A11Y Tips

for the

Mindful Developer

The case for accessibility

a c c e s s i b i l i t y

1 2 3 4 5 6 7 8 9 10 11

a 11 y

Accessibility

Can be used by everyone, regardless if they have disabilities.

“The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect.”

– Tim Berners-Lee, inventor of the World Wide Web

Types of disabilities

Types of disabilities

Permanent

VISUAL

AUDITORY

MOBILITY

COGNITIVE

Types of disabilities

Permanent

VISUAL

AUDITORY

MOBILITY

COGNITIVE

Temporary or Situational

AGING

NEW PARENT

SUN

MEETING

POWER USERS

INJURY

Why should we care about a11y?

Why should we care about a11y?

1 *It affects more people than you think.*

1.3 billion people with disabilities.

Why should we care about a11y?

1 *It affects more people than you think.*

1.3 billion people with disabilities.

2 *It's on the way to be mandatory.*

In some countries or fields, it already is.

Why should we care about a11y?

1. *It affects more people than you think.*

   1.3 billion people with disabilities.

2. *It's on the way to be mandatory.*

   In some countries or fields, it already is.

3. *It’s the right thing to do.*

   And our damn job.

The WebAIM Million

Test top 1M website's homepages using automated accessibility tools.

The WebAIM Million

Test top 1M website's homepages using automated accessibility tools.

30% can be verified with **automation**.

The WebAIM Million

Test top 1M website's homepages using automated accessibility tools.

30% can be verified with **automation**.

97.8% pages with accessibility **failures**.

The WebAIM Million

Test top 1M website's homepages using automated accessibility tools.

30% can be verified with **automation**.

97.8% pages with accessibility **failures**.

+10.1% errors in **React** websites.

We are preventing people from using our websites.

