

Headless Chrome & Cloudinary for progressively enhanced dynamic content on the web

Ire_Aderinokun

JAMstack_conf_sf 2019

73% of websites in 2006 were
reliant on Javascript for their
core functionality

🙄 Performance

🙄 Accessibility

🙄 Progressive Enhancement

We can use Javascript and the
JAMstack to improve Javascript

🖥️ **Frontend Developer &**
Cofounder, BuyCoins

🇳🇬 **Lagos**, Nigeria

Google Web Expert &
Clouinary Media Expert

Writer, **bitsofco.de**



bitsofco.de

Support and progressive enhancement

As I mentioned, the specification for CSS environment variables is still in Editor's Draft, which means things will probably change. Despite this, the support for this feature is already at 70% and therefore can be used.

CSS Environment Variables env()

Usage of environment variables like ``safe-area-inset-top``.

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Opera Mini	Chrome for Android	Android Browser	Samsung Internet
9	16	67	75	11.1	11.4			4.4	7.4
10	17	68	76	12	12.1			4.4.4	8.2
11	18	69	77	12.1	12.3	all	76	76	9.2
	76	70	78	13	13				

  Partial Support

Global: 80.18% + 0.27% = 80.45%

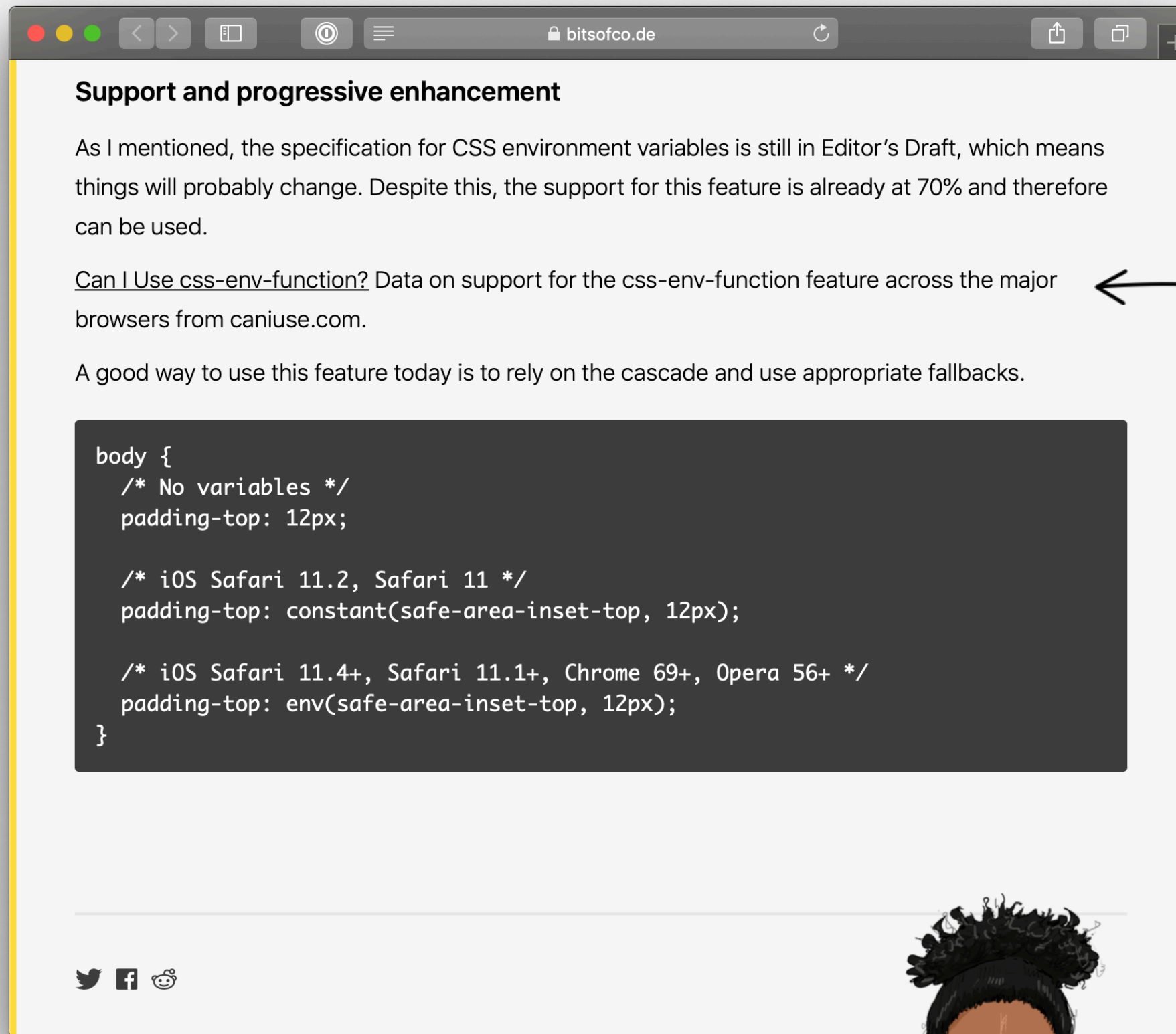
Data from caniuse.com | Embed from caniuse.bitsofco.de

