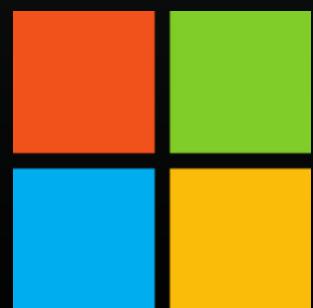


# Performance as User Experience



Aaron Gustafson  
@AaronGustafson  
[slideshare.net/AaronGustafson](http://slideshare.net/AaronGustafson)

/də'zīn/

JEFF VEEN

---

“I’ve been amazed at how often those outside the discipline of design assume that what designers do is decoration—likely because so much bad design simply is decoration. Good design isn’t. **Good design is problem solving.**”

/'frikSH(ə)n/

# Hallmarks of Good UX

- Streamlined flow

# Hallmarks of Good UX

- Streamlined flow
- Clear, concise copy

# Hallmarks of Good UX

- Streamlined flow
- Clear, concise copy
- Low cognitive load

# Hallmarks of Good UX

- Streamlined flow
- Clear, concise copy
- Low cognitive load
- Fast performance

[Store](#)[Mac](#)[iPod](#)[iPhone](#)[iPad](#)[iTunes](#)[Support](#)

# iPad

[Features](#)[Built-in Apps](#)[From the App Store](#)[iOS](#)[iCloud](#)[Tech Specs](#)[Pre-order Now](#)

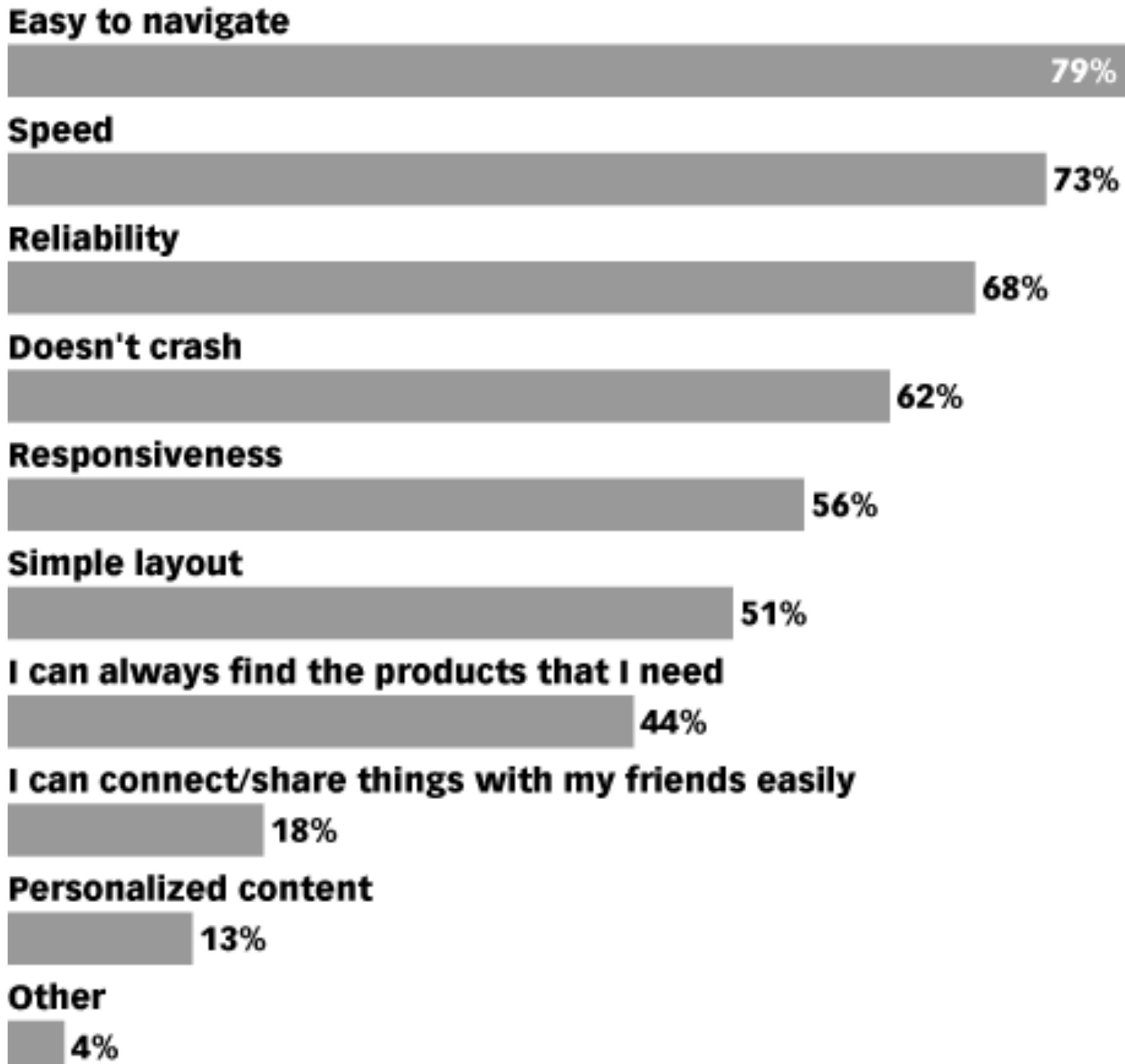
# Resolutionary

Introducing the new iPad. With the stunning Retina display.  
5MP iSight camera. And ultrafast 4G LTE.  
Starting at \$499.



Poor performance  
is **friction**

**Most Important Attributes of Website Performance  
According to US Internet Users, Sep 2015**  
*% of respondents*



*Note: n=2,044 ages 18+*

*Source: Soasta survey conducted by The Harris Poll as cited in press release, Nov 5, 2015*

A 1s delay in page  
load can reduce  
conversions by 7%

For an online shop earning  
\$100k/day, that's about  
**\$2.5m in lost sales**

For Amazon, 1s is worth  
about **\$1.6b** in sales

---

**Most Frustrating Aspects of the Mobile Shopping Experience According to US Mobile Device Users, Oct 2014**

*% of respondents*

---

**Slow mobile load times**



42%

**Inconsistent user experience across devices**



37%

**Not enough info accessible via mobile devices**



26%

---

*Source: MobiQuity, "The Frustrated Shopper: Mobile/Technology Satisfaction Report," Dec 22, 2014*

---

187033

[www.emarketer.com](http://www.emarketer.com)

Source: [eMarketer](http://www.emarketer.com)

**53% abandon  
websites that take  
more than 3s to load**

By shaving 7s off load,  
**Edmunds increased**  
**page views by 17%**  
**& ad revenue by 3%**

Mozilla reduced page load  
by 2.2s and saw a 15.4%  
increase in downloads

Performance  
matters



10k APART

Inspiring the Web with Just 10k

[Read the Rules](#)

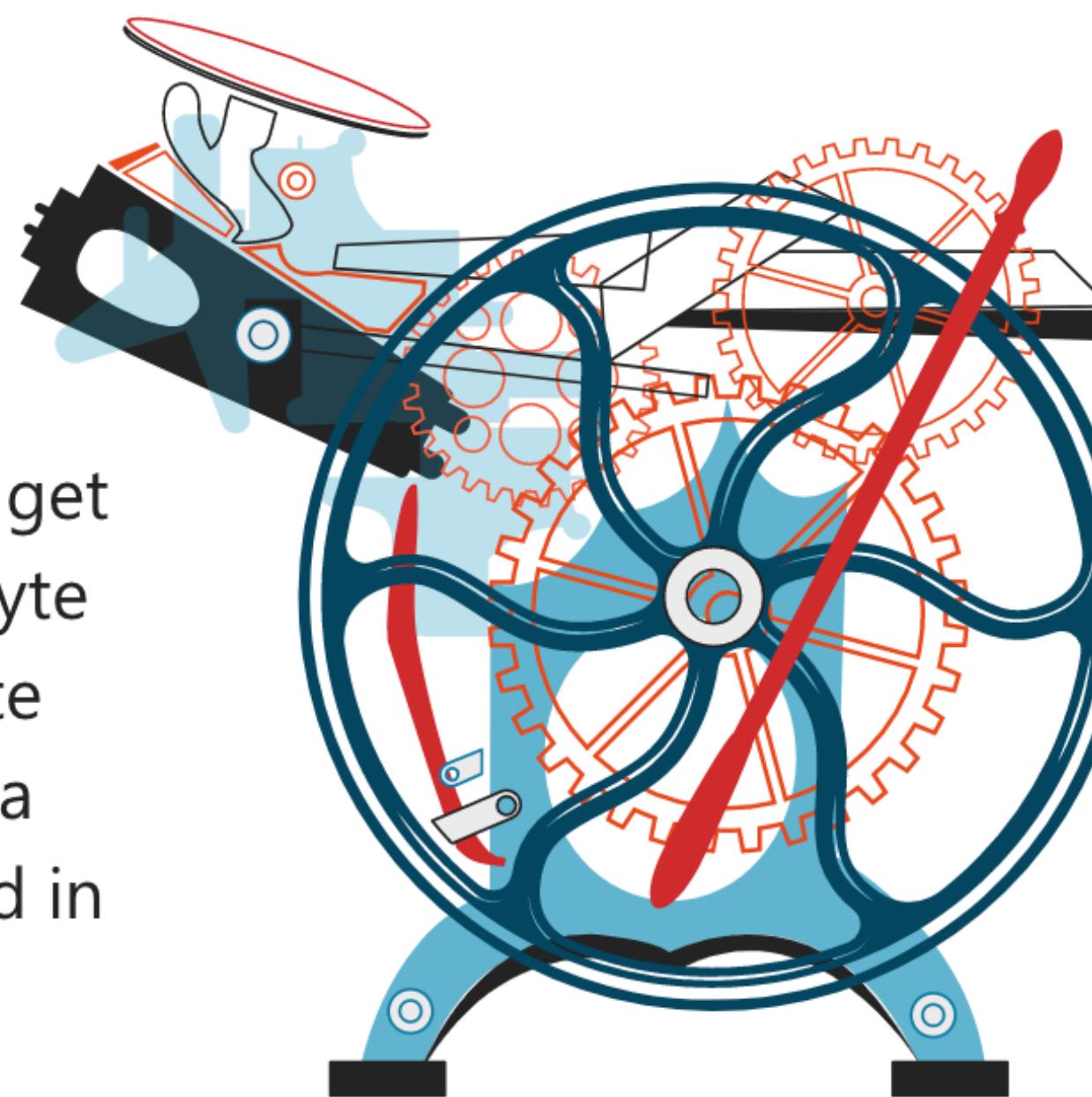
[View the Gallery](#)

[Meet the Judges](#)

*Brought to you by Microsoft and An Event Apart*

# Inspiring the Web with Just 10k

With so much of an emphasis on front-end frameworks and JavaScript runtimes, it's time to get back to basics—back to optimizing every little byte like your life depends on it and ensuring your site can work, no matter what. The Challenge? Build a compelling web experience that can be delivered in 10kB and works without JavaScript.



[View the Winners](#)

[Check Out the Gallery](#)

---

And the winners are...

**Best Overall:**

**Best Design:**

# What we were looking for

# What we were looking for

- Usable pages are delivered in 10kB or less

# What we were looking for

- Usable pages are delivered in 10kB or less
- Pages are accessible via screen readers, the keyboard, etc.

# What we were looking for

- Usable pages are delivered in **10kB or less**
- Pages are **accessible** via screen readers, the keyboard, etc.
- Entries follow the philosophy of **progressive enhancement**

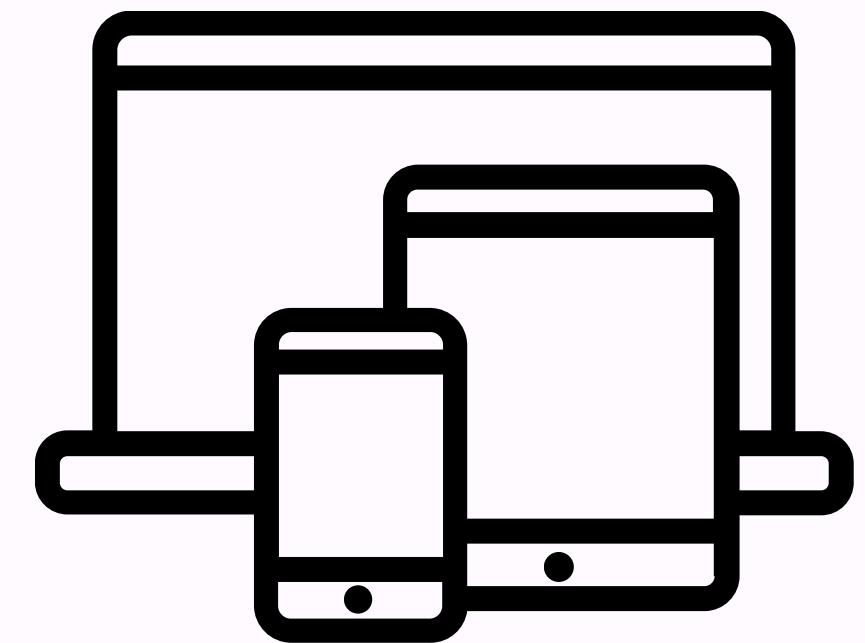
# What we were looking for

- Usable pages are delivered in 10kB or less
- Pages are accessible via screen readers, the keyboard, etc.
- Entries follow the philosophy of progressive enhancement
- Users can do what they need to without JavaScript

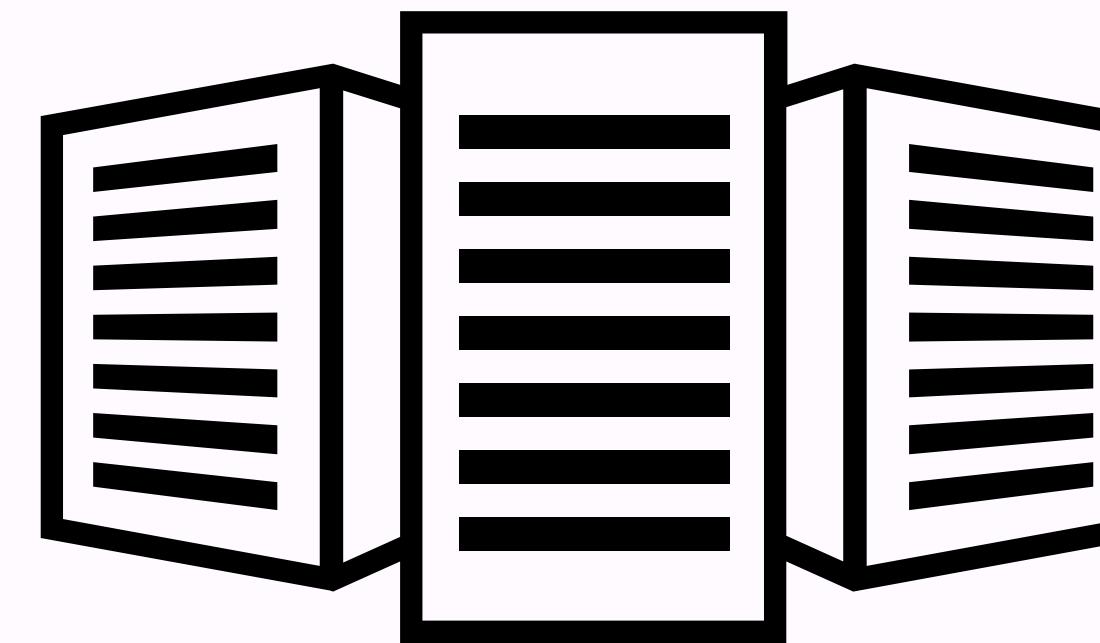
/dôg'fôodîNG/

Let's talk (briefly)  
about page load

# Your Device



# The Web

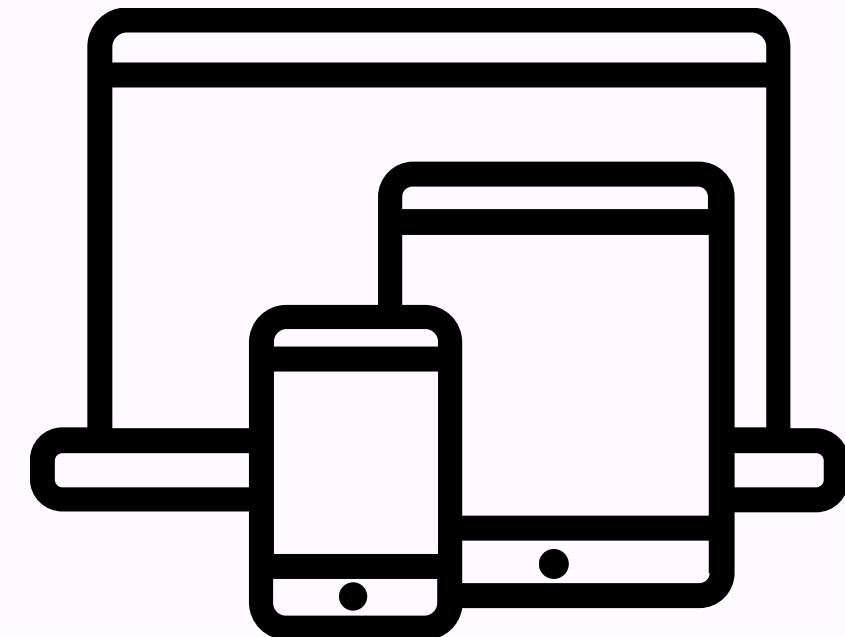


time

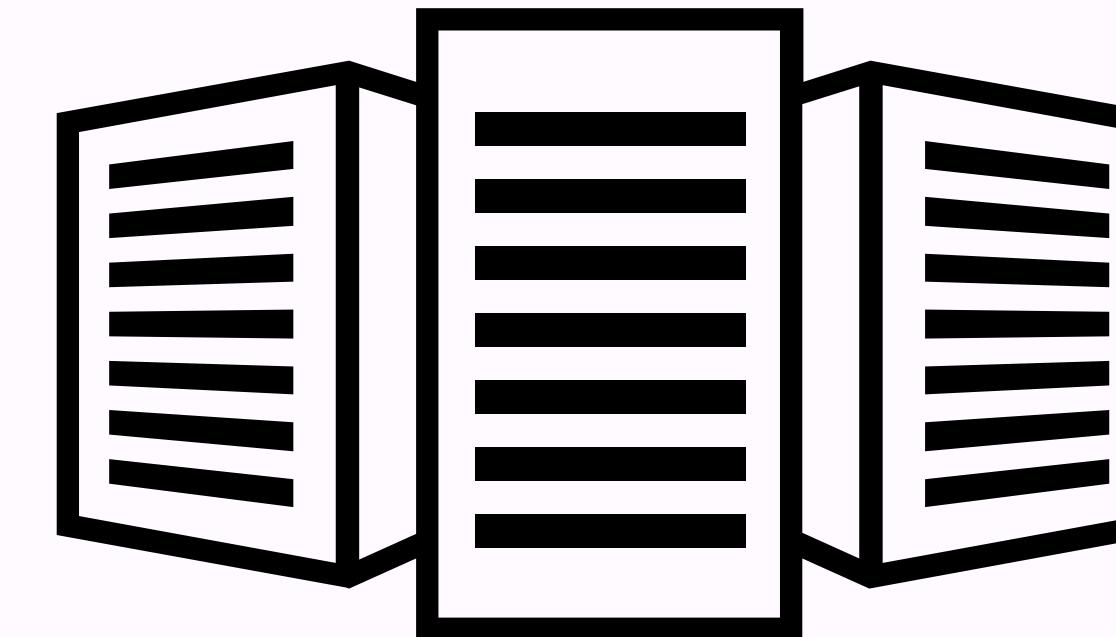


## DNS Lookup

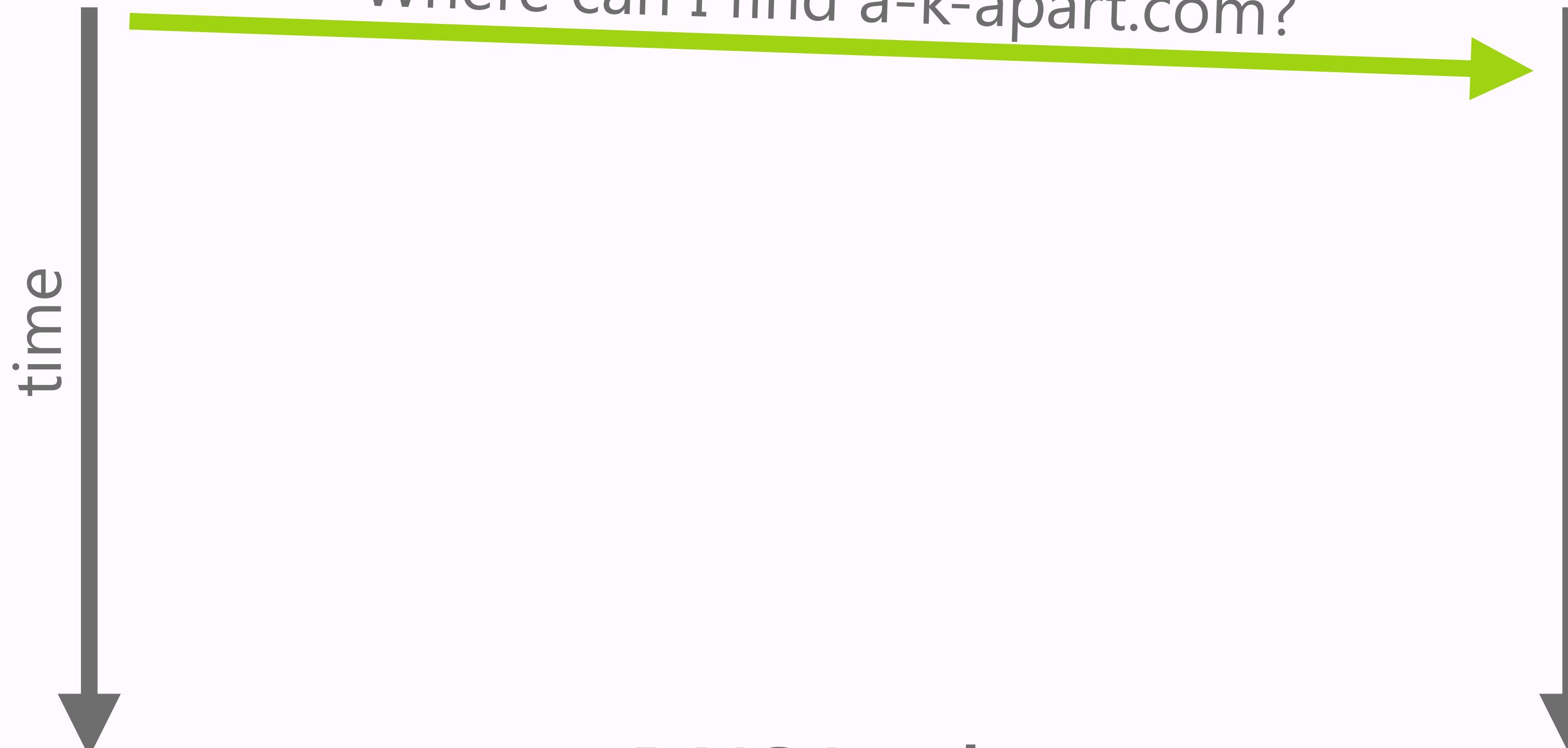
# Your Device



# The Web



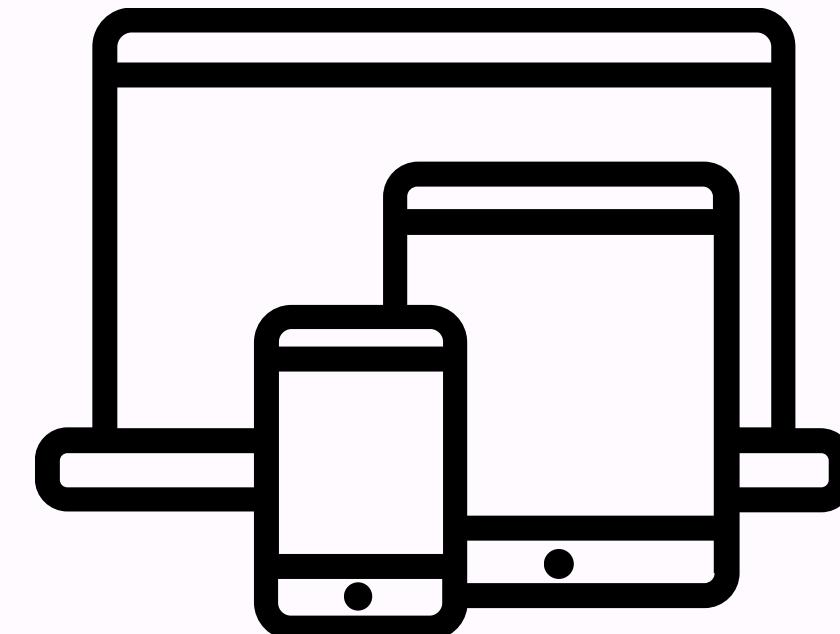
Where can I find a-k-apart.com?



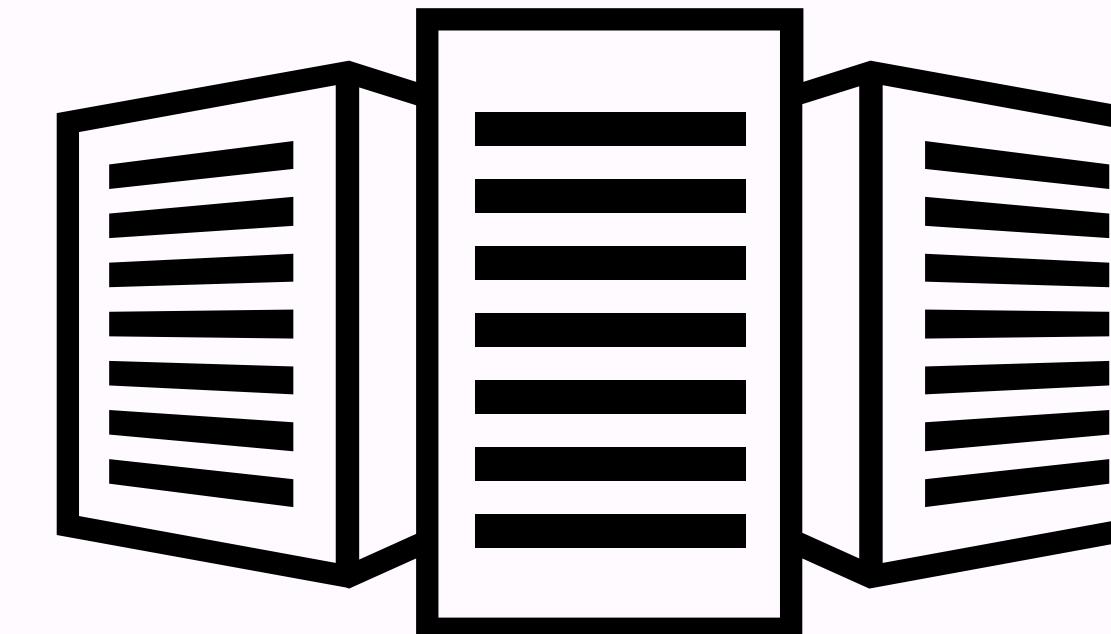
## DNS Lookup

Icons by [Mahmure Alp](#)

# Your Device

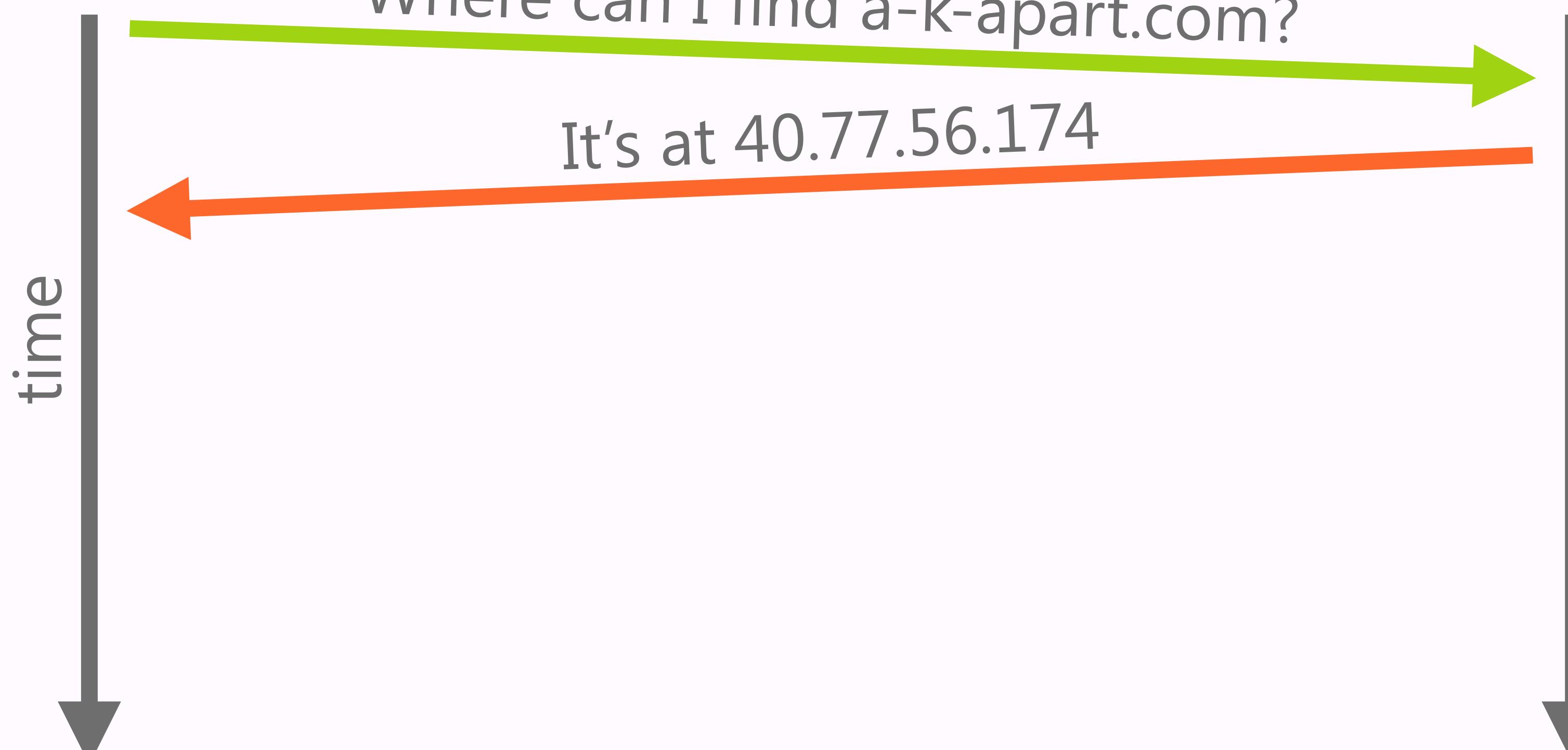


# The Web



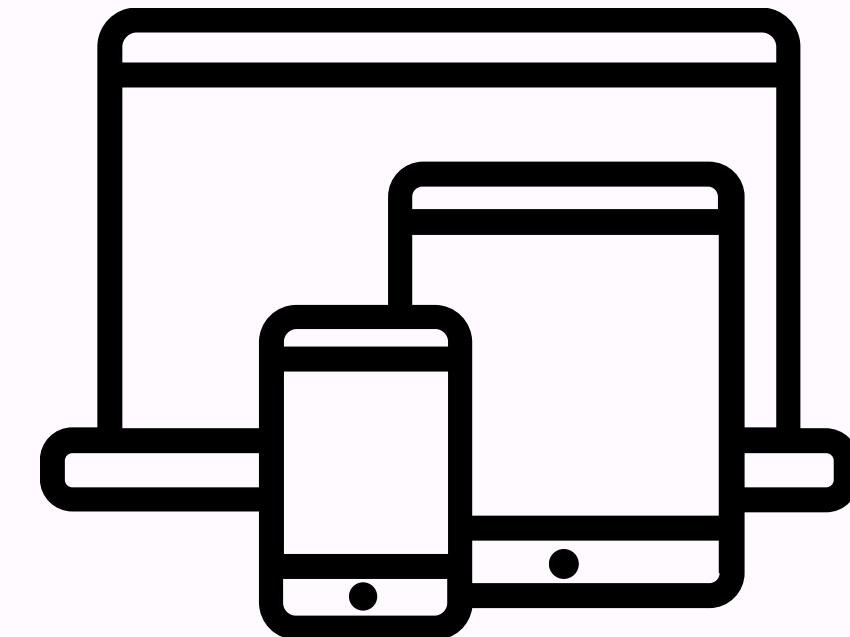
Where can I find a-k-apart.com?