*We need to do **way** better.*

10 A11Y Tips

★☆☆
Beginner

★★☆
Intermediate

★★★
Advanced

★☆☆

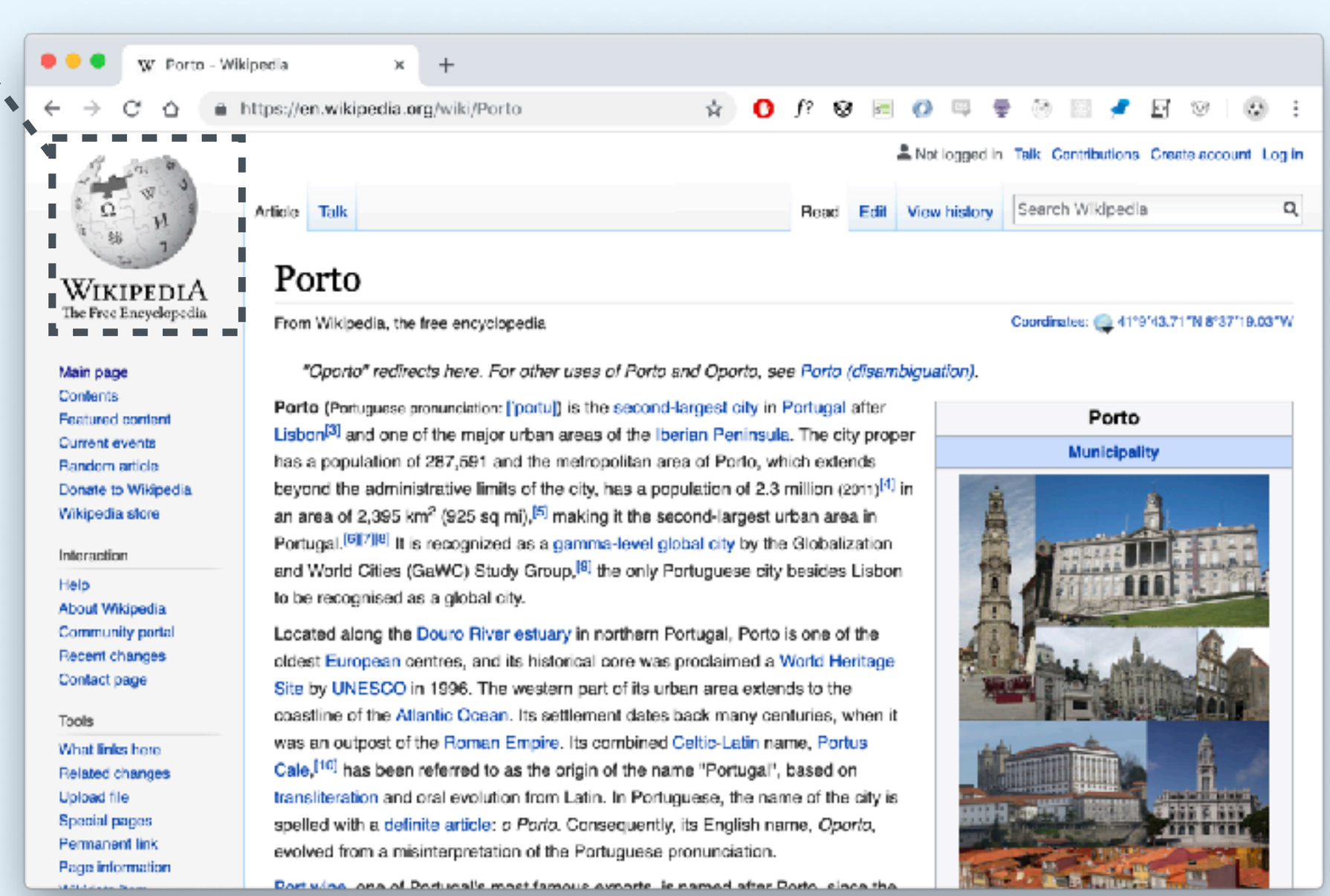
TIP #1

Landmarks

how we perceive a webpage

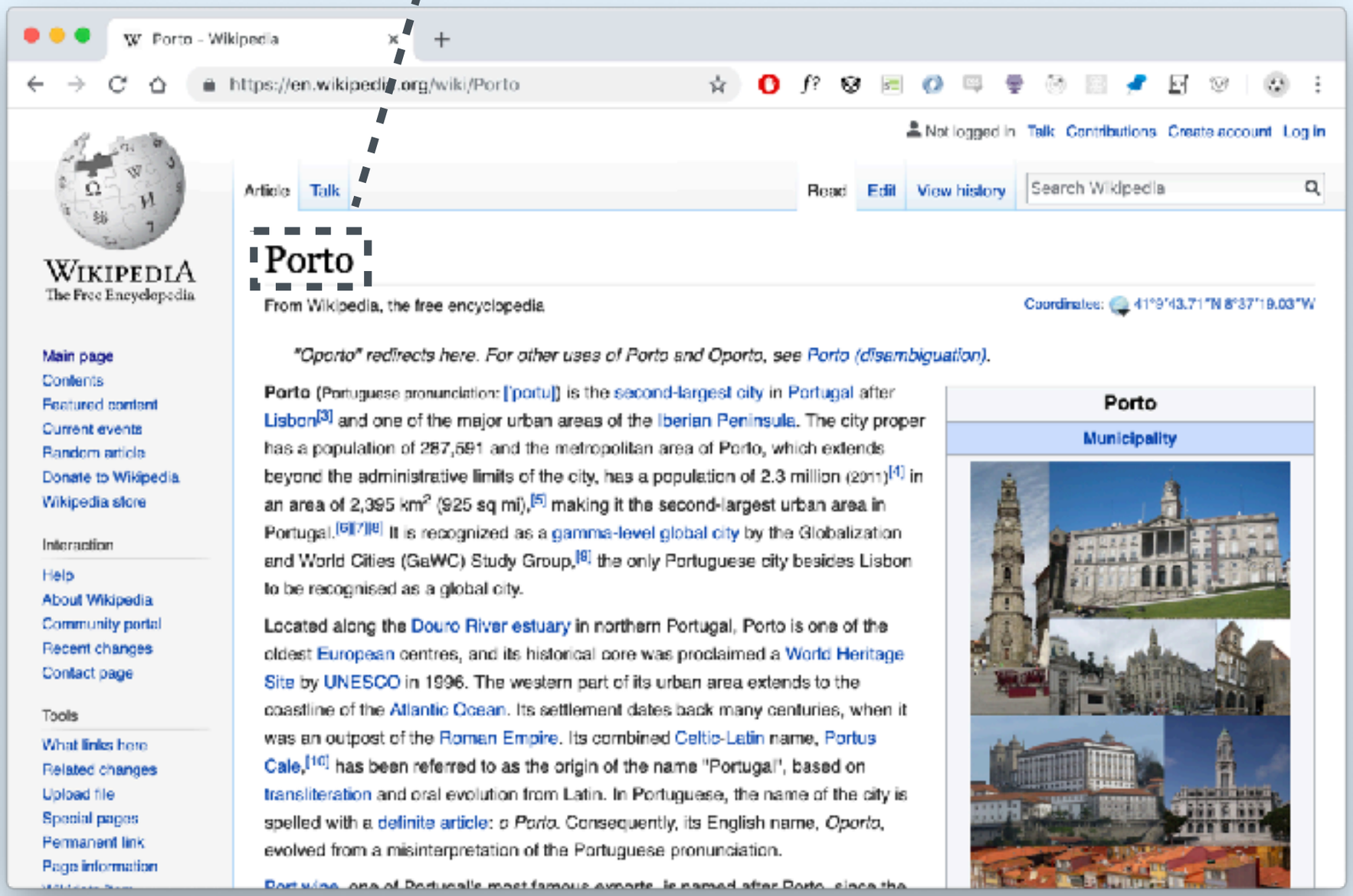
how we perceive a webpage

we are in wikipedia



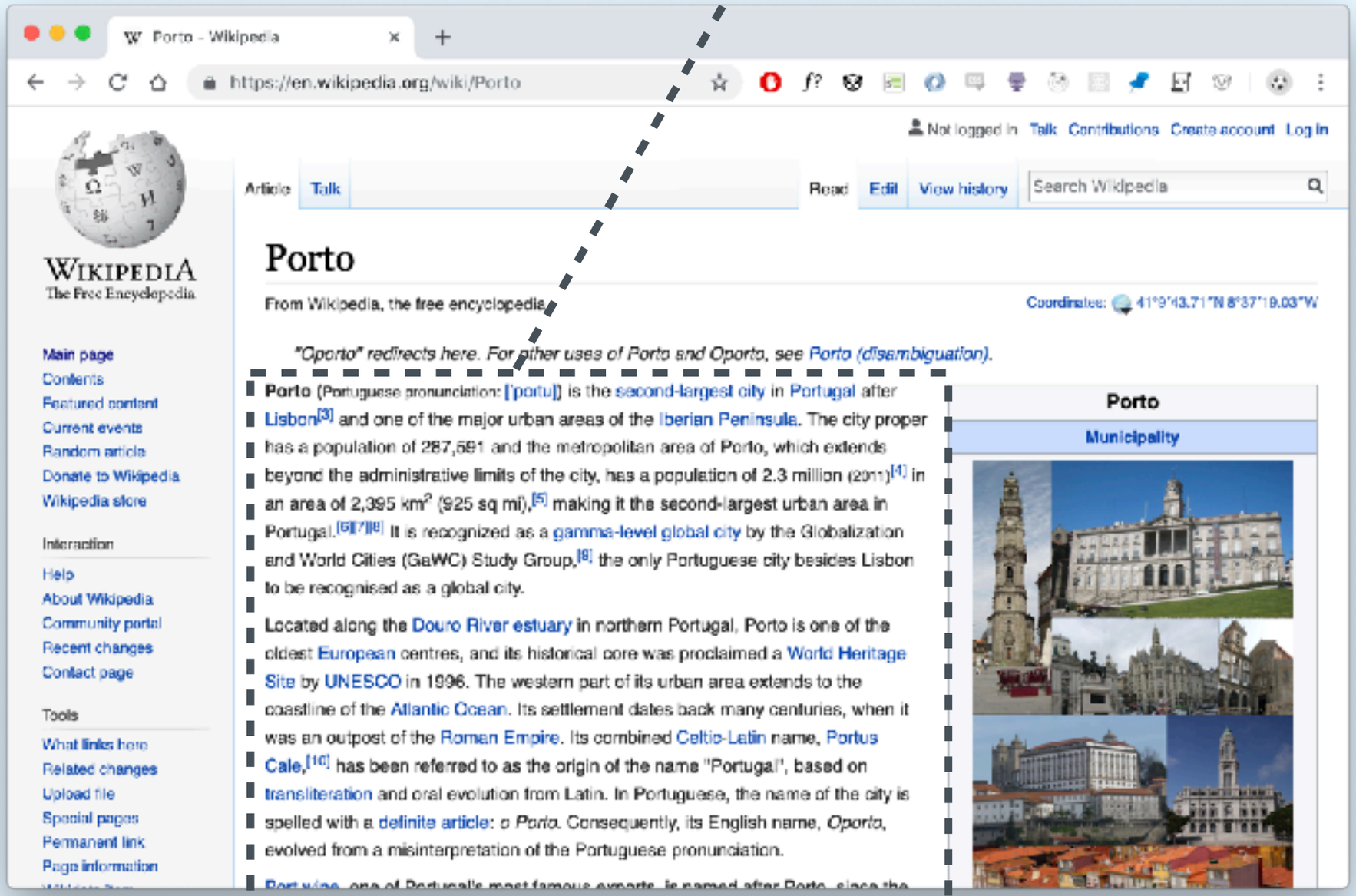
how we perceive a webpage

reading about Porto

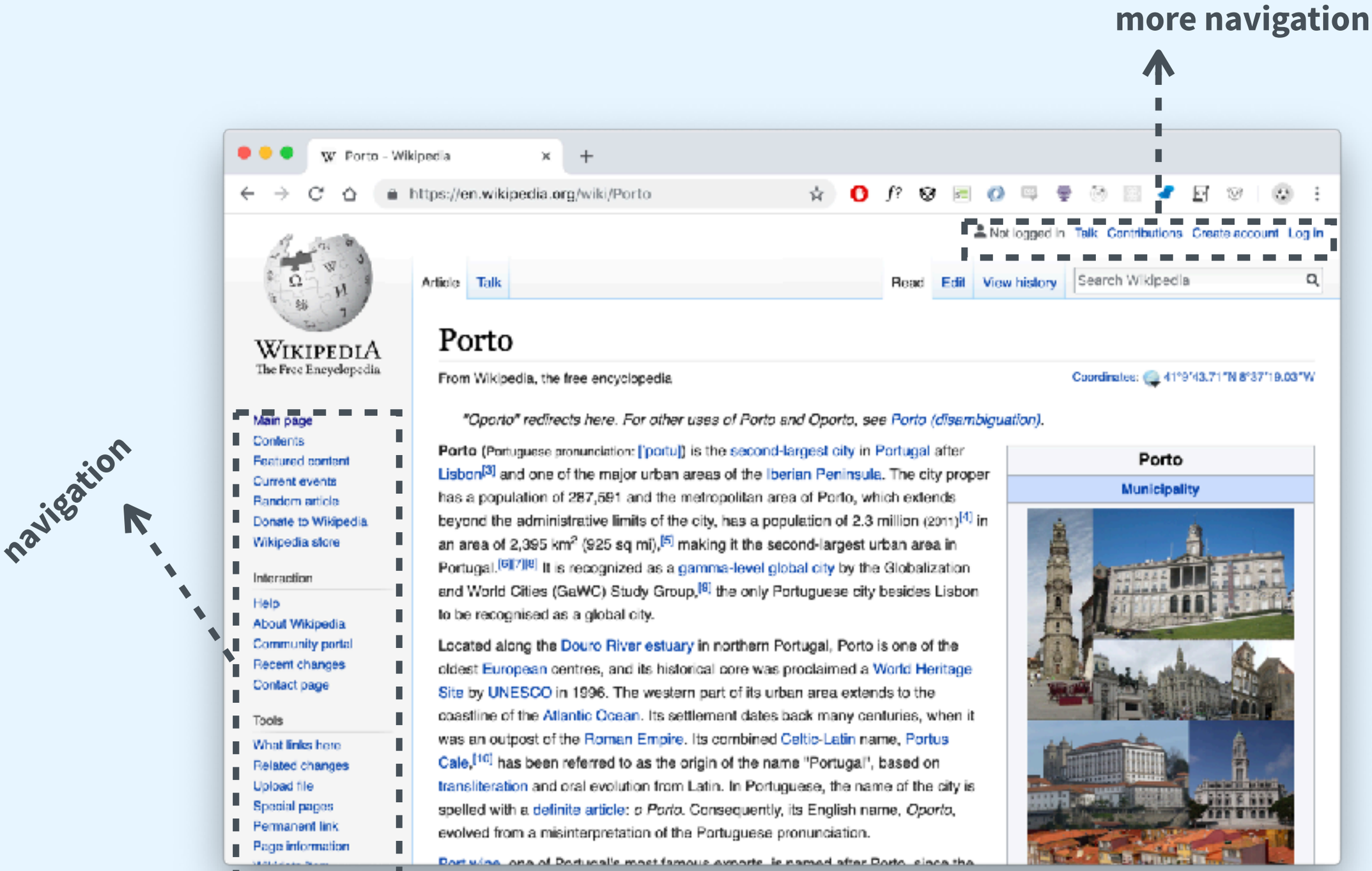


how we perceive a webpage

