JavaScript

The better Java*

redbrick.dcu.ie/~edu/js.pdf

*Subject to opinion

Wojtek Bednarzak

voy

Redbrick Webmaster

CPSSD

Github: VoyTechnology @MeVoyTech +WojtekBednarzak

Plan

- Boring Lecture
  - Variables
  - Functions
  - Objects
  - Callbacks
  - Errors
- Interesting practical
  - jQuery
  - To-do web app
- Pizza
- Where to continue

Ask Questions any time (but not too many)

WHAT DO I DO?

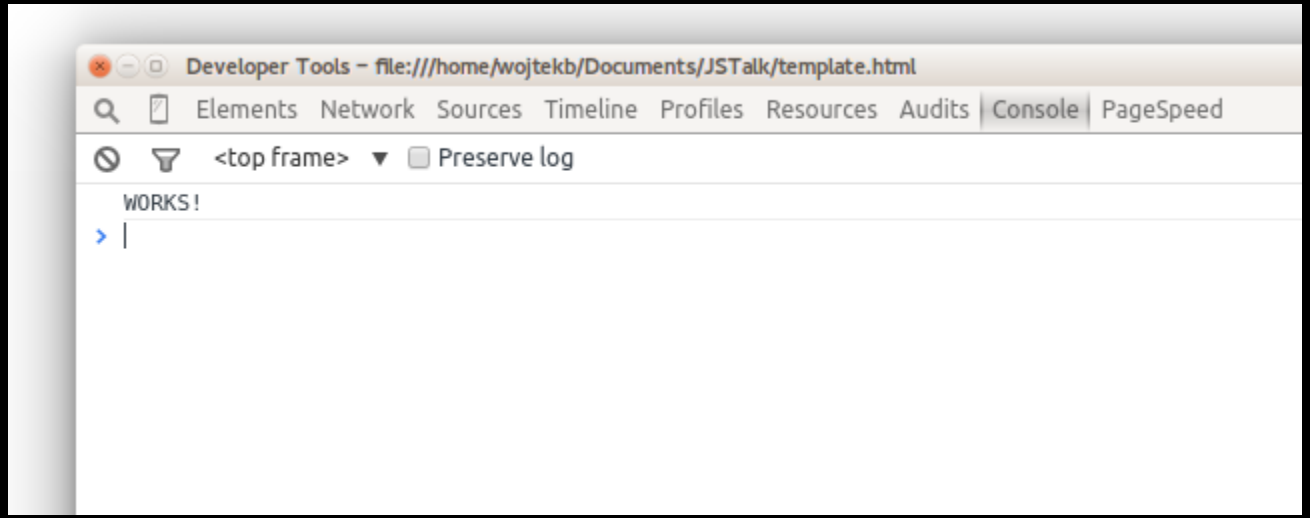
- Create this template:

```
<!DOCTYPE html>
<html>
	<head></head>
	<body>
		<script type=”application/javascript>

			console.log(‘WORKS!!!!’);

		</script>
	</body>
</html>
```

- Little writing in theory part

Google Dev Tools

- F12
- Console tab



What is it?

“JavaScript is a **dynamic** computer programming language. It is most commonly used as part of Web browsers, whose implementation allow client-side scripts to interact with the user, control the browser, communicate **asynchronously**, and alter the document content that is displayed [...]

JavaScript is classified as a prototype-based scripting language with dynamic typing and first-class functions. This mix of features makes it multi-paradigm, supporting object-oriented, imperative and functional programming styles.

Despite some naming [...], JavaScript and Java are otherwise **unrelated** and have very different semantics. The syntax is [...] derived from C”

- Wikipedia

Origins

- Standardized in ECMAScript specification
- Created in 1995
- ECMAScript v5 currently in use
- v6 coming soon

- Contained in RFC-262

What makes it special?

