

HolyJS

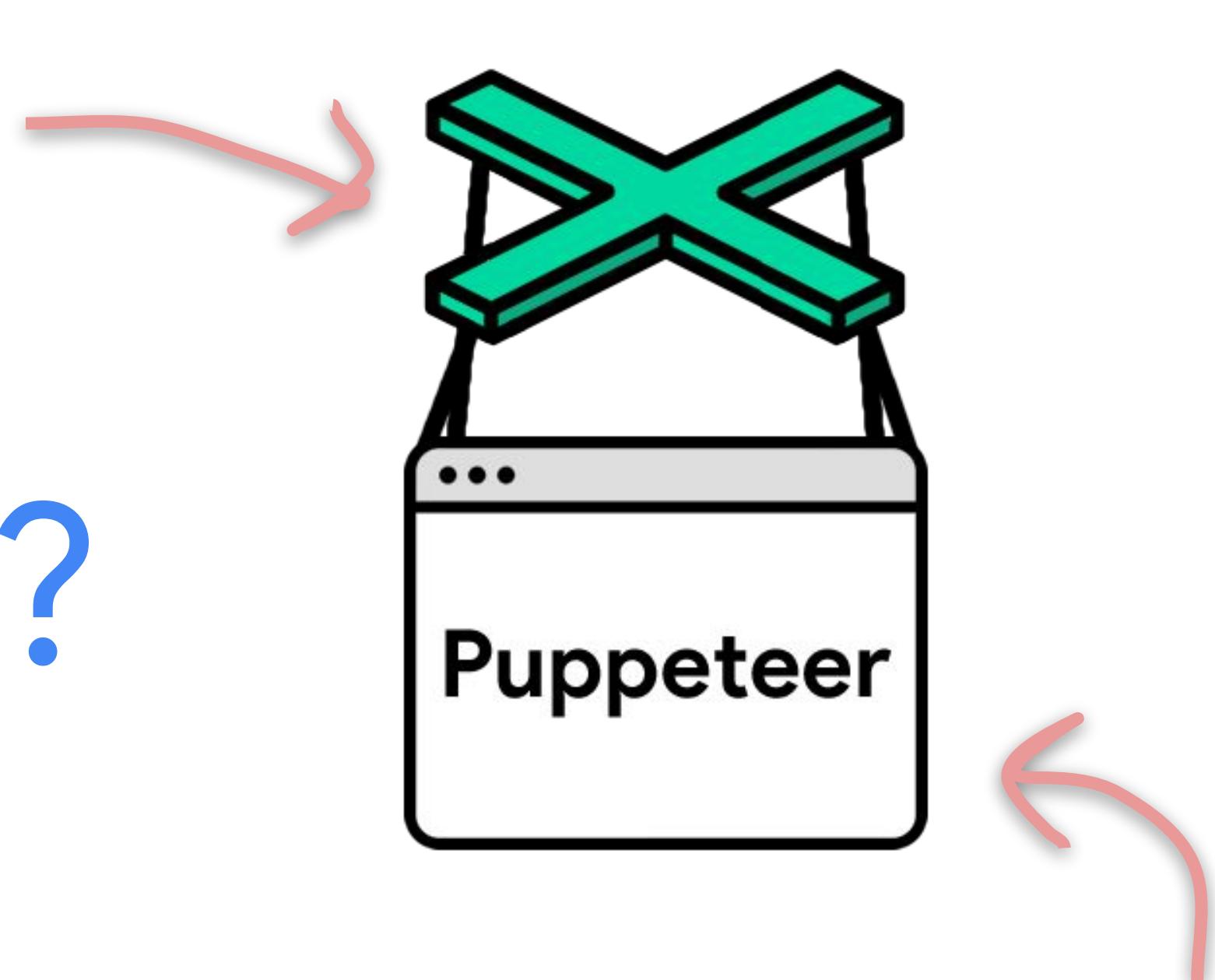
2019

Modern Web Testing And Automation with Puppeteer

Agenda

- 👉 What is Puppeteer?
- 👉 Announcement 🎁
- 👉 Testing Fundamentals
- 👉 Testing Modern Web
- 👉 Community 👤

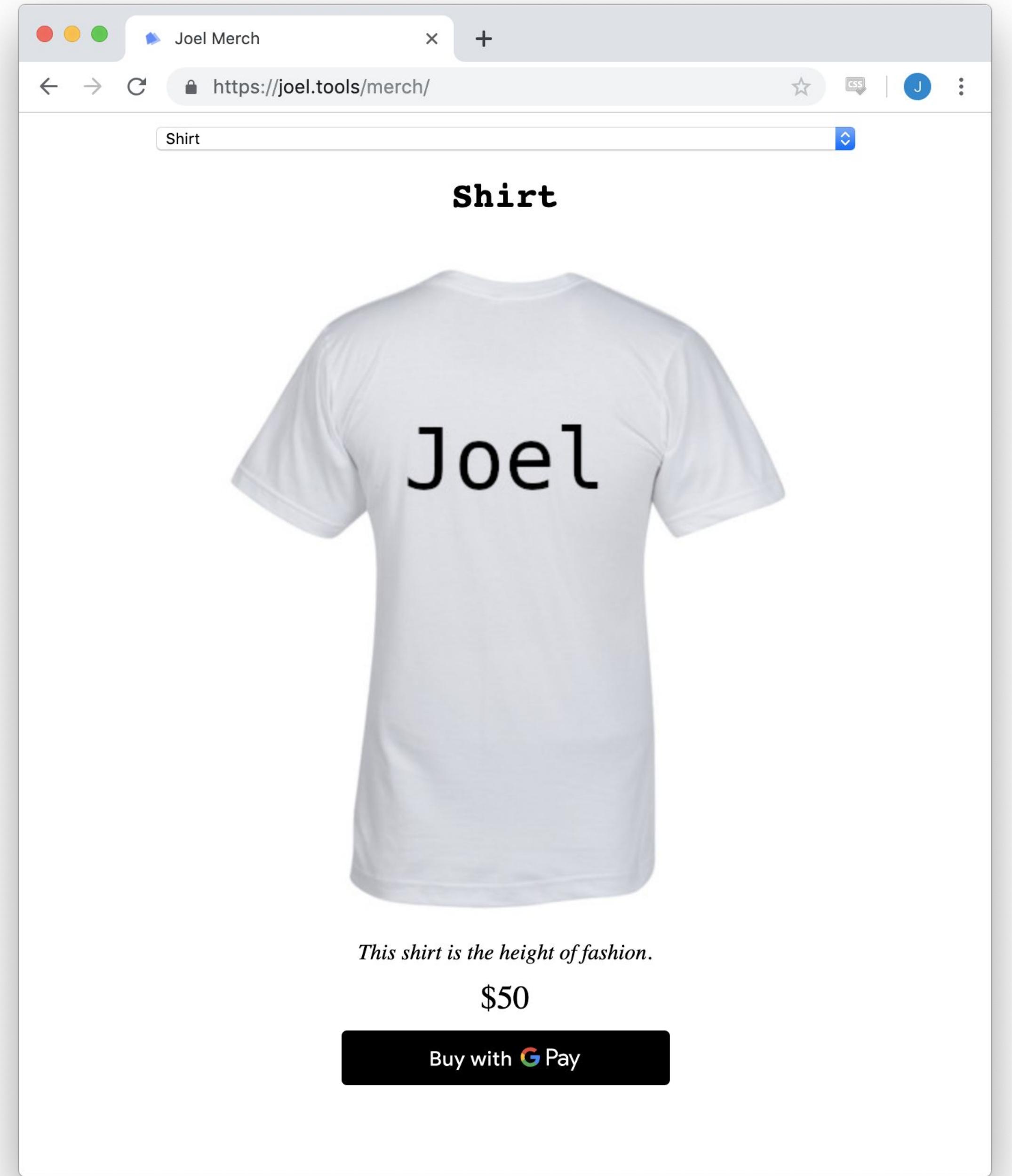




What is Puppeteer?

- 👉 Browser Automation Library
- 👉 Puppeteer = Node.js + Chrome
- 👉 Open pages, navigate to websites, evaluate javascript



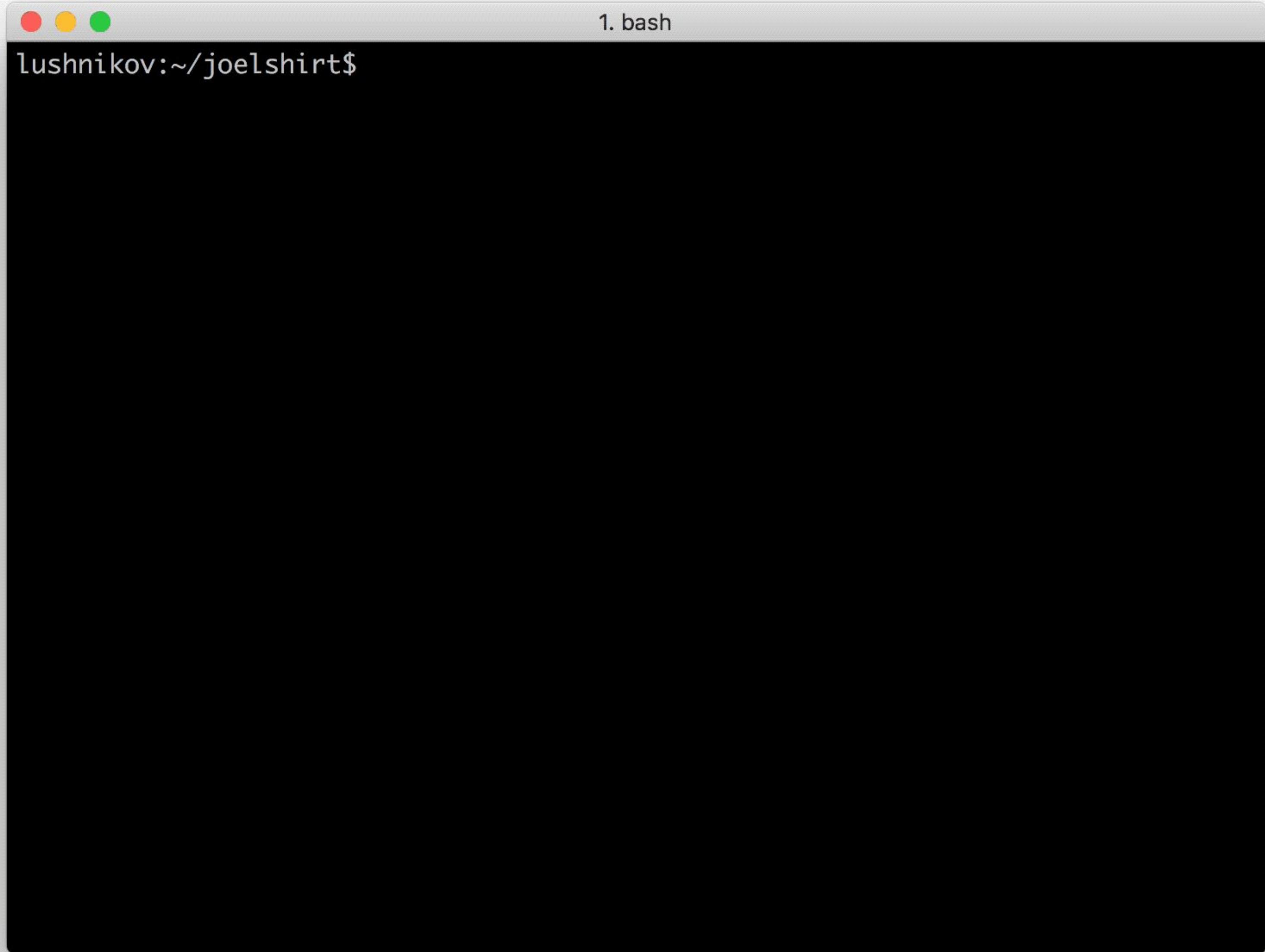


\$50



Puppeteer





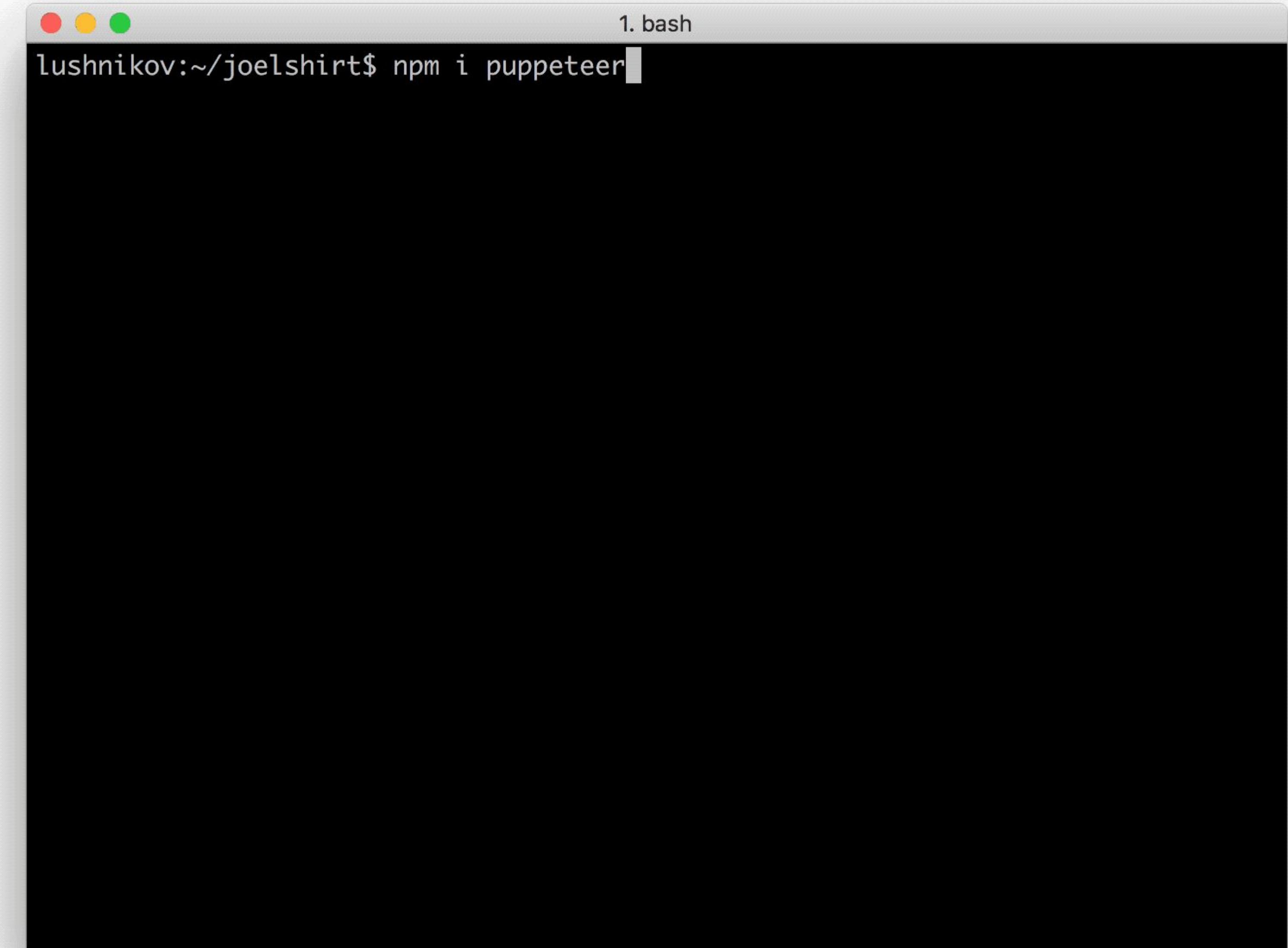
lushnikov:~/joelshirt\$

1. bash



Puppeteer





lushnikov:~/joelshirt\$ npm i puppeteer



```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

Downloading Chromium r650583 - 87.7 Mb [==] 10% 5.7s
```



```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

Downloading Chromium r650583 - 87.7 Mb [===== ] 38% 4.2s
```



```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

Downloading Chromium r650583 - 87.7 Mb [=====] 82% 1.1s
```



Puppeteer

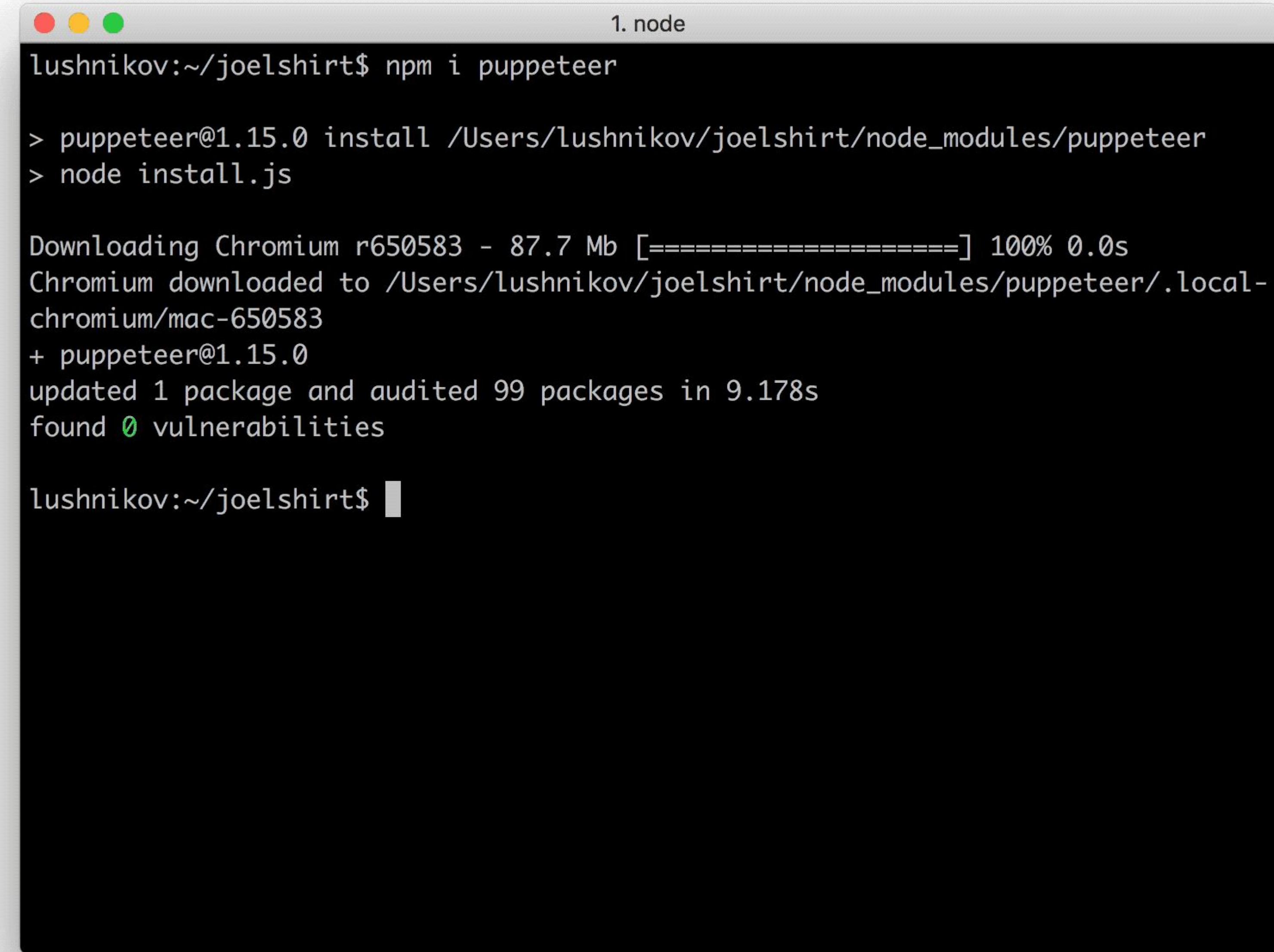


```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

Downloading Chromium r650583 - 87.7 Mb [=====] 100% 0.0s
```





```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

Downloading Chromium r650583 - 87.7 Mb [=====] 100% 0.0s
Chromium downloaded to /Users/lushnikov/joelshirt/node_modules/puppeteer/.local-
chromium/mac-650583
+ puppeteer@1.15.0
updated 1 package and audited 99 packages in 9.178s
found 0 vulnerabilities

lushnikov:~/joelshirt$
```



```
const puppeteer = require('puppeteer');

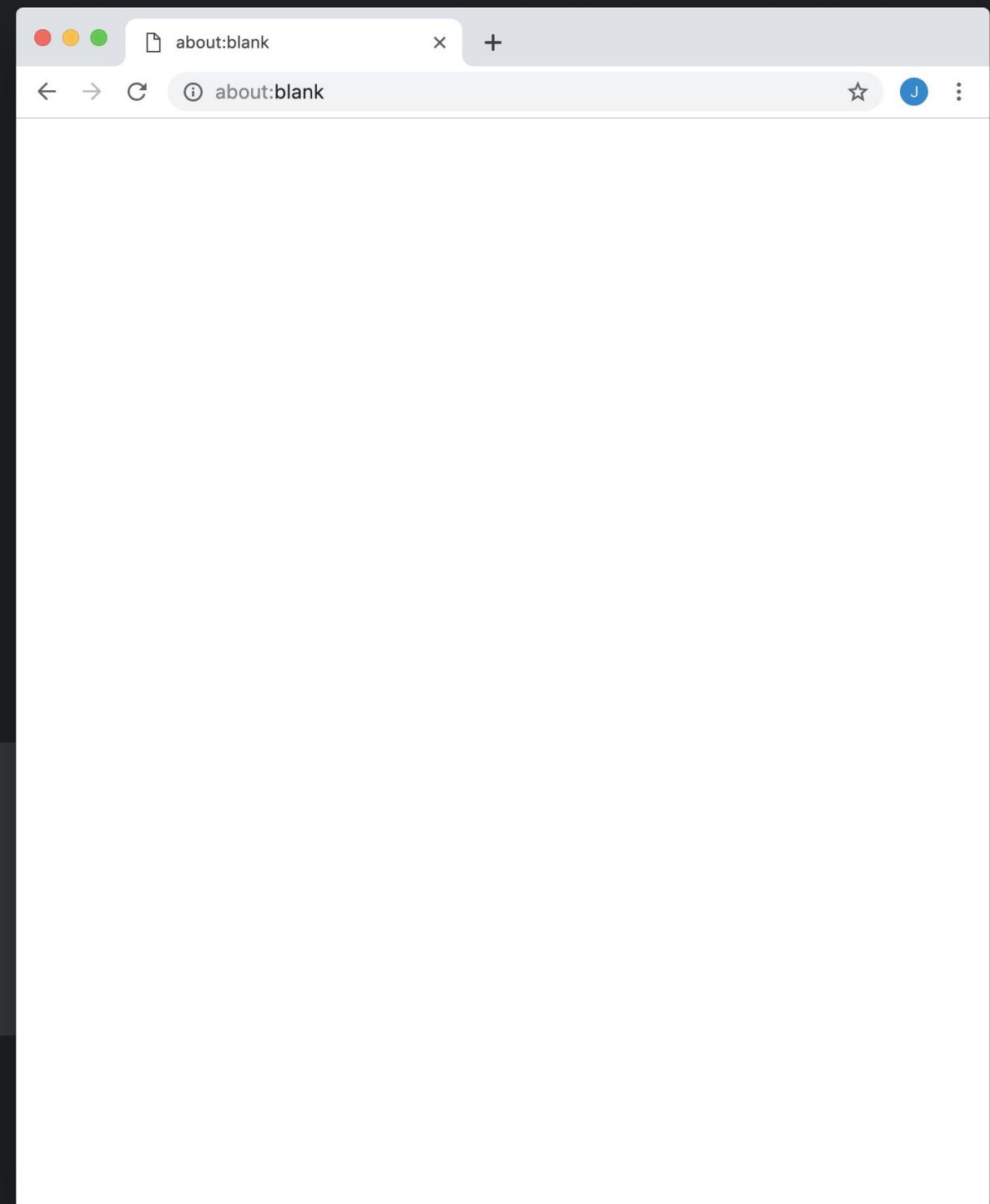
(async () => {
  // Automate here
})();
```

```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
})();
```

```
const puppeteer = require('puppeteer');

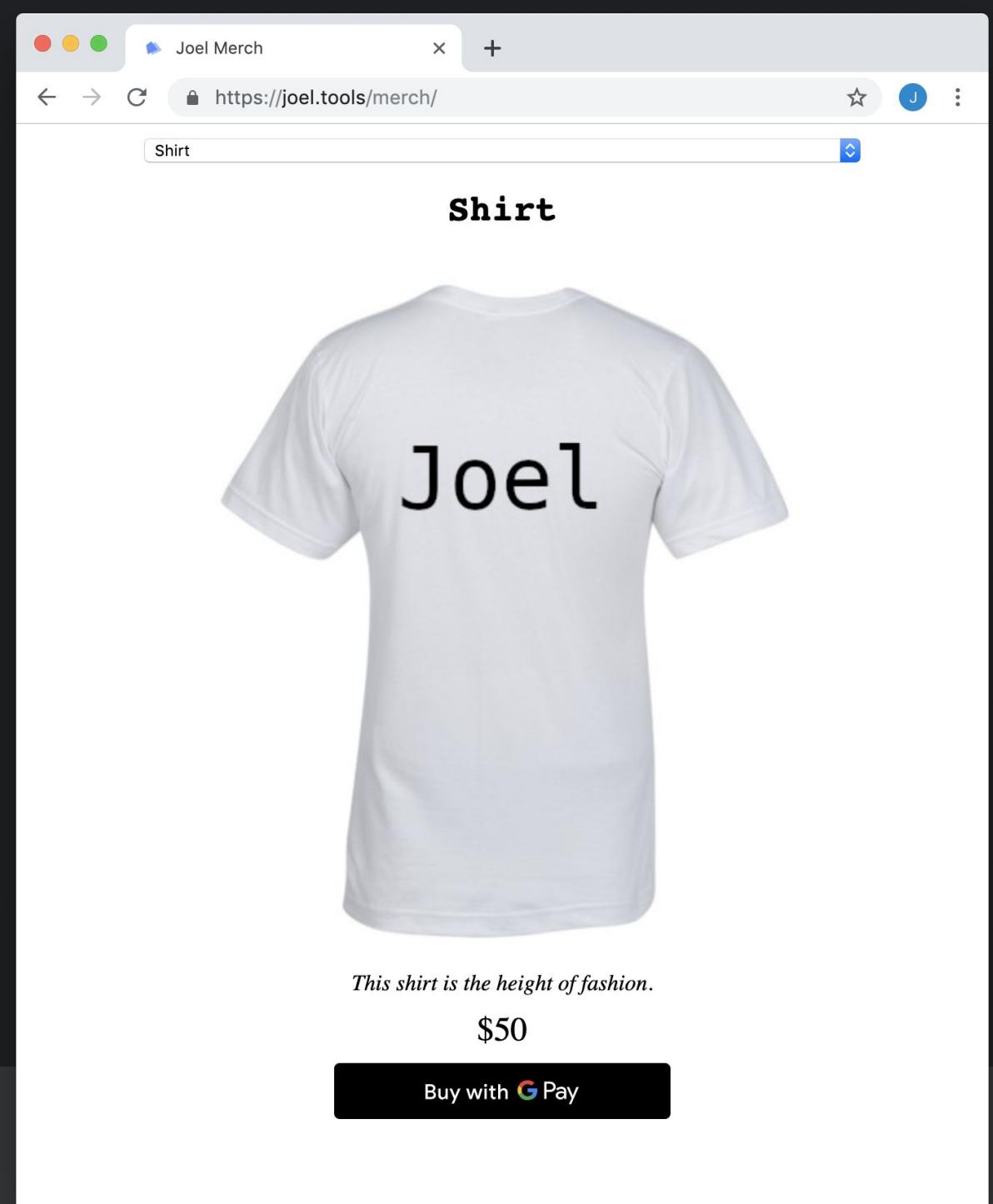
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');

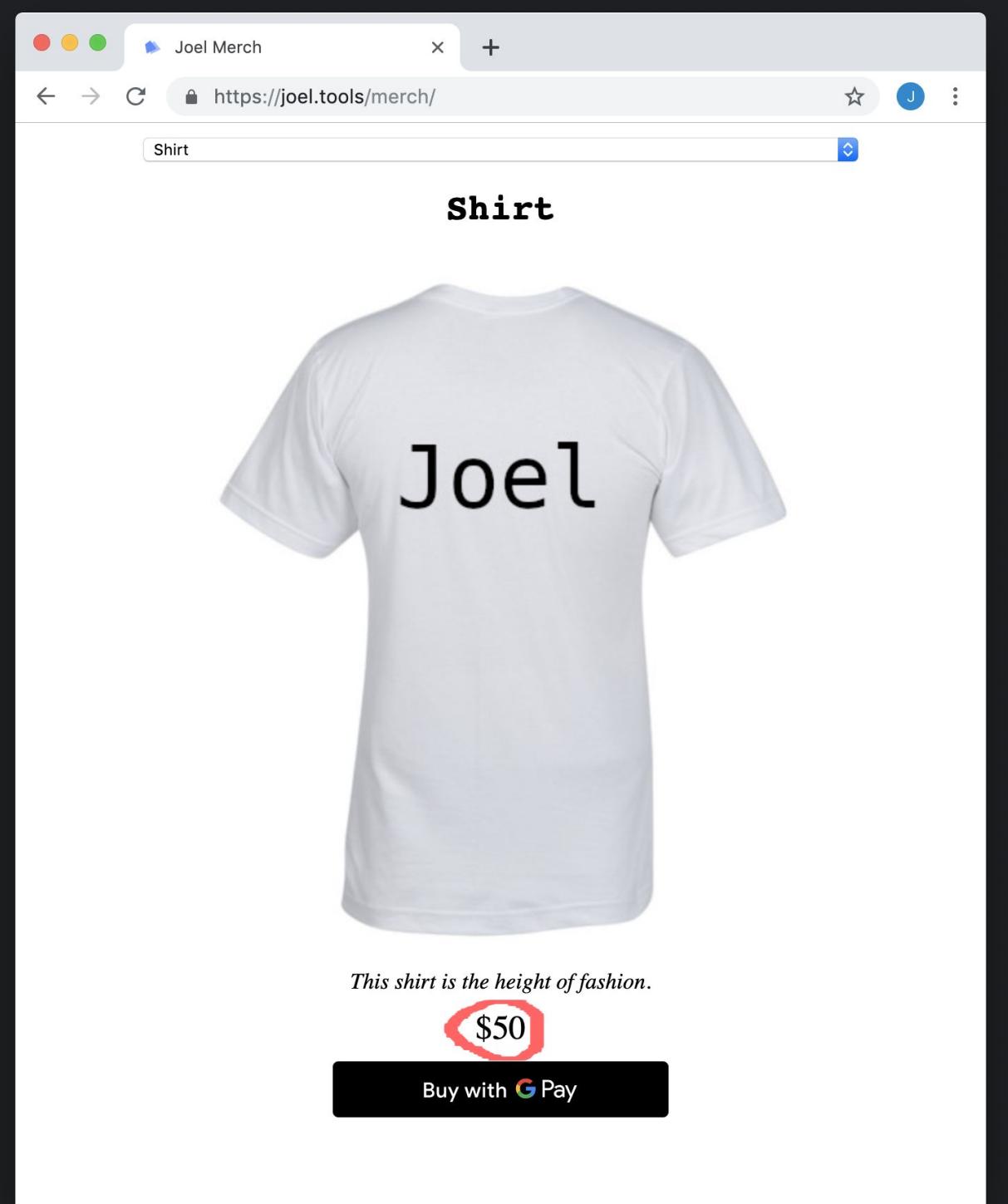
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');

  const price = await page.$eval('.price', div => div.textContent);
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  console.log(price);
})();
```



\$50

```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  console.log(price);
  await browser.close();
})();
```

\$50

Joel Shirt Prices





Puppeteer for Firefox

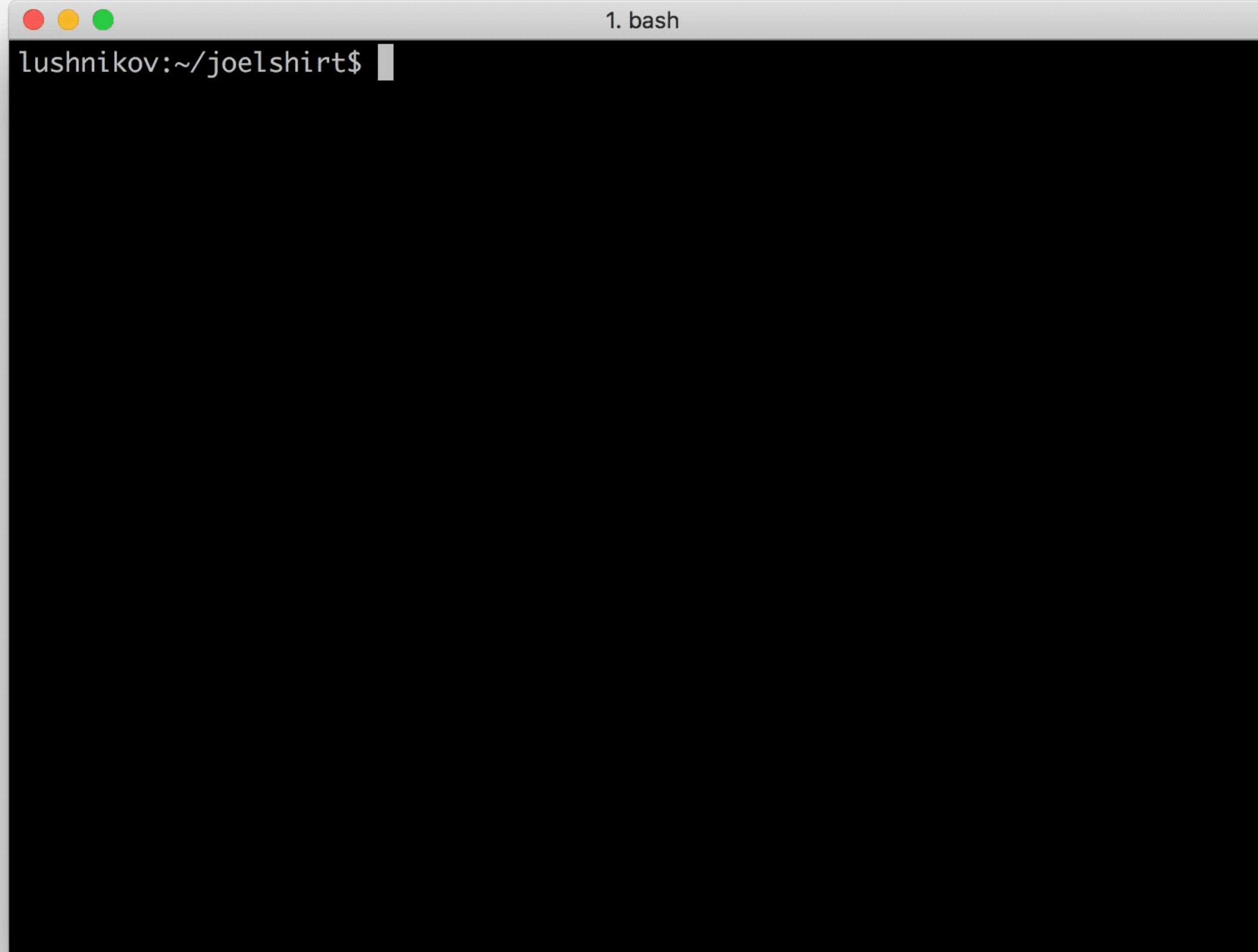
puppeteer-firefox

🐊 **experimental** project

👉 **npm i puppeteer-firefox**

👉 supports **90%** of Puppeteer API





1. bash

lushnikov:~/joelshirt\$



1. bash

```
lushnikov:~/joelshirt$ npm i puppeteer-firefox
```



Puppeteer

1. node

```
lushnikov:~/joelshirt$ npm i puppeteer-fxox

> puppeteer-fxox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-fxox
> node install.js

Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [| ] 5% 4.0s
```



Puppeteer

1. node

```
lushnikov:~/joelshirt$ npm i puppeteer-fxox

> puppeteer-fxox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-fxox
> node install.js

Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [|||||] 26% 2.0s
```



Puppeteer

1. node

```
lushnikov:~/joelshirt$ npm i puppeteer-firefox

> puppeteer-firefox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-firefox
> node install.js

Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [=====] 59% 1.1s
```



Puppeteer

1. node

```
lushnikov:~/joelshirt$ npm i puppeteer-fxox

> puppeteer-fxox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-fxox
> node install.js

Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [=====] 100% 0.0s
```



Puppeteer

1. bash

```
lushnikov:~/joelshirt$ npm i puppeteer-fx

> puppeteer-fx@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-fx
> node install.js

Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [=====] 100% 0.0s
Firefox downloaded to /Users/lushnikov/joelshirt/node_modules/puppeteer-fx/
.local-browser/firefox-mac-2f959d575a3d61f5dda12e4e2dca1f46a92ab569
Firefox preferences installed!
+ puppeteer-fx@0.5.0
added 1 package from 1 contributor and audited 149 packages in 5.391s
found 0 vulnerabilities

lushnikov:~/joelshirt$
```



Puppeteer

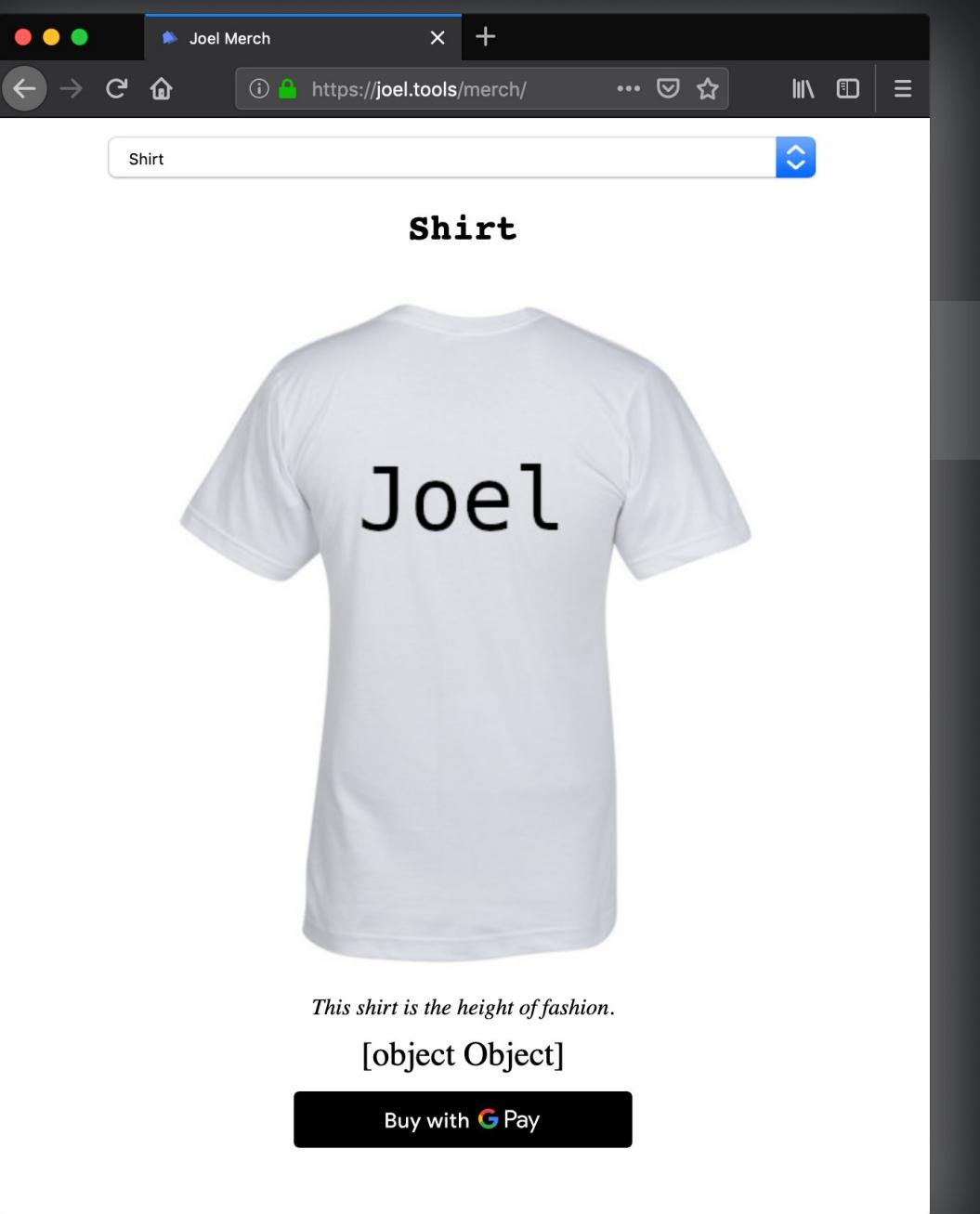
```
const puppeteer = require('puppeteer-firefox');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  console.log(price);
  await browser.close();
})();
```

```
const puppeteer = require('puppeteer-firefox');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  console.log(price);
  await browser.close();
})();
```

[object Object]





Joel Merch

https://joel.tools/merch/

Shirt

Shirt

This shirt is the height of fashion.

