



REACT NATIVE

By VNTALKING



LẬP TRÌNH REACT NATIVE THẬT ĐƠN GIẢN

XÂY DỰNG MOBILE APP BẰNG JAVASCRIPT



MỤC LỤC

LỜI NÓI ĐẦU.....	1
Yêu cầu trình độ	4
Cách học đúng cách	4
Liên hệ tác giả	5
GIỚI THIỆU REACT NATIVE	6
Ưu điểm của React Native.....	7
Nhược điểm của React Native.....	9
Khi nào nên chọn React Native cho dự án	10
Giới thiệu Expo và React Native CLI.....	12
Expo	12
React Native CLI	14
Cài đặt môi trường phát triển	15
Dành cho máy Window	16
Dành cho máy MacOS	24
TẠO ỨNG DỤNG ĐẦU TIÊN	27
Cấu trúc thư mục dự án	30
Debug trong React Native.....	40
CÔNG CỤ VÀ TÀI NGUYÊN.....	46
Visual Studio Code	47
Extensions hữu ích.....	48
REACT NATIVE COMPONENT	51
Khái niệm Component	52
Vòng đời của Component	54
Props	57
State	59
Truyền dữ liệu giữa các component.....	62
Xử lý Event trong Component	71
Functional component và Class component	75
React Hooks	78

STYLE VÀ LAYOUT	85
Áp dụng style	86
Inline Style	87
Đối tượng StyleSheet	89
Hiểu về Flexbox	91
Flex.....	93
Flex Direction	96
Justify Content.....	99
AlignItem.....	101
CORE COMPONENTS.....	104
Text.....	105
TextInput	109
View	114
Image.....	115
Image Background	120
FlatList.....	122
Modal	126
Tổng kết	132
REACT REDUX	133
Khái niệm chung Redux	134
Nguyên lý hoạt động Redux	135
Các thành phần của React Redux.....	135
Cài đặt và cấu hình Redux cho dự án.....	138
Tổng kết	147
REACT SAGAS	148
Middleware là gì?.....	150
Tổng quan Redux saga	151
Generator function	152
Cách thức hoạt động của Redux Saga	156
KẾT NỐI API BẰNG SAGAS	158
Cài đặt thư viện React Saga	163

Viết logic gọi API trong Saga	164
Cấu hình thêm Saga làm middleware của Store	167
Thêm loading UI khi kết nối API	170
NAVIGATION.....	175
Tại sao chọn React Navigation?	177
Phân loại Navigation	178
Stack Navigation	178
Tab Navigation	189
Drawer Navigation	200
XUẤT BẢN ỨNG DỤNG	207
Build App Bundle.....	207
Tạo keystore.....	208
Đưa keystore vào dự án	209
Generating ra .aab hoặc apk	211
Submit ứng dụng lên Google Play Store.....	212
REACT NATIVE STARTER KIT	223
React Native Starter	224
Ignite.....	225
ReactNative Full Template	226
DỰ ÁN MOBILE CUỐI KHÓA.....	228
Phân tích yêu cầu	229
Xây dựng mockup UI.....	235
Cấu hình Store và Axios	239
Gọi API và hiển thị ra màn hình	242
Tối ưu StatusBar	245
XIN CHÚC MỪNG!	249
KẾT NỐI VỚI VNTALKING	250
THÔNG TIN TÁC GIẢ.....	251
CUỐN SÁCH CỦA VNTALKING.....	252

1

LỜI NÓI ĐẦU

Trước đây, mỗi khi cần xây dựng một ứng dụng dành cho điện thoại di động, bạn sẽ phải đắn đo suy nghĩ mình sẽ dành nguồn lực phát triển cho Android trước hay iOS trước?

Lý do bởi vì bạn phải viết mã nguồn native cho từng nền tảng, ví dụ Android bạn sẽ phải viết bằng Java hoặc Kotlin, với nền tảng iOS thì cần viết bằng Objective-C hoặc Swift.

Điều này sẽ khiến bạn tốn nhiều thời gian để học và sử dụng thành thạo nhiều ngôn ngữ lập trình, chưa kể tới các nguồn lực khác khi phải xây dựng nhiều dự án độc lập chạy trên các nền tảng khác nhau.

Để khắc phục những vấn đề đó, các nhà phát triển đã cho ra đời các công nghệ, framework giúp bạn viết code một lần nhưng xuất bản cho nhiều nền tảng khác nhau (Android, iOS, Window...), khái niệm này người ta gọi chung là Cross-Platform.

Trong số những ứng cử viên thư viện/framework cross-platform sáng giá nhất, phải kể đến React Native.

Trong cuốn sách này, chúng ta sẽ cùng nhau tìm hiểu và thực hành xây dựng ứng dụng di động bằng React Native từ A-Z. Mình và bạn sẽ cùng tìm hiểu một loạt những kỹ thuật từ điều hướng màn hình, lưu trữ dữ liệu, quản lý trạng thái state, kết nối ứng dụng tới server, v.v...

Từ đó bạn hoàn toàn tự tin có thể làm chủ React native để xây dựng một ứng dụng của riêng mình và xuất bản lên Play Store hay App Store - để kiếm tiền như một nhà phát triển chuyên nghiệp hay đơn giản là khoe với crush (^_^)

Cuốn sách này là một dự án được mình ấp ủ trong một thời gian dài, với mong muốn truyền đạt và đơn giản hóa hết mức có thể về công nghệ React Native. Với mục tiêu là sau khi bạn hoàn thành cuốn sách này, bạn phải thốt lên rằng "*Ồ, hóa ra lập trình React Native cũng đơn giản nhỉ!*"

Bạn đã sẵn sàng cho hành trình chinh phục React Native chưa?

Vậy còn chờ gì nữa. Hãy tiếp tục đọc và nghiền ngẫm, bạn sẽ cảm thấy yêu thích cuốn sách này.

Mình đảm bảo!

2

SÁCH NÀY DÀNH CHO AI

Cuốn sách này rất phù hợp cho những ai yêu thích lập trình, muốn xây dựng ứng dụng cho điện thoại, trong khi trước đó bạn đã có kiến thức cơ bản về Javascript, ReactJS. Đặc biệt, những bạn bị xếp "dĩ" vào dự án phải dùng tới React Native, những người không có sự lựa chọn, bắt buộc phải học React Native nhanh nhất có thể.

Đây là cuốn sách "**Có yêu cầu kinh nghiệm ReactJS mức cơ bản**". Tức là để có thể tiếp thu được kiến thức trong cuốn sách này một cách nhanh nhất, bạn cần biết một chút về ReactJS, tất nhiên là bao gồm cả Javascript.

Tuy nhiên, nếu bạn chưa có hai nền tảng kiến thức trên, bạn vẫn có thể đọc cuốn sách này, chỉ là chậm hơn một xíu thôi.

Yêu cầu trình độ

React Native là một bộ thư viện xây dựng ứng dụng di động đa nền tảng sử dụng các kỹ thuật lập trình web, dựa trên ReactJS. Do đó, để có thể đọc cuốn sách này một cách trơn tru nhất, bạn nên biết:

- Kiến thức ReactJS cơ bản.
- Biết cú pháp Javascript cơ bản.
- Biết chỉnh sửa CSS.

Nếu bạn không biết những thứ trên thì sao? Cũng không sao, đọc xong cuốn sách này bạn cũng sẽ biết chúng thôi.

Cách học đúng cách

Cuốn sách này mình chia nhỏ nội dung thành nhiều phần, mỗi phần sẽ giới thiệu một chủ đề riêng biệt, kèm thực hành. Mục đích là để bạn có thể chủ động lịch học, không bị dồn nén quá nhiều, dễ dẫn tới "*tẩu hỏa nhập ma*", lúc đó lại oán trách mình ☺

Với mỗi phần lý thuyết, mình đều có ví dụ minh họa. Vì vậy, cách học tốt nhất vẫn là vừa học, vừa thực hành. Bạn nên tự mình gõ lại từng dòng code và kiểm tra kết quả trên thiết bị. Đừng copy cả đoạn code trong sách, điều này sẽ hạn chế khả năng viết code của bạn, cũng như khiến bạn nhiều khi không hiểu vì sao code bị lỗi.

"Học đi đôi với hành. Đọc đến đâu tự viết code đến đó, tự build và kiểm tra đoạn code đó chạy đúng không"

Ngoài ra, trong cuốn sách này, kiến thức phần sau được xây dựng từ phần trước. Do vậy, bạn đừng đọc lướt mà bỏ sót đoạn nào nhé.

