông tin và chọn tính chỉ số phần mềm sẽ xuất ra chỉ số cho người dùng kèm theo đánh giá về chỉ số này như sức khỏe tốt hoặc sức khỏe đang trong tình trạng nguy hiểm. Sau khi phần mềm tính toán xong sẽ lưu kết quả vào cơ sở dữ liệu bao gồm thời điểm sử dụng chức năng này, chỉ số vừa tính ra và thể trạng của người dùng. Việc lưu lại này giúp người dùng theo dõi quá trình thay đổi thể trạng của mình (chỉ số càng tăng hoặc càng giảm, tốt nhất là tiến tới 0.9 ở nam và 0.7 ở nữ) nếu không được như mong muốn người dùng có thể dựa theo sự thay đổi (thuộc phân khúc nào) để điều chỉnh chế độ cũng như nhận định về ăn uống, sinh hoạt.

* + 1. **Chức năng đo số bước chạy.**

Khi người dùng chọn chức năng đo số bước chạy phần mềm sẽ hiển thị màn hình cho người dùng thao tác. Khi người dùng sẽ đặt ra chỉ tiêu và chọn bắt đầu thì phần mềm sẽ bắt đầu ghi nhận số bước đi của người dùng. Khi người dùng chọn dừng lại thì phần mềm sẽ ngưng không đếm nữa. Nếu người dùng chọn lại start thì phần mềm sẽ lấy kết quả trước và cộng dồn vào. Nếu người dùng chọ start thì phần mềm sẽ lưu kết quả vào cơ sở dữ liệu bao gồm những thông tin như thời gian kết thúc lần chạy, chỉ tiêu của lần chạy này và số bước chạy thực tế đo được. Việc lưu lại dữ liệu này giúp người dùng theo dõi được quá trình tập luyện của người dùng, từ đó có thể nâng chỉ tiêu lên hoặc hạ chỉ tiêu xuống sao cho phù hợp.

* + 1. **Chức năng xem tin tức online.**

Chức năng xem tin tức online hỗ trợ người dùng xem tin tức ngay trên ứng dụng Health Care Pro. Ứng dụng tích hợp sẵn một số trang web cho phép người dùng xem tin tức về sức khỏe, đời sống cũng như những tiến bộ mới về y học.

* + 1. **Chức năng video call**

Chức năng hỗ trợ người dùng sử dụng video call ngay trên ứng dụng với một số danh sách user có sẵn, có thể đó là bác sĩ hoặc người thân của họ nhằm mục đích thông báo về tình hình sức khỏe của mình.

* + 1. **Chức năng chẩn đoán bệnh dựa theo triệu chứng**

Chức năng hỗ trợ việc chẩn đoán bệnh của người dùng thông qua triệu chứng của người dùng cung cấp. Người dùng sẽ chọn vị trí cần chẩn đoán dựa trên hình ảnh của ứng dụng, sau đó cung cấp một số triệu chứng mà mình đang cảm thấy khó chịu. Khi đó ứng dụng sẽ xử lý thông tin và xuất kết quả cho người dùng. Mặt khác gửi thông tin kết quả này về cho bác sĩ riêng của họ và bác sĩ sẽ xác nhận với họ về mức độ chính xác của kết quả chẩn đoán.

* + 1. **Chức năng tra cứu thuốc**

Tra cứu thuốc là chức năng hỗ trợ người dùng tra cứu thông tin các loại thuốc như thành phần, số đăng kí, chỉ định, chống chỉ định, cách dùng, liều dùng. Nhằm mục đích giúp người bệnh có thể biết mình đang uống thuốc gì hoặc nếu họ dùng thuốc mua ở tiệm thuốc tây mà không có sự chỉ định của bác sĩ thì cũng có thể biết được phần nào về việc họ có uống thuốc đúng bệnh hay không.

* 1. **Thiết kế ứng dụng.**
     1. **Use case Diagram**



Hình 3. 2: Sơ đồ Use case tổng thể.