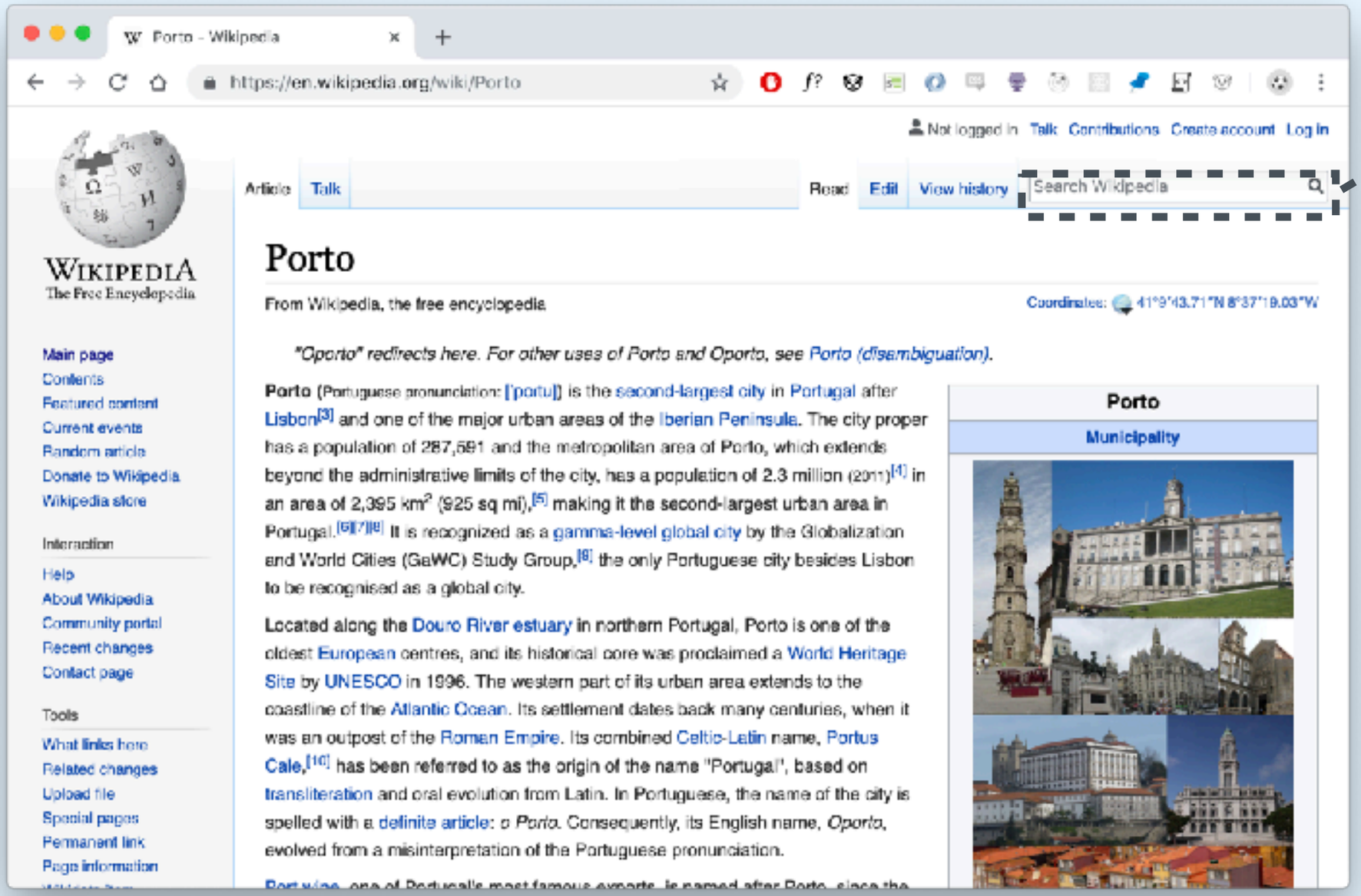
content starts here



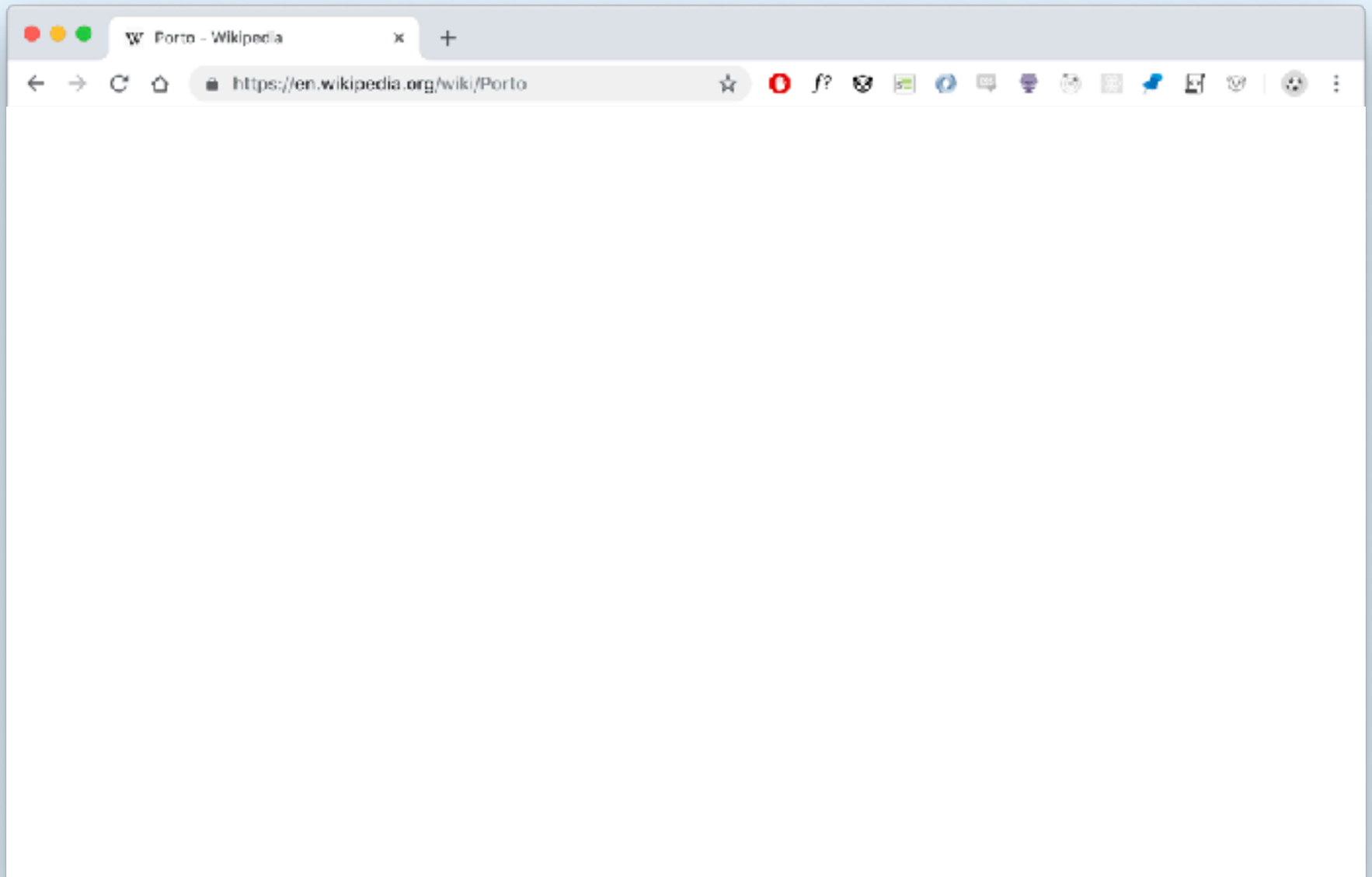
how we perceive a webpage



how we perceive a webpage

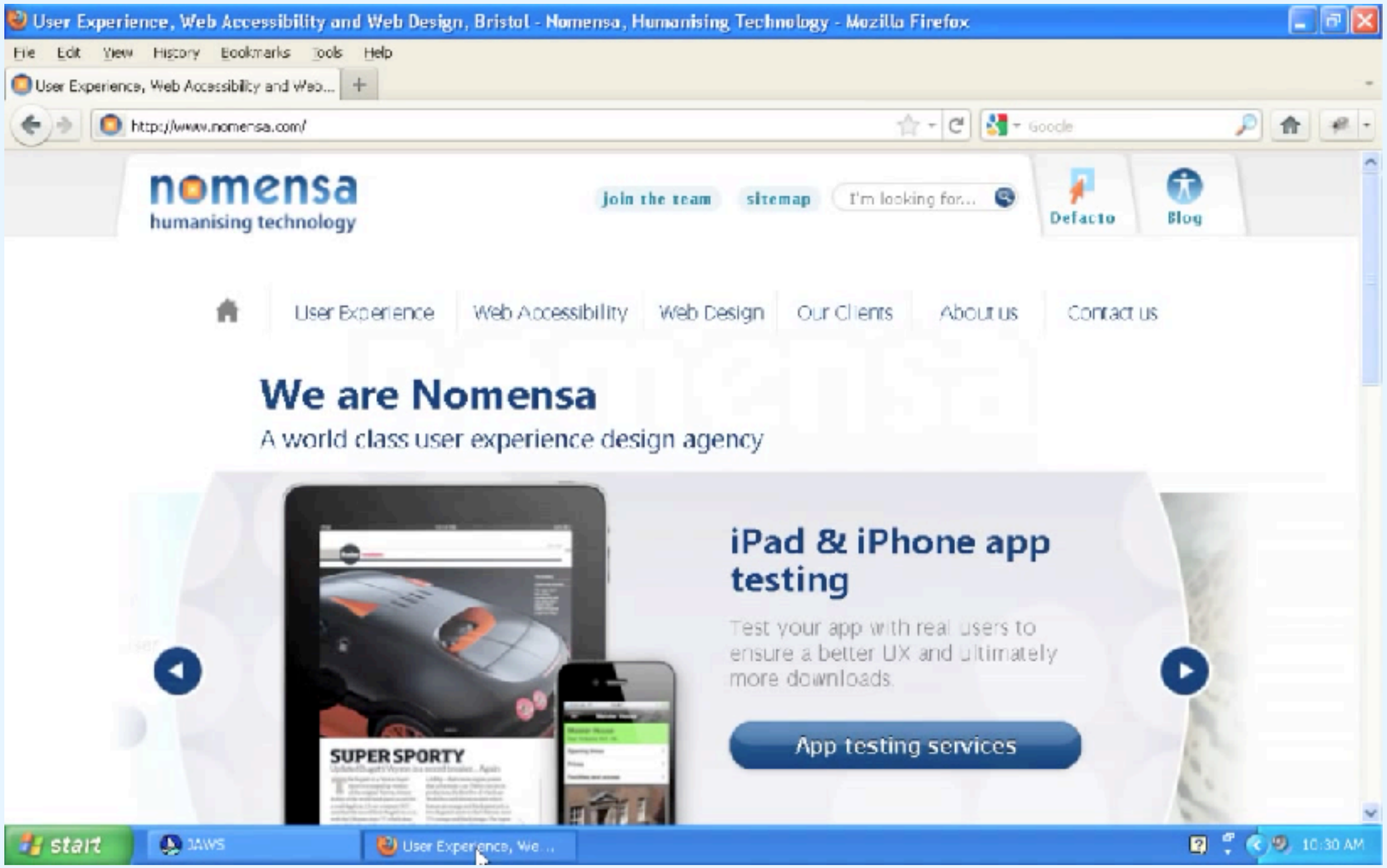


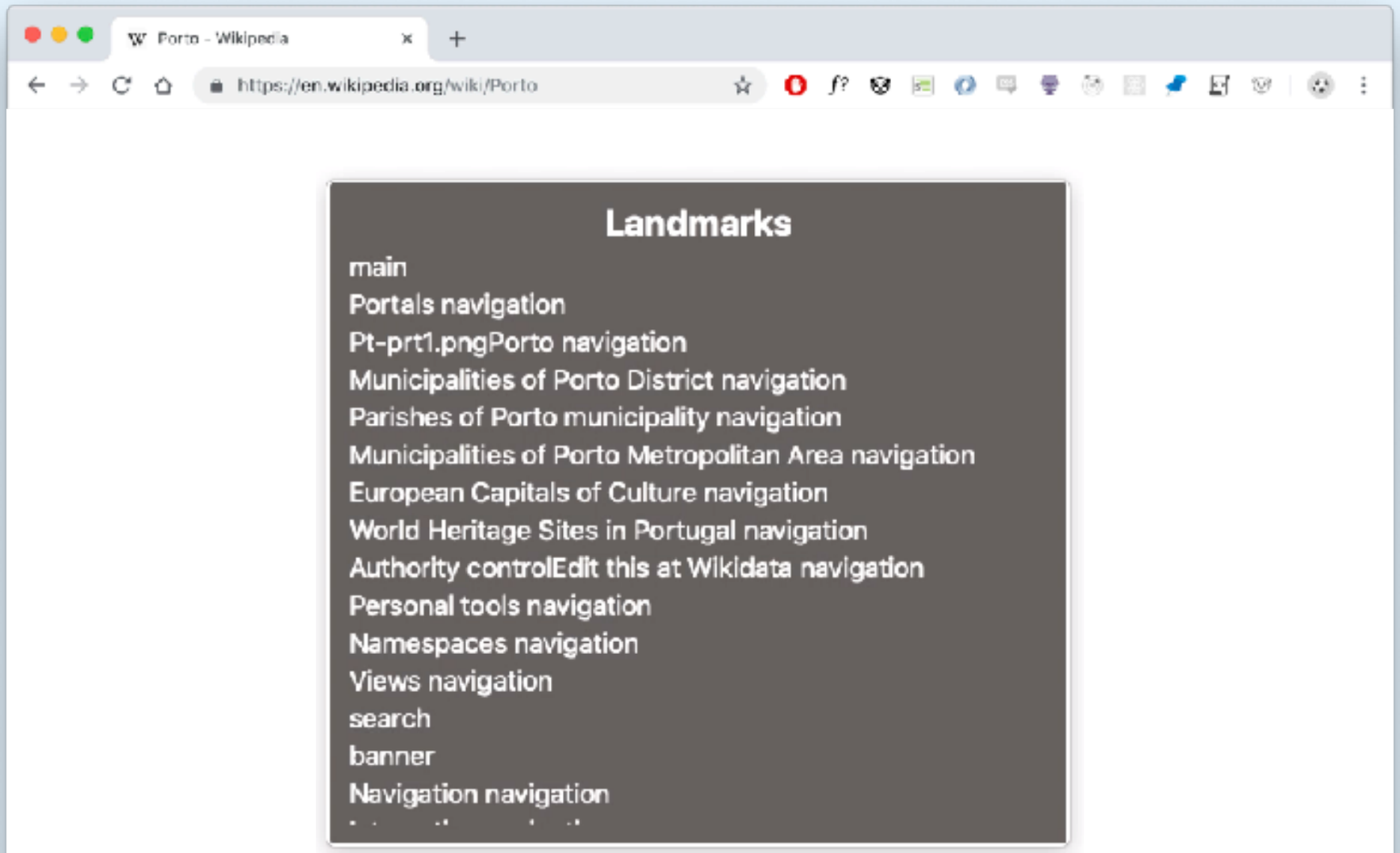
how people with disabilities perceive a webpage



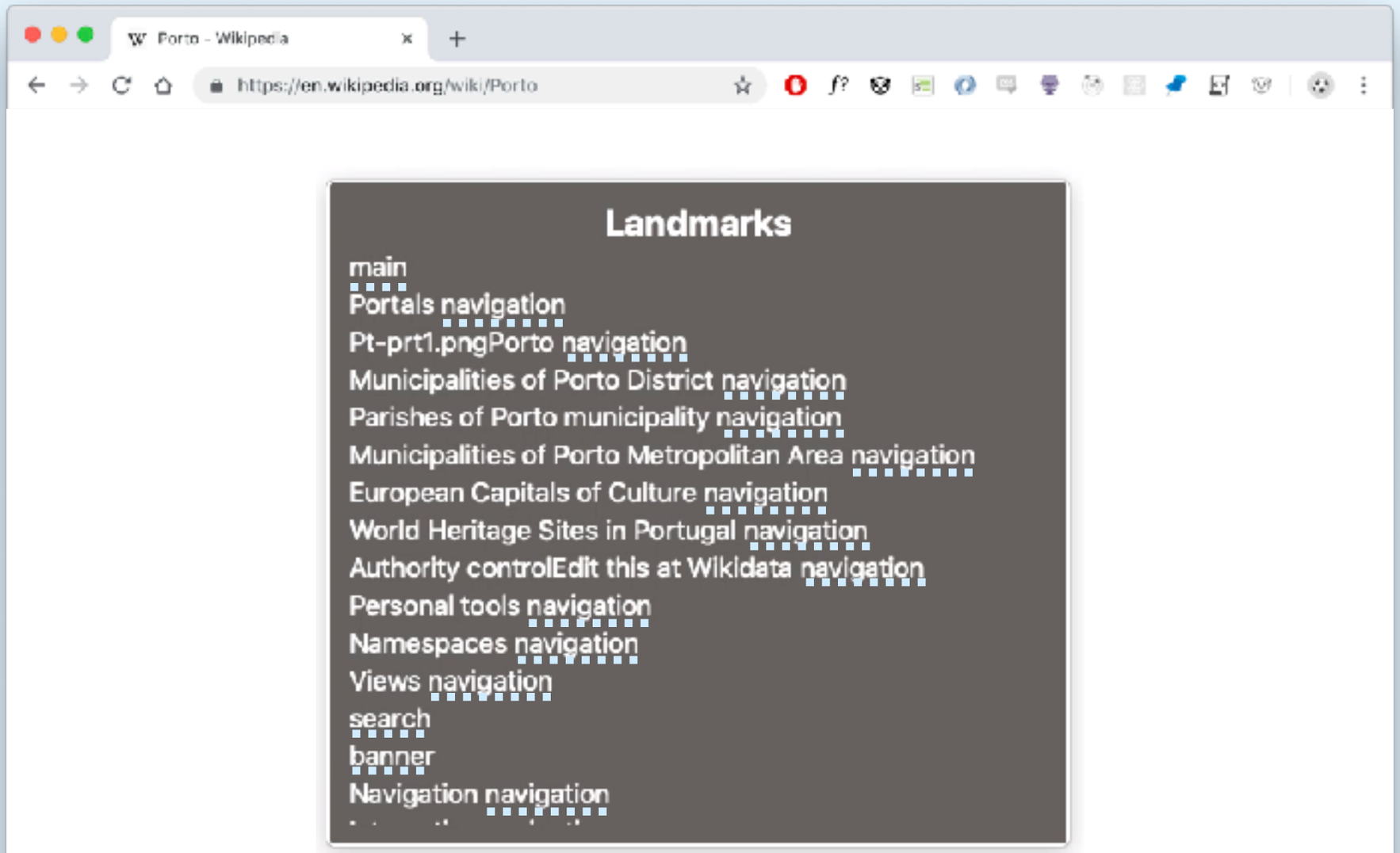
Screen Reader

Assistive technology that reads the content of a page for people who have visual disabilities.




