**Các vấn đề hiện đại Công nghệ thông tin**

CÔNG NGHỆ PHÁT TRIỂN ỨNG DỤNG mObILE

|  |
| --- |
| Document ID: |
| Total: 26 Page No. 1 |

**Contents**

[List of Figures 5](#_Toc52465236)

[List of Tables 6](#_Toc52465237)

[1. Giới thiệu 7](#_Toc52465238)

[1.1. Giới thiệu về công nghệ phát triển mobile 7](#_Toc52465239)

[1.2. fasdfsf 7](#_Toc52465240)

[1.3. fsdffaf 7](#_Toc52465241)

[2. Native App 8](#_Toc52465242)

[2.1. Ứng dụng Native 8](#_Toc52465243)

[2.2. Lịch sử phát triển và hình thành của iOS và Android 9](#_Toc52465244)

[2.2.1. Hệ điều hành iOS 9](#_Toc52465245)

[2.2.2. Hệ điều hành Android 9](#_Toc52465246)

[2.3. Một số khác biệt về UI của iOS Native App và Android Native App 11](#_Toc52465247)

[2.4. Công cụ phát triển Native App 13](#_Toc52465248)

[2.4.1. Công nghệ được sử dụng để phát triển iOS Native App 13](#_Toc52465249)

[2.4.2. Công nghệ được sử dụng để phát triển Android Native App 13](#_Toc52465250)

[2.5. Ưu và nhược điểm của Native App 14](#_Toc52465251)

[2.5.1. Ưu điểm 14](#_Toc52465252)

[2.5.2. Nhược điểm 14](#_Toc52465253)

[3. Web App – Hybird App 15](#_Toc52465254)

[3.1. Web App (Web Application) 15](#_Toc52465255)

[3.1.1. Khái niệm 15](#_Toc52465256)

[3.1.2. Khác nhau giữa Web App và Website 15](#_Toc52465257)

[3.1.3. Lịch sử 15](#_Toc52465258)

[3.1.4. Điểm mạnh và yếu của Webapp 16](#_Toc52465259)

[3.1.5. Webapp hoạt động như thế nào 16](#_Toc52465260)

[3.1.6. Một số thuật ngữ thể hiện đặc trưng của webapp 17](#_Toc52465261)

[3.2. Hybird App 17](#_Toc52465262)

[3.2.1. Khái niệm 17](#_Toc52465263)

[3.2.2. Ưu nhược điểm của Hybird App 18](#_Toc52465264)

[3.2.3. Phát triển Hybird 19](#_Toc52465265)

[3.2.4. Khi nào nên lựa chọn phát triển Hybird App? 20](#_Toc52465266)

[4. React Native 21](#_Toc52465267)

[4.1. Định nghĩa 21](#_Toc52465268)

[4.2. Lịch sử 22](#_Toc52465269)

[4.3. Thông tin cụ thể 22](#_Toc52465270)

[4.4. Hoạt động 23](#_Toc52465271)

[4.5. Ưu và nhược điểm của React Native 24](#_Toc52465272)

[4.5.1. Ưu điểm 24](#_Toc52465273)

[4.5.2. Nhược điểm 26](#_Toc52465274)

[4.6. Cài đặt React Native 27](#_Toc52465275)

[5. Flutter 28](#_Toc52465276)

[5.1. Ngôn ngữ Dart 28](#_Toc52465277)

[5.1.1. Cài đặt Flutter và viết chương trình đầu tiên 28](#_Toc52465278)

[5.1.2. Một số khái niệm cơ bản khi sử dụng ngôn ngữ Dart 30](#_Toc52465279)

[5.2. Flutter Framework 31](#_Toc52465280)

[5.2.1. Khái niệm 31](#_Toc52465281)

[5.2.2. Đặc điểm nổi bật 31](#_Toc52465282)

[5.2.3. Nhược điểm 31](#_Toc52465283)

[5.2.4. Cài đặt Flutter 32](#_Toc52465284)

[5.2.5. Cấu trúc thư mục dự án 34](#_Toc52465285)

List of Figures

[Figure 1: Sự khác biệt giữa giao diện iOS và Android 10](#_Toc52465220)

[Figure 2: Sự khác biệt giữa giao diện iOS và Android 10](#_Toc52465221)

[Figure 3: Giao diện Gmail của iOS và Android 11](#_Toc52465222)

[Figure 4: Giao diện Instagram của iOS và Android 11](#_Toc52465223)

[Figure 5: Công nghệ phát triển iOS và Android 12](#_Toc52465224)

[Figure 6: Web App 14](#_Toc52465225)

[Figure 7: Webapp hoạt động ra sao? 16](#_Toc52465226)

[Figure 8: Hybird App 17](#_Toc52465227)

[Figure 9: Công nghệ để phát triển Hybird App 18](#_Toc52465228)

[Figure 10: Hybird App 21](#_Toc52465229)

[Figure 11: React Native icon 21](#_Toc52465230)

[Figure 12: React Native hoạt động ra sao? 22](#_Toc52465231)

[Figure 13: Ưu điểm và nhược điểm của React Native 23](#_Toc52465232)

[Figure 14: Sự đa nền tảng của React Native 24](#_Toc52465233)

[Figure 15: Cài đặt React Native 26](#_Toc52465234)

[Figure 16: Flutter 27](#_Toc52465235)

List of Tables

[Table 1: Khác nhau cơ bản của Website và Webapp 14](#_Toc52454530)

# Giới thiệu

## Giới thiệu về công nghệ phát triển mobile

## fasdfsf

## fsdffaf

# Native App

## Ứng dụng Native

Ứng dụng native (hay còn gọi là ứng dụng gốc) là một chương trình phần mềm được phát triển để sử dụng trên một nền tảng hoặc thiết bị cụ thể.

Vì một ứng dụng gốc được xây dựng để sử dụng trên một thiết bị cụ thể và hệ điều hành của nó, nên nó có khả năng sử dụng phần cứng và phần mềm dành riêng cho thiết bị. Các ứng dụng gốc có thể cung cấp hiệu suất tối ưu hóa và tận dụng công nghệ mới nhất, chẳng hạn như GPS, so với các ứng dụng web hoặc ứng dụng đám mây di động được phát triển chung trên nhiều hệ thống.

Các ứng dụng Native thường được dùng để tạo ra các ứng dụng mặc định của thiết bị như Cài đặt, Danh bạ, Album,...

Native App bao gồm iOS Native App, Android Native App, Windows Mobile App,... nhưng đa số người dùng hiện tại chỉ sử dụng 2 hệ điều hành mobile thông dụng nhất là iOS và Android. Vì vậy, phần này em chủ yếu nói về iOS Native App và Android Native App

## Lịch sử phát triển và hình thành của iOS và Android

### Hệ điều hành iOS

iOS (trước đây là iPhone OS) là hệ điều hành trên các thiết bị di động của Apple. Đây là hệ điều hành chạy trên các sản phẩm iPhone, iPad, và iPod Touch và là hệ điều hành phổ biến thứ 2 trên toàn cầu, sau Android của Google.

Ban đầu hệ điều hành này chỉ được phát triển để chạy trên iPhone (gọi là iPhone OS), nhưng sau đó được mở rộng để chạy trên các thiết bị khác của Apple như iPod Touch (tháng 9 năm 2007) và máy tính bảng iPad (tháng 1 năm 2010). Tính đến tháng 1 năm 2017, App Store trên iOS chứa khoảng 2.2 triệu ứng dụng, 1 triệu trong số đó là ứng dụng chỉ dành cho iPad và được tải về tổng cộng khoảng 130 tỷ lần. Trong quý 4 năm 2010, có khoảng 26% điện thoại thông minh chạy hệ điều hành iOS, xếp sau về thị phần so với Android của Google và Symbian của Nokia.

Giao diện người dùng của iOS thân thiện với thao tác cảm ứng đa điểm bằng tay, cũng như hỗ trợ bút stylus Apple Pencil (chỉ dành cho iPad Pro). iOS chỉ cho phép thay đổi nhạc chuông và hình nền, không hỗ trợ thay đổi giao diện. Có nhiều hãng sản xuất Android đã tùy biến giao diện Android trên thiết bị của mình giống với giao diện iOS, có thể kể đến Color OS hay MIUI.

Phiên bản mới nhất là iOS 14 và iPadOS 14 được ra mắt thử nghiệm vào tháng 6 năm 2020, hỗ trợ cập nhật cho các thiết bị từ iPhone 6s trở lên, iPod Touch (thế hệ 7), iPad (2017) trở lên, iPad Air (thế hệ thứ hai) trở lên, iPad Mini (thế hệ thứ 4) trở lên và iPad Pro.

#### **Lịch sử:**

iOS được tiết lộ tại Hội nghị và Triển lãm Macworld diễn ra vào 9 tháng 1 năm 2007 và được phát hành vào tháng 9 năm đó cùng với thế hệ iPhone đầu tiên. Khi đó, hệ điều hành này chưa có một cái tên riêng nên chỉ đơn giản là "iPhone OS". Ban đầu, ứng dụng bên thứ ba không được hỗ trợ. Steve Jobs đã tuyên bố các nhà phát triển có thể xây dựng các ứng dụng của mình chạy trên iPhone mà Apple "sẽ cư xử như những ứng dụng mặc định trên iPhone". Vào ngày 17 tháng 10 năm 2007, Apple thông báo đang phát triển bộ công cụ phát triển phần mềm cho iPhone và sẽ đưa nó đến "tay của các nhà phát triển vào tháng 2". Ngày 6 tháng 3 năm 2008, Apple đã phát hành bản dùng thử đầu tiên, cùng với một cái tên đầu tiên cho iOS, đó là "iPhone OS".

Tháng 6 năm 2010, Apple đổi tên iPhone OS thành iOS. Tuy nhiên "iOS" đã được Cisco dùng để đặt tên cho hệ điều hành của mình trước đó. Để tránh các tranh chấp bản quyền trong tương lai, Apple đã xin giấy phép sử dụng thương hiệu iOS từ Cisco, điều này giống với cách Apple đã mua thương hiệu "iPhone" để sử dụng cho điện thoại thông minh của mình.

Trang web chính thức: http://www.apple.com/ios

iOS Dev Center: http://developer.apple.com/ios

### Hệ điều hành Android

Android là một hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Android, Inc. với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005.

Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào năm 2008.

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

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

Android chiếm 87,7% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 2 năm 2017, với tổng cộng 2 tỷ thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Sự thành công của hệ điều hành cũng khiến nó trở thành mục tiêu trong các vụ kiện liên quan đến bằng phát minh, góp mặt trong cái gọi là "cuộc chiến điện thoại thông minh" giữa các công ty công nghệ.

#### **Lịch sử:**

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

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

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

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

Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng; ví dụ như phiên bản 1.5 Cupcake (bánh bông lan nhỏ có kem) tiếp nối bằng phiên bản 1.6. Phiên bản mới nhất hiện nay là 11 Android R ra mắt vào tháng 2 năm 2020. Vào năm 2010, Google ra mắt loạt thiết bị Nexus—một dòng sản phẩm bao gồm điện thoại thông minh và máy tính bảng chạy hệ điều hành Android, do các đối tác phần cứng sản xuất. HTC đã hợp tác với Google trong chiếc điện thoại thông minh Nexus đầu tiên, Nexus One. Kể từ đó nhiều thiết bị mới hơn đã gia nhập vào dòng sản phẩm này, như điện thoại Nexus 4 và máy tính bảng Nexus 10, lần lượt do LG và Samsung sản xuất. Google xem điện thoại và máy tính bảng Nexus là những thiết bị Android chủ lực của mình, với những tính năng phần cứng và phần mềm mới nhất của Android.