Trong quá trình bạn đọc sách, nếu code của bạn không chạy hoặc chạy không đúng như mong muốn trong khi đã vất tay lên trán mấy hôm mà vẫn chưa giải đáp được thì đừng ngần ngại đặt câu hỏi tới tác giả thông qua các kênh liên lạc bên dưới nhé.

Liên hệ tác giả

Nếu gặp bất kỳ vấn đề gì trong quá trình đọc sách, code bị lỗi hoặc không hiểu, các bạn có thể liên hệ với mình theo bất kỳ kênh nào dưới đây:

- Website: <https://vntalking.com>
- Fanpage: <https://facebook.com/vntalking>
- Group: <https://facebook.com/groups/hoidaplaptrinh.vntalking>
- Email: support@vntalking.com
- Github: <https://github.com/vntalking/Book-ReactNative>

3

GIỚI THIỆU REACT NATIVE

React Native là một framework do Facebook phát triển, cho phép tạo các ứng dụng di động bằng các công nghệ Web quen thuộc mà hiệu suất vẫn rất tốt. Ngôn ngữ lập trình chính được sử dụng trong React Native là Javascript.



Do đó, bạn sẽ không cần phải học nhiều ngôn ngữ lập trình, cũng không cần phải tìm hiểu hệ sinh thái từng nền tảng. Đặc biệt, nếu bạn đang là một nhà phát triển ứng dụng web, muốn nhanh chóng xây dựng một ứng dụng mobile thì React native là giải pháp tốt nhất mà bạn nghĩ tới lúc này.

Như cái tên của nó, React Native được xây dựng dựa trên thư viện web nổi tiếng ReactJS. Do đó, để có thể nhanh chóng hiểu được React Native, trước tiên bạn cần phải khám phá ReactJS, nắm được những khái niệm cốt lõi của nó (vì React Native sẽ kế thừa cơ chế hoạt động, cách sử dụng hoàn toàn tương tự).



Để tìm hiểu ReactJS được đầy đủ, bạn có thể tham khảo cuốn sách "[Lập trình React thật đơn giản](#)" do VNTALKING biên soạn.

Với ReactJS, bạn chỉ cần hiểu được các khái niệm cơ bản như:

- JSX (một biến thể của Javascript)
- React Component
- Component Lifecycle – vòng đời của một Component.
- React Hook
- Props, State
- Xử lý xử kiện (Handling event)

Ngoài ra, nếu bạn mà biết nhiều hơn thì càng tốt ^_^, mình cũng không có ý kiến gì nhé.

Ưu điểm của React Native

Tại sao React Native lại trở nên phổ biến, được nhiều công ty gửi gắm niềm tin cho dự án của họ? Đó là do React Native có những ưu điểm

giúp giải quyết được những vấn đề của họ mà những giải pháp khác không có.

Dưới đây là một số ưu điểm của React Native:

- **Thời gian phát triển ứng dụng nhanh:** Điều mà các nhà phát triển “chót yêu” React native đó là họ có thể tái sử dụng và tái chế các component trước đó mà họ đã xây dựng cho web. Họ chỉ cần copy và paste, có khi code chạy luôn mà không cần phải sửa gì. Nói vậy thôi, nhưng về cơ bản, code của React Native giống với ReactJS, nên bạn sẽ tận dụng được tối đa code đã làm.
- **Chi phí phát triển ứng dụng rẻ:** Cái này quá rõ ràng rồi. Thay vì bạn sẽ phải bỏ công sức viết code riêng cho từng nền tảng. Giờ đây, bạn chỉ cần viết code một lần duy nhất, bằng một ngôn ngữ duy nhất - Javascript. Giảm chi phí training cho các thành viên trong dự án.
- **Một code base cho nhiều nền tảng:** Với đặc điểm, chỉ viết một code base, bạn có thể release cho cả Android và iOS. Bạn vừa tiết kiệm được công sức phát triển, vừa nhanh phát hành sản phẩm ra thị trường.
- **Đễ dàng kết hợp với native code:** Một số tính năng đặc thù cần phải tương tác nhiều với phần cứng của thiết bị như: tính năng camera, tính năng Bluetooth... bạn sẽ cần phải sử dụng code native (Java/Kotlin hoặc Swift). React Native cho phép bạn

code native ngay chính trong project và có thể tương tác với code Javascript bên ngoài.

- **Tính năng Hot Reloading/Live reloading:** Là tính năng cho phép cập nhật ứng dụng ngay sau khi chỉnh sửa mã nguồn mà không cần phải build lại toàn bộ. Nó giữ cho ứng dụng hoạt động bình thường và được cập nhật thêm những thay đổi theo thời gian thực. Điều này giúp rút ngắn thời gian chờ đợi rất nhiều mỗi khi chỉnh sửa mã nguồn.
- **Hệ sinh thái React lớn:** Vì React Native được xây dựng dựa trên thư viện ReactJS nên nó kế thừa khá nhiều thư viện, cũng như cộng đồng lập trình viên. Bạn dễ dàng tìm thấy hàng trăm, hàng ngàn thư viện, công cụ hỗ trợ cho việc xây dựng ứng dụng của bạn. Đặc biệt, với sự “chống lưng” của ông lớn công nghệ Facebook, bạn hoàn toàn yên tâm rằng React Native sẽ còn được phát triển dài dài.

Nhược điểm của React Native

Mặc dù khen ngợi không hết lời như vậy, nhưng bất kỳ framework hay giải pháp nào cũng có những hạn chế, nhược điểm nhất định.

Chúng ta có thể điểm qua một số nhược điểm của React Native:

- **Hiệu năng vẫn kém hơn so với Native một xíu:** Mặc dù React Native đã được tối ưu để có thể can thiệp tới code native của hệ thống. Với React Native, nó cho phép bạn render các View bằng native API, nó gọi tới chính SDK tương ứng từng nền tảng để

build ứng dụng. Tuy nhiên, dù gì nó vẫn phải chạy qua một tầng trung gian (là JS Runtime) nên hiệu năng chắc chắn sẽ không thể bằng các ứng dụng native được.

- **Thiết kế không hiệu quả:** Nếu ứng dụng của bạn có thiết kế phức tạp, coi tương tác nâng cao với người dùng là một trong những lợi thế kinh doanh thì bạn nên chọn giải pháp phát triển ứng dụng native.
- **Cập nhật version quá nhanh:** Nói chung cái gì nhanh quá cũng không tốt. Việc cập nhật version quá nhanh nói lên rằng React Native vẫn còn nhiều bugs, cần phải cải thiện nhiều. Chưa kể mỗi khi React Native nâng cấp, ứng dụng của bạn cũng cần xem xét có nên nâng cấp theo không, mà việc nâng cấp khi ứng dụng đang chạy ổn định và đã phát hành ra thị trường luôn là một công việc mệt mỏi, vì bạn sẽ phải test lại rất nhiều.

Khi nào nên chọn React Native cho dự án

Với những ưu điểm và nhược điểm của React Native, chắc hẳn bạn cũng có thể tự đưa ra quyết định khi nào thì nên chọn React Native cho dự án của mình và khi nào thì chọn giải pháp viết code native cho từng nền tảng.

Mình bỏ qua những yếu tố về con người như team của bạn toàn nhân sự giỏi về lập trình web, hoặc khách hàng yêu cầu phải làm bằng công nghệ "lạ", công nghệ "chai", v.v.. Dưới đây là những lý do bạn nên cân nhắc khi nào nên chọn React Native hay không.

Hãy cứ chọn React Native khi:

- Bạn muốn nhanh chóng có một ứng dụng cho cả Android và iOS nhưng chỉ cần viết code một lần.
- Ứng dụng chủ yếu tương tác với server thông qua API để lấy nội dung, những ứng dụng ít tương tác hoặc yêu cầu đặc biệt từ phần cứng, không yêu cầu những tính năng đặc thù của từng nền tảng. Ví dụ các ứng dụng phù hợp với React Native: ứng dụng kiểu đọc báo, tra cứu thời tiết, ứng dụng thanh toán, thương mại điện tử, v.v...



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

4

TẠO ỨNG DỤNG ĐẦU TIÊN

Sau khi hoàn thành xong việc cài đặt, thuận theo “truyền thống” khi học bất kỳ một ngôn ngữ lập trình hay một công cụ nào đó, việc đầu tiên là tạo một ứng dụng Hello world.

