

# Technologie Webowe

*Warstwa prezentacyjna*

Tomek Sułkowski

# Plan na dziś

- Ogólny zarys technologii webowych
- Warstwy front-endu
- Narzędzia i techniki

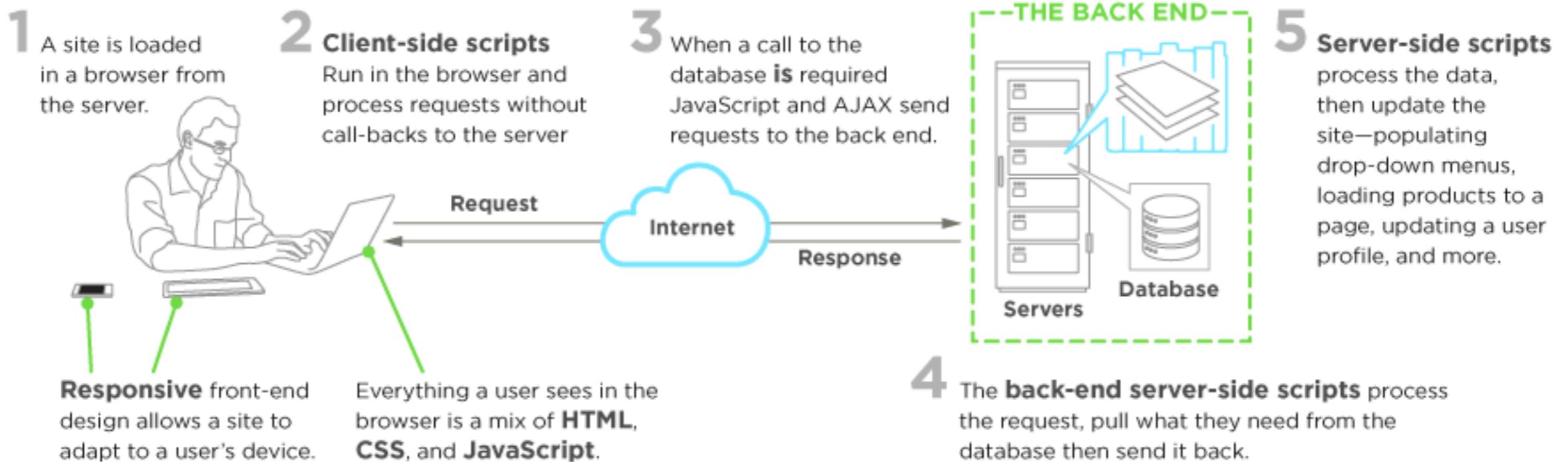
# *poznajmy się*

- Tomek Sułkowski
- Front-end Engineer @ StackBlitz / US
- Lead Trainer @ Sages / Warszawa
- 13 lat pracy z WWW
- 7 lat 100% z front-endem

# WWW: ogólnie

## *Co jest czym w internecie?*

### FRONT-END DEVELOPMENT

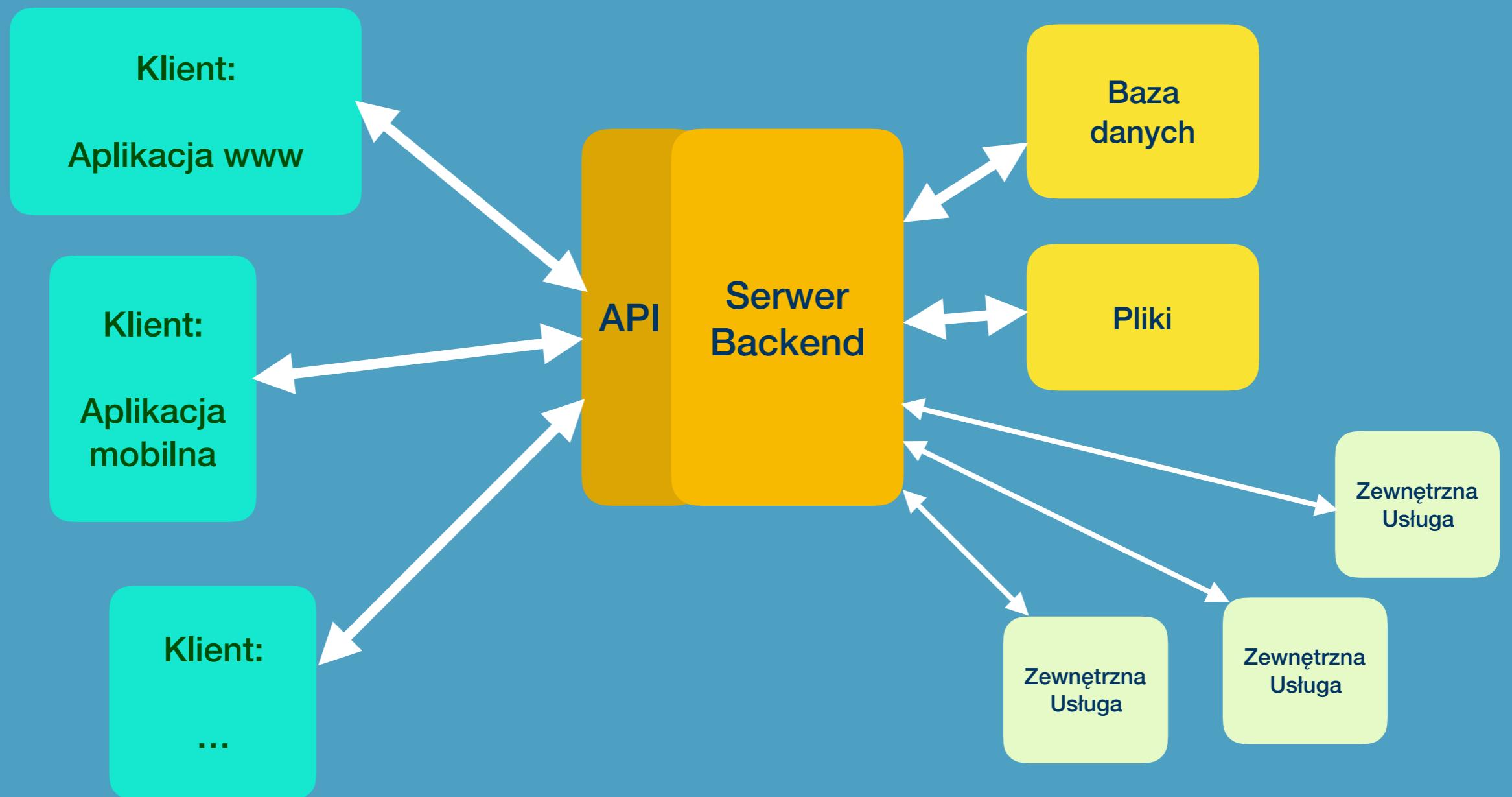


# Frontend vs Backend



# WWW: ogólnie

## Typowa architektura



# Backend?

- Typowe języki:  
Java, PHP, .Net, Ruby, Python
- Również bazy danych, DevOps
- Niezależny od front-endu  
Front-end może nie wiedzieć o technologii back-endu
- 2 typowe sposoby komunikacji:
  - “REST API” - frontend pyta->backend odpowiada
  - Socket connection - połączenie w czasie rzeczywistym

# Frontend?

- Typowe języki:  
JavaScript, CSS, HTML, TypeScript, Sass
- Niezależny od back-endu  
Back-end może nie wiedzieć o technologii front-endu
- 2 typowe sposoby komunikacji:
  - “REST API” - frontend pyta->backend odpowiada
  - Socket connection - połączenie w czasie rzeczywistym

# Frontend developer?

*jak nas zwą*

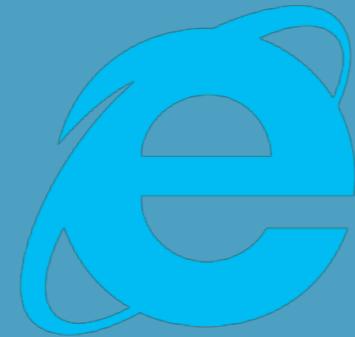
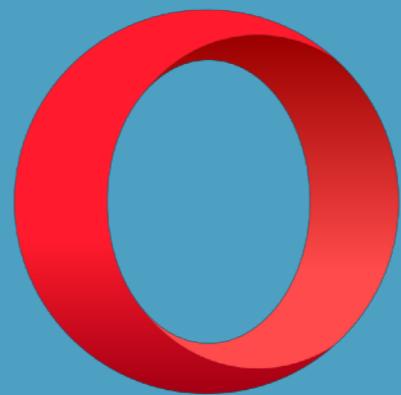
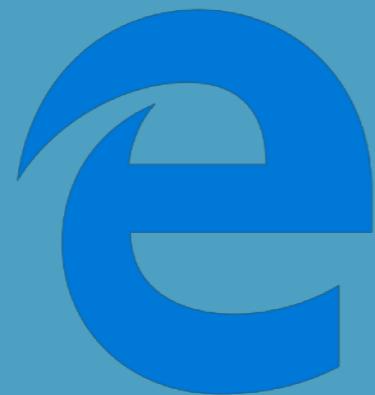
- ~~Webmaster~~
- Front-end Developer
- Front-end Engineer
- CSS/HTML Developer
- Front-End Web Designer
- UI Developer
- Front-end Accessibility Expert
- QA

