

**HEISENBUG**

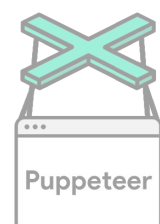
Heisenbug

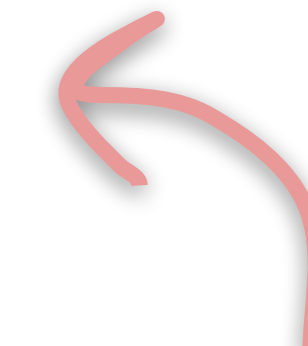
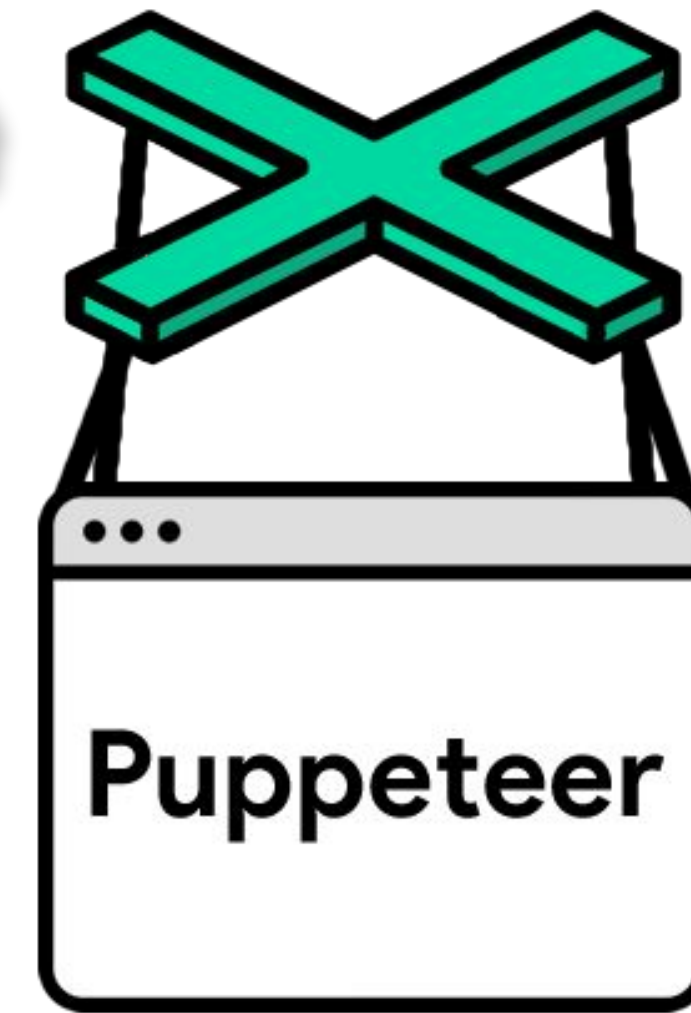
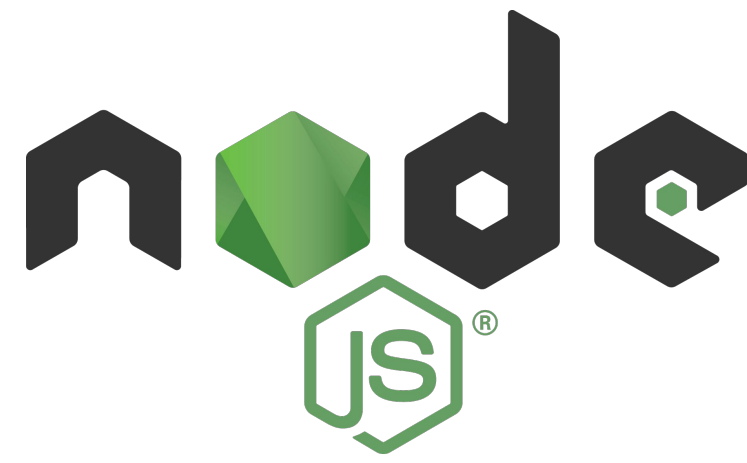
2019

# Modern Web Testing And Automation with Puppeteer

# Agenda

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



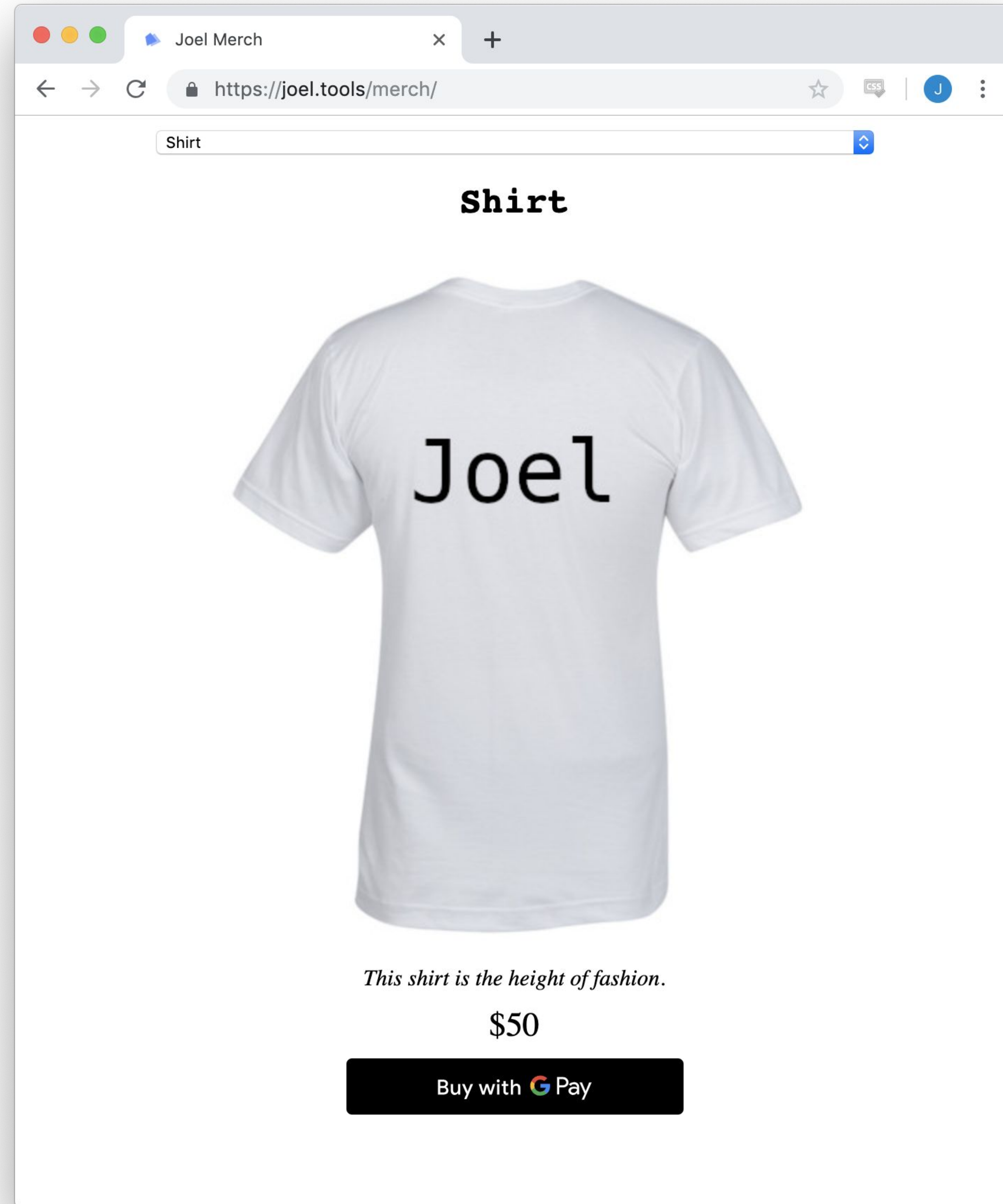


# What is Puppeteer?

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





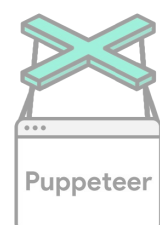


\$50

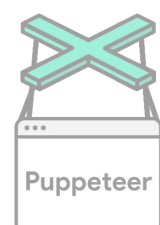


HEISENBUG

```
1. bash
lushnikov:~/joelshirt$
```



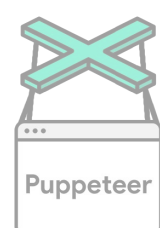
```
1. bash
lushnikov:~/joelshirt$ npm i puppeteer
```





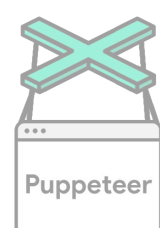
```
1. node
lushnikov:~/joelshirt$ npm i puppeteer
> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

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

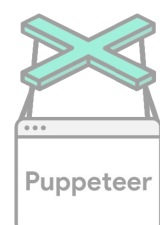
Downloading Chromium r650583 - 87.7 Mb [=====          ] 38% 4.2s
```





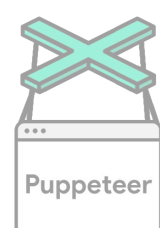
```
1. node
lushnikov:~/joelshirt$ npm i puppeteer
> puppeteer@1.15.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer
> node install.js

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



```
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
█
```



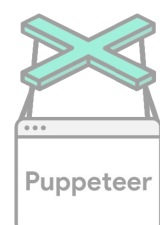


```
1. node
lushnikov:~/joelshirt$ npm i puppeteer

> 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
```

```
})();
```

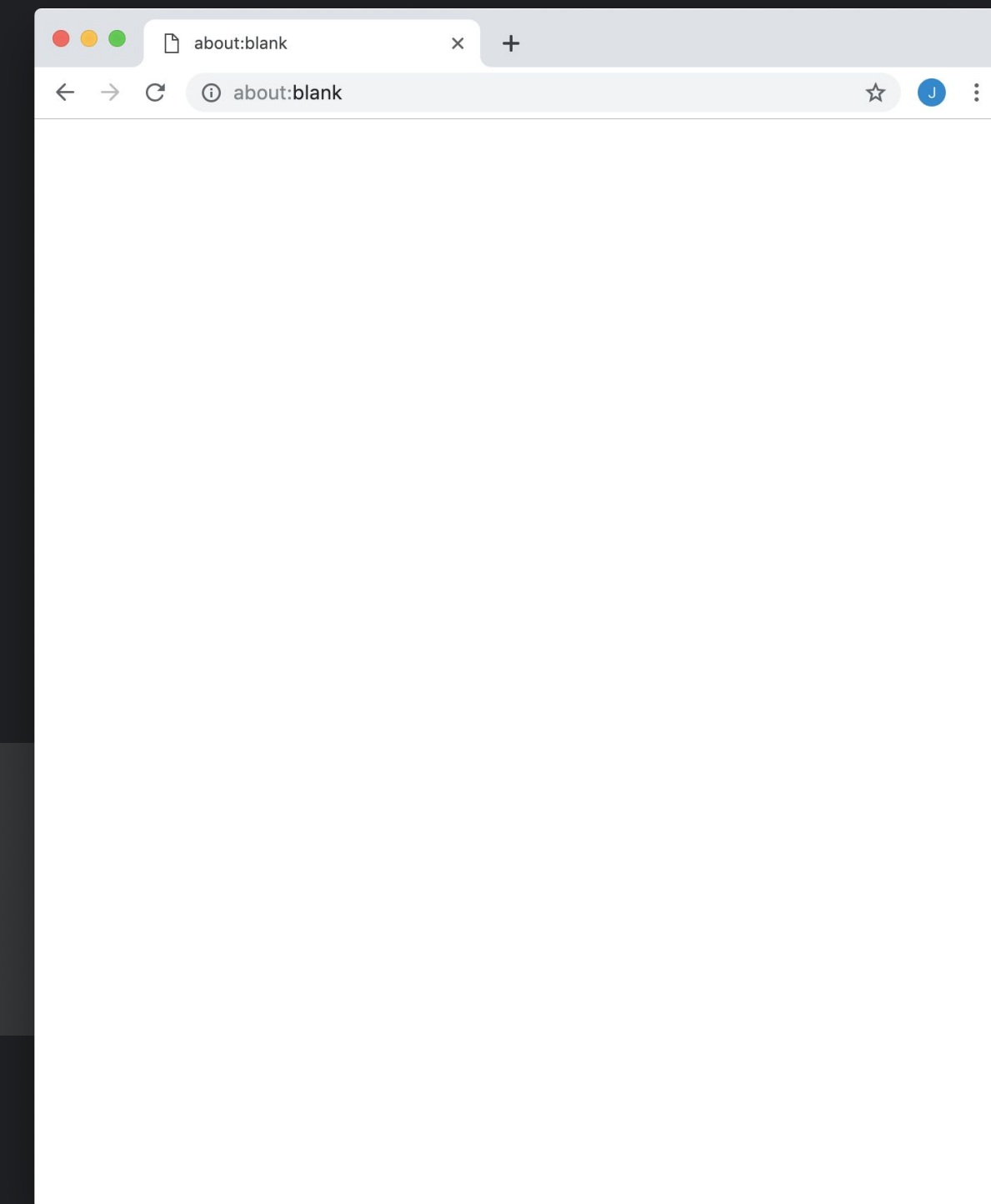
```
const puppeteer = require('puppeteer');
```

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

```
})();
```



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

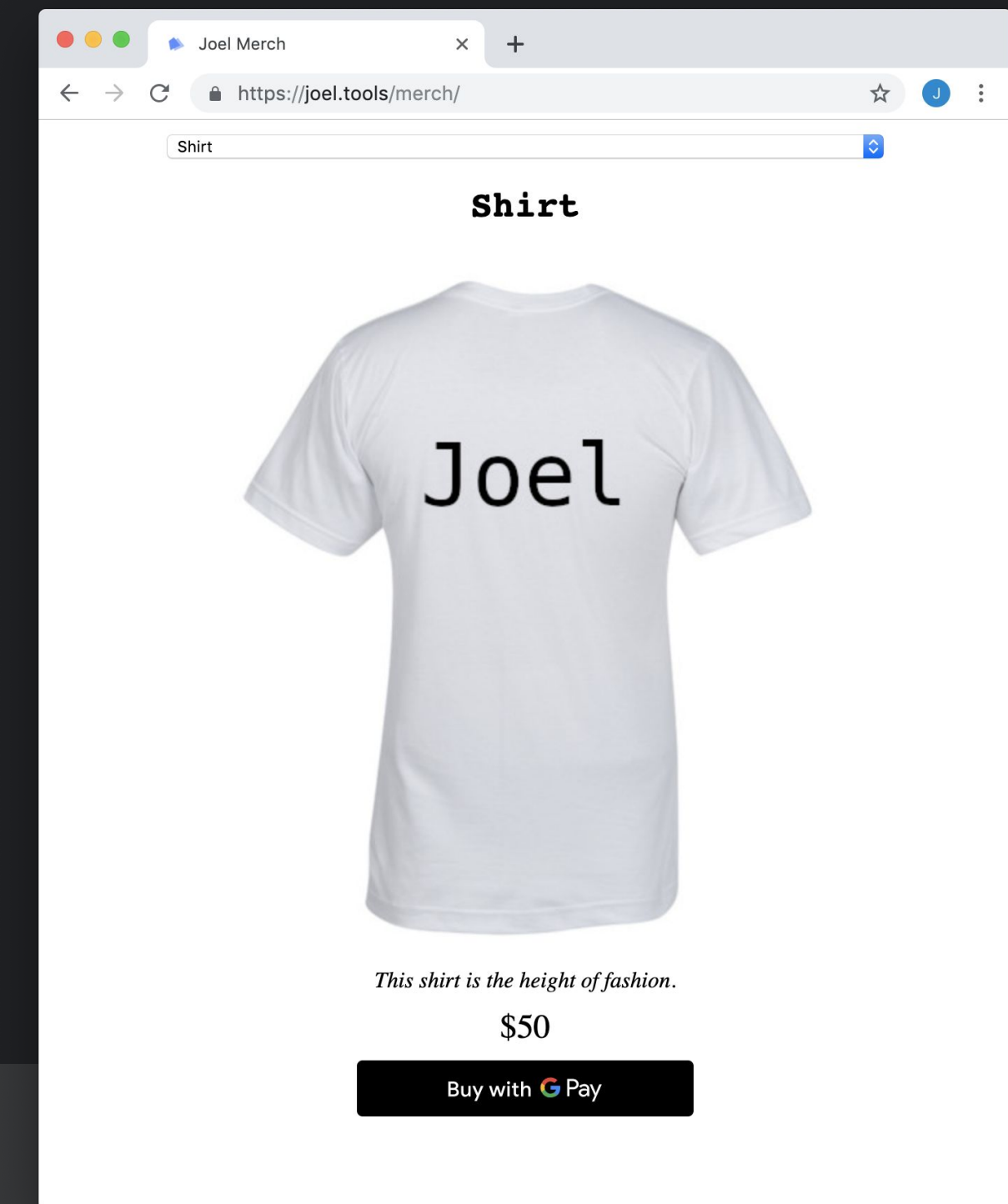
```
(async () => {
```

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

```
  const page = await browser.newPage();
```

```
  await page.goto('https://joel.tools/merch');
```

```
})();
```



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

```
(async () => {
```

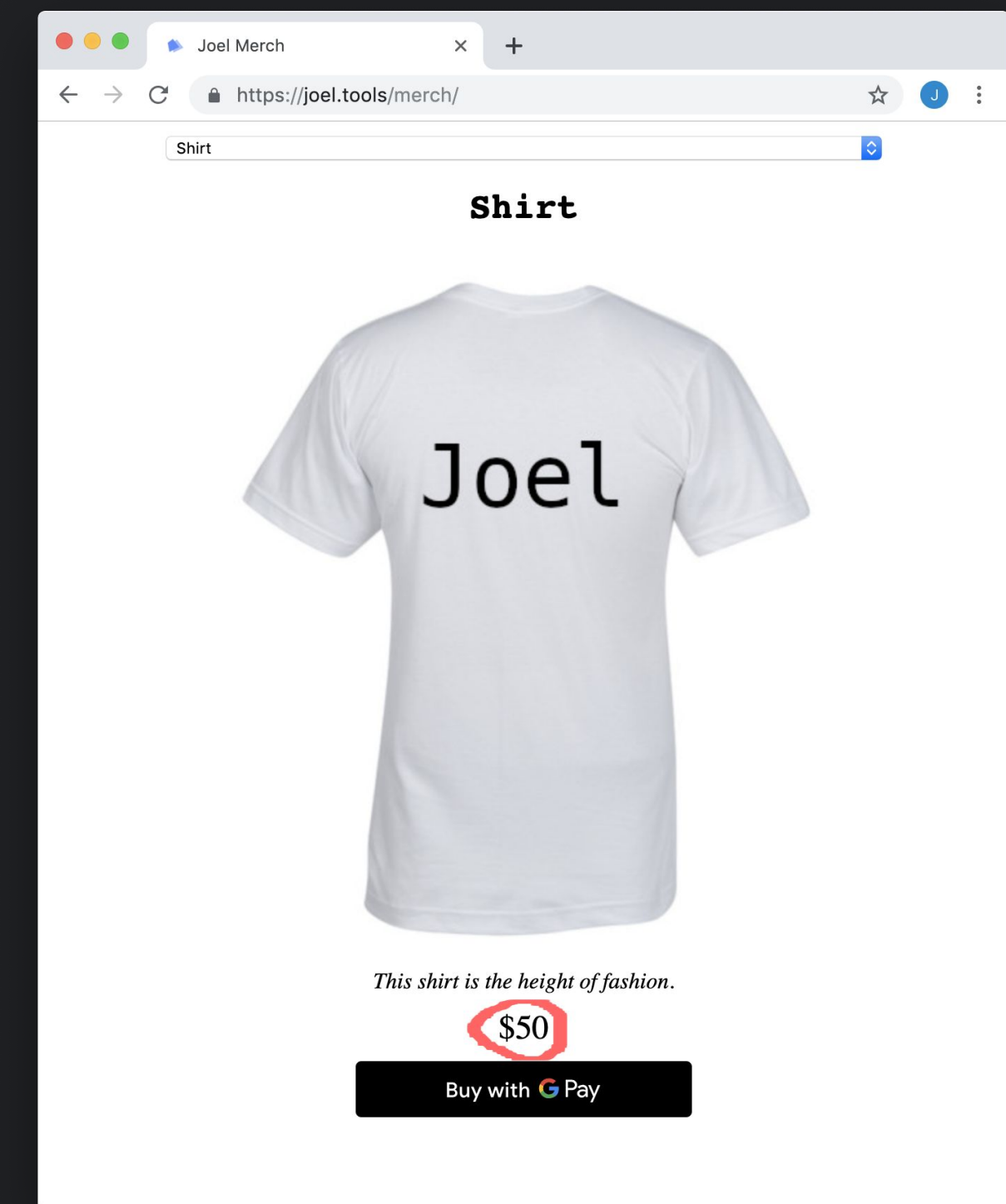
```
  const browser = await puppeteer.launch();
```

```
  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();
```

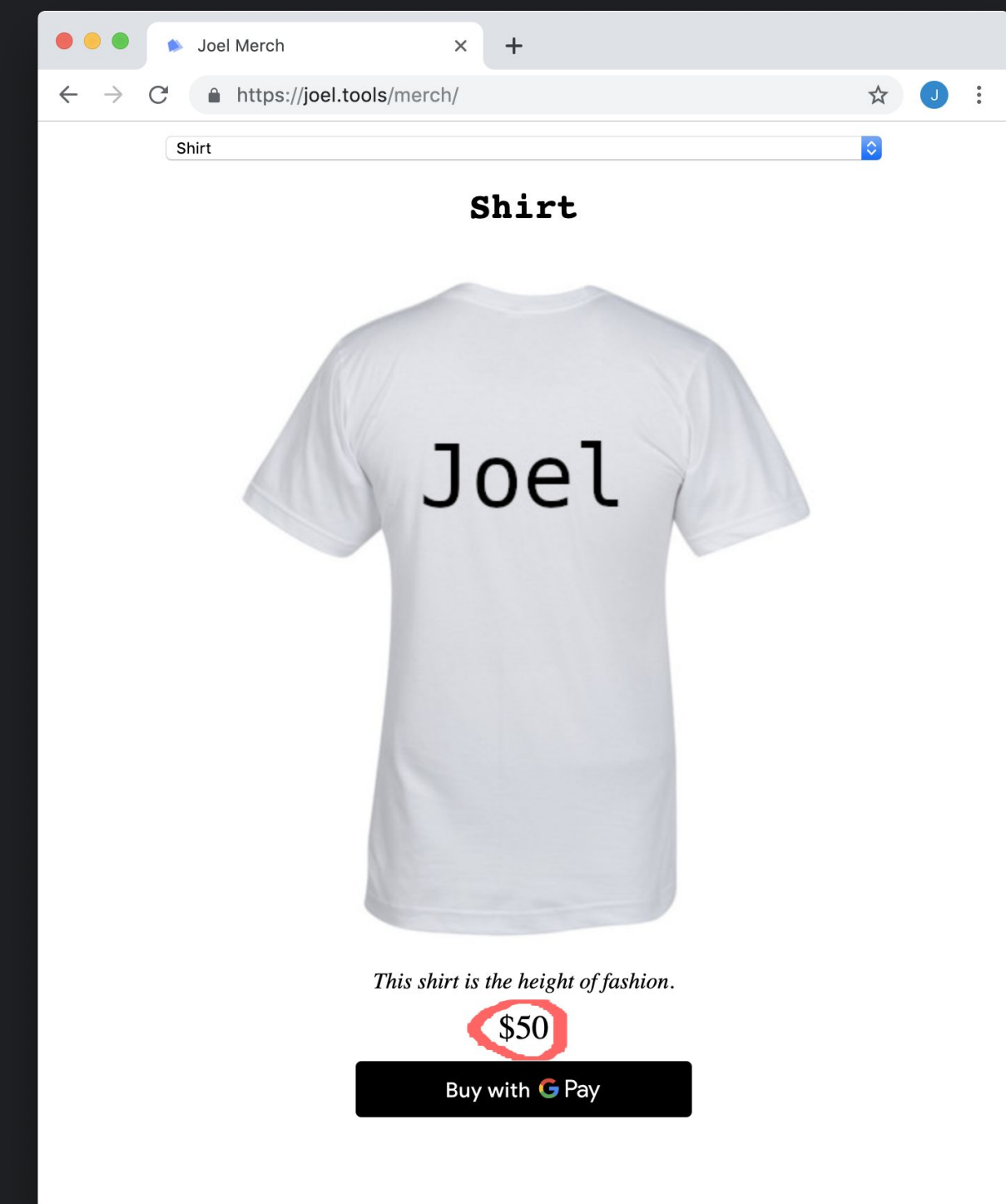
```
  const page = await browser.newPage();
```

```
  await page.goto('https://joel.tools/merch');
```

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

```
  console.log(price);
```

```
})();
```



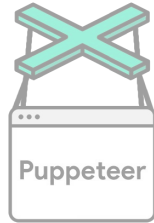
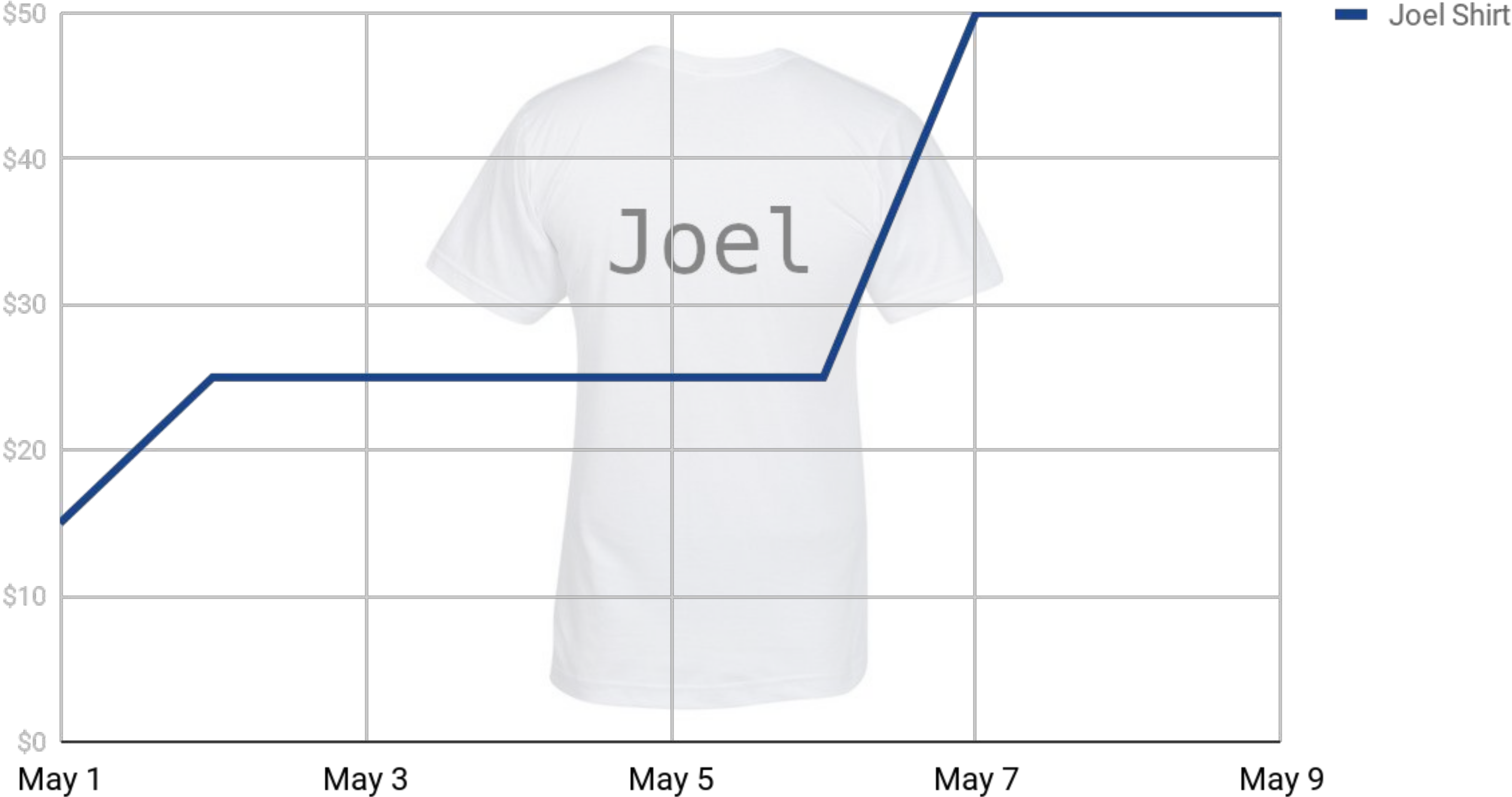
\$50

```
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

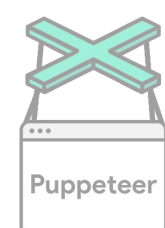
 experimental project

 `npm i puppeteer-firefox`

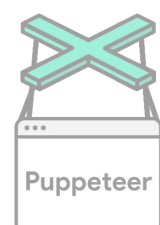
 supports **90%** of Puppeteer API



```
1. bash
lushnikov:~/joelshirt$
```

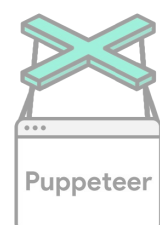


```
1. bash
lushnikov:~/joelshirt$ npm i puppeteer-firefox
```



```
1. node
lushnikov:~/joelshirt$ npm i puppeteer-firefox
> puppeteer-firefox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-firefox
> node install.js

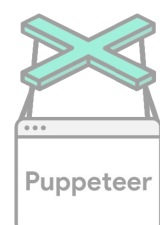
Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [ |          ] 5% 4.0s
```





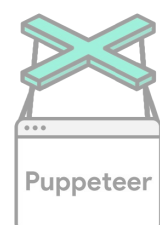
```
1. node
lushnikov:~/joelshirt$ npm i puppeteer-firefox
> puppeteer-firefox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-firefox
> node install.js

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



```
1. node
lushnikov:~/joelshirt$ npm i puppeteer-firefox
> puppeteer-firefox@0.5.0 install /Users/lushnikov/joelshirt/node_modules/puppeteer-firefox
> node install.js

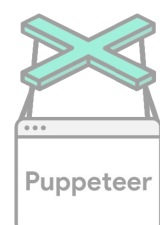
Downloading Firefox+Puppeteer 2f959d57 - 60.8 Mb [|||||] 59% 1.1s
```





```
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 [|||||||||||||||||] 100% 0.0s
█
```



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

lushnikov:~/joelshirt$
```





```
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');
```

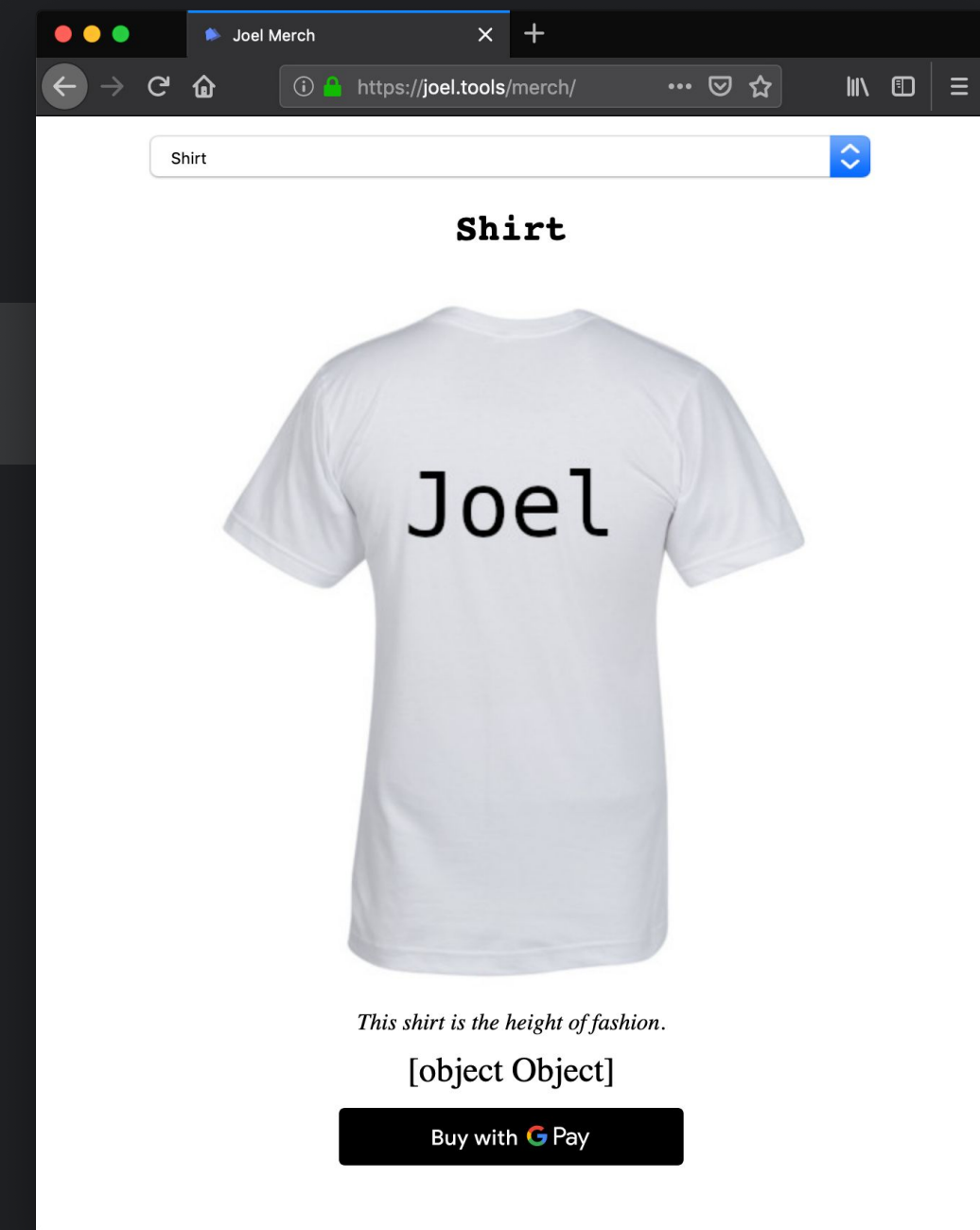
```
  const price = await page.$eval('.price', div => div.textContent);
```

```
  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  Pay

[object Object]

# Web Testing Challenges

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

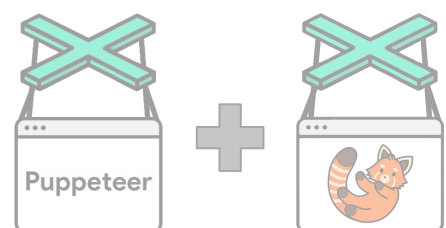


# New Web Testing

Powered by Puppeteer

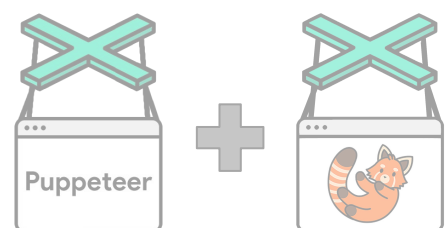
# *Simple* ~~Complicated Setup~~

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

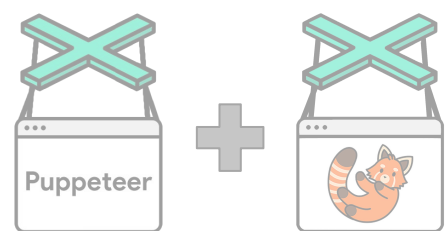
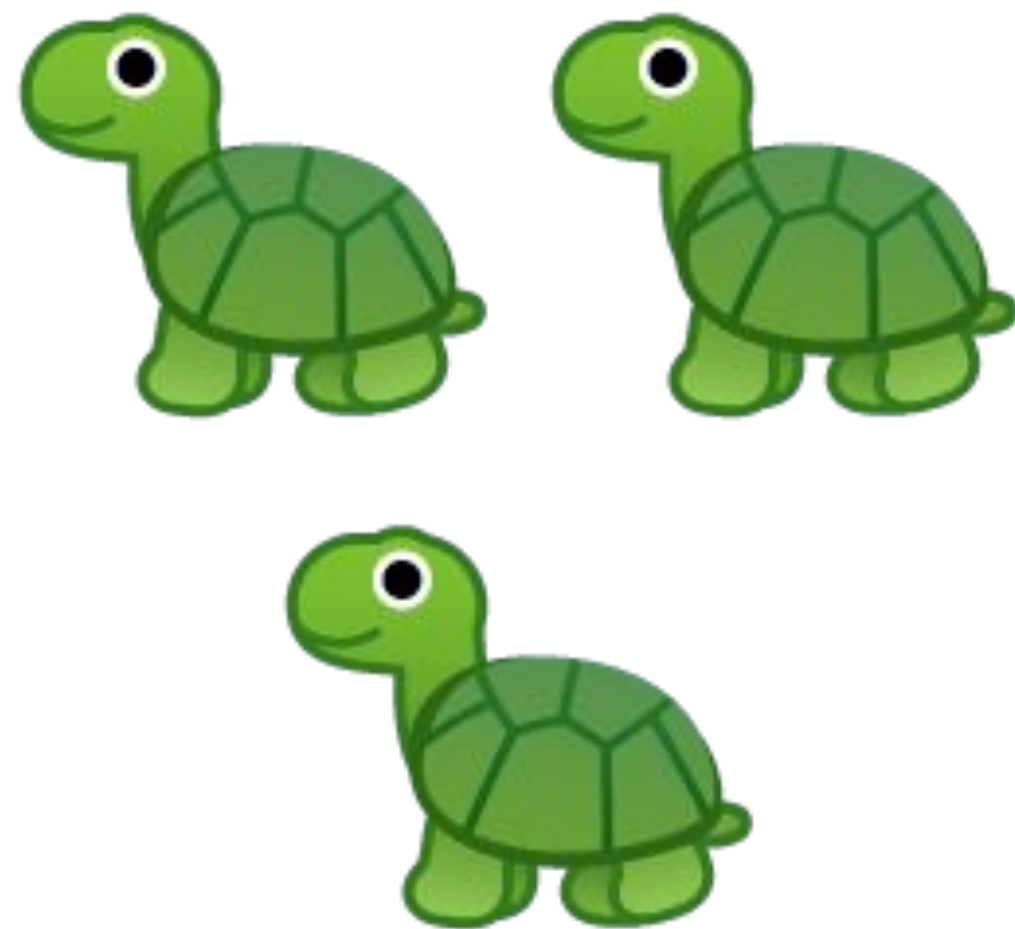


# Runs Everywhere

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




# Tests are Slow






Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay


Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay

Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt



*This shirt is the height of fashion.*

\$50

Buy with  Pay

Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay

Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt




*This shirt is the height of fashion.*

\$50

Buy with  Pay

Joel Merch x +  
https://joel.tools/merch/  
Shirt

### Shirt



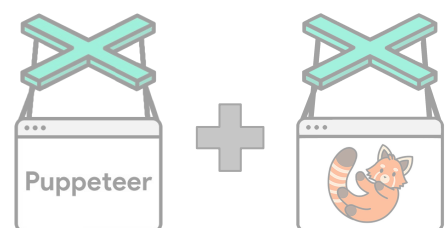
*This shirt is the height of fashion.*

\$50

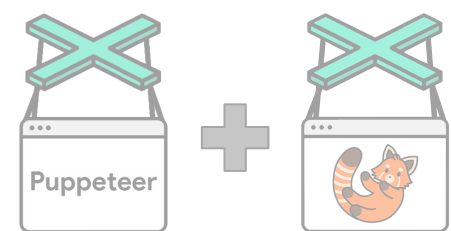
Buy with  Pay

# Tests are ~~Slow~~ *Fast*

 Browser Contexts



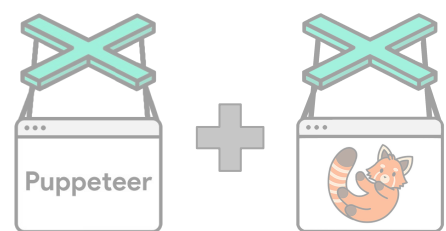
# 100x faster