Enable accessible colours

A good way to use this feature today is to rely on the cascade and use appropriate fallbacks.

```
body {  
  /* No variables */  
  ...  
}
```

```
<p class="ciu_embed" data-feature="flexbox" data-periods="future_1,current,past_1">
  <a href="http://caniuse.com/#feat=flexbox">
    Can I Use flexbox?
  </a>
  Data on support for the flexbox feature across the major browsers from
  caniuse.com.
</p>
```



Support and progressive enhancement

As I mentioned, the specification for CSS environment variables is still in Editor's Draft, which means things will probably change. Despite this, the support for this feature is already at 70% and therefore can be used.

[Can I Use css-env-function?](#) Data on support for the css-env-function feature across the major browsers from caniuse.com.

A good way to use this feature today is to rely on the cascade and use appropriate fallbacks.

```
body {  
  /* No variables */  
  padding-top: 12px;  
  
  /* iOS Safari 11.2, Safari 11 */  
  padding-top: constant(safe-area-inset-top, 12px);  
  
  /* iOS Safari 11.4+, Safari 11.1+, Chrome 69+, Opera 56+ */  
  padding-top: env(safe-area-inset-top, 12px);  
}
```

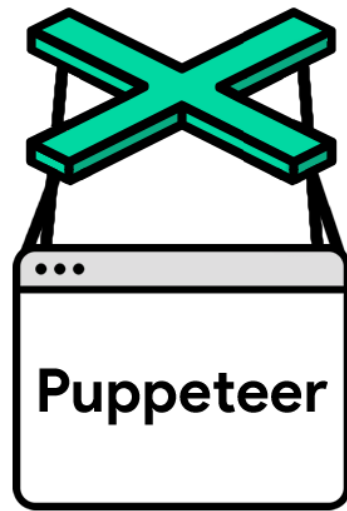


😞 Javascript disabled

😞 Slow connections

😞 Emails, READMEs, etc



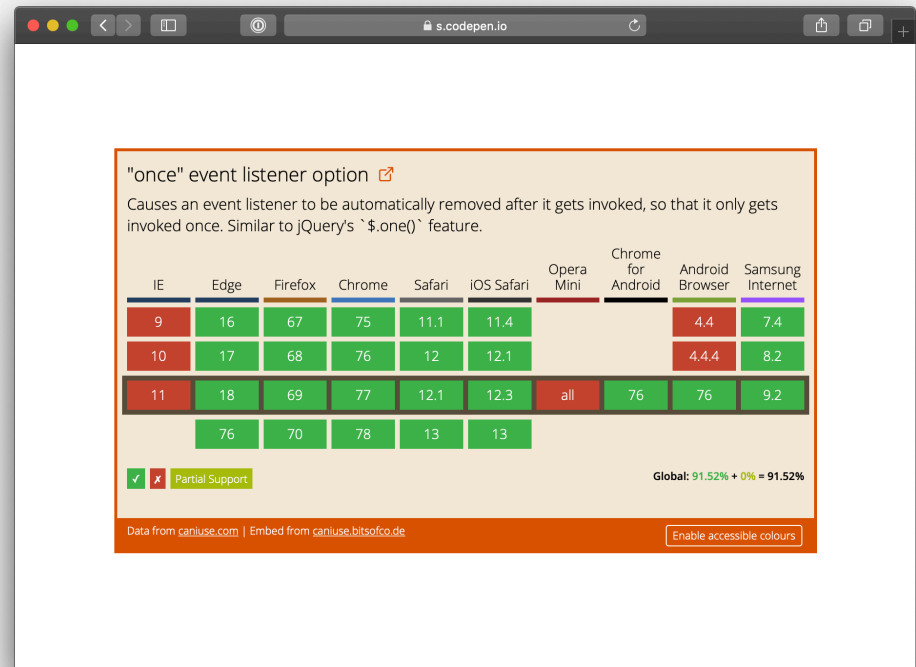
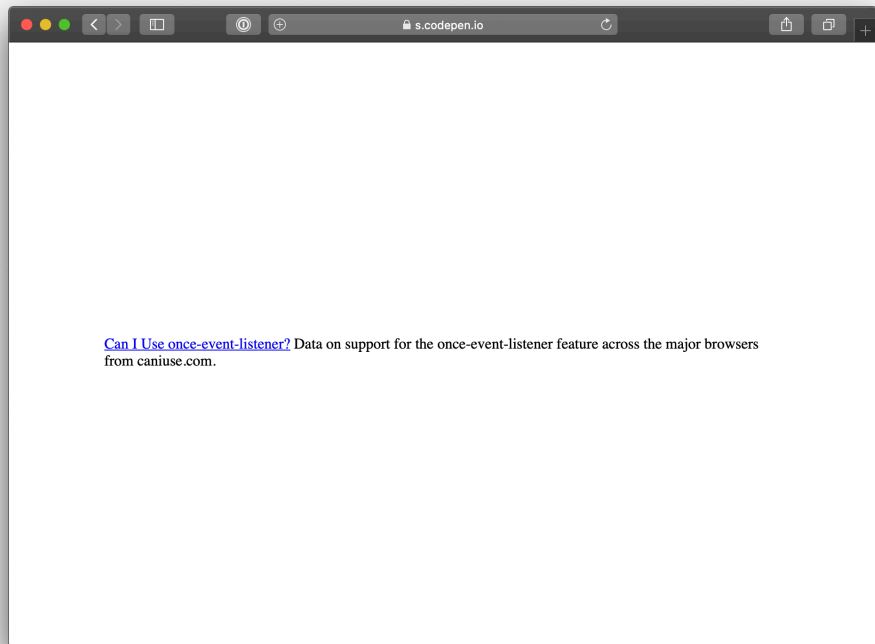




HTML Page



Screenshot





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

const browser = await puppeteer.launch({
  defaultViewport: {
    width: 800,
    height: 800,
    isLandscape: true
  }
});

const page = await browser.newPage();

await page.goto(
  embedUrl,
  { waitUntil: 'networkidle2' }
);

