

# Essentials of Javascript

Visit the Cultural View of Technology [JavaScript Tutorial page](#) for videos and exercises

# Contents

## Articles

JavaScript	1
JavaScript syntax	18
JavaScript Style Sheets	43
JavaScript engine	44
Ajax (programming)	48
AJAX.OOP	51
?:	52
Appcelerator Titanium	57
AppengineJS	58
BSON	59
Bookmarklet	60
Client-side JavaScript	62
CommonJS	66
Comparison of JavaScript frameworks	68
Comparison of JavaScript-based source code editors	74
Douglas Crockford	80
DWR (Java)	82
EMVC	84
Brendan Eich	85
JS/UIX	86
JSAN	87
JSDoc	88
JSLint	91
JSSP	92
JScript	93
JavascriptMVC	96
JSON	98
JsonML	107
Lightbox (JavaScript)	110
Lively Kernel	112
Log4javascript	117
Medireview	117
Minification (programming)	118
Objective-J	120

John Resig	122
Reverse Ajax	123
Rico (Ajax)	124
Seed (programming)	126
Server-side JavaScript	129
Comparison of Server-side JavaScript solutions	130
SproutCore	135
Unobtrusive JavaScript	136
Venkman	139
XMLHttpRequest	141


## References


Article Sources and Contributors	147
Image Sources, Licenses and Contributors	151

## Article Licenses

License	152
---------	-----

# JavaScript

<b>Paradigm</b>	Multi-paradigm: prototype-based, functional, imperative, scripting
<b>Appeared in</b>	1995
<b>Designed by</b>	Brendan Eich
<b>Developer</b>	Netscape Communications Corporation, Mozilla Foundation
<b>Stable release</b>	1.8.2 <sup>[1]</sup> (June 22, 2009)
<b>Preview release</b>	1.9.3 <sup>[2]</sup> (June 14, 2010)
<b>Typing discipline</b>	dynamic, weak, duck
<b>Major implementations</b>	KJS, Rhino, SpiderMonkey, V8, WebKit
<b>Influenced by</b>	Self, C, Scheme, Perl, Python, Java
<b>Influenced</b>	JScript, JScript .NET, Objective-J, TIScript
 JavaScript at Wikibooks	

	
<b>Filename extension</b>	.js
<b>Internet media type</b>	application/javascript,text/javascript <sup>[3]</sup>
<b>Uniform Type Identifier</b>	com.netscape.javascriptsource <sup>[4]</sup>
<b>Type of format</b>	Scripting language

**This article is part of the JavaScript series.**

JavaScript
JavaScript syntax
JavaScript topics

**JavaScript** is an implementation of the ECMAScript language standard and is typically used to enable programmatic access to computational objects within a host environment. It can be characterized as a prototype-based object-oriented<sup>[5]</sup> scripting language that is dynamic, weakly typed and has first-class functions. It is also considered a functional programming language<sup>[6]</sup> like Scheme and OCaml because it has closures and supports higher-order functions.<sup>[7]</sup>

JavaScript is primarily used in the form of client-side JavaScript, implemented as part of a web browser in order to provide enhanced user interfaces and dynamic websites. However, its use in applications outside web pages is also significant.

JavaScript and the Java programming language both use syntaxes influenced by that of C syntax, and JavaScript copies many Java names and naming conventions; but the two languages are otherwise unrelated and have very different semantics. The key design principles within JavaScript are taken from the Self and Scheme programming languages.<sup>[8]</sup>

## History

Anyway I know only one programming language worse than C and that is Javascript. [...] I was convinced that we needed to build-in a programming language, but the developers, Tim first, were very much opposed. It had to remain completely declarative. Maybe, but the net result is that the programming-vacuum filled itself with the most horrible kluge in the history of computing: Javascript.

Robert Cailliau<sup>[9]</sup>

JavaScript was originally developed by Brendan Eich of Netscape under the name *Mocha*, which was later renamed to *LiveScript*, and finally to JavaScript.<sup>[10]</sup><sup>[11]</sup> LiveScript was the official name for the language when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it was renamed JavaScript in a joint announcement with Sun Microsystems on December 4, 1995<sup>[12]</sup> when it was deployed in the Netscape browser version 2.0B3.<sup>[13]</sup>

The change of name from LiveScript to JavaScript roughly coincided with Netscape adding support for Java technology in its Netscape Navigator web browser. The final choice of name caused confusion, giving the impression that the language was a spin-off of the Java programming language, and the choice has been characterized by many as a marketing ploy by Netscape to give JavaScript the cachet of what was then the hot new web-programming language.<sup>[14]</sup><sup>[15]</sup> It has also been claimed that the language's name is the result of a co-marketing deal between Netscape and Sun, in exchange for Netscape bundling Sun's Java runtime with their then-dominant browser.

JavaScript very quickly gained widespread success as a client-side scripting language for web pages. As a consequence, Microsoft developed a compatible dialect of the language, naming it JScript to avoid trademark issues. JScript added new date methods to fix the non-Y2K-friendly methods in JavaScript, which were based on `java.util.Date`<sup>[16]</sup><sup>[17]</sup> JScript was included in Internet Explorer 3.0, released in August 1996. The dialects are perceived to be so similar that the terms "JavaScript" and "JScript" are often used interchangeably. (Microsoft, however, notes dozens of ways in which JScript is not ECMA-compliant.<sup>[18]</sup> )

In November, 1996 Netscape announced that it had submitted JavaScript to Ecma International for consideration as an industry standard, and subsequent work resulted in the standardized version named ECMAScript.<sup>[19]</sup>

JavaScript has become one of the most popular programming languages on the web. Initially, however, many professional programmers denigrated the language because its target audience was web authors and other such "amateurs", among other reasons.<sup>[20]</sup> The advent of Ajax returned JavaScript to the spotlight and brought more professional programming attention. The result was a proliferation of comprehensive frameworks and libraries, improved JavaScript programming practices, and increased usage of JavaScript outside of web browsers, as seen by the proliferation of server-side JavaScript platforms.

In January 2009 the CommonJS project was founded with the goal of specifying a common standard library mainly for JavaScript development outside the browser.<sup>[21]</sup>

## Trademark

"JavaScript" is a trademark of Sun Microsystems. It is used under license for technology invented and implemented by Netscape Communications and current entities such as the Mozilla Foundation.<sup>[22]</sup>

## Features

The following features are common to all conforming ECMAScript implementations, unless explicitly specified otherwise.

### Imperative and structured

JavaScript supports all the structured programming syntax in C (e.g., if statements, while loops, switch statements, etc.). One partial exception is scoping: C-style block-level scoping is not supported (instead, JavaScript has function-level scoping). JavaScript 1.7, however, supports block-level scoping with the `let` keyword. Like C, JavaScript makes a distinction between expressions and statements. One syntactic difference from C is automatic semicolon insertion, in which the semicolons that terminate statements can be omitted.<sup>[23]</sup>

### Dynamic

#### dynamic typing

As in most scripting languages, types are associated with values, not variables. For example, a variable `x` could be bound to a number, then later rebound to a string. JavaScript supports various ways to test the type of an object, including duck typing.<sup>[24]</sup>

#### object based

JavaScript is almost entirely object-based. JavaScript objects are associative arrays, augmented with prototypes (see below). Object property names are string keys: `obj.x = 10` and `obj["x"] = 10` are equivalent, the dot notation being syntactic sugar. Properties and their values can be added, changed, or deleted at run-time. Most properties of an object (and those on its prototype inheritance chain) can be enumerated using a `for...in` loop. JavaScript has a small number of built-in objects such as `Function` and `Date`.

#### run-time evaluation

JavaScript includes an `eval` function that can execute statements provided as strings at run-time.

## Functional

#### first-class functions

Functions are first-class; they are objects themselves. As such, they have properties and can be passed around and interacted with like any other object.

#### inner functions and closures

Inner functions (functions defined within other functions) are created each time the outer function is invoked, and variables of the outer functions for that invocation continue to exist as long as the inner functions still exist, even after that invocation is finished (e.g. if the inner function was returned, it still has access to the outer function's variables) — this is the mechanism behind closures within JavaScript.

---

## Prototype-based

### prototypes

JavaScript uses prototypes instead of classes for inheritance. It is possible to simulate many class-based features with prototypes in JavaScript.

### functions as object constructors

Functions double as object constructors along with their typical role. Prefixing a function call with `new` creates a new object and calls that function with its local `this` keyword bound to that object for that invocation. The constructor's `prototype` property determines the object used for the new object's internal prototype. JavaScript's built-in constructors, such as `Array`, also have prototypes that can be modified.

