Update your React Native apps seamlessly using Microsoft's CodePush

#react #reactnative #javascript #devops

Karan Pratap Singh May 13, 2020 · 5 min read

Greetings React Native Community, today I'll be helping you setup Microsoft's Codepush into your app. So that you can do seamless releases.

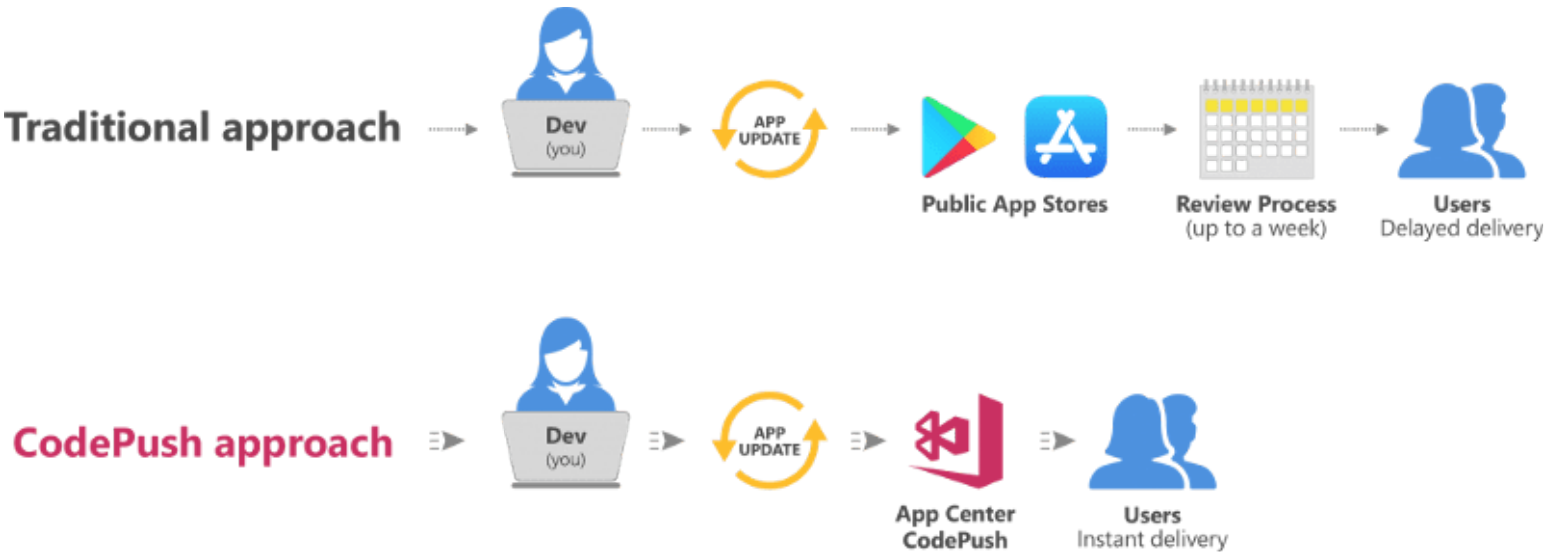
What is CodePush?

CodePush a technology that helps in the delivery of app updates and improvements to the end users instantly.

This is especially great if you want to do critical bug fixes and deliver instantly without going through the app store reviews.

You can think of it as "web-like" agility of side-loading updates as soon as they are available.

