

# **GIỚI THIỆU VỀ LẬP TRÌNH JAVA TRÊN ANDROID OS**

# Nội dung

1. Giới thiệu môn học
2. Thiết bị di động vs thiết bị cố định
3. Thị trường ứng dụng cho di động
4. Hệ điều hành Android
5. Lập trình trên android
6. Môi trường lập trình
  - Android Studio
  - Máy ảo Genymotion
1. Chương trình đầu tiên

# Phần 1: giới thiệu môn học

## Môn học “lập trình di động”

- Tên môn học chính xác phải là “làm quen với lập trình cho thiết bị di động”
- Ta chỉ làm quen một chút thôi, muốn lập trình thực sự thì cần học nhiều hơn
- Nên có một thiết bị android nào đó để thử
- Số tín chỉ: 3 (3 lý thuyết)
- Tại sao không có thực hành? Vì lý thuyết quá nhiều, thậm chí 3 tín chỉ vẫn chưa đủ
- Giảng viên:
- mail:

## Tài liệu học tập

- Bài giảng môn học + demo + tài liệu đọc thêm
- Tài liệu tham khảo:
  - ✓ Raymond Gallardo, Scott Hommel, Sowmya Kannan, Joni Gordon, Sharon Biocca Zakhour. The Java Tutorial: A Short Course on the Basics, 6th edition. Online version:  
<https://docs.oracle.com/javase/tutorial/>
  - ✓ Dave MacLean, Satya Komatineni, Grant Allen. Pro Android 5. Apress, 2015

## Kiến thức yêu cầu / nên biết

- Kiến thức về ngôn ngữ lập trình java
- Kiến thức về lập trình hướng đối tượng (cơ bản)
- Kiến thức về SQL (cơ bản)
- Kiến thức về XML (cơ bản)
- Kiến thức về kiến trúc máy tính (đặc biệt là của thiết bị di động)
- Kiến thức về hệ điều hành
- Đã từng sử dụng một thiết bị di động nào đó

## Đánh giá kết quả

- Điểm môn học = ĐQT x 50% + ĐTCK x 50%
- Điểm quá trình:
  - ✓ Điểm danh
  - ✓ Thảo luận
  - ✓ Bài tập
  - ✓ Mini project
- Điểm thi cuối kỳ:
  - ✓ Vấn đáp hoặc trắc nghiệm
  - ✓ Không có giới hạn nội dung thi



## Lợi ích của môn học

- Có kiến thức về lập trình cho thiết bị di động
- Có hiểu biết sâu sắc hơn về hoạt động của các thiết bị di động và phần mềm trên các thiết bị đó, khai thác tốt hơn các thiết bị đó
- Có khả năng viết chương trình đơn giản cho các thiết bị di động
- Có thêm lựa chọn cho đề tài làm tốt nghiệp
- Có điểm môn học và được ra trường



## Nội dung môn học

- Giới thiệu về lập trình di động và Android
- Activity, layout và các điều khiển cơ bản
- Xử lý sự kiện
- Intent, Notification và Menu
- Lưu trữ, SQLite và content provider
- Dịch vụ và Broadcast Receiver
- Khai thác các dịch vụ di động
- Các chủ đề nâng cao

## Công cụ học tập

- Công cụ đề xuất: Android Studio
  - ✓ Công cụ được Google khuyến cáo
  - ✓ Miễn phí, mạnh mẽ, tương thích tốt
  - ✓ Yêu cầu cấu hình cao
- Một số công cụ khác có thể thử
  - ✓ Eclipse, NetBeans, Xamarin, Unity,...
- Tất cả các công cụ trên đều cần bộ phát triển ứng dụng java: JDK (java development kit)
  - ✓ Đề xuất sử dụng phiên bản 8, 64 bit

## **Phần 2: Thiết bị di động vs thiết bị cố định**

## Di động vs Cố định

- Thiết bị di động (với ý nghĩa là giao tiếp không dây) đã xuất hiện từ rất lâu
- Tăng trưởng mạnh về số lượng khi xuất hiện thiết bị dành cho cá nhân (nhỏ, gọn, nhiều khách hàng)
- Bùng nổ khi giá thiết bị giảm (nhiều khách hàng có khả năng mua)
- Thiết bị di động dần thay thế cho thiết bị cố định do việc mua để thay thế thiết bị cũ
- Xuất hiện những chức năng mới, dịch vụ mới và cuối cùng là những loại thiết bị mới

## Chức năng mới

- Giao tiếp kiểu chạm-vuốt (bàn phím hạn chế)
- Tích hợp chụp ảnh, máy chơi nhạc, máy điện thoại và thêm nhiều thiết bị nữa trong tương lai
- Tích hợp các cảm biến, thiết bị có khả năng tương tác tốt hơn do “nhận ra” môi trường xung quanh
  - ✓ Ghi nhận được độ nghiêng của thiết bị
  - ✓ Ghi nhận được gia tốc và hướng di chuyển của thiết bị
  - ✓ Ghi nhận được âm thanh, nhiệt độ, ánh sáng xung quanh
- Nhiều giao tiếp không dây: bluetooth, wifi, nfc,...
- Khai thác tốt các dịch vụ online (GPS, OTT,...)

## Dịch vụ mới

- Tổng hợp tiếng nói (ví dụ: đọc email ra loa)
- Nhận dạng âm thanh, hình ảnh
- Dịch vụ vị trí, bản đồ và di chuyển
- Các dịch vụ sáng tạo trên nền giao thức mạng:
  - ✓ Chat, nhắn tin
  - ✓ Video thoại
  - ✓ Mạng xã hội
  - ✓ Đặt hàng online
  - ✓ Thông tin tức thời
  - ✓ ...

# Loại thiết bị mới



# Phần 3: Thị trường ứng dụng cho di động



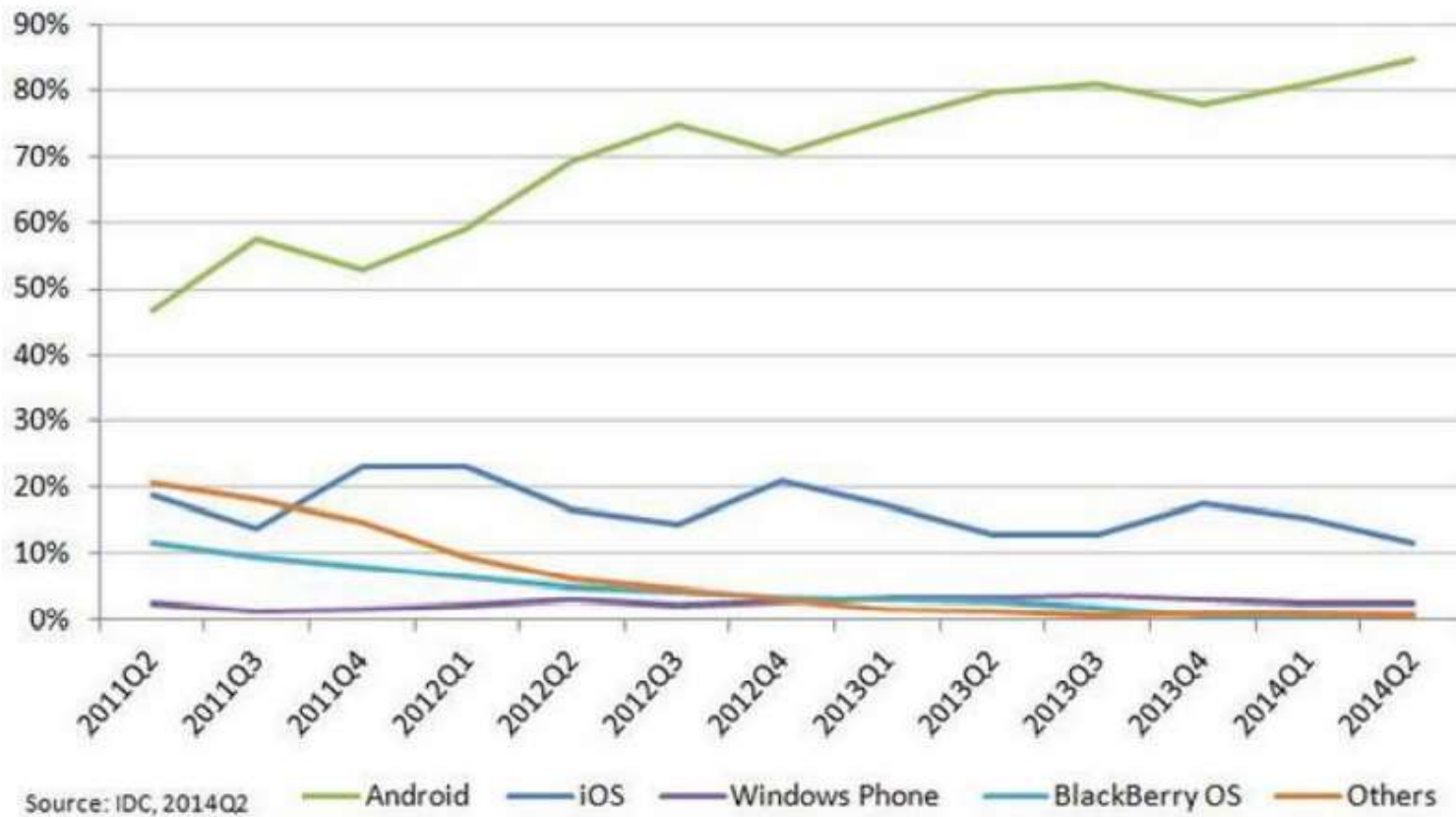
- Sự phát triển của các thiết bị di động thông minh
- Nhu cầu giải trí qua thiết bị di động tăng cao
  - ✓ Cần các ứng dụng giải trí cho di động
  - ✓ Cần nhiều dịch vụ giải trí hỗ trợ di động
- Xuất hiện nhu cầu làm việc qua thiết bị di động
  - ✓ Cần các ứng dụng hỗ trợ công việc
  - ✓ Hệ thống hiện tại cần mở rộng để hỗ trợ di động
- Các tương tác kiểu mới xuất hiện
- Sự phát triển của kênh phân phối
- Sự phát triển của kênh thanh toán

- Thị trường ứng dụng cho di động tăng trưởng nóng
  - ✓ Chuyển đổi các ứng dụng đã có lên di động
  - ✓ Chuyển đổi các ứng dụng di động sang loại thiết bị mới
  - ✓ Phát triển những ứng dụng mới hoàn toàn, khai thác khả năng đặc biệt của di động
- Nhu cầu nhân lực viết phần mềm cho di động cao
  - ✓ Tăng trưởng về lương cho người làm di động
  - ✓ Đỡ nhàm chán vì xuất hiện những công nghệ mới
- Cơ hội thực hiện các ý tưởng mới
  - ✓ Tự viết và bán ứng dụng: không còn quá khó như trước