### functions as methods

Unlike many object-oriented languages, there is no distinction between a function definition and a method definition. Rather, the distinction occurs during function calling; a function can be called as a method. When a function is called as a method of an object, the function's local `this` keyword is bound to that object for that invocation.

## Miscellaneous

### run-time environment

JavaScript typically relies on a run-time environment (e.g. in a web browser) to provide objects and methods by which scripts can interact with "the outside world". In fact, it relies on the environment to provide the ability to include/import scripts (e.g. HTML `<script>` elements). (This is not a language feature per se, but it is common in most JavaScript implementations.)

### variadic functions

An indefinite number of parameters can be passed to a function. The function can access them through formal parameters and also through the local `arguments` object.

### array and object literals

Like many scripting languages, arrays and objects (associative arrays in other languages) can each be created with a succinct shortcut syntax. In fact, these literals form the basis of the JSON data format.

### regular expressions

JavaScript also supports regular expressions in a manner similar to Perl, which provide a concise and powerful syntax for text manipulation that is more sophisticated than the built-in string functions.

## Vendor-specific extensions

JavaScript is officially managed by Mozilla Foundation, and new language features are added periodically. However, only some non-Mozilla JavaScript engines support these new features:

- property getter and setter functions (also supported by WebKit, Opera,<sup>[25]</sup> ActionScript, and Rhino)<sup>[26]</sup>
  - conditional catch clauses
  - iterator protocol adopted from Python
  - shallow generators/coroutines also adopted from Python
  - array comprehensions and generator expressions also adopted from Python
  - proper block scope via new `let` keyword
  - array and object destructuring (limited form of pattern matching)
  - concise function expressions (`function(args) expr`)
  - ECMAScript for XML (E4X), an extension that adds native XML support to ECMAScript
-

## Syntax and semantics

As of 2009, the latest version of the language is JavaScript 1.8.1. It is a superset of ECMAScript (ECMA-262) Edition 3. Extensions to the language, including partial E4X (ECMA-357) support and experimental features considered for inclusion into future ECMAScript editions, are documented here.<sup>[27]</sup>

### Example - syntax and semantics

This sample code showcases various JavaScript features. The example can be executed with the following steps: (1) Copy the code to a file with extension .htm. (2) Use the Mozilla or Firefox browser to open the file.

```
<html>
  <head><title>LCM Calculator</title></head>
  <body>
    <font face="Courier New" size="3">
    <script type="text/javascript">
      /* Finds the lowest common multiple of two numbers */
      function LCMCalculator(x, y) { // constructor function
        function checkInt(x) { // inner function
          if (x % 1 !== 0)
            throw new TypeError(x + " is not an integer"); // exception
        }
        //semicolons are optional (but beware since this may cause
        //erroneously treated as a single statement)
        this.a = checkInt(x)
        this.b = checkInt(y)
      }
      // The prototype of object instances created by a constructor is
      // that constructor's "prototype" property.
      LCMCalculator.prototype = { // object literal
        gcd : function() { // method that calculates the greatest common
          // Euclidean algorithm:
          var a = Math.abs(this.a), b = Math.abs(this.b), t;
          if (a < b) {
            t = b; b = a; a = t; // swap variables
          }
          while (b !== 0) {
            t = b;
            b = a % b;
            a = t;
          }
          // Only need to calculate gcd once, so "redefine" this method.
          // (Actually not redefinition - it's defined on the instance
          // so that this.gcd refers to this "redefinition" instead of
          itself,
```



```

LCMCalculator.prototype.gcd.)
    // Also, 'gcd' = "gcd", this['gcd'] = this.gcd
    this['gcd'] = function() { return a; };
    return a;
},
    "lcm" /* can use strings here */: function() {
        // Variable names don't collide with object properties, e.g.
||lcm|| is not |this.lcm|.
        // not using |this.a * this.b| to avoid FP precision issues
        var lcm = this.a / this.gcd() * this.b;
        // Only need to calculate lcm once, so "redefine" this method.
        this.lcm = function() { return lcm; };
        return lcm;
    },
    toString : function() {
        return "LCMCalculator: a = " + this.a + ", b = " + this.b;
    }
};

// Note: Array's map() and forEach() are predefined in JavaScript 1.6.
// They are currently not available in some major JavaScript engines
// such as Internet Explorer, but are available under Firefox.
// They are used here to demonstrate JavaScript's inherent functional
nature.

[[25,55],[21,56],[22,58],[28,56]].map(function(pair) { // array literal
    + mapping function
    return new LCMCalculator(pair[0], pair[1]);
}).sort(function(a, b) { // sort with this comparative function
    return a.lcm() - b.lcm();
}).forEach(function(obj) {
    /* Note: print() is a JS builtin function available in Mozilla's js
CLI;
    * It is functionally equivalent to Java's System.out.println().
    * Within a web browser, print() is a very different function
    * (opens the "Print Page" dialog),
    * so use something like document.write() or alert() instead.
    */
    // print        (obj + ", gcd = " + obj.gcd() + ", lcm = " +
obj.lcm());
    // alert      (obj + ", gcd = " + obj.gcd() + ", lcm = " +
obj.lcm());
    document.write(obj + ", gcd = " + obj.gcd() + ", lcm = " +
obj.lcm() + "<br>");
});

</script>

```

```
<noscript>
(Message from Wikipedia example) <br>
Your browser either does not support JavaScript, or you have JavaScript
turned off.
</noscript>
</body>
</html>
```

The following output should be displayed in the browser window.

```
LCMCalculator: a = 28, b = 56, gcd = 28, lcm = 56
LCMCalculator: a = 21, b = 56, gcd = 7, lcm = 168
LCMCalculator: a = 25, b = 55, gcd = 5, lcm = 275
LCMCalculator: a = 22, b = 58, gcd = 2, lcm = 638
```

If Internet Explorer is used, the example will generate an error. Hence the example illustrates a point discussed further in the body of the article, namely, the JavaScript interpreter supported by Firefox executes source differently from the JScript interpreter supported by Internet Explorer. (See comments in the source code for details on the relevant differences for this example.)

## Use in web pages

The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and that interact with the Document Object Model (DOM) of the page. Some simple examples of this usage are:

- Opening or popping up a new window with programmatic control over the size, position, and attributes of the new window (e.g. whether the menus, toolbars, etc. are visible).
- Validating input values of a web form to make sure that they are acceptable before being submitted to the server.
- Changing images as the mouse cursor moves over them: This effect is often used to draw the user's attention to important links displayed as graphical elements.

Because JavaScript code can run locally in a user's browser (rather than on a remote server), the browser can respond to user actions quickly, making an application more responsive. Furthermore, JavaScript code can detect user actions which HTML alone cannot, such as individual keystrokes. Applications such as Gmail take advantage of this: much of the user-interface logic is written in JavaScript, and JavaScript dispatches requests for information (such as the content of an e-mail message) to the server. The wider trend of Ajax programming similarly exploits this strength.

A JavaScript engine (also known as *JavaScript interpreter* or *JavaScript implementation*) is an interpreter that interprets JavaScript source code and executes the script accordingly. The first JavaScript engine was created by Brendan Eich at Netscape Communications Corporation, for the Netscape Navigator web browser. The engine, code-named SpiderMonkey, is implemented in C. It has since been updated (in JavaScript 1.5) to conform to ECMA-262 Edition 3. The Rhino engine, created primarily by Norris Boyd (formerly of Netscape; now at Google) is a JavaScript implementation in Java. Rhino, like SpiderMonkey, is ECMA-262 Edition 3 compliant.

A web browser is by far the most common host environment for JavaScript. Web browsers typically use the public API to create "host objects" responsible for reflecting the DOM into JavaScript. The web server is another common application of the engine. A JavaScript webserver would expose host objects representing an HTTP request and response objects, which a JavaScript program could then manipulate to dynamically generate web pages.

Because JavaScript is the only language that the most popular browsers share support for, it has become a target language for many frameworks in other languages, even though JavaScript was never intended to be such a language.<sup>[11]</sup> Despite the performance limitations inherent to its dynamic nature, the increasing speed of JavaScript engines has made the language a surprisingly feasible compilation target.

