JavaScript. Функції
Лекція 4

JavaScript. Функції
Лекція 4

- Вступ
- Оголошення, виклик функції
- Аргументи функції
- Arguments
- Методи функції
- Функції - тип даних
- Оголошення функцій
- Область видимості
- Нові можливості ECMAScript 2015

Вступ

ФУНКЦІЇ

- Не повторювати код
- Об'єднати групу дій
- Зручніше читати та змінювати код
- Створити область видимості
- Рекурсивний виклик

Оголошення, виклик функції

Оголошення функції

```
function add(x, y) {
    return x + y;
}
```

Виклик функції

```
function add(x, y) {
    return x + y;
}
add(2, 3);    // 5
```

Без return

```
function lazy() {
}
lazy();    // undefined
```

Аргументи функції

```
function asIs(x){
    return x;
}
asIs(); // undefined
```

```
function myMin(a, b) {
    return a < b ? a : b;
}
myMin(2, 7);  // 2
myMin(13, 7); // 7
myMin(13); // undefined
myMin(7, 5, 1); // 5
```

```
function myMin(a, b) {
    if (b === undefined) {
        return a;
    }
    return a < b ? a : b;
}
myMin(13);        // 13
```

Значення за замовчуванням

```
function myMin(a, b) {
    b = b || Infinity;
    return a < b ? a : b;
}
myMin(2, 7);      // 2
myMin(13, 7);     // 7
myMin(-13);       // -13
myMin(13, 0);     // 13
```

arguments

```
function myMin(a, b) {
    var min = a < b ? a : b;
    var c = arguments[2];
    return c < min ? c : min;
}
```

```
myMin(2, 3, 4);     // 2
myMin(20, 3, 4);    // 3
myMin(20, 30, 4);   // 4
```

```
function addMany() {
    var sum = 0;
    var length = arguments.length;
    for (var i = 0; i < length; i += 1) {
        sum += arguments[i];
    }
    return sum;
}
```

```
addMany(2, 3, 4);    // 9
```

Приведення до масиву

```
var args = [].slice.call(arguments);
```

Методи функції

```
Math.min(3, 5, 2, 6);          // 2
var numbers = [3, 5, 2, 6];
Math.min(numbers);             // NaN
```

```
var numbers = [3, 5, 2, 6];
Math.min.apply(null, numbers); // 2
```

```
Math.pow(base, exponent);
Math.pow(2, 4);              // 16
Math.pow(2, 10);             // 1024
```

```
var binPow = Math.pow.bind(null, 2);
binPow(4);   // 16
binPow(10);  // 1024
```

Функції - тип даних

```
function add(a, b) {
    return a + b;
}
var sum = add;
sum(1, 3); // 4
```

```
function multiplyBy2(x) {
    return x * 2;
}
var numbers = [3, 5, 9];
numbers.map(multiplyBy2);
// [6, 10, 18]
```

```
function getAdd() {
    function add(a, b) {
        return a + b;
    }
    return add;
}
var myAdd = getAdd();
myAdd(1, 3); // 4
```

function declaration

```
function add(x, y) {
    return x + y;
}
```

```
add(2, 3);    // 5

function add(x, y) {
    return x + y;
}
```

```
function add(x, y) {
    return x + y;
}

add(2, 3);    // 5
```

function expression

```
var add = function (x, y) {
    return x + y;
};
```

```
add(2, 3);
  // TypeError:
  // add is not a function

var add = function (x, y) {
    return x + y;
};
```

```
var add = function hidden(){
    return typeof hidden;
}
typeof add; // 'function'
typeof hidden;
   // ReferenceError:
   // hidden is not defined
add();  // 'function'
```

```
var factoral=function in(n){
 return (n === 1) ?
     1 : n * inner(n - 1);
}
factorial(3); // 6
```

Конструктор Function

```
var add = new Function('x', 'y', 'return x + y');

add(2, 3);    // 5
```

Область видимості

```
function f() {
    var text = 'Hello';
    console.log(text);  // Hello
}
console.log(text);    // ReferenceError:
                      // text is not defined
```

```
function outer() {
    var text = 'Hello';
    function inner() {
        console.log(text);
    }
    inner();
}
outer(); // Hello
```

```
function outer() {
    var text = 'outer';
    function inner() {
        var text = 'inner';
        console.log(text);
     }
    inner(); // 'inner'
    console.log(text);
}
outer(); // 'outer'
```

Область видимості

```
function f() {
    if (true) {
        var text = 'Hello';
    }

    console.log(text); // 'Hello'
}
```

```
function f() {
    var text;
    if (true) {
        text = 'Hello';
    }

    console.log(text); // 'Hello'
}
```

Функція, що негайно викликається
IIFE (immediately-invoked function expression)

```
(function(n) {
  for(var i = 0; i < n; i++) {
    console.log(i);
  }
})(5);
//0
//1
//2
//3
//4
console.log(typeof i); // => undefined
```

Нові можливості ECMAScript 2015

```
function foo() {
    let bar = 5;
    if (true) {
        let bar = 4;
    }
    console.log(bar);
}
foo(); //5
```

```
function add(x, y = 0) {
    return x + y;
}
```

```
function add(...args) {
    var sum = 0;
    args.forEach(function (arg) {
        sum += arg;
    });
    return sum;
}
```

Дякую!