It's at 40.77.56.174

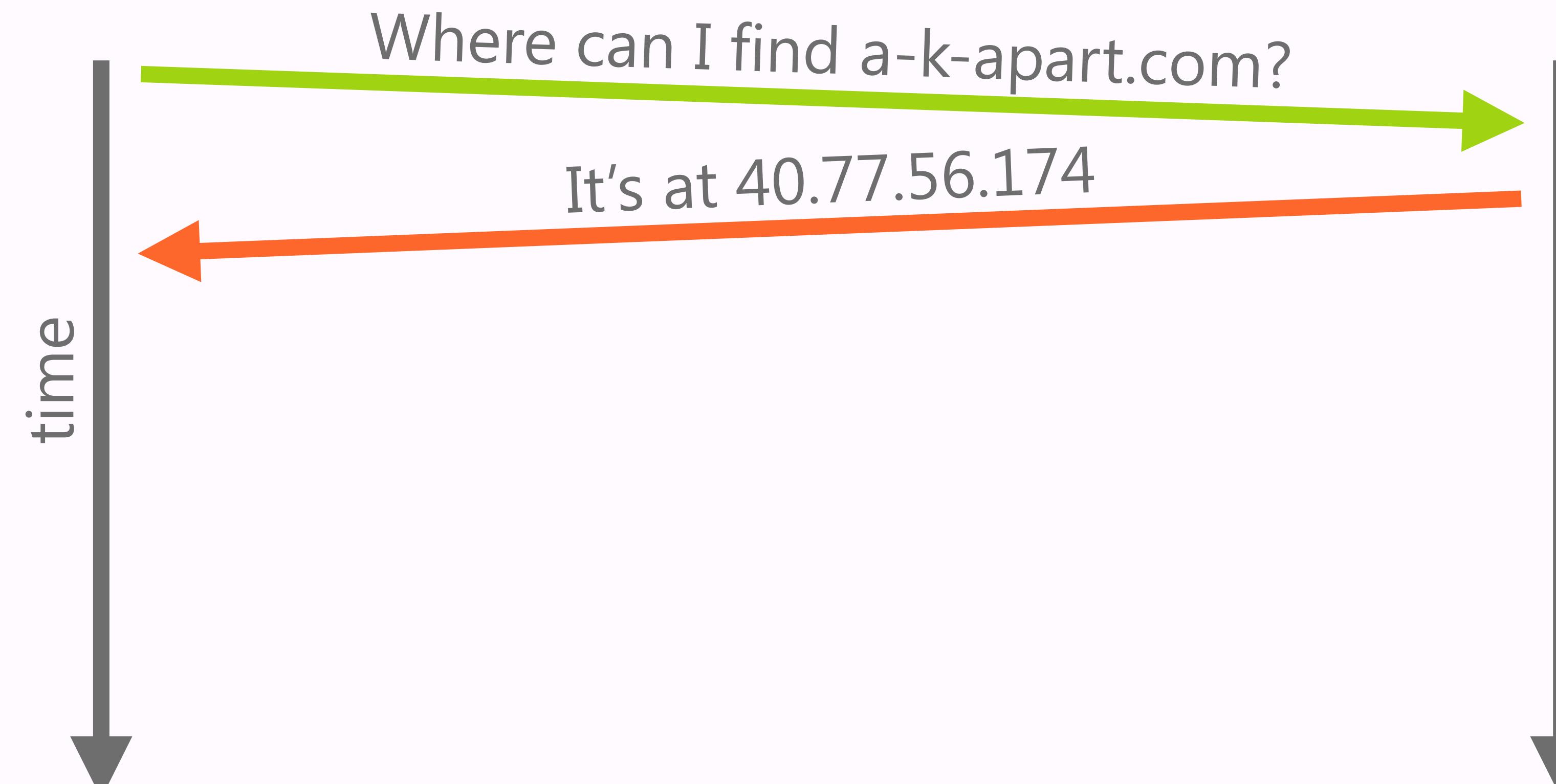
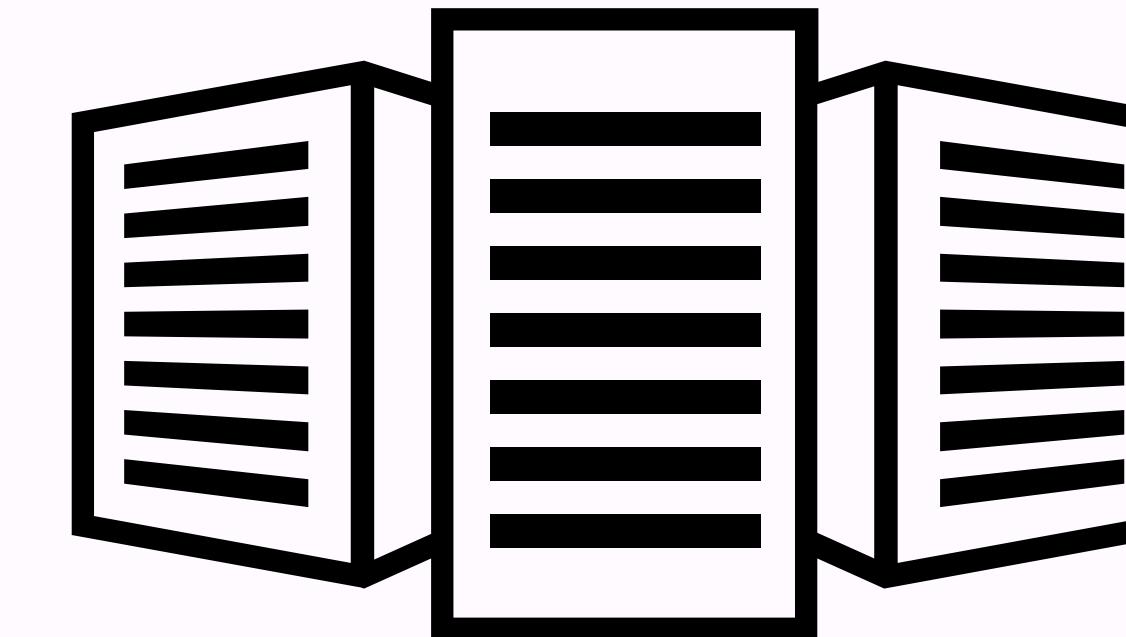


DNS Lookup

# Your Device

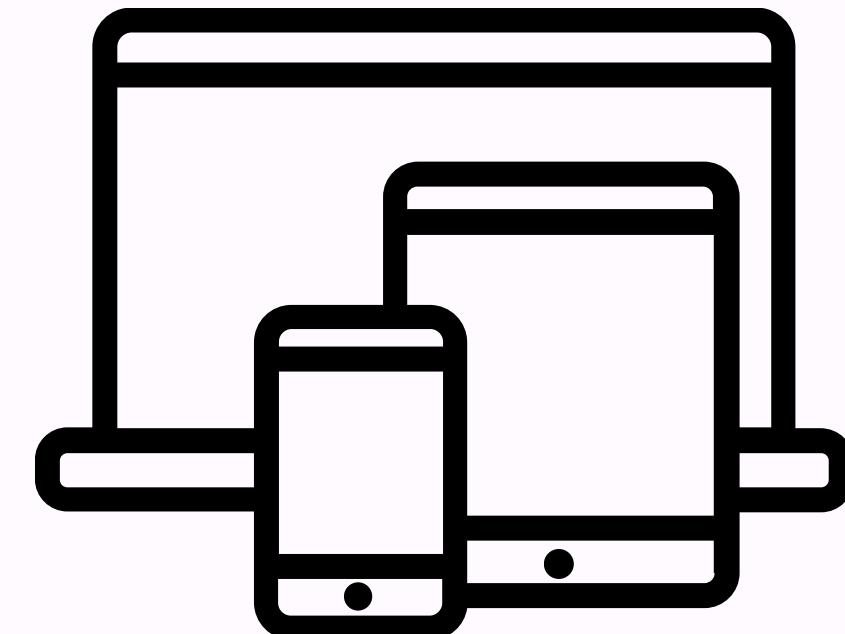


# The Web

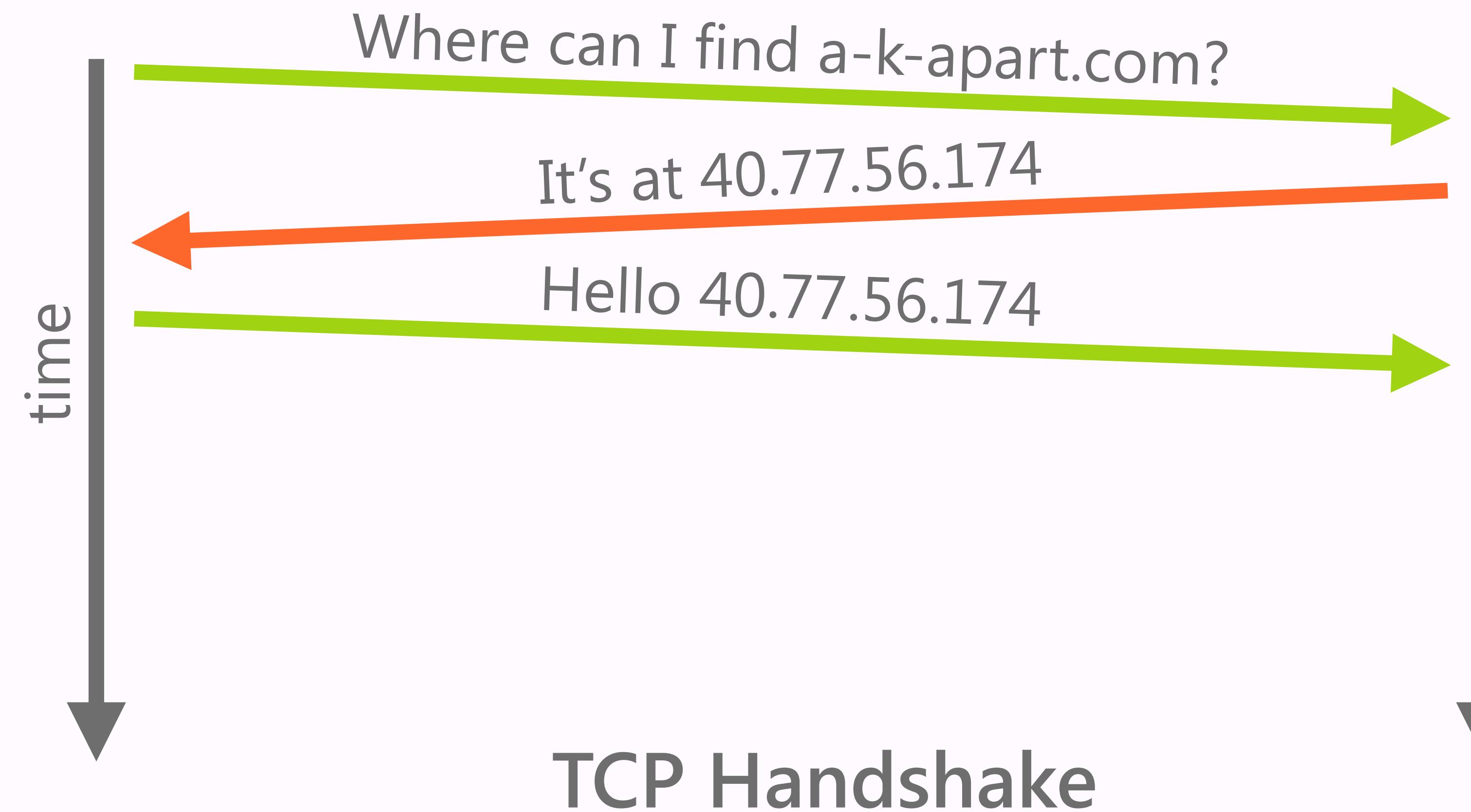
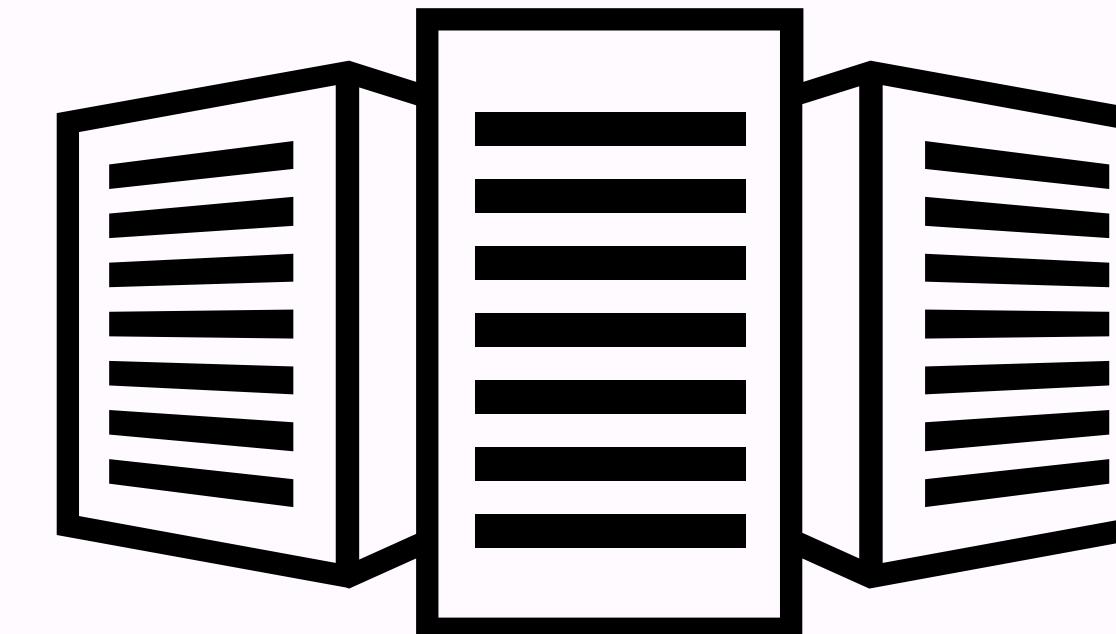


TCP Handshake

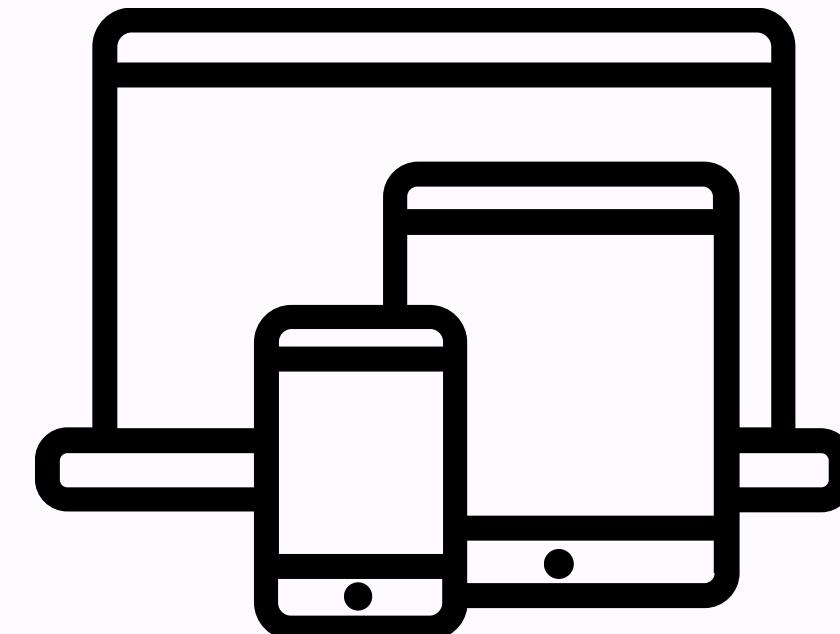
# Your Device



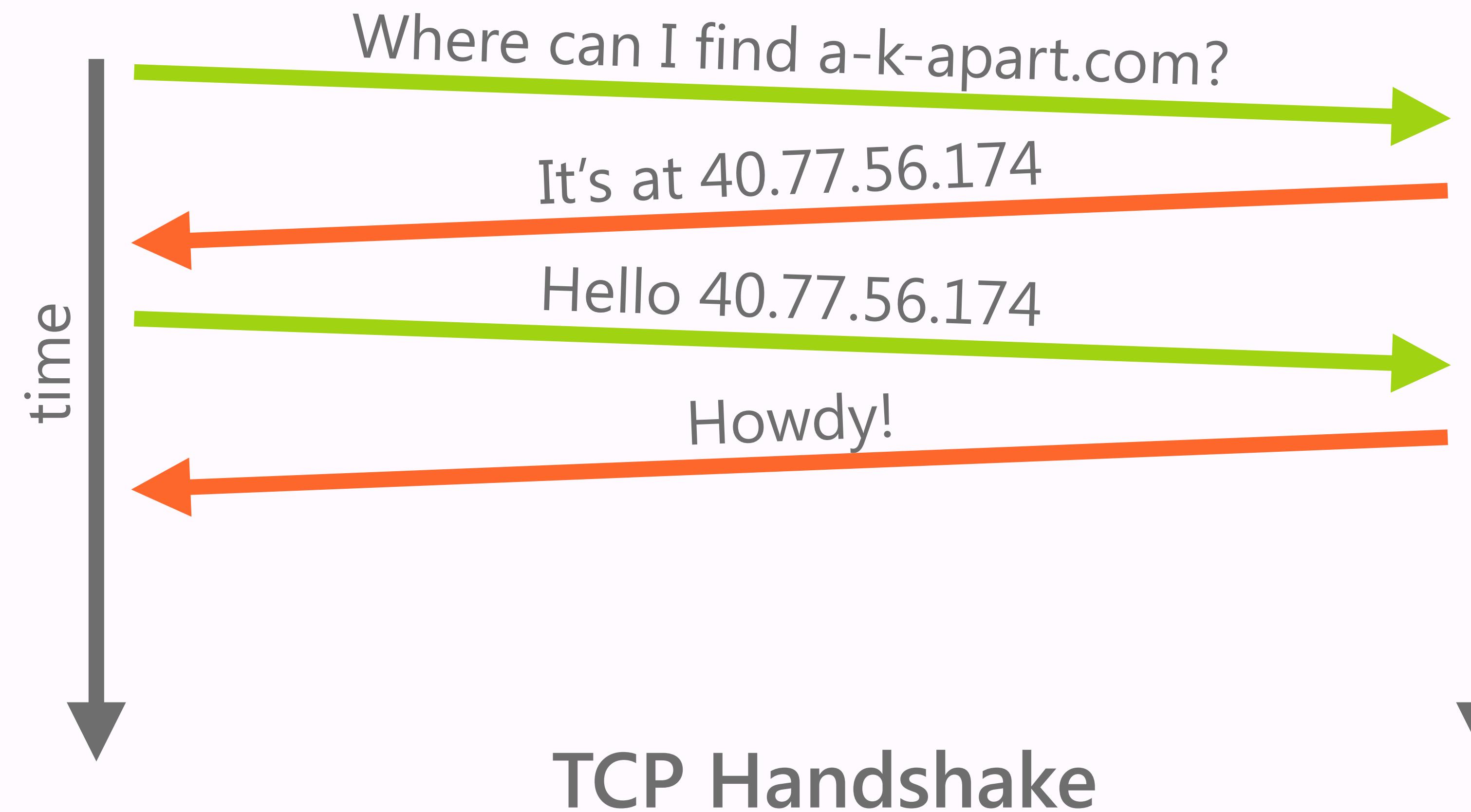
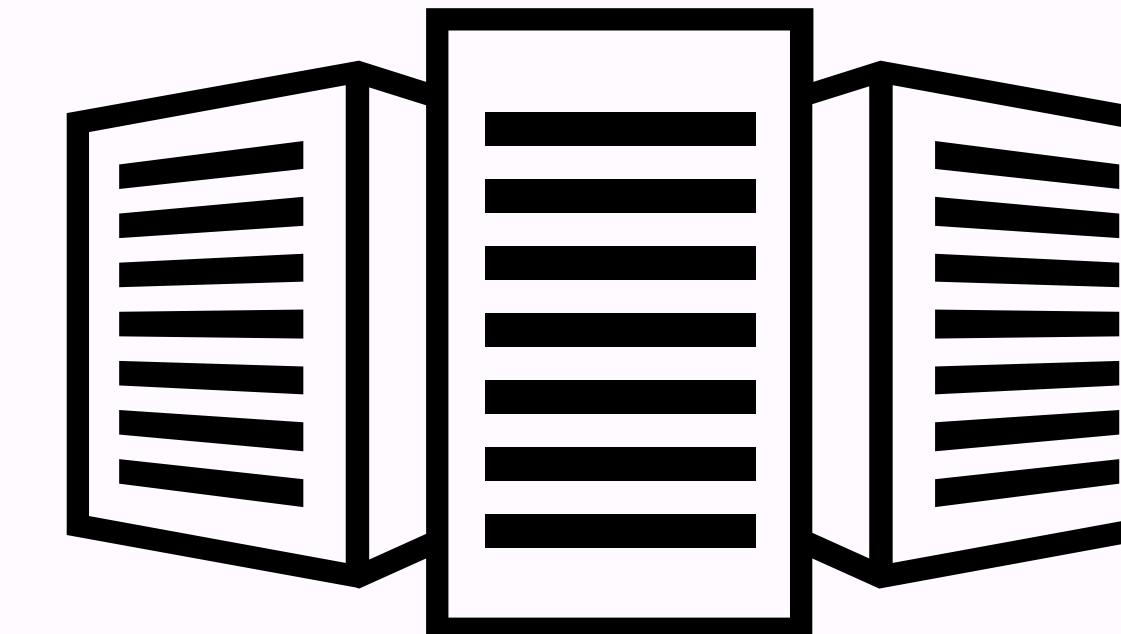
# The Web



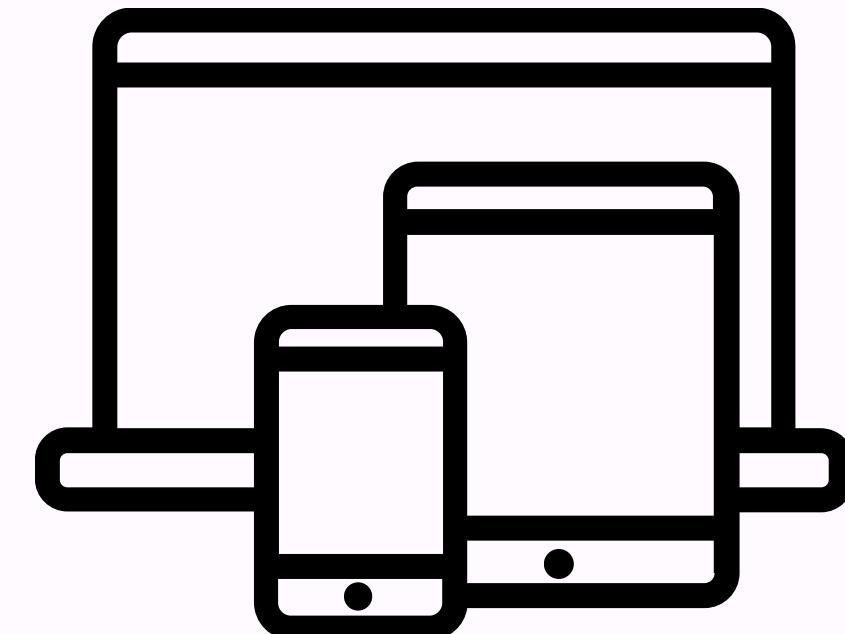
# Your Device



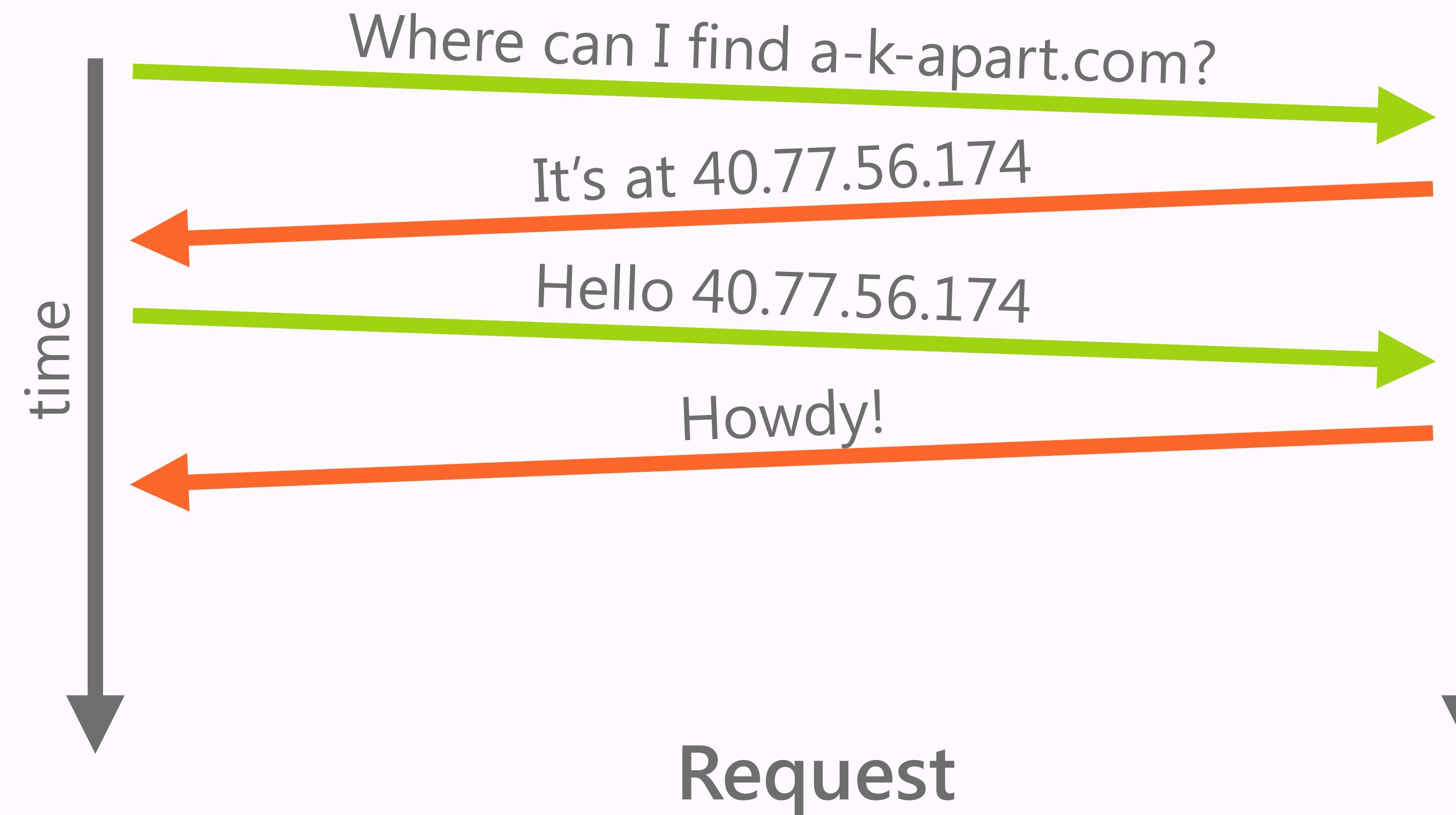
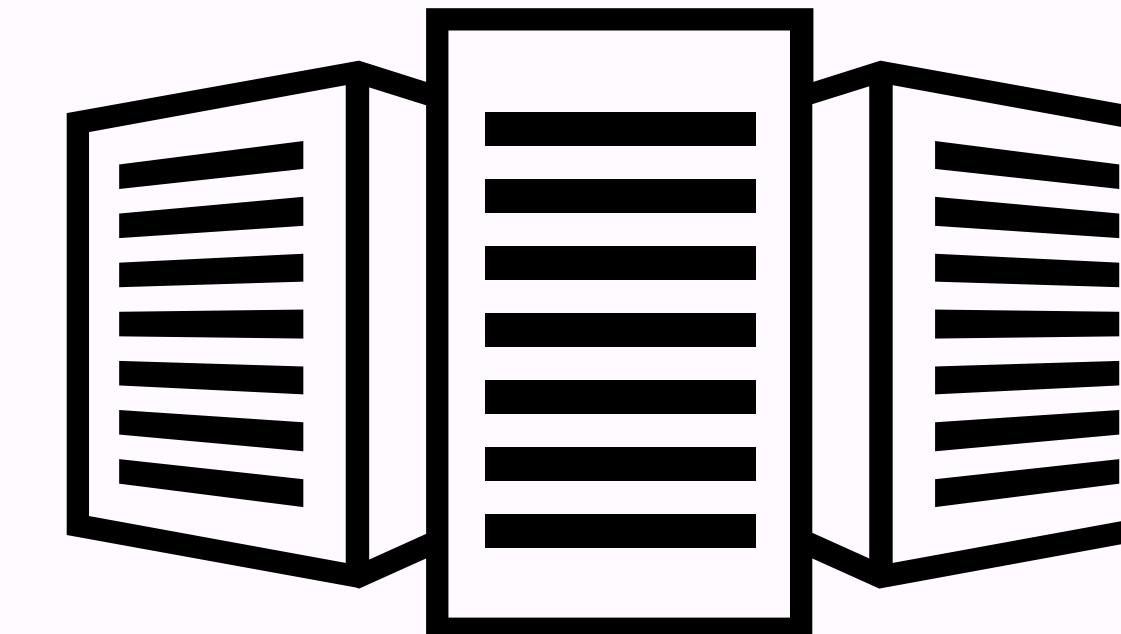
# The Web



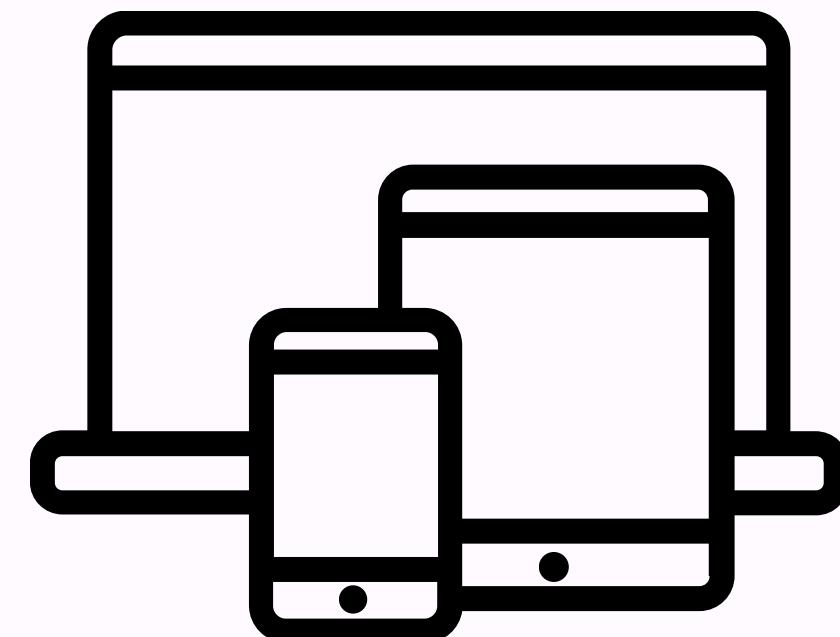
# Your Device



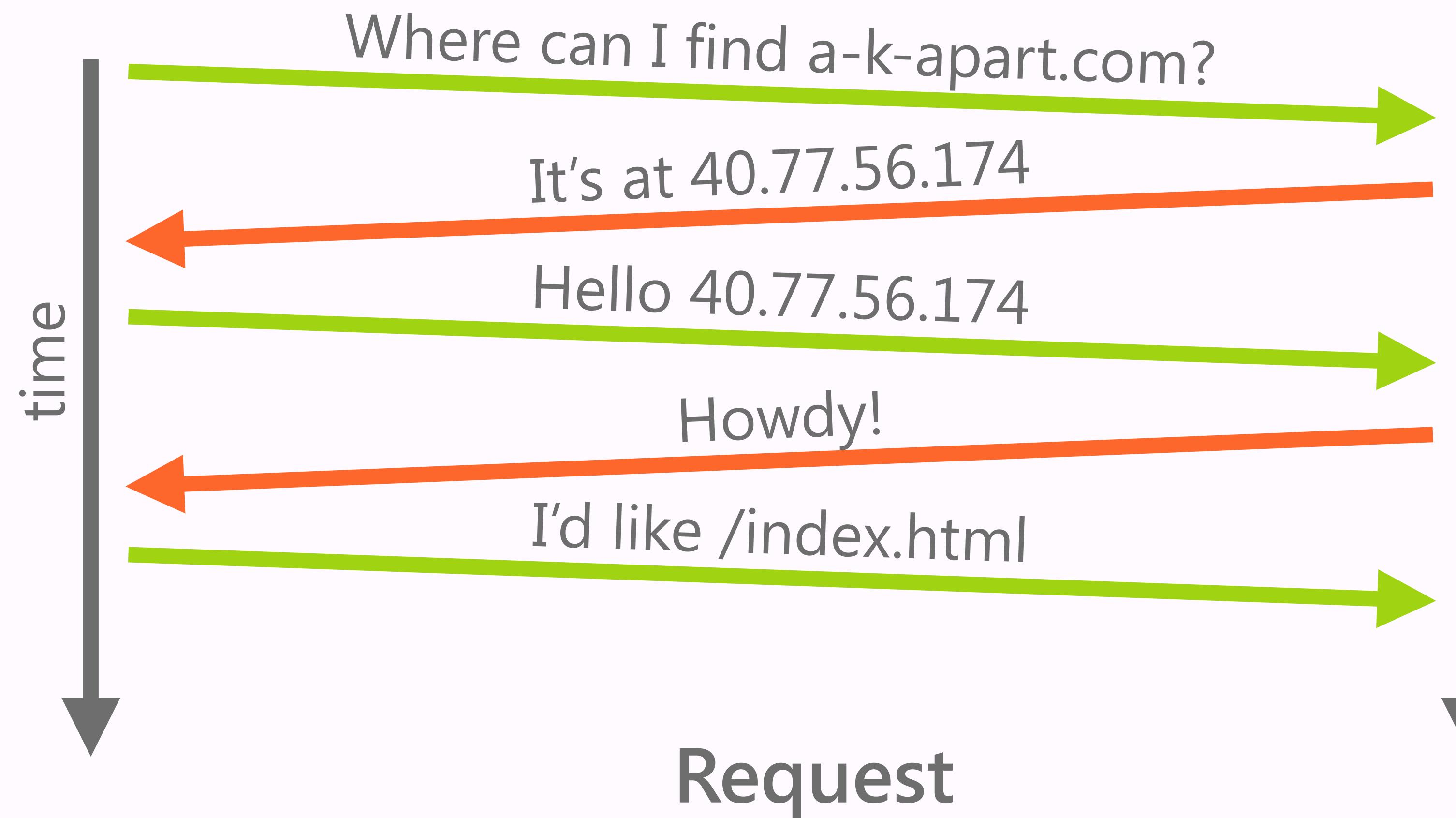
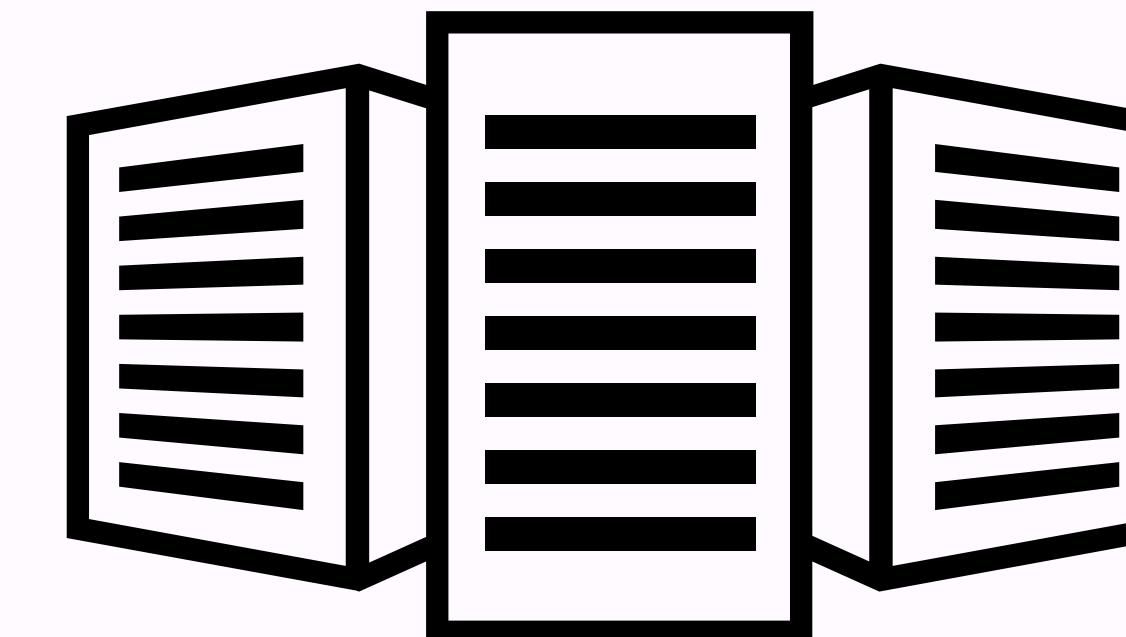
# The Web