* + 1. **Package Diagram**



Hình 3. 3: Sơ đồ package ứng dụng.

* + 1. **Activity Diagram**
* Đo nhịp tim.



Hình 3. 4: Activity Diagram chức năng đo nhịp tim

* Tính chỉ số BMI.



Hình 3. 5: Activity Diagram chức năng đo chỉ số BMI

* Tính chỉ số WHR.



Hình 3. 6: Activity Diagram chức năng đo chỉ số WHR

* Đo số bước chạy.



Hình 3. 7: Activity Diagram chức năng đo số bước chạy

* Lập lịch uống thuốc.



Hình 3. 8: Activity Diagram chức năng lập lịch uống thuốc

* + 1. **Sequence Diagram**
* Đo nhịp tim



Hình 3. 9: Sequence Diagram chức năng đo nhịp tim

* Tính chỉ số BMI



Hình 3. 10: Sequence Diagram chức năng tính BMI

* Tính chỉ số WHR



Hình 3. 11: Sequence Diagram chức năng tính WHR

* Lập lịch uống thuốc.



Hình 3. 12: Sequence Diagram chức năng lập lịch uống thuốc

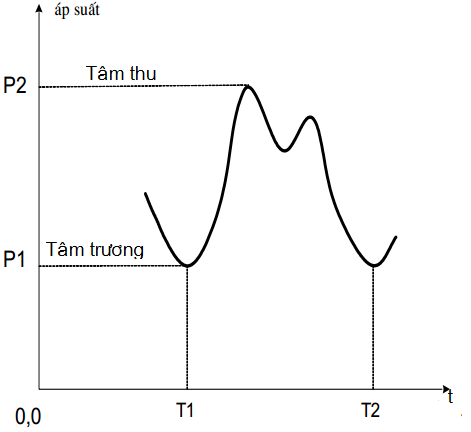
* Đo số bước chạy.



Hình 3. 13: Sequence Diagram chức năng đo số bước đi

* + 1. **Thiết kế thuật toán**
       1. **Thuật toán đo nhịp tim**

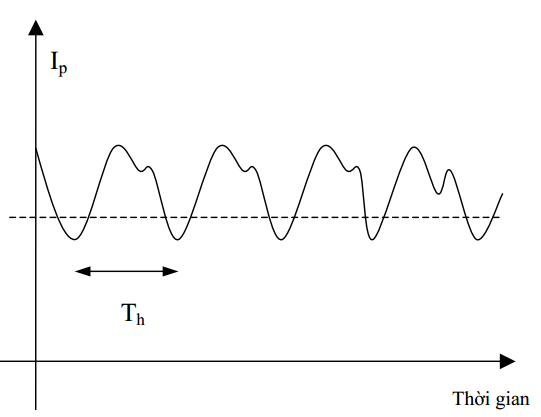
Khi tim đập, máu sẽ được dồn đi khắp cơ thể qua động mạch, tạo ra sự thay đổi về áp suất trên thành động mạch và lượng máu chảy qua động mạch. Vì vậy, ta có thể đo nhịp tim bằng cách đo những sự thay đổi đó.



Hình 3. 14: Dạng tín hiệu nhịp tim.

Khi lượng máu trong thành động mạch thay đổi sẽ làm thay đổi mức độ hấp thụ ánh sáng của động mạch, do đó khi một tia sáng được truyền qua động mạch thì cường độ ánh sáng sau khi truyền qua sẽ biến thiên đồng bộ với nhịp tim.

Khi tim giãn ra, lượng máu qua động mạch nhỏ nên hấp thụ ít ánh sáng, ánh sáng sau khi truyền qua động mạch có cường độ lớn, ngược lại khi tim co vào, lượng máu qua động mạch lớn hơn, ánh sáng sau khi truyền qua động mạch sẽ có cường độ nhỏ hơn.



Hình 3. 15: Sự hấp thụ ánh sáng của động mạch khi truyền qua ngón tay.