# Uwaga!

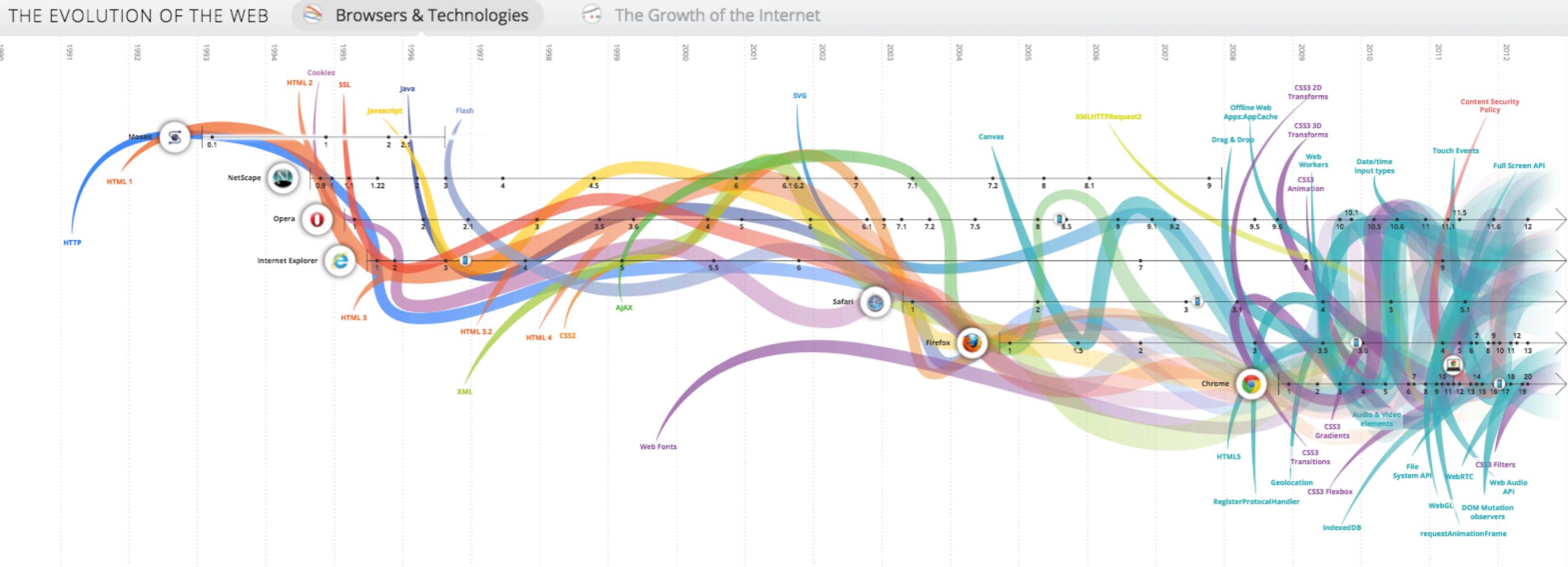


# Przeglądarka

*pierwszy front*



# “nie wszyscy na raz”



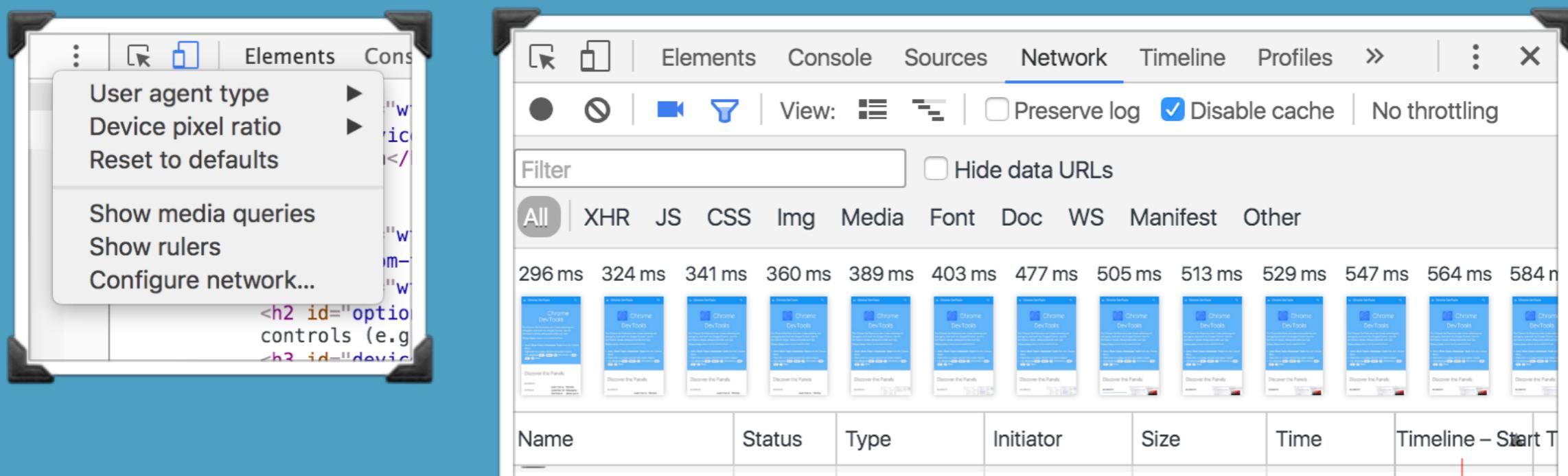
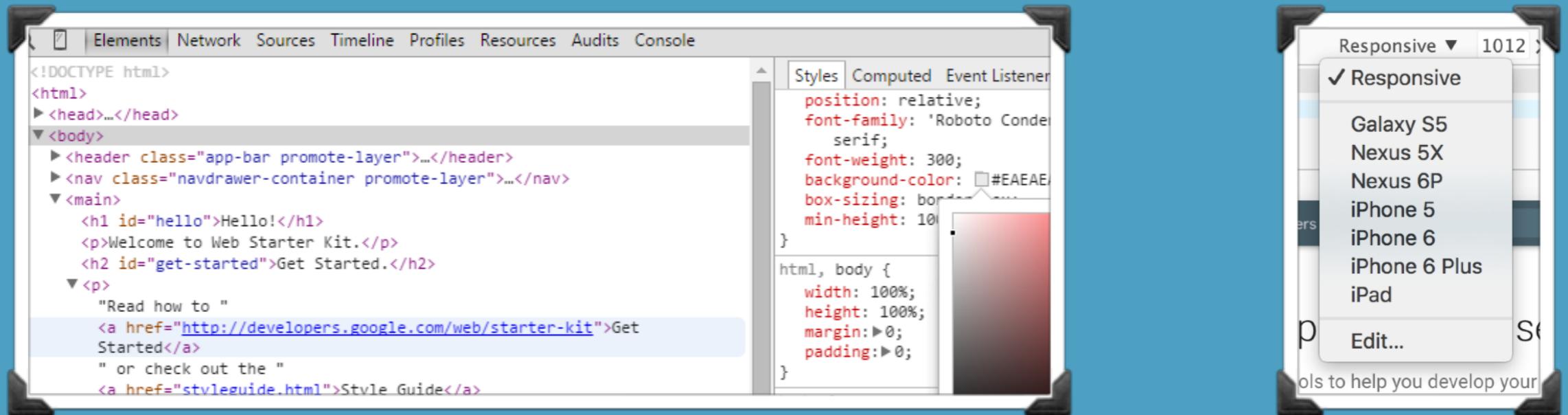
<http://www.evolutionoftheweb.com>

# Przeglądarka

*pierwszy front*

- Ale nie tylko 
  - Mniej popularne (?) silniki
  - Czytniki
  - “Headless browsers”
- 
- Statystyki: <http://gs.statcounter.com>
  - Ważniejsze statystyki: własne!

# Przeglądarka potężne narzędzie developerkie



# Warstwy Front-endu



# Warstwy Front-endu



*HTML*

- HyperText Markup Language
- Struktura
- Znaczenie / Semantyka
- HTML5

# Warstwy Front-endu



*CSS*

- Cascading Style Sheets
- Wygląd
- Statyczne i dynamiczne  
(animacje)
- CSS3 → 4

# Warstwy Front-endu



## *JavaScript*

- Zachowanie



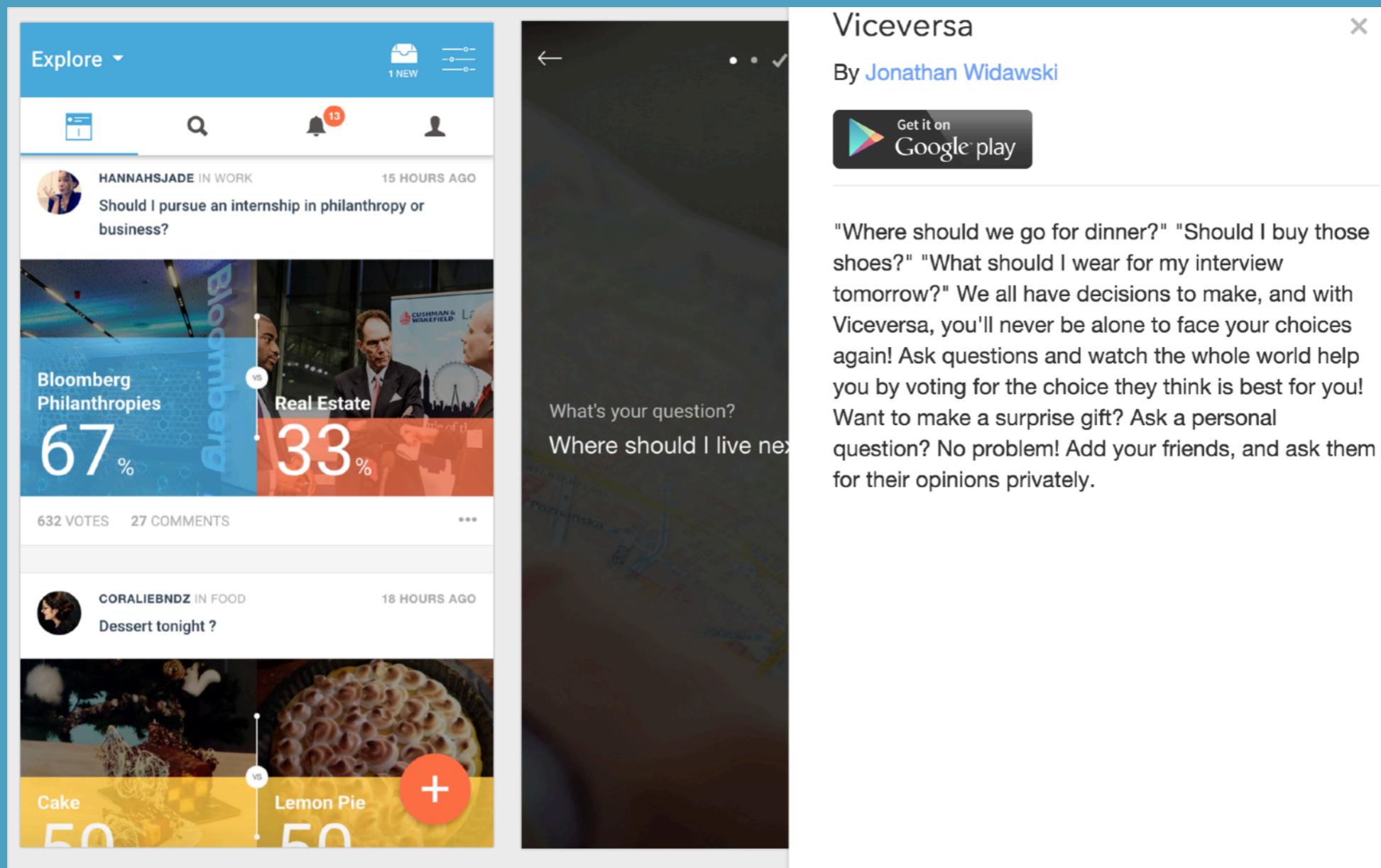
# Warstwy Front-endu

## *inne kluczowe akronimy*

- DOM  
Document Object Model
- Web APIs / Browser APIs/ HTML5 APIs
- JSON  
JavaScript Object Notation
- WCAG  
Web Content Accessibility Guidelines
- WAI-ARIA  
Web Accessibility Initiative: Accessible Rich Internet Applications

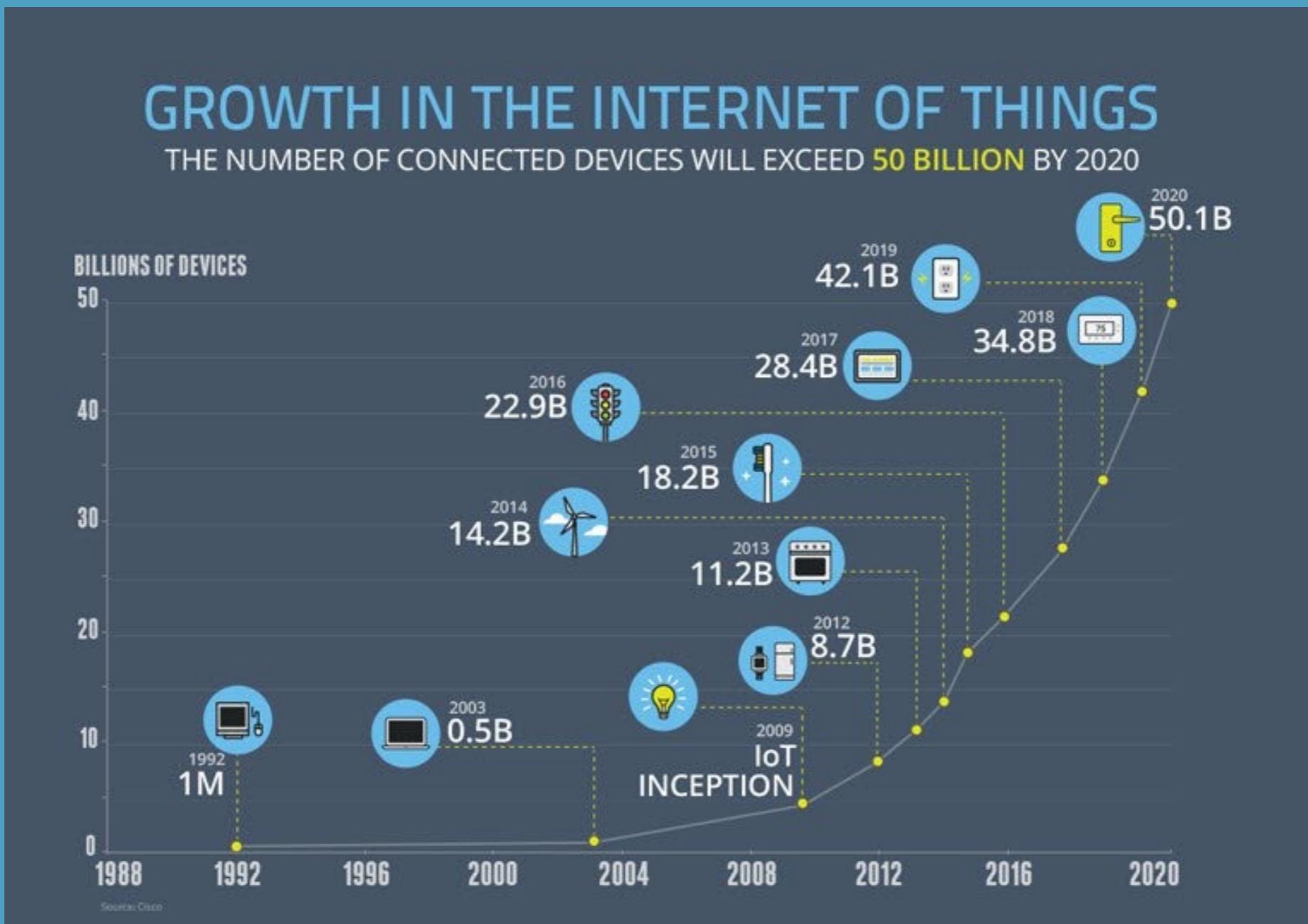
# Czy tylko front-end?

## *Desktop & Mobile*



# Czy tylko front-end?

## *Internet of Things*



# **HTML**

Podstawa

# HTML

## *Podstawowy dokument*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Hello world!</title>
    <meta name="description"
          content="My first basic HTML document">
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

# HTML

## *Elementy języka*

```
<!-- komentarz -->

<p>      <!-- znacznik otwierający -->
    If Tetris has taught me anything
    it's that errors pile up and accomplishments disappear.
</p>      <!-- znacznik zamykający -->

<br>      <!-- znacznik samozamykający się -->
<br/>

<!-- atrybuty -->

```

# HTML

## *2 rodzaje elementów*

### Blokowe:

- <div>
- <p>
- <ul>, <ol>
- <h1>, ... , <h6>
- <blockquote>
- <form>
- <article>, <section>, ...

