Курс:
Frontend с нуля

Тренер:
Максим Шепель

Занятие 9
JavaScript: операторы, инструкции

WEB ACADEMY
PROGRAMMING COURSES

Содержание занятия

- Операторы сравнения
- Операторы для работы с объектами
- Приоритетность операторов и типы операндов
- Логические выражения
- Выражения и объявления как инструкции
- Условные инструкции
- Циклы
- Обработка ошибок
- Другие инструкции

typeof

```
typeof 1 // "number"
typeof Infinity // "number"
typeof NaN // "number"
typeof 'text' // "string"
typeof true // "boolean"
typeof new Number(1) // "object"
typeof {} // "object"
typeof [] // "object"
typeof null // "object"
typeof undefined // "undefined"
typeof function () {} // "function"
```

Также допускается написание в формате `typeof(x)`

Нестрогое равенство

```
undefined == undefined
null == null
null == undefined
NaN != NaN
+0 == -0
{} != {}
[] != []
[] == 0
[] == ''
[1] == 1
[1, 2] == '1,2'
var a = {}, b = {}; a == b
1 == '1'
true == 1
false == 0
1 == new Number(1)
```

Строгое равенство

```
undefined === undefined
null === null
null !== undefined
NaN !== NaN
+0 === -0
1 === 1
'1' !== 1
1 !== new Number(1)
```

Сравнение

```
10 > 2
'10' < '2'
'A' < 'a'
'Abc' < 'abc'
'Butthead' > 'Beavis'
true > false
```

Побитовые операторы

Бинарные:
& (AND)
| (OR)
^ (XOR)
<< (левый сдвиг)
>> (правый сдвиг)
>> (правый сдвиг с заполнением нулями)

Унарный:
~ (NOT)

Работа с объектами

```
var obj = {
    index: 1,
    getIndexString: function () {
        return String(this.index);
    },
    numbers = [2, 4, 6, 8]
};

'index' in obj // true
'getIndexString' in obj // true
'toString' in obj // true
'0' in obj.numbers // true
1 in obj.numbers // true
10 in obj.numbers // false

delete obj.index;
'index' in obj // false
obj.index // undefined
delete obj.numbers[0];
'0' in obj.numbers // false
obj.numbers[0] // undefined
```

instanceof

```
var d = new Date(), arr = [0, 1];

d instanceof Date // true
d instanceof Object // true
d instanceof String // false
arr instanceof Array // true
arr instanceof Object // true

var str = 'text', strObj = new String('text');
str instanceof String // false
strObj instanceof String // true
strObj instanceof Object // true
```

Приоритетность операторов и типы операндов

Операторы	Типы
`++`	`lval` -> число
`--`	`lval` -> число
`-`	число -> число
`+`	число -> число
`~`	целое число -> целое число
`!`	булев -> булев
`delete`	`lval` -> булев
`typeof`	любой -> строка
`void`	любой -> `undefined`
`* / %`	число, число -> число
`+ -`	число, число -> число
`+`	строка, строка -> строка

Операторы	Типы
`<<`	целое число, целое число -> целое число
`>>`	целое число, целое число -> целое число
`>>>`	целое число, целое число -> целое число
`< <= > >=`	число, число -> булев
`< <= > >=`	строка, строка -> булев
`instanceof`	объект, функция -> булев
`in`	строка, объект -> булев
`==`	любой, любой -> булев
`!=`	любой, любой -> булев
`===`	любой, любой -> булев

Приоритетность операторов и типы операндов

Операторы	Типы
`!==`	любой, любой -> булев
`&`	целое число, целое число -> целое число
`^`	целое число, целое число -> целое число
`|`	целое число, целое число -> целое число
`&&`	любой, любой -> любой
`||`	любой, любой -> любой
`?:`	булев, любой, любой -> любой
`=`	`lval`, любой -> любой
`*= /= %= += -= &= ^= |= <<= >>= >>>=`	`lval`, любой -> любой
`,`	любой, любой -> любой