```
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');

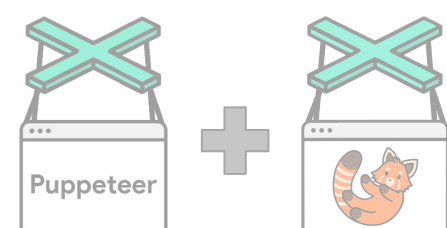
  assert(await page.$('.successful-purchase'));
});
```

```
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;

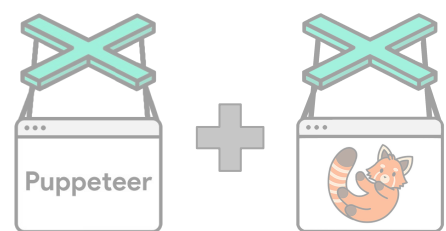
  assert(await page.$('.successful-purchase'));
});
```

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

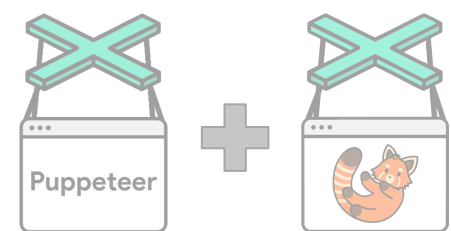
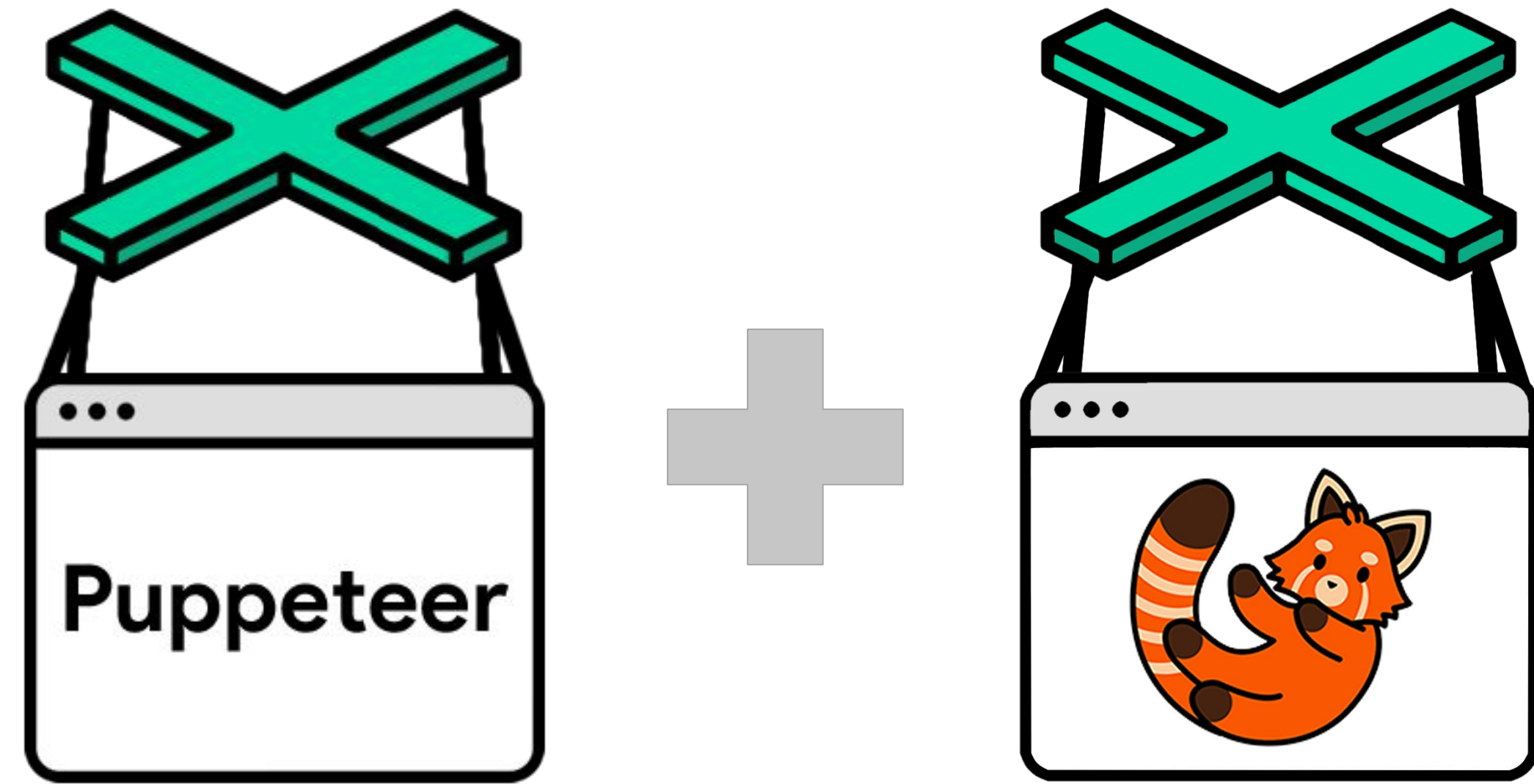
  await Promise.all([
    page.waitForResponse(res => res.url().endsWith('/pay')),
    page.click('button.gpay-button'),
  ]);
  assert(await page.$('.successful-purchase'));
});
```

# Recap

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



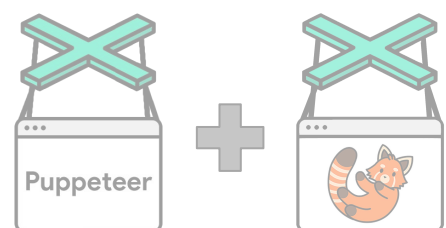




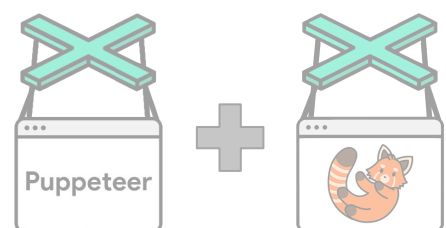
# Testing Fundamentals

# Web Testing Is...

- 👉 Navigation
- 👉 Evaluation
- 👉 Input Injection



# pptr.dev



page.evaluate

https://pptr.dev/#?product=Puppeteer&version=v1.15.0&show=api...

page.evaluate

**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"
```

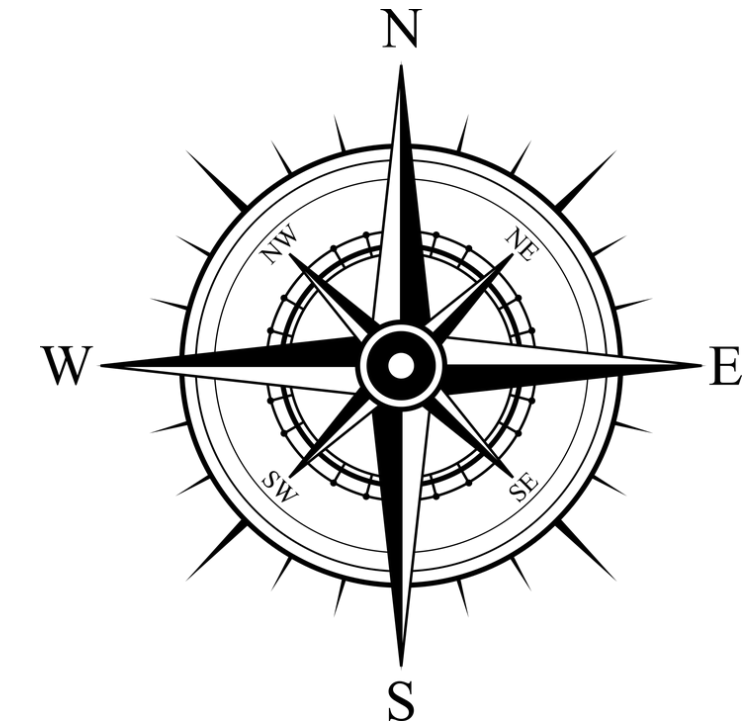
A string can also be passed in instead of a function:

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

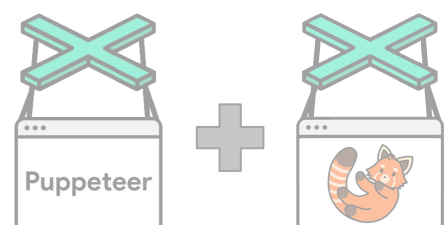
`ElementHandle` instances can be passed as arguments to the `page.evaluate`:



# Navigation

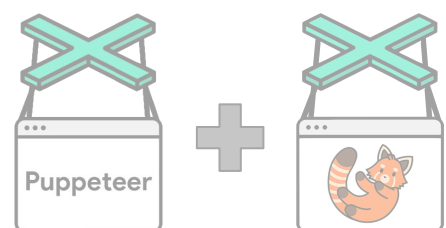
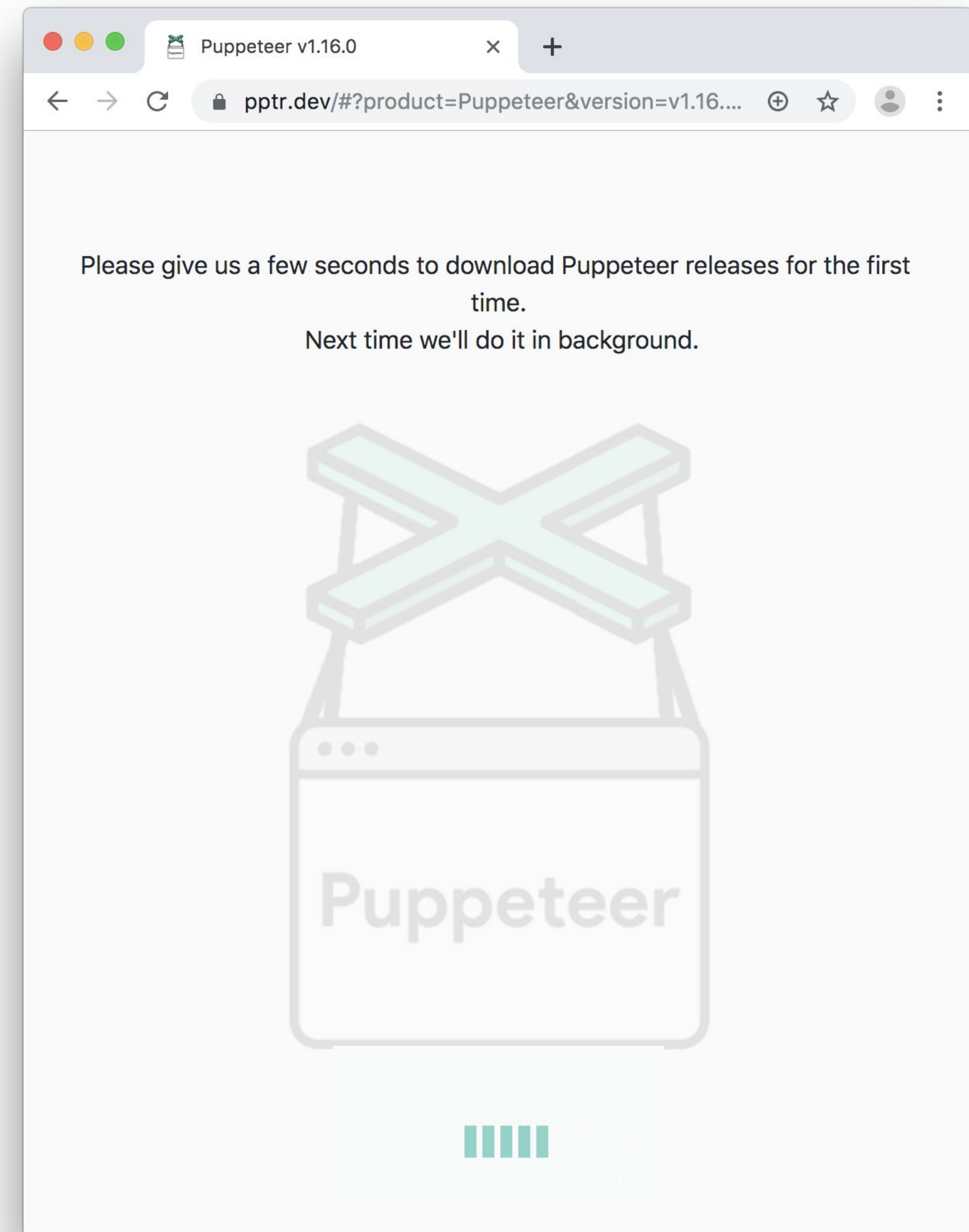


- 👉 `await page.goto( 'https://pptr.dev' );`
- 👉 resolves when page emits “load” event

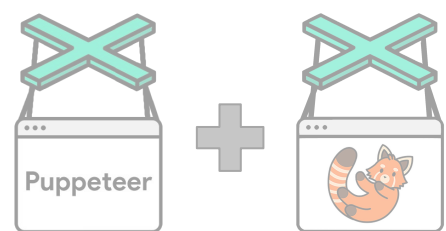
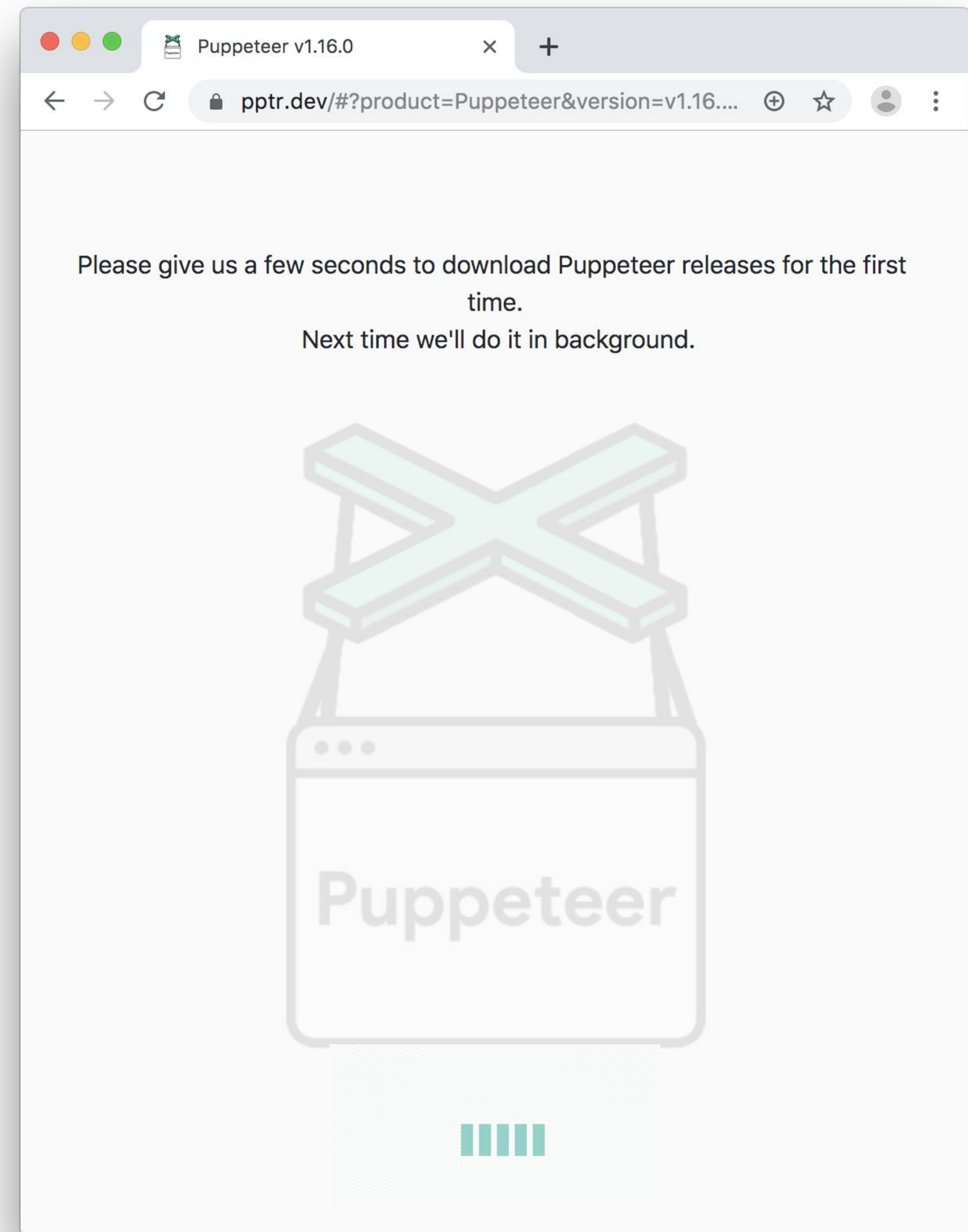




```
await page.goto('https://pptr.dev');
```

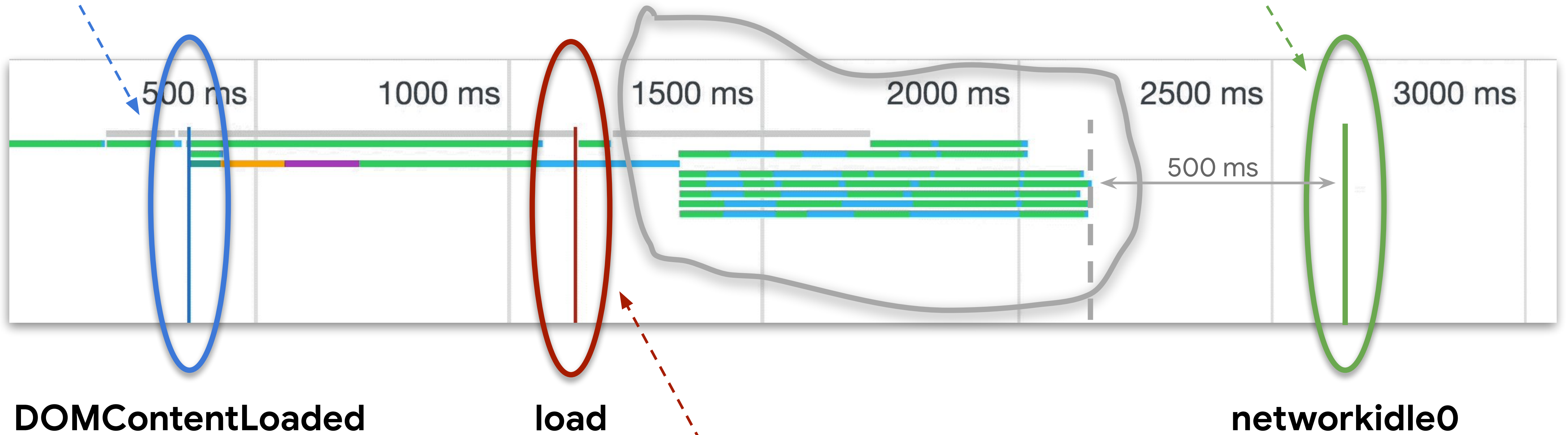


```
await page.goto('https://pptr.dev', {  
  waitUntil: '???'  
});
```

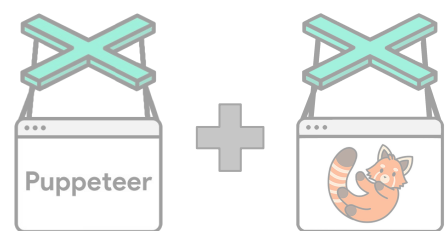


```
await page.goto('https://pptr.dev', {
  waitUntil: 'domcontentloaded'
});
```

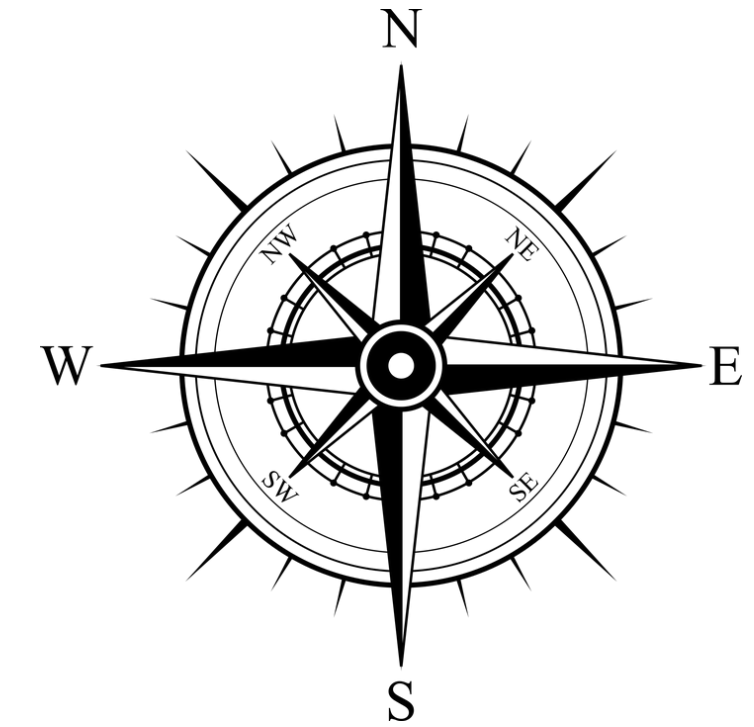
```
await page.goto('https://pptr.dev', {
  waitUntil: 'networkidle0'
});
```



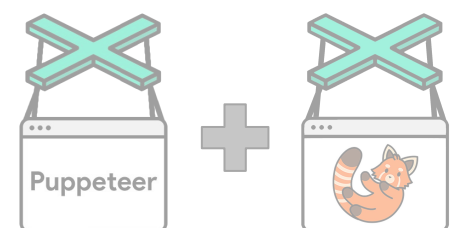
```
await page.goto('https://pptr.dev');
```



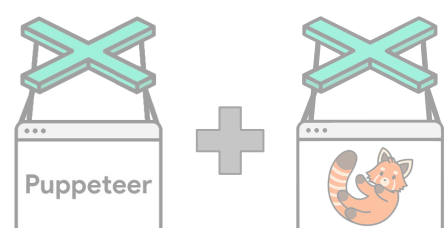
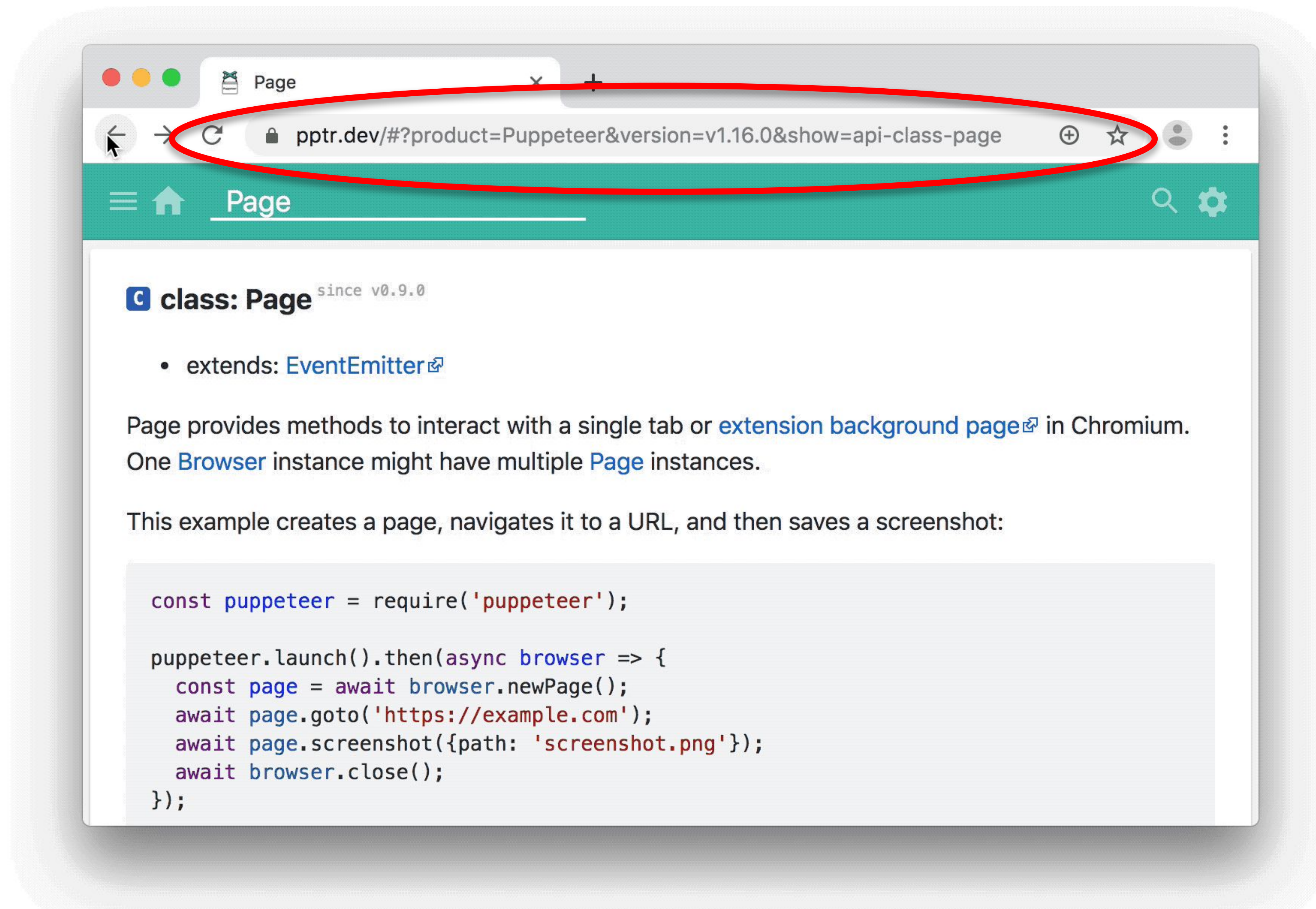
# Clicking Links



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

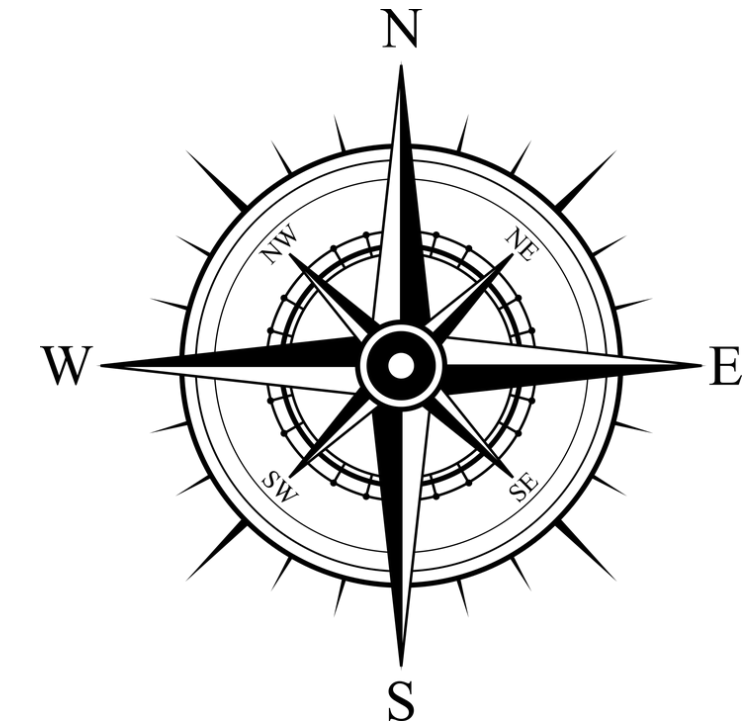




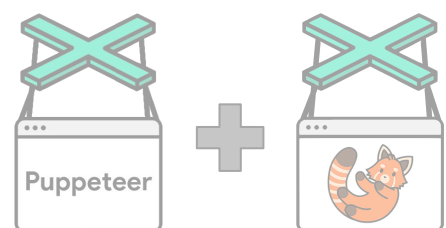




# Navigation Tracking



👉 `page.waitForNavigation();`



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

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

```
})();
```

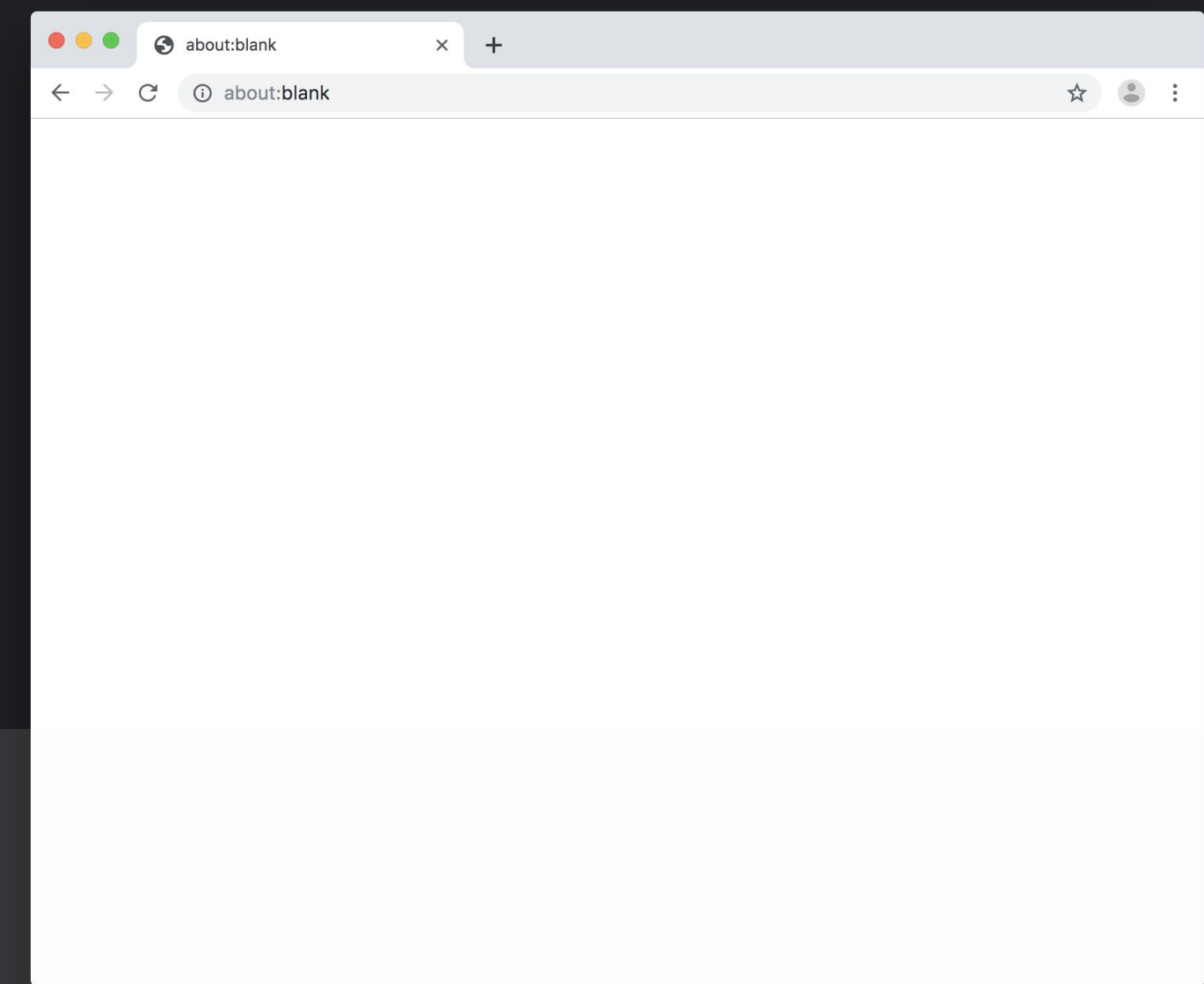
```
const puppeteer = require('puppeteer');
```

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

```
})();
```



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

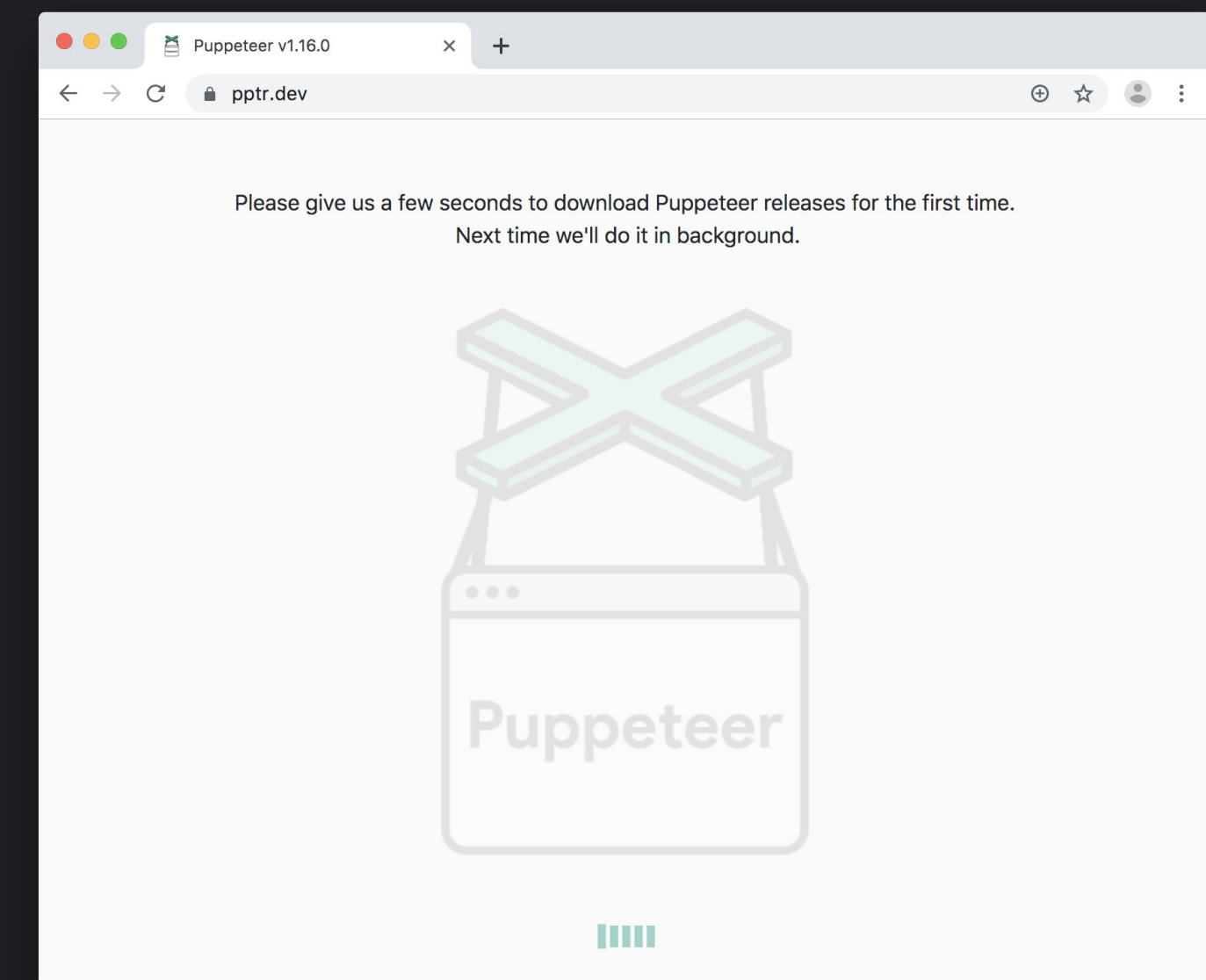
```
(async () => {
```

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