### Inline:

- <span>
- <a>
- <em>, <i>, <strong>, <b>
- <q>
- <abbr>
- <img>
- <input>, ...

# HTML

## Tabele

```
<table>
  <thead>          <!-- nagłówek -->
    <tr>           <!-- wiersz -->
      <th>ID</th>  <!-- komórka nagłówka -->
      <th>Name</th>
    </tr>
  </thead>          <!-- część główna -->
  <tbody>           <!-- nagłówek -->
    <tr>
      <td>1</td>  <!-- komórka -->
      <td>John</td>
    </tr>
    <tr>
      <td>2</td>
      <td>Anne</td>
    </tr>
  </tbody>           <!-- stopka -->
  <tfoot>
    <tr>
      <th colspan="2">
        2 people
      </th>
    </tr>
  </tfoot>
</table>
```

# HTML

## *Semantyka*

```
<div id="main">
  <div class="article">
    <div class="header">
      <h1>Blog post</h1>
      <p>Published: <span>21st Feb, 2015</span></p>
    </div>
    <p>...</p>
  </div>
</div>
```

# HTML

## *Accessibility (a11y)*

```
<!-- bad -->
<h1></h1>

<!-- good -->
<h1></h1>
```

- Linki używane do linkowania a przyciski do klikania
- Nie polegamy jedynie na kolorze jako nośniku informacji
- Jasno określamy etykiety w polach formularzy
- ... [http://romeo.elsevier.com/accessibility\\_checklist](http://romeo.elsevier.com/accessibility_checklist)

# HTML5 czyli...

- nowa specyfikacja znaczników
  - nowe elementy
  - nowe atrybuty do już istniejących elementów
- nowe specyfikacje interakcji z przeglądarką
  - canvas
  - drag & drop
  - offline storage
- technologie powiązane
  - file API
  - indexedDB
  - geolokacja
  - ...



# HTML5 czyli...

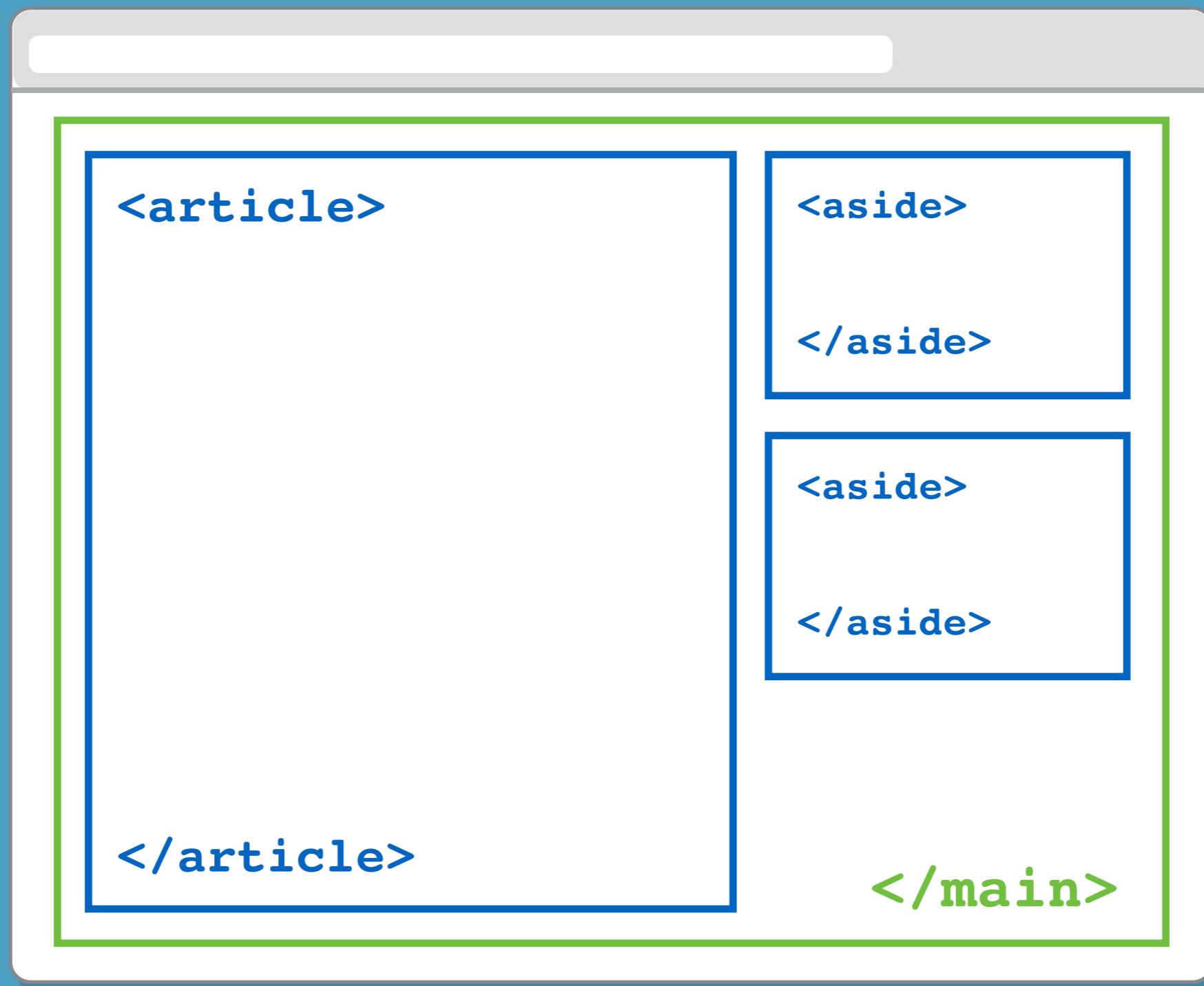
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">  
<html xmlns="http://www.w3.org/2009/xhtml">  
<head>  
    <meta http-equiv="content-type"  
          content="text/html; charset=utf-8" />
```

**HTML**

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">
```



# <article>, <aside>, <main>



# <video> i <audio>

```
<video>
  <source src="video.mp4"
    type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
  </source>
  <source src="video.3gp"
    type='video/3gpp; codecs="mp4v.20.8, samr"'>
  </source>
  <source src="video.ogv"
    type='video/ogg; codecs="theora, vorbis"'>
  </source>
  <track kind="subtitles" label="English Subtitles"
    src="en.vtt" srclang="en">
</video>
```

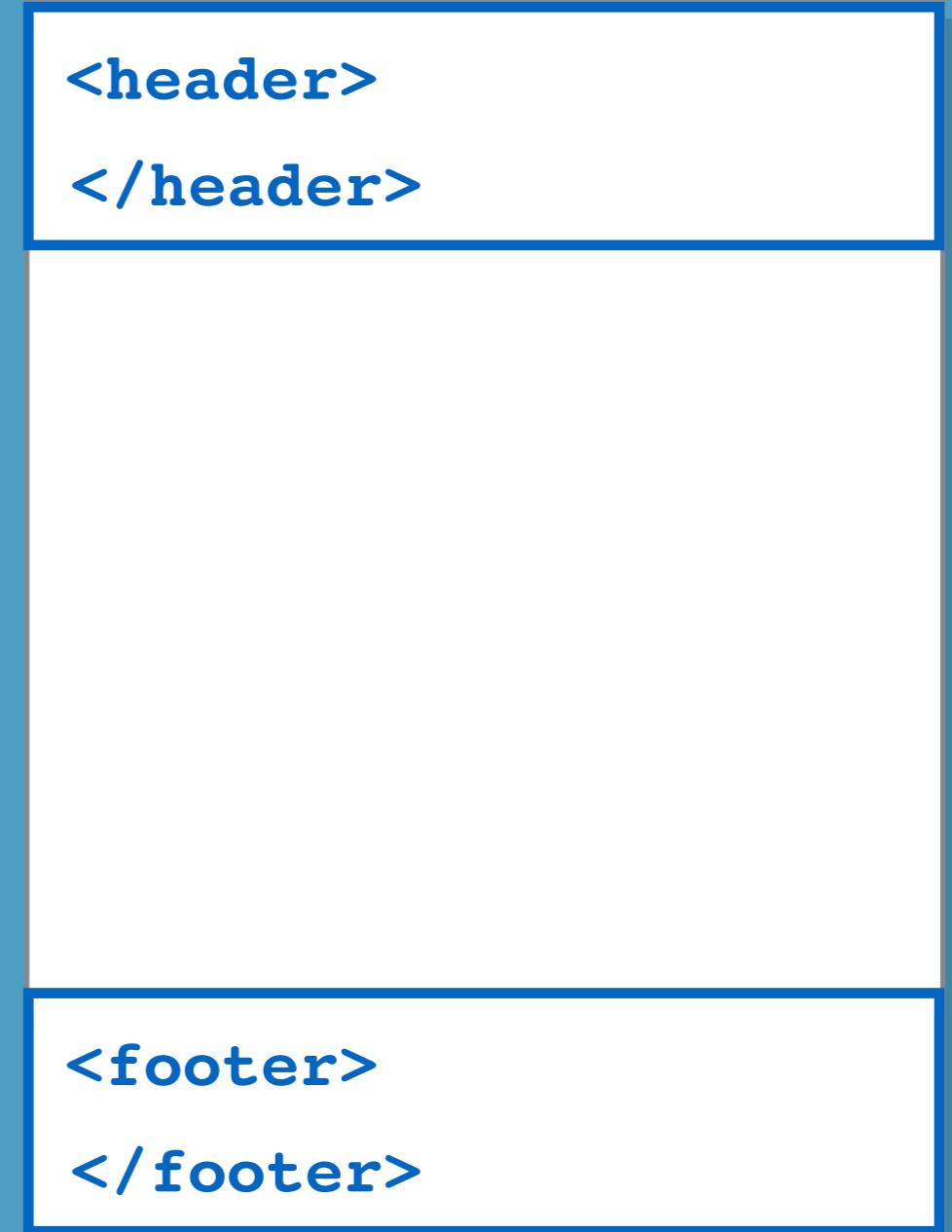
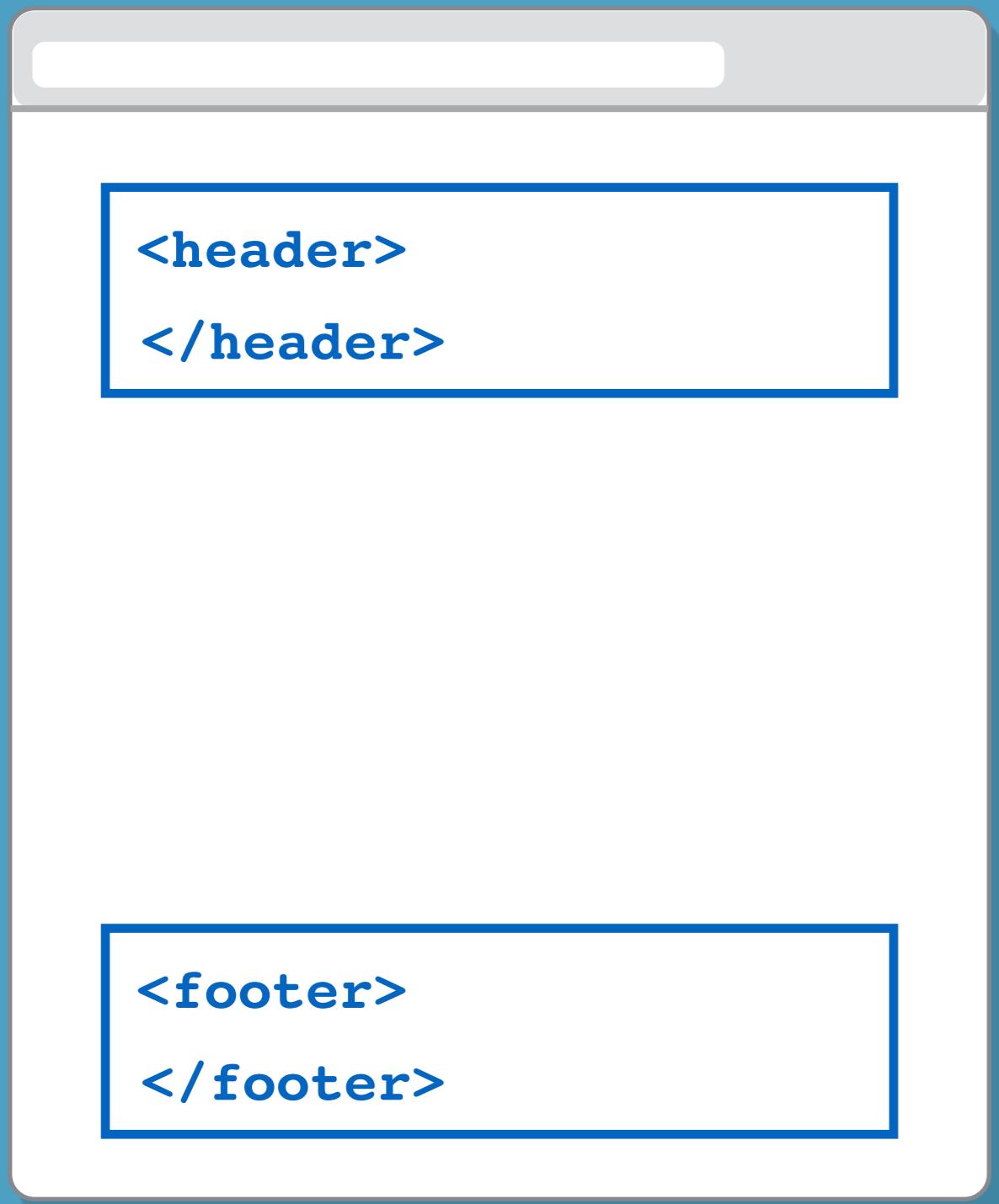
# <figure>