const screenshot = await page.screenshot({
  omitBackground: true,
  encoding: 'binary'
});

await browser.close();
```

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

const browser = await puppeteer.launch({
  defaultViewport: {
    width: 800,
    height: 800,
    isLandscape: true
  }
});
```

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

```
await page.goto(
  embedUrl,
  { waitUntil: 'networkidle2' }
);
```

```
const screenshot = await page.screenshot({
  omitBackground: true,
  encoding: 'binary'
});
```

```
await browser.close();
```

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

const browser = await puppeteer.launch({
  defaultViewport: {
    width: 800,
    height: 800,
    isLandscape: true
  }
});
```

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

await page.goto(
  embedUrl,
  { waitUntil: 'networkidle2' }
);
```

```
const screenshot = await page.screenshot({
  omitBackground: true,
  encoding: 'binary'
});
```

```
await browser.close();
```

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

const browser = await puppeteer.launch({
  defaultViewport: {
    width: 800,
    height: 800,
    isLandscape: true
  }
});

const page = await browser.newPage();

await page.goto(
  embedUrl,
  { waitUntil: 'networkidle2' }
);
```

```
const screenshot = await page.screenshot({
  omitBackground: true,
  encoding: 'binary'
});

await browser.close();
```



```
const cloudinary = require('cloudinary').v2;
cloudinary.config(/* config here */);

cloudinary.uploader.upload_stream((error, result) => {
  if (error) reject(error)

  return result.secure_url;
}).end(screenshot);
```

```
const cloudinary = require('cloudinary').v2;  
cloudinary.config(/* config here */);
```