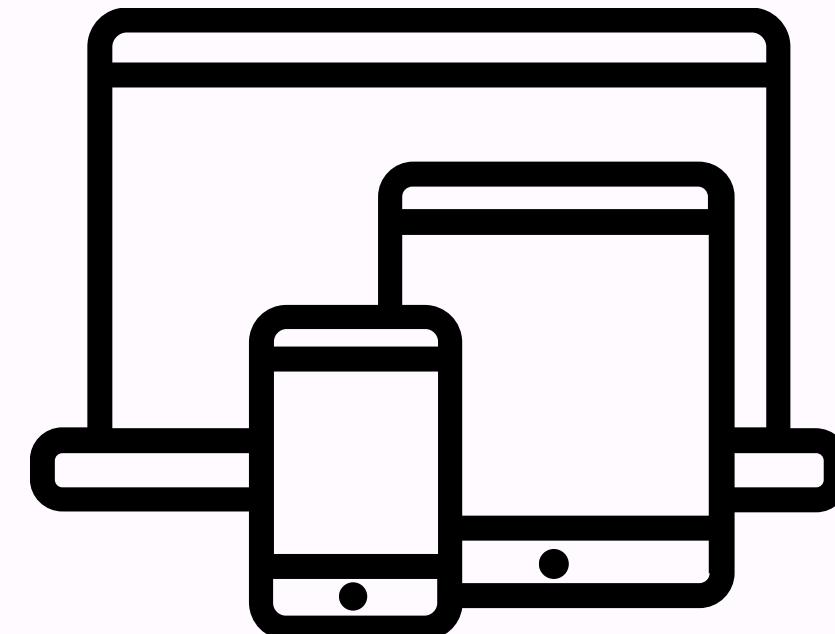
# Your Device



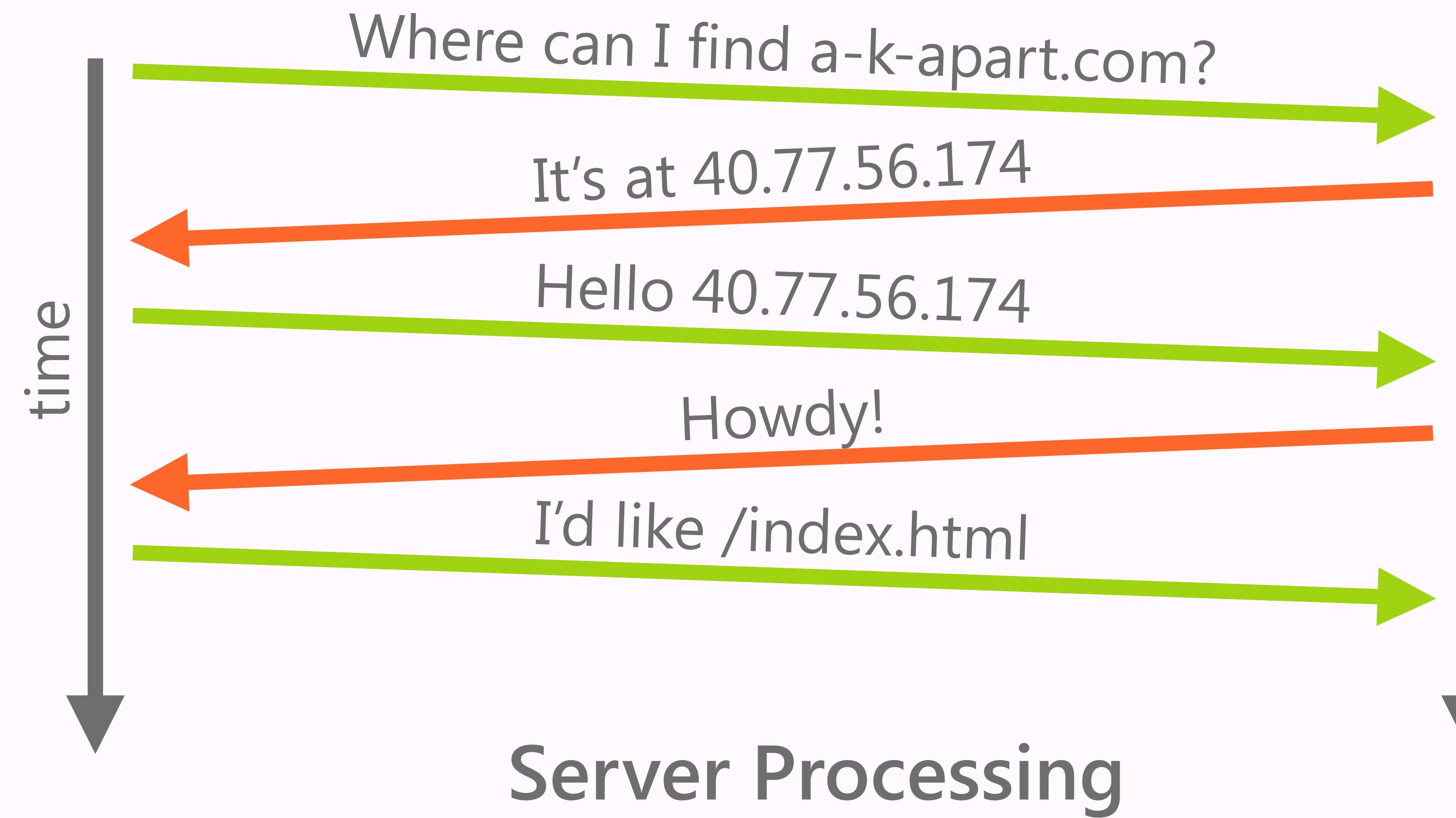
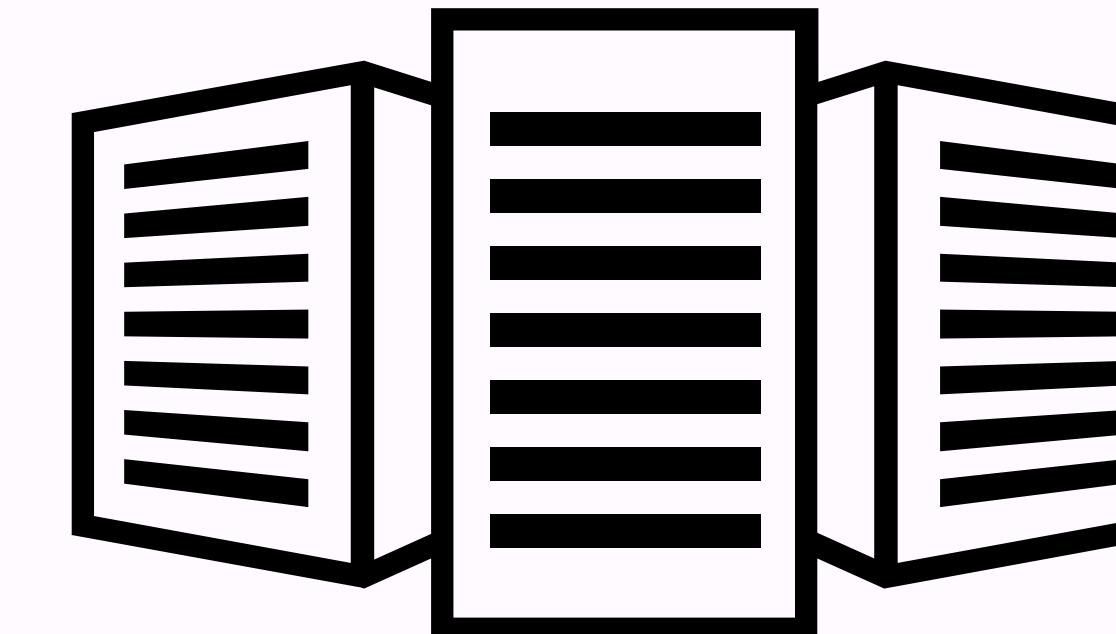
# The Web



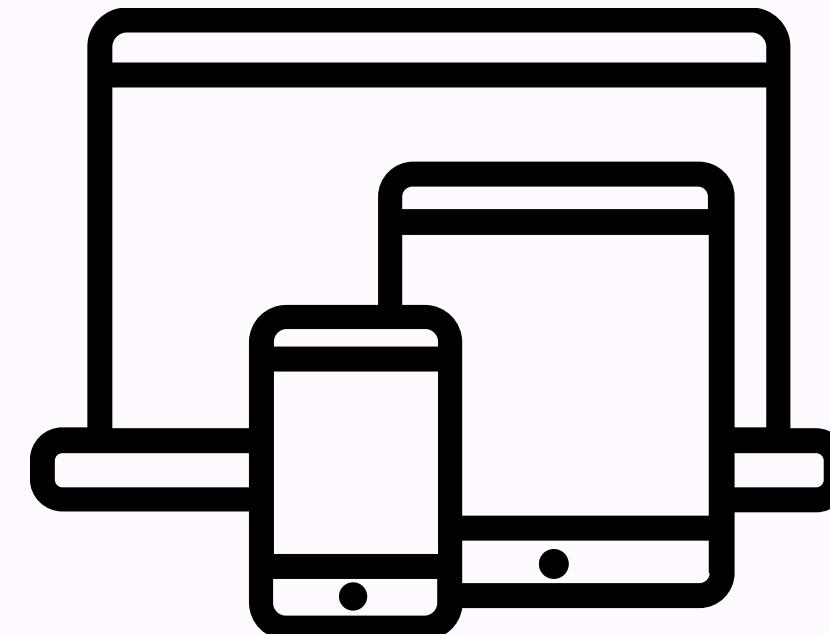
# Your Device



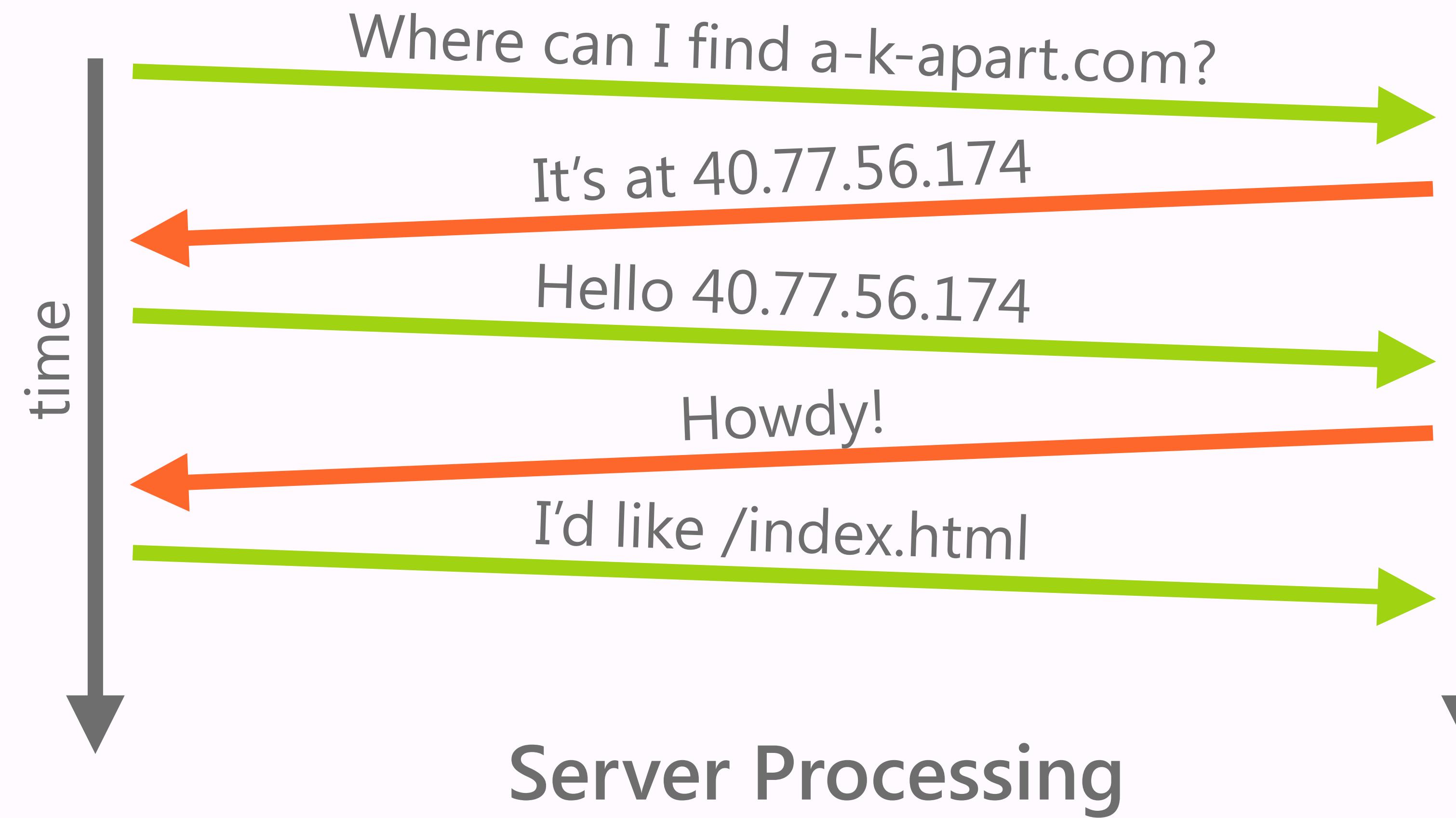
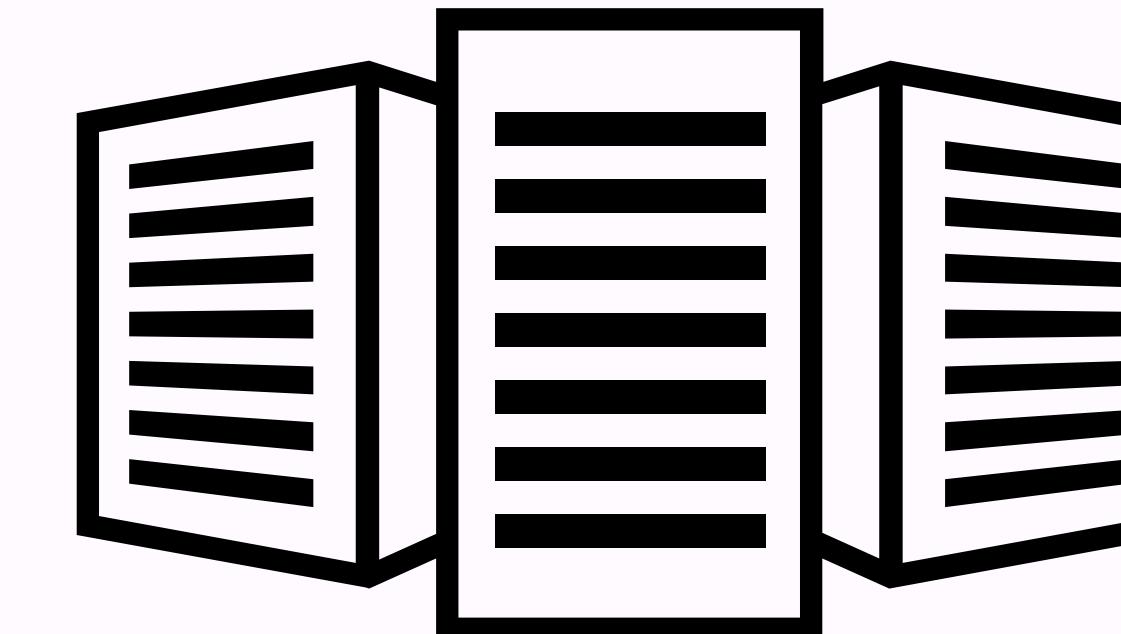
# The Web



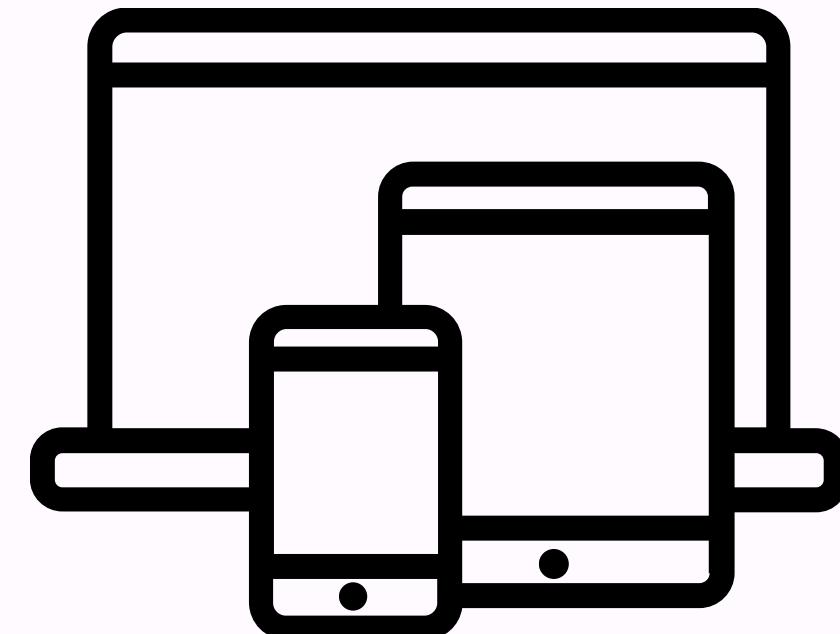
# Your Device



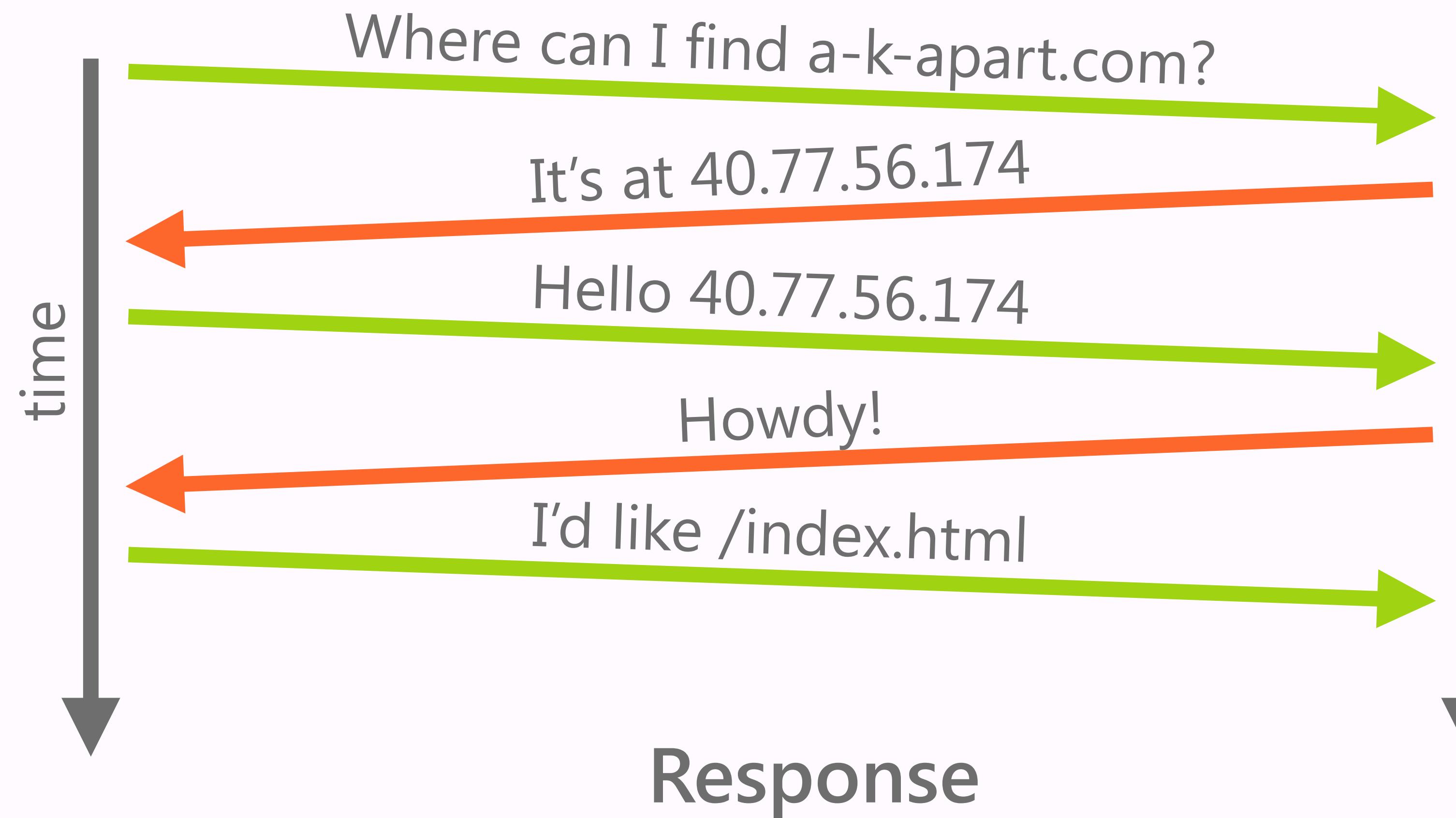
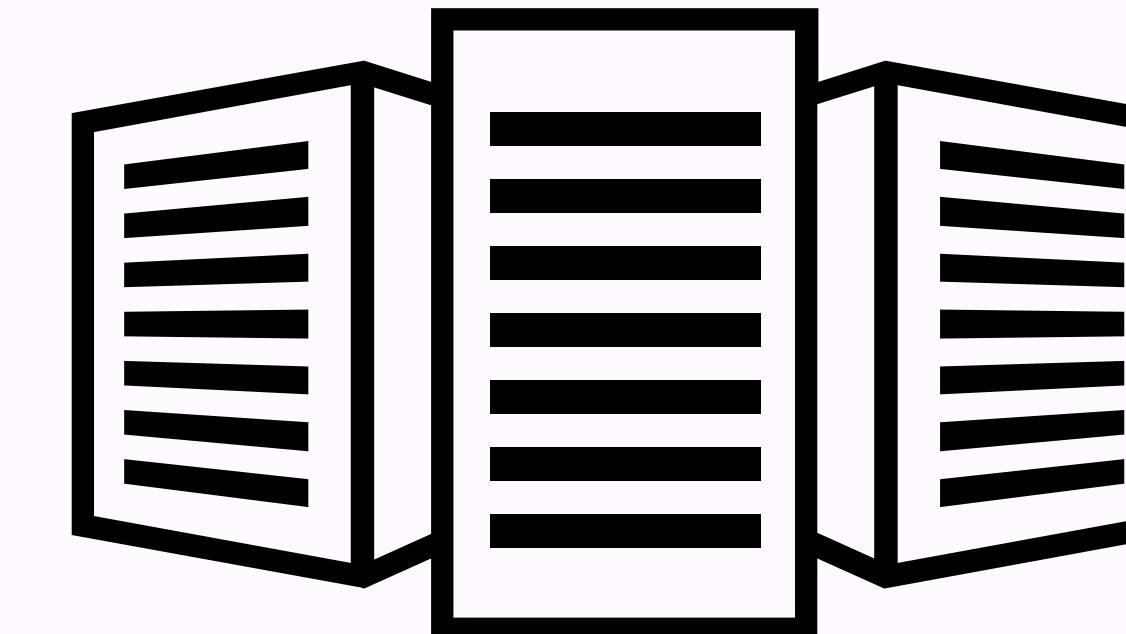
# The Web



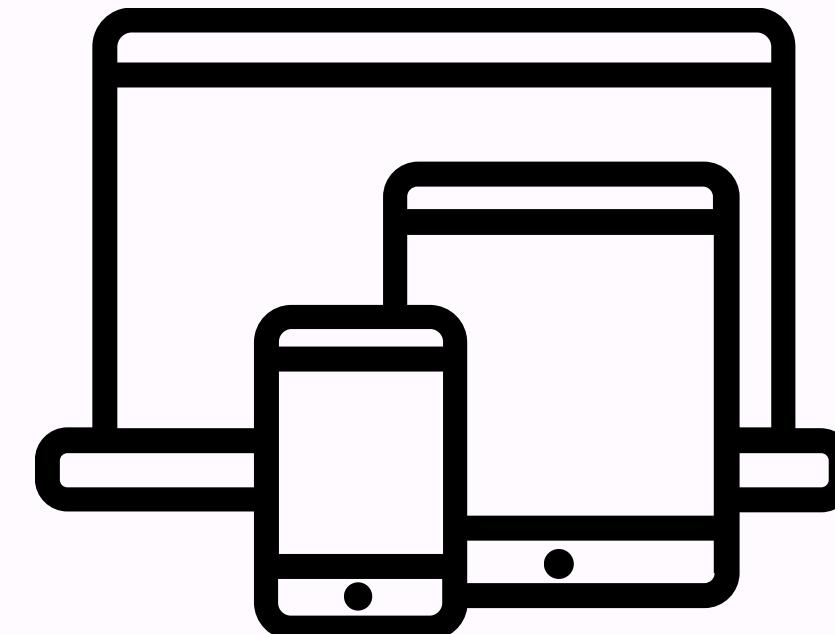
# Your Device



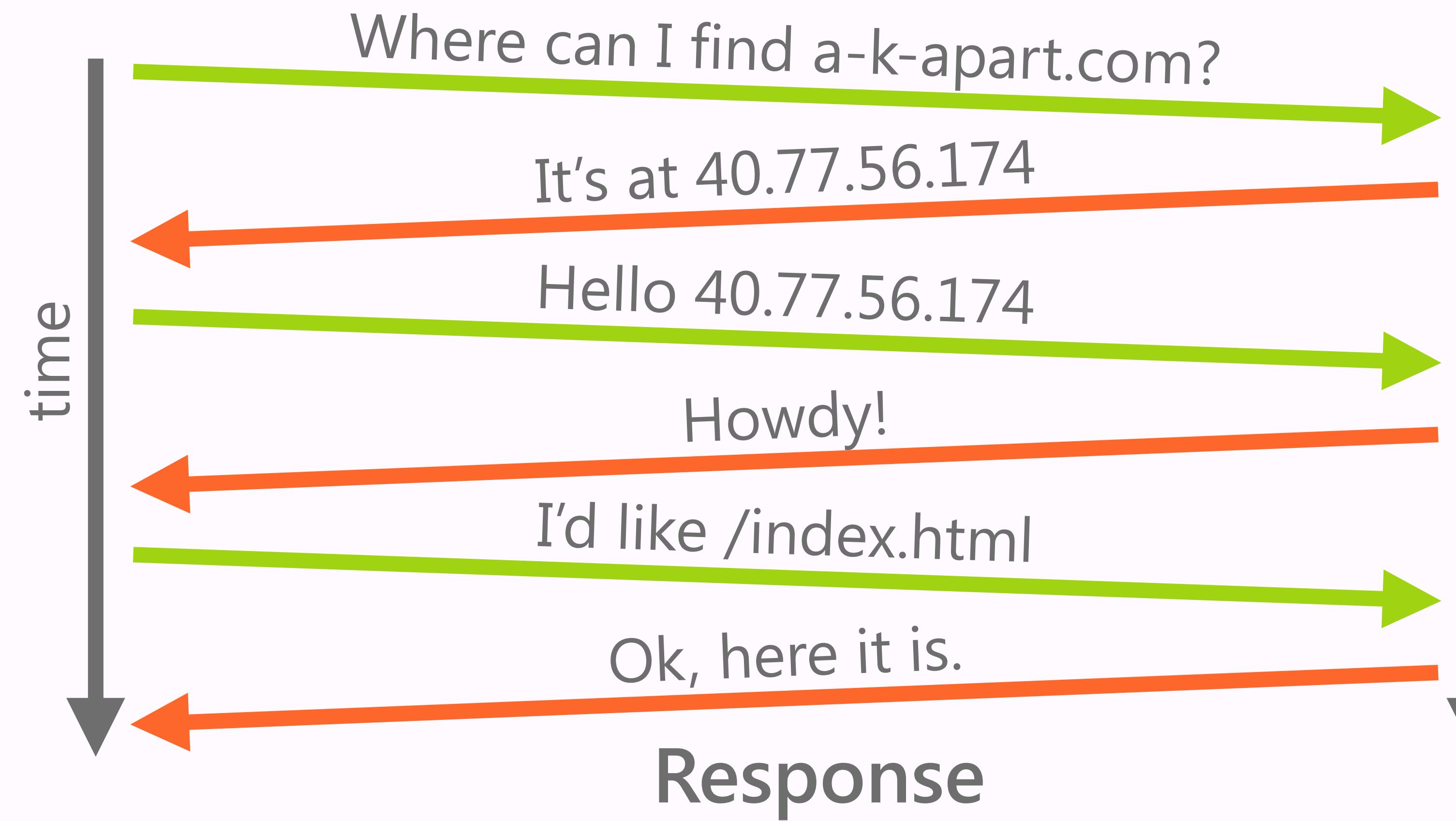
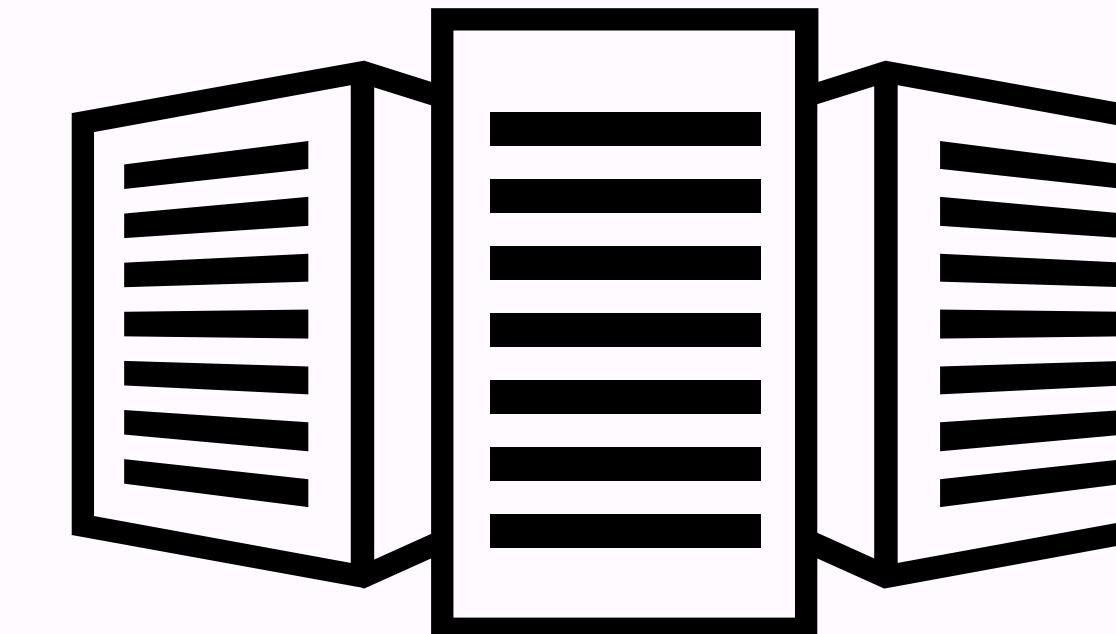
# The Web

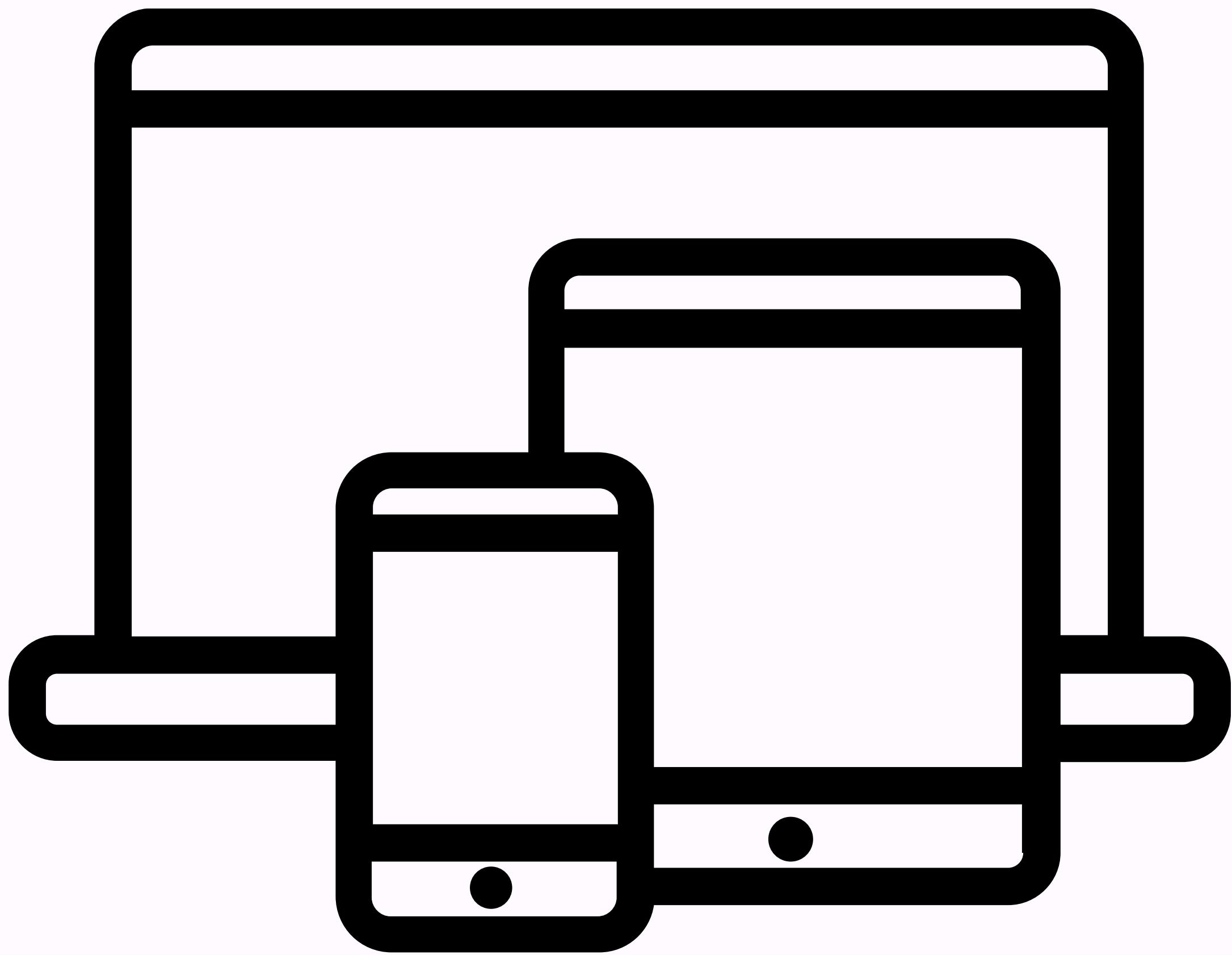


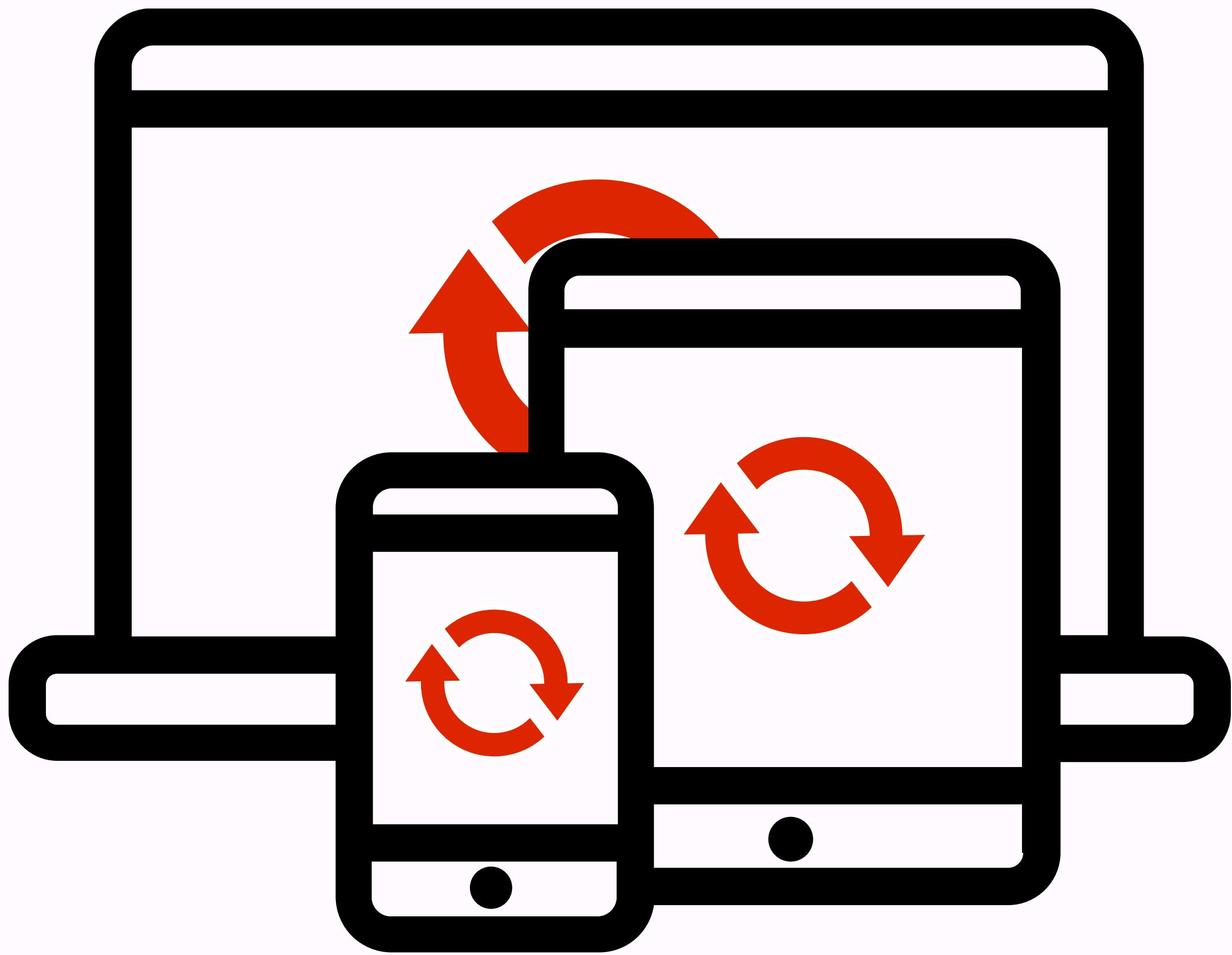
# Your Device



# The Web









Response



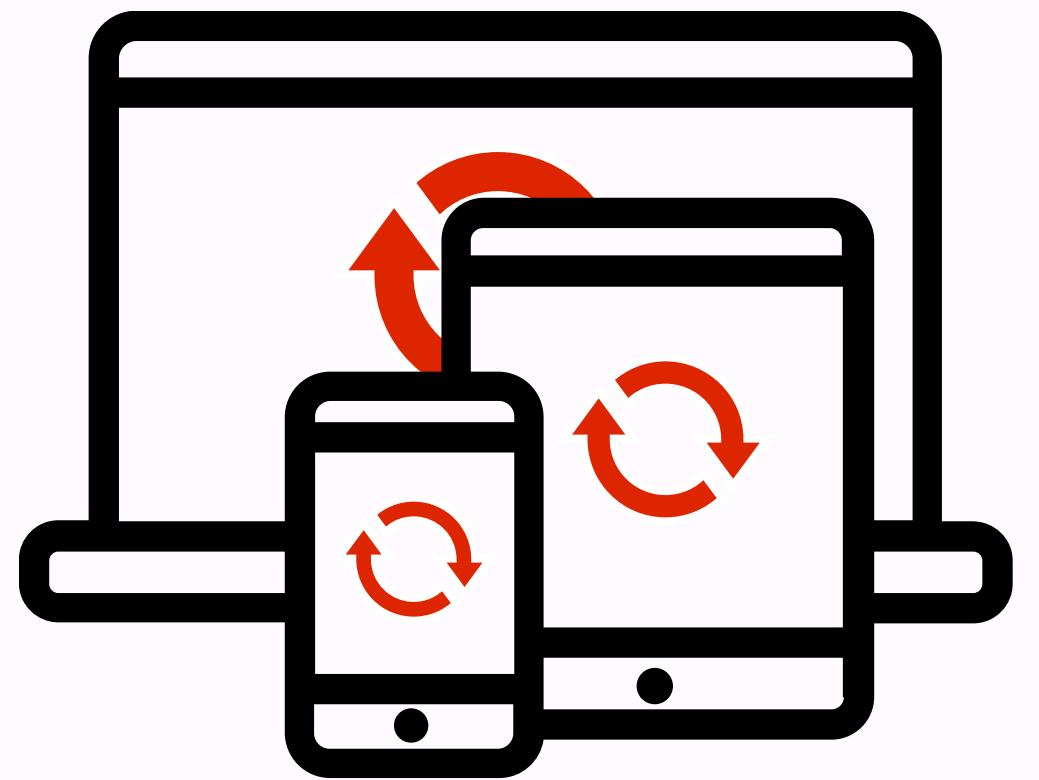
Load



Response

DOM

Load



Response

DOM

CSS

Load



Response

DOM

CSS

Render Tree

Load



Response

DOM

CSS

Render Tree

Layout

Load



Response

DOM

CSS

Render Tree

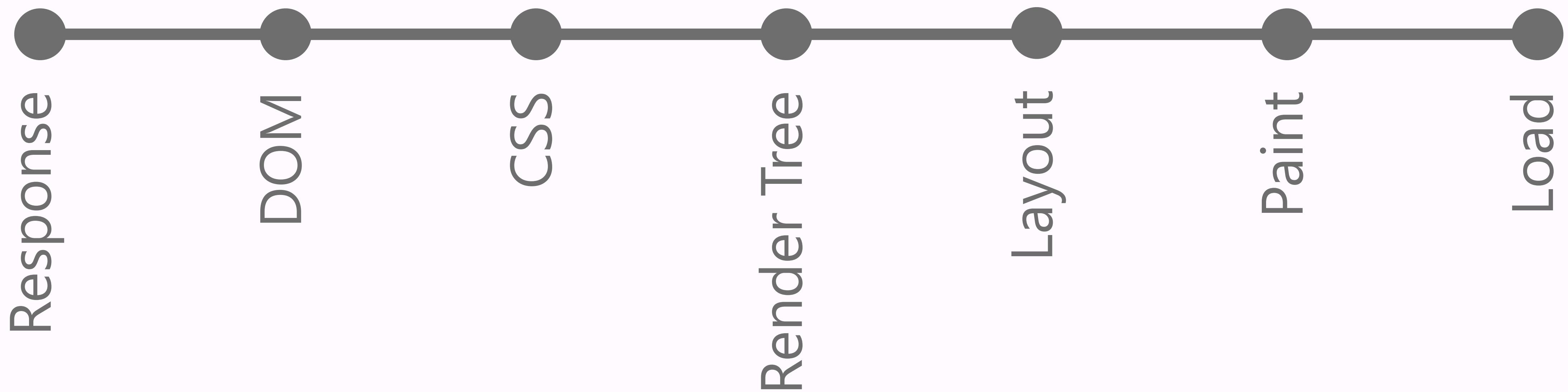
Layout

Paint

Load



CSS & JavaScript can  
delay rendering or cause  
these to be run again





CSS & JavaScript can  
delay rendering or cause  
these to be run again

Response

DOM

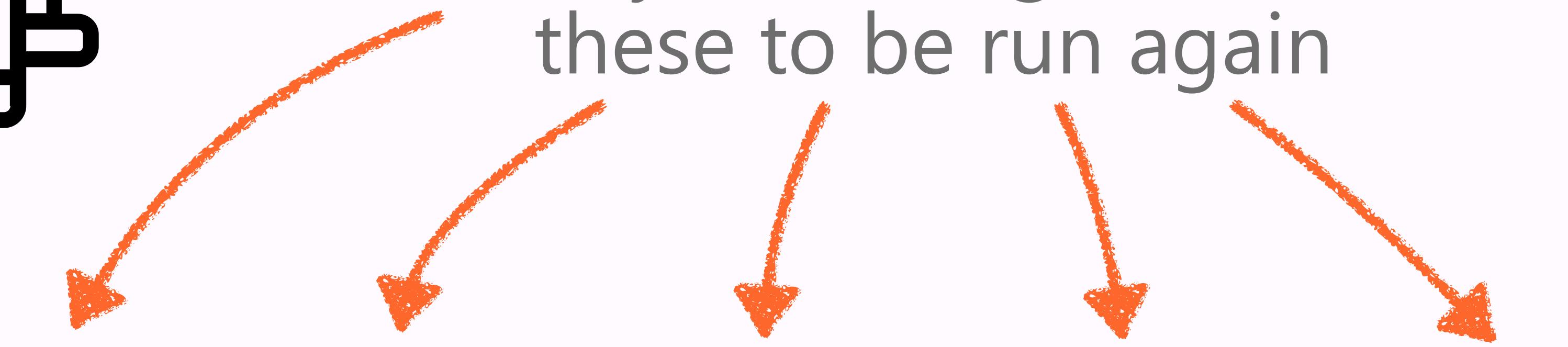
CSS

Render Tree

Layout

Paint

Load



# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script> ←
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

