JavaScript & jQuery

Module “Web-based information systems 1”

Michiel Noback (NOMI)
Institute for Life Sciences and Technology
Hanze University of Applied Sciences

Hello!

I am Michiel Noback

I am here because I love to give presentations on programming.

Introduction

- Client-side = all that happens in the browser
- Consists of HTML5, CSS3, JavaScript (JS) and JS libraries such as jQuery
  - JS is used to manipulate and query the viewed document (the DOM)
  - jQuery is the most popular JS library that makes your life so much easier when doing these tasks

JavaScript is not Java

"JavaScript, also known as ECMAScript, is a dynamic programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed." (Wikipedia)

1.
Divide responsibilities

Where should I put that stuff?

Always remember the
*“**Separation of concerns**”*
gospel

Repeat after me

"My HTML shall not contain JS code"
"My HTML shall not contain styling info"
"My JS code shall not contain HTML"
"My JS code shall not contain styling info"

The clean web mantra

https://www.flickr.com/photos/wicker-furniture/29920284633

My HTML shall not contain JS code

- This means filthy coders' dungeon

```
<button onclick="alert('Hi there')">
   Alert me
</button>
```

My HTML shall not contain JS code

- This is also questionable

```
<button id="alerter">Alert me</button>
<script>
  document.getElementById("alerter").onclick =
    function(){alert('Hi there');};
</script>
```

My HTML shall not contain JS code

- This should be the only place JS is referred to

```
<html>
   <head>
      <script type="text/javascript" src="js_lib/jquery-1.11.1.js" charset="utf-
8"></script>
      <script type="text/javascript" src="js/siteScript.js"></script>
   </head>
   <body>
...
   </body>
</html>
```

My HTML shall not contain styling

- ALL styling should be placed in your css files

```
<html>
   <head>
...
      <link rel="stylesheet" href="css/bootstrap.min.css">
      <link rel="stylesheet" href="css/bootstrap-theme.min.css">
      <link rel="stylesheet" href="css/main.css">
   </head>
   <body>
...
```

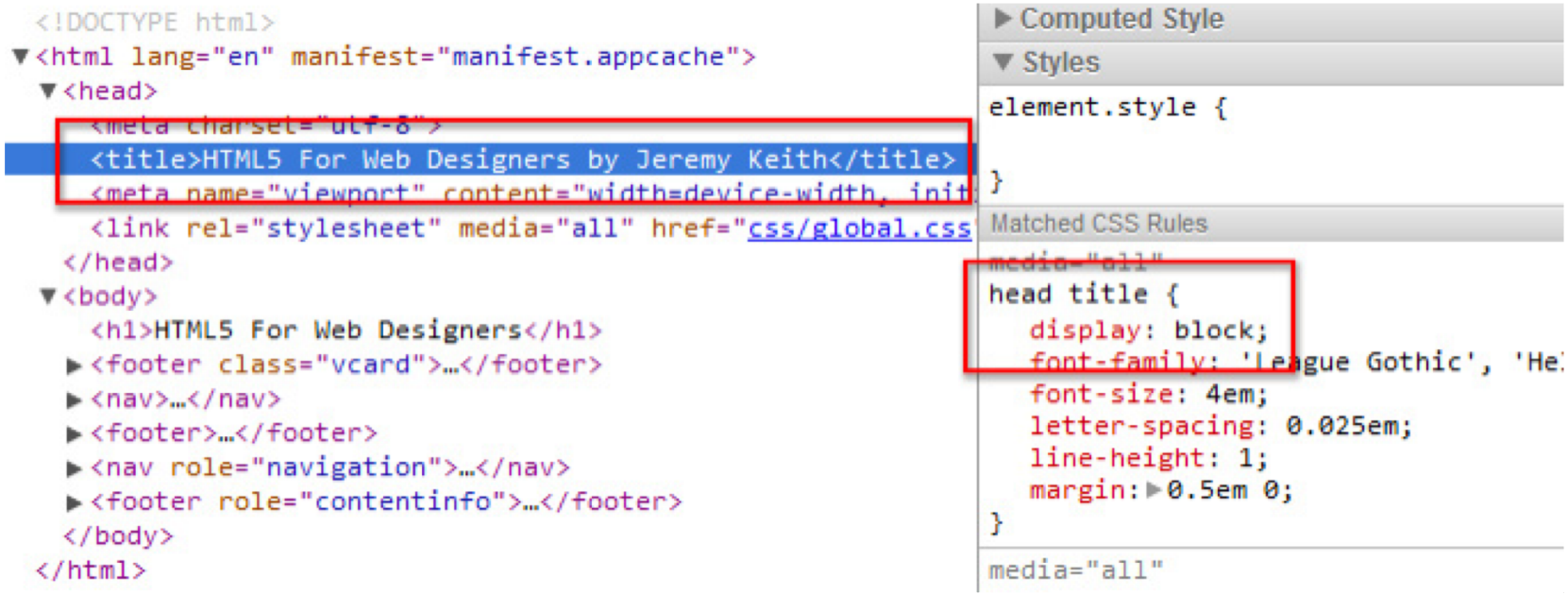
My JS code shall not contain HTML

- This is so tempting...

```
$("#alerter").append(
    "<div>My awesome spoiler</div>"
);
```