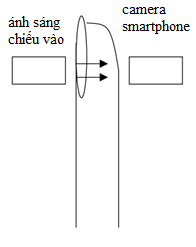
Ánh sáng sau khi truyền qua ngón tay gồm 2 thành phần AC và DC:

* Thành phần DC đặc trưng cho cường độ ánh sáng cố định truyền qua mô, xương và tĩnh mạch. Thành phần này không giúp ích trong quá trình đo đạt vì nó không bị ảnh hưởng bởi sự tuần hoàn máu của cơ thể.
* Thành phần AC đặc trưng cho cường độ ánh sáng thay đổi khi lượng máu thay đổi truyền qua động mạch, tần số của tín hiệu này đồng bộ với tần số nhịp tim. Đây là phần ánh sáng cần phân tích.

=> Nếu ta lọc bỏ thành phần DC sẽ thu được tín hiệu AC đồng bộ với tín hiệu nhịp tim.

Mô tả hoạt động thuật toán.

* Mở camera, đèn flash.
* Tiếp nhận ánh sáng xuyên qua ngón tay.
* Tính khoảng thời gian mỗi lần thay đổi ánh sáng từ đó tính ra nhịp tim của người dùng.



Hình 3. 16: Mô tả hoạt động thuật toán đo nhịp tim.

* + - 1. **Tính BMI.**
* Gọi W là khối lượng của một người (tính bằng kg) và H là chiều cao của người đó (tính bằng m), chỉ số khối cơ thể được tính theo công thức:
* Phân loại theo WHO

Theo WHO, người lớn có BMI trong phạm vi [18,50 - 24,99] là người bình thường. Dưới 18,5 là gầy, trên 25 là người béo và trên 30 là béo phì.

* Phân loại kiểu khác.
  + Người lớn hơn 20 tuổi

**Phân loại kiểu 1**

BMI < 18: người gầy

BMI = 18 - 24,9: người bình thường

BMI = 25 - 29,9: người béo phì độ I

BMI = 30 - 34,9: người béo phì độ II

BMI > 35: người béo phì độ III

**Phân loại kiểu 2**

Nam:

BMI < 20: người dưới cân

20 <= BMI < 25: người bình thường

25 <= BMI < 30: người quá cân

BMI > 30: người béo phì

Nữ:

BMI < 18: người dưới cân

18 <= BMI < 23: người bình thường

23 <= BMI < 30: người quá cân

BMI > 30: người béo phì

* + Trẻ em 2-20 tuổi

**Dựa vào thống kê theo nhóm tuổi và giới tính:**

Thiếu cân: nếu chỉ số BMI nằm trong vùng giá trị nhỏ hơn bách phân vị thứ 5 (percentile < 5th)

Sức khỏe dinh dưỡng tốt: nếu chỉ số BMI nằm trong khoảng bách phân vị thứ 5 🡪 85

Nguy cơ béo phì: nếu chỉ số BMI nằm trong khoảng bách phân vị 85 🡪 95

Béo phì: nếu chỉ số BMI nằm trong vùng lớn hơn bách phân vị 95

Dựa vào thống kê toán học, người ta có thể tính vị trí percentile của giá trị BMI tương ứng tuổi và giới tính. Nguyên nhân của béo phì hiện nay có rất nhiều tuy nhiên một trong những nguyên nhân chủ yếu là do nhận thức sai lầm về ăn uống

* + - 1. **Tính WHR.**

Thông thường WHR mang nghĩa như trên (phần mô tả chức năng), ngoài ra để bổ sung còn có một biến thể của nó đó là WHR bề mặt (frontal WHR). Sự khác nhau nằm ở chỗ nó là tỷ số của chiều dài bề ngang eo trên chiều dài bề ngang hông chứ không phải chu vi, do vậy WHR bề mặt đo được thông qua ảnh chụp còn WHR thì không đo được qua ảnh chụp.