OK, giờ mình mở cửa sổ terminal, giả sử mình để thư mục dự án ở màn hình Desktop. Bạn di chuyển con trỏ tới thư mục Desktop bằng lệnh:

```
cd ~/Desktop
```

Hoặc:

```
cd Desktop
```

Để tạo dự án React Native, chúng ta sẽ sử dụng CLI đã cài đặt trước đó, gõ lệnh:

```
npx react-native init VNTalking>Hello
```

Bạn chờ đợi một chút để nó tự động tải template và cài đặt các thư viện cần thiết. Sau khi hoàn thành, bạn sẽ thấy như này:


```
      Welcome to React Native!  
      Learn once, write anywhere  
  
✓ Downloading template  
✓ Copying template  
✓ Processing template  
✓ Installing dependencies  
  
Run instructions for Android:  
  • Have an Android emulator running (quickest way to get started), or a device connected.  
  • cd "C:\Users\Admin\Desktop\VNTalking_Hello" && npx react-native run-android  
  
Run instructions for Windows:  
  • See https://aka.ms/ReactNativeGuideWindows for the latest up-to-date instructions.
```

Tiếp theo, di chuyển con trỏ vào thư mục ứng dụng vừa tạo:

```
cd VNTalking_Hello
```

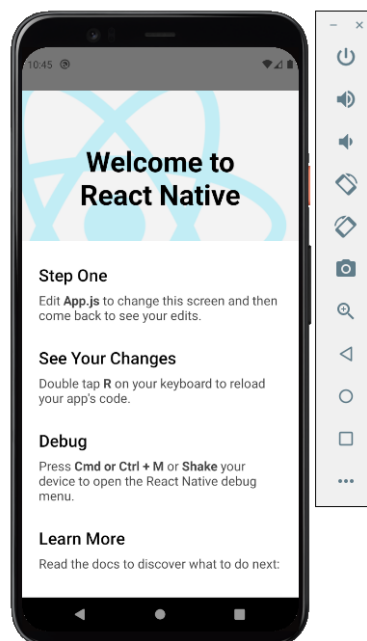
Theo như hướng dẫn trong cửa sổ lệnh trên, để chạy ứng dụng cho Android, bạn gõ lệnh:

```
npx react-native run-android
```

Để chạy ứng dụng trên iOS (chỉ dành cho máy tính Mac), bạn gõ lệnh:

```
npx react-native run-ios
```

OK, gõ luôn và thưởng thức thành quả nhé.



Với các dự án không phải là tạo tự đầu như trên mà đã có mã nguồn từ trước rồi thì bạn phải làm sao để chạy nó?

Để build và chạy ứng dụng từ một mã nguồn dự án React native đã có (ví dụ mã nguồn các bạn lấy trên github, hoặc các mã nguồn trong cuốn sách này), bạn làm như sau:

- Đầu tiên, bạn cần cài đặt các dependencies (chỉ lần đầu tiên chạy dự án, từ lần sau trở đi thì không cần): `npm install`
- Build và chạy ứng dụng:

- ✓ Với device android:

```
npx react-native run-android hoặc npm run android
```

- ✓ Với device iOS (chỉ dành cho máy tính Mac):

```
npx react-native run-ios hoặc npm run ios
```



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

5

CÔNG CỤ VÀ TÀI NGUYÊN

Với Javascript nói chung và React Native nói riêng, có rất nhiều Text Editor hỗ trợ bạn viết code được nhanh hơn, với nhiều tính năng như: highlight code, gợi ý code, định dạng code, v.v...

Một số “ứng viên” bạn có thể thử: Atom, Sublime Text, Notepad++, Visual Studio Code, v.v...

Trong số này, Visual Studio Code (VS Code) là Text Editor được mọi người ưu thích và sử dụng nhiều nhất. Mình cũng vậy, là một fan cứng của VS Code.

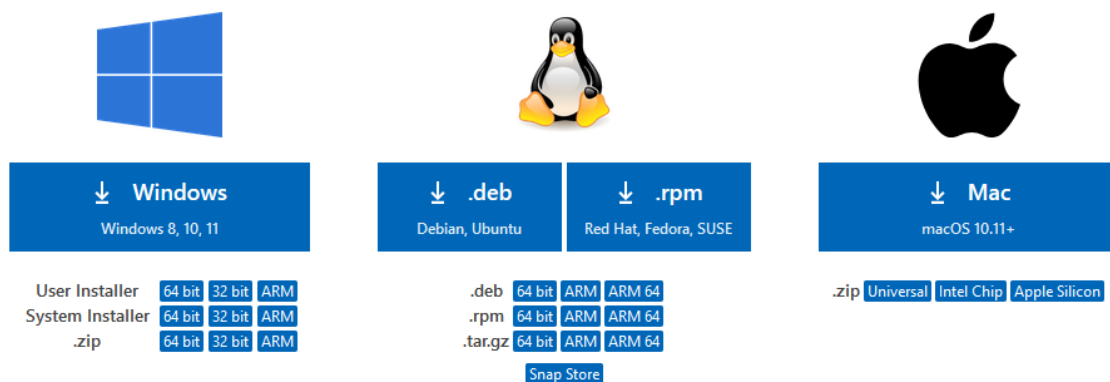
Trong chương này, mình sẽ giới thiệu và hướng dẫn cài đặt VS Code, cũng như cài thêm một số Extension hữu ích để hỗ trợ bạn code “nhanh như một cơn gió”.

Visual Studio Code

Visual Studio Code hay viết tắt là VS Code là một Text Editor miễn phí do Microsoft phát triển. Nó hỗ trợ đa nền tảng, từ Window, MacOS tới Linux. Bản thân công cụ này cũng được xây dựng bằng Javascript.

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



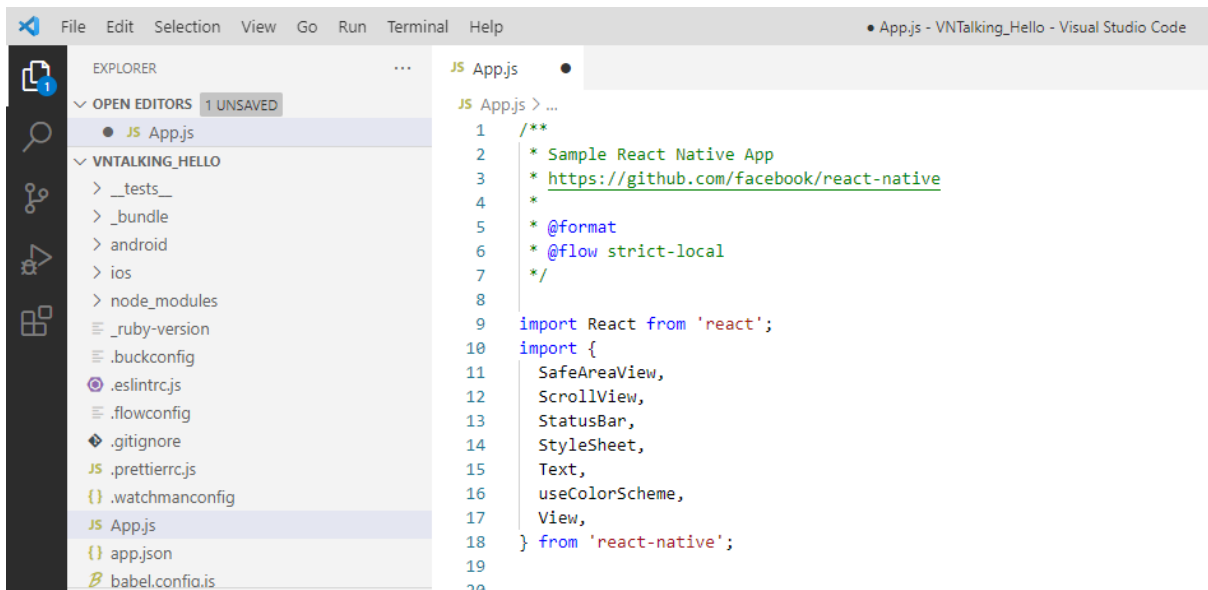
The screenshot shows the download options for Visual Studio Code. It is divided into three main sections: Windows, Linux, and Mac. The Windows section offers a 'User Installer' and 'System Installer' for 64-bit, 32-bit, and ARM architectures, along with a '.zip' file. The Linux section offers '.deb' and '.rpm' packages for various distributions like Debian, Ubuntu, Red Hat, Fedora, and SUSE, with options for 64-bit, ARM, and ARM 64 architectures. There is also a 'Snap Store' option. The Mac section offers a '.zip' file for Universal, Intel Chip, and Apple Silicon architectures.

Các bạn tải phiên bản tương ứng với hệ điều hành mình đang sử dụng, sau đó tiến hành cài đặt như bình thường.



Cuốn sách này mình sẽ không trình bày chi tiết cách cài đặt và sử dụng Visual Studio Code đâu nhé. Bạn có thể tham khảo hướng dẫn chi tiết từng bước [tại đây](#).