# Các nền tảng dùng cho di động

- (1973) embedded OS
- (1996) Palm OS
- (1996) Windows CE
- (1999) Nokia S40
- (2000) Symbian
- (2002) BlackBerry
- (2005) Maemo OS (Nokia)
- (2007) iOS
- (2008) Android
- (2009) webOS (Palm)
- (2009) Bada (Samsung)
- (2010) Windows Phone
- (2011) MeeGo
- (2012) Firefox OS
- (2013) Ubuntu Touch
- (2013) Sailfish OS
- (2013) Tizen

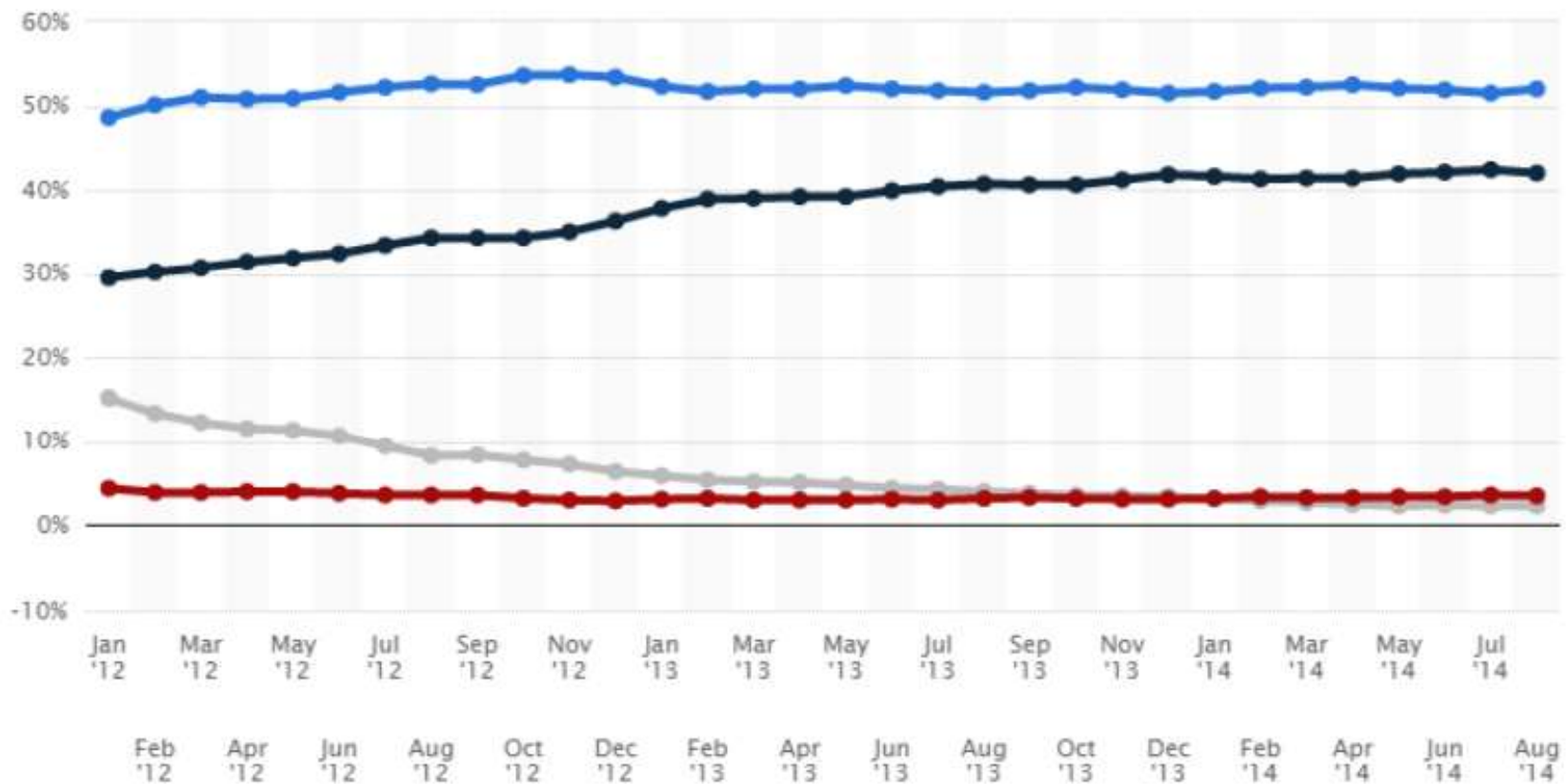
# Thị phần các nền tảng toàn cầu



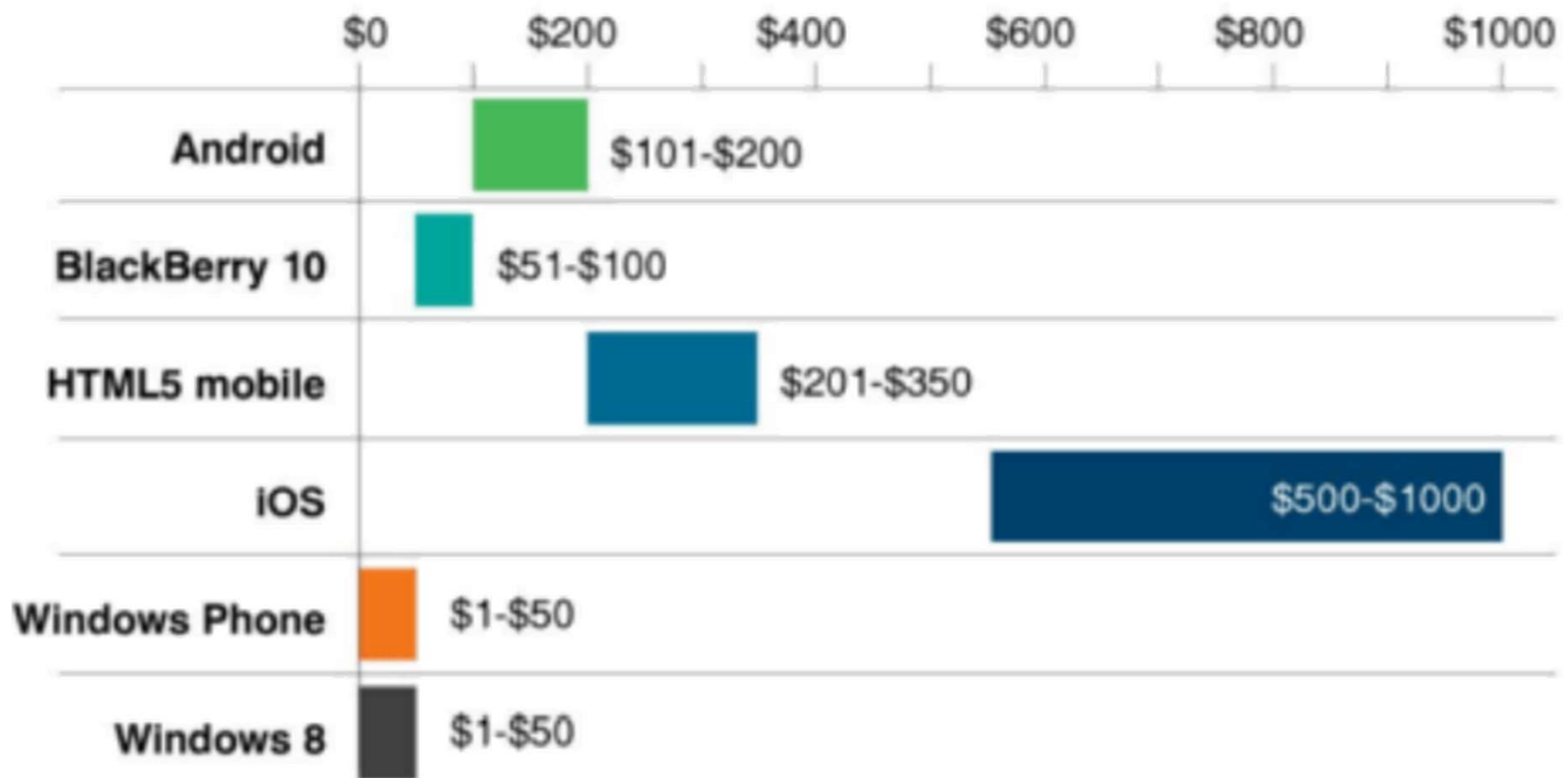
Period	Android	iOS	Windows Phone	BlackBerry OS	Others
Q2 2014	84.7%	11.7%	2.5%	0.5%	0.7%
Q2 2013	79.6%	13.0%	3.4%	2.8%	1.2%
Q2 2012	69.3%	16.6%	3.1%	4.9%	6.1%
Q2 2011	36.1%	18.3%	1.2%	13.6%	30.8%

- Nền tảng android thống trị về số lượng
- Nền tảng iOS giảm nhưng chậm
- Không có nhiều cơ hội cho các tay chơi khác ngoại trừ xuất hiện một loại thiết bị có tính đột phá (như iPhone trước kia)

# Thị phần các nền tảng tại Mỹ



## Thu nhập trên mỗi app/month



## So sánh 3 chợ ứng dụng

	Google	Apple	Microsoft
Number of users (in millions)	900	600	12
Number of apps (in thousands)	800	1250	160
Number of developers (in thousands)	150	235	45
Number of downloads (in billions)	48	50	.65
Paid to developers (in millions)	900	5000	100

	Google	Apple	Microsoft
Number of apps per developers	5	5	3
Number of downloads per app	60,000	40,000	4,062
Revenue per download	\$.01875	\$.1	\$.1538



# **Phần 4: Hệ điều hành Android**

# 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, Qualcomm,...
- Có 2 phiên bản song song: Android & Google API

## 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 27 Android: lịch

## Lịch sử phát triển

- Google mua Android Inc 17-8-2005
- Ra mắt cộng đồng tháng 11-2007, thành lập OHA
- Phiên bản 1.0 ra mắt tháng 9-2008
- Phiên bản 1.1 ra mắt tháng 2-2009
- Phiên bản 1.5 (Cupcake) ra mắt tháng 4-2009
- ...
- Phiên bản 5.0 (Lollipop) ra mắt tháng 10-2014
- Phiên bản 6.0 (Marshmallow) ra mắt tháng 11-2015
- Phiên bản 7.0 (Nougat) ra mắt tháng 3-2017