Trang web chính thức: http://www.android.com/

Android Developers: https://developer.android.com/

## Một số khác biệt về UI của iOS Native App và Android Native App

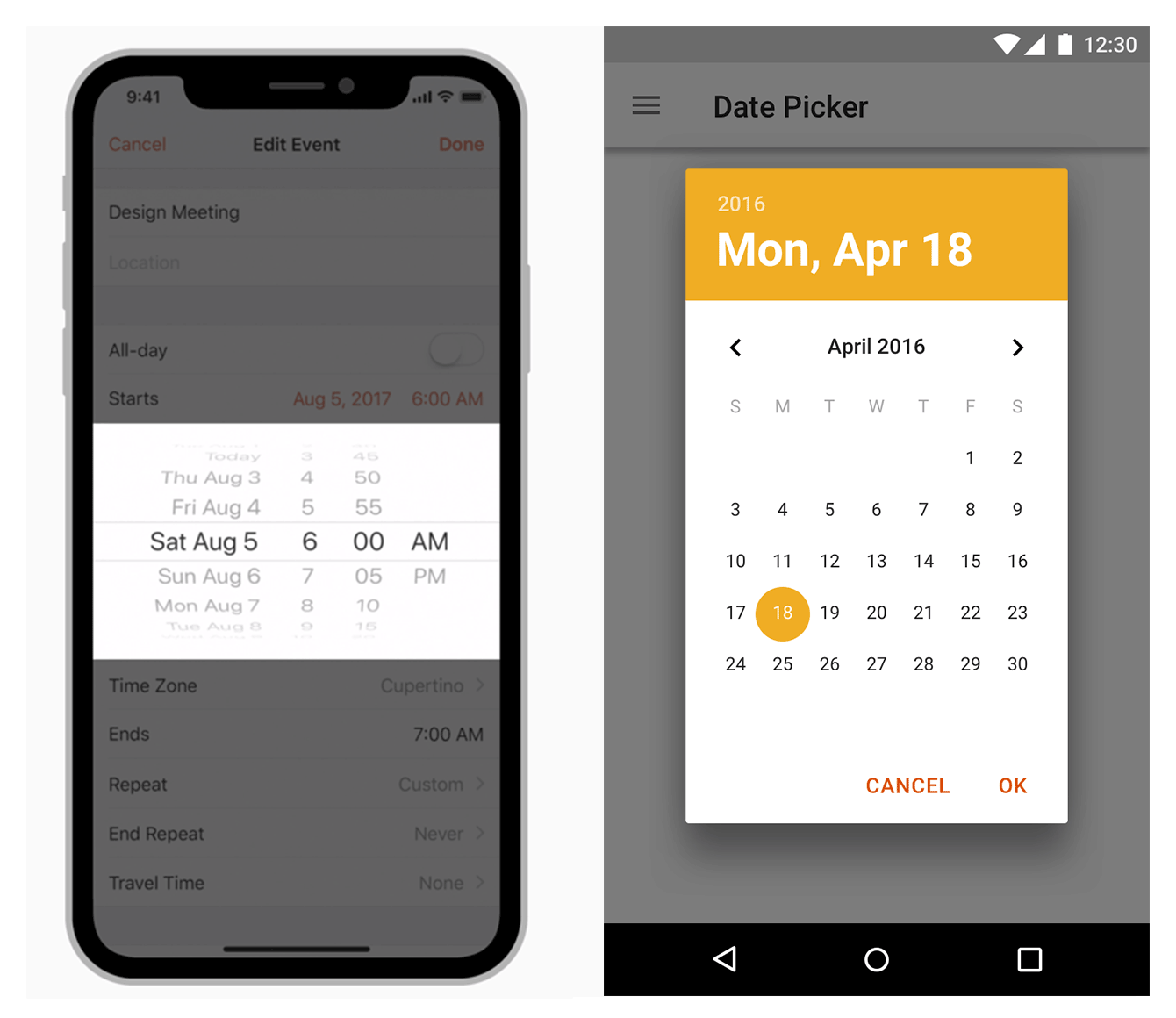


Figure 1: Sự khác biệt giữa giao diện iOS và Android

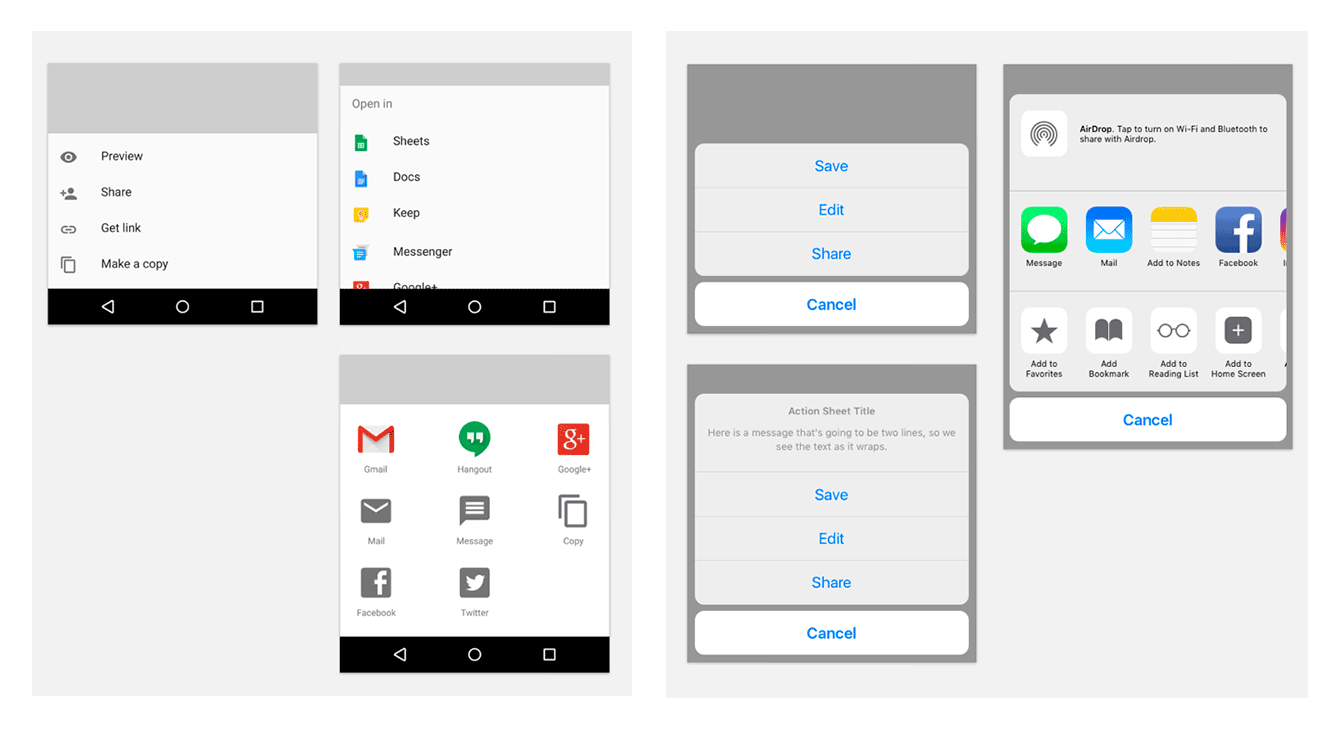


Figure 2: Sự khác biệt giữa giao diện iOS và Android

Qua một số hình trên ta có thể nhận thấy một số sự khác biệt về giao diện:

• Sự khác nhau của thanh điều hướng: kiểu dáng và vị trí.

• Sự khác nhau của các mẫu điều hướng: Android sử dụng Menu và Tabs để đièu hướng trong khi iOS chủ yếu chỉ sử dụng Tabs để điều hướng.

• Tuỳ chỉnh chế độ xem cho các điều khiển tiêu chuẩn: Ví dụ như công cụ chọn ngày của iOS theo kiểu cuộn thì Android trông giống 1 cuốn lịch.

• Thiết kế button,...

Tất nhiên là vẫn có một số ngoại lệ: một số ứng dụng iOS cũng tuân theo hướng dẫn Material Design (như Gmail) và một số ứng dụng Android tuân theo HIG (như Instagram).

**Chú thích:**

• Hướng dẫn Material Design: nguyên tắc thiết kế và quy ước chung của úng dụng Android.

• HGI: Human Interface Guidelines - hướng dẫn giao diện người dùng của Apple.

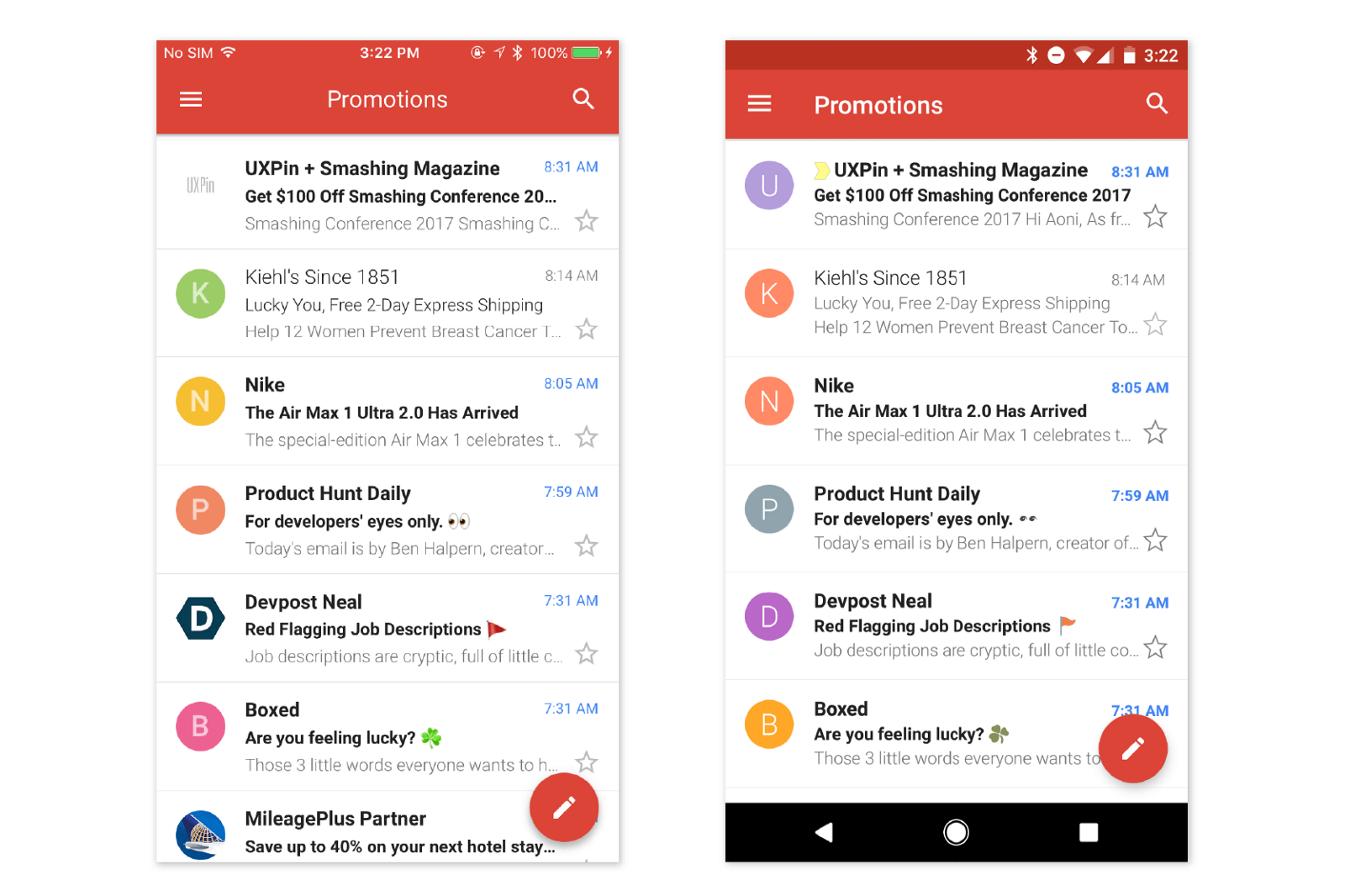


Figure 3: Giao diện Gmail của iOS và Android

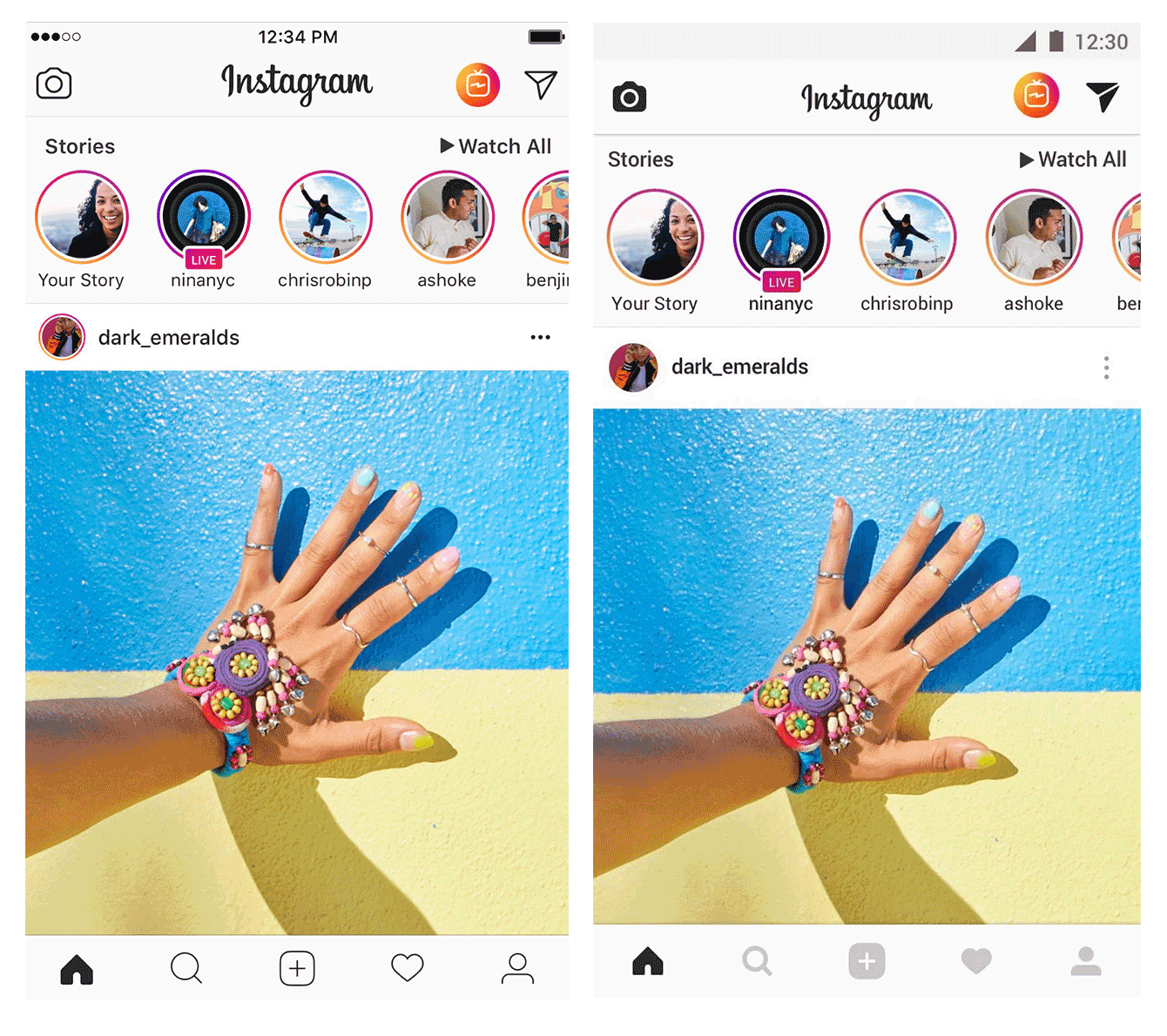


Figure 4: Giao diện Instagram của iOS và Android

## Công cụ phát triển Native App



Figure 5: Công nghệ phát triển iOS và Android

Native app được xây dựng riêng cho chỉ một nền tảng, do đó các công nghệ được áp dụng sẽ khác nhau. Để xử lý việc phát triển Native app của ứng dụng mobile Android, các kỹ sư có thể áp dụng Java hoặc Kotlin, Swift hoặc Objective-C sẽ cần cho phát triển iOS Native app. Các công ty thường chọn phương pháp này có sự cần thiết phải xây dựng hai ứng dụng cùng một lúc. Rõ ràng là quá trình kỹ thuật dài hơn và đòi hỏi nhiều nỗ lực hơn. Nhưng kết quả cuối cùng sẽ cân bằng nó. Các khách hàng có thể có một trải nghiệm nhanh nhạy hơn và nhanh hơn. Performance tốt hơn là lợi thế cho các ứng dụng mobile phức tạp xử lý hoạt hình và kết xuất. Hơn nữa, bảo mật dữ liệu là một tính năng có lợi khác.