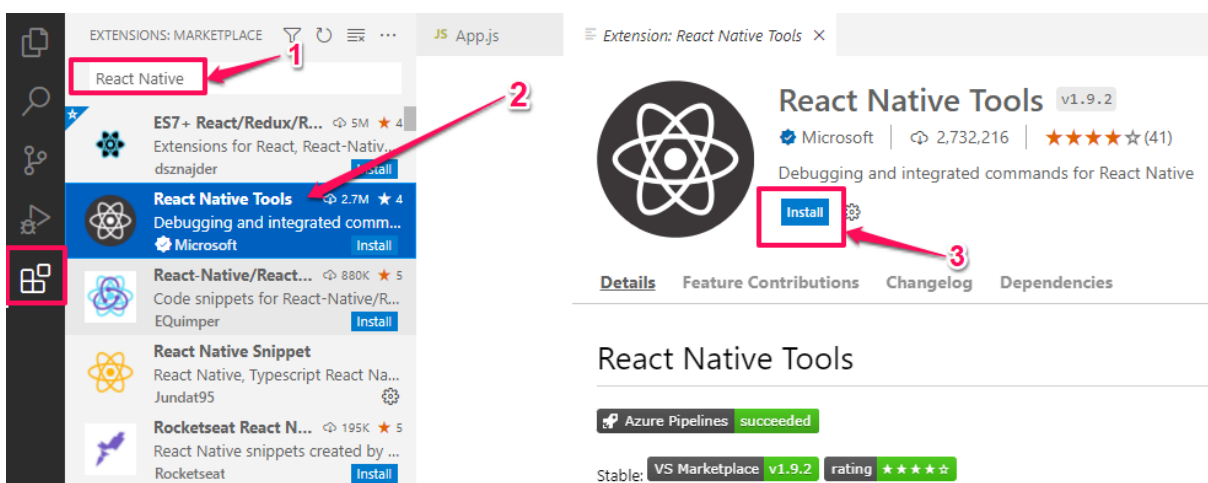
Đây là kết quả mình mở ứng dụng HelloWorld ở trên bằng Visual Studio Code.



Extensions hữu ích

Có một điều mình rất thích ở Visual Studio Code đó là kho Extensions vô cùng phong phú mà toàn miễn phí – thế mới sướng.

Để cài đặt thêm Extension lại cũng vô cùng đơn giản. Bạn vào tab “Extensions”, tìm kiếm extension và nhấn nút “install” là xong (một số extension khi cài đặt hoàn tất sẽ yêu cầu khởi động lại VS Code để có hiệu lực).



Dưới đây là danh sách một số Extensions hữu ích mà mình khuyến khích sử dụng:

1. React Native Tools

Đúng như tên gọi của nó, tiện ích này hỗ trợ môi trường phát triển cho các dự án React Native. Sử dụng tiện ích này, bạn có thể debug và chạy các câu lệnh react-native nhanh chóng trong màn hình terminal.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

6

REACT NATIVE COMPONENT

Hoàn toàn tương tự như ReactJS trong triết lý xây dựng giao diện ứng dụng web. Với React Native, tất cả giao diện ứng dụng sẽ đều là các components. Cơ bản chúng ta chia giao diện thành những block nhỏ hơn nhằm 2 mục đích chính:

- Tái sử dụng càng nhiều càng tốt.
- Đóng gói xử lý logic UI.

Có thể nói Component là một khái niệm vô cùng quan trọng trong React Native, bạn sẽ phải làm việc với nó rất nhiều. Mỗi màn hình trong ứng dụng là một component, trong đó, nó lại chứa nhiều component con. Do vậy, việc hiểu rõ và sử dụng thành thục các component là bạn đã nắm được 80% kiến thức về React Native rồi đấy.

Nếu bạn đã học và làm nhiều về ReactJS thì sẽ thấy phần kiến thức Component sẽ gần như giống tới 99%. Chỉ khác nhau đôi chút về những core components giữa web và mobile app. Ví dụ: để hiển thị

một text trong ReactJS bạn có thể dùng thẻ `<p>` *đây là một dòng text* `</p>` hoặc thẻ ``, thẻ `<h1>`, v.v... thậm chí không cần thẻ nào cũng được. Còn với React Native thì bắt buộc bạn phải dùng thẻ `<Text/>` (một core component có sẵn của React Native).

Tuy nhiên, để cho kiến thức được liền mạch, chương này mình cũng sẽ trình bày chi tiết về Component trong React Native.

Khái niệm Component

Để mà định nghĩa Component một cách tổng quát trừu tượng nhất và “chuẩn khoa học” thật là khó. Nhưng chúng ta có thể hiểu một cách nôm na và “nông dân” như sau:

Trên giao diện một màn hình ứng dụng, chúng ta chia nhỏ những phần giao diện có đặc điểm giống nhau thành một block, và gọi block này là component.

Các nhà phát triển React Native cũng tạo sẵn rất nhiều component mặc định, ví dụ như Text, Button, Image, v.v...

Mỗi component sẽ mang trong mình hai “sứ mệnh” chính:

- **Đóng gói:** Mọi xử lý liên quan tới logic của UI sẽ đóng gói trong component. Một component chỉ nhận dữ liệu qua props, sau đó xử lý và trả kết quả là UI ra màn hình. Tóm lại, mọi tác nhân bên ngoài không làm ảnh hưởng tới kết quả render UI, ngoại trừ dữ liệu mà bạn truyền vào cho component thông qua props.

- **Tái sử dụng:** Chính vì tính đóng gói mà các component có thể tái sử dụng ở bất kỳ đâu. Nhờ đó mỗi khi cần thay đổi UI, bạn chỉ cập nhật một lần code của component là tất cả mọi nơi được thay đổi theo.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

7

STYLE VÀ LAYOUT

Các bạn đã cảm thấy mệt mỏi với React Native chưa? Mình nghĩ là đọc được đến đây chứng tỏ bạn cũng rất nỗ lực và cố gắng để có thể làm chủ được công cụ React Native này. Ở các phần trước, bạn đã hiểu cách hoạt động cũng như tự tạo được component, sử dụng được props và state trong các component. Đó là những kiến thức rất cơ bản nhưng vô cùng quan trọng mà hầu như mọi dự án React Native sau này đều sử dụng.

Tuy nhiên, một ứng dụng muốn thu hút được người dùng thì ngoài việc nó chạy đúng nó còn phải đẹp đã, nhiều khi "*tốt gỗ chưa chắc đã hơn nước sơn*". Mình nói đùa vậy thôi! Chứ việc chúng ta chỉnh sửa style, bố cục của component là một công việc thường xuyên phải làm khi xây dựng ứng dụng React Native.

Ở phần này, chúng ta sẽ cùng nhau tìm hiểu và thực hành cách “mô phỏng” một component, giống như cách chúng ta đã làm với CSS trong lập trình web vậy.

Cách React Native có thể style (trang trí) và tạo layout (bố cục) được lấy cảm hứng từ Web. Hầu hết thuộc tính được sử dụng để trang trí, style cho component có nét tương đồng với CSS, nhờ đó mà bạn sẽ cảm thấy rất quen thuộc. Đối với layout, thay vì sử dụng float, nhóm phát triển React Native lại sử dụng flexbox, một hệ thống layout vô cùng mạnh mẽ, linh động và có vẻ phù hợp với giao diện đặc trưng của mobile.

Chính vì việc style trong React Native có vẻ giống với CSS trong web, nên nhiều bạn đã có kinh nghiệm làm việc với web có thể bỏ qua phần này. Tuy nhiên, vẫn có một số điểm khác biệt nhỏ giữa React native và CSS mà bạn không nên bỏ qua để có thể tự tin làm chủ phần style.

Nào bắt đầu thôi nhỉ? Cùng nhau khai phá *“vùng đất cũ ngõ quen thuộc mà lại lạ không tưởng!”*

Áp dụng style

Khi mới bắt đầu tìm hiểu style trong React Native, chắc hẳn bạn sẽ ngay lập tức thắc mắc: code cái style “khỉ gió” này nó trông như nào nhỉ? Thấy bảo giống CSS của web lắm mà. Và cái code style này được đặt ở đâu trong dự án React Native?

Để bạn khởi lặn tăn, chúng ta sẽ tìm hiểu và trả lời những thắc mắc đó ngay bây giờ đây.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách **tại đây** để đọc bản đầy đủ nhé.

