# Cypress Tips And Tricks

Anshita Bhasin

## Cypress

Cypress is a JavaScript-based end-to-end testing framework that enables automation developers to write and execute tests for web applications.Despite its many benefits, Cypress can be challenging to use effectively, especially for beginners. However, you can maximize productivity and streamline your testing process with a few Cypress tips and tricks.

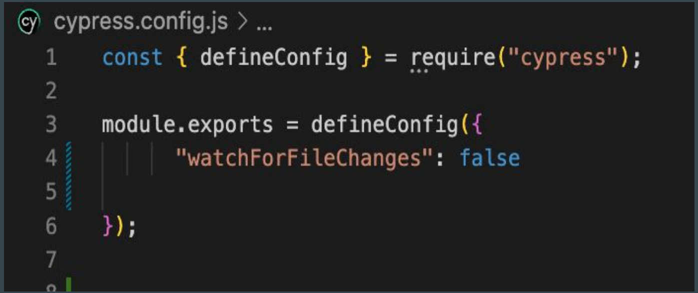
Whether you're new to Cypress or an experienced user, these Cypress tips and tricks will help you take your automation testing to the next level. So, let's dive in and explore how you can make Cypress work for you.

Here are some Cypress tips and tricks for getting the most out of the framework. When writing this tutorial, the latest version of Cypress is 12.5.1.

## (1) Modify Real-time reloads

By default, Cypress provides the convenient feature of automatically reloading the page and re-executing the test commands whenever any changes are made to the tests. But you can change this behavior in Cypress by setting the ***watchForFileChanges*** option to *false* in the configuration file.

```
"watchForFileChanges":false
```

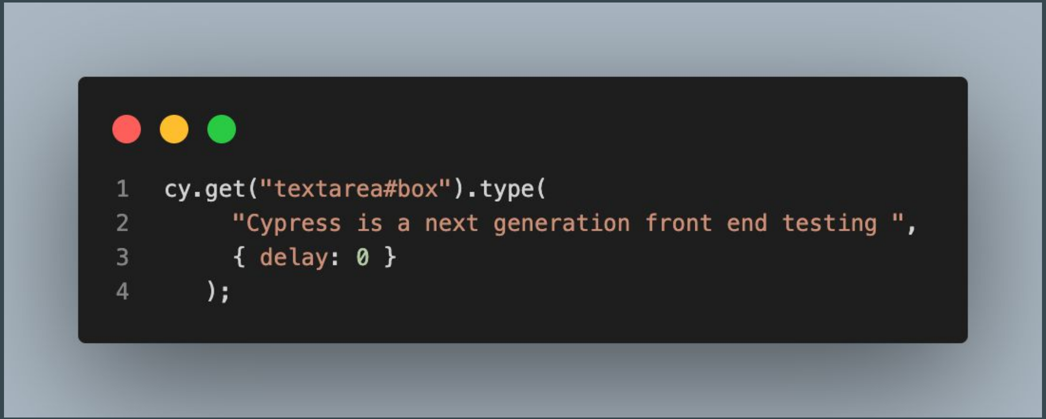


cypress.config.js > ...

```javascript
const { defineConfig } = require("cypress");

module.exports = defineConfig({
      "watchForFileChanges": false

});

```

## (2) Use {delay: 0} to quickly enter the text in Cypress

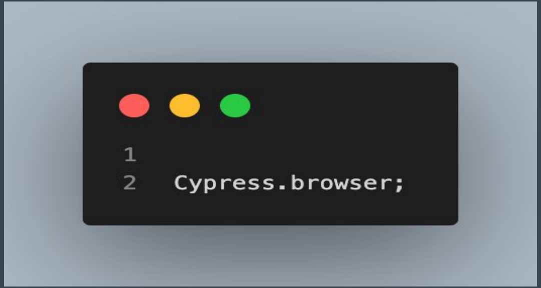
The delay option object can add a delay after each keypress.Setting the delay option to 0 i.e, *{delay:0}* may cause the text to be entered faster than a user could realistically type it.



```
cy.get("textarea#box").type(
     "Cypress is a next generation front end testing ",
     { delay: 0 }
   );
```

