

 [gitname](#) / [react-gh-pages](#) Public

Deploying a React App (created using create-react-app) to GitHub Pages

[gitname.github.io/react-gh-pages/](#)

★ 3.3k stars 🔗 576 forks

★ Star

👁 Watch ▼

Code

Issues

21


Pull requests

7

Actions

Security

Insights

 master ▼

...



gitname ...

on 4 Apr [View code](#)

Deploying a React App* to GitHub Pages

* created using `create-react-app`

Introduction

In this tutorial, I'll show you how you can create a React app and deploy it to GitHub Pages.

To create the React app, I'll be using [create-react-app](#), which is a tool people can use to create a React app from scratch. To deploy the React app, I'll be using [gh-pages](#), which is an npm package people can use to deploy things to [GitHub Pages](#), a free web hosting service provided by GitHub.

If you follow along with this tutorial, you'll end up with a new React app—hosted on GitHub Pages—which you can then customize.

Tutorial

Prerequisites

1. [Node](#) and [npm](#) are installed. Here are the versions I'll be using while making this tutorial:


```
$ node --version  
v16.13.2
```

```
$ npm --version  
8.1.2
```

Installing npm adds two commands to the system— `npm` and `npx` —both of which I'll be using while making this tutorial.

2. [Git](#) is installed. Here's the version I'll be using while making this tutorial:

```
$ git --version  
git version 2.29.1.windows.1
```

3. A [GitHub](#) account. 

Procedure

1. Create an empty repository on GitHub

1. Sign into your GitHub account.
2. Visit the [Create a new repository](#) form.
3. Fill in the form as follows:

- **Repository name:** You can enter any name you want*.

* For a [project site](#), you can enter any name you want. For a [user site](#), GitHub [requires](#) that the repository's name have the following format:

```
{username}.github.io (e.g. gitname.github.io )
```

The name you enter will show up in a few places: (a) in references to the repository throughout GitHub, (b) in the URL of the repository, and (c) in the URL of the deployed React app.

In this tutorial, I'll be deploying the React app as a project site.

I'll enter: `react-gh-pages`

- **Repository privacy:** Select *Public* (or *Private**).

* For [GitHub Free](#) users, the only type of repository that can be used with GitHub Pages is *Public*. For [GitHub Pro](#) users (and other paying users), both *Public* and *Private* repositories can be used with GitHub Pages.

I'll choose: *Public*

- **Initialize repository:** Leave all checkboxes empty.

That will make it so GitHub creates an empty repository, instead of pre-populating the repository with a `README.md`, `.gitignore`, and/or `LICENSE` file.

4. Submit the form.

At this point, your GitHub account contains an empty repository, having the name and privacy type that you specified.

2. Create a React app

1. Create a React app named `my-app` :

In case you want to use a different name from `my-app` (e.g. `web-ui`), you can accomplish that by replacing all occurrences of `my-app` in this tutorial, with that other name (i.e. `my-app --> web-ui`).

```
$ npx create-react-app my-app
```

That command will create a React app written in JavaScript. To create one written in [TypeScript](#), you can issue this command instead:

```
$ npx create-react-app my-app --template typescript
```

That command will create a new folder named `my-app`, which will contain the source code of a React app.

In addition to containing the source code of the React app, that folder is also a Git repository. That characteristic of the folder will come into play in Step 6.

2. Enter the newly-created folder:

```
$ cd my-app
```

At this point, there is a React app on your computer and you are in the folder that contains its source code. All of the remaining commands shown in this tutorial can be run from that folder.

3. Install the `gh-pages` npm package

1. Install the `gh-pages` npm package and designate it as a [development dependency](#):

```
$ npm install gh-pages --save-dev
```

At this point, the `gh-pages` npm package is installed on your computer and the React app's dependence upon it is documented in the React app's `package.json` file.

4. Add a homepage property to the `package.json` file

1. Open the `package.json` file in a text editor.

```
$ vi package.json
```

In this tutorial, the text editor I'll be using is [vi](#). You can use any text editor you want; for example, [Visual Studio Code](#).

2. Add a `homepage` property in this format*: `https://{username}.github.io/{repo-name}`

* For a [project site](#), that's the format. For a [user site](#), the format is: `https://{username}.github.io`. You can read more about the `homepage` property in the ["GitHub Pages" section](#) of the `create-react-app` documentation.

```
{
  "name": "my-app",
  "version": "0.1.0",
  + "homepage": "https://gitname.github.io/react-gh-pages",
  "private": true,
```

At this point, the React app's `package.json` file includes a property named `homepage`.

