NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

SL3003 - Web Engineering Lab

Lab Instructor: Shariq Masood “shariq.masood.v@nu.edu.pk”

Lab 5

Aim: - Introduction to Advanced JavaScript (ES6)

Arrow function

```
const sum = (a, b) => a + b
console.log(sum(2, 6)) // prints 8
```

Default parameters

```
function print(a = 5) {
 console.log(a)
}
print() // prints 5
print(22) // prints 22
```

let scope

```
let a = 3
if (true) {
 let a = 5
 console.log(a) // prints 5
}
console.log(a) // prints 3
```

const

```
// can be assigned only once:
const a = 55
a = 44 // throws an error
```

Template strings

```
const name = 'Leon'
const message = `Hello ${name}`
console.log(message) // prints "Hello Leon"
```

String includes()

```
console.log('apple'.includes('pl')) // prints true
console.log('apple'.includes('tt')) // prints false
```

String startsWith()

```
console.log('apple'.startsWith('ap')) // prints true
console.log('apple'.startsWith('bb')) // prints
false
```

String repeat()

```
console.log('ab'.repeat(3)) // prints "ababab"
```

Destructuring array

```
let [a, b] = [3, 7];
console.log(a); // 3
console.log(b); // 7
```

Destructuring object

```
let obj = {
 a: 55,
 b: 44
};
let { a, b } = obj;
console.log(a); // 55
console.log(b); // 44
```

object property assignment

```
const a = 2
const b = 5
const obj = { a, b }
// Before es6:
// obj = { a: a, b: b }
console.log(obj) // prints { a: 2, b: 5 }
```

spread operator

```
const a = [ 1, 2 ]
const b = [ 3, 4 ]
const c = [ ...a, ...b ]
console.log(c) // [1, 2, 3, 4]
```

Object.assign()

```
const obj1 = { a: 1 }
const obj2 = { b: 2 }
const obj3 = Object.assign({}, obj1, obj2)
console.log(obj3) // { a: 1, b: 2 }
```

spread operator

```
const a = {
 firstName: "Barry",
 lastName: "Manilow",
}
const b = {
 ...a,
 lastName: "White",
 canSing: true,
}
console.log(a) // {firstName: "Barry", lastName:
"Manilow"}
console.log(b) // {firstName: "Barry", lastName:
"White", canSing: true}
// great for modifying objects without side
effects/affecting the original
```

Exponent operator

```
const byte = 2 ** 8
// Same as: Math.pow(2, 8)
```

Promises with finally

```
promise
 .then((result) => { ⋯ })
 .catch((error) => { ⋯ })
 .finally(() => { // logic independent of
success/error })
// The handler is called when the promise is
fulfilled or rejected.
```

Destructuring Nested Objects

```
const Person = {
        name: "John Snow",
        age: 29,
        sex: "male",
        materialStatus: "single",
        address: {
                country: "Westeros",
                state: "The Crownlands",
                city: "Kings Landing",
                pinCode: "500014",
                },
        };

const { address : { state, pinCode }, name } =  Person;
console.log(name, state, pinCode)

 // John Snow
The Crownlands 500014
console.log(city) // ReferenceError
```

Modules

Language-level support for modules for component definition. Codifies patterns from popular JavaScript module loaders (AMD, CommonJS). Runtime behaviour defined by a host-defined default loader. Implicitly async model – no code executes until requested modules are available and processed.

JavaScript Copy

```javascript
// lib/math.js
export function sum(x, y) {
  return x + y;
}
export var pi = 3.141593;
```

JavaScript Copy

```javascript
// app.js
import * as math from "lib/math";
console.log("2π = " + math.sum(math.pi, math.pi));
```

JavaScript Copy

```javascript
// otherApp.js
import {sum, pi} from "lib/math";
console.log("2π = " + sum(pi, pi));
```