*A cheeky macaque, Lower Kintaganban River, Borneo. Original by Richard Clark*

```
<figure>
  
  <figcaption>A cheeky macaque, Lower
  Kintaganban River, Borneo. Original by <a
  href="http://www.flickr.com/photos/rclark/">
  Richard Clark</a></figcaption>
</figure>
```

# <header> i <footer>



# <nav>

```
<nav>
```

```
</nav>
```

```
<nav>
```

```
</nav>
```

```
<nav>
```

```
</nav>
```

# HTML5: przydatne zasoby

- HTML developer guide

[https://developer.mozilla.org/en-US/docs/  
Web/Guide/HTML](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML)



- HTML Doctor

<http://html5doctor.com/>



- Dive Into HTML5

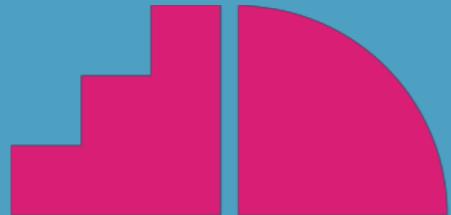
<http://diveintohtml5.info/>

- HTML5 validator

<http://html5.validator.nu/>

# *Sprawdzanie możliwości przeglądarek*

- Can I Use ... ?  
<http://caniuse.com>

-  Modernizr  
<http://modernizr.com/>

# Storage API

Zalety względem cookies:

- duże limity (  $>=5\text{MB}$  )
- nie przesyłane każdorazowo na serwer
- nie trzeba wyświetlać info o ciasteczkach ;)

Wady:

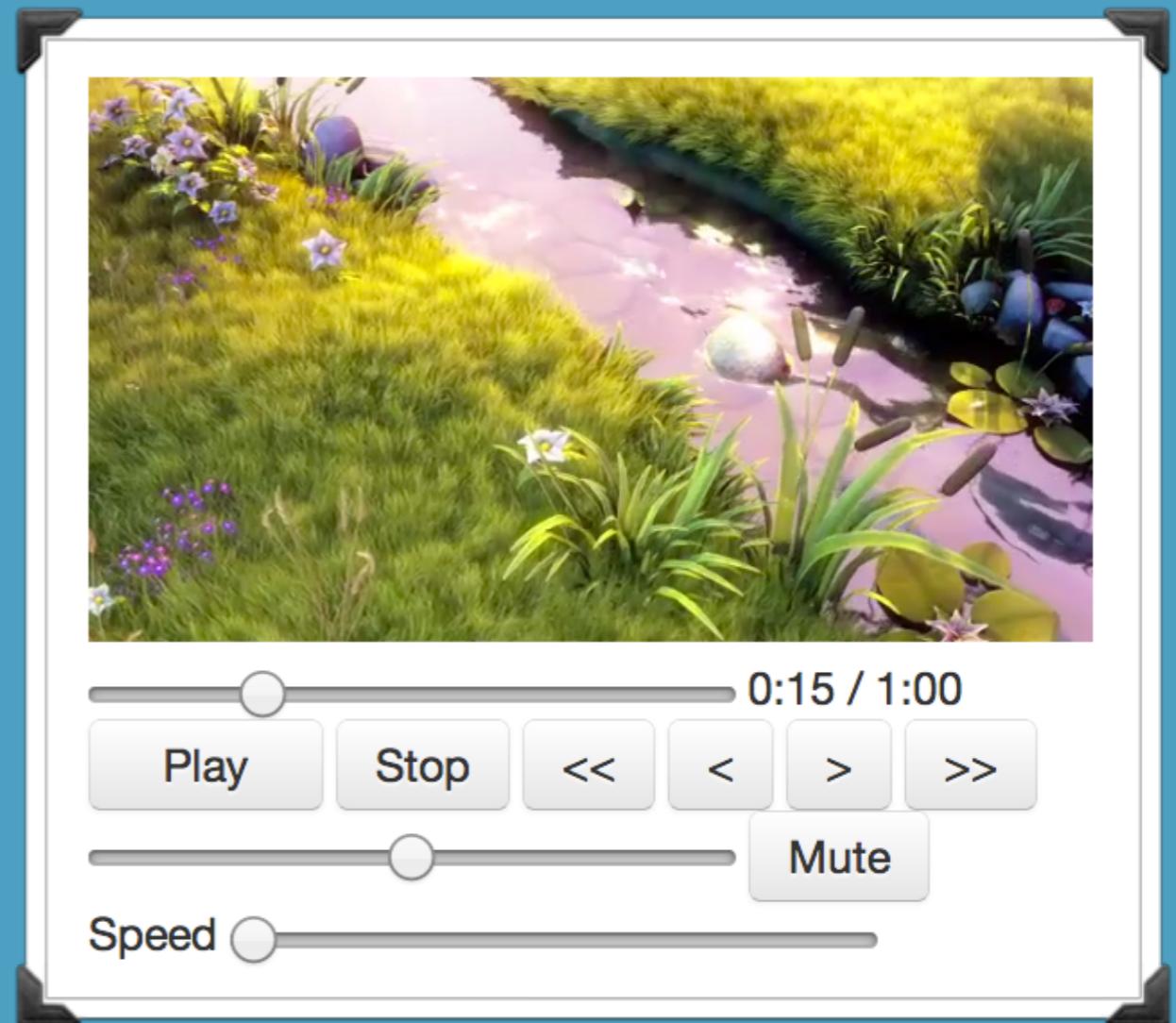
- brak mechanizmu do określania czasu ważności

Dwa rodzaje:

- `window.sessionStorage`
- `window.localStorage`

# Media API

- events:
  - paused
  - ended
- methods:
  - play()
  - pause()
  - load()
  - currentTime =
  - playbackRate =
  - volume =
  - muted =



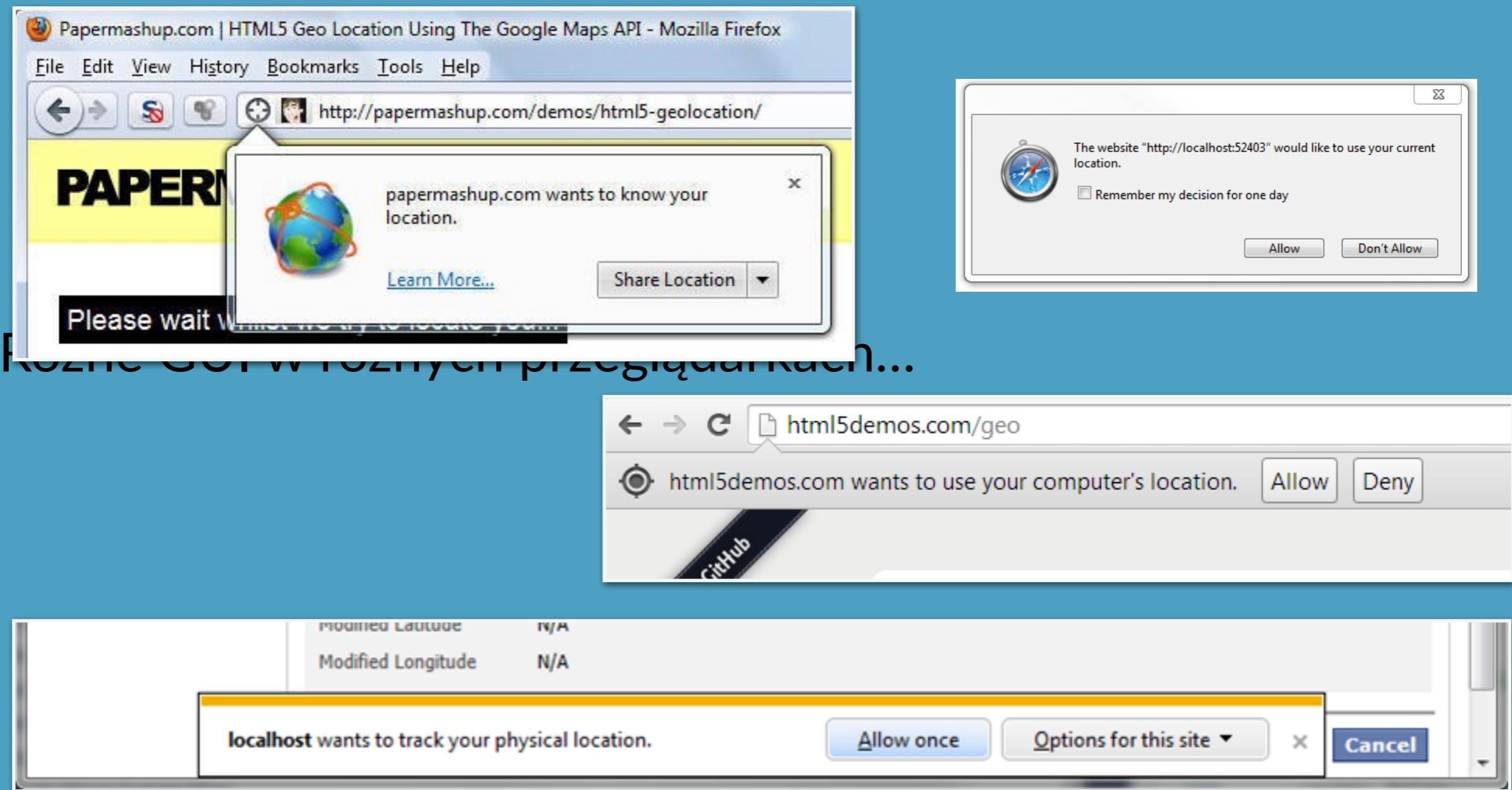
# Geolocation API

- Możliwość pobrania informacji o położeniu geograficznym urządzenia
- Za pomocą IP, GPS, WiFi lub danych sieci kom.
- Po udzieleniu zgody przez użytkownika!
- Informacje pobierane są **asynchronicznie**
- Jednorazowo lub stale

opcje:

```
{  
    enableHighAccuracy: true,  
    maximumAge          : 32000,  
    timeout             : 8000  
}
```

# Geolocation API



# HTML

## Formularze

# „Stare” typy elementu `<input>`

```
<input type="text">
<input type="radio">
<input type="checkbox">
<input type="file">
<input type="hidden">
<input type="password">
<input type="submit">
<input type="button">
```

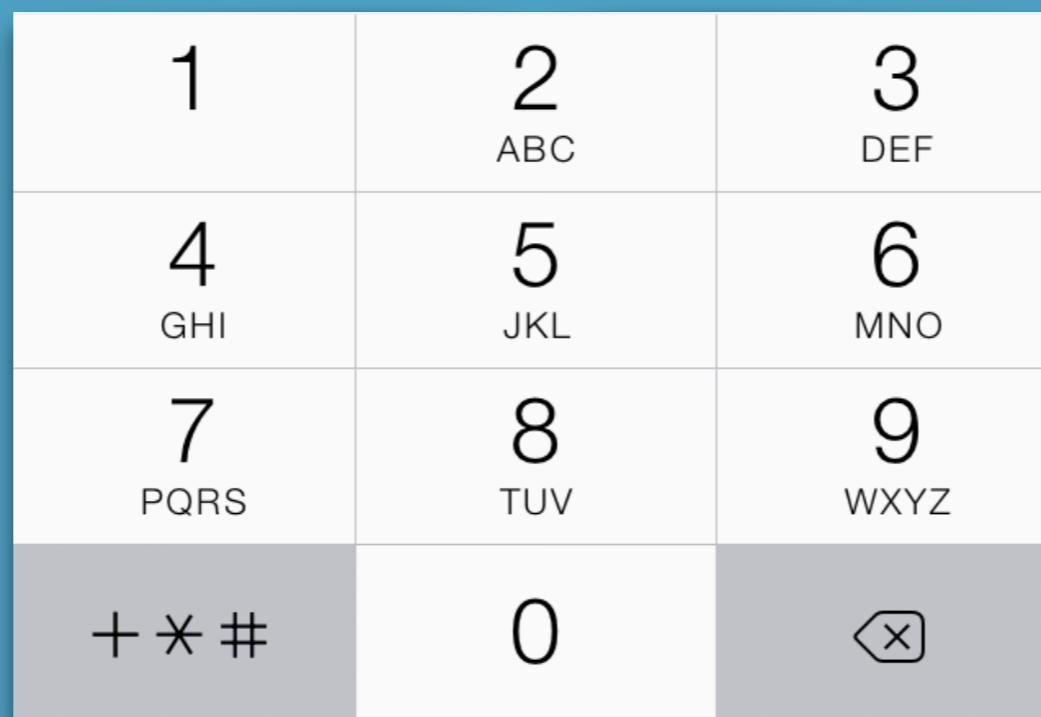
# Nowe typy elementu `<input>`

```
<input type="color">
<input type="datalist">
<input type="date">
<input type="datetime">
<input type="datetime-local">
<input type="email">
<input type="month">
<input type="number">
<input type="range">
<input type="search">
<input type="tel">
<input type="time">
<input type="url">
<input type="week">
```

