Table of Contents

[Currency Converter POC – A React Native Project 1](#_Toc502341924)

[Objective 1](#_Toc502341925)

[New Project Setup 1](#_Toc502341926)

[Project Folder Structure In Depth 4](#_Toc502341927)

[`.babelrc` 4](#_Toc502341928)

[`.gitignore` 4](#_Toc502341929)

[`package.json` 4](#_Toc502341930)

[`app.json` 5](#_Toc502341931)

[Visual Studio Code Setup 5](#_Toc502341932)

[Prettier Extension 5](#_Toc502341933)

[ESLint Extension 6](#_Toc502341934)

[Currency Convert Starter POC Project 7](#_Toc502341935)

[Video Tutorials Links 7](#_Toc502341936)

[App Design Links 7](#_Toc502341937)

[Terminology 7](#_Toc502341938)

[TDD With React Native 8](#_Toc502341939)

[Concept - Tools 8](#_Toc502341940)

[Setup 8](#_Toc502341941)

[Tests 8](#_Toc502341942)

# Currency Converter POC – A React Native Project

## Objective

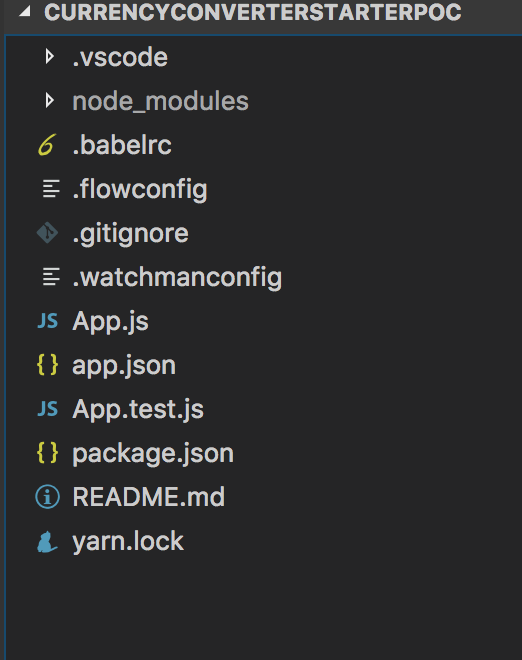
* Demonstrate the ability to develop your mobile app for both iOS and Android all within one single codebase.
* Demonstrate the ability to upload your app to Expo and test it right on your phone without the need to upload it to Google/Apple App Store.
* Demonstrate the folder structure the function of each individual components
* Show how to do TDD and BDD locally.
* Show Package Management and VSCode Setup.

## Getting Started

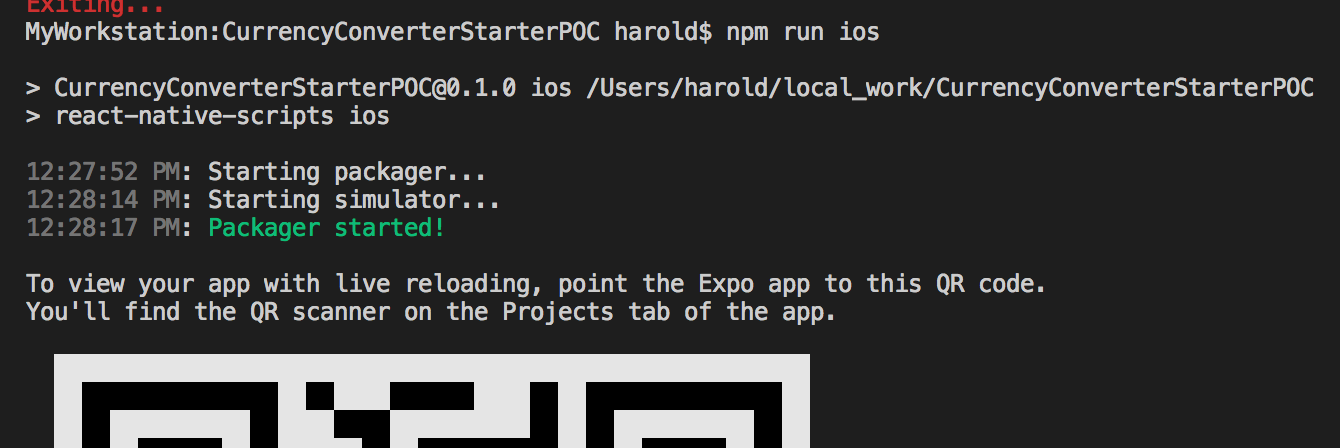
* Watch [React Redux Video](https://egghead.io/lessons/react-redux-the-single-immutable-state-tree)
* Read [Redux Doc](https://redux.js.org/)