8

CORE COMPONENTS

Ở chương 5 của cuốn sách, chúng ta đã tìm hiểu về component trong React Native rồi đúng không? Cũng đã biết cách để tự tạo ra một component của riêng mình. Nếu như cuốn sách này mà chỉ dành cho ReactJS thì như vậy đã đủ.

Tuy nhiên, với React Native lại khác một chút. Vì đặc thù là xây dựng các ứng dụng gần giống native chạy trên các nền tảng khác nhau (Android, iOS), và cũng để cho các bạn không phải “viết lại bánh xe” nên nhà phát triển đã viết sẵn các component cơ bản nhất, hay dùng nhất và tương thích với các UI component của từng nền tảng. Tương thích ở đây tức là bạn sử dụng component React Native đó và khi chạy ứng dụng trên Android hay iOS nó sẽ hiển thị tương ứng với giao diện của từng nền tảng. Nhưng cũng có những component lại chỉ sử dụng cho duy nhất một nền tảng vì tính đặc trưng của nền tảng đó. Ví dụ TabBarIOS, thanh điều hướng này chỉ có trên iOS.

Các component React Native mà nhà phát triển viết sẵn có rất nhiều, bạn có thể thoải mái sử dụng cho ứng dụng của mình. Trong khuôn khổ cuốn sách này, mình sẽ giới thiệu những component hay sử dụng nhất như: Text, Image, FlatList, Modal, v.v...

Tất nhiên, nhà phát triển họ không giới hạn số lượng các component này, nếu bạn có đủ khả năng hoàn toàn có thể tự tạo một component như của họ.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

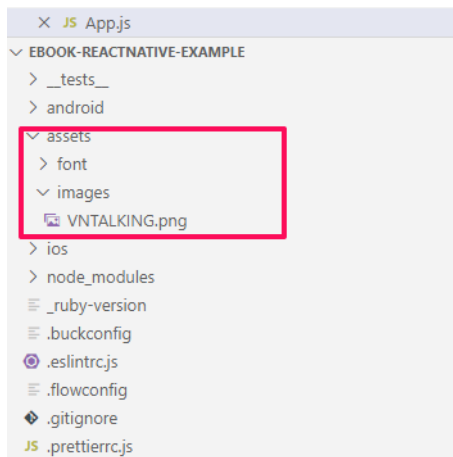
Image

Image là một trong những component rất hay sử dụng. Có hai cách để hiển thị hình ảnh trong ứng dụng. Một là lấy từ ảnh có sẵn trong dự án hoặc ảnh lấy từ trên server thông qua URL.

Chúng sử dụng props `source` để nhận đường dẫn của ảnh. Nếu ảnh lấy từ trong dự án thì sử dụng từ khóa `require(...)`,

```
<Image source={require('./assets/images/VNTALKING.png')} />
```

Như ví dụ trên, hình ảnh được để trong thư mục `assets/images` của dự án.



```
1  /**
2   * Tìm hiểu các core components có sẵn trong React Native
3   * Component: Image
4   * Biên tập: VNTALKING.COM
5   */
6  import React from 'react';
7  import { View, Image } from 'react-native';
8
9  const App = () => {
10   const [number, setNumber] = React.useState(null);
11
12   return (
13     <View style={{height: 150, padding: 10}}>
14       <Image source={require('./assets/images/VNTALKING.png')} />
15     </View>
16   );
17 };
18
```

Nếu ảnh từ server thì dùng từ khóa `uri:<Image_URL>`

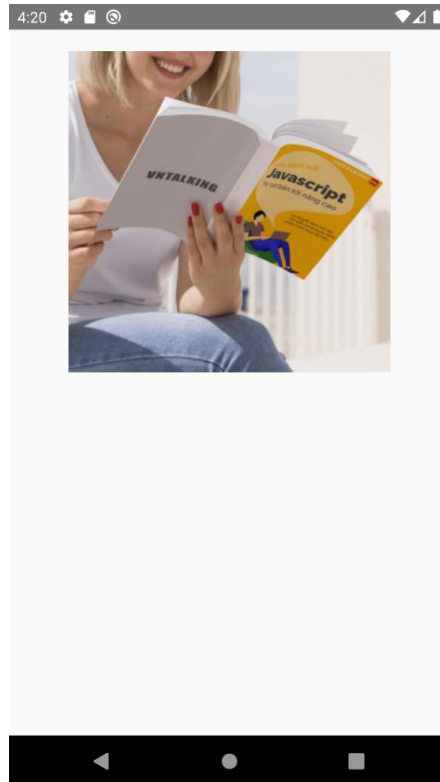
App.js

```
<Image source={{uri: 'https://vntalking.com/wp-content/uploads/2021/05/Javascript_cover-3D-N5-300x300.png'}}
  style={{width: 300, height: 300}} />
</View>
```



Với trường hợp ảnh lấy từ server, bạn cần phải thêm style để xác định kích thước cụ thể (width, height) thì nó mới hiển thị.

Kết quả thu được như sau:



Với trường hợp lấy ảnh từ server, thông thường sẽ mất một chút thời gian để tải ảnh về, nếu ảnh có kích thước lớn thì còn lâu hơn. Để tăng trải nghiệm người dùng, chúng ta có thể hiển thị một biểu tượng loading để báo cho người dùng biết là đang tải ảnh, chờ cho đến khi ảnh được tải về xong.

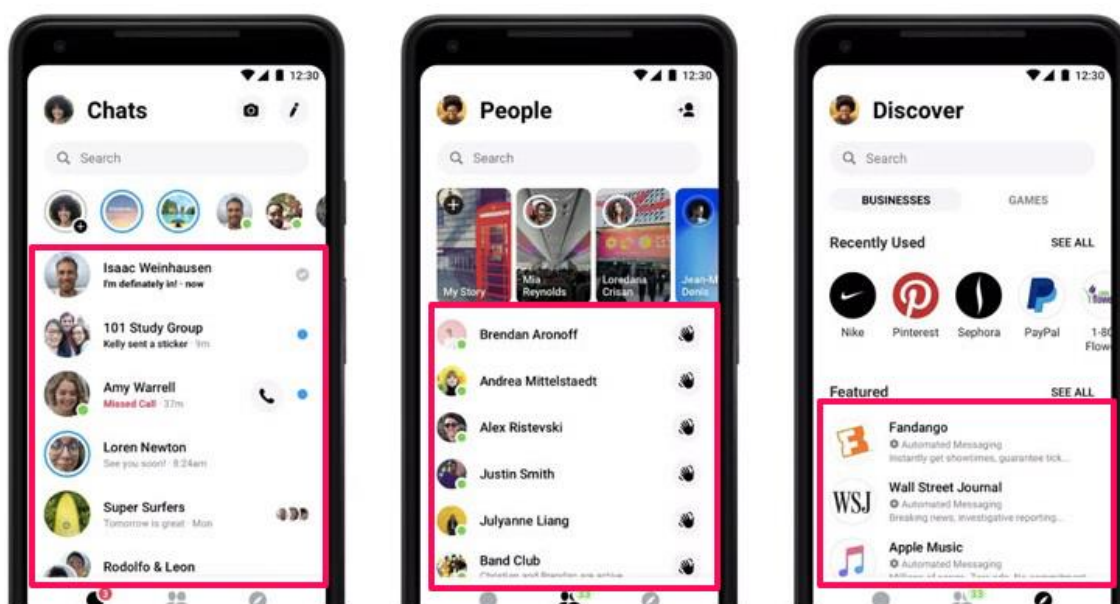


Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

FlatList

Bạn có để ý là hầu hết các ứng dụng đều có màn hình hiển thị danh sách “cái gì đó”, Các ứng dụng phổ thông như Danh bạ, Gmail, Messenger, WhatsApp... đều có.

Như giao diện ứng dụng Messenger dưới đây là một ví dụ:



Trong React Native, để làm được điều đó, bạn có thể sử dụng FlatList component.

FlatList là component được thiết kế để hiển thị số lượng lớn nội dung có thể cuộn trên màn hình. FlatList hiển thị từng mục trong mảng dữ liệu đầu vào bằng cách sử dụng prop: *renderItem*.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

9

REACT REDUX

Trong chương 6, chúng ta đã biết State là một trong những dạng dữ liệu quan trọng trong dự án React Native. State giống như một kho lưu trữ dữ liệu cho các component trong ReactJS. Nó được sử dụng để cập nhật component khi người dùng thực hiện một số hành động như nhấp vào nút, nhập văn bản, ẩn hiện view nào đó, v.v...

Hiện tại chúng ta mới biết cách khai báo và sử dụng state trực tiếp ngay trong từng component, thông qua sử dụng *useState* hook. Với cách làm này thì không có gì sai và mọi người cũng vẫn đang làm như vậy.