## Example - use in web pages

A minimal example of a standards-conforming web page containing JavaScript (using HTML 4.01 syntax) would be the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head><title>simple page</title></head>
  <body>
    <script type="text/javascript">
      document.write('Hello World!');
    </script>
    <noscript>
      Your browser either does not support JavaScript, or you have JavaScript turned off.
    </noscript>
  </body>
</html>
```

## Compatibility considerations

Since JavaScript runs in widely varying environments, an important part of testing and debugging it is testing across browsers.

The DOM interfaces for manipulating web pages are not part of the ECMAScript standard, or of JavaScript itself. Officially, they are defined by a separate standardization effort by the W3C; in practice, browser implementations differ from the standards and from each other, and not all browsers execute JavaScript.

To deal with these differences, JavaScript authors can attempt to write standards-compliant code which will also be executed correctly by most browsers; failing that, they can write code that checks for the presence of certain browser features and behaves differently if they are not available.<sup>[28]</sup> In some cases, two browsers may both implement a feature but with different behavior, and authors may find it practical to detect what browser is running and change their script's behavior to match.<sup>[29]</sup><sup>[30]</sup> Programmers may also use libraries or toolkits which take browser differences into account.

Furthermore, scripts may not work for some users. For example, a user may:

- use an old or rare browser with incomplete or unusual DOM support,
- use a PDA or mobile phone browser which cannot execute JavaScript,
- have JavaScript execution disabled as a security precaution,
- use a speech browser due to, for example, a visual disability.

To support these users, web authors can try to create pages which degrade gracefully on user agents (browsers) which do not support the page's JavaScript. In particular, the page should remain usable albeit without the extra features that the JavaScript would have added.

## Accessibility

Assuming that the user has not disabled its execution, client-side web JavaScript should be written to enhance the experiences of visitors with visual or physical disabilities, and certainly should avoid denying information to these visitors.<sup>[31]</sup>

Screen readers, used by the blind and partially sighted, can be JavaScript-aware and so may access and read the page DOM after the script has altered it. The HTML should be as concise, navigable and semantically rich as possible whether the scripts have run or not. JavaScript should not be totally reliant on mouse-specific events so as to deny its benefits to users who either cannot use a mouse or who choose to favor the keyboard for whatever reason. Equally, although hyperlinks and webforms can be navigated and operated from the keyboard, accessible JavaScript should not require keyboard events either. There are device-independent events such as `onfocus` and `onchange` that are preferable in most cases.<sup>[31]</sup>

JavaScript should not be used in a way that is confusing or disorientating to any web user. For example, using script to alter or disable the normal functionality of the browser, such as by changing the way the back-button or the refresh event work, is usually best avoided. Equally, triggering events that the user may not be aware of reduces the user's sense of control as do unexpected scripted changes to the page content.<sup>[32]</sup>

Often the process of making a complex web page as accessible as possible becomes a nontrivial problem where issues become matters of debate and opinion, and where compromises are necessary in the end. However, user agents and assistive technologies are constantly evolving and new guidelines and relevant information are continually being published on the web.<sup>[31]</sup>

## Security

JavaScript and the DOM provide the potential for malicious authors to deliver scripts to run on a client computer via the web. Browser authors contain this risk using two restrictions. First, scripts run in a sandbox in which they can only perform web-related actions, not general-purpose programming tasks like creating files. Second, scripts are constrained by the same origin policy: scripts from one web site do not have access to information such as usernames, passwords, or cookies sent to another site. Most JavaScript-related security bugs are breaches of either the same origin policy or the sandbox.

### Cross-site vulnerabilities

A common JavaScript-related security problem is cross-site scripting, or XSS, a violation of the same-origin policy. XSS vulnerabilities occur when an attacker is able to cause a target web site, such as an online banking website, to include a malicious script in the webpage presented to a victim. The script in this example can then access the banking application with the privileges of the victim, potentially disclosing secret information or transferring money without the victim's authorization. A solution to XSS vulnerabilities is to use *HTML escaping* whenever displaying untrusted data.

Some browsers include partial protection against *reflected* XSS attacks, in which the attacker provides a URL including malicious script. However, even users of those browsers are vulnerable to other XSS attacks, such as those where the malicious code is stored in a database. Only correct design of Web applications on the server side can fully prevent XSS.

XSS vulnerabilities can also occur because of implementation mistakes by browser authors.<sup>[33]</sup>

Another cross-site vulnerability is cross-site request forgery or CSRF. In CSRF, code on an attacker's site tricks the victim's browser into taking actions the user didn't intend at a target site (like transferring money at a bank). It works because, if the target site relies only on cookies to authenticate requests, then requests initiated by code on the attacker's site will carry the same legitimate login credentials as requests initiated by the user. In general, the solution to CSRF is to require an authentication value in a hidden form field, and not only in the cookies, to authenticate any

request that might have lasting effects. Checking the HTTP Referrer header can also help.

"JavaScript hijacking" is a type of CSRF attack in which a `<script>` tag on an attacker's site exploits a page on the victim's site that returns private information as JSON or JavaScript. Possible solutions include requiring an authentication token in the POST and GET parameters for any response that returns private JSON (even if it has no side effects); using POST and never GET for requests that return private JSON; and modifying the response so that it can't be used via a `<script>` tag (by, for example, wrapping the JSON in a JavaScript comment).

### Misplaced trust in the client

Client-server applications, whether they involve JavaScript or not, must recognize that untrusted clients may be under the control of attackers. Thus any secret embedded in JavaScript could be extracted by a determined adversary, and the application author can't assume that his JavaScript runs as intended, or at all. Some implications:

- Web site authors cannot perfectly conceal how their JavaScript operates, because the code is sent to the client, and obfuscated code can be reverse-engineered.
- JavaScript form validation only provides convenience for users, not security. If a site verifies that the user agreed to its terms of service, or filters invalid characters out of fields that should only contain numbers, it must do so on the server, not only the client.
- Scripts can be selectively disabled, so JavaScript can't be relied on to prevent operations such as "save image".<sup>[34]</sup>
- It would be extremely bad practice to embed a password in JavaScript (where it can be extracted by an attacker), then have JavaScript verify a user's password and pass "password\_ok=1" back to the server (since the "password\_ok=1" response is easy to forge).<sup>[35]</sup>

### Browser and plugin coding errors

JavaScript provides an interface to a wide range of browser capabilities, some of which may have flaws such as buffer overflows. These flaws can allow attackers to write scripts which would run any code they wish on the user's system.

These flaws have affected major browsers including Firefox,<sup>[36]</sup> Internet Explorer,<sup>[37]</sup> and Safari.<sup>[38]</sup>

Plugins, such as video players, Adobe Flash, and the wide range of ActiveX controls enabled by default in Microsoft Internet Explorer, may also have flaws exploitable via JavaScript, and such flaws have been exploited in the past.<sup>[39]</sup>  