Công thức I:

**WHR= Ce/Cm**

Công thức II:

**WHR bề mặt= De/Dm**

* + - 1. **Đo bước chạy.**

Thuật toán đo bước chạy được tính theo gia tốc trọng trường và các vector gia tốc của cảm biến trên điện thoại android:

**Ratio =**

Nếu ratio lớn hơn 2 nghĩa là gia tốc sẽ gấp 2 lần gia tốc trọng trường. Vì khi bước đi gia tốc rơi của điện thoại sẽ kèm theo bước đi của người nên sẽ lớn hơn gia tốc trọng trường. Nhóm khảo sát trên nhều thiết bị và thấy ratio bằng 2 là tương đối chính xác.

* + 1. **Màn hình ứng dụng.**
* Đo nhịp tim.



Hình 3. 17: Màn hình đo nhịp tim.



Hình 3. 18: Màn hình đo nhịp tim (kết quả)



Hình 3. 19: Màn hình đo nhịp tim (Hướng dẫn)

* Tính BMI.



Hình 3. 20: Màn hình tính BMI

* Tính WHR.



Hình 3. 21: Màn hình tính WHR

* Lập lịch uống thuốc



Hình 3. 22: Màn hình lập lịch uống thuốc

* Đo số bước chạy.



Hình 3. 23: Màn hình đo số bước chạy

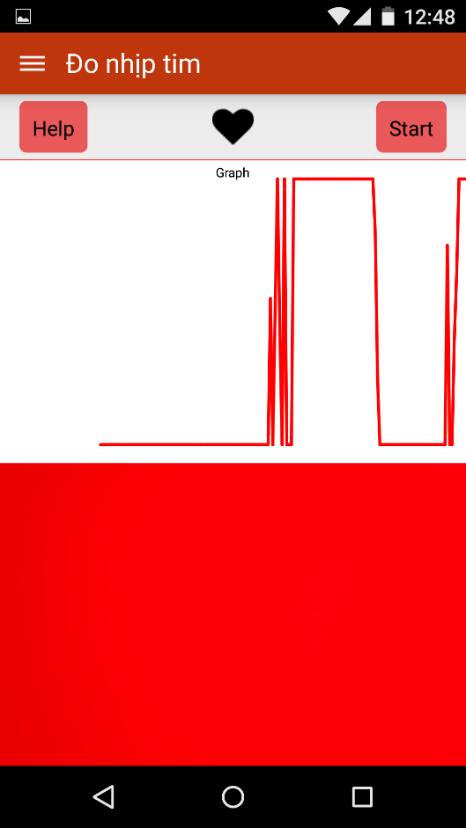
* Thông tin thêm.



Hình 3. 24: Màn hình thông tin thêm về phần mềm.

# **Chương 4: KẾT QUẢ**

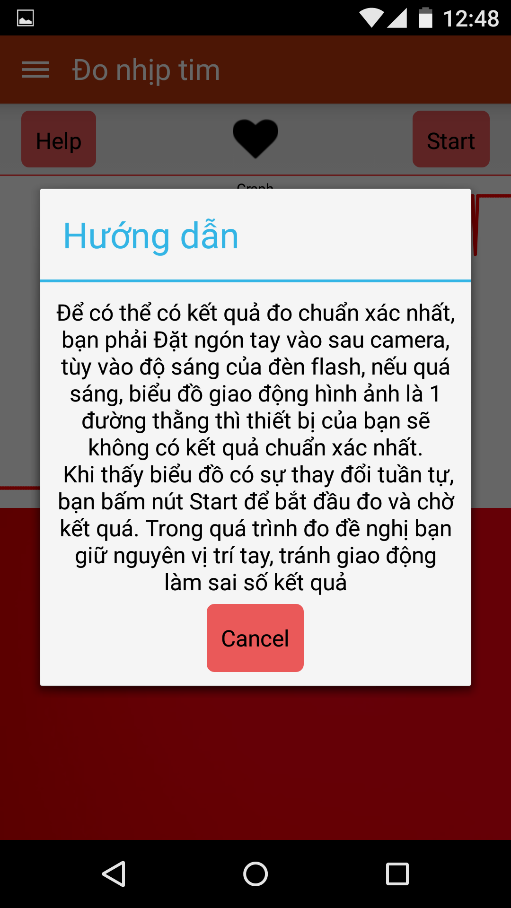
* 1. **Giao diện ứng dụng.**
     1. **Chức năng đo nhịp tim.**



Hình 4. 1: Màn hình đo nhịp tim.

