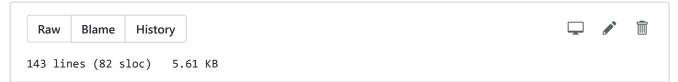


protractor / docs / timeouts.md





Timeouts

Because WebDriver tests are asynchronous and involve many components, there are several reasons why a timeout could occur in a Protractor test.

Timeouts from Protractor

Waiting for Page to Load

When navigating to a new page using browser.get, Protractor waits for the page to be loaded and the new URL to appear before continuing.

- Looks like: an error in your test results Error: Timed out waiting for page to load after 10000ms
- Default timeout: 10 seconds
- How to change: To change globally, add getPageTimeout: timeout_in_millis
 to your Protractor configuration file. For an individual call to get, pass an
 additional parameter: browser.get(address, timeout_in_millis)

Waiting for Angular

Before performing any action, Protractor waits until there are no pending asynchronous tasks in your Angular application. This means that all timeouts and http requests are finished.

- Looks like: an error in your test results Timed out waiting for asynchronous
 Angular tasks to finish after 11 seconds.
- Default timeout: 11 seconds
- How to change: Add allScriptsTimeout: timeout_in_millis to your Protractor configuration file.

You may also need to fix this problem with a change to your application.

AngularJS

If your AngularJS application continuously polls \$timeout or \$http, Protractor will wait indefinitely and time out. You should use the \$interval for anything that polls continuously (introduced in Angular 1.2rc3).

Angular

For Angular apps, Protractor will wait until the Angular Zone stabilizes. This means long running async operations will block your test from continuing. To work around this, run these tasks outside the Angular zone. For example:

```
this.ngZone.runOutsideAngular(() => {
    setTimeout(() => {
        // Changes here will not propagate into your view.
        this.ngZone.run(() => {
            // Run inside the ngZone to trigger change detection.
        });
    }, REALLY_LONG_DELAY);
});
```

As an alternative to either of these options, you could disable waiting for Angular, see below.

Waiting for Angular on Page Load

Protractor waits for the angular variable to be present when loading a new page.

- Looks like: an error in your test results Angular could not be found on the page: retries looking for angular exceeded
- Default timeout: 10 seconds

How to change: To change globally, add getPageTimeout: timeout_in_millis
to your Protractor configuration file. For an individual call to get, pass an
additional parameter: browser.get(address, timeout_in_millis)

How to disable waiting for Angular

If you need to navigate to a page which does not use Angular, you can turn off waiting for Angular by setting browser.waitForAngularEnabled(false). For example:

```
browser.waitForAngularEnabled(false);
browser.get('/non-angular-login-page.html');
element(by.id('username')).sendKeys('Jane');
element(by.id('password')).sendKeys('1234');
element(by.id('clickme')).click();
browser.waitForAngularEnabled(true);
browser.get('/page-containing-angular.html');
```

Timeouts from WebDriver

Asynchronous Script Timeout

Sets the amount of time to wait for an asynchronous script to finish execution before throwing an error.

- Looks like: an error in your test results ScriptTimeoutError: asynchronous script timeout: result was not received in 11 seconds
- Default timeout: 11 seconds
- How to change: Add allScriptsTimeout: timeout_in_millis to your Protractor configuration file.

Timeouts from Jasmine

Spec Timeout

If a spec (an 'it' block) takes longer than the Jasmine timeout for any reason, it will fail.

- Looks like: a failure in your test results timeout: timed out after 30000 msec
 waiting for spec to complete
- Default timeout: 30 seconds
- How to change: To change for all specs, add jasmineNodeOpts:
 {defaultTimeoutInterval: timeout_in_millis} to your Protractor configuration
 file. To change for one individual spec, pass a third parameter to it:
 it(description, testFn, timeout_in_millis).

Timeouts from Sauce Labs

If you are using Sauce Labs, there are a couple additional ways your test can time out. See Sauce Labs Timeouts Documentation for more information.

Maximum Test Duration

Sauce Labs limits the maximum total duration for a test.

- Looks like: Test exceeded maximum duration after 1800 seconds
- Default timeout: 30 minutes
- How to change: Edit the "max-duration" key in the capabilities object.

Command Timeout

As a safety measure to prevent Selenium crashes from making your tests run indefinitely, Sauce limits how long Selenium can take to run a command in browsers. This is set to 300 seconds by default.

- Looks like: Selenium took too long to run your command
- Default timeout: 300 seconds
- How to change: Edit the "command-timeout" key in the capabilities object.

Idle Timeout

As a safety measure to prevent tests from running too long after something has gone wrong, Sauce limits how long a browser can wait for a test to send a new command. This is set to 90 seconds by default. You can adjust this limit on a per-job basis.

- Looks like: Test did not see a new command for 90 seconds. Timing out.
- Default timeout: 90 seconds
- How to change: Edit the "idle-duration" key in the capabilities object.