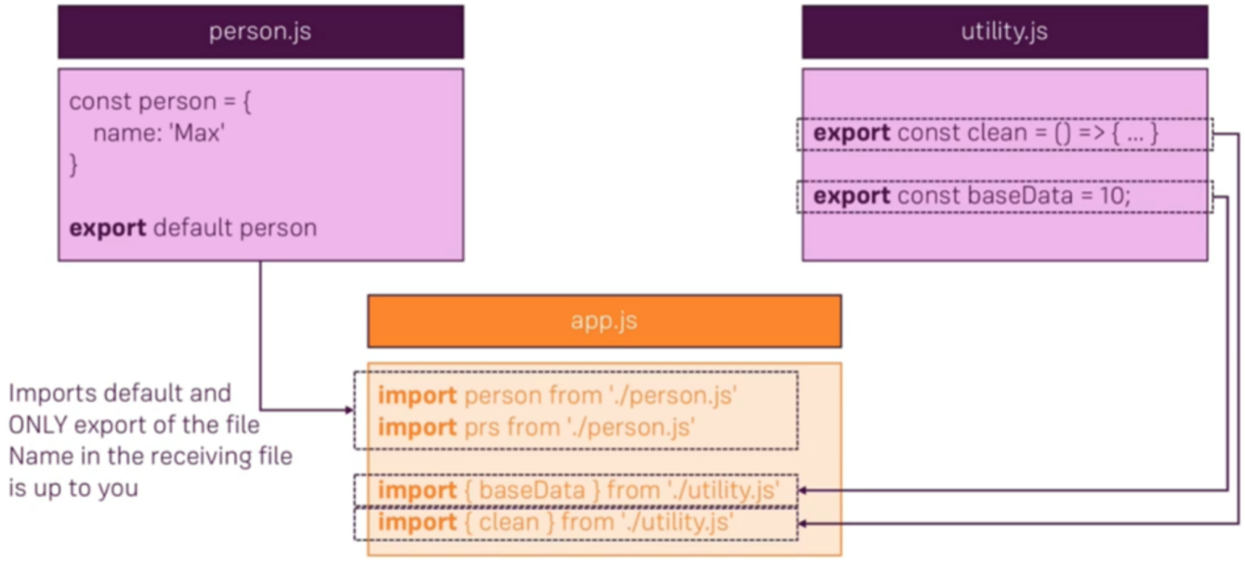
Some additional features include `export default` and `export *`:

JavaScript

```
// lib/mathplusplus.js
export * from "lib/math";
export var e = 2.71828182846;
export default function(x) {
    return Math.exp(x);
}
```

JavaScript

```
// app.js
import exp, {pi, e} from "lib/mathplusplus";
console.log("e^π = " + exp(pi));
```



```
const person = {
  name: 'Max'
}

export default person
```

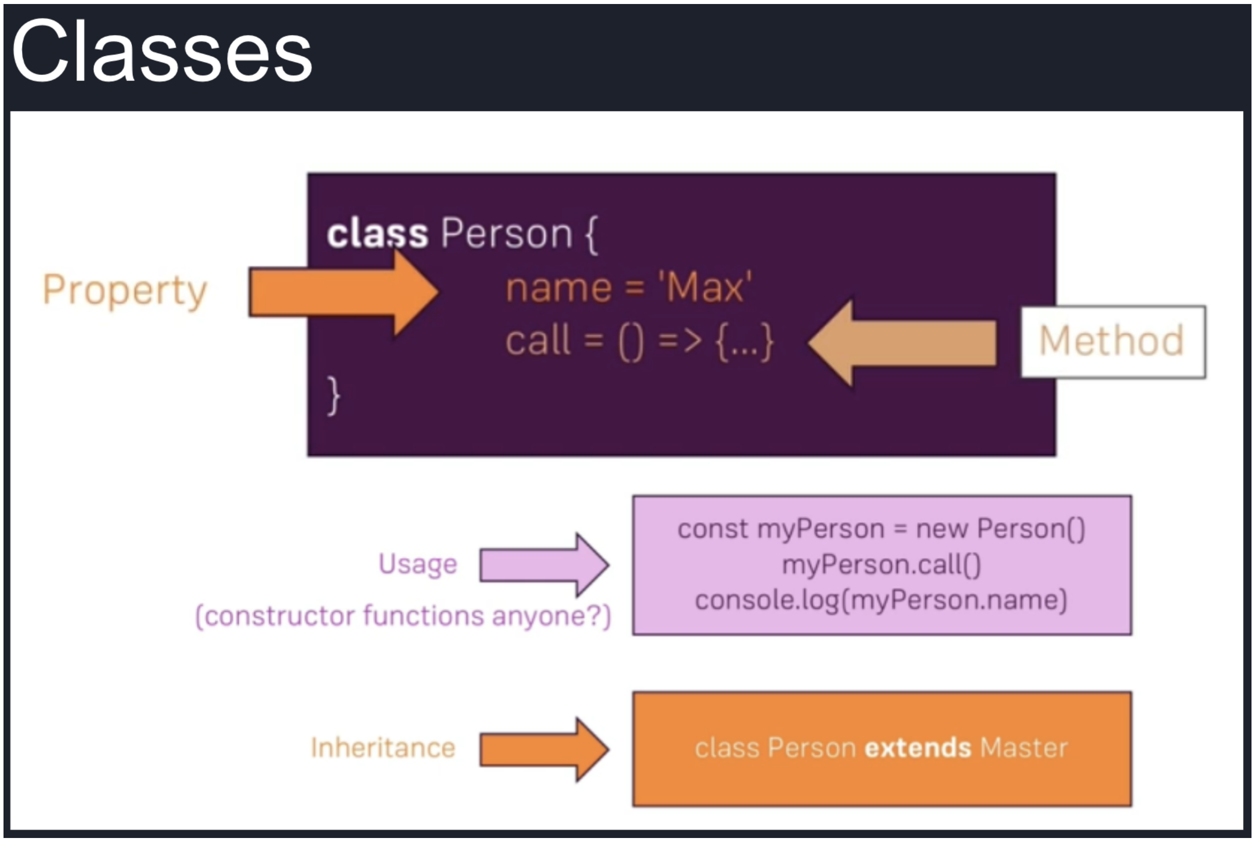
```
export const clean = () => { ... }

export const baseData = 10;
```

```
import person from './person.js'
import prs from './person.js'

import { baseData } from './utility.js'
import { clean } from './utility.js'
```