[object Object]

Buy with G Pay

[object Object]



Web Testing Challenges

- 👉 web testing is **complicated**
- 👉 web testing is **slow**
- 👉 web testing is **flaky**

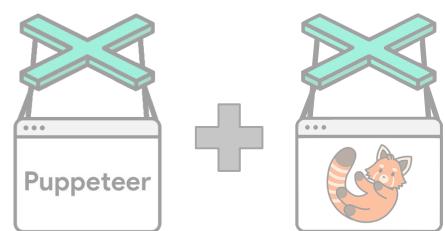


New Web Testing

Powered by Puppeteer

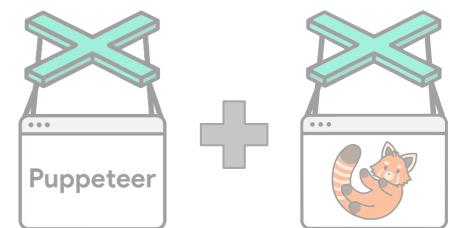
~~Simple Complicated Setup~~

- 👉 same thing as any NPM library
- 👉 any test runner: jest, ava, mocha...

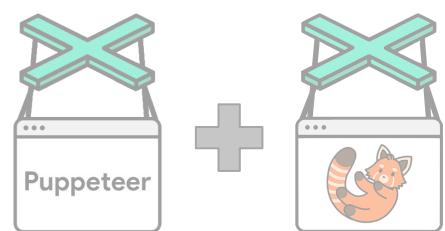
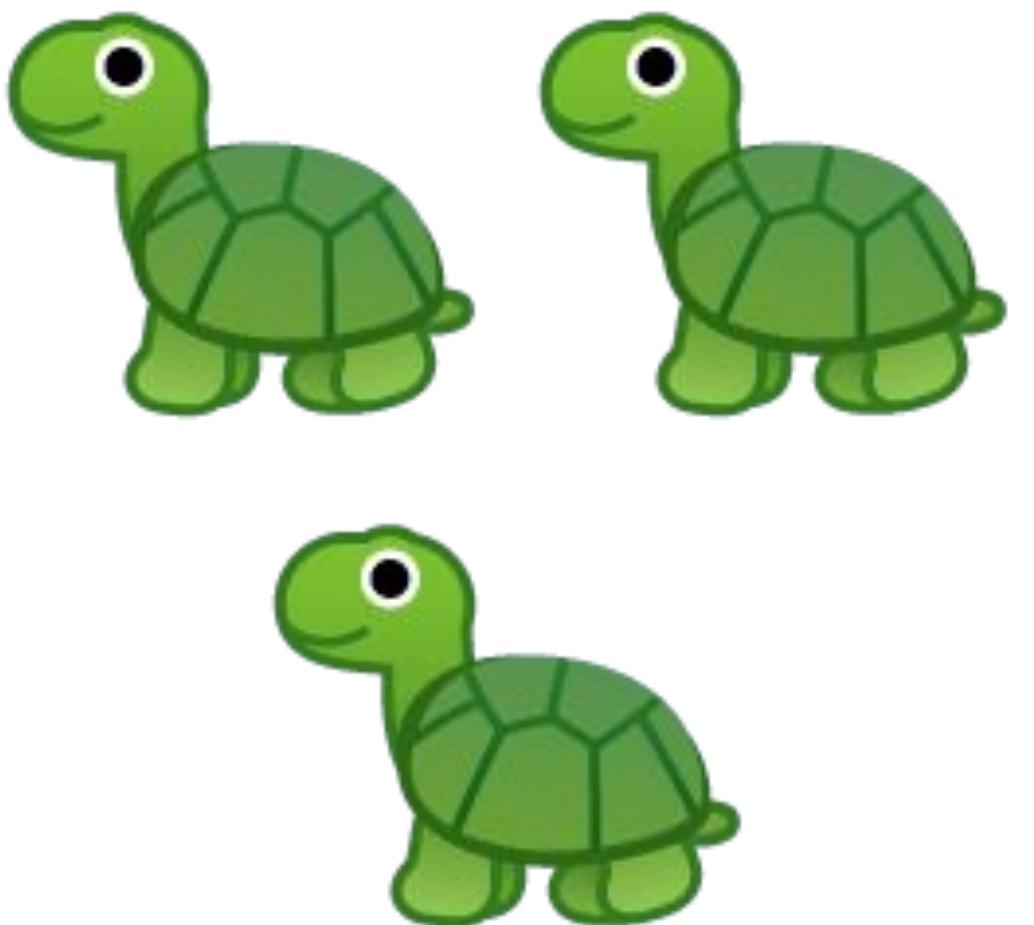


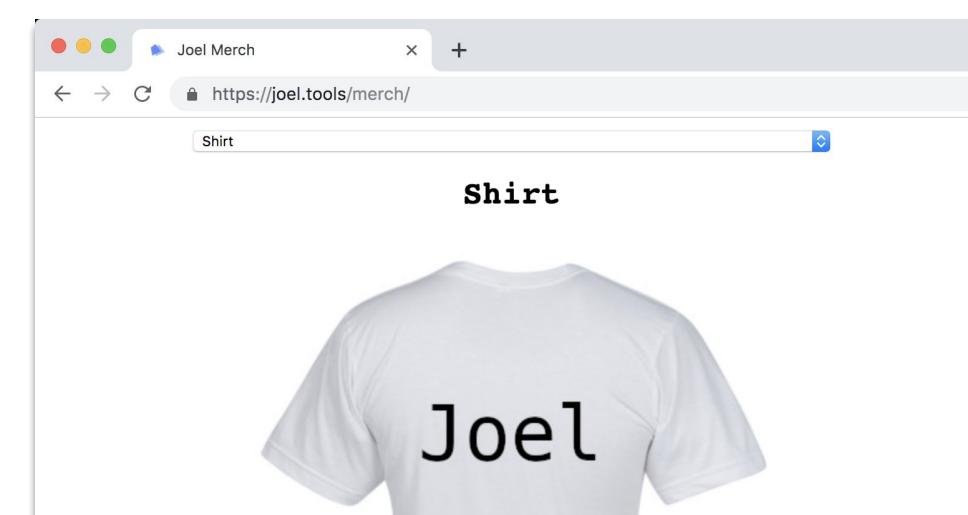
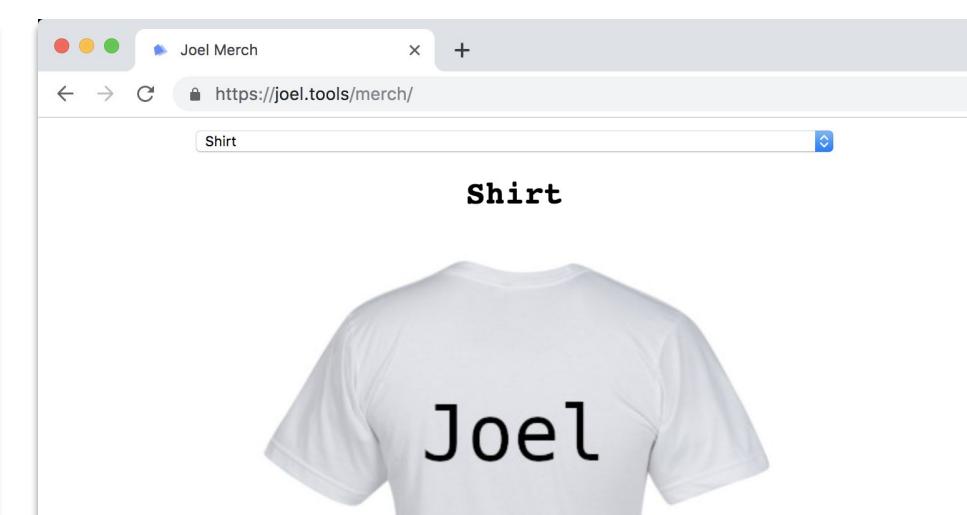
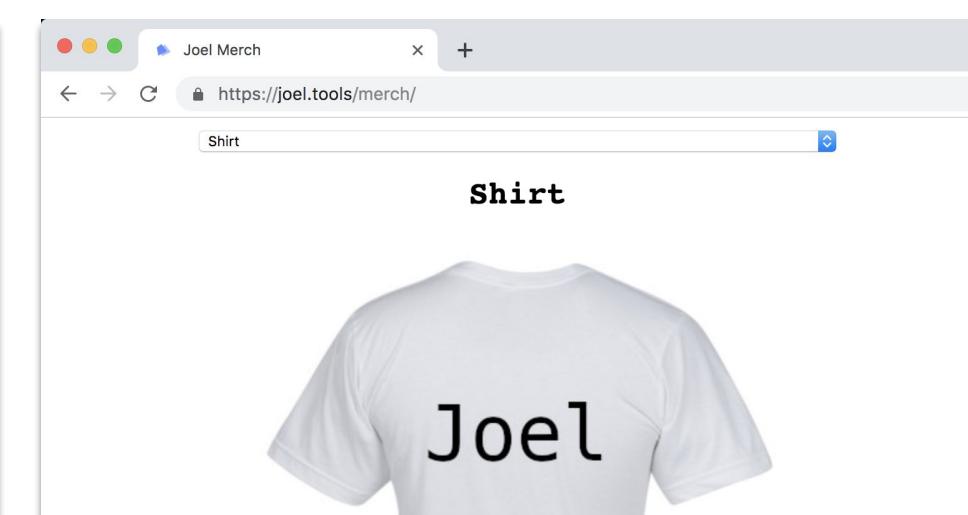
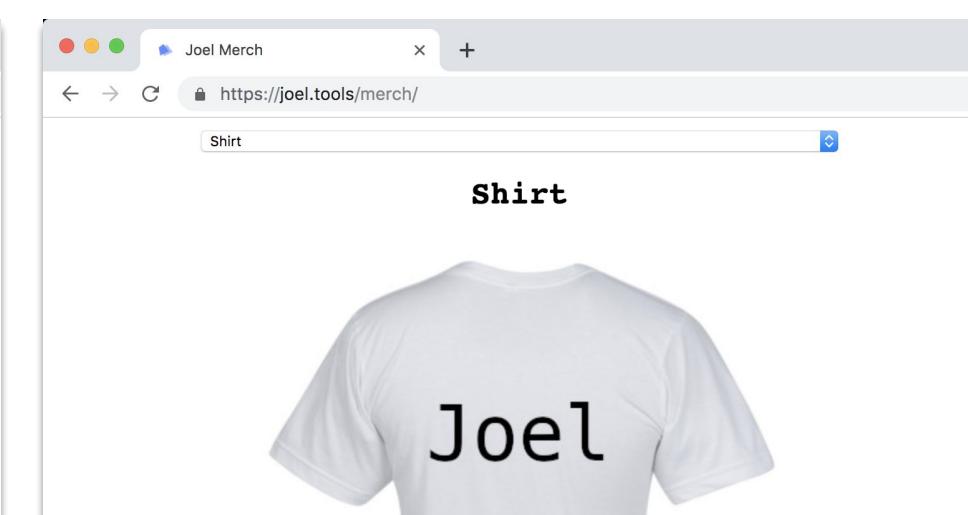
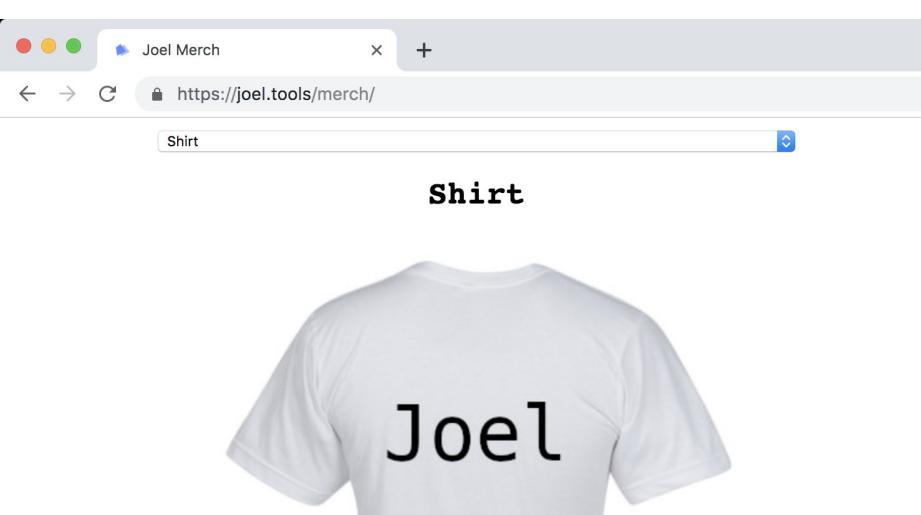
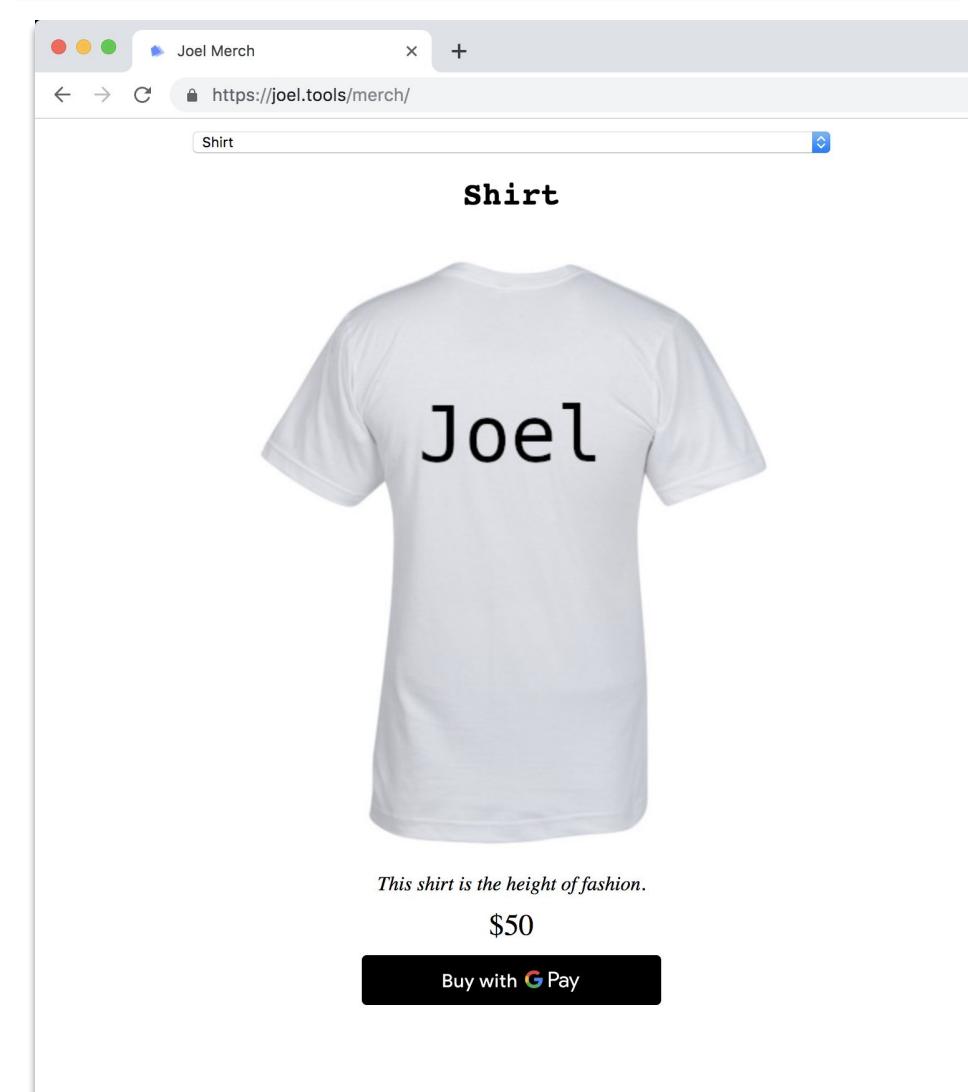
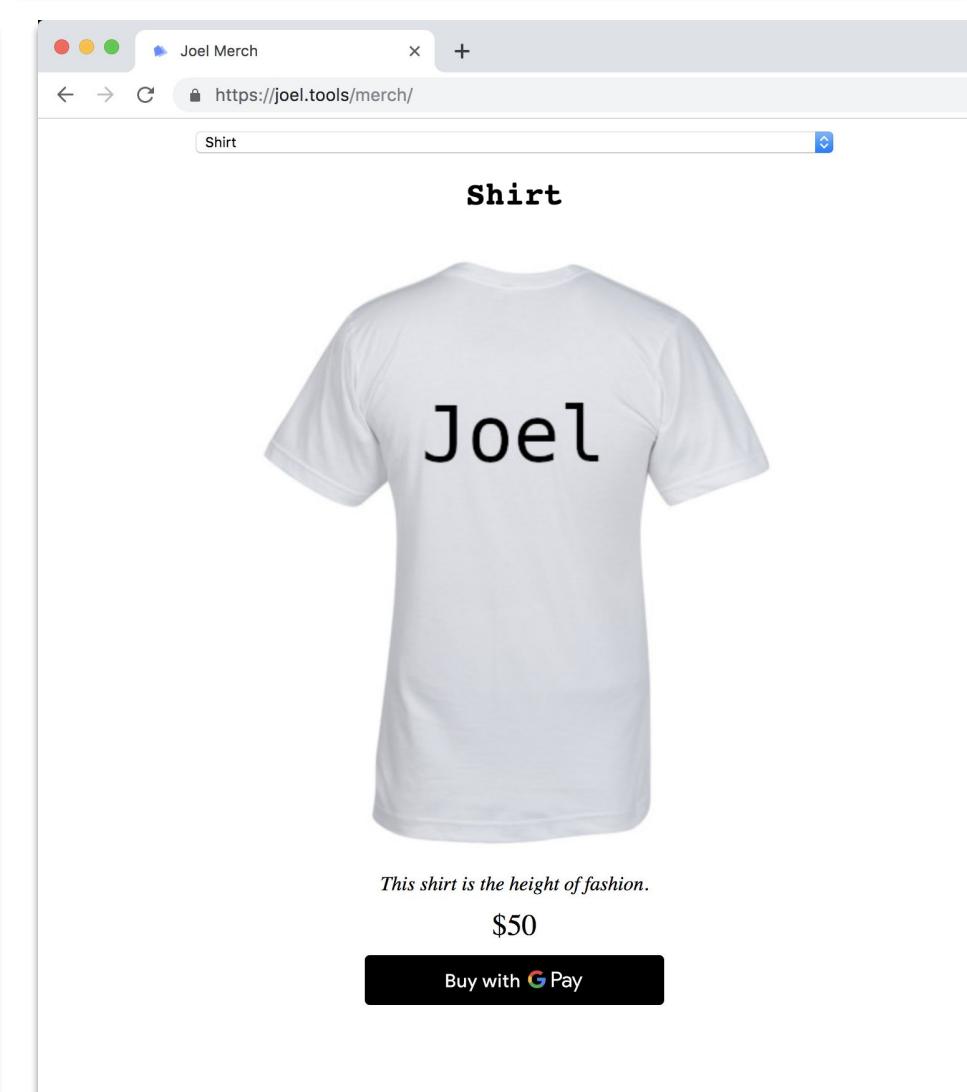
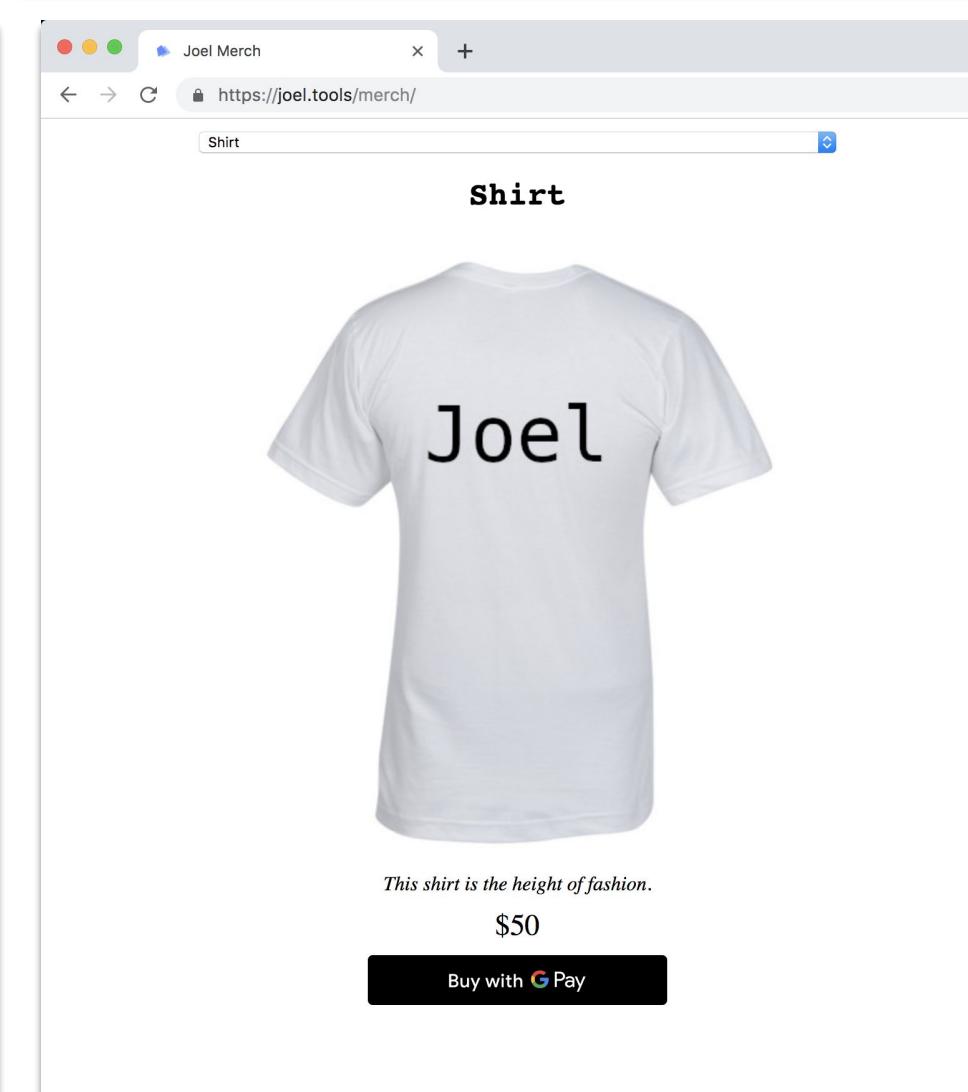
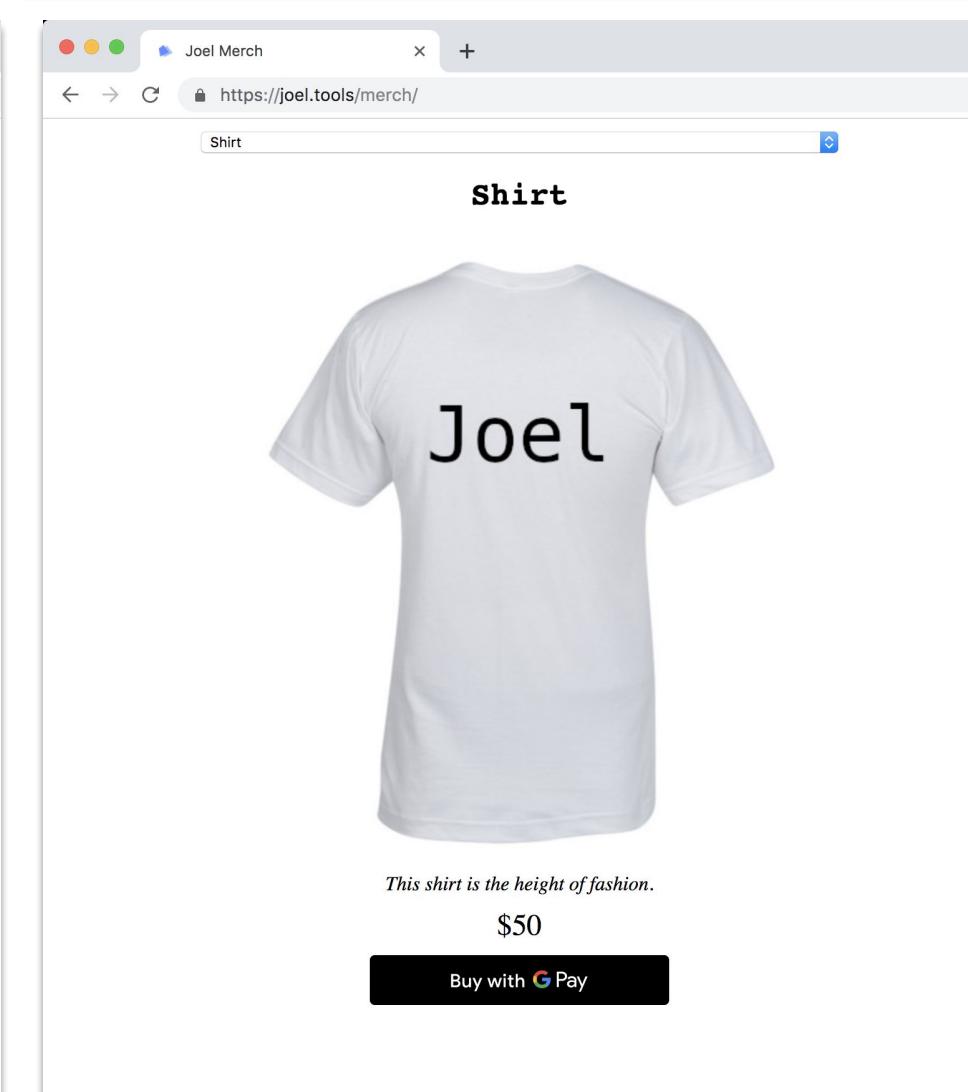
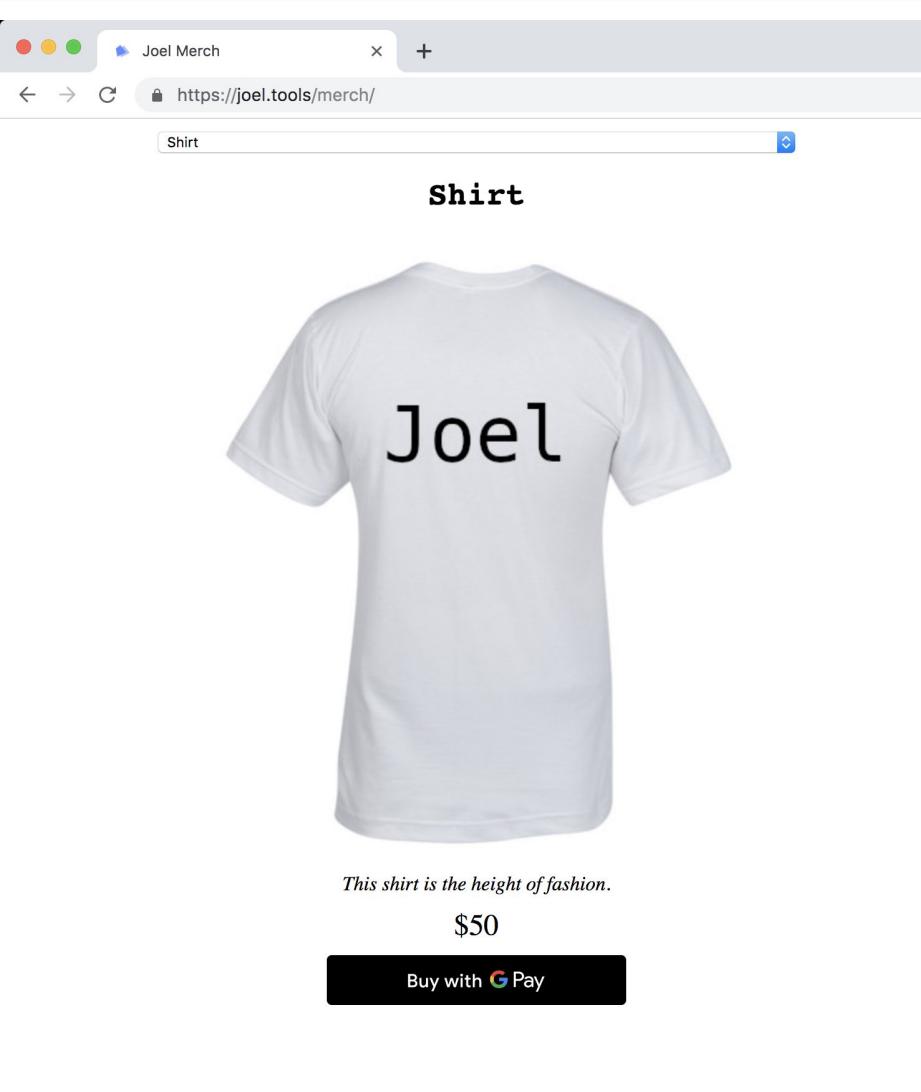
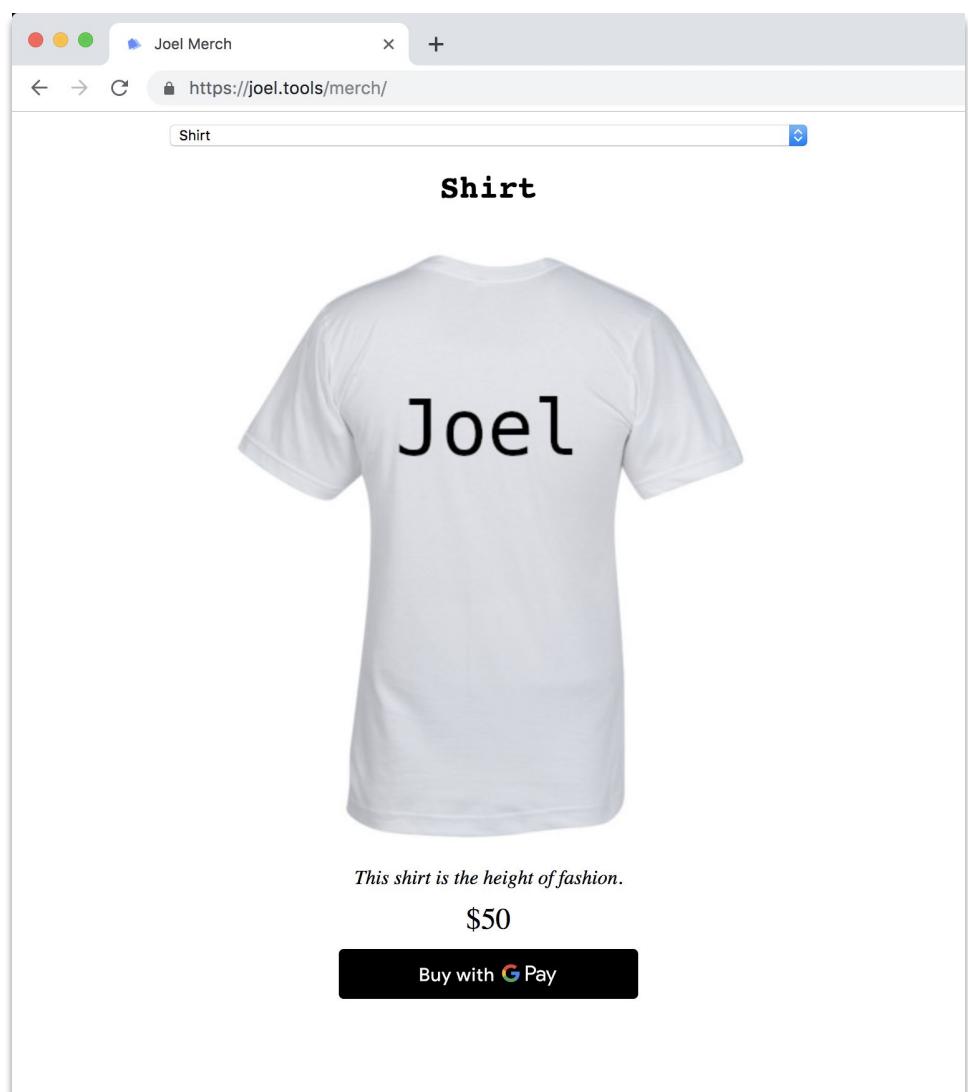
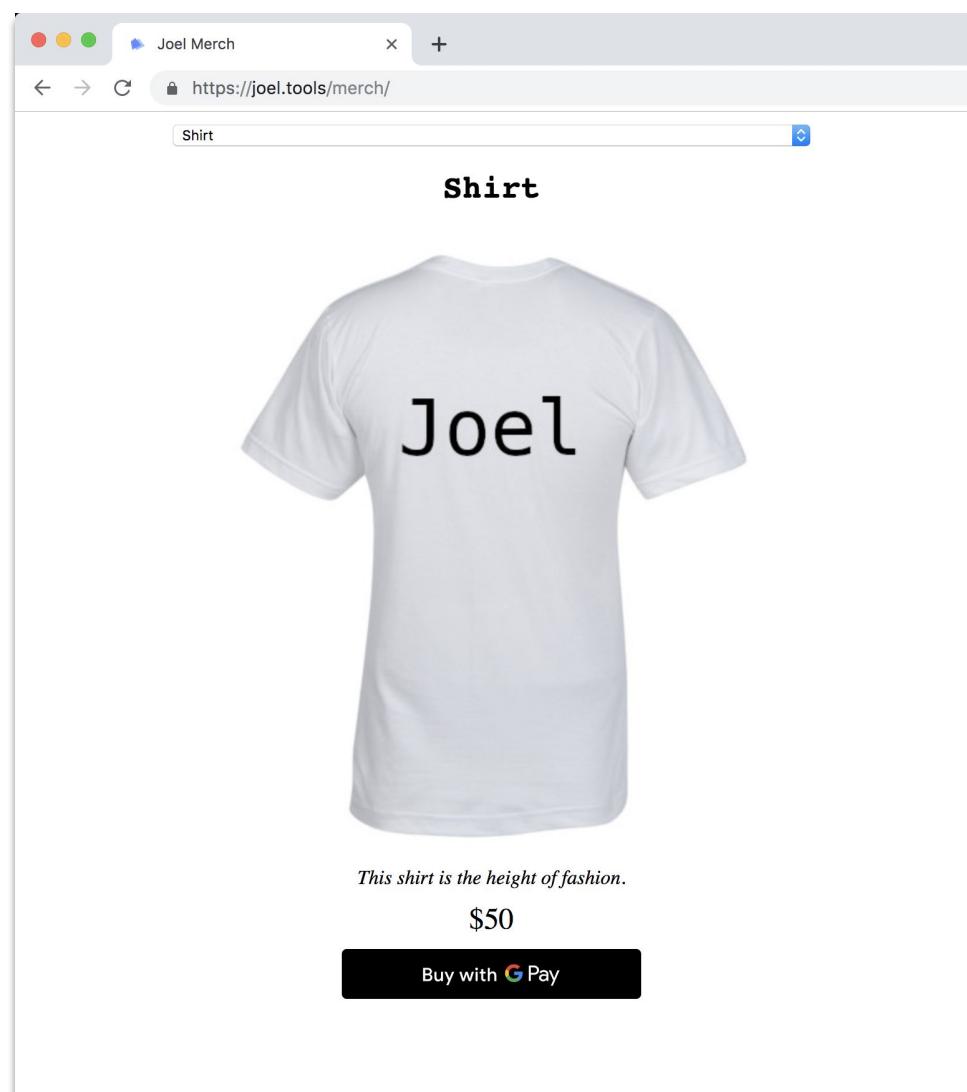
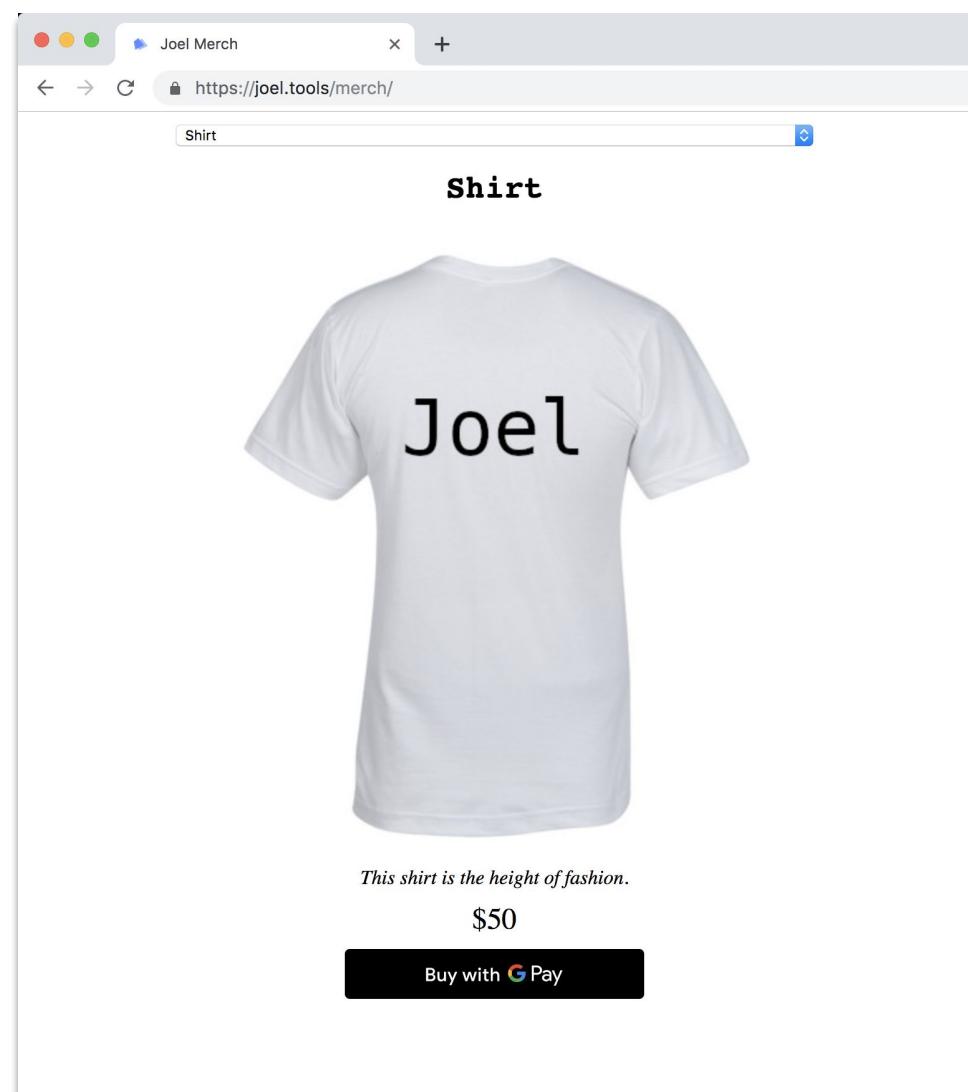
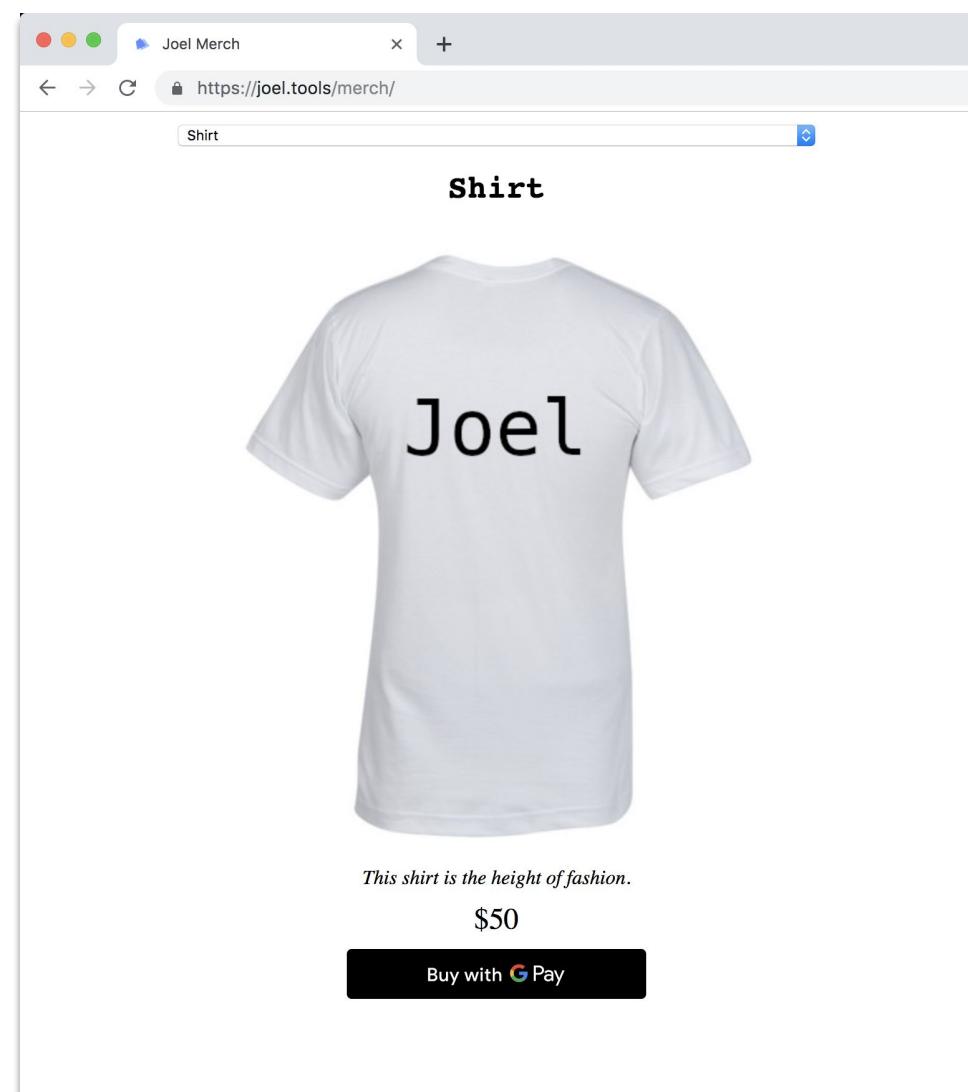
Runs Everywhere

