UNIVERSIDADE da MADEIRA

CTeSP CURSOS TÉCNICOS SUPERIORES PROFISSIONAIS

Tecnologias e Programação de Sistemas de Informação

JAVASCRIPT & MODULES

Desenvolvimento Web - Back-End | David Jardim

Cofinanciado por:

MADEIRA 14-20

REGIÃO AUTÓNOMA DA MADEIRA

PORTUGAL 2020

UNIÃO EUROPEIA
Fundo Social Europeu

Function declaration in JAVASCRIPT

```
function name([param[, param[, ... param]]]) {
   statements
}
```

- **name**
  - The function name
- **param**
  - The name of an argument to be passed to the function. A function can have up to 255 arguments
- **statements**
  - The statements comprising the body of the function

```
function factorial(number) {
    var fact = 1;
    for (i = 1; i <= number; i++) {
        fact *= i;
    }
    return fact;
}
```

```
function main() {

    var notaP1 = 15;
    var notaP2 = 12;
    var notaPF = 10;
    var notaF = notaFinal(notaP1, notaP2, notaPF);
    console.log("Nota final:" + notaF);

    var f = factorial(5);
    console.log("Factorial:" + f);

    var numbers = [1, 1, 1, 1, 1, 1, 1, 1, 1];
    var avg = average(numbers);
    console.log("Average:" + avg);
}
```

FIRST-CLASS FUNCTIONS: EVERYTHING YOU CAN DO WITH OTHER TYPES YOU CAN DO WITH FUNCTIONS.

You can use functions like strings, numbers, etc. (i.e. pass them around, set variables equal to them, put them in arrays, and more)

First-Class Functions

```
// functions are first-class
function hello(){
    console.log("hi");
}

function log(fn){
    fn();
}

log(hello);
```

Function expression in JAVASCRIPT

```
function [name]([param[, param[, ... param]]]) {
   statements
}
```

- **name**
  - Can be omitted, in which case the function becomes known as an anonymous function

Function expression in JAVASCRIPT

```
var myFunction = function() {
    statements
}

var myFunction = function namedFunction(){
    statements
}
```

One of the benefits of creating a named function expression is that in case we encountered an error, the stack trace will contain the name of the function, making it easier to find the origin of the error.

Function expression in JAVASCRIPT

```
var greetMe = function(){
      console.log("Hello");
 }

greetMe();
log(greetMe);
```

Modules in Node.js

- Module is a simple or complex functionality organized in single or multiple JavaScript files which can be reused throughout the Node.js application

- Each module in Node.js has its own context, so it cannot interfere with other modules or pollute global scope

- Types:
  - Core Modules
  - Local Modules
  - Third Party Modules

Core Modules in Node.js

Core Module	Description
http	http module includes classes, methods and events to create Node.js http server.
url	url module includes methods for URL resolution and parsing.
querystring	querystring module includes methods to deal with query string.
path	path module includes methods to deal with file paths.
fs	fs module includes classes, methods, and events to work with file I/O.
util	util module includes utility functions useful for programmers.

Loading Core Modules in Node.js

```
var module = require('module_name');
```

```
var http = require('http');

var server = http.createServer(function(req, res){

  //write code here

});

server.listen(5000);
```

Loading Local Modules in Node.js

Log.js

```
var log = {
            info: function (info) {
                console.log('Info: ' + info);
            },
            warning:function (warning) {
                console.log('Warning: ' + warning);
            },
            error:function (error) {
                console.log('Error: ' + error);
            }
    };

module.exports = log
```

app.js

```
var myLogModule = require('./Log.js');

myLogModule.info('Node.js started');
```

Export Literals

Message.js

```
module.exports = 'Hello world';

//or

exports = 'Hello world';
```

Now, import this message module and use it as shown below.

app.js

```
var msg = require('./Messages.js');

console.log(msg);
```

Export Object

data.js

```
module.exports = {
    firstName: 'James',
    lastName: 'Bond'
}
```

app.js

```
var person = require('./data.js');
console.log(person.firstName + ' ' + person.lastName);
```

Export Object

Log.js

```
module.exports.log = function (msg) {
    console.log(msg);
};
```

The above module will expose an object- `{ log : function(msg){ console.log(msg); } }`. Use the above module as shown below.

app.js

```
var msg = require('./Log.js');

msg.log('Hello World');
```

Export Function

Log.js

```
module.exports = function (msg) {
    console.log(msg);
};
```

Now, you can use the above module as below.

app.js

```
var msg = require('./Log.js');

msg('Hello World');
```

Export Function as Class

Person.js

```
module.exports = function (firstName, lastName) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.fullName = function () {
        return this.firstName + ' ' + this.lastName;
    }
}
```

The above module can be used as shown below.

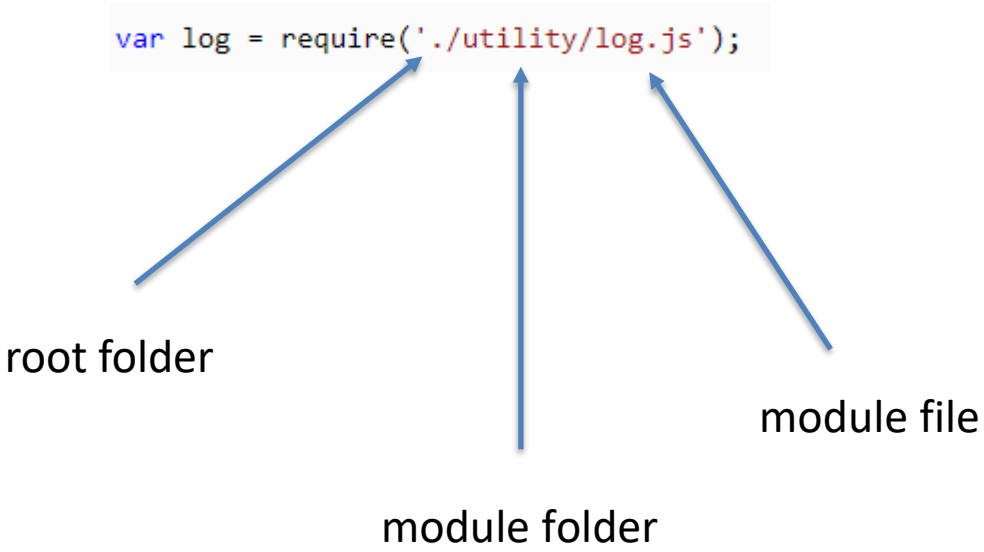
app.js

```
var person = require('./Person.js');

var person1 = new person('James', 'Bond');

console.log(person1.fullName());
```

Load module from another folder




UNIVERSIDADE da MADEIRA
CTeSP
CURSOS TÉCNICOS
SUPERIORES PROFISSIONAIS