# Nâng cấp & mở rộng



Cupcake  
1.5



Donut  
1.6



Eclair  
2.0/2.1



Froyo  
2.2



Gingerbread  
2.3



Honeycomb  
3.0



Android 5.0, Lollipop

android 6.0  
Marshmallow



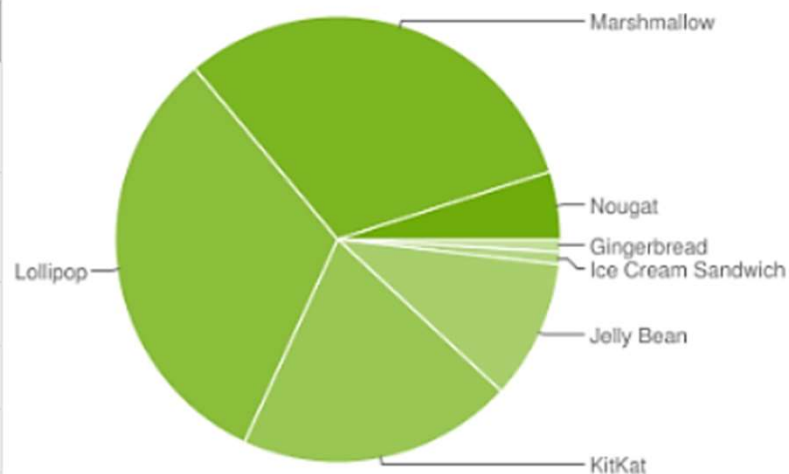
Jelly Bean



Android 7.0  
NOUGAT

# Phân mảnh (3/4/2017)

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.9%
4.1.x	Jelly Bean	16	3.5%
4.2.x		17	5.1%
4.3		18	1.5%
4.4	KitKat	19	20.0%
5.0	Lollipop	21	9.0%
5.1		22	23.0%
6.0	Marshmallow	23	31.2%
7.0	Nougat	24	4.5%
7.1		25	0.4%



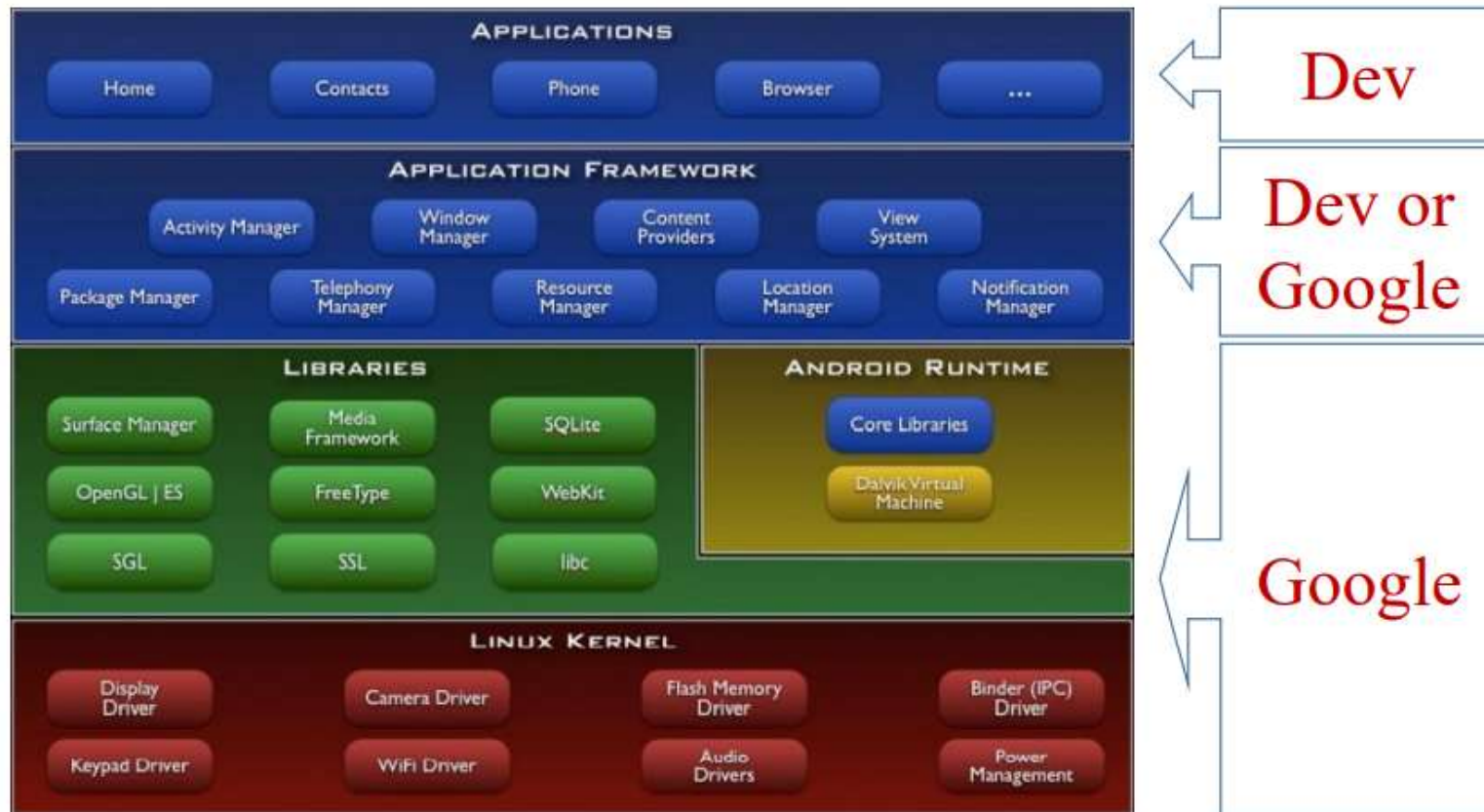
# **Phần 5: Lập trình trên Android**

# Kiến trúc OS

- Android OS chia thành tầng ứng dụng và các tầng trung gian để LTV có thể mở rộng hoặc tùy chỉnh theo mục đích ứng dụng họ viết
- Có 4 tầng trong HDH Android gồm:
  - ✓ Application Framework
  - ✓ Android Runtime
  - ✓ Native Libraries
  - ✓ Linux Kernel
- Tầng cao hơn sử dụng API của các tầng bên dưới
- Về lý thuyết thì LTV can thiệp được vào mọi tầng



# Kiến trúc OS



# Kiến trúc OS

- Linux Kernel: thấp nhất
  - ✓ 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 OS

- 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 (mở rộng từ PacketVideo's OpenCORE) thư viện hỗ trợ playback và recording của nhiều định dạng video, audio và image phổ biến
    - ✓ MPEG4
    - ✓ H.264
    - ✓ MP3
    - ✓ AAC
    - ✓ JPG/PNG/GIF

# Kiến trúc OS

- Native Libraries (tiếp):
  - ✓ Surface Manager: quản lý việc hiển thị và kết hợp đồ họa 2D và 3D
  - ✓ OpenGL: thư viện đồ họa tiêu chuẩn
  - ✓ 3D libraries: thư viện 3D dựa trên OpenGL ES, có nâng cấp tăng tốc "hardware 3D acceleration"
  - ✓ SSL: thư viện hỗ trợ mã hóa kết nối mạng
  - ✓ SQLite: động cơ cơ sở dữ liệu của ứng dụng
  - ✓ Webkit: bộ diễn dịch HTML, CSS & Javascript
  - ✓ ...

# Kiến trúc OS

- Android Runtime: hỗ trợ việc chạy ứng dụng
  - ✓ Máy ảo Dalvik: giúp thực thi các ứng dụng android, mỗi ứng dụng chạy trên một tiến trình riêng của Dalvik VM
  - ✓ Máy ảo Dalvik thực thi các file mang định dạng .dex (Dalvik Executable), định dạng này là định dạng đã được tối ưu hóa để chỉ chiếm một vùng nhớ vừa đủ dùng và nhỏ nhất có thể
  - ✓ Máy ảo ART, xuất hiện trong các phiên bản Android mới, sử dụng kỹ thuật biên dịch tức thời để có thể giúp ứng dụng chạy nhanh hơn, không hoàn toàn tương thích với mọi ứng dụng Android hiện thời

# Kiến trúc OS

- Application Framework
  - ✓ Nơi cung cấp các API có sẵn để sử dụng các tính năng của phân cứng mà không cần hiểu cấu trúc bên dưới
  - ✓ Các API được chia thành các nhóm: View UI, Content Providers, Resource Manager, Notification Manager, Activity Manager,...
  - ✓ Cung cấp các thành phần cơ bản để tạo nên ứng dụng Android mà ta thường thấy
  - ✓ Các thành phần của tầng này gần như tương đương 1-1 với các gói thư viện java trong Android SDK do Google cung cấp cho các nhà phát triển ứng dụng Android

## Ưu điểm

- Mã nguồn mở
- Miễn phí
- Đơn giản
- Mạnh mẽ
- Phổ biến (tài liệu, mã minh họa, thư viện)
- Sử dụng JAVA + XML để viết ứng dụng
- Thị phần lớn
- Kênh phân phối sẵn có
- Cộng đồng phát triển đông đảo 39 Lập trình android: SD

## SDK và NDK

- Có thể viết ứng dụng android bằng nhiều ngôn ngữ và nhiều cách khác nhau
  - ✓ Viết bằng Java, chạy trên máy ảo: dùng SDK
  - ✓ Viết bằng C/C++ chạy trực tiếp trên CPU: dùng NDK
- SDK: viết nhanh, chạy chậm, chi phí thấp, tương thích cao, bảo trì dễ
- NDK: viết lâu, chạy nhanh, chi phí cao, tương thích thấp, bảo trì khó
- Ngoài ra có thể dùng các ngôn ngữ lập trình khác hoặc các framework của nhà phát triển thứ ba



# **Phần 6: Môi trường lập trình**

## Môi trường lập trình

- 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) 1.6 or higher
  - ✓ Android SDK
  - ✓ IDE (Android Studio, Eclipse, Netbean,...)