- ✓ **Desktop:** Mac, Windows, Linux
- ✓ **CI:** Travis-CI, AppVeyor, ...
- ✓ **Cloud:** GCP, AWS, Azure, ...
- ✓ **Docker Containers**



Tests are Slow



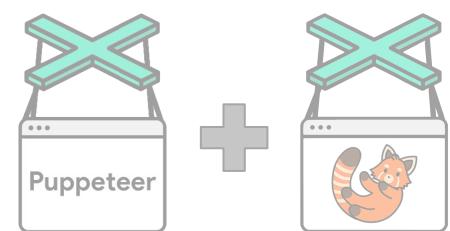


Fast

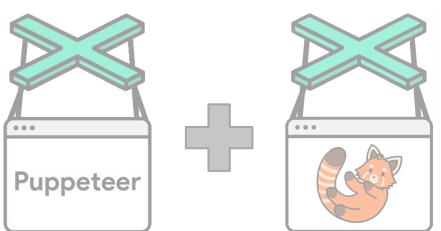
Tests are ~~Slow~~



Browser Contexts



100x faster

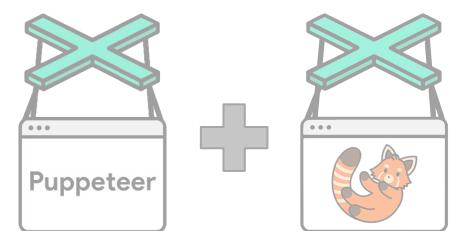
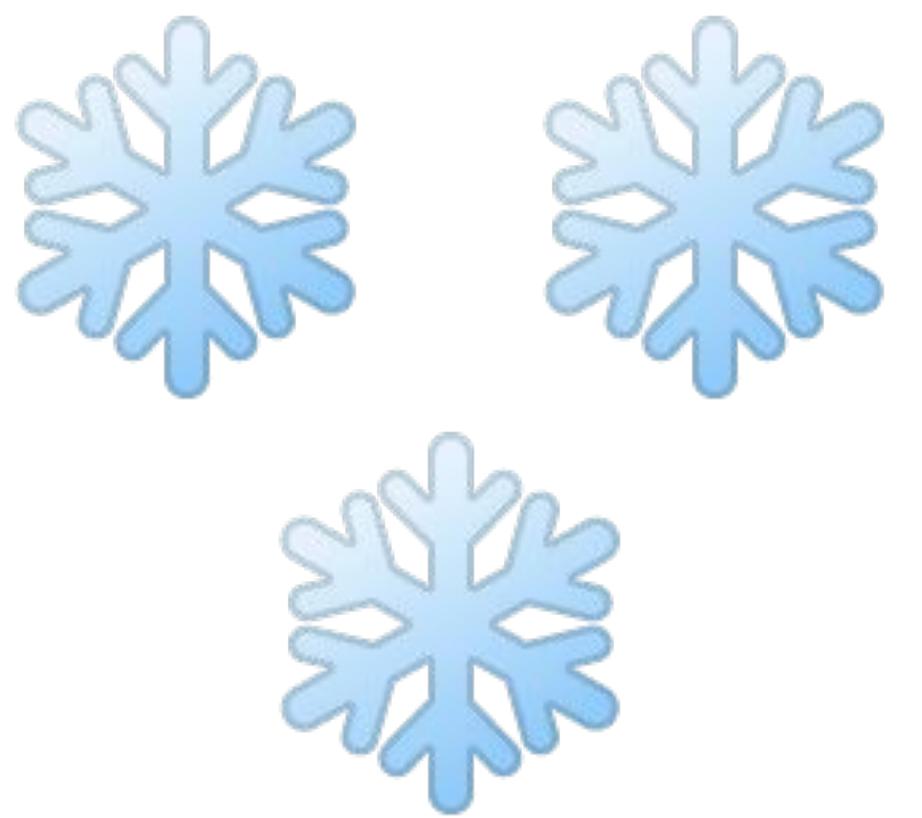


```
it('should have a pay button', async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  expect(await page.$('button.gpay-button')).toBeTruthy();
  await browser.close();
})
```

```
const browser = await puppeteer.launch();

...
it('should have a pay button', async () => {
  const context = await browser.createIncognitoBrowserContext();
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  expect(await page.$('button.gpay-button')).toBeTruthy();
  await context.close();
})
```

Tests are Flaky



```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.click('button.gpay-button');

  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitFor(1000);
  await page.click('button.gpay-button');
  await page.waitFor(1000);
  assert(await page.$('.successful-purchase')));
}) ;
```

~~Tests are Flaky~~ reliable

- 👉 `page.waitForSelector()`
- 👉 `page.waitForRequest()`
- 👉 `page.waitForResponse()`
- 👉 `page.waitForFunction()`
- 👉 `page.waitForNavigation()`



```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitFor(1000);
  await page.click('button.gpay-button');
  await page.waitFor(1000);
  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitFor(1000);
  await page.click('button.gpay-button');
  await page.waitFor(1000);
  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');
  await page.click('button.gpay-button');
  await page.waitFor(1000);
  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');
  await page.click('button.gpay-button');
  await page.waitFor(1000);
  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  await page.click('button.gpay-button');
  await page.waitFor(1000);

  assert(await page.$('.successful-purchase')));
});
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  await page.click('button.gpay-button');
  await page.waitFor(1000);

  assert(await page.$('.successful-purchase')));
});
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  await page.click('button.gpay-button');

  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  await page.click('button.gpay-button');

  assert(await page.$('.successful-purchase')));
}) ;
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  const response = page.waitForResponse(res => res.url().endsWith('/pay'));
  await page.click('button.gpay-button');

  assert(await page.$('.successful-purchase')));
});
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  const response = page.waitForResponse(res => res.url().endsWith('/pay'));
  await page.click('button.gpay-button');

  assert(await page.$('.successful-purchase')));
});
```

```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

  const response = page.waitForResponse(res => res.url().endsWith('/pay'));
  await page.click('button.gpay-button');
  await response;

  assert(await page.$('.successful-purchase')));
});
```

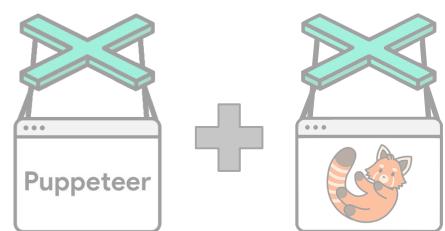
```
it('should pay', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch/');
  await page.waitForSelector('button.gpay-button');

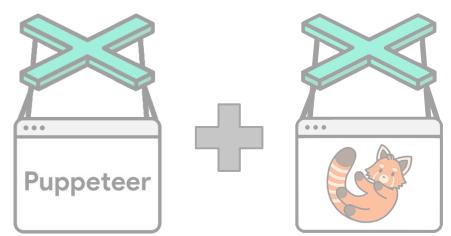
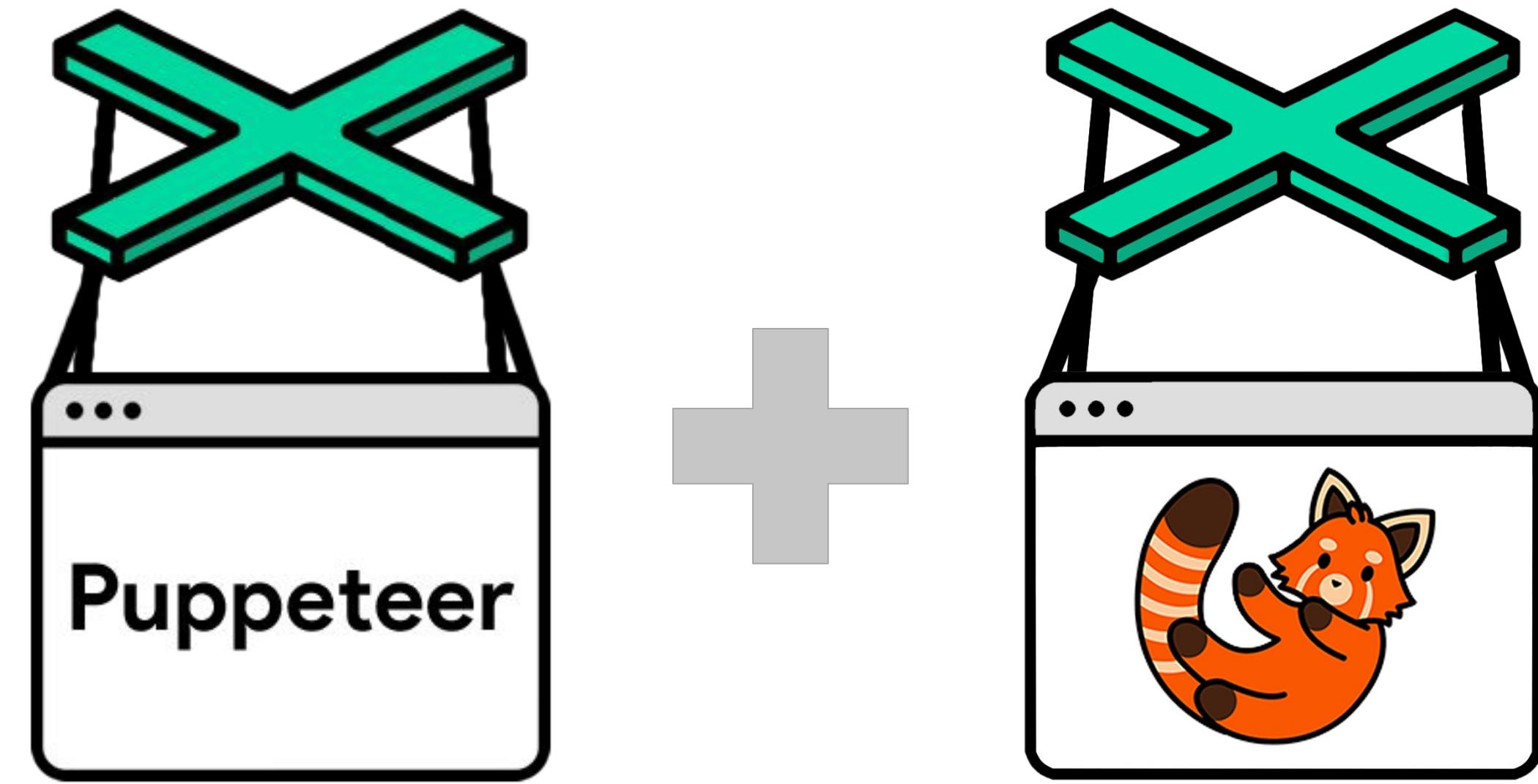
  await Promise.all([
    page.waitForResponse(res => res.url().endsWith('/pay')),
    page.click('button.gpay-button'),
  ]);

  assert(await page.$('.successful-purchase')));
});
```

Recap

- 👉 Testing with Puppeteer is **fast**
- 👉 Testing with Puppeteer is **reliable**
- 👉 Testing with Puppeteer is **simple**



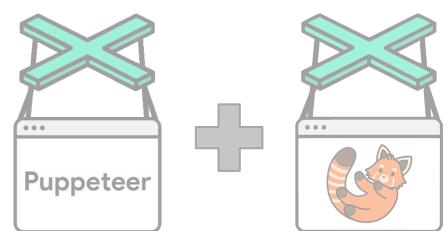


HOLY
JS!... 2019
PITER

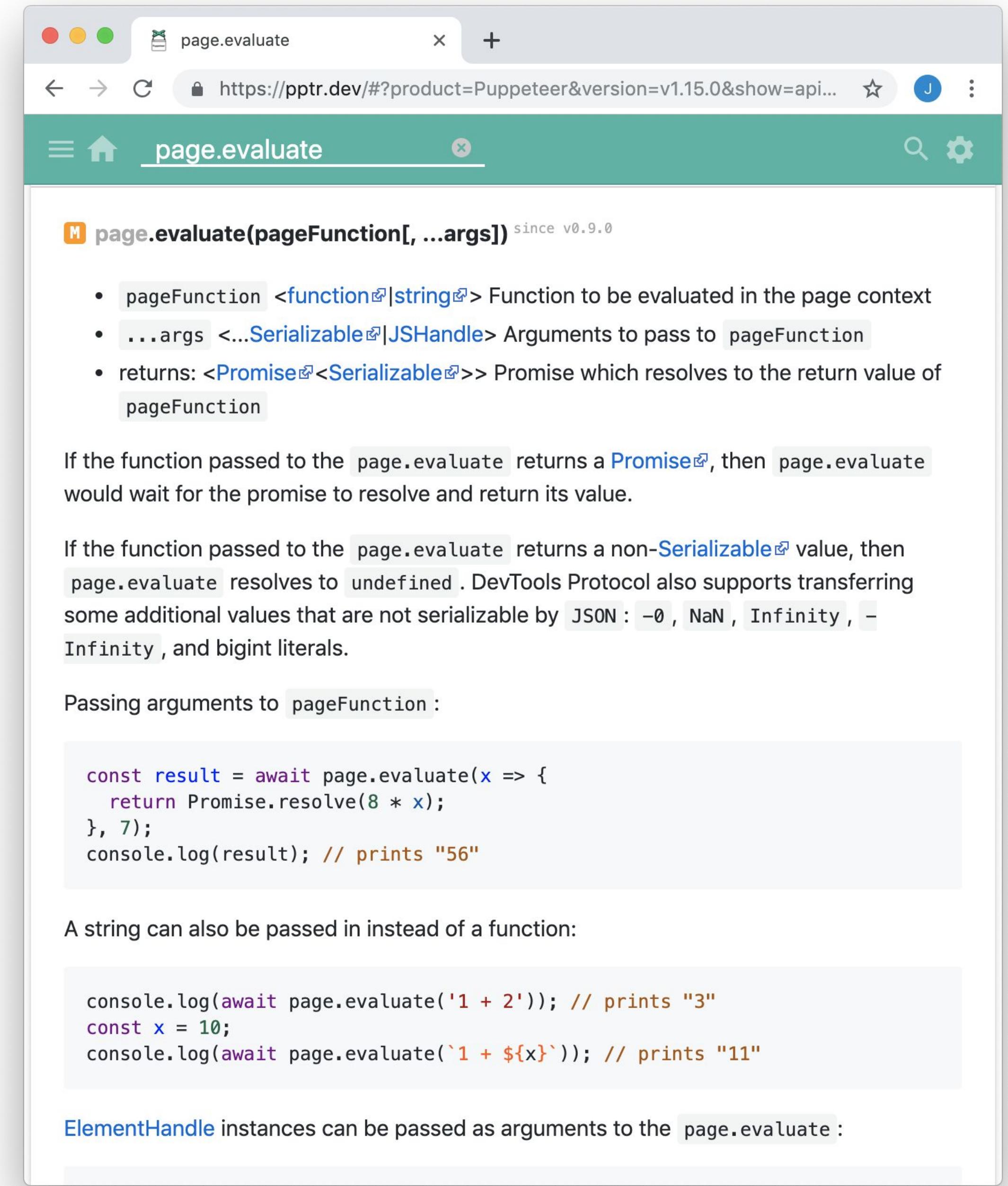
Testing Fundamentals

Web Testing Is...

- 👉 Navigation
- 👉 Evaluation
- 👉 Input Injection



pptr.dev



The screenshot shows a browser window with the title "page.evaluate" and the URL "https://pptr.dev/#?product=Puppeteer&version=v1.15.0&show=api...". The page content is as follows:

M page.evaluate(pageFunction[, ...args]) since v0.9.0

- `pageFunction <function|string>` Function to be evaluated in the page context
- `...args <...Serializable|JSHandle>` Arguments to pass to `pageFunction`
- returns: `<Promise<Serializable>>` Promise which resolves to the return value of `pageFunction`

If the function passed to the `page.evaluate` returns a `Promise`, then `page.evaluate` would wait for the promise to resolve and return its value.

If the function passed to the `page.evaluate` returns a non-`Serializable` value, then `page.evaluate` resolves to `undefined`. DevTools Protocol also supports transferring some additional values that are not serializable by `JSON`: `-0`, `Nan`, `Infinity`, `-Infinity`, and bigint literals.

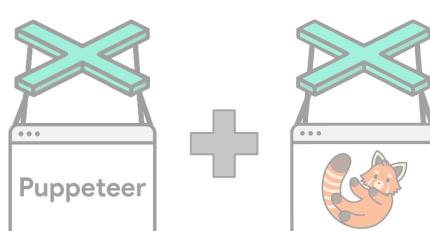
Passing arguments to `pageFunction`:

```
const result = await page.evaluate(x => {
  return Promise.resolve(8 * x);
}, 7);
console.log(result); // prints "56"
```

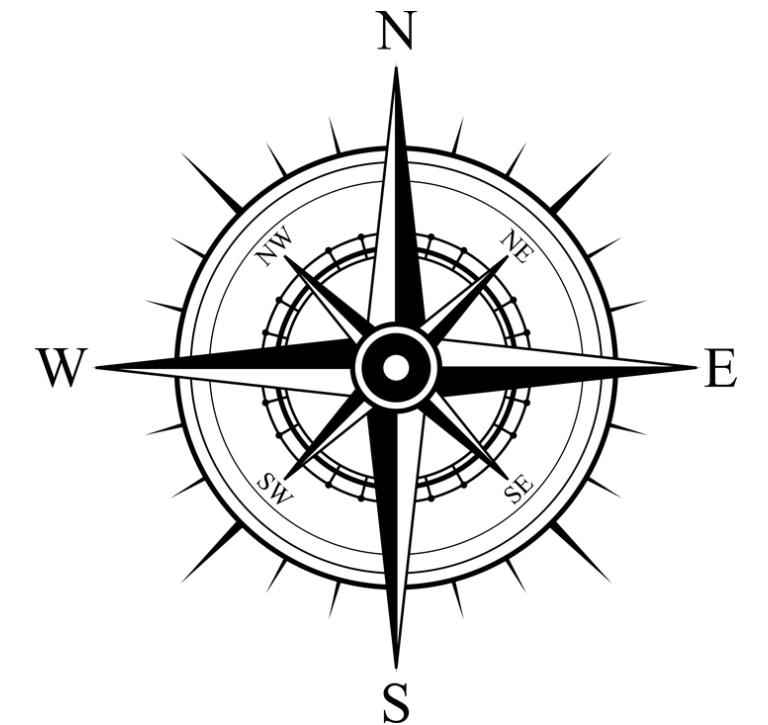
A string can also be passed in instead of a function:

```
console.log(await page.evaluate('1 + 2')); // prints "3"
const x = 10;
console.log(await page.evaluate(`1 + ${x}`)); // prints "11"
```

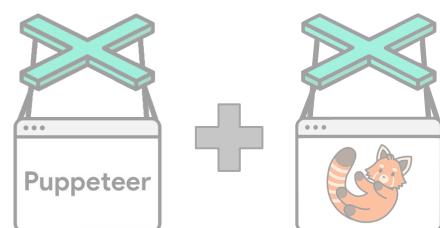
ElementHandle instances can be passed as arguments to the `page.evaluate`:



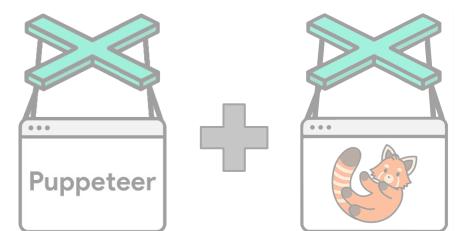
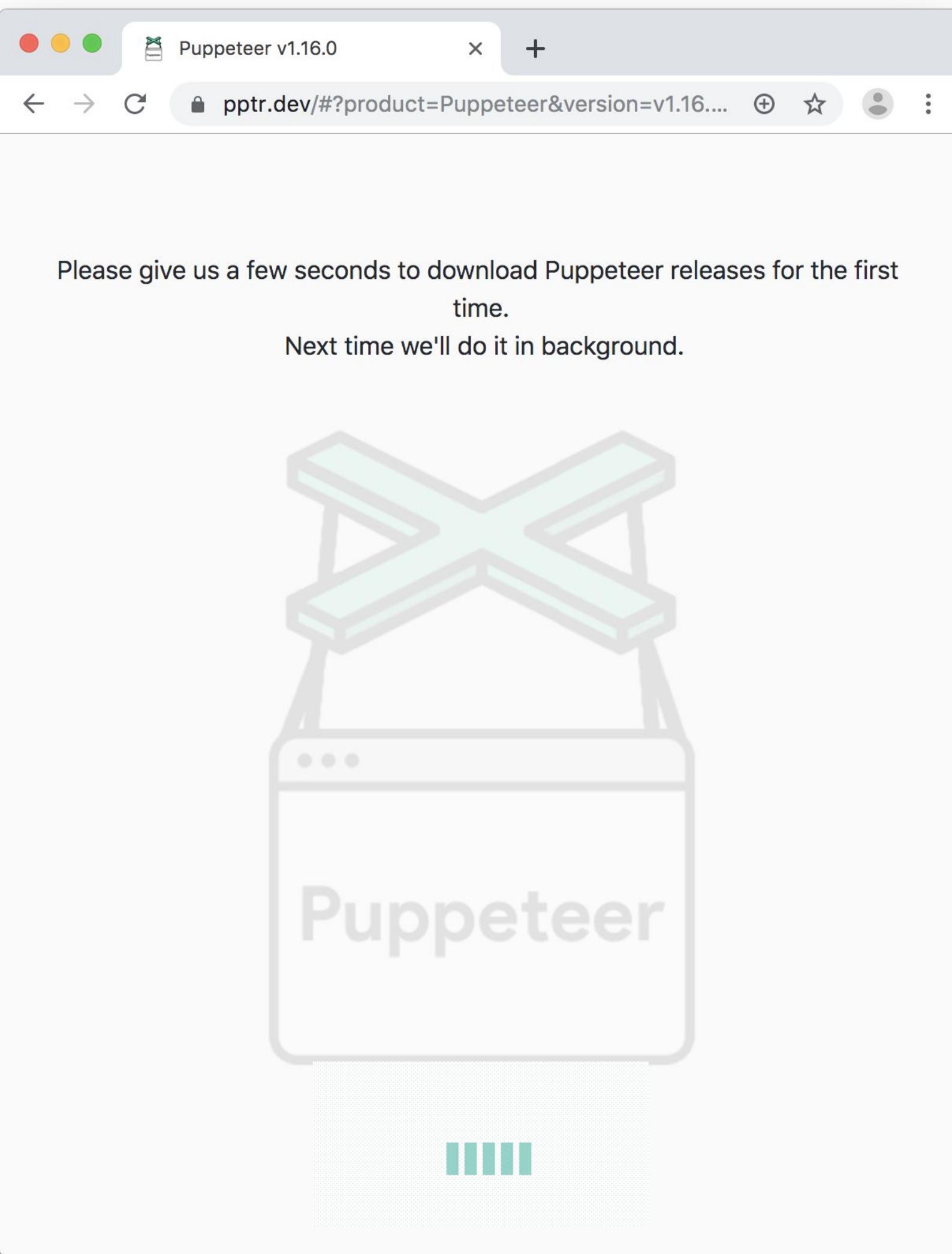
Navigation



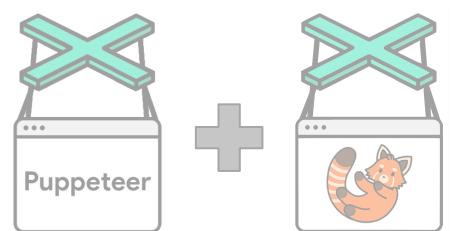
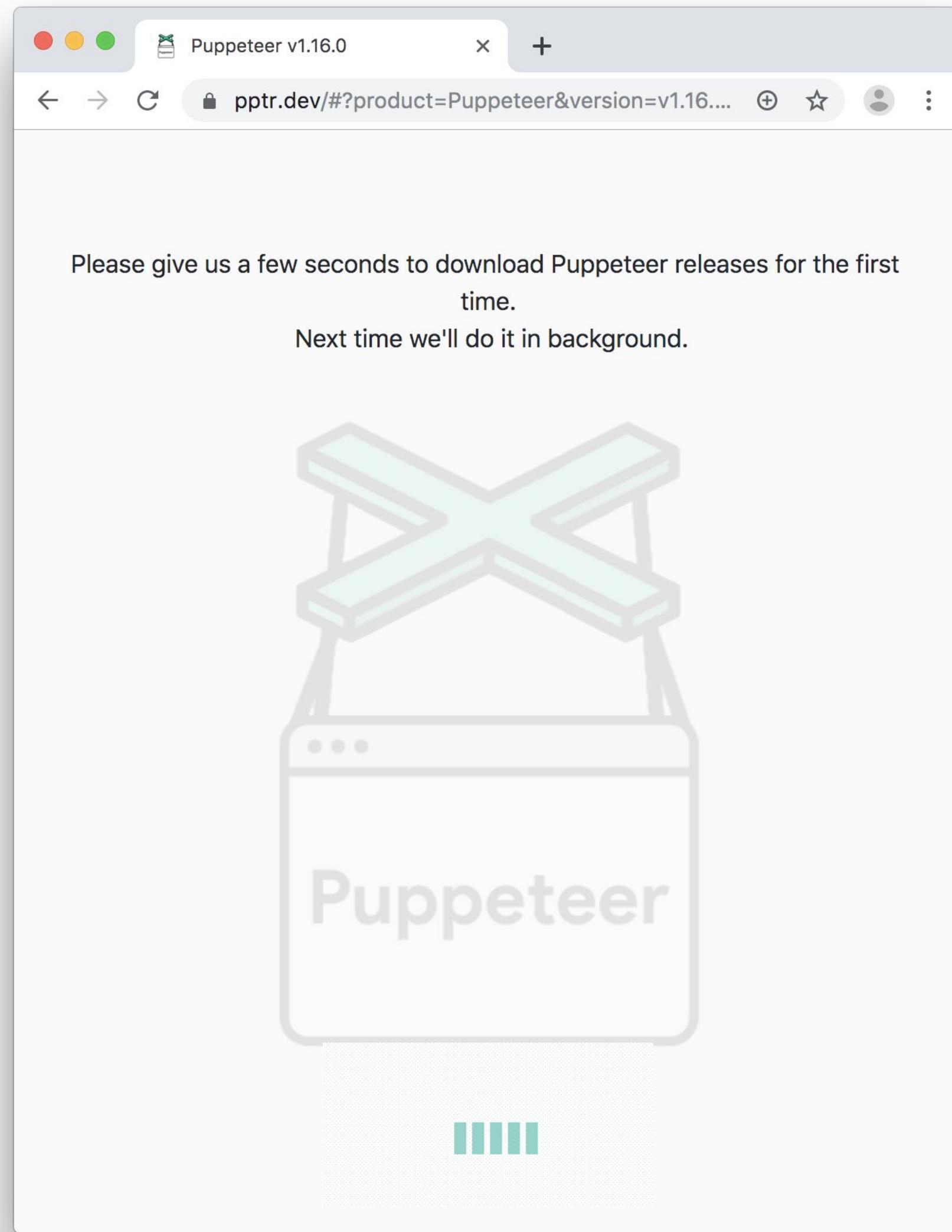
- 👉 `await page.goto('https://pptr.dev');`
- 👉 resolves when page emits “load” event



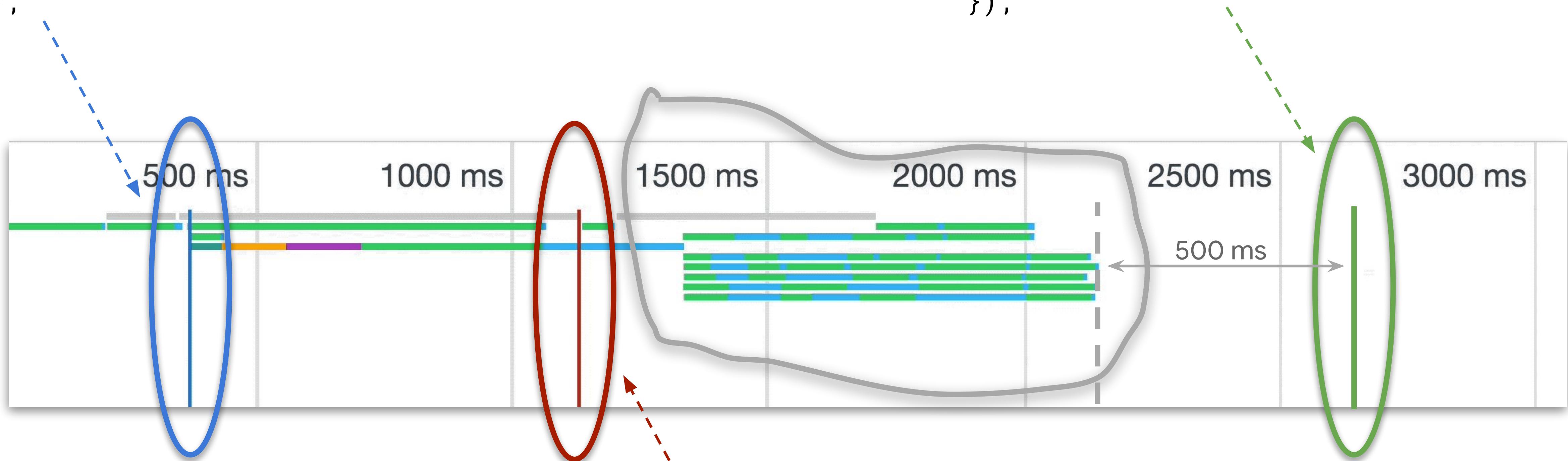
```
await page.goto('https://pptr.dev');
```



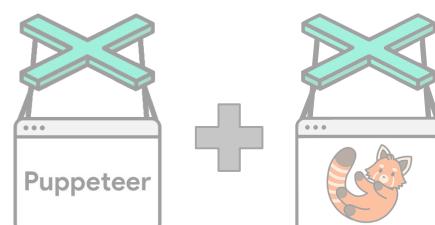
```
await page.goto('https://pptr.dev', {  
  waitUntil: '???'  
});
```



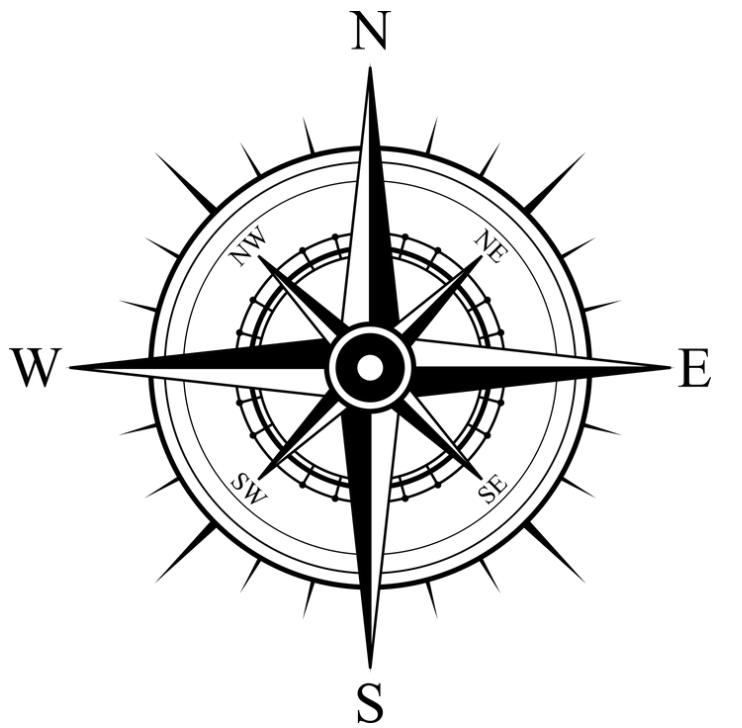
```
await page.goto('https://pptr.dev', {  
  waitUntil: 'domcontentloaded'  
});
```



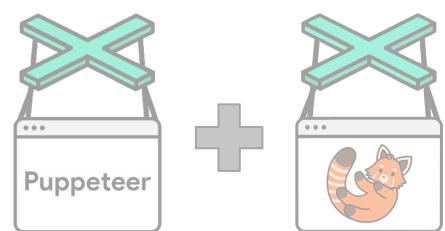
```
await page.goto('https://pptr.dev');
```



Clicking Links



- 👉 navigate to a different website
- 👉 anchor navigation
- 👉 Single-Page navigation: History API



A screenshot of a web browser window displaying the Puppeteer API documentation for the `Page` class. The browser's address bar shows the URL `pptr.dev/#?product=Puppeteer&version=v1.16.0&show=api-class-page`. The page title is "Page". A red oval highlights the browser's address bar and the top right corner of the page content.

c class: Page since v0.9.0

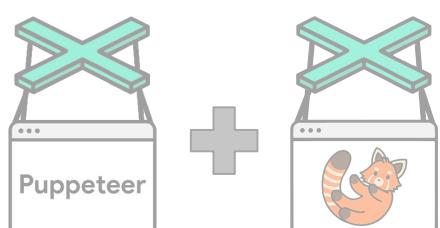
- extends: [EventEmitter](#)

Page provides methods to interact with a single tab or [extension background page](#) in Chromium. One [Browser](#) instance might have multiple [Page](#) instances.

