It’s important to establish and follow coding conventions—they make your code consistent, predictable, and much easier to read and understand. A new developer joining the team can read through the conventions and be productive much sooner, understanding the code written by any other team member.

**Indentation**

The rule is simple—anything within curly braces. This means the bodies of functions, loops (do, while, for, for-in), ifs, switches, and object properties in the object literal notation.

[复制代码](javascript:void(0);)

function outer(a, b) {

var c = 1,

d = 2,

inner;

if (a > b) {

inner = function () {

return {

r: c - d

};

};

} else {

inner = function () {

return {

r: c + d

};

};

}

return inner;

}

[复制代码](javascript:void(0);)

**Curly Braces**

Curly braces should always be used, even in cases when they are optional.

[复制代码](javascript:void(0);)

// bad practice

for (var i = 0; i < 10; i += 1)

alert(i);

// better

for (var i = 0; i < 10; i += 1) {

alert(i);

}

Similarly for if conditions:

// bad

if (true)

alert(1);

else

alert(2);

// better

if (true) {

alert(1);

} else {

alert(2);

}

[复制代码](javascript:void(0);)

**Opening Brace Location**

semicolon insertion mechanism—JavaScript is not picky when you choose not to end your lines properly with a semicolon and adds it for you.

[复制代码](javascript:void(0);)

// warning: unexpected return value

function func() {

return

{

name: "Batman"

};

}

[复制代码](javascript:void(0);)

If you expect this function to return an object with a  name property, you’ll be surprised. Because of the implied semicolons, the function returns undefined. The preceding code is equivalent to this one:

[复制代码](javascript:void(0);)

// warning: unexpected return value

function func() {

return undefined;

// unreachable code follows...

{

name: "Batman"

};

}

[复制代码](javascript:void(0);)

In conclusion, always use curly braces and always put the opening one on the same line as the previous statement:

function func() {

return {

name: "Batman"

};

}

**White Space**

Good places to use a white space include:

• After the semicolons that separate the parts of a for loop: for example, for (var i= 0; i < 10; i += 1) {...}

• Initializing multiple variables (i and max) in a for loop:  for (var i = 0, max = 10; i < max; i += 1) {...}

• After the commas that delimit array items: var a = [1, 2, 3];

• After commas in object properties and after colons that divide property names and their values: var o = {a: 1, b: 2};

• Delimiting function arguments: myFunc(a, b, c)

• Before the curly braces in function declarations: function myFunc() {}

• After function in anonymous function expressions:  var myFunc = function () {};

Another good use for white space is to separate all operators and their operands with

spaces, which basically means use a space before and after  +,  -,  \*,  =,  <,  >,  <=,  >=,  = = =,  != =, &&, ||, +=, and so on:

[复制代码](javascript:void(0);)

// generous and consistent spacing makes the code easier to read allowing it to "breathe"

var d = 0,

a = b + 1;

if (a && b && c) {

d = a % c;

a += d;

}

// antipattern

// missing or inconsistent spaces make the code confusing

var d= 0,

a =b+1;

if (a&& b&&c) {

d=a %c;

a+= d;

}

[复制代码](javascript:void(0);)

And a final note about white space—curly braces spacing. It’s good to use a space:

• Before opening curly braces ({) in functions,  if-else cases, loops, and object literals

• Between the closing curly brace (}) and else or while