<sup>[40]</sup>

In Windows Vista, Microsoft has attempted to contain the risks of bugs such as buffer overflows by running the Internet Explorer process with limited privileges.<sup>[41]</sup> Google Chrome similarly limits page renderers to an operating-system-enforced "sandbox."

### Sandbox implementation errors

Web browsers are capable of running JavaScript outside of the sandbox, with the privileges necessary to, for example, create or delete files. Of course, such privileges aren't meant to be granted to code from the web.

Incorrectly granting privileges to JavaScript from the web has played a role in vulnerabilities in both Internet Explorer<sup>[42]</sup> and Firefox.<sup>[43]</sup> In Windows XP Service Pack 2, Microsoft demoted JScript's privileges in Internet Explorer.<sup>[44]</sup>

Microsoft Windows allows JavaScript source files on a computer's hard drive to be launched as general-purpose, non-sandboxed programs. This makes JavaScript (like VBScript) a theoretically viable vector for a Trojan horse, although JavaScript Trojan horses are uncommon in practice.<sup>[45]</sup> (See Windows Script Host.)

## Uses outside web pages

In addition to web browsers and servers, JavaScript interpreters are embedded in a number of tools. Each of these applications provides its own object model which provides access to the host environment, with the core JavaScript language remaining mostly the same in each application.

### Embedded scripting language

- Apple's Dashboard Widgets, Microsoft's Gadgets, Yahoo! Widgets, Google Desktop Gadgets, Serence Klipfolio are implemented using JavaScript.
- Adobe's Acrobat and Adobe Reader (formerly Acrobat Reader) support JavaScript in PDF files.<sup>[46]</sup>
- Tools in the Adobe Creative Suite, including Photoshop, Illustrator, Dreamweaver and InDesign, allow scripting through JavaScript.
- OpenOffice.org office application suite allows for JavaScript as one of its scripting languages.
- The interactive music signal processing software Max/MSP released by Cycling '74, offers a JavaScript model of its environment for use by developers. It allows much more precise control than the default GUI-centric programming model.
- ECMAScript was included in the VRML97 standard for scripting nodes of VRML scene description files.
- Some high-end Philips universal remote panels, including TSU9600 and TSU9400, can be scripted using a JavaScript-based tool called ProntoScript.<sup>[47]</sup>
- Sphere is an open source and cross platform computer program designed primarily to make role-playing games that use JavaScript as a scripting language.
- The open-source Re-Animator<sup>[48]</sup> framework allows developing 2D sprite-based games using JavaScript and XML.
- Methabot is a web crawler that uses JavaScript as scripting language for custom filetype parsers and data extraction using E4X.
- The game engine Unity supports three scripting languages: JavaScript, C#, and Boo.<sup>[49]</sup>
- DX Studio (3D engine) uses the SpiderMonkey implementation of JavaScript for game and simulation logic.<sup>[50]</sup>
- Maxwell Render provides an ECMA standard based scripting engine for tasks automation.<sup>[51]</sup>
- Google Docs Spreadsheet has a script editor which allows users to create custom formulas, automate repetitive tasks and also interact with other Google products such as Gmail.<sup>[52]</sup>

### Scripting engine

- Microsoft's Active Scripting technology supports the JavaScript-compatible JScript as a scripting language used in HTML Applications.
- The Java programming language, in version SE 6 (JDK 1.6), introduced the javax.script package, including a JavaScript implementation based on Mozilla Rhino. Thus, Java applications can host scripts that access the application's variables and objects, much like web browsers host scripts that access the browser's Document Object Model (DOM) for a webpage.<sup>[53] [54]</sup>
- The Qt C++ toolkit includes a QtScript module to interpret JavaScript, analogous to javax.script.<sup>[55]</sup>
- Late Night Software's JavaScript OSA (aka JavaScript for OSA, or JSOSA), is a freeware alternative to AppleScript for Mac OS X. It is based on the Mozilla 1.5 JavaScript implementation, with the addition of a MacOS object for interaction with the operating system and third-party applications.<sup>[56]</sup>

## Application platform

- ActionScript, the programming language used in Adobe Flash, is another implementation of the ECMAScript standard.
- The Mozilla platform, which underlies Thunderbird, Firefox and some other web browsers, uses JavaScript to implement the graphical user interface (GUI) of its various products.
- Adobe Integrated Runtime is a JavaScript runtime that allows developers to create desktop applications.
- webOS uses the WebKit implementation of JavaScript as a part of its application framework.
- CA, Inc.'s AutoShell cross-application scripting environment is built on JavaScript/SpiderMonkey with preprocessor like extensions for command definitions and custom classes for various system related tasks like file i/o, operation system command invocation and redirection and COM scripting.
- GNOME Shell, the shell for the GNOME 3 desktop environment.<sup>[57]</sup> The Seed,<sup>[58]</sup> Gjs<sup>[59]</sup> (from Gnome) and Kjs<sup>[60][61]</sup> (from KDE) packages are aimed to utilize that needs.

## Development tools

Within JavaScript, access to a debugger becomes invaluable when developing large, non-trivial programs. Because there can be implementation differences between the various browsers (particularly within the Document Object Model) it is useful to have access to a debugger for each of the browsers that a web application targets.<sup>[62]</sup>

Script debuggers are available for Internet Explorer, Firefox, Safari, Google Chrome, and Opera.<sup>[63]</sup>

Three debuggers are available for Internet Explorer: Microsoft Visual Studio is the richest of the three, closely followed by Microsoft Script Editor (a component of Microsoft Office),<sup>[64]</sup> and finally the free Microsoft Script Debugger which is far more basic than the other two. The free Microsoft Visual Web Developer Express<sup>[65]</sup> provides a limited version of the JavaScript debugging functionality in Microsoft Visual Studio.

Web applications within Firefox can be debugged using the Firebug add-on, or the older Venkman debugger. Firefox also has a simpler built-in Error Console, which logs and evaluates JavaScript. It also logs CSS errors and warnings.

Opera includes a set of tools called DragonFly.<sup>[66]</sup>

WebKit's Web Inspector includes a JavaScript debugger<sup>[67]</sup> in Apple's Safari.

Some debugging aids are themselves written in JavaScript and built to run on the Web. An example is the program JSLint, developed by Douglas Crockford, currently senior JavaScript architect at Yahoo! who has written extensively on the language. JSLint scans JavaScript code for conformance to a set of standards and guidelines. Web development bookmarklets<sup>[68]</sup> and Firebug Lite<sup>[69]</sup> provide variations on the idea of the cross-browser JavaScript console.

## Versions

Version	Release date	Equivalent to	Netscape Navigator	Mozilla Firefox	Internet Explorer	Opera	Safari	Google Chrome
1.0	March 1996		2.0		3.0			
1.1	August 1996		3.0					
1.2	June 1997		4.0-4.05					
1.3	October 1998	ECMA-262 1 <sup>st</sup> edition / ECMA-262 2 <sup>nd</sup> edition	4.06-4.7x		4.0			
1.4			Netscape Server					

1.5	November 2000	ECMA-262 3 <sup>rd</sup> edition	6.0	1.0	5.5 (JScript 5.5), 6 (JScript 5.6), 7 (JScript 5.7), 8 (JScript 5.8)	6.0, 7.0, 8.0, 9.0, 10.0	3.0, 3.1, 3.2, 4.0	1.0
1.6	November 2005	1.5 + Array extras + Array and String generics + E4X		1.5				
1.7	October 2006	1.6 + Pythonic generators + Iterators + let		2.0				
1.8	June 2008	1.7 + Generator expressions + Expression closures		3.0				
1.8.1		1.8 + Native JSON support + Minor Updates		3.5				
1.8.2		1.8.1 + Minor updates		3.6				
1.9		1.8.1 + ECMAScript 5 Compliance		4				

[70]

## Related languages and features

Since the acceptance of JavaScript as a popular language, several languages and features have developed from it. Objective-J is a strict superset of JavaScript that adds traditional inheritance and Smalltalk/Objective-C style dynamic dispatch and optional pseudo-static typing to pure JavaScript.

TIScript is a superset of JavaScript that adds classes, namespaces and lambda expressions.

JSON, or JavaScript Object Notation, is a general-purpose data interchange format that is defined as a subset of JavaScript.

Mozilla browsers currently support LiveConnect, a feature that allows JavaScript and Java to intercommunicate on the web. However, Mozilla-specific support for LiveConnect is scheduled to be phased out in the future in favor of passing on the LiveConnect handling via NPAPI to the Java 1.6+ plug-in (not yet supported on the Mac as of March 2010).<sup>[71]</sup>

