LẬP TRÌNH DI ĐỘNG

Android và môi trường phát triển

Nội dung

- Tổng quan về hệ điều hành Android
- Kiến trúc Android
- Môi trường phát triển ứng dụng
- Tạo ứng dụng đầu tiên
- Các thành phần trong ứng dụng Android
- Vòng đời ứng dụng Android

Tổng quan về hệ điều hành Android

- Lịch sử ra đời
  - Năm 2003, Android Inc. được thành lập bởi Andy Rubin, Rich Miner, Nick Sears và Chris White tại California.
  - Năm 2005, Google sở hữu Android cùng với các vị trí quản lý.
  - Năm 2007, Liên minh thiết bị cầm tay mở (Open Handset Alliance) được thành lập. Công bố nền tảng phát triển Android.

Tổng quan về hệ điều hành Android

- Lịch sử ra đời
  - Năm 2008, thiết bị HTC Dream là phiên bản thế hệ đầu tiên hoạt động với hệ điều hành Android 1.0
  - 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

Tổng quan về hệ điều hành Android

- Các phiên bản hệ điều hành

Tổng quan về hệ điều hành Android

- Các phiên bản hệ điều hành
  - Phiên bản 1.x:
    - Android 1.0 (API 1)
    - Android 1.1 (API 2)
    - Android 1.5 Cupcake (API 3)
    - Android 1.6 Donut (API 4)
  - Phiên bản 2.x:
    - Android 2.0 Eclair (API 5) – Android 2.0.1 (API 6) – Android 2.1 (API 7)
    - Android 2.2 – 2.2.3 Froyo (API 8)
    - Android 2.3 – 2.3.2 Gingerbread (API 9)
    - Android 2.3.3 – 2.3.7 Gingerbread (API 10)

Tổng quan về hệ điều hành Android

- Các phiên bản hệ điều hành
  - Phiên bản 3.x:
    - Android 3.0 Honeycomb (API 11)
    - Android 3.1 Honeycomb (API 12)
    - Android 3.2 Honeycomb (API 13)
  - Phiên bản 4.x:
    - Android 4.0 – 4.0.2 Ice Cream Sandwich (API 14)
    - Android 4.0.3 – 4.0.4 Ice Cream Sandwich (API 15)
    - Android 4.1 Jelly Bean (API 16)
    - Android 4.2 Jelly Bean (API 17)
    - Android 4.3 Jelly Bean (API 18)
    - Android 4.4 Kit Kat (API 19)

Tổng quan về hệ điều hành Android

- Các phiên bản hệ điều hành
  - Phiên bản 5.x:
    - Android 5.0 Lollipop (API 21)
    - Android 5.1 Lollipop (API 22)
  - Phiên bản 6.0
    - Android 6.0 Marshmallow (API 23)
  - Phiên bản 7.x
    - Android 7.0 Nougat (API 24)
    - Android 7.1.1 Nougat (API 25)
  - Phiên bản 8.
    - Android 8.0 Oreo (API 26)

Tổng quan về hệ điều hành Android

- Hệ sinh thái

Mobile app

- Hiện nay có 3 hướng chính để phát triển một ứng dụng di động, đó là:
  - **Web App**
  - **Native App**
  - **Hybrid App**

Hệ điều hành Android

- Hệ điều hành tối ưu cho di động, dựa trên nhân Linux, dòng vi xử lý ARM
- Có thể được tùy biến cho thiết bị di động và những hệ thống nhúng
- Android được phát triển và hỗ trợ bởi liên minh OHA (Open Handset Allien) gồm nhiều công ty phần cứng, phần mềm và dịch vụ: Google, HTC, LG, Samsung, Motorola, Sprint, T-Mobile, NVIDIA, Intel, Broadcom, Qualcom,…
- Có 2 phiên bản song song: Android & Google API

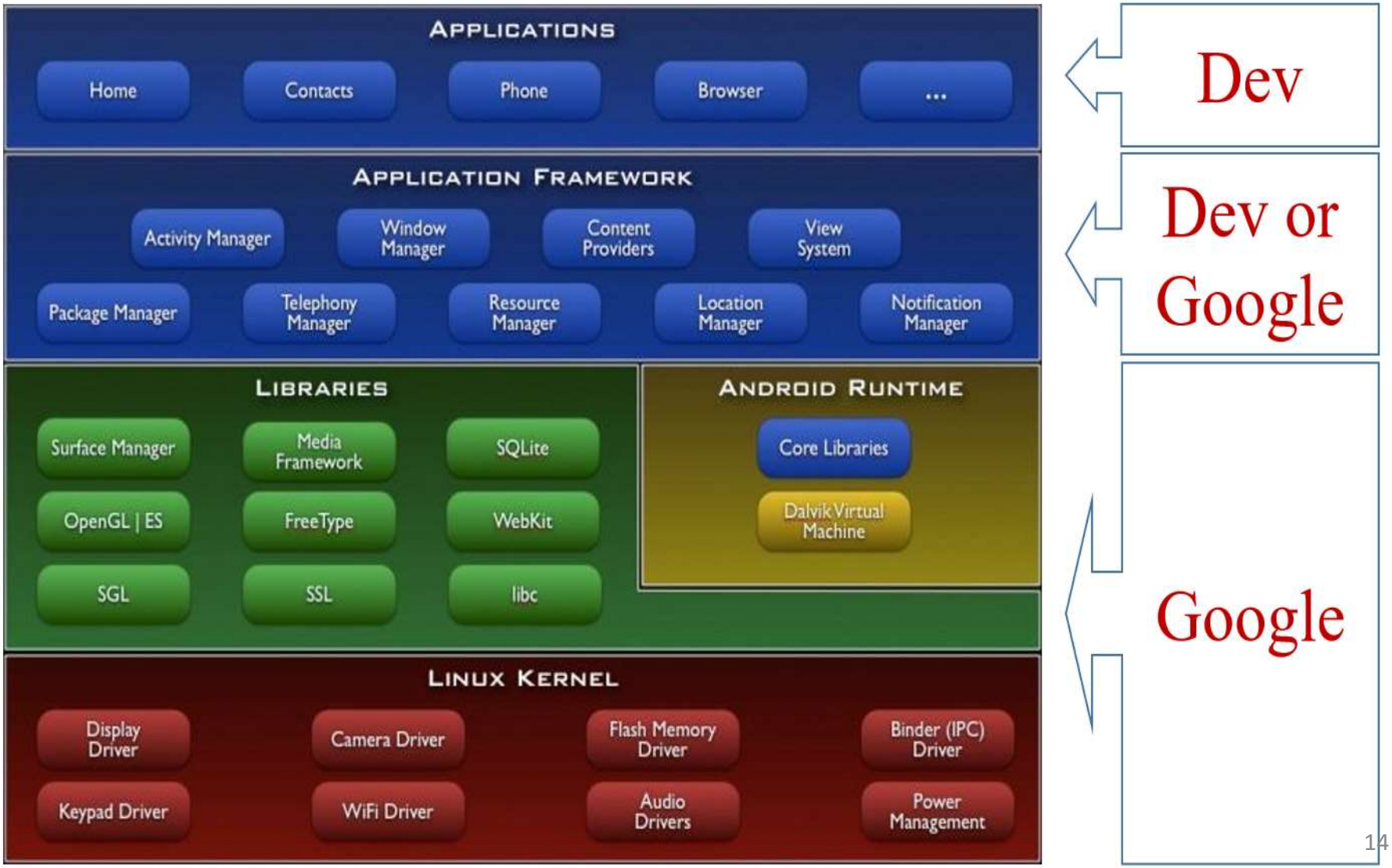
Android: đặc điểm nổi bật

- Đa luồng (multithread)
- Web ready (html5, css3, javascript, flash)
- Open GL
- Java
- Đa chạm (multitouch)
- Media (full HD video, mpeg4, H.264, mp3,…)
- Network ready (wifi, 3g, bluetooth,…)
- GPS
- Sensors

Kiến trúc Android

- Android được hình thành dựa trên nền tảng Linux nhân 2.6, từ phiên bản 4.0 sử dụng Linux nhân 3.x.
- Có 4 tầng trong HĐH Android gồm:
  - Application Framework
  - Android Runtime
  - Native Libraries
  - Linux Kernel
- Tầng cao hơn sử dụng API của các tầng bên dưới

Kiến trúc Android



Kiến trúc Android

- Linux Kernel
  - Mọi xử lý của hệ thống đều phải thông qua tầng này
  - Cung cấp các trình điều khiển thiết bị phần cứng
    - Camera
    - USB
    - Wifi / Bluetooth
    - Display
    - Power Management
    - …
  - Quản lý CPU và điều phối hoạt động các tiến trình
  - Quản lý bộ nhớ ở mức vật lý

Kiến trúc Android

- **Native Libraries**: thư viện các hàm lập trình
  - System C library: có nguồn gốc từ hệ thống thư viện chuẩn C (libc), điều chỉnh các thiết bị nhúng trên Linux
  - Media Libraries: thư viện hỗ trợ playback và recording của nhiều định dạng video, audio và image phổ biến
  - Thư viện hỗ trợ phát các tập tin đa truyền thông.
  - Bộ quản lý hiển thị
  - Thư viện hỗ trợ đồ họa OpenGL 2D và 3D
  - SQLite: hỗ trợ lưu trữ cơ sở dữ liệu
  - SSL: thư viện hỗ trợ mã hóa kết nối mạng
  - Webkit: bộ diễn dịch HTML, CSS & Javascript.

Kiến trúc Android

- **Android RunTime** – gồm máy ảo Dalvik và thư viện lõi. Android RunTime ngoài tăng tốc độ cho ứng dụng còn làm nền cho tầng Application Framework kết nối đến.
  - Core Libraries – Các thư viện lõi Android sẽ cung cấp hầu hết các chức năng chính có thể có trong thư viện Java cũng như thư viện riêng biệt của Android.
  - Máy ảo Dalvik – giúp thực thi các ứng dụng android, mỗi ứng dụng chạy trên một tiến trình riêng của Dalvik VM

Kiến trúc Android

- **Application Framework** – cung cấp các lớp cho việc xây dựng ứng dụng. Cho phép truy nhập phần cứng, quản lý giao diện người dùng và tài nguyên của ứng dụng.
- Android Framework gồm các dịch vụ chính sau:
  - Activitty Manager – Quản lý các Activity.
  - Content Providers – Cho phép các ứng dụng chia sẻ dữ liệu với các ứng dụng khác.
  - Resource Manager – Quản lý tài nguyên
  - Notifications Manager – Cho phép ứng dụng hiển thị cảnh báo và thông báo cho người dùng.
  - View System – Tập các thành phần giao diện (view) được sử dụng để tạo giao diện người dùng.

Kiến trúc Android

- **Applications** – gồm các ứng dụng được tích hợp sẵn và các ứng dụng của hãng thứ ba. Tầng ứng dụng trong Android RunTime sử dụng các lớp từ tầng Application Framework để thực thi ứng dụng.