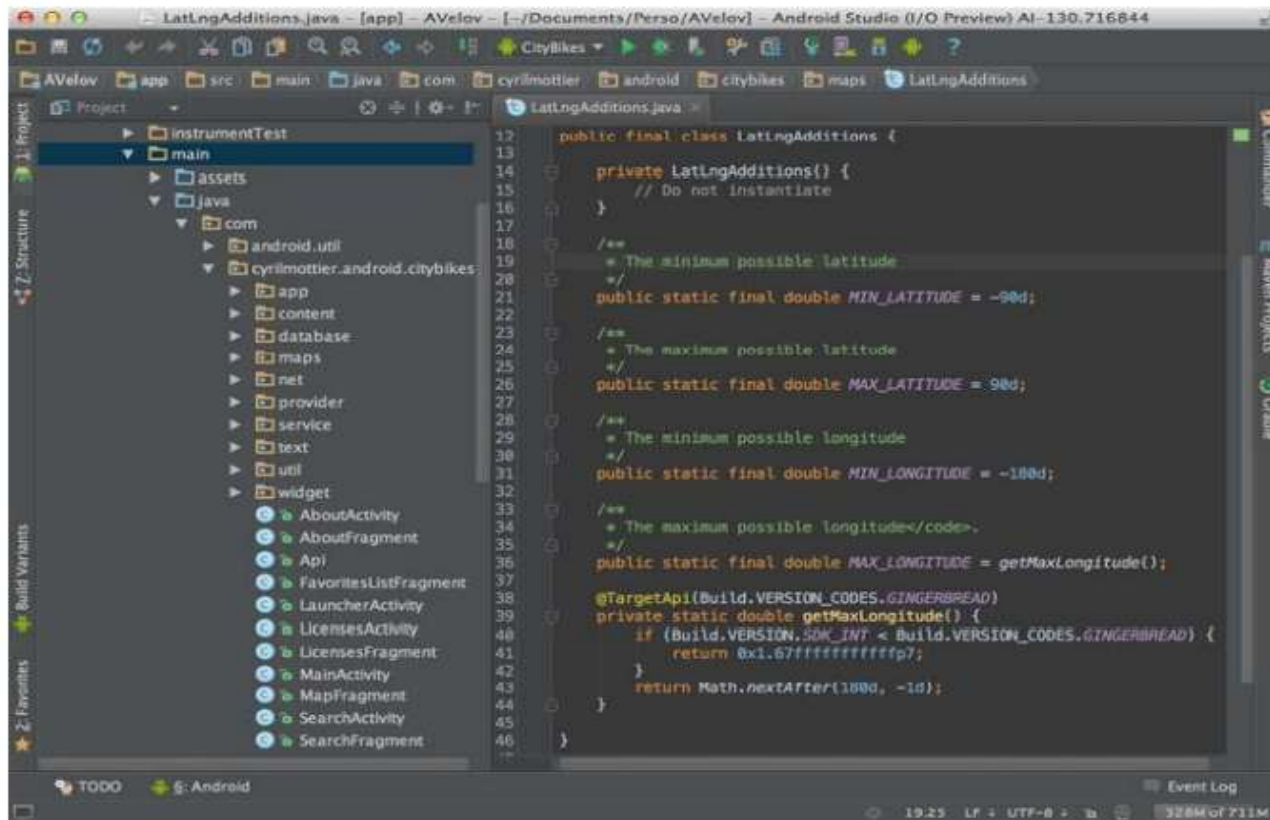
# Môi trường lập trình

- Android SDK (Android Software Development Kit): công cụ phát triển ứng dụng cho Android OS, chỉ gồm tài liệu, máy ảo và các công cụ dịch, không có giao diện phát triển tích hợp (IDE)
- Các thành phần chính của Android SDK:
  - ✓ Bộ công cụ giúp dịch mã java thành ứng dụng
  - ✓ Các công cụ tiện ích cho lập trình viên, để có thể dễ dàng tìm hiểu và xử lý các vấn đề đặt ra khi viết ứng dụng
  - ✓ Nhóm các tài nguyên ứng với từng bản Android OS
  - ✓ Thư viện bổ sung cho phép LTV dễ dàng khai thác các dịch vụ của Google (Maps, AdMod, YouTube,...)

## Môi trường lập trình

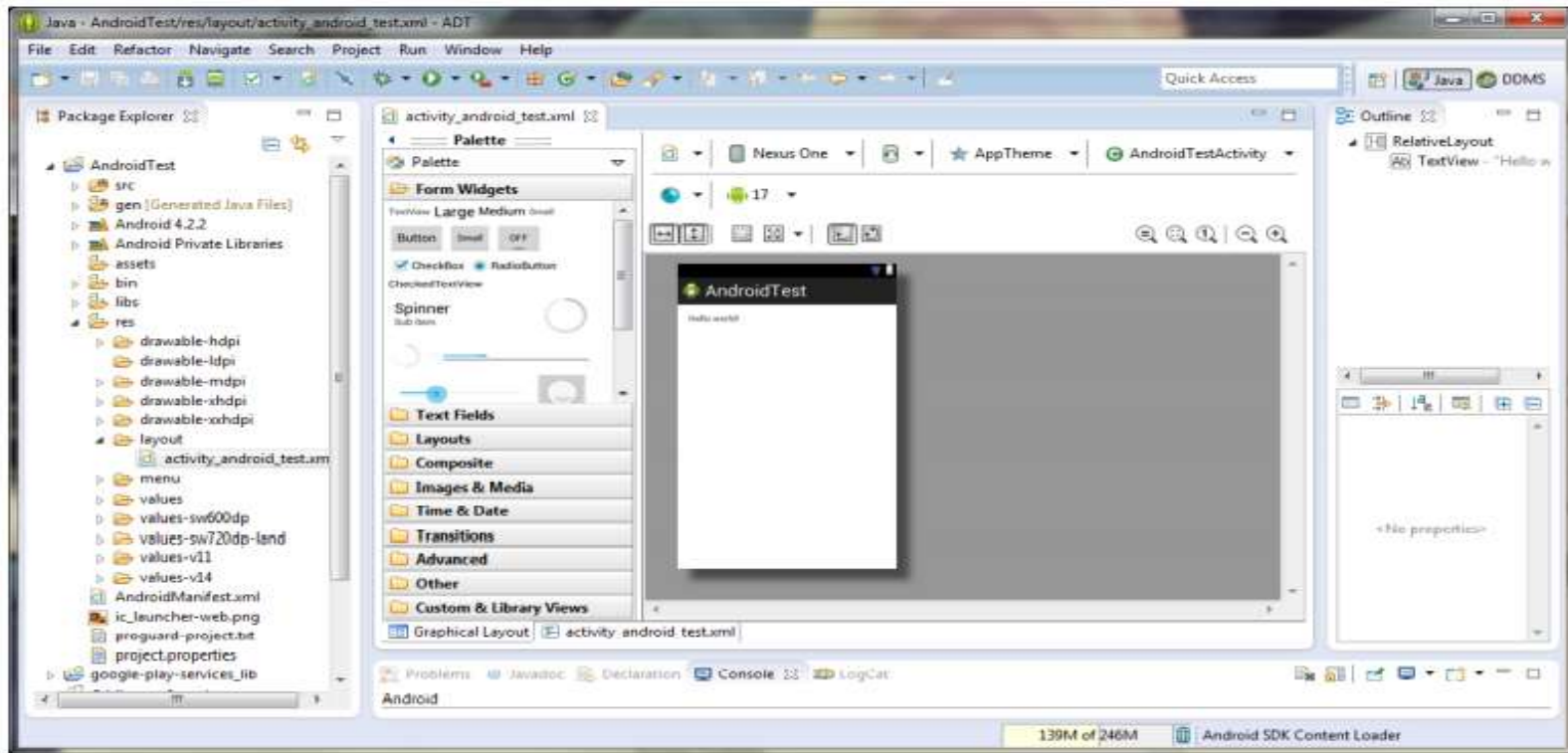
- Với một phiên bản Android OS, SDK cung cấp:
  - ✓ Tài liệu lập trình ứng với phiên bản đó
  - ✓ Thư viện các gói lập trình cơ bản cho phiên bản đó
  - ✓ Thư viện các gói lập trình bổ sung cho phép khai thác dịch vụ của Google (Google APIs) ứng với phiên bản đó
  - ✓ Các file ảnh để tạo máy ảo cho phiên bản hiện tại
  - ✓ Mã nguồn của phiên bản hiện tại
- Chú ý: Android SDK có thể tải về từng phần liên quan tới nội dung cần phát triển, bản đầy đủ kích thước khá lớn (vài chục GB)

# IDE cho phát triển android app



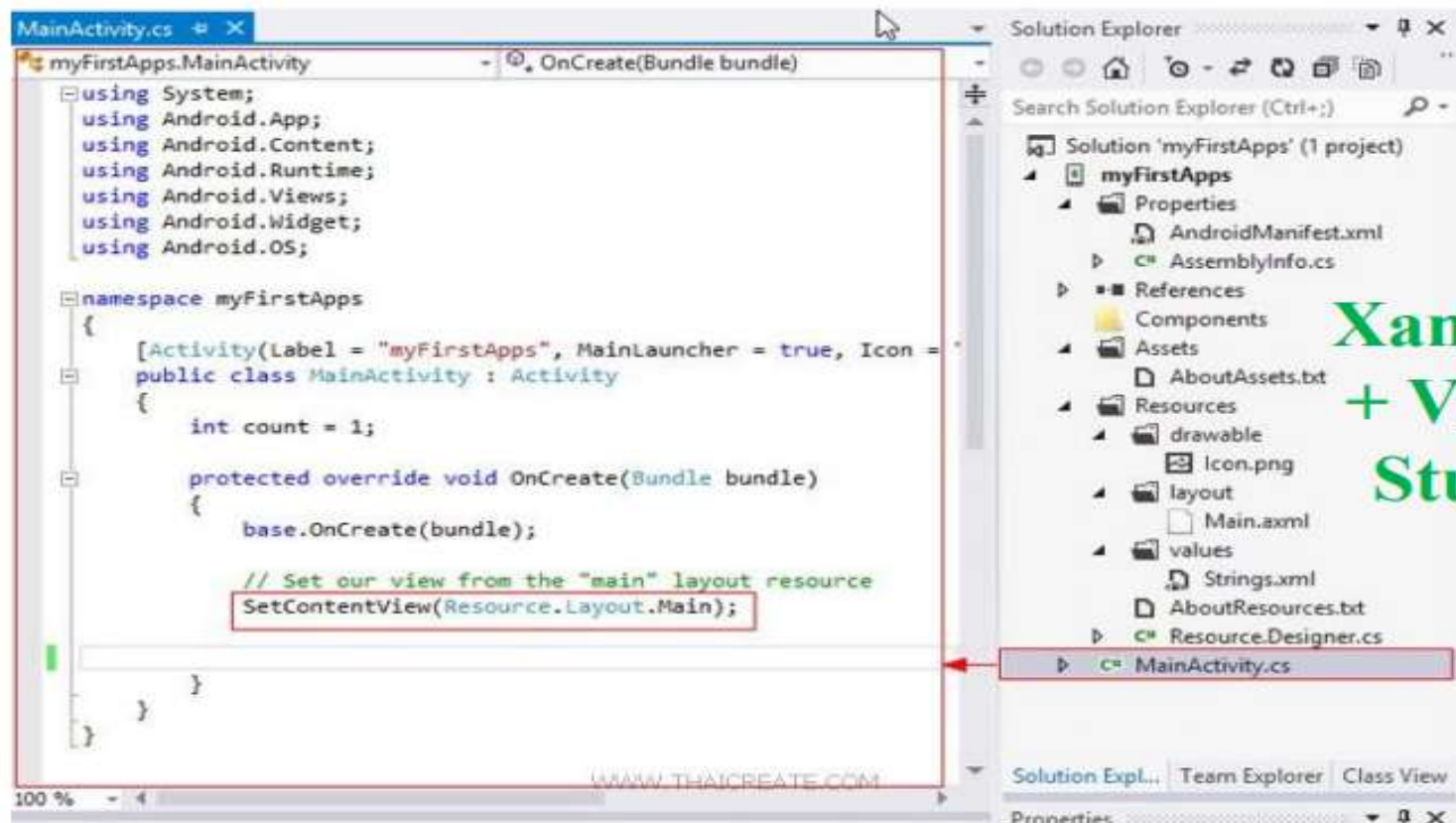
Android  
Studio

# IDE cho phát triển android app



**Eclipse + Android**

# IDE cho phát triển android app



# **Phần 6.1: Cài đặt Android Studio**



# Cài đặt Android Studio

- Android Studio là bộ công cụ phát triển riêng cho android, được google tự xây dựng, giới thiệu chính thức vào tháng 5/2013
- Dựa trên bộ IDE khá nổi tiếng IntelliJ IDEA
- Ưu điểm:
  - ✓ “Hàng chính chủ”
  - ✓ Hỗ trợ android tốt hơn so với Eclipse
- Nhược điểm:
  - ✓ Chạy khá chậm
  - ✓ Không ổn định khi làm việc trên Windows