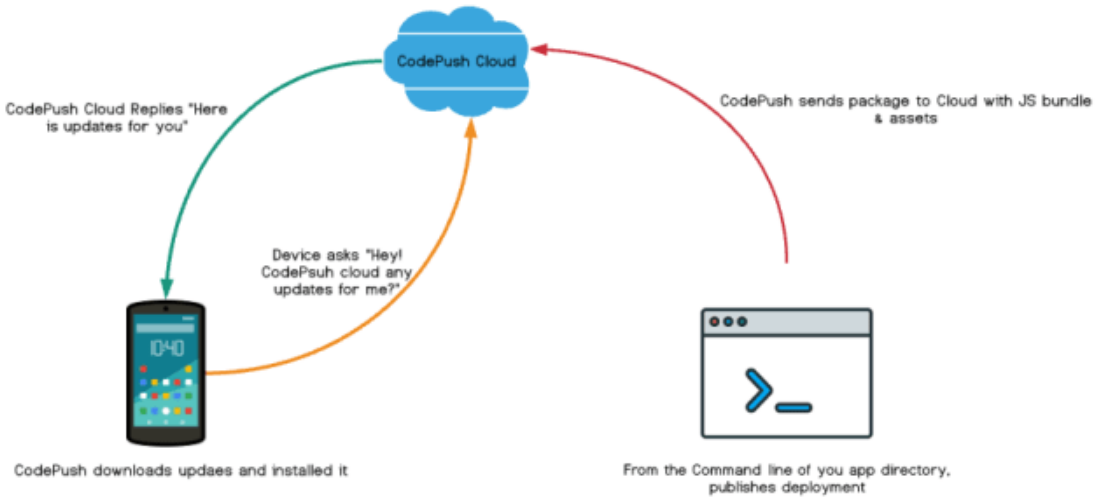
Moreover, it provides rollbacks if the new update crashed the app



How does it work?

CodePush keeps your app's javascript bundle in sync with the CodePush server, and every time the user opens the app it checks with the CodePush server if a new update is available to the bundle. And of course, it comes with tons of awesome configuration which can help us fine-tune our user's experience.

I personally use CodePush in almost all the React Native projects I work with as it is a very



