

# Advanced JavaScript Techniques



Advanced JavaScript is closely associated with Ajax, but this is not really a subject. There are not going to talk about this at all, and we will avoid using fancy advanced topics.

## Loose Typing

One of the greatest features of JavaScript is its loose typing. This allows you to use a variable of one type to store a value of another type without any special syntax.

```
var a = 5 + 4 + "px";  
var b = "$" + 5 + 4;  
var c = [1, 2, 3] + " and counting..";
```

```
alert(a);    9px  
alert(b);    $54  
alert(c);    1,2,3 and counting..
```

Loose Typing

## null, NaN & undefined

```
var a = null;  
var b = 5 - "$";
```

```
alert(a);    // null  
alert(b);    // NaN  
alert(c);    // undefined
```

null, NaN & undefined

Type of...

&& and ||

```
var a;  
if (b) {  
  a = b;  
} else {  
  a = 0; // default  
}
```

```
var a = b || 0;
```

&& and ||

```
var a;  
if (b) {  
  a = c;  
} else {  
  a = b;  
}
```

```
var a = b && c;
```

&& and ||

Anonymous Functions

```
function plus(a, b) {  
  return a + b;  
}  
var minus = function (a, b) {  
  return a - b;  
}  
function operation(a, b, sign) {  
  var f = sign?plus:minus;  
  return f(a, b);  
}
```

Anonymous Functions

```
function operation(a, b, sign) {
  var f = (typeof sign == "function") ?
    sign : sign ? plus : minus;
  return f(a, b);
}

var a = operation(3, 4, true);
var b = operation(3, 4, minus);
var c = operation(3, 4, function(a, b)
{ return a * b;});
```

## Anonymous Functions

## Object

```
var Orc = {
  name: "Zurk",
  level: 80,
  damage: 300,
  "go-to": function () {},
  die: function () {}
}
Orc.level++;
Orc.damage += 3;
Orc["go-to"](x, y);
Orc.die();
Orc.avatar = "http://orcs.org/zurk/plc.jpg";
```

## Object

```
function Human(name) {
  this.name = name;
  this.getName = function () {
    return this.name;
  };
}
function Orc(name) { this.name = name; }
Orc.prototype.getName = function () {
  return this.name;
};
john = new Human("John");
zorg = new Orc("Zorg");
```

## Object

## Prototype

```
var Elf = function () {};
Elf.prototype = {
  weapon: "bow",
  attack: function () {...},
  damage: 21
};
var Troll = function () {};
Troll.prototype = new Elf();
Troll.prototype.weapon = "axe";
var Zahar = new Troll(),
Zabah = new Troll(),
Féanor = new Elf();
```

## Prototype

Who is There?

```
for (var i in zorg) {  
  alert("zorg." + i + " = " + zorg[i]);  
}  
if ("wings" in unit) {  
  alert("the unit can fly!");  
}
```

Who is There?

'this' is fun

```
var Elf = function () {};  
Elf.prototype = {  
  weapon: "bow",  
  attack: function (target) {  
    target.health -= this.damage;  
  },  
  damage: 21  
};  
var Troll = {};  
Troll.weapon = "axe";  
Troll.damage = 28;  
Troll.attack = Elf.prototype.attack;
```

'this' is fun

Constructor

```
var zorg = new Orc("Zorg");  
if (zorg.constructor == Orc) {  
  alert("Run! Orcs in town!");  
}  
  
var borg = new function("Börg") {...};  
var zima = new borg.constructor("Zima");  
  
typeof ([1, 2, 3]) == "object";  
([1, 2, 3]).constructor == Array;
```

Constructor

## Instance of...

```
[[1, 2, 3]].constructor == Array; // true  
[[1, 2, 3]].constructor == Object; // false  
  
[[1, 2, 3]] instanceof Array; // true  
[[1, 2, 3]] instanceof Object; // true
```

## Instance of...

## Closures

```
var x = add(4)(5);  
  
var x = (add(4))(5);  
var z = add(4);  
var x = z(5); // 9  
  
function add(a) {  
  return function(b) {  
    return a + b;  
  }  
}
```

## Closures

## Arguments

```
function add(a, b) { return a + b; }  
function add(a, b, c) { return a + b + c; }  
  
function add() {  
  var res = 0;  
  for (var i = 0; i < arguments.length; i++){  
    res += arguments[i];  
  }  
  return res;  
}
```

## Arguments

## Scope

```
function die(cry, volume) {
  this.hitpoints = 0;
  this.damage = 0;
  this.look = "dead-body";
  this.play(cry, volume);
}
var Zhuk = new Orc();
Zhuk.die = die;
Zhuk.die();

die.call(Zhuk, cry, volume);
die.apply(Zhuk, [cry, volume]);
```

Scope

## Morphing Functions

```
function getButtonImage() {
  var res = "";
  if (browser.isIE) {
    res = "/i/ie-win.png";
  } else {
    res = "/i/not-win.png";
  }
  getButtonImage = function () {
    return res;
  }
  return res;
}
```

Morphing Functions

## Callee

```
var fibonacci = function () {
  if (!("numbers" in arguments.callee)) {
    arguments.callee.numbers = [0, 1];
    return 0;
  }
  var len =
    arguments.callee.numbers.length;
  arguments.callee.numbers.push(
    arguments.callee.numbers[len - 1] +
    arguments.callee.numbers[len - 2]);
  return arguments.callee.numbers[len - 1];
}
```

Callee

toString

```
var feanor = new Elf();  
alert(feanor); // [object]  
feanor.toString = function () {  
  return ":"; }  
alert(feanor); // :)
```

toString

# Thank You

<http://dmitry.baranovskiy.com>  
[dmitry@baranovskiy.com](mailto:dmitry@baranovskiy.com)