Porto - Wikipedia
https://en.wikipedia.org/wiki/Porto
Landmarks
main
Portals navigation
Pt-prt1.pngPorto navigation
Municipalities of Porto District navigation
Parishes of Porto municipality navigation
Municipalities of Porto Metropolitan Area navigation
European Capitals of Culture navigation
World Heritage Sites in Portugal navigation
Authority controlEdit this at Wikidata navigation
Personal tools navigation
Namespaces navigation
Views navigation
search
banner
Navigation navigation

HTML5 elements is a good start.



HTML5 (tag)	Role (attribute)
`<header />`	`banner *`
`<main />`	`main`
`<footer />`	`contentinfo`
`<nav />`	`navigation`
`<section />`	`region`
`<form />`	`form`
`<aside />`	`complementary`
	`search`

***** Does not apply if descendant of ***`article`***, ***`aside`***, ***`main`***, or ***`section`***.

Explicitly override a role when needed.

```
<form role='search'>
  <input type='search' aria-label='search text' />
  <button>search</button>
</form>
```

★☆☆

TIP #2

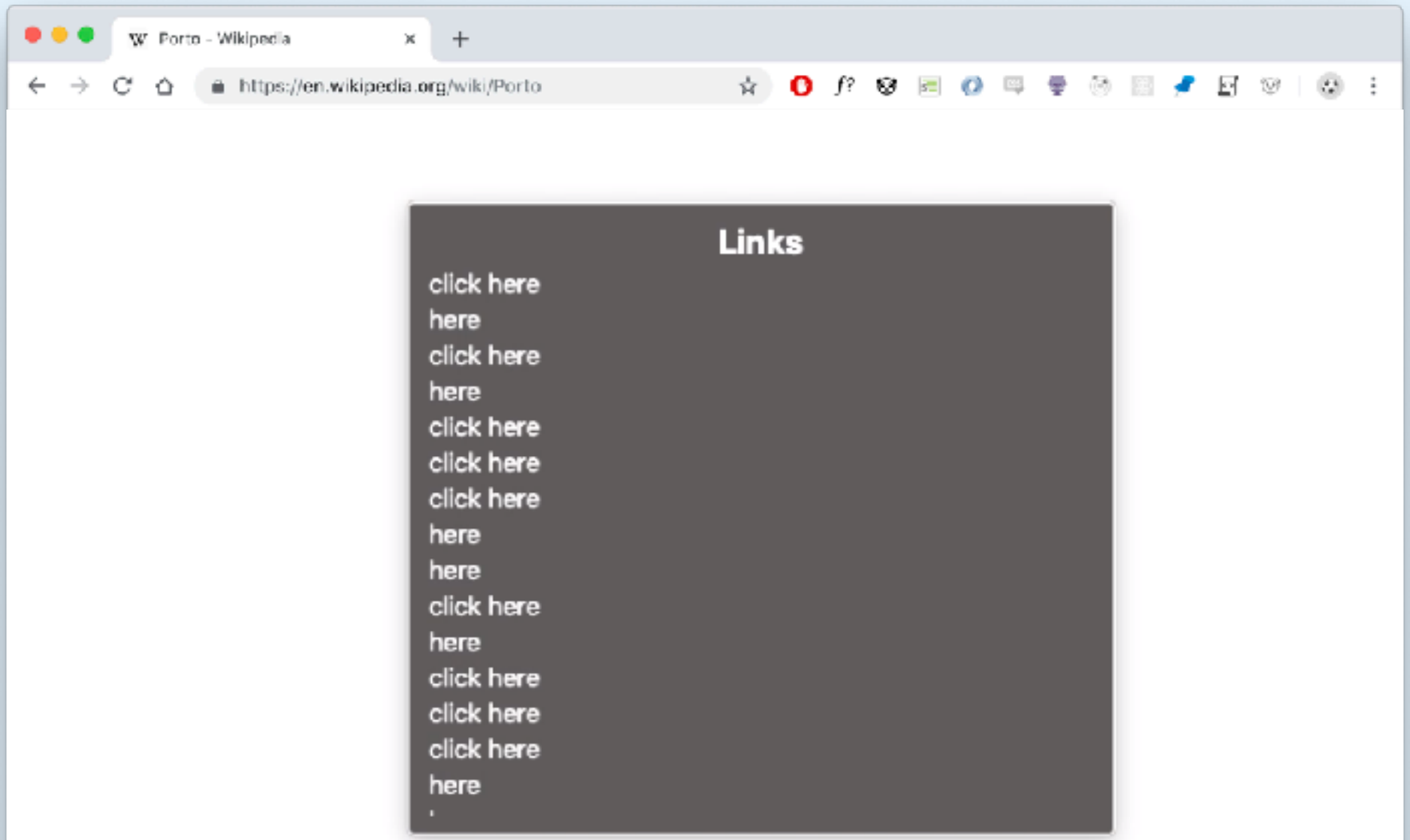
Read More Links

To see our documentation click here.

Interested in our roadmap? Read it here.

To see our documentation click here.

Interested in our roadmap? Read it here.



Option A: change copy.

Yes, accessibility is not only about code.

```
<!— ⛔ BAD —>
To see our documentation <a href="/README.md">click here</a>.
```

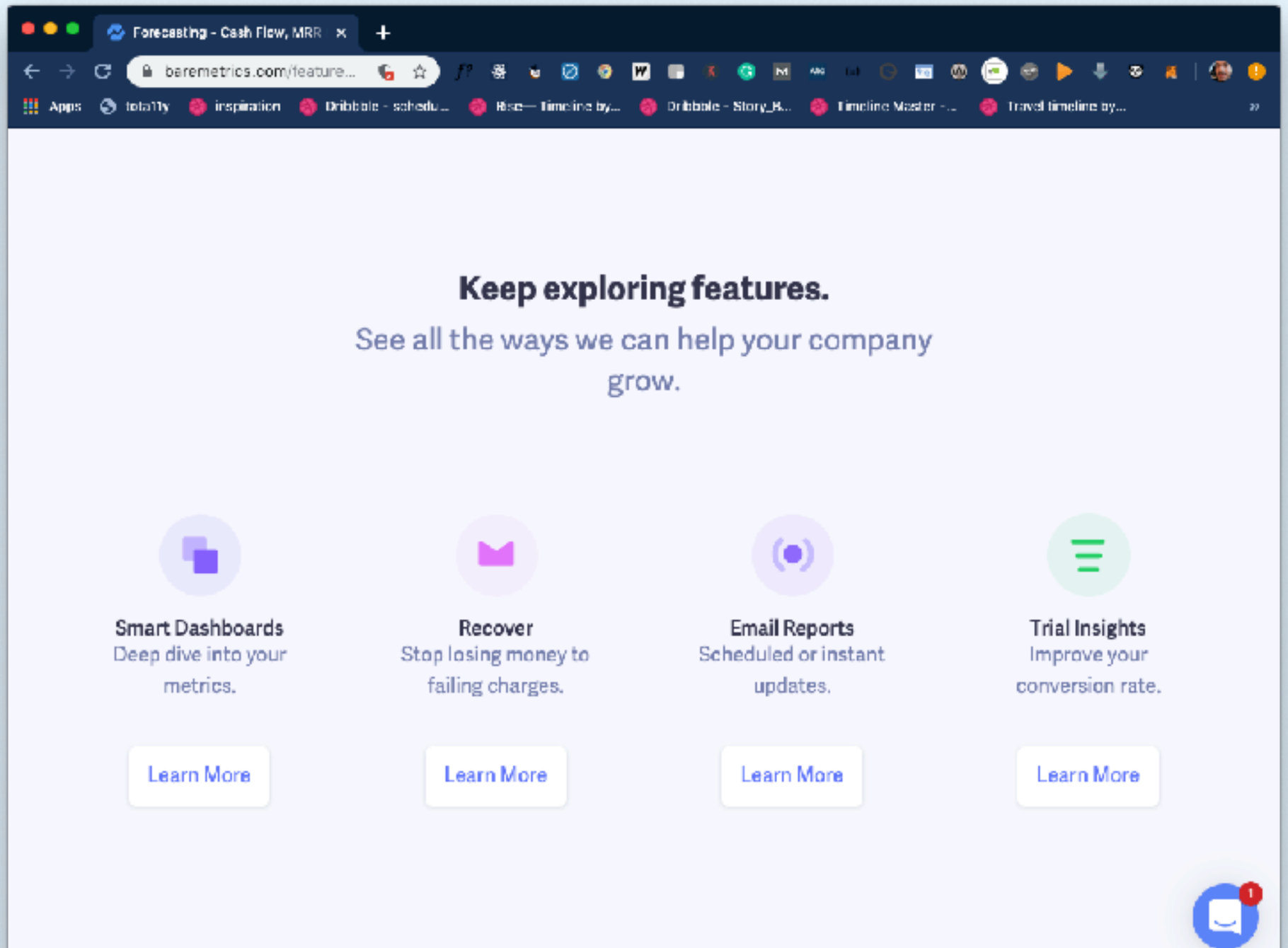
Option A: change copy.

Yes, accessibility is not only about code.

```
<!— ⛔ BAD —>
To see our documentation <a href="/README.md">click here</a>.

<!— ✅ Better —>
We have made our <a href="/README.md">documentation</a> available.
```

Sometimes there's nothing we can do.



