Modern Web Development

Jeroen Zwartepoorte

- 10+ years at Ciber
- Principal Web Consultant @ SL Microsoft/Mobile
- Java background; Ruby on Rails
- Front-end only since ~3 years
- Front-end architect & lead developer @ Malmberg

Introduce yourselves

Modern Web Development

Tips, tricks and best practices

- State of web development
- Techniques:
  - Javascript
  - CSS
  - HTML
  - Tooling

State of web development

Browsers

- Apple: (Mobile) Safari
- Google: Chrome (Desktop & Android)
- Microsoft: (IE10,) IE11 and Edge
- Mozilla: Firefox (Desktop & Android)

Safari

- OS-X only for desktop, iOS for mobile
- Best mobile browser today
- Least open development process
- Major releases coupled with OS-X/iOS
- Remote debugging of web apps running on iOS

IE10 & IE11

- Based on Trident engine; incremental development since IE6
- Supports most modern W3C standards (partially)
- Windows Vista and newer
- Requires hacks/workarounds due to bugs/incomplete implementations

Edge

- Greenfield browser; Windows 10 only
- Completely new engine; IE11 ships with Win10 for “enterprise” sites
- Same level as Safari, Chrome and Firefox
- Continuous open development; auto update

Chrome

- Best developer tools (extensions)
- Continuous open development; auto update
- Various channels: Canary, Dev, Beta and Release
- On iOS a wrapper around Safari
- Support only Chrome on Android

Firefox

- Developer Edition
- Similar developer tooling to Chrome
- Similar release channels as Chrome; also auto update
- On iOS a wrapper around Safari
- Personal preference

RESTful architecture

JSON

- Communication via JSON
- REST API (GET, POST, PUT & DELETE)
- Do not use a full-stack framework; separate presentation from business logic

Tooling

- Node.js & NPM for front-end
- Git version control (Github)
- Developer tools with framework specific extensions
- modern.ie VMs for IE10, IE11 and Edge browser testing

Techniques

Javascript, CSS, HTML and Tooling

ES6 aka ES2015

- ECMAScript is Javascript
- Major new features especially useful for large applications
- Renamed to ES2015 to indicate yearly releases
- Requires transpilers for now

```
class Person {
    constructor(name, age) {
        this.name = name;
        this.age = age;
    }

    get description() {
        return `${this.name} is ${this.age} years old`;
    }
}

let person = new Person('Jeroen', 38);
console.log(person.description);
```

```
[1, 2, 3].map(function (num) { return num * 2 })
// <- [2, 4, 6]
```

```
[1, 2, 3].map(num => num * 2)
// <- [2, 4, 6]
```

```
function Timer() {
    this.seconds = 0;

    var that = this;
    setInterval(function () { that.seconds++ }, 1000);
}

class Timer () {
    constructor() {
        this.seconds = 0;
        setInterval(() => this.seconds++, 1000);
    }
}

let timer = new Timer();
setTimeout(() => console.log(timer.seconds), 3100);
```

```
function doSomethingAsync() {
    return new Promise((resolve, reject) => {
        fetch('/api.json')
            .then(response => {
                resolve(response.json());
            })
            .catch(err => {
                reject(err);
            });
    });
}

doSomethingAsync().then(data => {
    console.log(data);
});
```

```
// foo.js
export class Foo {
    static bar() {
        console.log('bar!');
    }
}
```

```
// bar.js
import { Foo } from './foo';

Foo.bar();
```

And much much more…

https://ponyfoo.com/articles/es6

Transpilers

- No browser that runs ES6 100% (Chrome dev at 93%)
- Module loader is not part of ES6 (http://whatwg.github.io/loader/)
- Solution: polyfills & transpilers!

“Use next generation JavaScript, today.”

—Babel (babeljs.io)

```
let arr = [1, 2, 3];
arr.map(num => num * 2);

class Person {
  constructor(name) {
    this.name = name;
  }

  sayHello() {
    return `Hello ${this.name}`;
  }
}
```

```
"use strict";

function _classCallCheck(instance, Constructor) { if (!(instance instanceof
Constructor)) { throw new TypeError("Cannot call a class as a function"); } }

var arr = [1, 2, 3];
arr.map(function (num) {
  return num * 2;
});

var Person = (function () {
  function Person(name) {
    _classCallCheck(this, Person);

    this.name = name;
  }

  Person.prototype.sayHello = function sayHello() {
    return "Hello " + this.name;
  };

  return Person;
})();
```