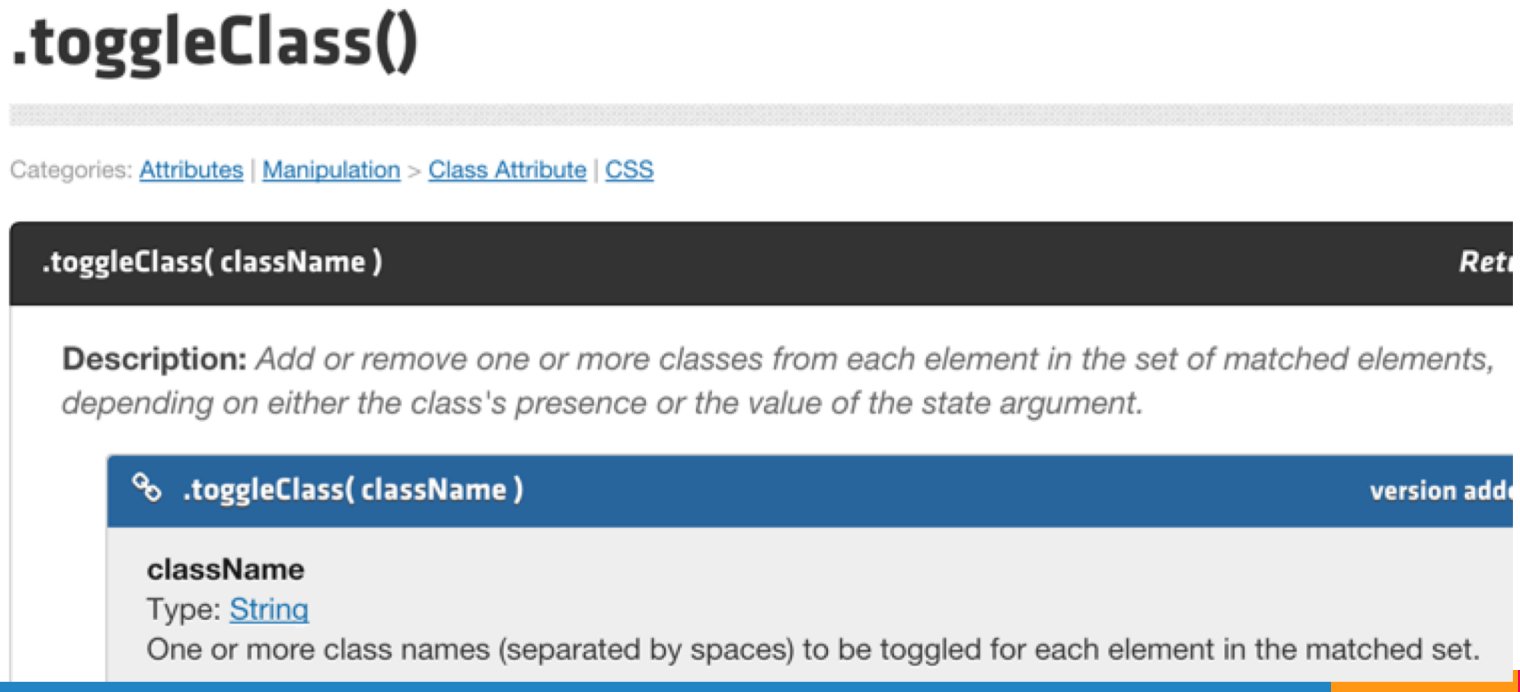
My JS code shall not contain HTML

- Get your HTML snippets from
  - hidden DOM elements
  - AJAX calls



My JS code shall not contain styling info

- Javascript can do all kinds of things with styling…but don’t
- Keep styling in your CSS files - use class toggling and other tools



The holy trinity HTML, CSS & JS

JavaScript

```
<script type=
"text/javascript"
```

Behavior

AJAX data manipulation
Error checking
pop-up calendars
special effects

CSS

```
<style>
Body {
  Color:...
ul#mylist {
  font-family
</style>
```

Presentation

Colors
Fonts
Positioning

HTML

```
<html>
<head>...
<body>
<h1>
<h2>
<p>...</p>
<ol><li><li>
```

Content

Headings
Paragraphs
Lists
Images
Links

2.