Option B: override labelling with `aria-label`

```
<!— ⛔ BAD —>
<a href="/smart-dashboards">Read more</a>
```

Option B: override labelling with `aria-label`

```
<!— ⛔ BAD —>
<a href="/smart-dashboards">Read more</a>

<!— ✅ Better —>
<a
  href=“/smart-dashboards"
  aria-label="read more about smart dashboards">Read more</a>
```

★☆☆

TIP #3

Decorative Images

See all attendees 

See all attendees >

```
<a href='/attendees'>
  See all attendees
  <svg> ... </svg>
</a>
```

See all attendees >

```
<a href='/attendees'>
  See all attendees
  <svg aria-hidden='true'> ... </svg>
</a>
```

See all attendees >

```
<!— ⛔ BAD: SC will announce the “src” ⟶
<a href='/attendees'>
  See all attendees
  <img src=‘/icons/arrow-1230dashd9a8h12s1892s1.svg’ />
</a>
```

See all attendees >

```
<!— ⛔ BAD: SC will announce the “src” ⟶
<a href='/attendees'>
  See all attendees
  <img src=‘/icons/arrow-1230dashd9a8h12s1892s1.svg’ />
</a>


<!—— ✅ Better ⟶
<a href='/attendees'>
  See all attendees
  <img
    src=‘/icons/arrow-1230dashd9a8h12s1892s1.svg'
    alt=‘'
  />
</a>
```

See all attendees >

```
<!— ⛔ BAD: SC will announce the “src” ⟶
<a href='/attendees'>
  See all attendees
  <img src=‘/icons/arrow-1230dashd9a8h12s1892s1.svg’ />
</a>


<!—— ✅ Better ⟶
<a href='/attendees'>
  See all attendees
  <img
    src=‘/icons/arrow-1230dashd9a8h12s1892s1.svg'
    role='presentation'
  />
</a>
```

★☆☆

TIP #4

Icon Buttons

```
<!—— ⛔ BAD ——>
<button>
  <svg> ... </svg>
</button>
```

```
<!— ⛔ BAD —>
<button>
  <svg> ... </svg>
</button>

<!— ✅ Better —>
<button>
  <svg aria-hidden='true'>…</svg>
</button>
```

```
<!— ⛔ BAD —>
<button>
  <svg> ... </svg>
</button>

<!— ✅ Better —>
<button aria-label='close'>
  <svg aria-hidden='true'>…</svg>
</button>
```

★★☆

TIP #5

Toggle Icon Buttons

```
<button
  aria-label='mute'
  aria-pressed='false'
>
  <svg aria-hidden='true'>
   …
  </svg>
</button>
```

```
<button
   aria-label=‘mute'
   aria-pressed=‘false'
>
   <svg aria-hidden='true'>
     …
   </svg>
</button>
```

```
<button
  aria-label='mute'
  aria-pressed='false'
>
  <svg aria-hidden='true'>
    …
  </svg>
</button>
```

```
<button
  aria-label=‘mute'
  aria-pressed=‘false'
>
  <svg aria-hidden='true'>
    …
  </svg>
</button>
```

```
<button
  aria-label=‘mute'
  aria-pressed=‘true’
>
  <svg aria-hidden='true'>
    …
  </svg>
</button>
```

You still have to manage state.

You still have to manage state.

```
function ButtonMute() {
  const [pressed, setPressed] = useState(false);

  return (
    <button
      aria-label='close'
      aria-pressed={pressed}
      onClick={() ⇒ {
        setPressed(!pressed);
        // do the thing ...
      }}
    >
      <svg aria-hidden='true'> ... </svg>
    </button>
  )
}
```

You still have to manage state.

```
function ButtonMute() {
  const [pressed, setPressed] = useState(false);

  return (
    <button
      aria-label='close'
      aria-pressed={pressed}
      onClick={() ⇒ {
        setPressed(!pressed);
        // do the thing ...
      }}
    >
      <svg aria-hidden='true'> ... </svg>
    </button>
  )
}
```

★★☆

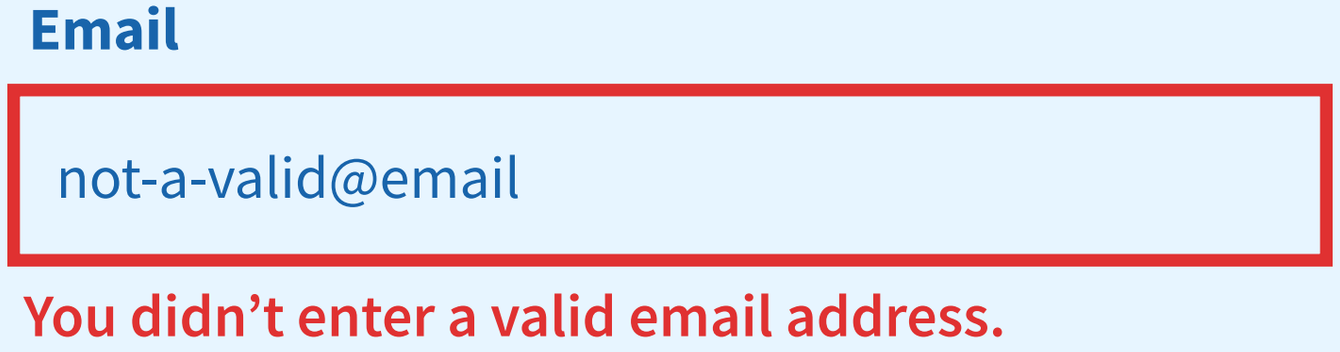
TIP #6

Form Validation

Email

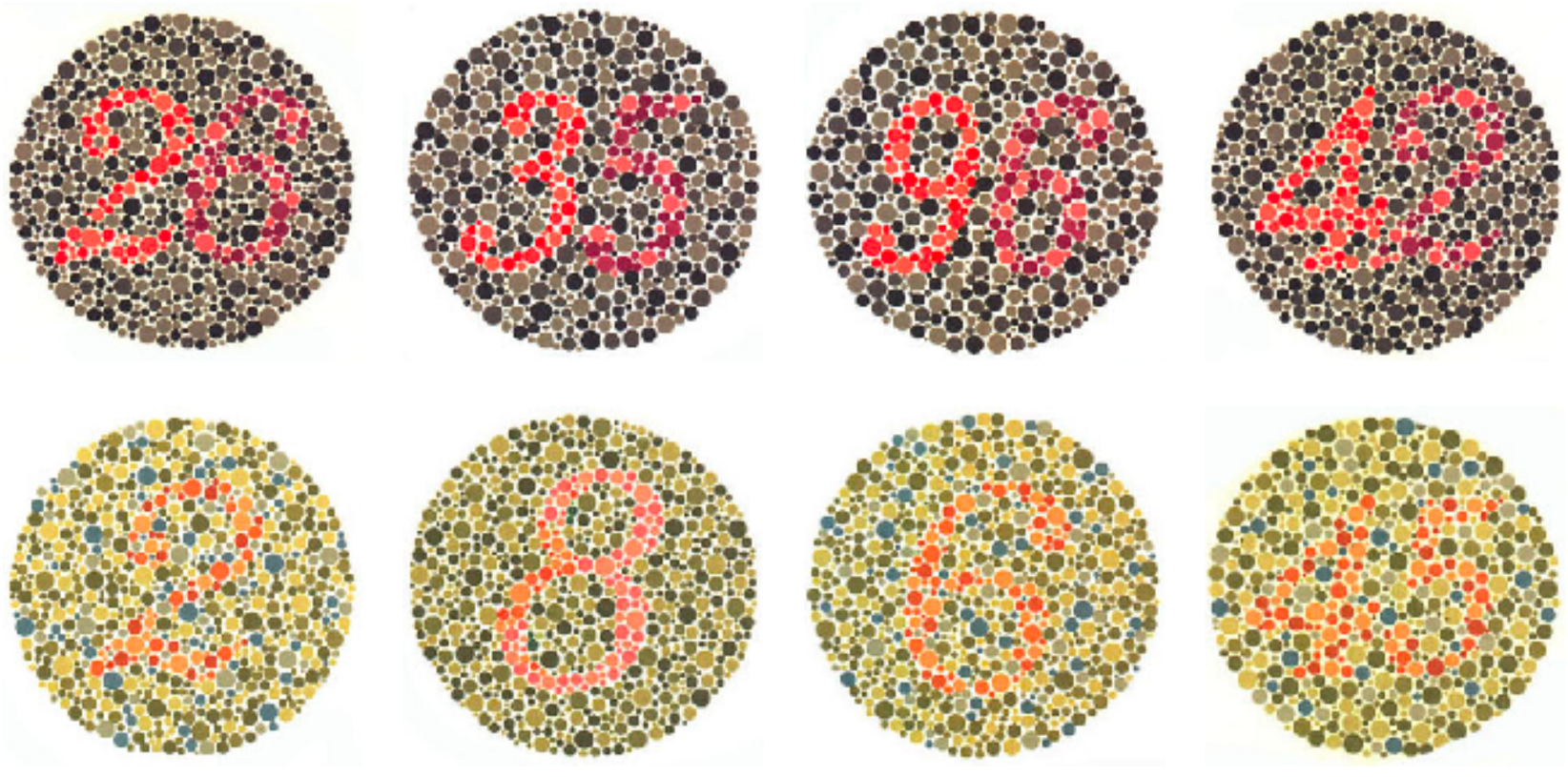
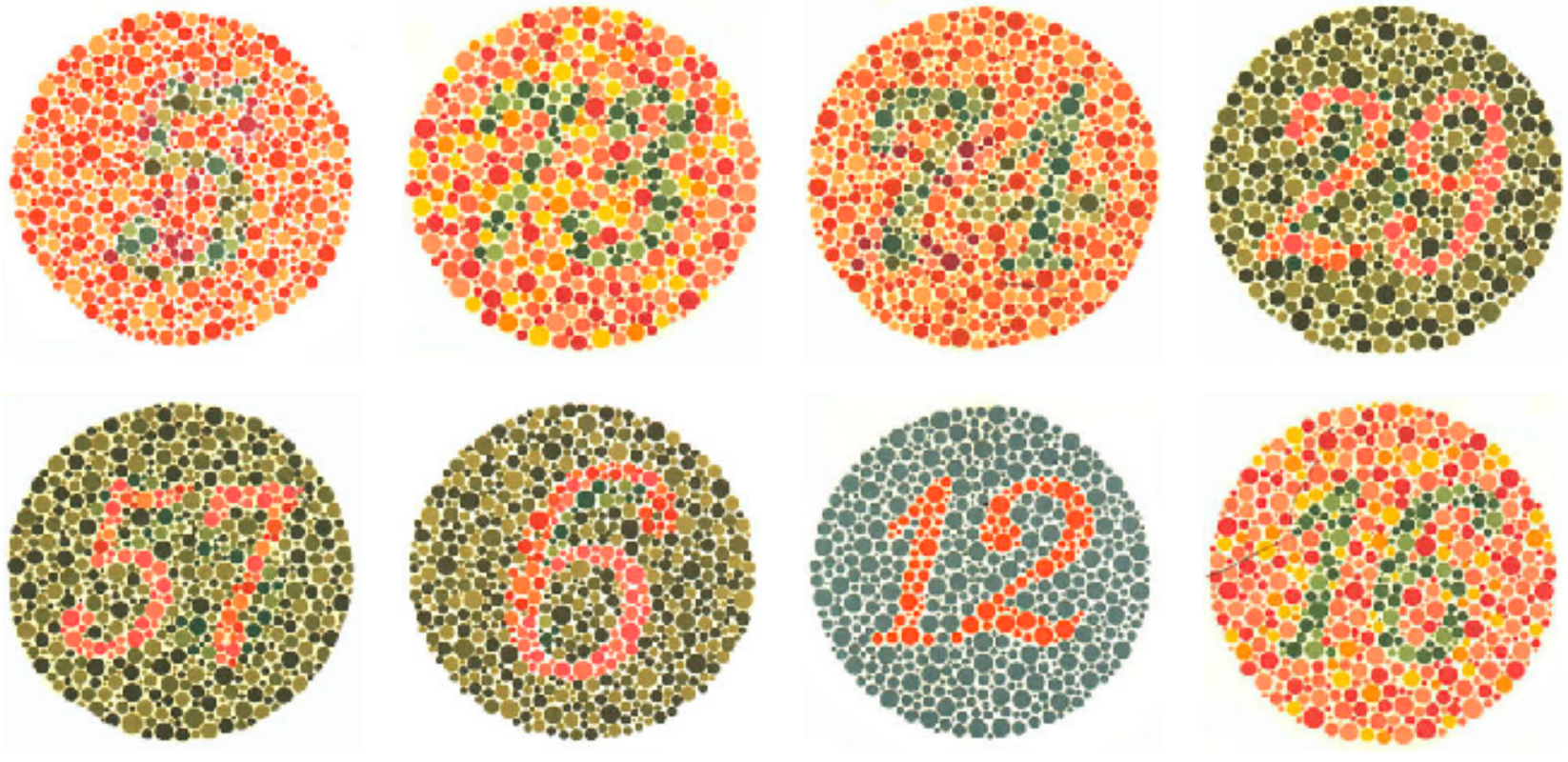
not-a-valid@email

You didn’t enter a valid email address.



Design Error: relying on color alone.