```
cloudinary.uploader.upload_stream((error, result) => {  
  if (error) reject(error)  
  
  return result.secure_url;  
  
}).end(screenshot);
```

```
const cloudinary = require('cloudinary').v2;  
cloudinary.config(/* config here */);
```

```
cloudinary.uploader.upload_stream((error, result) => {  
  if (error) reject(error)  
  
  return result.secure_url;  

```

```
}).end(screenshot);
```

```
const cloudinary = require('cloudinary').v2;  
cloudinary.config(/* config here */);  
  
cloudinary.uploader.upload_stream((error, result) => {  
  if (error) reject(error)  
  
  return result.secure_url;  
  
}).end(screenshot);
```

```
<p class="ciu_embed" data-feature="flexbox" data-periods="future_1,current,past_1">  
  <a href="http://caniuse.com/#feat=flexbox">  
    Can I Use flexbox?  
  </a>  
  Data on support for the flexbox feature across the major browsers from caniuse.com.  
</p>
```



```
<p class="ciu_embed" data-feature="flexbox" data-periods="future_1,current,past_1">  
  <a href="http://caniuse.com/#feat=flexbox">  
      
  </a>  
</p>
```



```
<p class="ciu_embed" data-feature="flexbox" data-periods="future_1,current,past_1">
  <a href="http://caniuse.com/#feat=flexbox">
    <picture>
      <source type="image/webp" srcset="https://res.cloudinary.com/
ireaderinokun/image/upload/v1571256076/caniuse-embed/flexbox-2019-10-16.webp">
      <source type="image/png" srcset="https://res.cloudinary.com/
ireaderinokun/image/upload/v1571256076/caniuse-embed/flexbox-2019-10-16.png">
      <source type="image/jpeg" srcset="https://res.cloudinary.com/
ireaderinokun/image/upload/v1571256076/caniuse-embed/flexbox-2019-10-16.jpg">
      