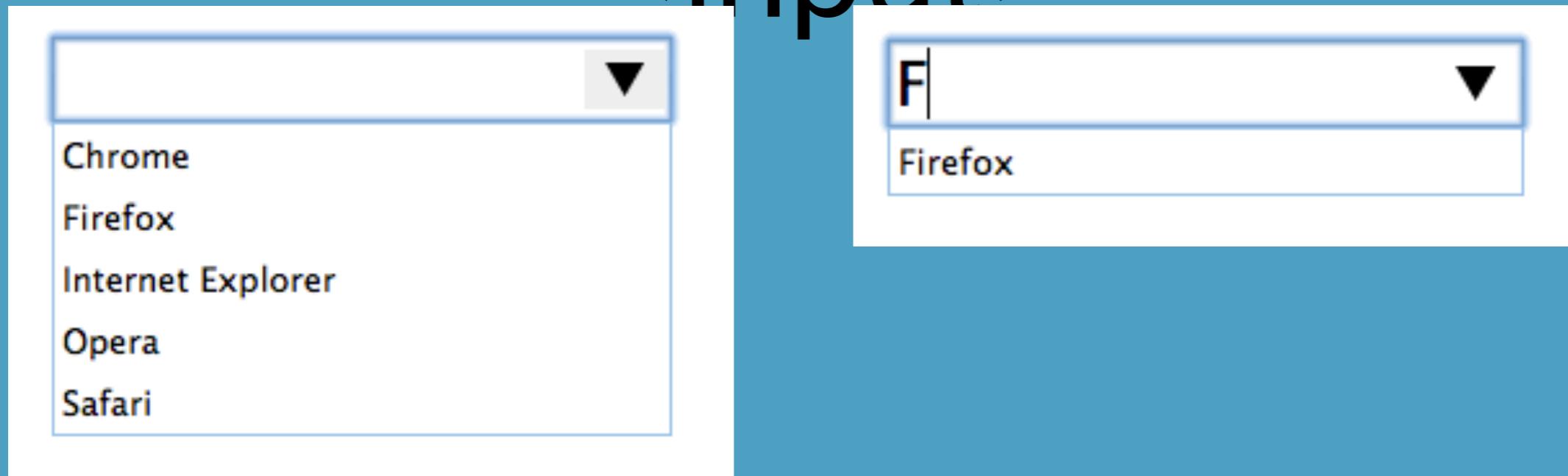
# Przykładowe typy elementu `<input>`

A screenshot of a numeric input field. The field contains the value "8,5" and has a small control button with up and down arrows to its right, indicating it's a slider input.

```
<input type="number" min="5" max="18" step="0.5"  
      value="8.5" name="shoe-size" >
```



# Przykładowe typy elementu `<input>`



```
<input list="browsers" />
<datalist id="browsers">
    <option value="Chrome">
    <option value="Firefox">
    <option value="Internet Explorer">
    <option value="Opera">
    <option value="Safari">
</datalist>
```

# Inne właściwości elementów formularzy

```
1 <form novalidate>
2   <input type="text" name="code"
3     placeholder="0-ABC"
4     autofocus
5     autocomplete="off"
6     pattern="[0-9]-[A-Z]{3}">
7 </form>
```

# CSS

## Podstawy

# CSS

## *Sktadnia*

```
/* Komentarz */

selektor { wlasnosc: wartosc;}
```

# Selektory podstawowe

```
1 <ul id="menu">
2   <li>Witamy!</li>
3   <li><a href="#">Start</a></li>
4   <li id="about" class="active"><a href="#">O nas</a>
5     <ul>
6       <li class="important active"><a href="#">O firmie</a></li>
7       <li><a href="#">Zarząd</a></li>
8       <li><a href="#">Partnerzy</a></li>
9     </ul>
10    </li>
11    <li class="important"><a href="#">Kontakt</a></li>
12 </ul>
```

```
1 li {text-transform: uppercase}
2 .important {font-weight: bold}
3 #about {font-style: italic}
```

# Selektory relacji

```
1 <ul id="menu">
2   <li>Witamy!</li>
3   <li><a href="#">Start</a></li>
4   <li id="about" class="active"><a href="#">O nas</a>
5     <ul>
6       <li class="important active"><a href="#">O firmie</a></li>
7       <li><a href="#">Zarząd</a></li>
8       <li><a href="#">Partnerzy</a></li>
9     </ul>
10    </li>
11    <li class="important"><a href="#">Kontakt</a></li>
12 </ul>
```

```
1 ul li {}
2 #menu > li {}
3 .important + li {}
4 .active ~ li {}
```

# Selektory atrybutów

```
1 a[href] {}
2 a[href="http://google.com"] {}
3 a[data-tags~="frameworks"] {}
4 a[lang|=en] {}
5 a[class^="btn-"] {}
6 a[class$="xs"] {}
7 a[href*="network.static"] {}
```

# CSS Specificity

## Który selektor wygra?

 <b>a</b> 1 x element selector  <b>Sith power: 0,0,1</b>	 <b>p a</b> 2 x element selectors  <b>Sith power: 0,0,2</b>	 <b>.foo</b> 1 x class selector *  <b>Sith power: 0,1,0</b>	 <b>a.foo</b> 1 x element selector 1 x class selector  <b>Sith power: 0,1,1</b>
 <b>p a.foo</b> 2 x element selectors 1 x class selector  <b>Sith power: 0,1,2</b>	 <b>.foo .bar</b> 2 x class selectors  <b>Sith power: 0,2,0</b>	 <b>p.foo a.bar</b> 2 x element selectors 2 x class selectors  <b>Sith power: 0,2,2</b>	 <b>#foo</b> 1 x id selector  <b>Sith power: 1,0,0</b>

# CSS Specificity

## Który selektor wygra? (cd)

The image features a grid of Star Wars Funko Pop! figures and a large Death Star model. The figures are arranged in four rows: the first row has two Stormtroopers and two Darth Vaders; the second row has a Stormtrooper, a Sith Mask, and a Darth Vader; the third row has three Sith Masks; the fourth row has one Kylo Ren figure. Below the first row, the selector **a#foo** is shown with a count of 1 x element selector and 1 x id selector. Below the second row, the selector **.foo a#bar** is shown with a count of 1 x element selector, 1 x class selector, and 1 x id selector. Below the third row, the selector **.foo .foo #foo** is shown with a count of 2 x class selectors and 1 x id selector. Below the fourth row, the selector **style** is shown with a count of 1 x style attribute. Each selector is accompanied by a button indicating its 'Sith power': **Sith power: 1,0,1** for a#foo, **Sith power: 1,1,1** for .foo a#bar, **Sith power: 1,2,0** for .foo .foo #foo, and **Sith power: 1,0,0,0** for style. In the bottom left corner, three more figures (two Sith Masks and one Darth Vader) are shown with the note **\* Same specificity**, explaining that class selector = attribute attribute = pseudo-classes. In the bottom center, a button labeled **!important** is positioned below the Death Star model.

**a#foo**  
1 x element selector  
1 x id selector

**.foo a#bar**  
1 x element selector  
1 x class selector  
1 x id selector

**.foo .foo #foo**  
2 x class selectors  
1 x id selector

**style**  
1 x style attribute

**Sith power: 1,0,1**

**Sith power: 1,1,1**

**Sith power: 1,2,0**

**Sith power: 1,0,0,0**

**\* Same specificity**  
class selector =  
attribute attribute =  
pseudo-classes

**!important**

# Pseudo-elementy

```
1 ::after {}
2 .ok::before {
3   content: "✓ " attr(data-reason);
4   display: inline-block;
5   position: absolute;
6   left: -20px;
7   width: 20px;
8   text-align: center;
9 }
10 ::first-letter {}
11 ::first-line {}
12 ::selection {}
```

::before i ::after

- działają dla **niepustych elementów**
- wymagają atrybutu **content**

# Pseudo-klasy

:active, :focus, :hover

:checked

:disabled, :enabled

:empty

:first-child

:last-child, :nth-child(), :nth-last-child()

:first-of-type, :last-of-type,

:nth-last-of-type(odd), :nth-of-type(2n+1)

:invalid, :valid

:lang()

:left, :right

:not( *\*\*simple selector\*\**)

:only-child, :only-of-type

:optional, :required

:target

# Kolory

```
1 {  
2     color: white;  
3     color: #fff;  
4     color: #FFFFFF;  
  
5     color: rgb(255, 255, 255);  
6     color: rgba(255, 255, 255, 1);  
7     color: rgba(100%, 100%, 100%, 1);  
8     color: hsl(0, 100%, 100%);  
9     color: hsla(0, 100%, 100%, 1);  
10    color: transparent;  
11    text-shadow: 0 0 currentColor;  
12 }
```

# transform

```
1 {  
2   transform:  
3     rotate(45deg)  
4     translate(200px, 0)  
5     skew(0, 0)  
6     scale(1);  
7   transform-origin: right bottom;  
8   transform-origin: 200px 100%;  
9 }
```

# transition

```
1 {  
2   transition-property: all;  
3   transition-duration: 2000ms;  
4   transition-timing-function: ease | ease-in |  
5     linear | steps(4, end);  
6   transition-delay: 2s;  
7   transition: transform 1s ease-in-out 2s;  
8 }
```

co podlega transition?

# animacje

```
1 @keyframes animacja {  
2   from {  
3     left: 0;  
4   }  
5   to {  
6     left: 100%;  
7   }  
8 }
```

VS

```
1 @keyframes animacja {  
2   0% {  
3     left: 0;  
4   }  
5   100% {  
6     left: 100%;  
7   }  
8 }
```

```
1 .dynamic {  
2   animation-name: animacja;  
3   animation-duration: 2s;  
4   animation-timing-function: ease-in-out;  
5   animation-iteration-count: 4; // lub infinite  
6   animation-delay: 1s;  
7   animation-direction: alternate; // normal | reverse  
8 }
```

# **Responsive** Web Design

# RWD: definicje

**Fixed** - Szerokość strony jest stała bez względu na rozmiar okna przeglądarki czy rodzaj urządzenia.

**Fluid** - Szerokości elementów na stronie określone są za pomocą wartości procentowych. Kolumny utrzymują rozmiary proporcjonalne do siebie i okna przeglądarki i płynnie zmieniają szerokość.

**Adaptive** - Strona posiada media queries, które definiują określone szerokości elementów dla poszczególnych rozmiarów urządzeń.

**Responsive** - strona jest zbudowana na płynnej siatce i używa media queries do określania zmian w layoucie w zależności od szerokości okna przeglądarki.

# RWD: proces

## Mobile first



# RWD: proces

## *Definiowanie breakpoints („wartości granicznych”)*

~~Standardowe wartości to:~~

- ▶ 320px i < 80px („iPhone”)
- ▶ 768px i 1024px („iPad”)
- ▶ powyżej 1024px („desktop”)

Bardziej uniwersalna metoda:

1. Projektujemy najmniejszy ekran
2. Rozszerzamy okno, póki design wygląda dobrze
3. W momencie, w którym przestaje - ustawiamy breakpoint

# RWD: meta viewport

```
<meta name="viewport" content="width=device-width,initial-scale=1">
```



Let's pretend  
I'm 960px, ok?

No, I need your  
actual width.

