First things first — you can install intern-cli using npm:

|  |  |
| --- | --- |
| 1 | $ npm install -g intern-cli |

Note that unlike a normal Intern install, we’re installing intern-cli globally. This is similar to how other popular npm-based dev tools work; the command-line interface is installed globally, and the tool itself will be installed in a project.

Now to try it out!

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | $ cd ~/tmp/new\_project  $ intern    You'll need a local install of Intern before you can use **this** command.  Install it **with**      npm install --save-dev intern    $ |

Ok, we weren’t quite ready yet. Intern must be locally installed in a project before you can use intern-cli.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | $ npm install --save-dev intern  ...  $ intern      Usage: intern [options] [command]      Run JavaScript tests      Commands:        init [options]    Setup a project **for** testing **with** Intern      run [options]     Run tests **in** Node or **in** a browser using WebDriver      serve [options]   Start a simple web server **for** running unit tests **in** a browser on your system      Options:        -h, --help     output usage information      -v, --verbose  show more information about what Intern is doing      -V, --version  output the version      --debug        enable the Node debugger |

Better! With no options or commands, intern-cli displays its top-level help menu. It provides three commands: init, run, and serve. The run command is just a convenience for running Intern’s Node.js client or WebDriver runner (intern-client and intern-runner). The init and serve commands are new, though.

The init command bootstraps a testing environment for a project. It creates a tests/ directory and populates it with a default test config and example functional and unit tests.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | $ intern init      Intern initialized! A test directory containing example unit and functional    tests has been created at tests/. See tests/intern.js **for** configuration options.      Run the sample unit test **with** `intern run`.      To run the sample functional test, first start a WebDriver server (e.g.,    Selenium), then run `intern run -w`. The functional tests assume chrome is    installed.      Note that running WebDriver tests **with** Chrome requires ChromeDriver to be    available **in** the system path. See        https://github.com/SeleniumHQ/selenium/wiki/ChromeDriver    **for** more information.    $ tree tests  tests  ├── functional  │   ├── hello.js  │   └── page.html  ├── intern.js  └── unit      └── hello.js |

At this point we can actually run tests. To do that, use the run command. By default this command will use Intern’s Node.js client. While intern-client and intern-runner require that a test config be specified, intern assumes the test config is at tests/intern.js, so a simple intern run command will suffice:

|  |  |
| --- | --- |
| 1  2  3  4 | $ intern run  PASS: hello - hello world (1ms)  0/1 tests failed  0/1 tests failed |

Chromedriver - - port=4444 - - url-base=wd/hub

That was neat, but what about tests that need a browser? One option is to use Intern’s WebDriver runner by passing a “-w” option to the run command. Assuming a Selenium server is running and Intern is configured to use it, this will tell Intern to start a browser and run unit and functional tests. However, intern-cli provides another option for running unit tests in a browser with its serve command.

Simply running intern serve will start Intern’s test proxy server, which is basically a static web server with some additional functionality, and display a URL that you can open in a browser to run unit tests using Intern’s browser client.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Listening on 0.0.0.0:9000      To run unit tests, browse to:        http://localhost:9000/node\_modules/intern/client.html?config=tests/intern.js      Press CTRL-C to stop serving. |

The serve command also accepts a -o option. When this option is provided, intern-cli will open the default system browser automatically after the server starts. This makes running browser-based unit tests as easy as running tests in the Node.js client.