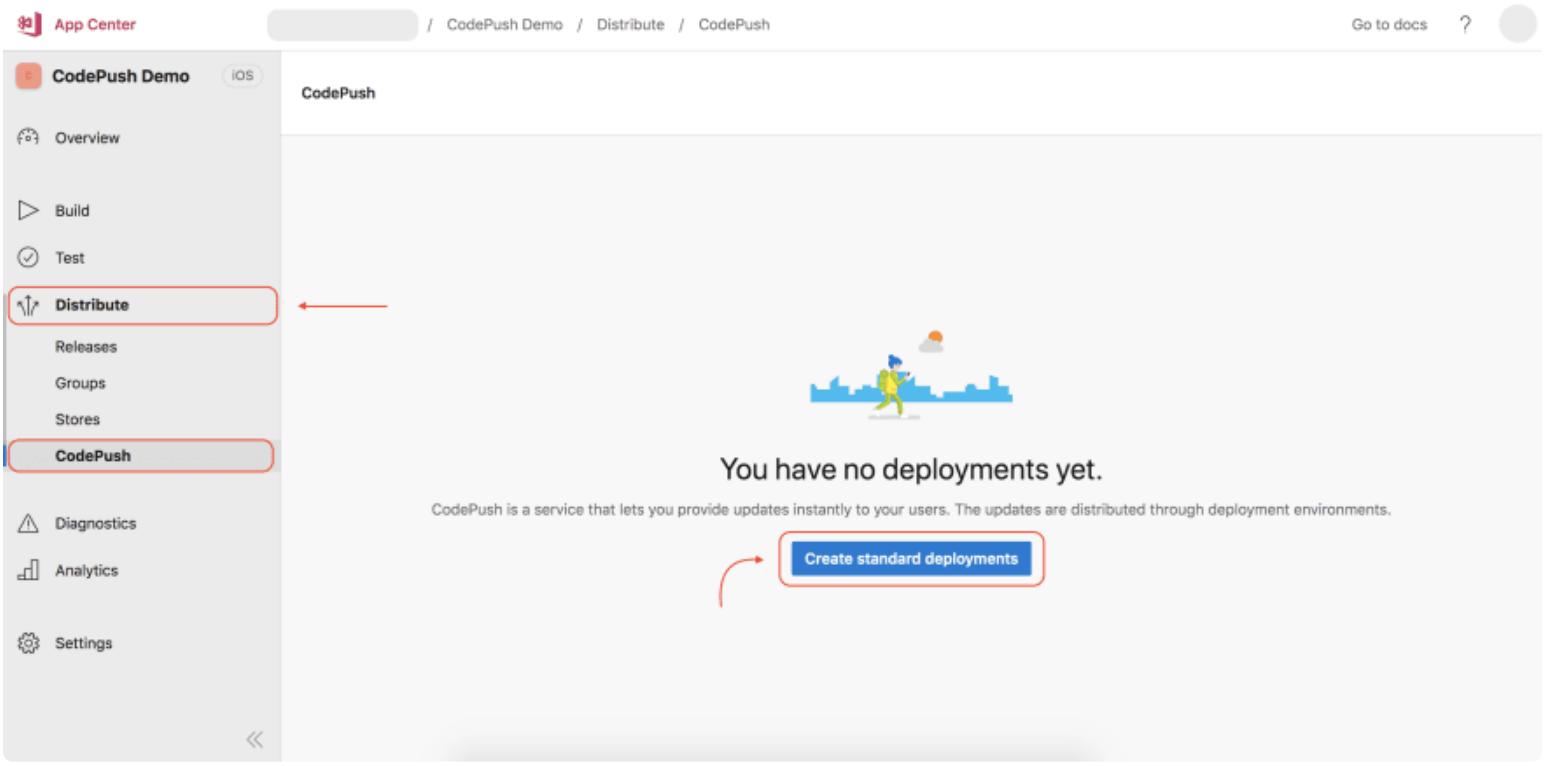
You can read more about it here

Let's get started 🚀

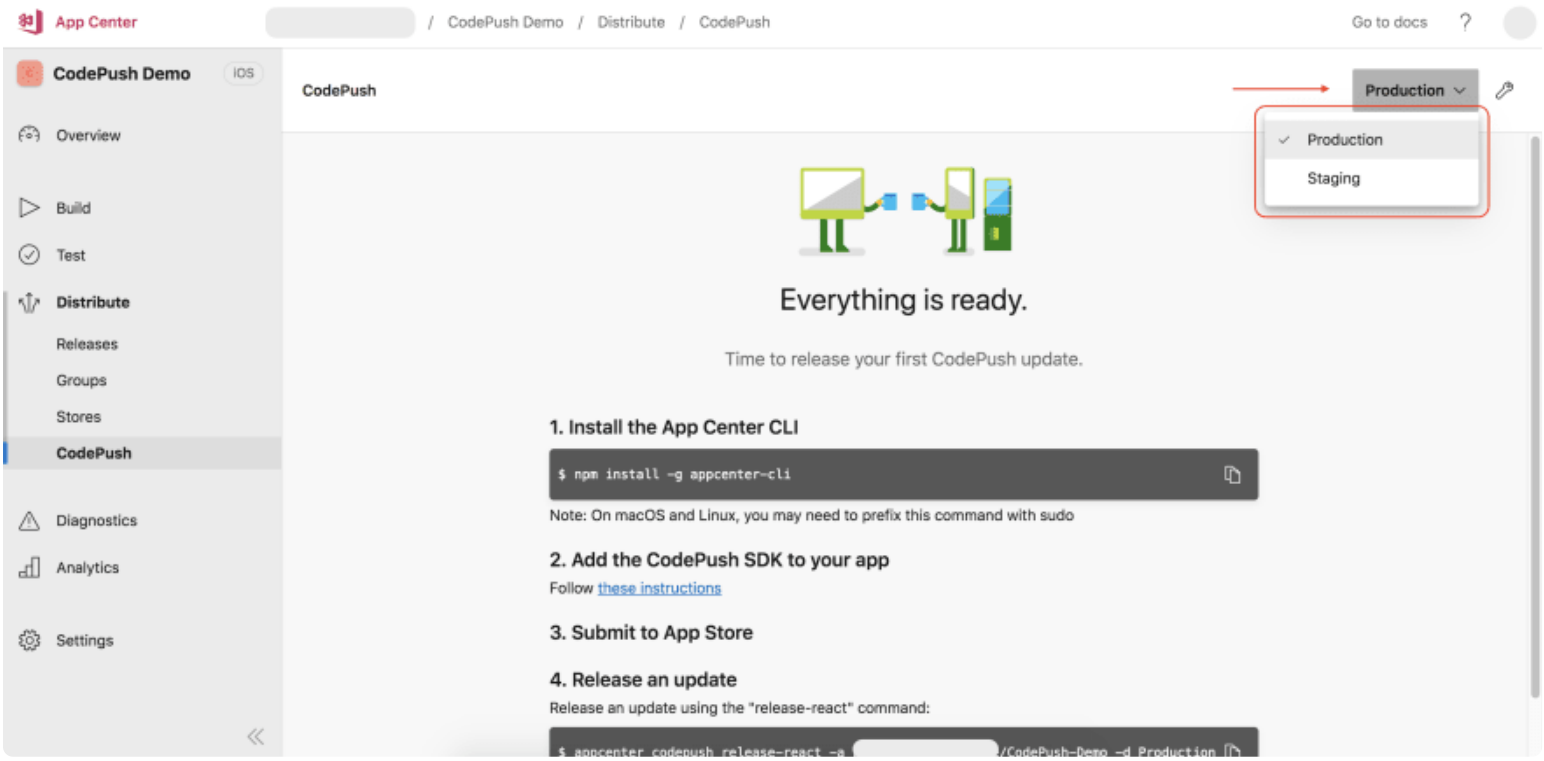
Let's get started by first creating standard deployments for CodePush in AppCenter.