## (3) Check browser version on runtime

Knowing the browser version that your tests are running on, can be useful for several reasons, including: Debugging, Compatibility testing,Customization. You can get the browser information using the below command:

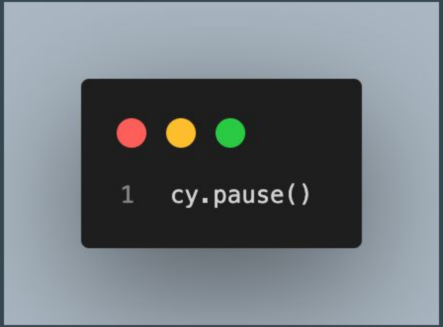


```

Cypress.browser;
```

For example, to print the browser version, you can use:
cy.log(Cypress.browser.version)

## (4) Pause the test execution

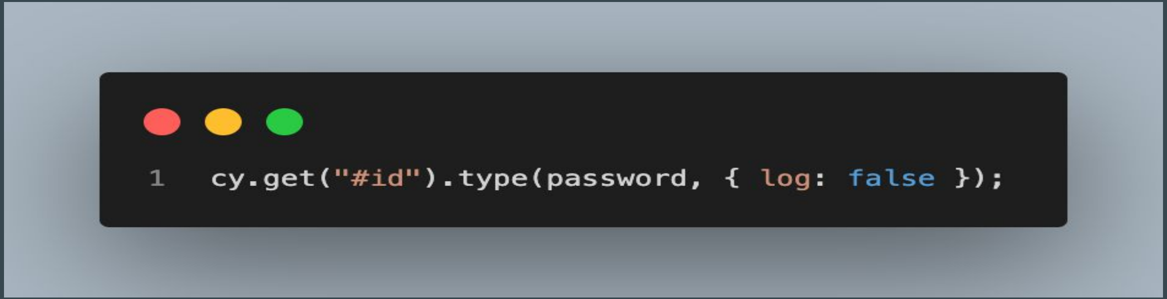
To pause the execution of a test to inspect the DOM /network to perform some actions manually can be done by using the below command:



```
cy.pause()
```

## (5) Add {log:false}

The *{log: false}* option can be used to disable logging for a specific command in Cypress Command Log. By default, Cypress logs the command and its arguments to the console when it is run. However, the command will be executed without being logged if you pass the *{log: false}* option.

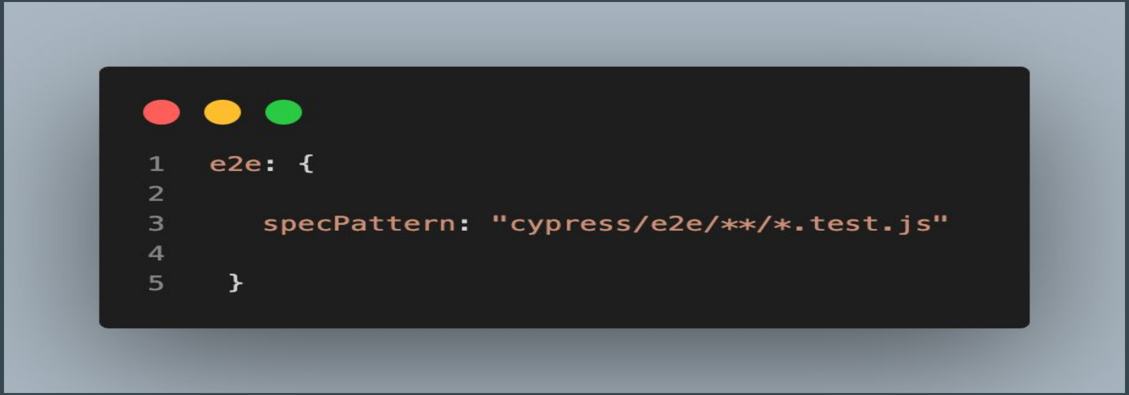


```
cy.get("#id").type(password, { log: false });
```

## (6) Modify specPattern as per the requirement

In Cypress, the *specPattern* can be customized in the configuration file to suit your specific requirements.