## JavaScript and Java

A common misconception is that JavaScript is similar or closely related to Java. It is true that both have a C-like syntax, the C language being their most immediate common ancestor language. They are both object-oriented, typically sandboxed (when used inside a browser), and are widely used in client-side Web applications. In addition, JavaScript was designed with Java's syntax and standard library in mind. In particular, all Java keywords are reserved in JavaScript, JavaScript's standard library follows Java's naming conventions, and JavaScript's Math and Date objects are based on classes from Java 1.0.<sup>[17]</sup>

But the similarities end there. Java has static typing; JavaScript's typing is dynamic (meaning a variable can hold an object of any type and cannot be restricted). Java is loaded from compiled bytecode; JavaScript is loaded as human-readable source code. Java's objects are class-based; JavaScript's are prototype-based. JavaScript also has many functional features based on the Self language.



## See also

- Client-side JavaScript
  - Ajax
  - Comparison of JavaScript frameworks
  - Dynamic HTML
- Comparison of layout engines (ECMAScript)
- Comparison of JavaScript-based source code editors
- ECMAScript
- JavaScript engine (Discussion of JavaScript engines (interpreters) with list of engines)
- JavaScript OSA - A system-level scripting language for the Apple Macintosh
- JavaScript syntax
- JSAN
- JScript
- JSDoc
- JSLint
- JSON
- List of ECMAScript engines
- Server-side JavaScript

## Further reading

- Bhargal, Sham; Jankowski, Tomasz (2003). *Foundation Web Design: Essential HTML, JavaScript, CSS, PhotoShop, Fireworks, and Flash*. APress L. P.. ISBN 1-59059-152-6.
- Burns, Joe; Growney, Andree S. (2001). *JavaScript Goodies*. Pearson Education. ISBN 0-7897-2612-2.
- Duffy, Scott (2003). *How to do Everything with JavaScript*. Osborne. ISBN 0-07-222887-3.
- Flanagan, David; Ferguson, Paula (2002). *JavaScript: The Definitive Guide* (4th ed.). O'Reilly & Associates. ISBN 0-596-00048-0.
- Flanagan, David (2006). *JavaScript: The Definitive Guide* (5th ed.). O'Reilly & Associates. ISBN 0-596-10199-6.
- Goodman, Danny; Markel, Scott (2003). *JavaScript and DHTML Cookbook*. O'Reilly & Associates. ISBN 0-596-00467-2.
- Goodman, Danny; Eich, Brendan (2001). *JavaScript Bible*. John Wiley & Sons. ISBN ISBN 0-7645-3342-8.
- Harris, Andy (2001). *JavaScript Programming for the Absolute Beginner*. Premier Press. ISBN 0-7615-3410-5.
- Heinle, Nick; Koman, Richard (1997). *Designing with JavaScript*. O'Reilly & Associates. ISBN 1-56592-300-6.
- McDuffie, Tina Spain (2003). *JavaScript Concepts & Techniques: Programming Interactive Web Sites*. Franklin, Beedle & Associates. ISBN 1-887-90269-4.
- McFarlane, Nigel (2003). *Rapid Application Development with Mozilla*. Prentice Hall Professional Technical References. ISBN 0-13-142343-6.
- Powell, Thomas A.; Schneider, Fritz (2001). *JavaScript: The Complete Reference*. McGraw-Hill Companies. ISBN 0-07-219127-9.
- Shelly, Gary B.; Cashman, Thomas J.; Dorin, William J.; Quasney, Jeffrey J. (2000). *JavaScript: Complete Concepts and Techniques*. Cambridge: Course Technology. ISBN 0-7895-6233-2.
- Watt, Andrew H.; Watt, Jonathan A.; Simon, Jinjer L. (2002). *Teach Yourself JavaScript in 21 Days*. Pearson Education. ISBN 0-672-32297-8.
- Vander Veer, Emily A. (2004). *JavaScript For Dummies* (4th ed.). Wiley Pub.. ISBN 0-7645-7659-3.

## External links

- Douglas Crockford's video lectures on Javascript<sup>[72]</sup>
- FAQ for Usenet's comp.lang.javascript<sup>[73]</sup>
- JavaScript<sup>[74]</sup> at the Open Directory Project
- Mozilla Developer Center
  - Mozilla's Official Documentation on JavaScript<sup>[75]</sup>
  - References for Core JavaScript versions: 1.5+<sup>[76]</sup>
  - New in JavaScript: 1.5<sup>[77]</sup>, 1.6<sup>[78]</sup>, 1.7<sup>[79]</sup>, 1.8<sup>[80]</sup>, 1.8.1<sup>[81]</sup>
  - List of JavaScript releases: versions 1.5+<sup>[82]</sup>
  - Re-Introduction to JavaScript<sup>[83]</sup>
- Programming languages implemented in JavaScript<sup>[84]</sup>

## References

- [1] Firefox 3.6 supports JavaScript 1.8.2 ([https://developer.mozilla.org/en/firefox\\_3.6\\_for\\_developers#JavaScript](https://developer.mozilla.org/en/firefox_3.6_for_developers#JavaScript))
- [2] Mozilla.org (<http://www.mozilla.org/projects/devpreview/releasenotes/>)
- [3] RFC 4329 (<http://www.apps.ietf.org/rfc/rfc4329.html>)
- [4] "System-Declared Uniform Type Identifiers" (<http://developer.apple.com/mac/library/documentation/Miscellaneous/Reference/UTITRef/Articles/System-DeclaredUniformTypeIdentifiers.html>). *Mac OS X Reference Library*. Apple Inc.. Retrieved 2010-03-05.
- [5] "ECMAScript Language Specification" (<http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-262.pdf>). .
- [6] Douglas Crockford on Functional JavaScript ([http://www.blinkx.com/video/douglas-crockford-on-functional-javascript/xscZz8XhfuNQ\\_aaVuyUB2A](http://www.blinkx.com/video/douglas-crockford-on-functional-javascript/xscZz8XhfuNQ_aaVuyUB2A)) (2:49): "[JavaScript] is also coincidentally the world's most popular functional programming language. JavaScript is and has always been, at least since [version] 1.2, a functional programming language."
- [7] The Little JavaScripter (<http://www.crockford.com/javascript/little.html>) shows the relationship with Scheme in more detail.
- [8] "ECMAScript Language Overview" (<http://www.ecmascript.org/es4/spec/overview.pdf>) (PDF). 2007-10-23. pp. 4. . Retrieved 2009-05-03.
- [9] wikinews:Wikinews:Story preparation/Interview with Robert Cailliau
- [10] Krill, Paul (2008-06-23). "JavaScript creator ponders past, future" ([http://www.infoworld.com/article/08/06/23/eich-javascript-interview\\_1.html](http://www.infoworld.com/article/08/06/23/eich-javascript-interview_1.html)). InfoWorld. . Retrieved 2009-05-19.
- [11] Hamilton, Naomi (2008-06-31). "The A-Z of Programming Languages: JavaScript" ([http://www.computerworld.com.au/article/255293/-z\\_programming\\_languages\\_javascript](http://www.computerworld.com.au/article/255293/-z_programming_languages_javascript)). computerworld.com.au. .
- [12] Press release announcing JavaScript (<http://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/newsrelease67.html>), "Netscape and Sun announce Javascript(TM)", PR Newswire, Dec 4, 1995
- [13] "TechVision: Innovators of the Net: Brendan Eich and JavaScript" ([http://web.archive.org/web/20080208124612/http://wp.netscape.com/comprod/columns/techvision/innovators\\_be.html](http://web.archive.org/web/20080208124612/http://wp.netscape.com/comprod/columns/techvision/innovators_be.html)). Web.archive.org. Archived from the original on 2008-02-08. . Retrieved 2010-06-14.
- [14] "Programming languages used on the Internet and the World Wide Web (WWW)" ([http://www.webdevelopersnotes.com/basics/languages\\_on\\_the\\_internet.php3](http://www.webdevelopersnotes.com/basics/languages_on_the_internet.php3)). Webdevelopersnotes.com. . Retrieved 2009-05-19.
- [15] "O'Reilly - Safari Books Online - 0596101996 - JavaScript: The Definitive Guide, 5th Edition" (<http://safari.oreilly.com/0596101996/jscript5-CHP-1>). Safari.oreilly.com. . Retrieved 2009-05-19.
- [16] <http://java.sun.com/j2se/1.4.2/docs/api/java/util/Date.html>
- [17] "Brendan's Roadmap Updates: Popularity" (<http://weblogs.mozillazine.org/roadmap/archives/2008/04/popularity.html>). Weblogs.mozillazine.org. . Retrieved 2009-05-19.
- [18] Microsoft JScript Features - Non-ECMA (<http://msdn2.microsoft.com/en-us/library/4tc5a343.aspx>)
- [19] ECMAScript 3rd Edition specification (<http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>)
- [20] "JavaScript: The World's Most Misunderstood Programming Language" (<http://www.crockford.com/javascript/javascript.html>). Crockford.com. . Retrieved 2009-05-19.
- [21] Kris Kowal (1 December 2009). "CommonJS effort sets JavaScript on path for world domination" (<http://arstechnica.com/web/news/2009/12/commonjs-effort-sets-javascript-on-path-for-world-domination.ars>). *Ars Technica*. Condé Nast Publications. . Retrieved 18 April 2010.
- [22] "Sun Trademarks" (<http://www.sun.com/suntrademarks/>). Sun Microsystems. . Retrieved 2007-11-08.
- [23] Flanagan, David (2006). *JavaScript: The definitive Guide*. p. 16. ISBN 978-0-596-10199-2. "Omitting semicolons is not a good programming practice; you should get into the habit of inserting them."
- [24] Flanagan, David (2006). *JavaScript: The Definitive Guide*. O'Reilly Media. pp. 176–178. ISBN 0596101996.

- [25] Robert Nyman, Getters And Setters With JavaScript – Code Samples And Demos (<http://robertnyman.com/2009/05/28/getters-and-setters-with-javascript-code-samples-and-demos/>), published 29 May 2009, accessed 2 January 2010.