```
  const page = await browser.newPage();
```

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

```
})();
```



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

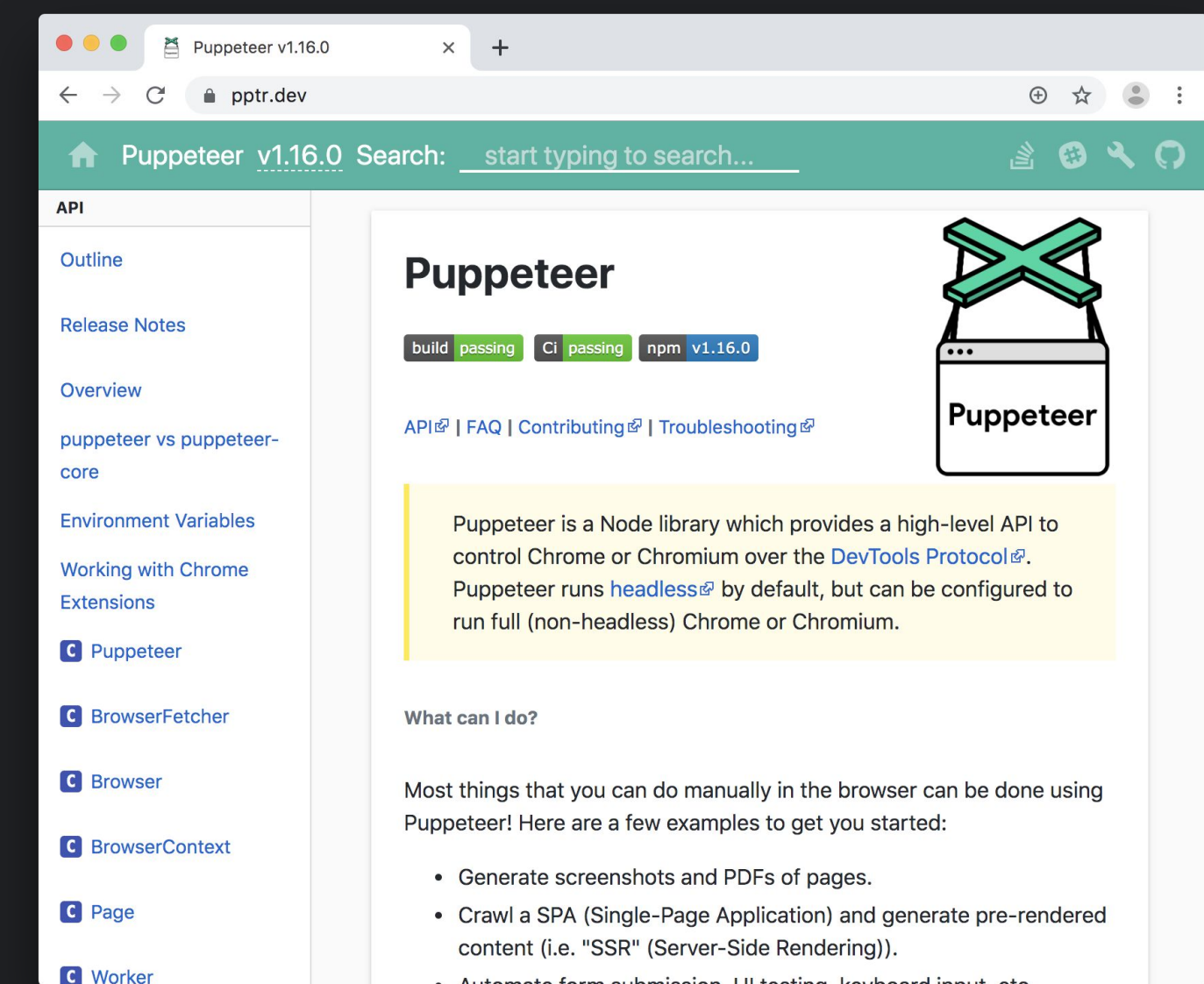
```
(async () => {
```

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

```
  const page = await browser.newPage();
```

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

```
})();
```



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

```
(async () => {
```

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

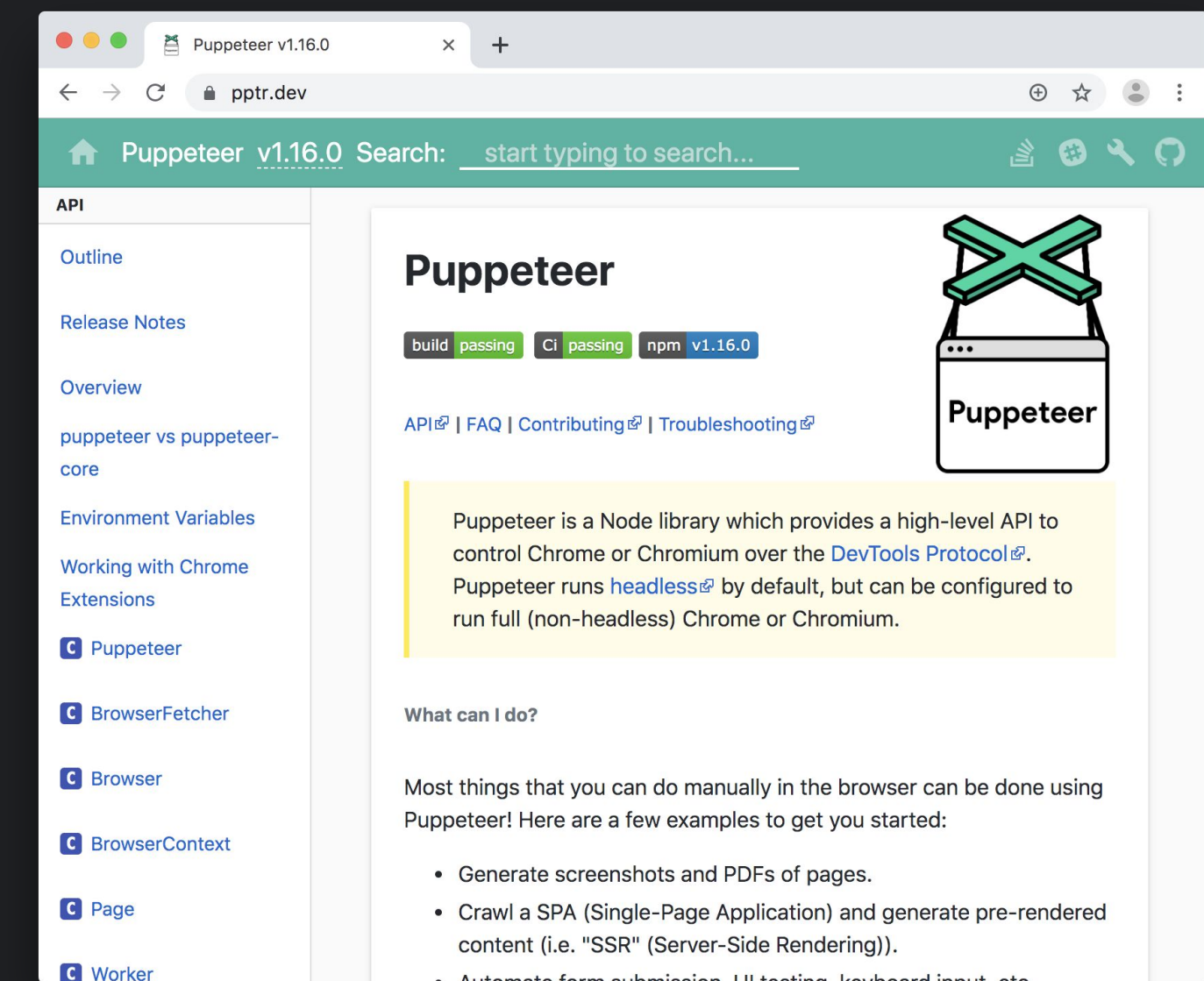
```
  const page = await browser.newPage();
```

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

```
  await Promise.all([
```

```
    ]);
```

```
})();
```





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

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

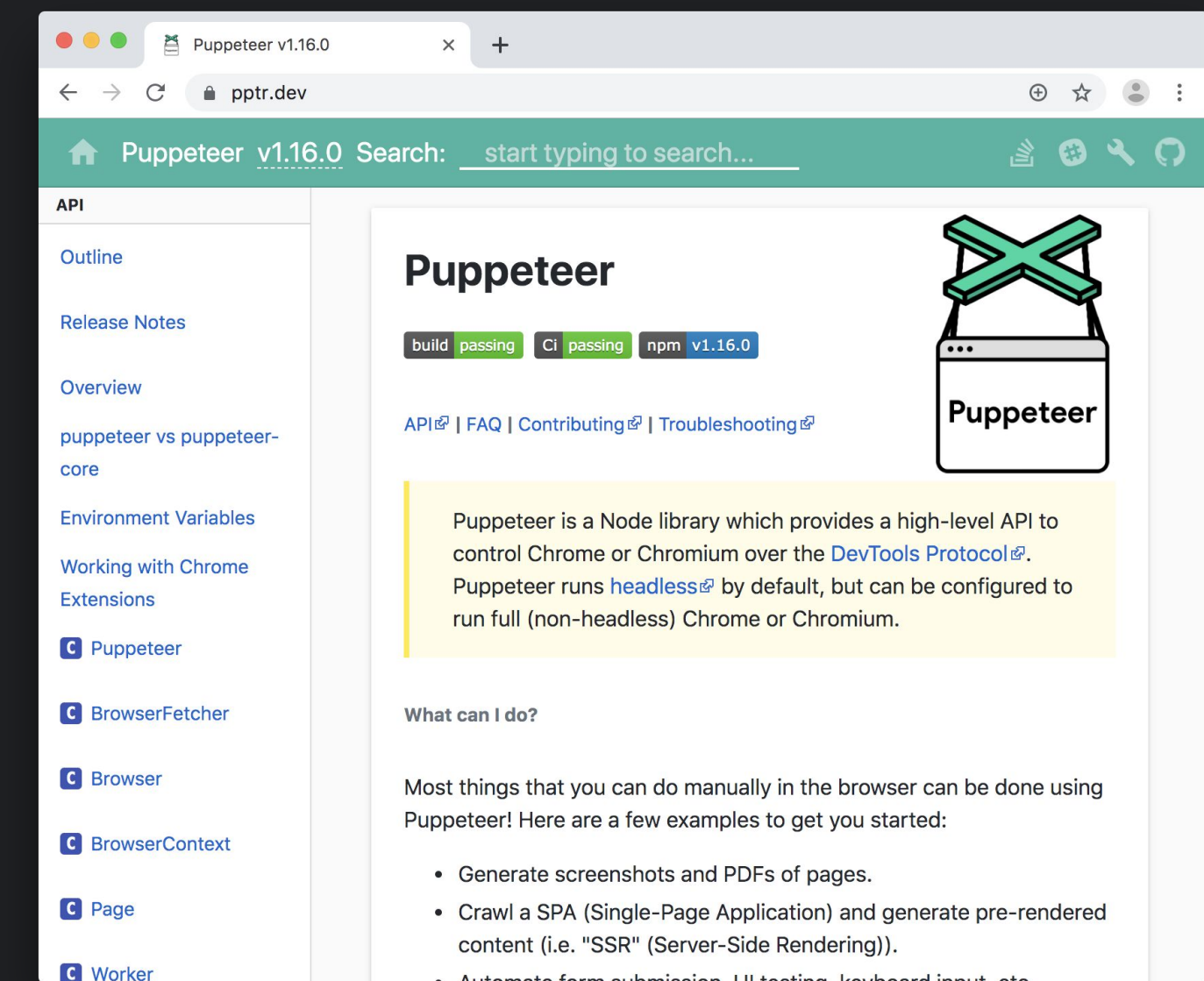
```
  await page.goto('https://pptr.dev', {waitUntil: 'networkidle0'});
```

```
  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'});
```

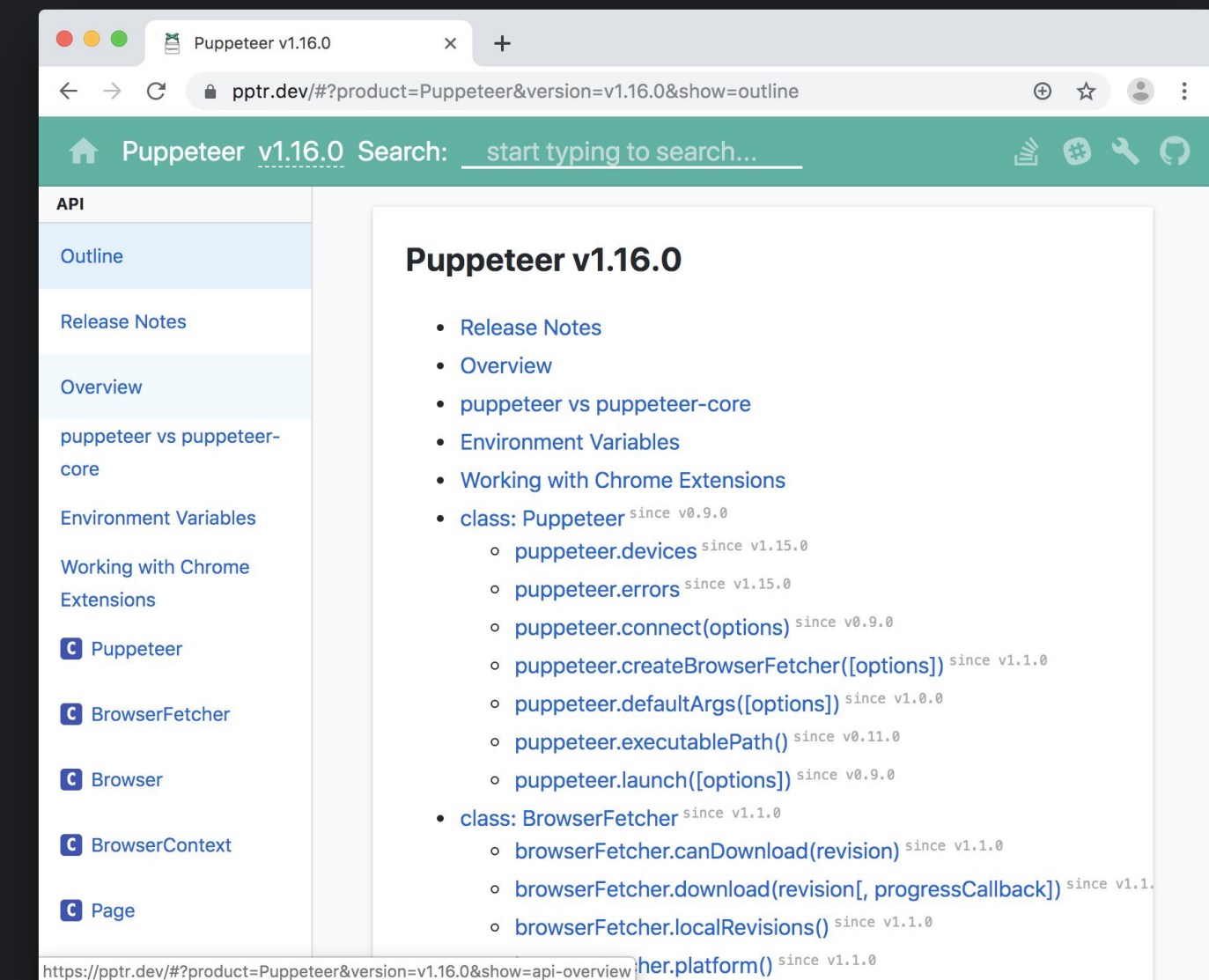
```
  await Promise.all([
```

```
    page.click('sidebar-component a'), // Documentation Outline
```

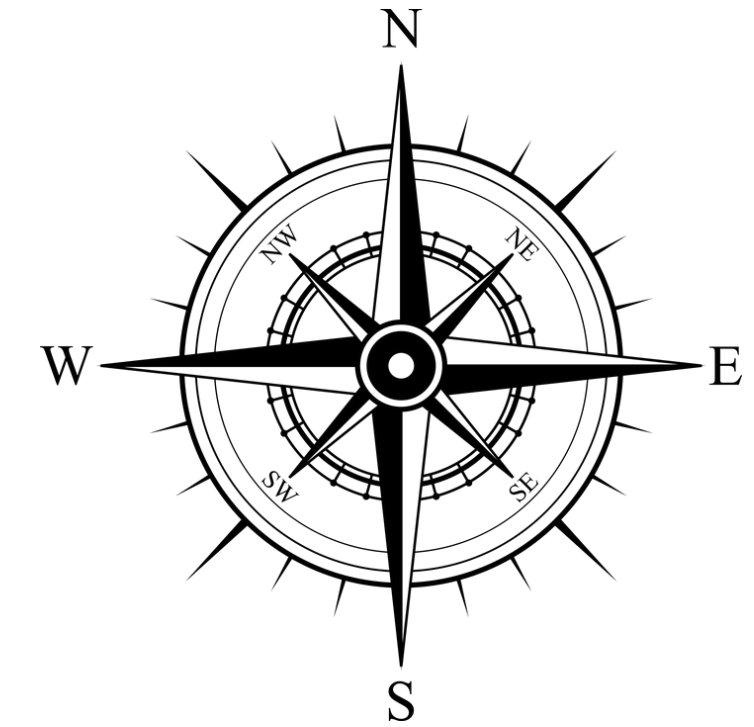
```
    page.waitForNavigation(),
```

```
  ]);
```

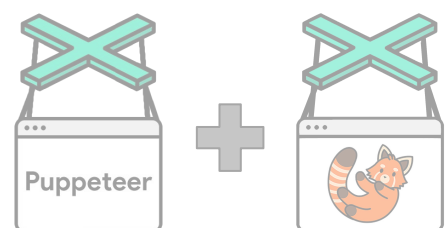
```
})();
```



# Navigation APIs

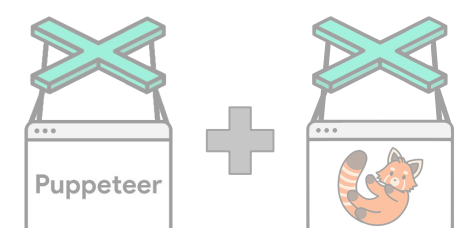


- 👉 `page.goto()`;
- 👉 `page.waitForNavigation()`;
- 👉 `page.goBack()`;
- 👉 `page.goForward()`;

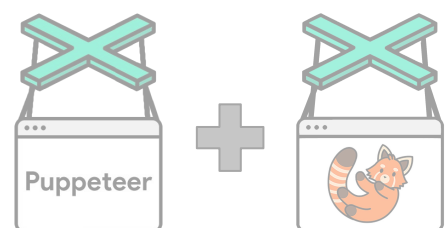
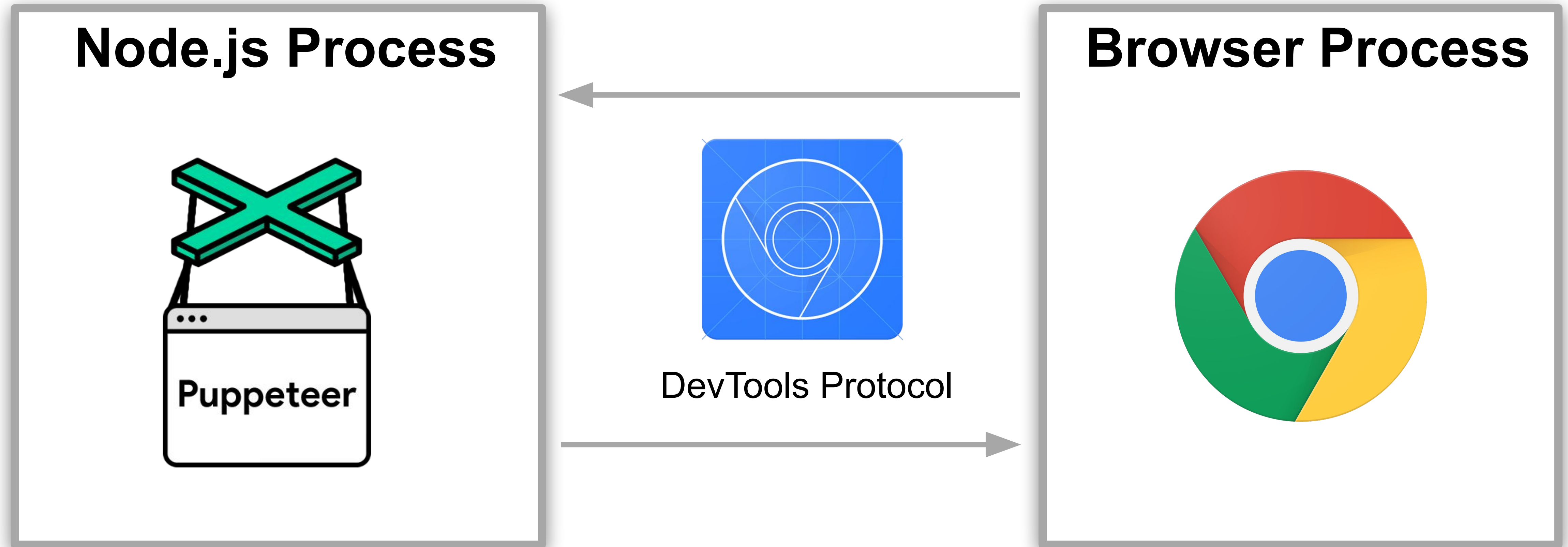


# Evaluation

- 👉 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;
  });
})();
```



*SERIALIZED!*



```
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!
  });
})();
```



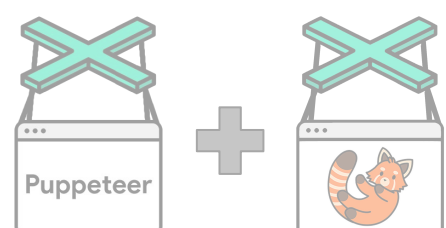
*SERIALIZED!*





# Test Isolation

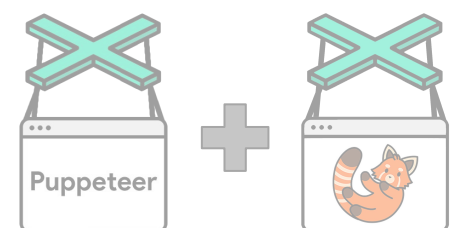
- 👉 Node.js **does not leak** into browser
- 👉 Website **doesn't know** it's been tested
- 👉 Websites **can't ship test-specific** code



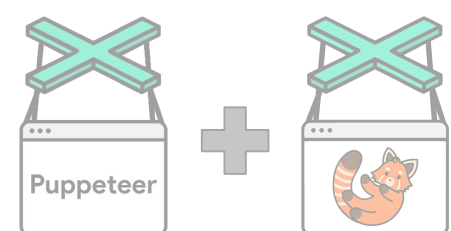
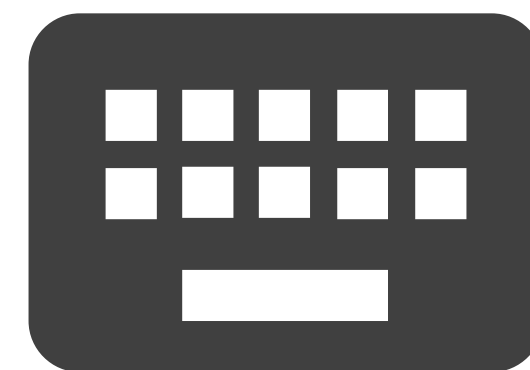


# Takeaway

👉 Testing with Puppeteer is **foolproof**

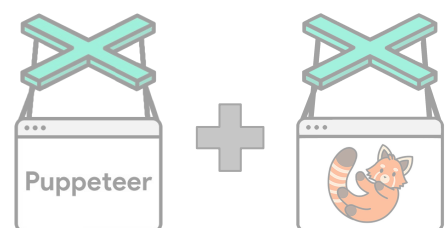


# Input Injection

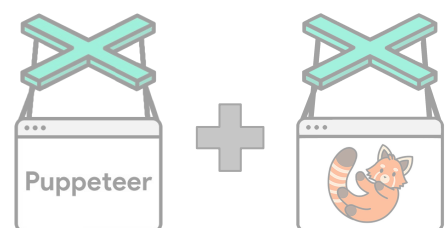
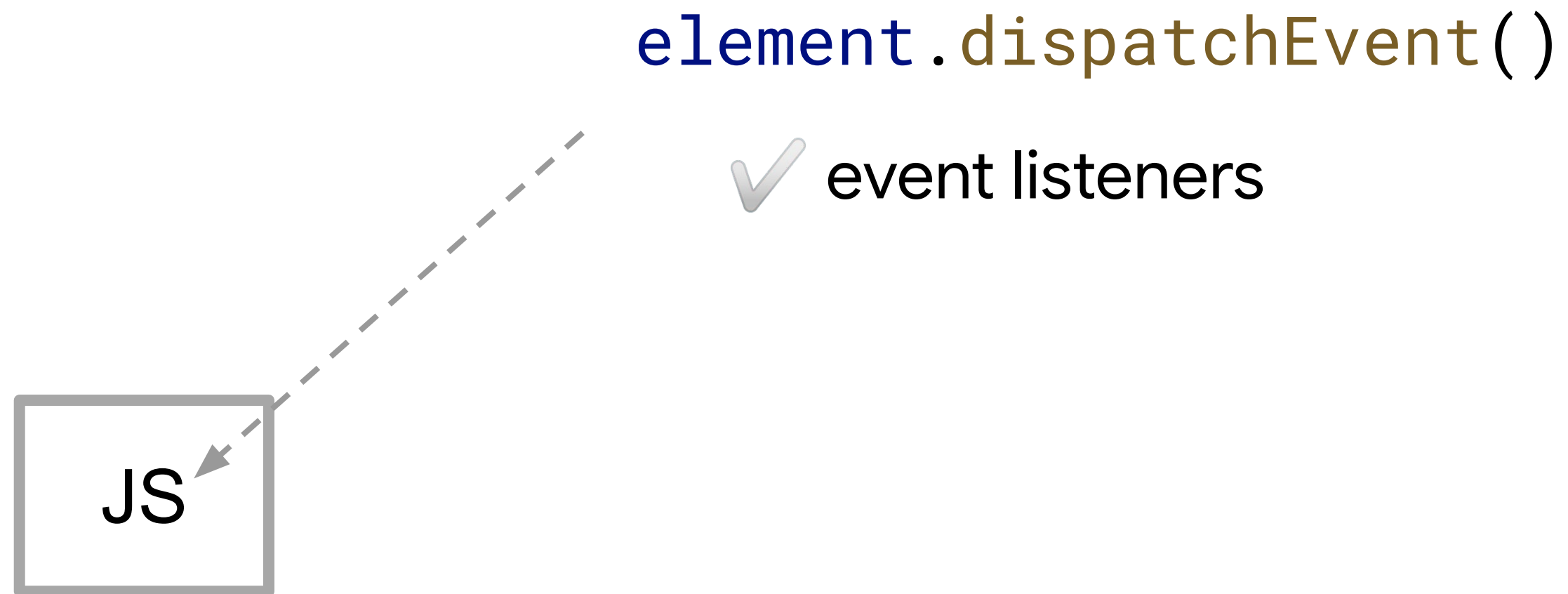


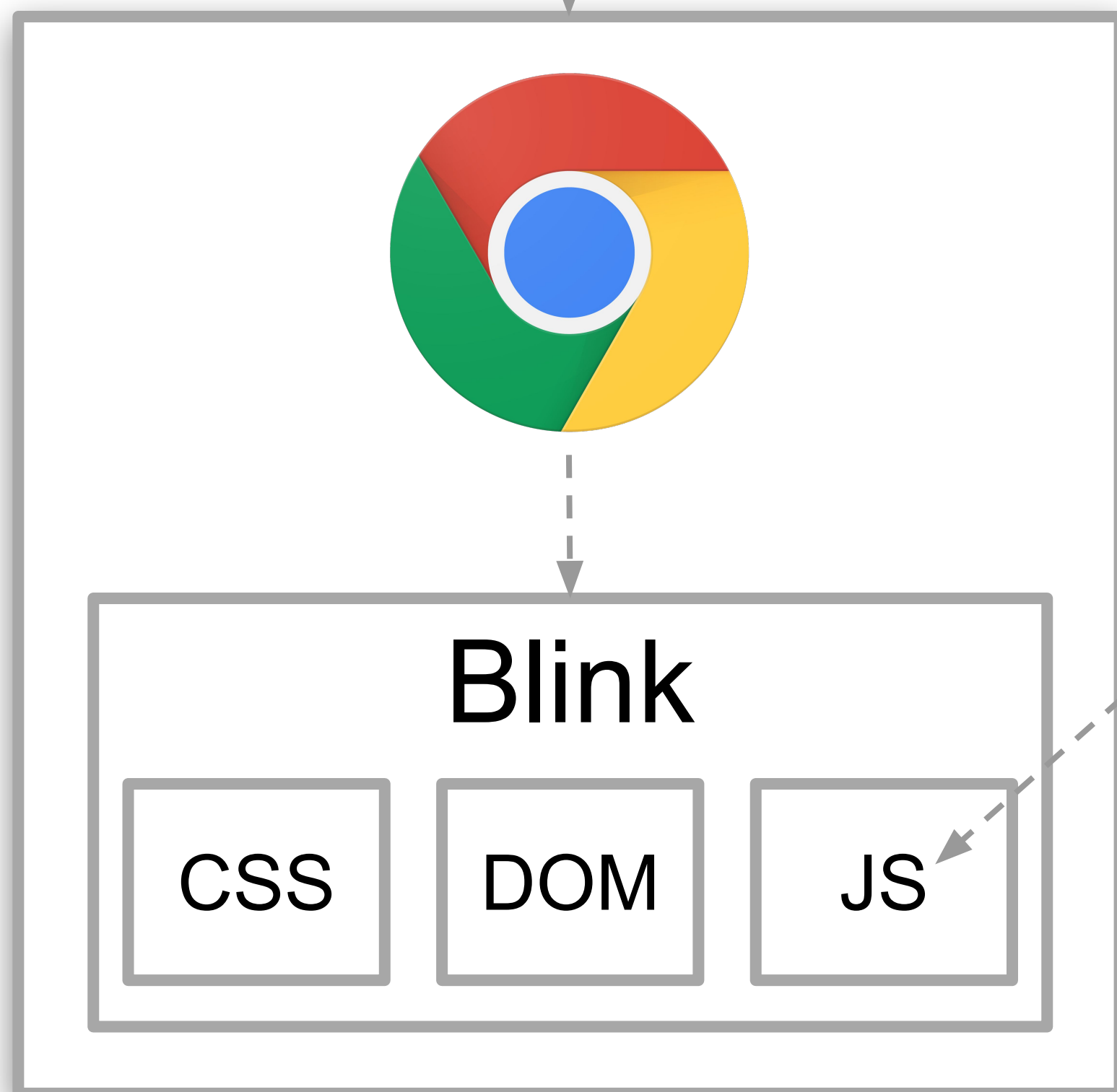
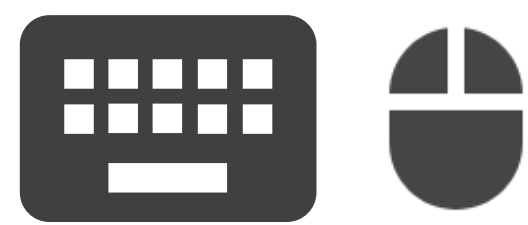


```
const mouseEvent = new MouseEvent( 'mousedown', {  
  clientX: 50,  
  clientY: 50,  
});  
document.body.dispatchEvent(mouseEvent);
```



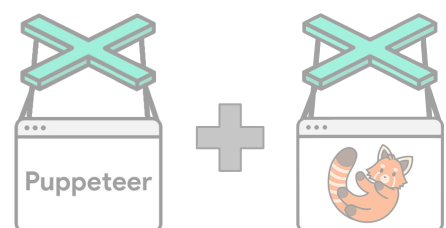
```
const mouseEvent = new MouseEvent( 'mousedown', {  
  clientX: 50,  
  clientY: 50,  
});  
document.body.dispatchEvent(mouseEvent);
```

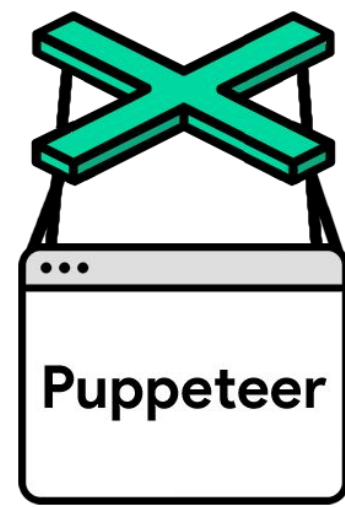
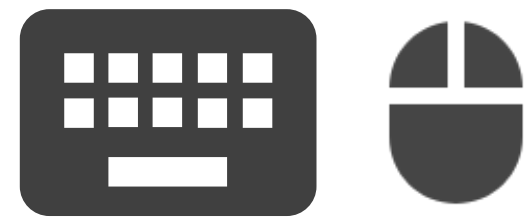




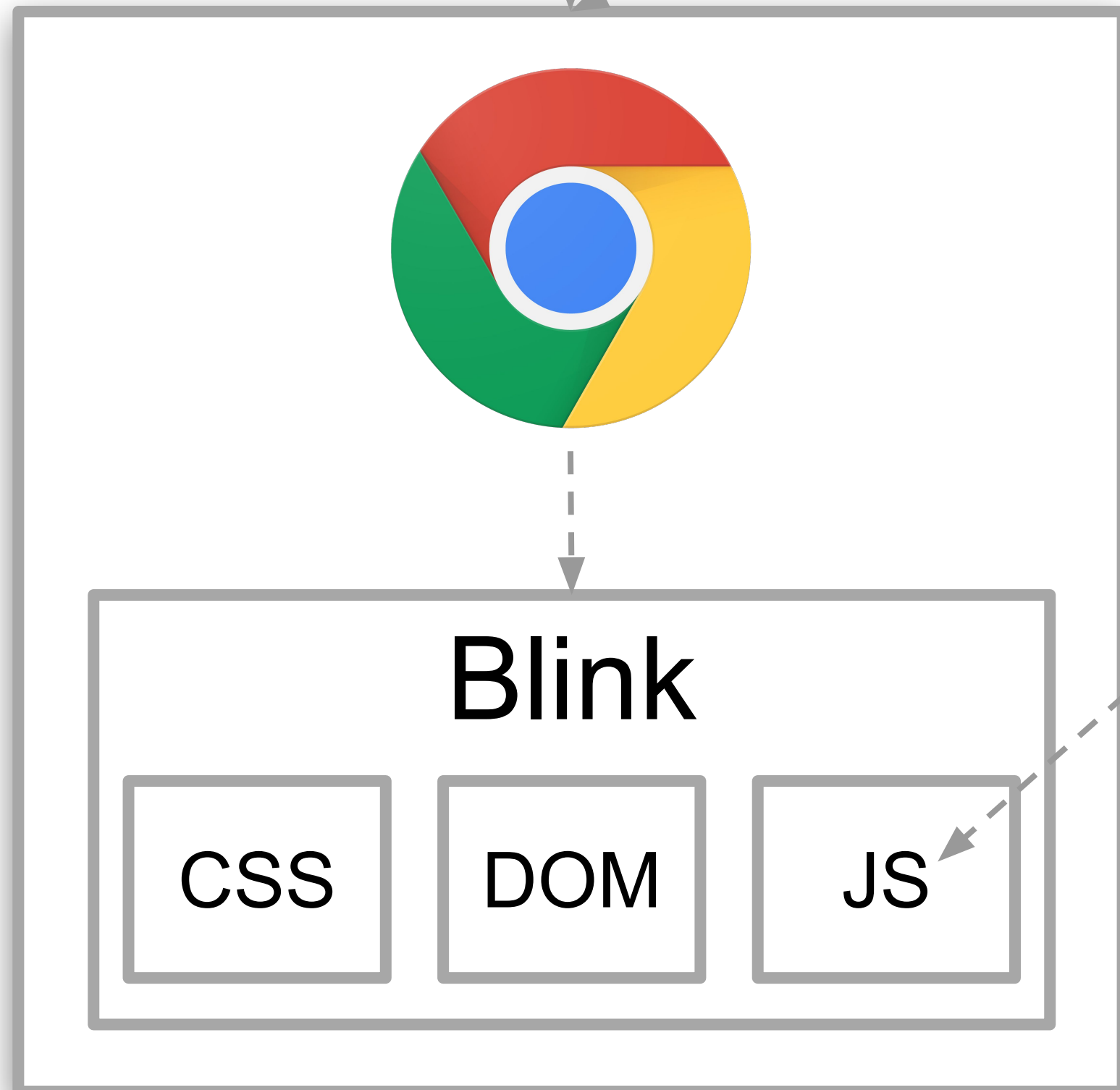
`element.dispatchEvent()`

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



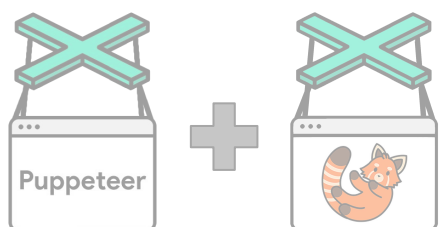


- ✓ event listeners
- ✓ focus
- ✓ selection
- ✓ :hover
- ✓ iframes
- ✓ shadow DOM
- ✓ insert text