Tuy nhiên, nếu state đó được sử dụng ở nhiều component khác nhau, nhiều màn hình trong ứng dụng cùng dùng chung một state. Lúc này bạn phải làm sao?

Mình lấy ví dụ điển hình đó là trường hợp bạn cần lưu trạng thái đăng nhập của một tài khoản. Trạng thái đăng nhập sẽ ảnh hưởng tới toàn

bộ các màn hình trong ứng dụng, bạn có thể hiểu nó như một dữ liệu global cho toàn ứng dụng vậy.

Tiếp tục nhé! Vấn đề bắt đầu khó khăn hơn khi ứng dụng ngày càng có nhiều state dạng global như vậy. Nào là trạng thái đăng nhập, theme (dark mode, light mode), các cấu hình chung của ứng dụng, v.v... Chúng ta cần phải có một giải pháp để quản lý các state global này. Trong React Native, giải pháp phổ biến nhất là sử dụng Redux.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

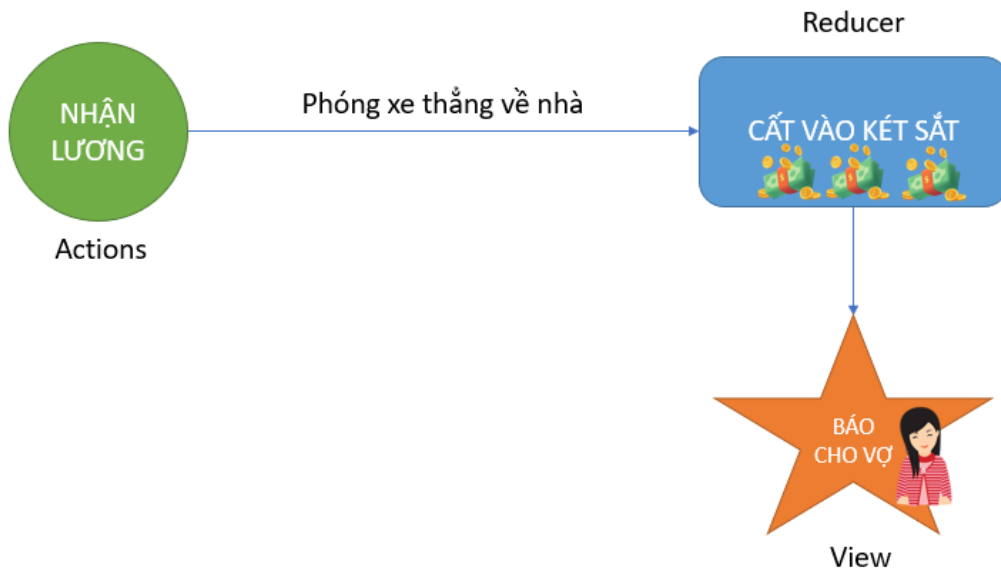
10

REACT SAGAS

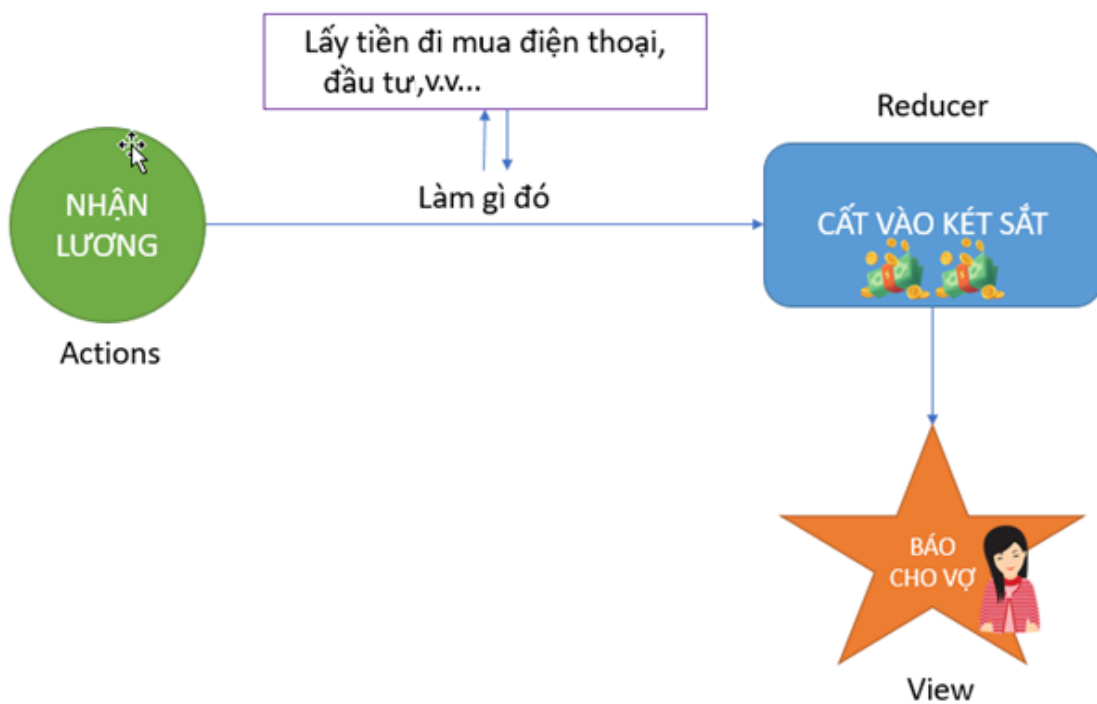
Ở chương trước, chúng ta đã tìm hiểu và thực hành cách sử dụng React Redux để quản lý state trong ứng dụng. Đọc đến chương này, mình tin rằng bạn thực sự là người rất kiên trì, có quyết tâm rất cao, chứ với người khác là bỏ cuộc lâu rồi.

Khi bạn nhìn kỹ hình mô tả các thành phần của Redux, bạn sẽ thấy có một phần nằm ở giữa Action và Reducer, đó là Middleware. Là phần trung gian làm một việc gì đó trước khi gọi tới reducer để lưu dữ liệu.

Để mình ví dụ minh họa nhé: Vào một ngày tuyệt vời nhất trong tháng, sếp gọi bạn đến “NHẬN LƯƠNG”. Lẽ thường, bạn sẽ cầm số tiền này đi về nhà đút ngay vào két sắt và báo cáo cho vợ.



Mọi lần bạn vẫn làm như này đúng không? Tuy nhiên, nếu trong quá trình phóng xe về nhà, bạn lại dùng số tiền đó mua một cái gì đó, hoặc mang đi đầu tư chứng khoán.v.v... Sau đó, số tiền còn lại (có thể ít đi, hoặc nhiều hơn hoặc thậm chí không phải là tiền mà cổ phiếu,v.v...) mới được đưa vào kết sắ. Những thao tác ngoài lề được thực hiện ở một nơi gọi chung là Middleware.



Trên đây là ví dụ vui vạ thôi! Nhưng cơ bản nó cũng giống với ý nghĩa và vai trò của React Saga.

Trong chương này, chúng ta sẽ cùng nhau tìm hiểu React Saga, một trong những middleware phổ biến nhất cho React Native.



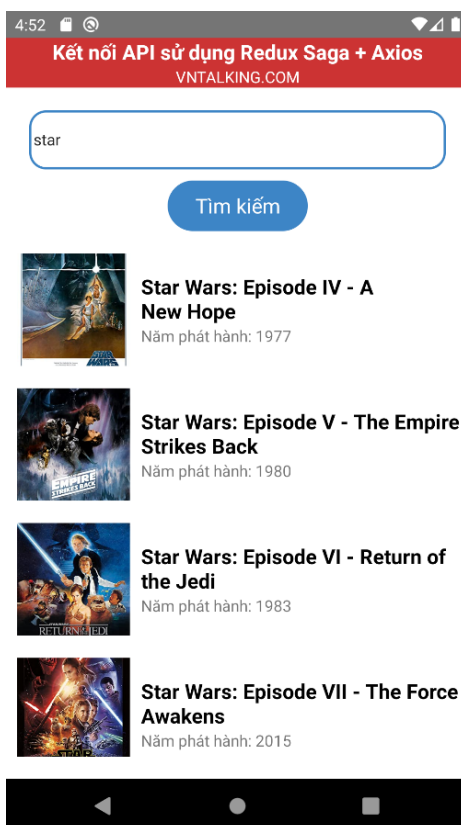
Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

11

KẾT NỐI API BẰNG SAGAS

Mô tả dự án: Xây dựng một ứng dụng cho phép người dùng tìm kiếm phim trên Internet.