Các ứng dụng gốc hoạt động với HĐH của thiết bị cho phép chúng hoạt động nhanh hơn và linh hoạt hơn các loại ứng dụng thay thế khác. Nếu ứng dụng được bán cho người dùng thuộc nhiều loại thiết bị khác nhau, nhà phát triển sẽ phải tạo một phiên bản ứng dụng riêng cho từng loại.

Ví dụ: Ứng dụng Facebook đã từng được viết bằng HTML5 để sử dụng cùng một mã cho iOS, Android và web di động. Tuy nhiên, ứng dụng chậm hơn đối với người dùng iOS, dẫn đến các nhà phát triển ứng dụng của Facebook phải tạo mã riêng cho iOS.

### Công nghệ được sử dụng để phát triển iOS Native App

#### **Objective-C:**

Ngôn ngữ lập trình này được biết đến với trải nghiệm phát triển tuyệt vời, thư viện có sẵn và một nhóm chuyên gia rộng lớn. Objective-C là một mature framework, cũng có được khả năng tương thích với các công nghệ lập trình khác. Bất chấp những lợi ích đó, chúng ta nhận ra rằng các ngôn ngữ mới xuất hiện và đôi khi nó sẽ không có khả năng hỗ trợ tất cả các tính năng mới nhất.

#### **Swift:**

Đây là một framework lập trình tương đối mới được Apple giới thiệu đã trở thành lựa chọn thay thế để xây dựng các iOS Native app. Các nhà phát triển đã chỉ ra performance nhanh hơn của nó và nó dễ dàng hơn để tìm hiểu và làm việc với. Nó liên tục phát triển đó là lý do tại sao có thể thiếu một số thành phần. Tuy nhiên, người ta nói rằng có thể thay thế Objective-C trong tương lai.

### Công nghệ được sử dụng để phát triển Android Native App

#### **Java:**

Ngôn ngữ lập trình không chỉ chủ yếu được sử dụng cho các ứng dụng mobile Android mà còn cho các mục đích khác. Phần lớn phát triển desktop và web dựa trên Java. Nó đã trở thành một hệ thống công cụ đặc biệt, các nhà phát triển có quyền truy cập vào một thư viện mạnh mẽ. Nó giúp đơn giản hóa quá trình lập trình tổng thể. Tuy nhiên, các ứng dụng trên Java đòi hỏi nhiều bộ nhớ hơn và hoạt động chậm hơn so với các framework khác.

#### **Kotlin:**

Nó là một ngôn ngữ lập trình được phát triển đặc biệt để làm việc với Java và Java Virtual Machine. Do đó, việc sử dụng nó được Google hỗ trợ và phê duyệt để phát triển các ứng dụng Android. Một trong những ưu điểm chính của Kotlin so với Java thông thường là giao diện của nó cho phép làm việc theo cú pháp ngắn hơn. Thực tế này giúp giảm thời gian lập trình cần thiết để tạo một ứng dụng cho Android. Bây giờ nó được bao gồm như là một thay thế cho trình biên dịch Java tiêu chuẩn cho Android Studio. Expedia, Square, Pinterest và Flipboard có thể được nêu tên trong số những ví dụ điển hình nhất về các công ty sử dụng Kotlin cho các ứng dụng Android của họ.

## Ưu và nhược điểm của Native App

### Ưu điểm

• Điểm mạnh đầu tiên và cũng là lợi thế lớn nhất của native app là việc cho phép người dùng truy cập, sử dụng tất cả tính năng một cách dễ dàng; đồng thời cũng rất ít khi gặp lỗi nếu được thiết kế một cách hoàn chỉnh. Các tính năng native app truy cập gồm phần cứng như camera, GPS, thiết bị thu âm,…; các quyền cá nhân như sao chép, tạo, ghi, đọc tập tin trên bộ nhớ của điện thoại, thực hiện cuộc gọi, nhắn tin,…

• Trải nghiệm người dùng: hầu hết người dùng ít quan tâm đến công nghệ, họ chỉ cần biết nếu dùng có thao tác dễ dàng và mượt mà hay không. Nghiên cứu được thực hiện bởi Equation Reach: 79% người dùng sẽ chỉ thử lại một ứng dụng một hoặc hai lần nếu nó không hoạt động trong lần đầu tiên. Và Native app đã chứng minh được khả năng chiều lòng người dùng tốt hơn những người anh em khác.

• Native app phù hợp với các sản phẩm yêu cầu hiệu năng cao và có các tính năng tùy chỉnh tương thích tốt với nền tảng.

• Data protection / Bảo mật dữ liệu – Nó dễ dàng hơn nhiều để làm cho Native app an toàn. Đó là lợi thế mà rất nhiều công ty quan tâm trong việc cung cấp cho khách hàng của họ đặc biệt là trong lĩnh vực doanh nghiệp, fintech và các ứng dụng có dữ liệu nhạy cảm.

• Overall functionality / Chức năng tổng thể – Ứng dụng sẽ có khả năng quan trọng để kết nối các tính năng phần cứng của thiết bị và các cơ sở dữ liệu khác nhau. Không cần plugin hay công cụ bổ sung.

• Bản build nhẹ hơn – Dung lượng của bản build gọn nhẹ, ít tốn bộ nhớ của device

### Nhược điểm

• Time and money-consuming / Tốn thời gian và tiền bạc – Chắc chắn đòi hỏi thời gian để xây dựng phần mềm phức tạp. Phân phối người dùng trên hai platforms chính nên tăng gấp đôi số lượng công việc và testing cần thiết để duy trì hai ứng dụng riêng biệt cho cả iOS và Android hoạt động.

• Distributed codebase / Codebase phân tán – Có một số tính năng không có sẵn cho iOS hoặc Android là một thực tế lớn có ngay cả trong năm 2019. Điều này xảy ra do những hạn chế về ngân sách hoặc hạn chế của platform. Đôi khi, ứng dụng trong App Store có thể bị hủy bỏ trong nhiều năm trong khi phiên bản Android nhận được cập nhật thường xuyên và ngược lại. Ngược lại, Hybrid khuyến khích các nhà phát triển tiếp cận UI và các tính năng một cách chu đáo hơn và chỉ giới thiệu các tính năng có thể hoạt động trên cả hai hệ điều hành.

# Web App – Hybird App

## Web App (Web Application)

### Khái niệm

Đầu tiên ta cần hiểu khái niệm Application là gì? Application là "Ứng dụng là một loại chương trình có khả năng làm cho máy tính thực hiện trực tiếp một công việc nào đó người dùng muốn thực hiện".

Ban đầu, các website chỉ bao gồm text, hình ảnh và video, liên kết với nhau thông qua các link. Tác dụng của website là lưu trữ và hiển thị thông tin. Người dùng chỉ có thể đọc, xem, click các link để di chuyển giữa các page.

Về sau, với sự ra đời của các ngôn ngữ server: CGI, Perl, PHP, … các website đã trở nên “động” hơn, có thể tương tác với người dùng. Từ đây, người dùng có thể dùng web để “thực hiện một công việc nào đó bằng máy tính“, do đó web app web app có thể được thiết kế chạy trên nền tảng web hoặc các trình duyệt của thiết bị di động.

****

Figure 6: Web App

Nói dễ hiểu, web app là những ứng dụng chạy trên web. Thông qua web app, người dùng có thể thực hiện một số công việc: tính toán, chia sẻ hình ảnh, mua sắm … Tính tương tác của web app cao hơn website rất nhiều.

Với một số người không rành về IT, tất cả những thứ online, vào được bằng trình duyệt đều là website cả. Do đó họ thường yêu cầu bạn là: website quản lý siêu thị, website bán hàng, … thực chất chúng đều là webapp.

### Khác nhau giữa Web App và Website

Trên thực tế, ranh giới giữa web app và website có vẻ hơi mong manh. Một trang báo mạng – Vnexpress chẳng hạn, trong mắt người đọc nó là website. Nhưng trong mắt biên tập viên hoặc admin, nó lại là web app. Một số trang web cho phép người dùng search, comment nhưng nó vẫn chỉ là website, chưa phải là webapp. Dưới đây là bảng so sánh (tương đối).

Table 1: Khác nhau cơ bản của Website và Webapp

|  |  |
| --- | --- |
| Website | Webapp |
| Tính tương tác thấp, ít chức năng (Xem, đọc, click qua lại giữa các link…) | Tính tương tác cao, nhiều chức năng (Đăng thông tin, upload file, xuất báo cáo…) |
| Được tạo thành từ các trang html tĩnh và một số tài nguyên (hình ảnh, âm thanh, video) | Được tạo bởi html và code ở backend (PHP, C#, Java, …) |
| Được dùng để lưu trữ, hiển thị thông tin | Được dùng để “thực hiện một công việc”, thực hiện các chức năng của một ứng dụng |

### Lịch sử

Trong dạng tính toán chủ-khách trước đây, mỗi ứng dụng có chương trình khách riêng của nó sẽ phục vụ như giao diện người dùng và phải được cài đặt riêng rẽ trên mỗi máy tính cá nhân của người dùng. Sự nâng cấp phần máy chủ của ứng dụng sẽ cần nâng cấp tất cả máy khách đã được cài trên mỗi máy trạm người dùng, thêm vào đó là chi phí hỗ trợ và giảm năng suất.

Ngược lại, ứng dụng web linh hoạt tạo ra một loạt các tài liệu Web ở định dạng chuẩn được hỗ trợ bởi những trình duyệt phổ biến như HTML/XHTML. Ngôn ngữ kịch bản phía người dùng ở dạng ngôn ngữ chuẩn như JavaScript thường được thêm vào để có thêm những yếu tố động trong giao diện người dùng. Nói chung, mỗi trang Web đơn lẻ được gửi tới người dùng như một tài liệu ổn định, nhưng thứ tự các trang có thể cung cấp cảm giác trực quan, khi những gì người dùng nhập vào sẽ được trả về thông qua thành phần mẫu Web được nhúng vào trong đánh dấu trang. Trong quá trình giao dịch đó, trình duyệt Web sẽ thông dịch và hiển thị trang, và hoạt động như một người dùng chung cho bất kỳ ứng dụng Web nào.

### Điểm mạnh và yếu của Webapp

#### **Điểm mạnh:**

* Cross platform: Có thể chạy trên tất cả trình duyệt của mobile hỗ trợ phiên bản HTML và javascript.
* Không cần cài đặt trên máy.
* Thuận lợi cho các nhà phát triển:
  + Với một phiên bản duy nhất cho tất cả, nên giảm chi phí và thời gian cho phát triển, bảo trì, cũng như nâng cấp sau này.
  + Ngôn ngữ lập trình phổ dụng là HTML và Javascript mà hầu hết các lập trình viên đều biết.
  + Không giới hạn môi trường lập trình, lập trình viên có thể làm việc trên hầu hết các HDH.
  + Ứng dụng không cần phải được build lại qua SDK hoặc một công cụ phát triển độc quyền khác như xCode (chỉ cần deploy lên server và chạy).
  + Việc cập nhật phiên bản sẽ đến ngay lập tức và trong suốt với người dùng (ngay sau khi deploy) mà không cần thông qua thủ tục và chờ đợi approval như các chợ ứng dụng.
* Có thể được đánh dấu bởi search engines (tốt cho SEO).
* Dễ dàng quảng bá: người dùng có thể tìm thấy thông qua search engine, share link từ email, từ social networks hoặc từ các dịch vụ quảng cáo trực tuyến như Google Ads

#### **Điểm yếu:**

* Mặc dù HTML5 có tính tương thích rộng lớn nhưng cùng với đó là cả điểm yếu. Nó phụ thuộc vào trình duyệt, nền tảng, loại máy. Đặc biệt khi mức độ phân mảnh thị trường của các thiết bị di động lớn như hiện nay lại càng làm các nhà lập trình khó khăn trong việc lựa chọn đúng phần nào của HTML5 có thể sử dụng.
* Về performance, web-based app không chạy nhanh như native app, và không mang lại trải nghiệm trọn vẹn mà nền tảng đó có thể cung cấp, đây cũng là lý do mà hầu hết các mobile game đều không dùng web app.
* Luôn phải chạy online. Khi mạng chập chờn thì ứng dụng web ngay lập tức thể hiện điểm yếu của mình.

### Webapp hoạt động như thế nào

Các ứng dụng web thường được mã hóa bằng ngôn ngữ được trình duyệt hỗ trợ như JavaScript và HTML vì các ngôn ngữ này dựa trên trình duyệt để render chương trình thực thi. Có một số ứng dụng động yêu cầu quá trình xử lý phía máy chủ, còn lại các ứng dụng tĩnh sẽ hoàn toàn không cần xử lý ở phía máy chủ.

