JavaScript

Laboratory 2

Main features

- Dynamic typing
  - types associated with values, not variables
- Object-oriented
  - objects – associative arrays
  - functions – 1st order objects
  - no classes
- Execution
  - external environment
  - automatic semicolon completion

Variables and functions

- Variables and constants

```
let x = 5;
const c = 23;
```

- Functions

```
//Declaration
function a(p) { ... }
//or
const a = function(p) { ... }
//or
const a = (p) => { ... }

//Call
a(5); //OK
a("asdf"); //OK
a(); //OK
```

Types

- Object
  - Collection of pairs <field, value>
- Boolean
  - Values: true/false
- Undefined
  - Values: undefined
- Null
  - Values: null
- Number
  - Values: 64 bit, float
- String
  - Values: ’text’ or ”text”

Types – what to look out for

- Type coercion
  - 4 == "4" → true
  - 4 === "4" → false

- Boolean
  - false = false, null, undefinded, "", 0, NaN
  - true = true, "asdf", 1, -1, "false", in general – everything that isn't false.

Arrays

```
const t = new Array();
t[0] = 23;
t[1] = "asdf";
t[2] = function(x) { return x; };
t[100] = { name: "Maciej" };

console.log(t[2](5));
console.log(t[100].name);
```

```
const t = ["Peter", "Dante", "Patrokles"];

for (let i = 0; i < t.length; i++) {
    console.log(t[i]);
}

for (let i in t) {
    console.log(t[i]);
}
```

DOM

`window`

- Global context

`document`

- Document tree
- Useful methods

```
getElementById("id")
getElementsByTagName("div")
querySelector(".class div")
```

Arrow functions

- A shorter version of an anonymous function

```
(param1, param2, ..., paramN) => { instrukcje }
(param1, param2, ..., paramN) => wyrażenie

(singleParam) => { instrukcje }
singleParam => { instrukcje }
singleParam => wyrażenie

() => { instrukcje }

(...args) => {
  for (const e of args) {
    console.log(e);
  }
}
```

```
const t = [1, 2, 3];
const t2 = t.map(x => x * x);
```

Loops

```
const t = [5, 10, 15];

for (var i = 0; i < t.length; i++) {
  console.log(t[i]);
}

for (var i in t) {
  console.log(t[i]);
}

t.forEach((e) => { console.log(e); });

for (const e of t) {
  console.log(e);
}

for (const [i, e] of t.entries()) {
  console.log(`t[${i}] = ${e}`);
}
```

Useful resources

- https://exploringjs.com/
- https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Async_JS