# RWD: media queries - zapis

Media Queries (2.1) używamy od lat:

```
1 <link rel="stylesheet" href="css/styles.css"
2           media="screen" type="text/css">
3 <link rel="stylesheet" href="css/print.css"
4           media="print" type="text/css">
```

```
1 @media screen {
2     p {color: blue;}
3 }
4
5 @media print {
6     p {color: red;}
7 }
```

# RWD: media queries - zapis

Którego zapisu używać?

```
@media (max-width:632px) {}
```

```
@media screen and (max-width:632px) {}
```

```
@media only screen and (max-width:632px) {}
```

# RWD: media queries - dostępne warunki

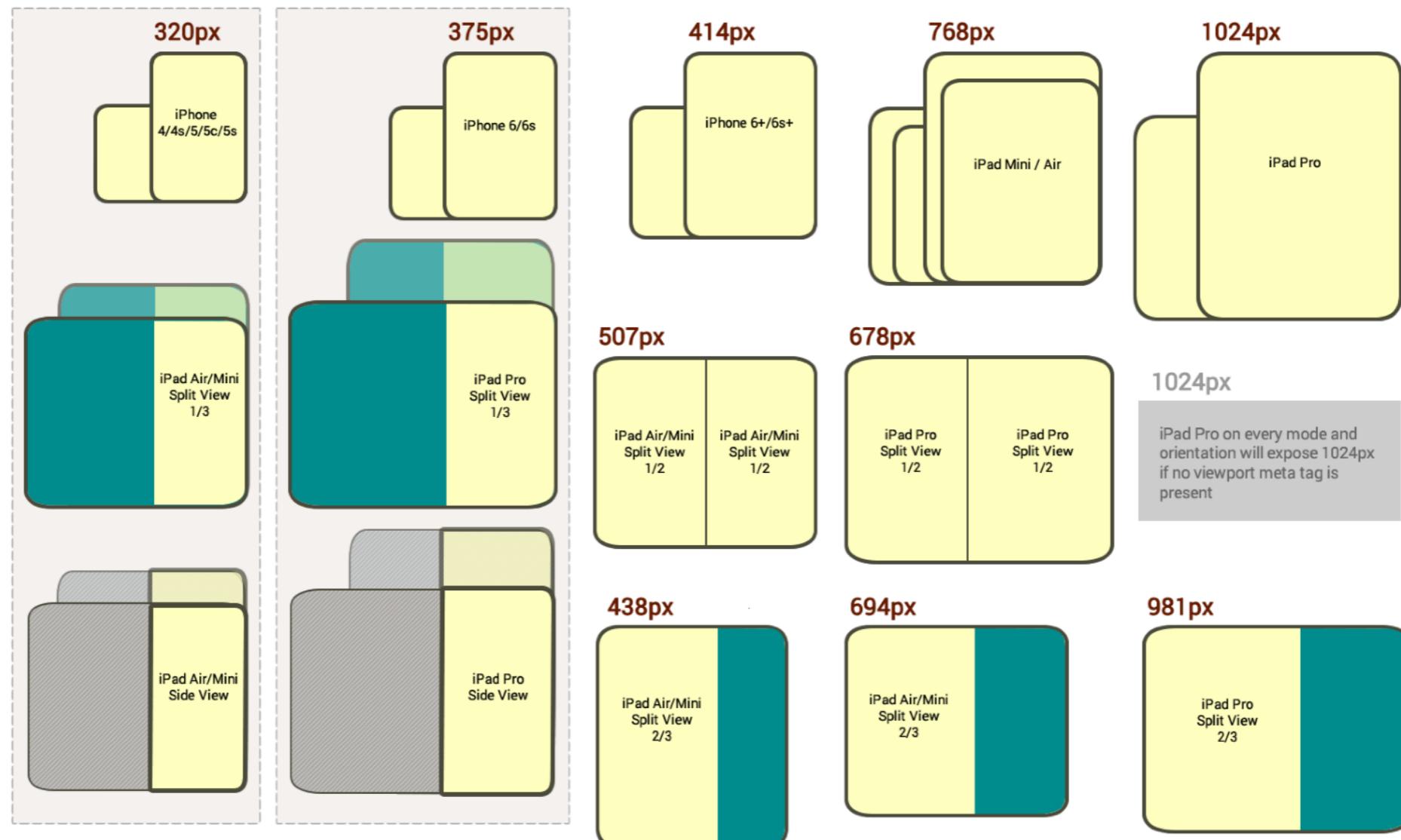
## CSS2.1

1. all
2. aural
3. handheld
4. braille
5. embossed
- 6. print**
7. projection
- 8. screen**
9. tty
10. tv

## CSS3

1. **width** ←
2. **height** ?
3. **device-width** ←
4. **device-height**
5. **orientation**
6. **aspect-ratio**
7. **device-aspect-ratio**
8. color
9. color-index
10. monochrome
11. resolution
12. scan
13. grid

# RWD: media queries - width vs device-width



*Szerokość widoku w CSS  
przy width=device-width  
w urządzeniach z iOS 9 AD 2015*

# RWD: img

podstawa:

```
img {max-width:100%;}
```

lepiej:

```
img {max-width:100%;  
height:auto;}
```

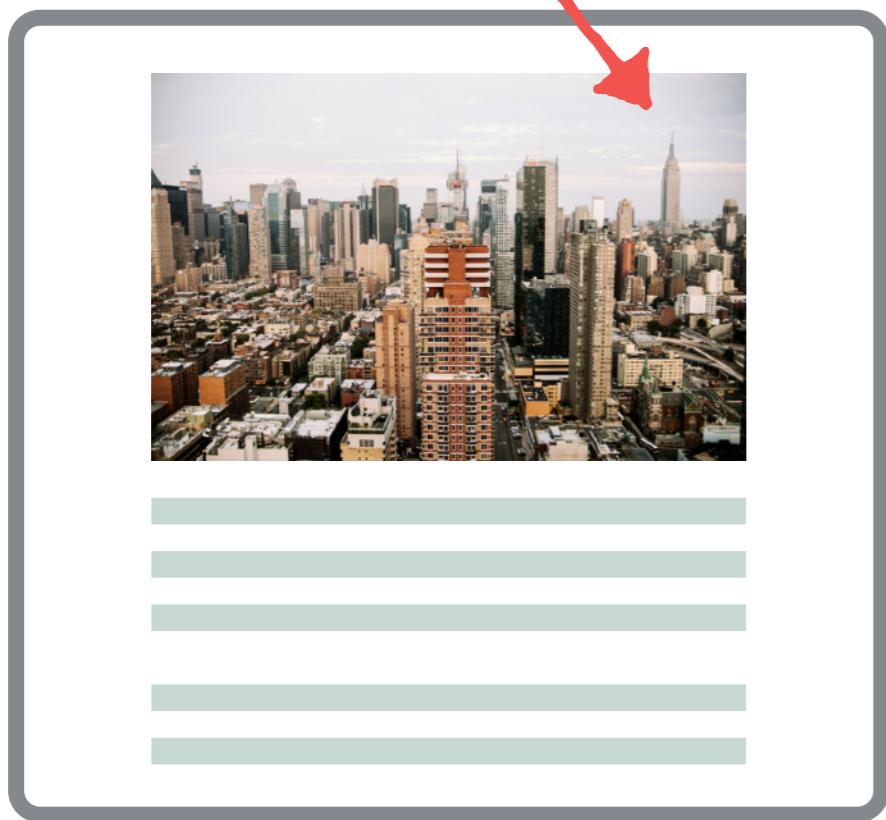


Screenshot of a browser developer tools Elements tab showing the HTML code for an image element. The width and height attributes are highlighted with a red oval.

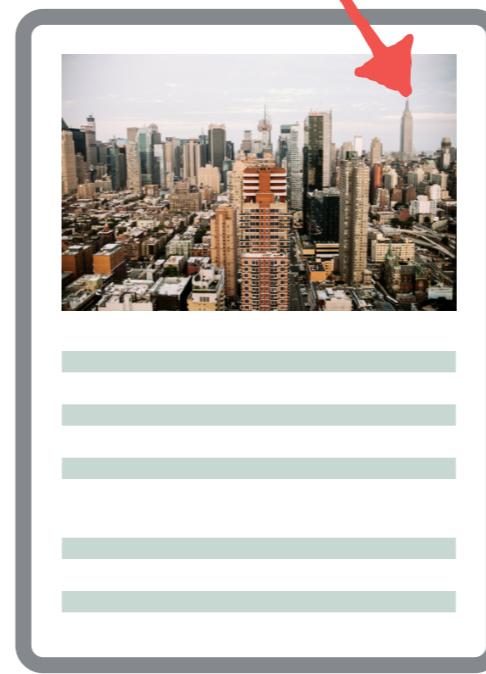
```
<img alt="A man in a suit holding a glass." width="800" height="600" style="max-width: 100%;">
```

# RWD: img

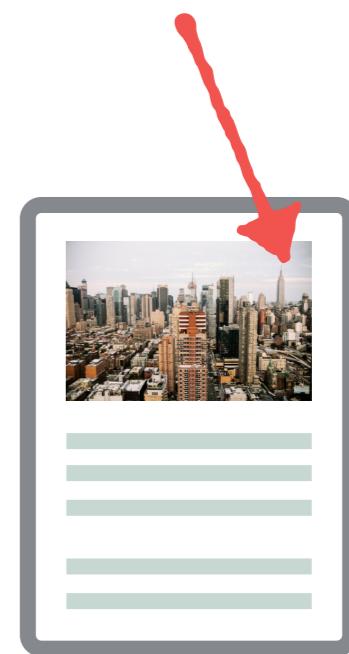
city.png  
[337kB]



city.png  
[337kB]

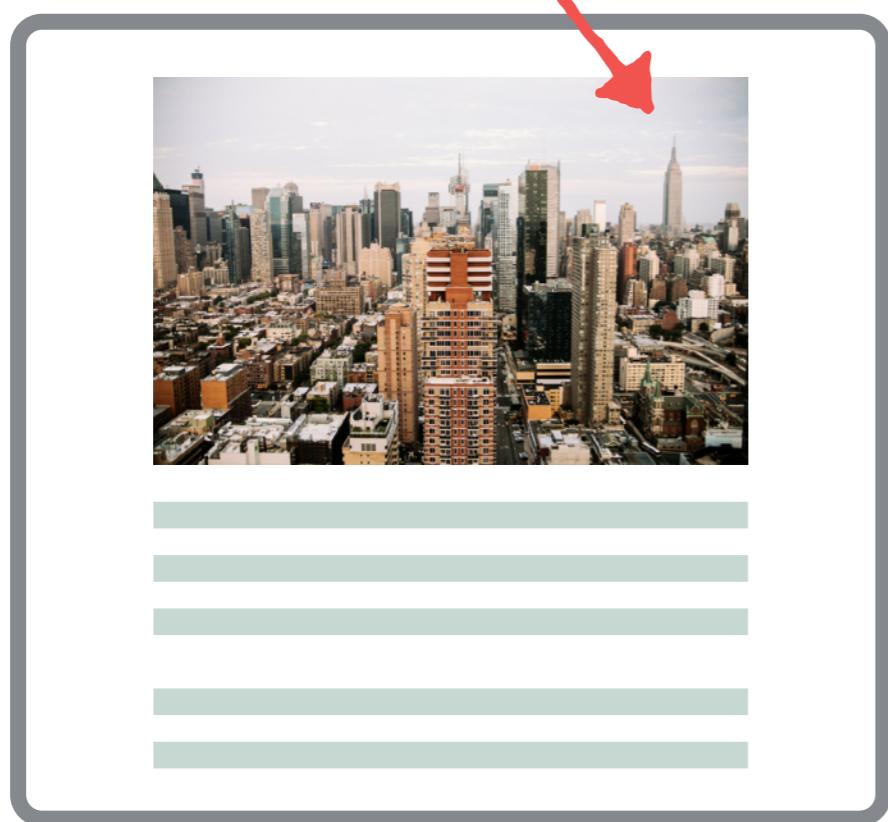


city.png  
[337kB]

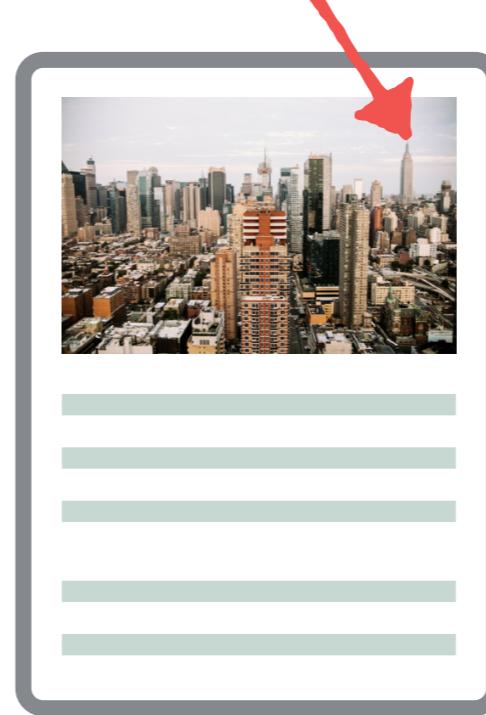


# RWD: img

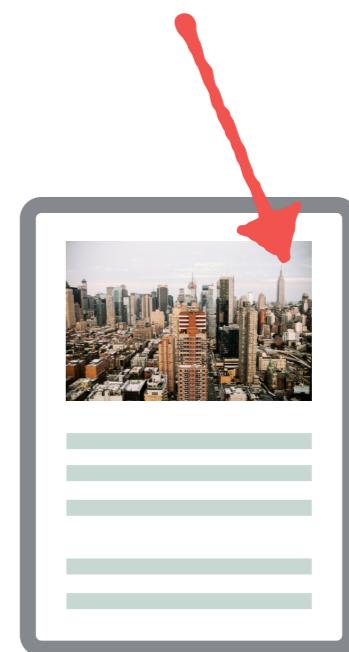
city-lg.png  
[337kB]



city-md.png  
[188kB]



city-sm.png  
[90kB]



# RWD: `img srcset`

## `srcset`

Lista adresów URL wskazujących na pliki, które mogą być wykorzystane przez przeglądarkę.

```

```

# RWD: `img srcset`

## `w (Width Descriptor)`

Opcjonalny opis wskazujący na szerokość każdej z grafik w `srcset`.

```

```

# RWD: `img srcset`

## `x` (*Pixel Density Descriptor*)

Opcjonalny opis wskazujący na docelową gęstość pikseli dla każdej z grafik w `srcset`.

```

```

# RWD: picture

## <picture>

Określa warunki, dla których mają zostać załadowane odpowiednie grafiki.