*Wait while the browser fetches & parses the script*

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

wait while the browser fetches & parses the CSS

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
     ←
    <script src="body.js"></script>
  </body>
</html>
```

kicks off downloading  
the image

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script> ←
  </body>
</html>
```

wait while the browser  
fetches & parses the script

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# DOM Parsing

```
<html>
  <head>
    <title>Silly example</title>
    <script src="head.js"></script>
    <link rel="stylesheet" href="main.css">
    <style>h2 { font-weight: bold; }</style>
    <script>console.log('hi');</script>
  </head>
  <body>
    
    <script src="body.js"></script>
  </body>
</html>
```

# Steps for better performance

# Steps for better performance

1. Use native features whenever possible

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets
5. Consider how you load assets

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets
5. Consider how you load assets
6. Only load assets when they add value

Step 1:  
**Use native features  
whenever possible**

**(they're effectively free)**

# Por exemplo

```
<header>
  Header content...
</header>
```

# Por exemplo

```
<header>  
  Header content...  
</header>
```

not only shorter than  
<div id="header">, but  
semantic too

# Por exemplo

```
<header>  
  Header content...  
</header>
```

not only shorter than  
`<div id="header">`, but  
semantic too

*depending on its location  
in the document, could also  
provide a landmark  
for navigation*

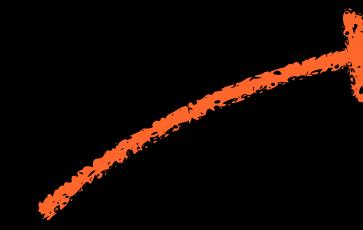
# Por exemplo

```
<input id="n" name="n"  
      required aria-required="true"  
      autocorrect="off"  
      autocapitalize="words"  
      placeholder="Sir Tim Berners Lee"  
      autocomplete="name"  
>
```

# Por exemplo

```
<input id="n" name="n"  
      required aria-required="true"  
      autocorrect="off"  
      autocapitalize="words"  
      placeholder="Sir Tim Berners Lee"  
      autocomplete="name"  
      >
```

modern browsers require  
users to enter content



# Por exemplo

modern browsers require users to enter content

```
<input id="n" name="n"  
       required aria-required="true"  
       autocorrect="off"  
       autocapitalize="words"  
       placeholder="Sir Tim Berners Lee"  
       autocomplete="name"  
     >
```

browser can inform assistive tech that the field is required

# Por exemplo

```
<input id="n" name="n"  
       required aria-required="true"  
       autocorrect="off" ←  
       autocapitalize="words"  
       placeholder="Sir Tim Berners Lee"  
       autocomplete="name"  
>
```

modern browsers require  
users to enter content

browser can inform  
assistive tech that  
the field is required

don't let the browser  
try to correct  
someone's name

# Por exemplo

```
<input id="n" name="n"  
       required aria-required="true"  
       autocorrect="off" ←  
       autocapitalize="words"  
       placeholder="Sir Tim Berners Lee"  
       autocomplete="name"  
>
```

modern browsers require  
users to enter content

browser can inform  
assistive tech that  
the field is required

don't let the browser  
try to correct  
someone's name

auto-disappearing suggestion  
without JavaScript

# Por exemplo

```
<input id="n" name="n"  
       required aria-required="true"  
       autocorrect="off" ←  
       autocapitalize="words"  
       placeholder="Sir Tim Berners Lee"  
       autocomplete="name"  
>
```

modern browsers require  
users to enter content

if the browser already  
knows the user's name,  
by all means, let it fill it in

auto-disappearing suggestion  
without JavaScript

browser can inform  
assistive tech that  
the field is required

don't let the browser  
try to correct  
someone's name

# Por exemplo

```
<input type="email"  
       id="e" name="e"  
       required aria-required="true"  
       autocorrect="off"  
       autocapitalize="off"  
       autocomplete="email"  
       placeholder="you@yourdomain.tld"  
>
```

# Por exemplo

```
<input type="email" ←  
  id="e" name="e"  
  required aria-required="true"  
  autocorrect="off"  
  autocapitalize="off"  
  autocomplete="email"  
  placeholder="you@yourdomain.tld"  
>
```

modern browsers validate  
the email address  
and may supply a  
custom keyboard layout

# Por exemplo

```
<input type="email" →  
  id="e" name="e"  
  required aria-required="true"  
  autocorrect="off"  
  autocapitalize="off"  
  autocomplete="email"  
  placeholder="you@yourdomain.tld"  
>
```

modern browsers validate  
the email address  
and may supply a  
custom keyboard layout

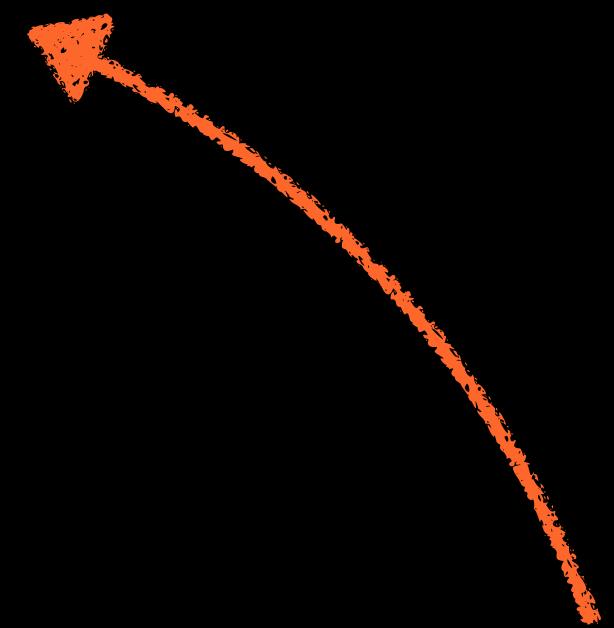
let the browser suggest  
an email address if it has one

# Por exemplo

```
@media (min-width:600px) {  
  .gridded {  
    display: grid;  
    grid-template-columns: 1fr 300px;  
    grid-gap: 20px;  
  }  
}
```

# Por exemplo

```
@media (min-width:600px) {  
  .gridded {  
    display: grid;  
    grid-template-columns: 1fr 300px;  
    grid-gap: 20px;  
  }  
}
```



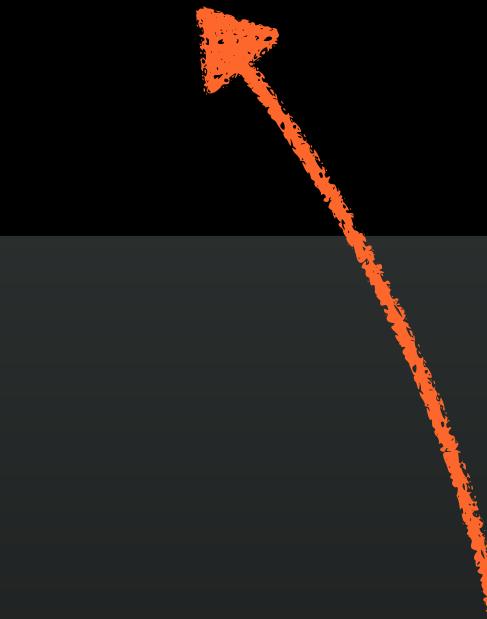
So much better than floats

# Por exemplo

```
var $radios = document.querySelectorAll(  
    'input[type=radio]'  
);
```

# Por exemplo

```
var $radios = document.querySelectorAll(  
    'input[type=radio]'  
);
```



CSS selector-based  
DOM traversal  
without a JS library

# Por exemplo

```
font-family: Georgia, Times,  
"Times New Roman", serif
```

Serif

```
font-family: "Segoe UI", Frutiger,  
"Frutiger Linotype",  
"Dejavu Sans", "Helvetica Neue",  
Arial, sans-serif;
```

Sans Serif

**This is a Title**

Georgia

**This is a Title**

Times

**This is a Title**

Times New Roman

**This is a Title**

System Serif

**This is body text.**

Segoe UI

**This is body text.**

Frutiger

**This is body text.**

Frutiger Linotype

**This is body text.**

DejaVu Sans

**This is body text.**

Helvetica Neue

**This is body text.**

Arial

**This is body text.**

System Serif

If you need  
a custom font:  
**subset, subset, subset**

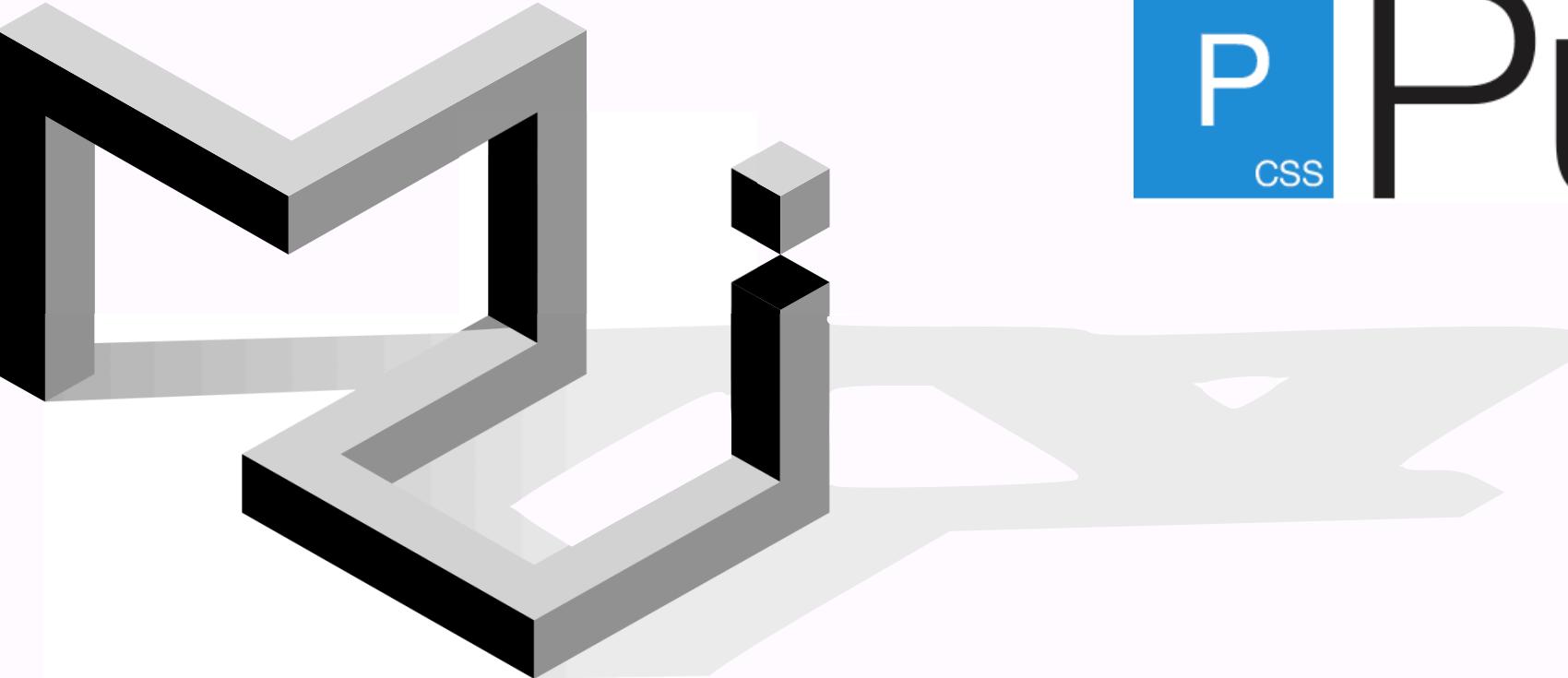
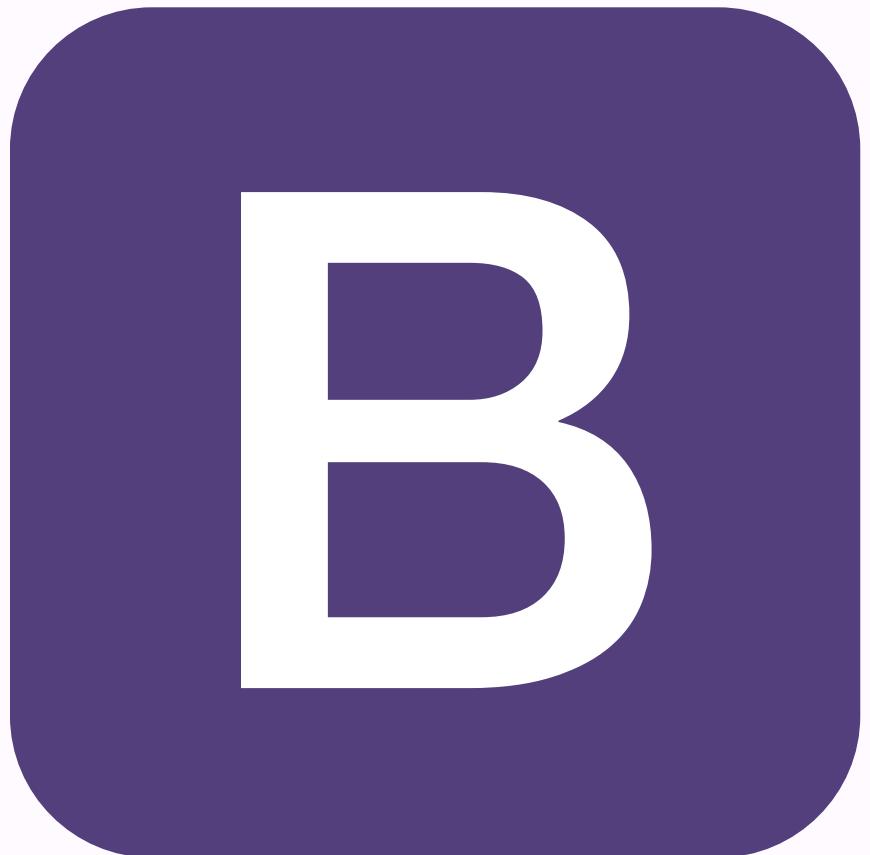
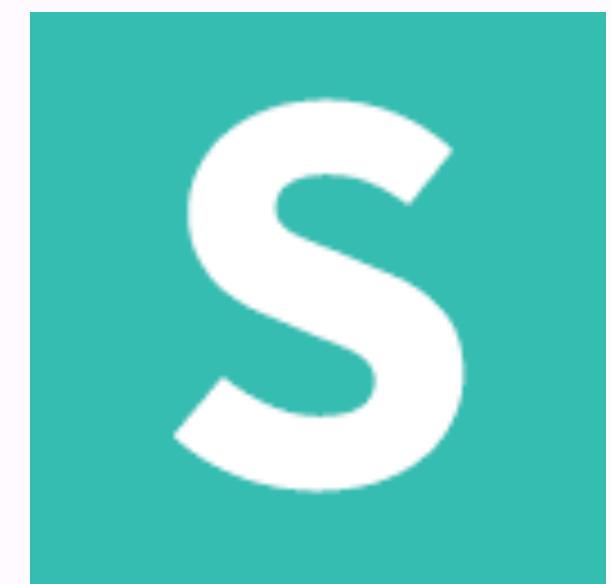
# Proceed with caution



Source: [Bran Stein](#)

Step 2:  
Only include assets  
**you actually need**





Great tools,  
possibly overkill

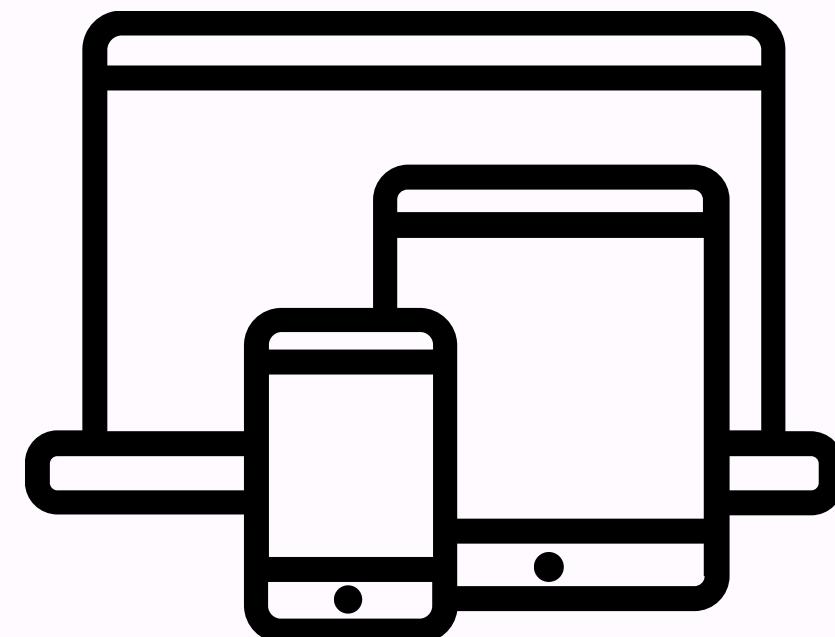


# Every tool has a cost

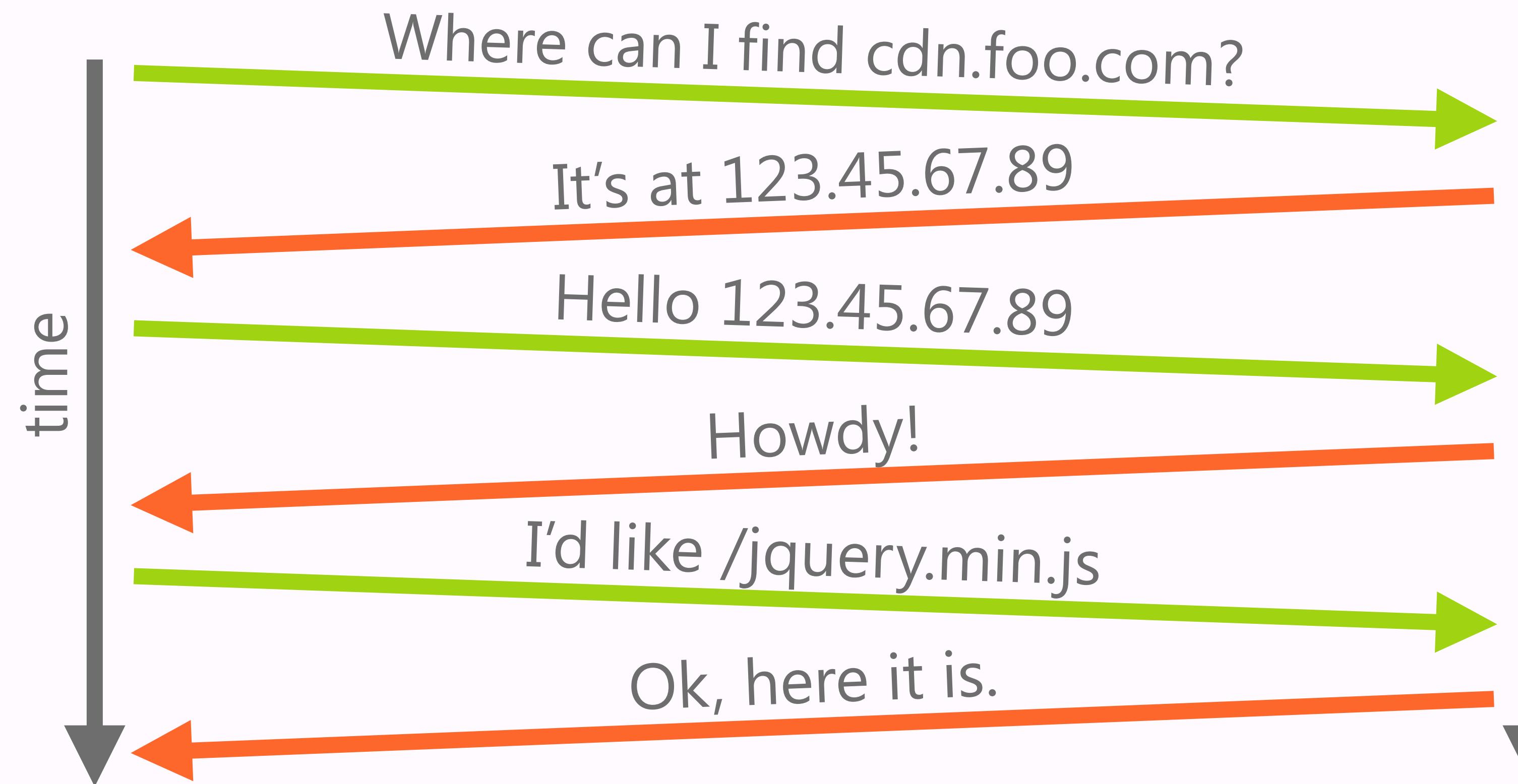
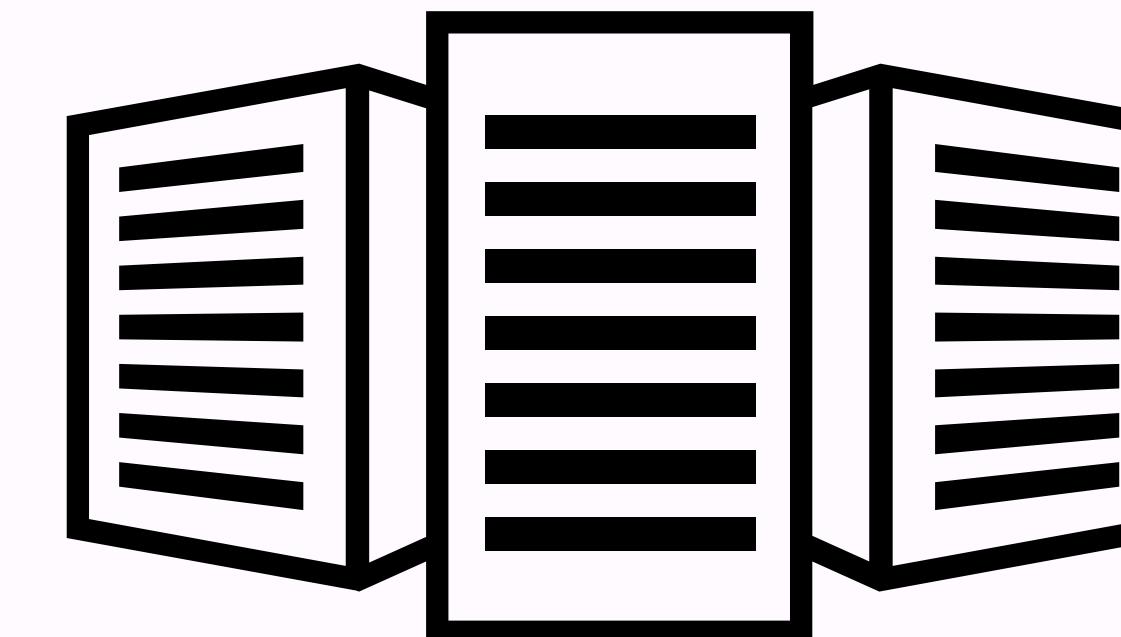
Framework	Size (Compressed)
Bootstrap 2	149 kB
Bootstrap 3	103 kB
Angular 1.4	51 kB
Ember 2.2.0	111 kB
Foundation 4	266 kB
jQuery	32 kB
UI Kit	86 kB
React 16 + React DOM	35 kB
Vue 2.4.2	20 kB

Chances are, your  
library of choice  
is on a CDN

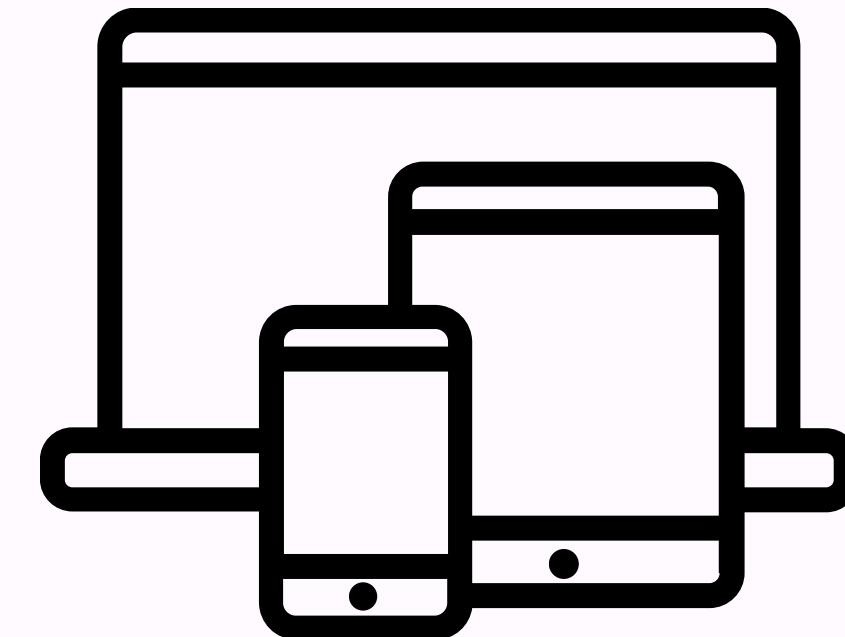
# Your Device



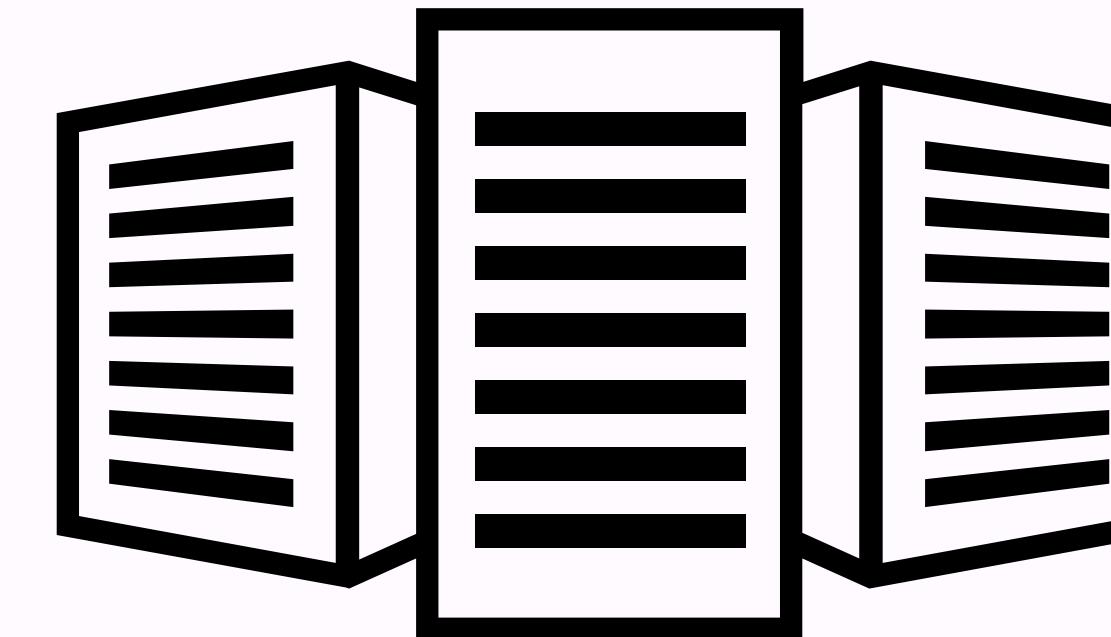
# The Web



# Your Device

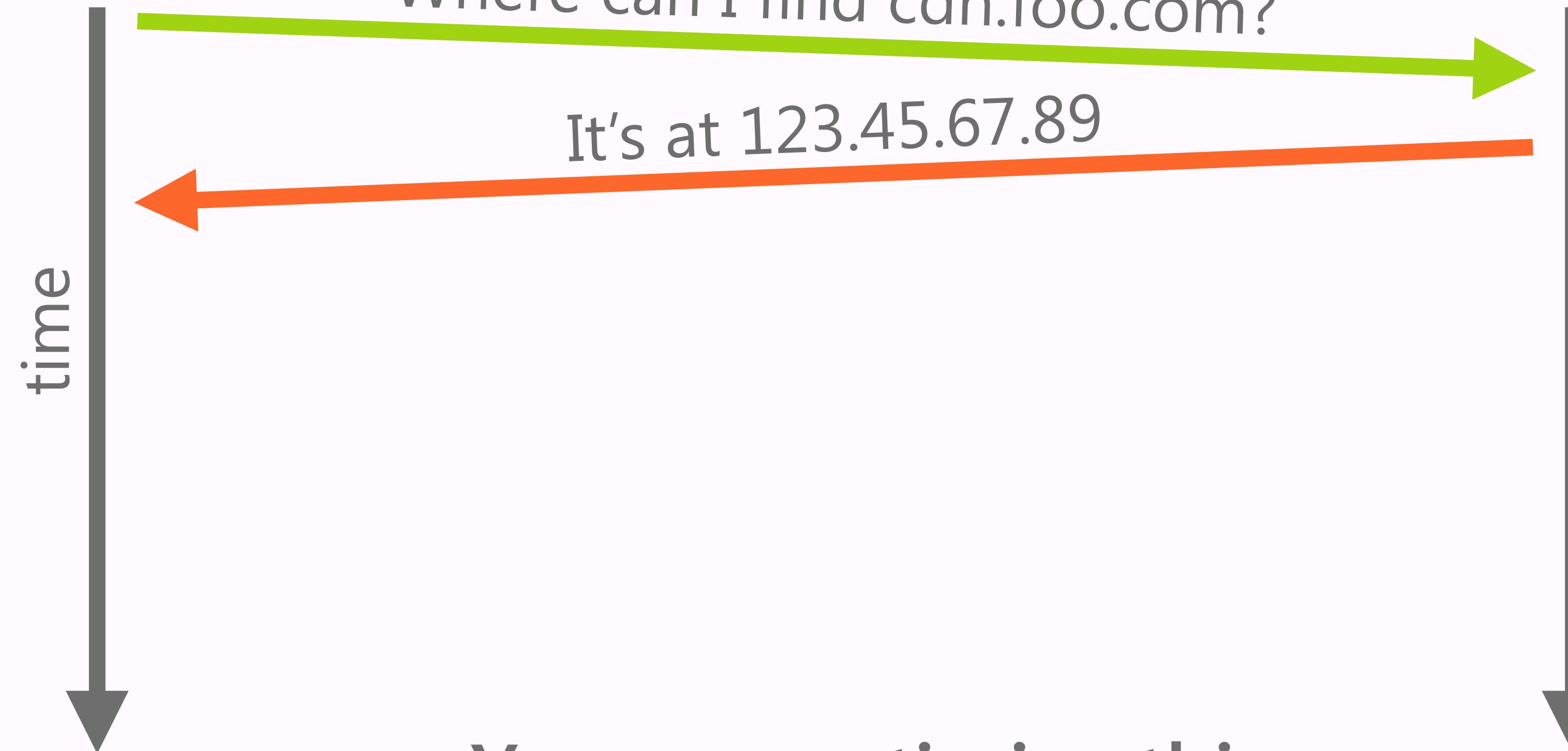


# The Web



Where can I find cdn.foo.com?

