Front-End II: JavaScript

Language Technology and Web Applications

Jannis Vamvas [1]

Department of Computational Linguistics
University of Zurich

October 2, 2024

University of Zurich UZH

[1] Parts of this lecture are based on the MDN web docs, which are licensed under CC-BY-SA 2.5.

Project Preferences

Specify your project ideas by the end of the week in the file

`Project Preferences.xlsx`

Quiz: CSS

https://t.uzh.ch/1BW

Learning Goals for this Week

- You recall the main differences between Python and JavaScript
- You can write a simple JavaScript program (with help of documentation) ...
- ... that interacts with a webpage

Topics

1. Introduction

2. Syntax

3. Web Programming

Topics

1. Introduction

2. Syntax

3. Web Programming

Hello World

```
console.log("Hello World!");
```

Linking an External Script

```html
<html>
    <head>
        <script src="script.js" defer></script>
    </head>
    <body>
        ...
    </body>
</html>
```

Internal Script

```
<html>
    <body>
        ...

        <script>
            console.log("Hello World!");
        </script>
    </body>
</html>
```

Topics

1. Introduction

2. Syntax

3. Web Programming

C-like Syntax

```
// Means the same:

console.log("Hello World!");

console
    .log
(
"Hello World!"
)    ;
```

Variables

```
// Declaration
let name;

// Initialization
name = 'Chris';

// Usually: both at the same time
let name = 'Chris';
```

let and const

```
// Variable
let name = 'Chris';

// Constant
const name = 'Chris';
```

Functions

Python

```python
def say_hello(name):
    print("Hello, " + name + "!")
```

JavaScript

```javascript
function sayHello(name) {
    console.log("Hello, " + name + "!");
}
```

Functions as Variables

Python

```python
def say_hello(name):
    print("Hello, " + name + "!")

myfunc = say_hello
```

JavaScript

```javascript
function sayHello(name) {
    console.log("Hello, " + name + "!");
}

let myfunc = sayHello;
```

Functions as Variables

```
let sayHello = function(name) {
    console.log("Hello, " + name + "!");
}
```

Task

Write a function that reduplicates a given string (e.g., `"hellohello"`).

For Loops

Python

```python
for i in range(10):
    print(i)
```

JavaScript

```javascript
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

For Loops

Python

```python
for i in range(0, 10, 1):
    print(i)
```

JavaScript

```javascript
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

Task

Write a function that reduplicates a given string *n* times.

Iterating over Elements

Python

```python
for c in 'Hello':
    print(c)
```

JavaScript

```javascript
for (const c of 'Hello') {
    console.log(c);
}
```

While Loops

Python

```python
n = 0

while n < 3:
    n += 1
```

JavaScript

```javascript
let n = 0;

while (n < 3) {
    n++;
}
```

Primitive Data Types

```
// Number
let n = 123;
let n2 = 12.3;

// String
let s = 'foo';
let s2 = "foo";

// Boolean
let b = true;
let b2 = false;
```

Type Coercion

Python

```python
print("The result is " + 5)
```

```
TypeError: can only concatenate str (not "int") to str
```

JavaScript

```javascript
console.log("The result is " + 5);
```

Type Coercion

```javascript
let sum = '5' + 5;
// '55' (!)
```

Loose Equality and Strict Equality

```
'5' == 5  // true

'5' === 5  // false
```

Conditionals

Python

```python
if n < 0:
    ...
elif n == 0:
    ...
else:
    ...
```

JavaScript

```javascript
if (n < 0) {
    ...
} else if (n === 0) {
    ...
} else {
    ...
}
```

Logical Operators

Python

```
a and b

a or b
```

JavaScript

```
a && b

a || b
```

Task

Write a function that removes all the vowels from a given string.

Built-in String Methods

Python	**JavaScript**
`len(str)`	`str.length`
`str[0]`	`str[0]`
`str[-1]`	`str[str.length-1]`
`str[2:5]`	`str.slice(2,5)`
`str.split()`	`str.split(' ')`
`str.strip()`	`str.trim()`
`str.replace('a', 'b')`	`str.replace('a', 'b')`
`'a' in str`	`str.indexOf('a') !== -1`

Null and Undefined

`undefined` means a variable has been declared but has not yet been initialized.

```
let a;  // a === undefined
```

The value `null` can be assigned to variable to indicate the absence of a value.

```
a = null;
```

Arrays

```
let fruits = ['Apple', 'Banana'];
```

Built-in Array Methods

Python	**JavaScript**
`len(l)`	`l.length`
`l[0]`	`l[0]`
`l[-1]`	`l[l.length-1]`
`l.append(newitem)`	`l.push(newitem)`
`l.pop()`	`l.pop()`
`l1 + l2`	`l1.concat(l2)`
`a in l`	`l.indexOf(a) !== -1`

Objects

```
let dog = {
    "name": "Bello",
    "breed": "Dalmatian",
};

// Two ways to access a member:
dog["name"]  // "Bello"
dog.name  // "Bello"
```

JSON

The syntax of JSON is inspired by JavaScript objects:

```
{
    "name": "Bello",
    "breed": "Dalmatian"
}
```

JSON is an abbreviation for *JavaScript Object Notation*.

Classes

```
class Dog extends Animal {

    constructor(name) {
        super();
        this.name = name;
    }

    bark() {
        console.log('Woof, my name is ' + this.name);
    }
}

let dog = new Dog("Rex");
```

Topics

1. Introduction

2. Syntax

3. Web Programming

Document Object Model

Definition

The **Document Object Model (DOM)** is an API for controlling HTML and styling information that makes heavy use of the `window` and `document` objects.

Try this in the browser console:

```
    console.log(window.innerWidth);
    console.log(window.innerHeight);
```

Accessing HTML Elements

```
// Single element
let body = document.querySelector('body');
let link = document.querySelector('a');

// Array of elements
let links = document.querySelectorAll('a');
```

Task

Use JavaScript to count the hyperlinks on `https://www.uzh.ch/`.

Modifying HTML Elements

```
let link = document.querySelector('a');

link.textContent = 'Click this link!';

link.getAttribute('href');

link.setAttribute('href', 'https://www.example.com');

link.classList.add('hidden');
link.classList.remove('hidden');
```

Creating a New Element

```
let paragraph = document.createElement('p');

paragraph.textContent = 'Thanks for the message!';

document.querySelector('body').appendChild(paragraph);
```

Advanced Selectors

Most types of selectors work for both CSS and JS:

```css
#lead {
  /* ... */
}
```

```js
document.querySelector('#lead');
```

Descendant Combinators

```
/* Direct or indirect descendant */
footer em {
  /* ... */
}

/* Direct descendant */
footer > em {
  /* ... */
}
```

Attribute Selectors

```css
/* Links with an href matching "https://example.com/" */
a[href="https://example.com/"] {
  /* ... */
}

/* All paragraphs in Portuguese */
div[lang="pt"] {
  /* ... */
}
```

Events

“When something happens in the browser window.”

Examples:

- The user clicks on an element
- The user types a key on the keyboard
- A form is submitted

Events

“When something happens in the browser window.”

Examples:

- The user clicks on an element **`(click)`**
- The user presses a key on the keyboard **`(keypress)`**
- A form is submitted **`(submit)`**

Event Listeners

```
let h1 = document.querySelector('h1');

h1.addEventListener('click', function(event) {
    console.log('The title was clicked.');

    event.target.classList.add('red-title');
});
```

Quiz: JavaScript

https://t.uzh.ch/1BW