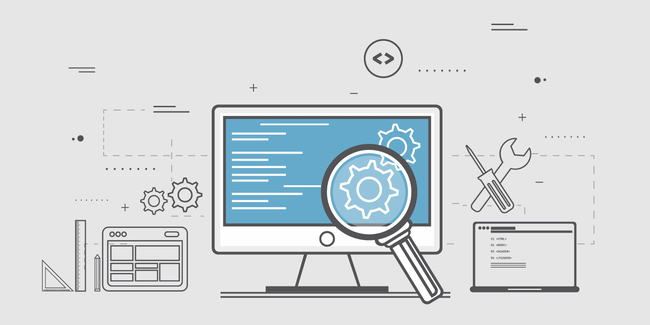


Figure 7: Webapp hoạt động ra sao?

Ứng dụng web yêu cầu một web server để quản lý các yêu cầu từ máy khách, một application server để thực hiện các tác vụ được yêu cầu và đôi khi, một database để lưu trữ thông tin. Công nghệ application server có các loại từ ASP.NET, ASP và ColdFusion, đến PHP và JSP.

#### **Flow hoạt động của webapp:**

* Người dùng kích hoạt request tới web server qua Internet, thông qua trình duyệt web hoặc giao diện người dùng của ứng dụng.
* Web server chuyển tiếp request này đến web application server thích hợp.
* Máy chủ ứng dụng Web (web application server) thực hiện nhiệm vụ được yêu cầu - chẳng hạn như truy vấn cơ sở dữ liệu hoặc xử lý dữ liệu - sau đó tạo ra các kết quả của dữ liệu được yêu cầu.
* Máy chủ ứng dụng web gửi kết quả đến máy chủ web với thông tin được yêu cầu hoặc dữ liệu đã được xử lý.
* Máy chủ web phản hồi response lại cho khách hàng các thông tin được yêu cầu sau đó xuất hiện trên màn hình của người dùng.

### Một số thuật ngữ thể hiện đặc trưng của webapp

* Responsive: Giao diện tương thích với kích thước của màn hình thiết bị.
* App-like feel: Trải nghiệm Web gần giống với trải nghiệm App.
* Offline support: sử dụng bộ nhớ của thiết bị để cung cấp trải nghiệm offline.
* Re-engaging: Đẩy thông báo (notification), giúp người dùng khám phá lại ứng dụng một khi được cài đặt.
* Discoverable: Các công cụ tìm kiếm và việc tối ưu hoá SEO có thể mang nhiều người dùng đến với ứng dụng của bạn hơn.
* Fresh: Ứng dụng tự động cập nhật một khi online.
* Safe: Ứng dụng sử dụng HTTPS.
* Progressive: Ứng dụng hoạt động trên mọi thiết bị thậm chí cả các thiết bị cũ, mặc dù có thể chúng có ít tính năng hơn.
* Linkable: Dễ dàng để chia sẻ, kết nối đến tới ứng dụng chỉ với đường dẫn URL.

## Hybird App

### Khái niệm

Đây là ứng dụng lai. Ứng dụng này là sự kết hợp hoàn hảo giữa những ưu điểm của Native App và Mobile Web. Ứng dụng này cũng giống với hầu hết những ứng dụng khác dành cho điện thoại di động.

Chúng được cài đặt trên điện thoại để bạn có thể chụp ảnh, kết nối với bạn bè thông qua các mạng xã hội, chơi game giải trí hay nhiều chức năng hữu ích khác.

Hybrid là ứng dụng được xây dựng dựa trên các công nghệ Web phổ biến như Javascript, HTML hay CSS. Điểm khác biệt của Hybrid là ứng dụng này được lưu trong ứng dụng native sử dụng cửa sổ trình duyệt Web View.

Điểm khác biệt này giúp cho Hybrid có thể truy cập vào hầu hết các chức năng thuộc phần cứng của điện thoại di động bao gồm máy ảnh, danh bạ, cảm biến gia tốc, âm thanh… Đây là một lợi thế của Hybrid so với các ứng dụng khác khi không bị hạn chế khi truy cập vào các trình duyệt trên điện thoại di động.



Figure 8: Hybird App

### Ưu nhược điểm của Hybird App

#### **Ưu điểm:**

* Single code base – Đây là lợi thế đem đến tất cả các lợi ích sau. Các ứng dụng này được các công ty và nhà phát triển ưa thích vì chúng có thể hoạt động trên cả hai nền tảng. Sẽ không cần building hai code riêng cho iOS và Android vì lý do code common chức năng.
* Lower cost / Chi phí thấp - Công ty có thể tiết kiệm một chi phí rất lớn bằng cách phát triển một ứng dụng mobile trong khi giải quyết vấn đề người dùng trên các nền tảng khác nhau. Bạn có nhu cầu thuê một nhóm có chuyên môn về cả hai nhưng chỉ cần chuyên môn tốt về phát triển web là đủ. Chi phí của bạn sẽ gần giống như chi phí xây dựng chỉ cho một nhưng được thiết kế để chạy ở mọi nơi.
* Simpler to build and test / Đơn giản để build và test – Team sẽ đạt được kết quả mong đợi nhanh hơn. Họ không thỏa thuận với từng nền tảng. Code được tạo một lần và thời gian test cũng sẽ được giảm.
* Easier to maintain / Dễ dàng bảo trì – Tất cả các thay đổi và cập nhật bắt buộc sẽ được duy trì đồng thời trên cả hai nền tảng. Nó không chỉ thuận tiện cho các nhà phát triển cho người dùng là tốt. Rất nhiều vấn đề có thể khắc phục từ phía máy chủ và người dùng sẽ chỉ nhận được các bản cập nhật tự động.
* Faster delivery time / Thời gian delivery nhanh hơn – Như đã đề cập ở trên, bạn không cần phải có hai nhóm iOS và Android hoặc nhóm đa chức năng lớn, bạn chỉ cần tìm một nhóm chuyên gia tương đối nhỏ. Họ có thể tạo ra một sản phẩm được tiếp thị thông minh sẽ thú vị cho những người dùng khác nhau. Các Hybrid app đã tự đề xuất là định hướng nội dung.

#### **Nhược điểm:**

* Bounded efficiency / Giới hạn hiệu quả – Cross-platform framework phụ thuộc vào các plugin được kết nối với các tính năng của thiết bị. Đôi khi các nhà phát triển phải tạo chúng bằng tay để tiếp cận chức năng cụ thể của thiết bị.
* Internet connection – Đối với hiệu quả của Native app và Hybrid app, có một sự khác biệt đáng kể. Các cross-platform framework yêu cầu kết nối internet thường xuyên. Một số chức năng sẽ không có sẵn khi người dùng thiếu kết nối internet.
* Bản build dung lượng lớn – Dung lượng của bản build khá lớn, tốn bộ nhớ của device nhieufehow so với Native app.
* Mặc nhiên, kiểu lập trình này vẫn còn tồn tại một vài hạn chế, nhất là độ mượt mà không bằng Native App và lập trình viên sẽ rất khó để tùy biến các tính năng UI/ UX trên ứng dụng.

### Phát triển Hybird

Khi đưa ra quyết định tạo Hybrid app, nên tìm hiểu quy trình phát triển của nó, kiểm tra tất cả các tính năng và tìm ra điểm mạnh và điểm yếu. Ứng dụng này ban đầu được coi là một trang web được đưa vào các container. Do đó, mỗi page phải sử dụng các API được thiết kế đặc biệt để đạt được các tính năng cơ bản cũng như thu hút các phương tiện của bên thứ ba cho nó. Các ứng dụng này vận hành kiểm soát chế độ webview để nhận tệp HTML và JavaScript toàn màn hình tận dụng công cụ kết xuất của các trình duyệt trong hệ điều hành tích hợp.

#### **Công nghệ được sử dụng để phát triển Hybird App:**

Như đã đề cập, các công ty phát triển mobile đang làm việc với sự kết hợp của các công nghệ. Có ba khung phổ biến nhất để phát triển ứng dụng lai:

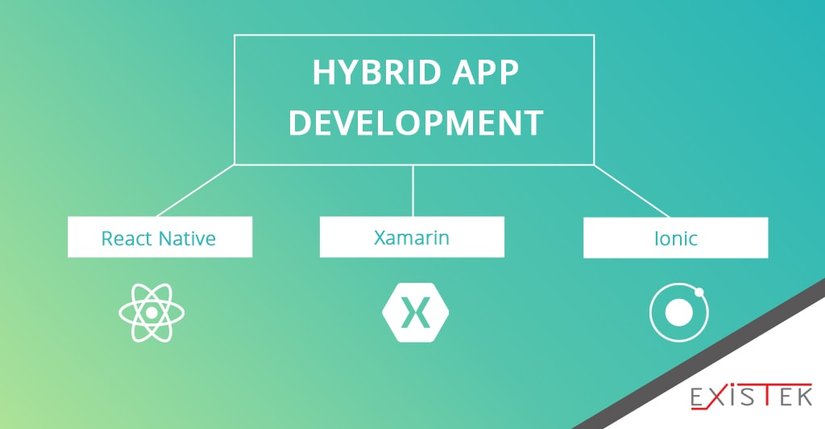


Figure 9: Công nghệ để phát triển Hybird App

* React Native : Facebook đã tạo ra công nghệ open-sourced này để tương thích đa nền tảng. Các UI được đơn giản hóa có hiệu suất tăng đáng kể khi reloading. React Native được ưa chuộng sử dụng vì thời gian phát triển ngắn. Các kỹ sư phần mềm có quyền truy cập vào các đơn vị có sẵn để áp dụng, tuy nhiên, framework có thể thiếu các thành phần nhất định. Cả Facebook và cộng đồng thường xuyên cung cấp thông tin cập nhật.
* Xamarin : Open framework được Microsoft hỗ trợ và duy trì nâng cấp mạnh mẽ. Gần đây, họ đã đưa ra cách tiếp cận phát triển Hybrid mobile app mới cho phép code cross platform. Nó đã giúp cải thiện tốc độ phát triển và đơn giản hóa bảo trì. Các nhà phát triển có thể sử dụng logic chung, tuy nhiên, các quy định UI sẽ duy trì cụ thể cho từng nền tảng. Xamarin sẽ không xử lý đồ họa phức tạp nhưng phù hợp với các ứng dụng đơn giản. Công nghệ này thường được chọn cho các dự án định hướng kinh doanh.
* Ionic : Framework này tạo ra các ứng dụng mobile với các công nghệ web tiêu chuẩn như JavaScript, CSS, HTML, Angular,... Các nhà phát triển cũng có rất nhiều UIcomponent có thể truy cập đơn giản để xử lý. Thời gian phát triển là tuyệt vời như với tất cả các khung khác cho các ví dụ Hybrid app. Mặc dù có tất cả các lợi ích, việc bảo trì có thể khó khăn hơn. Ionic cung cấp rất nhiều plugin, vì vậy khi sự cố xuất hiện, nó có thể yêu cầu điều chỉnh thủ công thêm.

### Khi nào nên lựa chọn phát triển Hybird App?

Khi chúng ta đã kiểm tra tất cả các ưu và nhược điểm của Native app so với Hybrid app, bạn có thể nhận thấy sự khác biệt cốt lõi giữa phát triển Native app và Hybrid app nằm ở cách tiếp cận triển khai. Native app là hai phần mềm hoàn toàn khác nhau được thiết kế cho nền tảng Android hoặc iOS, trong khi các Hybrid app là cùng một phần mềm hoạt động trên cả hai platform với các tinh chỉnh nhỏ. Sự khác biệt đáng kể này đi trước danh sách các tiêu chí sau.

Quan sát performance của Hybrid app so với Native app, nhiều nhà phát triển đã thừa nhận rằng các Native mobile app được chọn vì performance tốt hơn. Tuy nhiên, các Hybrid app (cross-platform framework) không ngừng cải tiến để theo kịp.

Đối với bảo trì, có thể dễ dàng quản lý một ứng dụng mobile hơn là hai ứng dụng cùng một lúc. Theo các phân tích, so sánh của Native app so với Hybrid app, Hybrid app chắc chắn dễ dàng hơn nhiều để hỗ trợ và duy trì.

Cuối cùng, tất cả các công ty đều được định hướng cho trải nghiệm người dùng tốt nhất. Hầu hết thời gian, khách hàng sẽ không nhận thấy sự khác biệt giữa hiệu suất Hybrid app và Native app. Cả hai biến thể đều không ngừng cải tiến với các tính năng và công nghệ mới. Các ví dụ Native app có thể thực hiện nhanh hơn một chút vì ban đầu chúng được tạo cho một platform nhất định. Tuy nhiên, một cross-platform framework thường được chọn vì tính linh hoạt của nó.

# React Native



## Định nghĩa

React Native là một framework mã nguồn mở được sáng tạo bởi Facebook. Nó được sử dụng để phát triển ứng dụng di động Android, iOS, Web và UWP bởi các nhà phát triển sử dụng React cùng với môi trường ứng dụng gốc (native). (Wikipedia)

Nói cách khác, React Native là một framework giúp lập trình viên xây dựng các ứng dụng đa nền tảng mà chỉ sử dụng ngôn ngữ lập trình Javascript, nó sinh ra nhằm mục đích giải quyết bài toán hiệu năng Hybird và bài toán chi phí khi mà phải viết nhiều loại ngôn ngữ native cho từng nền tảng di động.

Chúng ta sẽ build được ứng dụng Native, và chúng ta cũng có thể build ứng dụng đó một cách đa nền tảng (multi-platform) chứ không phải một “mobile web app”, không phải là “HTML5 app”, và cũng không phải là một “hybrid app” hay cũng không chỉ build trên iOS hay Android mà chúng ta build và chạy được cả hai hệ sinh thái.