Imports default and
ONLY export of the file
Name in the receiving file
is up to you



Classes

```
class Person {
    name = 'Max'
    call = () => {...}
}
```

Property

Method

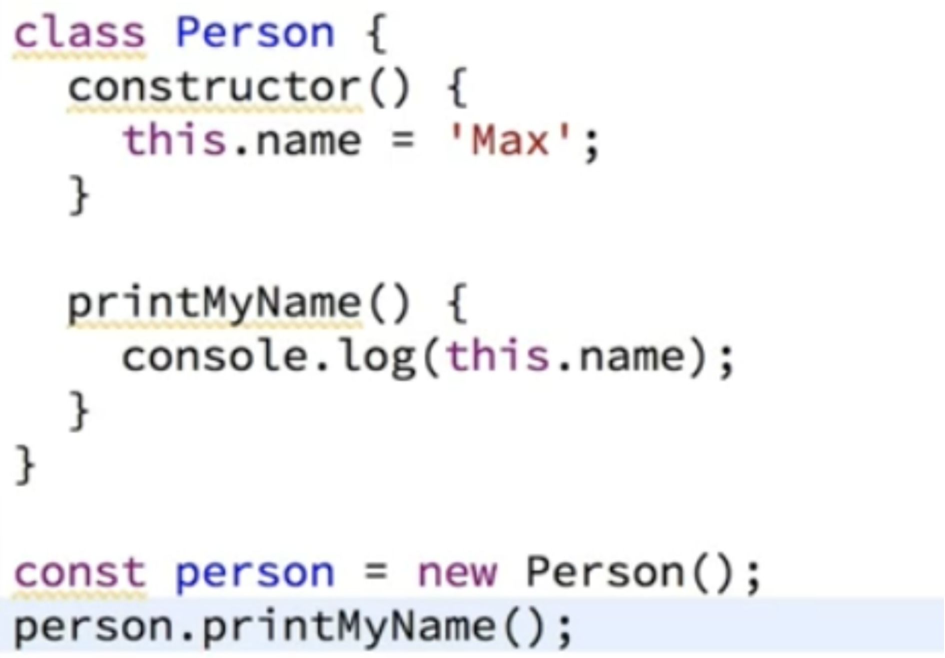
Usage

(constructor functions anyone?)

```
const myPerson = new Person()
myPerson.call()
console.log(myPerson.name)
```

Inheritance

```
class Person extends Master
```



```
class Person {
  constructor() {
    this.name = 'Max';
  }

  printMyName() {
    console.log(this.name);
  }
}

const person = new Person();
person.printMyName();
```

Primitive VS Reference ?

Array Functions:

Map, join, splice, slice, push,pop,shift,unshift

What will be the output?

```
function greet (person) {
  if (person == { name: 'amy' }) {
    return 'hey amy'
  } else {
    return 'hey arnold'
  }
}
greet({ name: 'amy' })
```

Task:

Will be discussed in class

The End…!