Polyfill

- Add missing features to browsers
- NOOP if the browser already supports the feature
- Example: Array.prototype.find; Supported in all browsers except IE and Edge

```
if (Array.prototype.find) return;

var find = function(predicate) {
    ...
};

if (!Array.prototype.find) {
    Array.prototype.find = find;
}
```

github.com/paulmillr/array.prototype.find

Source maps

- “By using source maps, the debugger can map the code being executed to the original source files, making debugging much, much easier.”
- Debug as if the browser is running ES6
- Not just for Javascript

```
// Minified (almost) unreadable code here....
//# sourceMappingURL=student.js.map
```

jQuery

vs

Vanilla JS

Fight!

```
// jQuery
$(document).ready(function() {
  // code
})

// Vanilla
document.addEventListener('DOMContentLoaded', function() {
  // code
})
```

```
// jQuery
var divs = $('div')

// Vanilla
var divs = document.querySelectorAll('div')
```

```
// jQuery
var newDiv = $('<div/>')

// Vanilla
var newDiv = document.createElement('div')
```

```
// jQuery
newDiv.addClass('foo')

// Vanilla
newDiv.classList.add('foo')
```

```
// jQuery
newDiv.toggleClass('foo')

// Vanilla
newDiv.classList.toggle('foo')
```

```
// jQuery
$('body').append($('<p/>'))

// Vanilla
document.body.appendChild(document.createElement('p'))
```

```
// jQuery
var parent = $('#about').parent()

// Vanilla
var parent = document.getElementById('about').parentNode
```

```
// jQuery
$('img').filter(':first').attr('alt', 'My image')

// Vanilla
document.querySelector('img').setAttribute('alt', 'My image')
```

https://gist.github.com/liamcurry/2597326

“When developing your website in HTML, you might be tempted to use the tag to solve problems. Things not lining up in every browser? Maybe another containing would help. Don't know how to select that particular element? Wrap a around it! Unfortunately this can lead to divitis: weighty, convoluted code that is slow to load and difficult to maintain.”

http://www.apaddedcell.com/div-itis-what-it-and-how-avoid-it

```
▼<body>
  ▶<div class="mobile-navigation">…</div>
  ▶<header id="header">…</header>
    <!-- Contact Us Modal with Overlay -->
  ▶<div class="modalOverlay invisible" id="modal_1_Overlay">…</div>
    <div>
    </div>
  ▶<div class="panel-lang-loc" style="height: 963px;">…</div>
  ▼<div class="page-container">
    ▶<div class="hero-slider-container">…</div>
    ▶<div class="background-white">…</div>
    ▼<div class="container-content">
      ▼<div id="card-view" class="tabs">
        ▶<div class="header-container">…</div>
        ▼<div class="slides-container" style="height: 853px; padding-bottom: 80p
          ▼<ul class="slides" style="width: 1000%; transform: translate(-224px,
            ▼<li class="trend-cards" cardtabs-num="0" style="position: relative;
            2004.56px; width: 224px;">
              ▼<div class="card-items" style="position: relative; left: 0%; top:
              height: 2004.56px;">
                ▼<div class="block trendingtopiccardblock  item one-third-width-
                style="position: absolute; left: 0%; top: 0px;">
                    <p>Trending Topics</p>
                  ▼<div class="hashtag-list">
                    ▼<span class="active">
                      ▶<a href="#">…</a> == $0
                      </span>
                    ▶<span class="active">…</span>
```

www.ciber.com/nl/

```
▶ <head>…</head>
▼ <body class="homepage" data-comscore="{"name":"track.click.homepage"}"> == $0
  ▶ <header id="nav">…</header>
  ▼ <main id="content" role="main">
      ::before
    ▶ <section id="topstories" class="js-topstories js-topstories-interactive" data-
    comscore="{"nos_origin":"topstory"}">…</section>
    ▼ <div id="main">
        ::before
      ▼ <section id="featured">
          <h2 class="vh">Uitgelicht nieuws</h2>
        ▼ <ul class="list-featured cf padded-small js-items" data-comscore="{"nos_origin":
        "featured"}">
            ::before
          ▼ <li class="list-featured__item " data-item-id="2087230">
            ▼ <a href="/artikel/2087230-is-aanhangers-opgepakt-bij-actie-in-belgie.html"
              class="link-block js-event-click" data-comscore="{"nos_id":
              2087230,"nos_position":1}">
                ::before
              ▼ <figure class="list-featured__img">
                  <img src="http://nos.nl/data/image/2016/02/16/257126/480x270.jpg" alt
                  class="media-full">
                </figure>
              ▼ <div class="list-featured__wrap">
                  <span class="list-featured__title link-hover">IS-aanhangers opgepakt bij
                  actie in België</span>
                ▶ <p class="list-featured__text">…</p>
                </div>
                ::after
              </a>
            </li>
          ▶ <li class="list-featured__item " data-item-id="2087234">…</li>
          ▶ <li class="list-featured__item " data-item-id="2087233">…</li>
```


nos.nl

- Use semantic HTML(5) tags: <main> <header> <footer> <article> <section> <aside> <figure>
- Use a list if it’s a list of things: <ul>,
- Use <button> for non-href actions