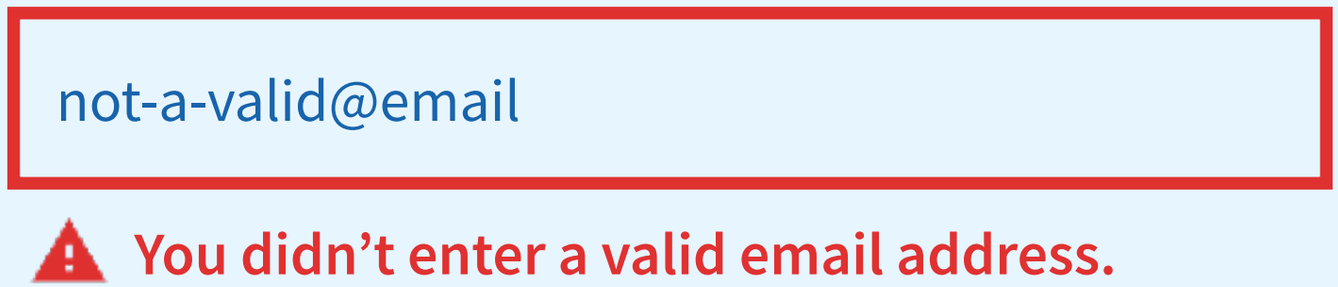
1 in 12 men and 1 in 200 women are color blind.

Email

not-a-valid@email

You didn’t enter a valid email address.



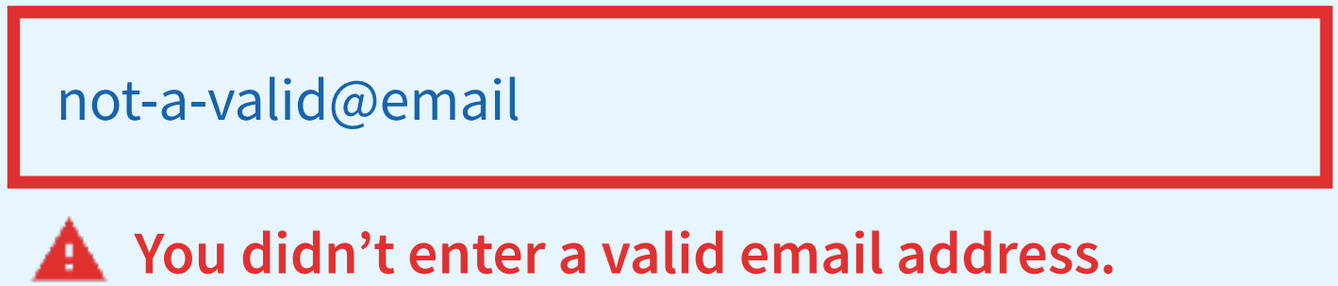

```
<p>Email:</p>
<input type=‘email’ />
<p>
  <svg> … </svg>
  You didn't enter a valid email address.
</p>
```

Email

not-a-valid@email

You didn’t enter a valid email address.

```
<p>Email:</p>
<input type=‘email’ />
<p>
  <svg aria-hidden='true'> … </svg>
  You didn't enter a valid email address.
</p>
```




```
<label>Email:</label>
<input type=‘email’ />
<p>
  <svg aria-hidden='true'> ... </svg>
  You didn't enter a valid email address.
</p>
```

Email

not-a-valid@email

⚠ **You didn’t enter a valid email address.**

```
<label for='email'>Email:</label>
<input type=‘email’
       id=‘email’ />
<p>
  <svg aria-hidden='true'> ... </svg>
  You didn't enter a valid email address.
</p>
```

Email

not-a-valid@email

⚠ **You didn’t enter a valid email address.**

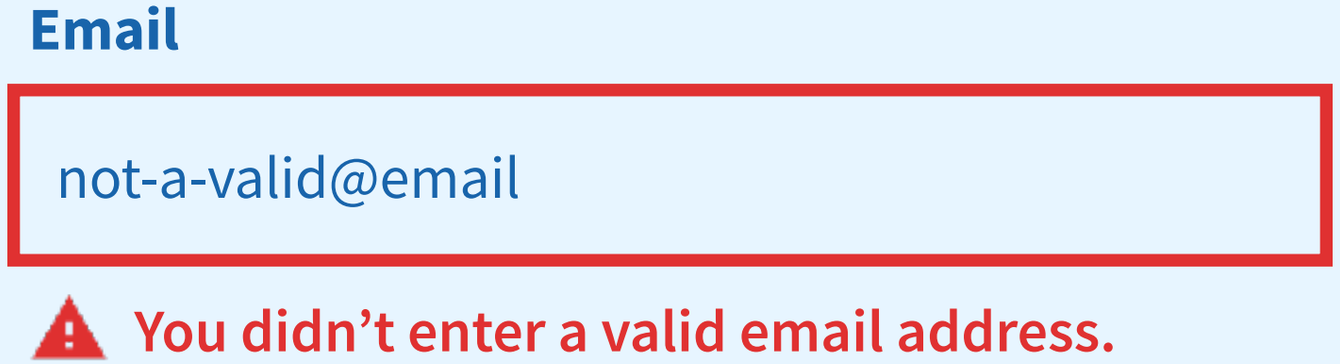
```
<label for='email'>Email:</label>
<input type=‘email’
       id=‘email’
       aria-describedby=‘message’ />
<p id=‘message'>
  <svg aria-hidden=‘true' >...</svg>
  You didn't enter a valid email address.
</p>
```

Email

not-a-valid@email

⚠ **You didn’t enter a valid email address.**

```
<label for='email'>Email:</label>
<input type=‘email’
       id=‘email’
       aria-describedby=‘message’ />
<p id=‘message’
   aria-live=‘polite|assertive’>
  <svg aria-hidden=‘true' >…</svg>
  You didn't enter a valid email address.
</p>
```



`aria-live="polite"`

Any updates made to this region should only be announced if the user is not currently doing anything

`aria-live="assertive"`

Any updates made to this region are important enough to be announced to the user as soon as possible, but it is not necessary to immediately interrupt the user.

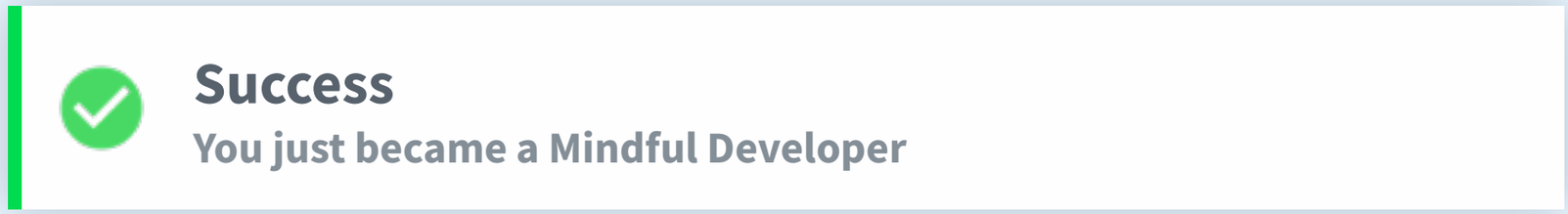
★★☆

TIP #6

Notifications

Success

You just became a Mindful Developer

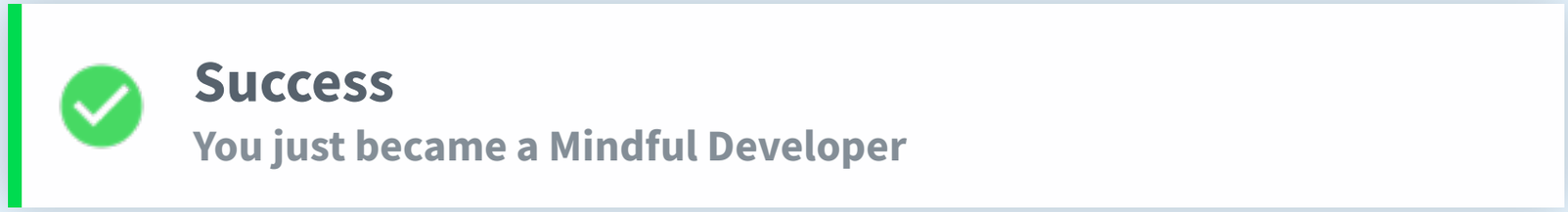



```
<div>
  <svg> ... </svg>
  <span>Success</span>
  <span>You just became a Mindful Developer</span>
</div>
```

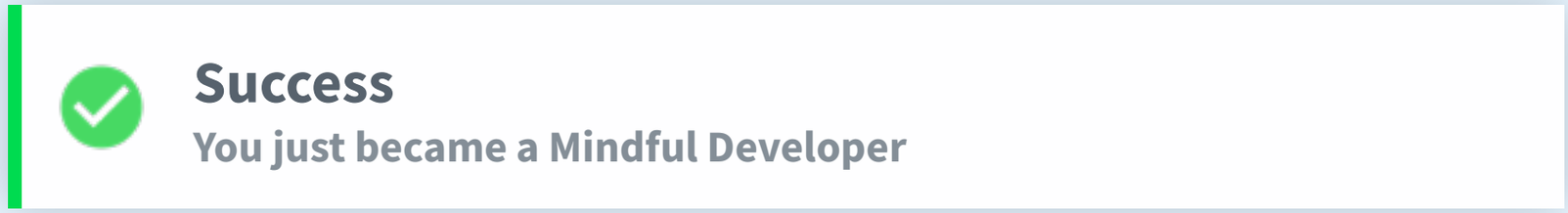
Success

You just became a Mindful Developer

```
<div>
  <svg aria-hidden='true'> ... </svg>
  <span>Success</span>
  <span>You just became a Mindful Developer</span>
</div>
```




```
<div role='status'>
  <svg aria-hidden='true'> ... </svg>
  <span>Success</span>
  <span>You just became a Mindful Developer</span>
</div>
```




```
<div role='status'
     aria-live='polite'>
  <svg aria-hidden='true'> ... </svg>
  <span>Success</span>
  <span>You just became a Mindful Developer</span>
</div>
```

By adding `role="status"`, the element gets `aria-live` set to `"polite"`.

★☆☆

TIP #8

Tooling

Linting

Accessibility issues as linting errors.