Kiến trúc Android

- Có thể sử dụng các ngôn ngữ lập trình:
  - Java
  - C/C++
  - JNI (Java Native Interface)
  - XML
  - Render Script

Môi trường phát triển ứng dụng

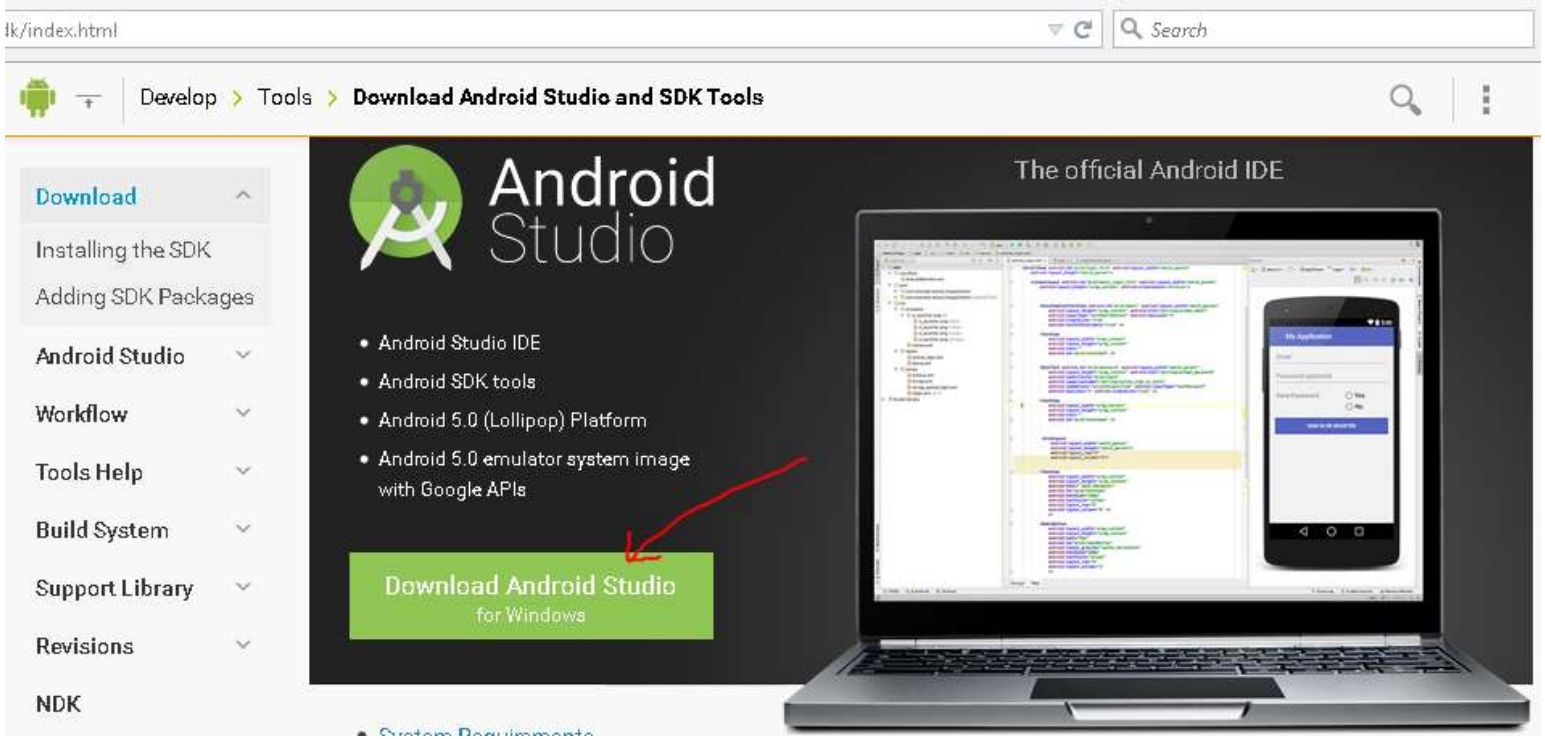
- Android có thể phát triển trên hầu hết các hệ điều hành phổ biến hiện nay:
  - Windows 32 bit: từ Windows XP trở lên
  - Windows 64 bit: từ Windows Vista trở lên
  - Mac OS X 10.4.8 or later (x86 only)
  - Ubuntu
- Môi trường phát triển:
  - JDK (Java Development Kit) từ 1.6
  - Android Studio
  - Máy ảo giả lập Android (Genymotion)

Môi trường phát triển ứng dụng

- Android Studio là công cụ lập trình dựa trên nền IntelliJ, cung cấp các tính năng mạnh mẽ hơn ADT, bao gồm:
  - Hỗ trợ xây dựng dự án dạng Gradle.
  - Hỗ trợ sửa lỗi và tái sử dụng cấu trúc phương thức
  - Cung cấp công cụ kiểm tra tính khả dụng, khả năng hoạt động của ứng dụng, tương thích nền tảng…
  - Hỗ trợ bảo mật mã nguồn và đóng gói ứng dụng.
  - Trình biên tập giao diện cung cấp tổng quan giao diện ứng dụng và các thành phần, cho phép tuỳ chỉnh trên nhiều cấu hình khác nhau.
  - Cho phép tương tác với Google Cloud.

Môi trường phát triển ứng dụng

- Android Studio bao gồm các thành phần:
  - Android Studio IDE
  - Android SDK tools
- Link: http://developer.android.com/sdk/index.html

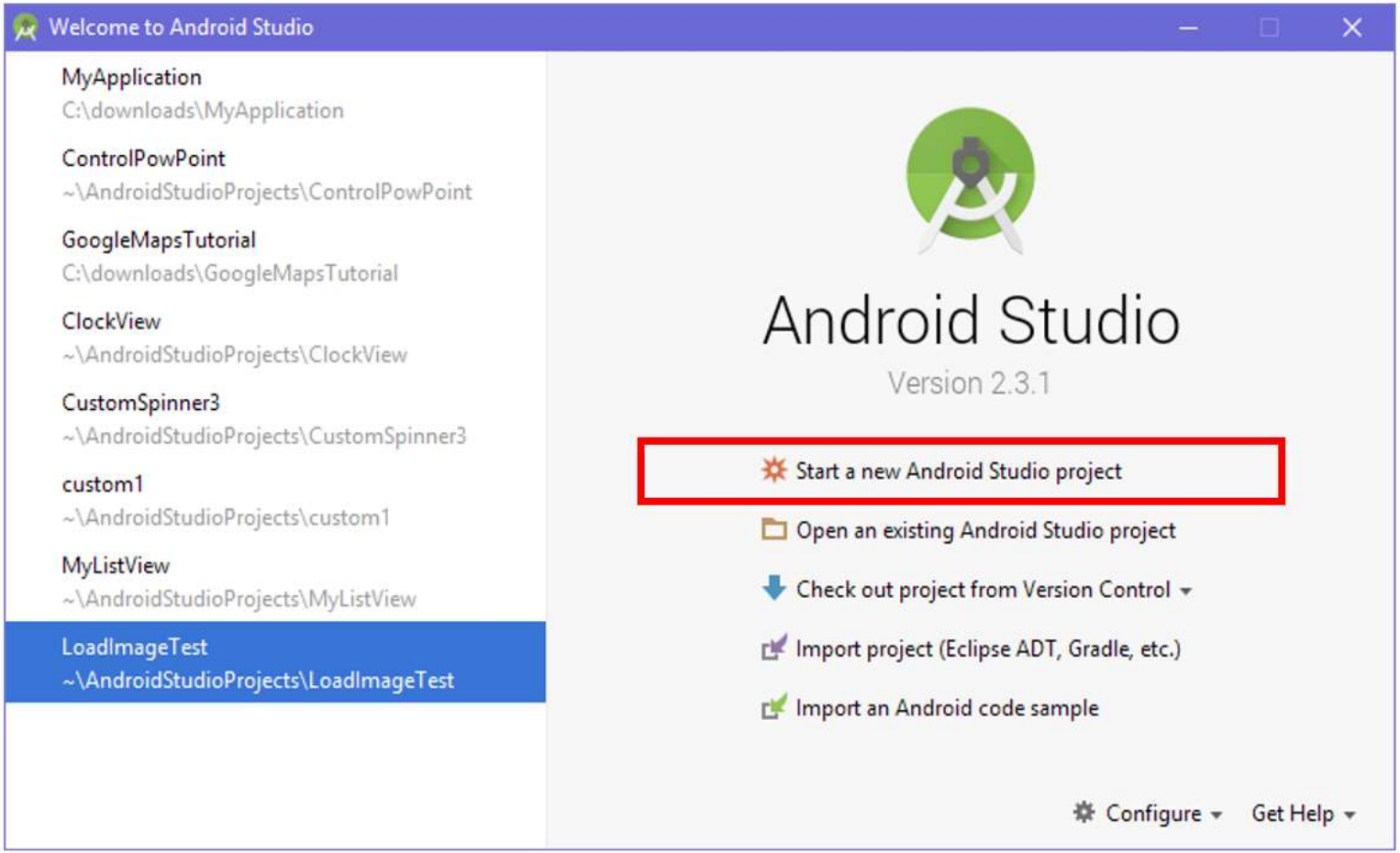


Làm quen với Android Studio

- Tạo Project trong Android Studio
- Cấu trúc một Project trong Android Studio
- Các chức năng thường dùng trong Android Studio
- Cách tạo và sử dụng Android Emulator trong Android Studio
- Quy trình thực thi một phần mềm lên thiết bị (thật, emulator) trong Android Studio.