Giao diện ứng dụng sau khi hoàn thành như sau:



Các dependencies được sử dụng trong dự án:

- **axios**: Thư viện hỗ trợ gọi các HTTP request
- **react-Redux**: Thư viện bindings cho redux.
- **redux-saga**: Thư viện mà chúng ta đã tìm hiểu ở chương trước

Phần API tìm kiếm phim, mình đăng ký sử dụng API miễn phí của omdbapi: <https://omdbapi.com/?s=<Từ khóa> &apikey=5008fdeb>



Đây chỉ là bản demo - phần này còn nữa
Đặt sách **[tại đây](#)** để đọc bản đầy đủ nhé.

12

NAVIGATION

Hello! Bạn vẫn còn đọc sách đó chứ?!

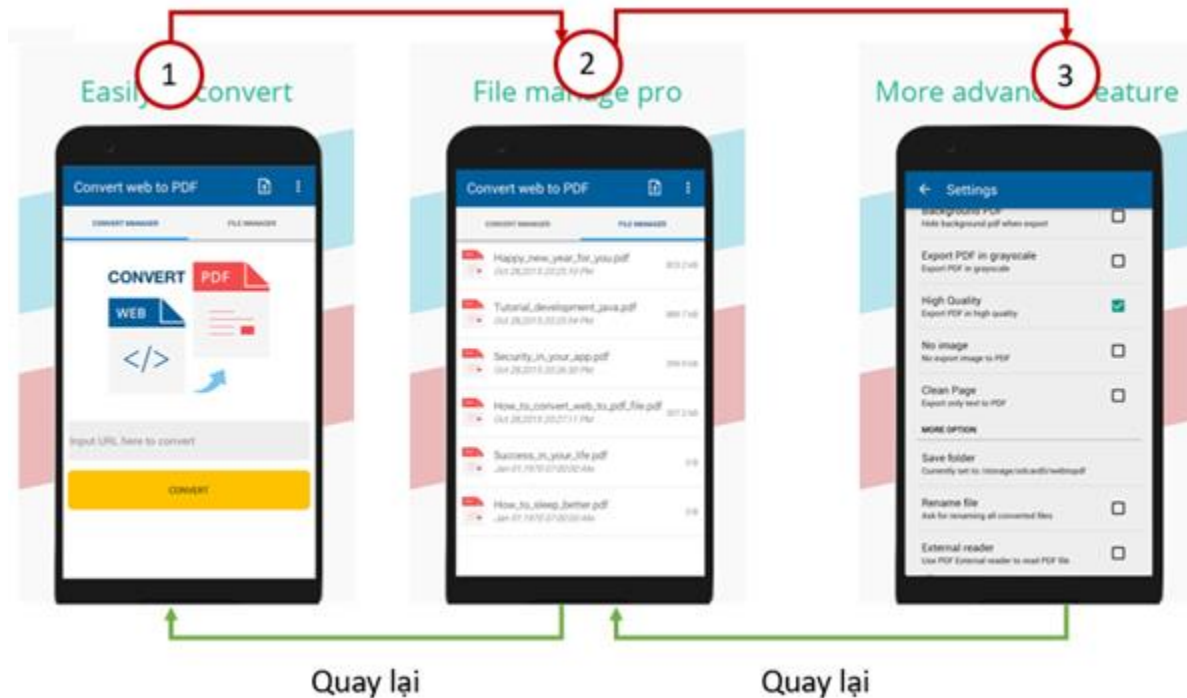
Đọc được đến dòng này là bạn đã đi được một quãng đường khá dài rồi đấy. Mình tin chắc rằng chỉ có 20% độc giả đủ kiên trì và nghị lực để đọc được đến chương này. Bạn là một trong số đó!

Mình nên có một món quà để tặng thưởng cho bạn. Hãy liên hệ với mình để nhận quà nhé 😊.

Quay lại với nội dung của cuốn sách. Từ đầu đến giờ, chúng ta thực hành tạo ứng dụng chủ yếu chỉ có một màn hình. Mà thực tế thì đâu có đơn giản như vậy, các ứng dụng luôn có rất nhiều màn hình với các tính năng khác nhau, người dùng di chuyển qua lại giữa các màn hình rất nhiều. Đó là lý do chúng ta sẽ tìm hiểu về Navigation trong React Native.

Mình ví dụ một ứng dụng mobile như dưới đây. Nó sẽ có một số màn hình như: Màn hình chính nhập URL để convert sang PDF, màn hình

danh sách các file PDF đã được converted, và một màn hình setting để thiết lập tổng thể ứng dụng.



Navigator trong React Native được chia làm 2 loại:

- **Javascript navigator:** Là những package được viết hoàn toàn bằng javascript và chạy trên môi trường javascript.
- **Native navigator:** Được viết bằng ngôn ngữ native của từng platform (ví dụ iOS có Swift hay Android có Kotlin) và nó được kết nối tới React Native thông qua javascript API. Sự khác biệt chính là tách navigation ra khỏi Javascript thread. Giúp có cảm giác sử dụng đồ chính chủ hơn là đồ giả lập. Từ đó sẽ tối ưu performance và ứng dụng được mượt mà hơn.

Để hỗ trợ tính năng navigation – điều hướng giữa các màn hình, chúng ta có thể sử dụng một số thư viện như:

- [React Navigation](#)
- [React Native Router Flux](#)
- [React Router Native](#)

Trong cuốn sách mình sẽ hướng dẫn các bạn sử dụng thư viện React Navigation, đây là thư viện nổi bật nhất được chính nhà phát hành React Native khuyến nghị.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

13

XUẤT BẢN ỨNG DỤNG

Cám ơn các bạn đã đồng hành cùng mình trong một quãng đường khá dài. Sau tất cả, khi ứng dụng đã hoàn thiện, giờ là lúc chúng ta sẽ release sản phẩm để phát hành ra thị trường.

Hiện tại, React Native hỗ trợ build ra hai nền tảng Android và iOS. Tương ứng với đó là chúng ta có thể phân phối ứng dụng qua hai market là Google Play và App Store.

Trong khuôn khổ cuốn sách này, mình sẽ hướng dẫn các bạn cách để build ra App Bundle và submit ứng dụng lên Google Play.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách [tại đây](#) để đọc bản đầy đủ nhé.

14

REACT NATIVE STARTER KIT

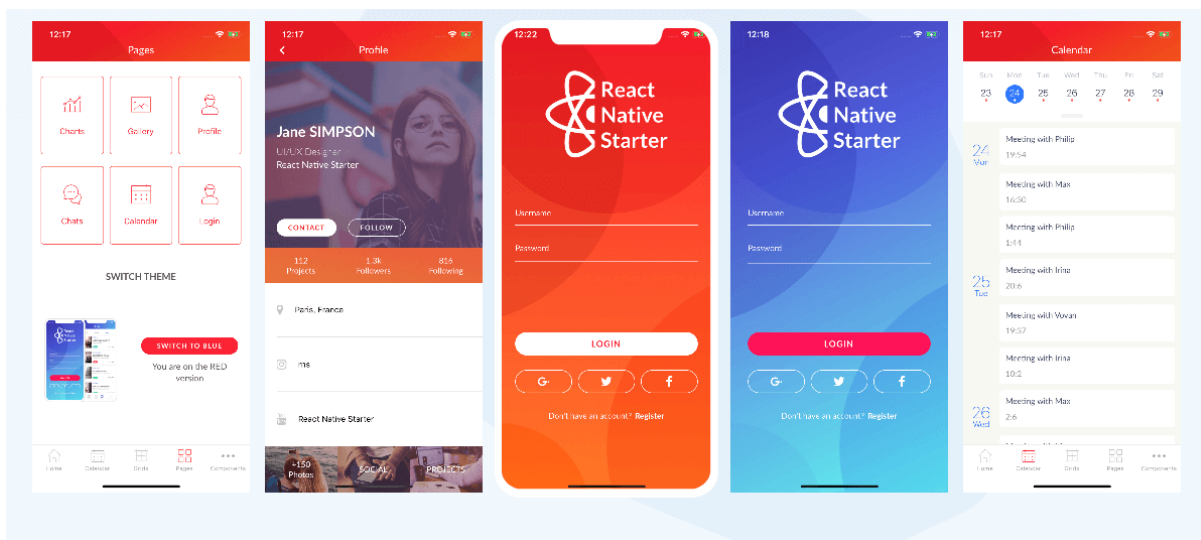
Để bắt đầu một dự án mới, bạn có thể bắt đầu từ con số bằng cách khởi tạo bằng React Native CLI.