\* ***Giải thích một số định nghĩa:***

* UWP: Universal Windows Platform (UWP) là một cấu trúc ứng dụng thống nhất giữa các nền tảng được tạo bởi Microsoft và được giới thiệu lần đầu trong Windows 10. Mục đích của nền tảng phần mềm này là giúp phát triển các ứng dụng kiểu Metro chạy trên cả Windows 10 và Windows 10 Mobile mà không cần phải viết lại ứng dụng cho mỗi nền tảng. Nó hỗ trợ cho việc phát triển các ứng dụng Windows bằng C++, C#, VB.NET, hoặc XAML. API được thực hiện bằng C++, và được hỗ trợ trong C++, VB.NET, C#, và JavaScript. Được thiết kế như một phần mở rộng của nền tảng Windows Runtime được giới thiệu lần đầu trong Windows Server 2012 và Windows 8, UWP cho phép nhà phát triển tạo các ứng dụng có thể chạy trên nhiều loại thiết bị.
* Native App: Native App là tên gọi dành cho những ứng dụng được xây dựng và phát triển bằng những công cụ do chính nhà phát triển cung cấp cho lập trình viên. Hiện nay, trên thế giới có hai nhà phát triển lớn tồn tại song song và cạnh tranh lẫn nhau đó là Android và iOS.

Native App được tạo ra bằng ngôn ngữ của hệ điều hành và các tính năng có sẵn trên hệ điều hành đó. Điều này cho phép các ứng dụng vận hành với tốc độ cao nhất mà không cần thông qua bất kì ứng dụng bên thứ ba hay engine nào.

Các ứng dụng Android có thể được viết bằng hai ngôn ngữ là Java và Kotlin. Trong khi đó, iOS app được viết bằng ngôn ngữ hướng đối tượng Swift, trước đây là Objective C. Công cụ hỗ trợ phát triển ứng dụng cho hệ điều hành Android và iOs có thể tham khảo là Android Studio và XCode.

* Hybird App: Hybrid App là sự kết hợp giữa ứng dụng Web và ứng dụng mobile. Tức là chúng vừa có thể cài đặt lên điện thoại người dùng giống như là những ứng dụng Native bình thường và bạn có thể tìm thấy chúng trên chợ ứng dụng nhưng khó có thể nhận ra đâu là ứng dụng Native và đâu là ứng dụng Hybrid thường thì người dùng sẽ không để ý quá nhiều.

Đồng thời như ứng dụng Web chúng được xây dựng dựa trên HTML + CSS + JS. Nhược điểm của Hybrid App đó chính là vấn đề hiệu năng sẽ bị ảnh hưởng đáng kể cũng như không tương tác được hết những tài nguyên hệ thống phần mềm cũng như phần cứng.



Figure 10: Hybird App

Vì vậy, mà Native App là sự lựa chọn hàng đầu của giới lập trình trong việc phát triển phần mềm di động bởi vì nó mang lại hiệu năng cao, trải nghiệm người dùng tốt.

* Native Code: Là sử dụng ngôn ngữ của chính nền tảng để xây dựng nên ứng dụng native. So với cross-platform thì cách sử dụng native code giúp mang lại hiệu suất hoạt động tốt nhất, đồng thời nhận được nhiều tài liệu hỗ trợ từ nhà phát triển cũng như từ phía cộng đồng. Tuy nhiên, hạn chế của native code là nếu muốn phát triển ứng dụng trên nhiều nền tảng khác nhau thì sẽ tốn nhiều thời gian và nguồn lực để xây dựng ứng dụng trên từng nền tảng riêng biệt.

## Lịch sử

Năm 2012 Mark Zuckerberg đã phát biểu, "Sai lầm lớn nhất của chúng tôi khi làm công ty là dựa trên quá nhiều HTML hơn là môi trường phát triển gốc". Ông hứa rằng Facebook sẽ sớm cung cấp trải nghiệm di động tốt hơn.

Kỹ sư Jordan Walke tại Facebook đã tìm ra cách xây dựng các thành phần UI cho iOS bằng một luồng JavaScript. Họ quyết định tổ chức cuộc thi Hackathon để hoàn thiện nguyên mẫu hệ thống để có thể xây dựng các ứng dụng di động gốc (native app) bằng công nghệ này.

Sau nhiều tháng phát triển, Facebook đã phát hành phiên bản đầu tiên cho React Native vào năm 2015. Trong một cuộc hội thảo công nghệ, Christopher Chedeau cho biết Facebook đã sử dụng React Native trong phát triển ứng dụng nhóm và ứng dụng quản lí quảng cáo của họ.

## Thông tin cụ thể

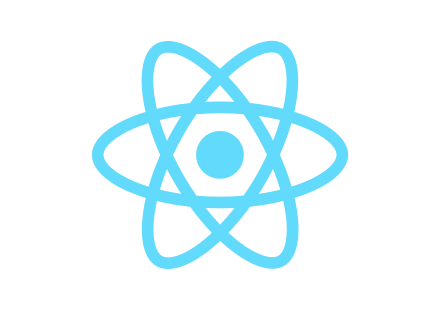


Figure 11: React Native icon

* + Phát triển bởi: Facebook và cộng đồng
  + Phát hành lần đầu: 26/03/2015
  + Phiên bản ổn định: 0.61.5 / 23 tháng 11 năm 2019
  + Repository: https://github.com/facebook/react-native
  + Viết bằng: JavaScript, Java, C++, Objective-C, Objective-C++, Python
  + Giấy phép: MIT License
  + Website: facebook.github.io/react-native/

## Hoạt động

Các nguyên tắc hoạt động của React Native gần như giống hệt với React ngoại trừ việc React Native không thao tác với DOM thông qua DOM ảo. Nó chạy một quá trình xử lý nền (phiên dịch JavaScript viết bởi các nhà phát triển) trực tiếp trên thiết bị đầu cuối và giao tiếp với nền tảng gốc qua một cầu trung gian, bất đông bộ và theo đợt.

Các thành phần React bao bọc mã gốc và tương tác với API gốc qua mô hình UI khai báo và Javascript của React. Điều này giúp việc phát triển ứng dụng cho nhiều nền tảng nhanh hơn.

React Native không sử dụng HTML. Thay vào đó, nó sử dụng các thành phần khác từ luồng Javascript.

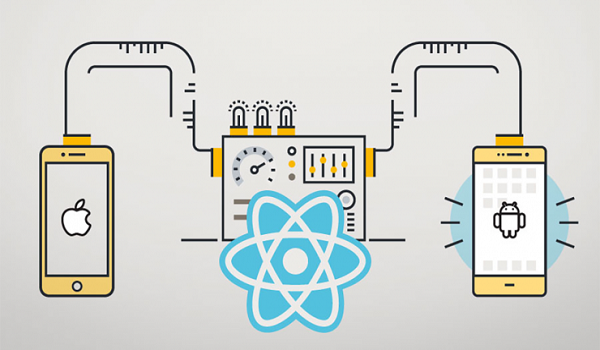


Figure 12: React Native hoạt động ra sao?

Bằng cách tích hợp 2 thread là Main Thread và JS Thread cho ứng dụng mobile. Với Main Thread sẽ đảm nhận vai trò cập nhật giao diện người dùng(UI). Sau đó sẽ xử lý tương tác người dùng. Trong khi đó, JS Thread sẽ thực thi và xử lý code Javascript. Hai luồng này hoạt động độc lập với nhau.

Để tương tác được với nhau hai Thread sẽ sử dụng một Bridge (cầu nối). Cho phép chúng giao tiếp mà không phụ thuộc lẫn nhau, chuyển đổi dữ liệu từ thread này sang thread khác. Dữ liệu từ hai Thread được vận hành khi tiếp nối dữ liệu cho nhau.

## Ưu và nhược điểm của React Native

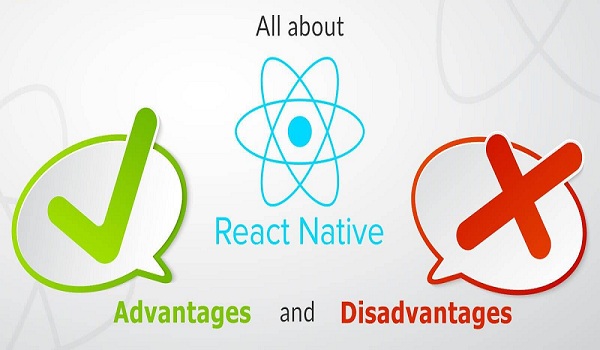


Figure 13: Ưu điểm và nhược điểm của React Native

### Ưu điểm

#### **Trong lĩnh vực phát triển ứng dụng mobile:**

* *Hot reloading, Live reload*: Với bản thân đã code native application bằng Android, mỗi lần thay đổi UI thì phải chờ nó rebuild lại project rồi mới run application trên device làm nó mất rất nhiều thời gian.

Còn với React Native, giai đoạn build app khá nhanh và nếu có thay đổi gì, chỉ cần Edit và Save lại, thì lập tức nó reload lại project rất nhanh, đó là tính năng của Hot reloading.

Còn live reload nó sẽ load lại tất cả các file của Project. Thì đôi lúc Hot reloading nó tiện trong lúc handle với UI design, còn thay đổi logic đôi lúc nó lại không cập nhật được (cái này cũng hiếm khi) thì Live reload là giải pháp đó, live reload cũng rất nhanh. Hot reloading thì nó chỉ load lại cái file có sự thay đổi, còn live reload thì nó sẽ load lại toàn bộ các file trong Project. Nên chúng ta sẽ cân nhắc trước khi sử dụng 2 cái đó

* *Native Application:*

Không giống những framework khác như Cordova cái mà thường chỉ là một Webview (giống kiểu bật cái ứng dụng điện thoại lên thật chất là đang lướt web mà không có cái thanh địa chỉ trên trình duyệt, tạo ảo giác như đang xài application). React Native được sử dụng để xây dựng một ứng dụng Native thật sự. Những Webview không thể cung cấp trải nghiệm chân thật cho người dùng – điều mà chỉ React Native có thể làm được.

Với React Native, những component cơ bản(View, Image, Button, Checkbox) đều là những component native. Do đó ứng dụng sẽ mang một trải nghiệm liền mạch với từng hệ điều hành. Điều đó thực sự làm lên một khác biệt lớn so với các framework lập trình cross-platform khác.

Không chỉ là tốc độ vì nó gần native nhất, mà nó còn làm cho cuộc sống của lập trình viên trở nên dễ dàng hơn. Họ không cần phải code lại một logic cho các nền tảng khác nhau mà vẫn có ứng dụng native cho từng nền tảng đó.

Nếu nó đã build được như 1 native application thì hiệu năng nó sẽ rất là tốt.

* *Tiết kiệm nhân lực và thời gian:*
  + Có thể build để chạy trên cả iOS và Android nên hiệu quả về mặt thời gian khi mà bạn muốn phát triển một ứng dụng nhanh chóng
  + Tiết kiệm tiền
  + Team phát triển nhỏ
  + Có thể tái sử dụng code rất dễ dàng, lên đến 75%
  + Cung cấp trải nghiệm người dùng rất tốt và tốt hơn khi so sánh với ứng dụng Hybird
  + Học một lần và viết ở mọi nơi
  + Được hỗ trợ bởi cộng động lập trình viên đông đảo
  + Có hỗ trợ vân tay

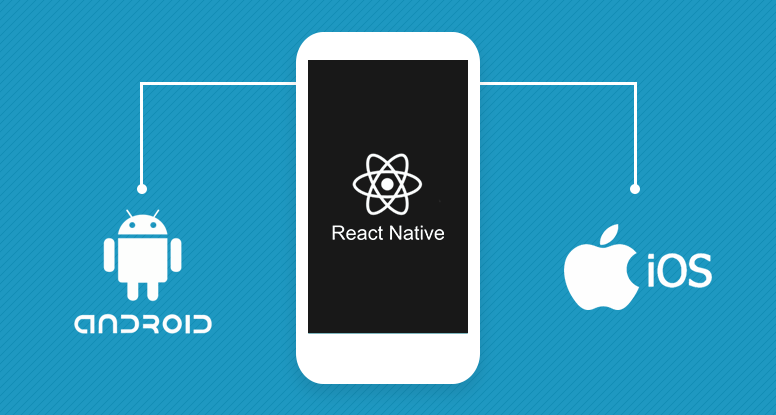


Figure 14: Sự đa nền tảng của React Native

* *Kiến thức nền để tiếp cận:*

Nó không đòi hỏi bạn phải thật giỏi hay quá cao siêu mới có thể tiếp cận được nó. Thứ nhật bạn chỉ cần biết những cái căn bản như XML, HTML, CSS hay chỉ cần biết javascript hoặc Java core, hoặc cho dù bạn có là Web developer hay mobile developer thuần cũng đã có thể học tốt React native rồi.

#### **Trong lĩnh vực nghề nghiệp và trình độ:**

* *Nền tảng React Native tốt hơn và đầy hứa hẹn:*

React Native hiển thị các ứng dụng native cho cả điện thoại di động iOS và Android. React chịu trách nhiệm về view-controller và tạo ra các chế độ xem native bằng cách sử dụng JavaScript. Nó nhắm mục tiêu đến nền tảng di động thay cho các trình duyệt. Nói cách khác, chúng ta có thể tóm tắt bằng lời, các nhà lập trình viên web cảm thấy thoải mái khi viết các ứng dụng dành cho thiết bị di động có giao diện tự nhiên, tất cả từ thư viện JavaScript. Điểm nổi bật của framework này là hầu hết các đoạn code bạn viết có thể được chia sẻ trên các nền tảng- React native giúp phát triển đồng thời cho cả Android và iOS dễ dàng hơn.

