On this page

Selenium Grid (experimental)

Introduction

Playwright can connect to Selenium Grid Hub that runs Selenium 4 to launch **Google Chrome** or **Microsoft Edge** browser, instead of running browser on the local machine. Note this feature is **experimental** and is prioritized accordingly.

WARNING

There is a risk of Playwright integration with Selenium Grid Hub breaking in the future. Make sure you weight risks against benefits before using it.

MORE DETAILS Internally, Playwright connects to the browser using Chrome DevTools Protocol websocket. Selenium 4 currently exposes this capability. However, this might not be the case in the future. If Selenium drops this capability, Playwright will stop working with it.

Before connecting Playwright to your Selenium Grid, make sure that grid works with Selenium WebDriver. For example, run one of the examples and pass `SELENIUM_REMOTE_URL` environment variable. If webdriver example does not work, look for any errors at your Selenium hub/node/standalone output and search Selenium issues for a possible solution.

Starting Selenium Grid

If you run distributed Selenium Grid, Playwright needs selenium nodes to be registered with an accessible address, so that it could connect to the browsers. To make sure it works as expected, set `SE_NODE_GRID_URL` environment variable pointing to the hub when running selenium nodes.

```
# Start selenium node
SE_NODE_GRID_URL="http://<selenium-hub-ip>:4444" java -jar selenium-server-<version>.jar node
```

Connecting Playwright to Selenium Grid

To connect Playwright to **Selenium Grid 4**, set `SELENIUM_REMOTE_URL` environment variable pointing to your Selenium Grid Hub. Note that this only works for Google Chrome and Microsoft Edge.

```
SELENIUM_REMOTE_URL=http://<selenium-hub-ip>:4444 npx playwright test
```

You don't have to change your code, just use your testing harness or browserType.launch() as usual.

Passing additional capabilities

If your grid requires additional capabilities to be set (for example, you use an external service), you can set `SELENIUM_REMOTE_CAPABILITIES` environment variable to provide JSON-serialized capabilities.

```
SELENIUM_REMOTE_URL=http://<selenium-hub-ip>:4444
SELENIUM_REMOTE_CAPABILITIES="{'mygrid:options':{os:'windows',username:'John',password:'secure'}}" npx playwright test
```

Passing additional headers

If your grid requires additional headers to be set (for example, you should provide authorization token to use browsers in your cloud), you can set `SELENIUM_REMOTE_HEADERS` environment variable to provide JSON-serialized headers.

```
SELENIUM_REMOTE_URL=http://<selenium-hub-ip>:4444
SELENIUM_REMOTE_HEADERS="{'Authorization':'OAuth 12345'}" npx playwright test
```

Detailed logs

Run with `DEBUG=pw:browser*` environment variable to see how Playwright is connecting to Selenium Grid.

```
DEBUG=pw:browser* SELENIUM_REMOTE_URL=http://internal.grid:4444 npx playwright test
```

If you file an issue, please include this log.

Using Selenium Docker

One easy way to use Selenium Grid is to run official docker containers. Read more in selenium docker images documentation. For experimental arm images, see docker-seleniarm.

Standalone mode

Here is an example of running selenium standalone and connecting Playwright to it. Note that hub and node are on the same `localhost`, and we pass `SE_NODE_GRID_URL` environment variable pointing to it.

First start Selenium.

```
docker run -d -p 4444:4444 --shm-size="2g" -e
SE_NODE_GRID_URL="http://localhost:4444" selenium/standalone-chrome:4.3.0-
20220726

# Alternatively for arm architecture
docker run -d -p 4444:4444 --shm-size="2g" -e
SE_NODE_GRID_URL="http://localhost:4444" seleniarm/standalone-
chromium:103.0
```

Then run Playwright.

```
SELENIUM_REMOTE_URL=http://localhost:4444 npx playwright test
```

Hub and nodes mode

Here is an example of running selenium hub and a single selenium node, and connecting Playwright to the hub. Note that hub and node have different IPs, and we pass `SE_NODE_GRID_URL` environment variable pointing to the hub when starting node containers.

First start the hub container and one or more node containers.

```
docker run -d -p 4442-4444:4442-4444 --name selenium-hub
selenium/hub:4.3.0-20220726
docker run -d -p 5555:5555 \
    --shm-size="2g" \
    -e SE_EVENT_BUS_HOST=<selenium-hub-ip> \
    -e SE_EVENT_BUS_PUBLISH_PORT=4442 \
    -e SE_EVENT_BUS_SUBSCRIBE_PORT=4443 \
    -e SE_NODE_GRID_URL="http://<selenium-hub-ip>:4444"
    selenium/node-chrome:4.3.0-20220726

# Alternatively for arm architecture
docker run -d -p 4442-4444:4442-4444 --name selenium-hub
seleniarm/hub:4.3.0-20220728
docker run -d -p 5555:5555 \
    --shm-size="2g" \
    -e SE_EVENT_BUS_HOST=<selenium-hub-ip> \
    -e SE_EVENT_BUS_PUBLISH_PORT=4442 \
    -e SE_EVENT_BUS_SUBSCRIBE_PORT=4443 \
    -e SE_NODE_GRID_URL="http://<selenium-hub-ip>:4444"
    seleniarm/node-chromium:103.0
```

Then run Playwright.

```
SELENIUM_REMOTE_URL=http://<selenium-hub-ip>:4444 npx playwright test
```

Selenium 3

Internally, Playwright connects to the browser using Chrome DevTools Protocol websocket. Selenium 4 exposes this capability, while Selenium 3 does not.

This means that Selenium 3 is supported in a best-effort manner, where Playwright tries to connect to the grid node directly. Grid nodes must be directly accessible from the machine that runs Playwright.

Previous
Continuous Integration

Next
Supported languages