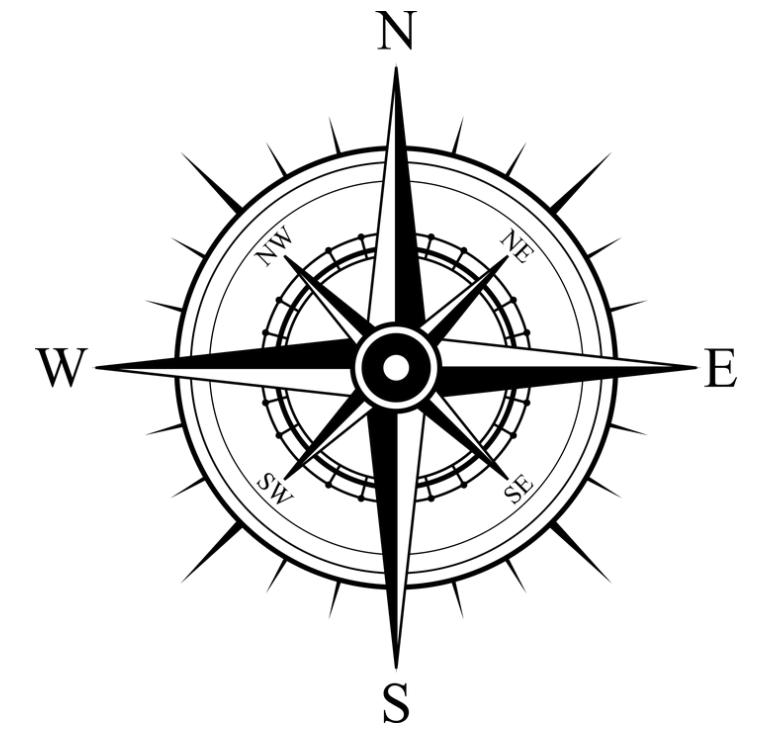
This example creates a page, navigates it to a URL, and then saves a screenshot:

```
const puppeteer = require('puppeteer');

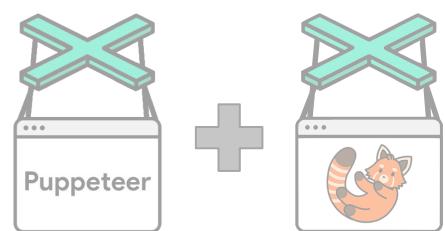
puppeteer.launch().then(async browser => {
  const page = await browser.newPage();
  await page.goto('https://example.com');
  await page.screenshot({path: 'screenshot.png'});
  await browser.close();
});
```



Navigation Tracking



👉 `page.waitForNavigation();`

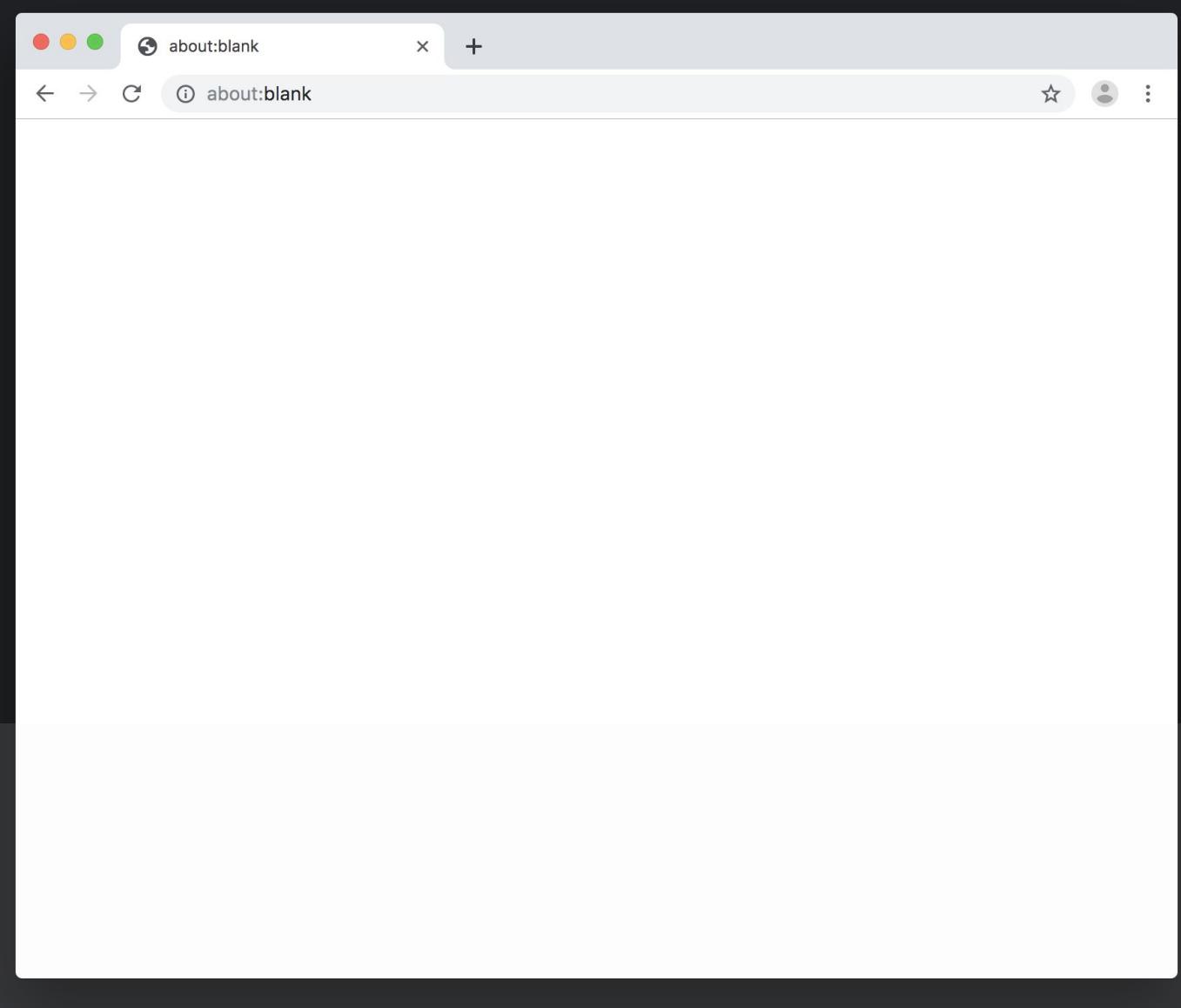


```
const puppeteer = require('puppeteer');

(async () => {
  // Automate here
})();
```

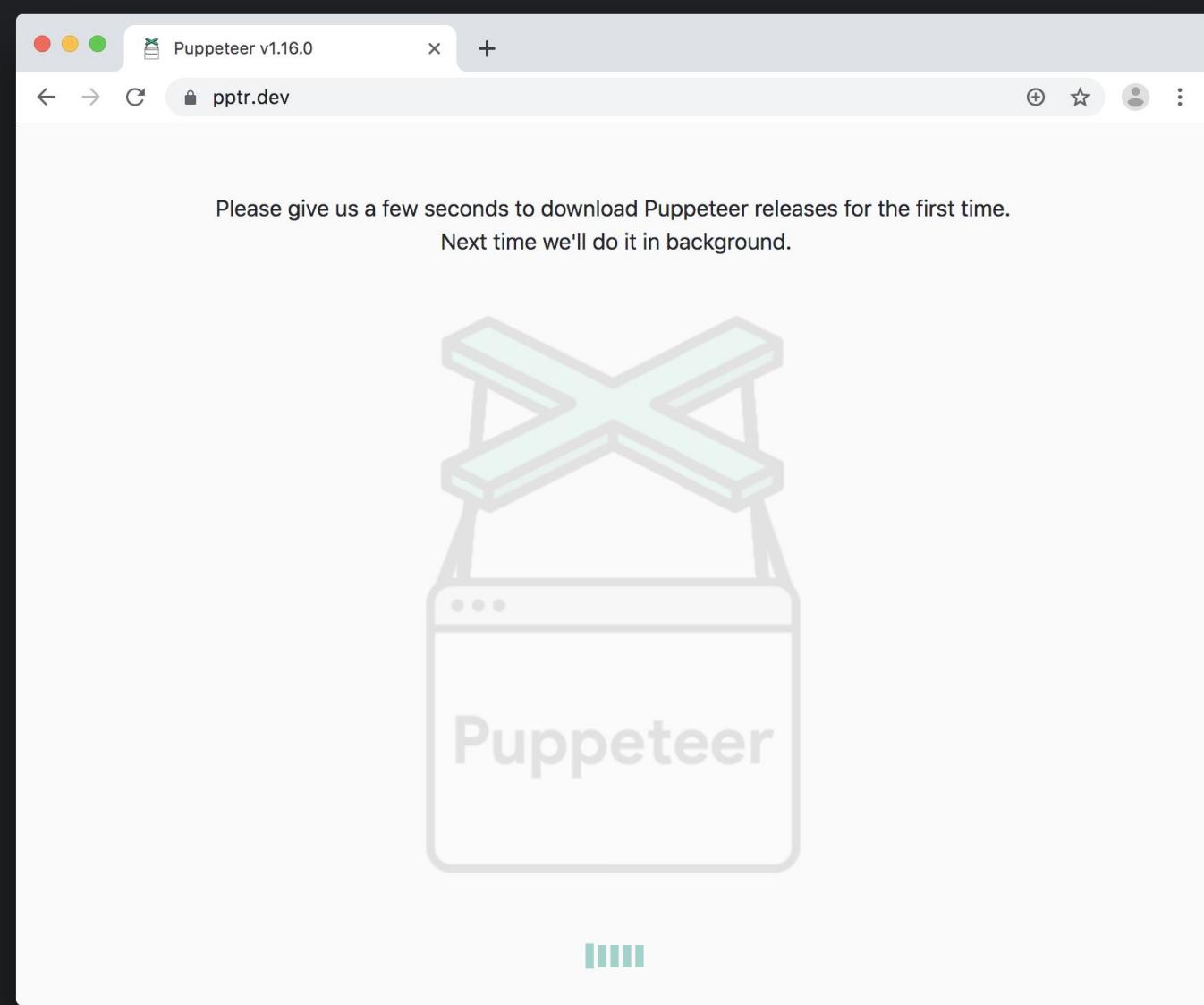
```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
})();
```



```
const puppeteer = require('puppeteer');

(async () => {

  const browser = await puppeteer.launch();
  const page = await browser.newPage();

  await page.goto('https://pptr.dev', {waitFor: 'networkidle0'});

})();
```

The screenshot shows a web browser window displaying the Puppeteer v1.16.0 documentation at pptr.dev. The page has a green header bar with the title "Puppeteer v1.16.0". On the left, there is a sidebar titled "API" with a tree view of the API structure. The main content area features a large yellow callout box with the text: "Puppeteer is a Node library which provides a high-level API to control Chrome or Chromium over the DevTools Protocol. Puppeteer runs headless by default, but can be configured to run full (non-headless) Chrome or Chromium." Below this, there's a section titled "What can I do?" with a bulleted list of examples.

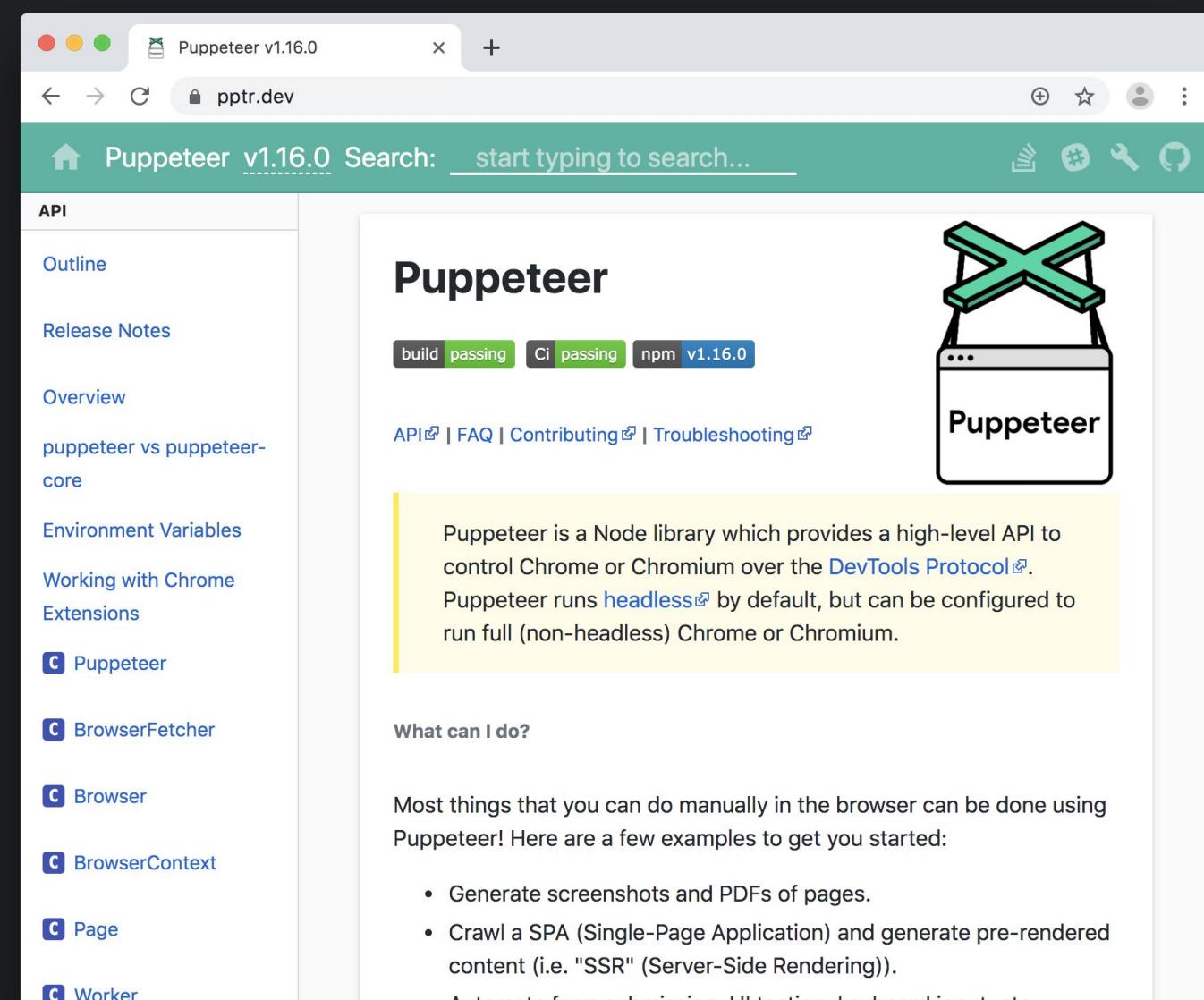
```
const puppeteer = require('puppeteer');

(async () => {

  const browser = await puppeteer.launch();
  const page = await browser.newPage();

  await page.goto('https://pptr.dev', {waitUntil: 'networkidle0'});

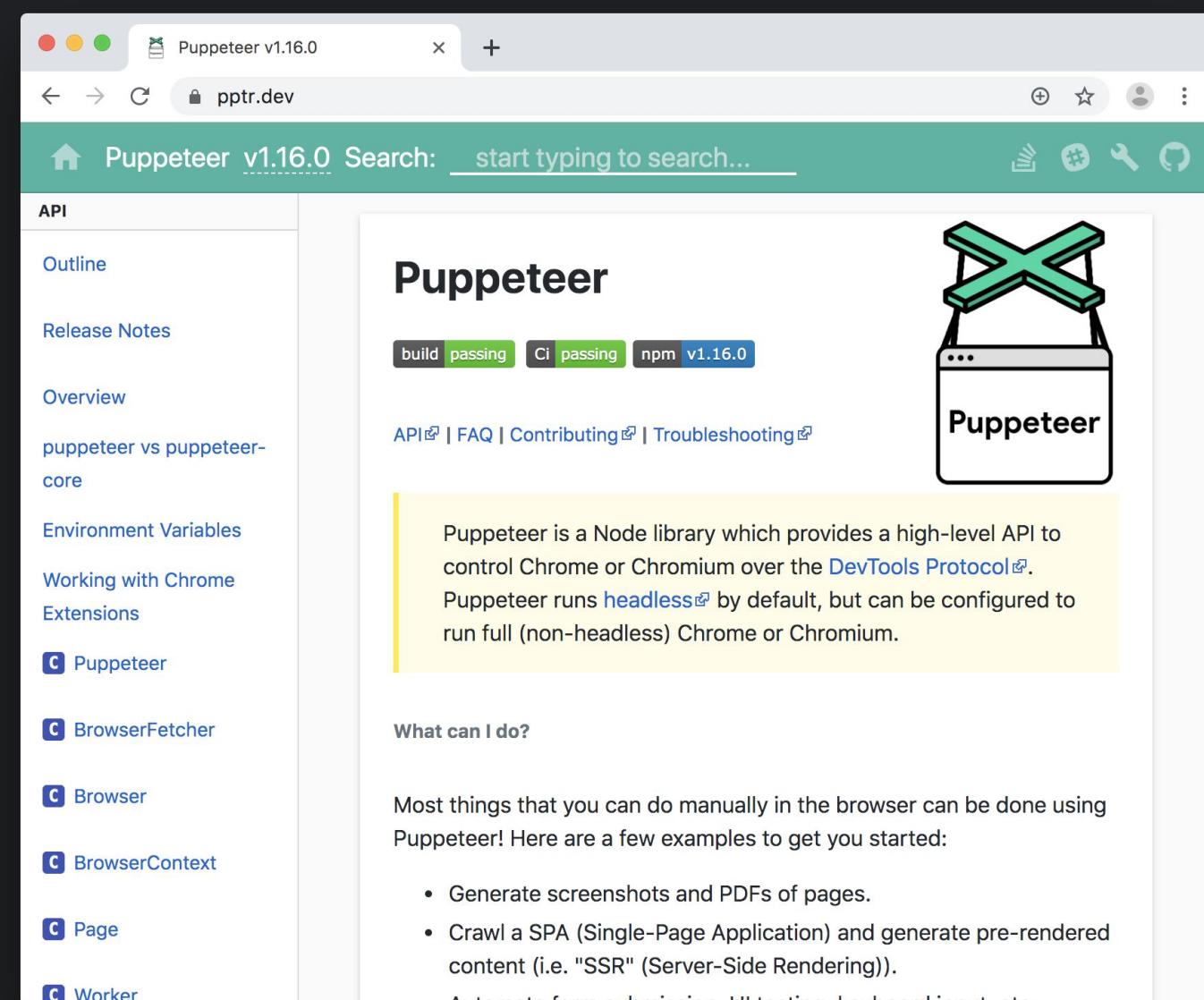
  await Promise.all([
    ...
  ])();
})();
```



The screenshot shows a web browser displaying the Puppeteer v1.16.0 documentation at pptr.dev. The page has a green header bar with the title "Puppeteer v1.16.0". On the left, there's a sidebar with a tree view of the API structure, including sections like "API", "Outline", "Release Notes", "Overview", and various classes like "Puppeteer", "BrowserFetcher", etc. The main content area features a large yellow callout box with the text: "Puppeteer is a Node library which provides a high-level API to control Chrome or Chromium over the DevTools Protocol. Puppeteer runs headless by default, but can be configured to run full (non-headless) Chrome or Chromium." Below this, there's a section titled "What can I do?" with a bulleted list of examples: "Generate screenshots and PDFs of pages.", "Crawl a SPA (Single-Page Application) and generate pre-rendered content (i.e. "SSR" (Server-Side Rendering)).", and "Automate form submission, UI testing, keyboard input, etc.". A small icon of a computer monitor with a green X on it is located in the top right corner of the main content area.

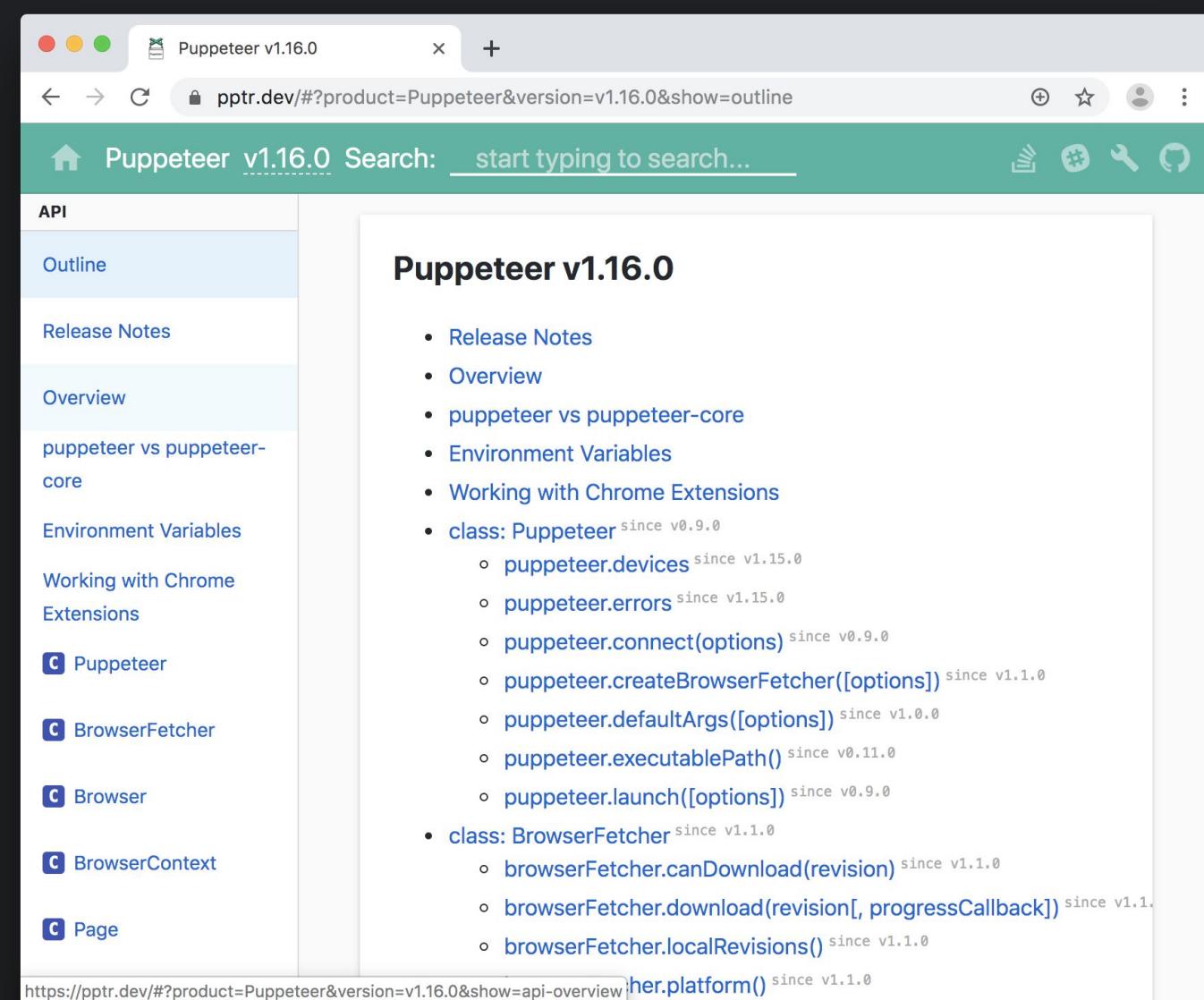
```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev', {waitUntil: 'networkidle0'});
  await Promise.all([
    page.click('sidebar-component a'), // Documentation Outline
  ]);
})();
```

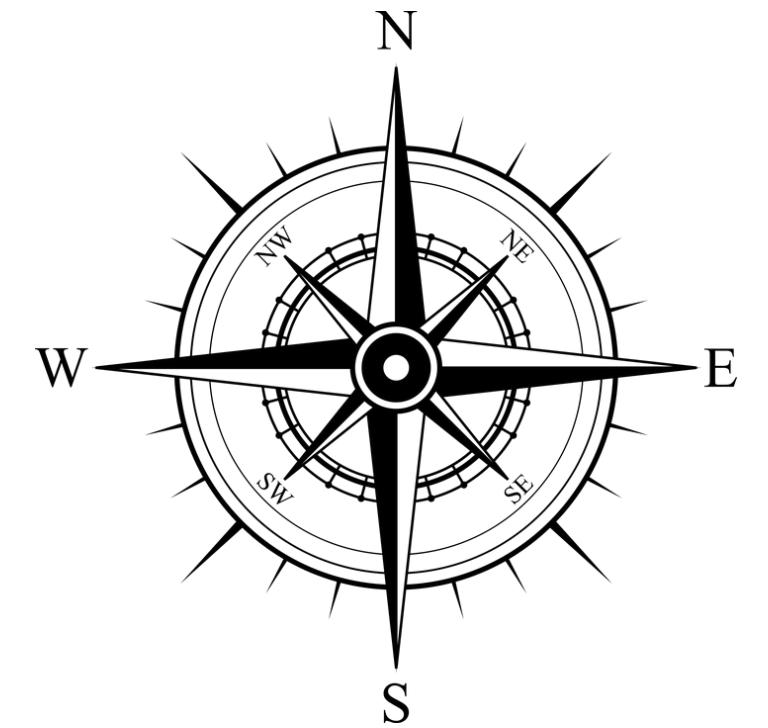


```
const puppeteer = require('puppeteer');

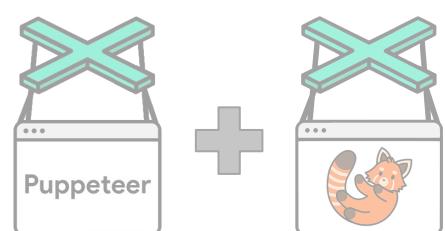
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev', {waitUntil: 'networkidle0'});
  await Promise.all([
    page.click('sidebar-component a'), // Documentation Outline
    page.waitForNavigation(),
  ]);
})();
```



Navigation APIs



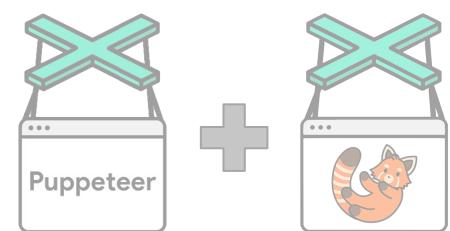
- 👉 `page.goto();`
- 👉 `page.waitForNavigation();`
- 👉 `page.goBack();`
- 👉 `page.goForward();`



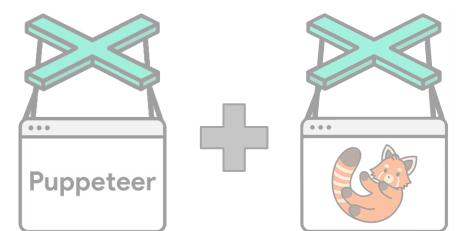
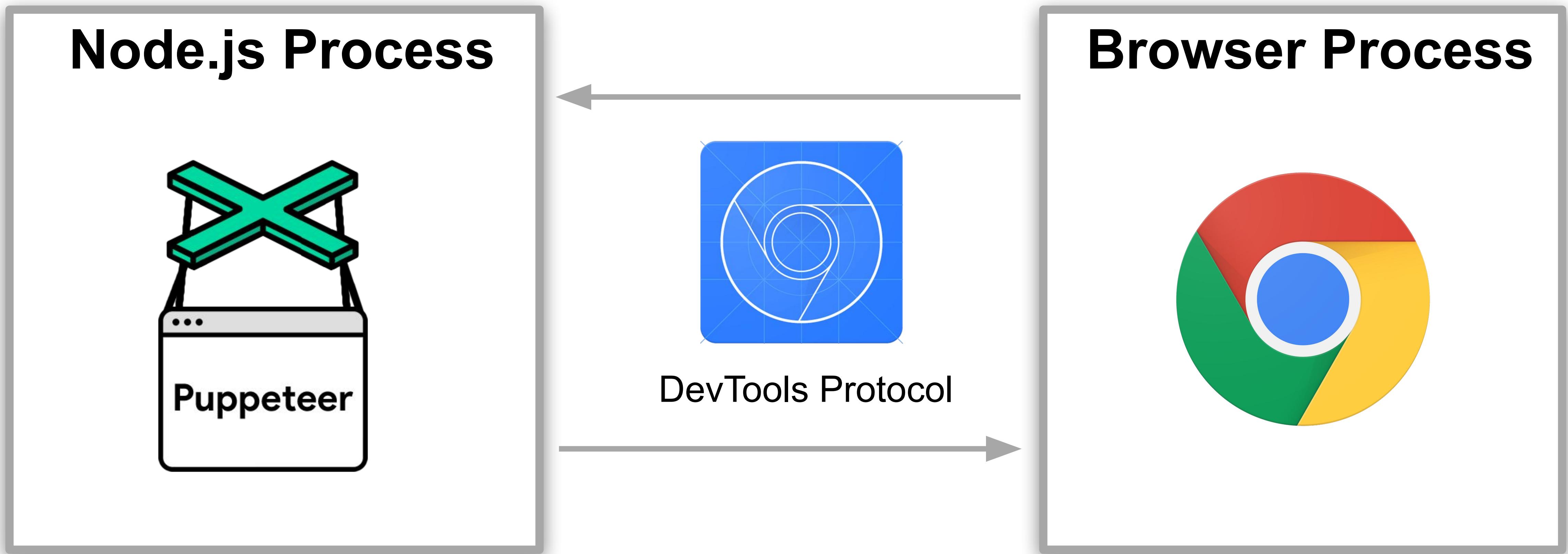
Evaluation



- 👉 evaluate JavaScript in website contexts
- 👉 use Web APIs to drive automation



ASYNC!



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const linkCount = await page.evaluate(() => {
    const links = document.querySelectorAll('a');
    return links.length;
  });
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const linkCount = await page.evaluate(() => {
    const links = document.querySelectorAll('a');
    return links.length;
  });
})();
```



```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const linkCount = await page.evaluate(() => {
    });
})();
});
```



```
const puppeteer = require('puppeteer');

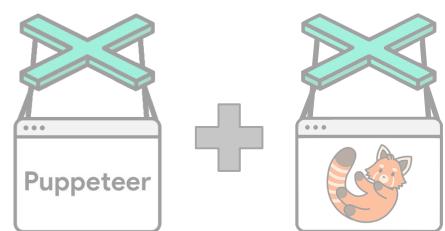
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://joel.tools/merch');
  const linkCount = await page.evaluate(() => {
    browser; // undefined!
    page; // undefined!
  });
})();
```





Test Isolation

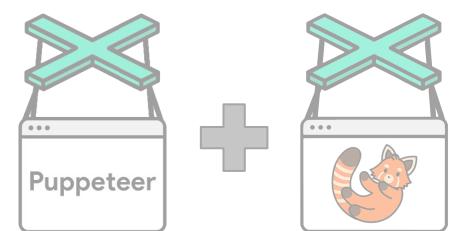
- 👉 Node.js **does not leak** into browser
- 👉 Website **doesn't know** it's been tested
- 👉 Websites **can't ship test-specific** code



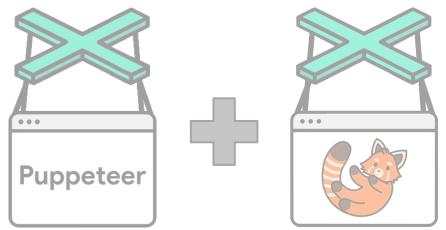
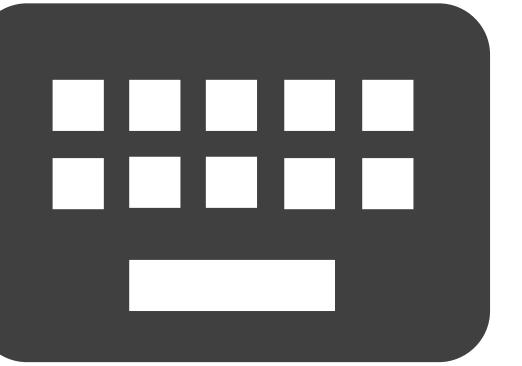


Takeaway

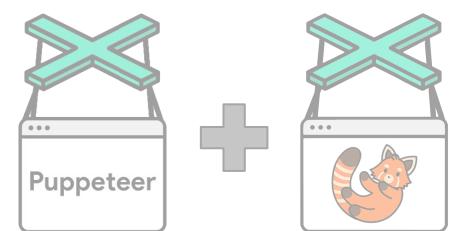
👉 Testing with Puppeteer is **foolproof**



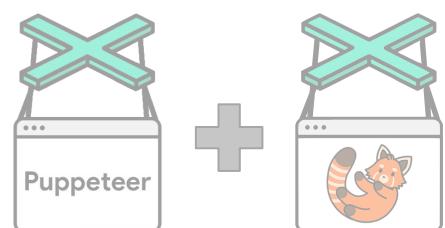
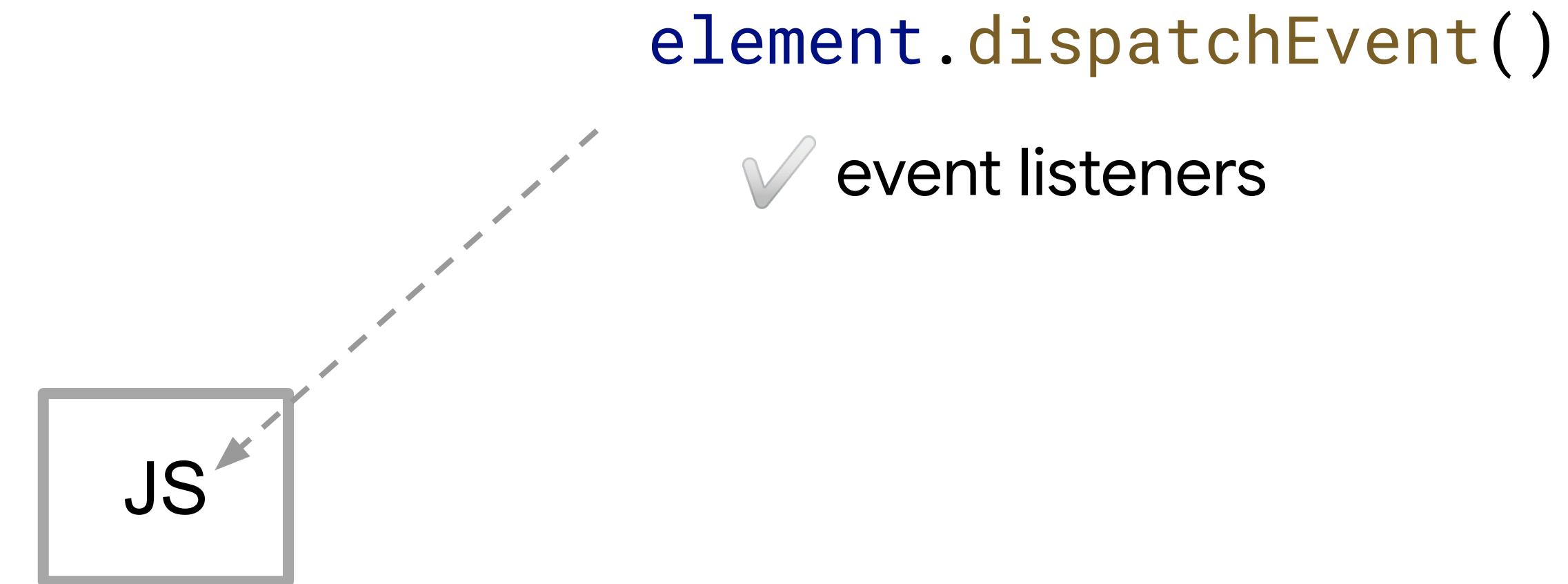
Input Injection

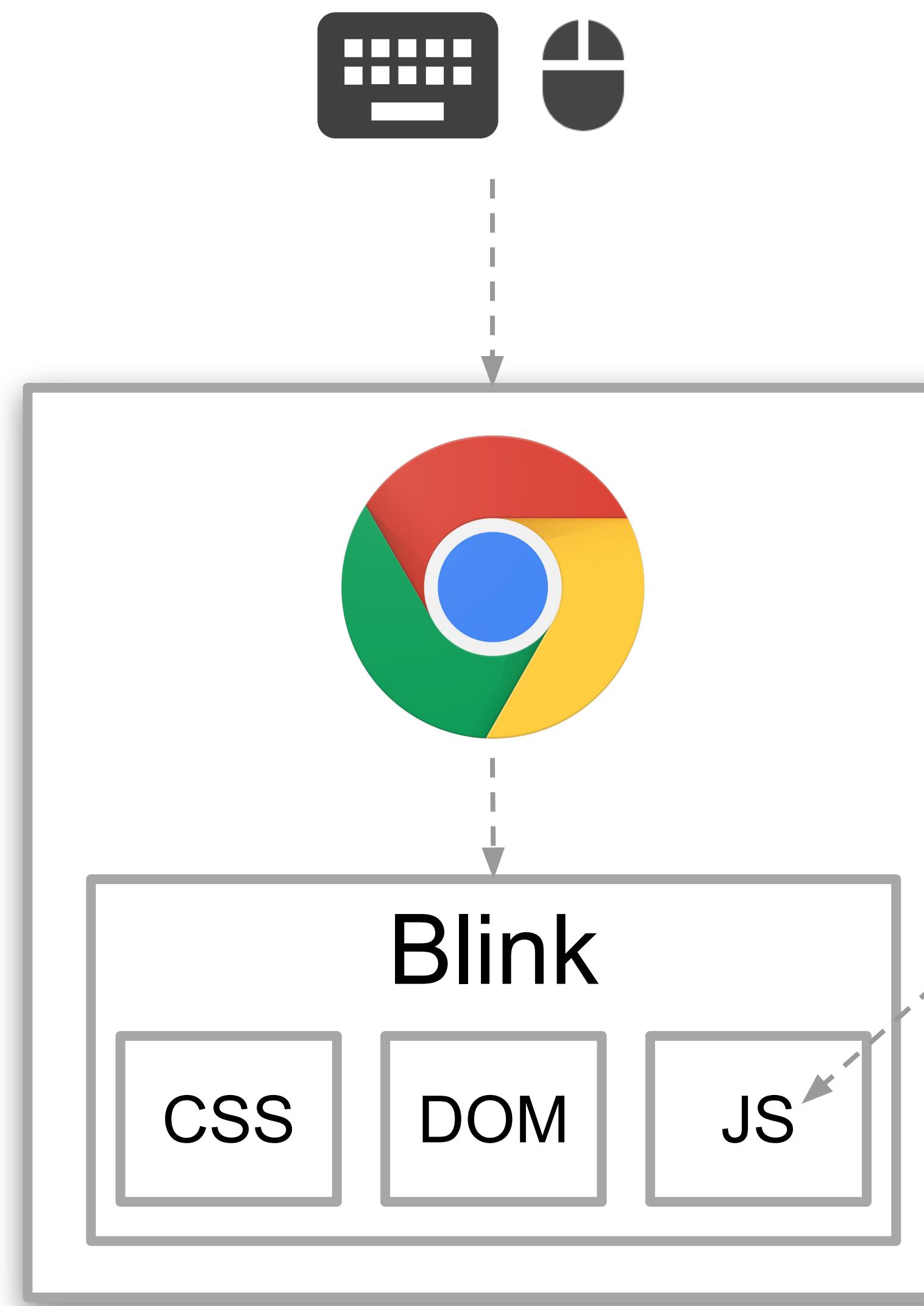


```
const mouseEvent = new MouseEvent('mousedown', {  
  clientX: 50,  
  clientY: 50,  
});  
document.body.dispatchEvent(mouseEvent);
```



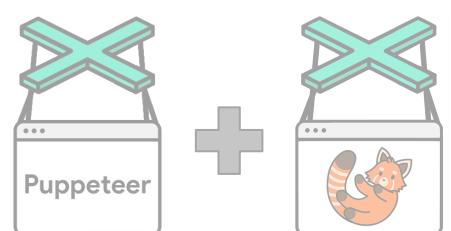
```
const mouseEvent = new MouseEvent('mousedown', {  
  clientX: 50,  
  clientY: 50,  
});  
document.body.dispatchEvent(mouseEvent);
```

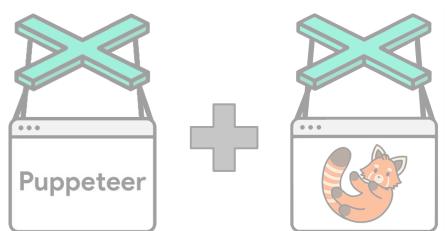
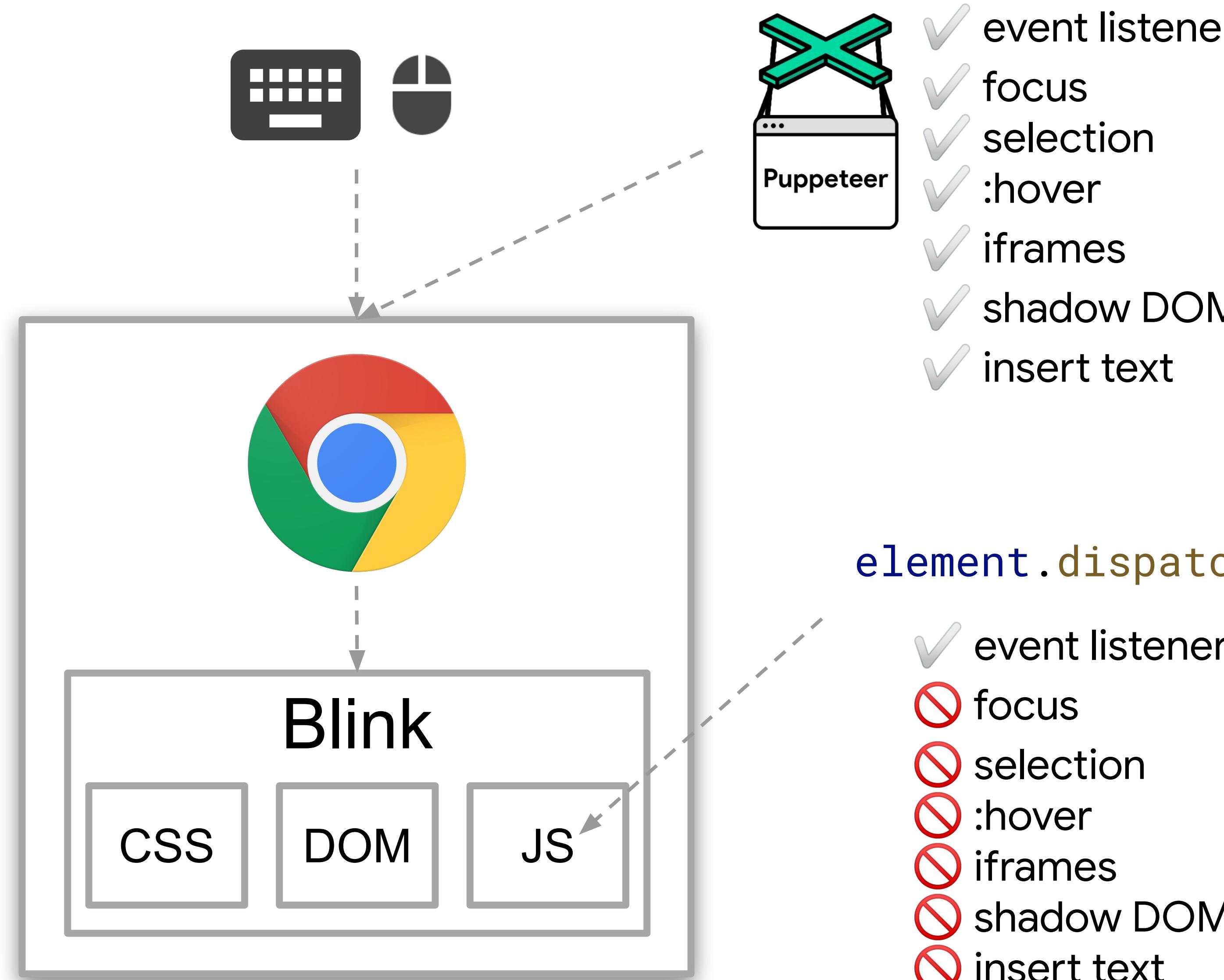




`element.dispatchEvent()`

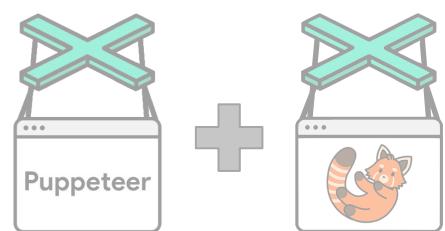
- ✓ event listeners
- 🚫 focus
- 🚫 selection
- 🚫 :hover
- 🚫 iframes
- 🚫 shadow DOM
- 🚫 insert text





Input Injection APIs

- 👉 `page.keyboard.*` 
- 👉 `page.mouse.*` 
- 👉 `page.type(selector, text);`
- 👉 `page.click(selector);`



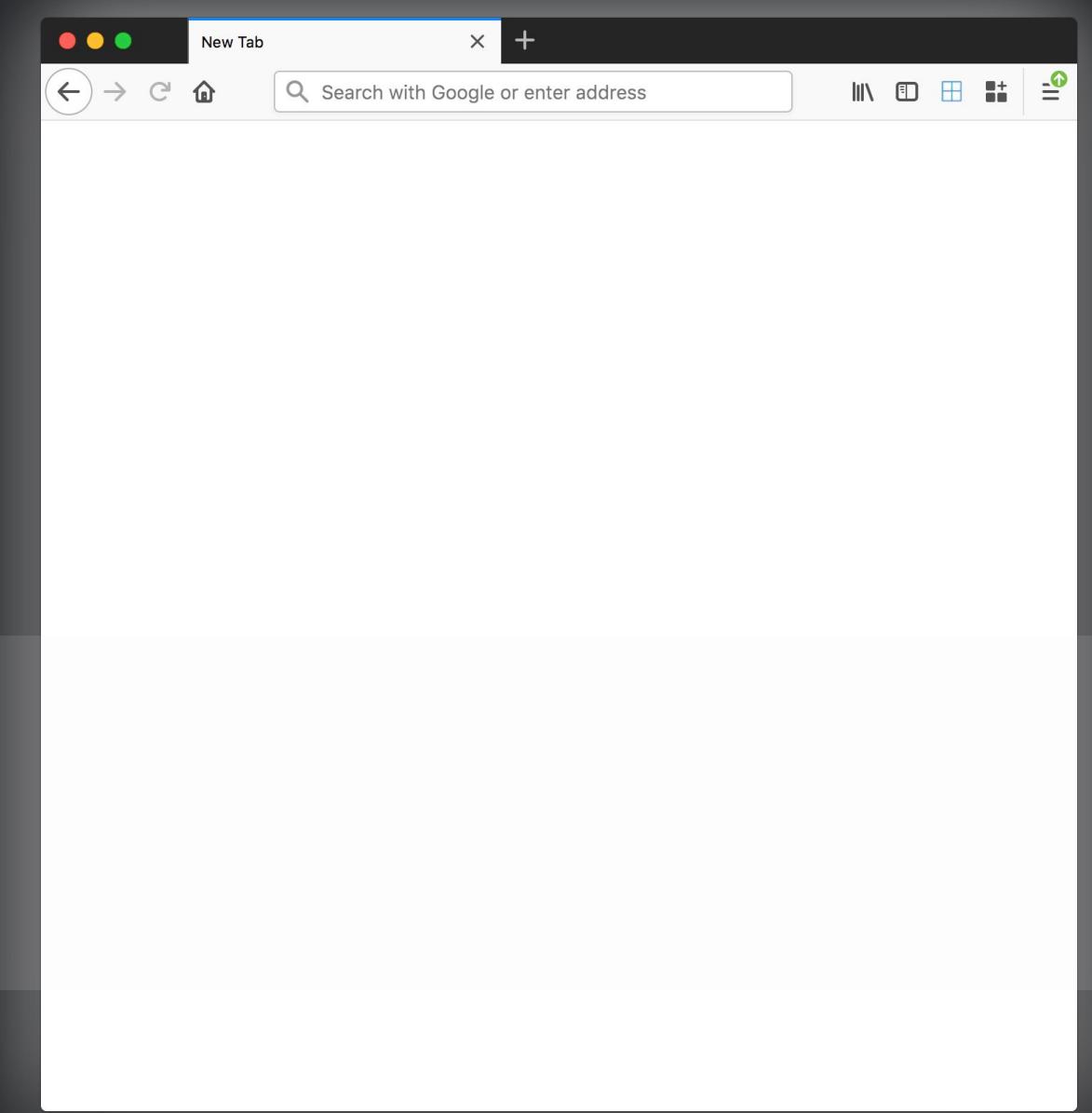
```
const puppeteer = require('puppeteer-firebase');