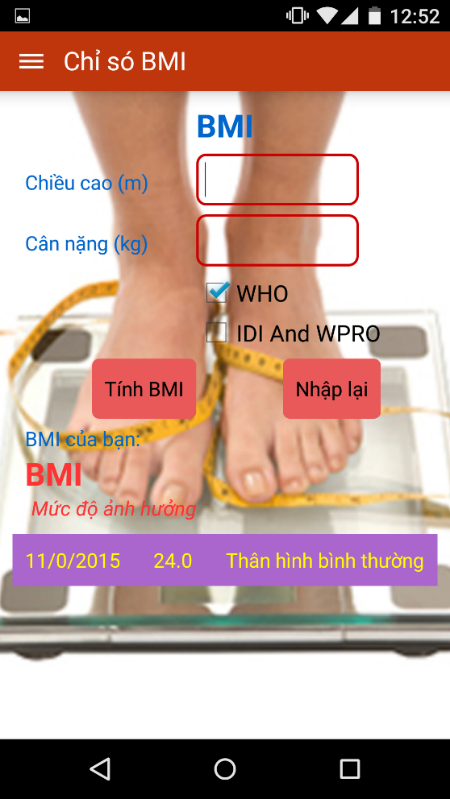


Hình 4. 2: Màn hình đo nhịp tim (kết quả).



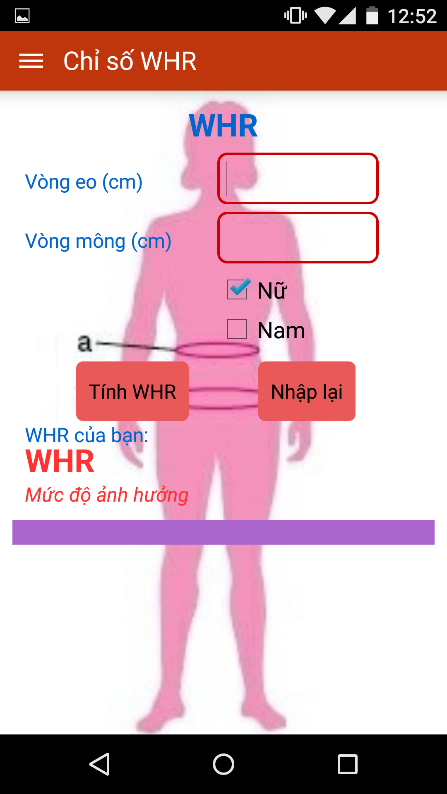
Hình 4. 3: Màn hình đo nhịp tim (Hướng dẫn).

* + 1. **Chức năng tính BMI.**



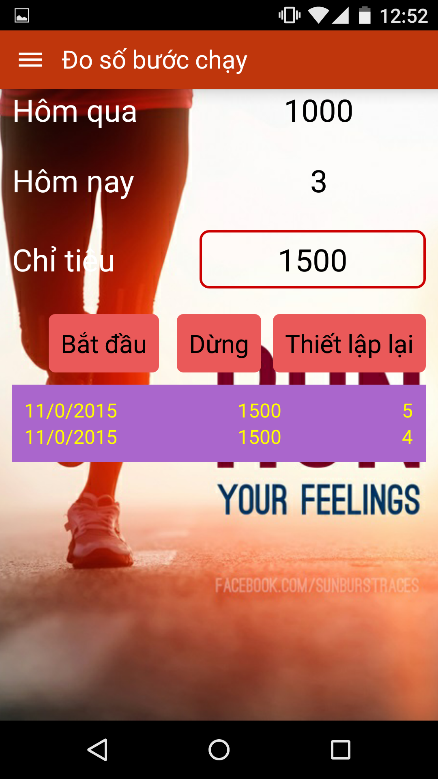
Hình 4. 4: Màn hình tính BMI.