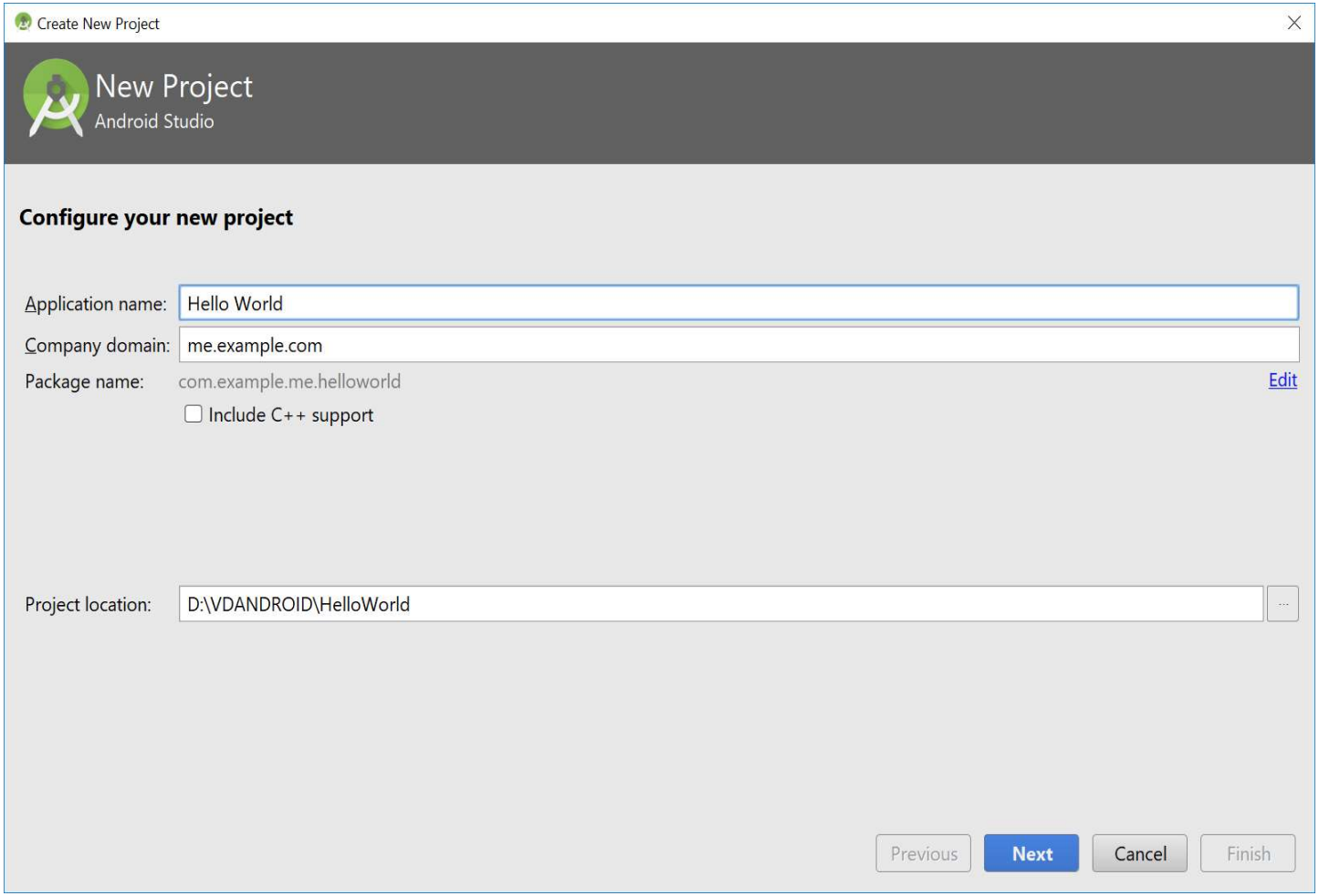
Chương trình đầu tiên

- Khởi động Android Studio



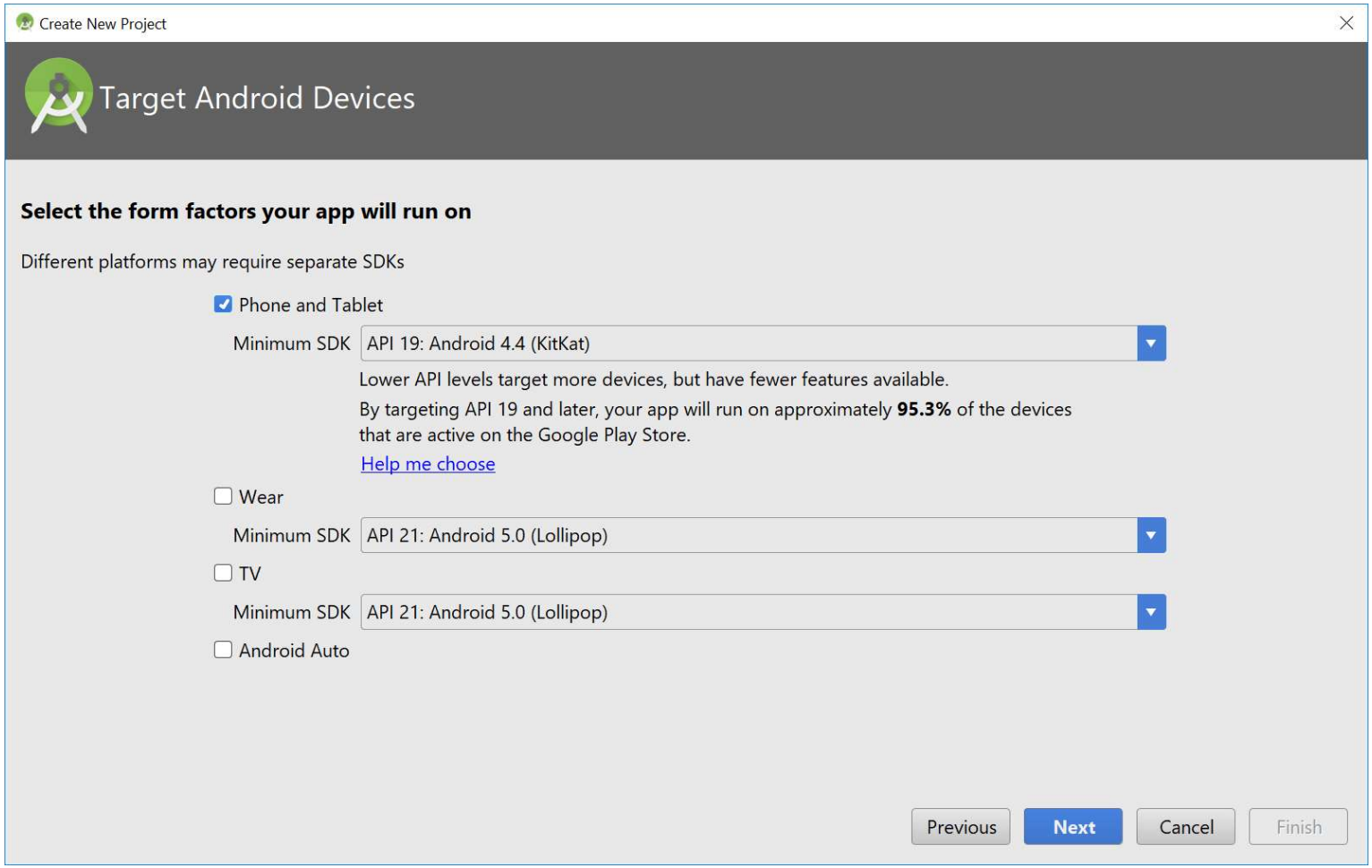
Chương trình đầu tiên

- Đặt tên ứng dụng, tên package,…



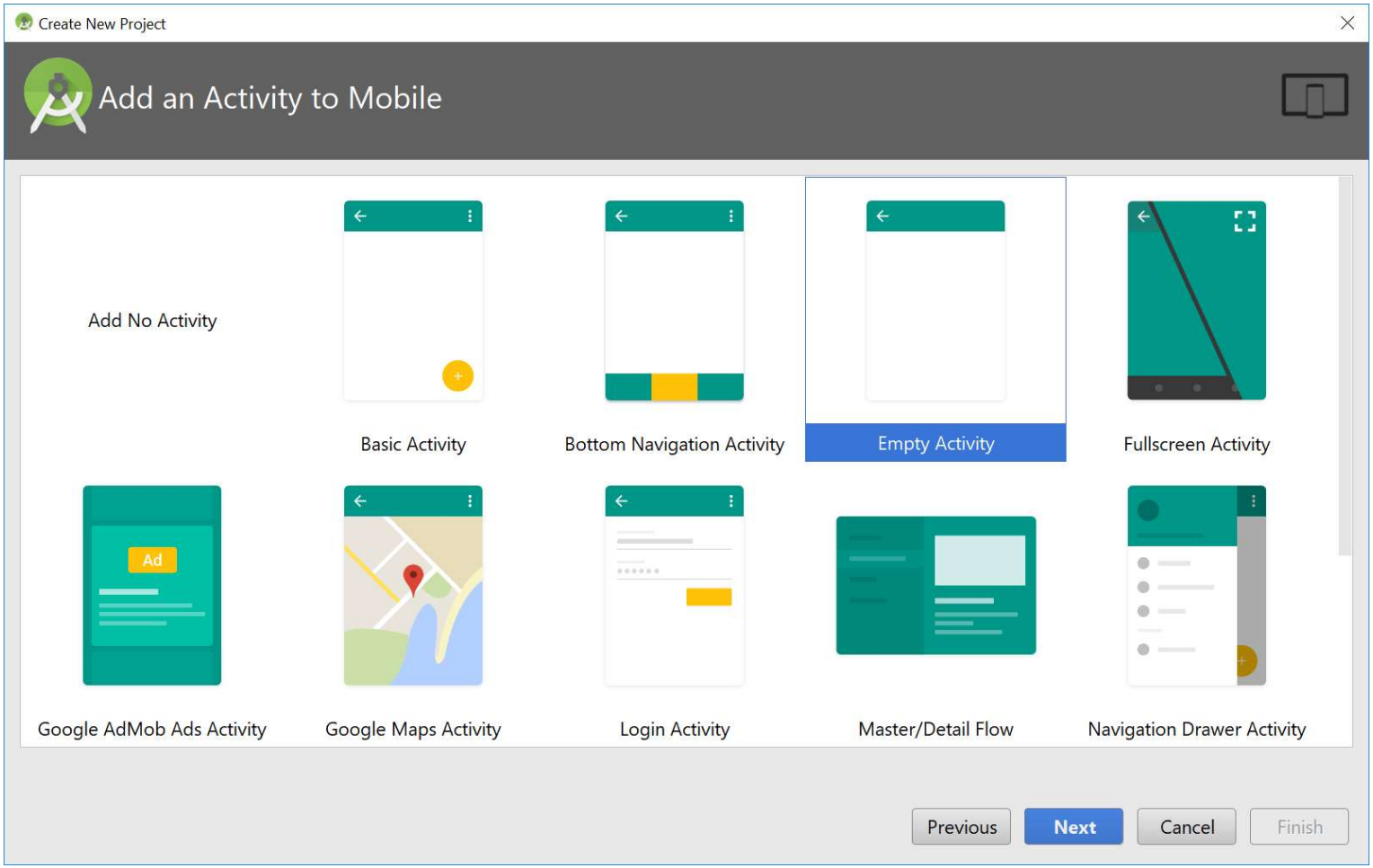
Chương trình đầu tiên

- Chọn phiên bản hệ điều hành



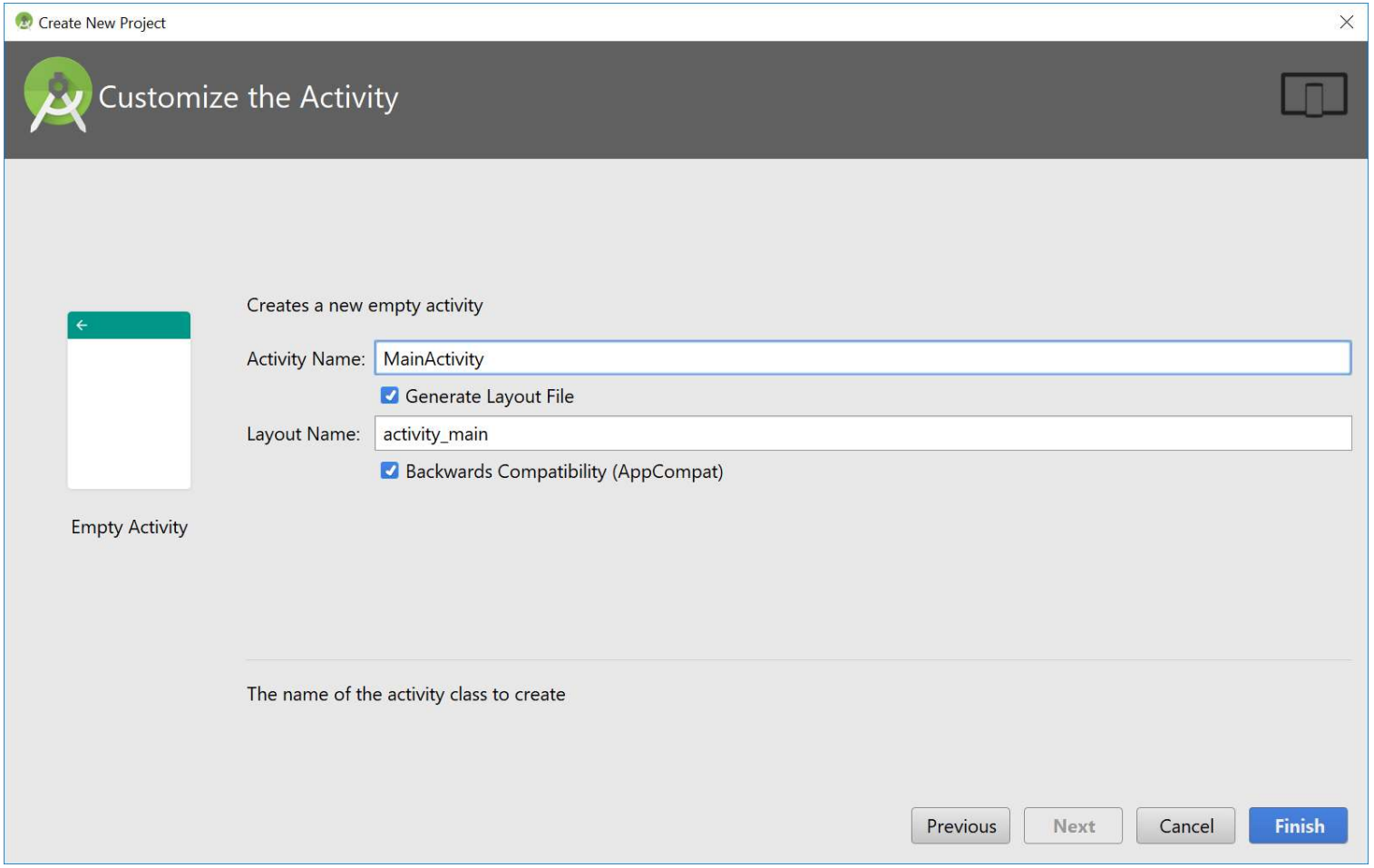
Tạo ứng dụng đầu tiên

- Chọn giao diện ban đầu

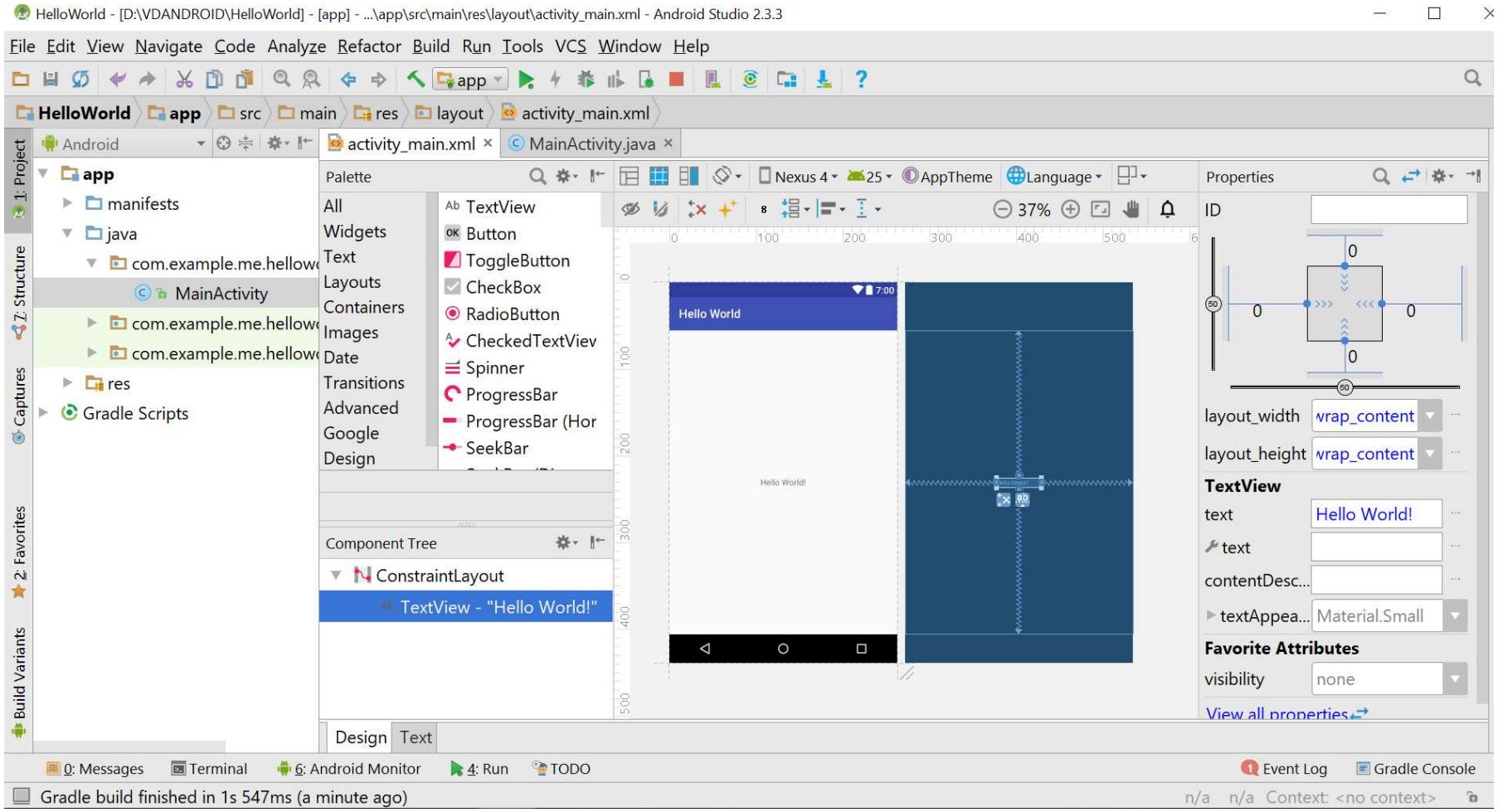


Chương trình đầu tiên

- Đặt tên cho giao diện

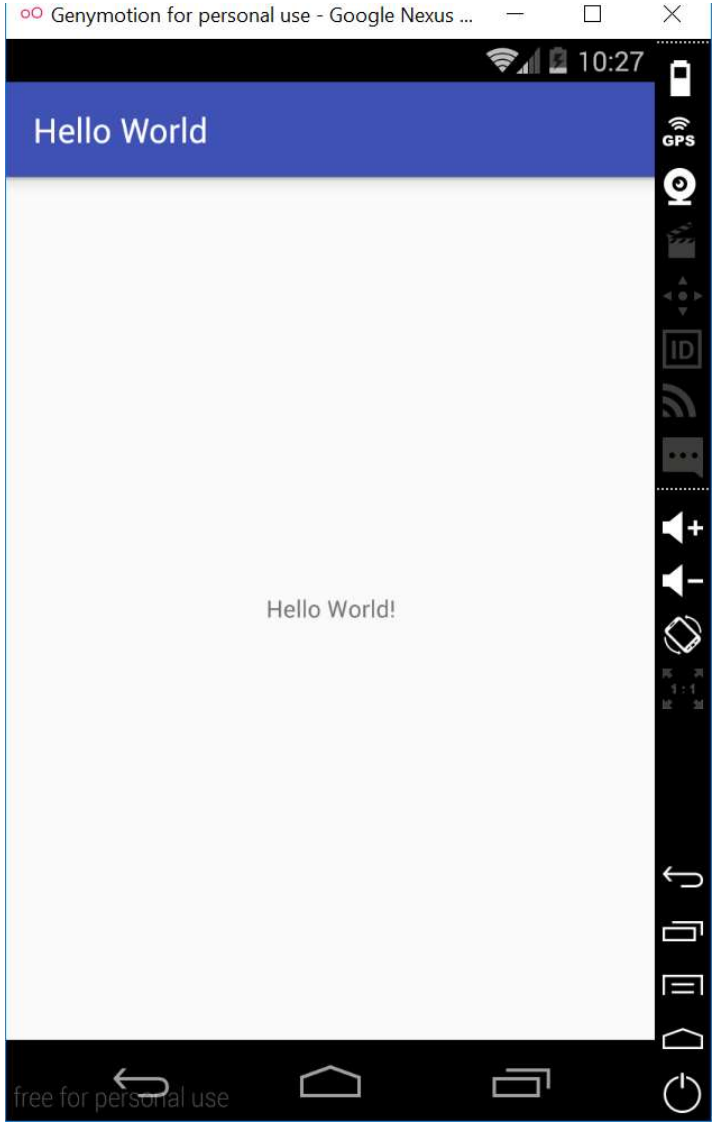