# Cài đặt Android Studio

- Cài đặt theo hướng dẫn trên trang chủ  
✓ <http://developer.android.com/sdk/index.html>
- Hỗ trợ cả Windows, Mac OS và Linux
- Yêu cầu phải có Java SDK cài đặt sẵn từ trước
- Bản thông dụng đã tích hợp sẵn Android SDK
- Vẫn có thể sử dụng lại Android SDK từ trước (chẳng hạn như tình huống dùng chung Android SDK với eclipse hoặc IDE khác)

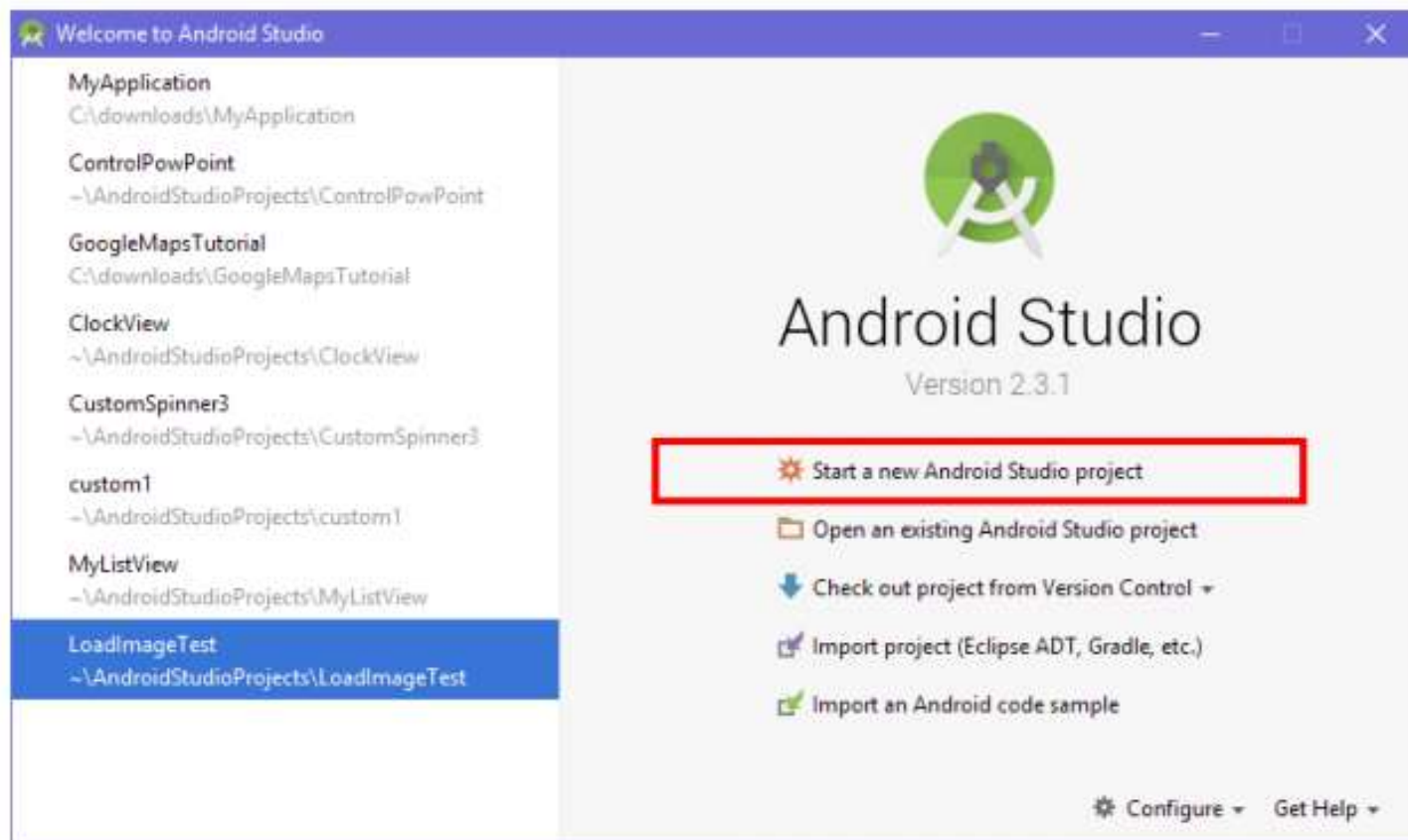
## **Phần 6.2: Máy ảo genymotion**

## Máy ảo genymotion

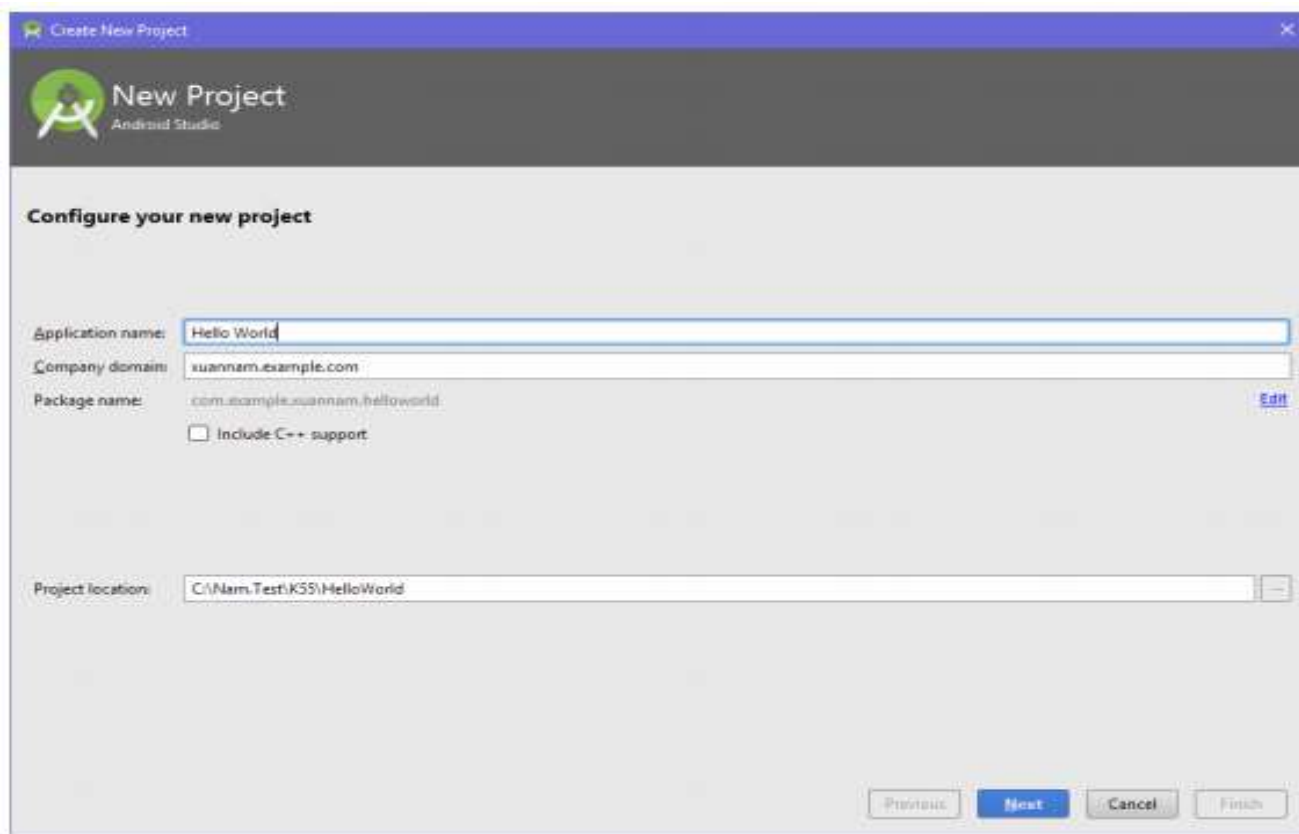
- Một trong những công đoạn quan trọng trong phát triển phần mềm là chạy thử sản phẩm
- Sử dụng thiết bị thật:
  - ✓ Bật chế độ usb debug
  - ✓ Bật chế độ developer
  - ✓ Cho phép sử dụng app từ “unknown source”
- Sử dụng thiết bị ảo: chạy giả lập trên máy tính, hỗ trợ hạn chế, chậm, chi phí thấp
- Máy ảo genymotion: nhanh, giống với máy thật
  - ✓ <http://www.genymotion.com>

