JavaScript CodeStyle

Online Code Beautifier

See the online mrdoobapproves code beautifier.

ESlint Plugin

This style guide is also available as an eslint plugin on NPM.

General Provisions

- Code should be in UTF-8.
- A new line should be LF.
- Indentation should be tabs.
- No extra spaces at the end of lines (set up your text editor, so that it removes extra spaces when saving).

Naming

Naming should be as descriptive as possible. The only exception is the indexing variable in a loop. That can be shortened to a single letter starting from i.

- variableNamesLikeThis
- functionNamesLikeThis
- ClassNamesLikeThis
- methodNamesLikeThis
- CONSTANTS_LIKE_THIS

Private properties and methods of objects begin with an underscore _.

Literals

Objects

- Object should always be created by `{}` and not `new Object()`.
- When creating an empty Object, use `{}` without spaces.
- When creating an Object with keys,
  - There should be a single space after the opening bracket.
  - There should be a single space before the closing bracket.
  - Keys should be written without quotes. The exception is when it is necessary.
  - There should be no space between the key word and the colon.
  - There should be a space between the colon and the value.
  - The comma should have no space before and 1 space behind itself.
  - The last value shouldn't be trailed with a comma.
  - The comma shouldn't start a new line. It should always trail the previous value.

good:

```
var obj = { A: 1, b: 2, C: 3 };

var obj = {
    A: 1,
    b: 2,
    C: 3
};
```

poor:

```
var obj = {A:1,b:2,C:3};

var obj = {A:1, b:2, C:3};

var obj = {A : 1, b : 2, C : 3};

var obj = { "A" : 1, "b" : 2, "C" : 3 };

var obj = { A : 1, b : 2, C : 3 };

var obj = { A :1, b :2, C :3 };

var obj = { A : 1 , b : 2 , C : 3 };

var obj = {
    A : 1,
    b : 2,
    C : 3,
};


var obj = {
    A : 1
  , b : 2
  , C : 3
};
```

Classes

- Private properties should start with an underscore.

Arrays

- Arrays should always be created by `[]` and not `new Array()`.
- When creating an empty Array, use `[]` without spaces.
- When creating an Array with values,

- There should be a single space after the opening bracket.
- There should be a single space before the closing bracket.
- The comma should have no space before and 1 space behind itself.
- The last value shouldn't be trailed with a comma.
- The comma shouldn't start a new line. It should always trail the previous value.

good:

```
var arr = [ 1, 2, 3 ];

var arr = [
    1,
    2,
    3
];
```

poor:

```
var arr = [1,2,3];

var arr = [1, 2, 3];

var arr = [ 1 , 2 , 3 ];

var arr = [
    1,
    2,
    3,
];

var arr = [
    1
  , 2
  , 3
];
```

Strings

Strings are written using single quotes:

good:

```
var  lyrics  =  'Never gonna Give you up, Never gonna Let you down' ;
```

Semicolon

Semicolons are always placed.

Blocks

- The opening brackets should be followed by 1 empty line.
- The closing brackets should be behind 1 empty line.
- The opening brackets should always follow a space and not start at a new line

good:

```
if ( a === 0 ) {

   // this is good
   return true;

}
```

poor:

```
if ( a === 0 ) {
    // this is bad: missing empty line after '{'
    return true;

}

if ( a === 0 ) {

    // this is bad: missing empty line before '}'
    return true;
}

if ( a === 0 ) { // this is bad:  stuff after '{'

    return true;

}

if ( a === 0 ){

   // this is bad: no space before '{'
   return true;

}
```

Conditional instructions

if statement

- The if keyword should always be followed with a space, an opening parenthesis and another space.

- The test should end with a space, a closing parenthesis and another space.
- The if statement should always contain a block.
- If there is an else statement, it should be on the same line as the closing bracket of the block.
- The else statement is followed by another block and should be separated from both blocks with a single space on both sides.
- Assignment should not be used in a test.

good:

```
if ( test ) {

    // ...

} else {

    // ...

}
```

poor:

```
if (test) {

    // ...

}

if( test ) {

    // ...

}

if (test ) {

    // ...

}

if ( test) {

    // ...

}

if ( test ){
```

```
    // ...

}

if ( test ) {

    // ...

}else {

    // ...

}

if ( test ) {

    // ...

} else{

    // ...

}
```

switch

- The switch keyword should always be followed with a space, an opening parenthesis and another space.
- The test value should end with a space, a closing parenthesis and another space.
- Every case should have a break statement, except for the default case or when returning a value.
- Every case should have a space between the colon and the test.
- Before and after each case, there should be a new line.
- Assignment should not be used in a test.

good:

```
switch ( value ) {

    case 1 :

        // ...
        break ;

    case 2 :
```

```
        // ...
        break ;

    default :

        // ...
        // no break keyword on the last case

}
```

Cycles

for

- Be aware that `foreach` and `for in` are much slower than `for`.
- The for keyword should always be followed with a space, an opening parenthesis and another space.
- The assignment part should start with a space.
- The assignment part should end with a semicolon and a space.
- The test part should start with a space.
- The test part should end with a semicolon and a space.
- The update part should start with a space.
- The update part should end with a space, a closing parenthesis and another space

Operators

with

Operator `with` not used.

The equality operator

Always use strict equality `===` (inequality `!==`).

Ternary operator

Always use spaces around the colon and question mark.

Unary

All unary operators are written together with the following operands:

var foo = ! bar;

eval

Avoid using eval. To parse json, use JSON.parse.

undefined

Check the value through a strict comparison.

Good:

```
x === undefined;
```

Poor:

```
/ / In modern browsers already defined immutable undefined.
var undefined;
x === undefined;

typeof x === 'undefined'

x === void 0
```