It's at 123.45.67.89



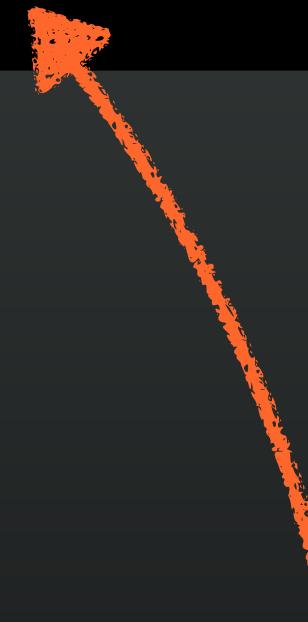
You can optimize this

# Find the server early

```
<link rel="dns-prefetch"  
      href="https://cdn.foo.com">
```

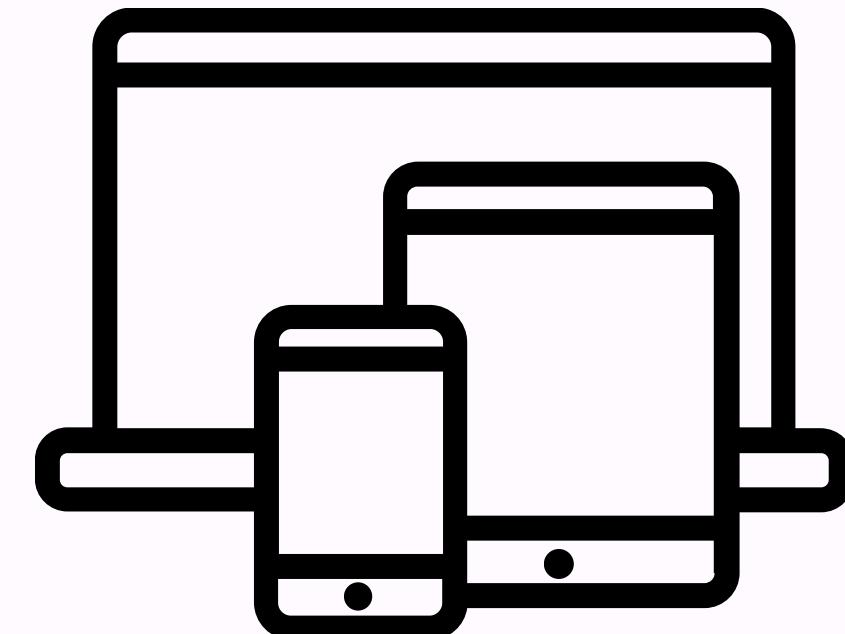
# Find the server early

```
<link rel="dns-prefetch"  
      href="https://cdn.foo.com">
```

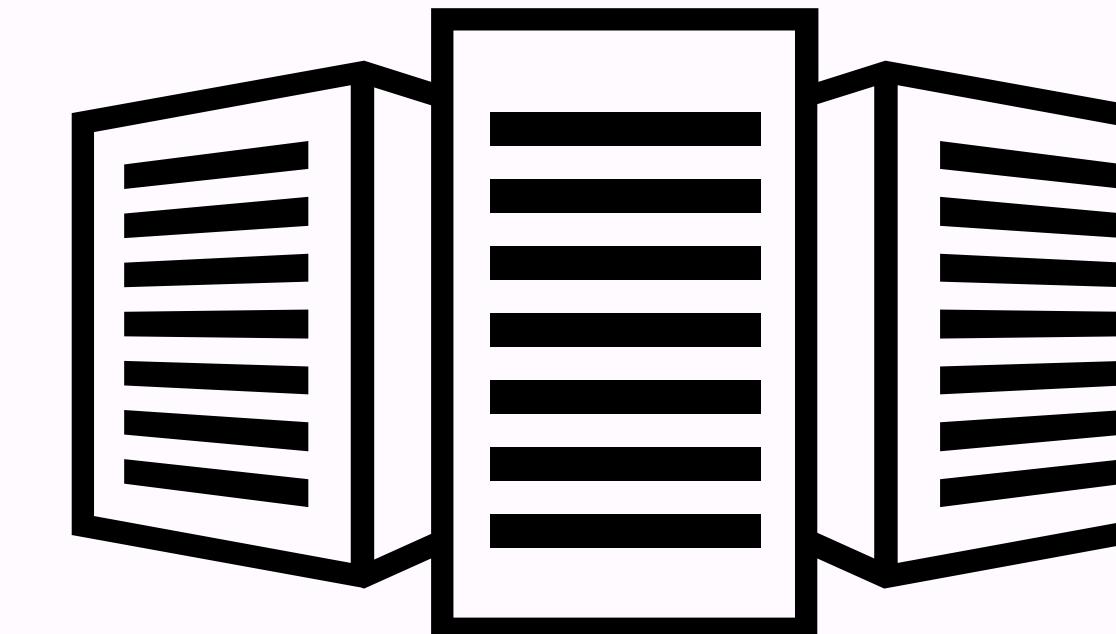


Drop this in the  
head of your pages

# Your Device



# The Web

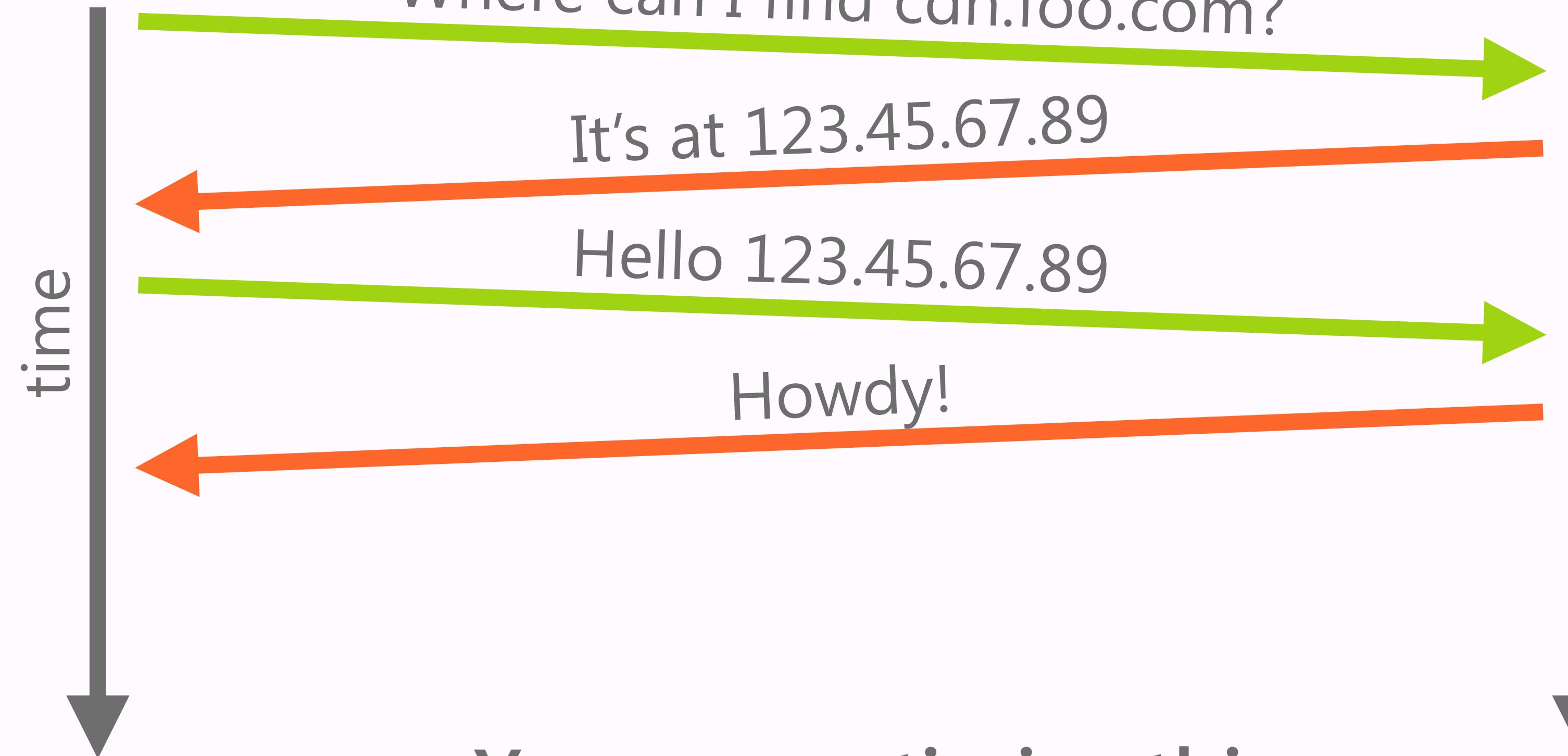


Where can I find cdn.foo.com?

It's at 123.45.67.89

Hello 123.45.67.89

Howdy!

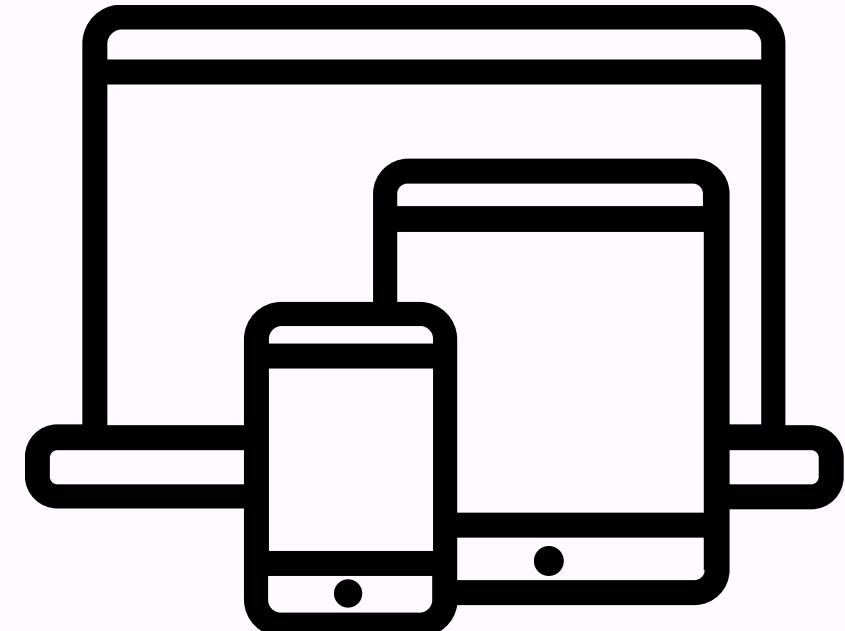


You can optimize this

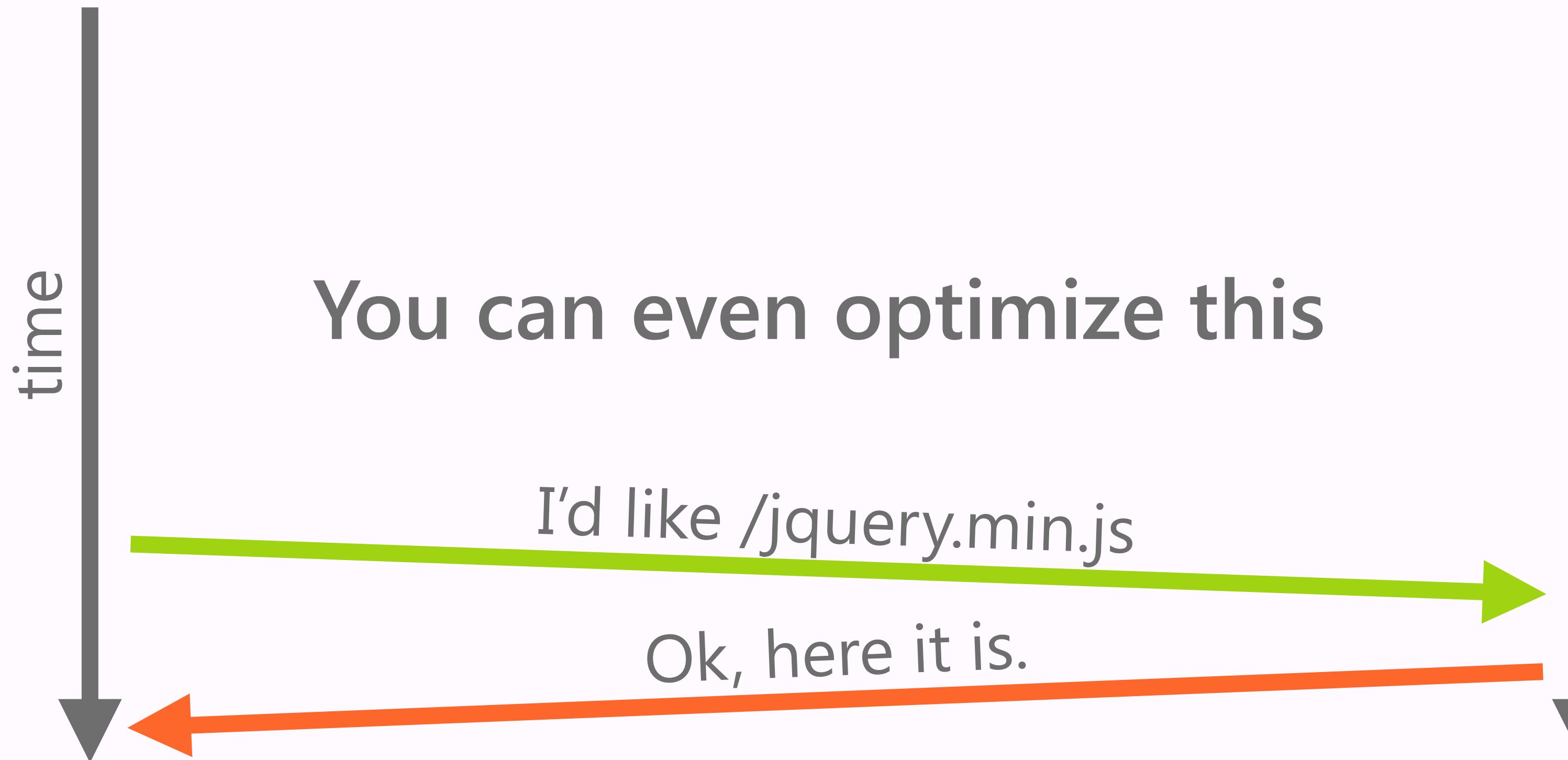
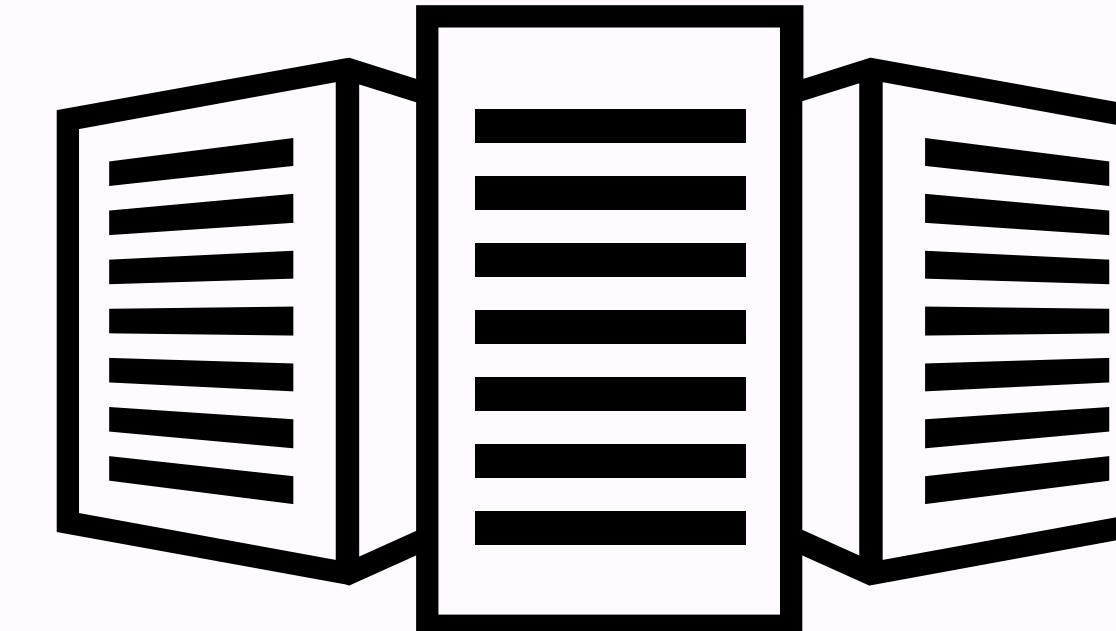
# Go for the handshake

```
<link rel="preconnect"  
      href="https://cdn.foo.com">
```

# Your Device



# The Web

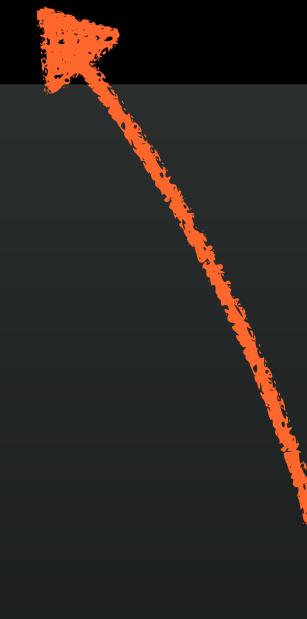


# Grab that resource

```
<link rel="preload"  
      href="https://cdn.foo.com/jquery.min.js"  
      as="script">
```

# Grab that resource

```
<link rel="preload"  
      href="https://cdn.foo.com/jquery.min.js"  
      as="script">
```

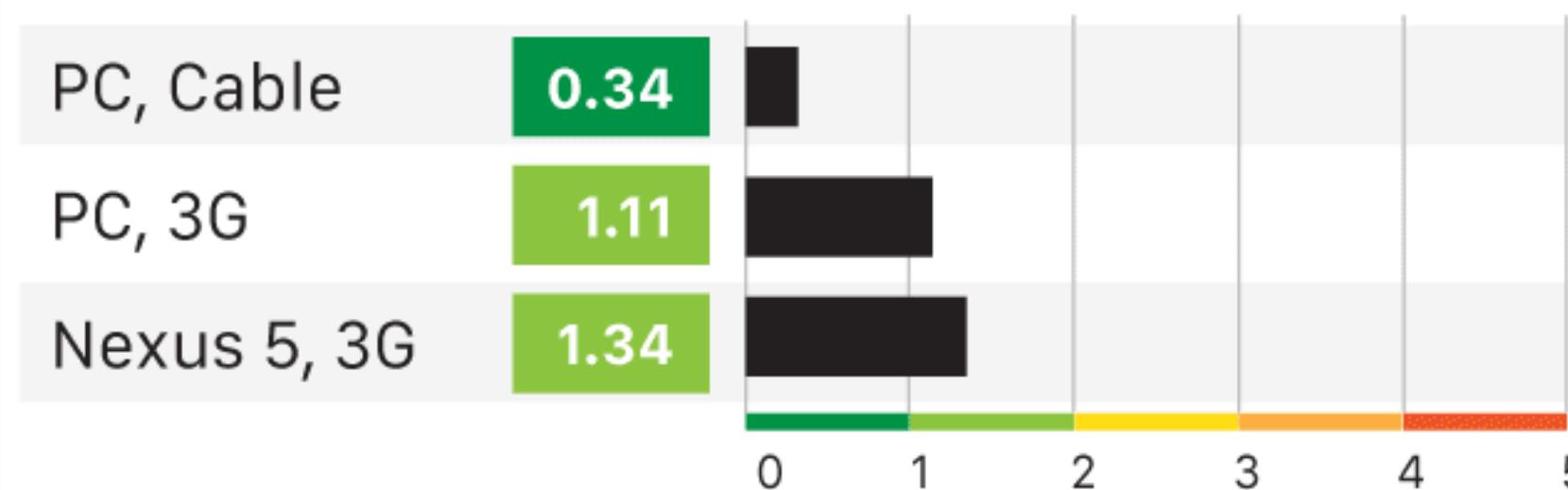


Helps optimize the process

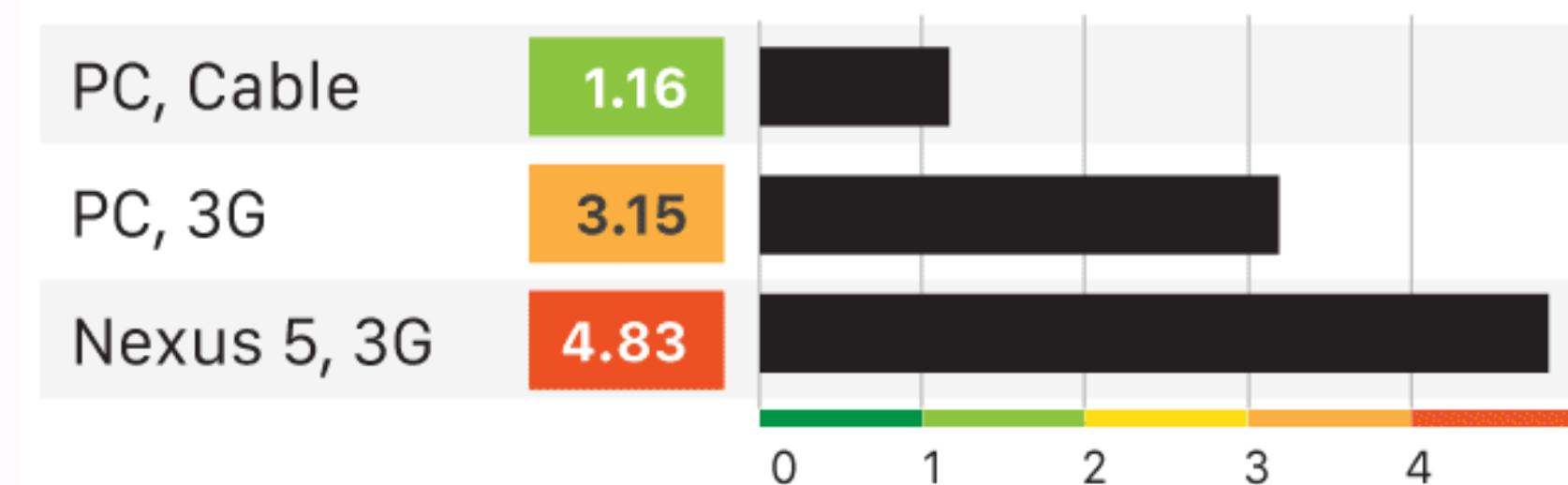
# Download isn't everything

**Ampersand**

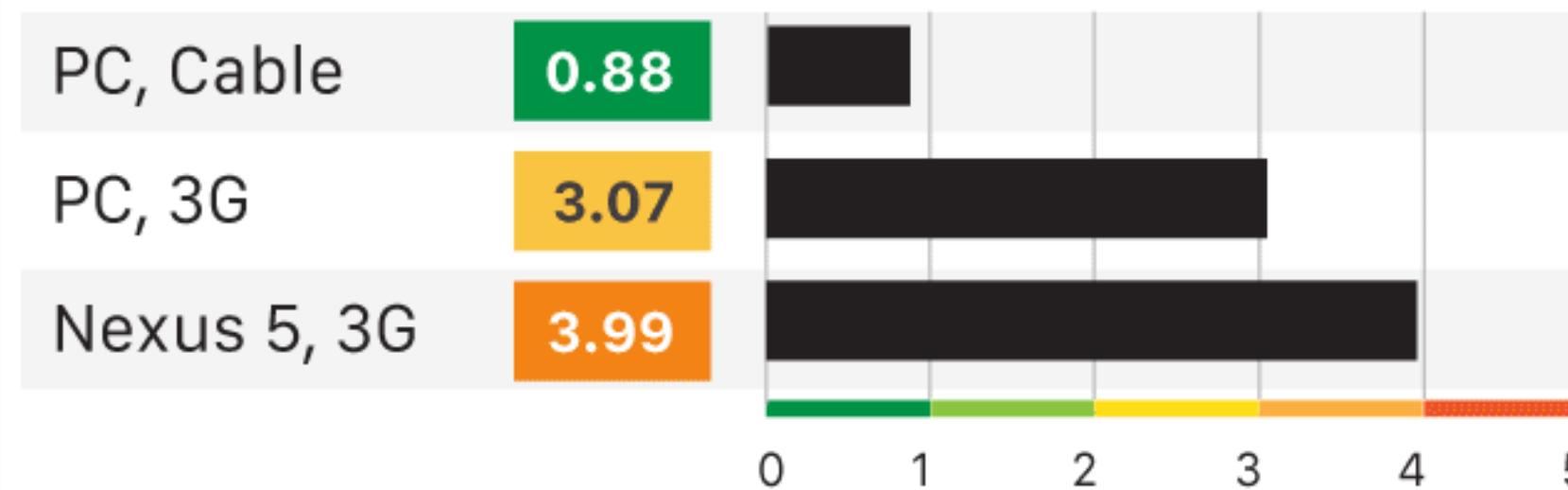
32k script size

**Ember 1.9.0**

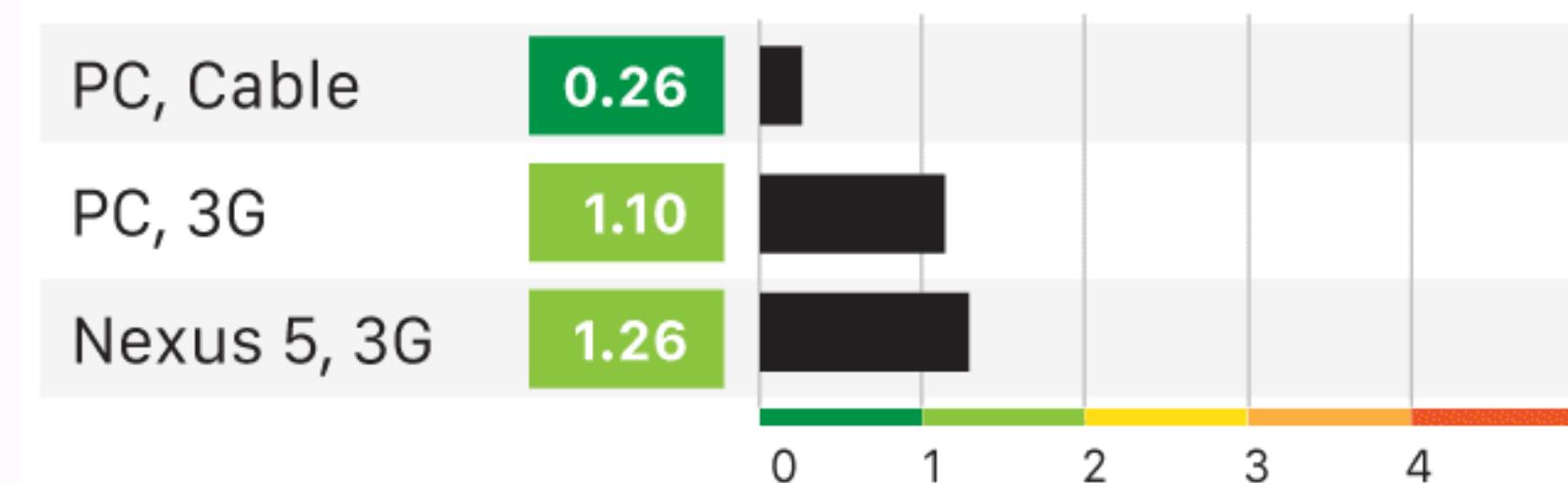
177k script size

**Angular 1.3.6**

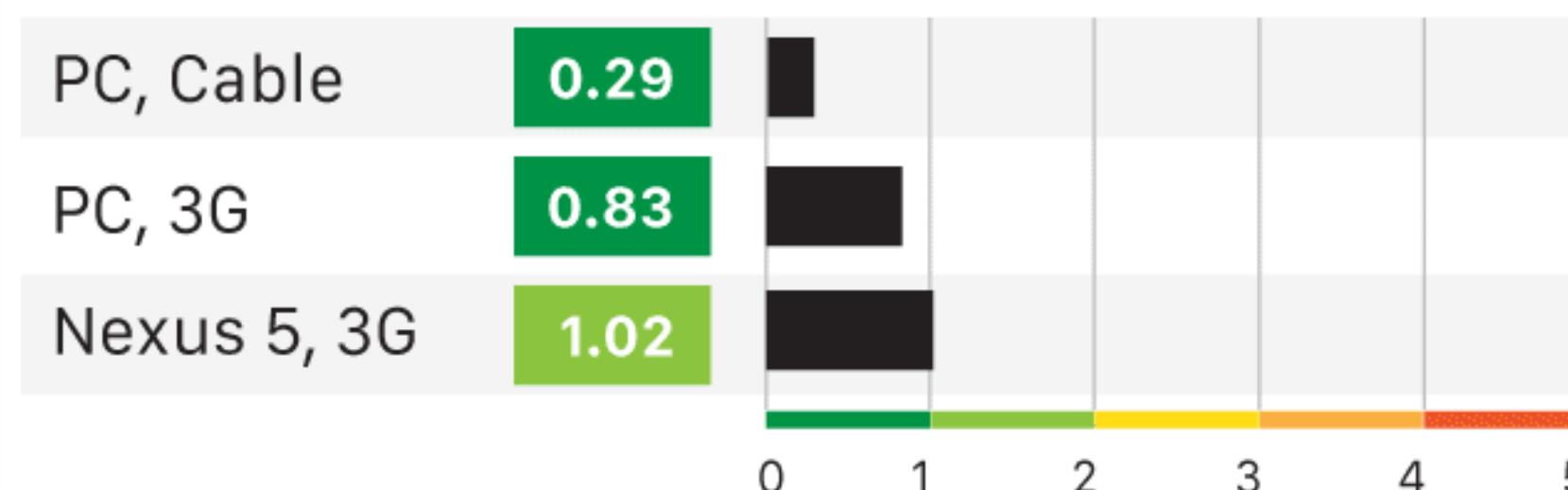
51k script size

**React 0.12.2**

53k script size

**Backbone 1.1.2**

43k script size



# Download isn't everything

Framework	Method/function	operations/s
Vanilla JS	<code>document.getElementById()</code>	12,137,211
Dojo	<code>dojo.byId();</code>	5,443,343
Prototype	<code>\$()</code>	2,940,734
Ext JS	<code>Ext.get()</code>	997,562
jQuery	<code>\$()</code>	350,557
YUI	<code>YAHOO.util.Dom.get()</code>	326,534
MooTools	<code>document.id()</code>	78,802



10k APART

Inspiring the Web with Just 10k

[Read the Rules](#)

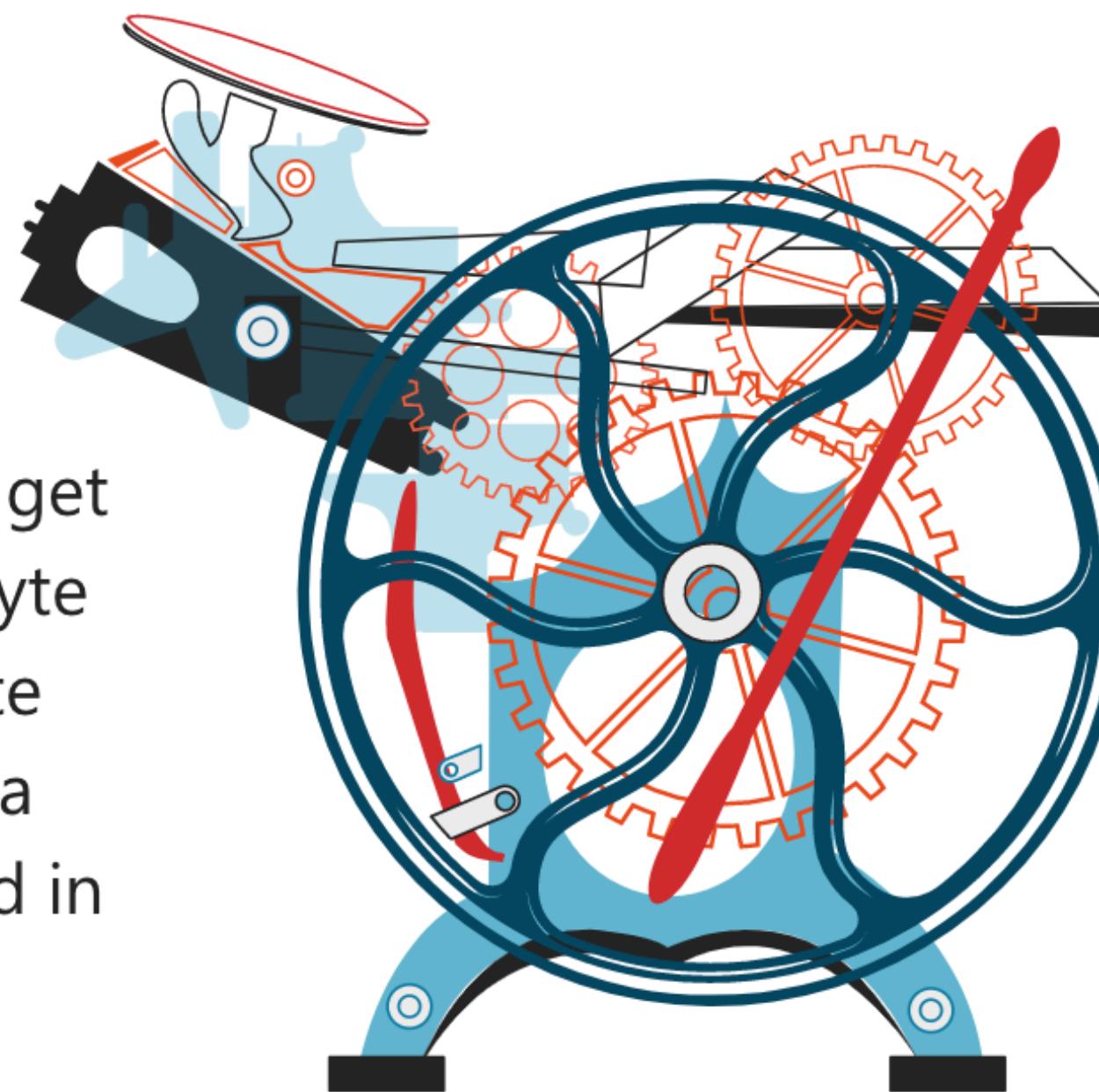
[View the Gallery](#)

[Meet the Judges](#)

*Brought to you by Microsoft and An Event Apart*

# Inspiring the Web with Just 10k

With so much of an emphasis on front-end frameworks and JavaScript runtimes, it's time to get back to basics—back to optimizing every little byte like your life depends on it and ensuring your site can work, no matter what. The Challenge? Build a compelling web experience that can be delivered in 10kB and works without JavaScript.



[View the Winners](#)

[Check Out the Gallery](#)

---

And the winners are...

**Best Overall:**

**Best Design:**



10k APART

Inspiring the Web with Just 10k

Read the Rules

View the Gallery

Meet the Judges

*Brought to you by Microsoft and An Event Apart*

# Inspiring the Web with Just 10k

With so much of an emphasis on front-end frameworks and JavaScript runtimes, it's time to get back to basics—back to the basics of web design, like you did in college.

100% library free



[View the Winners](#)

[Check Out the Gallery](#)

---

And the winners are...