Điều này có nghĩa là bạn có thể có tất cả tốc độ và sức mạnh của một ứng dụng native, với việc phát triển dễ dàng đi kèm với React. Hai lý do chính là:

* + Với React Native, ứng dụng logic được viết và chạy trong JavaScript, trong khi UI của ứng dụng hoàn toàn native; nhờ đó, bạn không cần các thỏa hiệp thường được liên kết với HTML5 UI.
  + React giới thiệu một cách tiếp cận mới, cấp tiến và có chức năng cao để xây dựng các giao diện người dùng. Tóm lại, ứng dụng UI được biểu thị đơn giản như một chức năng của trạng thái ứng dụng hiện tại.
* *React Native nối liền những khoảng cách:*

Tương tự như React cho Web, các ứng dụng React Native được viết bằng cách sử dụng hỗn hợp JavaScript và XML, được gọi là JSX. Hỗ trợ cho cả iOS và Android, React Native có tiềm năng mở rộng trên tất cả các nền tảng trong tương lai.

Bạn thực sự có thể sử dụng React Native để tạo ứng dụng dành cho thiết bị di động sẵn sàng cho sản xuất, sử dụng các thành phần giao diện người dùng trên thiết bị di động thực, chứ không phải dạng xem trên web và sẽ trông giống như bất kỳ ứng dụng di động nào khác. Nó cũng hiển thị các giao diện JavaScript cho nền tảng APIs, vì vậy các ứng dụng React Native có thể truy cập các tính năng nền tảng như máy ảnh của điện thoại hoặc vị trí của người dùng, mang đến trải nghiệm người dùng tốt hơn.

***React Native nối liền khoảng cách giữa hiệu suất của ứng dụng native và sự phát triển dễ dàng trên ứng dụng web***

React Native đang chạm đến cái đích cuối cùng để nối liền khoảng cách giữa hiệu suất của ứng dụng native và sự phát triển dễ dàng trên ứng dụng web. Khi hầu hết các framework đang cố gắng bắt chước các thành phần UI native, kết quả thường là chỉ “cảm thấy” một chút; reverse-engineering một thứ gì đó như hình ảnh động có thể phải mất rất nhiều nỗ lực, và chúng có thể nhanh chóng trở nên lỗi thời. Ngược lại, React Native chuyển các đánh dấu của bạn thành các thành phần UI native, thực tế, tận dụng các phương tiện hiển thị có trên bất kỳ nền tảng nào bạn đang làm việc.

Ngoài ra, React hoạt động riêng biệt với main UI thread, vì vậy ứng dụng của bạn có thể duy trì hiệu suất cao mà không bị mất khả năng. Chu trình cập nhật trong React Native cũng giống như trong React: khi props hoặc state thay đổi, React Native tái hiển thị các khung nhìn. Sự khác biệt chính giữa React Native và React trong trình duyệt là React Native thực hiện điều này bằng cách tận dụng các thư viện UI của nền tảng máy chủ của nó, thay vì sử dụng đánh dấu HTML và CSS.

* *Vùng thoải mái của lập trình viên*

Đối với các lập trình viên đã quen làm việc trên Web với React, nghĩa là bạn có thể viết các ứng dụng dành cho thiết bị di động với hiệu suất và giao diện của ứng dụng native, đồng thời sử dụng các công cụ quen thuộc. React Native cũng đại diện cho một cải tiến so với phát triển di động thông thường ở hai lĩnh vực khác: trải nghiệm của nhà phát triển và tiềm năng phát triển đa nền tảng.

React Native mang lại tất cả hiệu suất ứng dụng tốt hơn của React.JS, DOM abstraction (cũng như thao tác trong virtual DOM) và các phương pháp lập trình đơn giản cho phát triển di động hybrid – tốc độ và sự linh hoạt của phát triển ứng dụng web với kết quả native.

React Native cho phép bạn liên kết plugin với module native, do đó bạn có thể link bản đồ với các chức năng của thiết bị như thu phóng, xoay và la bàn, trong khi sử dụng ít bộ nhớ hơn và tải nhanh hơn. Nếu ứng dụng của bạn hỗ trợ các hệ điều hành cũ hơn (và các thiết bị cũ hơn), điều này có thể giúp bạn giữ cho ứng dụng chạy trơn tru. Bắt đầu với React Native thật dễ dàng — đặc biệt là với các ưu điểm JavaScript.

Tóm lại, React Native nhằm mục đích chủ yếu mang sức mạnh của mô hình lập trình React đến phát triển ứng dụng di động. Đó là framework thế hệ tương lai.

### Nhược điểm

React Native là một giải pháp tuyệt vời cho các nhà phát triển ứng dụng mobile, tuy nhiên đến thời điểm hiện tại, vẫn còn tồn tại một số khuyết điểm:

* Vẫn còn thiếu các component view cho Android: Map, Slider, Spiner hoặc các module như Camera Roll, Media, PushNotificationIOS
* Không hỡ trợ Window Phone: đây sẽ là điểm trừ lớn nếu so với ứng dụng hybird. Tuy nhiên, nếu bạn chỉ tập trung cho iOS và Android thì tất nhiên đây không phải là vấn đề lớn.
* Không build được ứng dụng iOS trên Window và Linux: tất nhiên đây không phải là yếu kém của bản thân React Native, mà đó là do yêu cầu từ Apple.
* React Native không thể build được ứng dụng “quá phức tạp” nếu bạn không biết Swift/Objective-C, Java – tính phức tạp ở đâu là ứng dụng của bạn cần phải chỉnh sửa các component. Nên nhớ là để viết được một ứng dụng native bằng javascript “luôn luôn” có sẵn các component đã được viết từ Swift/Objective-C (iOS) và Java (Android) với tính chất 1-1. Trường hợp muốn chỉnh sửa 1 component nào đó: thay đổi thành phần hoặc thêm API thì phải “tự viết” bằng chính ngôn ngữ tương ứng của iOS hoặc Android. Tuy nhiên cũng nhiều lập trình viên khác đã viết nhiều component khác cần thiết cho hầu hết ứng dụng (đây cũng là lý do vì sao Facebook biến React Native thành mã nguồn mở)
* Không nên dùng để viết game có tính đồ họa và cách chơi phức tạp
* Dùng ES6 – cú pháp mới của javascript nên đối với một số lập trình viên có thể không quen
* Mất nhiều thời gian để theo hết hệ sinh thái của React, bao gồm: Reactjs, React Native, Flux, Relay, GraphQL.
* Khả năng cập nhật không được nhanh được như native.
* Không hỗ trợ 64 bit, trên trang của RN đã xác nhận điều này và cũng không có sự hứa hẹn nào trong tương lai.
* Chưa có IDE hỗ trợ
* Bảo mật không cao
* Hiệu năng sẽ thấp hơn với app thuần native code.

## Cài đặt React Native



Figure 15: Cài đặt React Native

React Native hỗ trợ đa hệ điều hành bao gồm Windows, Linux, MacOS. Dưới đây là hướng dẫn cài đặt React Native:

* Bước 1: Cài đặt Nodejs

Việc đầu tiên ta phải làm là cài đặt Node để có được trình quản lý package npm. Trong quá trình phát triển ứng dụng React Native ta sẽ phải sử dụng rất nhiều đến giao diện cosole. Bởi vậy nên cài đặt Node là điều bắt buộc.

* Bước 2: Tiến hành cài đặt React Native App với CLI bằng cú pháp sau:

npm install –g create-react-native-app

* Bước 3: Tạo và run projet mẫu

create-react-native-app my-app

cd my-app

npm start

* Bước 4: Chạy project trên điện thoại

# Flutter



Figure 16: Flutter

Khi nhắc đến việc làm sao để tạo ra 1 ứng dụng mobile thì thứ mọi người sẽ lặp tức nghĩ ngay đến là native app (Android và iOS), nhưng bên cạnh đó vẫn còn rất nhiều công nghệ có thể giúp bạn tạo ra 1 ứng dụng mobile như Cordova, Webview (WeChat), Ionic, Xamarin và React Native. Tất cả các framework, platform đó đều hổ trợ người dùng tạo ra được 1 ứng dụng mobile theo ý muốn của mình.

Và để không bỏ lỡ cuộc vui thì Google cũng đã mang đến cho người dùng đứa con mới nhất của mình - Flutter - nó kế thừa cũng như nổi bật hơn các công nghệ hybrid app kia. Vậy thì làm sao để có thể sử dụng, có thể tạo ra được những sản phẩm theo ý muốn của mình?

## Ngôn ngữ Dart

• Là ngôn ngữ lập trình đơn giản, dễ hiểu và dễ tiếp cận.

• Là ngôn ngữ lập trình được sử dụng để build lên Flutter Framework.

• Nó ra đời vào năm 2011, nhắm đến việc tạo ra các ứng dụng đa nền tảng - web, mobile, desktop và IoT.

• Phiên bản hiện là 2.9 và nó khác là ổn định trong việc lập trình.

• Là ngôn ngữ tĩnh, theo hướng đối tượng (OOP), functional programming và lexical scoped.

• Nó như 1 sự kết hợp giữa Java và JavaScript nên khi học nó nếu ai đã có nền tảng 1 trong 2 ngôn ngữ kia thì lúc tiếp cận sẽ khá dễ.

### Cài đặt Flutter và viết chương trình đầu tiên

#### **Cài đặt:**

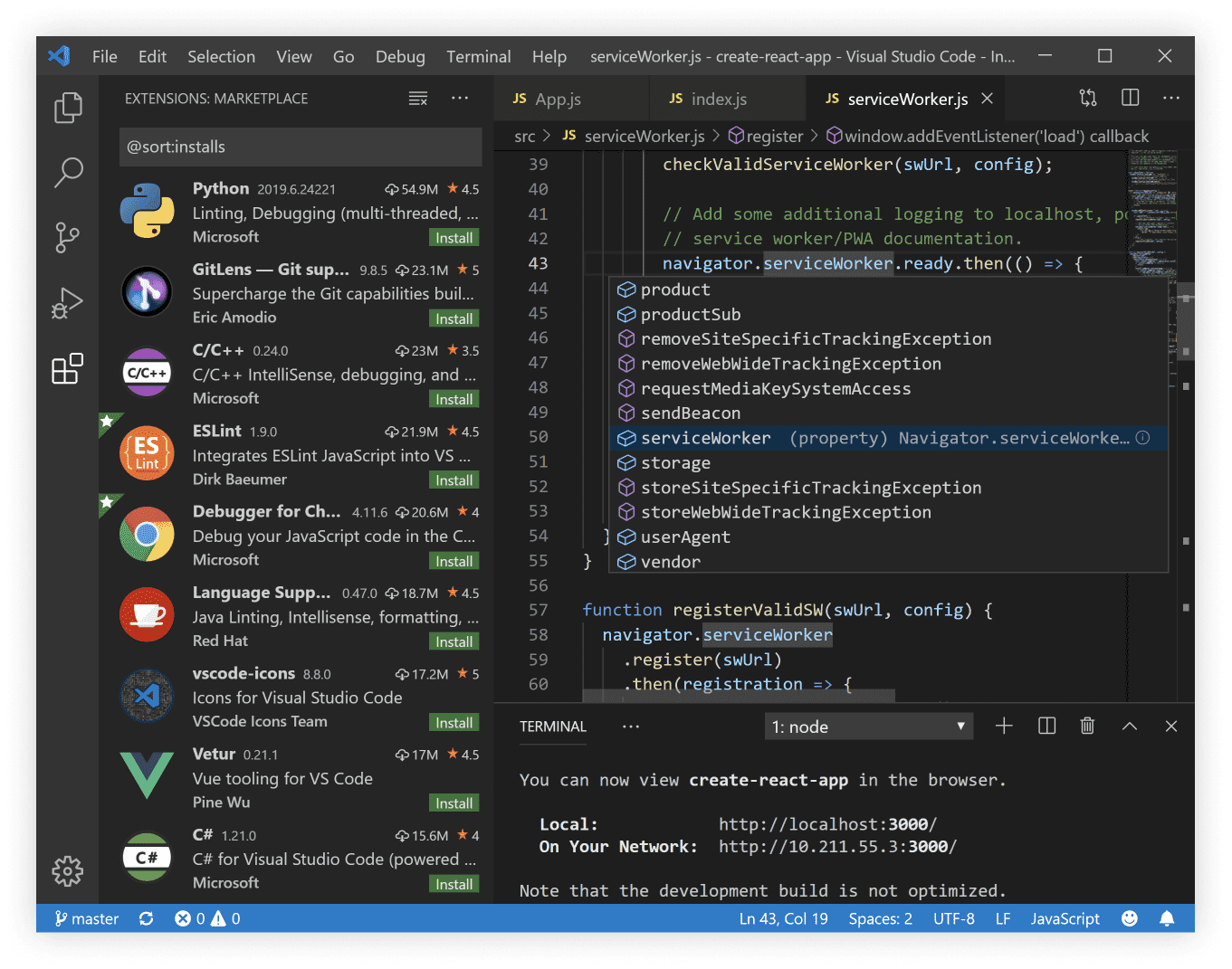
Dart SDK có các thư viện và công cụ dòng lệnh (command-line tools) mà bạn cần để phát triển các ứng dụng Web với ngôn ngữ Dart. Bạn có thể truy cập dart.dev/get-dart để cài đặt Dart SDK.