- Don’t use DOM ids unless you need to reference it from Javascript
- Don’t put `block` elements inside `inline` elements
- Use a mixture of tag names and classes in CSS
- Don’t nest your CSS too deep (max ~4 levels deep)

Tip: Add a class to the <body> tag to indicate the page template

Page flow

```
display: block | inline | …

float: left | right | none

position: absolute | fixed | …
```

Display

- Lots of possible values, but mainly `block` or `inline`
- Know your defaults!

	Block	Inline
Width	100%	As needed
Size	CSS	Inherent
Nesting	Yes	Inline only
Tag		

Float

- Content flows around it
- Useful for images, sidebars
- Used a lot before flexbox
- See CSS Shapes for advanced layout

Position

- Static
- Absolute
- Relative
- Fixed
- Sticky (needs polyfill)

Flexbox

Hell yeah!

- Solution to all your (CSS) problems (almost)
- IE10 and newer (with prefixes)
- Bugs in IE10 and IE11

```
align-items: center;
display: flex;
flex-direction: row;
flex-wrap: wrap;
justify-content: center;
```

```
<div>
    <p>Pick an action how you want to proceed:</p>
    <button>Cancel</button>
    <button>Save</button>
</div>
```

```
display: -webkit-flex;
display: -moz-flex;
display: -ms-flex;
display: -o-flex;
display: flex;
```

```
@include border-radius(10px);

-webkit-border-radius: 10px;
-moz-border-radius: 10px;
border-radius: 10px;
```

SASS

github.com/postcss/autoprefixer

```
:fullscreen a {
    display: flex
}
```

```
:-webkit-full-screen a {
    display: -webkit-box;
    display: -webkit-flex;
    display: flex
}
:-moz-full-screen a {
    display: flex
}
:-ms-fullscreen a {
    display: -ms-flexbox;
    display: flex
}
:fullscreen a {
    display: -webkit-box;
    display: -webkit-flex;
    display: -ms-flexbox;
    display: flex
}
```

```
var postcss = require('postcss'),
    processors = [
    require('autoprefixer')({ browsers: ['last 2 versions', 'IE >=
10'] }),
    require('css-mqpacker'),
    require('csswring')
];

return gulp.src('app.css').pipe(postcss(processors));
```

“CSS with superpowers”

http://sass-lang.com/

```
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;

  main {
    background-color: lighten($primary-color, 10%);
  }
}
```

```
@import 'mixins';

.eb-icon {
  background: $eb-splitter-control-bg;
  border: $eb-splitter-control-border;
  border-radius: 50%;
  fill: $eb-splitter-control;
  margin-top: $eb-splitter-control-size / -2;
  margin-left: ($eb-splitter-control-size - $eb-splitter-control-
width) / -2;
  padding: 3px;
  @include position(absolute, 50% 0 0 0);
  @include size($eb-splitter-control-size);
}
```

```
h1 {
  color: $secondary;
  font: 700 16px $font-header;
  @include bp(medium, small) {
    font-size: 13px;
    line-height: 15px;
  }
}
```

```scss
@mixin bp($points...) {
  @each $point in $points {
    @if $point == small {
      // Landscape phones and down
      @media (max-width: 480px) { @content; }
    }
    @else if $point == medium {
      // Landscape phone to portait tablet
      @media (min-width: 480px) and (max-width: 767px) { @content; }
    }
    @else if $point == large {
      // Portrait tablet to landscape and desktop
      @media (min-width: 768px) and (max-width: 1023px) { @content; }
    }
    @else if $point == xlarge {
      // Large desktop
      @media (min-width: 1024px) and (max-width: 1399px) { @content; }
    }
    @else if $point == xxlarge {
        // Extra large desktop
        @media (min-width: 1400px) { @content; }
    }
  }
}
```

“A media query consists of a media type and at least one expression that limits the style sheets' scope by using media features, such as width, height, and color. Media queries, added in CSS3, let the presentation of content be tailored to a specific range of output devices without having to change the content itself.”

—*MDN*

```
@media (min-width: 700px) { ... }

@media (min-width: 700px) and
       (orientation: landscape) { ... }

@media tv and
       (min-width: 700px) and
       (orientation: landscape) { ...  }
```

“Everybody should use SASS!”