**Best Overall:**

**Best Design:**

# We used some hints though

```
<link rel="preload"
      href="https://10kapart.blob.core.windows.net">
<link rel="preload"
      href="https://cdnjs.cloudflare.com">
<link rel="preload"
      href="https://www.google-analytics.com">
```

Step 3:  
**Optimize**  
everything

# Our approach to CSS (Gulp)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)
5. Remove any unused declarations/rule sets (gulp-uncss)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)
5. Remove any unused declarations/rule sets (gulp-uncss)
6. Optimize the files (gulp-css)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)
5. Remove any unused declarations/rule sets (gulp-uncss)
6. Optimize the files (gulp-css)
7. Minify (gulp-clean-css)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)
5. Remove any unused declarations/rule sets (gulp-uncss)
6. Optimize the files (gulp-css)
7. Minify (gulp-clean-css)
8. Gzip (gulp-zopfli)

# Our approach to CSS (Gulp)

1. Write modular CSS in Sass (+ Breakup for MQ management)
2. Compile CSS with a precision of 4 decimal places (gulp-sass)
3. Fallbacks for the last 2 browser versions (gulp-autoprefixer)
4. CSS shorthand declarations if possible (gulp-shorthand)
5. Remove any unused declarations/rule sets (gulp-uncss)
6. Optimize the files (gulp-css)
7. Minify (gulp-clean-css)
8. Gzip (gulp-zopfli)
9. Brotli (gulp-brotli)

# Before

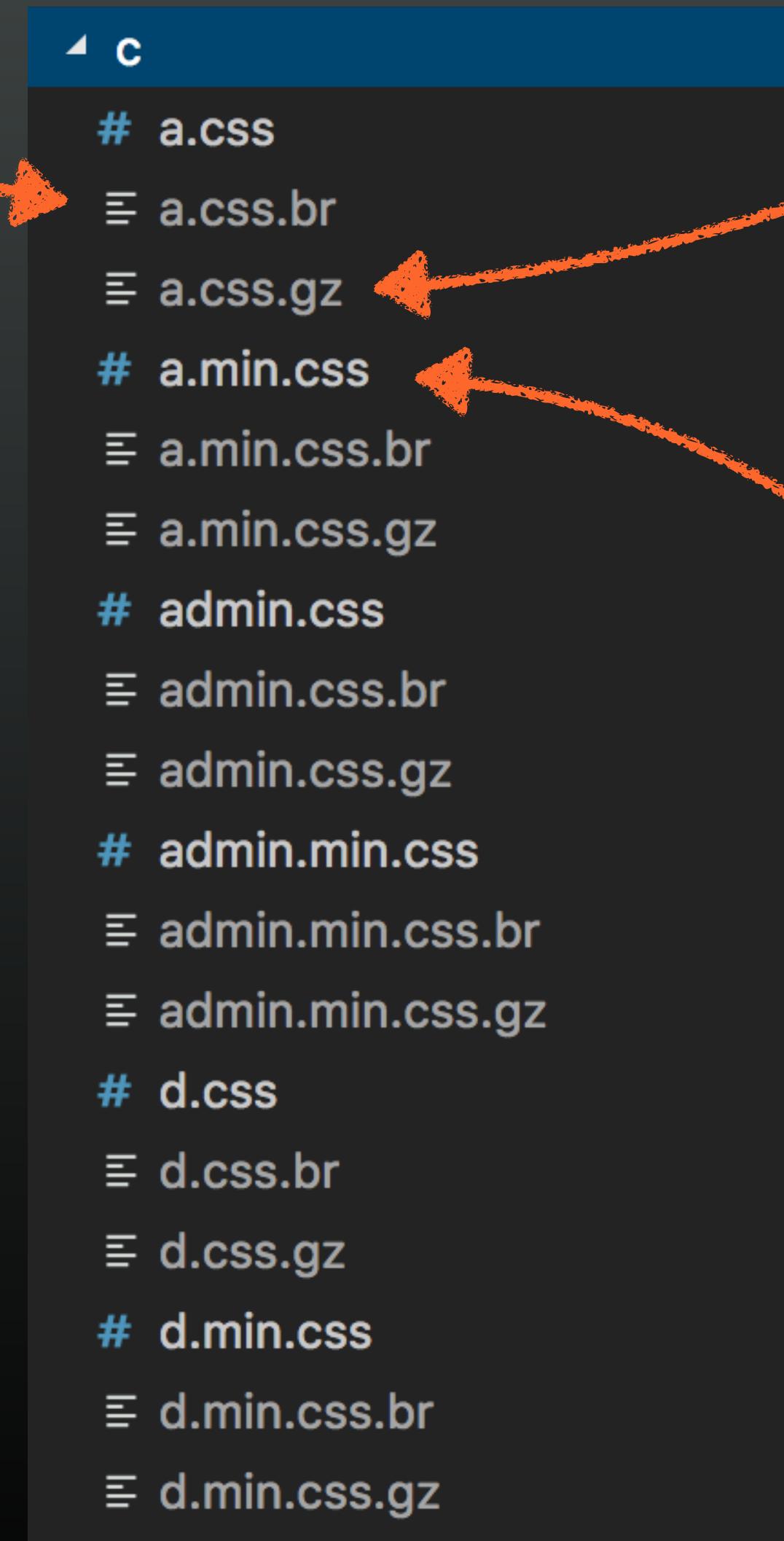
```
▲ scss
  ▲ base
    ↗ _base.scss
    ↗ _blocks.scss
    ↗ _body.scss
    ↗ _forms.scss
    ↗ _headings.scss
    ↗ _links.scss
    ↗ _lists.scss
    ↗ _media.scss
    ↗ _text.scss
  ▲ helpers
    ↗ _helpers.scss
    ↗ _config.scss
    ↗ _functions.scss
    ↗ _mixins.scss
    ↗ _placeholders.scss
    ↗ _variables.scss
  ▲ components
    ↗ _components.scss
    ↗ _boxed.scss
    ↗ _cookie-banner.scss
    ↗ _count.scss
    ↗ _cta.scss
    ↗ _faq.scss
    ↗ _footer.scss
    ↗ _gallery.scss
    ↗ _header.scss
    ↗ _hero.scss
    ↗ _judges.scss
    ↗ _main.scss
    ↗ _mq-watcher.scss
    ↗ _nav.scss
    ↗ _pagination.scss
    ↗ _project.scss
    ↗ _target.scss
    ↗ _vote.scss
    ↗ _winners.scss
  ▲ pages
    ↗ _pages.scss
    ↗ _home.scss
  ▲ vendors
    ▶ breakup
      ↗ _vendors.scss
    ↗ a.scss
    ↗ admin.scss
    ↗ d.scss
```

# After

```
◀ c
# a.css
≡ a.css.br
≡ a.css.gz
# a.min.css
≡ a.min.css.br
≡ a.min.css.gz
# admin.css
≡ admin.css.br
≡ admin.css.gz
# admin.min.css
≡ admin.min.css.br
≡ admin.min.css.gz
# d.css
≡ d.css.br
≡ d.css.gz
# d.min.css
≡ d.min.css.br
≡ d.min.css.gz
```

# After

browsers that support  
brotli compression  
2 kB



browsers that support  
gzip compression  
3 kB

everyone else  
8 kB

# Our approach to JS (Gulp)

# Our approach to JS (Gulp)

1. Write modular JavaScript, grouped as appropriate

# Our approach to JS (Gulp)

1. Write modular JavaScript, grouped as appropriate
2. Combine files based on folder structure (gulp-folders, gulp-concat)

# Our approach to JS (Gulp)

1. Write modular JavaScript, grouped as appropriate
2. Combine files based on folder structure (gulp-folders, gulp-concat)
3. Create an wrapping closure to isolate from other JS (gulp-insert)

# Our approach to JS (Gulp)

1. Write modular JavaScript, grouped as appropriate
2. Combine files based on folder structure (gulp-folders, gulp-concat)
3. Create an wrapping closure to isolate from other JS (gulp-insert)
4. Minify (gulp-uglify)

# Our approach to JS (Gulp)

1. Write modular JavaScript, grouped as appropriate
2. Combine files based on folder structure (gulp-folders, gulp-concat)
3. Create an wrapping closure to isolate from other JS (gulp-insert)
4. Minify (gulp-uglify)
5. Gzip (gulp-zopfli)

# Our approach to JS (Gulp)

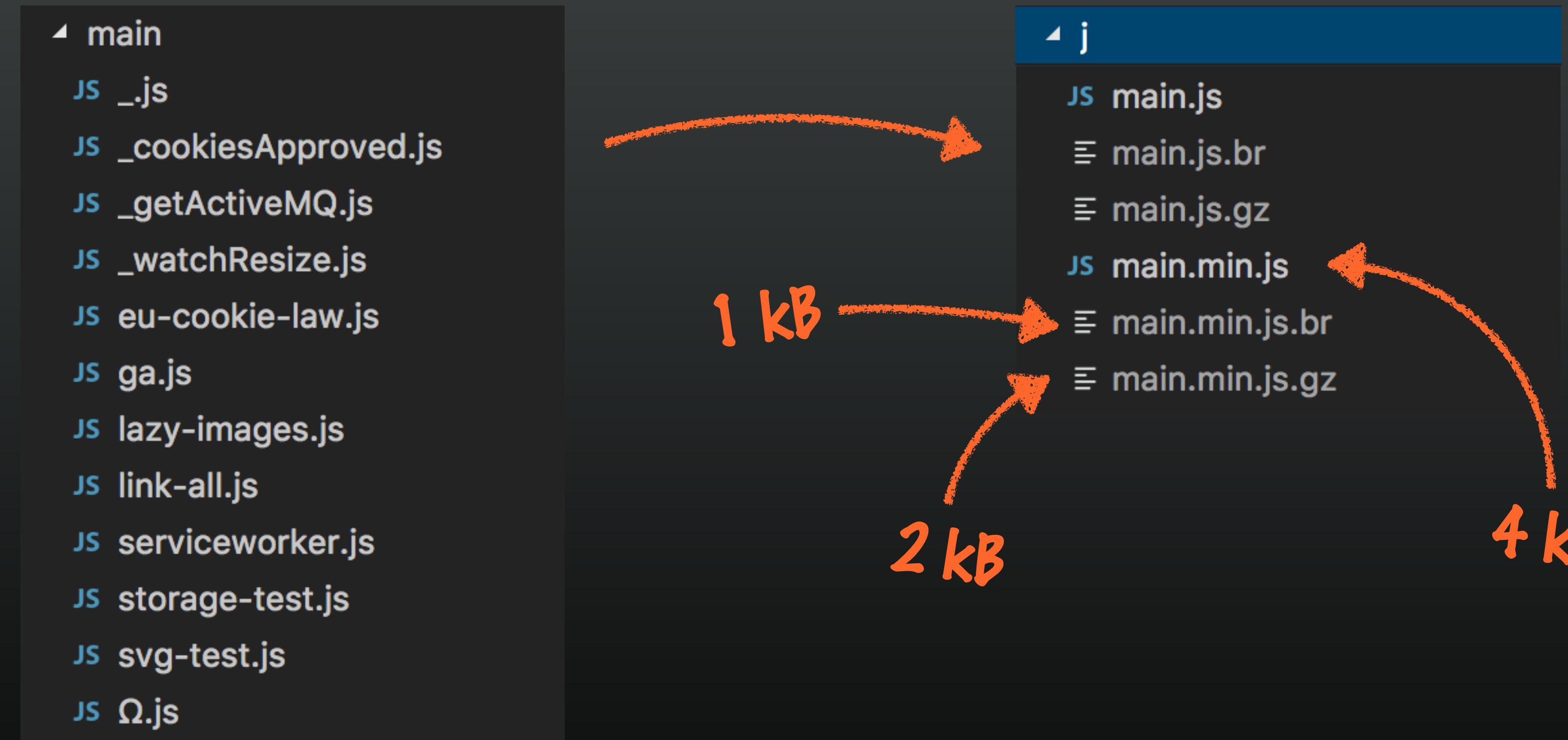
1. Write modular JavaScript, grouped as appropriate
2. Combine files based on folder structure (gulp-folders, gulp-concat)
3. Create an wrapping closure to isolate from other JS (gulp-insert)
4. Minify (gulp-uglify)
5. Gzip (gulp-zopfli)
6. Brotli (gulp-brotli)

# Results

```
▸ main
  JS _.js
  JS _cookiesApproved.js
  JS _getActiveMQ.js
  JS _watchResize.js
  JS eu-cookie-law.js
  JS ga.js
  JS lazy-images.js
  JS link-all.js
  JS serviceworker.js
  JS storage-test.js
  JS svg-test.js
  JS Ω.js
```

about 8 kB all-up

# Results

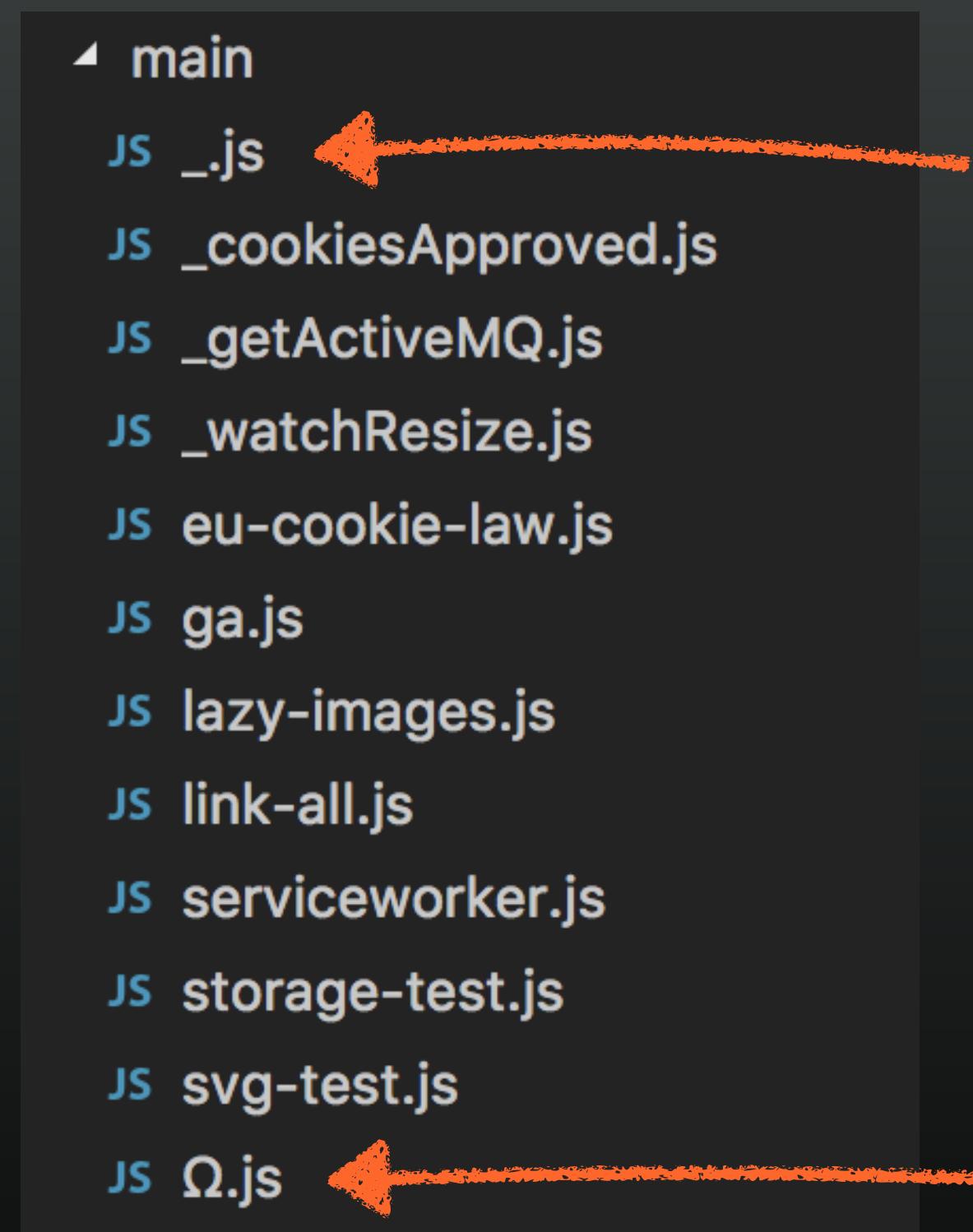


about 8 kB all-up

# Interesting side note

```
▸ main
  JS _.js
  JS _cookiesApproved.js
  JS _getActiveMQ.js
  JS _watchResize.js
  JS eu-cookie-law.js
  JS ga.js
  JS lazy-images.js
  JS link-all.js
  JS serviceworker.js
  JS storage-test.js
  JS svg-test.js
  JS Ω.js
```

# Interesting side note



indexes first

indexes last

We also minified  
& pre-compressed  
our HTML

**Step 4:**  
**Think about when**  
**you load assets**

# We had 10 JS files

- **Global**
  - ▶ main.js - the site's library
  - ▶ serviceworker.js - The site's service worker
- **Browser-specific**
  - ▶ html5shiv.js - local copy of the HTML5 Shiv for < IE9

# We had 10 JS files

- **Page-specific**
  - ▶ enter.js - Entry form-related code
  - ▶ form-saver.js - Used to save form entries locally until submitted
  - ▶ hero.js - Runs the SVG animation on the homepage
  - ▶ home.js - Handles homepage-specific tasks
  - ▶ project.js - Used on project pages during voting
  - ▶ update.js - Handles the winner update form

# Per the common wisdom

```
<script src="/j/main.min.js"></script>
</body>
</html>
```

# No need to run immediately

```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js"
      defer
    ></script>
  </body>
</html>
```

# No need to run immediately

```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js"
    defer
    ></script>
</body>
</html>
```

run after the DOM is loaded

# Run whenever you can

```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js"
      async
      ></script>
    </body>
</html>
```

# Run whenever you can

```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js"
      async
    ></script>
  </body>
</html>
```

run whenever it becomes  
available, but don't  
delay page load

# Consider dependencies

```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js" async></script>
```

# Consider dependencies

```
<script src="/j/main.min.js" async></script>
<script src="/j/home.min.js" async></script>
```

# Consider dependencies

```
<script src="/j/main.min.js" async></script>
<script src="/j/home.min.js" async></script>
```



what if this calls a function  
in main.min.js?

“race condition”

# Avoid race conditions

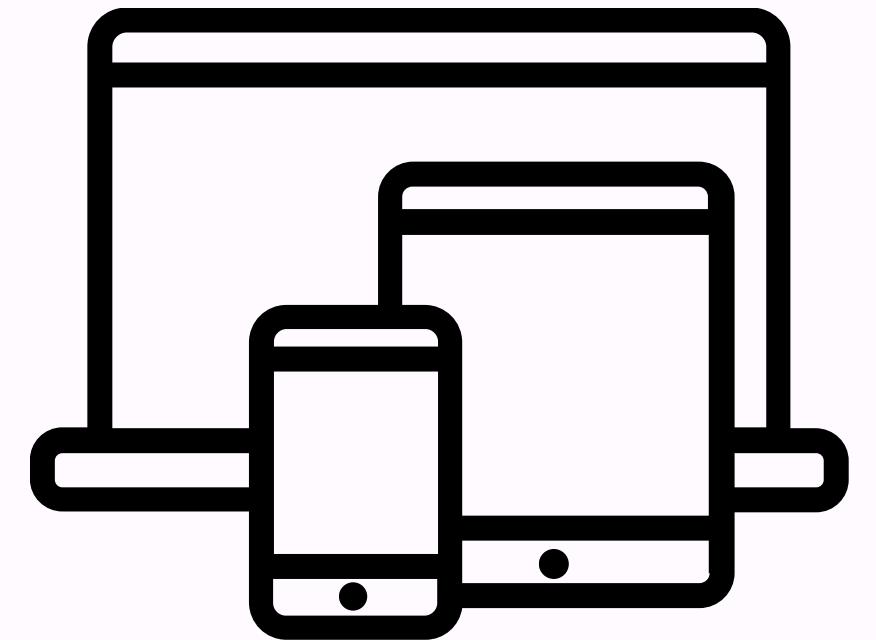
```
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js" async></script>
```

Why so many  
separate files?

# Connections in HTTP/1.1

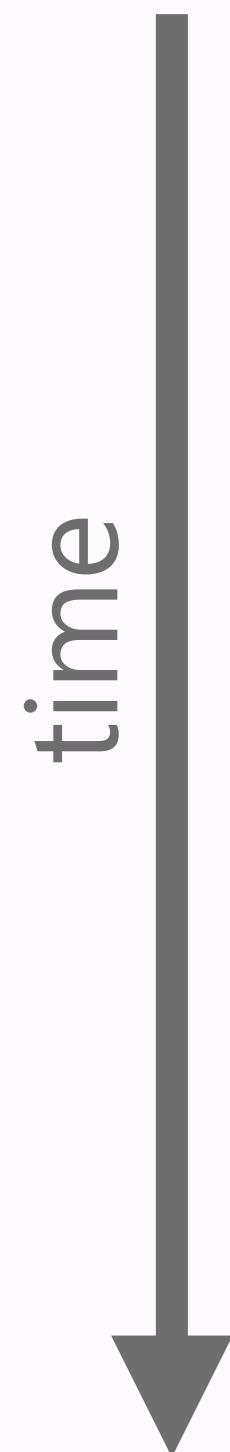
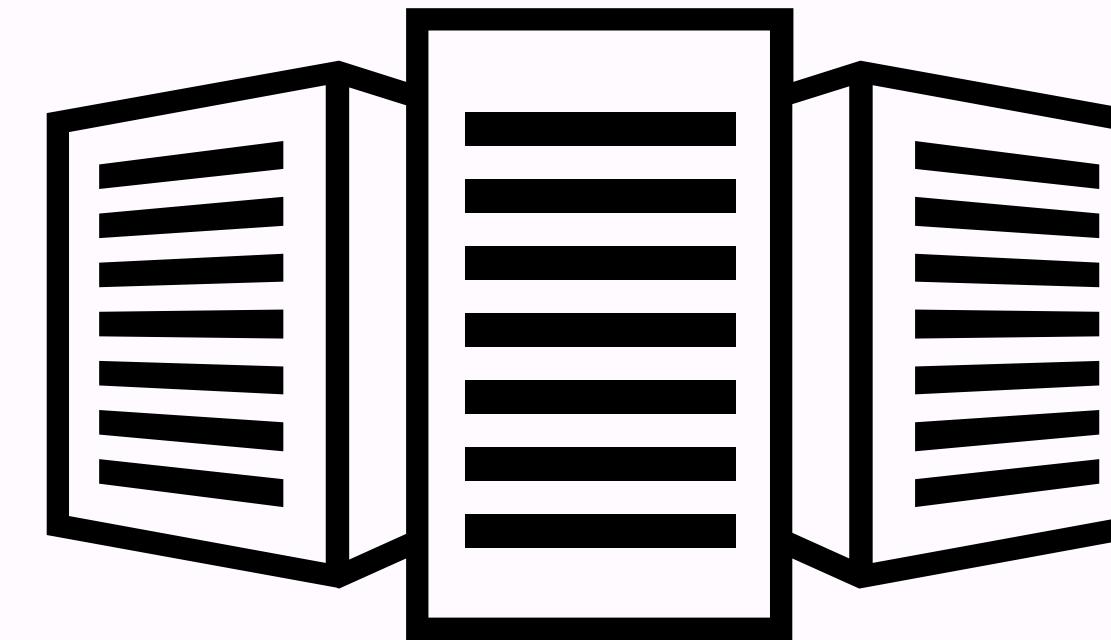
Browser	Per host	Overall
IE 9	6	35
IE 10	8	17
IE 11	13	17
Firefox 4+	6	17
Opera 11+	6	user defined
Chrome 4+	6	10
Safari 7+	6	17

# Your Device

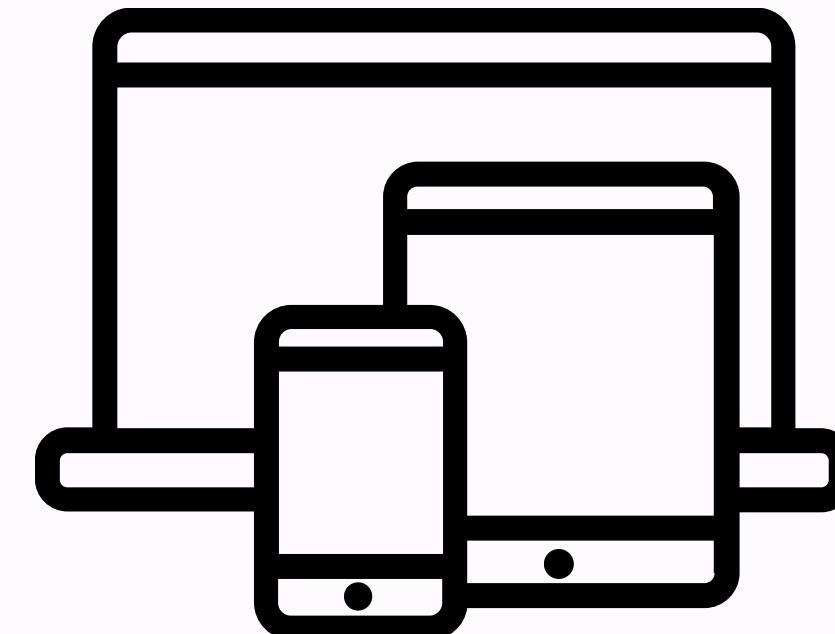


# HTTP/1.1

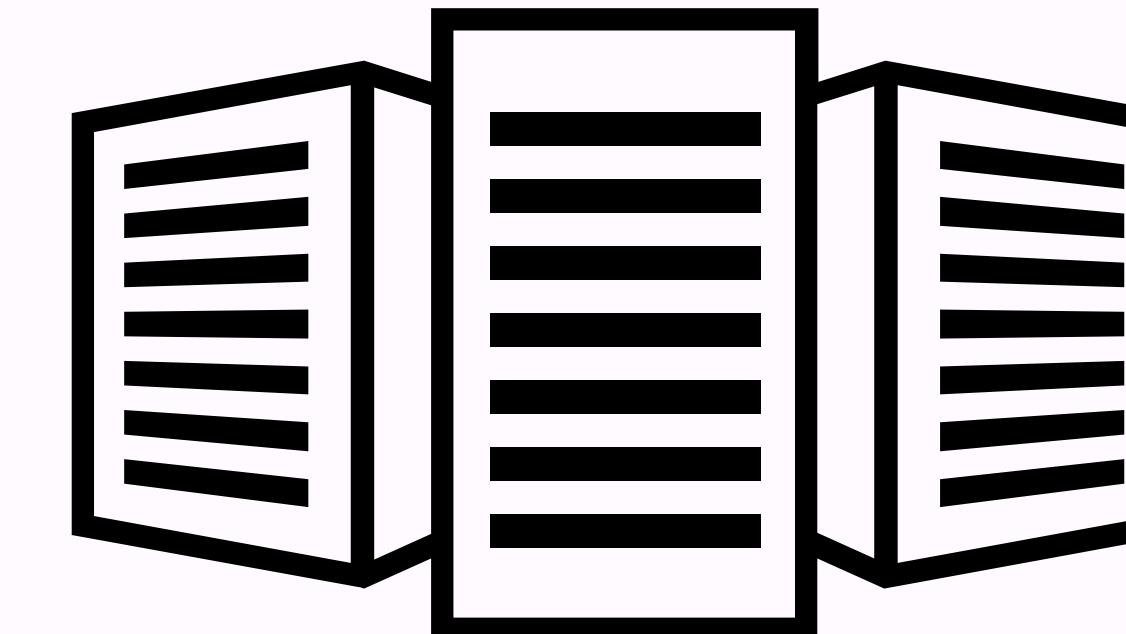
# The Web



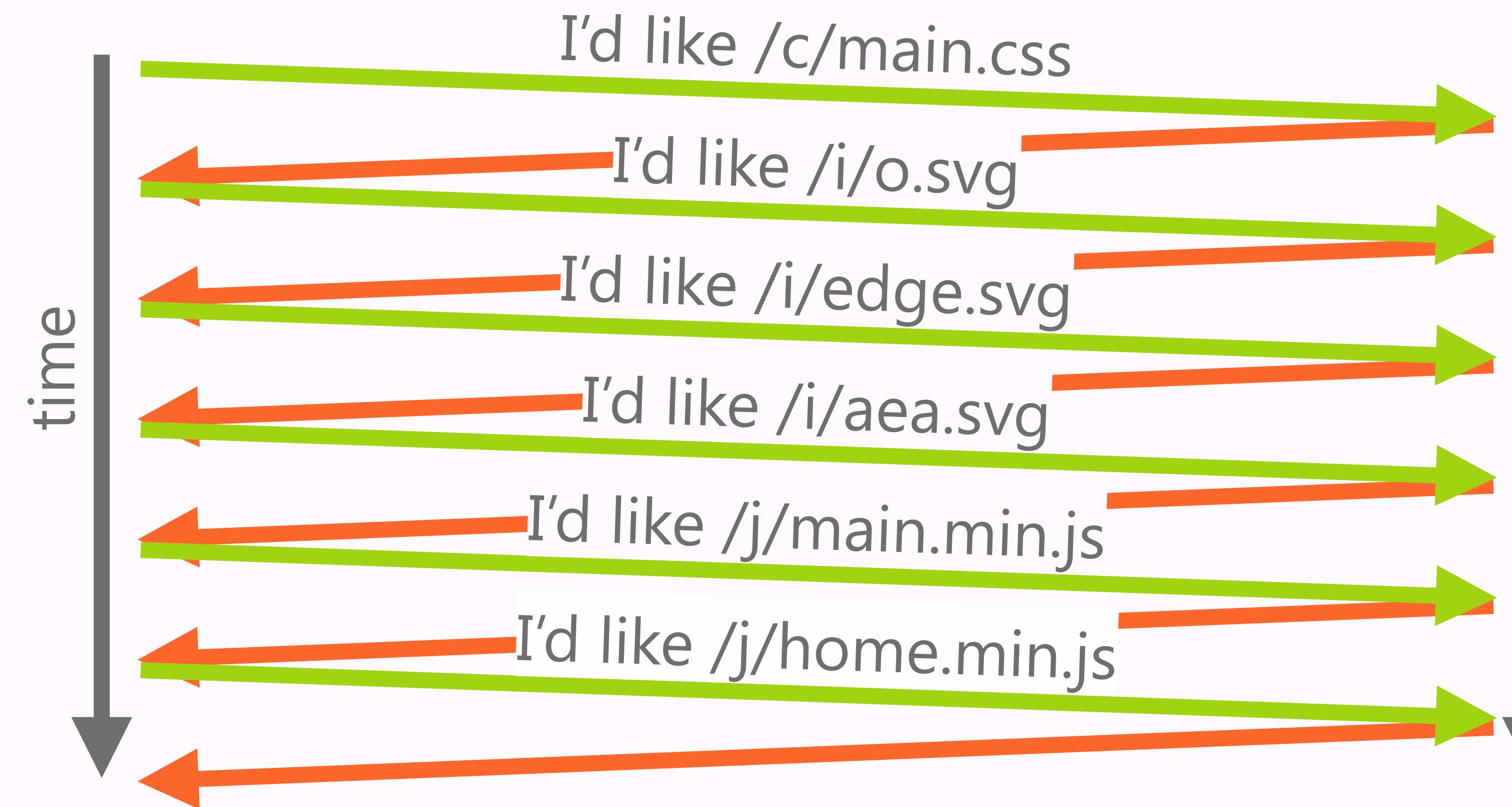
# Your Device



# The Web

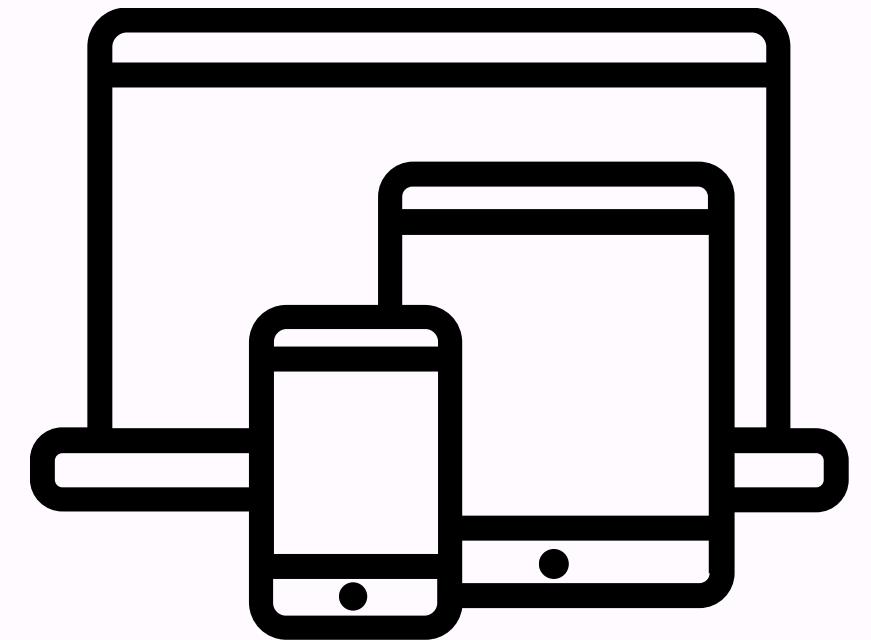


HTTP/1.1



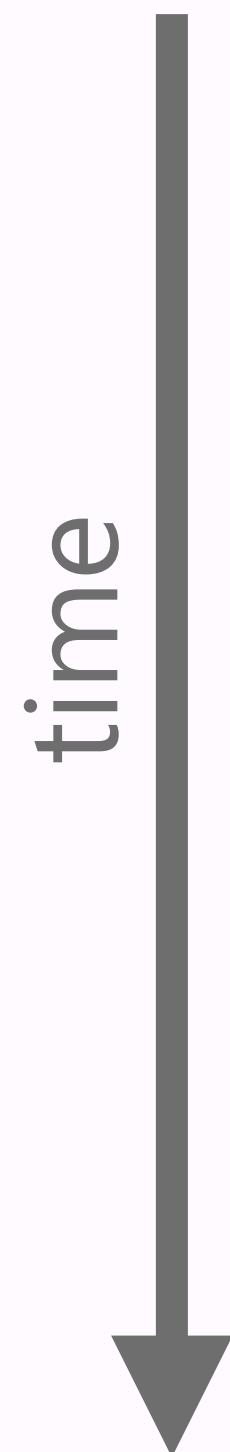
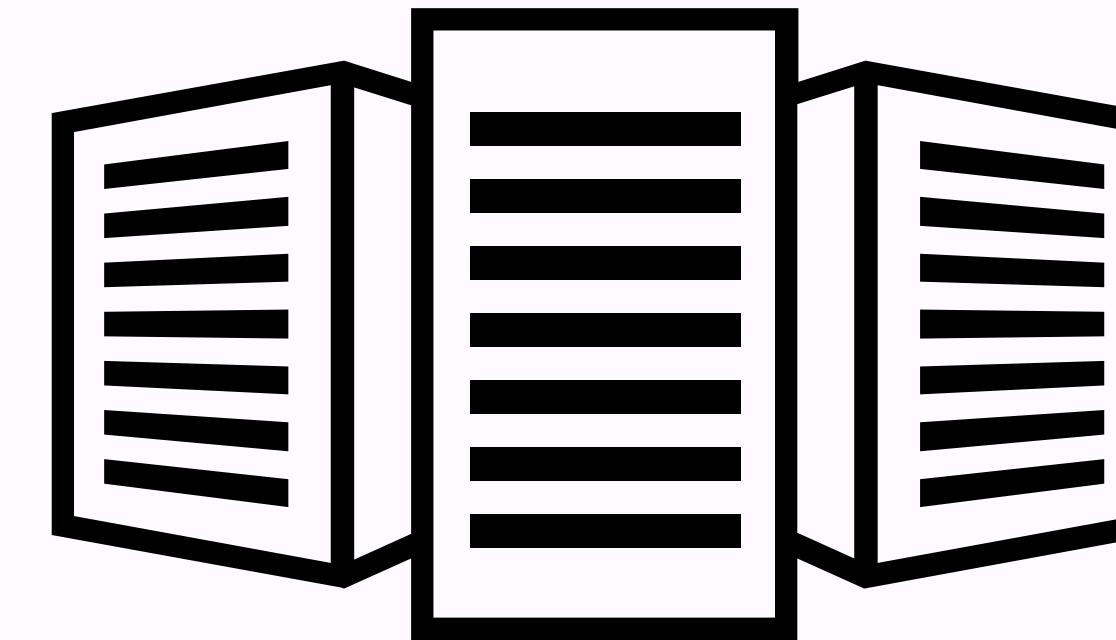
**HTTP/2 creates  
a single connection and  
contents stream in**

