JavaScript & JSON

Language Technology and Web Applications

Igor Mustač

Department of Computational Linguistics &
Linguistic Research Infrastructure (LiRI)

October 18, 2023

Learning Goals for this Week

- You recall the main differences between Python and JavaScript
- You can write a simple JavaScript program (with the help of documentation) ...
- You are familiar with the structure of JSON

Topics

Topics

What is JavaScript?

- Scripting Language
- Mainly a client-side language
- ECMAScript - language specification
- Node.js (V8)

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

C-like Syntax

```
// Means the same:

console.log("Hello World!");

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

Variables

```
// Declaration
let name;

// Initialization
name = 'Chris';
```

Variables

```
// Declaration
let name;

// Initialization
name = 'Chris';


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

let and const

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

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

var?

```javascript
// Variable
var name = 'Chris';
```

Functions

Python

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

Functions

Python

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

JavaScript

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

Functions as Variables – anonymous function

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

Functions as Variables – anonymous function

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

sayHello("World")
```

Arrow Function

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

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

Template string

Python

```python
str = f"Hello, {name}!"
```

JavaScript

```javascript
let str = `Hello, ${name}`
```

Task

Write a function that reduplicates a given string (e.g., “hellohello”).

```
// function prototype
function reduplicateString(str)
```

For Loops

Python

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

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);
}
```

i++ VS ++i

```
let i = 2;
let a = ++i;
let b = i++;

console.log("Result:", a, b, i);
```

`i++` VS `++i`

```
let i = 2;
let a = ++i;
let b = i++;

console.log("Result:", a, b, i);

// Result: 3, 3, 4
```

i++ VS ++i

```
let i = 2;
let a = ++i;
let b = i++;

console.log("Result:", a, b, i);

// Result: 3, 3, 4
```

Python

```
i = i + 1
i += 1
```

Task

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

```
// function prototype
function reduplicateString(str, n)
```

Iterating over Elements

Python

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

Iterating over Elements

Python

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

JavaScript

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

While Loops

Python

```
n = 0

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

While Loops

Python

```python
n = 0

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

JavaScript

```javascript
let n = 0;

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

Break Loop

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


let n = 0;

while (true) {
    n++;
    break;
}
```

Primitive Data Types

Primitive Data Types

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

Primitive Data Types

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


// String
let s = 'foo';
let s2 = "foo";
let s3 = `foo`;
```

Primitive Data Types

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


// String
let s = 'foo';
let s2 = "foo";
let s3 = `foo`;


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

Type Coercion

```
let sum = '5' + 5;
```

Type Coercion

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

Type Coercion

```
let div = '5'/5;
```

Type Coercion

```
let div = '5'/5;
// 1 (!)
```

Loose Equality and Strict Equality

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

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

Conditionals

Python

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

JavaScript

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

Logical Operators

Python

```
a and b

a or b
```

JavaScript

```
a && b

a || b
```

Bitwise Operators

Python

```
a & b

a | b
```

JavaScript

```
a & b

a | b
```

Conditionals – ternary operator

Python

```python
c = True if a > b else False
```

Conditionals – ternary operator

Python

```
c = True if a > b else False
```

JavaScript

```
let c = a > b ? true : false
```

Task

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

```
// function prototype
function removeVowels(str)
```

Built-in String Methods

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
```

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]` or `l.at(-1)`
`l.append(newitem)`	`l.push(newitem)`
`l.pop()`	`l.pop()`
`l1 + l2`	`l1.concat(l2)`
`a in l`	`l.indexOf(a) !== -1`

Built-in Array Methods - `forEach`

```
let arr = ["A", "B", "C"]

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

Built-in Array Methods - `forEach`

```
let arr = ["A", "B", "C"]

arr.forEach(item => {
    console.log(item)
})
```

Built-in Array Methods - `forEach`

```
let arr = ["A", "B", "C"]

arr.forEach(item => {
    console.log(item)
})

arr.forEach((item, index) => {
    console.log(index, item)
})
```

Built-in Array Methods - `filter`

```
let arr = ["A", "B", "C"]
let newArr = []

arr.forEach(item => {
    if (item !== "A"){
        newArr.push(item)
    }
})
console.log(newArr)

// ["B", "C"]
```

Built-in Array Methods - `filter`

```
let arr = ["A", "B", "C"]

let newArr = arr.filter(item => item !== "A")

console.log(newArr)

// ["B", "C"]
```

Built-in Array Methods

```
let arr = ["A", "B", "C"]

arr.includes("A")   // true



let arr2 = [1, 2, 3]

arr2.map(el => el * 2)   // [2, 4 ,6]
```

Objects

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

Objects

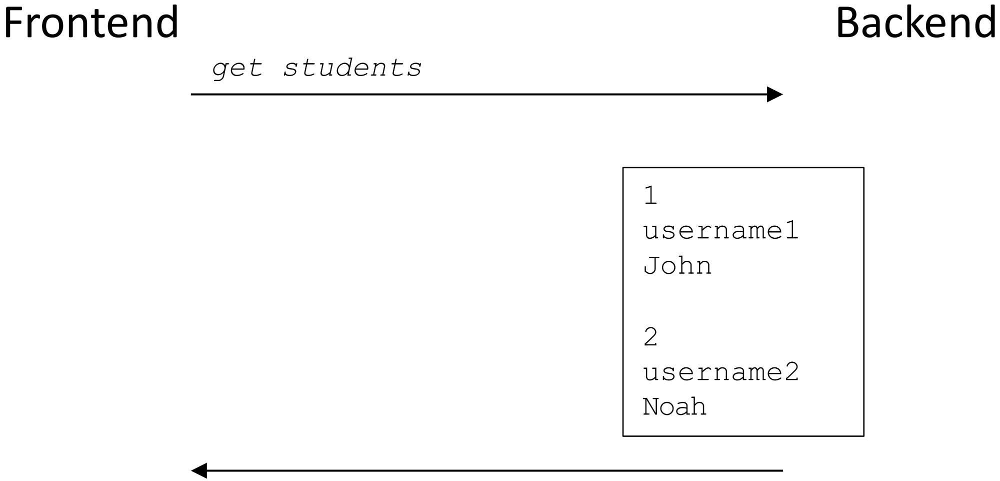
```
let dog = {
    'name': 'Bello',
    'breed': 'Dalmatian',
};

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

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

JavaScript Object Notation



JavaScript Object Notation

- Data Representation Format (data normalization)
- Commonly Used for APIs and Configs
- Lightweight and Easy to Read/Write
- Integrates Easily With Most Languages

JSON Data Types

• String	`"Hello World"`
• Numbers	`20    1.5    -2    1.2e10`
• Booleans	`true    false`
• Null	`null`
• Arrays	`[1, 2, 3] ["Hello", "World"]`
• Objects	`{"key": "value"}  {"age": 20}`

Example 1

```
[{
  "name": "John",
  "isStudent": false,
  "address": {
    "city": "Zürich",
    "postalCode": "8000"
  },
  "friends": [{
    "name": "Noah",
    "friends": [...],
    ...
  }]
}]
```

Example 1

- Can we improve the organization of data in the example?

Task

Decrease the complexity and size of the example JSON file. (*lookups*)

Example 2

- JSON – make it easier to read and write, but …
- What when we have a large amount of data

Example 2

```
ID Name         Courses Enrolled
=====================================================
1  Student1     Math, Science, Web development, English
2  Student2     Math, English
3  Student3     JavaScript, Math, Science
4  Student4     Python, Math, JavaScript
```

Task

Reduce the size of the example JSON file. (*arrays*)

- Questions?