- Easy to write
- Easy to read
- All you need is a web browser

Variables

var name = value

Basic Syntax - Variables

```
var a = 4; // Integer
var b = a; // assignment
var c = true; // Boolean
var d = 3.14; // Float or double
var e = ‘Hello’; // String, both “ ” and ‘ ’ can be used (single quotes prefered)
var f = {‘hello’: ‘world’}; // Object
var g = function(){ alert( ‘Hello World’ ); }; // Function
var h = [a, b, c, d, e, f]; // Array (of anything)
var i = undefined;   \
var j = null;          |  <- All undefined / null
var k;                 /
```

Basic Syntax - Functions

```
function hello( name ){
    console.log( 'Hello, ' + name );
}
hello( 'Han Solo' );


function add( a, b ){
    return a + b;
}
console.log( add( 3.14, 42 ) );
```

Basic Syntax - Arrays

```
var array = [‘dog’, ‘cat’, ‘human’];

console.log( array[0] );

array.push( ‘shark’ );

console.log( array );
```

Basic Syntax - Objects

```
var person = {
        age: 18,
        interests: [
                'JavaScript'
        ],
        getAge: function(){
                return this.age;
        }
};
console.log( person.age );
console.log( person['age'] );
console.log( person.interests[0] );
console.log( person.getAge() );
```

Basic Syntax - String Operations

```
var a = ‘Hello ’ + ‘ World’;

var b = (‘Hello World’).split(‘ ‘);

var c = [‘Hello’, ‘World’].join(‘ ’);

var d = ‘VeryLongWord’.slice(0, 4);
```

Casting

Integer to String (1 -> “1”)

```
1 + “”
(1).toString();
```

String to Integer (“1” -> 1)

```
+’1’
Number(‘1’)
```

Bool to String

```
(true).toString()
String(true)
```

String to Bool

```
Boolean(‘true’)

! WARNING !
‘anything’ == true

!’false’ == false
!’true’ == false
```

Other Basics - Types

typeof {} == ‘object’

typeof “” == ‘string’

typeof 1 == ‘number’

typeof true = ‘boolean’

Other Basics - Maths

10 / 4 = 2.5

2 * 3 = 6

7 % 2 = 1

4 - true = 3 (apparently, since true==1 and false==0)

a++, ++a

b--, --b

2 / 0 = Infinity

Infinity / 0 = Infinity

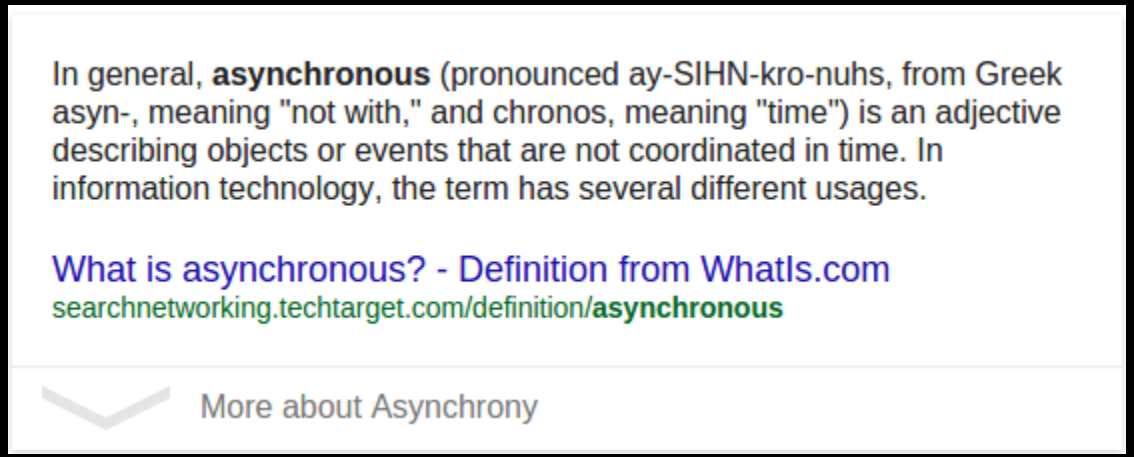
2 / Infinity = 0

Java vs JavaScript

```java
class Auto {
        String make;
        String model;
        Auto(String carMake, String carModel){
                this.make = carMake;
                this.model = carModel;
        }
}

class Test {
        public static void main(String[] args){
                Auto car = new Auto("Tesla", "Model S");
                System.out.println(car);
        }
}
```

```javascript
var Auto = function( carMake, carModel ){
        this.make = carMake;
        this.model = carModel;
};

var car = new Auto( 'Tesla', 'Model S' );
console.log( car );
```

Asynchronous JavaScript



What does it mean?