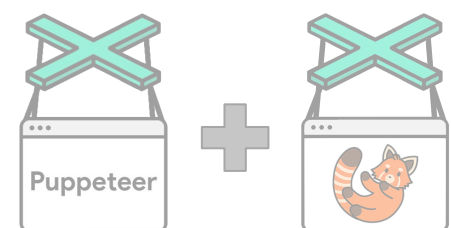
`element.dispatchEvent()`

- ✓ 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-firefox');
```

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

```
})();
```

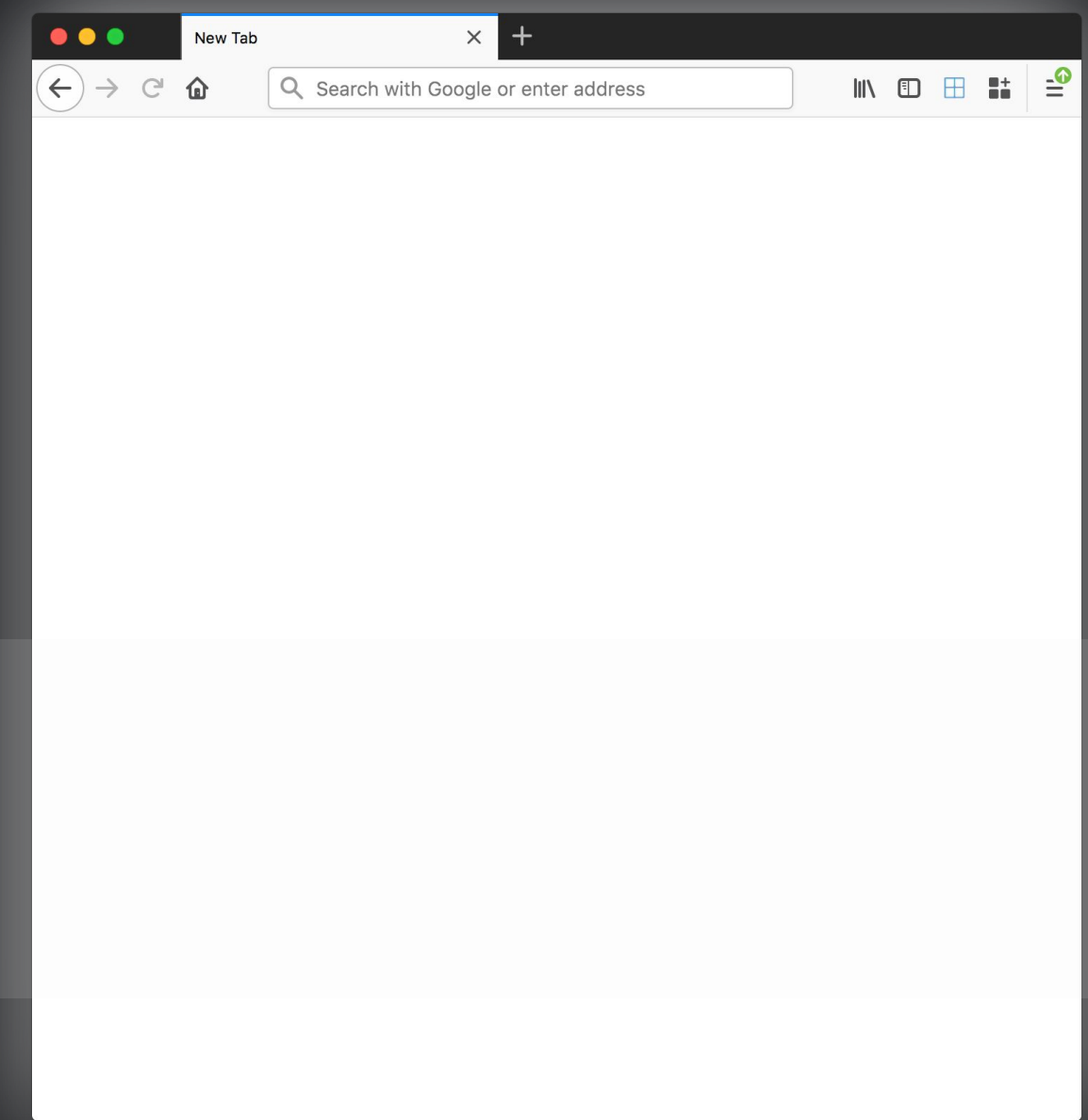
```
const puppeteer = require('puppeteer-firefox');
```

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

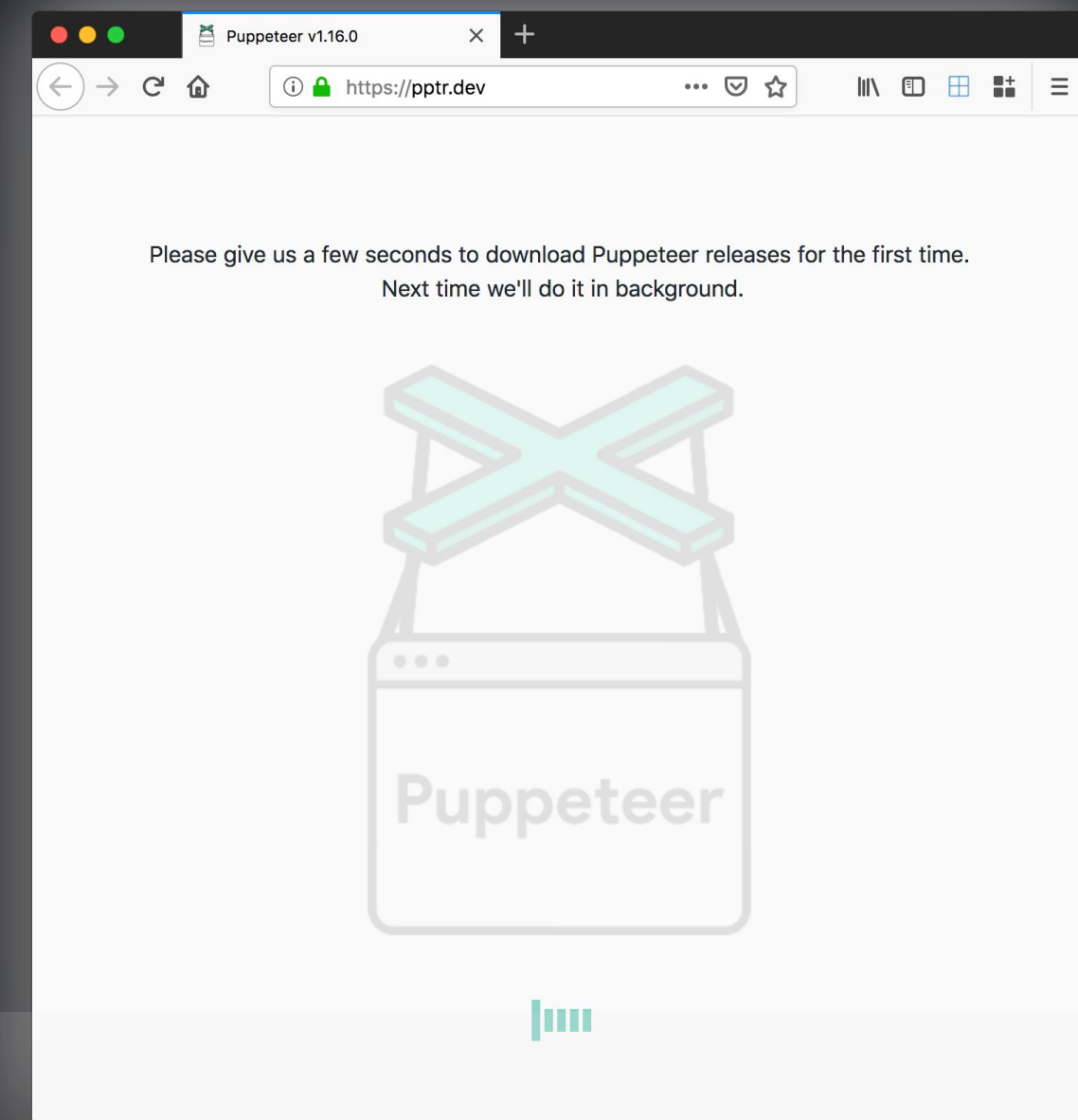
```
})();
```



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

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

```
})();
```



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

```
(async () => {
```

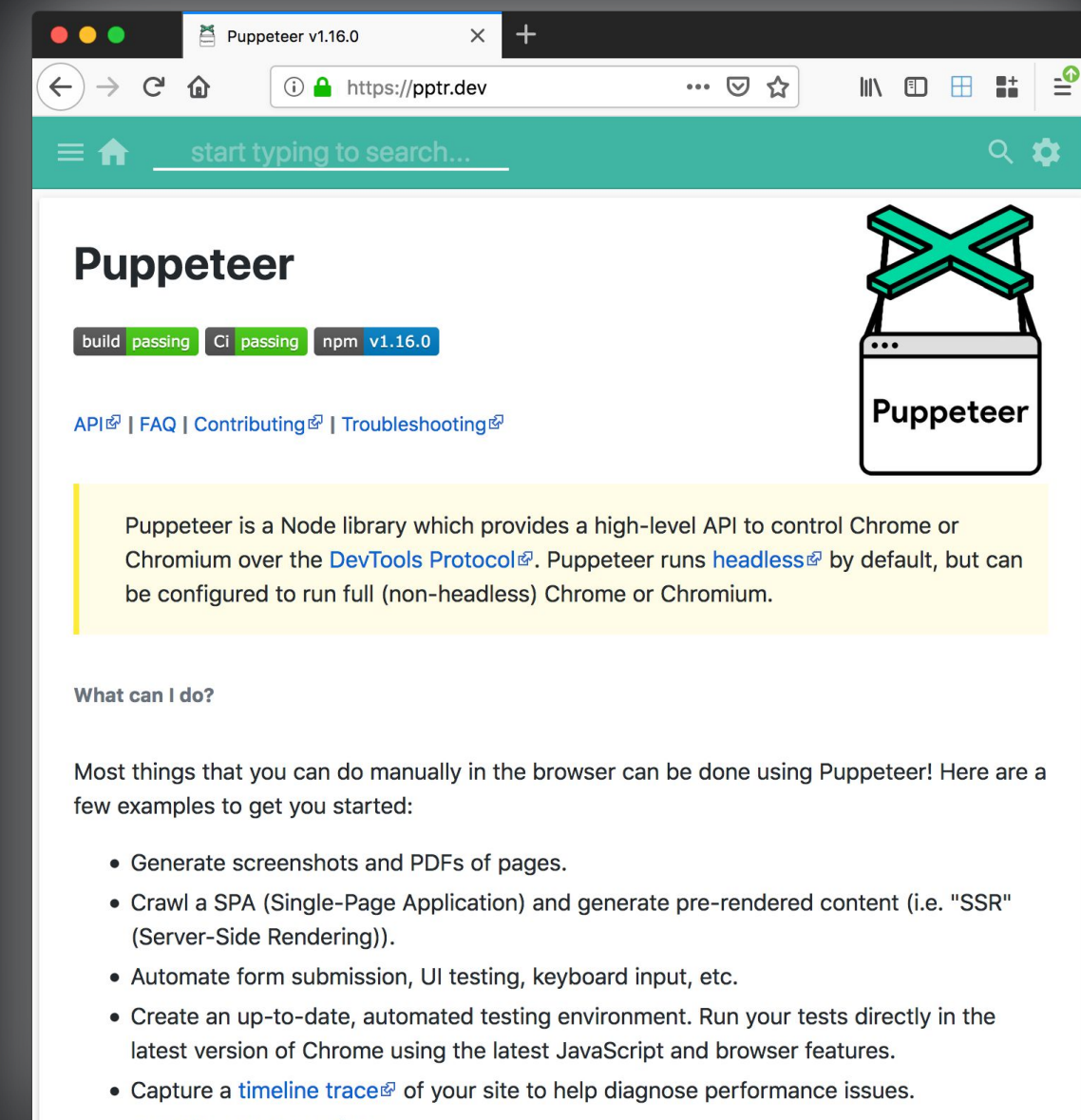
```
  const browser = await puppeteer.launch();
```

```
  const page = await browser.newPage();
```

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

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

```
})();
```



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

```
(async () => {
```

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

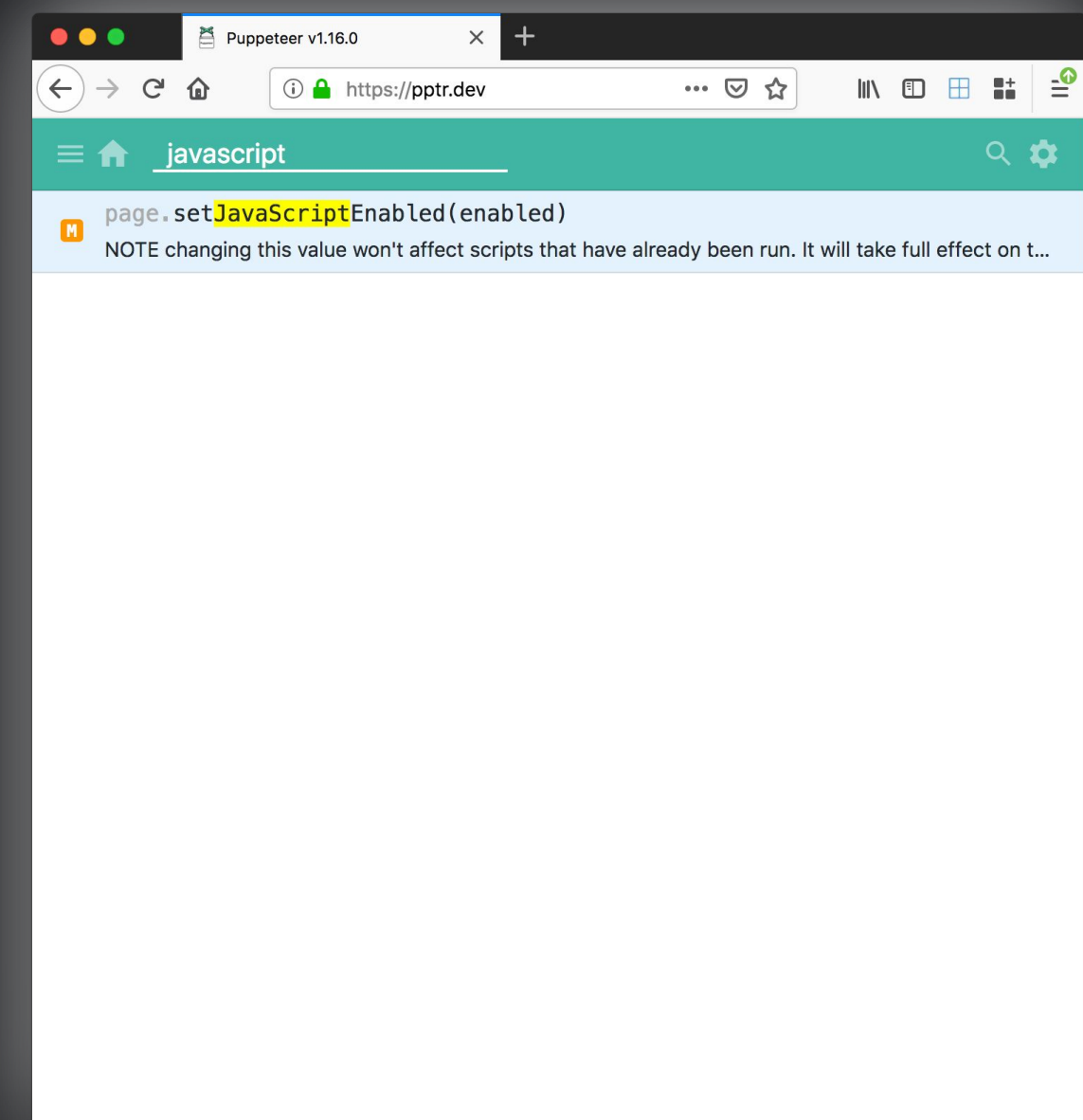
```
  const page = await browser.newPage();
```

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

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

```
  await input.type('javascript');
```

```
})();
```



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

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

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

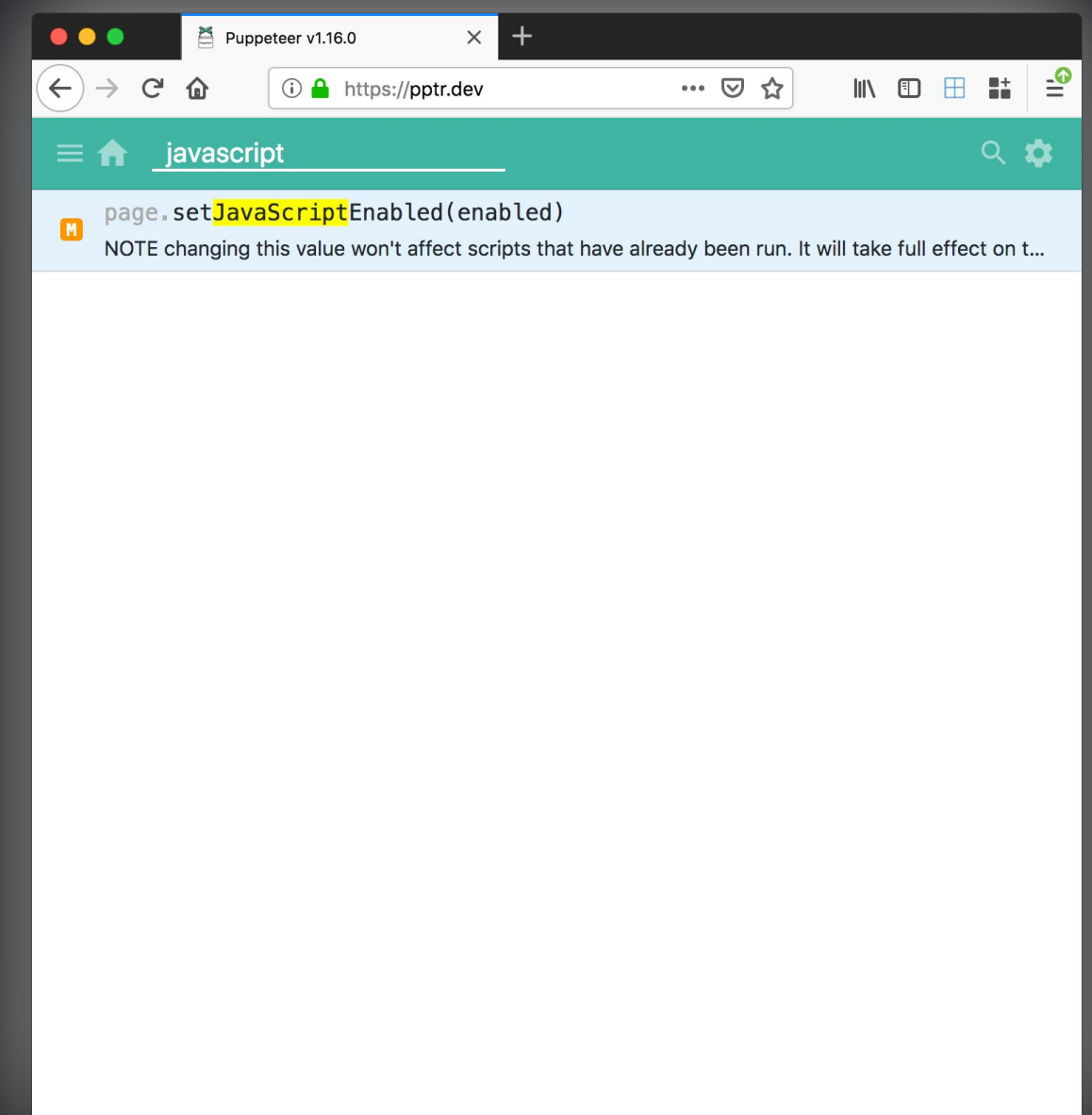
```
  const input = await page.waitForSelector('[type=search]');
```

```
  await input.type('javascript');
```

```
  await Promise.all([
```

```
    ]);
```

```
})();
```





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

```
(async () => {
```

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

```
  const page = await browser.newPage();
```

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

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

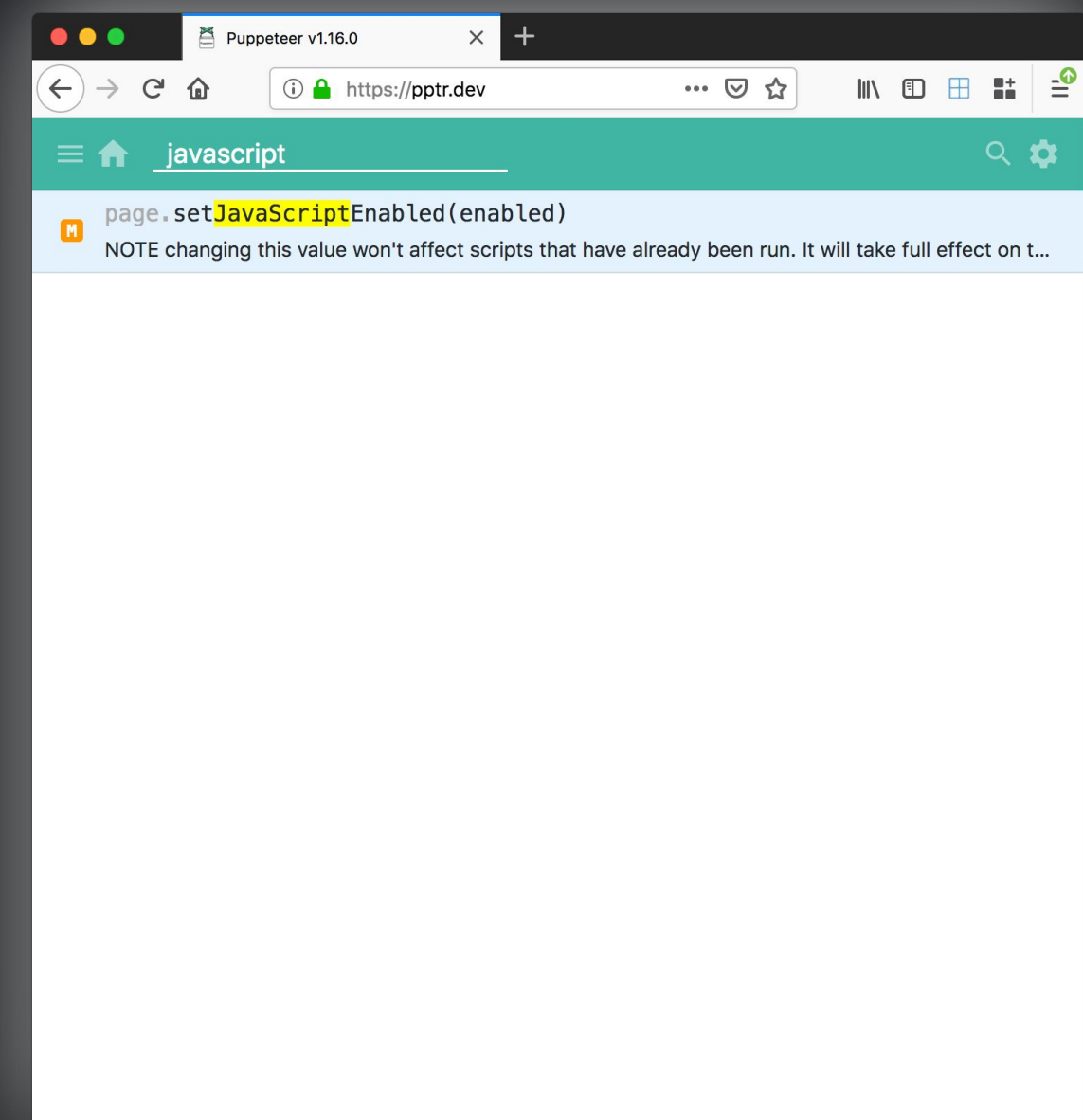
```
  await input.type('javascript');
```

```
  await Promise.all([
```

```
    page.keyboard.press('Enter'),
```

```
  ]);
```

```
})();
```



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

```
(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');
```

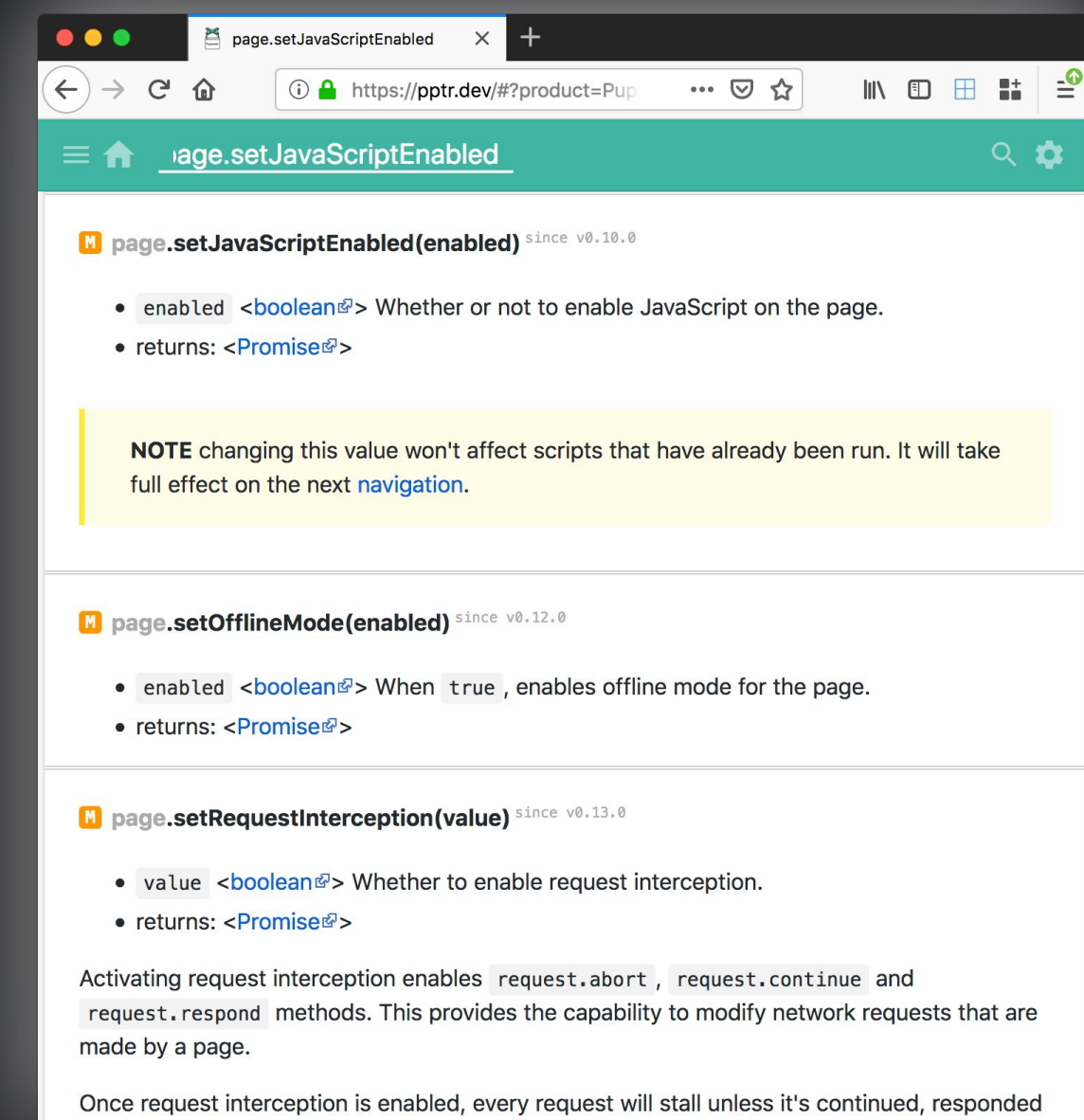
```
  await Promise.all([
```

```
    page.keyboard.press('Enter'),
```

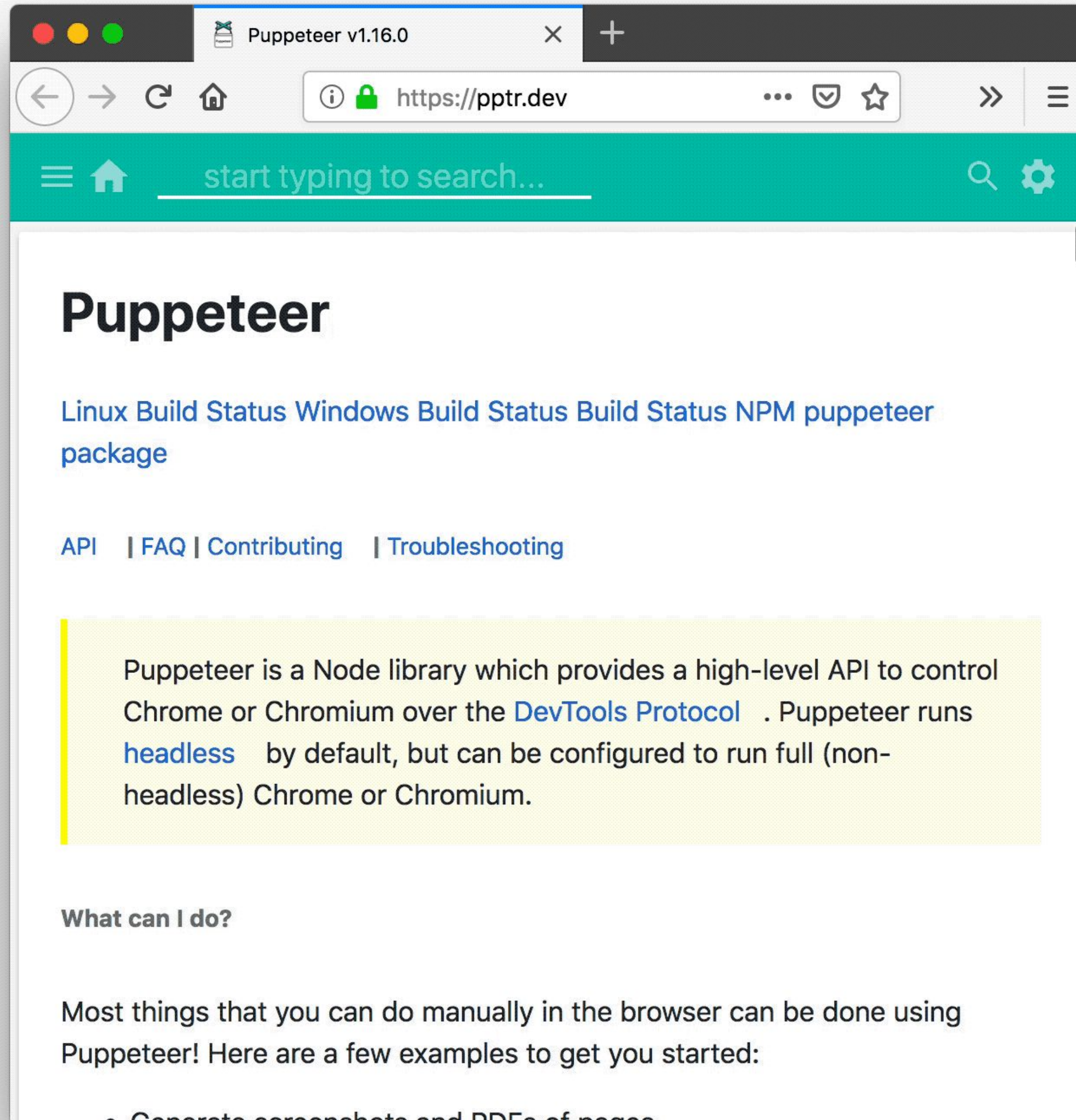
```
    page.waitForNavigation(),
```

