# Web Driver IO Framework

WebDriver IO is an open-source testing utility for NodeJS. It makes it possible to write super easy selenium tests with JavaScript in your favorite BDD or TDD test framework. It basically sends requests to a Selenium server via the WebDriver Protocol and handles its response. These requests are wrapped in useful commands and can be used to test several aspects of your site in an automated way.

In simple words – WebDriver IO is a custom implementation for selenium's W3C web driver API. It is written in JavaScript and packaged into 'npm' and runs on Node.js.

## Why should I use WebDriver IO?

WebDriver IO is a flexible and powerful automation framework that can be used to automate web applications (WebDriver/Chrome Dev Tools), mobile (Hybrid/Native) applications using Appium, and even Native desktop applications (Windows and macOS), using a custom configuration of Appium (mobile automation).

The tests performed on WebDriver IO are simple and concise. You can control browsers with just a few lines of code. As it is built in NodeJS, you can write your tests using JavaScript, as well as with TypeScript.

NodeJS has a vast community around the world that has helped the framework grow with new, powerful, and useful features and plugins to make suite tests much more robust, complete, and stable. Compared to many automation tools, WebDriver IO is a truly open-source project that is run with open governance and owned by a non-profit entity called Open-JS Foundation.

## Why you should consider using WebDriver IO over Selenium WebDriver?

1. **Front-end friendly –** It is built for testing modern web applications written in React, Vue, Angular, Svelte, or other front-end frameworks built in NodeJS.
2. **Power of Selenium -** Selenium, is an incredibly robust platform and an industry leader for running browser automation, and WebDriver IO stands on top of Selenium. As all the great things about Selenium are available, without the overhead of writing Java based tests.
3. **Easy to use -** Commands you use in your WebDriver IO tests are crisp, concise and all about common sense.
4. **Parallel Execution –** We can execute our test cases parallelly without any complex configuration.
5. **Cross Browser Testing -** We can execute our test cases on different browsers without any complex configuration.
6. **Easy Project Setup -** Easy project setup, flexible configuration in only one configuration file `(wdio.conf)`, and integration with other test automation tools out of a box.

## Setup Project

To add a full WebDriver IO setup to an existing or new project using the WebDriver IO Starter Toolkit, run:

If you're in the root directory of an existing project, run:

```
npm init wdio .
```

or if you want to create a new project:

```
npm init wdio ./path/to/new/project
```

The wizard will prompt a set question that guides you through the setup. You can pass a `--yes` parameter to pick a default set up which will use Mocha with Chrome using the Page Object pattern.

```
npm init wdio . --yes
```

## Run Tests

You can start your test suite by using the run command and pointing to the WebdriverIO config that you just created:

```
npx wdio run ./wdio.conf.js
```

If you like to run specific test files you can add a `--spec` parameter:

```
npx wdio run ./wdio.conf.js --spec example.e2e.js
```

define suites in your config file and run just the test files defined by in a suite:

```
npx wdio run ./wdio.conf.js --suite exampleSuiteName
```

## WebDriver IO POC

There are three different projects -

1. gem-webdriver-io
2. sample-webdriver-io-project
3. webdriver-io-template

## Gem-WebDriver-IO

It contains the web-actions methods and custom gem-report. This project works as a npm module. We are going to use this in other projects and utilize the its methods.

**Step 1: Clone the project -**

```
git clone https://github.com/Gemini-Solutions/TheBrainiacsCrew -b
WebDriver-IO
```

**Step 2: Install Dependencies -**

```
npm install
```

**Step 3: Create local package module of this project -**

```
npm link
```

**Step 4:** Open WebDriver-IO-Template.

## WebDriver-IO-Template

This is the basic starter template that you can clone and start writing your test and using the web-actions from gem-webdriver-io project. We are using page object model approach, creating different folders for locators, steps and features files.

**Step 1:** Open the project.

**Step 2: Install Dependencies**

```
npm install
```

**Step 3: Add local package module into this project**

```
npm link gem-webdriver-io
```

**Step 4: Run the project**

```
npm test
```

**Step 5:** Start writing the test cases, Refer the Sample WebDriver-IO-Project.

## Sample-WebDriver-IO-Project

This is the sample project you can clone it for reference.

## Key Features of WebDriver IO

- Beginners can easily understand the setup.
- It also supports built-in waits and retries.
- It needs an external library.
- JS users can easily understand the test created in it.
- User custom implementation.

## Limitations of WebDriver IO

- Debugging the test is difficult.
- It doesn’t have any paid features.
- It contains a lot of customizations which can lead to confusion.
- Async-await syntax can be confusing for beginners.
- CLI test runner only.