## New Project Setup

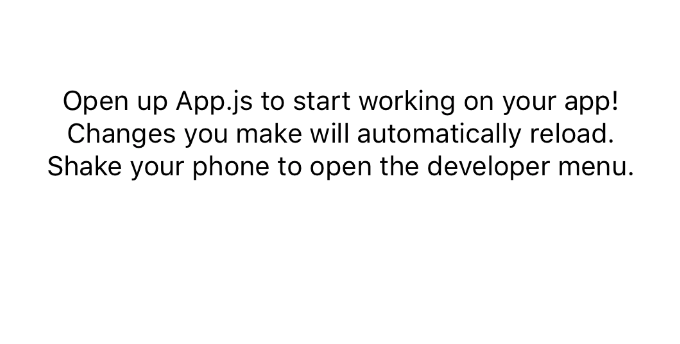
* Create a project
  + ` create-react-native-app CurrencyConverterStarterPOC `
    - It will automatically create a project for you and install all of the dependencies and set up package.json



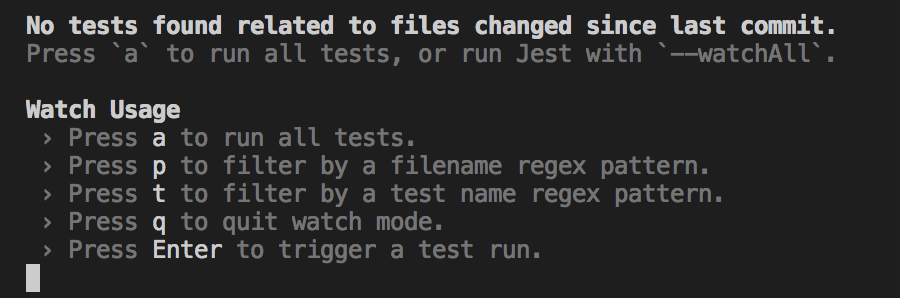
* Run the project
  + `npm run ios` ( if android, do `npm run android`)
  + You should something similar to below in your terminal



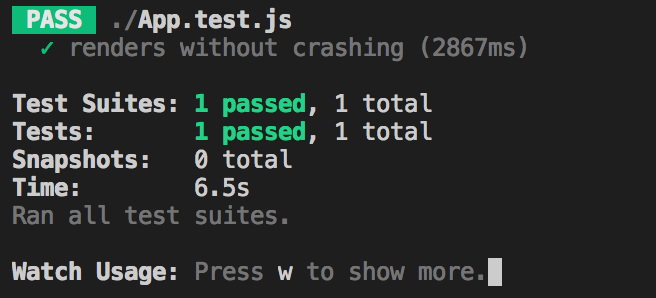
* + You should see something similar to below in your emulator or iphone with Expo client.



* Run a test
  + `npm run test`
    - run Jest in watch mode
  + You should something similar to below in your terminal



* + - Press `a` to run all tests.
    - Keep this window open and it will run all tests whenever it detects any changes in your test files or new test files are created.



* + - Check mark hints a specific test case with the name ‘render without crashing) is passed in 2.8 seconds.

## Project Folder Structure In Depth

* If you are experienced with `create-react-native-app`, you can skip to the next section

### `.babelrc`

* A file used by Babel JavaScript transpiler



* ***A preset* is a set of *plugins* used to support particular language features**
  + `babel-preset-expo` will contain react-native preset , module-resolver for expo vector icons, transform plugins, etc., that’s needed for expo
  + ` transform-react-jsx-source` plugin will add source file and line number to JSX elements.
  + `babel --plugins transform-react-jsx-source App.js | js-beautify`



### `.gitignore`

* prevents expo, log and config related files to be committed to git

### `package.json`

* Versioning for Create React Native App
  + Follow the versioning listed or else things will break.
  + [expo-react-native-dependency-versions](https://github.com/react-community/create-react-native-app/blob/master/VERSIONS.md)
* `devDependencies`
  + jest-expo
    - A preset (A plugin to support particular language feature) to painlessly test Expo apps.
* `dependencies`
  + Versions must match ones stated in

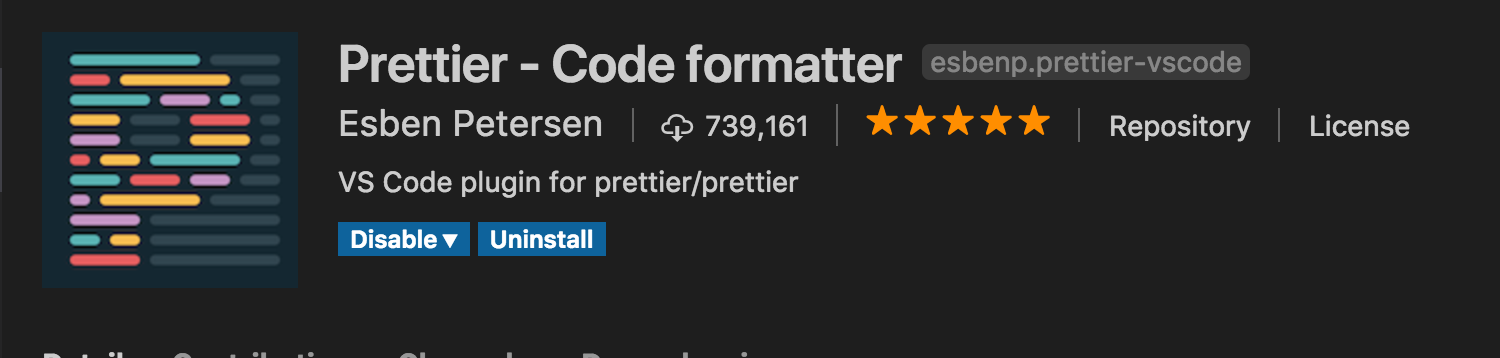
### `app.json`

* `expo`
  + sdkVersion that match to [expo-react-native-dependency-versions](https://github.com/react-community/create-react-native-app/blob/master/VERSIONS.md)
  + Expo apps are React Native apps which contain the [Expo SDK](https://docs.expo.io/versions/latest/sdk/index.html#expo-sdk). The SDK is a native-and-JS library which provides access to the device’s system functionality (things like the camera, contacts, local storage, and other hardware). That means you don’t need to use Xcode or Android Studio, or write any native code, and it also makes your pure-JS project very portable because it can run in any native environment containing the Expo SDK.

## Visual Studio Code Setup

### Prettier Extension

* Problem Statement
  + Working on a JavaScript project on a code editor can be messy because it doesn’t always provide a specific style guide or code format and if you do, you have to format it manually like from Eclipse or Intellij (Ctrl+Shift+F)
* Solver
  + Use prettier plugin and configure VSCode so that it will automatically format your code when you save.
* Get `Prettier – Code Formatter` plugin from `extensions:marketplace`