I'll be assuming that you already know how to log in with AppCenter and create or link a new Android/iOS app, if you don't then please check out adding/linking part of this guide here

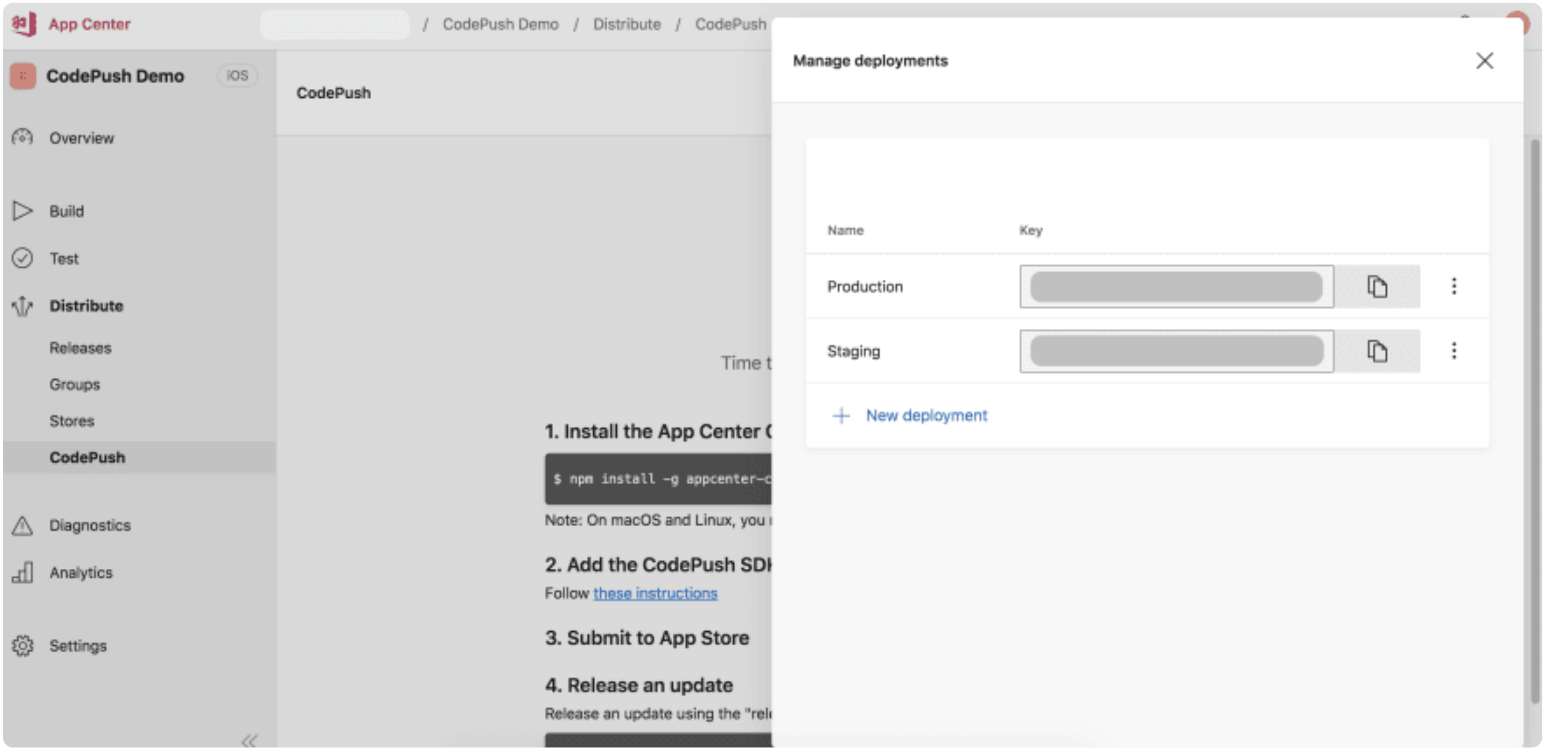
- Navigate to `Codepush` under `Distribute` and click on `Create Standard Deployment`