# **Phần 7: Chương trình đầu tiên**

# Khởi động Android Studio



# Đặt tên ứng dụng, tên package,...



Create New Project

New Project  
Android Studio

Configure your new project

Application name: Hello World

Company domain: xuannam.example.com

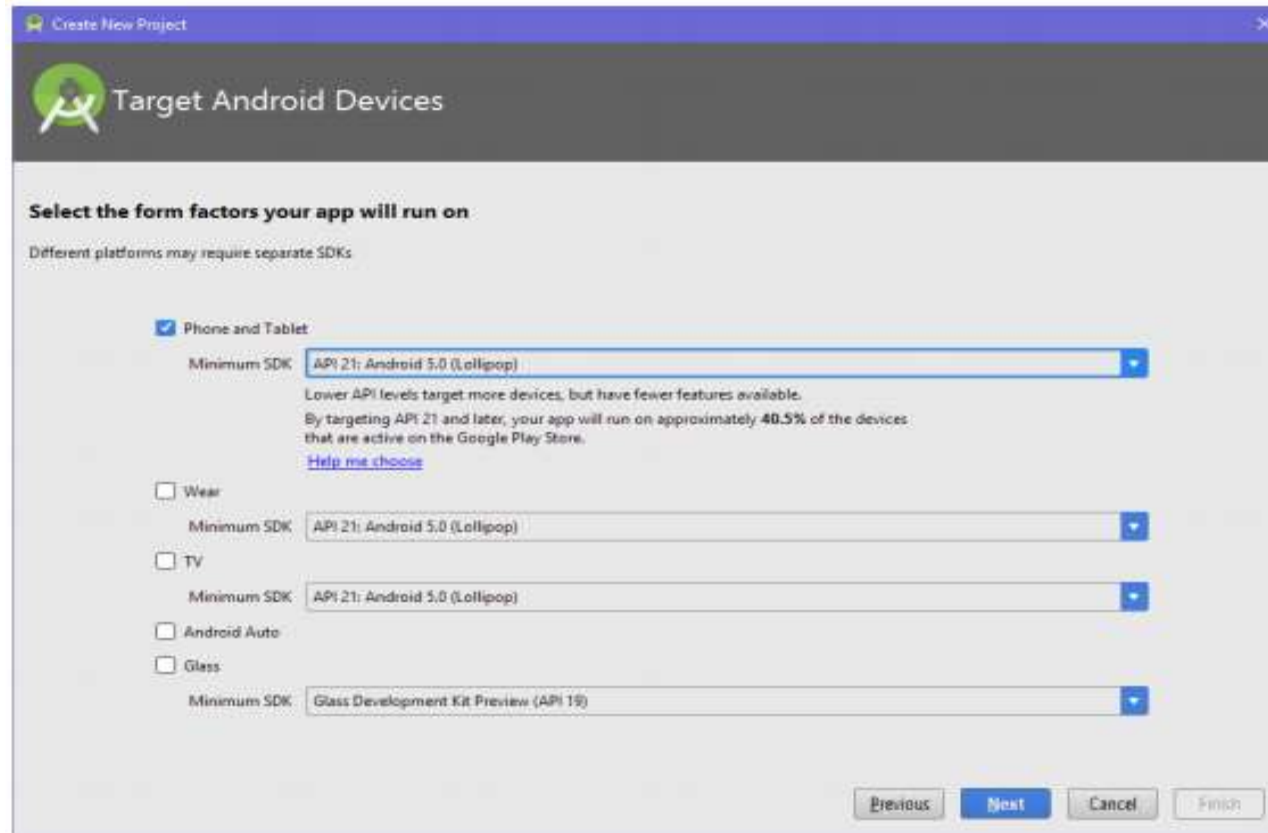
Package name: com.example.xuannam.helloworld [Edit](#)

☐ Include C++ support

Project location: C:\Nam.Test\KSS\HelloWorld

Previous Next Cancel Finish

# Chọn phiên bản hệ điều hành



Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK: API 21: Android 5.0 (Lollipop)

Lower API levels target more devices, but have fewer features available.  
By targeting API 21 and later, your app will run on approximately 40.5% of the devices that are active on the Google Play Store.  
[Help me choose](#)

☐ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ Android Auto

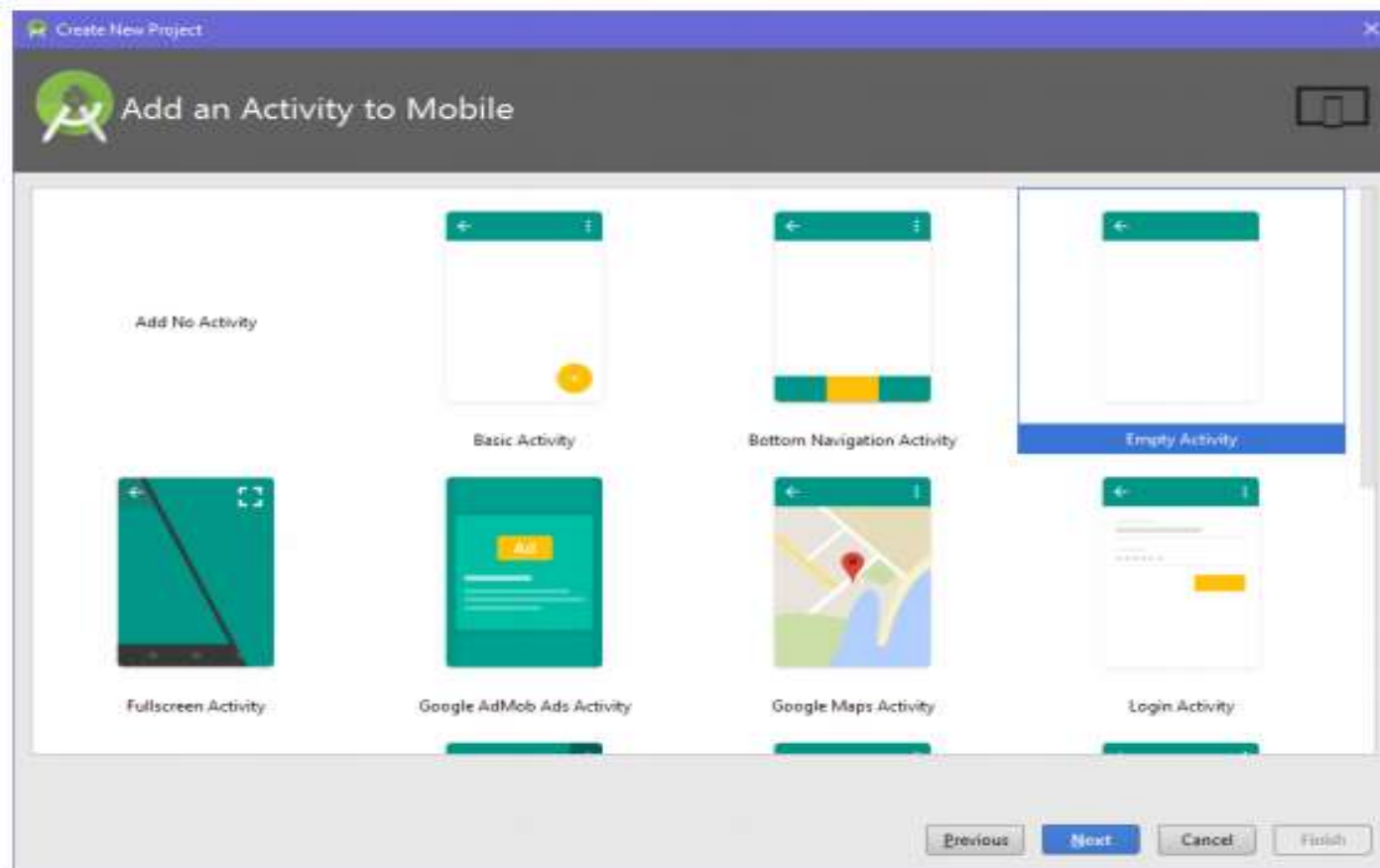
☐ Glass

Minimum SDK: Glass Development Kit Preview (API 19)

