Upgrading to new versions

Upgrading to new versions of React Native will give you access to more APIs, views, developer tools and other goodies. Upgrading requires a small amount of effort, but we try to make it straightforward for you.

Expo projects

Upgrading your Expo project to a new version of React Native requires updating the `react-native`, `react`, and `expo` package versions in your `package.json` file. Expo provides an `upgrade` command to handle upgrading these and any other known dependencies for you. See the Upgrading Expo SDK Walkthrough for up-to-date information about upgrading your project.

React Native projects

Because typical React Native projects are essentially made up of an Android project, an iOS project, and a JavaScript project, upgrading can be rather tricky. There's currently two ways for upgrading your React Native project: by using React Native CLI or manually with Upgrade Helper.

React Native CLI

The React Native CLI comes with `upgrade` command that provides a one-step operation to upgrade the source files with a minimum of conflicts, it internally uses rn-diff-purge project to find out which files need to be created, removed or modified.

1. Run the `upgrade` command

> The `upgrade` command works on top of Git by using `git apply` with 3-way merge, therefore it's required to use Git in order for this to work, if you don't use Git but still

> want to use this solution then you can check out how to do it in the Troubleshooting section.

Run the following command to start the process of upgrading to the latest version:

```
npx react-native upgrade
```

You may specify a React Native version by passing an argument, e.g. to upgrade to `0.61.0-rc.0` run:

```
npx react-native upgrade 0.61.0-rc.0
```

The project is upgraded using `git apply` with 3-way merge, it may happen that you'll need to resolve a few conflicts after it's finished.

2. Resolve the conflicts

Conflicted files include delimiters which make very clear where the changes come from. For example:

```
13B07F951A680F5B00A75B9A /* Release */ = {
  isa = XCBuildConfiguration;
  buildSettings = {
    ASSETCATALOG_COMPILER_APPICON_NAME = AppIcon;
<<<<<<< ours
    CODE_SIGN_IDENTITY = "iPhone Developer";
    FRAMEWORK_SEARCH_PATHS = (
      "$(inherited)",
      "$(PROJECT_DIR)/HockeySDK.embeddedframework",
      "$(PROJECT_DIR)/HockeySDK-iOS/HockeySDK.embeddedframework",
    );
=======
    CURRENT_PROJECT_VERSION = 1;
>>>>>>> theirs
    HEADER_SEARCH_PATHS = (
      "$(inherited)",

/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/include,
      "$(SRCROOT)/../node_modules/react-native/React/**",
```

```
    "$(SRCROOT)/../node_modules/react-native-code-push/ios/CodePush/**",
  );
```

You can think of "ours" as "your team" and "theirs" as "the React Native development team".

Upgrade Helper

The Upgrade Helper is a web tool to help you out when upgrading your apps by providing the full set of changes happening between any two versions. It also shows comments on specific files to help understanding why that change is needed.

1. Select the versions

You first need to select from and to which version you wish to upgrade, by default the latest major versions are selected. After selecting you can click the button "Show me how to upgrade".

💡 Major updates will show an "useful content" section on the top with links to help you out when upgrading.

2. Upgrade dependencies

The first file that is shown is the `package.json`, it's good to update the dependencies that are showing in there. For example, if `react-native` and `react` appears as changes then you can install it in your project by running `yarn add`:

```
# {{VERSION}} and {{REACT_VERSION}} are the release versions showing in the diff
yarn add react-native@{{VERSION}}
yarn add react@{{REACT_VERSION}}
```

3. Upgrade your project files

The new release may contain updates to other files that are generated when you run `npx react-native init`, those files are listed after the `package.json` in the Upgrade Helper

page. If there aren't other changes then you only need to rebuild the project to continue developing.

In case there are changes then you can either update them manually by copying and pasting from the changes in the page or you can do it with the React Native CLI upgrade command by running:

```
npx react-native upgrade
```

This will check your files against the latest template and perform the following:

- If there is a new file in the template, it is created.
- If a file in the template is identical to your file, it is skipped.
- If a file is different in your project than the template, you will be prompted; you have options to keep your file or overwrite it with the template version.

> Some upgrades won't be done automatically with the React Native CLI and require manual work, e.g. `0.28` to `0.29`, or `0.56` to `0.57`. Make sure to check the release notes when upgrading so that you can identify any manual changes your particular project may require.

Troubleshooting

I want to upgrade with React Native CLI but I don't use Git

While your project does not have to be handled by the Git versioning system -- you can use Mercurial, SVN, or nothing -- you will still need to install Git on your system in order to use `npx react-native upgrade`. Git will also need to be available in the `PATH`. If your project doesn't use Git, initialize it and commit:

```
git init # Initialize a Git repository
git add . # Stage all the current files
git commit -m "Upgrade react-native" # Save the current files in a commit
```

After you finish upgrading you may remove the `.git` directory.

I have done all the changes but my app is still using an old version

These sort of errors are usually related to caching, it's recommended to install react-native-clean-project to clear all your project's cache and then you can run it again.

Is this page useful? 

Edit this page

*Last updated on **Jun 21, 2023***