- [26] John Resig, JavaScript Getters and Setters (<http://ejohn.org/blog/javascript-getters-and-setters/>), 18 July 2007, accessed 2 January 2010
- [27] "About - MDC" ([https://developer.mozilla.org/en/Core\\_JavaScript\\_1.5\\_Reference:About](https://developer.mozilla.org/en/Core_JavaScript_1.5_Reference:About)). Developer.mozilla.org. 2008-08-31. . Retrieved 2009-05-19.
- [28] Peter-Paul Koch, Object detection (<http://www.quirksmode.org/js/support.html>)
- [29] Peter-Paul Koch, Mission Impossible - mouse position (<http://www.evolt.org/node/23335>)
- [30] Peter-Paul Koch, Browser detect (<http://www.quirksmode.org/js/detect.html>)
- [31] Flanagan, David (2006). *JavaScript: The definitive guide*. O'Reilly. pp. 262–263. ISBN 978-0-596-10199-2.
- [32] "Creating Accessible JavaScript" (<http://www.webaim.org/techniques/javascript/>). WebAIM. . Retrieved 8 June 2010.
- [33] MozillaZine, Mozilla Cross-Site Scripting Vulnerability Reported and Fixed (<http://www.mozillazine.org/talkback.html?article=4392>)
- [34] *Right-click "protection"? Forget about it* (<http://blog.anta.net/2008/06/17/right-click-â€œprotectionâ€•-forget-about-it/>). 2008-06-17. ISSN 1797-1993. . Retrieved 2008-06-17.
- [35] For an example of this bad practice, see Javascript.internet.com (<http://javascript.internet.com/passwords/>)
- [36] Mozilla Corporation, Buffer overflow in crypto.signText() (<http://www.mozilla.org/security/announce/2006/mfsa2006-38.html>)
- [37] Paul Festa, CNet, Buffer-overflow bug in IE (<http://news.com.com/2100-1001-214620.html>)
- [38] SecurityTracker.com, Apple Safari JavaScript Buffer Overflow Lets Remote Users Execute Arbitrary Code and HTTP Redirect Bug Lets Remote Users Access Files (<http://securitytracker.com/alerts/2006/Mar/1015713.html>)
- [39] SecurityFocus, Microsoft WebViewFolderIcon ActiveX Control Buffer Overflow Vulnerability (<http://www.securityfocus.com/bid/19030/info>)
- [40] Fusion Authority, Macromedia Flash ActiveX Buffer Overflow (<http://www.fusionauthority.com/security/3234-macromedia-flash-activex-buffer-overflow.htm>)
- [41] Mike Friedman, Protected Mode in Vista IE7 (<http://blogs.msdn.com/ie/archive/2006/02/09/528963.aspx>)
- [42] US CERT, Vulnerability Note VU#713878: Microsoft Internet Explorer does not properly validate source of redirected frame (<https://www.kb.cert.org/vuls/id/713878>)
- [43] Mozilla Foundation, Mozilla Foundation Security Advisory 2005-41: Privilege escalation via DOM property overrides (<http://www.mozilla.org/security/announce/2005/mfsa2005-41.html>)
- [44] Microsoft Corporation, Changes to Functionality in Microsoft Windows XP Service Pack 2: Part 5: Enhanced Browsing Security (<http://technet.microsoft.com/en-us/library/bb457150.aspx#EHAA>)
- [45] For one example of a rare JavaScript Trojan Horse, see Symantec Corporation, JS.Seeker.K ([http://www.symantec.com/security\\_response/writeup.jsp?docid=2003-100111-0931-99](http://www.symantec.com/security_response/writeup.jsp?docid=2003-100111-0931-99))
- [46] "JavaScript for Acrobat" (<http://www.adobe.com/devnet/acrobat/javascript.html>). . Retrieved 2009-08-18.
- [47] Koninklijke Philips Electronics NV (<http://www.pronto.philips.com/prontoscript/index.cfm?id=1422>)
- [48] <http://www.green-eyed-monster.com/reanimator/>
- [49] "Best Of All Worlds" (<http://unity3d.com/unity/features/scripting>). unity3d.com. . Retrieved 2009-09-12.
- [50] "Technical Specification" ([http://www.dxstudio.com/features\\_tech.aspx](http://www.dxstudio.com/features_tech.aspx)). dxstudio.com. . Retrieved 2009-10-20.
- [51] THINK! The Maxwell Render Resourcer Center, Scripting References ([http://think.maxwellrender.com/scripting\\_references-269.html](http://think.maxwellrender.com/scripting_references-269.html))
- [52] Google Apps Script, Welcome to Google Apps Script (<http://www.google.com/google-d-s/scripts/scripts.html>)
- [53] "javax.script release notes" (<http://java.sun.com/javase/6/webnotes/index.html#scripting>). Java.sun.com. . Retrieved 2009-05-19.
- [54] Flanagan 5th Edition, Pp 214 et seq
- [55] Nokia Corporation, QtScript Module (<http://doc.qt.nokia.com/4.6/qtscript.html>)
- [56] Open Scripting Architecture
- [57] "Behind the Scenes with Owen Taylor" (<http://gnomejournal.org/article/74/behind-the-scenes-with-owen-taylor>). The GNOME Journal. . Retrieved 2010-01-23.
- [58] Devel.akbkhomes.com (<http://devel.akbkhomes.com/seed/index.shtml>)
- [59] <http://live.gnome.org/Gjs>
- [60] <http://xmelegance.org/kjsembed/>
- [61] Xmelegance.org (<http://xmelegance.org/kjsembed/jsref/index.html>)
- [62] "Advanced Debugging With JavaScript" (<http://www.alistapart.com/articles/advanced-debugging-with-javascript/>). alistapart.com. 2009-02-03. . Retrieved 2010-05-28.
- [63] "The JavaScript Debugging Console" (<http://javascript.about.com/od/problemsolving/ig/JavaScript-Debugging/>). javascript.about.com. 2010-05-28. . Retrieved 2010-05-28.
- [64] JScript development in Microsoft Office 11 ([http://msdn2.microsoft.com/en-us/library/aa202668\(office.11\).aspx](http://msdn2.microsoft.com/en-us/library/aa202668(office.11).aspx)) (MS InfoPath 2003)
- [65] <http://www.microsoft.com/express/vwd/>
- [66] "Opera DragonFly" (<http://www.opera.com/dragonfly/>). Opera Software. .
- [67] "Introducing Drosera - Surfin' Safari" (<http://webkit.org/blog/61/introducing-drosera/>). Webkit.org. 2006-06-28. . Retrieved 2009-05-19.
- [68] <https://www.squarefree.com/bookmarklets/webdevel.html>
- [69] <http://getfirebug.com/lite.html>
- [70] John Resig, "Versions of JavaScript" (<http://ejohn.org/blog/versions-of-javascript>). Ejohn.org. . Retrieved 2009-05-19.

- [71] Java.sun.com (<http://java.sun.com/javase/6/webnotes/6u10/plugin2/liveconnect/>)
  - [72] <http://yuiblog.com/crockford/>
  - [73] <http://jibbering.com/faq/>
  - [74] <http://www.dmoz.org/Computers/Programming/Languages/JavaScript/>
  - [75] <https://developer.mozilla.org/en/JavaScript>
  - [76] [https://developer.mozilla.org/en/Core\\_JavaScript\\_1.5\\_Reference](https://developer.mozilla.org/en/Core_JavaScript_1.5_Reference)
  - [77] [https://developer.mozilla.org/en/New\\_in\\_JavaScript\\_1.5](https://developer.mozilla.org/en/New_in_JavaScript_1.5)
  - [78] [https://developer.mozilla.org/en/New\\_in\\_JavaScript\\_1.6](https://developer.mozilla.org/en/New_in_JavaScript_1.6)
  - [79] [https://developer.mozilla.org/en/New\\_in\\_JavaScript\\_1.7](https://developer.mozilla.org/en/New_in_JavaScript_1.7)
  - [80] [https://developer.mozilla.org/en/New\\_in\\_JavaScript\\_1.8](https://developer.mozilla.org/en/New_in_JavaScript_1.8)
  - [81] [https://developer.mozilla.org/en/New\\_in\\_JavaScript\\_1.8.1](https://developer.mozilla.org/en/New_in_JavaScript_1.8.1)
  - [82] [https://developer.mozilla.org/en/Core\\_JavaScript\\_1.5\\_Reference/About](https://developer.mozilla.org/en/Core_JavaScript_1.5_Reference/About)
  - [83] [https://developer.mozilla.org/en/A\\_re-introduction\\_to\\_JavaScript](https://developer.mozilla.org/en/A_re-introduction_to_JavaScript)
  - [84] <http://www.is-research.de/info/jslanguages/>
-

# JavaScript syntax

---

This article is part of the JavaScript series.
JavaScript
JavaScript syntax
JavaScript topics

The syntax of JavaScript is the set of rules that define a correctly structured JavaScript program.

**Note:** The examples below make use of the `alert` function for standard text output. The JavaScript standard library lacks an official standard text output function. However, given that JavaScript is mainly used for client-side scripting within modern web browsers, and that almost all web browsers provide the `alert` function, `alert` is used in the examples.

## Origin of Syntax

Brendan Eich summarized the ancestry of the syntax in the first paragraph of the JavaScript 1.1 <sup>[1]</sup> specification as follows:

JavaScript borrows most of its syntax from Java, but also inherits from Awk and Perl, with some indirect influence from Self in its object prototype system.

## Syntax Basics

### Case sensitivity

JavaScript is case sensitive. It is common to start the name of a constructor with a capitalised letter, and the name of a function or variable with a lower-case letter.

### Whitespace and semicolons

Spaces, tabs and newlines used outside of string constants are called whitespace. Unlike C, whitespace in JavaScript source can directly impact semantics. Because of a technique called "semicolon insertion", some statements that are well formed when a newline is parsed will be considered complete (as if a semicolon were inserted just prior to the newline). Programmers are advised to supply statement-terminating semicolons explicitly, although it degrades readability, because it may lessen unintended effects of the automatic semicolon insertion.<sup>[2]</sup>

**return**