5. Add deployment scripts to the `package.json` file

1. Open the `package.json` file in a text editor (if it isn't already open in one).

```
$ vi package.json
```

2. Add a `predeploy` property and a `deploy` property to the `scripts` object:

```
"scripts": {  
  + "predeploy": "npm run build",  
  + "deploy": "gh-pages -d build",  
  "start": "react-scripts start",  
  "build": "react-scripts build",  
}
```

☰ README.md

6. Add a "remote" that points to the GitHub repository

1. Add a "remote" to the local Git repository.

You can do that by issuing a command in this format:

```
$ git remote add origin https://github.com/{username}/{repo-name}.git
```

To customize that command for your situation, replace `{username}` with your GitHub username and replace `{repo-name}` with the name of the GitHub repository you created in Step 1.

In my case, I'll run:

```
$ git remote add origin https://github.com/gitname/react-gh-pages.git
```

That command tells Git where I want it to push things whenever I—or the `gh-pages` npm package acting on my behalf—issue the `$ git push` command from within this local Git repository.

At this point, the local repository has a "remote" whose URL points to the GitHub repository you created in Step 1.

7. Deploy the React app to GitHub Pages

1. Deploy the React app to GitHub Pages

```
$ npm run deploy
```

That will cause the `predeploy` and `deploy` scripts defined in `package.json` to run.

Under the hood, the `predeploy` script will build a distributable version of the React app and store it in a folder named `build`. Then, the `deploy` script will push the contents of that folder to a new commit on the `gh-pages` branch of the GitHub repository, creating that branch if it doesn't already exist.

By default, the new commit on the `gh-pages` branch will have a commit message of "Updates". You can [specify a custom commit message](#) via the `-m` option, like this:

```
$ npm run deploy -- -m "Deploy React app to GitHub Pages"
```

GitHub Pages will automatically detect that a new commit has been added to the `gh-pages` branch of the GitHub repository. Once it detects that, it will begin serving the files that make up that commit — in this case, the distributable version of the React app — to anyone that visits the `homepage` URL you specified in Step 4.

That's it! The React app has been deployed to GitHub Pages! 🚀

At this point, the React app is accessible to anyone who visits the `homepage` URL you specified in Step 4. For example, the React app I deployed is accessible at <https://gitname.github.io/react-gh-pages>.

8. (Optional) Store the React app's *source code* on GitHub

In the previous step, the `gh-pages` npm package pushed the distributable version of the React app to a branch named `gh-pages` in the GitHub repository. However, the *source code* of the React app is not yet stored on GitHub.

In this step, I'll show you how you can store the source code of the React app on GitHub.

1. Commit the changes you made while you were following this tutorial, to the `master` branch of the local Git repository; then, push that branch up to the `master` branch of the GitHub repository.

```
$ git add .  
$ git commit -m "Configure React app for deployment to GitHub Pages"  
$ git push origin master
```

I recommend exploring the GitHub repository at this point. It will have two branches: `master` and `gh-pages`. The `master` branch will contain the React app's source code, while the `gh-pages` branch will contain the distributable version of the React app.

References

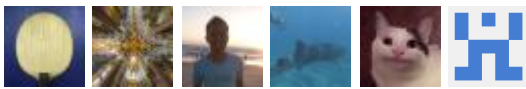
1. [The official create-react-app deployment guide](#)
2. [GitHub blog: Build and deploy GitHub Pages from any branch](#)

Notes

- Special thanks to GitHub (the company) for providing us with the GitHub Pages hosting service for free.
- And now, time to turn the default React app generated by `create-react-app` into something unique!
- This repository consists of two branches:
 - `master` - the *source code* of the React app
 - `gh-pages` - the React app *built from* that source code

Contributors

Thanks to these people for contributing to the maintenance of this tutorial.



This list is maintained manually—for now—and includes (a) each person who submitted a pull request that was eventually merged into `master`, and (b) each person who contributed in a different way (e.g. providing constructive feedback) and who approved of me including them in this list.

Releases

No releases published

Packages

No packages published

Used by 56



Contributors 4



gitname gitname



dependabot[bot]



rhulse Richard Hulse



AbhishekCode Abhishek Singh

Environments 1



github-pages

Active

Languages

TypeScript 43.0% HTML 37.0% CSS 20.0%