–Me

```
gulp.src('app.scss')
    .pipe($.sourcemaps.init())
    .pipe($.sass())
    .pipe($.postcss(processors))
    .pipe($.sourcemaps.write('.'))
    .pipe(gulp.dest('target'));
```

Livereload

- Supported by IDE/tooling or roll your own (using node & npm)
- Instant CSS changes
- Auto reload on HTML & Javascript changes

CSS Frameworks

- Bootstrap
- Foundation
- Material Design (Google)
- and lots more…

- Be aware of what you’re getting
- Be aware of what you’re NOT getting

- Has no knowledge of your app & DOM structure
- Often does not have the same requirements as your app
- Was not created with the design of your app in mind

- Need to specify everything using classes
- Wrap lots of things in s
- Only supports a uniform grid
- Be careful that your site does not have that “bootstrap” look

<svg>

- Icons, logos, infographics
- 1 asset for all devices
- Stylable using CSS (if done right)
- Good support since IE9

```
<body>
  <svg xmlns="http://www.w3.org/2000/svg" style="display:none">
    <symbol id="icon-alert" viewBox="0 0 21 18">
      <path d="....."/>
    </symbol>
  </svg>

  <svg class="icon">
    <use xmlns:xlink="http://www.w3.org/1999/xlink"
         xlink:href="#icon-alert"></use>
  </svg>
</body>
```

```
.icon {
  fill: blue;
  transition: fill 0.2s ease-in-out;
  @include size(12px);

  &:hover {
    fill: red;
  }
}
```

<path fill="#AABBCC" d="..."></path>

```
var icons = gulp.src('app/images/icons/*.svg')
  .pipe($.svgmin())
  .pipe($.svgstore({ inlineSvg: true }))
  .pipe($.cheerio(function ($) {
    $('svg').attr('style', 'display:none');
  }));

gulp.src('demo-app/app/index.html')
    .pipe($.inject(icons, { ... }))

<!-- inject:svg --><!-- endinject -->
```

Cleanup your SVG!

Animation

- CSS Transitions
- CSS Animations
- Javascript animations
- Combination

```
button {
  background-color: #ccc;
  color: #fff;
  transition: 0.2s ease-in-out;
  transition-property: background-color color;

  &:hover {
    background-color: #fff;
    color: #ccc;
  }
}
```

```scss
span {
  animation: bouncedelay 1.4s infinite ease-in-out both;

  &.bounce1 {
    animation-delay: -0.60s;
  }

  &.bounce2 {
    animation-delay: -0.40s;
  }
}

@keyframes bouncedelay {
  0%, 80%, 100% {
    opacity: 0;
    transform: scale(0.4);
  } 40% {
    opacity: 1;
    transform: scale(1.0);
  }
}
```

- CSS is eligible for GPU rendering (use translate3d over translate)
- Be careful with transforming <svg> elements; has a different coordinate system
- Most advanced animations are a mix of Javascript adding/removing CSS animations

<table>

The root of all evil

- ONLY use for simple tabular data
- Anything more complex, use flexbox with lists
- Trying to style <table> lands you back in the browser stone age

Miscellaneous

CSS Variables

```
:root {
  --main-color: #06c;
}
#foo h1 {
  color: var(--main-color);
}
```

Firefox and Chrome only
caniuse.com/#feat=css-variables

WebRTC/ORTC

- Take an avatar photo using your webcam
- Record audio/video in the browser
- Skype in the browser
- Chrome, Firefox and Edge
  caniuse.com/#search=webrtc

CSS Grid Layout

- Ability to define a grid using CSS
- In development by most vendors

```
.container {
  display: grid;
  grid-template-rows: 200px 100px;
  grid-template-columns: repeat(4, 100px);
}
```

Web Workers

- Computation intensive tasks freeze the browser
- Web workers are basically separate threads
- Limits to how a web worker can interact with the rest of the browser

WebGL

- 3D graphics in the browser
- OpenGL based
- Supported in >= IE11
  caniuse.com/#feat=webgl
- www.nytimes.com/interactive/2015/01/09/sports/the-dawn-wall-el-capitan.html

Web Components

- Create your own custom elements
- Angular & Ember & React already simulate this
- Custom elements, HTML templates, Shadow DOM and HTML imports

HTTP 2.0

- Supported by most browsers (>= IE11)
- HTTPS by default
- Compression, multiplexing and prioritisation
- Conflicts with today’s best practices

TypeScript

- ES6 with types
- Much better error messages, code completion during development
- Very useful for large applications
- Preferred for Angular 2.0

```ts
class Greeter {
  greeting: string;

  constructor(message: string) {
    this.greeting = message;
  }

  greet() {
    return "Hello, " + this.greeting;
  }
}

var greeter = new Greeter(123);
// Error: Argument of type 'number' is
// not assignable to parameter of type 'string'
```

Thank you

github.com/jpzwarte/modern-web-development