```
a + b;
```

```
// Returns undefined. Treated as:
```

```
// return;
```

```
// a + b;
```

But:

```
a = b + c
```

```
(d + e).foo()
```

```
// Treated as:
```

```
// a = b + c(d + e).foo();
```

## Comments

Comment syntax is the same as in C++ and many other languages.

```
// a short, one-line comment

/* this is a long, multi-line comment
   about my script. May it one day
   be great. */
```

Note that JavaScript explicitly forbids nesting of comments, e.g.

```
/* You can't do
   /* this */
*/
/* But you can
//Do this.
*/

// And you can /* also do this */

// As /* well/* as this */ */
```

## Variables

Variables in standard JavaScript have no type attached, and any value can be stored in any variable. Variables can be declared with a `var` statement. These variables are lexically scoped and once a variable is declared, it may be accessed anywhere inside the function where it is declared. Variables declared outside any function, and variables first used within functions without being declared with 'var', are global. Here is an example of variable declarations and global values:

```
x = 0; // A global variable
var y = 'Hello!'; // Another global variable

function f(){
  var z = 'foxes'; // A local variable
  twenty = 20; // Global because keyword var is not used
  return x; // We can use x here because it is global
}
// The value of z is no longer available
```

## Primitive data types

The JavaScript language provides a handful of primitive data types. Some of the primitive data types also provide a set of named values that represent the extents of the type boundaries. These named values are described within the appropriate sections below.

### Undefined

The value of "undefined" is assigned to all uninitialized variables, and is also returned when checking for object properties that do not exist. In a Boolean context, the undefined value is considered a false value.

Note: Undefined is considered a true primitive-type. As such, when performing checks that enforce type checking, the undefined value will not equal other false types.

```
var test;           // variable declared but not defined, ...
                    // ... set to value of undefined

var testObj = {};
alert(test);         // test variable exists but value not ...
                    // ... defined, displays undefined
alert(testObj.myProp); // testObj exists, property does not, ...
                    // ... displays undefined
alert(undefined == null); // unenforced type during check, displays
true
alert(undefined === null); // enforce type during check, displays false
```

Note: There is no built-in language literal for undefined. Thus `(x == undefined)` is not a foolproof way to check whether a variable is undefined, because it is legal for someone to write `var undefined = "I'm defined now";`. A more robust comparison can be made using `(typeof x == 'undefined')` or a function like this:

```
function isUndefined(x) { var u; return x === u; }
```

or

```
function isUndefined(x) { return x === void(0); }
```

### Null

Unlike undefined, null is often set to indicate that something has been declared but has been defined to be empty. In a Boolean context, the value of null is considered a false value in JavaScript.

Note: Null is a true primitive-type within the JavaScript language, of which **null** (note case) is the single value. As such, when performing checks that enforce type checking, the null value will not equal other false types.

```
alert(null == undefined); // unenforced type during check, displays
true
alert(null === undefined); // enforce type during check, displays false
```

## Number

Numbers are represented in binary as IEEE-754 Doubles, which provides an accuracy to about 14 or 15 significant digits JavaScript FAQ, Numbers <sup>[3]</sup>. Because they are floating point numbers, they do not always exactly represent real numbers, including fractions.

This becomes an issue when formatting numbers. For example:

```
alert(0.94 - 0.01); // displays 0.9299999999999999
```

As a result, a routine such as the `toFixed()` method should be used to round numbers whenever they are formatted for output <sup>[4]</sup>.

Numbers may be specified in any of these notations:

```
345;    // an "integer", although there is only one numeric type in
JavaScript
34.5;    // a floating-point number
3.45e2;  // another floating-point, equivalent to 345
0377;    // an octal integer equal to 255
0xFF;    // a hexadecimal integer equal to 255, digits represented by
the ...
        // ... letters A-F may be upper or lowercase
```

In some ECMAScript implementations such as `ActionScript`, RGB color values are sometimes specified with hexadecimal integers:

```
var colorful = new Color( '_root.shapes' );
colorful.setRGB( 0x003366 );
```

The extents of the number type may also be described by named constant values:

```
Infinity; // Construct equivalent to positive Infinity
-Infinity; // Negated Infinity construct, equal to negative Infinity
NaN;      // The Not-A-Number value, often returned as a failure in ...
          // ... string-to-number conversions
```

The `Number` constructor, or a unary `+` or `-`, may be used to perform explicit numeric conversion:

```
var myString = "123.456";
var myNumber1 = Number( myString );
var myNumber2 = + myString;
```

When used as a constructor, a numeric *wrapper* object is created, (though it is of little use):

```
myNumericWrapper = new Number( 123.456 );
```



## String

A String in Javascript is a sequence of characters. Strings in JavaScript can be created directly by placing the series of characters between double or single quotes.

```
var greeting = "Hello, world!";
var another_greeting = 'Greetings, people of Earth.';
```

Strings are instances of the String class:

```
var greeting = new String("Hello, world!");
```

You can access individual characters within a string using the `charAt()` method (provided by `String.prototype`). This is the preferred way when accessing individual characters within a string, as it also works in non-Mozilla-based browsers:

```
var h = greeting.charAt(0); // Now h contains 'H' - Works in both ...
                          // ... Internet Explorer and Mozilla ...
                          // ... based browsers
```

In Mozilla based browsers, individual characters within a string can be accessed (as strings with only a single character) through the same notation as arrays:

```
var h = greeting[0]; // Now h contains 'H' - Works in Mozilla based
browsers
```

But JavaScript strings are immutable:

```
greeting[0] = "H"; // ERROR
```

Applying the equality operator ("`==`") to two strings returns true if the strings have the same contents, which means: of same length and same cases (for alphabets). Thus:

```
var x = "world";
var compare1 = ("Hello, " + x == "Hello, world"); // Now compare1
contains true
var compare2 = ("Hello, " + x == "hello, world"); // Now compare2
contains ...
// ... false since
the ...
// ... first
characters ...
// ... of both
operands ...
// ... are not of the
same case
```

You cannot use quotes of the same type inside the quotes unless they are escaped.

```
var x = '"Hello, world!" he said.' //Just fine.
var x = '"Hello, world!" he said.' //Not good.
var x = "\"Hello, world!\" he said." //That works by replacing " with \"
```

## Boolean

JavaScript provides a Boolean data type with true and false literals. The typeof operator returns the string "boolean" for these primitives. In a logical context, automatic type coercion evaluates 0, -0, null, NaN, undefined or the empty string ("") as false. Everything else evaluates as true, including the strings "0", "false" and all objects. Automatic coercion can sometimes be avoided by using the type checked comparison operators, (=== and !==).

```
// Automatic type coercion
alert(true == 1);      // true.... Only ±0 and NaN are false Numbers.
alert(true == "0");    // true.... Only the empty String is false.

// Type checked comparison
alert(true === 1);     // false... The data types don't match.
alert(true === "0");   // false... The data types don't match.
```

The binary logical operators returned a Boolean value in early versions of JavaScript, but now they return one of the operands instead. The left-operand is returned if it can be evaluated as: false, in the case of conjunction (a && b), or true, in the case of disjunction (a || b); otherwise the right-operand is returned. An expression can be explicitly cast to a Boolean primitive by: doubling the logical negation operator (!), using the Boolean() function, or using the conditional operator (c ? t : f).

```
alert({} || null);     // Object {}... Any Object is true.
alert(!0 && null);      // null..... The negation of zero is true.
alert(!undefined);     // true..... The negation of undefined is true.
alert(true === !!"0"); // true..... Double negation of a non-empty String is both true and Boolean.
alert("" ? "T" : "F"); // "F"..... The empty String is false.
alert(Boolean("false")); // true..... A non-empty String is true.
alert(typeof Boolean()); // "boolean"... A missing parameter is undefined and thus false.
```

The new operator can be used to create an object wrapper for a Boolean primitive. However, the typeof operator does not return "boolean" for the object wrapper, it returns "object". Because all objects evaluate as true, a method such as .valueOf(), or .toString(), must be used to retrieve the wrapped value. For explicit coercion to the Boolean type, Mozilla recommends that the Boolean() function (without new) be used in preference to the Boolean object.