Using JavaScript

Using JavaScript in your web pages

Attaching JS to your DOM

1. The Document Object Model (DOM) can be manipulated via JS to create a dynamic web experience
2. The main technique by which you do this is through *event handlers*

Event handler

A piece of code that should run whenever the user triggers an event with a mouse click, a hover, a keyboard stroke etc.

Event types

Keyboard

This is the ancestor of all events

- onkeydown
- onkeypress
- onkeyup
- special case: the Enter key

Mouse

Mouse is on desktop computers the main event source

- onclick
- onmouseover
- onmouseout

Top level Document

The browser also fires (many) events

- onload
- onscroll
- onresize
- ...

Touch

A whole new world

ontouch...

Mouse extras

clientX
clientY
wheelevents
...many more

Media events

onplay
onabort
onprogress
...

Getting real: HTML & CSS

js_demo.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>JS demo</title>
    <link href="css/js_demo.css" rel="stylesheet" type="text/css"/>
    <script src="js/js_demo.js" type="text/javascript"></script>
  </head>
  <body>
    <button id="unclickable">Click me</button>
  </body>
</html>
```

js_demo.css

```
#unclickable {
    color: #2aabd2;
    position: absolute;
    top: 48%;
    left: 48%;
}
```

Getting real: the pure JS solution

js_demo.js

```
/* Initialization on window loading
 * Uses an anonymous function to avoid cluttering the global namespace*/
window.onload = function(){
    function relocate(){
        var width = window.innerWidth;
        var height = window.innerHeight;
        var newLeft = Math.random() * width;
        var newTop = Math.random() * height;
        document.getElementById("unclickable").style.left = newLeft + "px";
        document.getElementById("unclickable").style.top = newTop + "px";
    };
    //relocate when the user attempts to click
    document.getElementById("unclickable").onmouseover = relocate;
};
```

3.

JS best practices

Don’t rely on all those bad GIYF examples

Use libraries that have been optimized for cross-browser compatibility

Keep the namespace clean

Namespacing

- see
  http://www.w3.org/wiki/JavaScript_best_practices

  http://addyosmani.com/blog/essential-js-namespacing/

Strategies for namespacing

- Completely anonymous
- Defined namespace
- …

Anonymous
when nobody needs to know

```
(function(){
  var current = null;
  function init(){...}
  function change(){...}
  function verify(){...}})(); //← This
// is called an immediately invoked
// function definition (expression),
// or IIFE
```

Module pattern
when functionality is needed externally

```
var Module = (function () {
  var privateMethod = function () {};

  return {
    publicMethodOne: function () {
      // I can call `privateMethod()`
    },
    publicMethodTwo: function () {
    },
    publicMethodThree: function () {
    }
  };
})();
```

Module pattern alternative form

```
var Module = (function () {
  // locally scoped Object
  var myObject = {};

  // declared with `var`, must be "private”
  var privateMethod = function () {};

  myObject.someMethod = function () {
    // take it away Mr. Public Method
  };

  return myObject;
})();
```

Module pattern third form

```
var Module = (function () {
  var privateMethod = function () {
    // private
  };
  var someMethod = function () {
    // public
  };
  var anotherMethod = function () {
    // public
  };
  return {
    publicName: someMethod,
    anotherPublicName: anotherMethod
  };
})();
```

Constructor pattern

```
var Scoreboard = function() {
    console.log('Creating a new scoreboard...');
    var message = 'Welcome to the game!';
    function printMessage() {
        console.log(message);
    }
    return {
        printMessage: printMessage
    }
};

var myScoreboard = new Scoreboard();
myScoreboard.printMessage();
```

Further reading on Modules

- https://toddmotto.com/mastering-the-module-pattern/

4.

jQuery

jQuery is the Swiss army knife of JS

jQuery & the DOM

jQuery is used to traverse, select from and manipulate the DOM and its attributes in an easy and cross-browser compatible manner.

When jQuery supports it, don't implement it yourself

- Although it has improved a lot, cross-browser compatibility remains a big issue in Javascript web programming.
- jQuery solves (almost) all of these issues for you!

jQuery anatomy

Most jQuery code consists of these three (four) elements:

- jQuery main call (“jQuery()” or the “$” alias)
- selector
- action
- (action) parameters

jQuery anatomy

selector	action	parameters
`jQuery('p')`	`.css`	`('color', 'blue');`
`$('p')`	`.css`	`('color', 'blue');`

SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change fill color and opacity.
- Change line color, width and style.

Isn't that nice? :)

Examples: