Get Started with Windows

Edit

This guide will help you get started on setting up your very first React Native for Windows app.

Make sure you have installed all of the development dependencies.

For information around how to set up React Native, see the React Native Getting Started Guide.

Install React Native for Windows

Remember to call `react-native init` from the place you want your project directory to live.

```
npx react-native init <projectName> --version "latest"
```

Copy

Navigate into this newly created directory

Once your project has been initialized, React Native will have created a new sub directory where all your generated files live.

```
cd projectName
```

Copy

Install the Windows extension

> The --overwrite flag copies a custom `metro.config.js` file. If you are starting a new app, this should have no impact. If you are adding Windows to your existing app and you have modified the `metro.config.js` file, please back up your changes, run the command and copy over to take effect.

For information on the options that `react-native-windows-init` takes see React Native Windows Init CLI.

Running a React Native Windows App

> Make sure a browser is launched and running before running a React Native Windows app. Also ensure your system meets all the requirements to build a Windows app as well.

- Without Using Visual Studio

  In your React Native Windows project directory, run:

  ```
  npx react-native run-windows
  ```

  Copy

  For information on the options that `@react-native-windows/cli` takes see React Native Windows CLI.

  A new Command Prompt window will open with the React packager as well as a `react-native-windows` app. This step may take a while during first run since it involves building the entire project and all dependencies. You can now start developing! :tada:

- Using Visual Studio

```
npx react-native autolink-windows
```

- Open the solution file in the application folder in Visual Studio (e.g., `AwesomeProject/windows/AwesomeProject.sln` if you used `AwesomeProject` as `<projectName>`)
- Select the `Debug` configuration and the `x64` platform from the combo box controls to the left of the `Run` button and underneath the `Team` and `Tools` menu item.
- Run `yarn start` (or `npm start`) from your project directory, and wait for the React Native packager to report success.
- Click the `Run` button to the right of the platform combo box control in VS, or select the `Debug -> Start without Debugging` menu item. You now see your new app and Chrome should have loaded `http://localhost:8081/debugger-ui/` in a new tab. Press `F12` or `Ctrl+Shift+I` in Chrome to open its Developer Tools. :tada:

- With VS Code
  - Open your applications folder in VS Code.
  - Install the React Native Tools plugin for VS Code.
  - Create a new file in the applications root directory, `.vscode/launch.json` and paste the following configuration:

Copy

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Debug Windows",
      "cwd": "${workspaceFolder}",
      "type": "reactnative",
      "request": "launch",
      "platform": "windows"
    }
```

and in the Debug drop-down select "Debug Windows" and press the green arrow to run the application.

Authoring Native Modules

See Native Modules and React Native Windows.

Building a standalone React Native Windows App

Follow these steps to build a version of your app that you can install or publish to the store. This version will package your bundle and assets into the APPX package so you don't need to run Metro.

- Open the solution in Visual Studio
- Select the Release configuration from the Configuration Manager drop-down.
- Build the solution. You can now launch without first launching Metro.
- If you want to build an APPX package to share or publish, use the **Project** > **Publish** > **Create App Packages...** option.

> The Debug configuration uses the Web Debugger by default, which means the application's JavaScript code runs in Chrome.
> If you're getting different runtime behavior between the Release and Debug configurations, consider disabling the `UseWebDebugger` setting in `App.cpp` or `App.xaml.cs` to get the same behavior in the Debug configuration.

See also this article for additional details: https://techcommunity.microsoft.com/t5/windows-dev-appconsult/getting-started-with-react-native-for-windows/ba-p/912093#

REACT NATIVE DOCS

Getting Started

Tutorial

Components and APIs

More Resources

REACT NATIVE FOR WINDOWS + MACOS DOCS

Get Started with Windows

Get Started with macOS

React Native Windows Components and APIs

Native Modules

Native UI Components

CONNECT WITH US ON

Blog

Twitter

GitHub

Samples