# Your Device

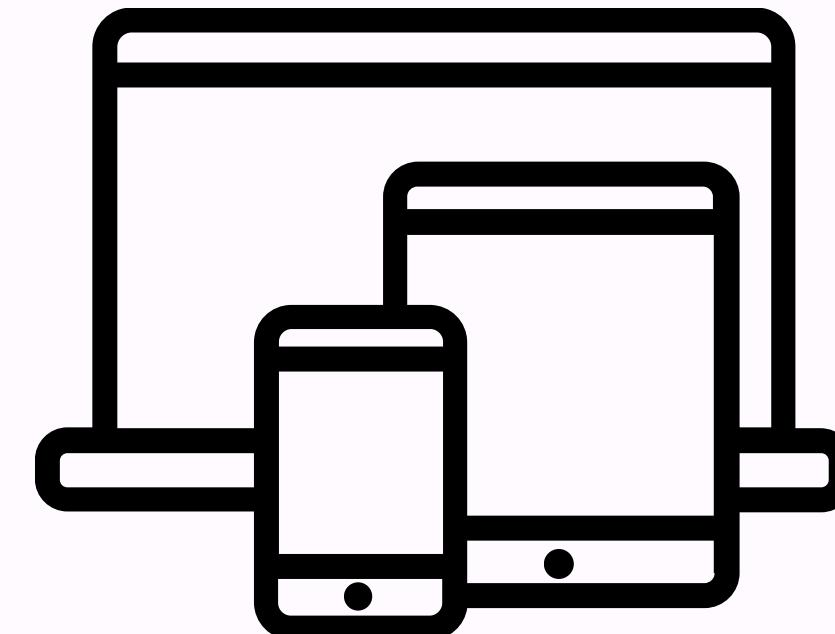


# HTTP/2

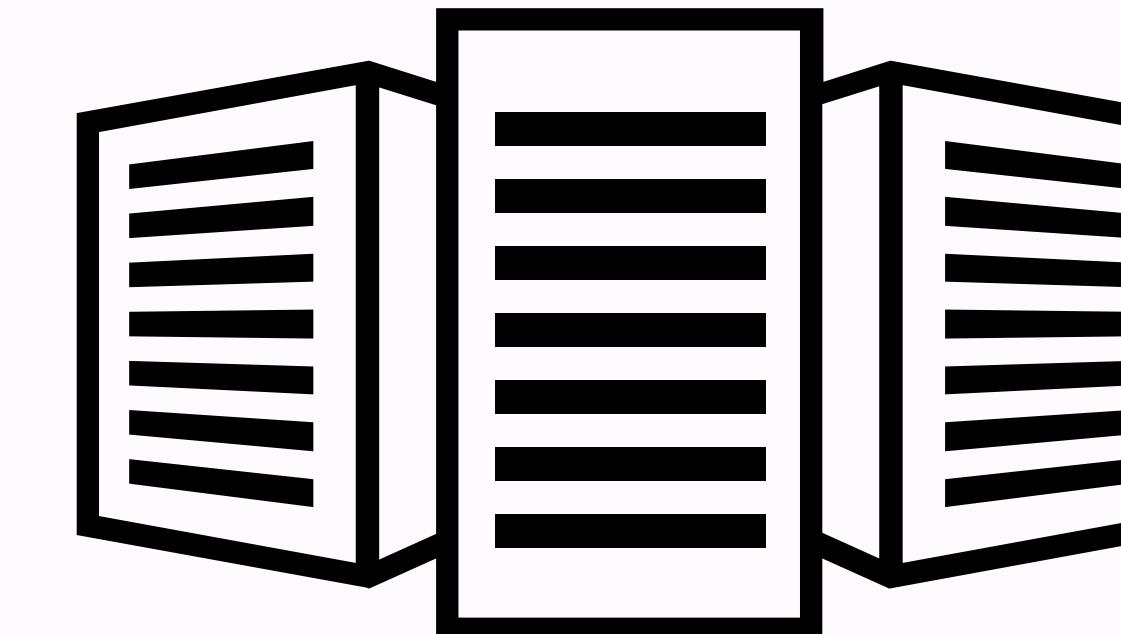
# The Web



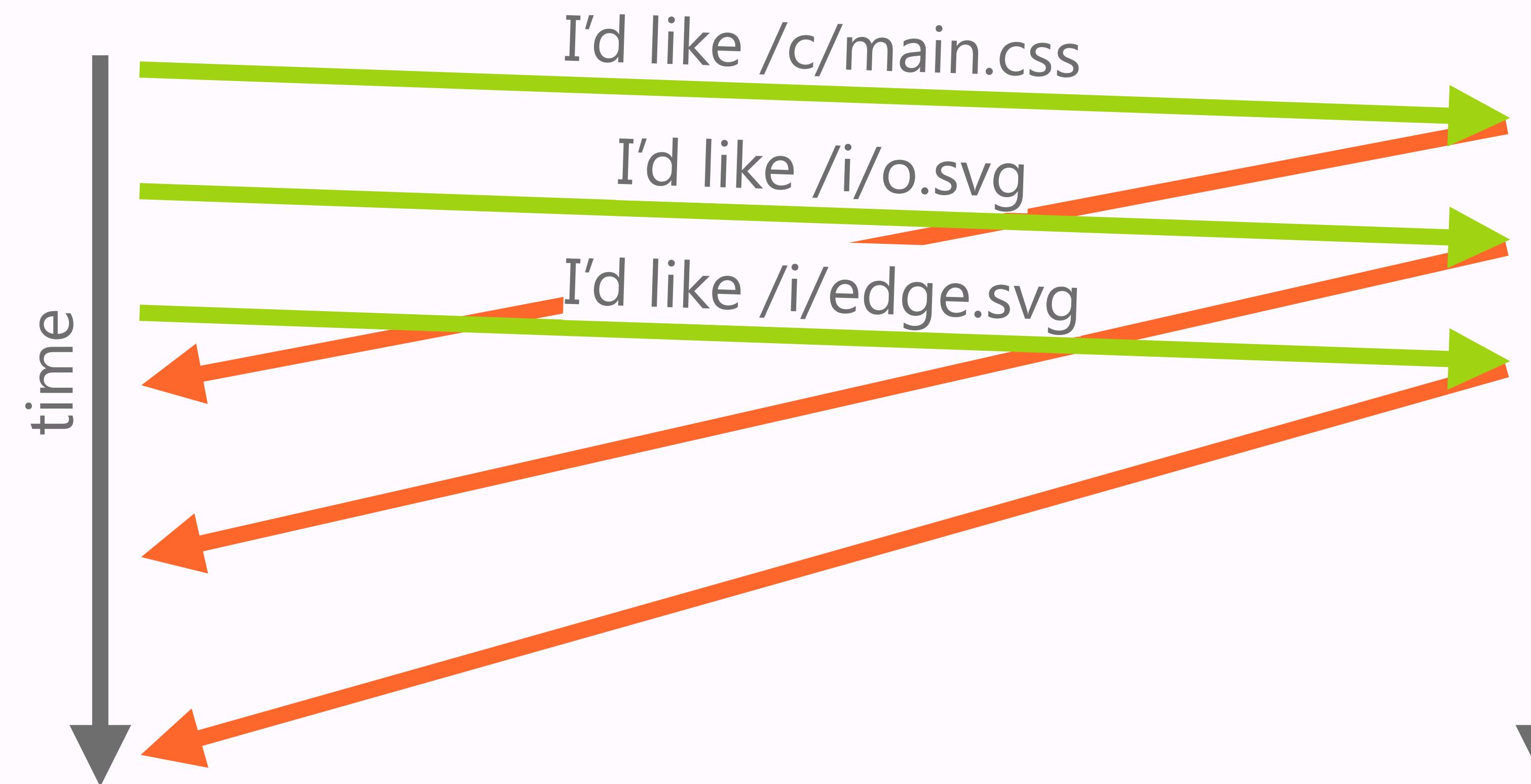
# Your Device

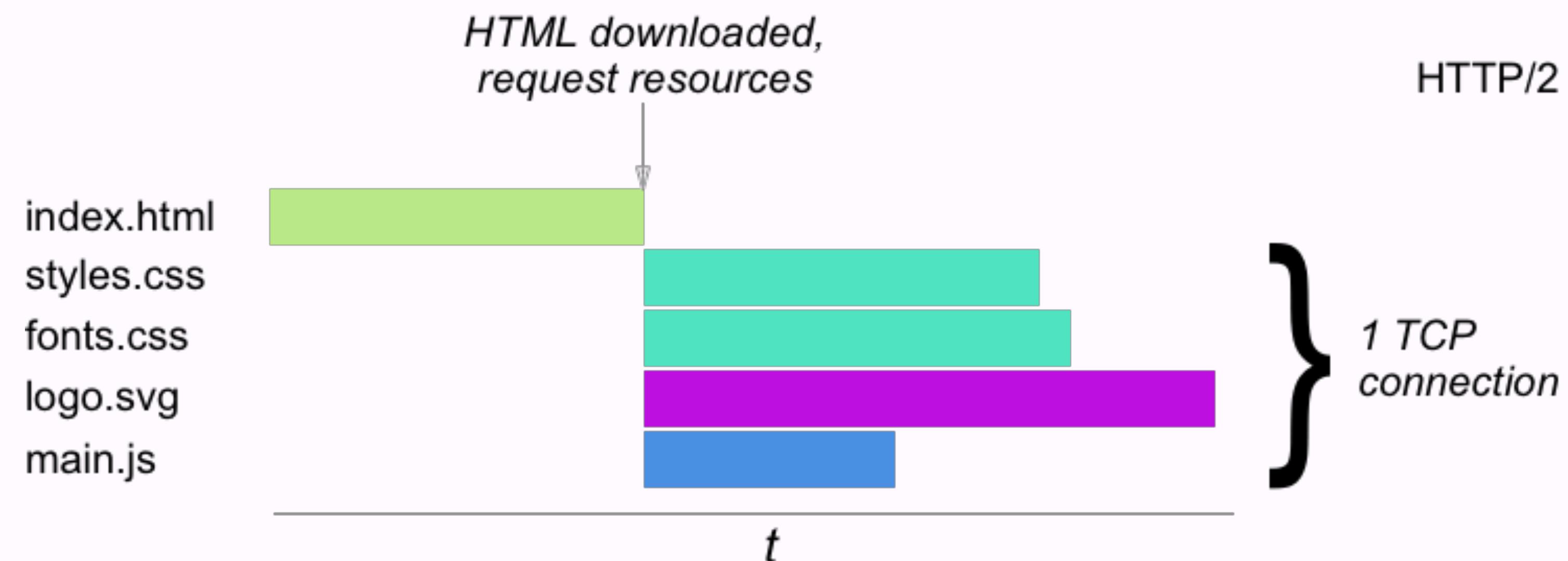
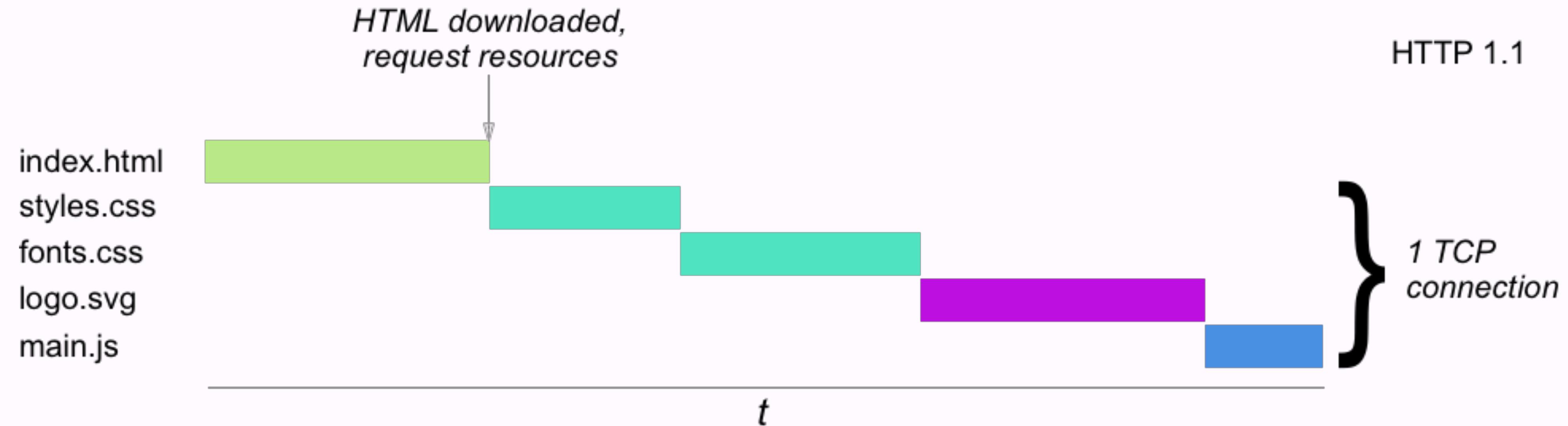


# The Web



HTTP/2











# A LIST APART



Illustration by [Dougal MacPherson](#)

## The Best Request Is No Request, Revisited

by [Stefan Baumgartner](#) · November 28, 2017

Published in [Browsers](#), [The Server Side](#)

Over the last decade, web performance optimization has been controlled by one  
individual: Alfonso.

Source: [A List Apart](#)

**Step 5:  
Consider how  
you load assets**

# Start simple

```
<link rel="stylesheet" href="/c/d.min.css">
<link rel="stylesheet" href="/c/a.min.css"
      media="only screen">
```

# Start simple

```
<link rel="stylesheet" href="/c/d.min.css">  
<link rel="stylesheet" href="/c/a.min.css"  
      media="only screen">
```

*default styles  
(all browsers)*



# Start simple

```
<link rel="stylesheet" href="/c/d.min.css">  
<link rel="stylesheet" href="/c/a.min.css"  
      media="only screen">
```

*default styles  
(all browsers)*

*advanced styles  
(modern browsers)*

# Fault tolerance FTW!

```
<link rel="stylesheet" href="/c/d.min.css">
<link rel="stylesheet" href="/c/a.min.css"
      media="only screen">
```

# Fault tolerance FTW!

```
<link rel="stylesheet" href="/c/d.min.css">
<link rel="stylesheet" href="/c/a.min.css"
      media="only screen">
```



browsers that don't grok  
media queries ignore this file

# Conditional scripting

```
<!--[if lt IE 9]>
  <script src="/j/html5shiv.min.js"></script>
<![endif]-->
```

# Conditional scripting

```
<!--[if lt IE 9]>
<script src="/j/html5shiv.min.js"></script>
<![endif]-->
```

delivers this script  
to <= IE 8

# Conditional scripting

```
<!--[if gt IE 8]><!-->
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js" async
       ></script>
<!--<![endif]-->
</body>
</html>
```

# Conditional scripting

```
<!--[if gt IE 8]><!-->
<script src="/j/main.min.js"></script>
<script src="/j/home.min.js" async
       ></script>
<!--<![endif]-->
</body>
</html>
```

*hides scripts from <= IE8  
(no need to test!)*

# Conditional imagery

Brought to you by [Microsoft Edge](#), in association with  
[An Event Apart](#).

©2007–16 Microsoft | [Legalese](#) |  
[Privacy Statement & Cookies](#) | [Contact Us](#)  
[Find out how this site was made](#)

Microsoft Edge  An Event Apart

Brought to you by [Microsoft Edge](#), in association with [An Event Apart](#).

©2007–16 Microsoft | [Legalese](#) | [Privacy Statement & Cookies](#) | [Contact Us](#)  
[Find out how this site was made](#)

# Conditional images

```
@media (min-width: 36.375em) {  
  .presented-by [href*=microsoft]::before {  
    background-image: url(/i/c/edge.png);  
    background-image: url(/i/c/edge.svg),  
    none;  
  
    ...  
  }  
}
```

# Conditional images

```
@media (min-width: 36.375em) {  
  .presented-by [href*=microsoft]::before {  
    background-image: url(/i/c/edge.png);  
    background-image: url(/i/c/edge.svg),  
    none;  
  
    ...  
  }  
}  
}
```

# Conditional images

```
@media (min-width: 36.375em) {  
  .presented-by [href*=microsoft]::before {  
    background-image: url(/i/c/edge.png);  
    background-image: url(/i/c/edge.svg),  
    none;  
  
    ...  
  }  
}
```

# Conditional images

```
@media (min-width: 36.375em) {  
  .presented-by [href*=microsoft]::before {  
    background-image: url(/i/c/edge.png);  
    background-image: url(/i/c/edge.svg),  
    none;  
  
    ...  
  }  
}
```

# Conditional images

```
@media (min-width: 36.375em) {  
  .presented-by [href*=microsoft]::before {  
    background-image: url(/i/c/edge.png);  
    background-image: url(/i/c/edge.svg),  
    none;  
  }  
  ...  
}
```

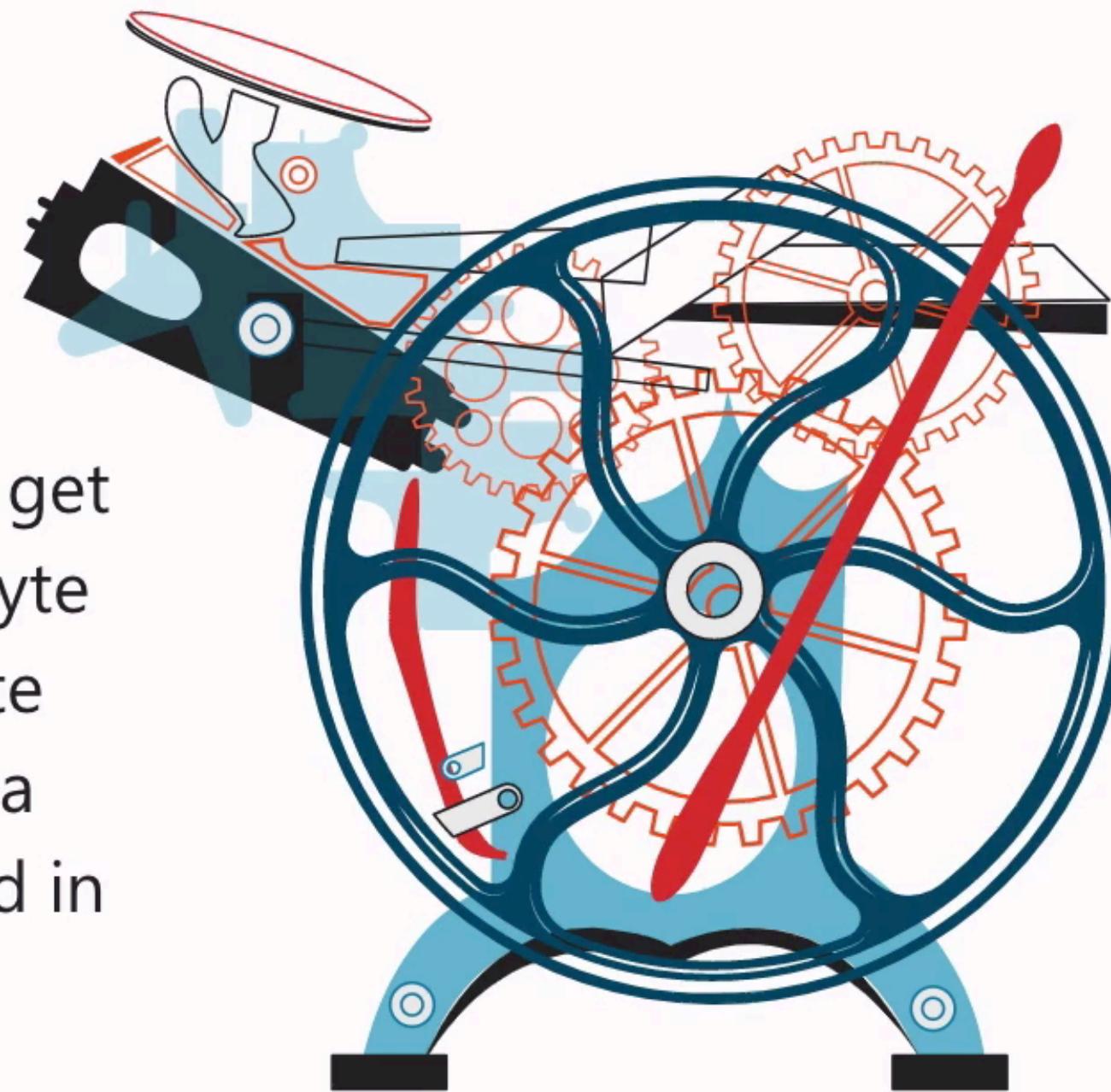


*browsers that support  
multiple backgrounds  
also support SVG*

# Conditional animation

## Inspiring the Web with Just 10k

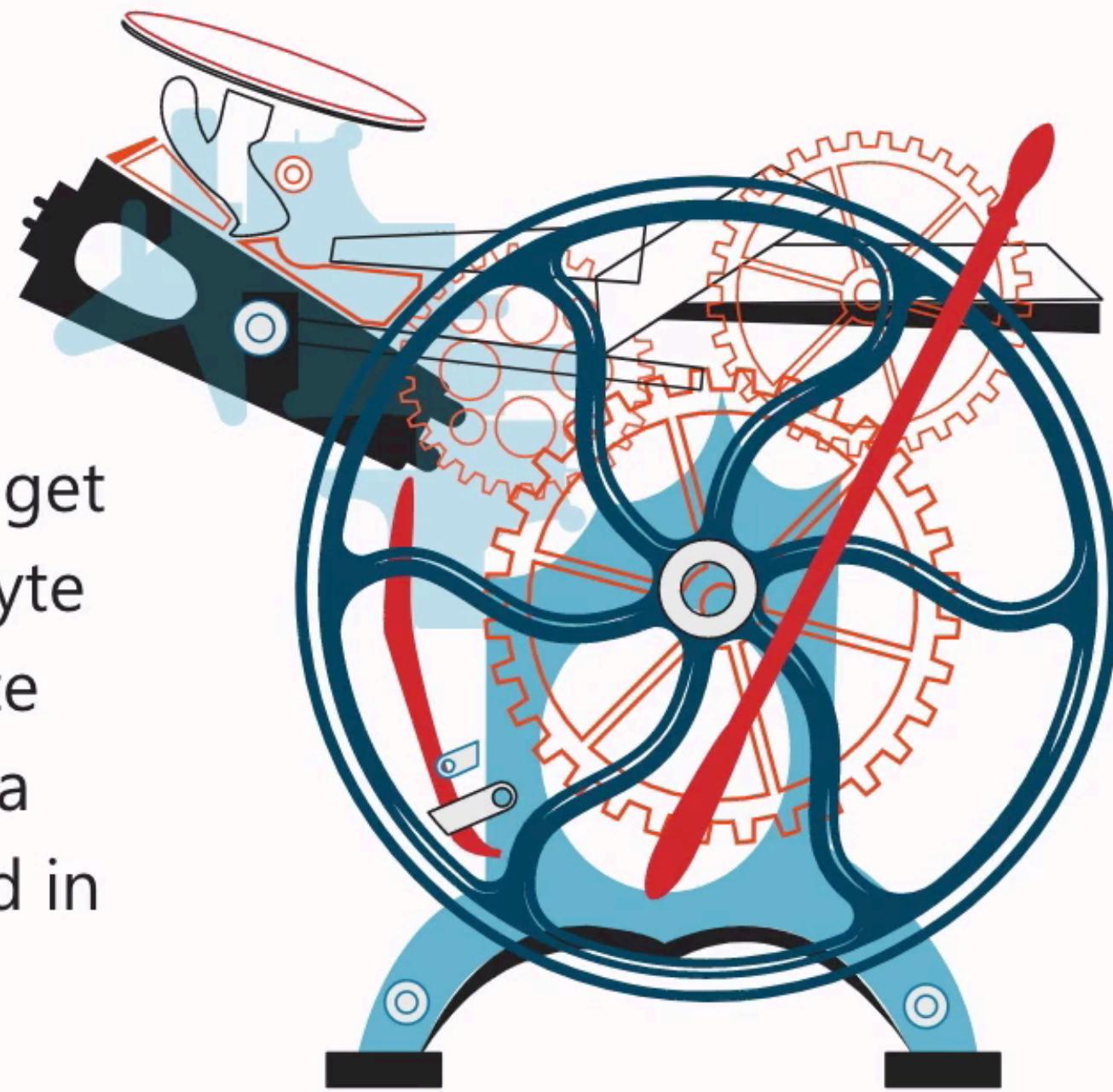
With so much of an emphasis on front-end frameworks and JavaScript runtimes, it's time to get back to basics—back to optimizing every little byte like your life depends on it and ensuring your site can work, no matter what. The Challenge? Build a compelling web experience that can be delivered in 10kB and works without JavaScript.



# Conditional animation

## Inspiring the Web with Just 10k

With so much of an emphasis on front-end frameworks and JavaScript runtimes, it's time to get back to basics—back to optimizing every little byte like your life depends on it and ensuring your site can work, no matter what. The Challenge? Build a compelling web experience that can be delivered in 10kB and works without JavaScript.



# How do we get there?

Load

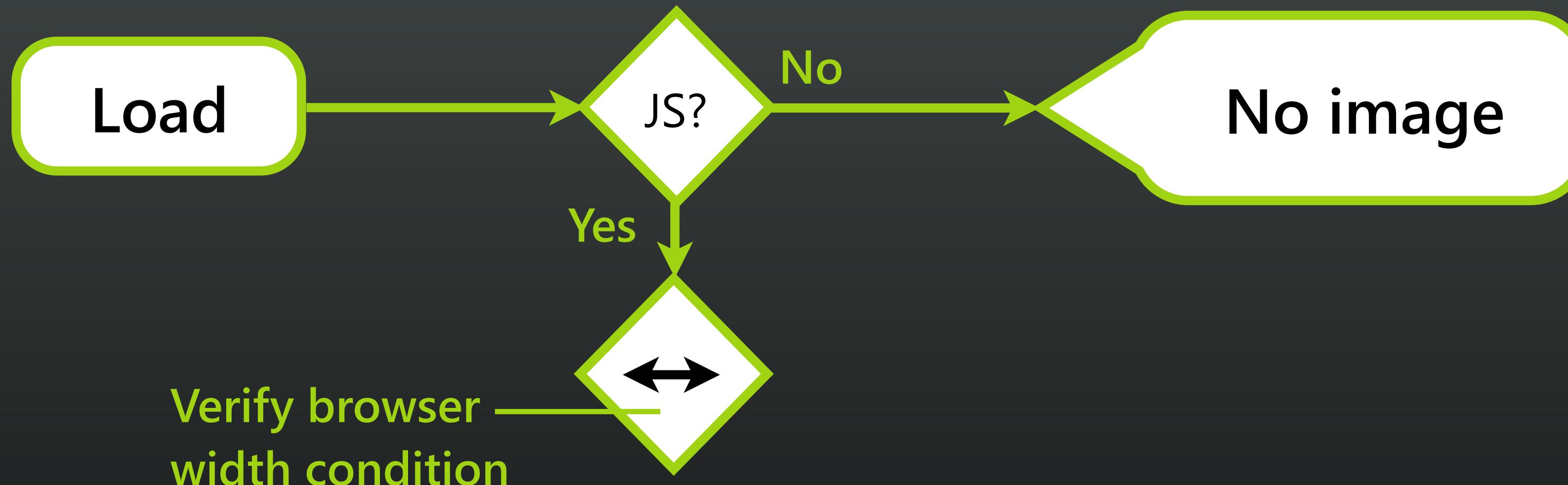
# How do we get there?



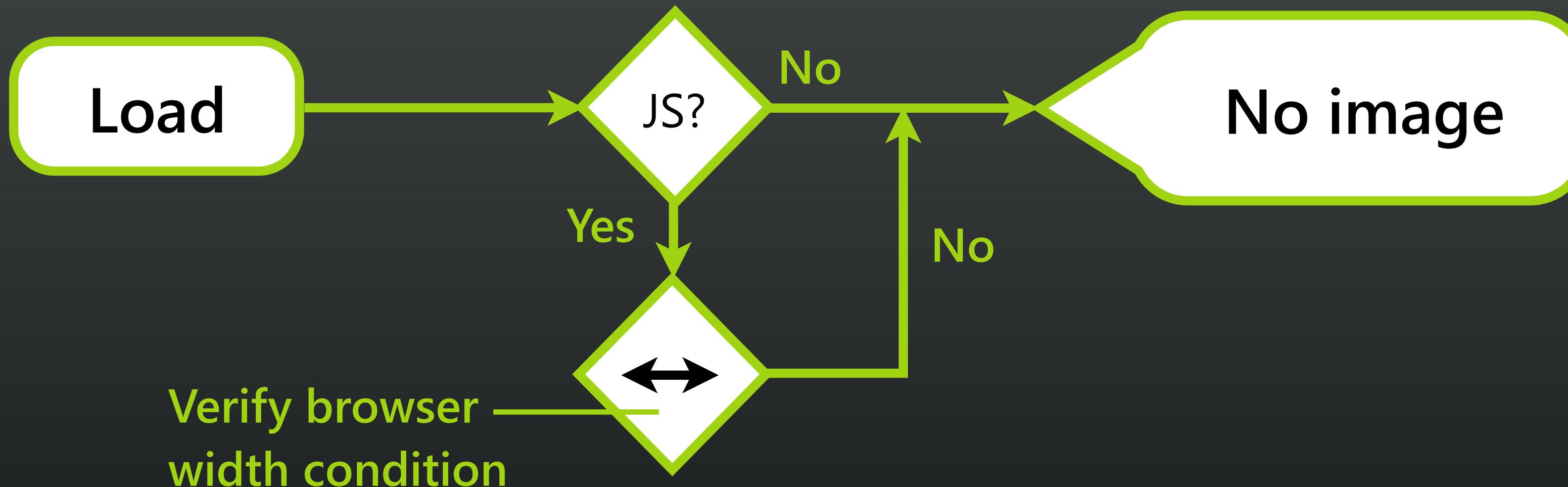
# How do we get there?



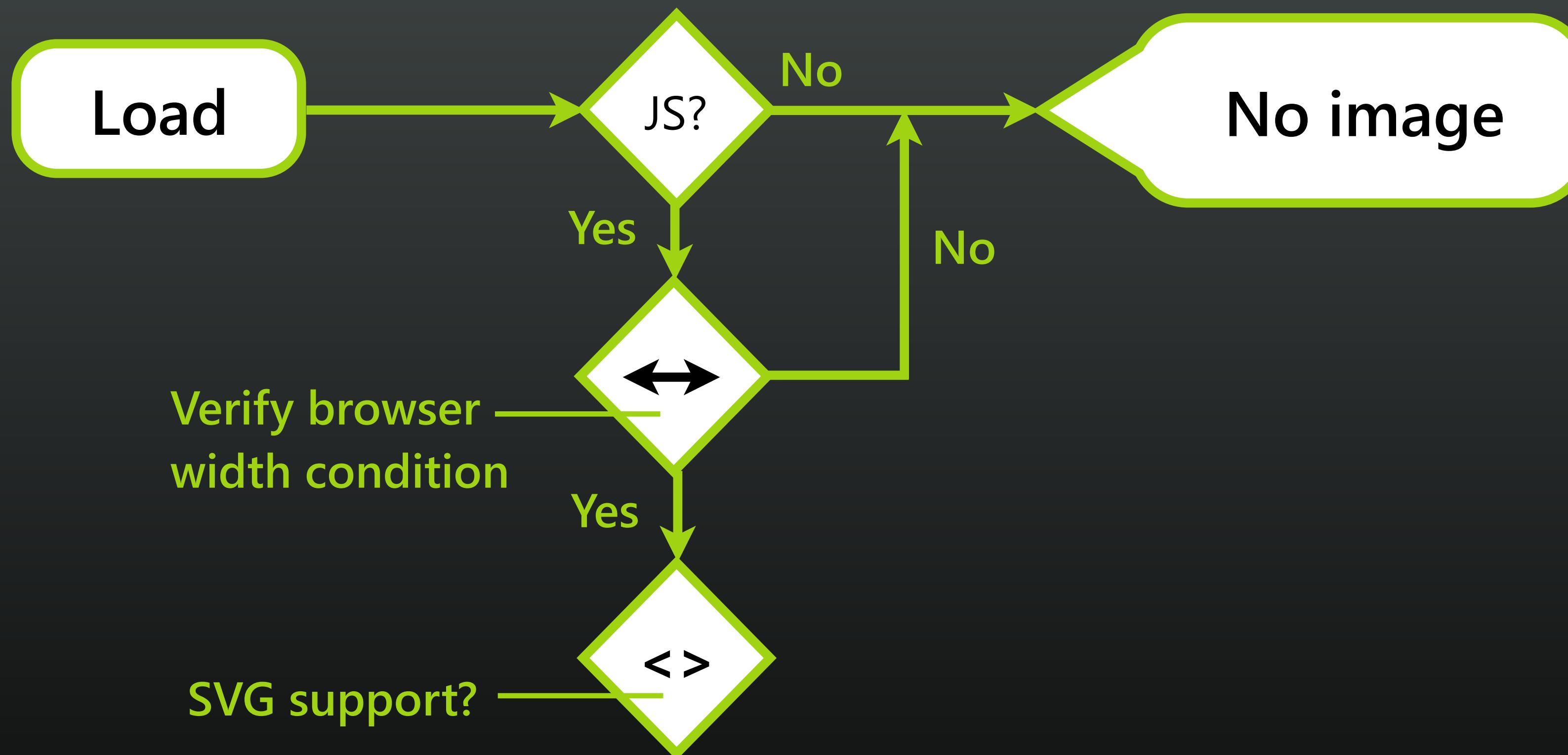
# How do we get there?



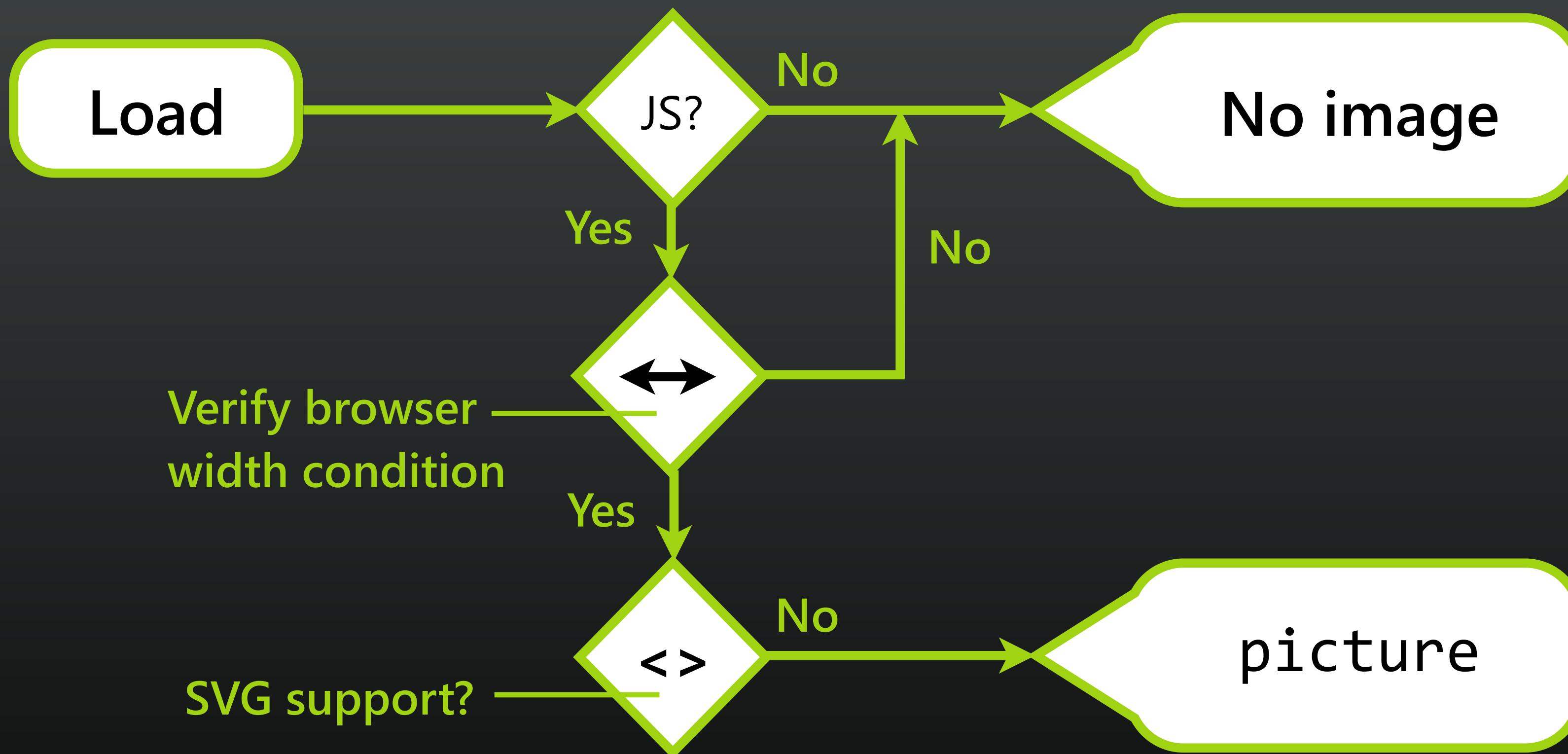
# How do we get there?