Tuy nhiên, nếu dự án cần gấp, deadline bị dí sát gáy rồi, khách hàng muốn có sản phẩm demo ngay và luôn. Chưa kể, bạn cũng không muốn phải “phát minh lại các bánh xe”, khi các chức năng phổ thông, các UI component phổ biến đã được thiết kế sẵn, lại rất đúng chuẩn UX nữa. Đó là lúc bạn nghĩ tới sử dụng các bộ RN Starter KIT.

Starter Kit thực chất là tuyển tập những component/template người ta đã viết sẵn, việc của bạn chỉ là gọi ra và sử dụng.

Các starter kit đã viết sẵn style cho Button, Edittext, Checkbox... Thậm chí cả giao diện cho các màn hình phổ biến như: Chat, login, Profiles...

Các components/template này được tối ưu, thiết kế rất đẹp, tương thích với style của cả Android và iOS.



Phần này, mình sẽ giới thiệu một số bộ React Native Starter Kit miễn phí, được nhiều người tin dùng, trong đó có cả một bộ template do chính VNTALKING phát triển.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách **[tại đây](#)** để đọc bản đầy đủ nhé.

ReactNative Full Template

[ReactNative Full Template](#) Đây là một dự án đầy tham vọng của mình và các bạn VNTALKING. Thông qua dự án mã nguồn mở này, mình muốn tạo một bộ khung với đầy đủ các tính năng phổ biến như:

- Hỗ trợ đầy đủ 3 loại navigator: Stack Navigator, Tab Navigator và Drawer Navigator được phân chia khoa học.
- Tích hợp sẵn Redux + saga để quản lý gọi API.

- Tích hợp sẵn cơ chế quản lý Authentication.
- Tích hợp bộ component Native Base, với những component phổ biến dùng được ngay.
- Và còn nhiều nữa, dự án đang đợi những đóng góp của bạn.

Công việc của bạn là tải về và đổi tên package là chiến 😊

Ngoài ra, Nếu bạn muốn tham gia đóng góp vào dự án, đừng ngần ngại tạo pull request nhé. <https://github.com/vntalking/ReactNative-Full-Template/pulls>

15

DỰ ÁN MOBILE CUỐI KHÓA

Xin chúc mừng bạn đã hoàn thành xong phần lý thuyết về lập trình ứng dụng di động bằng React Native.

Để những kiến thức lý thuyết đó không phải là lý thuyết suông, cũng như luyện tập khả năng thực chiến trước khi apply vào các công ty phần mềm hàng đầu, phần bài tập cuối khóa này như một điểm nhấn trong CV của bạn đấy.

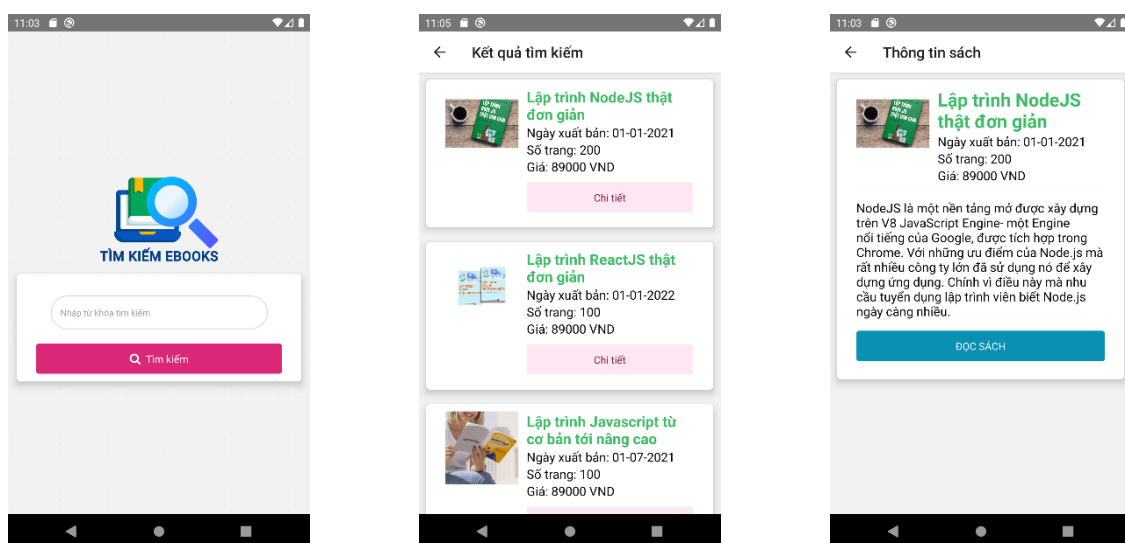
Có rất nhiều ý tưởng về một ứng dụng di động, bạn cứ thử suy nghĩ, rồi liệt kê các tính năng rồi biến nó thành hiện thực. Sau khi hoàn thành, bạn có thể gửi sản phẩm về VNTALKING qua địa chỉ email support@vntalking.com để mình có thể đánh giá, gợi ý và giúp bạn hoàn thiện sản phẩm hơn.

Dưới đây là một ý tưởng xây dựng ứng dụng mobile mà bạn có thể tham khảo và thực hành theo mình: ***Dự án xây dựng ứng dụng tìm kiếm Ebooks trên Internet.***

Các tính năng chính:

- Tìm kiếm Ebooks theo từ khóa.
- Hiển thị kết quả tìm được.
- Hiển thị thông tin chi tiết một cuốn Ebook.
- Sử dụng Google Book API làm search engine.
- Placeholder components.
- Giao diện đẹp bắt mắt.
- Và còn nhiều hơn thế nữa... (đang chờ bạn đóng góp).

Giao diện ứng dụng dự kiến như sau:



Chúng ta sẽ cùng nhau thực hành xây dựng ứng dụng này nhé.



Đây chỉ là bản demo - phần này còn nữa
Đặt sách **tại đây** để đọc bản đầy đủ nhé.

KẾT NỐI VỚI VNTALKING

Một lần nữa, VNTALKING đánh giá rất cao sự nỗ lực của bạn, được minh chứng bởi việc bạn đã đọc hết cuốn sách và đọc đến tận trang sách này. Cảm ơn bạn rất nhiều ^_^

Đặc biệt, VNTALKING cũng rất vui khi được đồng hành cùng bạn trên con đường học để trở thành một lập trình viên chuyên nghiệp nói chung và Javascript nói riêng.

Vì vậy, bất cứ khi nào bạn cần tư vấn, có thắc mắc hay khó khăn hãy "trút bầu tâm sự" với VNTALKING.

Liên hệ với VNTALKING bằng bất kỳ hình thức nào dưới đây.

- Website: <https://vntalking.com>
- Fanpage: <https://facebook.com/vntalking>
- Group: <https://facebook.com/groups/hoidaplaptrinh.vntalking>
- Email: support@vntalking.com

THÔNG TIN TÁC GIẢ

VNTALKING.COM là một website được thành lập từ 25/12/2016 và đang được vận hành bởi Dương Anh Sơn (một developer “kì cựu” – chuẩn bị về quê chăn lợn).

Bọn mình luôn hướng tới một trải nghiệm miễn phí mà hiệu quả. VNTALKING gồm những thành viên luôn muốn đem đến cho độc giả những kiến thức, kinh nghiệm thực tiễn, được cập nhật nhanh nhất. Đồng hành cùng VNTALKING để khám phá niềm đam mê lập trình trong bạn.

Thông tin thêm về tác giả.



Tên đầy đủ là Dương Anh Sơn, gọi tắt là Sơn Dương (^_^). Tốt nghiệp ĐH Bách Khoa Hà Nội. Mình bắt đầu nghiệp coder khi ra trường không xin được việc đúng chuyên ngành. Mình tin rằng chỉ có chia sẻ kiến thức mới là cách học tập nhanh nhất.

CUỐN SÁCH CỦA VNTALKING

Đến thời điểm hiện tại, VNTALKING đã hoàn thành 4 dự án sách học lập trình. Sách học React Native này là cuốn tiếp theo mà VNTALKING thực hiện.

Nếu muốn tìm hiểu nhiều hơn về Back-end hoặc Front-end, mời bạn tham khảo cuốn sách:



Lập trình Node.JS thật đơn giản

[Đọc sách](#)



Lập trình React thật đơn giản

[Đọc sách](#)



Lập trình Javascript từ cơ bản tới nâng cao

[Đọc sách](#)

Hi vọng trong thời gian tới, VNTALKING sẽ tiếp tục nhận được sự ủng hộ của độc giả. Thành công của bạn chính là động lực để VNTALKING cho ra nhiều cuốn sách với chất lượng tốt hơn nữa, đáp ứng nhu cầu học lập trình của mọi người.