Логические выражения

```
var a = 0, b = 1, c = 2;
console.log(a || b); // >>> 1
console.log(b || c); // >>> 1
console.log(a && b); // >>> 0
console.log(b && c); // >>> 2
```

Блоки

```
var x = 10,
    y = 20;
{
    let x = 1; // ES6
    y += 5;
    var z = 30;
    console.log(x + 1); // >>> 2
}
console.log(x); // 10
console.log(y); // 25
console.log(z); // 30
```

Условные инструкции: if, if/else

```
var isNumber = typeof x == 'number';

if (isNumber) console.log(x);

if (isNumber) {
    console.log(x);
    x++;
}

if (isNumber) {
    console.log(x);
    x++;
} else {
    console.log('not a number');
}
```

Условные инструкции: if/else if

```
var isString = typeof x == 'string',
    isFiniteNumber = typeof x == 'number' && isFinite(x),
    isUndefined = typeof x == 'undefined';

if (isFiniteNumber) {
    console.log(x);
    x++;
} else if (isString) {
    console.log('text: ' + x);
} else if (isUndefined) {
    console.log('value not defined');
    x = 0;
} else {
    console.log('unknown case');
}
```

Условные инструкции: switch

```
switch (language) {
    case 'HTML':
        console.log('markup');
        break;
    case 'CSS':
        console.log('styling');
        break;
    case 'JS':
    case 'JavaScript':
        console.log('behaviour');
        break;
    default:
        console.log('something different');
        break;
}
```

Циклы: while

```
var i = 0, j = 0, k = 0;

// будут выведены числа от 0 до 9
while (i < 10) console.log(i++);

// будут выведены числа от 0 до 9
while (j < 10) {
    console.log(j);
    j++;
}

// не выполнится ни разу
while (k < 0) {
    console.log(k);
    k++;
}
```

Циклы: do/while

```
var i = 0, j = 0, k = 0;

// будут выведены числа от 0 до 9
do console.log(i++); while (i < 10)

// будут выведены числа от 0 до 9
do {
    console.log(j);
    j++;
} while (j < 10);

// выполнится 1 раз
do {
    console.log(k);
    k++;
} while (k < 0);
```

Циклы: for

Синтаксис:

```
for (инициализация; условие; переход на следующую итерацию)
    тело цикла

var arr = ['Veni', 'Vidi', 'Vici'];

for (var i = 0, count = arr.length; i < count; i++) {
    console.log(arr[i]);
}

for (var count = arr.length, i = count; i--;) {
    console.log(arr[count - i - 1]);
}
```

Циклы: for/in

```
var obj = {
    index: 0,
    name: 'plain object'
};

for (var propName in obj) {
    console.log(propName, obj[propName]);
}

// >>> index 0
// >>> name plain object
```

break

```
var arr = ['Veni', 'Vidi', 'Vici'];

for (var i = 0, count = arr.length; i < count; i++) {
    var item = arr[i];
    if (item == 'Vidi') {
        break;
    }
    console.log(item);
}

// >>> Veni
```

continue

```
var arr = ['Veni', 'Vidi', 'Vici'];

for (var i = 0, count = arr.length; i < count; i++) {
    var item = arr[i];
    if (item == 'Vidi') {
        continue;
    }
    console.log(item);
}

// >>> Veni
// >>> Vici
```

Метки инструкций

```
var arr = [
    ['Veni', 'Vidi', 'Ut perii'], ['Veni', 'Vidi', 'Vici'],
    ['Veni', 'Vidi']
];

outerCycle: for (var i = 0, n = arr.length; i < n; i++) {
    for (var j = 0, m = arr[i].length; j < m; j++) {
        var item = arr[i][j];
        if (item == 'Vici') break outerCycle;
        if (item == 'Ut perii') continue outerCycle;
        console.log(item);
    }
}

// >>> Veni
// >>> Vidi
// >>> Veni
// >>> Vidi
```

Обработка ошибок

```
try {
    if (!isFinite(x)) {
        throw new Error('x is not valid number');
    }
} catch (err) {
    console.log('operation failed: ', err);
} finally {
    Console.log('operation finished');
}
```