Cấu trúc một Project trong Android Studio



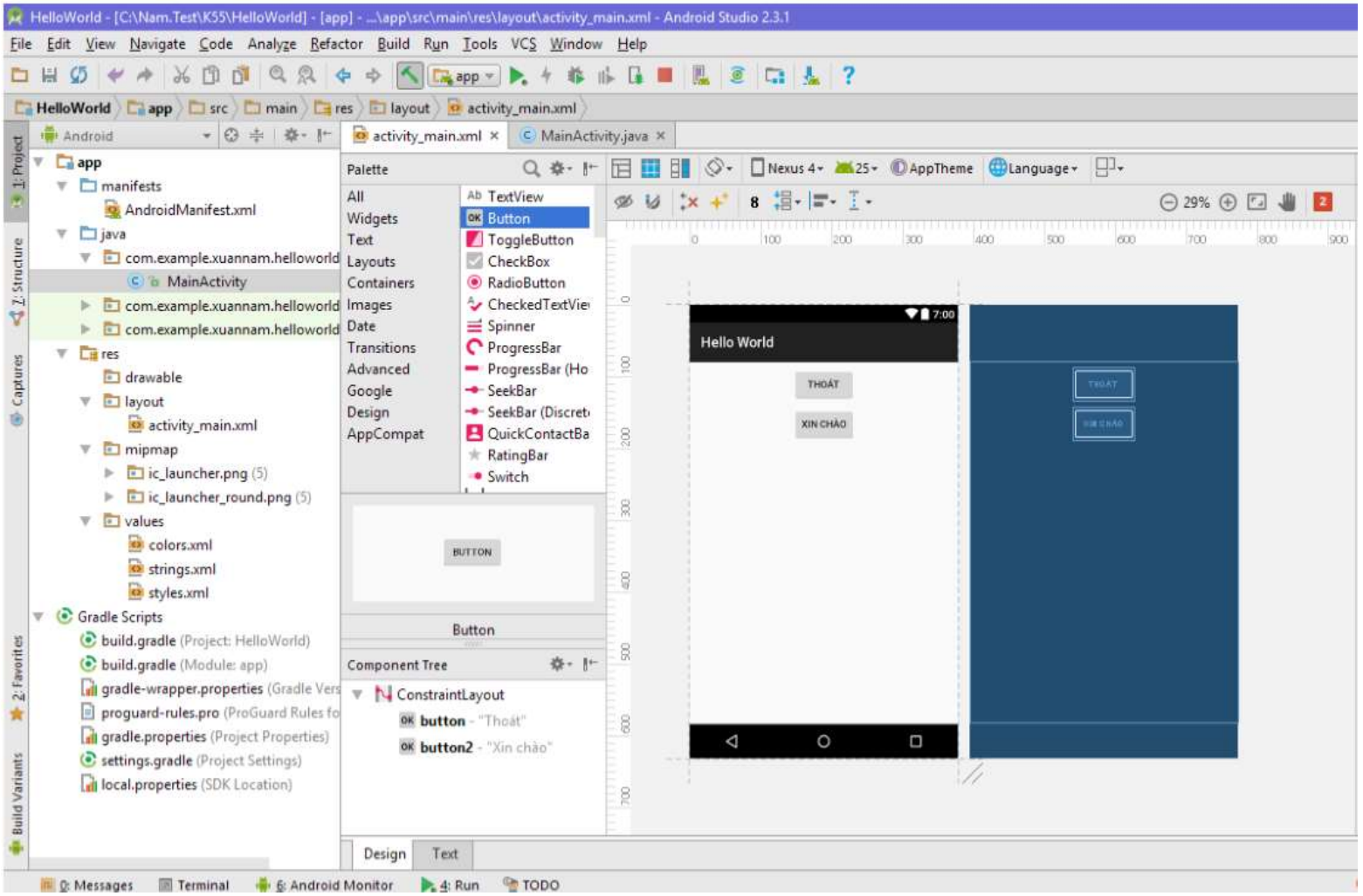
Chương trình đầu tiên

- Chạy chương trình



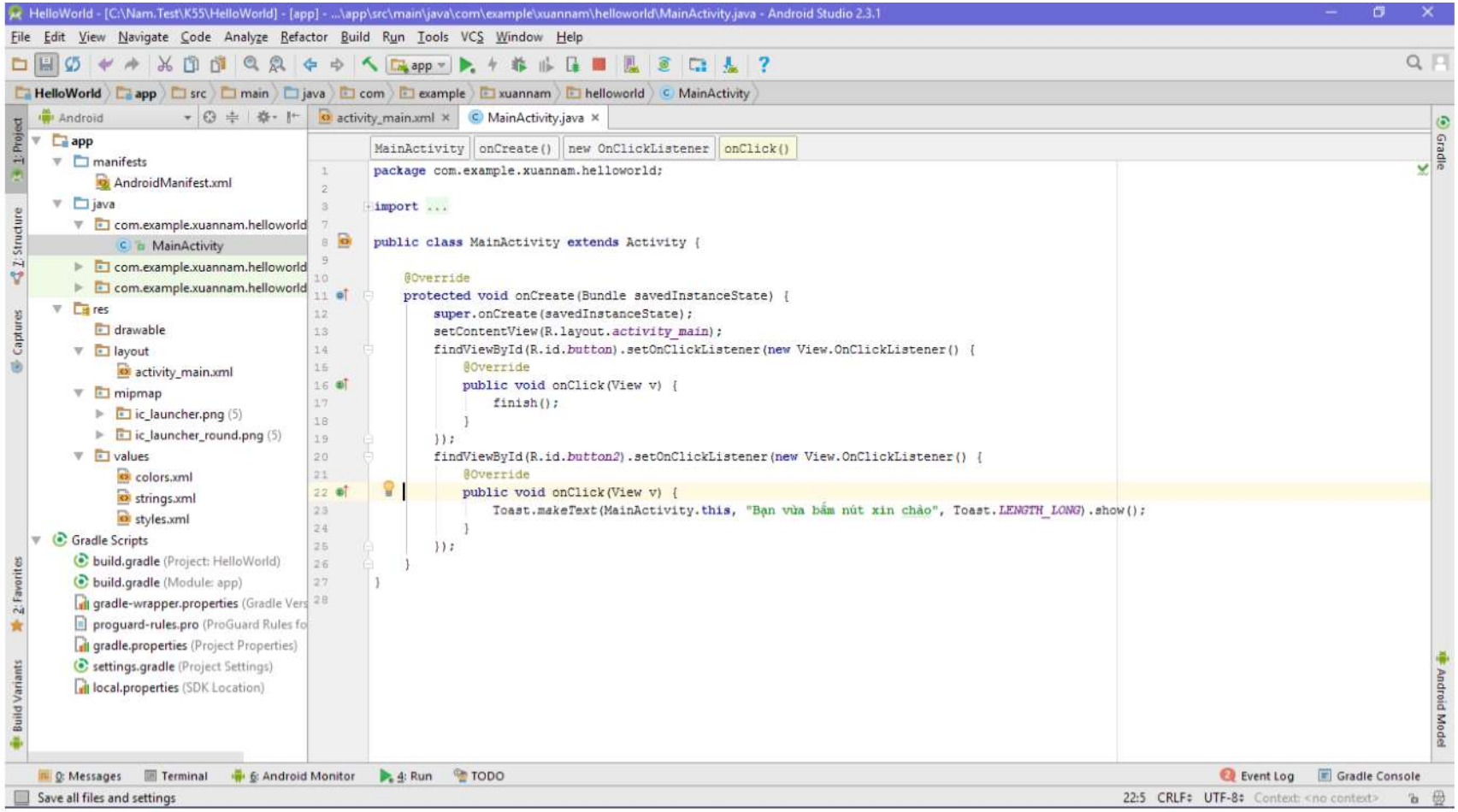
Chương trình đầu tiên

- Thêm 2 nút bấm vào giao diện



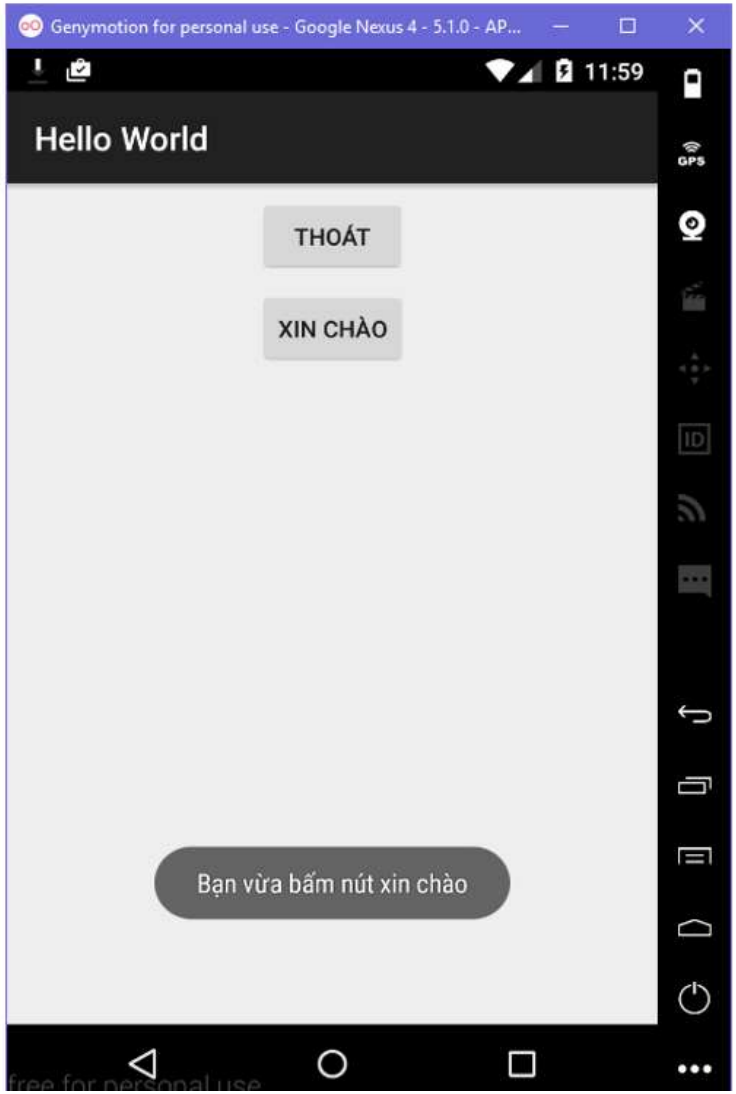
Chương trình đầu tiên

- Viết mã xử lý sự kiện

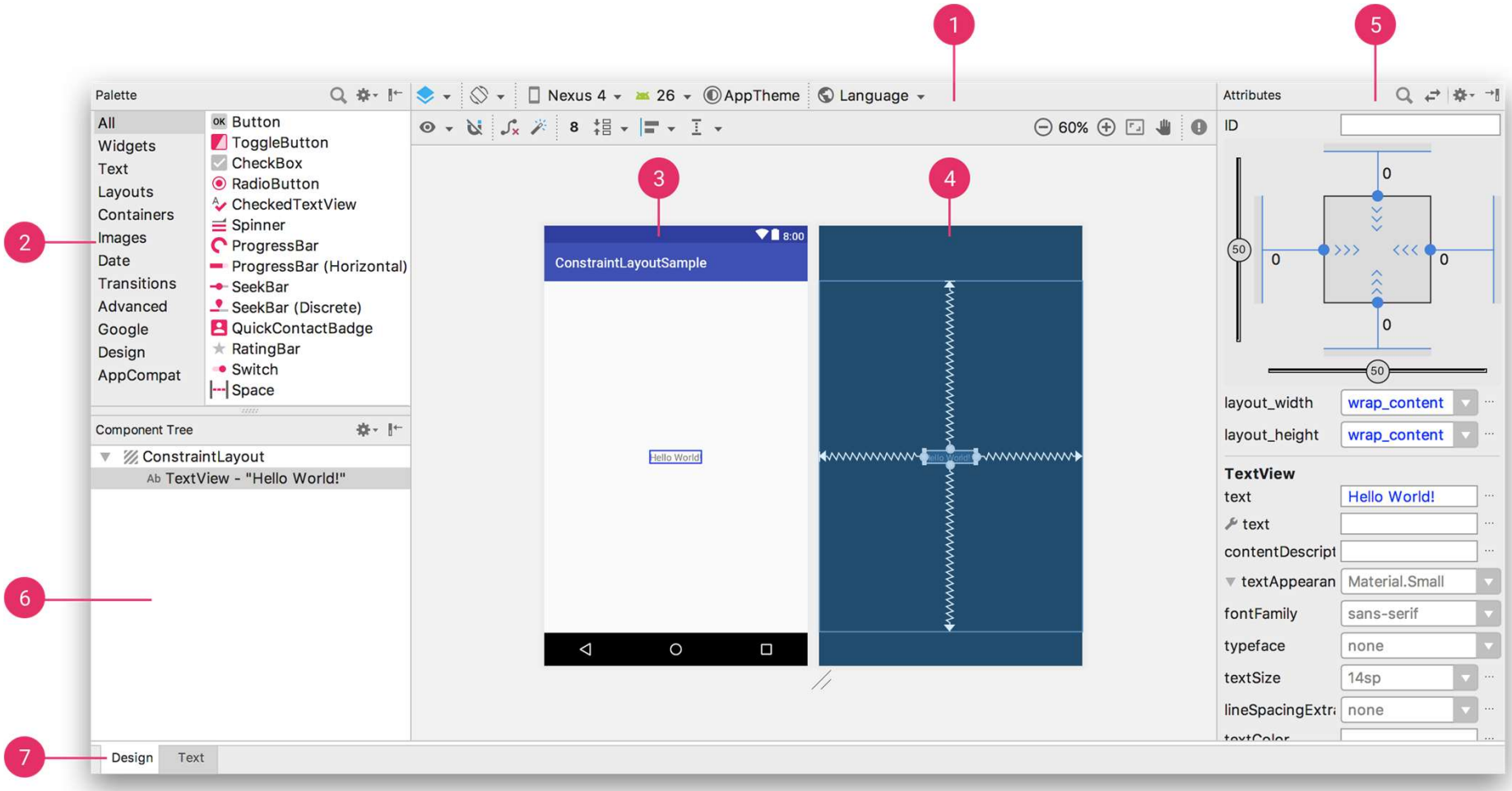


Chương trình đầu tiên

- Chạy lại chương trình



Một số thành phần của Android Studio



Một số thành phần của Android Studio

1. *Toolbar*: Thanh công cụ
2. *Palette*: Bảng lựa chọn các view.
3. *Design view*: Màn hình trực quan. Có thể sử dụng màn hình này để thiết kế và xem kết quả thiết kế.
4. *Blueprint view*: Màn hình xanh đặc biệt dùng cho thiết kế.
5. *Attribute*: chứa các thông số cho việc canh chỉnh giao diện, và một số các thuộc tính khác.
6. *Component tree*: Các view hiển thị ở *Design view* và *Blueprint view* cũng sẽ được hiển thị ở *Component tree* và được sắp xếp trực quan theo dạng cây.
7. *Design & Text*: thay đổi cách thức editor hiển thị giao diện ở dạng kéo thả (tab *Design*) hoặc code XML (tab *Text*).