    </picture>
  </a>
</p>
```



Support and progressive enhancement

As I mentioned, the specification for CSS environment variables is still in Editor's Draft, which means things will probably change. Despite this, the support for this feature is already at 70% and therefore can be used.

CSS Environment Variables `env()` [↗](#)

Usage of environment variables like ``safe-area-inset-top``.

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Opera Mini	Chrome for Android	Android Browser	Samsung Internet
9	16	61	68	11	11.2			4.4	5
10	17	62	69	11.1	11.4			4.4.4	6.2
11	18	63	70	12	12	all	70	67	7.2
		64	71	TP					

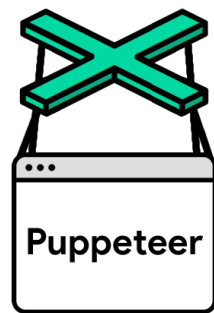
✓ ✗ ? Partial Support

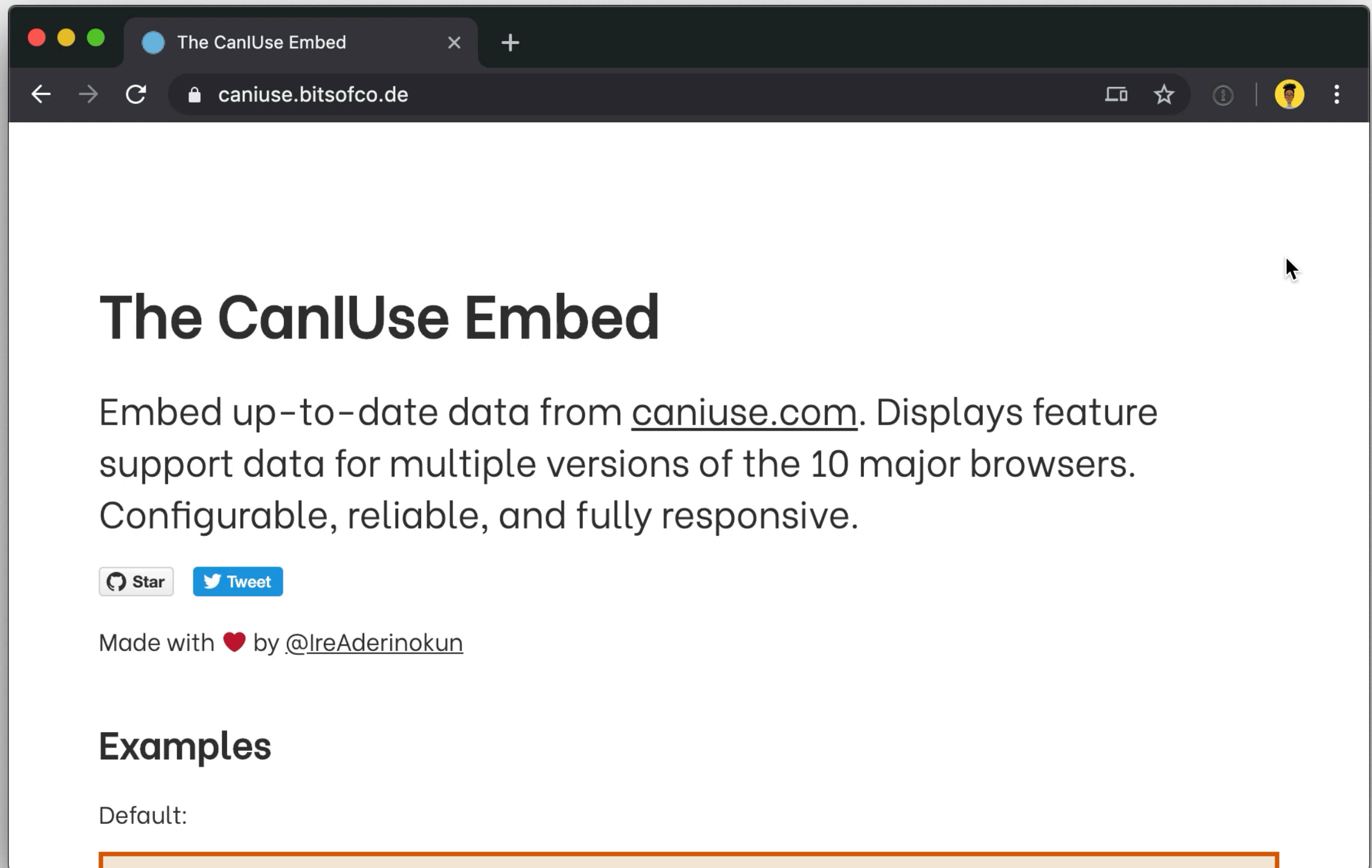
Global: 69.83% + 1.03% = 70.86%

Data from caniuse.com | Embed from caniuse.bitsofco.de

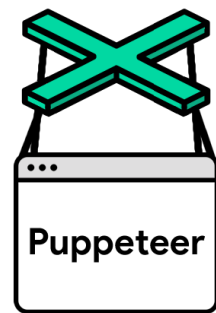
[Enable accessible colours](#)

JAMstackifying





JAMstackifying



```
exports.handler = async (event, context) => {  
  
  const params = JSON.parse(event.body);  
  const feature = params.feature;  
  const periods = params.periods;  
  const a11yColours = params.a11yColours;  
  
  try {  
    const screenshot = await takeScreenshot(feature, periods, a11yColours);  
    const image = await uploadScreenshot(feature, screenshot);  
  
    return { statusCode: 200, body: JSON.stringify(image) };  
  } catch (err) {  
    return { statusCode: 500, body: err.toString() }  
  }  
  
};
```

Thank you!

Where to find me

- ireaderinokun.com
- bitsofco.de
- [@ireaderinokun](https://twitter.com/ireaderinokun)

Use the embed

- caniuse.bitsofco.de

Further reading & sources

- <http://news.bbc.co.uk/2/hi/technology/6210068.stm>
- <https://bitsofco.de/using-a-headless-browser-to-capture-page-screenshots/>
- <https://bitsofco.de/how-to-upload-a-screenshot-from-puppeteer-to-cloudinary/>