(async () => {
  // Automate here
})();
```

```
const puppeteer = require('puppeteer-firebase');

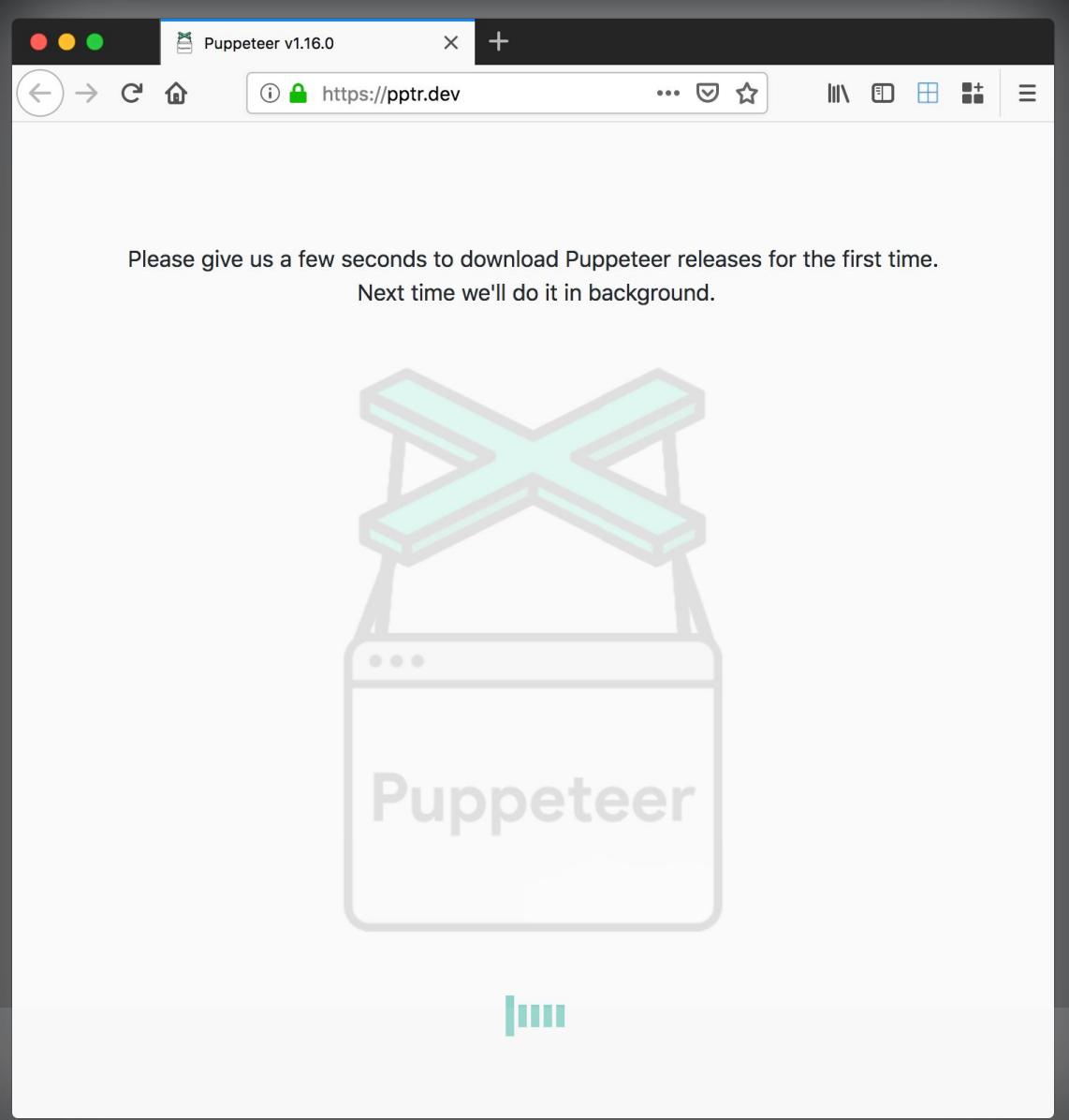
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();

})();
```



```
const puppeteer = require('puppeteer-firebase');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
})();
```



```
const puppeteer = require('puppeteer-firebase');

(async () => {

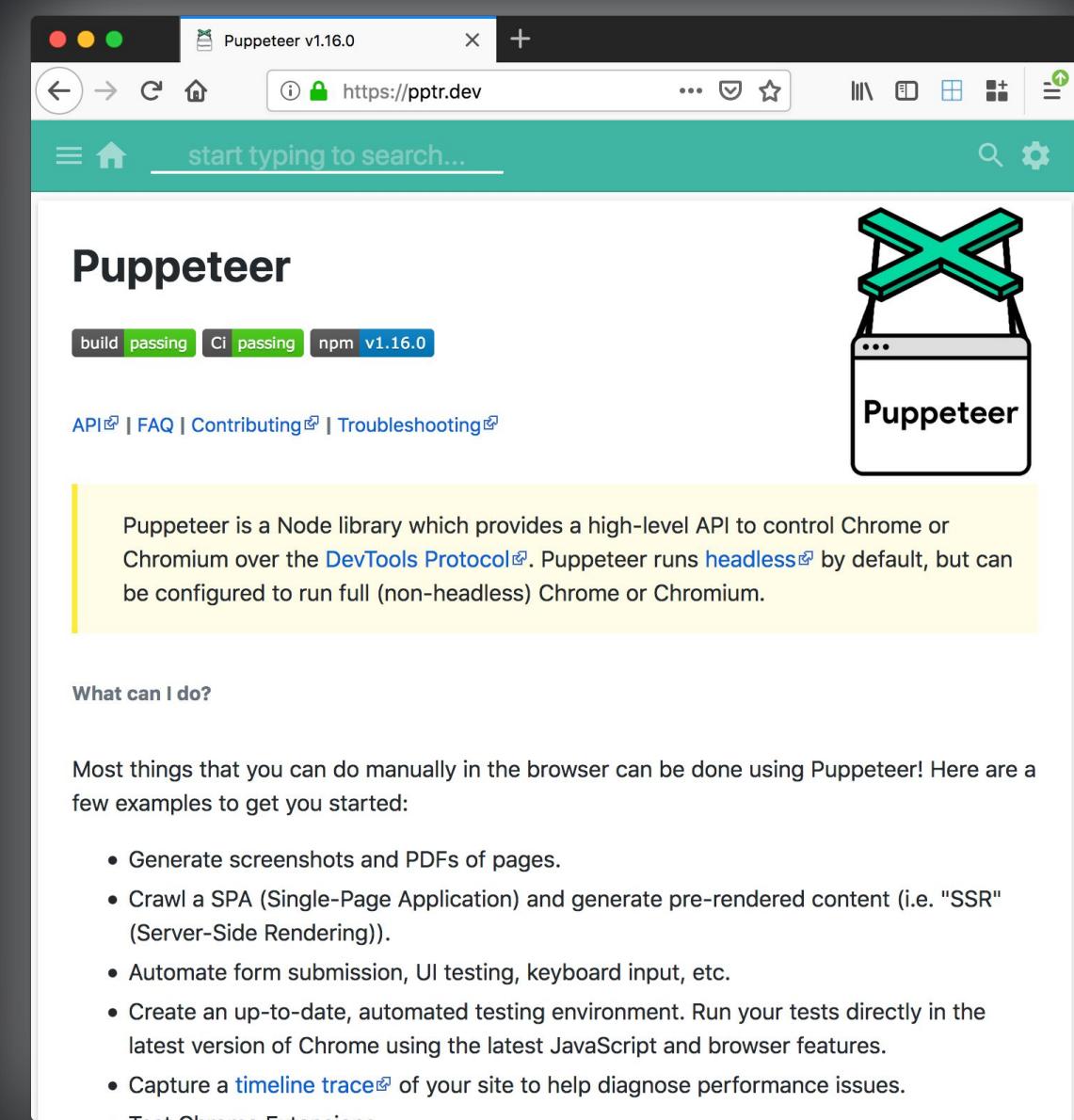
  const browser = await puppeteer.launch();

  const page = await browser.newPage();

  await page.goto('https://pptr.dev');

  const input = await page.waitForSelector('[type=search]');

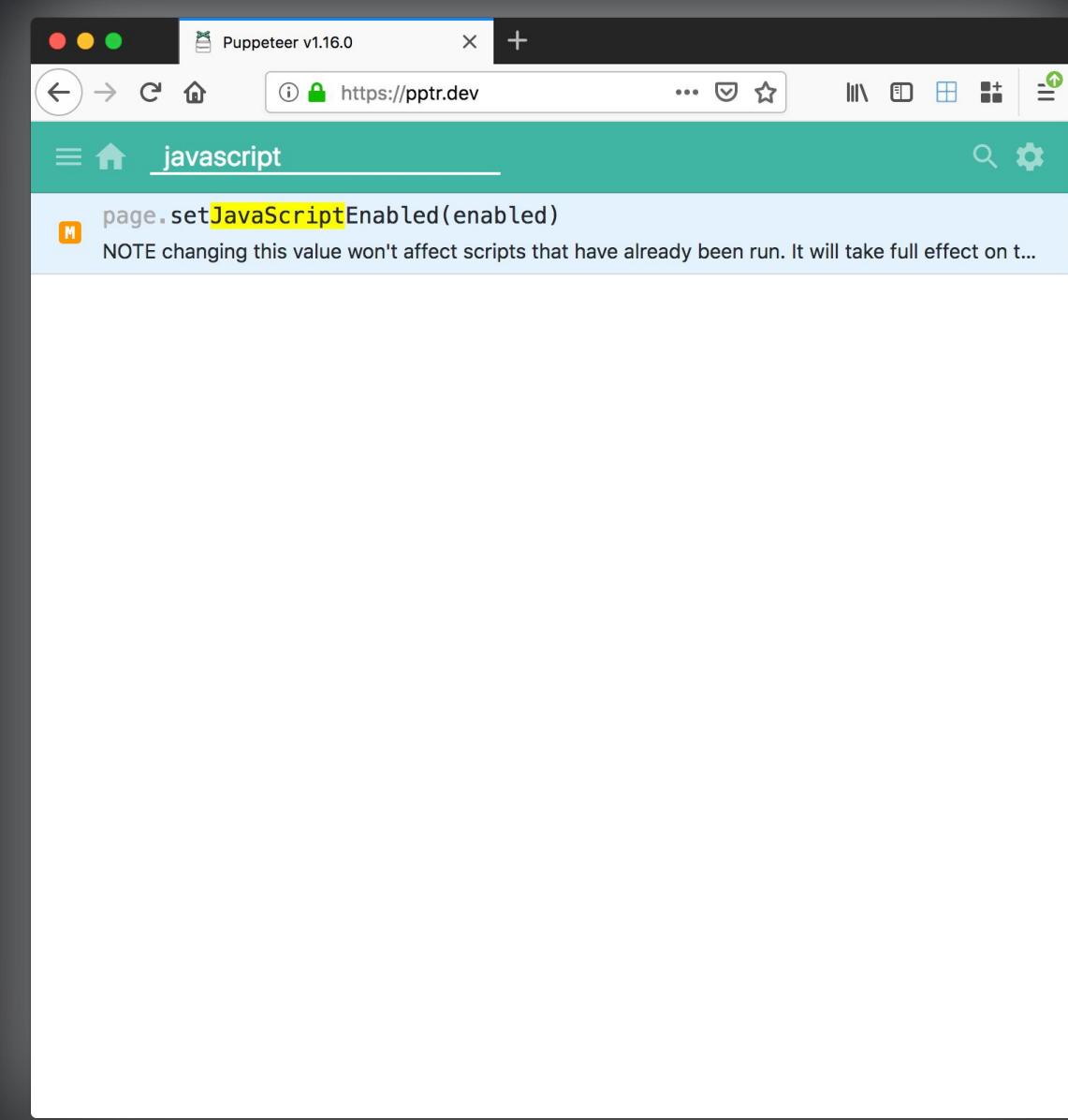
})();
```



```
const puppeteer = require('puppeteer-firebase');

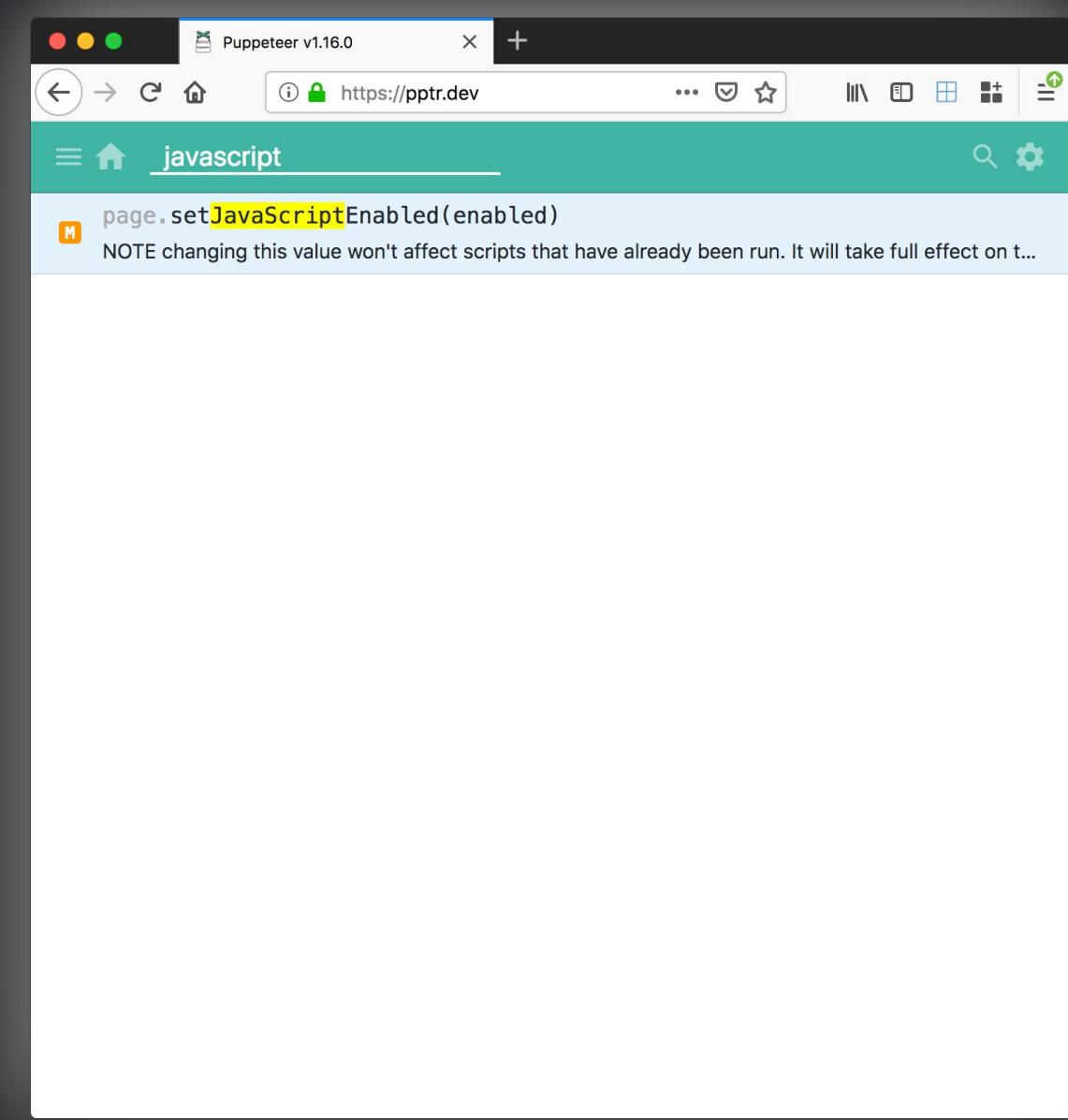
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
  const input = await page.waitForSelector('[type=search]');
  await input.type('javascript');

})();
```



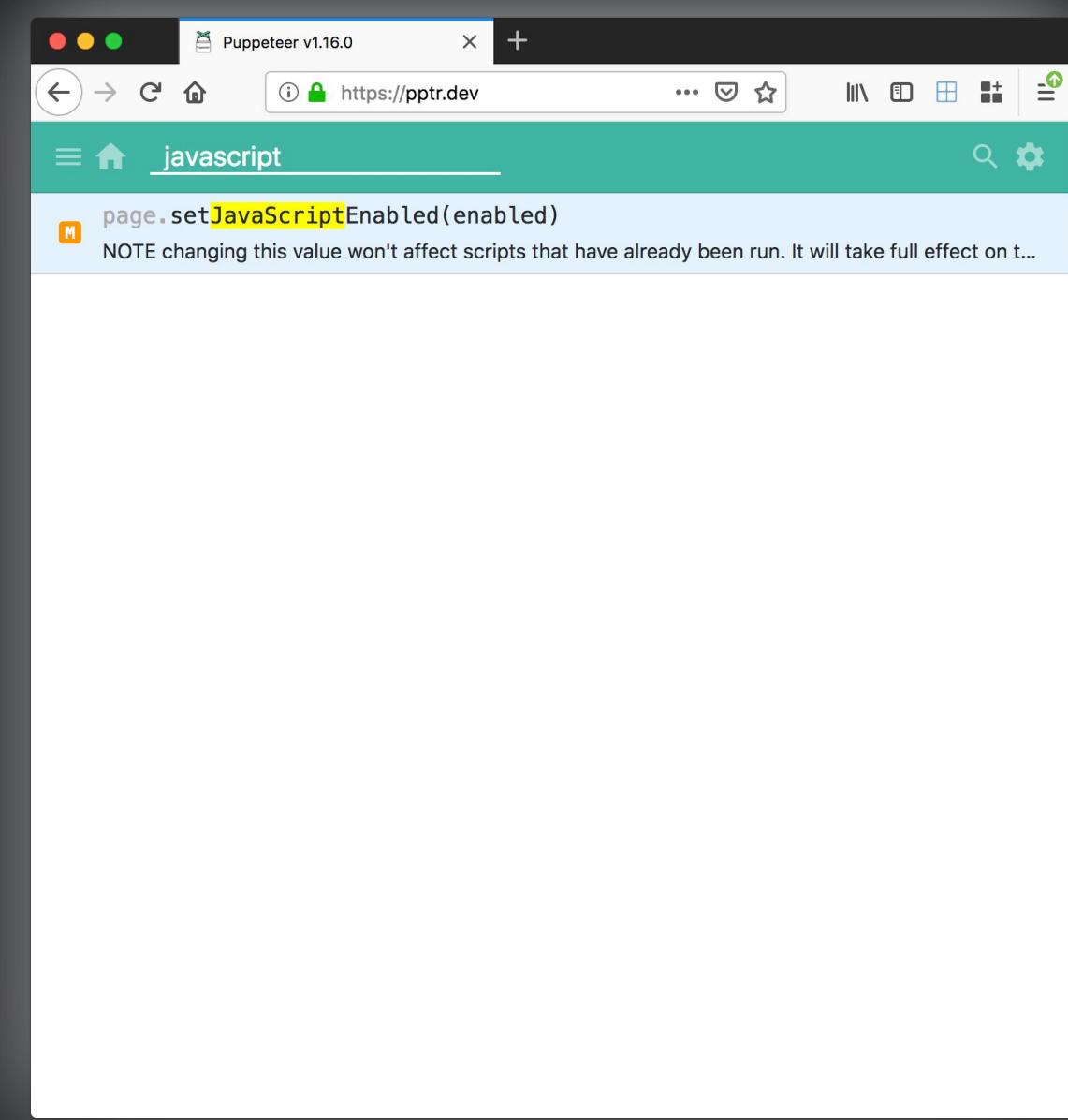
```
const puppeteer = require('puppeteer-firebase');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
  const input = await page.waitForSelector('[type=search]');
  await input.type('javascript');
  await Promise.all([
    () => {
      const result = await page.evaluate(() => {
        const script = document.createElement('script');
        script.textContent = `console.log('Hello from JS!');`;
        document.head.appendChild(script);
      });
      return result;
    }
  ]);
})();
```



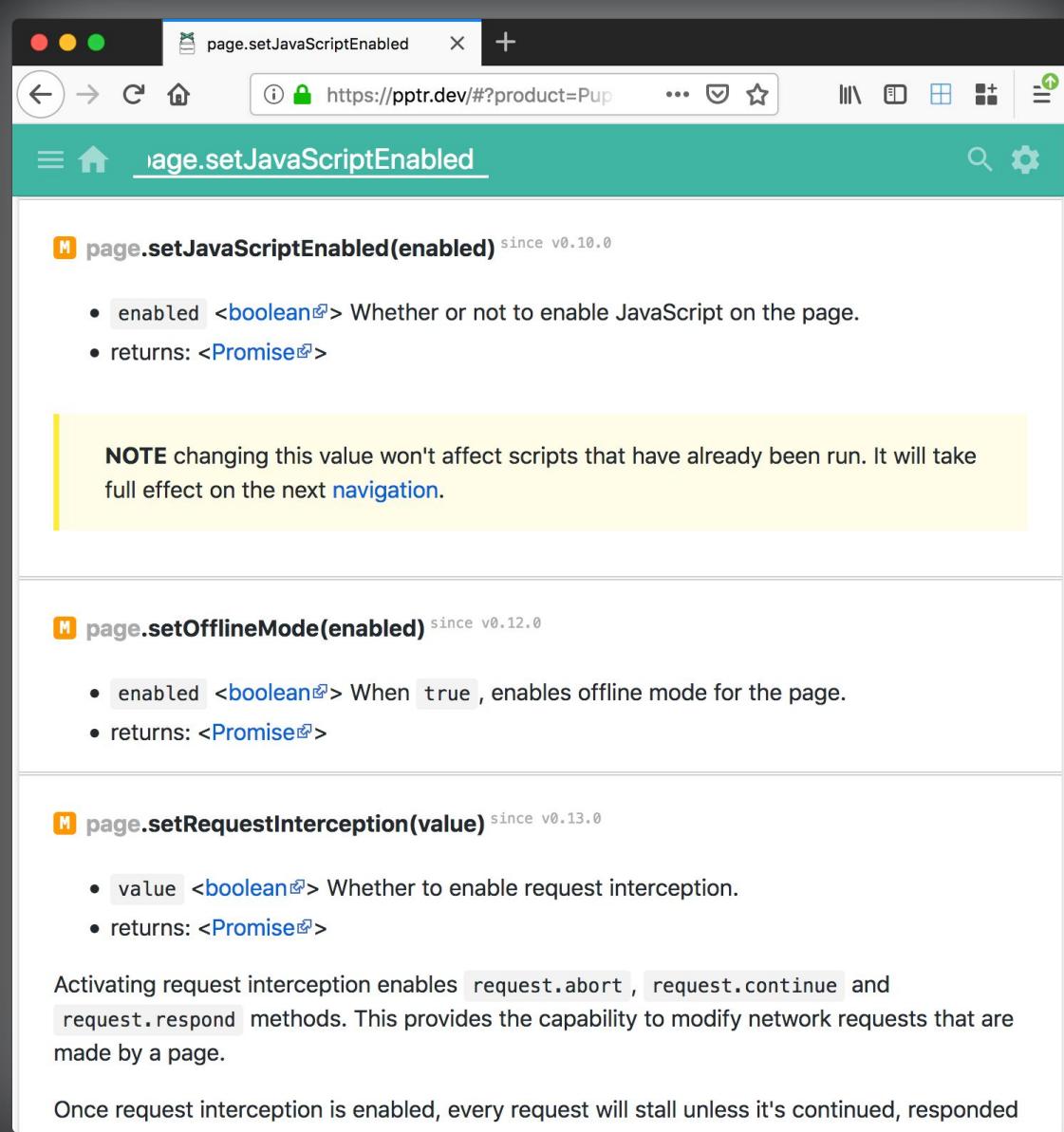
```
const puppeteer = require('puppeteer-firebase');

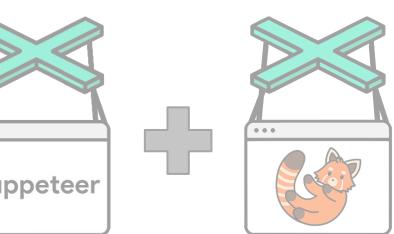
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
  const input = await page.waitForSelector('[type=search]');
  await input.type('javascript');
  await Promise.all([
    page.keyboard.press('Enter'),
  ]);
})();
```



```
const puppeteer = require('puppeteer-firebase');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://pptr.dev');
  const input = await page.waitForSelector('[type=search]');
  await input.type('javascript');
  await Promise.all([
    page.keyboard.press('Enter'),
    page.waitForNavigation(),
  ]);
})();
```





Puppeteer v1.16.0

https://pptr.dev

Puppeteer

Linux Build Status Windows Build Status Build Status NPM puppeteer package

[API](#) | [FAQ](#) | [Contributing](#) | [Troubleshooting](#)

Puppeteer is a Node library which provides a high-level API to control Chrome or Chromium over the [DevTools Protocol](#). Puppeteer runs `headless` by default, but can be configured to run full (non-headless) Chrome or Chromium.

What can I do?

Most things that you can do manually in the browser can be done using Puppeteer! Here are a few examples to get you started:

- Generate screenshots and PDFs of pages

Summary

- 👉 All navigations are supported
- 👉 Tests are isolated from web content
- 👉 Real input



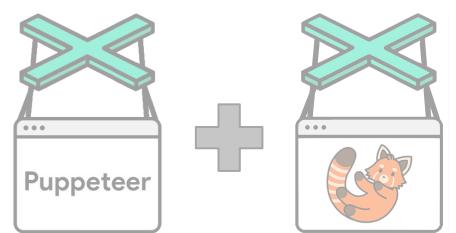
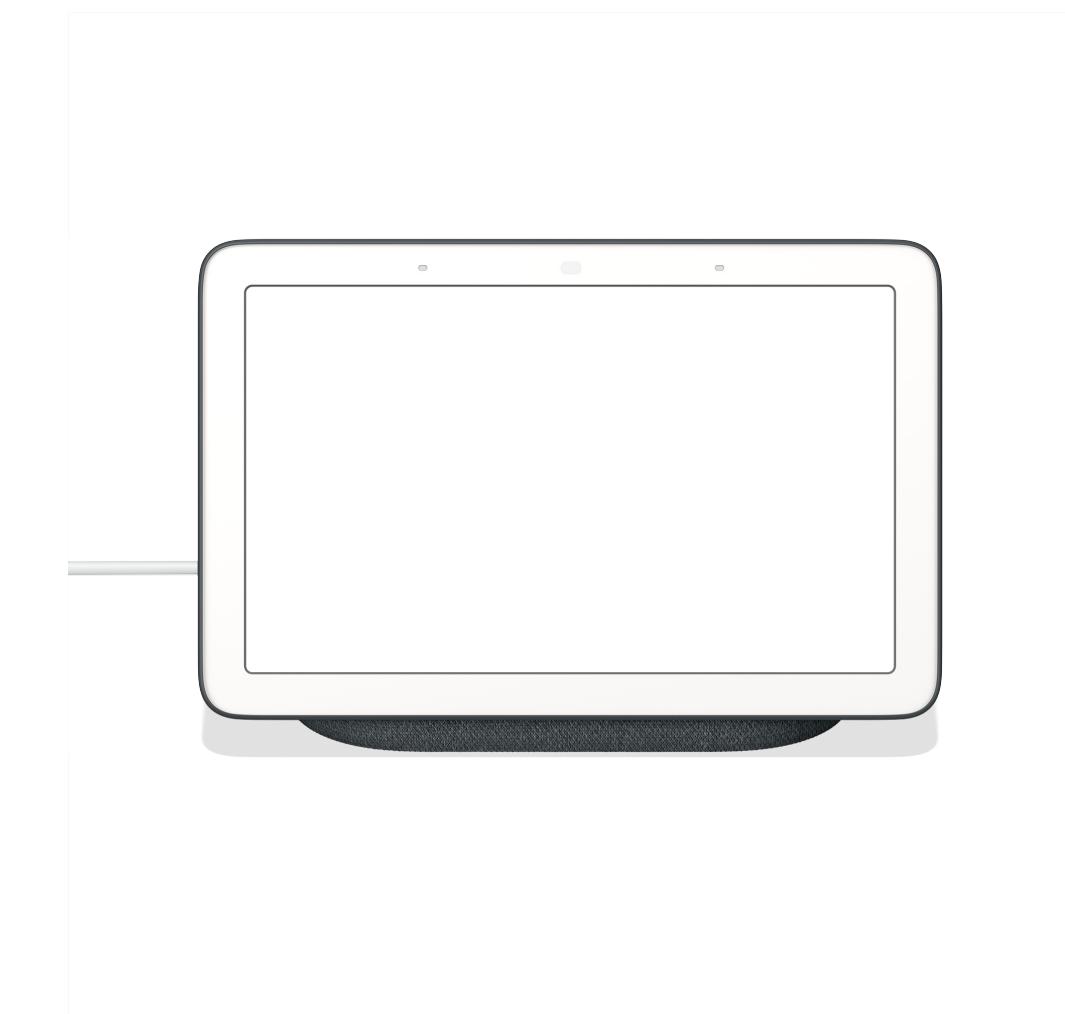
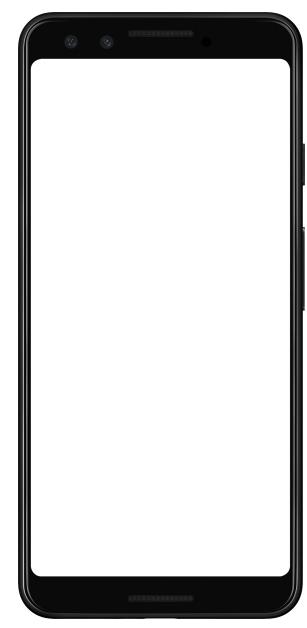
Testing Modern Web

Modern Web

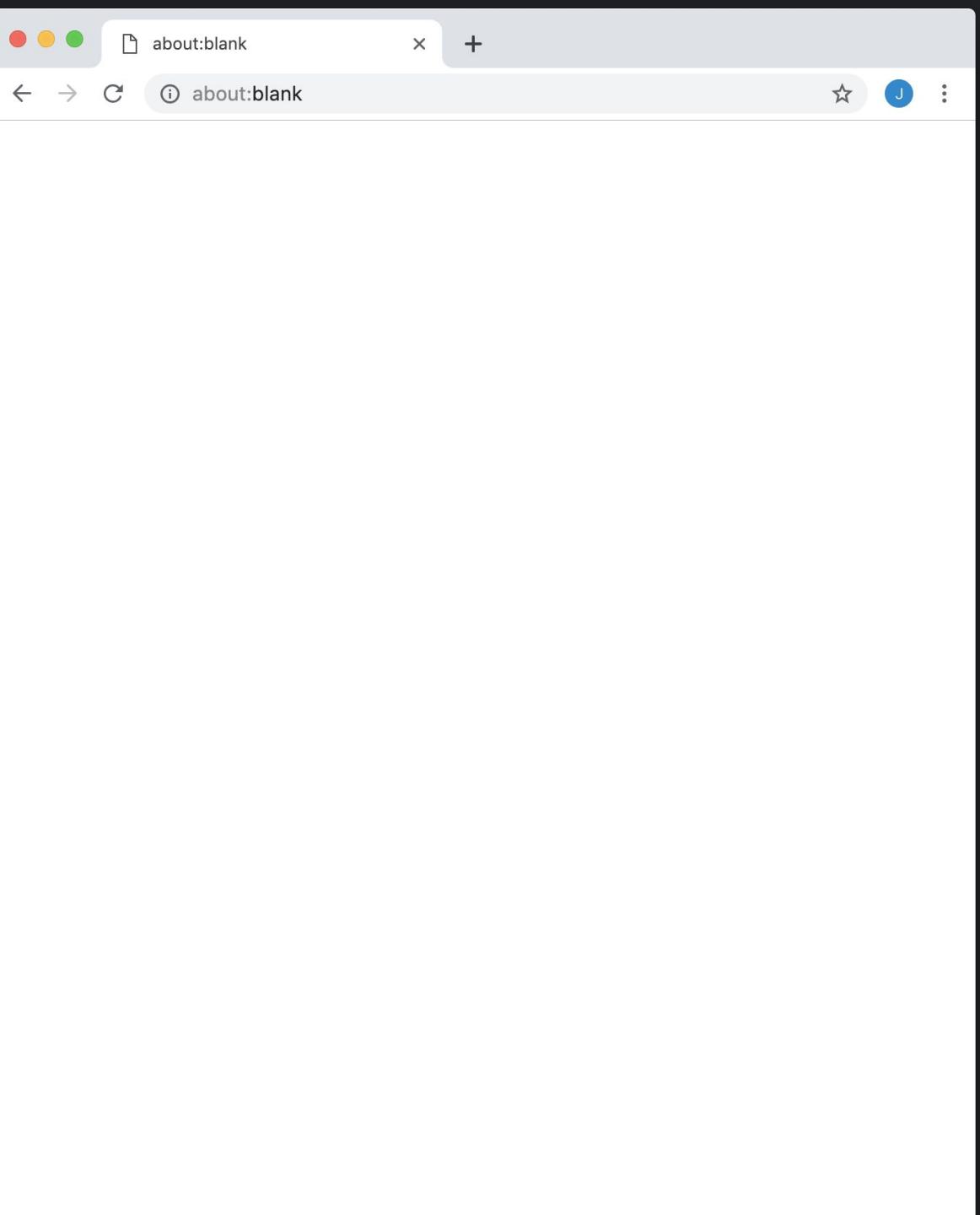
- 👉 WebApps
- 👉 Performance
- 👉 Accessibility



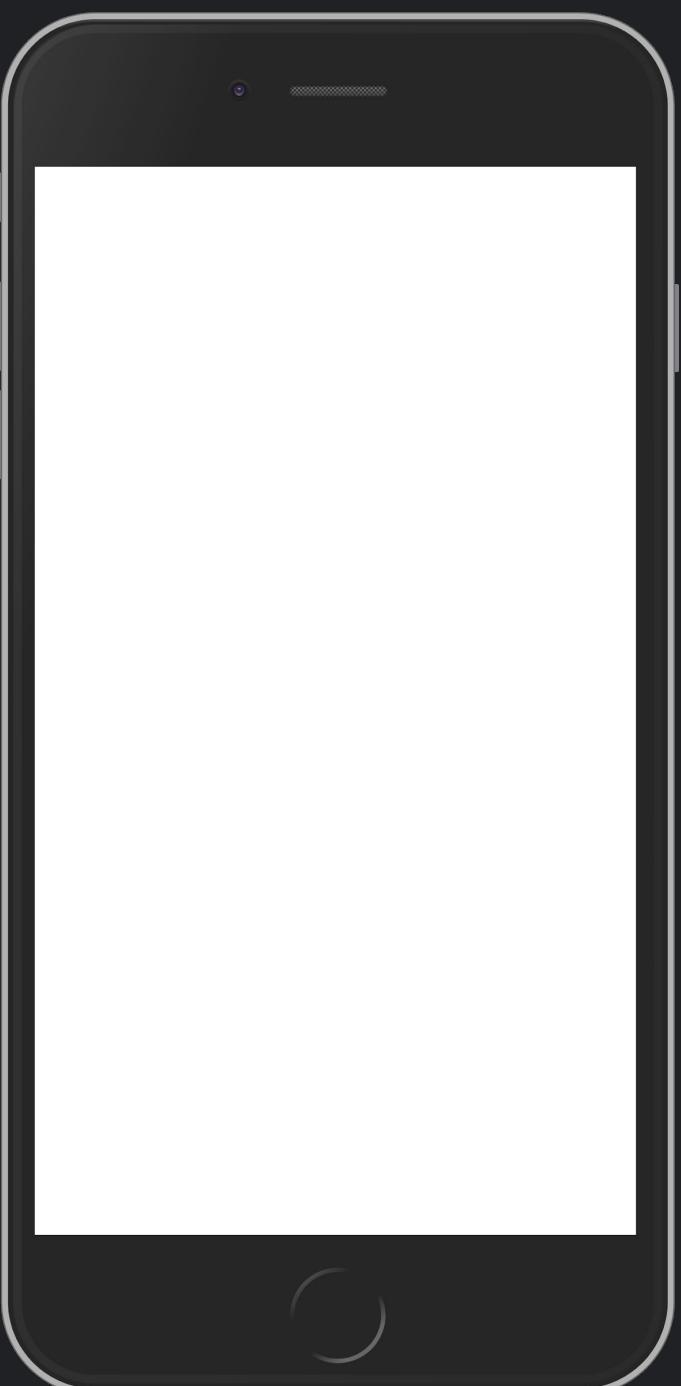
Mobile