## <picture>

```
<source srcset="city-lg.png" media="(min-width: 800px)">
<source srcset="city-md.png" media="(min-width: 600px)">

</picture>
```

# RWD: can I use „picture”?



RESPONSIVE IMAGES COMMUNITY GROUP

<https://responsiveimages.org/>

# Picturefill

A RESPONSIVE IMAGE POLYFILL

<https://scottjehl.github.io/picturefill/>

# Typografia: dlaczego to ważne?

„Web design is 95% typography”

Naprawdę?

„Typography is the foundation of web design”

Tak lepiej.

# Typografia: gdzie?

## *Popularne zastosowania tekstu:*

- nagłówki
- długie paragrafy
- cytaty, zajawki mające przykuć uwagę
- nawigacja

# Typografia: jak?

*Na co będziemy zwracać uwagę:*

- **konsistentna** reprezentacja w różnych kontekstach
- czytelna długość linii na wszystkich rozmiarach ekranów
- waga strony

# Typografia: skąd?

*Źródło fontów:*

Self-hosted:



Fonts providers:

**Google** Fonts



Adobe Typekit



# Typografia: konsekwencja = skala

The screenshot shows a web browser window for 'Modularscale' at [www.modularscale.com](http://www.modularscale.com). The interface includes a sidebar with 'Bases' (set to 1em) and 'Ratios' (set to 1,5), and tabs for 'Table' and 'Text'. The 'Text' tab is active, displaying five examples of text with increasing font sizes and line heights. Each example is preceded by a small icon and its corresponding value: '3.375em', '2.25em', '1.5em', '1em', and '0.667em'. The text examples are: 'The quick brown fox jumps over the lazy dog', 'The quick brown fox jumps over the lazy dog', 'The quick brown fox jumps over the lazy dog', 'The quick brown fox jumps over the lazy dog', and 'The quick brown fox jumps over the lazy dog'. The background features a large, bold, dark font version of the same text.

Modular Scale

Not sure how this calculator works? [Here are some tips.](#)

Bases

1em

Ratios

1,5

Web Sass JS

This site is built & maintained by [Scott Kellum](#) and [Tim Brown](#). Copyright © 2010–2015 Tim Brown.

Table Text

a 3.375em

The quick brown fox jumps over the lazy dog

a 2.25em

The quick brown fox jumps over the lazy dog

a 1.5em

The quick brown fox jumps over the lazy dog

a 1em

The quick brown fox jumps over the lazy dog

a 0.667em

The quick brown fox jumps over the lazy dog

a 0.444em

Budowa skali typograficznej

# Typografia: jednostki

<b>px</b>	<b>em</b>	<b>rem</b>
wartości absolutne	wartości relatywne	wartości relatywne
 łatwe do przeniesienia z designu	liczone względem kontekstu	liczone względem globalnego kontekstu (html)
„najbardziej precyzyjne”	łatwe do globalnej modyfikacji	łatwe do globalnej modyfikacji
 problemy z dostępnością w IE	problemy ze zmianą kontekstu	wsparcie przez IE >8 (pełne przez IE >10)
mało elastyczne	mniej „przenośne”	

# Typografia: jednostki

## **vw / vh / vmin / vmax**



liczone względem wymiarów okna

„bardzo responsywne”



wsparcie przez IE >8  
(oprócz vmax, z jakiegoś powodu)

zwykle wymagają dookreślenia  
dla mocno różniących się rozmiarów ekranów

# Typografia: parametry wpływające na czytelność

typografia

css

stopień pisma

size

leading

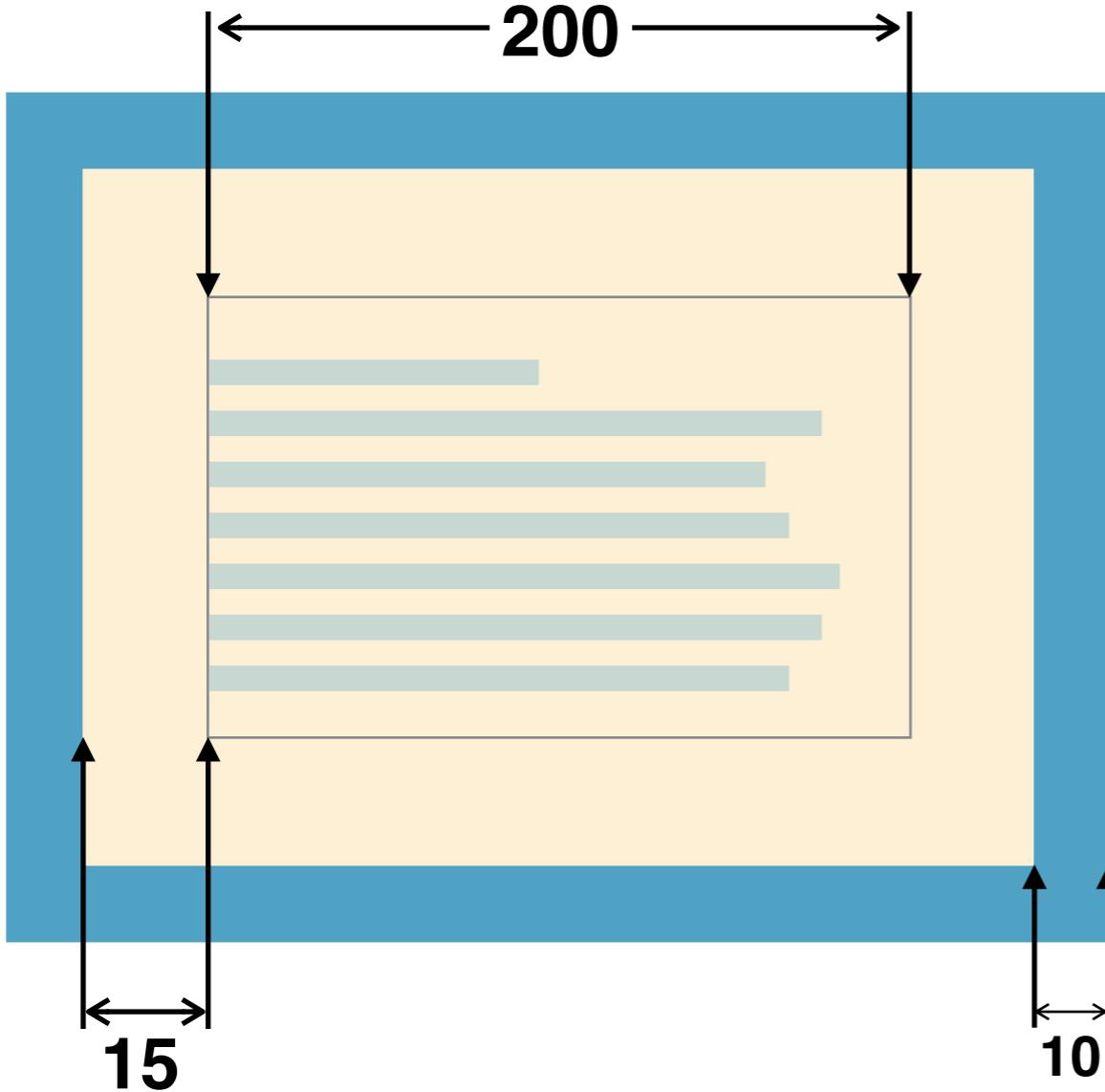
line-height

measure

width

- ▶ kombinacja tych trzech parametrów wpływa na wygodę czytania
- ▶ optymalna szerokość linii to 45-75 znaków
- ▶ *text-align: left* również pomaga

# CSS: box model



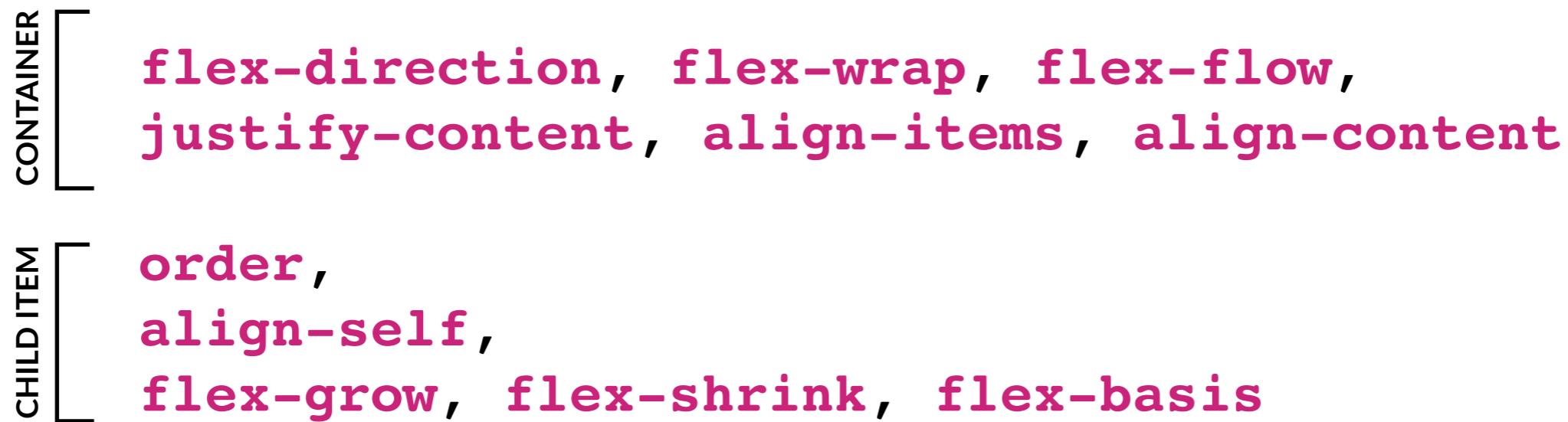
*„Ile miejsca zajmuje ten blok?”*

```
.box {  
    width: 200px;  
    margin: 15px;  
    border-width: 15px;  
}
```

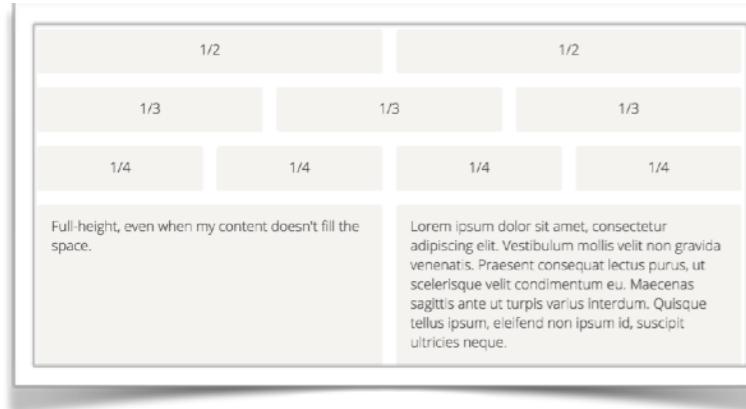
# CSS: flexbox

*Flexible Box Layout* to nowy tryb definiowania układu elementów w CSS 3, który pozwala na uzyskanie zaawansowanych layoutów aplikacji internetowych.

Dostępne właściwości:



# CSS: flexbox - rozwiązania



Grid system

The Holy Grail Layout is a classic CSS problem with various solutions presented over time. If you're unfamiliar with the history of the Holy Grail layout, this [A List Apart article](#) offers a pretty good summary and links to a few of the more well-known solutions.

At its core, the Holy Grail Layout is a page with a header, footer, and three columns. The center column contains the main content, and the left and right columns contain supplemental content like ads or navigation.

Most CSS solutions to this problem aim to meet a few goals:

„Holy Grail Layout”

Add-on Prepended      Add-on Appended

Amount      Go

Q      ★

Appended and Prepended Add-ons

Example One      Send      Example One      Encrypt

Input addons

Standard Media Object

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur ac nisl quis massa vulputate adipiscing. Vivamus sit amet risus ligula. Nunc eu pulvinar augue.

Standard Media Object

Donec imperdiet sem leo, id rutrum risus aliquam vitae. Cras tincidunt porta mauris, vel feugiat mauris accumsan eget.

Media Object Reversed

Phasellus vel felis purus. Aliquam consequat pellentesque dui, non mollis erat dictum sit amet. Curabitur non quam dictum, consectetur arcu in, vehicula justo. Donec tortor massa, eleifend nec viverra in, aliquet at eros. Mauris laoreet condimentum mauris, non tempor massa fermentum ut. Integer gravida pharetra cursus. Nunc in suscipit nunc.

Media element

Sticky Footer

Click the button below to hide the contents of this page. Notice how the footer sticks to the bottom of the window even when there's not enough content to fill the page.

Star 6,671      Tweet      @philwalton

Hosted on GitHub by [Philip Walton](#). Source code and examples released under the [MIT license](#). Website and documentation licensed under the [Creative Commons Attribution license](#).