```
  ]);
```

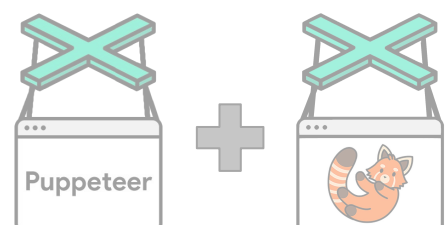
```
})();
```





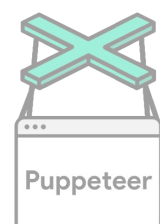


The screenshot shows a browser window with the title "Puppeteer v1.16.0" and the URL "https://pptr.dev". The page content includes a search bar, a main heading "Puppeteer", and several links: "Linux Build Status", "Windows Build Status", "Build Status NPM puppeteer package", "API", "FAQ", "Contributing", and "Troubleshooting". A highlighted text block states: "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, a section titled "What can I do?" begins with the text: "Most things that you can do manually in the browser can be done using Puppeteer! Here are a few examples to get you started:".



# Summary

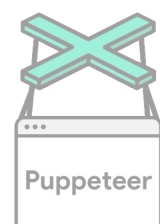
- 👉 All navigations are supported
- 👉 Tests are isolated from web content
- 👉 Real input



# Testing Modern Web

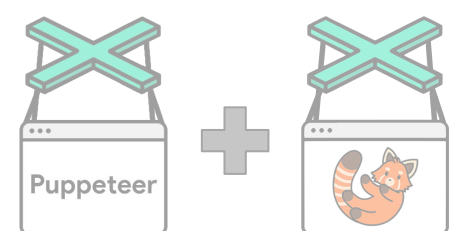
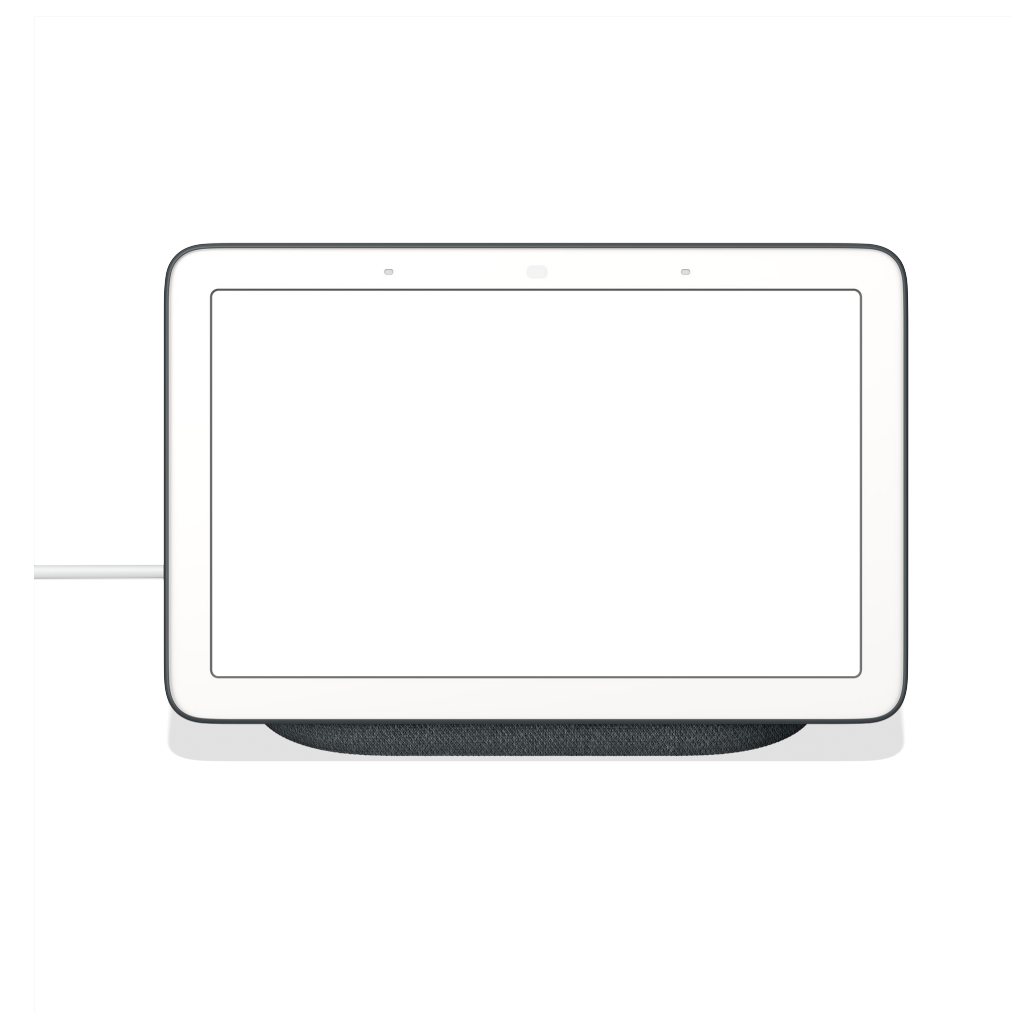
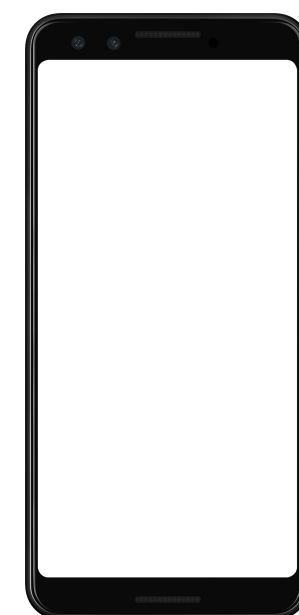
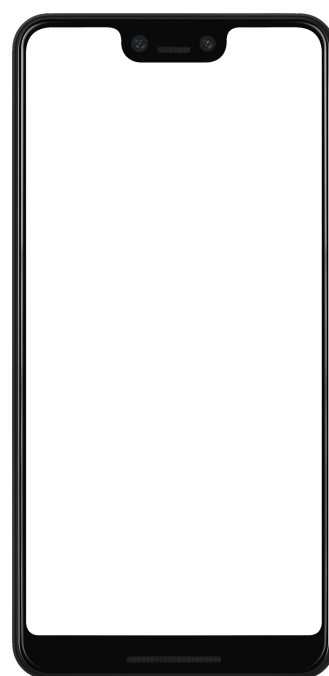
# Modern Web

- 👉 WebApps
- 👉 Performance
- 👉 Accessibility





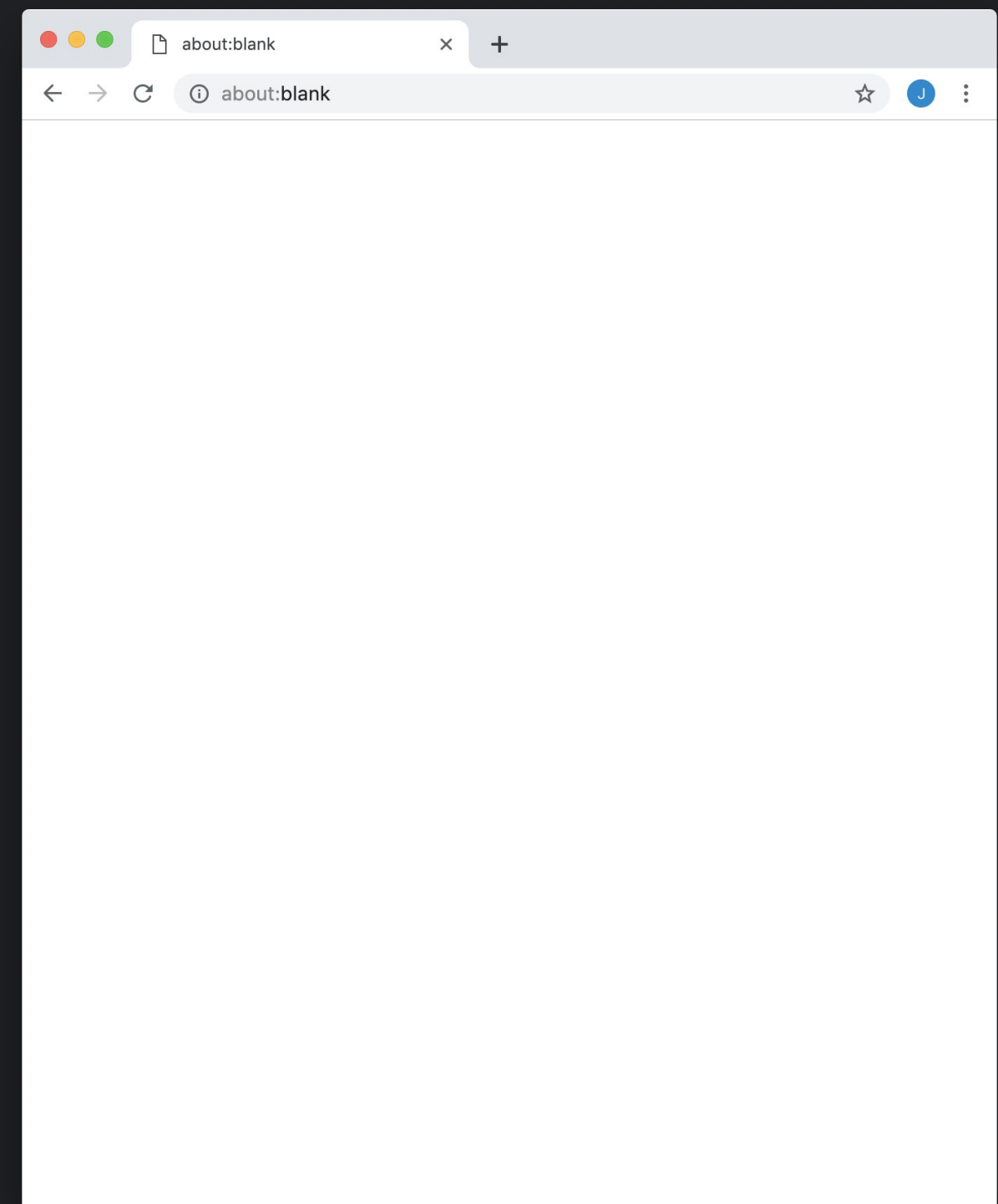
# Mobile





```
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/');  
  
  })
```



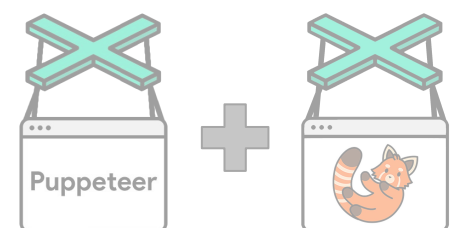
```
it('should work on mobile', async () => {  
  const page = await context.newPage();  
  await page.emulate(puppeteer.devices['iPhone 8']);  
  await page.goto('https://joel.tools/merch/');  
  const payButton = await page.waitForSelector('button.gpay-button');  
  expect(payButton).toBeTruthy();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



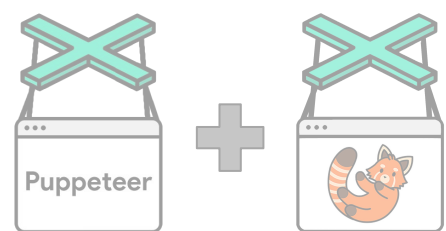


# Device Emulation

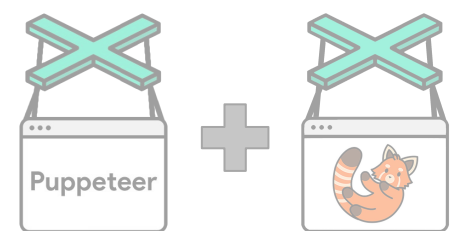
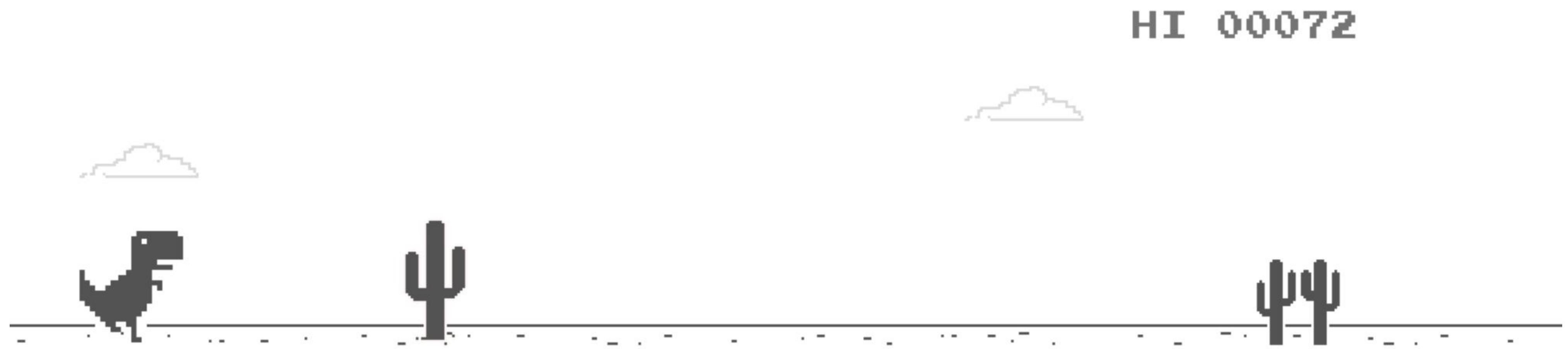
- 👉 User Agent
- 👉 Device Pixel Ratio
- 👉 Viewport size
- 👉 Touch Support

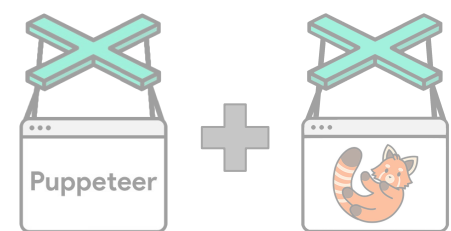
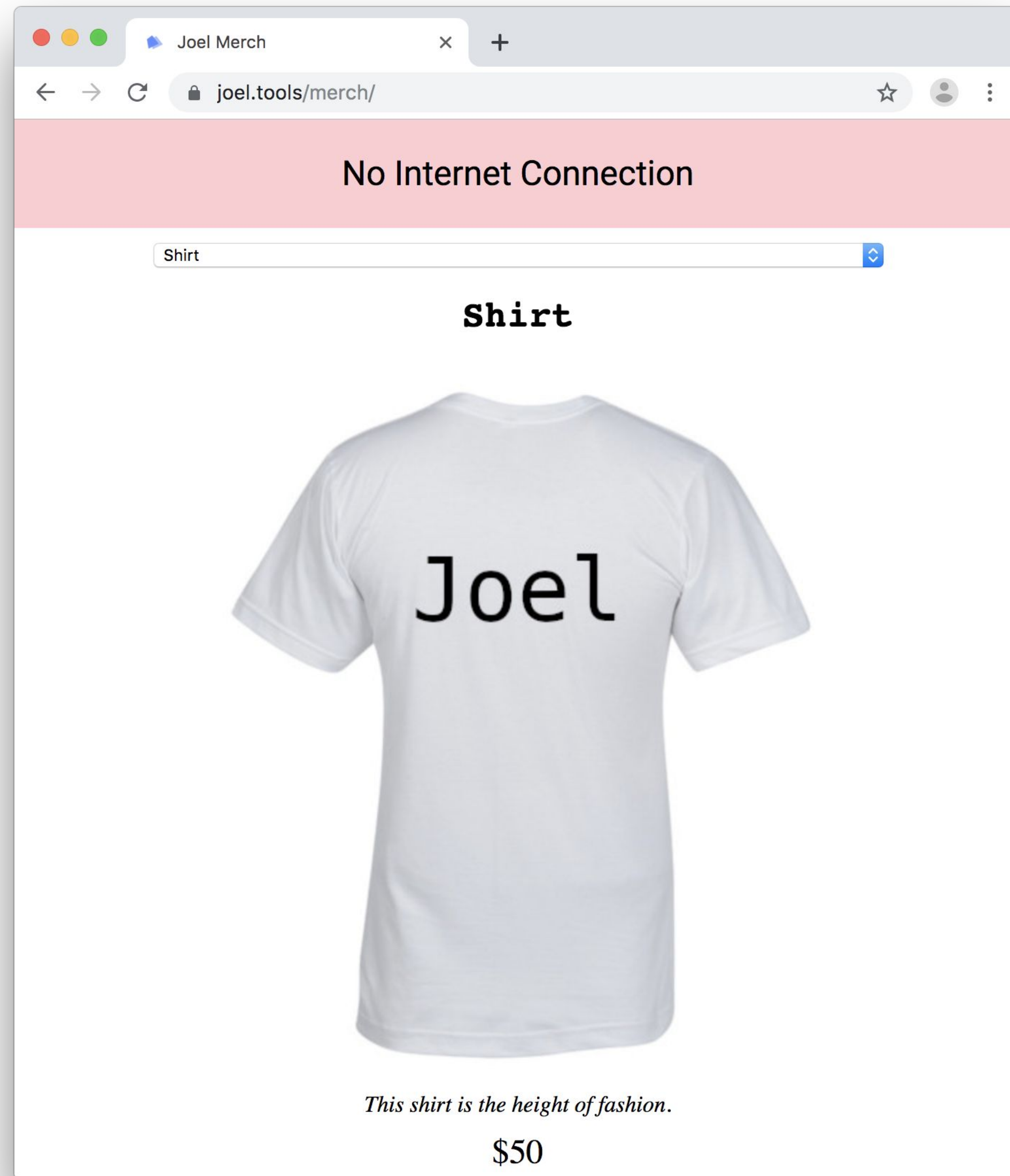


# 100+ devices



# Offline Emulation

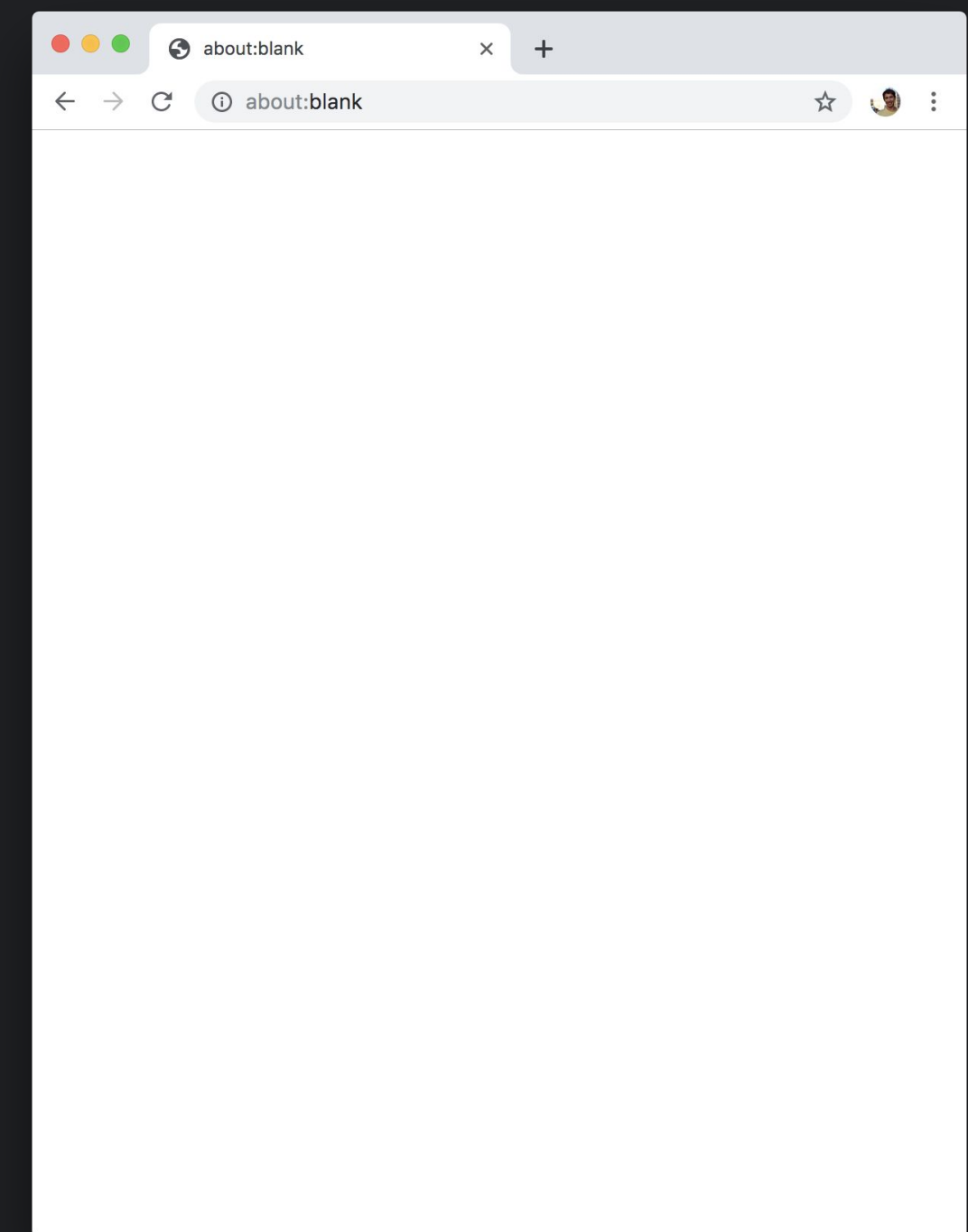






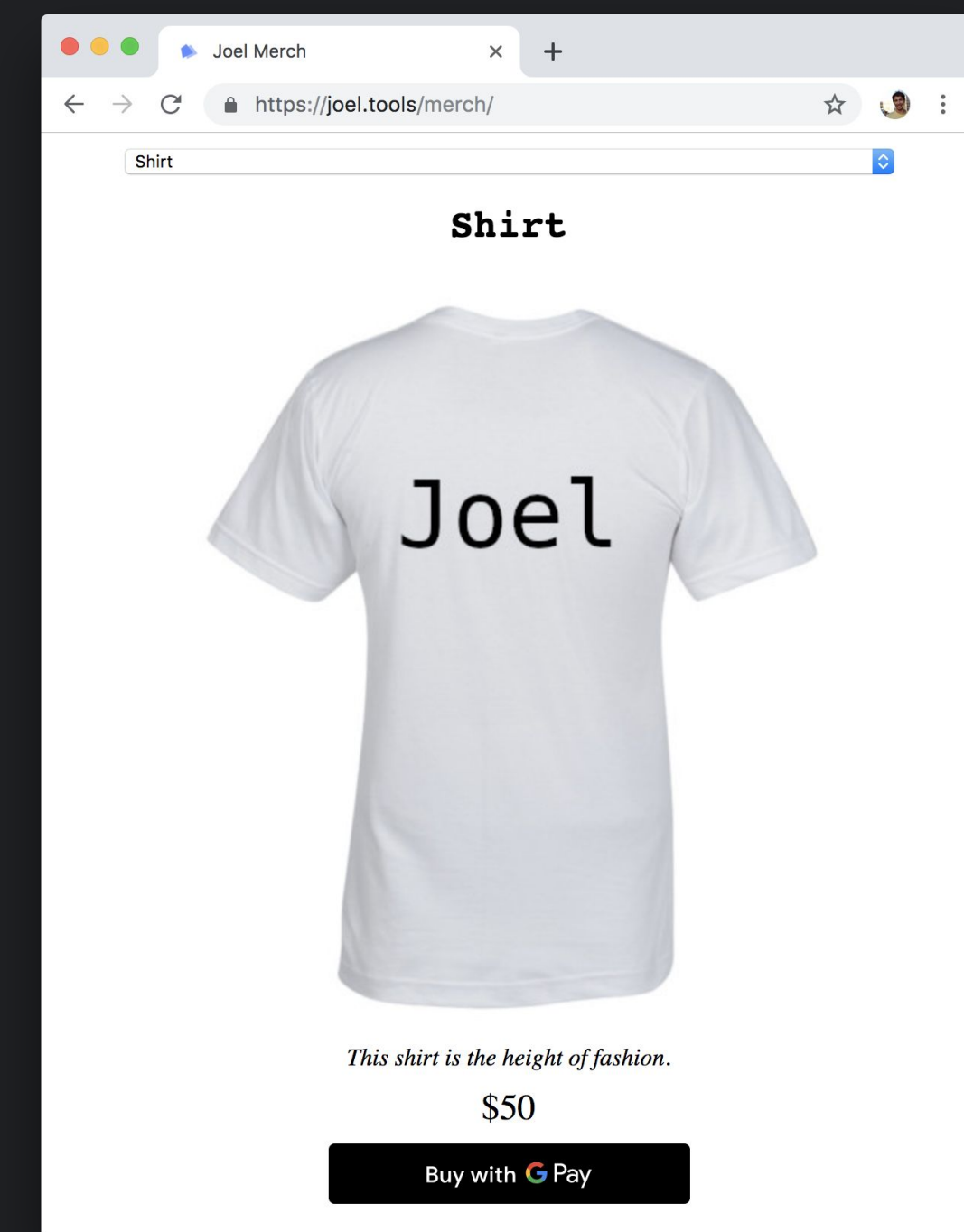
```
it('should notify user when goes offline', async () => {  
  const page = await context.newPage();
```

```
});
```

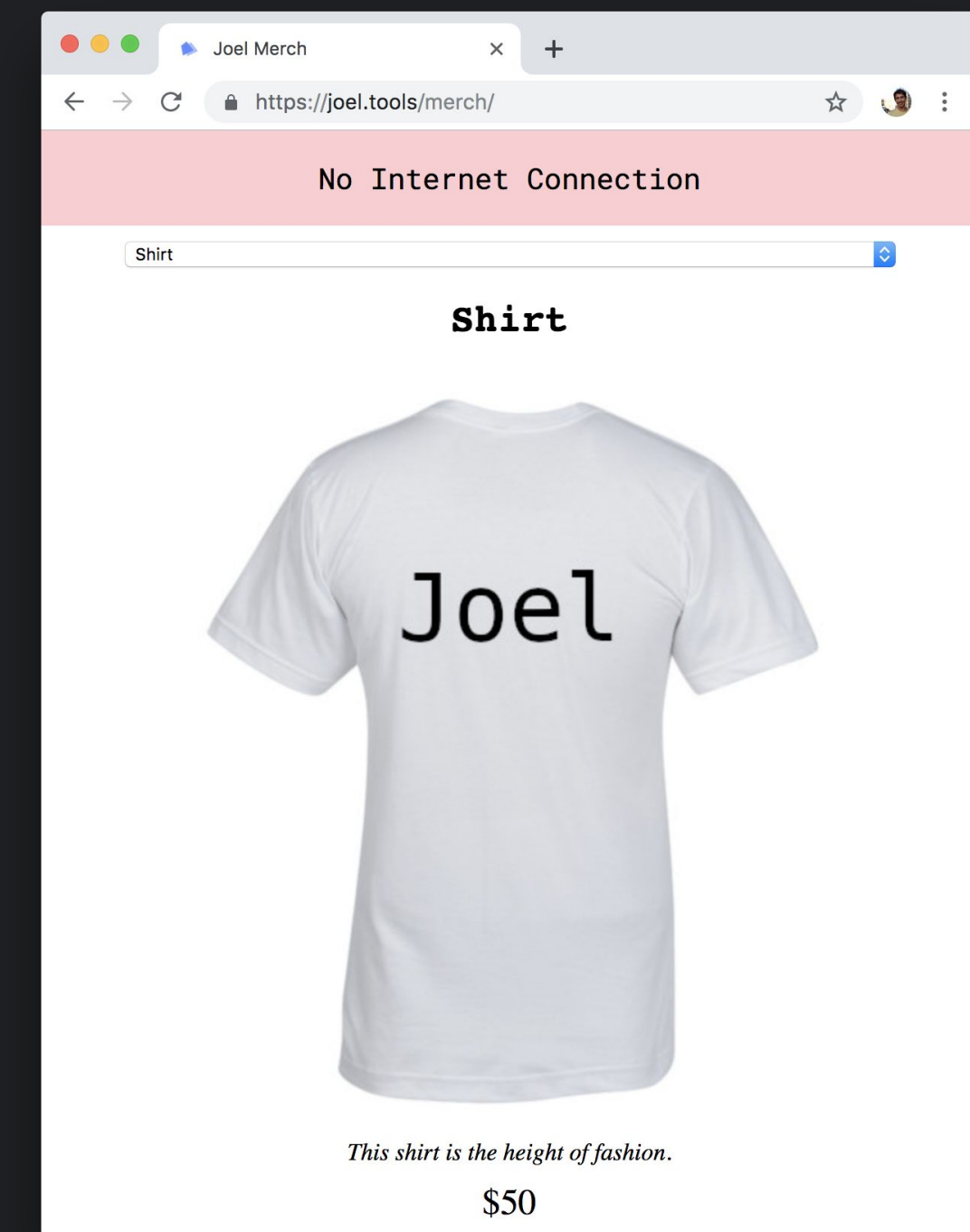




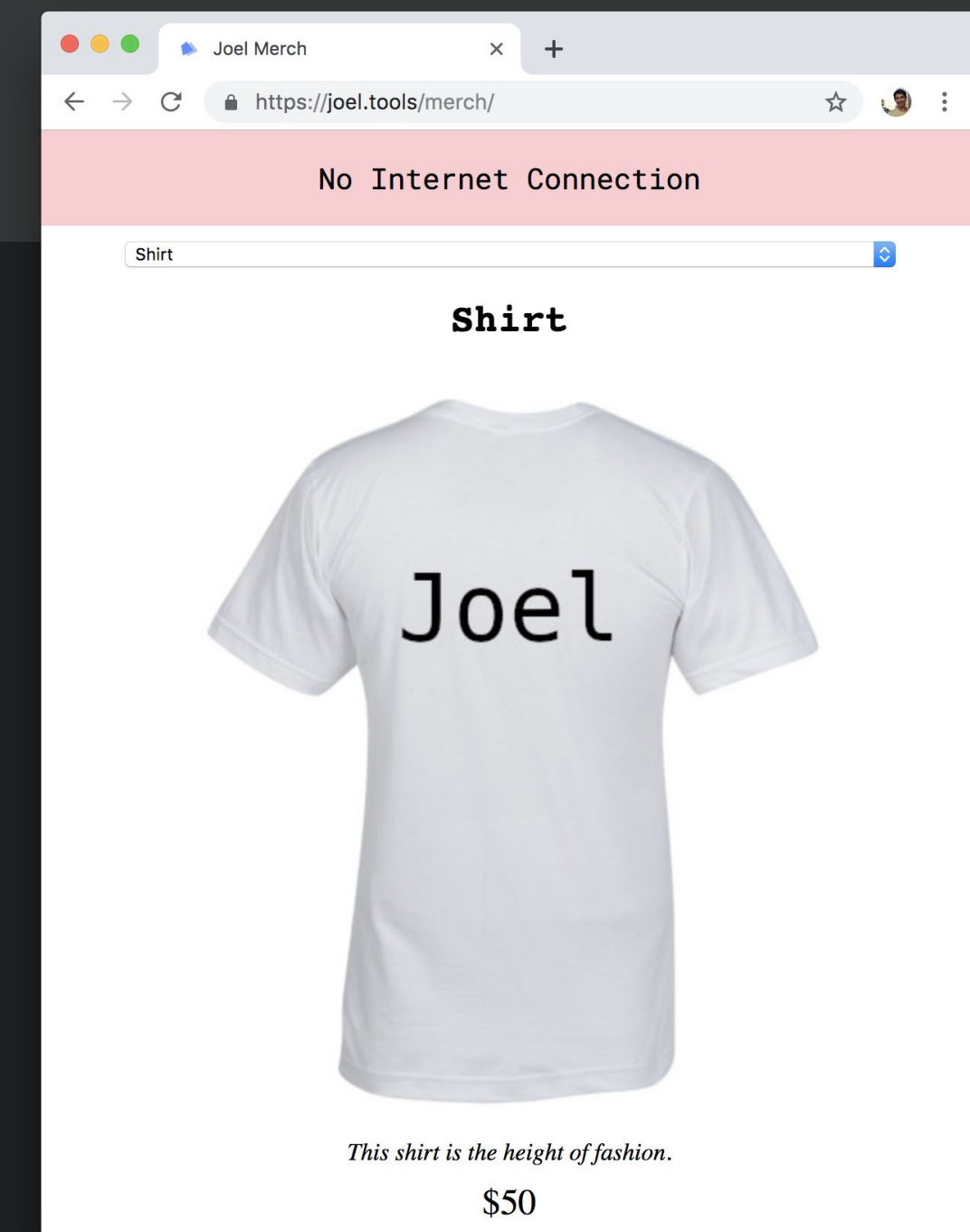
```
it('should notify user when goes offline', async () => {  
  const page = await context.newPage();  
  await page.goto('https://joel.tools/merch');  
  
});
```



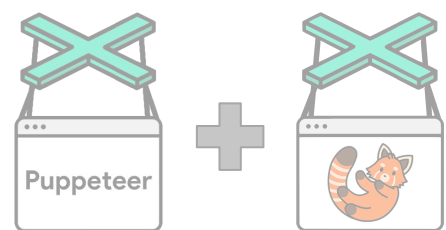
```
it('should notify user when goes offline', async () => {  
  const page = await context.newPage();  
  await page.goto('https://joel.tools/merch');  
  await page.setOfflineMode(true);  
  
});
```



```
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();  
});
```



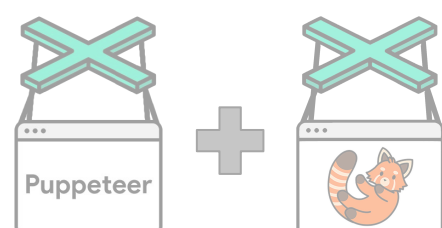
# Service Workers







# pptr.dev



page.evaluate

page.evaluate

**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`:

HEISENBUG



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

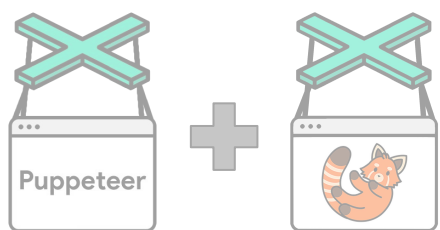
Status ● #33041 activated and is running [stop](#)

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

Push  [Push](#)

Sync  [Sync](#)

▶ Service workers from other domains

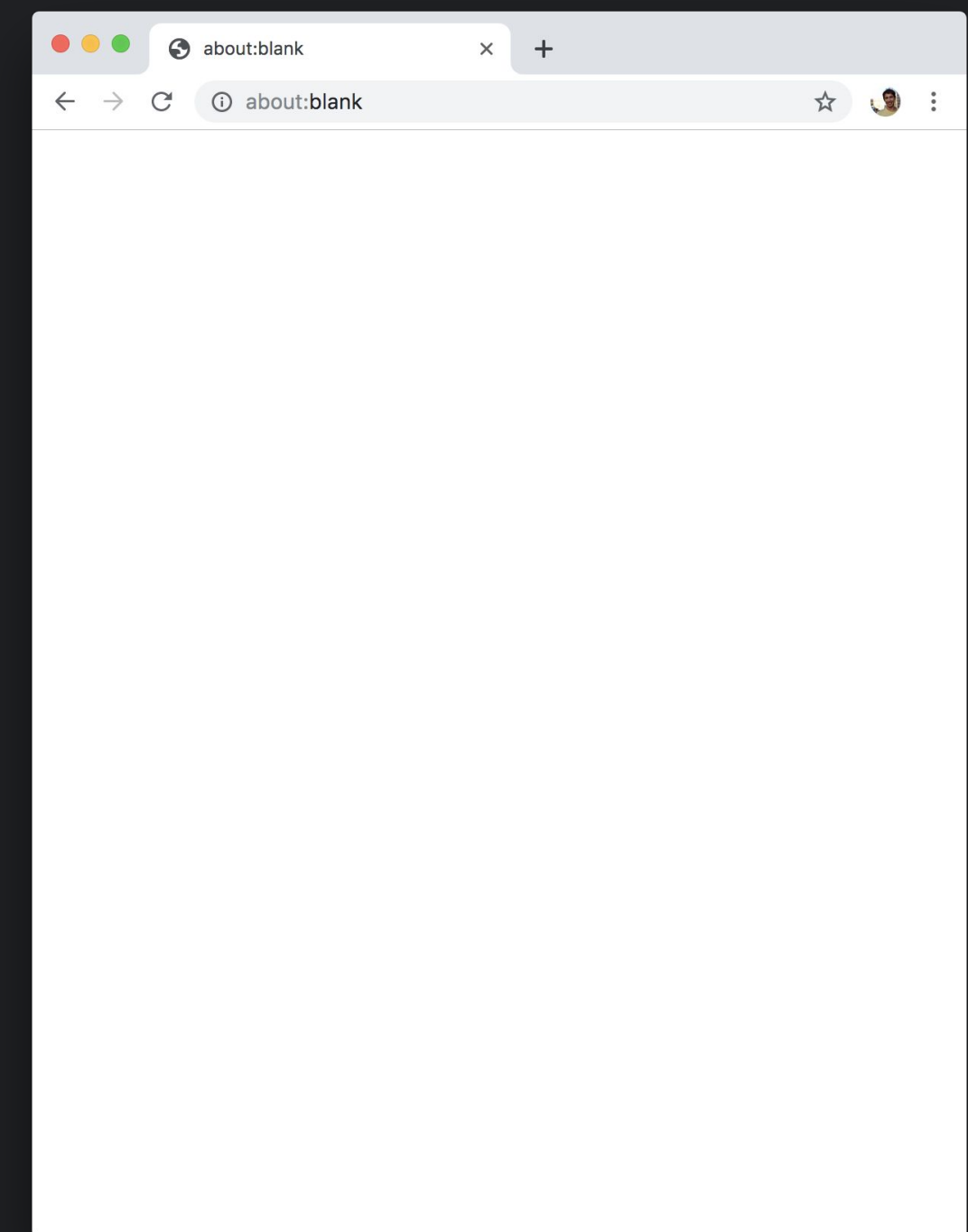




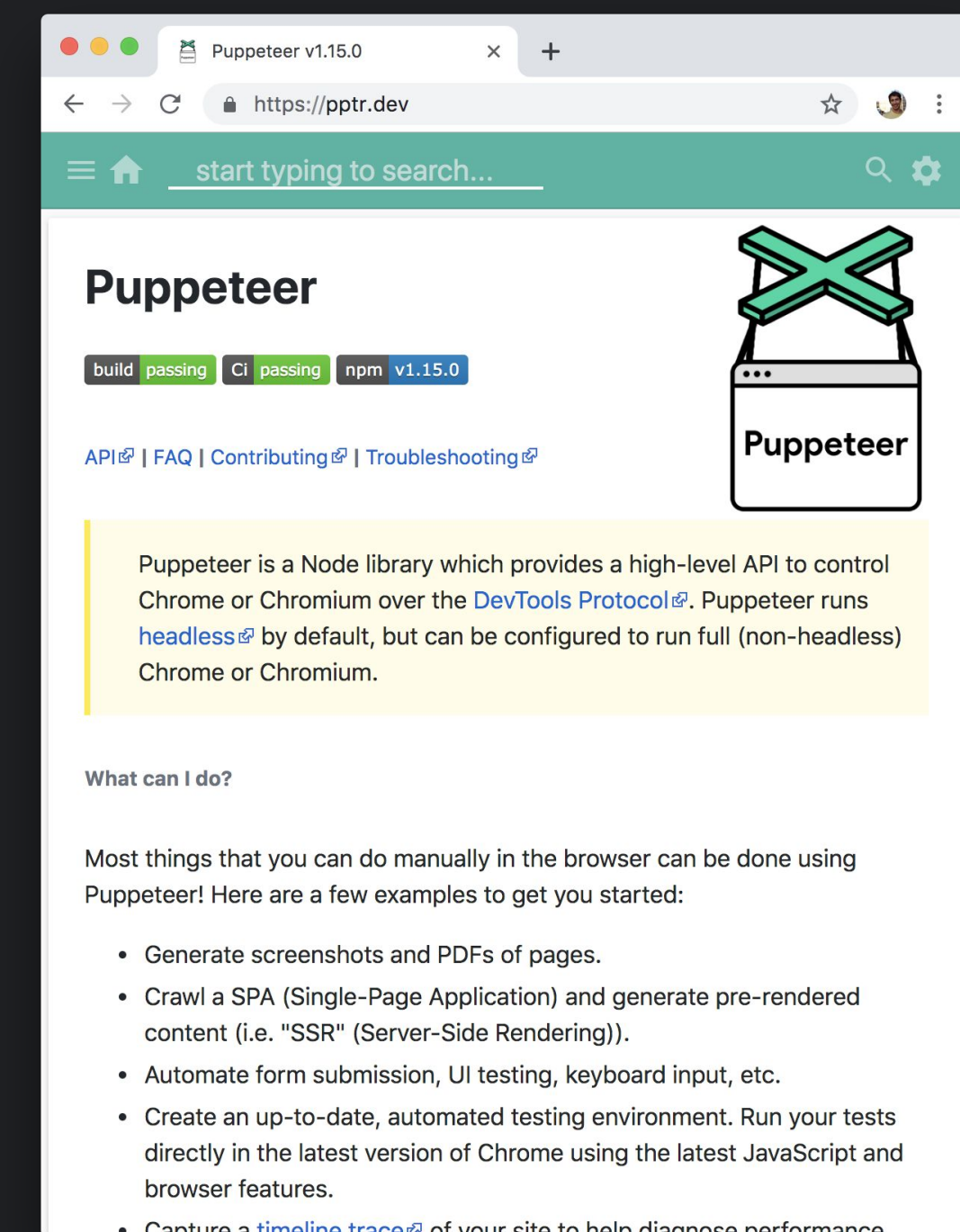


```
it('should register a service worker', async () => {  
  const page = await context.newPage();
```