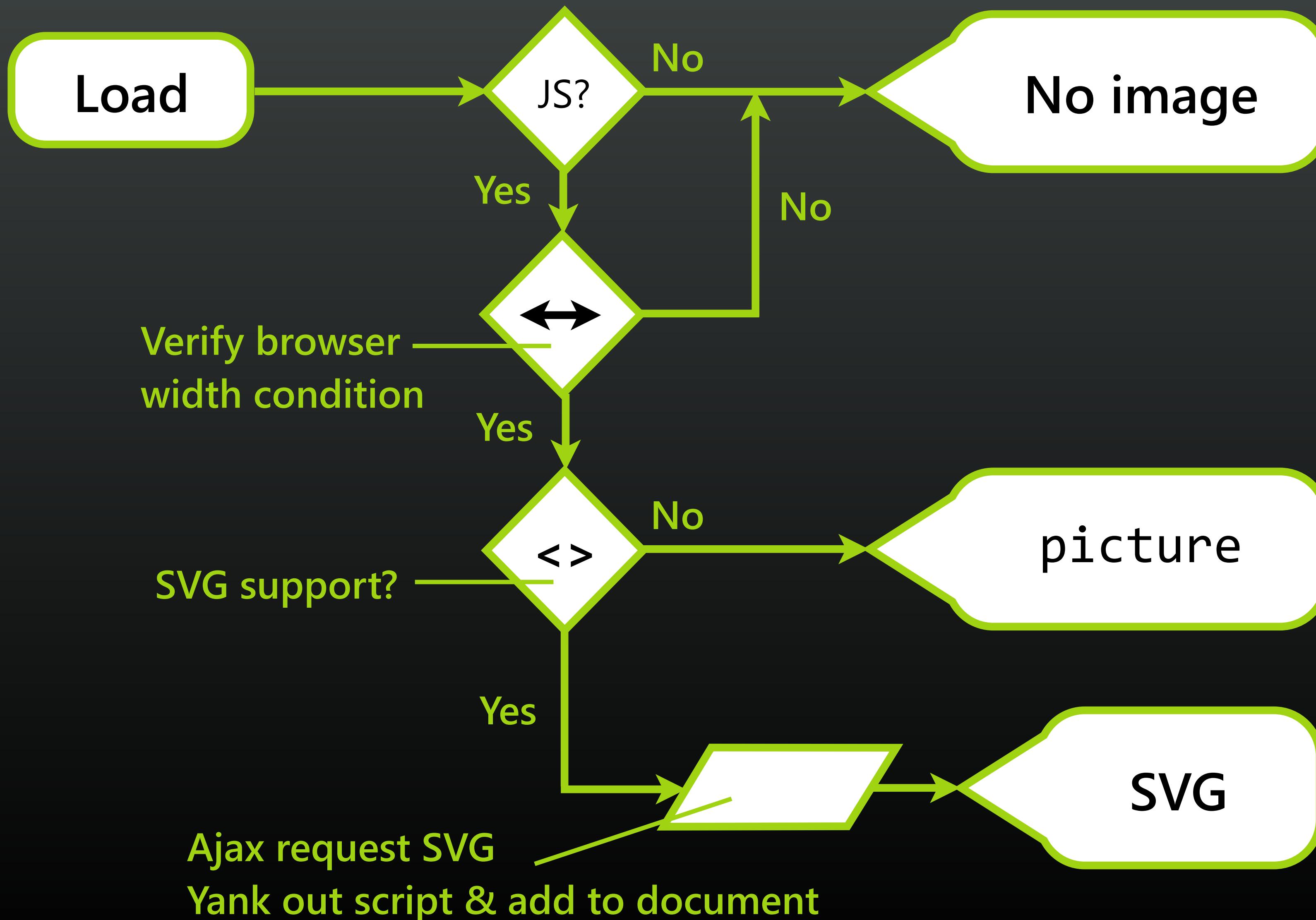
# How do we get there?



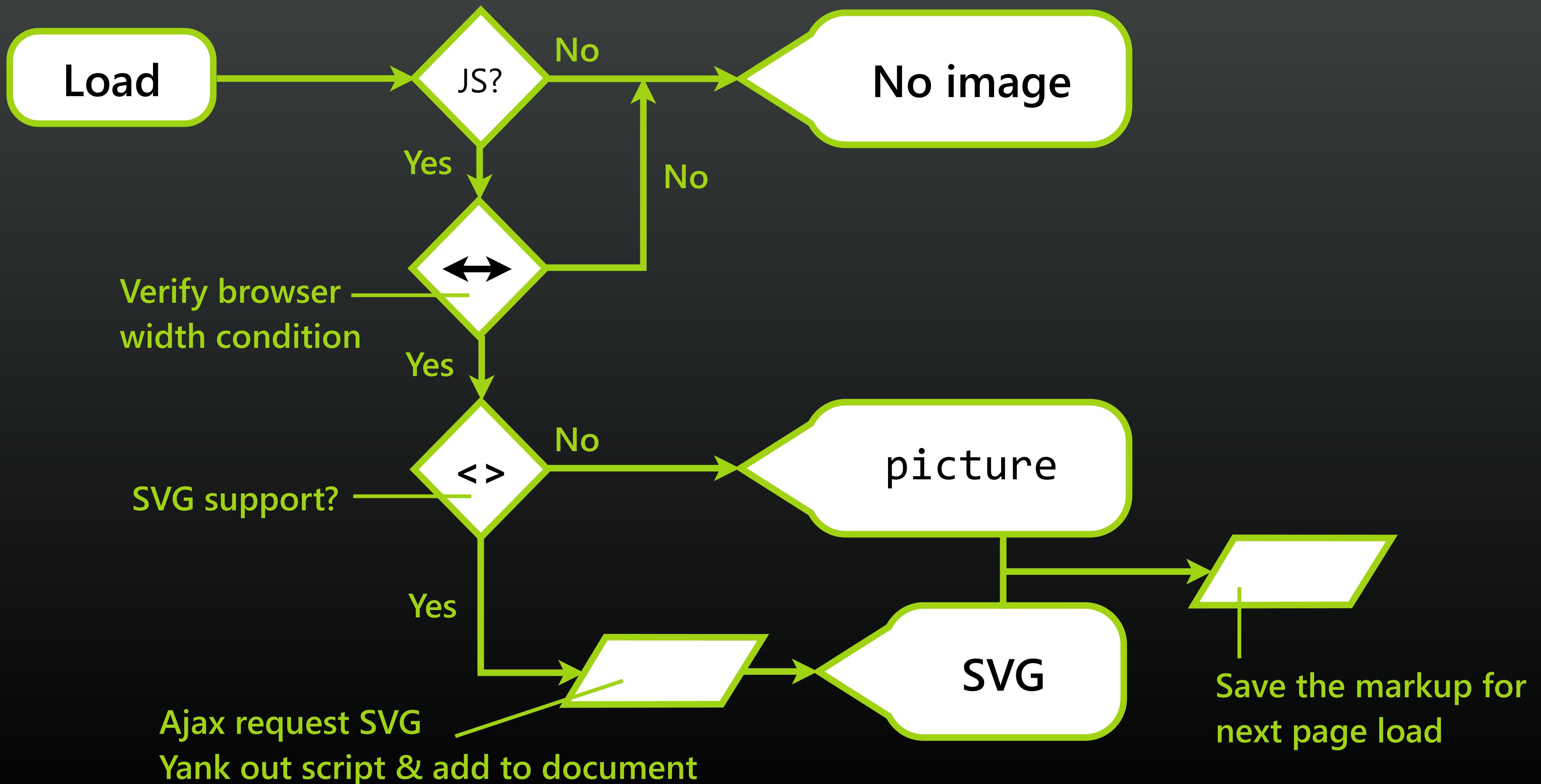
# How do we get there?



# How do we get there?



# How do we get there?



Step 6:  
Only load assets  
**when they add value**

# The New York Times

World U.S. Politics N.Y. Business Opinion Tech Science Health Sports Arts Style Food Travel Magazine T Magazine Real Estate ALL

G.O.P. TAX PLAN

## Adverse Report Threatened Tax Bill, So G.O.P. Went on Attack

- A Republican requirement that Congress consider the full cost of major legislation threatened to derail the party's \$1.5 trillion tax rewrite last week.
- Lawmakers went on the offensive to discredit the agency performing the analysis.

21m ago

At this quarry, how to spend G.O.P. tax cut isn't set in stone.

The failure to repeal Obamacare provided Senate Republicans with a road map of what not to do when pursuing their tax agenda.

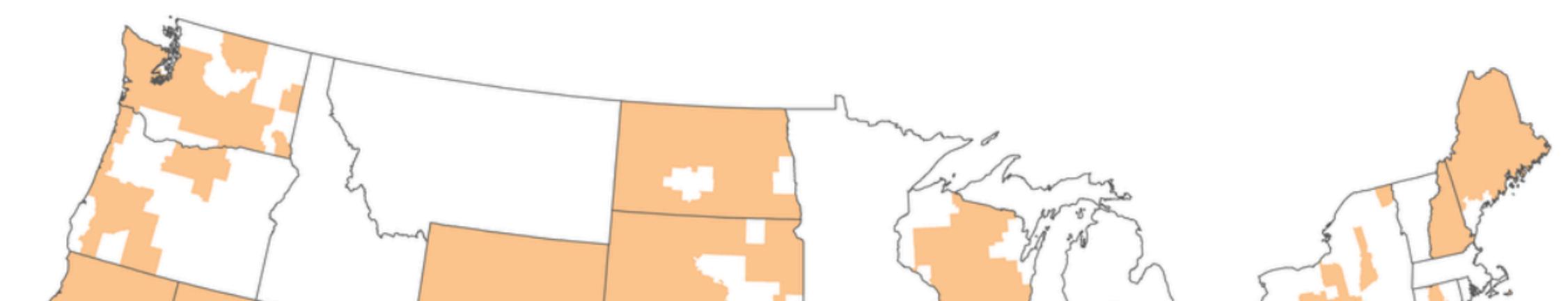


Tom Brenner/The New York Times

## Republicans Try to Avert Federal Shutdown, Due Within Days

- With government funding set to expire Friday, Republicans are pushing for a stopgap spending measure that would extend

## Where a bronze plan costs \$50 more than last year



You're seeing a test of a new version of the New York Times home page.

What do you think?  
Send us your feedback.

Miss the old experience?  
Opt out.

## Opinion



Rob Kim/Getty Images

## Billy Bush Yes, Donald Trump, You Said That

The president is currently engaging in some revisionist history.

Dec. 3

Robert P. George and Sherif

## G.O.P. TAX PLAN



Tom Brenner/The New York Times

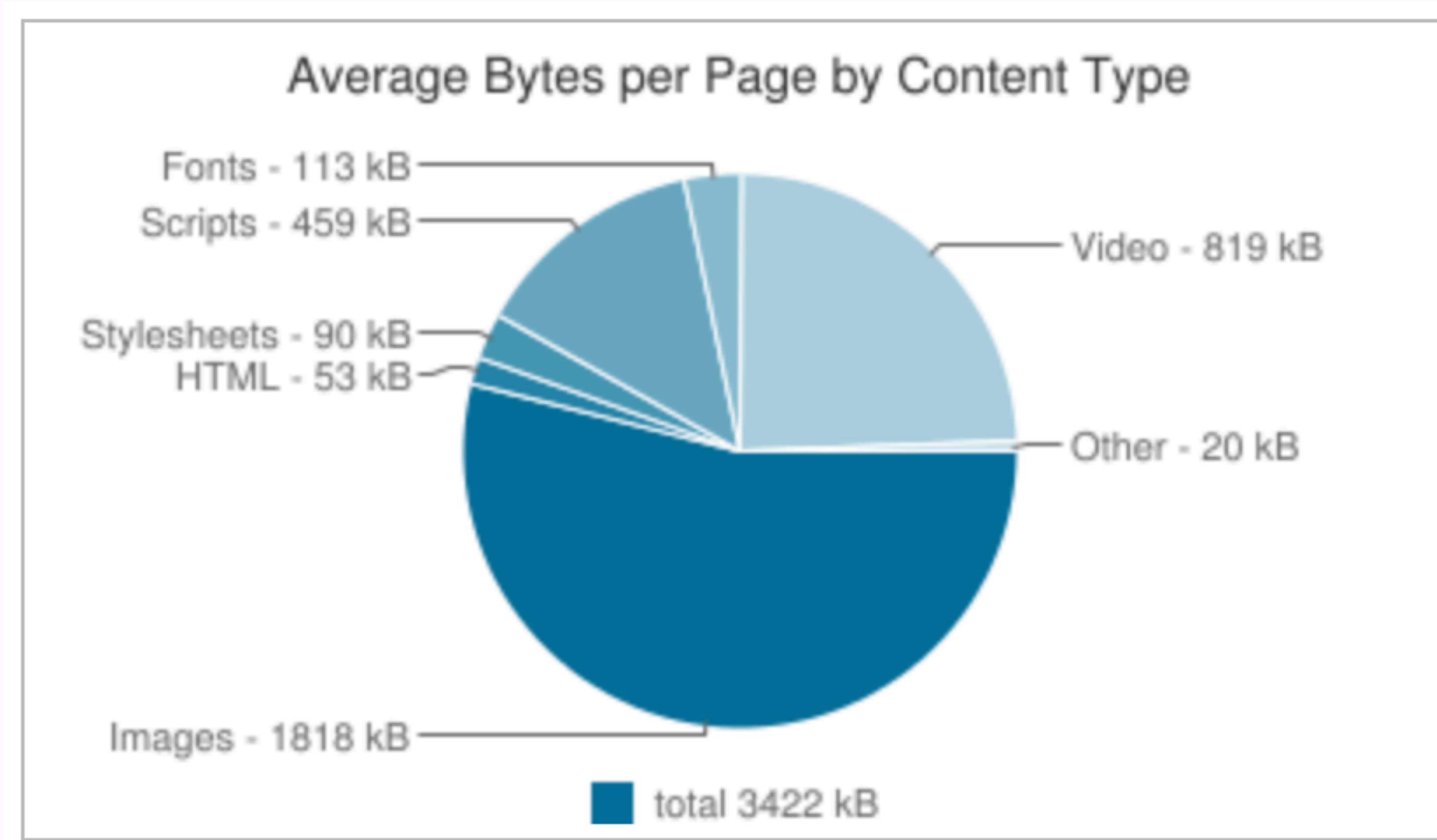
## Adverse Report Threatened Tax Bill, So G.O.P. Went on Attack

- A Republican requirement that Congress consider the full cost of major legislation threatened to derail the party's \$1.5 trillion tax rewrite last week.
- Lawmakers went on the offensive to discredit the agency performing the analysis.

# Evaluate images case-by-case

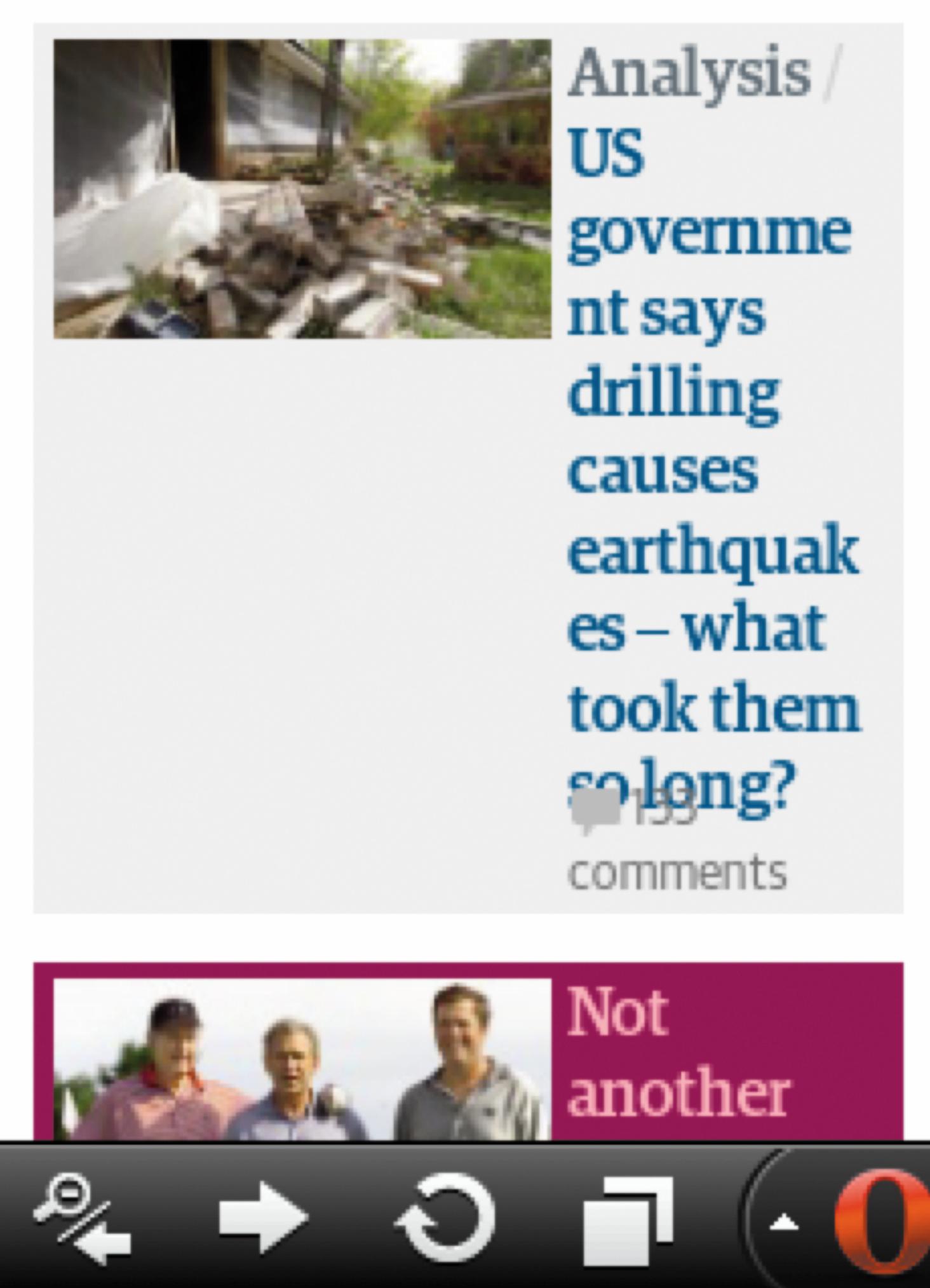
- Does the image reiterate information found in the surrounding text?
- Is the image necessary to understand the surrounding content?
- Does the image contain text?
- Is the image a graph, chart, or table?
- Could the content of the image be presented in a different format that would not require an image?
- Is the image purely presentational?

# 53% of the average web page



Source: [Internet Archive](#)

# And they don't always fit





MEDIA

# NOT EVERY ARTICLE NEEDS A PICTURE

It is dumb to keep forcing images into every story online.



Source: [The Outline](#)

If you can avoid  
using an image, do it

If you need an image,  
choose the best format

# Quick format recap

- **GIF**
  - ▶ for images with large swaths of solid colors
  - ▶ Binary transparency
- **JPG**
  - ▶ For photographs and images with gradations of color
  - ▶ Can be compressed (introduces artifacts)

# Quick format recap

- **PNG (8-Bit)**
  - ▶ Alternative to GIF
  - ▶ Can support alpha transparency (with the right creation software)
- **PNG (24-bit)**
  - ▶ Alternative to JPG
  - ▶ Usually larger than JPGs
  - ▶ Supports alpha transparency

# Quick format recap

- **WebP**
  - ▶ Newer format, not universally supported
  - ▶ Smaller than JPGs and 24-bit PNGs
  - ▶ Support alpha transparency
  - ▶ and so much more...

Sometimes images  
are “nice to have”

---

## Who judged this thing?



Rachel  
Andrew



Lara Hogan



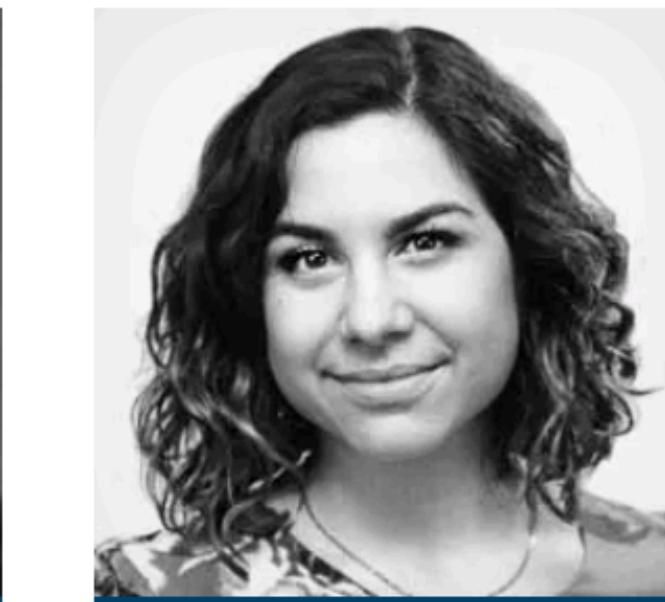
Mat Marquis



Heydon  
Pickering



Jen  
Simmons



Una Kravets

---

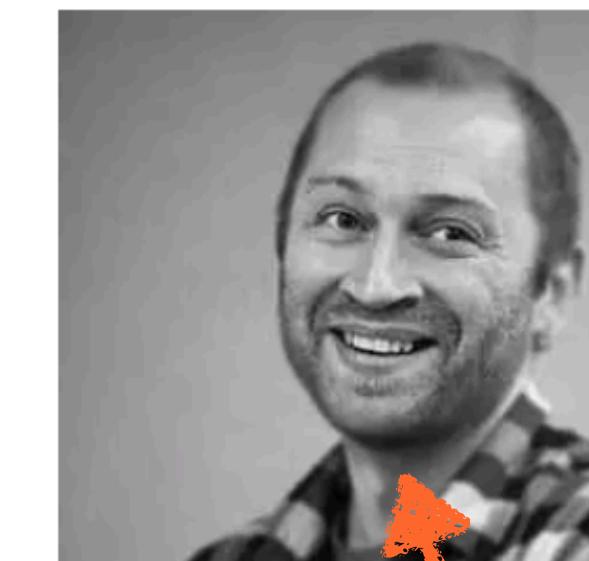
## Who judged this thing?



Rachel  
Andrew



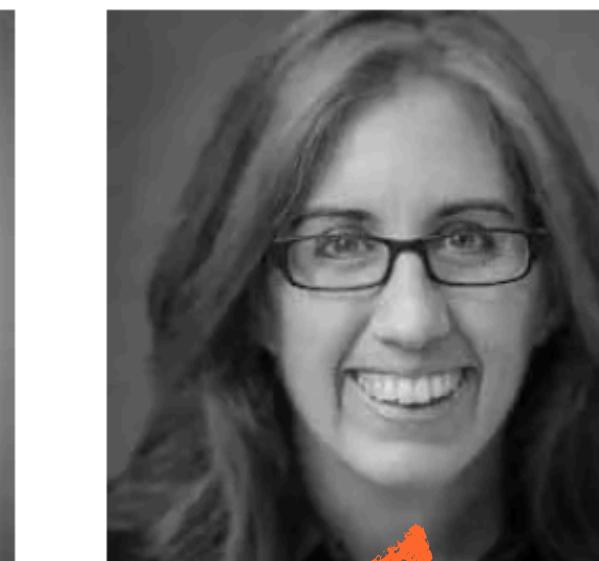
Lara Hogan



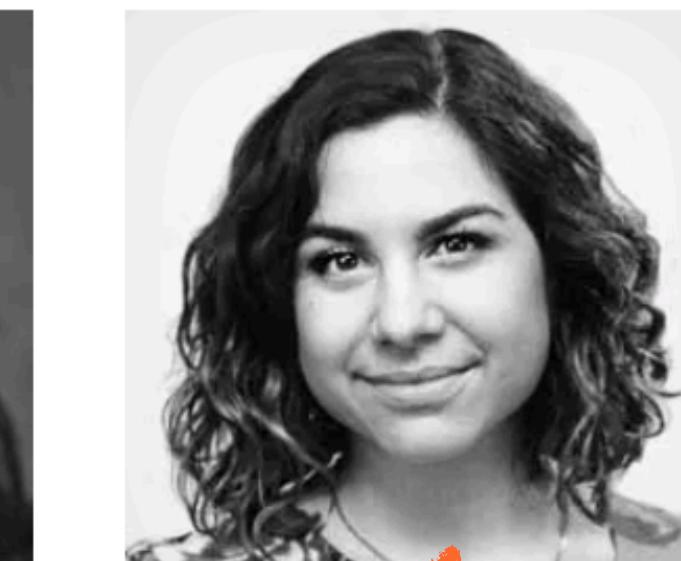
Mat Marquis



Heydon  
Pickering



Jen  
Simmons



Una Kravets

that's 29 kB of images

---

## Who judged this thing?

Rachel  
Andrew

Lara Hogan

Mat Marquis

Heydon  
Pickering

Jen  
Simmons

Una Kravets

# How it works

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
  >
  <a href="https://twitter.com/rachelandrew">
    <b>Rachel Andrew</b>
  </a>
</li>
```

# How it works

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
  >
  <a href="https://twitter.com/rachelandrew">
    <b>Rachel Andrew</b>
  </a>
</li>
```

# How it works

```
<li class="gallery__item h-card"  
    data-img="/i/j/r.jpg|ly"  
    >  
    <a href="https://twitter.com/rachelandrew">  
        <b>Rachel Andrew</b>  
    </a>  
</li>
```

*image path*



# How it works

```
<li class="gallery__item h-card"  
    data-img="/i/j/r.jpg|y"  
    >  
    <a href="https://twitter.com/rachelandrew">  
        <b>Rachel Andrew</b>  
    </a>  
</li>
```

*image path*

*no alt necessary*

# How it works

```
<li class="gallery__item h-card"  
    data-img="/i/j/r.jpg|y"  
    >  
    <a href="https://twitter.com/rachelandrew">  
        <b>Rachel Andrew</b>  
    </a>  
</li>
```

*image path*

*no alt necessary*

*prepend to  
this list item*

# Getting CSS & JS in sync

```
var $watcher = document.createElement('div'),  
    re = /["]/g;  
  
$watcher.setAttribute( 'id', 'getActiveMQ-watcher' );  
$watcher.style.display = 'none';  
document.body.appendChild( $watcher );  
  
window.getActiveMQ = function() {  
    return window.getComputedStyle( $watcher, null )  
        .getPropertyValue( 'font-family' )  
        .replace( re, '' );  
};
```

# Getting CSS & JS in sync

```
#getActiveMQ-watcher { font-family: global; }
@media (min-width: 15em) {
  #getActiveMQ-watcher { font-family: tiny; }
}
@media (min-width: 20em) {
  #getActiveMQ-watcher { font-family: small; }
}
@media (min-width: 30em) {
  #getActiveMQ-watcher { font-family: medium; }
}
@media (min-width: 40em) {
  #getActiveMQ-watcher { font-family: large; }
}
@media (min-width: 48.75em) {
  #getActiveMQ-watcher { font-family: larger; }
}
@media (min-width: 60em) {
  #getActiveMQ-watcher { font-family: full; }
}
```

# Getting CSS & JS in sync

```
#getActiveMQ-watcher { font-family: global; }
@media (min-width: 15em) {
  #getActiveMQ-watcher { font-family: tiny; }
}
@media (min-width: 20em) {
  #getActiveMQ-watcher { font-family: small; }
}
@media (min-width: 30em) {
  #getActiveMQ-watcher { font-family: medium; }
}
@media (min-width: 40em) {
  #getActiveMQ-watcher { font-family: large; }
}
@media (min-width: 48.75em) {
  #getActiveMQ-watcher { font-family: larger; }
}
@media (min-width: 60em) {
  #getActiveMQ-watcher { font-family: full; }
}
```

# Getting CSS & JS in sync

```
var $watcher = document.createElement('div'),  
    re = /["]/g;  
  
$watcher.setAttribute( 'id', 'getActiveMQ-watcher' );  
$watcher.style.display = 'none';  
document.body.appendChild( $watcher );  
  
window.getActiveMQ = function() {  
    return window.getComputedStyle( $watcher, null )  
        .getPropertyValue( 'font-family' )  
        .replace( re, '' );  
};
```

# Getting CSS & JS in sync

```
var MQ = getActiveMQ();

if ( MQ == 'larger' || MQ == 'full' ) {
  lazyLoadImages();
}
```

# How it works

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
  >
  <a href="https://twitter.com/rachelandrew">
    <b>Rachel Andrew</b>
  </a>
</li>
```

# Result!

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
    data-imaged="true"
    >

<a href="https://twitter.com/rachelandrew">
    <b>Rachel Andrew</b>
</a>
</li>
```

Oh wait...  
optimize **everything**

---

## Who judged this thing?



Rachel  
Andrew



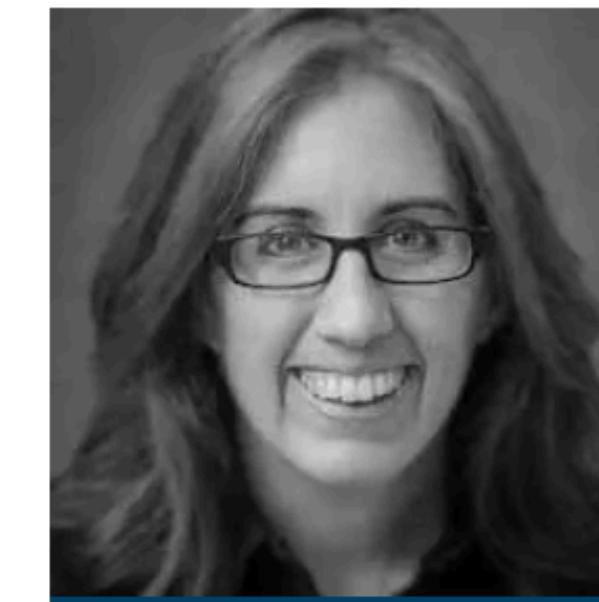
Lara Hogan



Mat Marquis



Heydon  
Pickering



Jen  
Simmons



Una Kravets



Rachel  
Andrew



Rachel  
Andrew



Rachel  
Andrew



Source: 38 kB JPG



B&W: 35 kB JPG (-7%)



Crop & Resize: 12 kB JPG (-68%)



Blur & optimize: 9 kB JPG (-76%)

---

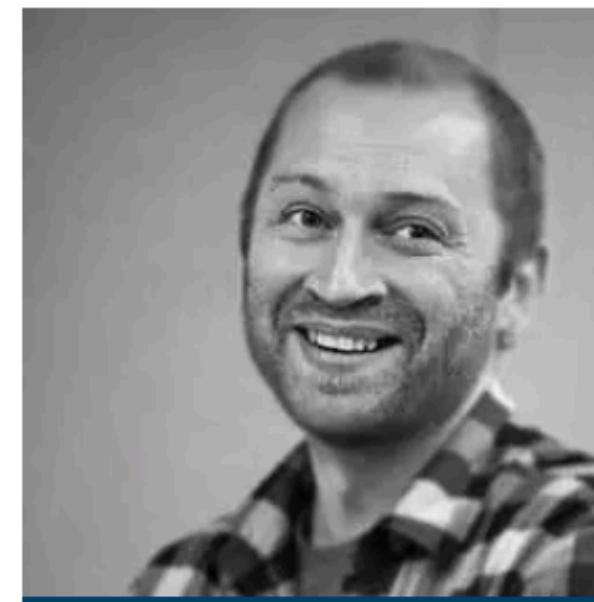
## Who judged this thing?



Rachel  
Andrew



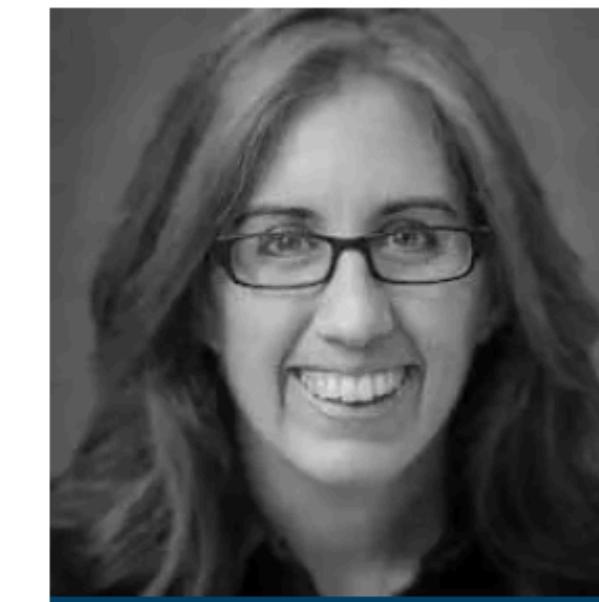
Lara Hogan



Mat Marquis



Heydon  
Pickering



Jen  
Simmons



Una Kravets

---

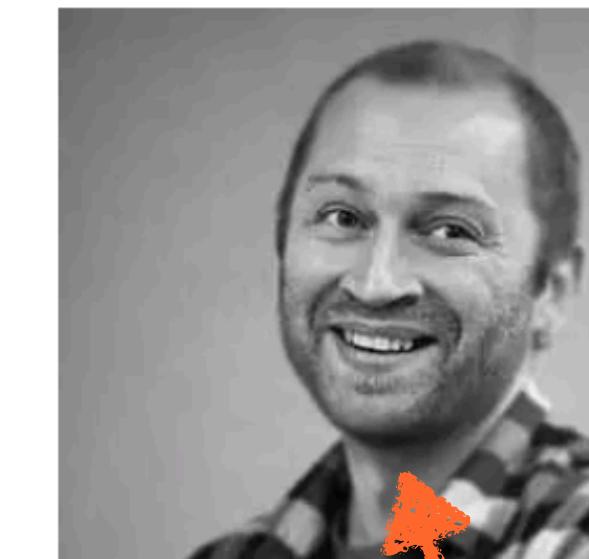
## Who judged this thing?



Rachel  
Andrew



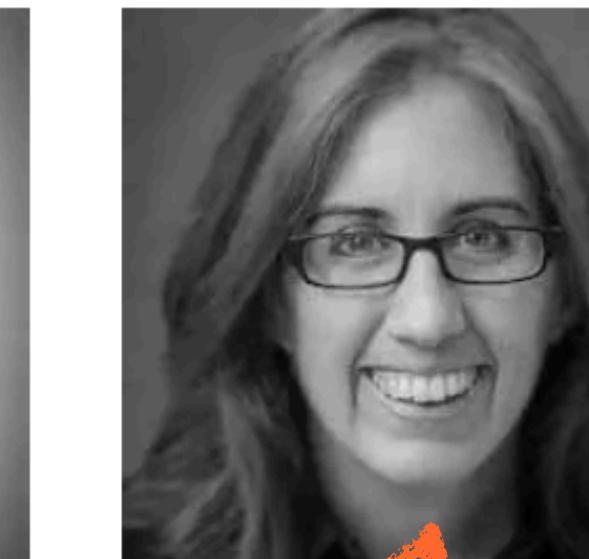
Lara Hogan



Mat Marquis



Heydon  
Pickering



Jen  
Simmons



Una Kravets

blurred everything  
but their faces



**JPG: 9 kB (-76%)**



**WebP: 4 kB (-89%)**



# Not every browser supports WebP

Browser	User Share (%)
IE	11
Edge *	18
Firefox	63
Chrome	70
Safari	12
Opera	57
iOS Safari *	12.1
Opera Mini *	all
Android Browser	67
Blackberry Browser	10
Opera Mobile *	46
Chrome for Android	70
Firefox for Android	63
IE Mobile	11
UC Browser for Android	11.8

Legend for browser versions:

- Light Gray: Primary version
- Red: Secondary version
- Green: Primary version
- Blue: Secondary version
- Dark Gray: Primary version
- Maroon: Secondary version
- Orange: Primary version
- Yellow: Secondary version
- Purple: Primary version

# Indicating alternate formats

```
<picture>
  <source type="image/webp" srcset="my.webp">
  
</picture>
```

# Indicating alternate formats

```
<picture>
  <source type="image/webp" srcset="my.webp">
    
</picture>
```

*first choice if  
WebP is supported*

# Indicating alternate formats

```
<picture>
  <source type="image/webp" srcset="my.webp">
    
  </picture>
```

*first choice if  
WebP is supported*

*fallback image  
if it isn't*

# Indicating alternate formats

```
<picture>
  <source type="image/svg+xml"
          srcset="my.svg">
  <source type="image/webp" srcset="my.webp">
  
</picture>
```

# Indicating alternate formats

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
    data-imaged="true"
    >

<a href="https://twitter.com/rachelandrew">
    <b>Rachel Andrew</b>
</a>
</li>
```

# Indicating alternate formats

```
<li class="gallery__item h-card"
    data-img="/i/j/r.jpg|ly"
    data-imaged="true"
    >
  <picture>
    <source type="image/webp"
            srcset="/i/j/r.webp">
      
    </picture>
    <a href="https://twitter.com/rachelandrew">
      <b>Rachel Andrew</b>
    </a>
  </li>
```

# Steps for better performance

# Steps for better performance

1. Use native features whenever possible

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets
5. Consider how you load assets

# Steps for better performance

1. Use native features whenever possible
2. Only include assets you actually need
3. Optimize everything
4. Think about when you load assets
5. Consider how you load assets
6. Only load assets when they add value

**Every choice we  
make affects our  
users' experiences**

Let's spend our time to  
**save it for our users**

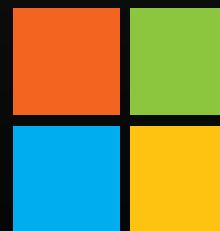
Speedy performance is  
a great user experience

# Thank you!

@AaronGustafson

aaron-gustafson.com

slideshare.net/AaronGustafson



Microsoft