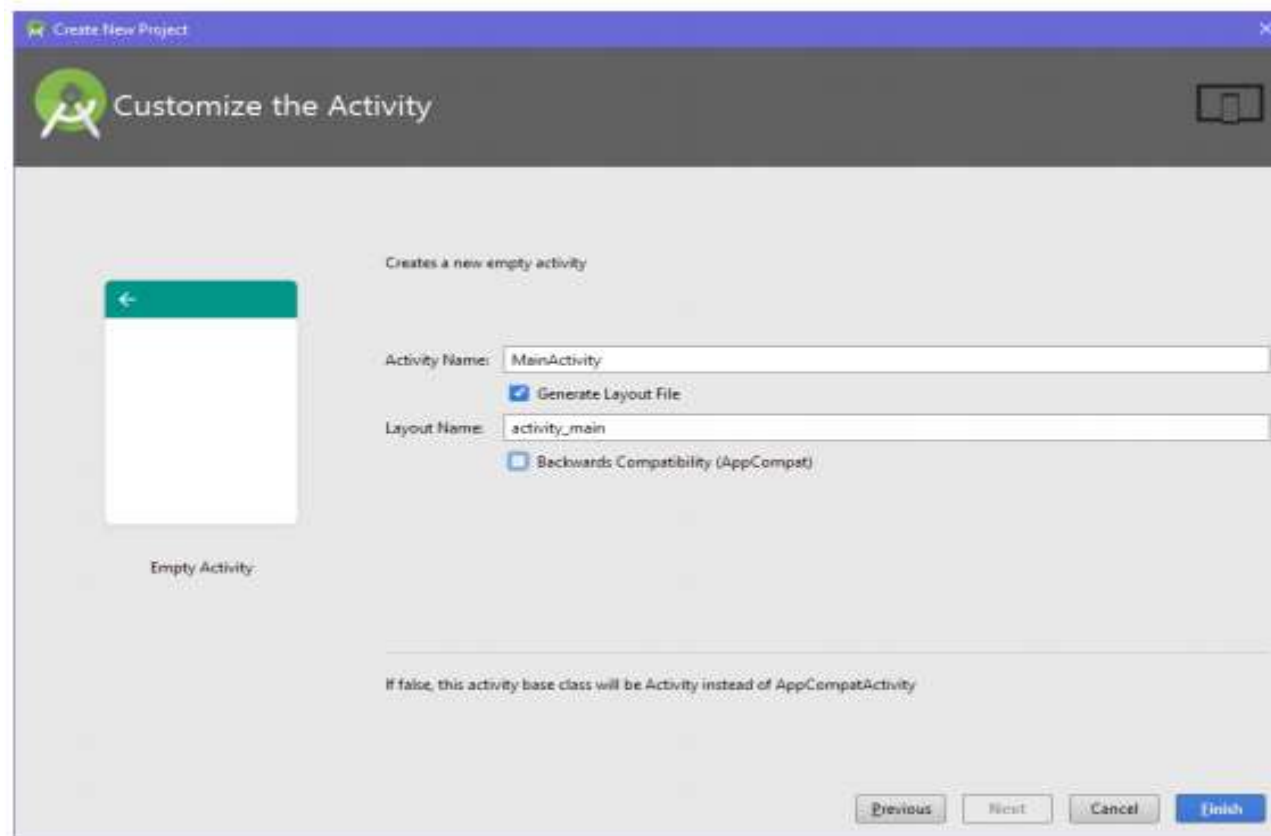
Previous Next Cancel Finish



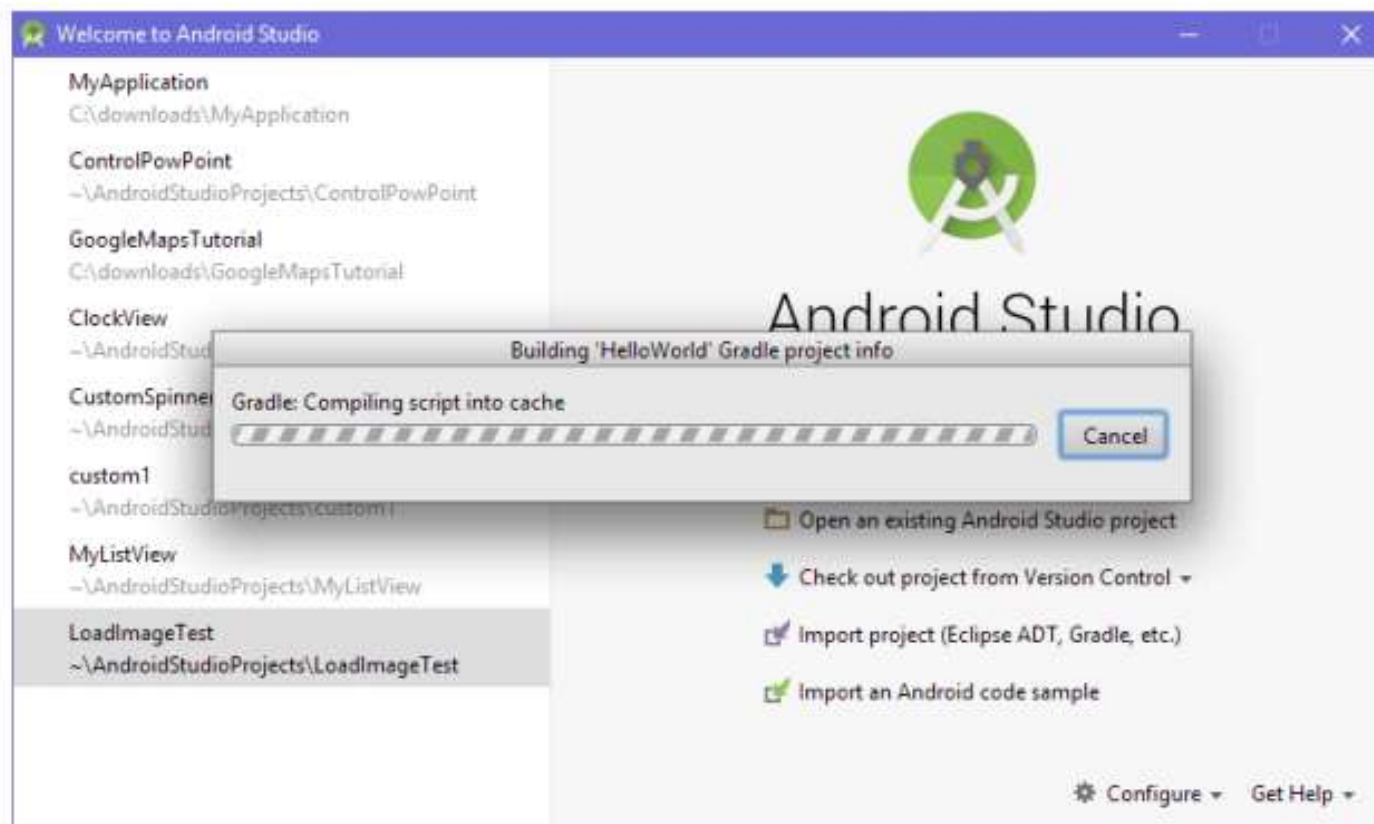
# Chọn giao diện ban đầu



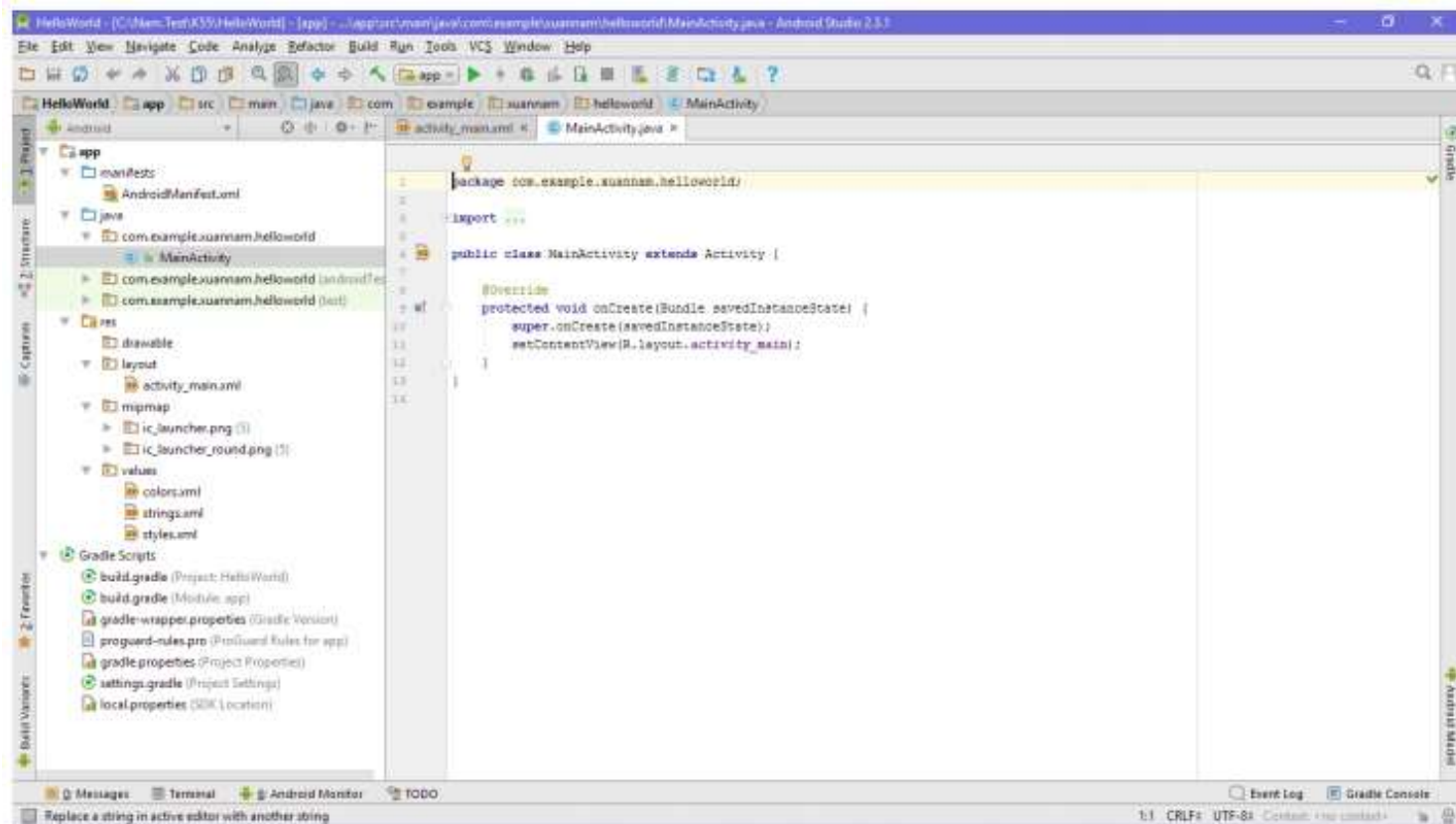
# Đặt tên cho giao diện



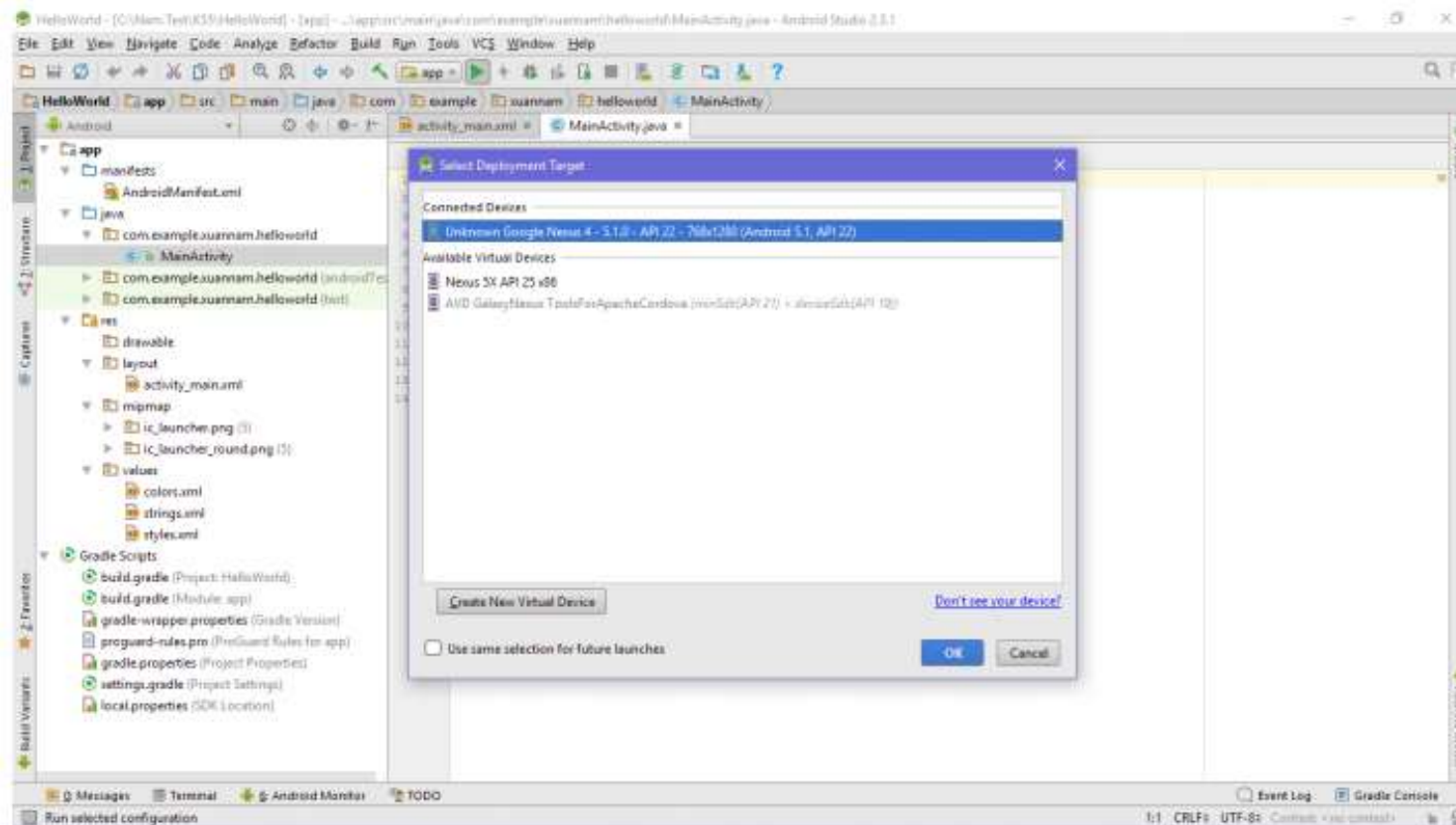
# Android Studio thiết đặt project



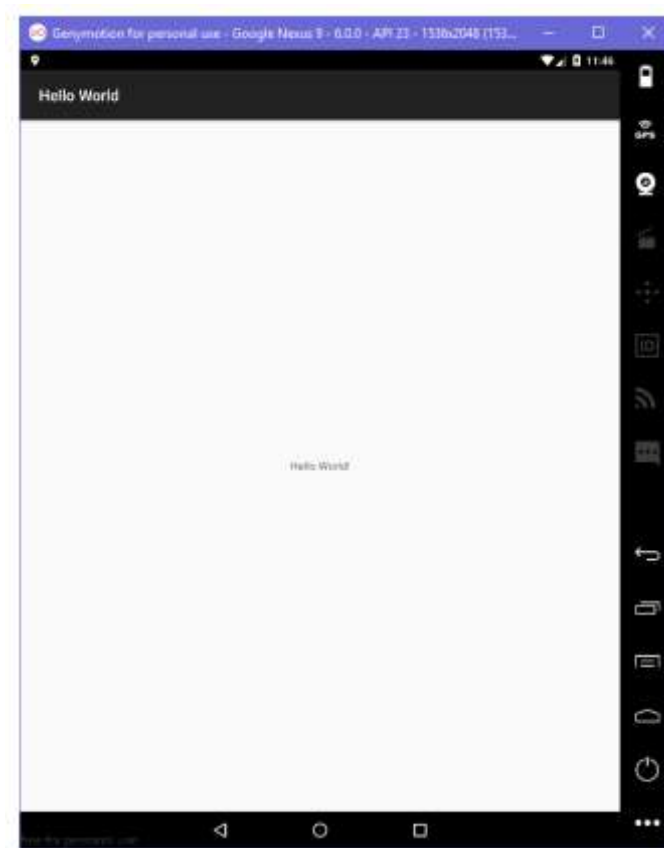