```
it('should work on mobile', async () => {  
  const page = await context.newPage();  
  
}) )
```



```
it('should work on mobile', async () => {  
  const page = await context.newPage();  
  await page.emulate(puppeteer.devices['iPhone 8']);  
})
```



```
it('should work on mobile', async () => {
  const page = await context.newPage();
  await page.emulate(puppeteer.devices['iPhone 8']);
  await page.goto('https://joel.tools/merch/');
})
```

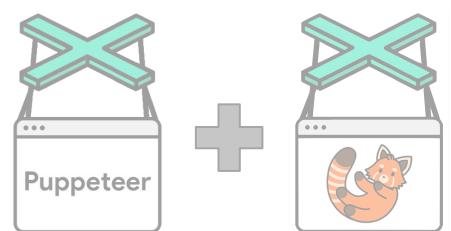


```
it('should work on mobile', async () => {
  const page = await context.newPage();
  await page.emulate(puppeteer.devices['iPhone 8']);
  await page.goto('https://joel.tools/merch/');
  const payButton = await page.waitForSelector('button.gpay-button');
  expect(payButton).toBeTruthy();
})
```

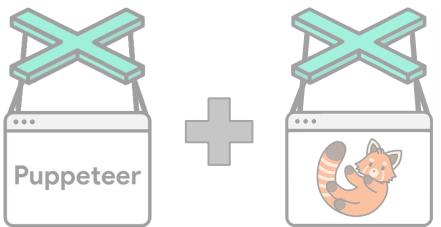


Device Emulation

- 👉 User Agent
- 👉 Device Pixel Ratio
- 👉 Viewport size
- 👉 Touch Support

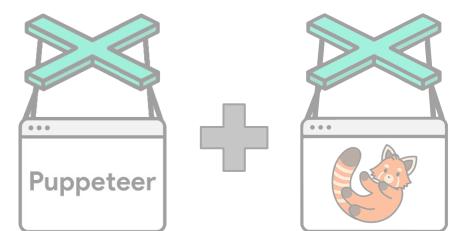
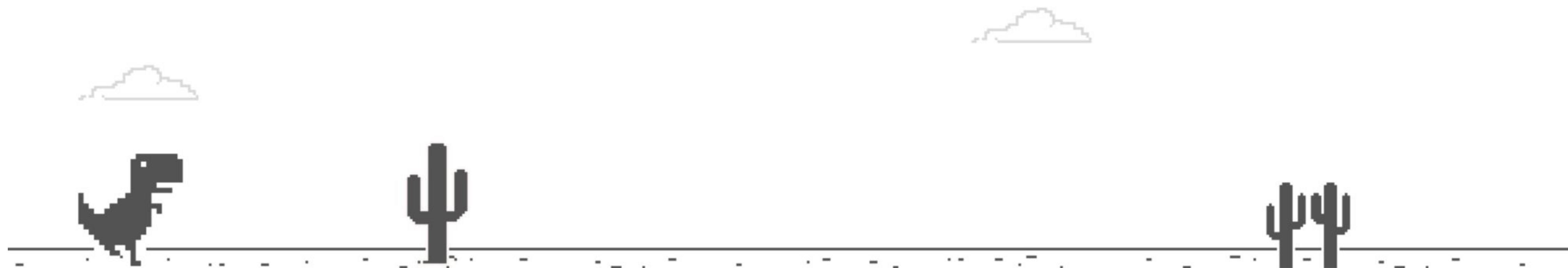


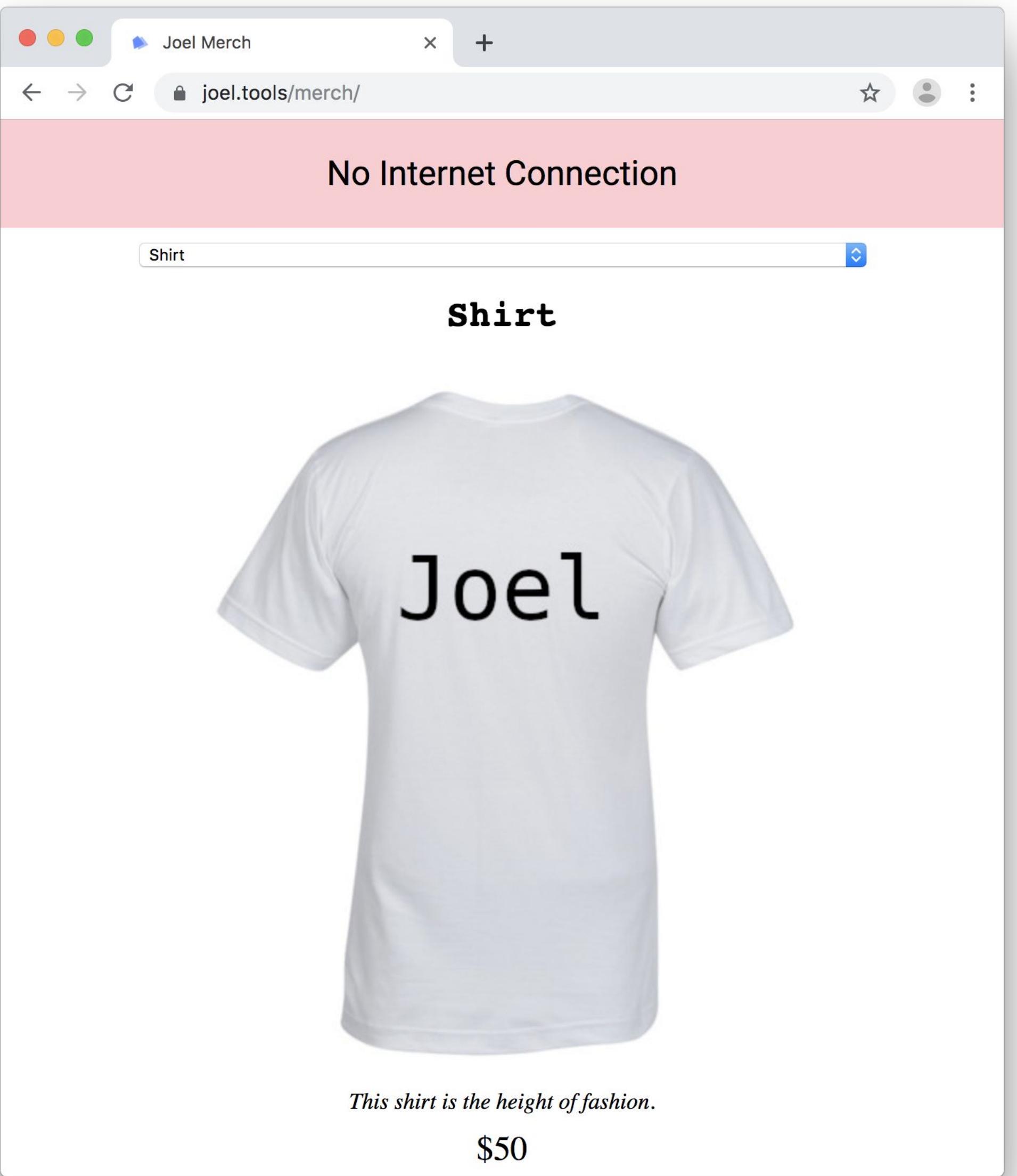
100+ devices



Offline Emulation

HI 00072






```
it('should notify user when goes offline', async () => {  
  const page = await context.newPage();  
  
});
```



```
it('should notify user when goes offline', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
}) ;
```



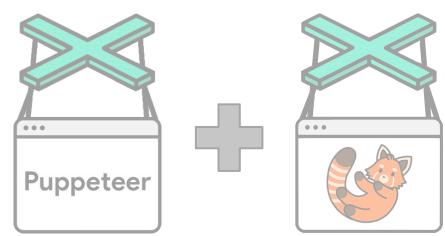
```
it('should notify user when goes offline', async () => {  
  const page = await context.newPage();  
  await page.goto('https://joel.tools/merch');  
  await page.setOfflineMode(true);  
});
```



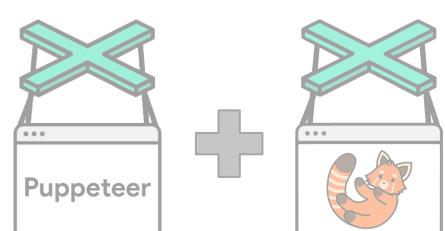
```
it('should notify user when goes offline', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  await page.setOfflineMode(true);
  const msg = await page.waitForSelector('.offline-alert');
  expect(msg).toBeTruthy();
});
```



Service Workers



pptr.dev



The screenshot shows a browser window displaying the `page.evaluate` API documentation from [pptr.dev](https://pptr.dev/#?product=Puppeteer&version=v1.15.0&show=api-page-evaluate). The page has a teal header with the title `page.evaluate`. The main content describes the `page.evaluate` method, which takes a function or string and returns a promise. It includes examples of passing arguments and strings, and mentions ElementHandle instances.

M `page.evaluate(pageFunction[, ...args])` since v0.9.0

- `pageFunction <function|string>` Function to be evaluated in the page context
- `...args <...Serializable|JSHandle>` Arguments to pass to `pageFunction`
- returns: `<Promise<Serializable>>` Promise which resolves to the return value of `pageFunction`

If the function passed to the `page.evaluate` returns a `Promise`, then `page.evaluate` would wait for the promise to resolve and return its value.

If the function passed to the `page.evaluate` returns a non-`Serializable` value, then `page.evaluate` resolves to `undefined`. DevTools Protocol also supports transferring some additional values that are not serializable by `JSON`: `-0`, `Nan`, `Infinity`, `-Infinity`, and bigint literals.

Passing arguments to `pageFunction`:

```
const result = await page.evaluate(x => {
  return Promise.resolve(8 * x);
}, 7);
console.log(result); // prints "56"
```

A string can also be passed in instead of a function:

```
console.log(await page.evaluate('1 + 2')); // prints "3"
const x = 10;
console.log(await page.evaluate(`1 + ${x}`)); // prints "11"
```

ElementHandle instances can be passed as arguments to the `page.evaluate`:



Elements Network Console Performance Sources Application >⋮×

Application

- Manifest
- Service Workers**
- Clear storage

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

Cache

- Cache Storage
- Application Cache

Frames

- top

Service Workers

Offline Update on reload Bypass for network

pptr.dev [Update](#) [Unregister](#)

Source [sw.js](#)
Received 3/20/2019, 7:13:00 PM

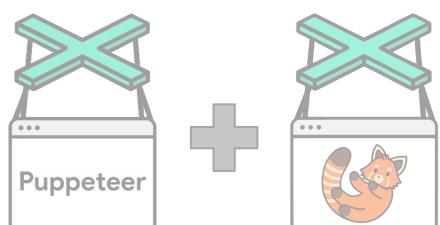
Status ● #33041 activated and is running [stop](#)

Clients <https://pptr.dev/> [focus](#)

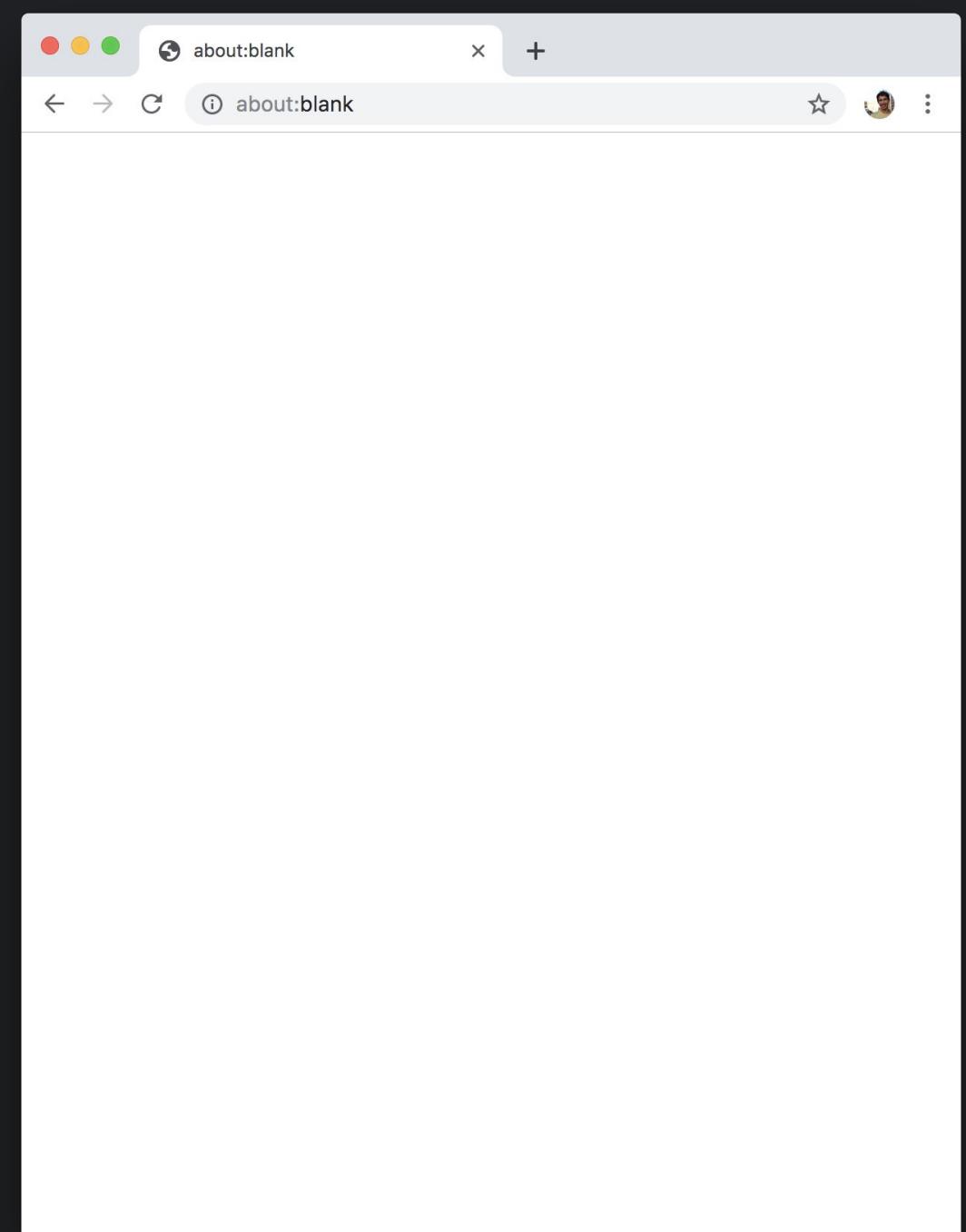
Push [Push](#)

Sync [Sync](#)

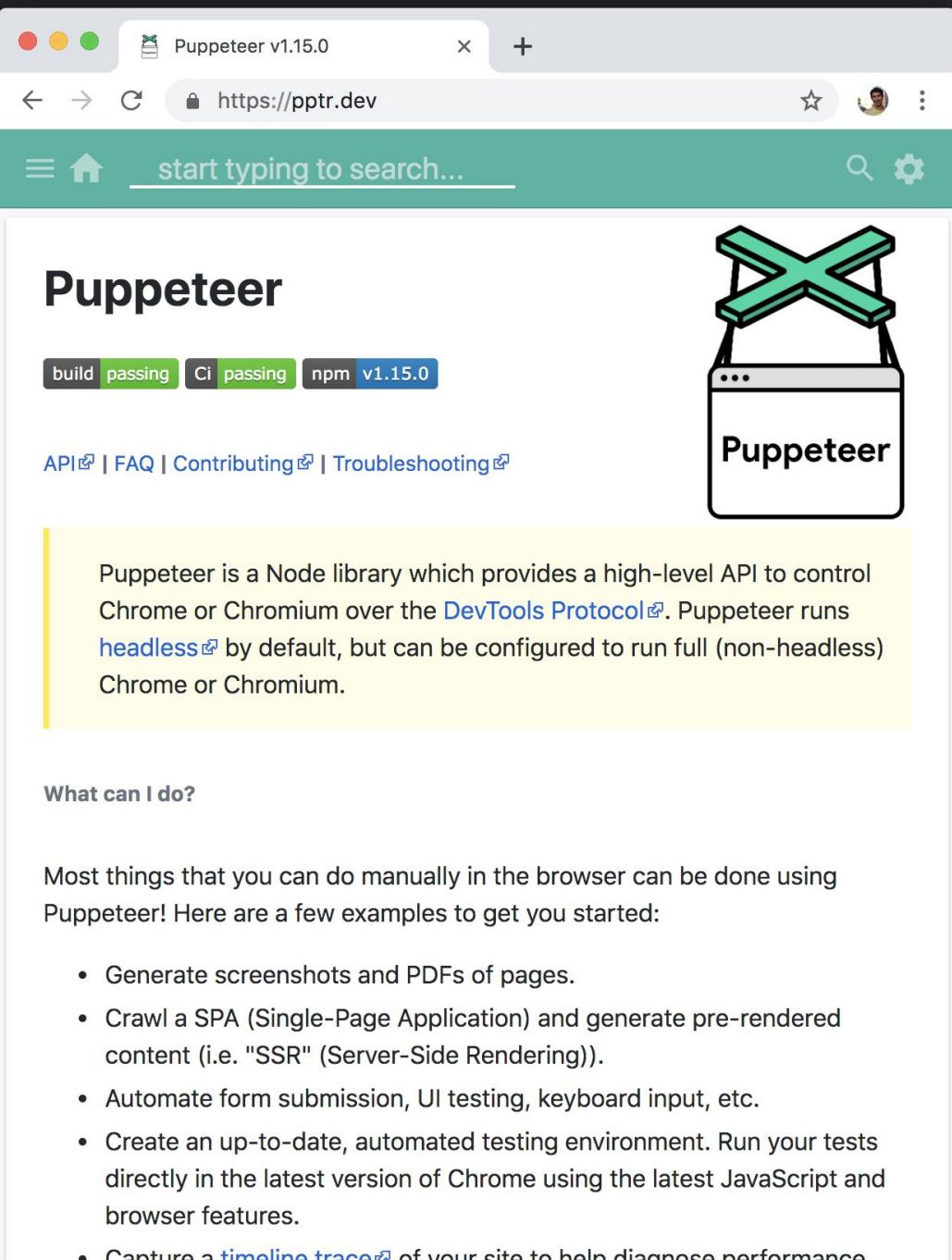
▶ Service workers from other domains



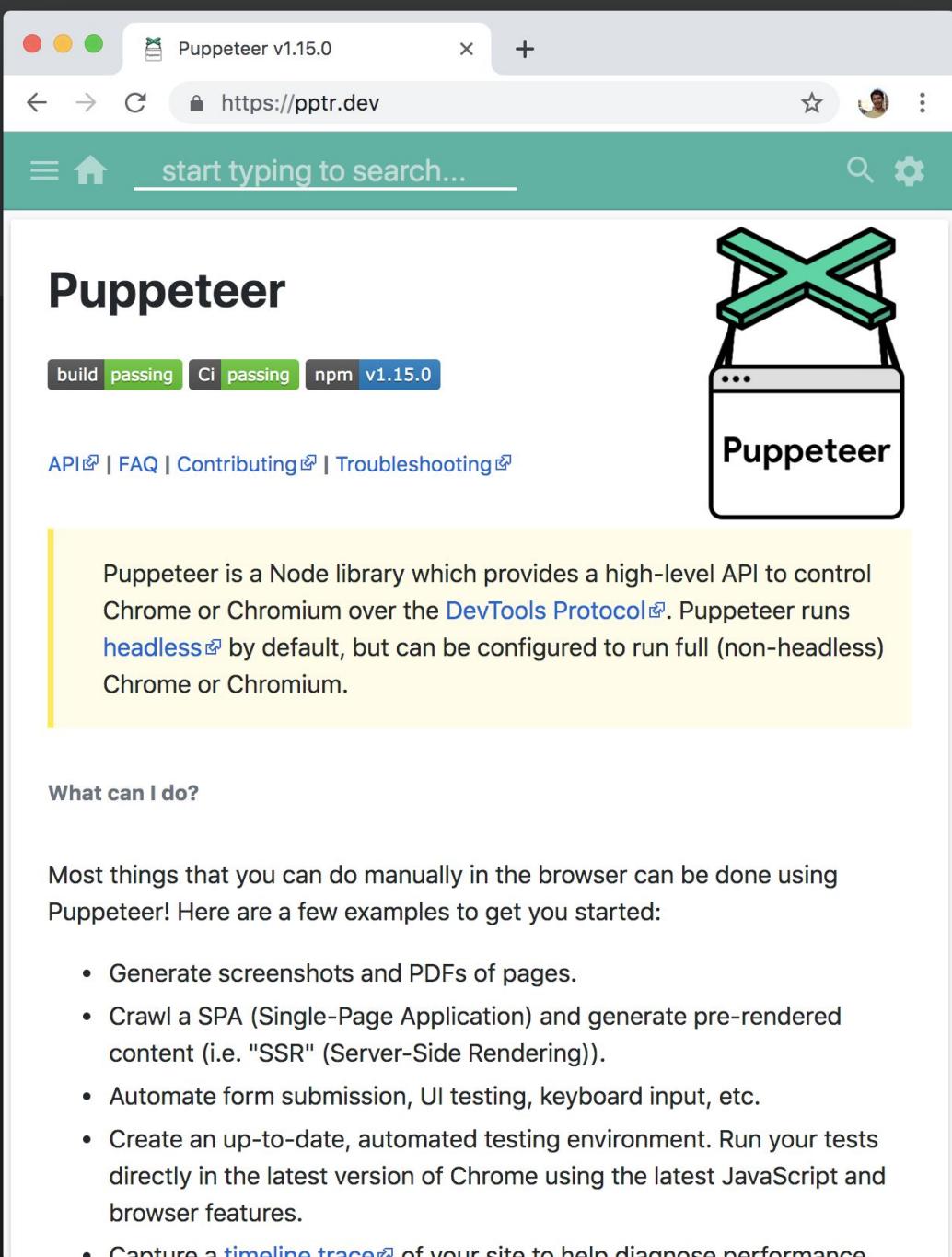

```
it('should register a service worker', async () => {  
  const page = await context.newPage();  
  
})
```



```
it('should register a service worker', async () => {
  const page = await context.newPage();
  await page.goto('https://pptr.dev');
})
```



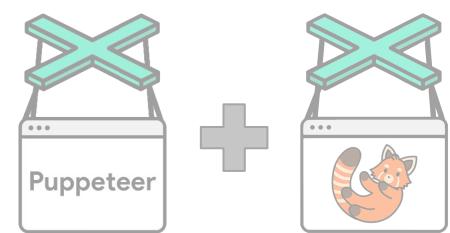
```
it('should register a service worker', async () => {
  const page = await context.newPage();
  await page.goto('https://pptr.dev');
  const swTarget = await context.waitForTarget(target => {
    return target.type() === 'service_worker';
  });
})
```



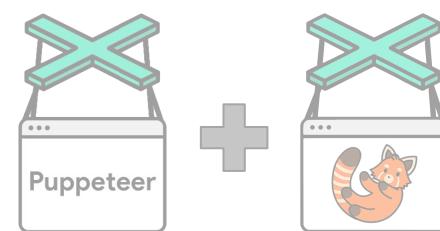
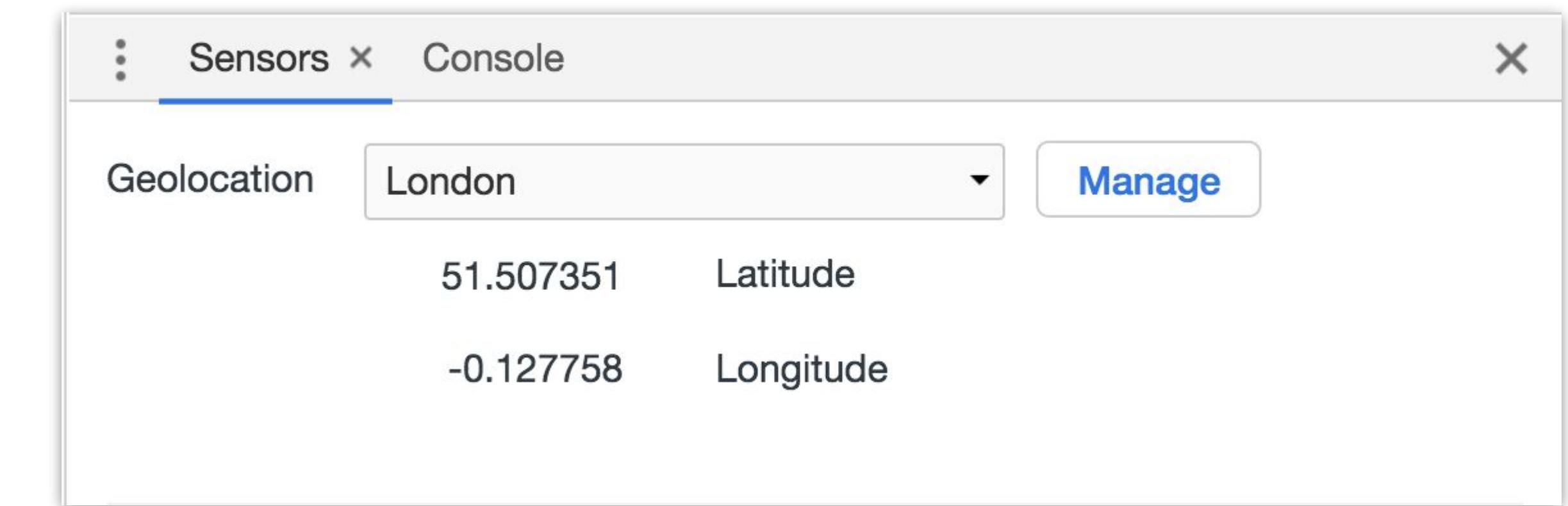
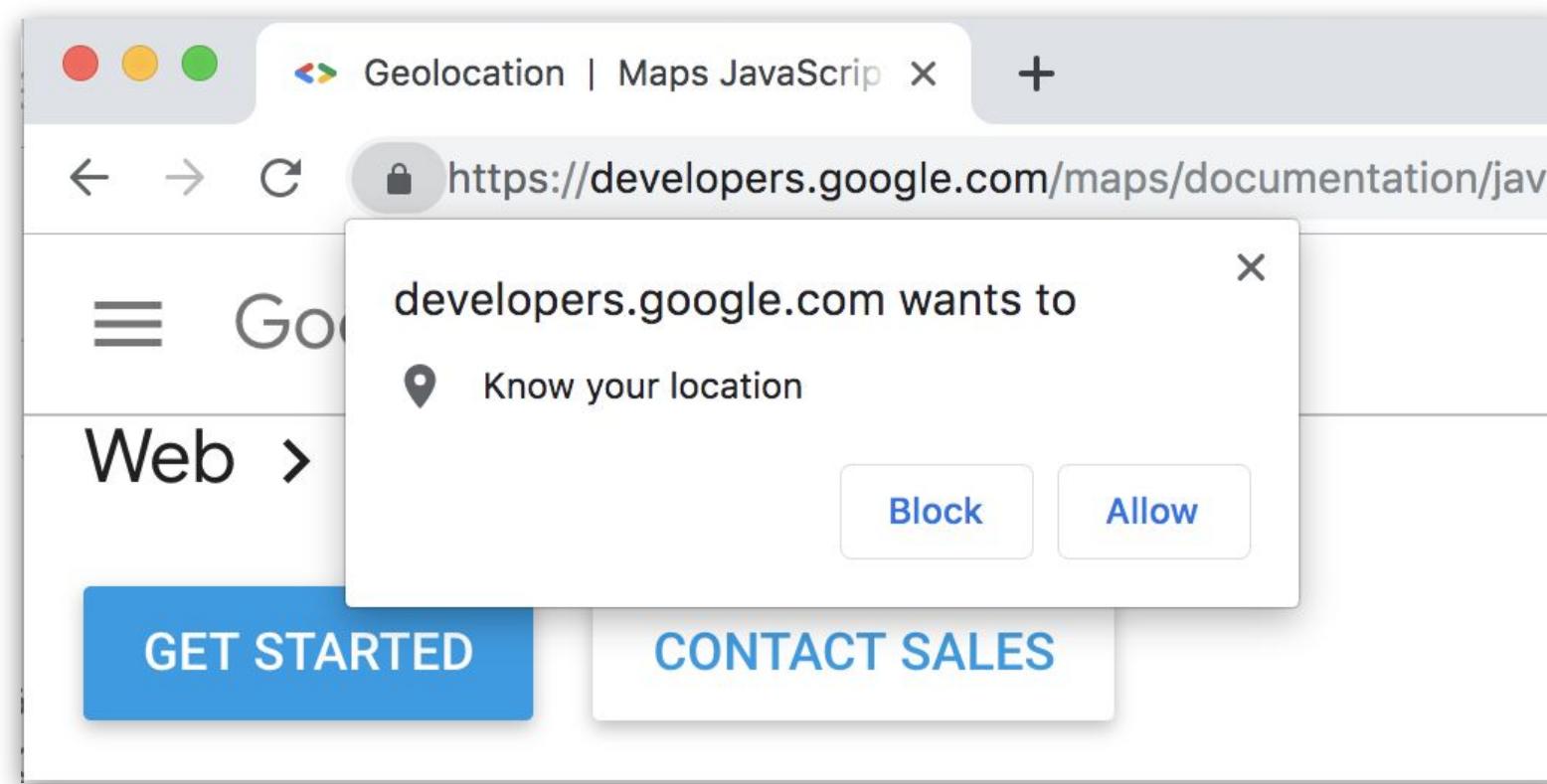

```
...  
const swTarget = await context.waitForTarget(target => {  
    return target.type() === 'service_worker';  
});  
const serviceWorker = await swTarget.worker();
```

```
...  
const swTarget = await context.waitForTarget(target => {  
    return target.type() === 'service_worker';  
});  
const serviceWorker = await swTarget.worker();  
const isCached = await serviceWorker.evaluate(async () => {  
    return !!(await caches.match('https://pptr.dev/logo.png'));  
});  
expect(isCached).toBe(true);
```

Geolocation



Geolocation: London

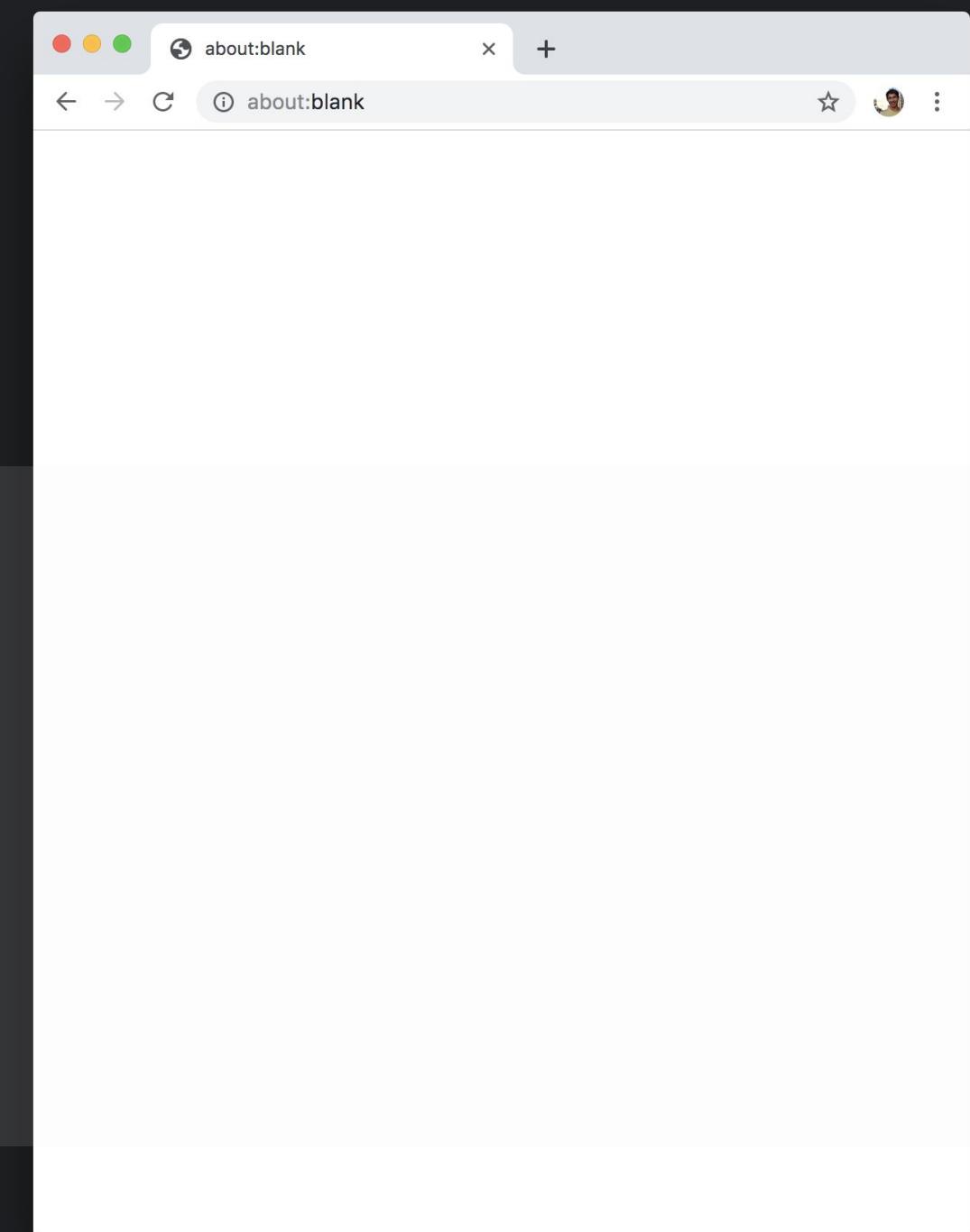



```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
})
```

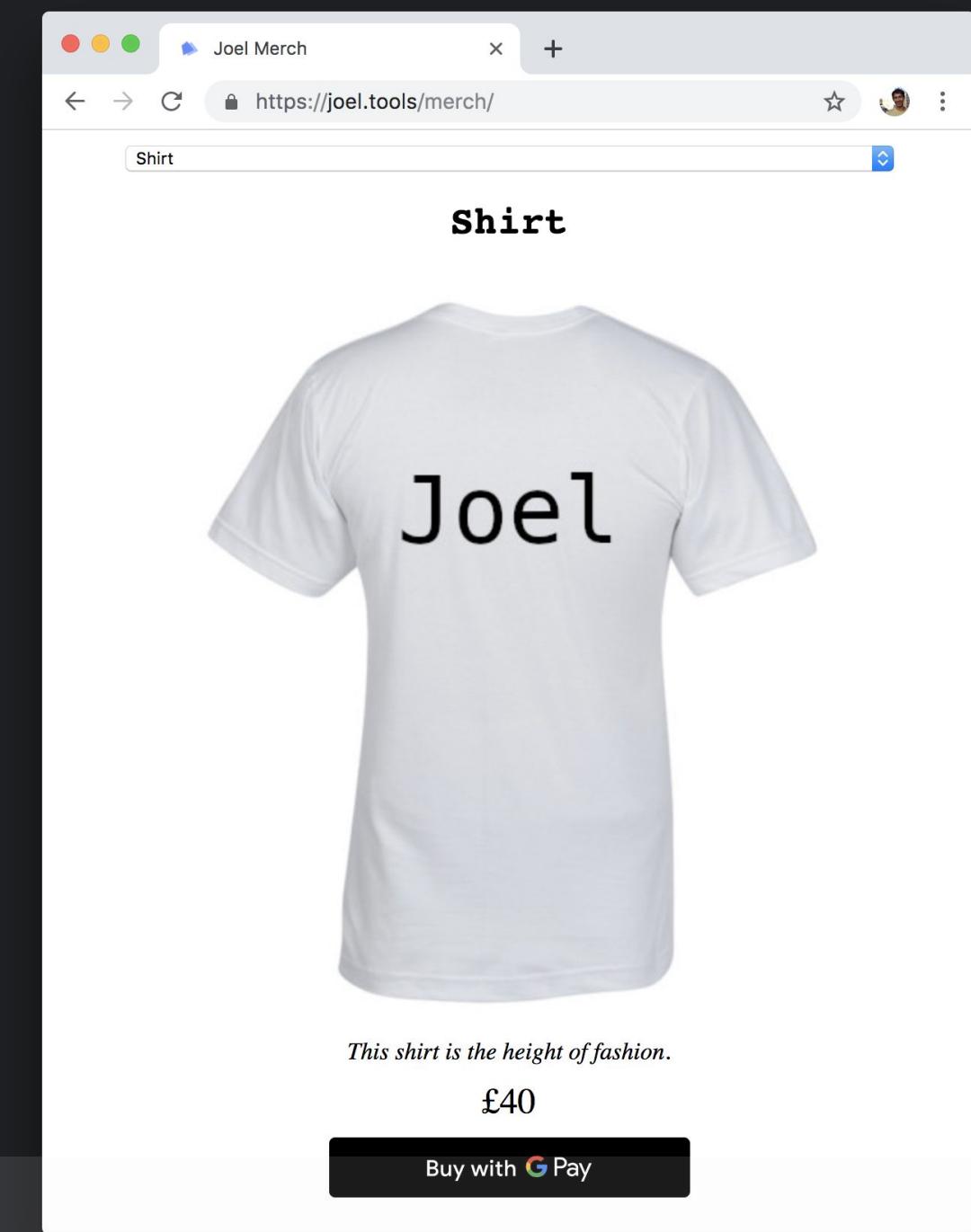
```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
  const page = await context.newPage();
})
```



```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
  const page = await context.newPage();
  await page.setGeolocation({
    latitude: 51.5,
    longitude: -0.12,
  });
})
```