Quy trình thực thi ứng dụng

- **B1:** Biên dịch và đóng gói ứng dụng thành file **APK**.
- **B2:** Hệ thống tiến hành tìm thiết bị (thật, ảo)
- **B3:** Nếu tìm được thiết bị, tiến hành tải file APK vào Remote Path /data/local/tmp/com.tranduythanh.helloandroidstudio
- **B4:** Cài đặt ứng dụng vào thiết bị
- **B5:** Kích hoạt ứng dụng để chạy
  - Tìm trong AndroidManifest, Activity nào được thiết lập ACTION MAIN sẽ được thực hiện.

Các thành phần ứng dụng Android

- Activity
- View
- Service
- Broadcast Receiver
- Intent
- Content Provider
- Notification

Activity

- Trong ứng dụng Android, Activity là một màn hình, nơi người dùng có thể tương tác với ứng dụng, ví dụ: chụp hình, xem bản đồ, gửi mail…
- Một ứng dụng có thể có một hoặc nhiều Activity, Activity được khởi chạy đầu tiên khi ứng dụng hoạt động được gọi là “MainActivity”.
- Activity có thể hiển thị ở chế độ toàn màn hình, hoặc ở dạng cửa sổ với kích thước nhất định.
- Activity này có thể gọi đến các Activity khác, Activity được gọi sẽ nhận tương tác ở thời điểm đó.

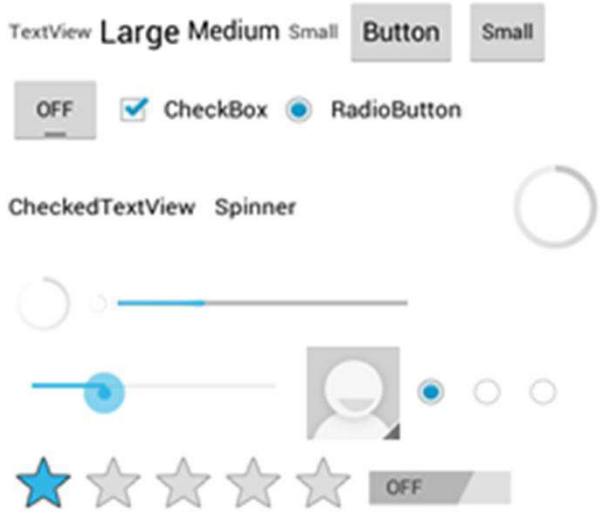
Activity

- Activity settings, được cung cấp bởi hệ thống

- Một activity được bên thứ 3 tự xây dựng

View

- View được sử dụng để tạo ra các điều khiển trên màn hình cho phép nhận tương tác từ người dùng cũng như hiển thị thông tin
- View gồm hai dạng:
  - View: các điều khiển đơn lẻ
  - ViewGroup: tập hợp nhiều điều khiển đơn lẻ



Service

- Service dùng để thực thi các tác vụ ở chế độ ngầm và thường không cần giao diện.
- Service có thể được khởi chạy và hoạt động xuyên suốt ngay cả khi ứng dụng không hoạt động.
- Ví dụ:
  - Điều khiển việc chạy file nhạc
  - Thực hiện việc download/upload dữ liệu
  - Theo dõi và cảnh báo dung lượng pin
  - Ghi nhận ngầm thông tin (GPS chẳng hạn)
  - …

Broadcast Receiver

- Thành phần ứng dụng cho phép truyền tải các thông báo trên phạm vi toàn hệ thống. Không có giao diện nhưng có thể thực hiện thông báo qua thanh trạng thái.
- Ví dụ:
  - Tín hiệu pin yếu
  - Tín hiệu mất kết nối mạng
  - Tín hiệu có cuộc gọi tới
- Có thể chặn các tín hiệu này và xử lý theo cách riêng của mình. Chẳng hạn:
  - Ứng dụng ngắt cuộc gọi đến từ số điện thoại quấy rối
  - Bật âm thanh cảnh báo khi điện thoại đã nạp đầy pin

Intent

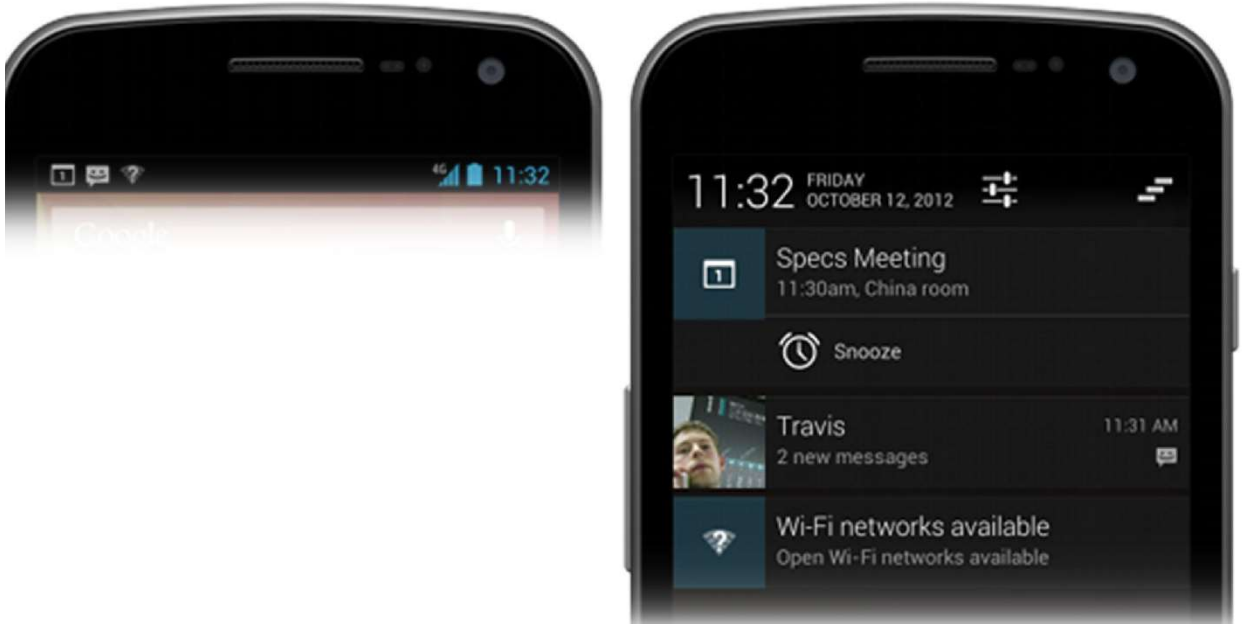
- Intent là cơ chế chuẩn của Android OS để truyền thông tin giữa các thành phần cho nhau (giữa activity với nhau, activity cho service, receiver cho service,…)
- Chẳng hạn có thể sử dụng Intent để:
  - Gọi một service
  - Mở một activity
  - Hiển thị một trang web hoặc danh sách contacts
  - Hiển thị gallery để chọn ảnh

Content Provider

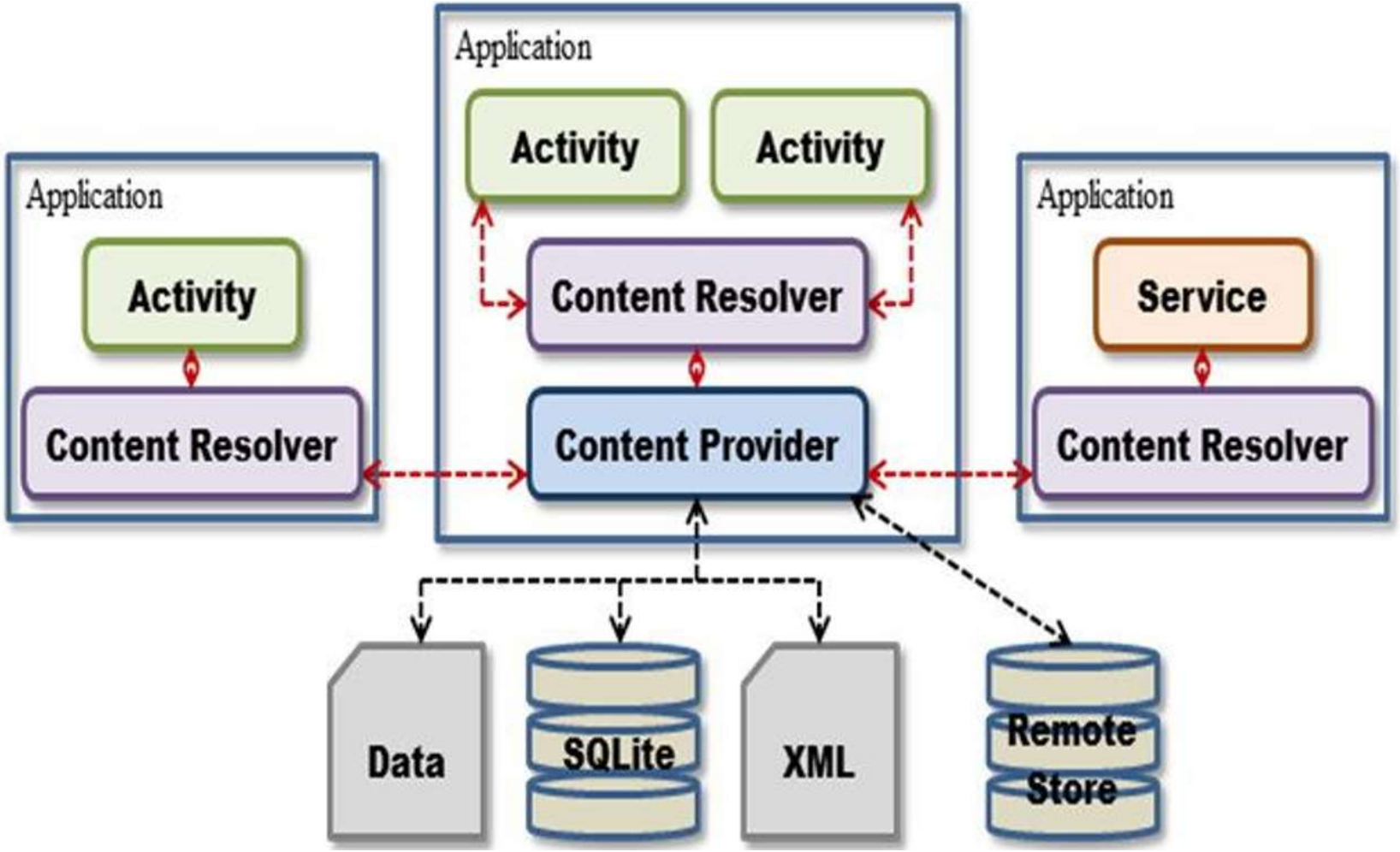
- Content provider dùng để quản lý việc chia sẻ (dùng chung) một nguồn dữ liệu nào đó. Ví dụ:
  - Danh sách người dùng trên điện thoại
  - Dữ liệu về các cuộc gọi
  - Dữ liệu về tin nhắn
- Bằng cách chia sẻ dữ liệu để dùng chung, Android OS làm cho các ứng dụng dễ dàng cung cấp trải nghiệm nhất quán cho người dùng
  - Vd: các ứng dụng thoại dùng chung danh bạ điện thoại