Sticky footer

Top

Centered

Bottom

Centering!

# Wzorce RWD: nawigacja

*Popularne wzorce resonsywnej nawigacji:*

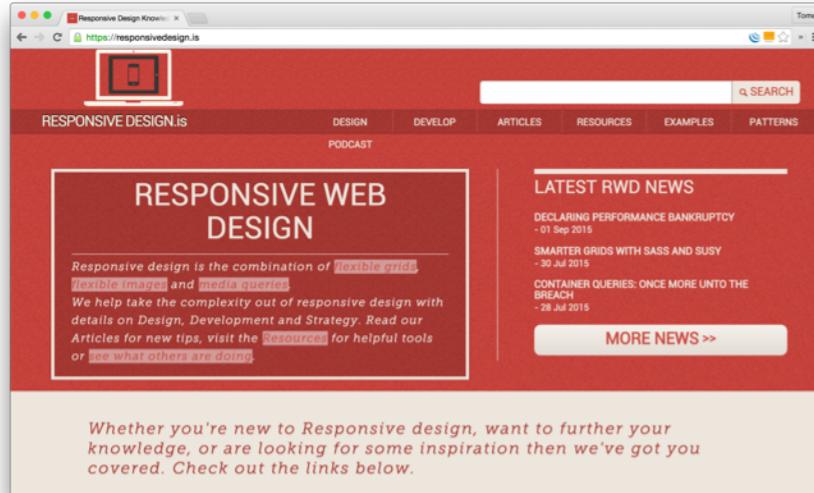
- ▶ zwężane menu („bez zmian”)
- ▶ menu w stopce
- ▶ menu <select>
- ▶ dropdown toggle
- ▶ off-canvas
- ▶ pull-down

# Wzorce RWD: nawigacja

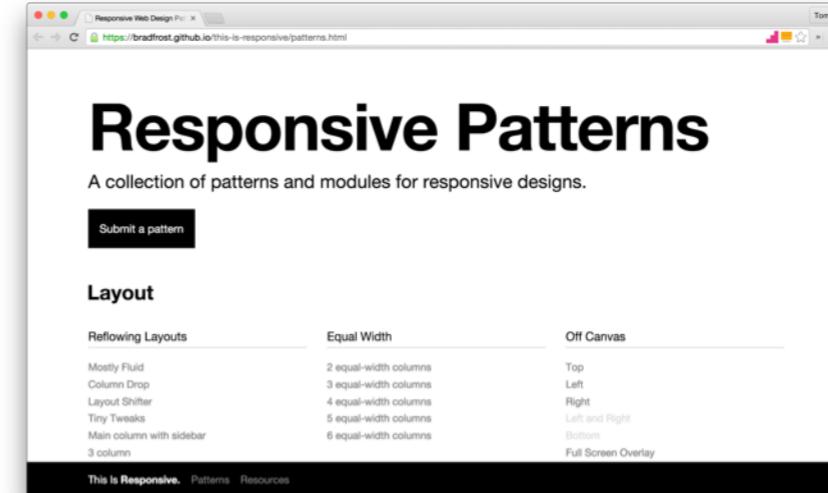
*Rozwiązania responsywnej zaawansowanej nawigacji:*

- ▶ multi-toggle („zagnieżdżony akordeon”)
- ▶ „prawy do lewego”
- ▶ bez podmenu
- ▶ „priority+”

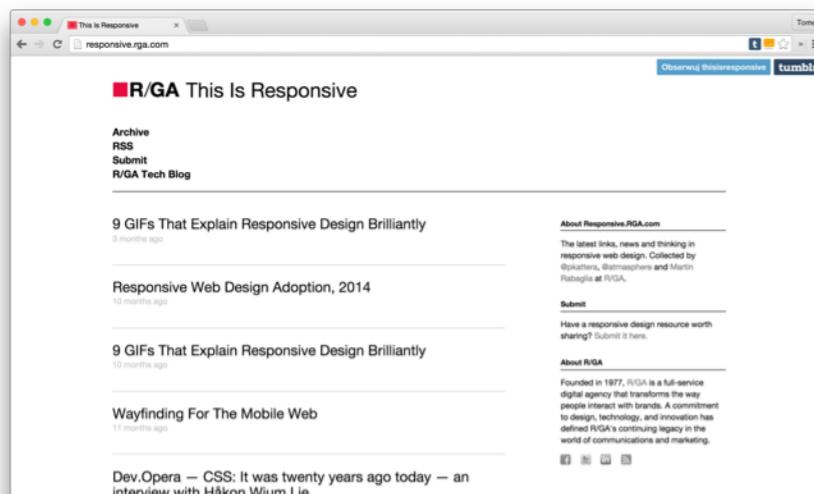
# RWD: przydatne zasoby



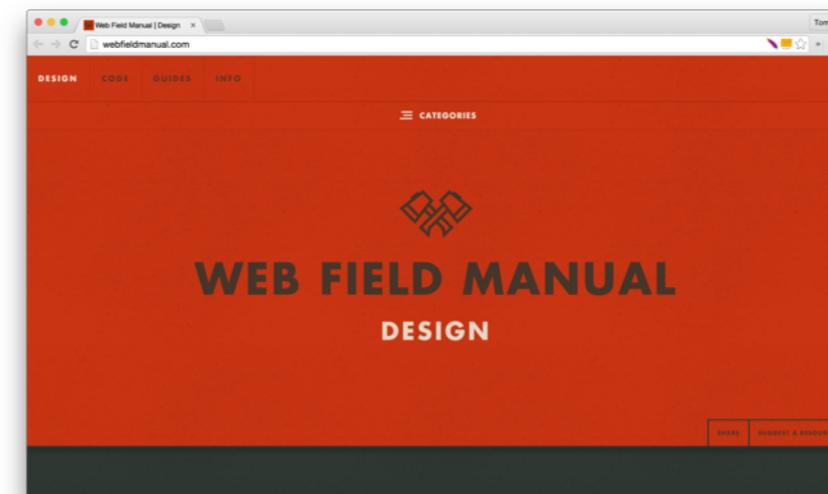
<https://responsivedesign.is/>



<https://bradfrost.github.io/this-is-responsive/patterns.html>



<http://responsive.rga.com/>



<http://webfieldmanual.com/>

# CSS Frameworks

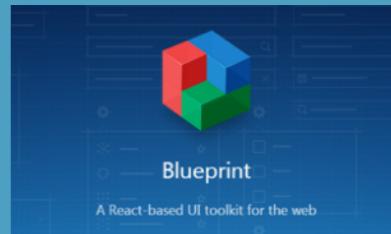
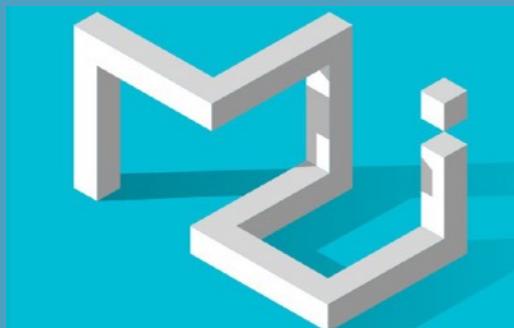
## *typowe problemy - rozwiążane*

The image displays a collection of user interface (UI) elements and navigation patterns, likely from a CSS framework's documentation or design system. The components include:

- Buttons: Submit (grey), Send (blue), Create (green), Delete (red).
- Form fields: Placeholder... (text input), valid@email.com (text input with green checkmark), invalid email (text input with red X).
- Checkboxes and Radio buttons: Various states of checked/unchecked checkboxes and radio buttons.
- Switches: OFF/ON toggle switches.
- Toolbars: Reload (purple), Undo (orange), Review (blue), Accept (dark grey).
- Navigation: Settings, Profile (highlighted in blue), Apps.
- Progress bars: A blue progress bar and a green progress bar.
- Dropdown menus: Dropdown (with arrow), Cut, Copy, Paste.
- Search bar: Search... (placeholder).
- Breadcrumbs: Dashboard > Projects > App > Design.
- Alerts: Done! You successfully completed this project. (green background), Oops! Something is technically wrong. (yellow background).
- Table: A table with columns #, Title, Status, and Views. Data rows:
  - #1: Title: Lorem ipsum dolor sit amet, Status: Published, Views: 4215
  - #2: Title: Consectetur adipiscing elit, Status: Draft, Views: -
  - #3: Title: Integer molestie lorem at massa, Status: Published, Views: 2316

# CSS Frameworks

*typowe problemy - rozwiążane*



# JavaScript

# JavaScript frameworks

*nie wymyślajmy kota od nowa*

- Rozwiążanie podstawowych, typowych problemów
- Zapewnienie przewidywalnej struktury\*
- I konwencji\*
- Abstrakcja nad trudniejszymi problemami
- Własne ekosystemy dodatków

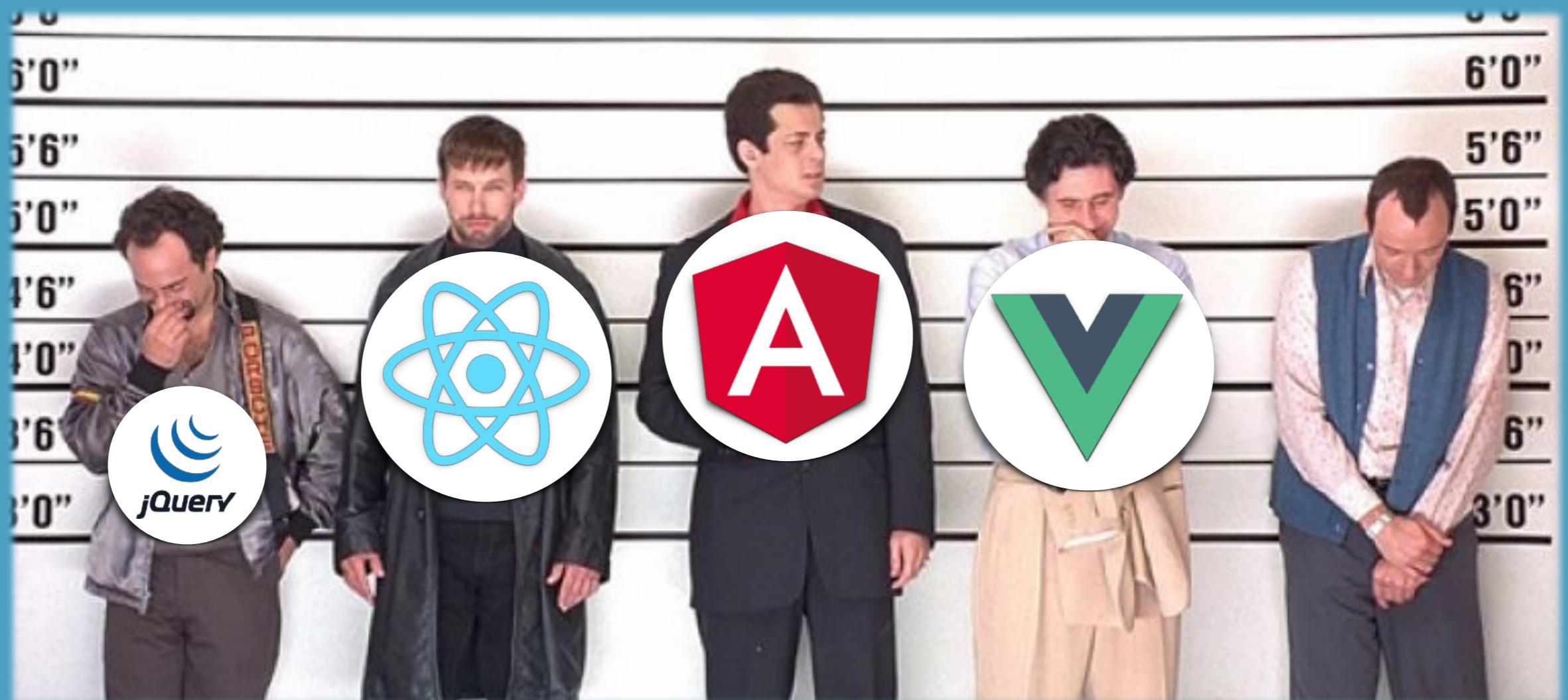
# JavaScript frameworks

Aktualnie najpopularniejsze:

*React*

*Angular*

*Vue.js*



# Linki

*przykłady i inspiracje:*

- <https://stuffandnonsense.co.uk/>
- <https://boagworld.com/>
- <https://clearleft.com/casestudies>
- <https://www.awwwards.com/>

*narzędzia:*

- <https://codesandbox.io>
- <https://stackblitz.com/>
- <https://codepen.io/>
- <https://webflow.com/>

*html, css:*

- <https://marksheet.io/>
- <https://bradfrost.github.io/this-is-responsive/patterns.html>
- <https://philipwalton.github.io/solved-by-flexbox/>



- <https://www.invisionapp.com/>
- <https://www.figma.com/>
- <https://www.sketchapp.com/>
- <https://zeplin.io/>
- <https://twitter.com/sulco>
- <https://medium.com/@tomsu>