#### **Viết chương trình đầu tiên**

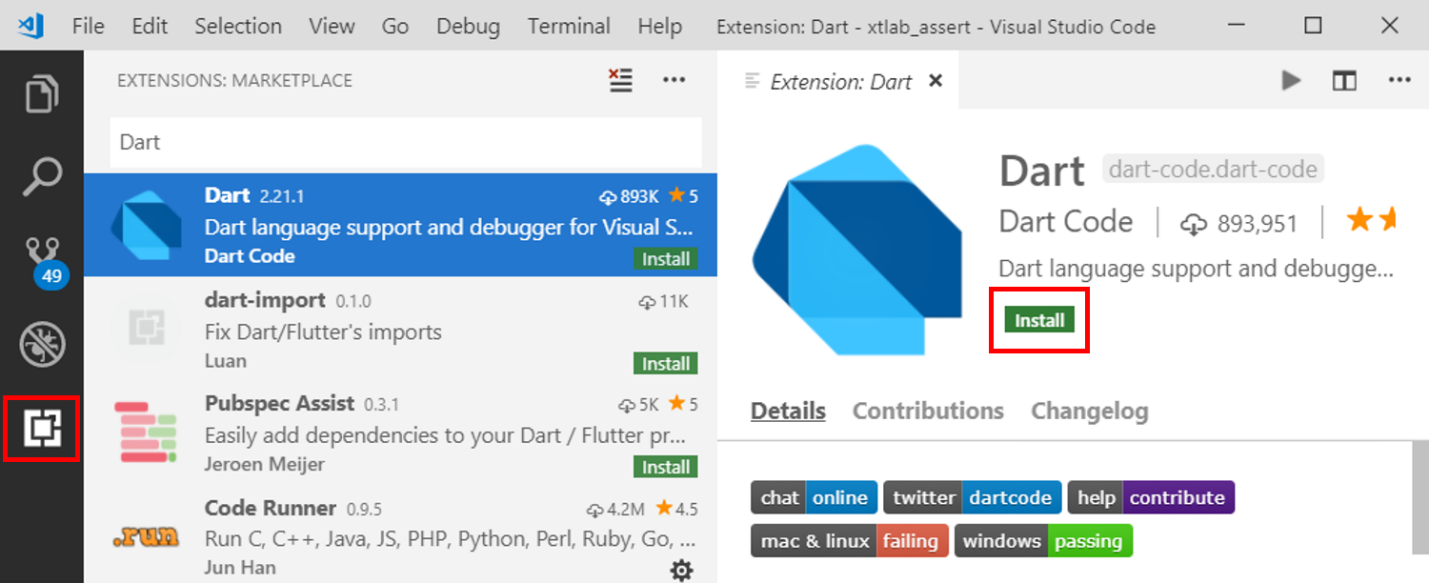
Có nhiều IDE phức tạp mà nó hỗ trợ Dart thông qua các plugin như: Android Studio, IntelliJ IDEA, Visual Studio Code ... Tùy mục đích phát triển ứng dụng mà chọn IDE tương ứng.

Nếu ta muốn thực hành viết code với ngôn ngữ Dart thì Visual Studio Code là một lựa chọn khá ổn. Nhưng nếu để build một ứng dụng Flutter thì ta nên dùng Android Studio.

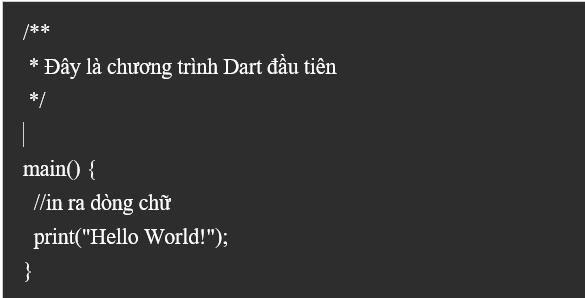
* Đầu tiên ta download VSCode từ trang chủ, sau đó ta cài đặt và mở nó lên.



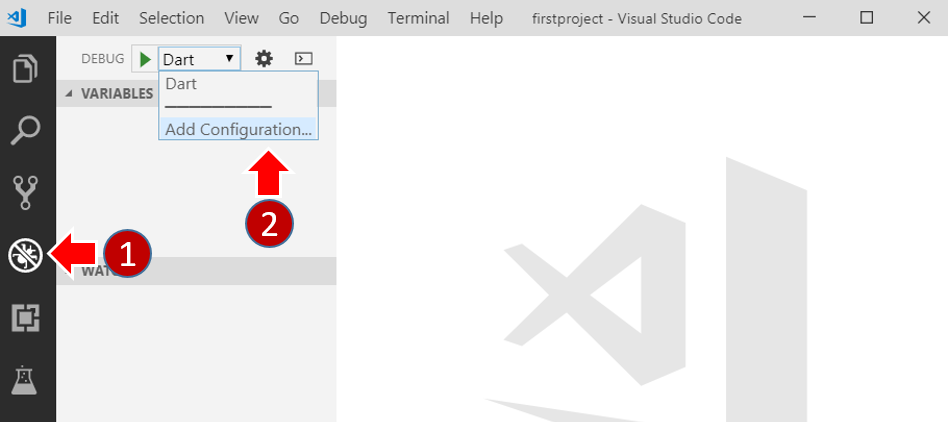
* Để VS Code hỗ trợ kiểm tra cú pháp code, định dạng code, debug ... thì cần cài đặt Plugin do Google cung cấp là Dart code. Chọn mục plugin (đánh dấu màu đỏ) gõ Dart, chọn nó, rồi bấm và install



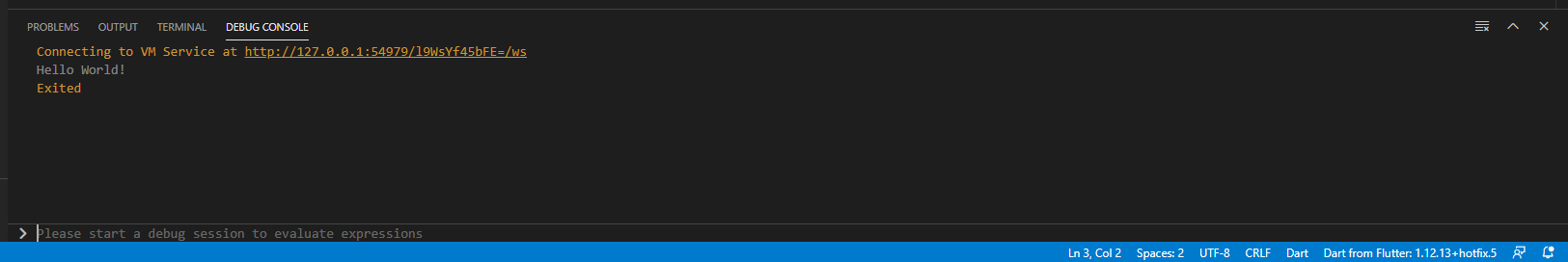
* Sau khi đã cài đặt xong, ta tạo 1 thư mục với tên tùy chọn, trong thư mục sẽ là tên file .dart cũng tùy ta đặt theo ý thích. Sau đó mở thư mục bằng VSCode.
* Ở đây ta sẽ tạo file hello.dart với nội dung như sau :



* Giờ ta sẽ cầu hình để chạy và Debug code. Bấm chọn biểu tượng Debug, sau đó chọn Add configuration ... như hình dưới, nó sẽ tạo ra file launch.json



* Giờ thì có chạy và Debug hello.dart, để bắt đầu chạy và debug bấm vào Start debugging (1), khi code đang chạy sẽ có công cụ debug (2) với các chức năng rất quen thuộc như continue, Step into, restart ...



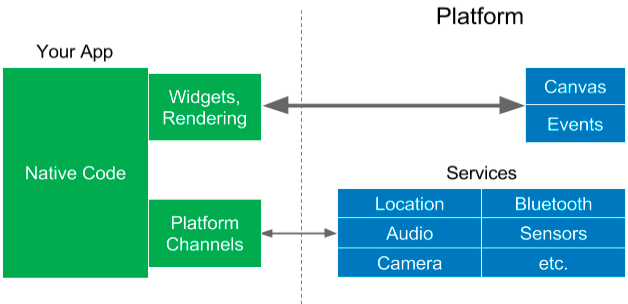
### Một số khái niệm cơ bản khi sử dụng ngôn ngữ Dart

* Bất kì thứ gì có thể đặt vào 1 biến đều là object và mọi object đều là thể hiện của 1 class. Số, hàm, null đều là object. Mọi object thì đều xuất phát từ Object class.
* Tuy Dart là strong typed nhưng Dart vẫn hỗ trợ loose typed. Có nghĩa là nếu bạn chưa chắc chắn về kiểu dữ liệu cho biến thì vẫn có thể khai báo với kiểu dynamic.
* Dart hỗ trợ generic type, ví dụ List<int> và List<dynamic>.
* Dart hỗ trợ top-level function (như hàm main()). Bạn có thể tạo ra 1 hàm bên trong hàm khác (nested hoặc local function).
* Dart cũng hỗ trợ top-level variables.
* Không như Java, Dart không có các từ khóa để set access modifier. Nếu 1 indentifier bắt đầu với ( \_ ) thì nó là private trong library của nó.
* Identifier có thể bắt đầu bằng một chữ cái hoặc dấu gạch dưới ( \_ ).
* Dart vừa hỗ trợ expressions, vừa hỗ trợ statements. Bạn có thể sử dụng biểu thức ? exp1 : exp2 và cũng có thể sử dụng câu lệnh if else.
* Dart tools có thể báo cho bạn 2 loại vấn đề: warnings và errors. Warnings là những dấu hiểu chỉ ra rằng code của bạn có thể không hoạt động, nhưng chương trình của bạn vẫn có thể chạy. Errors có thể là error lúc compile-time hoặc run-time. Error lúc compile-time hiển nhiên sẽ khiến code bạn không chạy được, còn kết quả của error run-time sẽ là những exceptions được throw ra khi chạy.

## Flutter Framework

### Khái niệm

Flutter là 1 mobile SDK do Google phát triển, nó giúp người dùng có thể tạo ra được 1 ứng dụng chạy trên cả iOS và Android. Là một Cross-flatform framework nhưng khác với các Cross-flatform hiện tai, Flutter không thông qua bridge, mà nó sẽ chạy engine render riêng (viết bằng C++) và dùng Flutter framework (viết bằng Dart) để giao tiếp với các service. Cả 2 bộ này sẽ được đóng gói cùng ứng dụng và thông qua SDK nó có thể chạy trên đa nền tảng. Kì vọng mà team phát triển Flutter nhắm đến là có thể chạy trên đa nền tảng, Flutter ngoài chạy trên mobile thì còn có thể chạy trên nền web thông qua dự án mang tên Hummingbird, chạy trên các thiết bị IoT và cả desktop. Tuy vậy, Flutter cũng còn rất "non trẻ", nó cần thêm thời gian để có thể phát triển hơn nữa và việc chọn Flutter để học cũng là 1 cách để đầu tư cho tương lai.



### Đặc điểm nổi bật

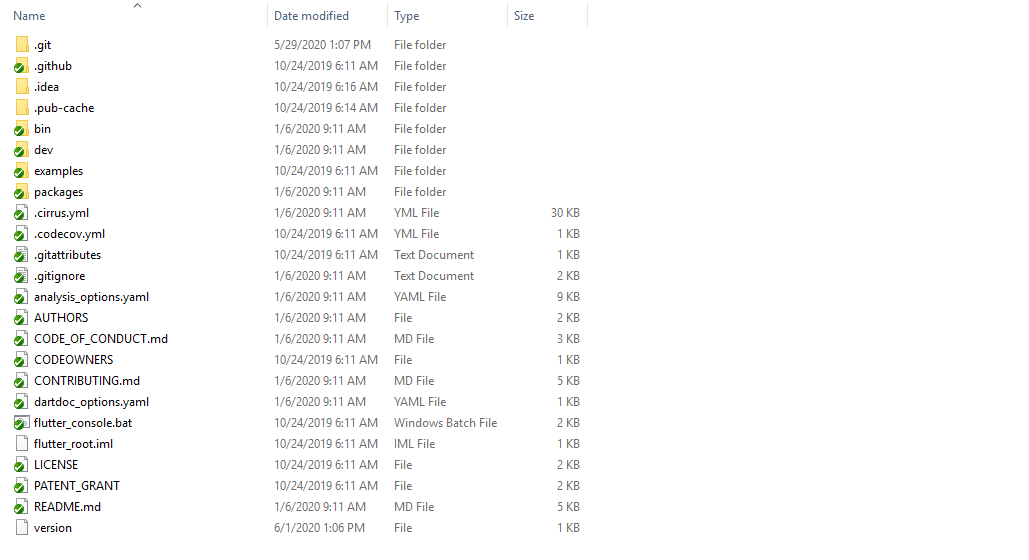
* Fast Development: Tíng năng Hot Reload hoạt động trong milliseconds để hiện thị giao diện tới bạn. Sử dụng tập hợp các widget có thể customizable để xây dựng giao diện trong vài phút. Ngoài ra Hot Reload còn giúp bạn thêm các tính năng, fix bug tiết kiệm thời gian hơn mà không cần phải thông qua máy ảo, máy android hoặc iOS.
* Expressive and Flexible UI: Có rất nhiều các thành phần để xây dựng giao diện của Flutter vô cùng đẹp mắt theo phong cách Material Design và Cupertino, hỗ trợ nhiều các APIs chuyển động, smooth scrolling...
* Native Performance: Các widget của fluter kết hợp các sự khác biệt của các nền tảng ví dụ như scrolling, navigation, icons, font để cung cấp một hiệu năng tốt nhất tới iOS và Android.