Notification

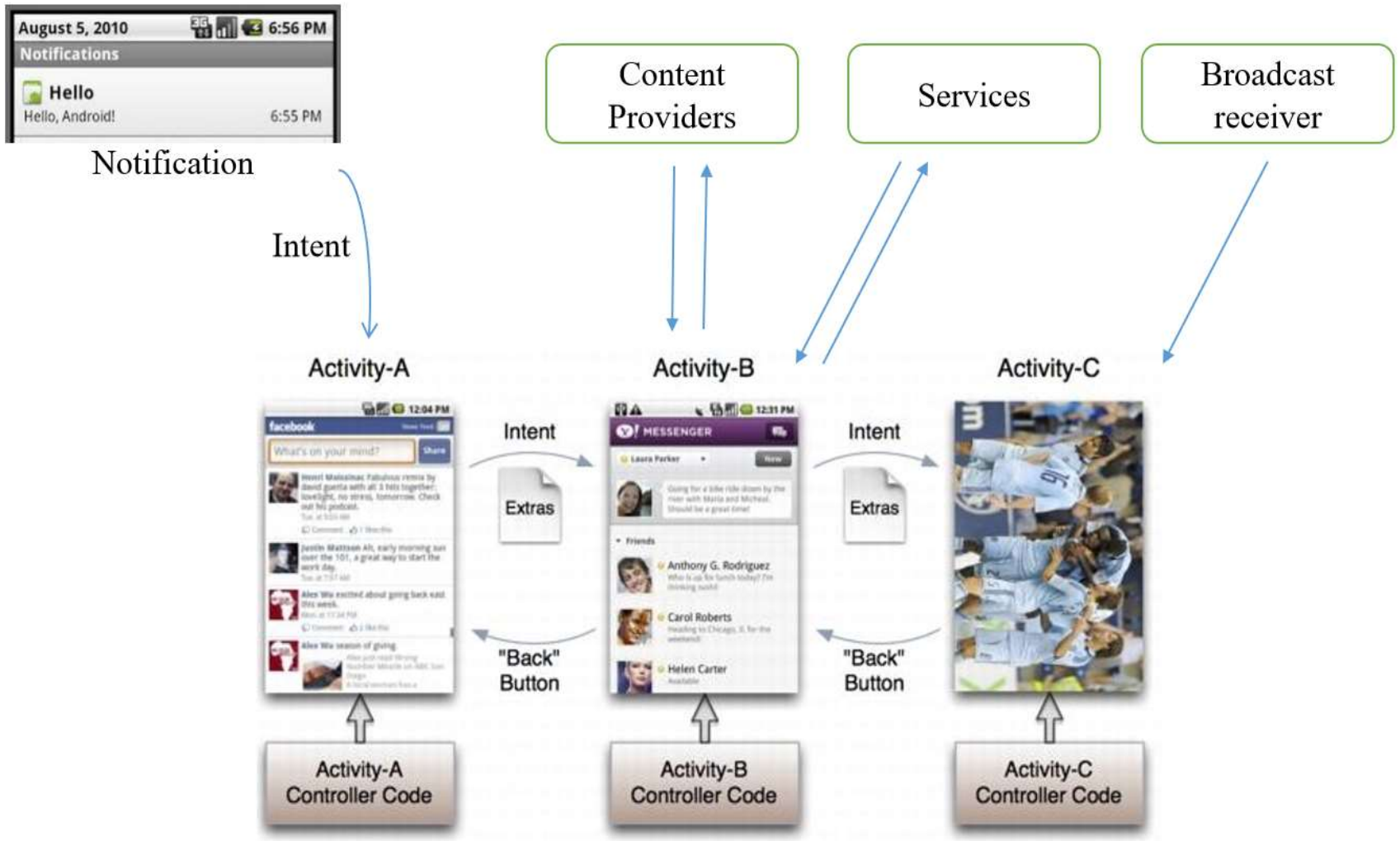
- Notification được xây dựng cho mục đích gửi các thông báo đến người dùng thông qua thanh trạng thái.
- Giao diện Notification không thuộc giao diện ứng dụng, nhưng có thể tuỳ chỉnh giao diện Notification thông qua các phương thức có sẵn.



Các thành phần ứng dụng Android



Cách thực thi điển hình



Ứng dụng Android và cơ chế hoạt động

- Đóng gói và thực thi ứng dụng
  - Ứng dụng Android được viết bằng ngôn ngữ Java và biên dịch, đóng gói cùng các tập tin tài nguyên thành tập tin *.apk
  - Cài đặt trên thiết bị theo đường dẫn data/app/<Tên đóng gói>:
    - Mỗi ứng dụng được cấp một ID, chỉ duy nhất ứng dụng mới có thể truy xuất các tập tin liên quan đến ứng dụng đó.
    - Ứng dụng thực thi riêng biệt trên từng máy ảo.
    - Tiến trình Linux được cấp phát khi bất cứ thành phần ứng dụng được gọi thực thi, và thu hồi khi chấm dứt hoạt động.
    - Các ứng dụng có cùng ID và chứng chỉ (Cerfiticate) có thể truy xuất tài nguyên của nhau, hoặc xin quyền nếu truy xuất hệ thống.

Ứng dụng Android và cơ chế hoạt động

- Tính tương thích ứng dụng với thiết bị bao gồm:
  - Trang bị tính năng của thiết bị
  - Phiên bản hệ điều hành
  - Kích thước màn hình
- Trang bị tính năng của thiết bị:
  - Mỗi tính năng phần cứng và phần mềm trên thiết bị Android được cung cấp một ID, quy định thiết bị đó có được trang bị tính năng đó hay không.
  - Ví dụ:
    - FEATURE_SENSOR_COMPASS: tính năng la bàn
    - FEATURE_APP_WIDGET: tính năng gắn Widget

Ứng dụng Android và cơ chế hoạt động

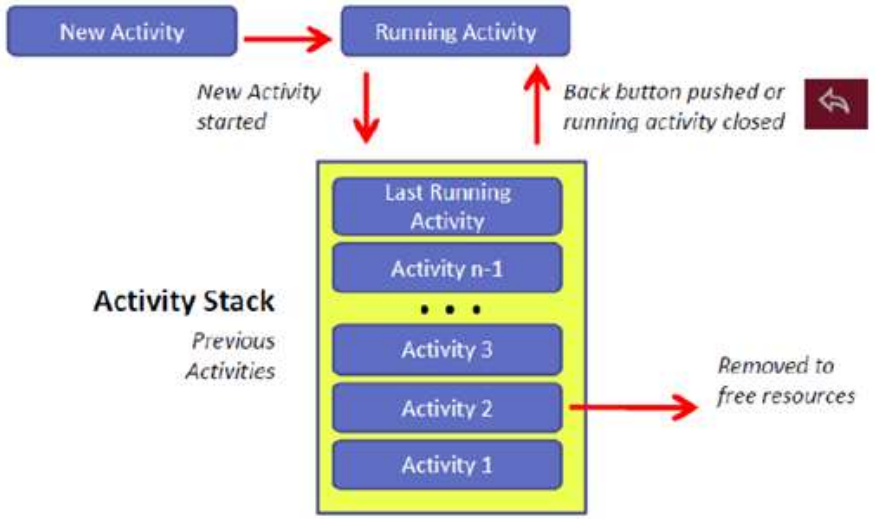
- Tính tương thích ứng dụng với thiết bị bao gồm:
  - Phiên bản hệ điều hành:
    - Mỗi phiên bản kế tiếp được bổ sung hoặc lược bỏ các hàm API, cần thực thi khai báo các thông tin phiên bản để sử dụng đầy đủ các tính năng cho ứng dụng.
    - Ví dụ:
      - Android ICS 4.0: bổ sung API cho Calendar
      - Android Kit Kat 4.4: bổ sung API cho WirelessPrint
  - Kích thước màn hình:
    - Ứng dụng Android cần tương thích với nhiều kích cỡ thiết bị, được phân chia thành hai thuộc tính:
      - Kích thước vật lý màn hình.
      - Độ phân giải (mật độ điểm ảnh)

Activity và độ ưu tiên ứng dụng

- ***Activity*** là thành phần quan trọng nhất và đóng vai trò chính trong xây dựng ứng dụng Android.
- Hệ điều hành Android quản lý Activity theo dạng stack:
  - Khi một Activity mới được khởi tạo, nó sẽ được xếp lên đầu của stack và trở thành running activity
  - Các Activity trước đó sẽ bị tạm dừng và chỉ hoạt động trở lại khi Activity mới được giải phóng.

Activity và độ ưu tiên ứng dụng

- ***Activity Stack là gì?***
  - Activity Stack hoạt động theo cơ chế LIFO (LAST IN FIRST OUT)
  - Mỗi một Activity mới được nó sẽ ở bên trên Activity cũ, để trở về Activity chỉ cần nhấn nút “Back” hoặc viết lệnh. Tuy nhiên nếu nhấn nút Home thì không thể dùng nút “Back” để quay lại màn hình cũ được.



Activity và độ ưu tiên ứng dụng

- Một ***Activity*** cơ bản có 4 trạng thái:
  - *Active*: Activity đang hiển thị trên màn hình (foreground).
  - *Paused*: Activity vẫn hiển thị (visible) nhưng không thể tương tác (lost focus).
  - *Stopped*: Activity bị thay thế hoàn toàn bởi Activity mới sẽ tiến đến trạng thái stopped.
  - *Inactive:* Khi hệ thống bị thiếu bộ nhớ, nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái *stopped* hoặc *paused* cũng có thể bị giải phóng và khi nó được hiển thị lại thì các Activity này phải khởi động lại hoàn toàn và phục hồi lại trạng thái trước đó.