- Now, to the top right you should be able to select your environment



- Click on the settings items at the top right and keys panel should open reveling your keys (we'll be needing them later)



Integration

With the keys now available, let's integrate CodePush into our apps. For this we'll need to install [react-native-code-push]

```
yarn add react-native-code-push
```

```
npm i --save react-native-code-push
```

Android

In this section, we'll see how to integrate CodePush plugin with our native android project.

- In your `android/settings.gradle`, add the following:

```
include ':app', ':react-native-code-push'
project(':react-native-code-push').projectDir = new File(rootProject.projectDir, '../nod
```

- In your `android/app/build.gradle`, add the `codepush.gradle` file as an additional build task definition underneath `react.gradle`

```
...
apply from: "../../node_modules/react-native/react.gradle"
apply from: "../../node_modules/react-native-code-push/android/codepush.gradle"
...
```

- Update the `MainApplication.java` file to use CodePush via the following changes:

```
...
// 1. Import the plugin class.
import com.microsoft.codepush.react.CodePush;

public class MainApplication extends Application implements ReactApplication {

    private final ReactNativeHost mReactNativeHost = new ReactNativeHost(this) {
        ...

        // 2. Override the getJSBundleFile method in order to let
        // the CodePush runtime determine where to get the JS
        // bundle location from on each app start
        @Override
        protected String getJSBundleFile() {
            return CodePush.getJSBundleFile();
        }
    };
}
```

- Optional: You can add key in `android/app/src/main/res/values/strings.xml` file like this or You can also skip adding deployment key here as you can dynamically override it via js (isn't that amazing 😆), which we'll get to soon.

```
  <resources>
    <string name="app_name">AppName</string>
    <string moduleConfig="true" name="CodePushDeploymentKey">DeploymentKey</string>
```



Note: you can checkout the official CodePush android docs here for more in depth look.

iOS

In this section, we'll see how to integrate CodePush plugin with our native iOS project.

- Run `cd ios && pod install && cd ..` to install all the necessary CocoaPods dependencies.
- Open up the `ios/<Your-Project>/AppDelegate.m` file, and add an import statement for the CodePush headers:

```
  #import <CodePush/CodePush.h>
```

- Find the following line of code, which sets the source URL for bridge for production releases:

```
  return [[NSBundle mainBundle] URLForResource:@"main" withExtension:@"jsbundle"];
```

And Replace it with this line:

```
  return [CodePush bundleURL];
```

- Optional: Go to `ios/<Your-Project>/Info.plist` and add a new key as `CodePushDeploymentKey` of type `string` and add your iOS key.

```
<key>CodePushDeploymentKey</key>
<string>DeploymentKey</string>
```

Note: you can checkout the official CodePush ios docs here for more in depth look.