- Don’t hold and wait, continue running
- Do something when callback is called

In practice:

```
longRunningProcess( function(){
        console.log( ‘Message 1’ );
});

console.log( ‘Message 2’ );
```

Which will show first?

Why Async?

- Greatest friend
- Greatest enemy
- Doesn't stop and wait, multiple things can be running at the same time
- Example:

```
document.addEventListener( 'click', function(){
        alert( 'CLICKED' );
});
```

- Useful for:
  - Database queries
  - Server requests
  - Event listeners

Async Disadvantages

- Difficult to understand at first
- Can create problems
  - You expect something to be ready but its not
  - No return statements
  - Remember the scope
  - Nesting gets messy

Async - Not yet ready

```
var myData = {};

slowProcess( ‘data.json’ , function( data ){
        myData = data;
});

console.log( myData );
```

Result: {}

Async - no return / scope

```
var myData = slowProcess( ‘data.json’ , function( data ){
        return data;
});

console.log( myData );
```

Result: undefined

Async - nasty nesting

```
$(document).ready( function() {
    $(document).on( ‘click’, function(){
        $.getJSON( ‘data.json’, function( data ){
            setTimeout( function(){
                $(document).html( data.status );
            }, 1000);
        });
    });
});
```

JavaScript Errors

- Runtime / JIT Compiled language
  - No need to pre-compile
  - things can be created as you go
  - no real IDE
- Many errors will just show when running

Error Examples

console.log(a):	Uncaught ReferenceError: a is not defined
function(){}:	Uncaught SyntaxError: Unexpected token {
a[1000]:	undefined

Coding

Yay! Finally

What we are doing

- Learning jQuery
- Expanding JS knowledge
- Writing a small to-do web app

HTML Template

```
<!DOCTYPE html>
<html>
    <head>

    </head>
    <body>

    </body>
</html>
```

HTML - head

```
<head>
    <title>To-Do Web App</title>
    <script src="//code.jquery.com/jquery-1.11.2.min.js"></script>
    <script src=”main.js”></script>
    <link rel=”stylesheet” href=”style.css”>
</head>
```

HTML - body

```
<body>

    <div id=”todo”></div>

    <form id=”inputForm”>
        <input id=”action” placeholder=”Task”>
    </form>

</body>
```

Style.css

```
@import url(http://fonts.googleapis.com/css?family=Roboto:300);

body {
        background-color: #eee;
        font-family: ‘Roboto’, sans-serif;
        font-weight: 300;
        border: 0;
        padding: 10%;
}
```

Style.css - Continued

```
#action {
    border: 1px solid #aaa;
    border-radius: 1px;
    background-color: #fff;
    font-size: 2em;
}

#action:selected {
    overline: none;
}
```

Style.css - Continued

```
#todo div {
  margin: 0.5em;
  cursor: pointer;
}
```

main.js

```
$(document).ready( function(){

});

function appendTask( task ){

}
```

main.js - Continued

```
function appendTask( task ){
    if( !localStorage[task.id] ){
        localStorage[task.id] = task.name;
    }
    var  taskHTML = document.createElement( ‘div’ );
        taskHTML.id = task.id;
        taskHTML.innerHTML = task.name;
    var  todo = document.getElementById( ‘todo’ );
        todo.appendChild( taskHTML );
}
```

main.js - Continued

```
$(document).ready( function(){

    for( var id in localStorage ){
        var dbTask = {
            id: id,
            name: localStorage[id]
        };
    }

...
```

main.js - Continued

```
...
$( '#todo div' ).click( function(){
        var el = $(this);

        if(el.css( 'text-decoration' ) == 'line-through' ){
                el.css( 'text-decoration', 'none' );
        } else {
                el.css( 'text-decoration', 'line-through' );
        }
});
```

main.js - Continued

```
...
$( '#inputForm' ).submit( function(){
        var addedTask = $( '#action' ).val();
        $( '#action' ).val(‘’);
        var task = {
                id: ( new Date() ).getTime(),
                name: addedTask
        };
        appendTask( task );
        return false;
});
```

Appendix

JavaScript Resources: https://developer.mozilla.org/en-US/docs/Web/JavaScript

TypeScript: http://www.typescriptlang.org/

Node.js: https://nodejs.org/

nwjs (Node-Webkit): http://nwjs.io/