

Module 2: Functions in JavaScript

Agenda

- Functions in JS [1]
- Input and Output [2]
- JS Code Processing [3]
- Declaration and Expression [4]

Functions in JS

Basic Information

In mathematics:

Function is a relation between a set of inputs and a set of permissible outputs. [1]

$$y = f(x) \quad [2]$$

In classical programming

Function is a named part of a code that performs a distinct service. [3]

Example

```
var i, base, power, result;      [1]
base = 2; power = 2; result = 1; [2]
for(i = 0; i < power; i++) {    [3]
    result *= base;
}
console.log(result);           [4]
base = 3; power = 4; result = 1;
for(i = 0; i < power; i++) {    [5]
    result *= base;
}
console.log(result);
```

Declaration of function

function is a special keyword for creation of function in JavaScript. [1]

```
function name () {  
    body; [2]  
}
```

Example

```
var i, base, power, result;
```

```
base = 2; power = 2; result = 1;
```

```
for(i = 0; i < power; i++) {
```

```
    result *= base;
```

```
}
```

```
console.log(result);
```

```
base = 3; power = 4; result = 1;
```

```
for(i = 0; i < power; i++) {
```

```
    result *= base;
```

```
}
```

```
console.log(result);
```

Example

```
function pow () {  
    result = 1;  
    for (i = 0; i < power; i++) {  
        result *= base;  
    }  
}
```


Function call

Call - operation for execution of function. [1]

() - operator for this action. [2]

Usually function can be **called** by name. [3]

Example

```
var i, base, power, result;

base = 2; power = 2;
pow();
console.log(result);

base = 3; power = 4;
pow();
console.log(result);

function pow () {
    result = 1;
    for(i = 0; i < power; i++) {
        result *= base;
    }
}
```

Input and Output

Input and Output

```
function name (a, b) {  
    return a + b; [1]  
}
```

* you can return one value only [2]

* **return** always interrupts the execution. [3]

* place your **return** at the end of a function

[3]

Example

```
function pow () {  
    result = 1;  
    for (i = 0, l < power; i++) {  
        result *= base;  
    }  
}
```

Example

```
function pow (base, power) {  
    var result = 1;  
    for (i = 0, i < power; i++) {  
        result *= base;  
    }  
    return result;  
}
```

Example

```
var i, out;

out = pow(2, 2);
console.log(out);

out = pow(3, 4);
console.log(out);

function pow (base, power) {
  var result = 1;
  for(i = 0; i < power; i++) {
    result *= base;
  }
  return result;
}
```

JS Code Processing

Code processing

```
var a = 10;
test();
function test () {
    a = 30;
    var b = 40;
}
var b = 20;
console.log(a, b);
```

Code processing

```
var a = 10;
```

```
test();
```

```
1. function test () {
```

```
    a = 30;
```

```
    var b = 40;
```

```
}
```

```
var b = 20;
```

```
console.log(a, b);
```

Code processing

2. `var a = 10;`

`test();`

1. `function test () {`

`a = 30;`

`var b = 40;`

`}`

3. `var b = 20;`

`console.log(a, b);`

Code processing

2. `var a = 10;`

4. `test();`

1. `function test () {`

`a = 30;`

5. `var b = 40;`

`}`

3. `var b = 20;`

6. `console.log(a, b);`

Code processing

2. `var a = 10;`

4. `test();`

1. `function test () {`

`a = 30; 5.2`

5. `var b = 40; 5.1`
`}`

3. `var b = 20;`

6. `console.log(a, b);`

Declaration and Expression

Declaration and Expression

```
function name () {  
    body;  
}
```

[1]

```
var name = function () {  
    body;  
};
```

[2]

Additional Facts About Functions

Functions in JavaScript are Objects.

[1]

As a result, functions are accessible by reference.

[2]

Functions can be used as a parameter in other function.

[3]

References to functions can be saved in any other variable.

[4]