# Giao diện của Android Studio



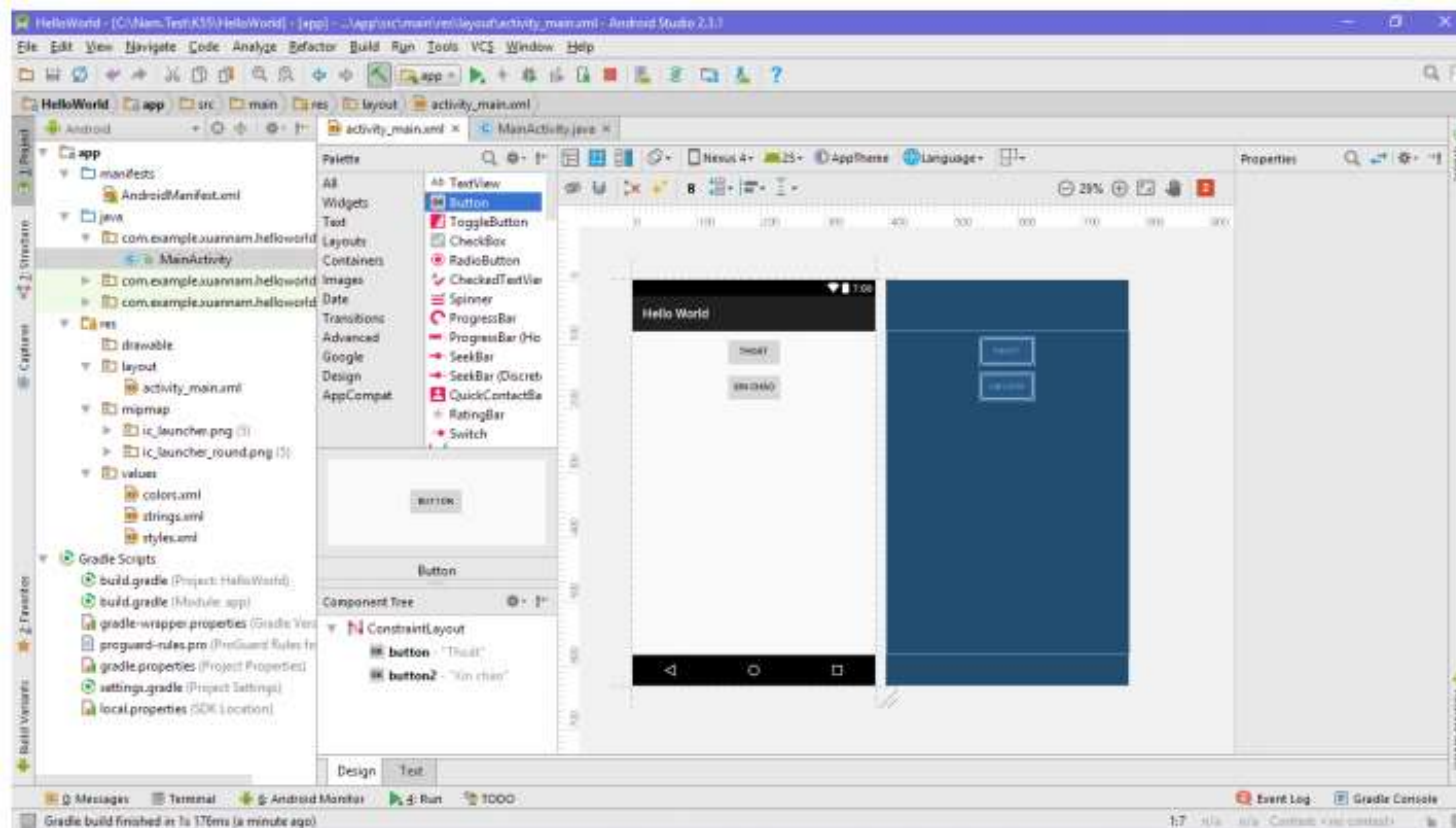
# Chọn thiết bị chạy thử



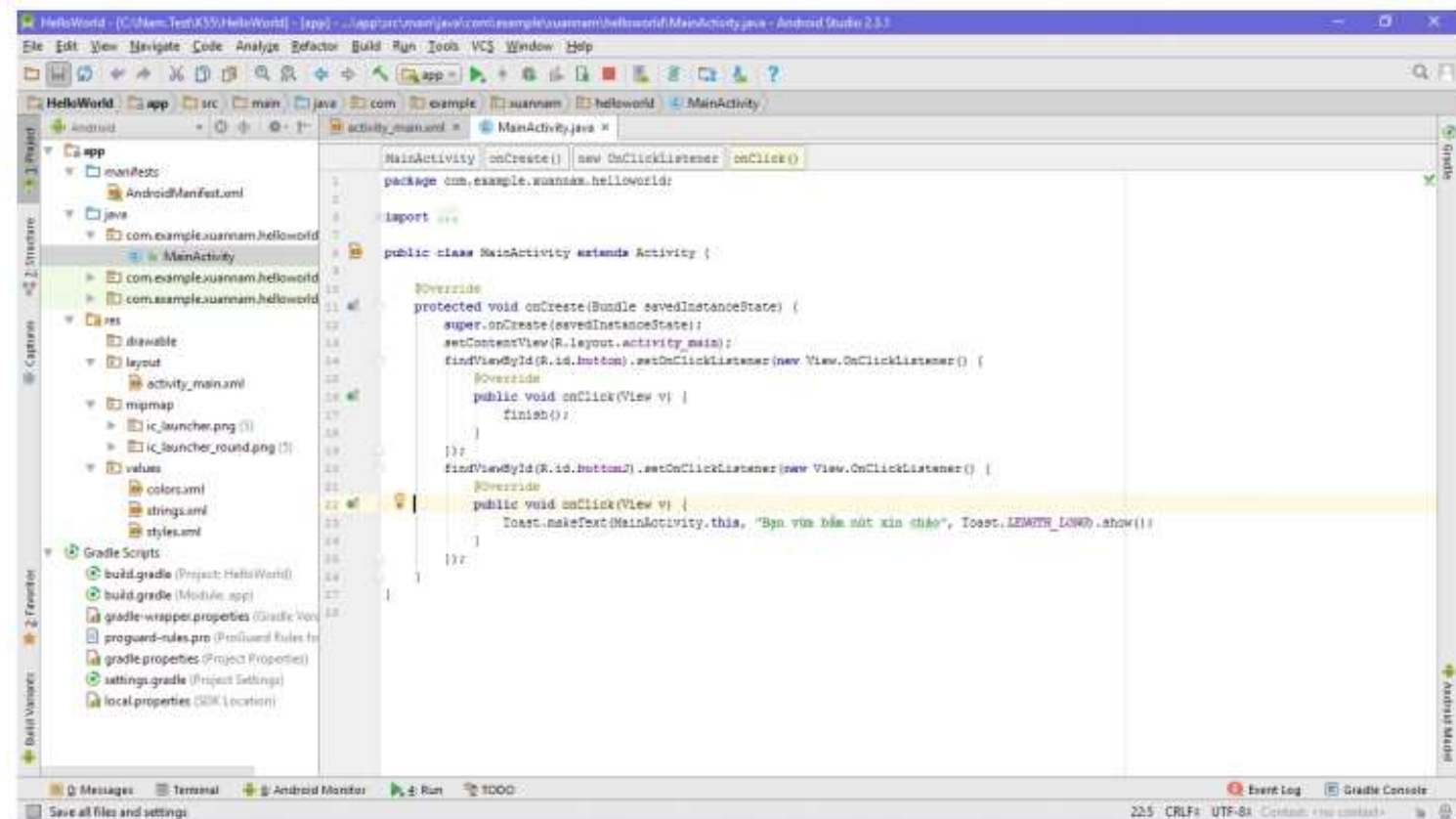
# Kết quả chạy thử



# Thêm 2 nút bấm vào giao diện



# Viết mã xử lý sự kiện





# Chạy lại chương trình mới