Initialization

In this section we'll be following a simple example for initializing our CodePush plugin as there's no way I can do justice to all the options and configuration available in this plugin, so make sure to checkout the official CodePush js api reference here

```
import codePush from 'react-native-code-push';

...

const codePushOptions = {
  installMode: codePush.InstallMode.IMMEDIATE,
  deploymentKey: "<YOUR KEY HERE>",
  checkFrequency: codePush.CheckFrequency.ON_APP_START,
};

export default codePush(codePushOptions)(App);
```

```
rm -rf ios/build android/app/build
```

```
yarn start -c

# or if you use npm

npm start --reset-cache
```

Deployments

As our app is now ready to use CodePush, let's now look into how we'll be releasing updates. For this we'll need [appcenter-cli]

```
yarn global add appcenter-cli
```

Or if you prefer npm then,

```
npm i -g appcenter-cli
```

Note: You can also use `npx` if you don't like installing a lot of packages globally

- Now, we have to login with the cli. We can do that simply using the command below and authenticating with our AppCenter account.

```
appcenter login
```

- That's it, we're almost there. We can use the command below to make releases.

```
appcenter codepush release-react -a <user>/<app> -d <environment>
```

For Example:

```
appcenter codepush release-react -a Karan-Pratap-Singh/CodePushDemo -d Staging
```

Note: To find out which apps are currently available to use in the `-a` argument then just use `appcenter apps list` command (you need to be authenticated)

- Bonus Tip 🔥

Typing these might get tedious, so what I like to do is add these scripts to my package.json like:

```
"scripts": {
  "codepush:ios": "appcenter codepush release-react -a Karan-Pratap-Singh/CodePushDemo -
  "codepush:android": "appcenter codepush release-react -a Karan-Pratap-Singh/CodePushDe
```

- After making the release it should be visible on your dashboard with tons of cool info about no. of installs, rollbacks etc.

Well, this was all about setting up Codepush with App Center. However App Center has tons of great features like CI/CD, Analytics, Test Runs, Diagnostics, Push Notification, Crash Reporting.

If you're interested in CI/CD with AppCenter, checkout my other article about it.

Hopefully, you were able to integrate CodePush into your build and enjoy seamless updates 😄

If you liked this article, or faced any issues, feel free to reach out via Twitter or Email 🚀

Happy Coding 🎉

Discussion (7)

Subscribe

DEV

Add to the discussion

Miguel Cárdenas • May 27 '20 • Edited

Hi, thanks for your post, I've been working in an RN project using CodePush and I've found several issues with this, for example, Codepush always tries to download the whole assets and in my projects they're so big, and I'd prefer to just load and download only JS code, another issue is CodePush sometimes but frequently show a 503 error and it's a big problem because that cause that my App does not download the last version of code, have you led with those problems?

2

Karan Pratap Singh • May 27 '20

Glad you liked the post, I haven't experienced such issues so I have no idea unfortunately

2

ErnesII • Sep 21 '20

Thank your for this guide, it really help me. I have some doubts... Talking about android and that i want to have one build with multiple deployment keys. Should I add the CodePushDeploymentKey config in the strings.xml file? Because right now I am using the production key in this file and I am not able to get updades. Also one more question... is it suppose that when i run this command appcenter codepush release-react -a / is the update going to be ready for Staging without using promote?I saw a variation of this command that include this: -d Staging

1

Karan Pratap Singh • Sep 22 '20

Glad you liked this article, yes you'll need to use -d option to let app center know which stage it is. Not sure about the CodePushDeploymentKey config in the strings.xml as I never add it there, I always add it via codePush(codePushOptions)(App);

1

Ehsan sarshar • Jul 13 '20

Awesome blog post. but I prefer expo-updates over code-push because it's both available for expo and react-native and also I can push native changes not just javascript part

2

abhishek gupta • Jul 2 '20

Hi Karan, Great writeup. I have read at many places that Apple app store rejected apps that used to push via codepush. Did you face anything like that?

1

Karan Pratap Singh • Jul 2 '20

Thanks, No I did not face any such issue, as far as I know they only reject when your codepush update alters the core functionality of the app as shown on appstore

1

Karan Pratap Singh

A full stack developer who loves open source and values learning and growing with people, teams, and technologies

WORK

Software Developer

More from Karan Pratap Singh

Amazing image placeholders with blurhash

#react #typescript #javascript

Fullstack GraphQL starter kit November update

#javascript #typescript #react #node

Private, Public and Restricted routes in React

#react #webdev #javascript #typescript