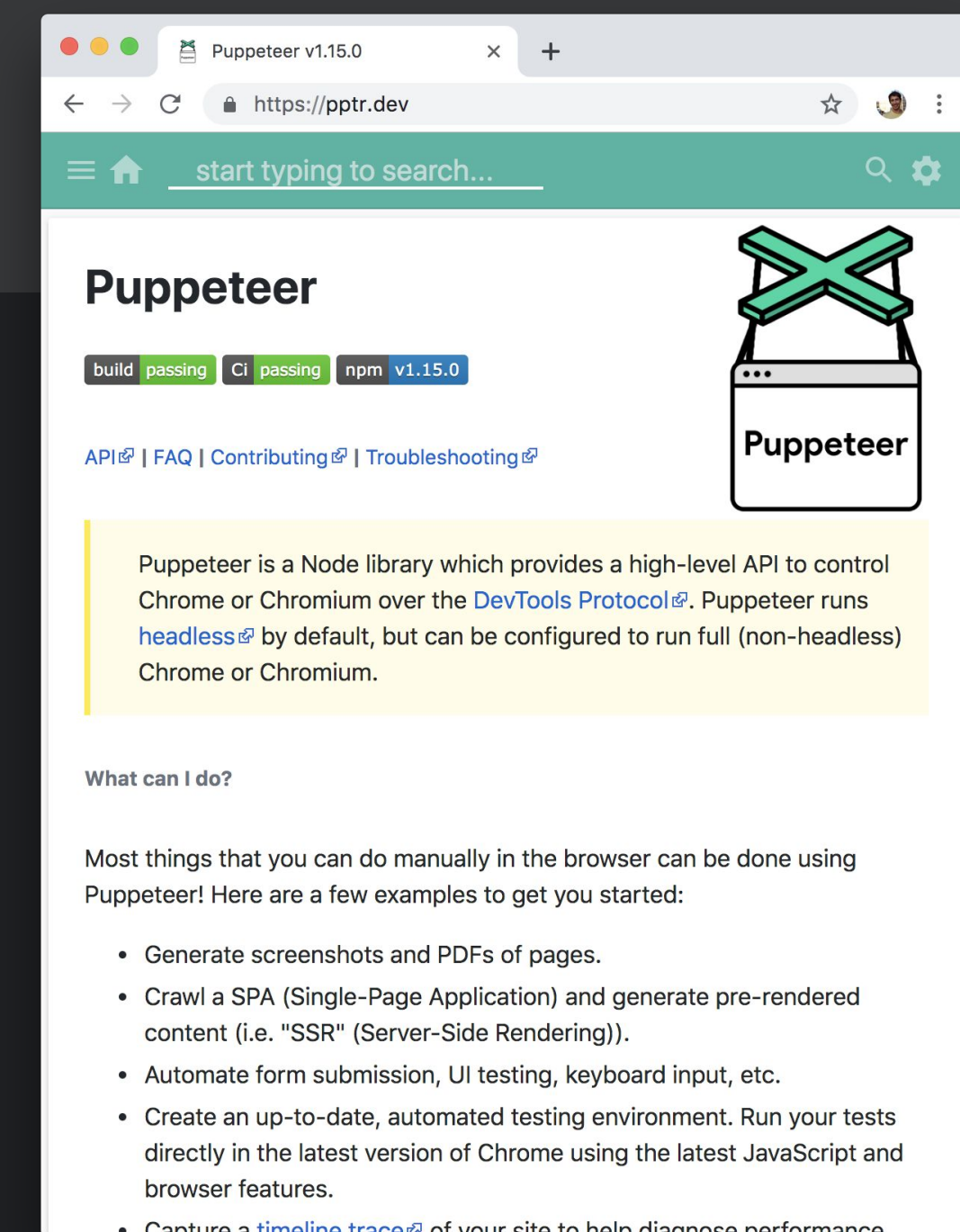
```
})
```



```
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';  
});
```

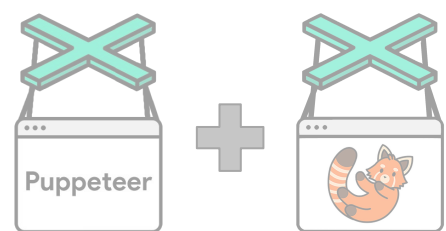
...

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

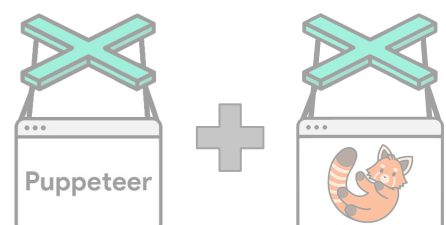
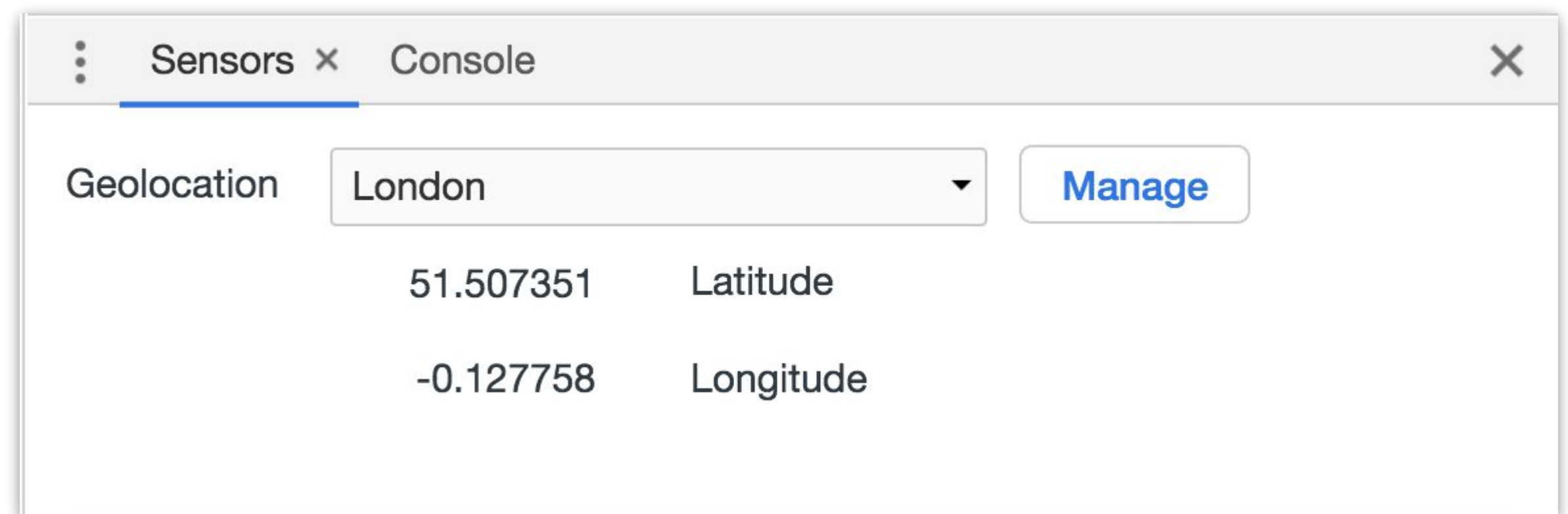
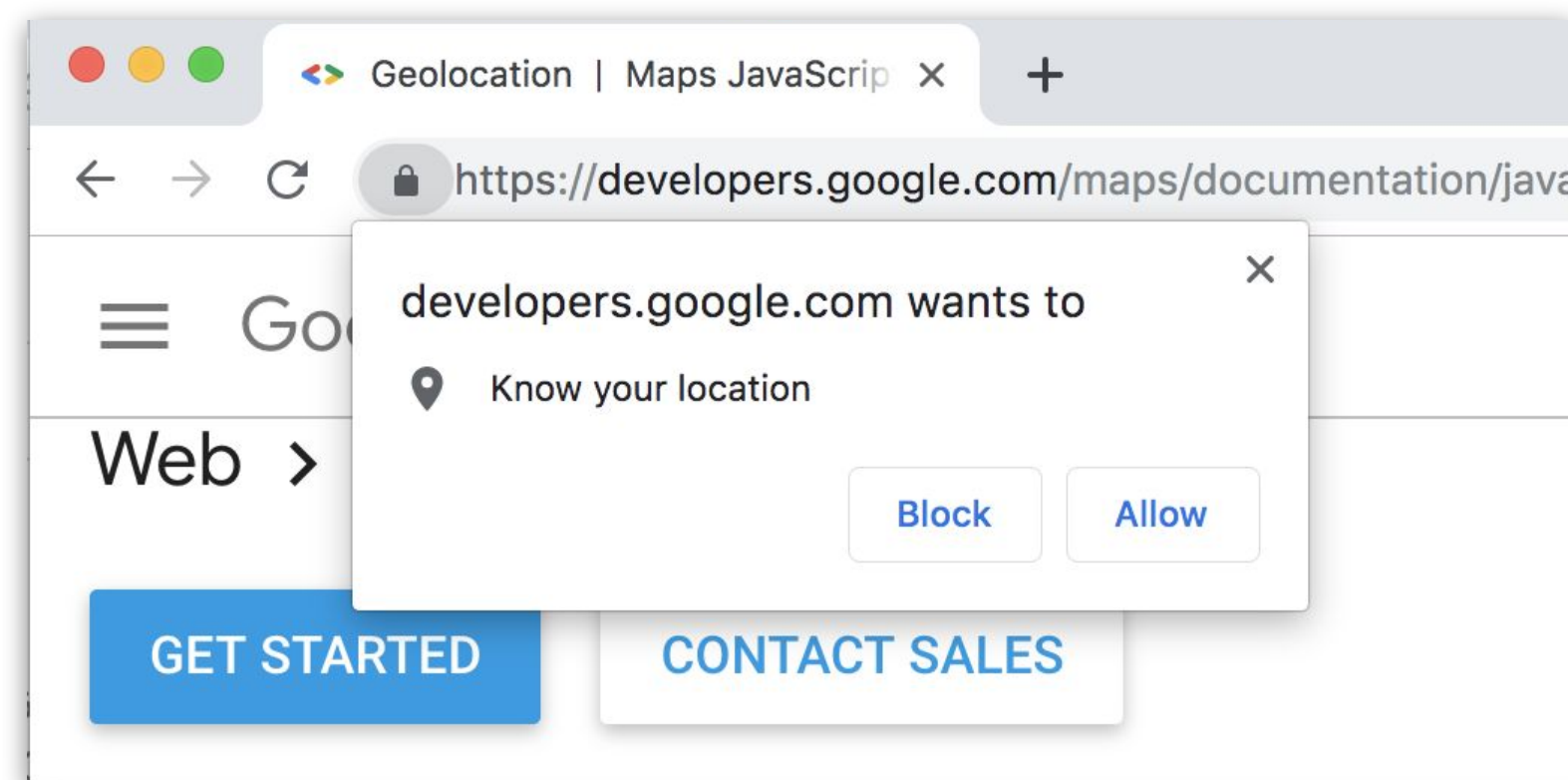


```
...
const swTarget = await context.waitForTarget(target => {
  return target.type() === 'service_worker';
});
const serviceWorker = await swTarget.worker();
const isCached = await serviceWorker.evaluate(async () => {
  return !!(await caches.match('https://pptr.dev/logo.png'));
});
expect(isCached).toBe(true);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

# Geolocation



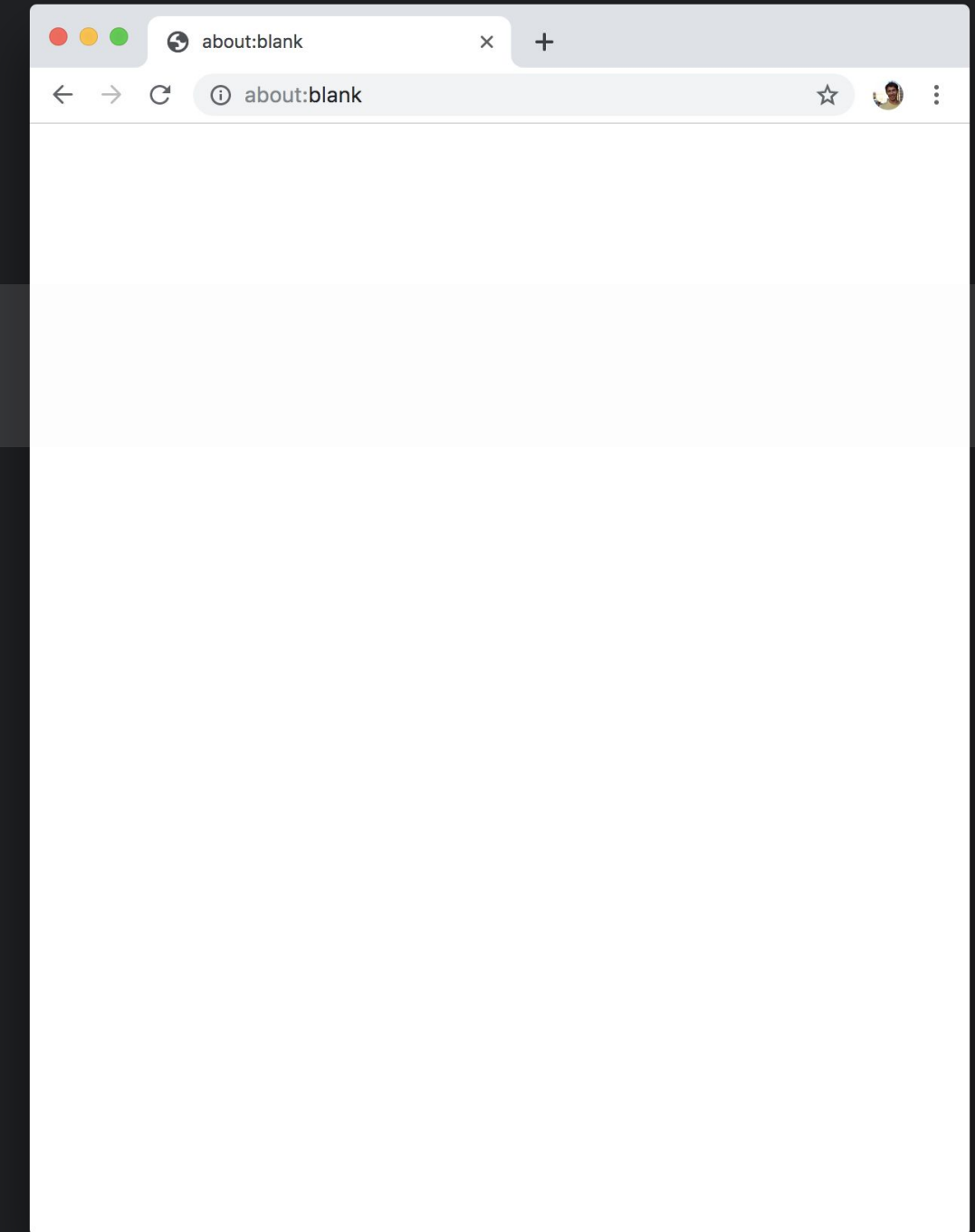
# Geolocation: London





```
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();
```



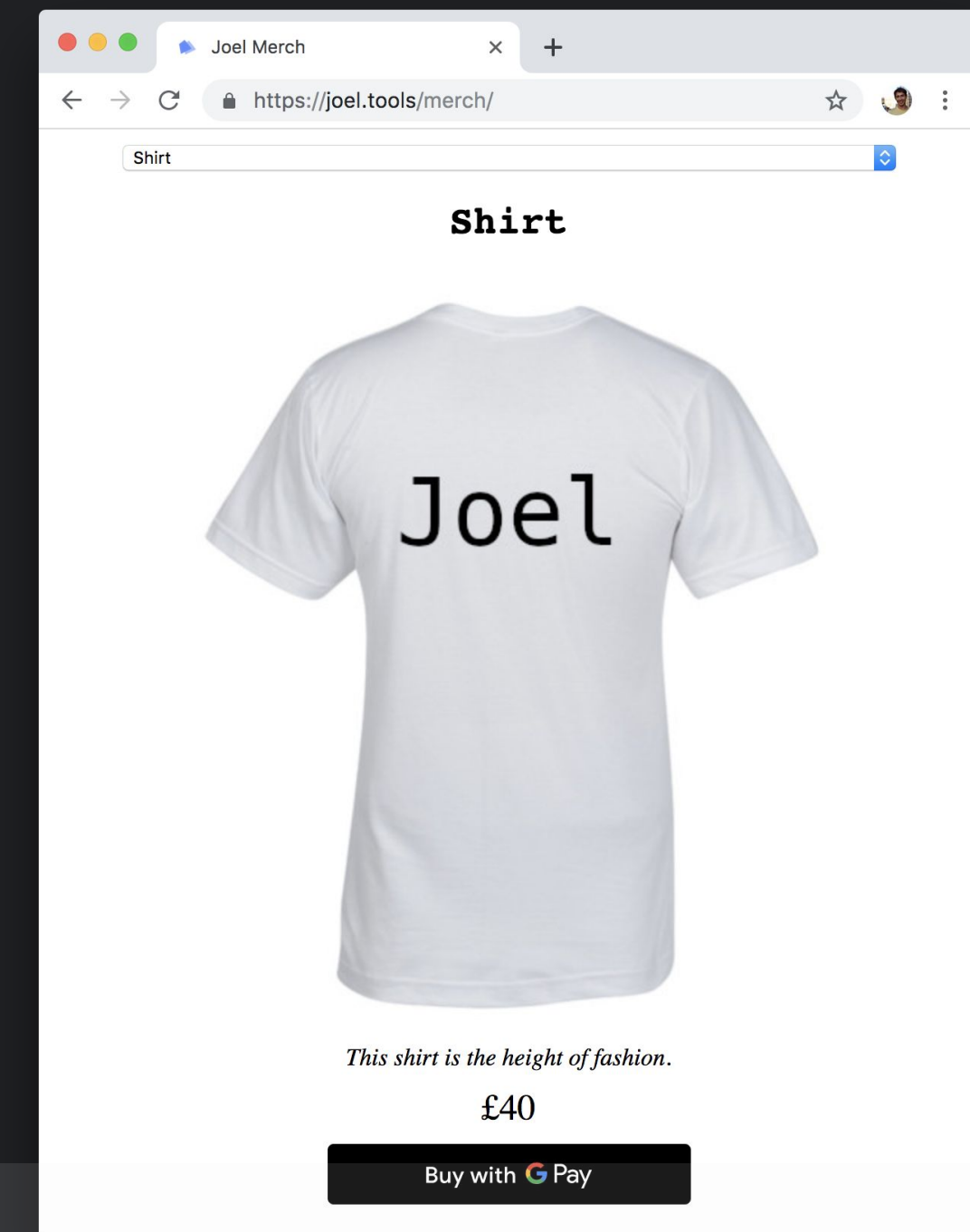
```
})
```



```
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: 51.50,  
    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: 51.50,  
    longitude: -0.12,  
  });  
  await page.goto('https://joel.tools/merch');  
  
  })
```

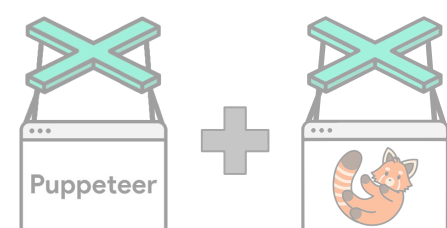
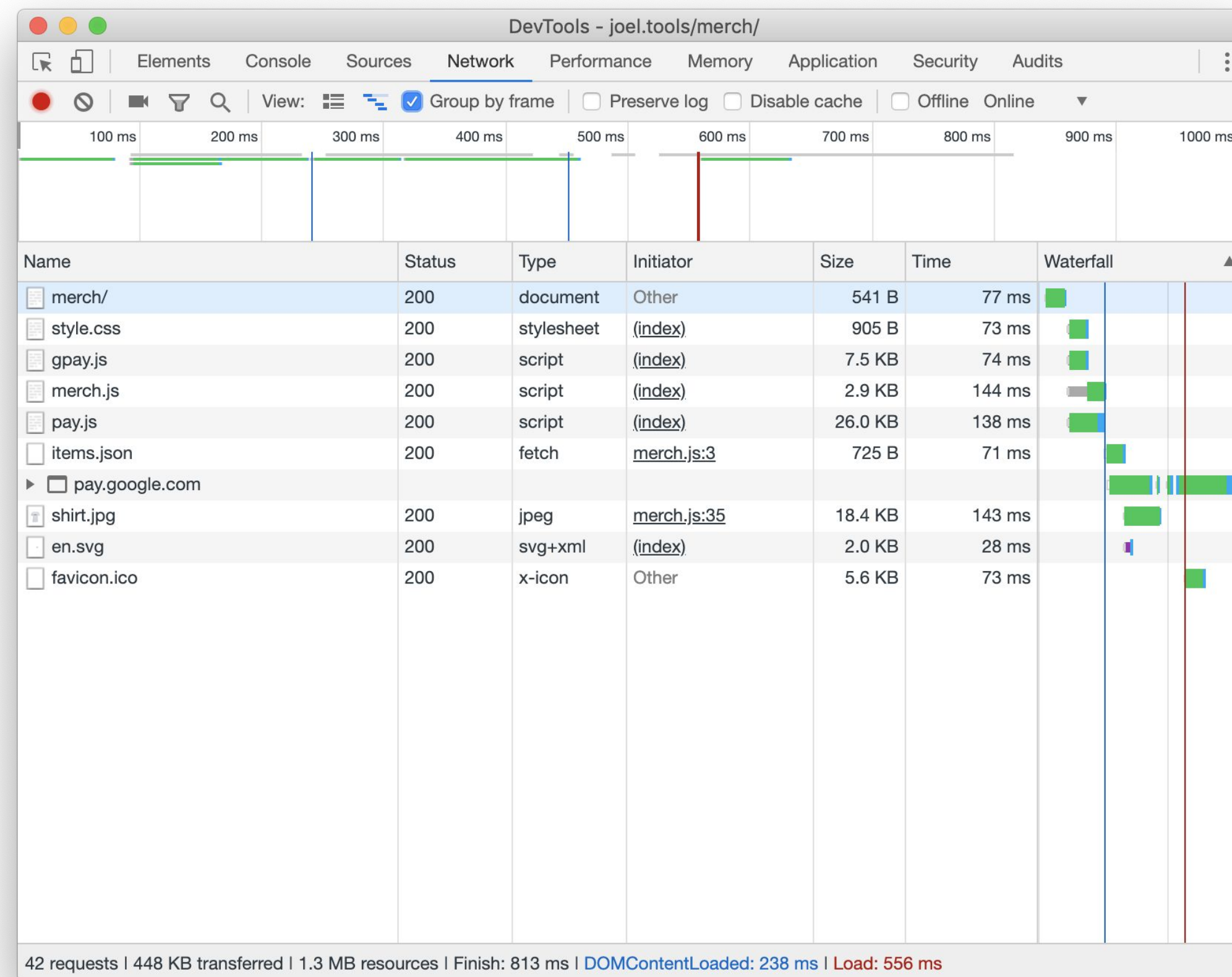


```
it('should have prices in pounds in London', async () => {
  await context.overridePermissions('https://joel.tools', [
    'geolocation',
  ]);
  const page = await context.newPage();
  await page.setGeolocation({
    latitude: 51.50,
    longitude: -0.12,
  });
  await page.goto('https://joel.tools/merch');
  const price = await page.$eval('.price', div => div.textContent);
  expect(price).toBe('£40');
})
```



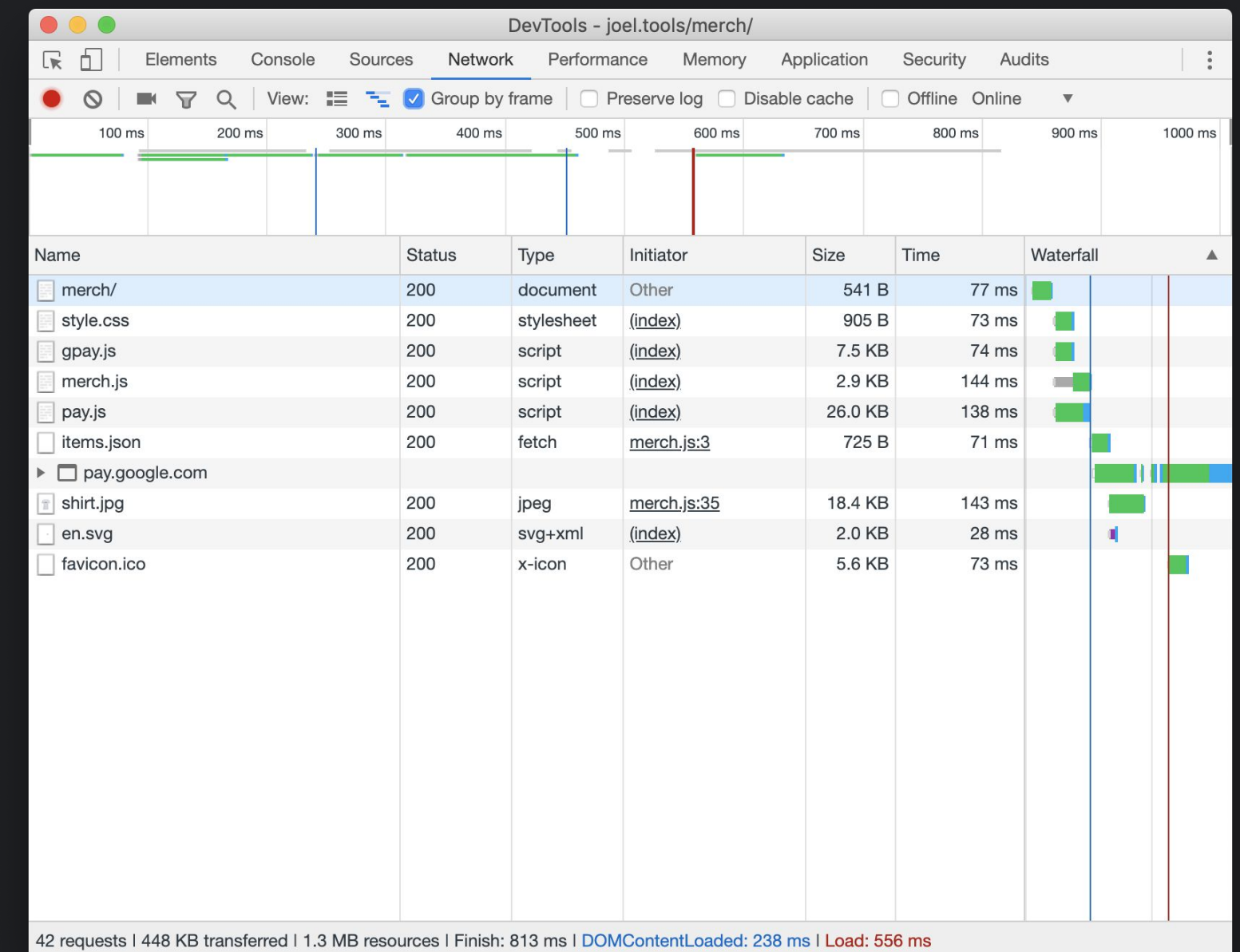
£40

# Network Monitoring



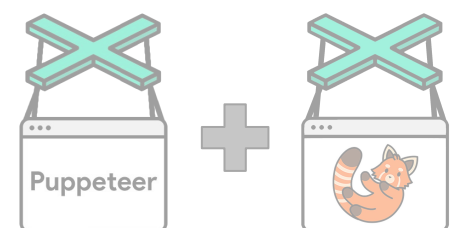
```
page.on('request', request => {
  console.log(request.method(), request.url());
});

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



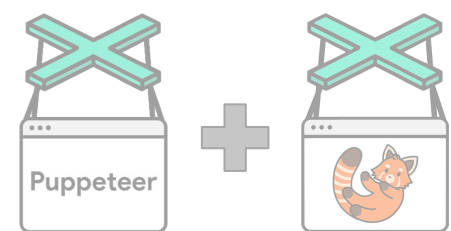
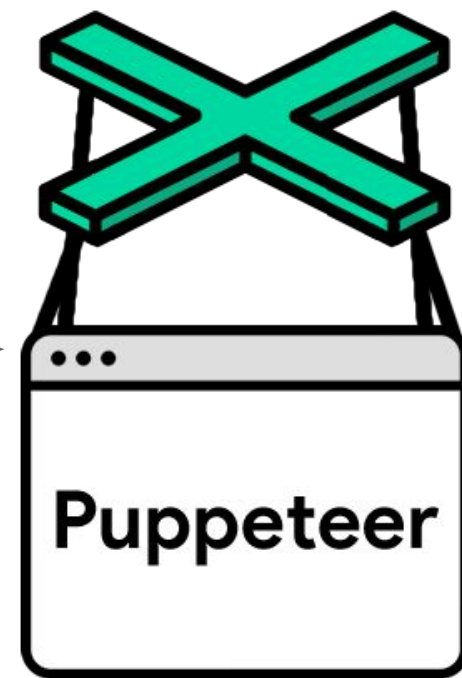
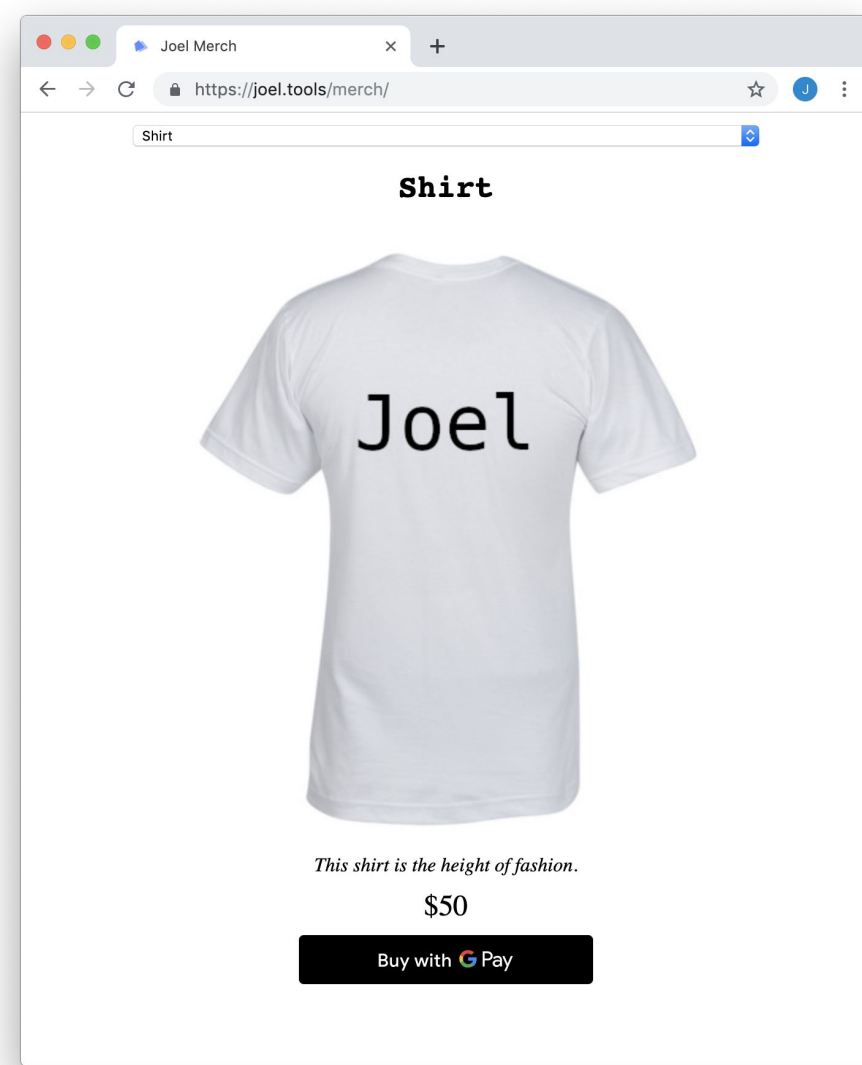
# Network Monitoring

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






# Request Interception



Google Developers x +

← → × <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](#)

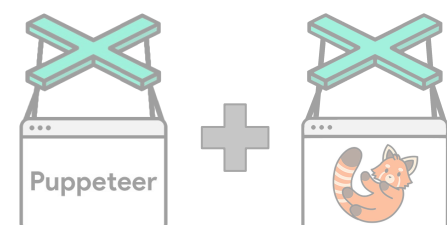


[Google I/O'19](#)  
May 7-9 | Mountain View, CA,  
USA

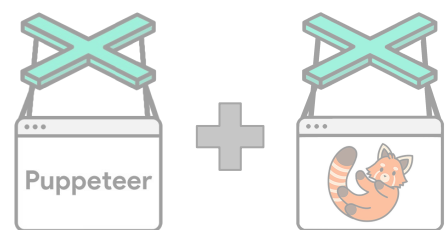
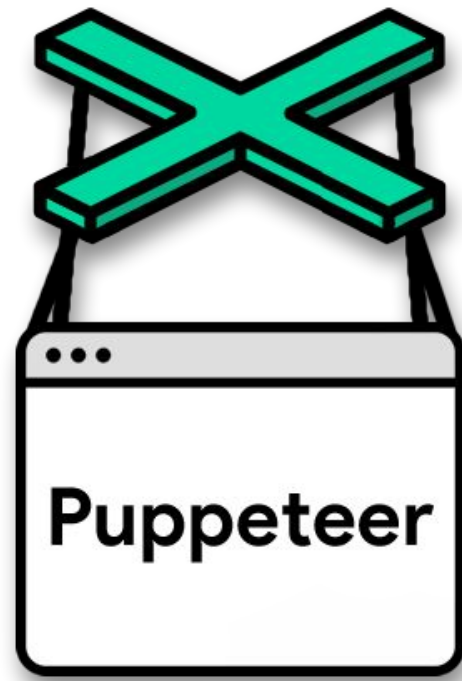


