

How to set up JavaScript Test Coverage

JULY 20, 2019 BY ROBIN WIERUCH - [EDIT THIS POST](#)

 Follow on Twitter

17k

[Follow on Facebook](#)



This tutorial is part 2 of 2 in the series.

Part 1: [How to set up Continuous Integration for JavaScript](#)

This tutorial is part 3 of 3 in the series.

Part 1: [How to set up React with Webpack and Babel](#)

Part 2: [How to test React components with Jest](#)


Coveralls is used to show you the test coverage of your JavaScript application. Let's see how it can be used for your JavaScript project which is already on GitHub and connected to your Travis CI due to the previous CI setup tutorial. First, sign up at [Coveralls.io](https://coveralls.io) with your GitHub account. Second, synchronize your GitHub repositories and toggle a specific repository to be used for code coverage.


ADD REPO

To add repositories that are private on Github or Bitbucket you will need a Coveralls Pro subscription. Click the 'Add Subscription' button next to the user or organization name to add a private repository.

☒ **RWIERUCH** / *react-components-test-setup* [DETAILS](#) [GITHUB](#)

 ☐ **THE-ROAD-TO-LEARN-REACT** / *react-controlled-components-examples* [GITHUB](#)

 ☐ **THE-ROAD-TO-LEARN-REACT** / *react-styled-components-example* [GITHUB](#)

 Afterward, hit the "Details" button to copy your `coveralls_repo_token` to your clipboard. Since you don't want to add this private token directly to your public project, you can add it on your Travis CI dashboard to your repository's environment variables. You will find it via the settings option of your Travis repository.

rwieruch / react-components-test-setup  build passing

[Current](#)
[Branches](#)
[Build History](#)
[Pull Requests](#)

 **master** last changes

 Commit 762f5d2

 Compare 55dc72b..762f5d2

 Branch master

 Robin Wieruch authored and committed

 Ran for 52 sec

 a day ago

[More options](#)


[Settings](#)

[Requests](#)

[Caches](#)

[Trigger build](#)

Then, create a new environment variable for your project. You can name it `coveralls_repo_token`:

Environment Variables

Notice that the values are not escaped when your builds are executed. Special characters (for hash) should be escaped accordingly.

coveralls_repo_token

Name Value OFF Display value in build log Add

Last but not least, modify your project the following way. First, install the coveralls library on the command line to your dev dependencies:

```
npm install --save-dev coveralls
```

Second, add a new script to your *package.json* file to introduce Coveralls to it:

```
"scripts": {
  "start": "webpack serve --config ./webpack.config.js --mode development",
  "test": "jest --config ./jest.config.json",
  "coveralls": "cat ./coverage/lcov.info | node node_modules/.bin/coveralls"
},
```

f And third, extend your Travis CI configuration for reporting the coveralls information to your coveralls.io dashboard.



```
language: node_js

node_js:
  - stable

install:
  - npm install

script:
  - npm run test -- --coverage

after_script:
  - COVERALLS_REPO_TOKEN=$coveralls_repo_token npm run coveralls
```

That's it. By adding, committing and pushing your changes to GitHub now, you can see how a report shows up on your Coveralls.io dashboard.



LAST BUILD ON BRANCH MASTER				
COMMITTED 20 MAR 2018 - 4:31			FIRST BUILD ON MASTER AT 73.333%	
BUILD #	BUILD TYPE	COMMITTED BY	COMMIT MESSAGE	RUN DETAILS
4	push travis-ci	 rwieruch	add coveralls	11 of 15 relevant lines covered (73.33%) 0.8 hits per line

Perhaps you can see that the coverage isn't too high. Then it's up to you to add tests to increase the percentage for your project.

Last but not least, you can add the fancy Coveralls badge to your GitHub's *README.md* file. You find the badge on the Coveralls dashboard for embedding it as markdown:

My JavaScript Project

[![Coverage Status](https://coveralls.io/repos/github/rwieruch/my-javascript-1



Make sure to change the URL to your repository's URL.



If you are using Jest as a test runner, you can enforce a certain coverage for your JavaScript project. Also you can include and exclude specific folders/files from your source code to be added/removed from your testing coverage report:

```
module.exports = {
  ...
  coverageThreshold: {
    global: {
      functions: 95,
      lines: 95
    }
  },
  collectCoverageFrom: [
    '<rootDir>/src/**/*.js',
    '!<rootDir>/src/pages/**/*.js'
  ]
};
```

That's everything in a nutshell about testing coverage in JavaScript projects.

Continue Reading: [How to test React components with Jest](#)

Continue Reading: [How to end-to-end test React components with Cypress](#)

Show Comments

KEEP READING ABOUT [NODE](#) >

HOW TO DEPLOY A GITHUB WEBHOOK IN NODE.JS

The following implementation shows you how to deploy a Github Webhook in Node.js. Personally I am using this kind of GitHub Webhook to deploy my websites automatically on Digital Ocean . Without...



HOW TO SET UP CONTINUOUS INTEGRATION FOR JAVASCRIPT

The following tutorial will show you how to set up continuous integration with Travis CI. The continuous integration should make sure that the JavaScript application is built and tested with every...



THE ROAD TO REACT

Learn React by building real world applications. No setup configuration. No tooling. Plain React in 200+ pages of learning material. Learn React like

50.000+ readers.

GET THE BOOK >

Get it on Amazon.



TAKE PART

NEVER MISS AN ARTICLE ABOUT WEB DEVELOPMENT AND JAVASCRIPT.

- ✓ Join 50.000+ Developers
- ✓ Learn Web Development with JavaScript
- ✓ Tips and Tricks
- ✓ Access Tutorials, eBooks and Courses
- ✓ Personal Development as a Software Engineer

Your email address

SUBSCRIBE

View our [Privacy Policy](#).

PORTFOLIO

[Online Courses](#)

[Open Source](#)

[Tutorials](#)

ABOUT

[About me](#)

[What I use](#)

[How to work with me](#)

[How to support me](#)

© Robin Wieruch



[Contact Me](#)

[Privacy & Terms](#)