### Nhược điểm

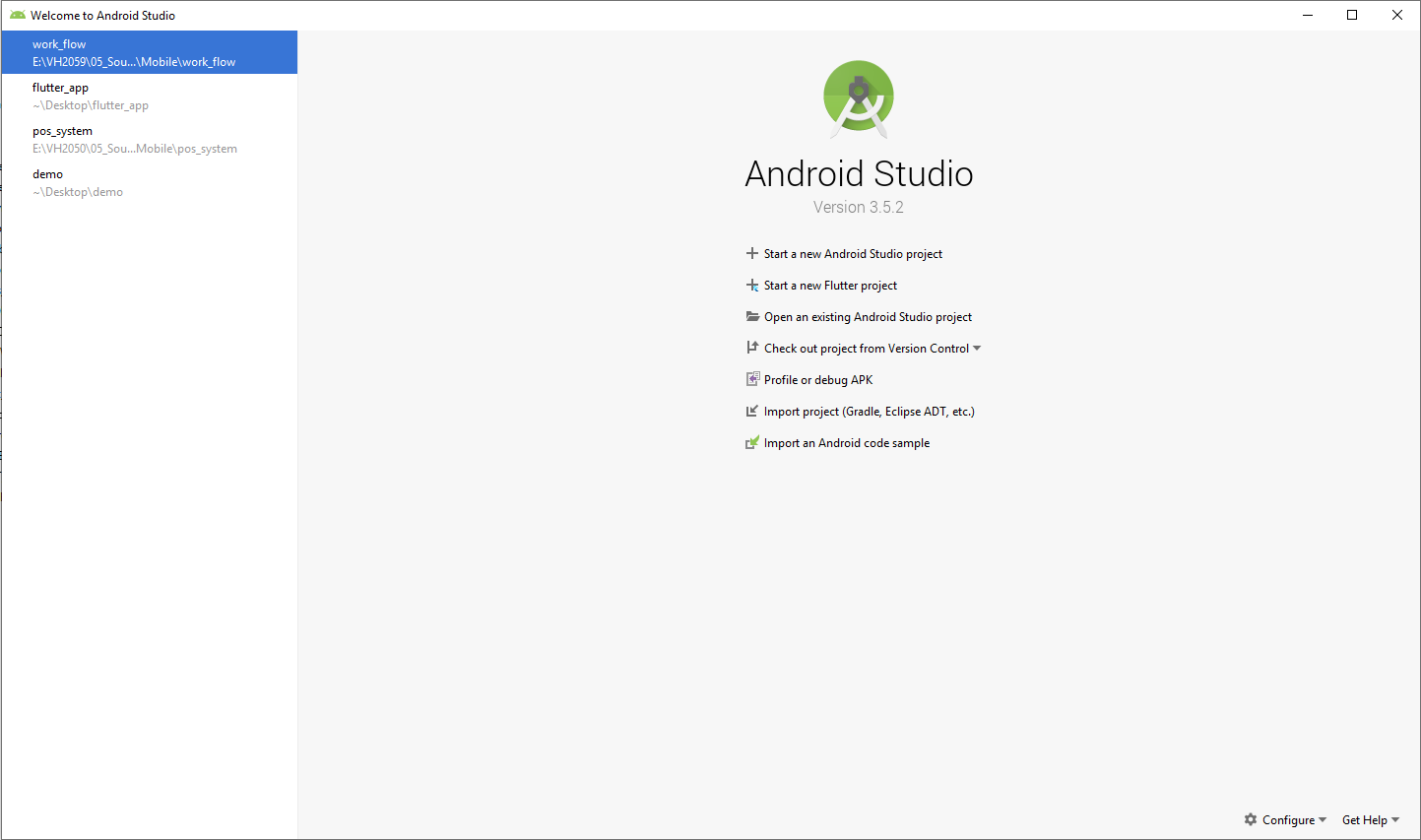
* Bộ render UI được nhóm phát triển gần như viết lại, không liên quan tới UI có sẵn của Framework native, dẫn đến memory sử dụng khá nhiều. Hơn nữa, các UI không đi chung với OS, mà được phát triển riêng, nghĩa là cùng 1 phiên bản Flutter khi tạo ra ứng dụng cho iOS thì iOS 8.x -> 12.x đều y chang nhau, tương tự như với Android. Nhưng UI của Android thì tất nhiên khác với iOS.
* Phải học thêm ngôn ngữ DART: lập trình viên biết về DART không nhiều, cũng có rủi ro là học xong DART sẽ dính liền luôn với DART ở mảng phát triển ứng dụng mobile. Chứ không uyển chuyển như JS hay Python có thể nhảy qua lại giữa front, back hay AI…
* Mô hình dữ liệu mới: bloc pattern, DART Streaming; nếu đã quen với Redux khi làm phát triển React Native, bạn sẽ mất thời gian để học thêm mô hình dữ liệu trong Flutter, mặc dù nó không khó.
* Là con cưng của Google, tuy nhiên hãng dính nhìu phốt với thói quen “quăng con giữa chợ” nên cũng cần cân nhắc. Tuy nhiên, Flutter rất tốt, tốt hơn nhiều so với những cái mà Google từng làm ra như Angular.

### Cài đặt Flutter

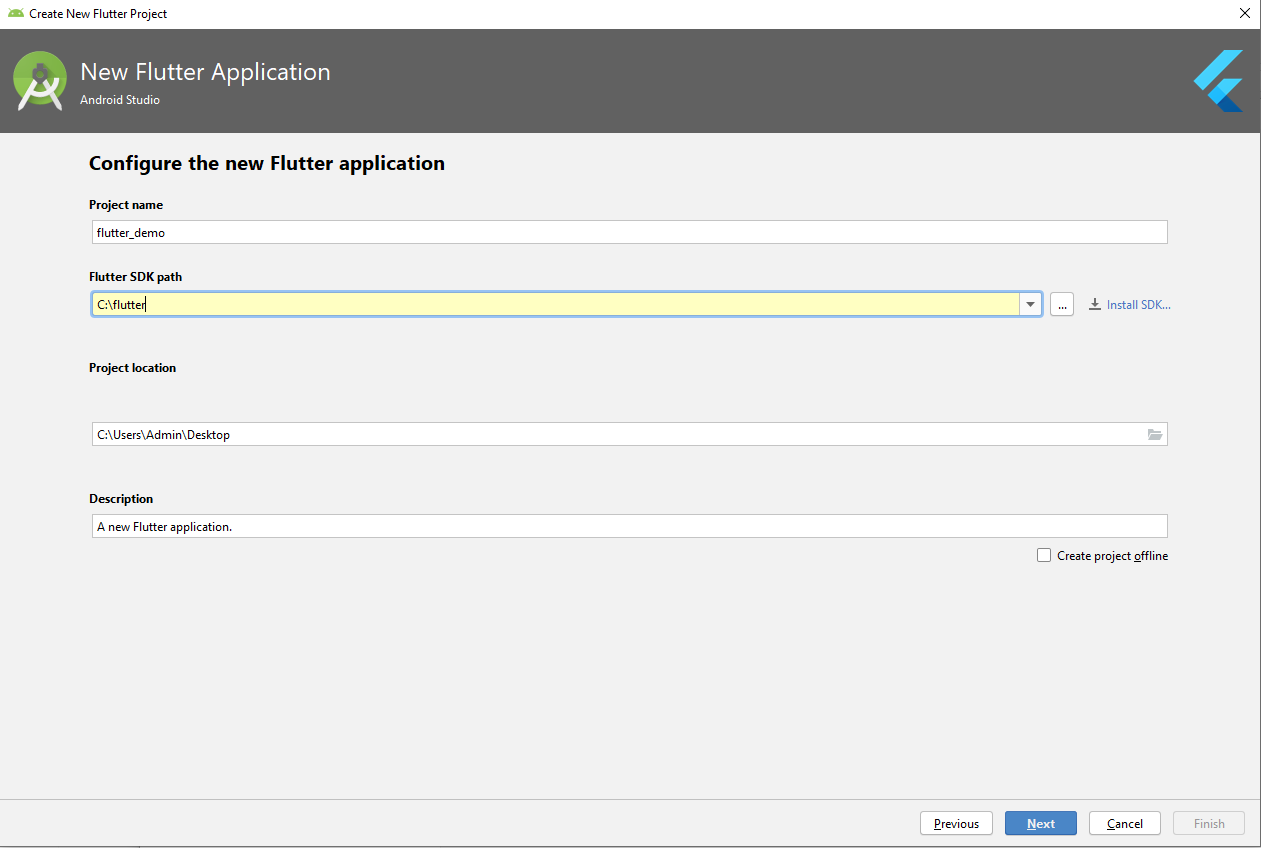
* Để thực hành với ngôn ngữ Dart thì VSCode là sự lựa chọn ổn nhưng với các bạn lập trình java adroid thì tạo ứng dụng với Flutter với Android Studio lại là lựa chọn tuyệt vời.
* Trước tiên ta cần phải download android studio
* Tiếp theo là download Flutter SDK. Đây là các phiên bản mà Google đã release và hiện tại phiên bản mới nhất là 1.20.4.
  + Sau khi download flutter sdk về máy, ta giải nén nó vào bất kỳ ổ cứng hoặc thư mục nào đó mà mình mong muốn ( Ví dụ : C:\src\flutter, lưu ý sẽ không giải nén vào C:\Program Files\ vì thư mục này cần một quyền truy cập cao).
  + Đây là toàn bộ thư mục và file của flutter sdk



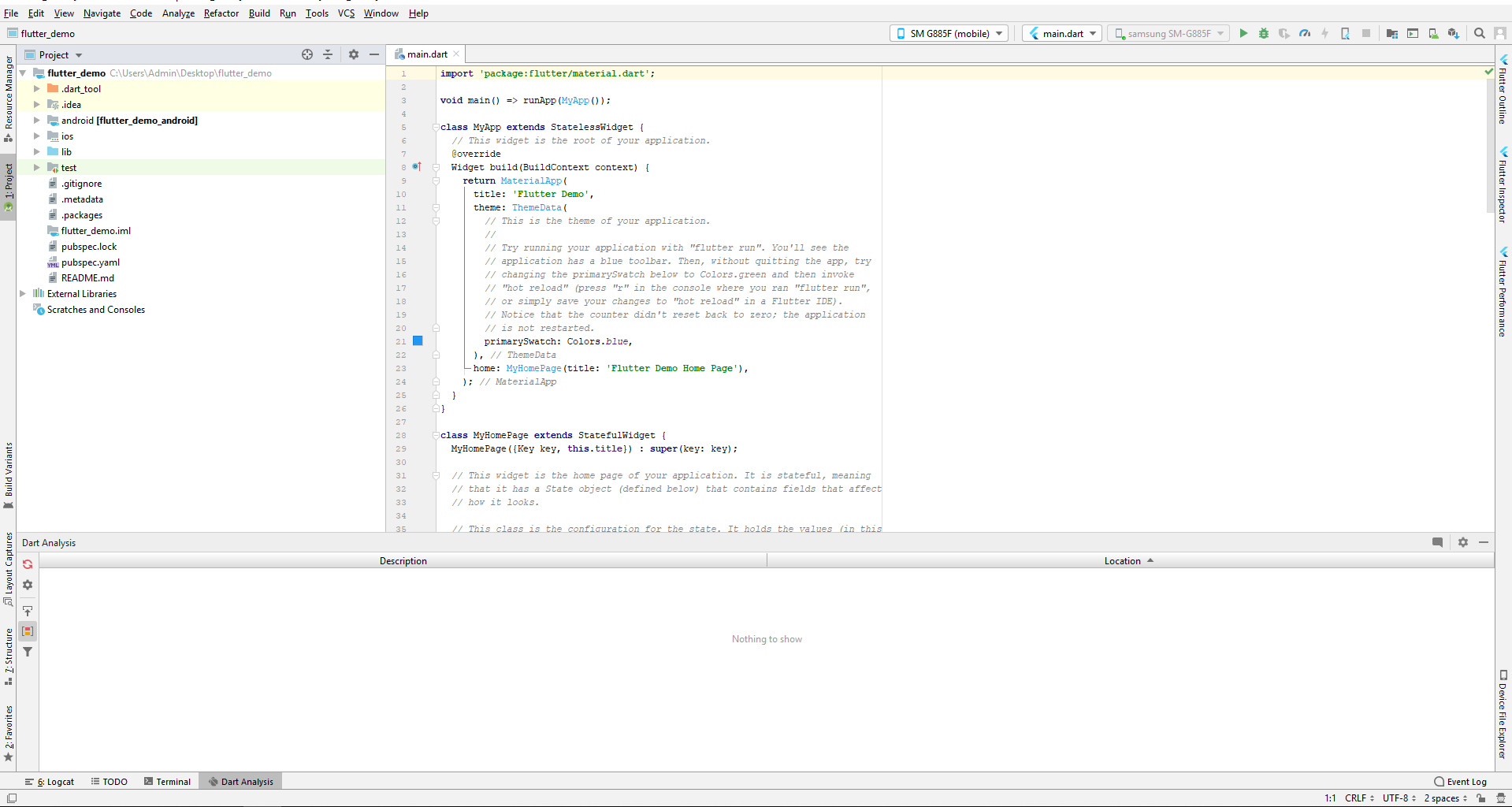
* Sau khi đã cài đặt xong Android Studio và Flutter SDK, mở Android Studio lên và Lựa chọn Start a new Flutter project



* Đến bước cài đặt Flutter SDK path, ta sẽ chọn đến đường dẫn flutter vừa giải nén. Sau đó click Next và đợi Android Studio build ứng dụng mang tên flutter\_demo được viết bằng ngôn ngữ Dart



* Sau khi Android Studio build xong sẽ được một project với cấu trúc như sau :



### Cấu trúc thư mục dự án

* Thư mục lib: đây là thư mục lưu trữ các file .dart mà ta sẽ viết code lên file đó. Khi vừa mới khởi tạo thì chỉ có 1 tệp tin duy nhất là main.dart, main.dart phải tồn tại và nó phải nằm trong thư mục gốc của lib. Đây là tệp mà Dart và Flutter biết để chạy vào ứng dụng của bạn
* File pubspec.yaml : đây là file chứa các package, khai báo các ảnh offline cho ứng dụng, fonts, ...
* Với những bạn mới bắt đầu thì chỉ cần quan tâm đến thư mục lib và tệp pubspec.yaml vì đây là nơi bạn liệt kê các dependencies của dự án