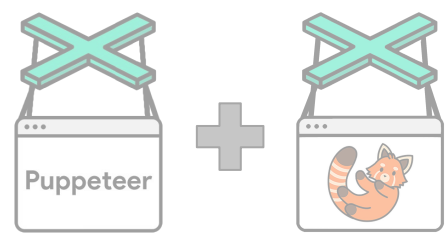
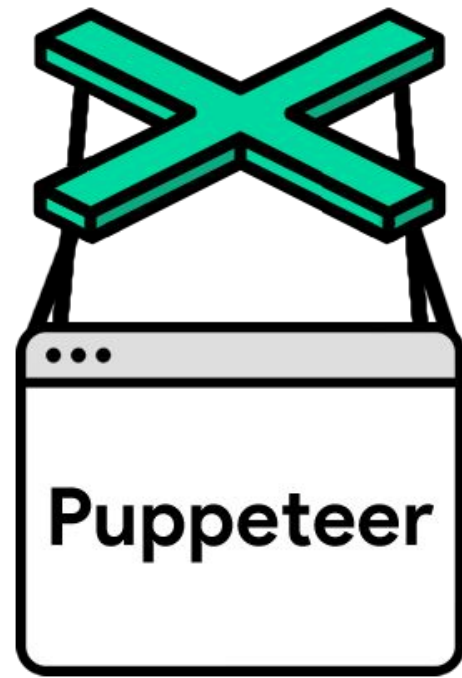
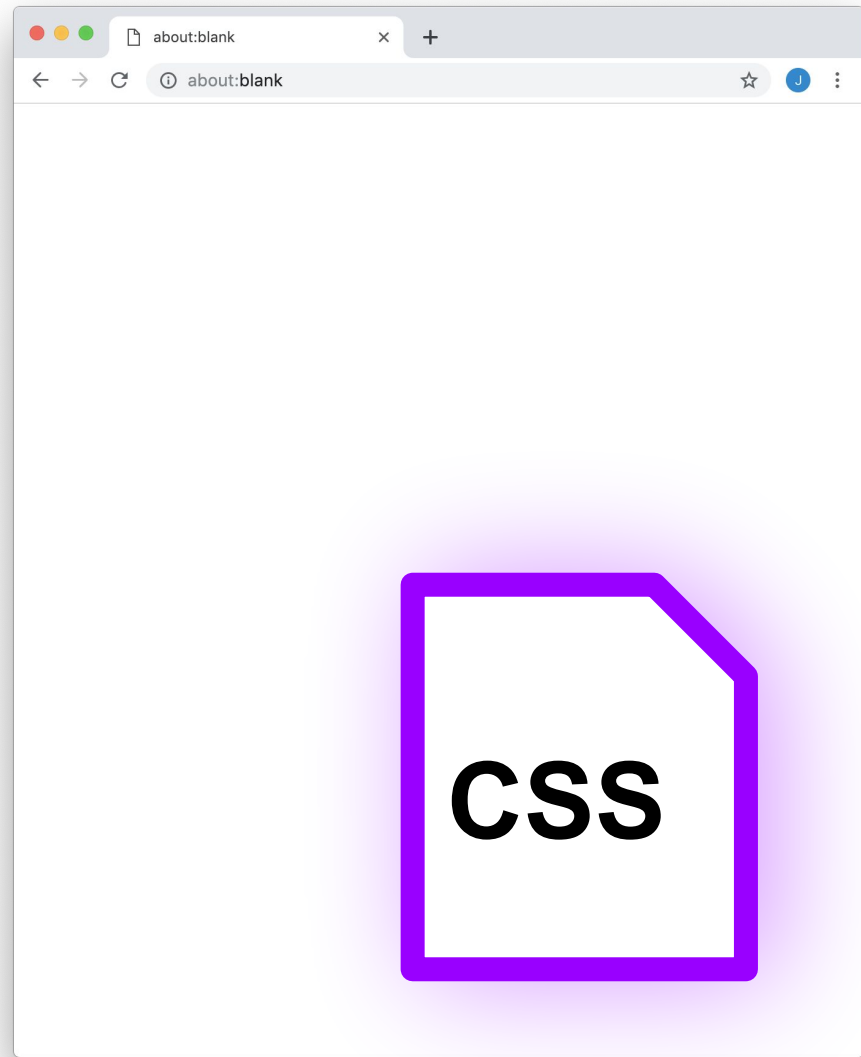
[Android Dev Summit 2019](#)  
October 23-24 | Sunnyvale, CA,  
USA

[https://events.google.com/io/?utm\\_source=devsite&utm\\_medium=events&utm\\_campaign=upcoming&utm\\_content=DevSite\\_HPP](https://events.google.com/io/?utm_source=devsite&utm_medium=events&utm_campaign=upcoming&utm_content=DevSite_HPP)









```
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 x +

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](#)


**Top news** [View all](#)



**Google brings AR and Lens closer to the future of search**

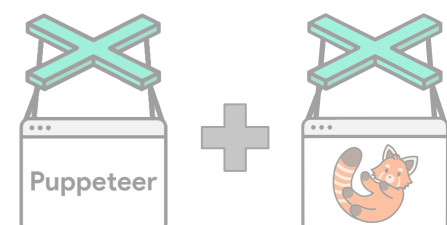
Exclusive: How Google's building the AR glue to a browsable reality.

 **CNET** May 07, 2019

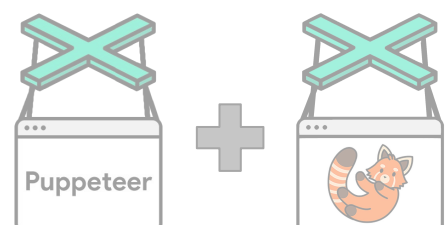
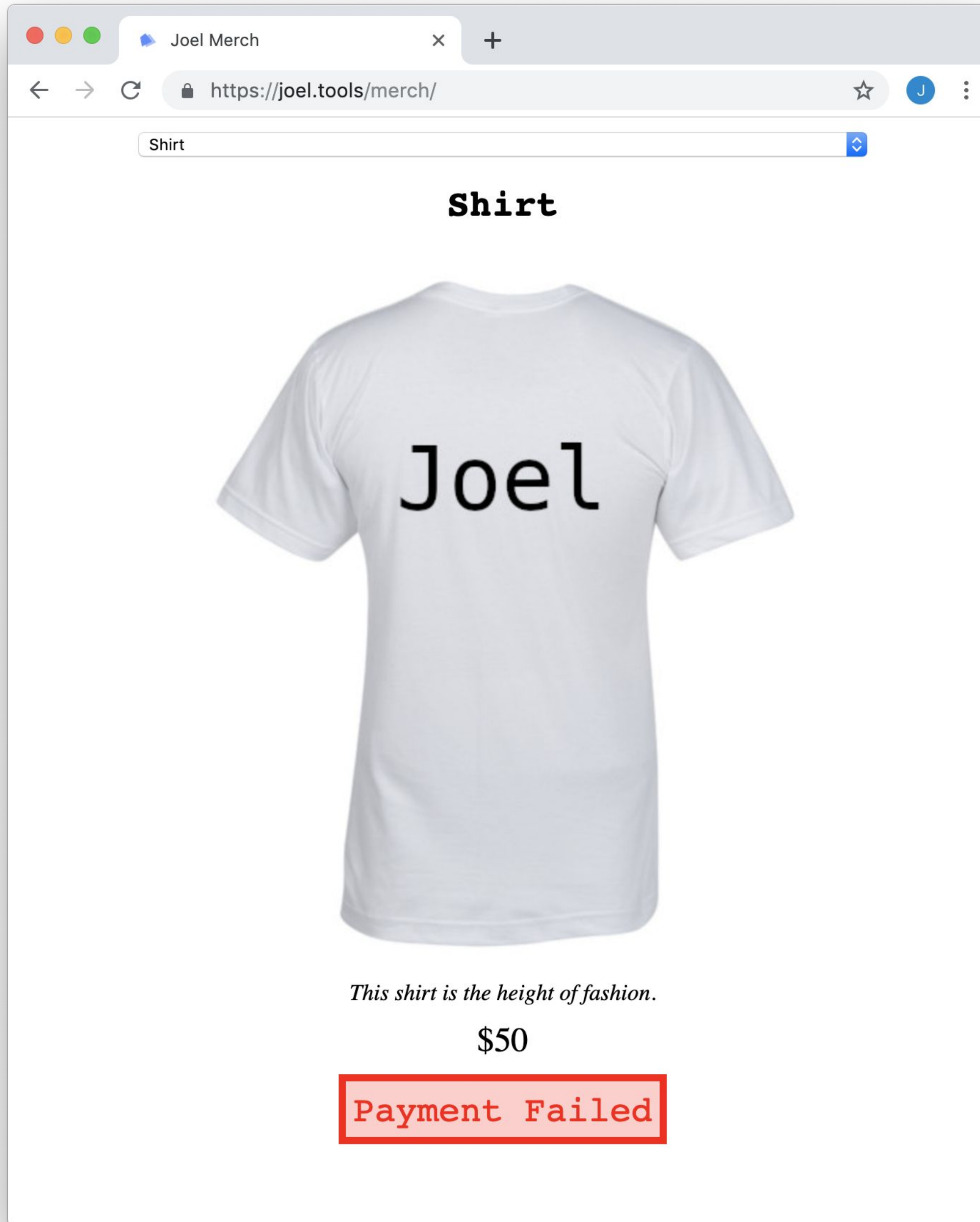
 **Lance Ulanoff** [@LanceUlanoff](#) · 3h

You know [@bts\\_bighit](#) has arrived when they're included in a [#GoogleIO2019](#) keynote. [#io19](#)

Waiting for twitter.com...











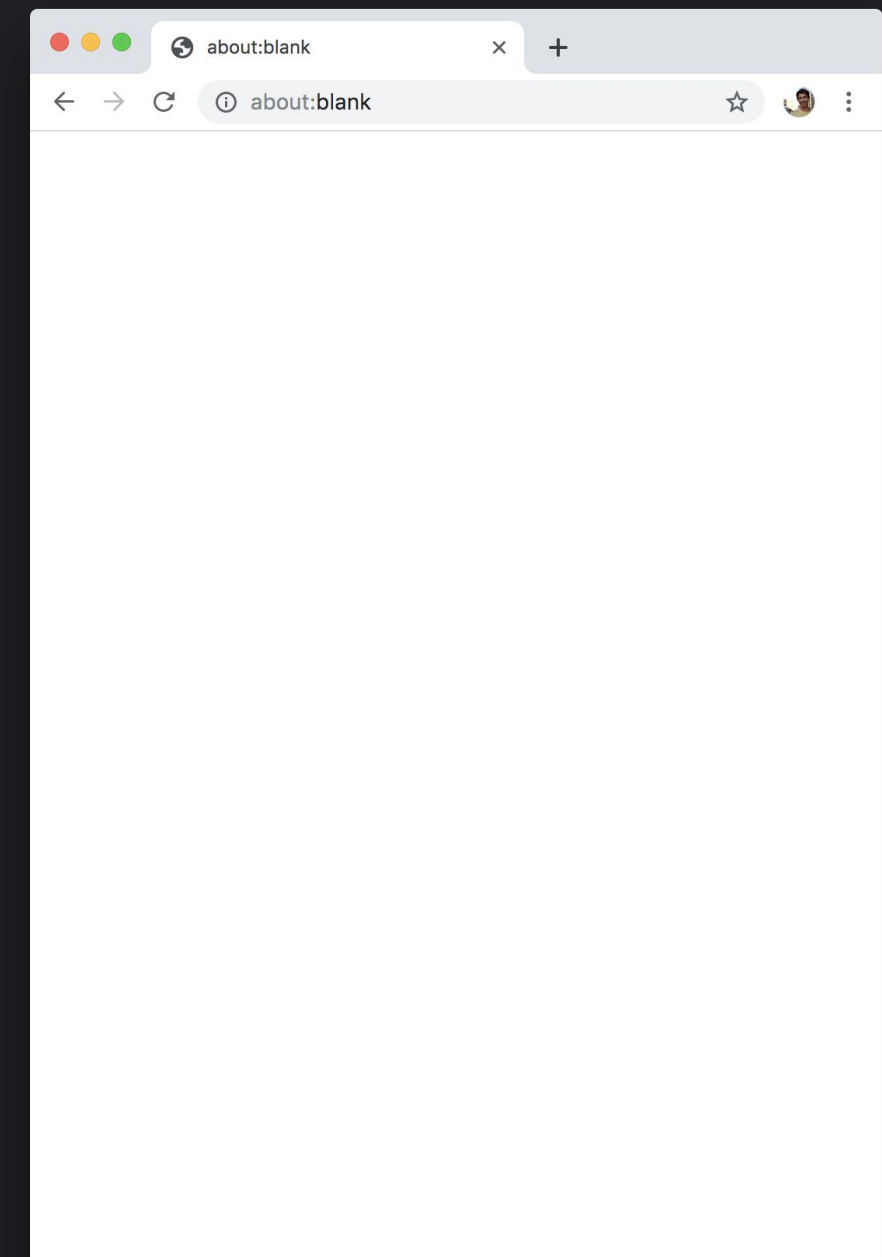
```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();
```

```
})
```

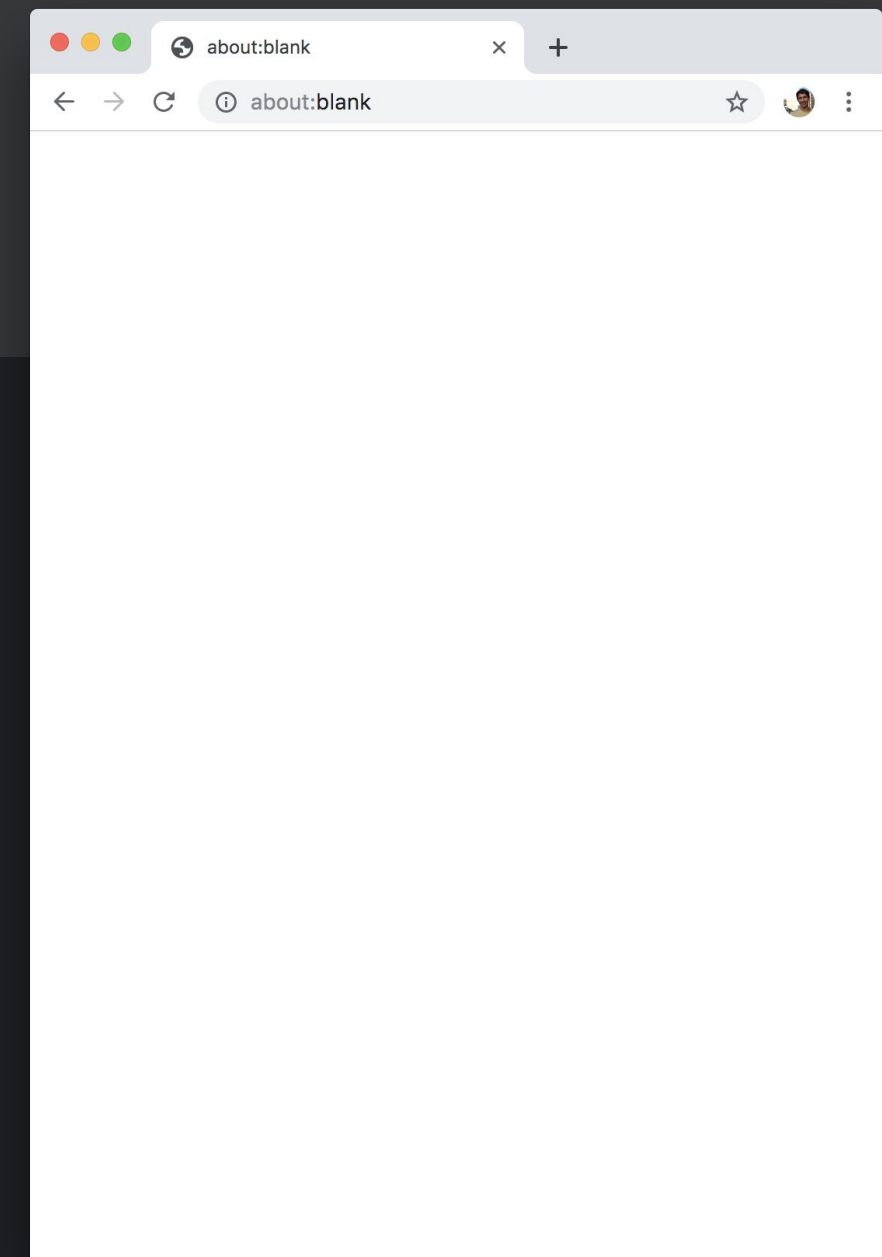


```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();  
  await page.setRequestInterception(true);
```

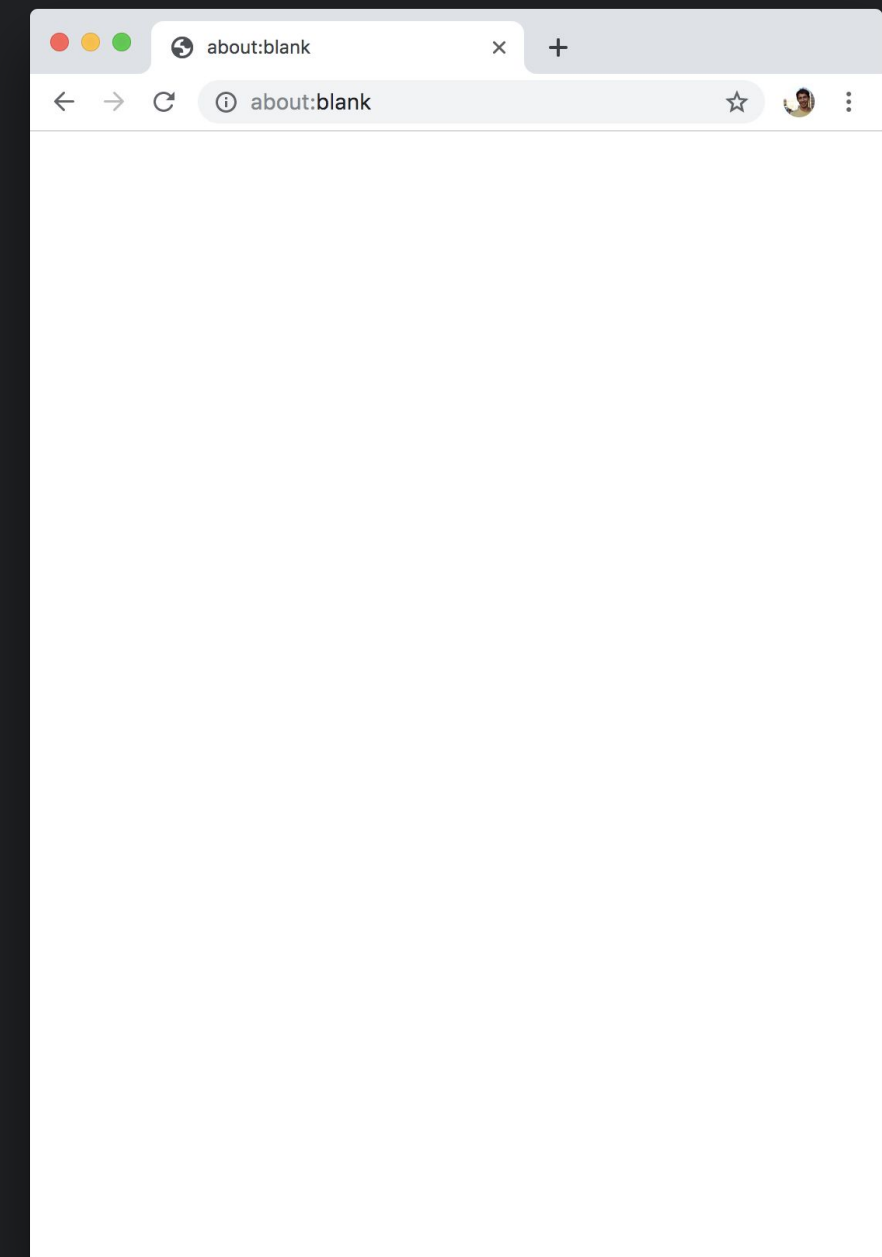
```
})
```



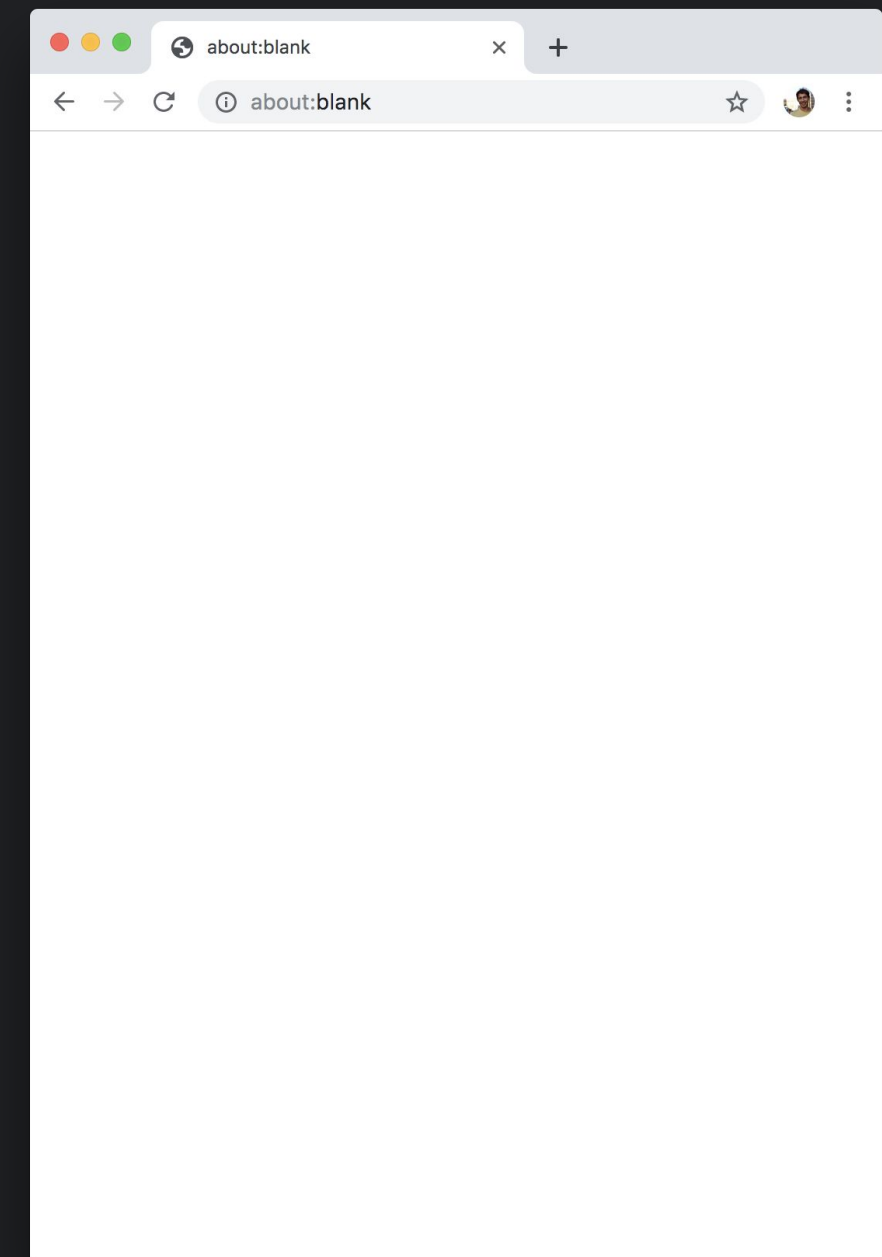
```
it('should notify when payment fails', async () => {  
  const page = await context.newPage();  
  await page.setRequestInterception(true);  
  page.on('request', request => {  
  
  });  
  
})
```



```
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')  
  
  });  
  
})
```

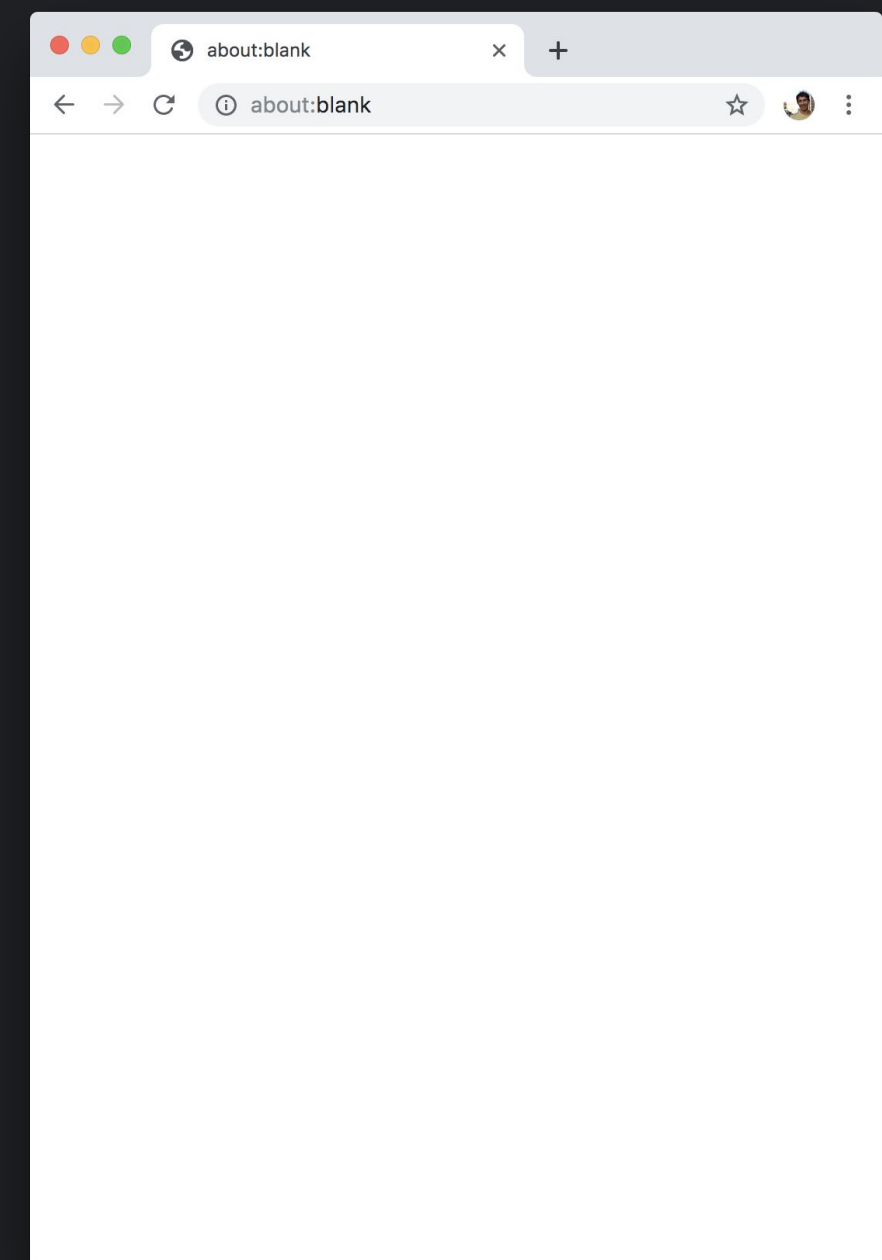


```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
  });
});
```

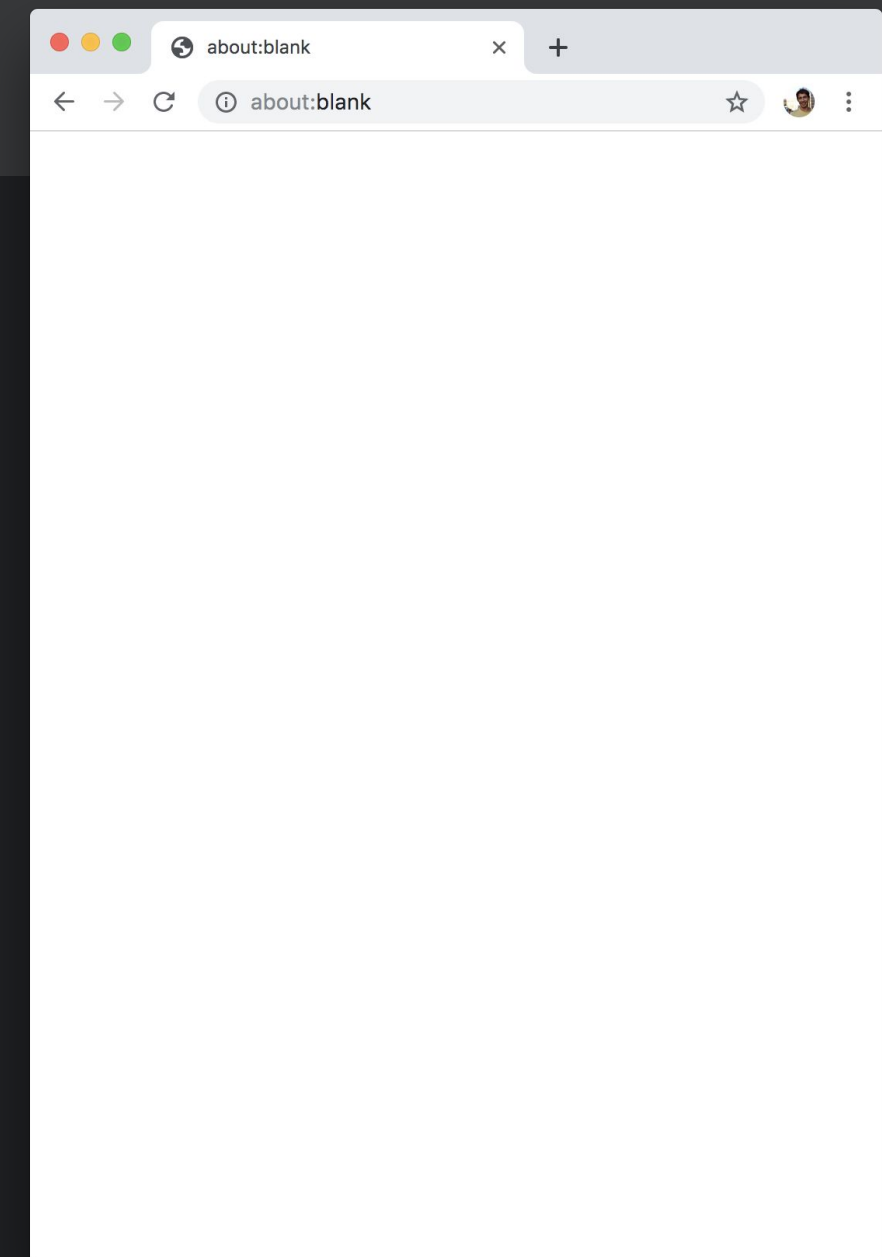




```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
  });
});
```



```
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');

  })
```



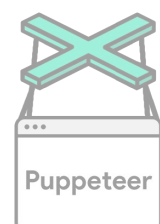
```
it('should notify when payment fails', async () => {
  const page = await context.newPage();
  await page.setRequestInterception(true);
  page.on('request', request => {
    if (request.url() === 'https://joel.tools/pay')
      request.abort();
    else
      request.continue();
  });

  await page.goto('https://joel.tools/merch');
  await page.click('button.gpay-button');
  await page.waitForSelector('.payment-failed');
})
```



# Performance Testing

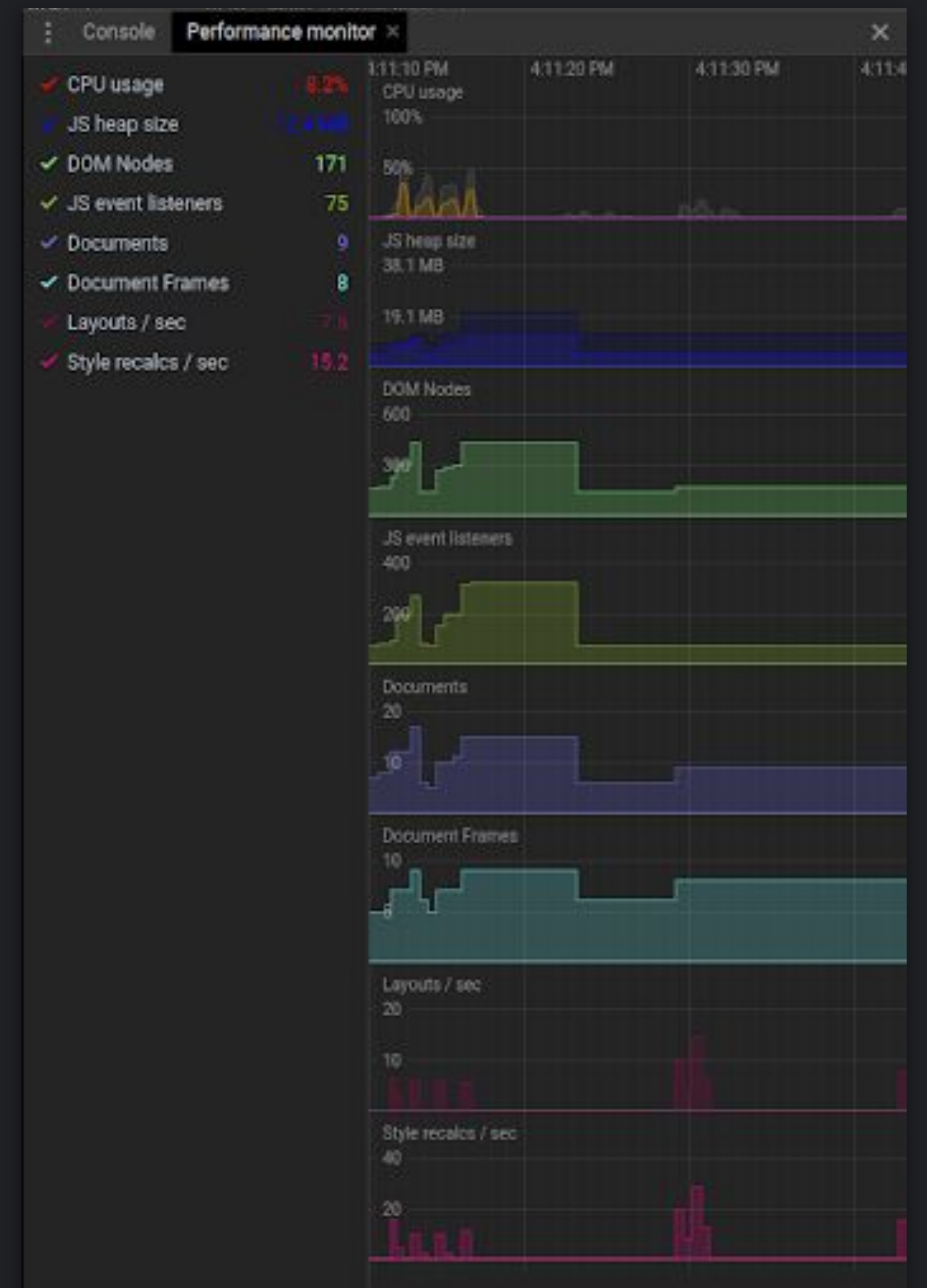
- 👉 Metrics
- 👉 Chrome Tracing
- 👉 Code Coverage



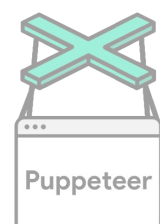
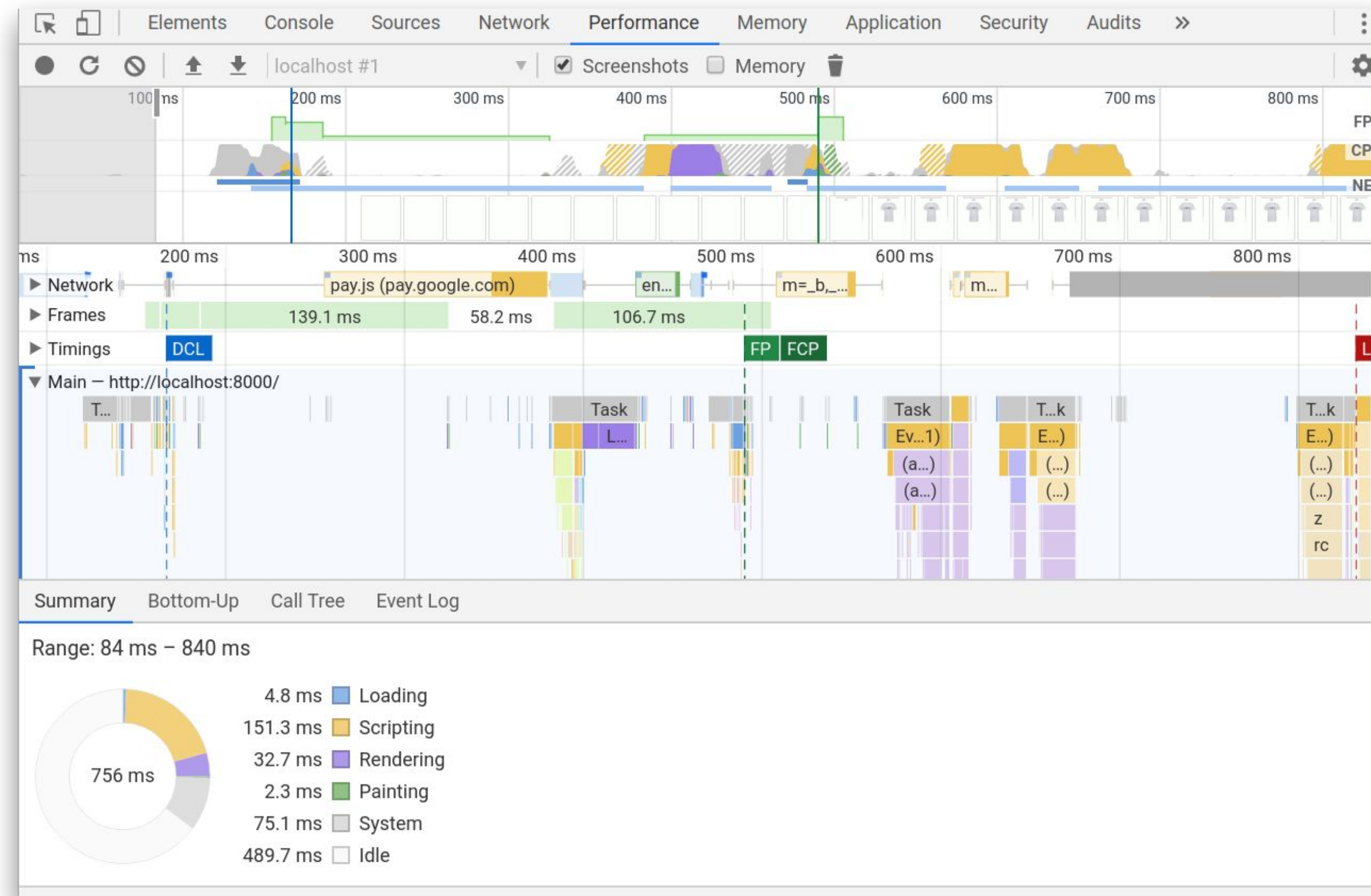