* Add settings.json file inside to your .vscode folder

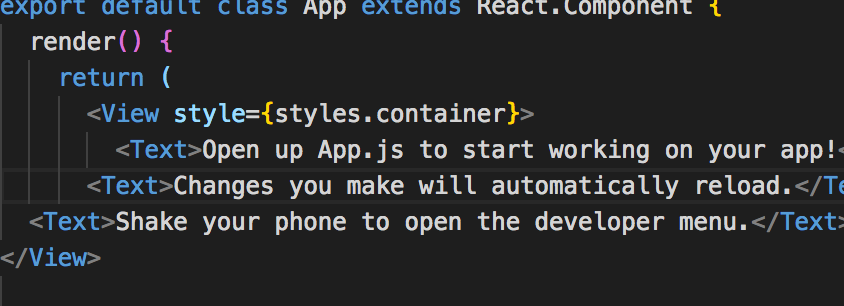
{

"prettier.eslintIntegration": true,

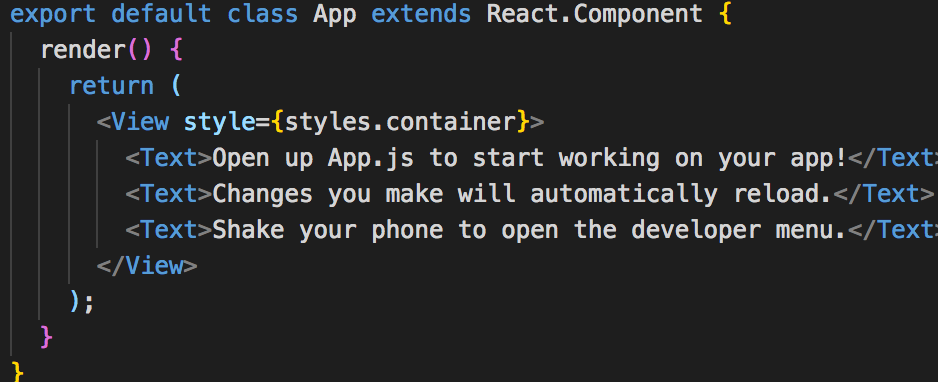
"editor.formatOnSave": true

}

* Now
  + when you do mess up your code like

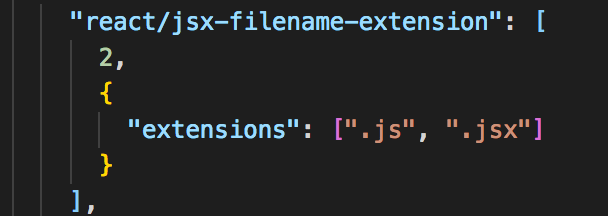


* + You hit save, you get a formatted code look like



### ESLint Extension

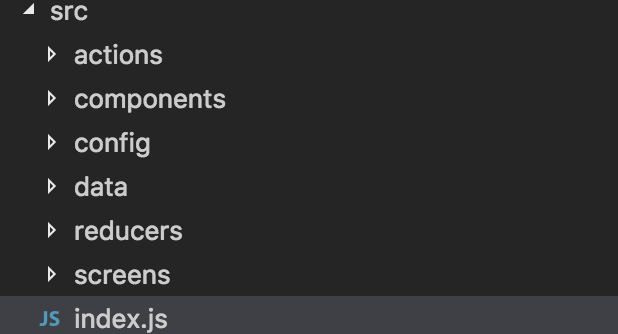
* What is ESLint
  + A tool used for basic syntax validation, mix and match the bundled rules and your custom rules.
  + [Reference URL](https://eslint.org/docs/user-guide/configuring)
* What’s Configurable
  + Environments
    - A set of environments your script is designed to run.
    - Each environment brings with it a certain set of predefined global variables
  + Globals
    - Additional global variables your script accesses during execution
  + Rules
    - Defines what rule are enabled at what error level
* `.eslintrc.json`
  + Uses Airbnb eslint config
  + Uses babel-eslint parser
    - Allows you to lint **ALL** valid Babel code with the fantastic ESLint.
    - Only necessary if you are using types (Flow) or experimental feature not supported in ESLint.
  + Rules
    - `react/jsx-filename-extension`
      * react plugin has a which defines what file acceptable and raise error otherwise



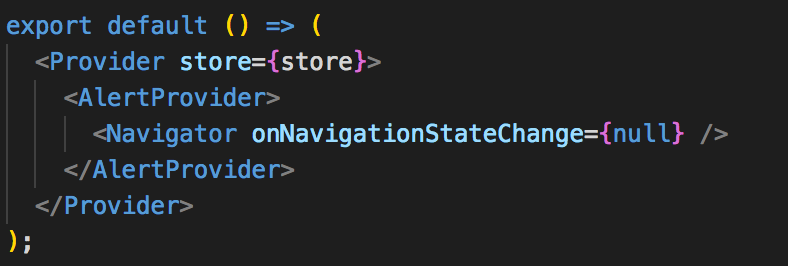
* + - `global-require`: [0]
      * disable require at the global level
    - ` react/require-default-props`: [0]
      * disable default props requirement when you create your react component.
        + `defaultProps={prop1:[], prop2:[]}`
    - `react/forbid-prop-types`: [0]
      * to disable warnings on some prop types such as object.
  + [React Plugin Reference Page](https://github.com/yannickcr/eslint-plugin-react)

## Currency Convert Starter POC Project

### Source Folder Layout

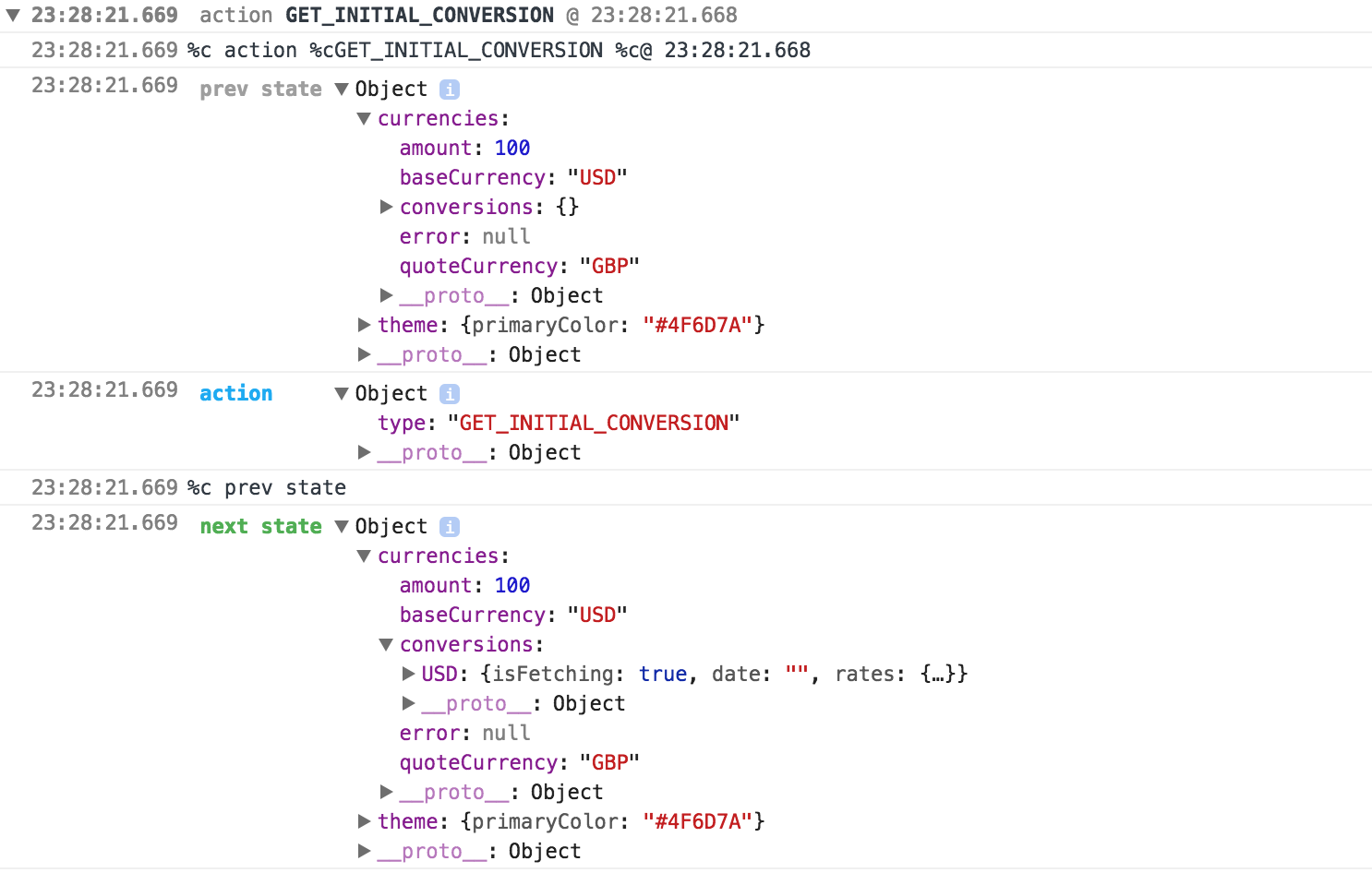


* actions
  + Action to be dispatched within redux
* Components
  + Re-useable components for wiring up different screens
* Config
  + Routing for different view screens.
  + Stores for store middleware and logging concerns.
  + Sagas for side-effect such as asynchronous data fetching and impure things like accessing the browser cache.
* Data
  + Fixed data for all different currencies – used for making async request when user changes currency for conversion. (It’s probably better to extract this logic as an API call to get the currency data)
* Reducers
  + Takes a state and an action and returns a new state.
* Screens
  + Screens that user will be able to see.
* `index.js`
  + The entry point to the App with react-redux bridge Provider interfacing redux store with reducers
  + Navigator for navigating between views surrounded by AlertProvider providing alerts if action dispatched is unsuccessful.



* + - Provider component from react-redux takes a store, which contains reducers (in app) and middlewares like logging (dev env) and sagas (async req).
    - AlertProvider provides dropdown alert
    - Navigator provides routing and navigation for different screens

### App State Transition – Will Go Over

1. 
2. 
3. 

### Expo

* Publish to Exp
  + `Exp publish`

### TODOS

* [Redux Integration](https://reactnavigation.org/docs/guides/redux)

Addendum

## Publishing Your Expo App

* [Expo Quick Start](https://docs.expo.io/versions/latest/introduction/project-lifecycle.html)

## Video Tutorials Links

* [getting-started-with-redux](https://egghead.io/courses/getting-started-with-redux)