```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
  const page = await context.newPage();
  await page.setGeolocation({
    latitude: 51.5,
    longitude: -0.12,
  });
  await page.goto('https://joel.tools/merch');
})
```

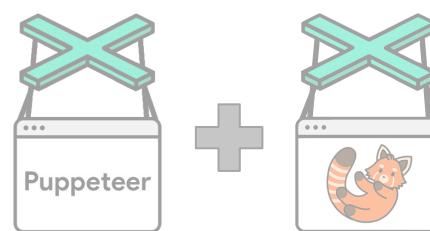
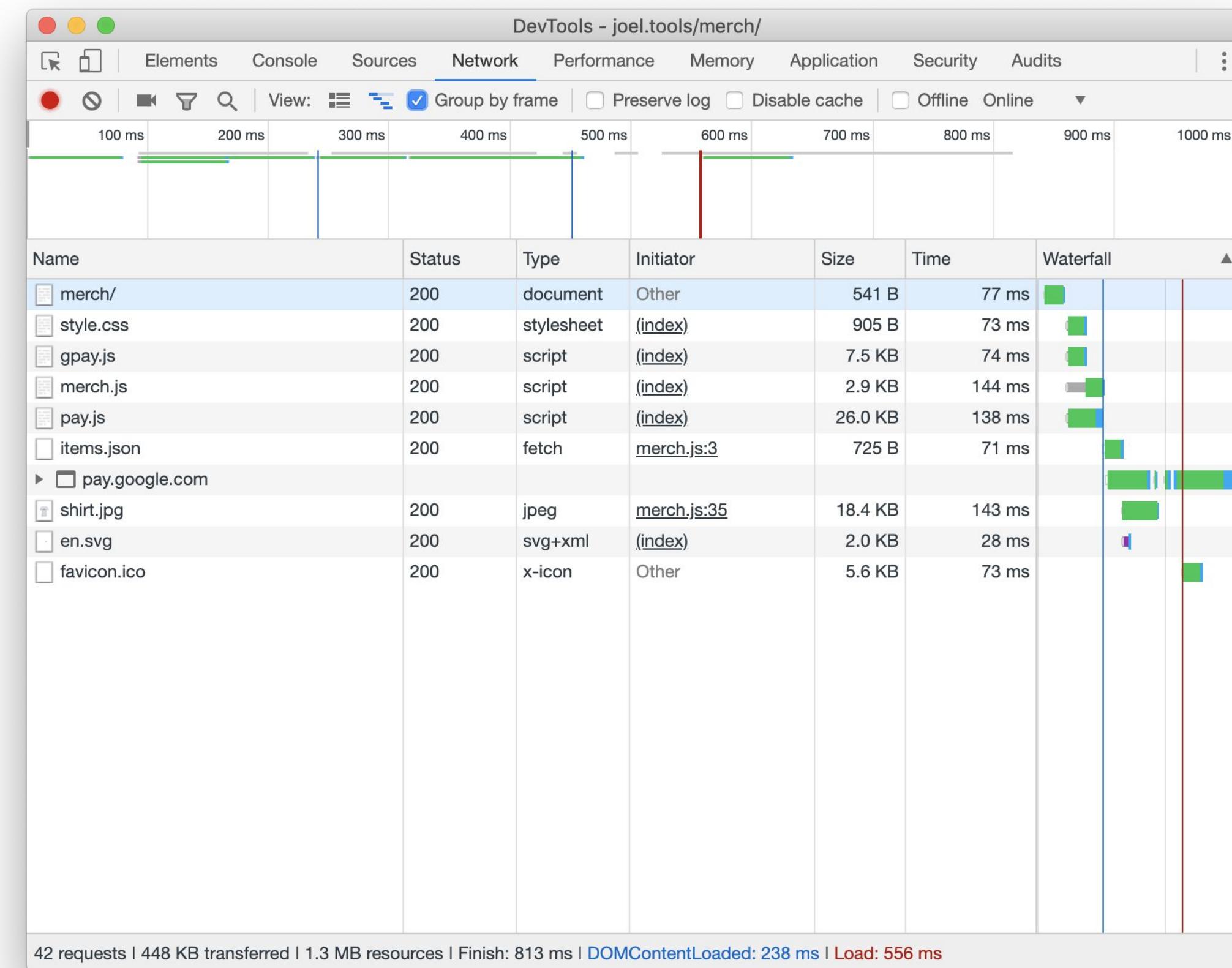


```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
  const page = await context.newPage();
  await page.setGeolocation({
    latitude: 51.5,
    longitude: -0.12,
  });
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  expect(price).toBe('£40');
})
```



£40

Network Monitoring

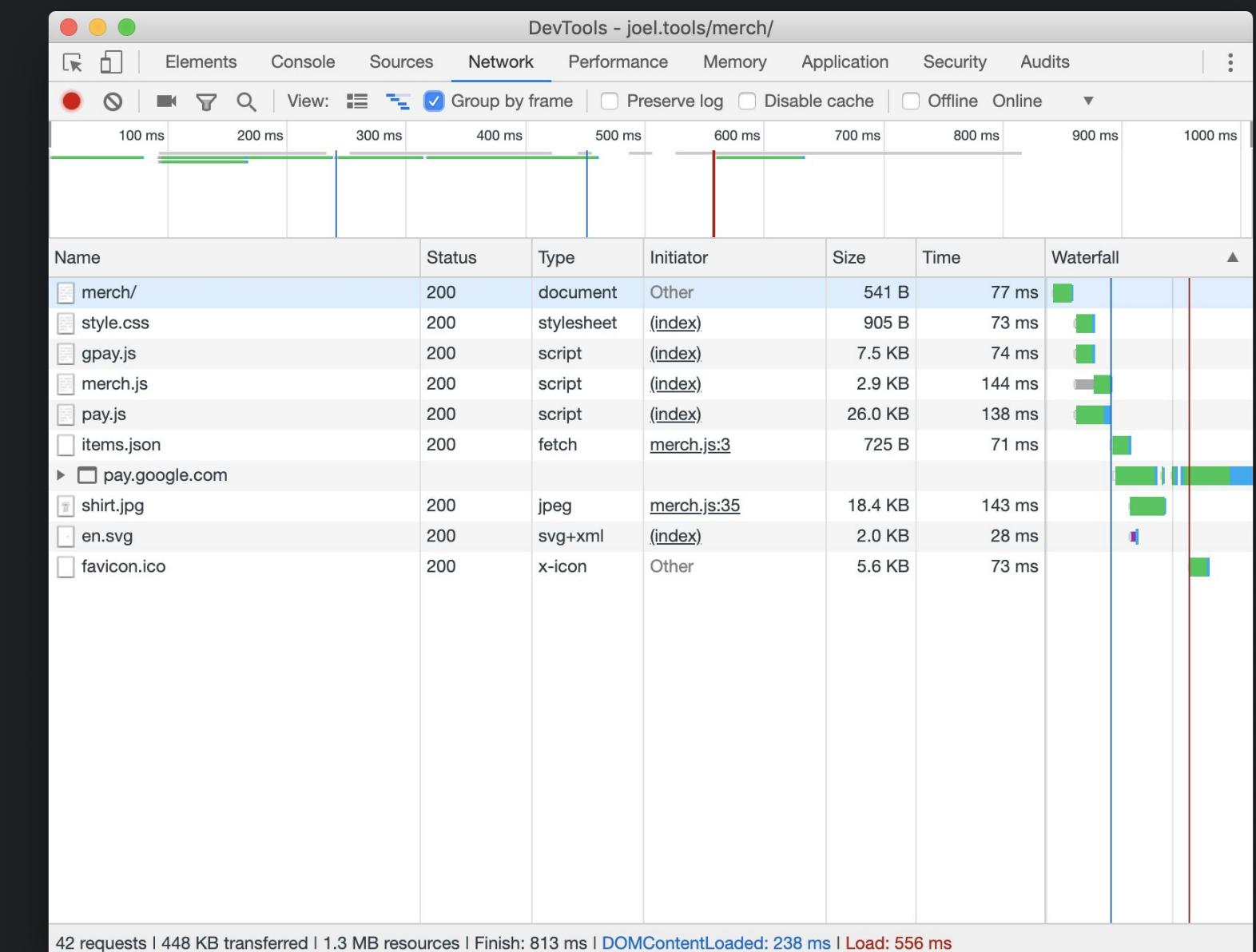


```

page.on('request', request => {
  console.log(request.method(), request.url());
}) ;

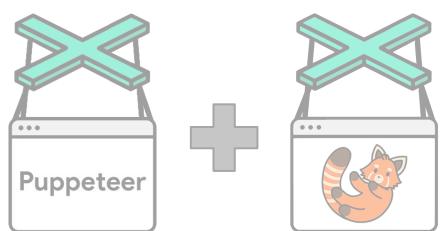
page.on('response', response => {
  console.log(response.status(), response.url());
}) ;

```

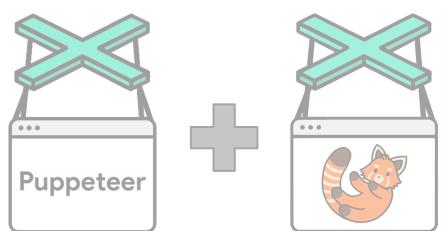
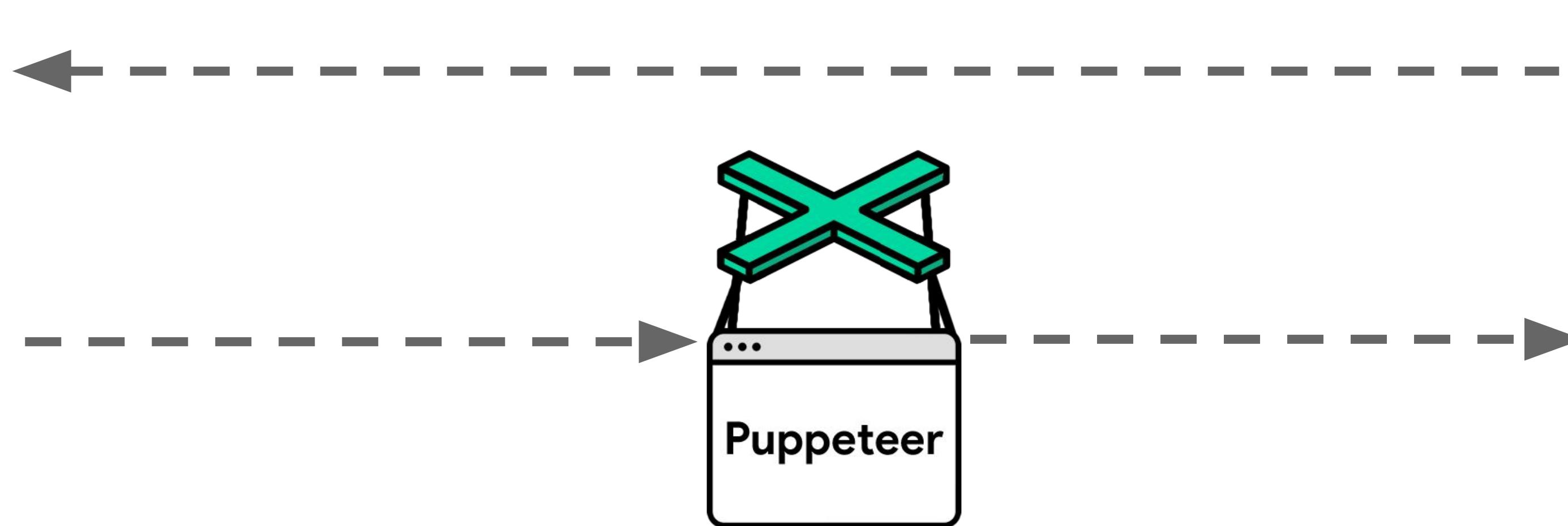
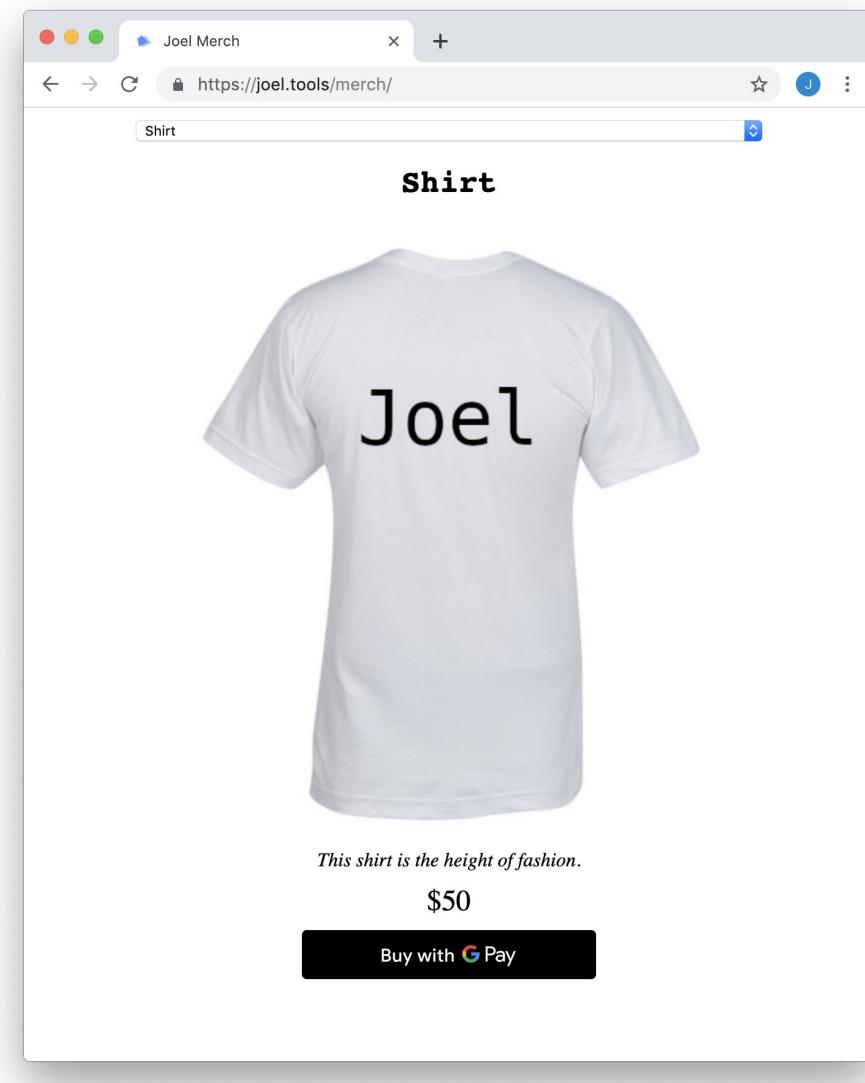


Network Monitoring

- 👉 Headers
- 👉 POST Data
- 👉 Content
- 👉 From cache / From Service Worker



Request Interception



Google Developers

https://developers.google.com

Chrome is being controlled by automated test software.

Products Events Developer Programs Blog SIGN IN

Build anything with Google

Search products and documentation

Events

[VIEW ALL](#)

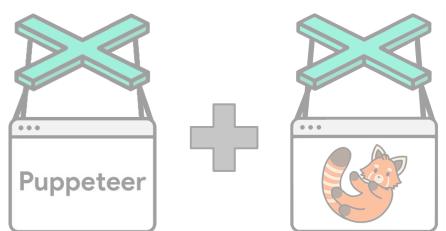


Google I/O'19
May 7-9 | Mountain View, CA,
USA

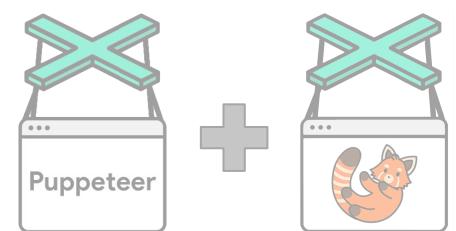
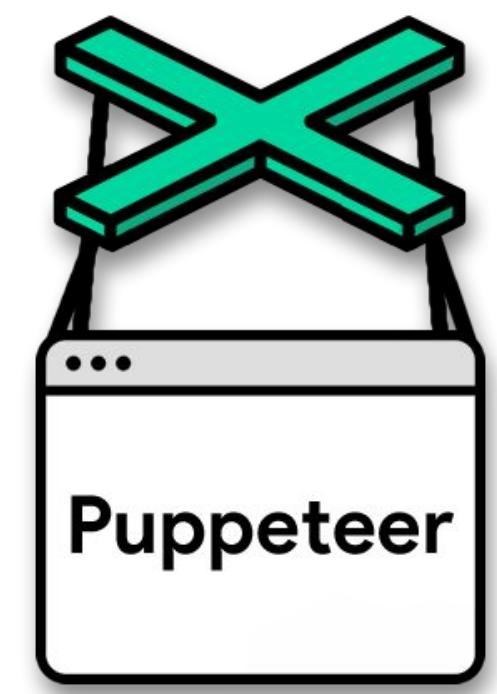
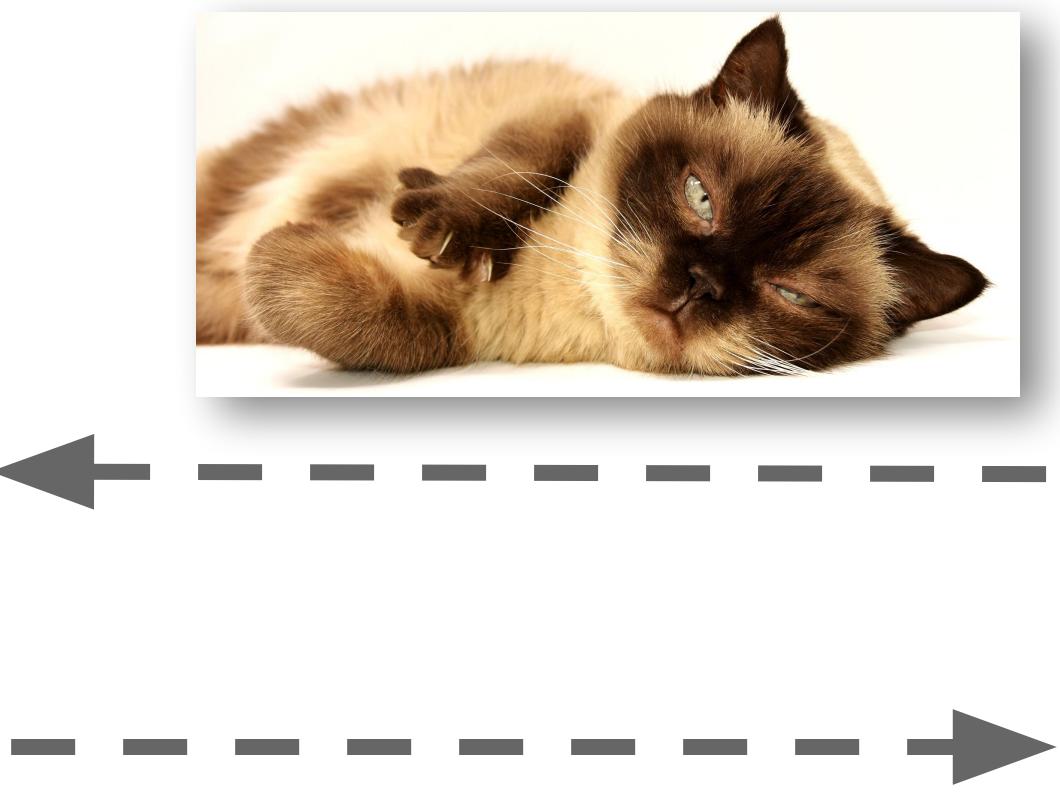
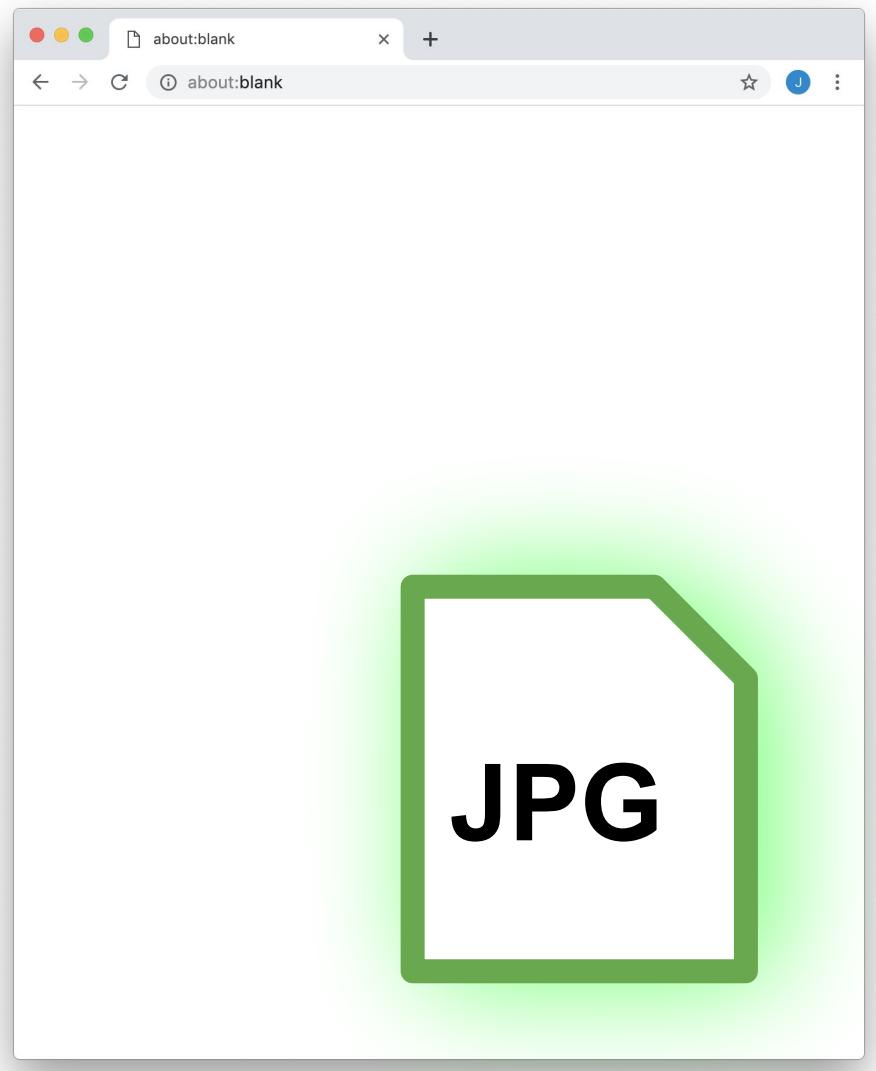


Android Dev Summit 2019
October 23-24 | Sunnyvale, CA,
USA

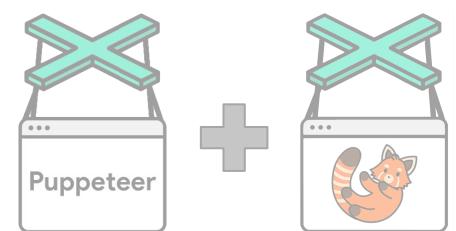
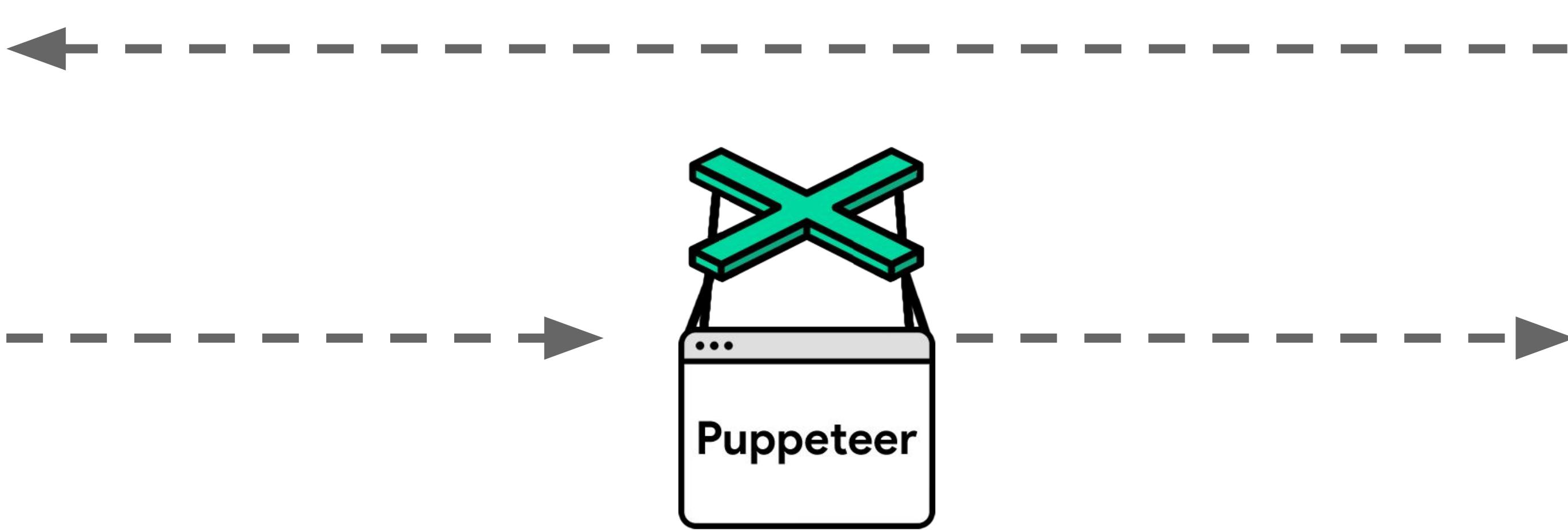
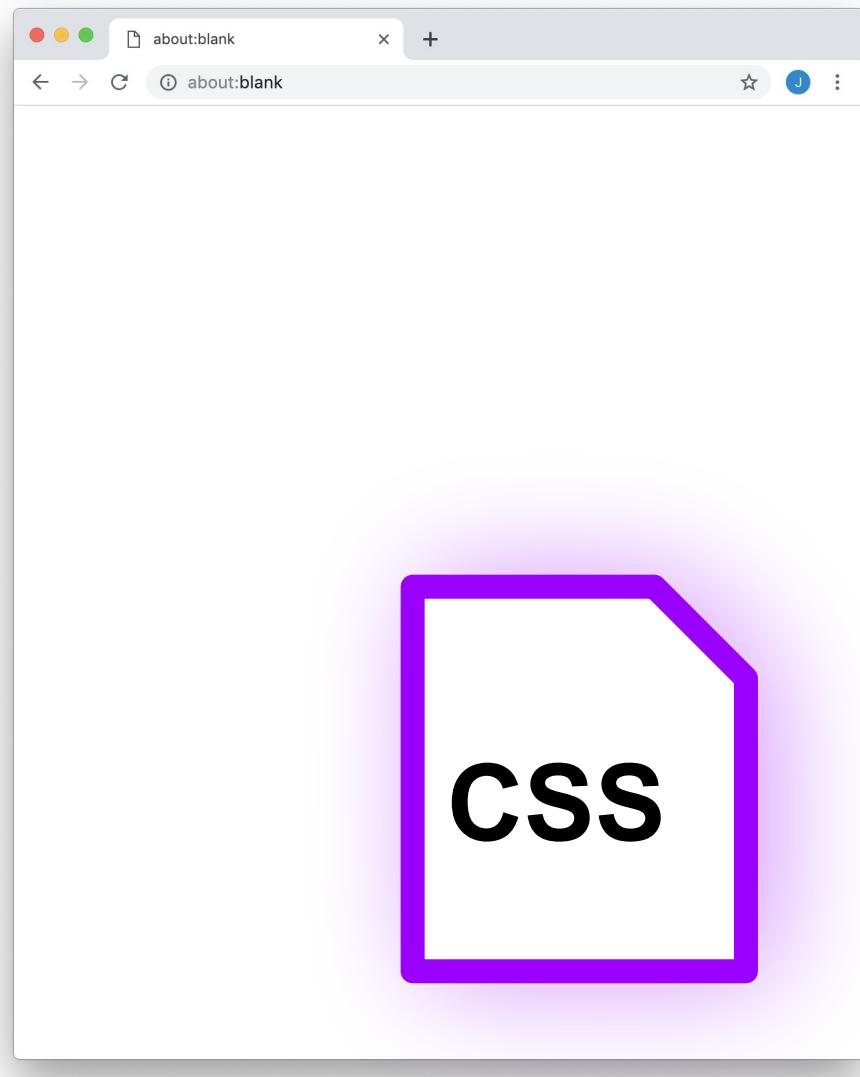
https://events.google.com/io/?utm_source=devsite&utm_medium=events&utm_campaign=upcoming&utm_content=DevSite_HPP



HOLY JS!...
2019
PITER



HOLY
JS!...
2019
PITER



HOLY
JS!...
2019
PITER

```
await page.setRequestInterception(true);

page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```

```
await page.setRequestInterception(true);

page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```

```
await page.setRequestInterception(true);

page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```

```
await page.setRequestInterception(true);

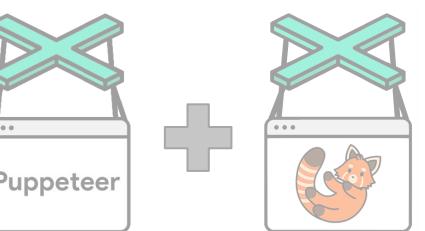
page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```

```
await page.setRequestInterception(true);

page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```

```
await page.setRequestInterception(true);

page.on('request', request => {
  if (request.resourceType() === 'image')
    request.respond({ body: randomCatImage() });
  else
    request.continue();
});
```



News about #io19 on Twitter

https://twitter.com/hashtag/io19?lang=en

Chrome is being controlled by automated test software.

#io19

Have an account? Log in

#io19

Top Latest People Photos Videos News Broadcasts

Search filters · Show

Related searches

- #googleio
- #googleio2019

New to Twitter?

Sign up now to get your own personalized timeline!

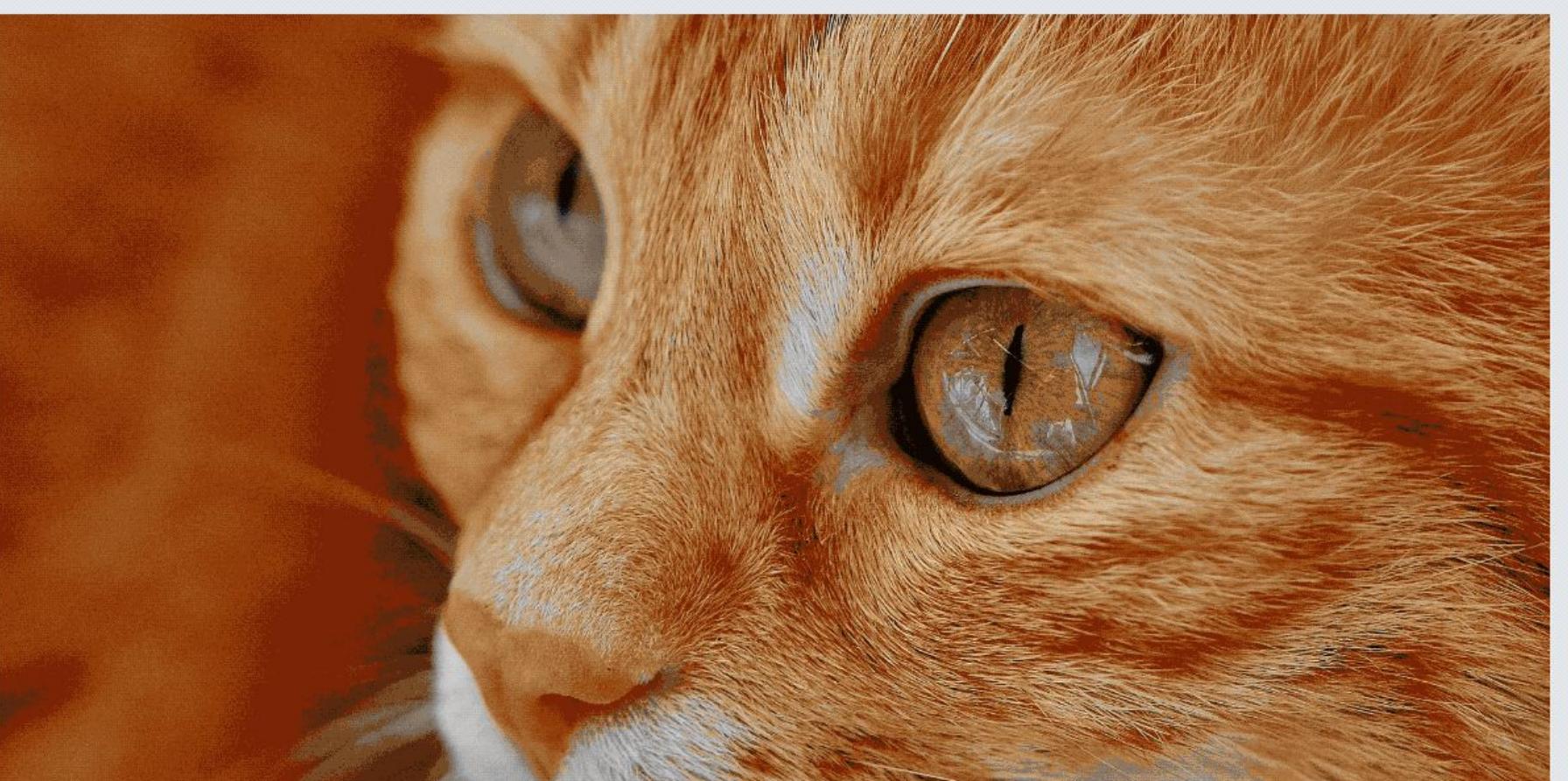
Sign up

© 2019 Twitter About Help Center Terms
Privacy policy Cookies Ads info

Waiting for twitter.com...

Top news

View all



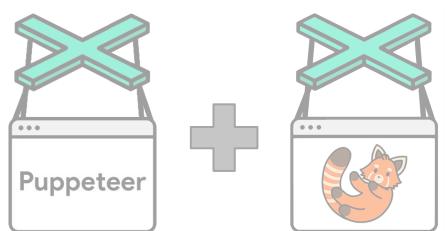
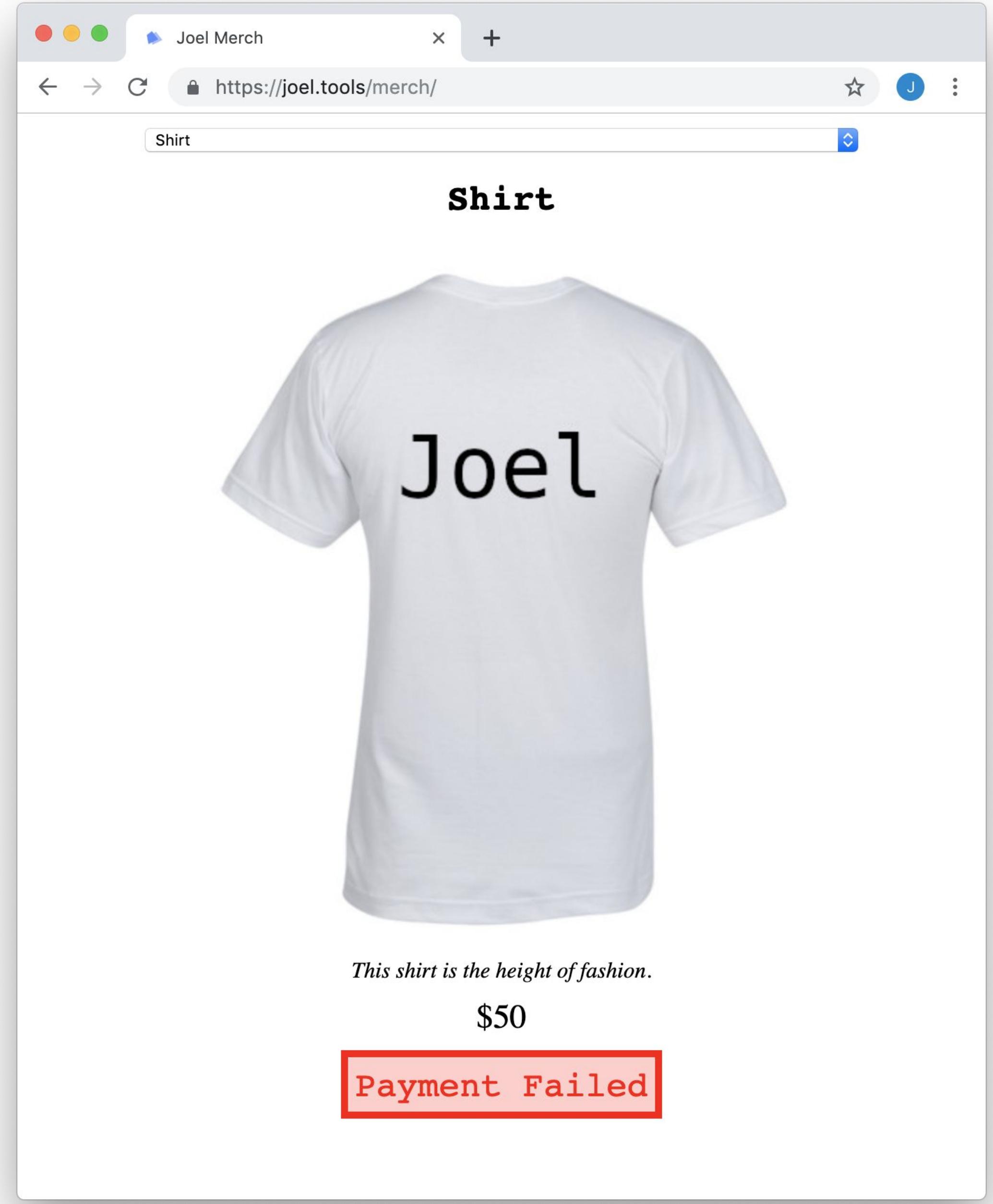
Google brings AR and Lens closer to the future of search

Exclusive: How Google's building the AR glue to a browsable reality.