Linting

Accessibility issues as linting errors.

```
npm install eslint-plugin-jsx-a11y --save-dev
```

```
// .eslintrc
{
  "extends": [
    "plugin:jsx-a11y/recommended"
  ],
  "plugins": [
    "jsx-a11y"
  ]
}
```

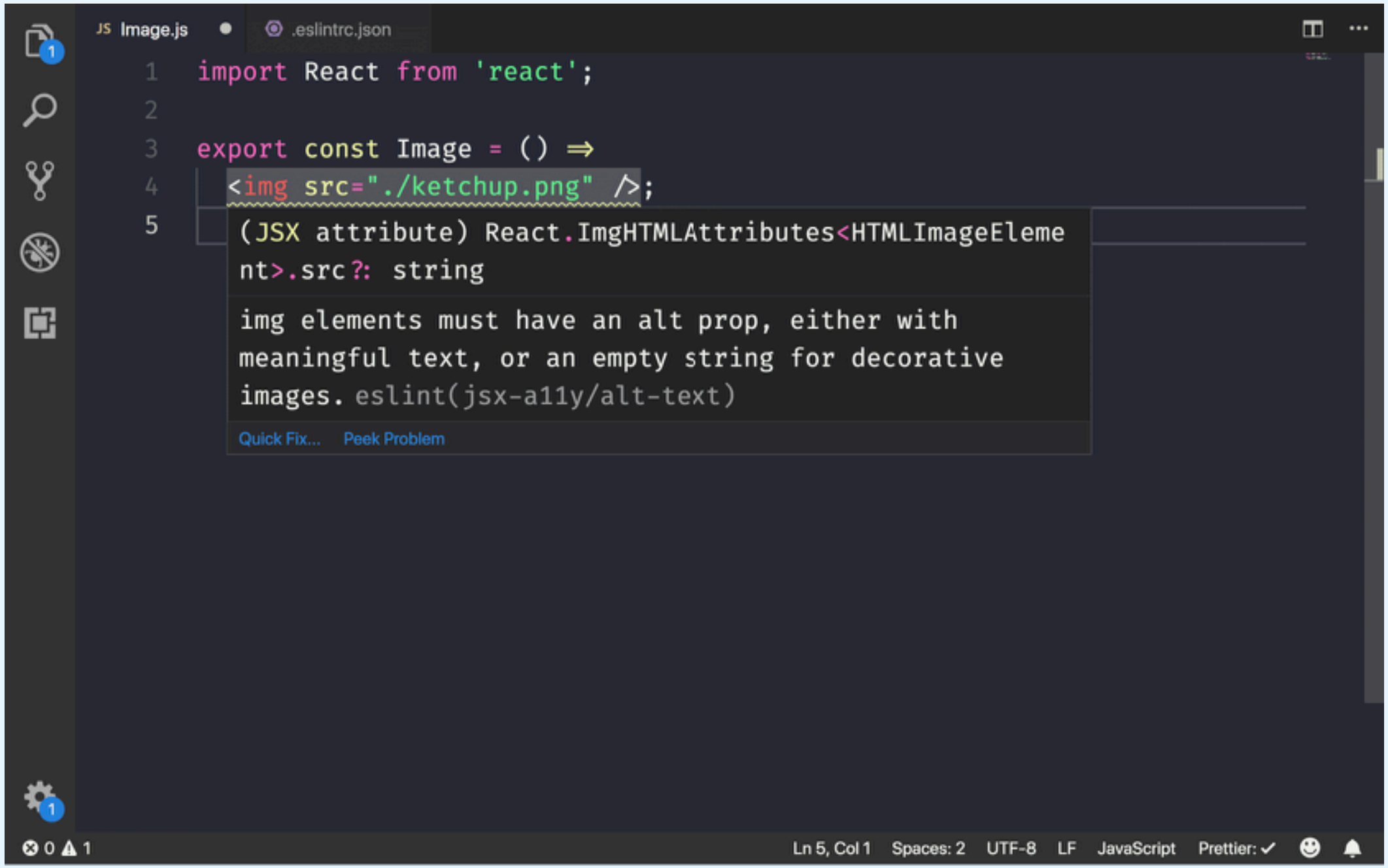


Image.js
.eslintrc.json
1 import React from 'react';
2
3 export const Image = () =>
4 ;
5
(JSX attribute) React.ImgHTMLAttributes<HTMLImageElement>.src?: string
img elements must have an alt prop, either with meaningful text, or an empty string for decorative images. eslint(jsx-a11y/alt-text)
Quick Fix...
Peek Problem
0 1
Ln 5, Col 1
Spaces: 2
UTF-8
LF
JavaScript
Prettier:

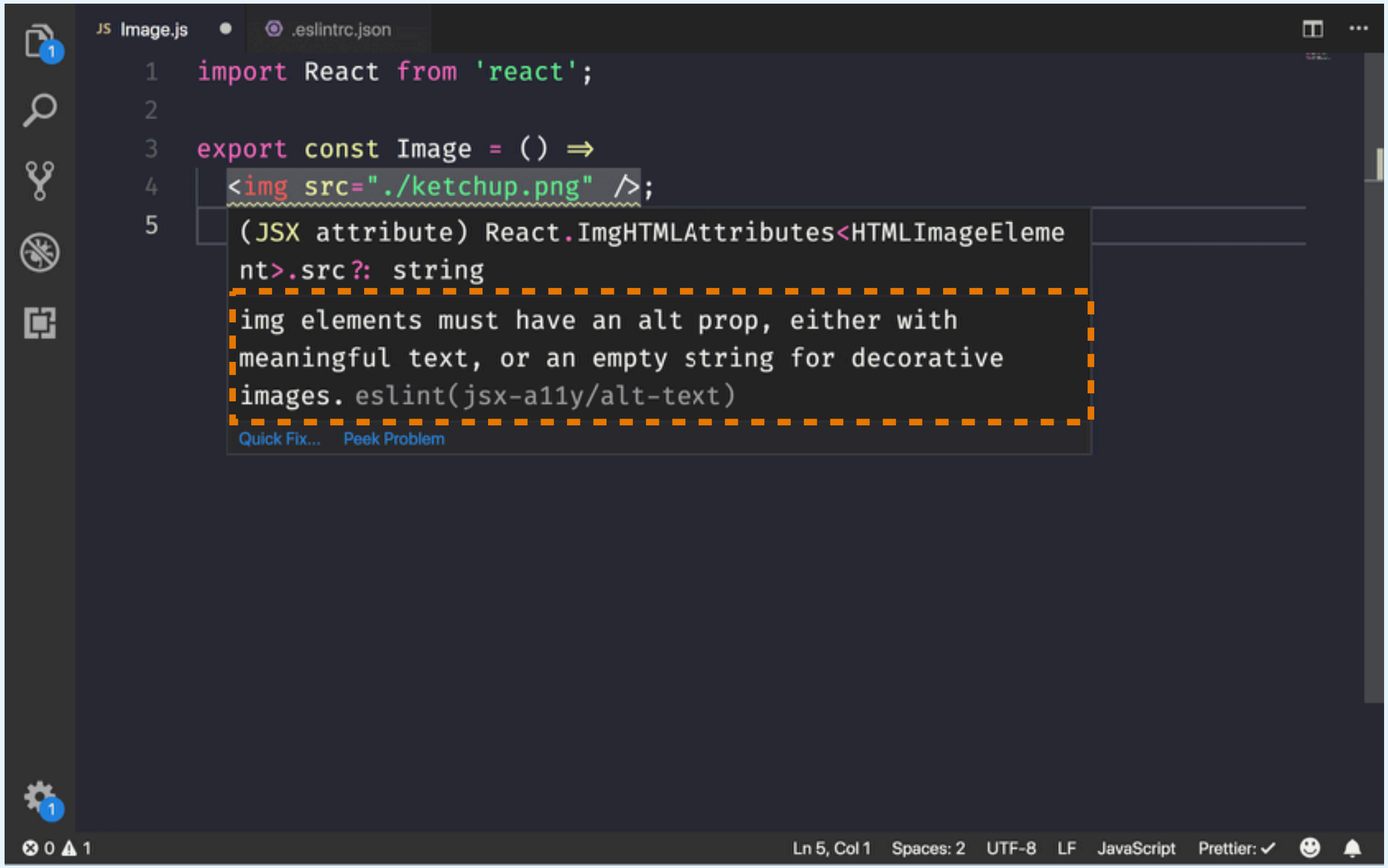


Image.js
.eslintrc.json
1 import React from 'react';
2
3 export const Image = () =>
4 ;
5
(JSX attribute) React.ImgHTMLAttributes<HTMLImageElement>.src?: string
img elements must have an alt prop, either with meaningful text, or an empty string for decorative images. eslint(jsx-a11y/alt-text)
Quick Fix... Peek Problem
0 1
Ln 5, Col 1 Spaces: 2 UTF-8 LF JavaScript Prettier:

Console

Accessibility issues as console errors.

Console

Accessibility issues as console errors.

```
npm install --save-dev react-axe
```

```
import React from 'react';
import ReactDOM from 'react-dom';

if (process.env.NODE_ENV !== 'production') {
  import('react-axe').then(axe => {
    axe(React, ReactDOM, 1000);
    ReactDOM.render(<App />, document.getElementById('root'));
  });
} else {
  ReactDOM.render(<App />, document.getElementById('root'));
}
```

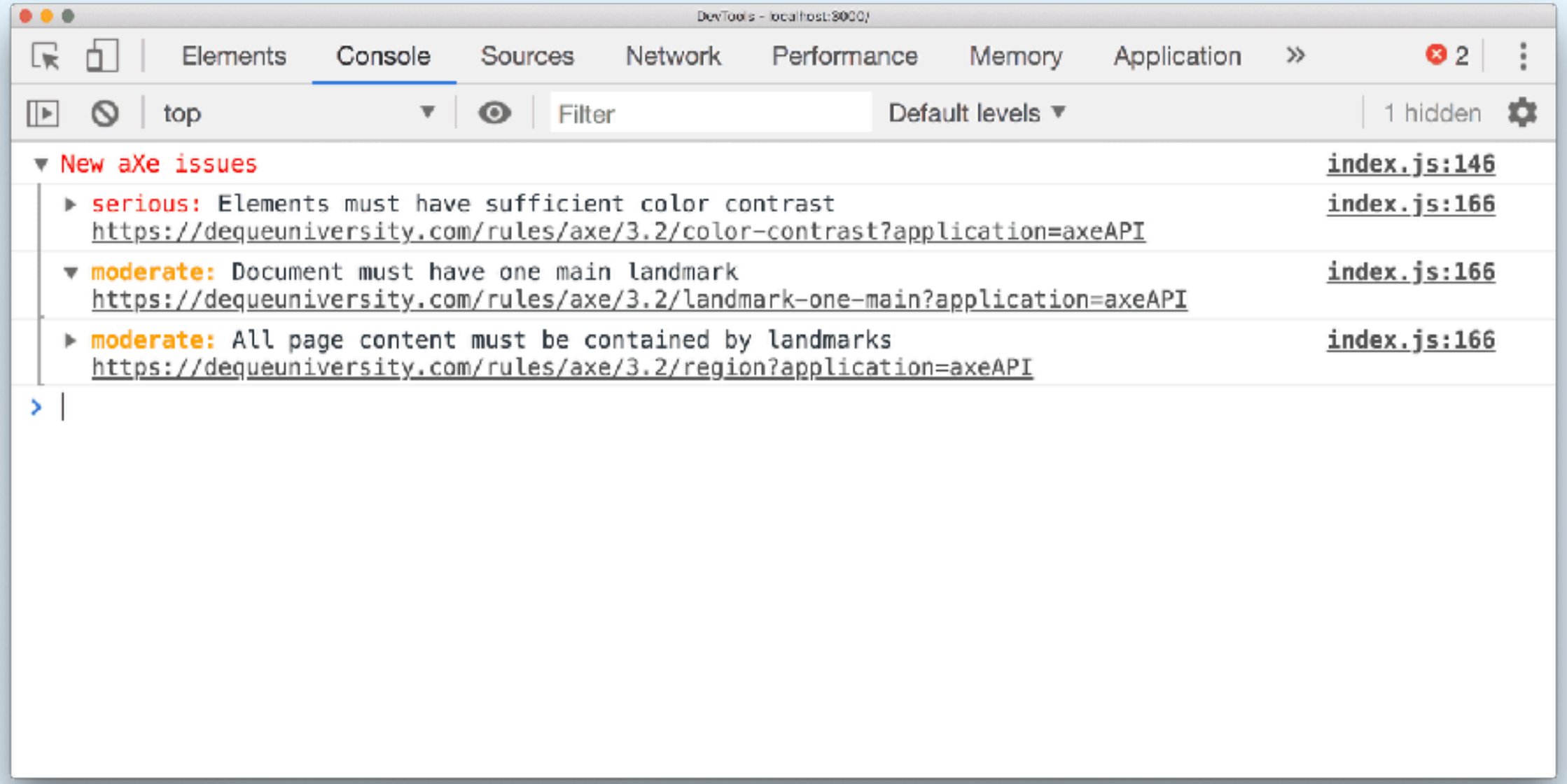
Console

Accessibility issues as console errors.

```
npm install --save-dev react-axe
```

```
import React from 'react';
import ReactDOM from 'react-dom';

if (process.env.NODE_ENV !== 'production') {
  import('react-axe').then(axe => {
    axe(React, ReactDOM, 1000);
    ReactDOM.render(<App />, document.getElementById('root'));
  });
} else {
  ReactDOM.render(<App />, document.getElementById('root'));
}
```


DevTools - localhost:3000/
Elements
Console
Sources
Network
Performance
Memory
Application
2
top
Filter
Default levels
1 hidden
New aXe issues
index.js:146
serious: Elements must have sufficient color contrast
https://dequeuniversity.com/rules/axe/3.2/color-contrast?application=axeAPI
index.js:166
moderate: Document must have one main landmark
https://dequeuniversity.com/rules/axe/3.2/landmark-one-main?application=axeAPI
index.js:166
moderate: All page content must be contained by landmarks
https://dequeuniversity.com/rules/axe/3.2/region?application=axeAPI
index.js:166

Unit Test

Accessibility issues can be integrated in your CI.