* + 1. **Chức năng tính WHR.**



Hình 4. 5: Màn hình tính WHR.

* + 1. **Chức năng đo bước chạy.**



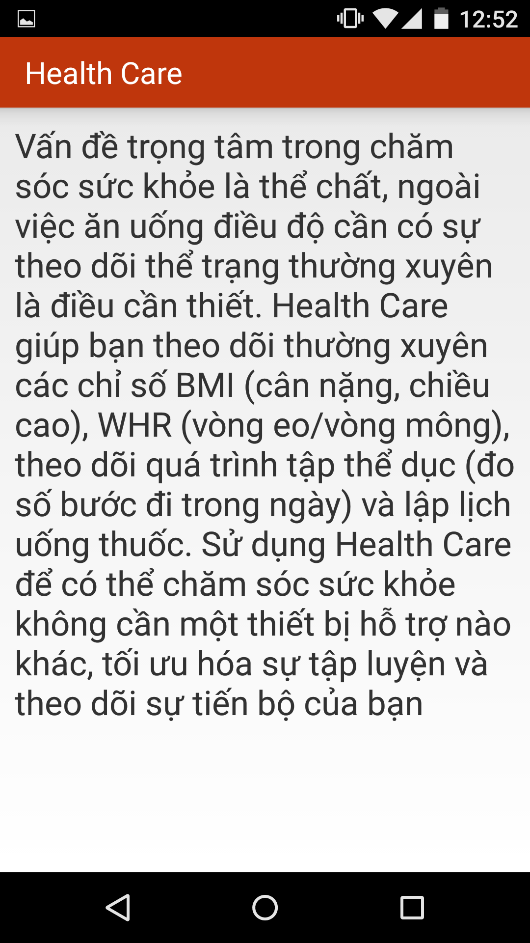
Hình 4. 6: Màn hình đo số bước chạy.

* + 1. **Chức năng lập lịch uống thuốc.**



Hình 4. 7: Màn hình lập lịch uống thuốc.

* + 1. **Màn hình thông tin phần mềm.**



Hình 4. 8: Màn hình thông tin phần mềm.

* 1. **Đánh giá độ chính xác và các tác nhân ảnh hưởng.**
* Chức năng đo nhịp tim
  + Các tác nhân ảnh hưởng:
    - Đèn flash và camera là 2 nhân tố quan trọng nhất trong chức năng này, nếu đèn flash quá sáng ứng dụng sẽ không nhận diện được các điểm ảnh thay đổi theo thời gian (đèn quá sáng nên chỉ thấy một màu đỏ duy nhất). Nếu đèn flash quá tối camera sẽ không nhận được ánh sáng thay đổi theo thời gian (chỉ nhận được màu đen). Chất lượng camera cũng ảnh hưởng đến chức năng này tuy nhiên nó không gây nhiều tác động như đèn flash. Camera tốt sẽ giúp phần mềm nhận dạng các điểm ảnh rõ ràng hơn.
    - Cách đặt ngón tay: nếu đặt ngón tay xa quá camera sẽ bị ảnh hưởng bởi ánh sáng ngoài (ứng dụng không biết đâu là ánh sáng qua ngón tay đâu là ánh sáng ngoài) nên kết quả sẽ có xu hướng tăng cao. Nếu đặt ngón tay quá chặt thì sự tuần hoàn của máu sẽ bị tắt nghẽn. Khi bị tắc ngẽn thì sẽ không có (hoặc rất ít) ánh sáng thay đổi nên kết quả có xu hướng giảm đi.
    - Thời tiết: nếu thời tiết quá lạnh ngón tay sẽ teo lại, sự tuần hoàn máu ít nhiều cũng bị ảnh hưởng nên thường kết quả sẽ giảm đi.
    - Người dùng di chuyển khi đang sử dụng chức năng: khi di chuyển thì ánh sáng môi trường cũng sẽ ảnh hưởng đến sự nhận diện của camera, ngoài ra khi người dùng di chuyển nhịp tim cũng có xu hướng tăng lên.
* Chức năng tính chỉ số WHR và BMI: độ chính xác 100%. Người dùng có thể tham khảo bảng sau:

|  |  |  |
| --- | --- | --- |
| **Phân loại** | **WHO BMI (kg/m2)** | **IDI & WPRO BMI (kg/m2)** |
| Cân nặng thấp (gầy) | <18.5 | <18.5 |
| Bình thường | 18.5 - 24.9 | 18.5 - 22.9 |
| Thừa cân | 25 | 23 |
| Tiền béo phì | 25 - 29.9 | 23 - 24.9 |
| Béo phì độ I | 30 - 34.9 | 25 - 29.9 |
| Béo phì độ II | 35 - 39.9 | 30 |
| Béo phì độ III | 40 | 40 |

Bảng 4. 1: Các chỉ số BMI.

|  |  |  |
| --- | --- | --- |
| **Nam** | **Nữ** | **Mức nguy hiểm đến sức khỏe** |
| 0,9 | 0,7 | Không nguy hiểm (sức khỏe tốt) |
| 0,9 – 0,95 | 0,7 – 0,8 | Ít |
| 0,96 - 1 | 0,81 – 0,85 | Trung bình |
| Trên 1 | Trên 0,85 | Cao (Rất nguy hiểm) |

Bảng 4. 2: Chỉ số WHR.

* Chức năng đo số bước chạy:
  + Tác nhân ảnh hưởng:
    - Phần cứng của máy: nếu cảm biến quá nhạy hoặc không nhạy thì sẽ phát sinh sai số. Ứng dụng chỉ tính trung bình trên một số máy nên việc sai số là hiển nhiên.
    - Người dùng cho người khác mượn điện thoại: ứng dụng không thể nhận biết ai đang sử dụng điện thoại nên khi người khác dùng ứng dụng cũng sẽ tính số bước của người đó.
  + độ chính xác khoảng 90%.
* Chức năng lập lịch uống thuốc.
  + Tác nhân ảnh hưởng:
    - Điện thoại hết pin hoặc tắt máy: khi đó bộ đếm thời gian sẽ ngưng hoạt động nên không thể báo động.
  + Nếu người dùng không thay đổi thời gian hoặc không tắt nguồn trong quá trình sử dụng thì tính năng này chính xác 100%
  1. **Hạn chế**
* Chưa hoàn thành tính năng tính thời gian ngủ

Lý do: chưa thiết kế được thuật toán tự động phát hiện trạng thái ngủ của người dùng

* Giao diện tuy đã chỉnh sửa nhiều so với các control mặt định của google nhưng còn chưa đẹp mắt.
* Chưa thực hiện được tính năng đo nhiệt độ

Lý do: cảm biến đo k chính xác, chỉ để dùng cho đo nhiệt độ CPU và PIN.

* 1. **Hướng phát triển**
* Phát triển tính năng share tình trạng sức khỏe lên các mạng xã hội
* Thực hiện tính năng tính thời gian ngủ
* Phát triển giao diện đẹp hơn.
  1. **Kết luận**
* Sau khi thực hiện đồ án này nhóm em đã nắm rõ hơn về quy trình phát triển một ứng dụng cũng như cấu trúc các dạng ứng dụng trên android.hiểu thêm vê kiến thức y khoa và áp dụng nó vào các thiết bị công nghệ giúp chiếc điện thoại đã tiện dụng ngày càng tiện dụng hơn.