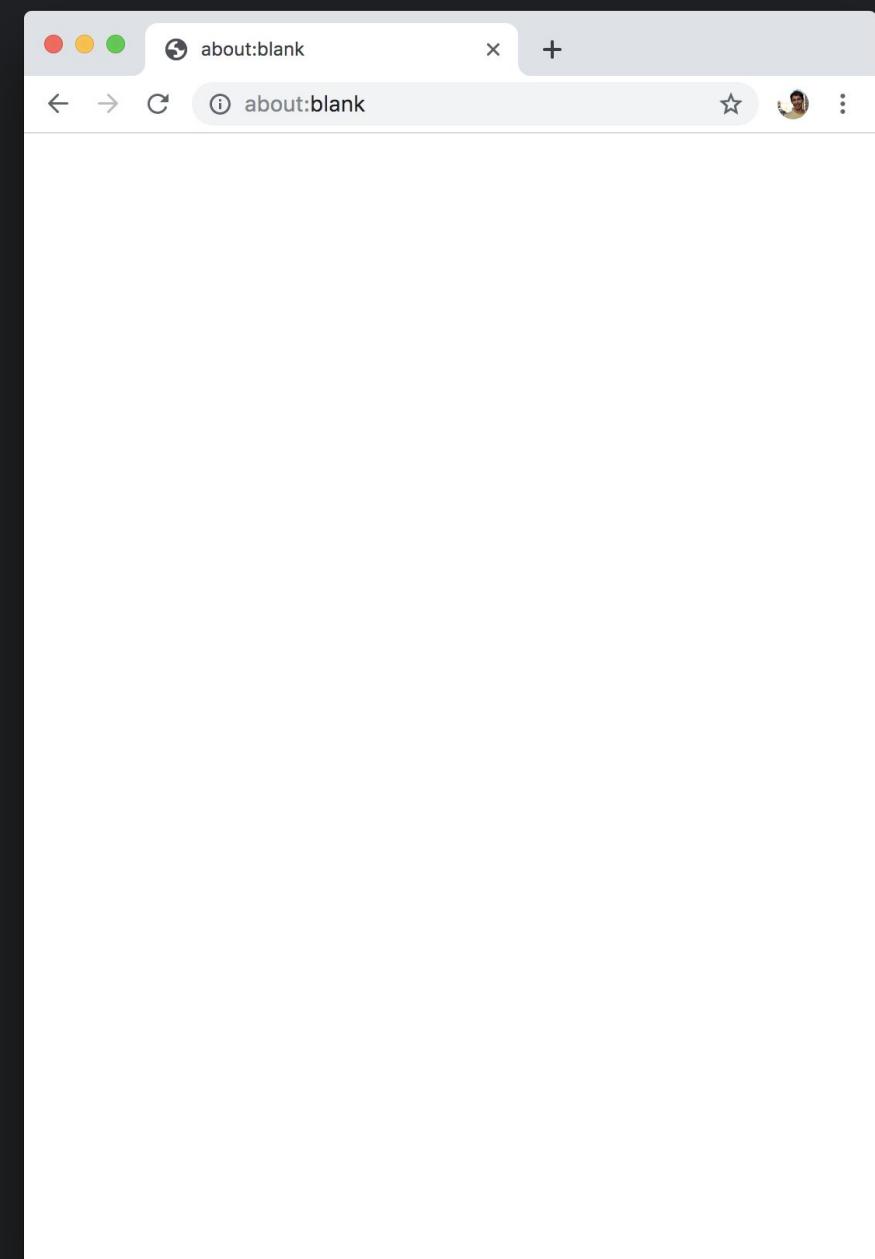
CNET May 07, 2019

Lance Ulanoff @LanceUlanoff · 3h
You know @bts_bighit has arrived when they're included in a #GoogleIO2019 keynote. #io19

HOLY JS!...
2019 PITER




```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();  
  
}) )
```



```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();  
  await page.setRequestInterception(true);  
}  
})
```



```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();  
  await page.setRequestInterception(true);  
  page.on('request', request => {  
    } );  
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
  } );
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
  });
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
  } );
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
      request.continue();
  });
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
      request.continue();
  });
  await page.goto('https://joel.tools/merch');
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
      request.continue();
  });
  await page.goto('https://joel.tools/merch');
  await page.click('button.gpay-button');
})
```



```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
      request.continue();
  });
  await page.goto('https://joel.tools/merch');
  await page.click('button.gpay-button');
  await page.waitForSelector('.payment-failed');
})
```

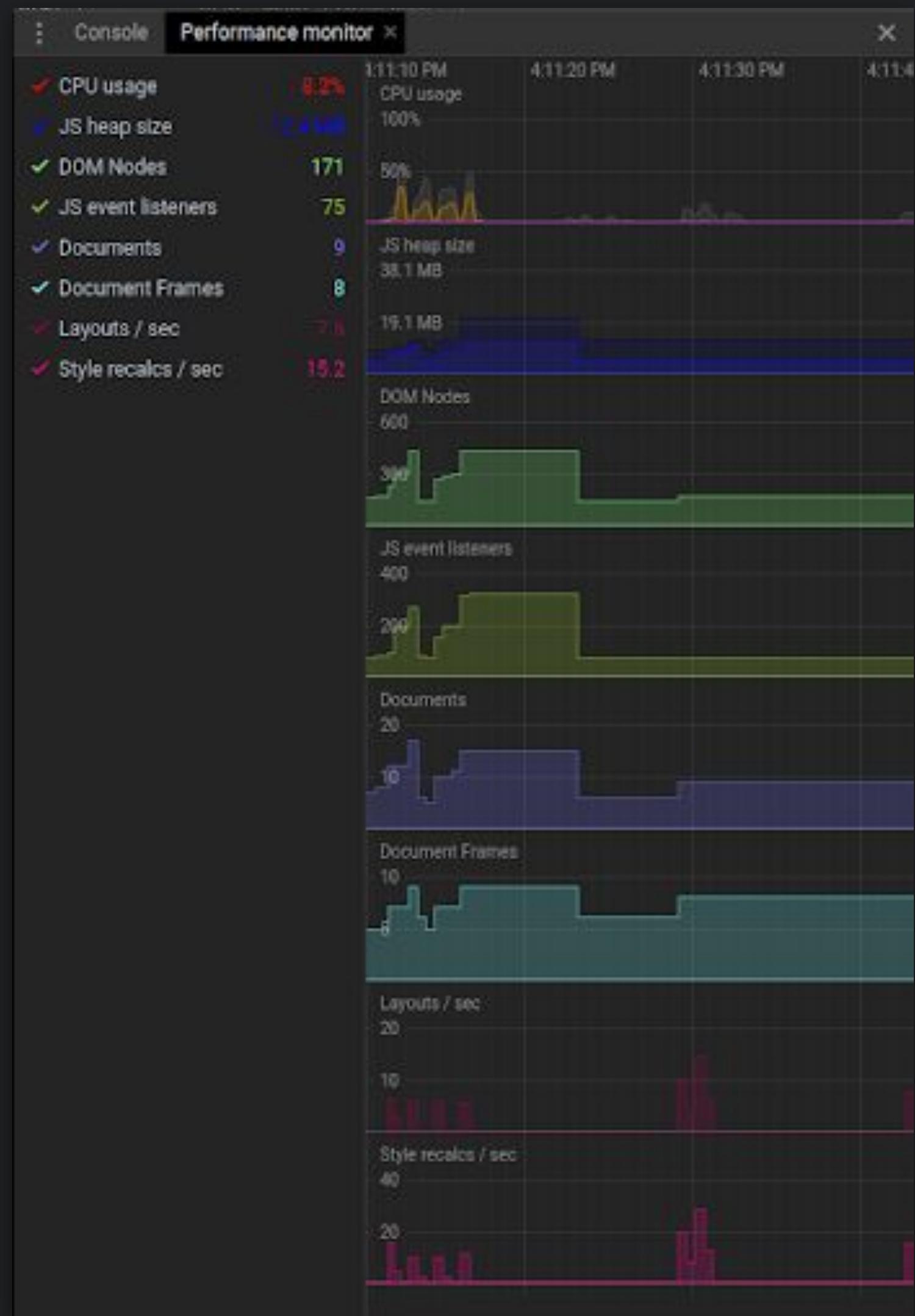


Performance Testing

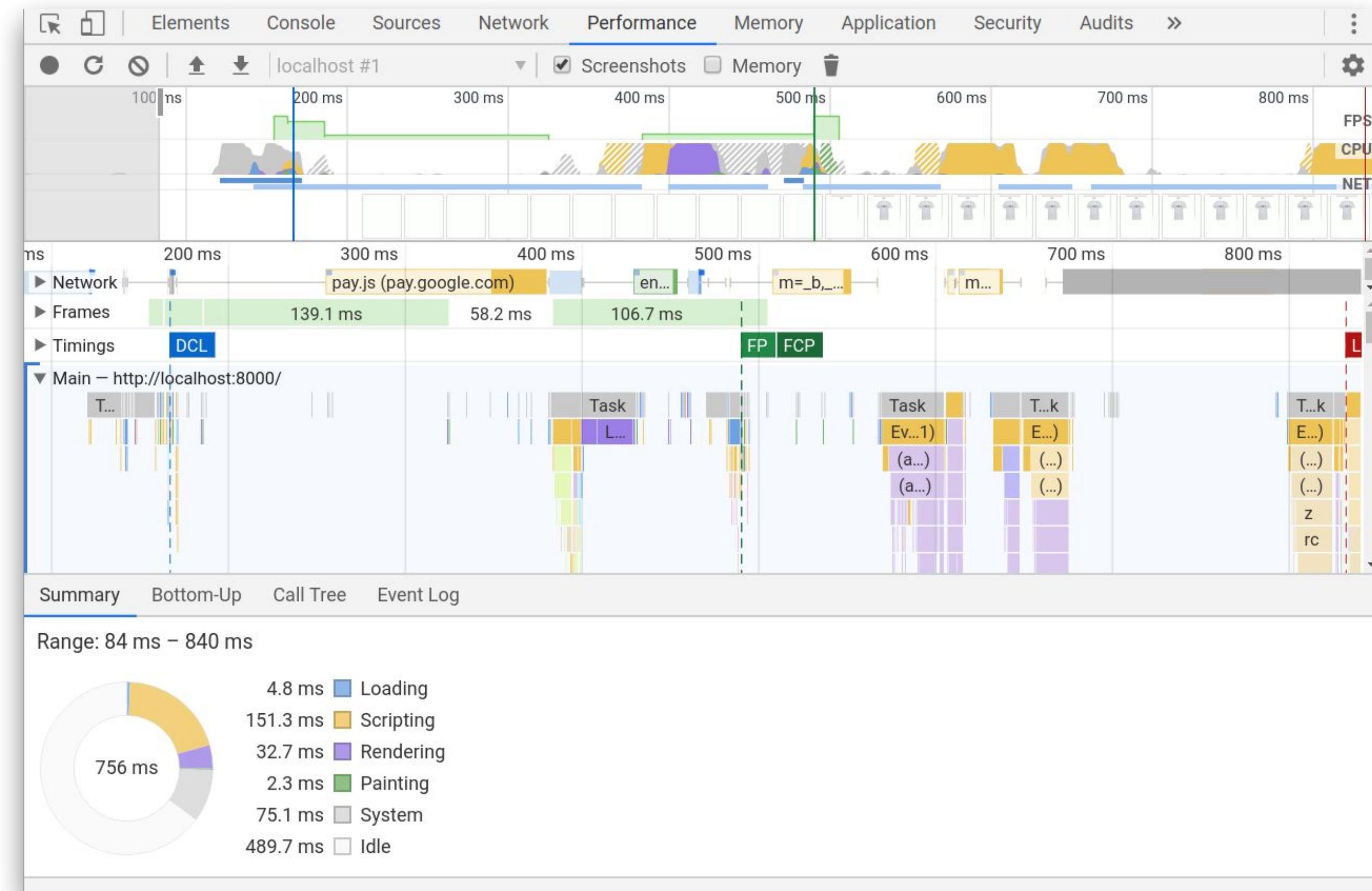
- 👉 Metrics
- 👉 Chrome Tracing
- 👉 Code Coverage



```
const metrics = await page.metrics();  
  
console.log(metrics);  
  
{ Timestamp: 3180099.406978,  
  Documents: 7,  
  Frames: 3,  
  JSEventListeners: 121,  
  Nodes: 156,  
  LayoutCount: 6,  
  RecalcStyleCount: 7,  
  LayoutDuration: 0.021108,  
  RecalcStyleDuration: 0.009698,  
  ScriptDuration: 0.158487,  
  TaskDuration: 0.401342,  
  JSHeapUsedSize: 7107312,  
  JSHeapTotalSize: 11689984 }
```



Chrome Tracing



Puppeteer



```
await page.goto('http://joel.tools/merch/');  
await page.waitForSelector('select');  
await page.select('select', 'mug');
```

```
await page.tracing.start({ path: './trace.json' });

await page.goto('http://joel.tools/merch/');
await page.waitForSelector('select');
await page.select('select', 'mug');
```

```
await page.tracing.start({ path: './trace.json' });

await page.goto('http://joel.tools/merch/');
await page.waitForSelector('select');
await page.select('select', 'mug');

await page.tracing.stop();
```

Code Coverage

The screenshot shows the Google Chrome DevTools interface with the "Sources" tab selected. The left sidebar displays a file tree for the page, including files like `style.css`, `index`, `gpay.js`, and `merch.js`. The right pane shows the content of `style.css` with line numbers 11 through 28. A vertical red bar highlights line 22, which contains the CSS rule for the `.out-of-stock` class. Below the code editor, a message indicates "Line 22, Column 2". At the bottom of the DevTools window, the "Coverage" tab is active, showing a table of resource usage statistics:

URL	Type	Total Bytes	Unused Bytes	% Unused
https://pay.google.com/gp/p/js/pay.js	JS (coarse)	79 146	57 517	72.7 %
https://joel.tools/merch/gpay.js	JS (coarse)	7 389	2 173	29.4 %
https://joel.tools/merch/merch.js	JS (coarse)	2 710	829	30.6 %
https://joel.tools/merch/style.css	CSS	659	263	39.9 %
https://joel.tools/merch/	JS	23	0	0 %

A note at the bottom states "59.4 KB of 87.8 KB bytes are not used. (68%)".



Puppeteer



Code Coverage

- 👉 `page.coverage.startCSSCoverage()`
- 👉 `page.coverage.stopCSSCoverage()`
- 👉 `page.coverage.startJSCoverage()`
- 👉 `page.coverage.stopJSCoverage()`



Accessibility API



- 👉 `page.accessibility.snapshot()`
- 👉 returns “**Chrome Accessibility Tree**”
- 👉 foundation for all assistive technologies



```
const snapshot = await page.accessibility.snapshot();
console.log(snapshot);

{
  role: 'WebArea',
  name: 'Joel Merch',
  children: [
    { role: 'combobox', name: '', value: 'Shirt', children: [Array] },
    { role: 'heading', name: 'Shirt', level: 1 },
    { role: 'img', name: 'Shirt' },
    { role: 'text', name: 'This shirt is the height of fashion.' },
    { role: 'text', name: '$50' },
    { role: 'button', name: 'Buy with Google Pay', focused: true }
  ]
}
```

```
const snapshot = await page.accessibility.snapshot();
console.log(snapshot);

{
  role: 'WebArea',
  name: 'Joel Merch',
  children: [
    { role: 'combobox', name: '', value: 'Shirt', children: [Array] },
    { role: 'heading', name: 'Shirt', level: 1 },
    { role: 'img', name: 'Shirt' },
    { role: 'text', name: 'This shirt is the height of fashion.' },
    { role: 'text', name: '$50' },
    { role: 'button', name: 'Buy with Google Pay', focused: true }
  ]
}
```

```
const snapshot = await page.accessibility.snapshot();
console.log(snapshot);

{
  role: 'WebArea',
  name: 'Joel Merch',
  children: [
    { role: 'combobox', name: '', value: 'Shirt', children: [Array] },
    { role: 'heading', name: 'Shirt', level: 1 },
    { role: 'img', name: 'Shirt' },
    { role: 'text', name: 'This shirt is the height of fashion.' },
    { role: 'text', name: '$50' },
    { role: 'button', name: 'Buy with Google Pay', focused: true }
  ]
}
```

```
const snapshot = await page.accessibility.snapshot();
console.log(snapshot);

{
  role: 'WebArea',
  name: 'Joel Merch',
  children: [
    { role: 'combobox', name: '', value: 'Shirt', children: [Array] },
    { role: 'heading', name: 'Shirt', level: 1 },
    { role: 'img', name: 'Shirt' },
    { role: 'text', name: 'This shirt is the height of fashion.' },
    { role: 'text', name: '$50' },
    { role: 'button', name: 'Buy with Google Pay', focused: true }
  ]
}
```



```
it('should have a well named pay button', async () => {  
  const page = await context.newPage();  
  
});
```



```
it('should have a well named pay button', async () => {  
  const page = await context.newPage();  
  await page.goto('https://joel.tools/merch');  
});
```



```
it('should have a well named pay button', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  const button = await page.$('button');
}) ;
```

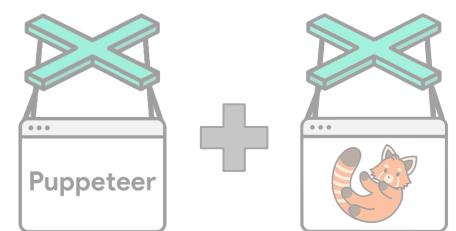
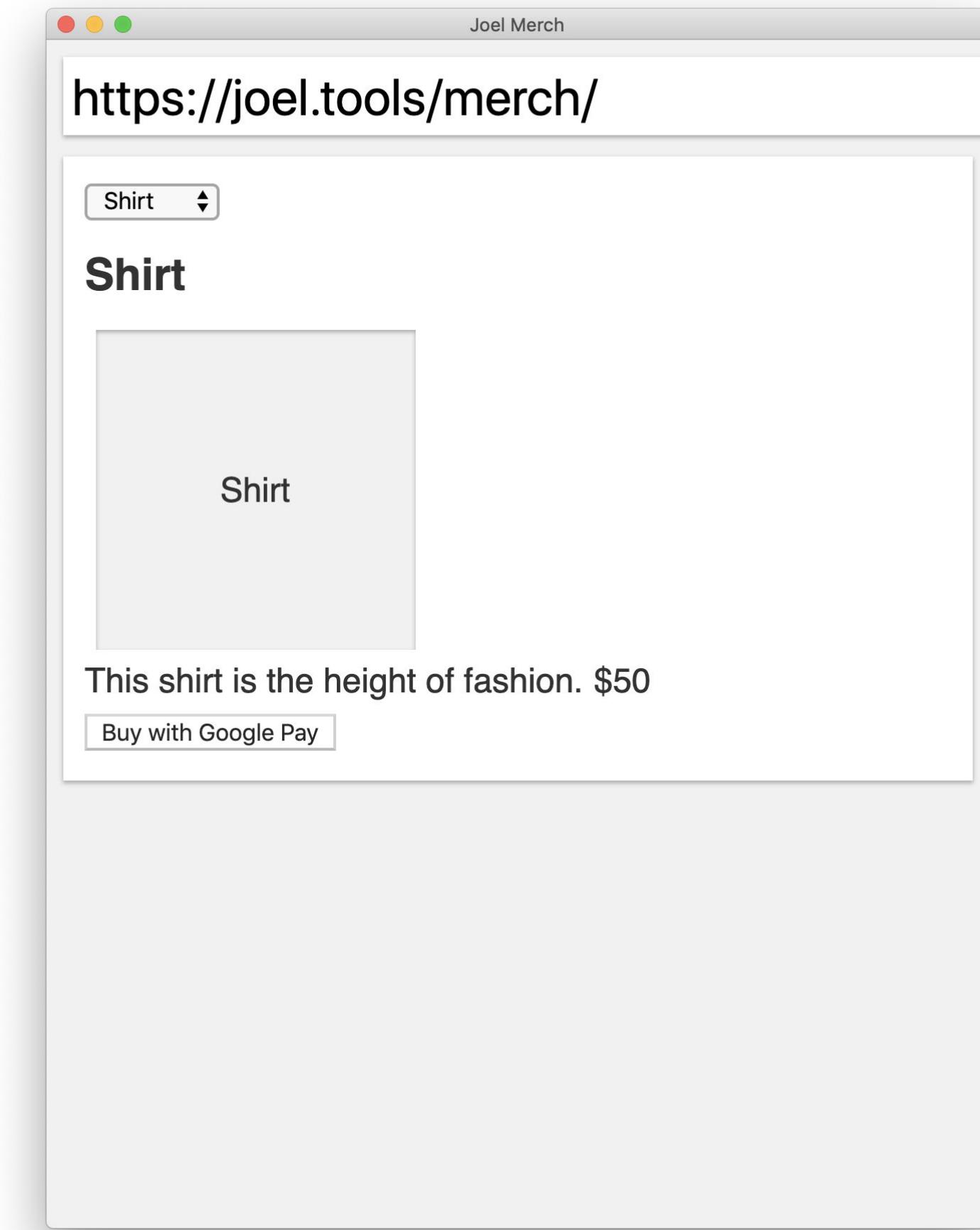
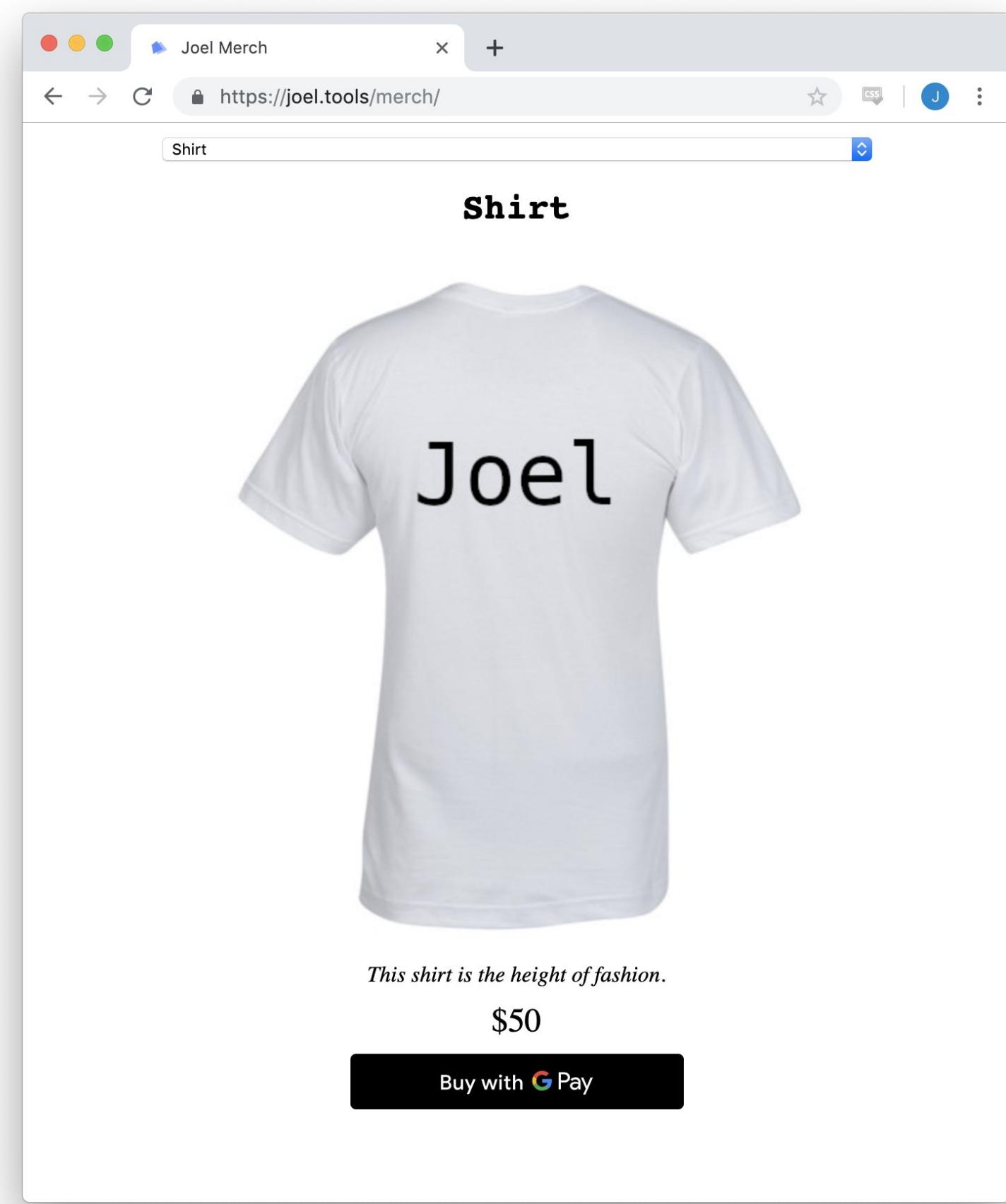


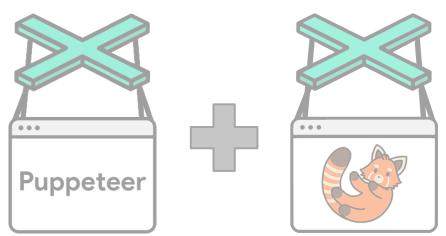
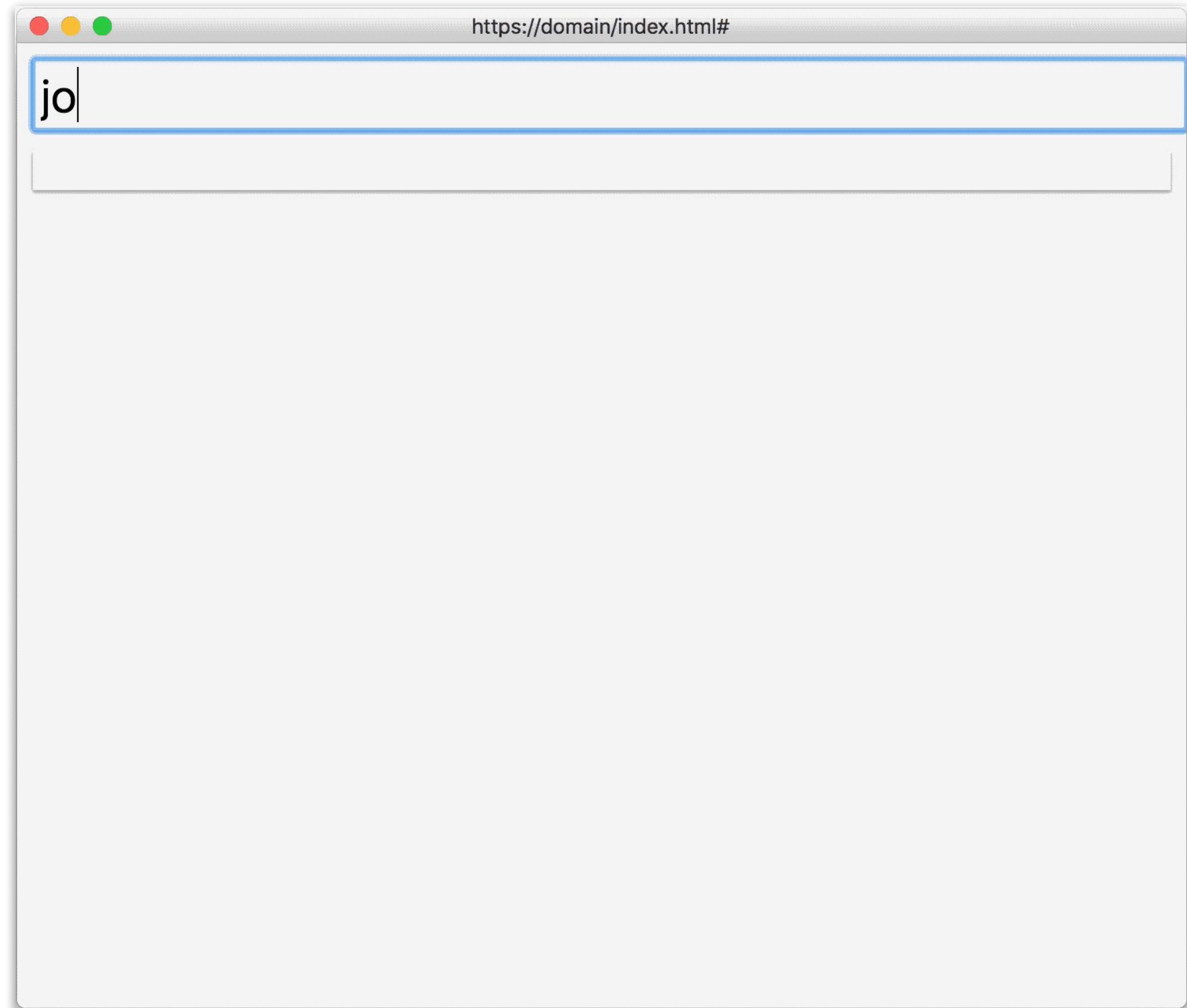
```
it('should have a well named pay button', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  const button = await page.$('button');
  const buttonAX = await page.accessibility.snapshot({
    root: button,
  });
});
```



```
it('should have a well named pay button', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  const button = await page.$('button');
  const buttonAX = await page.accessibility.snapshot({
    root: button,
  });
  expect(buttonAX.name).toBe('Buy with Google Pay');
});
```



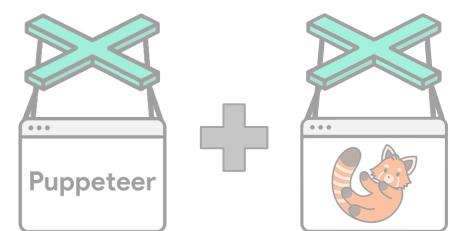


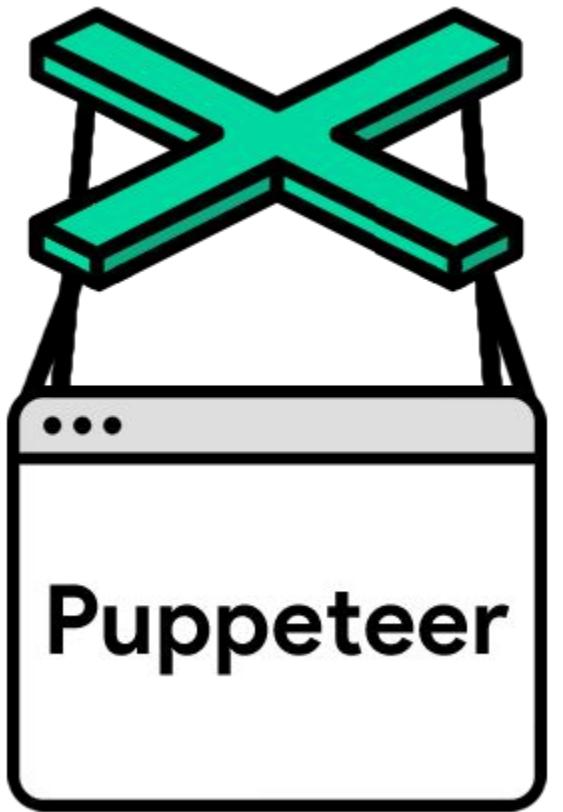




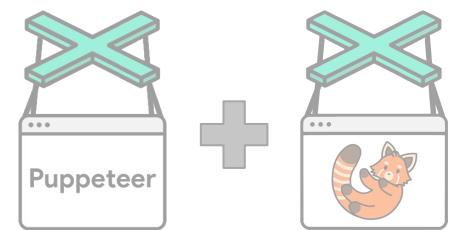
Accessibility Visualizer

👉 github.com/GoogleChromeLabs/html-a11y-renderer





github.com/GoogleChrome/puppeteer





Google™

NETFLIX



HOLY
JS!...
2019
PITER



Community



👉 2700+ folks on puppeteer.slack.com

The screenshot shows the Slack interface for the '#general' channel in the 'puppeteer' workspace. The sidebar on the left lists channels like '# general', '# psa', and '# puppeteer-sharp', along with a list of direct message recipients. The main pane displays messages from users like Karabo Maimane, aslushnikov, and joelgriffith. The messages include a link to the v1.15.0 release of Puppeteer and a mention of a TypeError.

#general

general | puppeteer Slack

Jump to... All Threads Channels # general # psa # puppeteer-sharp + Add a channel Direct Messages Slackbot aslushnikov (you) abraham Bill Eric Bidelman Joel Einbinder joelgriffith Kenneth Auchenberg Lisa rodrigo

Karabo Maimane 5:19 AM Does anyone have an example of how to make use of puppeteer on an angular 7 app

Thursday, April 25th

Friday, April 26th

aslushnikov 12:47 AM [PSA] v1.15.0 is live: <https://github.com/GoogleChrome/puppeteer/releases/tag/v1.15.0>

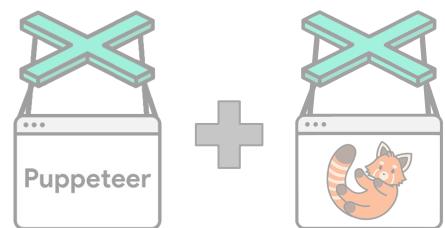
Posted in #psa | Apr 26th | View message

4 2

joelgriffith 8:15 AM TypeError: Cannot read property 'call' of null :thank you:

1

+ Message #general



Community

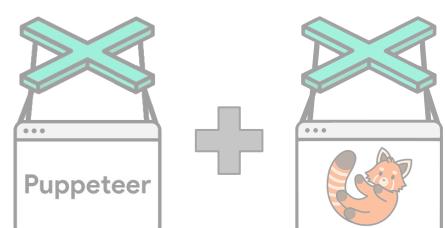


👉 100+ people contributed to api docs

The screenshot shows a GitHub repository page for `puppeteer / docs / api.md`. The page displays the commit history for version `v1.15.0`, which was tagged by `aslushnikov` on `b9f4a95` 10 days ago. The commit message is `chore: mark version v1.15.0 (#4346)`. Below the commit, it shows `110 contributors` with small profile pictures. At the bottom, there are links for `Raw`, `Blame`, `History`, and icons for `Copy file` and `Copy path`. The file has `3636 lines (2806 sloc)` and is `175 KB` in size.

Puppeteer API v1.15.0

- Interactive Documentation: <https://pptr.dev>
- API Translations: [中文|Chinese](#)
- Troubleshooting: [troubleshooting.md](#)
- Releases per Chromium Version:
 - Chromium 75.0.3738.0 - Puppeteer v1.14.0
 - Chromium 74.0.3723.0 - Puppeteer v1.13.0
 - Chromium 73.0.3679.0 - Puppeteer v1.12.2
 - Chromium 72.0.3582.0 - Puppeteer v1.11.0



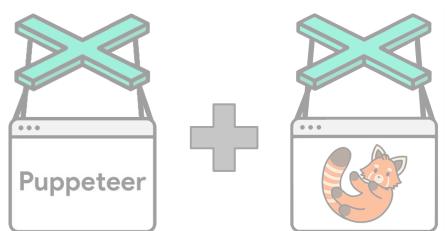
Community



👉 20% of patches to //lib are community contributions

The screenshot shows a GitHub pull request page for the `Add .content() method. Fixes #406.` pull request (#419) in the `GoogleChrome/puppeteer` repository. The pull request has been merged. The commit was made by `jeresig` on Aug 20, 2017, and merged by `aslushnikov` on Aug 21, 2017. The commit message states: "This adds a new method: `.content()` which is the counterpart of `.setContent()`, retrieving the HTML contents of the page, including the doctype (if there is one)." The pull request has 11 conversations, 1 commit, 0 checks, and 3 files changed. The review status is +36 -2. The right sidebar shows the following details:

- Reviewers:** aslushnikov (checked)
- Assignees:** No one assigned
- Labels:** None yet
- Projects:** None yet
- Milestone:** No milestone
- Participants:** 5 participants (including jeresig, aslushnikov, and others)





```
ai@blackjack:~/Dev/my-js-lib
~/Dev/my-js-lib
→ npx size-limit

Total time limit has exceeded
Time limit: 800 ms
Package size: 36.46 KB with all dependencies, minified and gzipped
Loading time: 730 ms on slow 3G
Running time: 200 ms on Snapdragon 410
Total time: 929 ms

Try to reduce size or increase limit in .size-limit.json

~/Dev/my-js-lib
→
```

github.com/ai/size-limit



Puppeteer





A screenshot of a GitHub repository page for 'penthouse'. The page title is 'penthouse' and it is described as a 'Critical Path CSS Generator'. It shows an 'npm package' badge with version 2.1.0, a green 'build passing' badge, and a badge showing 'downloads 91k/month'. The 'About' section explains that Penthouse generates critical CSS for websites. The 'Usage' section provides instructions to install the package using 'yarn add --dev penthouse' or 'npm install' if not using Yarn. The GitHub URL is https://github.com/pocketjoso/penthouse.

github.com/pocketjoso/penthouse



Puppeteer





API monitoring and Site Transaction Monitoring

checklyhq.com

Checkly

A better way to monitor your APIs and site click flows

Monitor the performance and correctness of your API endpoints & vital site transactions from a single, simple dashboard.

Start your free trial

STATUS	NAME	TYPE	LAST RESULTS	24H RATIO	AVG. RESP.	SCHED.
Passing	Add to shopping cart	commerce	100% (green bars)	100%	2.1 sec	5
Passing	Customer signup form	commerce	100% (green bars)	100%	2.68 sec	60
Failing	Customer signup form	commerce	0% (red bars)	0%	2.68 sec	60

checklyhq.com



Puppeteer





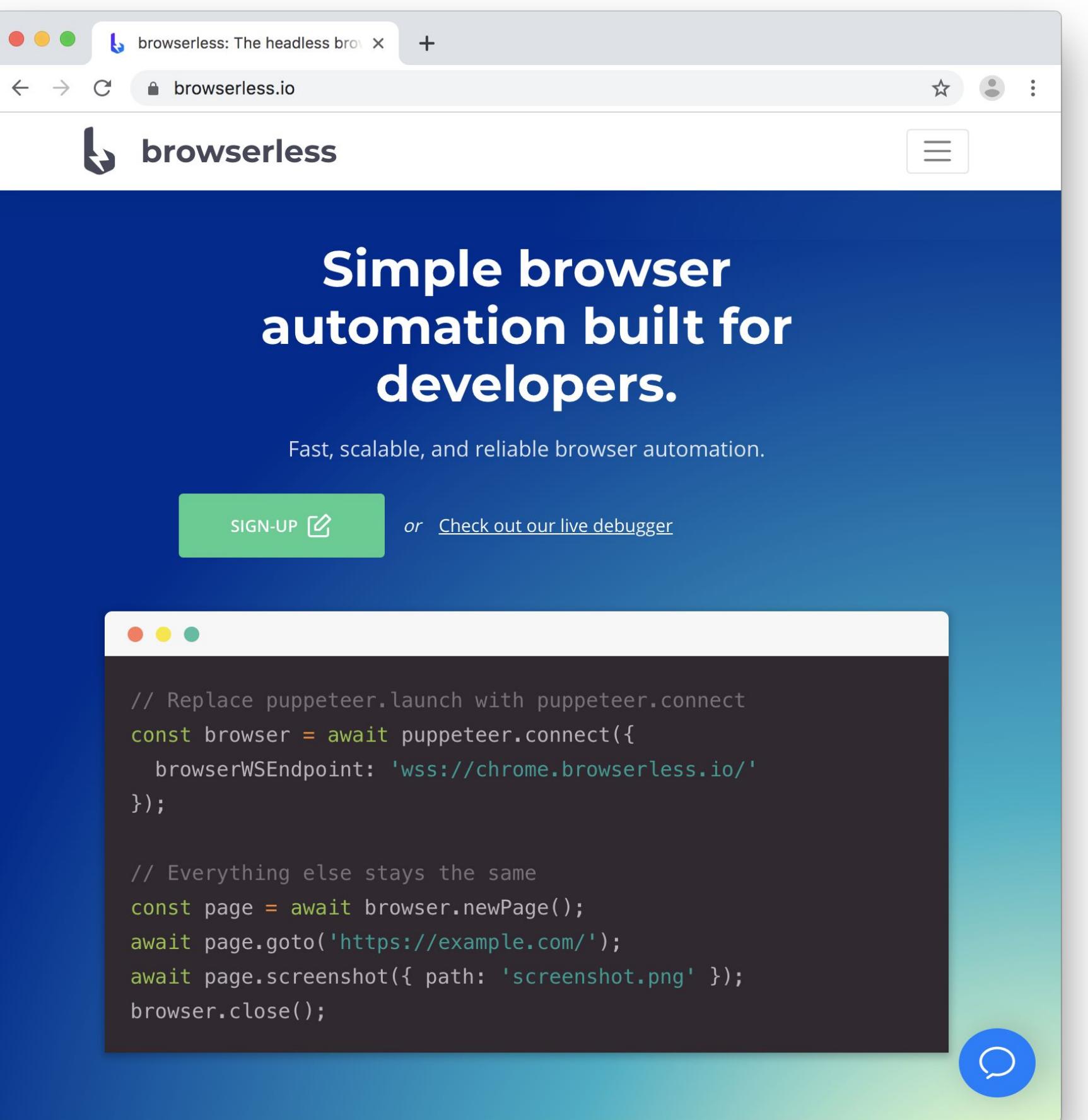
The screenshot shows the GitHub README page for the `puppeteer-recorder` repository. The page includes a green "build passing" badge, a "Buy me a coffee" button, and three steps illustrating the recorder's workflow:

- 1. Click record**: Shows the initial state of the recorder interface with a cactus icon and a "Record" button.
- 2. Track recorded events**: Shows the recorder interface after recording several events, with a list of tracked interactions (clicks and keydowns) and the generated Puppeteer script.
- 3. Generate a Puppeteer script**: Shows the final generated Puppeteer script, which includes launching a browser, navigating to a URL, and performing recorded interactions.

<https://github.com/checkly/puppeteer-recorder>

github.com/checkly/puppeteer-recorder





browserless.io



Puppeteer

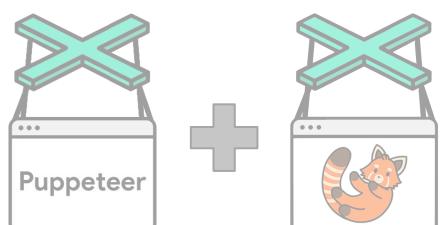


Community



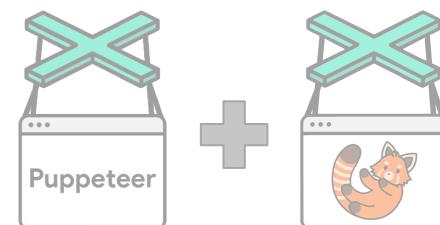
- 👉 33000+ project on GitHub
- 👉 1700+ packages on NPM

The screenshot shows the npmjs.com package page for 'puppeteer'. The page header includes the npm logo, a search bar, and links for 'New Package Monster', 'npm Enterprise', 'Products', 'Solutions', 'Resources', 'Docs', and 'Support'. A banner at the top right promotes 'npm's 2019 JavaScript ecosystem survey analysis'. The main content area shows the package 'puppeteer' version 1.15.0, which is public and was published 10 days ago. It has 8 dependencies and 1,715 dependents. Below this, a list of dependents is provided, including @ui-autotools/utils, vue-prerender-plugin, @xlaoyu/leetcode-submission-table, alive-pension, @vue/cli-test-utils, @daisy/ace-core, m65, snappydoo, headless-chrome-crawler, delibee, telecar, open-anais, js-sequence-diagrams-cli, herohero, puppeteer-spider, wtrans, pretty-google, authan, @coco-platform/render, and @jbmoelker/webshot-service. To the right of the dependents list are sections for 'install' (with a command line icon), 'weekly downloads' (showing a chart with 859,045), 'version' (1.15.0), 'license' (Apache-2.0), 'open issues' (348), and 'pull requests' (16). Navigation tabs include 'Readme', '8 Dependencies', '1,715 Dependents' (which is highlighted in purple), and '623 Versions'.



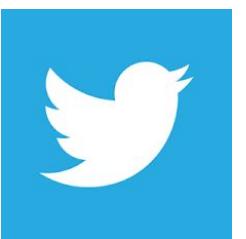
TL;DR

- 👉 **The NodeJS API for Chrome**
- 👉 **Experimental Firefox support**
- 👉 **Test the Web with Puppeteer**





Andrey Lushnikov, Google



@aslushnikov



aslushnikov@gmail.com