Unit Test

Accessibility issues can be integrated in your CI.

```
npm install jest-axe --save-dev
```

```
const React = require('react')
const { render } =  require('react-dom')
const Page = require('./Page')

const { axe, toHaveNoViolations } = require('jest-axe')
expect.extend(toHaveNoViolations)

it(‘<App /> should pass automated accessibility
tests‘, async () ⇒ {
  render(<App/>, document.body)
  const results = await axe(document.body)
  expect(results).toHaveNoViolations()
})
```

Unit Test

Accessibility issues can be integrated in your CI.

```
npm install jest-axe --save-dev
```

```
const React = require('react')
const { render } =  require('react-dom')
const Page = require('./Page')

const { axe, toHaveNoViolations } = require('jest-axe')
expect.extend(toHaveNoViolations)

it(‘<App /> should pass automated accessibility
tests‘, async () ⇒ {
  render(<App/>, document.body)
  const results = await axe(document.body)
  expect(results).toHaveNoViolations()
})
```

★★☆

TIP #9

Client-Side Routing

PROBLEM

When users click a link, screen readers don’t announce new dynamic content.

Option A: focus on wrapper

Option A: focus on wrapper

```
function MyPage() {
  const pageRef = useRef(null);
  useEffect(() ⇒ {
    pageRef.current.focus();
  }, []);
  return (
    <div tabIndex={-1} aria-labelledBy='title' ref={pageRef}>
      <h1 id='title'>Page Title</h1>
      {/*  ...  */}
    </div>
  )
}
```

Make wrapper “focusable”.

Option A: focus on wrapper

```
function MyPage() {
  const pageRef = useRef(null);
  useEffect(() ⇒ {
    pageRef.current.focus();
  }, []);
  return (
    <div tabIndex={-1} aria-labelledBy='title' ref={pageRef}>
      <h1 id='title'>Page Title</h1>
      {/*  ...  */}
    </div>
  )
}
```

Provide labelling to the wrapper.

Option A: focus on wrapper

```
function MyPage() {
  const pageRef = useRef(null);
  useEffect(() ⇒ {
    pageRef.current.focus();
  }, []);
  return (
    <div tabIndex={-1} aria-labelledBy='title' ref={pageRef}>
      <h1 id='title'>Page Title</h1>
      {/*  ...  */}
    </div>
  )
}
```

Move focus to wrapper when component mounts.

Option B: focus on title

```
function MyPage() {
  const titleRef = useRef(null);
  useEffect(() ⇒ {
    titleRef.current.focus();
  }, []);
  return (
    <div>
      <h1 id='title'
          tabIndex={-1}
          ref={titleRef}>
        Page Title
      </h1>
      {/*  ...  */}
    </div>
  )
}
```

Option B: focus on title

```
function MyPage() {
  const titleRef = useRef(null);
  useEffect(() ⇒ {
    titleRef.current.focus();
  }, []);
  return (
    <div>
      <h1 id='title'
          tabIndex={-1}
          ref={titleRef}>
        Page Title
      </h1>
      {/*  ...  */}
    </div>
  )
}
```

Make title “focusable”.

Option B: focus on title

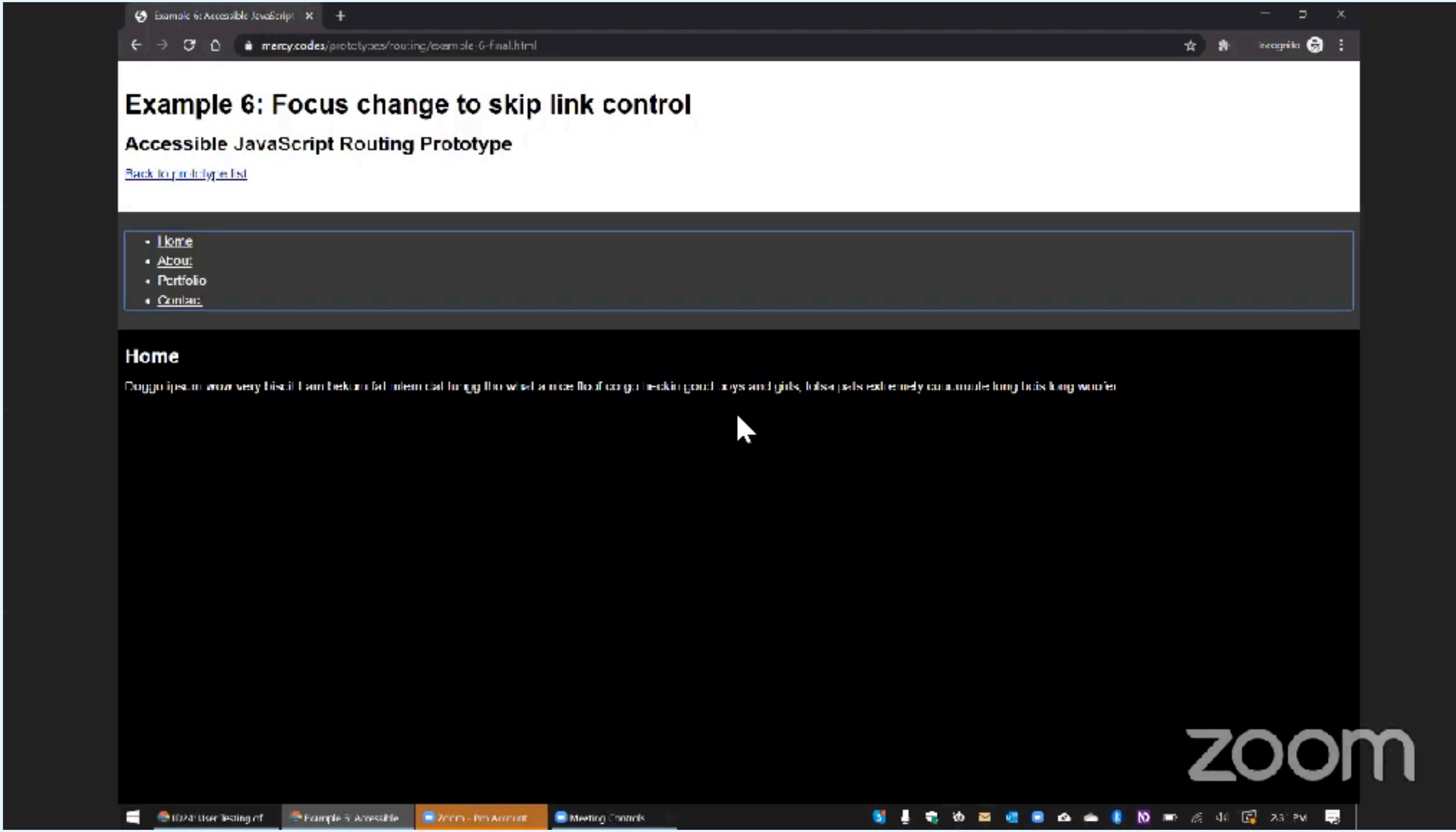
```
function MyPage() {
  const titleRef = useRef(null);
  useEffect(() => {
    titleRef.current.focus();
  }, []);
  return (
    <div>
      <h1 id='title'
          tabIndex={-1}
          ref={titleRef}>
        Page Title
      </h1>
      {/* ... */}
    </div>
  )
}
```

Make title “focusable”.

What we learned from user testing of accessible client-side routing techniques.

– Marcy Sutton, Head of Learning at Gatsby

https://www.gatsbyjs.org/blog/2019-07-11-user-testing-accessible-client-routing/


Example 6: Accessible JavaScript
mercy.codes/prototypes/routing/example-6-final.html
Example 6: Focus change to skip link control
Accessible JavaScript Routing Prototype
Back to prototype list
Home
About
Portfolio
Contact
Home
zoom

★★★

TIP #10

Accordion

Delivery

Billing

Payment

Delivery

Mr John Smith
132, My Street,
Kingston, New York 12401
United States

Edit

Billing

Payment

Payment +

```
<div>
    Payment
</div>
<div>
  …
</div>
```

Payment +

```
<h3>
    Payment
</h3>
<section>
  …
</section>
```

Add semantic meaning
(good for landmark or heading navigation)

Payment +

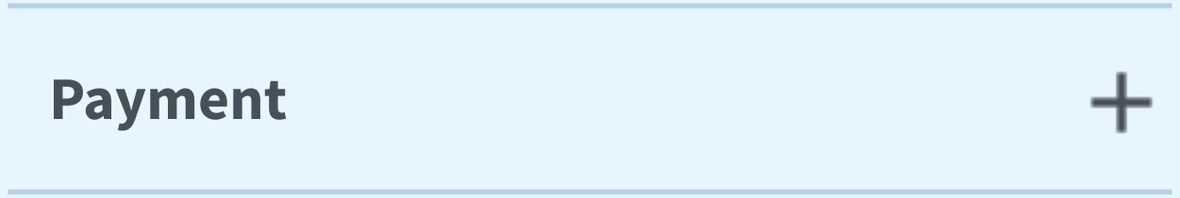
```
<h3>
    <button>
      Payment
    </button>
</h3>
<section>
  …
</section>
```

Make trigger usable to mouse and keyboard users.

Payment +

```
<h3>
    <button>
      Payment
      <svg aria-hidden="true"> … </svg>
    </button>
</h3>
<section>
  …
</section>
```

Add icon and make it decorative.




```
<h3>
    <button id="accordion-trigger-1">
      Payment
      <svg aria-hidden="true"> ... </svg>
    </button>
</h3>
<section aria-labelledby="accordion-trigger-1">
  …
</section>
```

Defines the accessible name
(required in region landmark).

Payment +

```
<h3>
  <button aria-controls="accordion-1"
          id="accordion-trigger-1">
    Payment
    <svg aria-hidden="true"> ... </svg>
  </button>
</h3>
<section id=“accordion-1"
         aria-labelledby=“accordion-trigger-1">
  …
</section>
```

Points to the ID of the panel which the header controls.

Payment +

```
<h3>
  <button aria-expanded="false"
          aria-controls="accordion-1"
          id="accordion-trigger-1">
    Payment
    <svg aria-hidden="true"> ... </svg>
  </button>
</h3>
<section id="accordion-1"
         hidden
         aria-labelledby="accordion-trigger-1">
  …
</section>
```

Add state to screen readers.
When closed, hide content from screen reader users.

Payment —

…

```
<h3>
  <button aria-expanded="true"
          aria-controls="accordion-1"
          id="accordion-trigger-1">
    Payment
    <svg aria-hidden="true"> … </svg>
  </button>
</h3>
<section id=“accordion-1"
         aria-labelledby=“accordion-trigger-1">
  …
</div>
```

When opened, remove the `hidden` *attribute.*

Where to go from here

Where to go from here

1 *Be an a11y advocate within your own projects .*

Accessibility is a team's job. You need to get everyone on board.

Where to go from here

1 *Be an a11y advocate within your own projects .*

Accessibility is a team's job. You need to get everyone on board.

2 *Incrementally improve your projects.*

Slow and steady wins the race.
Start by forcing yourself to only use keyboard, for example.

Weekly tips on Twitter

@lucalanca

aditus.io **joao@aditus.io**

That’s a wrap. Thank you.

Weekly tips on Twitter

@lucalanca

aditus.io **joao@aditus.io**