For example, if you want to change *specPattern* to *test.js,* you can make the changes in config files as shown below:

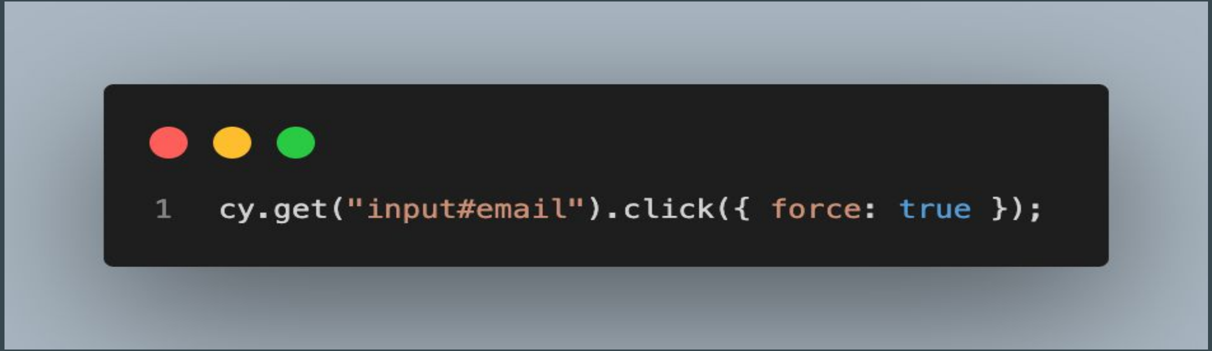


```
e2e: {

    specPattern: "cypress/e2e/**/*.test.js"

 }
```

## (7) Handle Invisible/Hidden elements

To interact with hidden or disabled elements in Cypress, you must pass option *{force:true}* to the Cypress command.

Example:

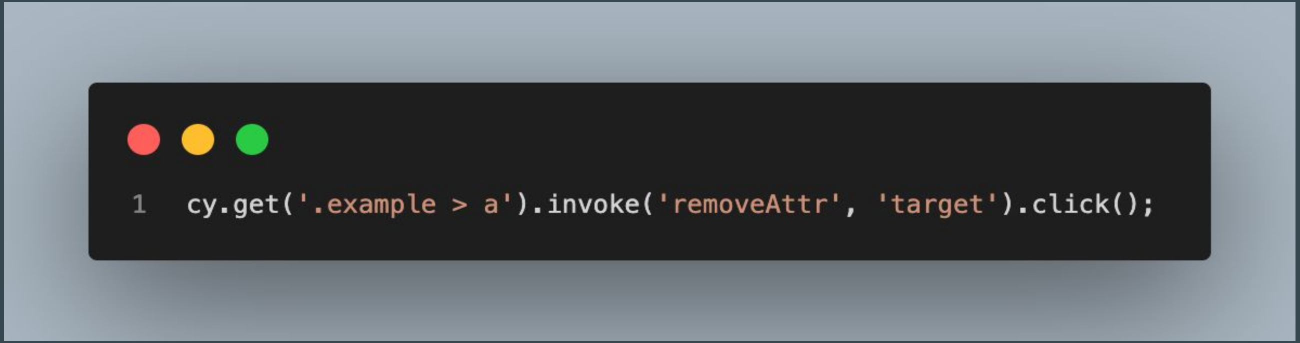


```
cy.get("input#email").click({ force: true });
```

## (8) Open the Link in the same tab

Cypress, as it runs inside the browser, does not support multiple tabs. However, there are ways to work around this and open a link in the same tab. One of the solution is described as below:

Sample code:



```
cy.get('.example > a').invoke('removeAttr', 'target').click();
```

For a more detailed understanding of additional tips and tricks, please refer to the blog provided below.

Blog Link - [https://www.lambdatest.com/learning-hub/cypress-tips-and-tricks?utm_source=twitter&utm_medium=organic&utm_campaign=feb20_sb&utm_term=sb&utm_content=learning_hub](https://www.lambdatest.com/learning-hub/cypress-tips-and-tricks?utm_source=twitter&utm_medium=organic&utm_campaign=feb20_sb&utm_term=sb&utm_content=learning_hub)