```
const metrics = await page.metrics();
console.log(metrics);
```

```
{ 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



```
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

```
11 margin: 0;  
12 }  
13 img {  
14   display: block;  
15   margin: auto;  
16   max-width: 100vmin;  
17   max-height: 100vmin;  
18 }  
19 button {  
20   max-width: 100vmin;  
21   margin: 0;  
22 }  
23 .out-of-stock {  
24   font-weight: bold;  
25   color: red;  
26   padding: 10px;  
27   font-size: 24px;  
28 }
```

URL	Type	Total Bytes	Unused Bytes	
https://pay.google.com/gp/p/js/pay.js	JS (coarse)	79 146	57 517 72.7 %	<div style="width: 72.7%; background-color: red;"></div>
https://joel.tools/merch/gpay.js	JS (coarse)	7 389	2 173 29.4 %	<div style="width: 29.4%; background-color: green;"></div>
https://joel.tools/merch/merch.js	JS (coarse)	2 710	829 30.6 %	<div style="width: 30.6%; background-color: green;"></div>
https://joel.tools/merch/style.css	CSS	659	263 39.9 %	<div style="width: 39.9%; background-color: green;"></div>
https://joel.tools/merch/	JS	23	0 0 %	

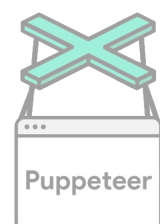
59.4 KB of 87.8 KB bytes are not used. (68%)





# 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: 'Joe1 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: 'Joe1 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: 'Joe1 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);
```

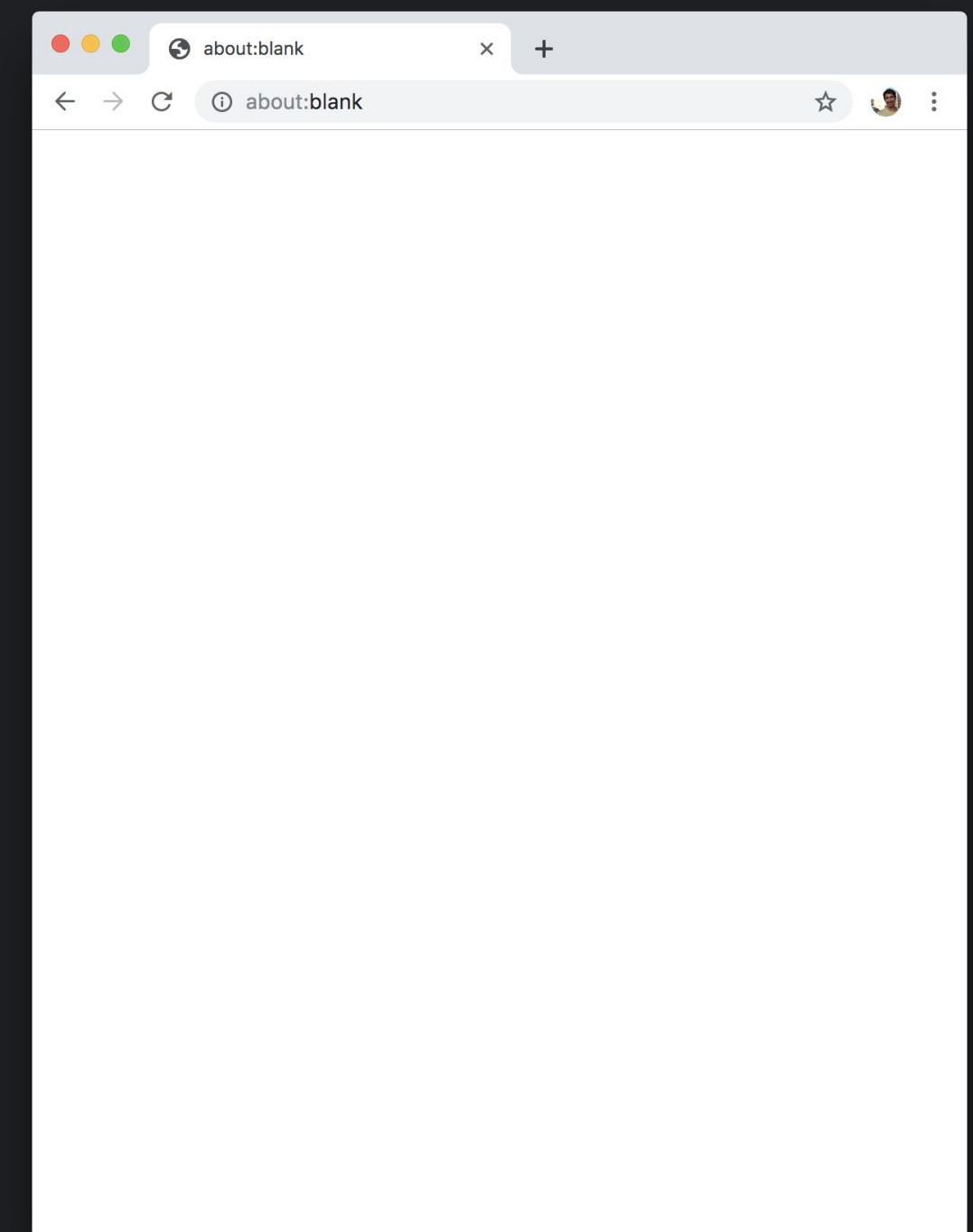
```
{ role: 'WebArea',
  name: 'Joe1 Merch',
  children: [
    { role: 'combobox', name: '', value: 'Shirt', children: [Array] },
    { role: 'heading', name: 'Shirt', level: 1 },
    { role: 'img', name: 'Shirt' },
    { role: 'text', name: 'This shirt is the height of fashion.' },
    { role: 'text', 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();
```

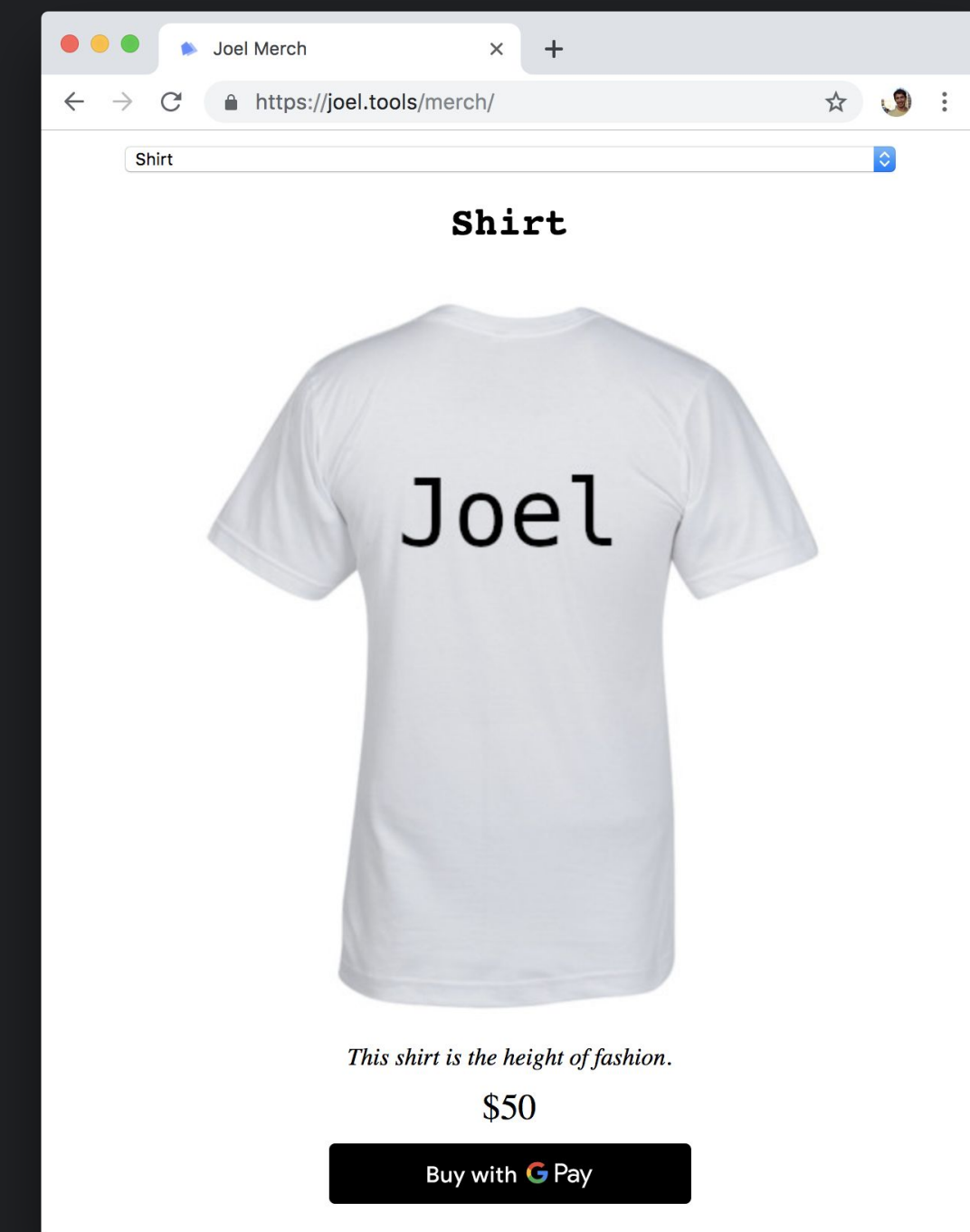
```
});
```



```
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');  
  
});
```



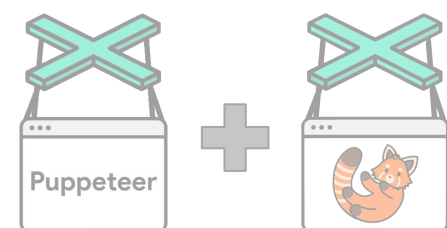
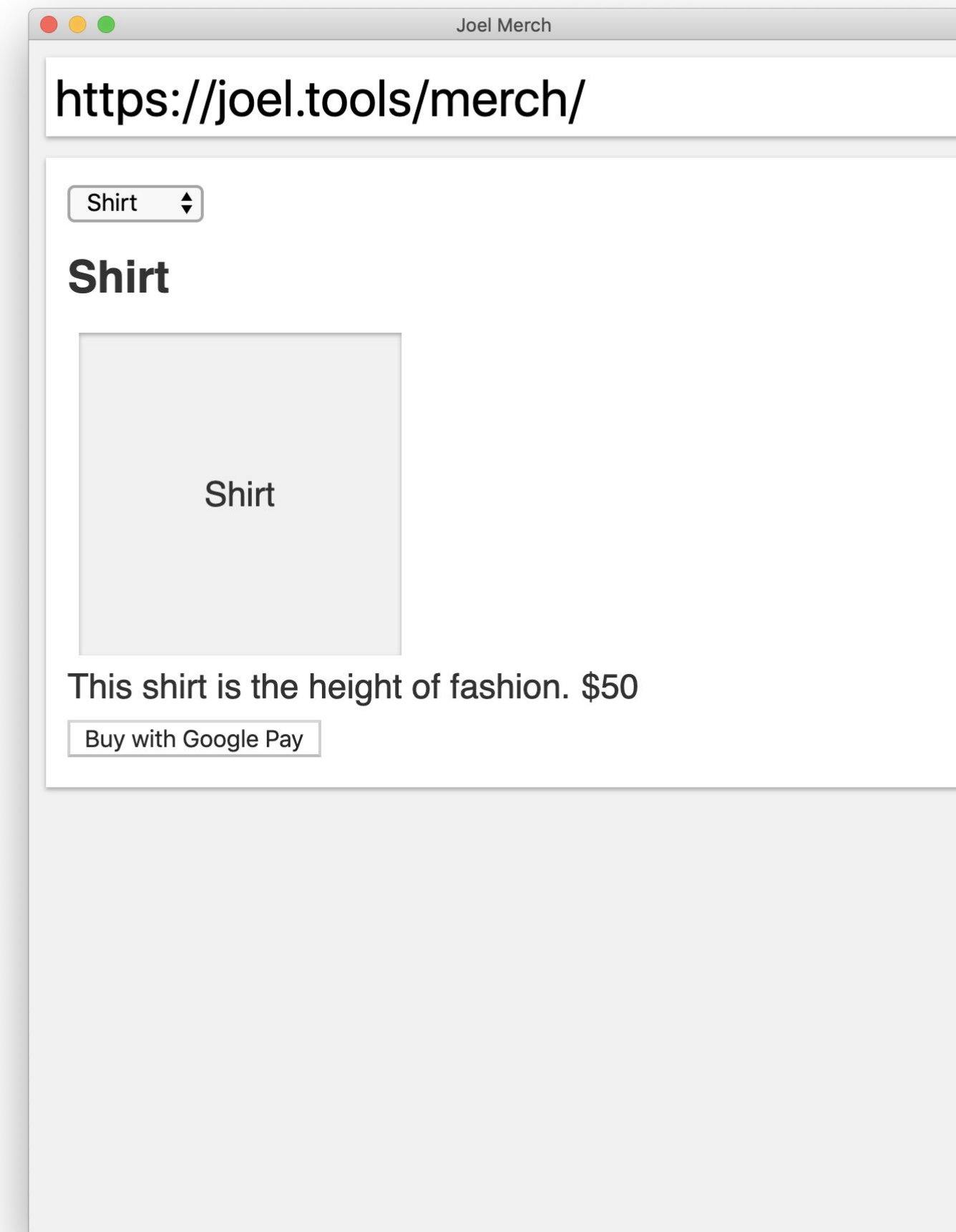
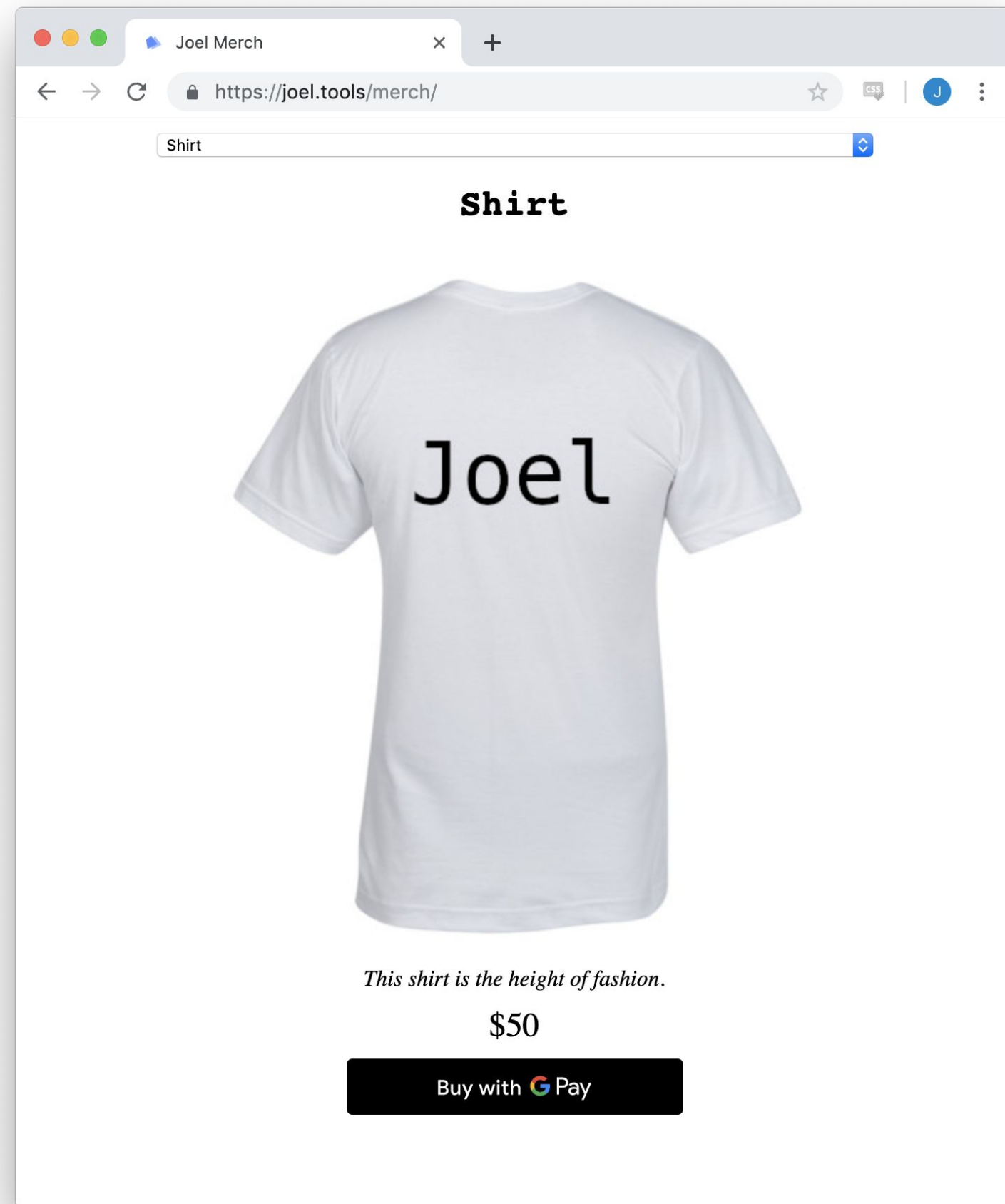
```
it('should have a well named pay button', async () => {  
  const page = await context.newPage();  
  await page.goto('https://joel.tools/merch');  
  const button = await page.$('button');  
  const buttonAX = await page.accessibility.snapshot({  
    root: button,  
  });  
  
});
```

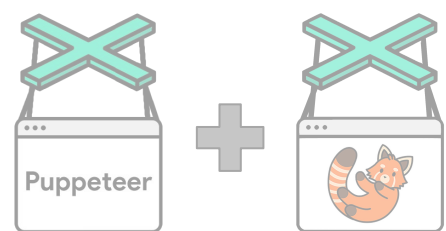
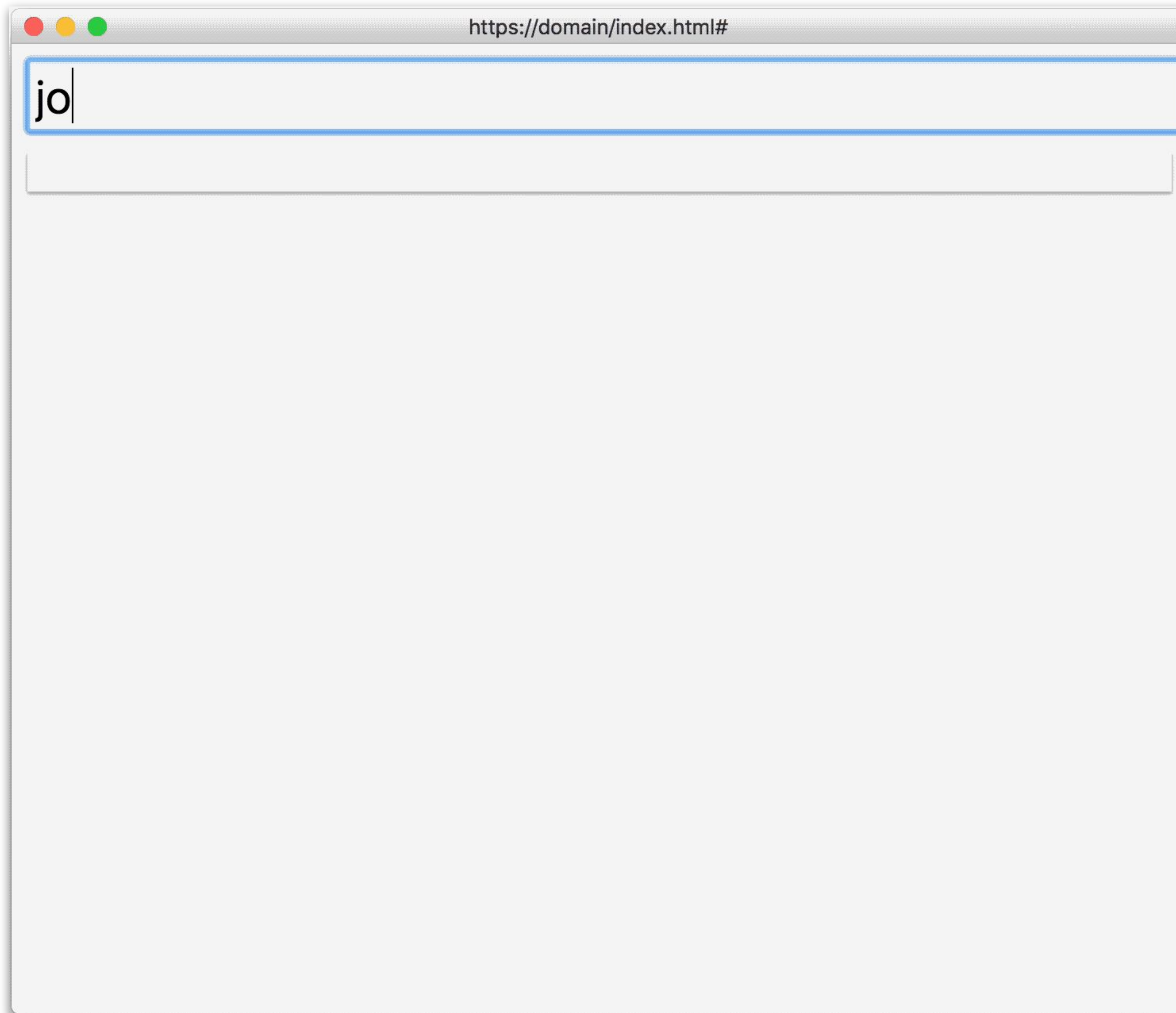


```
it('should have a well named pay button', async () => {
  const page = await context.newPage();
  await page.goto('https://joel.tools/merch');
  const button = await page.$('button');
  const buttonAX = await page.accessibility.snapshot({
    root: button,
  });
  expect(buttonAX.name).toBe('Buy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





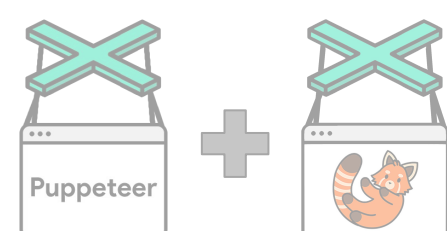


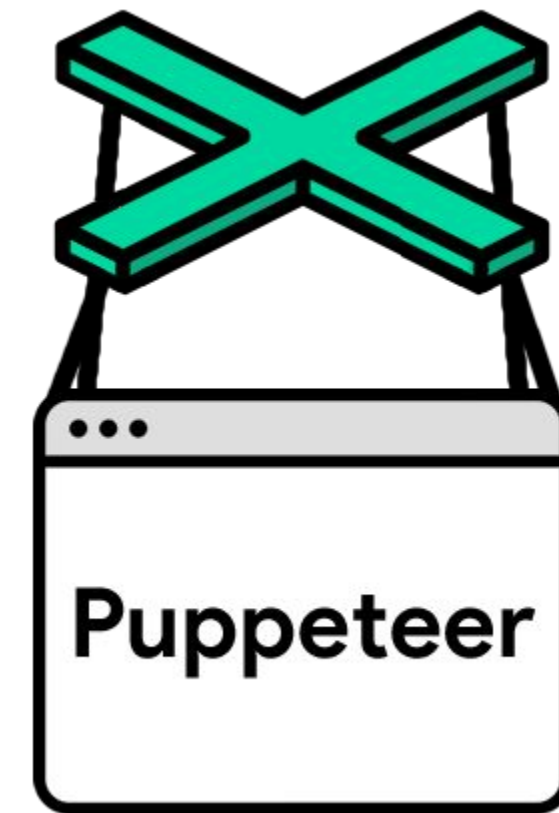




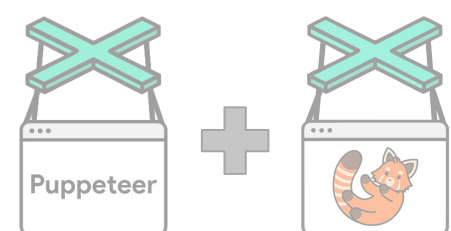
# Accessibility Visualizer

👉 [github.com/GoogleChromeLabs/html-a11y-renderer](https://github.com/GoogleChromeLabs/html-a11y-renderer)





[github.com/GoogleChrome/puppeteer](https://github.com/GoogleChrome/puppeteer)



Google™

NETFLIX



HEISENBUG

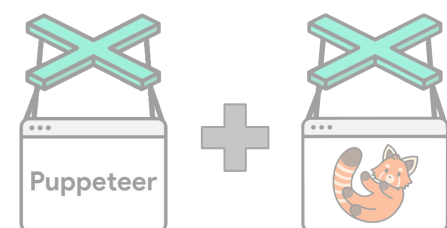
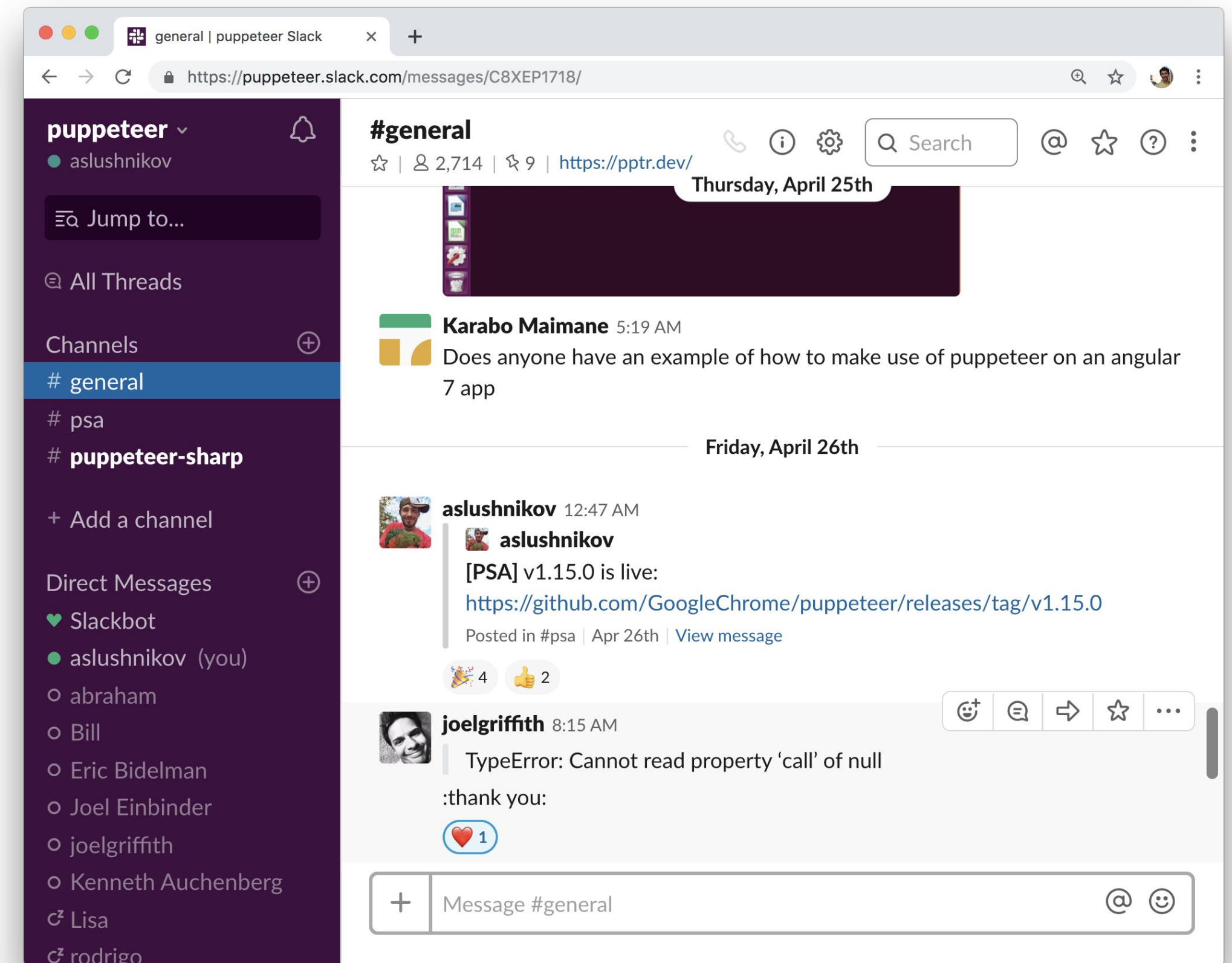




# Community



👉 2700+ folks on [puppeteer.slack.com](https://puppeteer.slack.com)





# Community



👉 100+ people contributed to api docs

puppeteer/api.md at v1.15.0 · GitHub

Code Issues 348 Pull requests 16 Pulse Community

Tag: v1.15.0 puppeteer / docs / api.md Find file Copy path

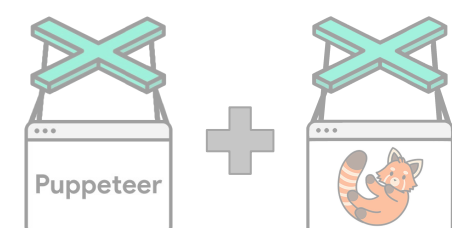
aslushnikov chore: mark version v1.15.0 (#4346) b9f4a95 10 days ago

110 contributors

3636 lines (2806 sloc) 175 KB Raw Blame History

## 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](#)

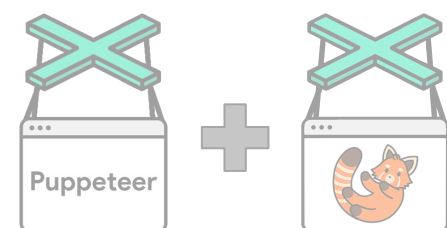


# Community



👉 20% of patches to `//lib` are community c

The screenshot shows a GitHub pull request page for the repository 'GoogleChrome / puppeteer'. The pull request title is 'Add .content() method. Fixes #406. #419'. It is marked as 'Merged' and was merged by 'aslushnikov' on August 21, 2017. The pull request includes 1 commit and 3 files changed, with a net change of +36 lines and -2 lines. The pull request description 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 1 reaction (👍) and 2 reactions (❤️). The pull request is reviewed by 'aslushnikov' with a green checkmark. The pull request is assigned to no one. The pull request has no labels, projects, or milestones. The pull request has 5 participants. The pull request is a response to a request from 'googlebot' on August 20, 2017, which states: 'Thanks for your pull request. It looks like this may be your first contribution to a Google open source project. Before we can look at your pull request, you'll need to sign a Contributor License Agreement (CLA). Please visit <https://cla.developers.google.com/> to sign. Once you've signed, please reply here (e.g. I signed it!) and we'll verify. Thanks. If you've already signed a CLA, it's possible we don't have your GitHub username or you're using a





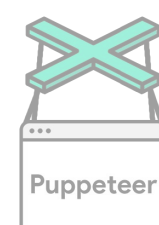
```
ai@blackjack:~/Dev/my-js-lib
~/Dev/my-js-lib
→ npx size-limit

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](https://github.com/ai/size-limit)

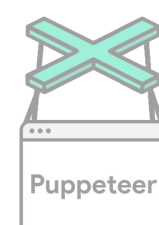


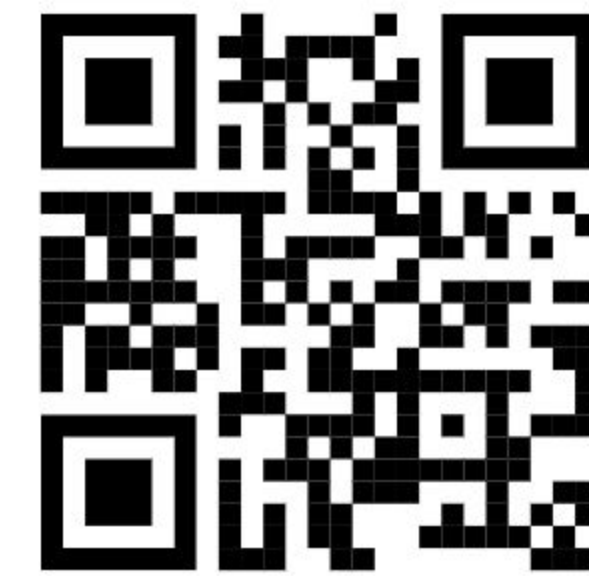




Browser window showing the GitHub repository page for `pocketjoso/penthouse`. The page title is "penthouse" and it is described as a "Critical Path CSS Generator". The page shows the current version as 2.1.0, with a passing build status and 91k/month downloads. The "About" section explains that Penthouse is the original critical path CSS generator, helping to speed up page rendering by returning the critical CSS needed to perfectly render the above the fold content of the page. The process is automatic and the generated CSS is production ready. Behind the scenes, Penthouse uses `puppeteer` to generate the critical CSS via `chromium:headless`. The "Usage" section provides the command to install the package: `yarn add --dev penthouse` (or `npm install` if not using `yarn`).

[github.com/pocketjoso/penthouse](https://github.com/pocketjoso/penthouse)





API monitoring and Site Transa X +

checklyhq.com

**Checkly**

# A better way to monitor your APIs and site click flows

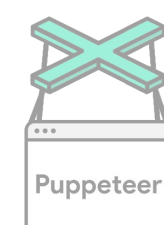
Monitor the performance and correctness of your API endpoints & vital site transactions from a single, simple dashboard.

[Start your free trial](#)

5 PASSING 1 FAILING

STATUS	NAME	TYPE	LAST RESULTS	24H RATIO	AVG. RESP.	SCHED.
✓	Add to shopping cart 4 minutes ago • commerce			100%	2.1 sec	5
✓	Customer signup form 40 minutes ago			100%	2.68 sec	60

checklyhq.com





GitHub - checkly/puppeteer-re X +

GitHub, Inc. [US] | github.com/checkly/puppeteer-recorder

README.md

# Puppeteer Recorder

build passing

Buy me a coffee

1. Click record
2. Track recorded events
3. Generate a Puppeteer script

The screenshot shows three sequential screenshots of the Puppeteer Recorder application. The first shows a 'Record' button and 'No recorded events yet'. The second shows a 'recording' status and a list of events: 1. click (https://github.com/checkly/puppeteer-recorder), 2. click (.focus > .form-control), 3. keydown (.focus > .form-control), 4. keydown (.jump-to-dropdown-visible), and 5. keydown (.jump-to-dropdown-visible). The third shows the generated Puppeteer script and 'Restart' and 'copy to clipboard' buttons.

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch()
  const page = await browser.newPage()
  await page.goto('https://github.com/checkly')
  await page.click('.focus > .form-control')
  await page.type('.jump-to-dropdown-visible')
  await page.click('nth-child(3) > .js-sele')
  await browser.close()
})();
```

[github.com/checkly/puppeteer-recorder](https://github.com/checkly/puppeteer-recorder)







browserless: The headless bro x +

browserless.io

browserless

# Simple browser automation built for developers.

Fast, scalable, and reliable browser automation.

[SIGN-UP](#) or [Check out our live debugger](#)

```
// Replace puppeteer.launch with puppeteer.connect
const browser = await puppeteer.connect({
  browserWSEndpoint: 'wss://chrome.browserless.io/'
});

// Everything else stays the same
const page = await browser.newPage();
await page.goto('https://example.com/');
await page.screenshot({ path: 'screenshot.png' });
browser.close();
```

# browserless.io



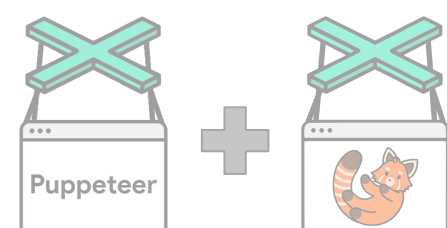
HEISENBUG

# Community



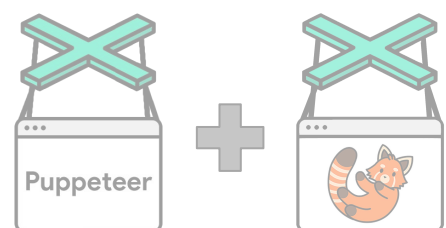
👉 1700+ packages on NPM

The screenshot shows the NPM package page for puppeteer. The page includes a search bar, navigation links, and package details. The package name is puppeteer, version 1.15.0, published 10 days ago. It has 8 dependencies, 1,715 dependents, and 623 versions. The page lists several dependents, 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. The page also shows the install command, weekly downloads (859,045), version (1.15.0), license (Apache-2.0), open issues (348), and pull requests (16).



# TL;DR

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



# Thank you!



Andrey Lushnikov

Google, @aslushnikov