Activity và độ ưu tiên ứng dụng

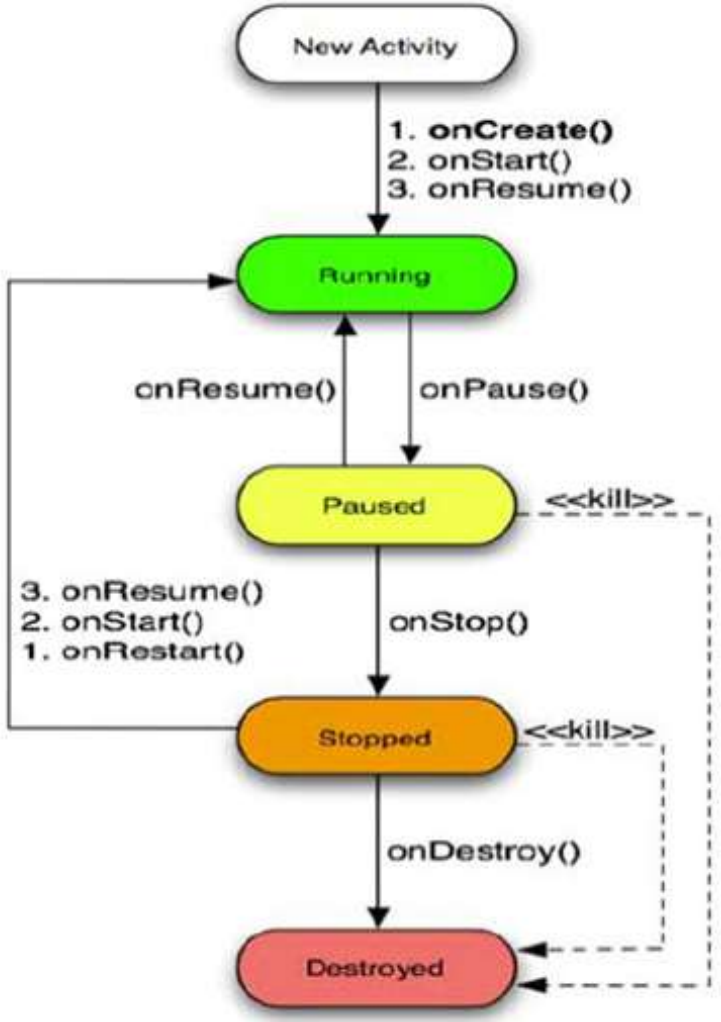
- Khi một activity bị chuyển qua chuyển lại giữa các trạng thái, nó được cảnh báo việc chuyển này bằng hàm chuyển trạng thái (transition)
- Thực hiện gọi các hàm quản lý trạng thái:
  - protected void onCreate(Bundle b);
  - protected void onStart();
  - protected void onRestart();
  - protected void onResume();
  - protected void onPause();
  - protected void onStop();
  - protected void onDestroy();

Activity và độ ưu tiên ứng dụng

- Các hàm quản lý trạng thái
  - onCreate(...): gọi khi activity khởi tạo
  - onStart(): gọi khi acivity xuất hiện trên màn hình
  - onResume(): gọi ngay sau onStart hoặc người dùng focus, hàm này đưa ứng dụng lên top màn hình
  - onPause(): gọi khi hệ thống focus đến activity khác
  - onStop(): gọi khi activity bị che hoàn toàn
  - onRestart(): gọi khi ứng dụng khởi chạy lại
  - onDestroy(): gọi khi ứng dụng chuẩn bị được gỡ khỏi bộ nhớ

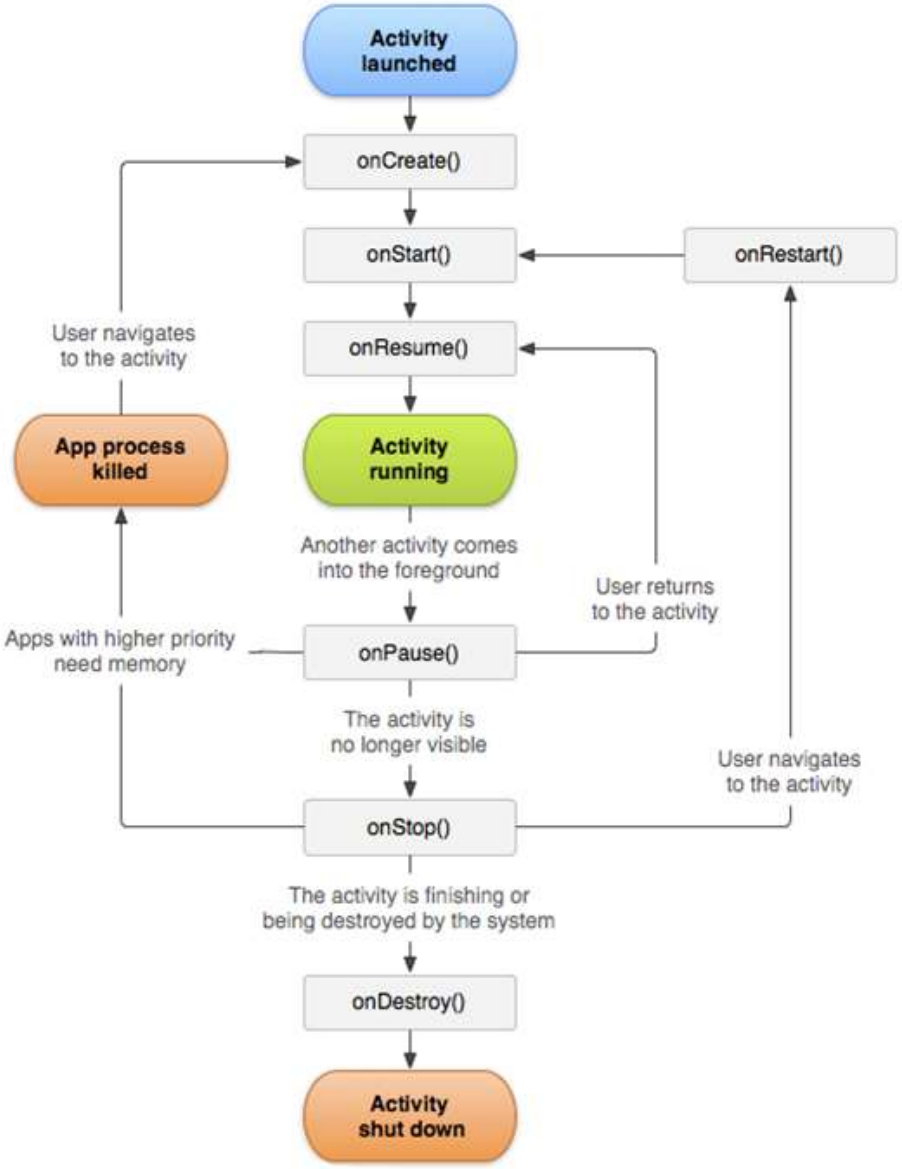
Activity và độ ưu tiên ứng dụng

- Mỗi Activity thường vòng đời có 3 trạng thái sau:
  - 1- **Running** (đang kích hoạt)
  - 2- **Paused** (tạm dừng)
  - 3- **Stopped** (dừng – không phải Destroyed)



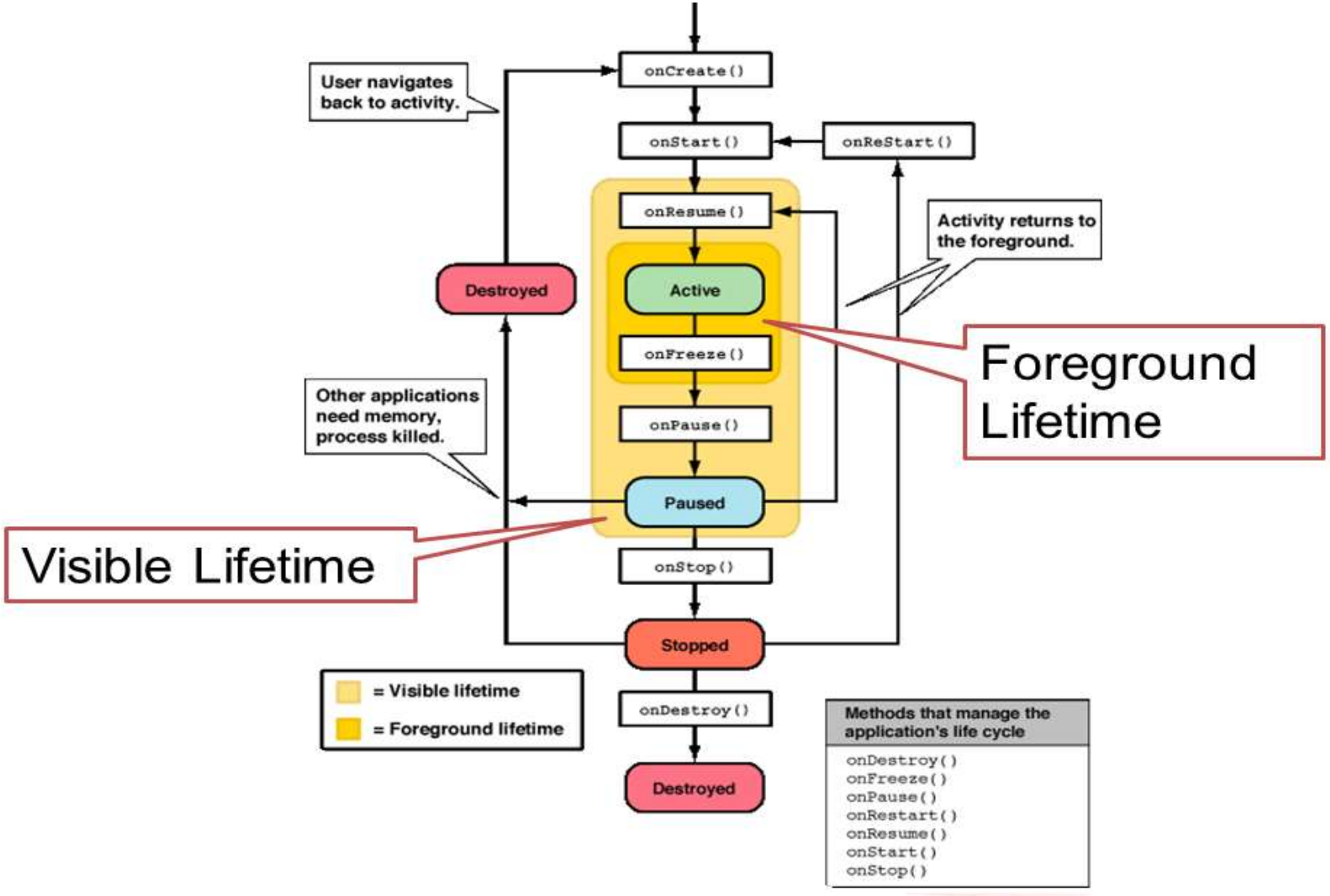
Activity và độ ưu tiên ứng dụng

Application's Life Cycle



Activity và độ ưu tiên ứng dụng

- **Visible Lifetime** và **Foreground Lifetime**



Activity và độ ưu tiên ứng dụng

- **Visible Lifetime** và **Foreground Lifetime**
  - **Visible Lifetime:**
    - Xảy ra từ sau khi gọi onStart –> cho tới lúc gọi onStop : trong trường hợp này ta vẫn có thể thấy màn hình Activity (có thể tương tác khi nó là foreground, không tương tác được khi nó không phải foreground)
  - **Foreground Lifetime:**
    - Xảy ra từ khi gọi onResume –> cho tới lúc gọi onPause: trong suốt thời gian này Activity luôn nằm ở trên cùng và ta có thể tương tác được với nó

Activity và độ ưu tiên ứng dụng

- **Cơ chế quản lý bộ nhớ:**
  - Android quản lý các ứng dụng dựa trên độ ưu tiên.
  - Nếu hai ứng dụng có cùng trạng thái thì ứng dụng nào đã chạy lâu hơn sẽ có *độ ưu tiên* thấp hơn.
  - Nếu ứng dụng đang chạy một *Service* hay *Content Provider* do một ứng dụng khác hỗ trợ thì sẽ có cùng độ ưu tiên với ứng dụng đó.
  - Các ứng dụng sẽ bị đóng mà không có sự báo trước.

Activity và độ ưu tiên ứng dụng

- Độ ưu tiên