* [building-react-applications-with-idiomatic-redux](https://egghead.io/courses/building-react-applications-with-idiomatic-redux)

## Starter Links

* [ReactJS Introduction](https://reactjs.org/docs/hello-world.html)

## App Design Links

* [Presentational and Container Components](https://medium.com/@dan_abramov/smart-and-dumb-components-7ca2f9a7c7d0)
* [Redux Basics](https://github.com/reactjs/redux/tree/master/docs/basics)

## Terminology

* [Babel](https://babeljs.io/): A Javascript compiler to transpile our javascript code so that it is compatible everywhere.
* [Mocha](https://mochajs.org/): A Javascript testing framework running on [node.js](https://nodejs.org/en/) to run our tests.
* [Chai](http://chaijs.com/) : A library that provides us with interfaces to write assertions in our test.
* [Sinon](http://sinonjs.org/): A library that provides us with spies, stubs and mocks which are used extensively while testing.
* [React Native Mock](https://github.com/lelandrichardson/react-native-mock) : Library that provides a completely mocked version of react-native that is easily testable.
* [Enzyme](http://airbnb.io/enzyme/): A React test utility that helps us to write painless tests for react components. JavaScript Testing utility for React that makes it easier to assert, manipulate, and traverse your React Components.

## TDD With React Native

* This branch created after Visual Studio Code Setup

### Concept - Tools

* When writing and running tests, we generally need three things
  + A test runner
    - Finds and executes your tests and reports on the output
  + A test framework
    - Some way of defining, organizing and describing tests
  + An assertion library
    - A set of convenience methods used to assert result from expectation
* Testing Tools
  + Jest - Provides all three of the above
* Other Testing tools
  + Enzyme, Mocha, Jasmine, AVA, Tape.

### Setup

* ` npm install --save prop-types`
  + Adds prop type check for components
* ` npm install enzyme chai sinon enzyme-adapter-react-15 --save-dev`
  + Set up enzyme and chai for better test and assertion

## Test References:

### https://github.com/reactjs/redux/blob/master/docs/recipes/WritingTests.md