```
var b = new Boolean(false); // Object false {}
var t = Boolean(b);         // Boolean true var
f = Boolean(b.valueOf());   // Boolean false var n
= new Boolean(b);           // Not recommended
n = new Boolean(b.valueOf()); // Preferred

if ( 0 || -0 || "" || null || undefined || b.valueOf() || !new
Boolean() || !t ) {
    alert("Never this");
} else if ( [] && {} && b && typeof b === "object" && b.toString() ===
"false" ) {
    alert("Always this");
}
```

## Native Objects

The JavaScript language provides a handful of native objects. JavaScript native objects are considered part of the JavaScript specification. JavaScript environment notwithstanding, this set of objects should always be available.

### Array

An Array is a native JavaScript type specifically designed to store data values indexed by integer keys. Arrays, unlike the basic Object type, are prototyped with methods and properties to aid the programmer in routine tasks (e.g., join, slice, and push).

As in the C family, arrays use a zero-based indexing scheme: A value that is inserted into an empty array by means of the push method occupies the 0th index of the array.

```
var myArray = [];           // Point the variable myArray to a newly
...                          // ... created, empty Array
myArray.push("hello world"); // Fill the next empty index, in this case
0
alert(myArray[0]);          // Equivalent to alert("hello world");
```

Arrays have a length property that is guaranteed to always be larger than the largest integer index used in the array. It is automatically updated if one creates a property with an even larger index. Writing a smaller number to the length property will remove larger indices.

Elements of Arrays may be accessed using normal object property access notation:

```
myArray[1]; //this gives you the 2nd item in myArray
myArray["1"];
```

The above two are equivalent. It's not possible to use the "dot"-notation or strings with alternative representations of the number:

```
myArray.1; // syntax error
myArray["01"]; // not the same as myArray[1]
```

Declaration of an array can use either an Array literal or the Array constructor:

```
myArray = [0,1,,,4,5];           // array with length 6 and 6
elements, ...                    // ... including 2 undefined elements
myArray = new Array(0,1,2,3,4,5); // array with length 6 and 6 elements
myArray = new Array(365);         // an empty array with length 365
```

Arrays are implemented so that only the elements defined use memory; they are "sparse arrays". Setting myArray[10] = 'someThing' and myArray[57] = 'somethingOther' only uses space for these two elements, just like any other object. The length of the array will still be reported as 58.

You can use the object declaration literal to create objects that behave much like associative arrays in other languages:

```
dog = {"color":"brown", "size":"large"};
dog["color"]; // this gives you "brown"
dog.color;    // this also gives you "brown"
```

You can use the object and array declaration literals to quickly create arrays that are associative, multidimensional, or both.

```
cats = [{"color":"brown", "size":"large"},
        {"color":"black", "size":"small"}];
cats[0]["size"];    // this gives you "large"

dogs = {"rover":{"color":"brown", "size":"large"},
        "spot":{"color":"black", "size":"small"}};
dogs["spot"]["size"]; // this gives you "small"
dogs.rover.color;    // this gives you "brown"
```

## Date

A Date object stores a signed millisecond count with zero representing 1970-01-01 00:00:00 UT and a range of  $\pm 10^8$  days. One can create a Date object representing "now":

```
var d = new Date();
```

Initial value of Date object is current date and time. Other ways to construct a Date instance:

```
var d = new Date(2010,4,5);    // create a new Date instance with
value:                          // 2010-4-5 00:00:00

var d = new Date(2010,4,5,14,25,30); // create a new Date instance with
value:                          // 2010-4-5 14:25:30
```

Format the display of a Date instance:

```
var d = new Date(2010,5,6,14,25,30); // May,2010
var display = d.getFullYear() + '-' + (d.getMonth()+1) + '-' +
d.getDate()
              + ' ' + d.getHours() + ':' + d.getMinutes() + ':' +
d.getSeconds();
alert(display);    // Displays '2010-5-6 14:25:30'
```

Construct from a String:

```
var d = new Date("2010-4-6 14:25:30");
alert(d);
```

## Error

Custom error messages can be created using the Error class:

```
throw new Error("What the...");
```

Nested within conditional statements, such instantiations can substitute for try/catch blocks:

```
var eMail = prompt("Please enter your e-mail address:", "");
if (!eMail || eMail.length == 0)
{
    throw new Error("Excuse me: You must enter your e-mail address to
continue.");
}
```

}

## Math

The Math object contains various math-related constants (e.g.  $\pi$ ) and functions (e.g. cosine). (Note the "Math" object has no constructor, unlike Array or Date. All its methods are "static" aka "class" methods.) All the trigonometric functions use angles expressed in radians; not degrees or grads.

### Properties of the Math object

Property	Value Returned rounded to 5 digits	Description
Math.E	2.7183	e: Euler's number
Math.LN2	0.69315	Natural logarithm of 2
Math.LN10	2.3026	Natural logarithm of 10
Math.LOG2E	1.4427	Logarithm to the base 2 of e
Math.LOG10E	0.43429	Logarithm to the base 10 of e
Math.PI	3.14159	$\pi$ : circumference/diameter of a circle
Math.SQRT1_2	0.70711	Square root of $\frac{1}{2}$
Math.SQRT2	1.4142	Square root of 2

### Methods of the Math object

Example	Value Returned rounded to 5 digits	Description
Math.abs(-2.3)	2.3	Absolute value: $(x < 0) ? -x : x$
Math.acos(Math.SQRT1_2)	0.78540 rad. = $45^\circ$	Arccosine
Math.asin(Math.SQRT1_2)	0.78540 rad. = $45^\circ$	Arcsine
Math.atan(1)	0.78540 rad. = $45^\circ$	Half circle arctangent $(-\pi/2 \text{ to } +\pi/2)$
Math.atan2(-3.7, -3.7)	-2.3562 rad. = $-135^\circ$	Whole circle arctangent $(-\pi \text{ to } +\pi)$
Math.ceil(1.1)	2	Ceiling: round up to smallest integer $\geq$ argument
Math.cos(Math.PI/4)	0.70711	Cosine
Math.exp(1)	2.7183	Exponential function: e raised to this power
Math.floor(1.9)	1	Floor: round down to largest integer $\leq$ argument
Math.log(Math.E)	1	Natural logarithm, base e
Math.max(1, -2)	1	Maximum: $(x > y) ? x : y$
Math.min(1, -2)	-2	Minimum: $(x < y) ? x : y$
Math.pow(-3, 2)	9	Exponentiation (raised to the power of): Math.pow(x, y) gives $x^y$
Math.random()	0.17068	Pseudorandom number between 0 (inclusive) and 1 (exclusive)
Math.round(1.5)	2	Round to the nearest integer; half fractions are rounded up (e.g. 1.5 rounds to 2)
Math.sin(Math.PI/4)	0.70711	Sine
Math.sqrt(49)	7	Square root
Math.tan(Math.PI/4)	1	Tangent

## Regular Expression

```

/expression/.test(string);
"string".search(/expression/);
"string".replace(/expression/,replacement);

// Here are some examples
if(/Tom/.test("My name is Tom")) alert("Hello Tom!");
alert("My name is Tom".search(/Tom/));           // = 11
(letters before Tom)
alert("My name is Tom".replace(/Tom/,"John"));    // = "My
name is John"

```

Character Classes:

```

// \d - digit
// \D - non digit
// \s - space
// \S - non space
// \w - word char
// \W - non word
// [ ] - one of
// [^] - one not of
// - - range

if(/^d/.test('0')) alert('Digit');
if(/[0-9]/.test('5')) alert('Digit');
if(/[13579]/.test('1')) alert('Odd number');
if(/^\S\S\S\S\S\S\S\S/.test('My name')) alert('Format OK');
if(/^\w\w\w\w/.test('Tom')) alert('Format OK');
if(/[a-z]/.test('b')) alert('Small letter');
if(/[A-Z]/.test('B')) alert('Big letter');
if(/[a-zA-Z]/.test('B')) alert('Letter');

```

Character matching:

```

// A...Z a...z 0...9 - alphanumeric
// \u0000... \uFFFF - Unicode hexadecimal
// \x00... \xFF - ASCII hexadecimal
// \t - tab
// \n - new line
// \r - CR
// . - any character
// | - OR

if(/T.m/.test('Tom')) alert ('Hi Tom, Tam or Tim');
if(/A|B/.test("A")) alert ('A or B');

```

Repeaters:

```
// ?      - 0 or 1 match
// *      - 0 or more
// +      - 1 or more
// {n}    - exactly n
// {n,}   - n or more
// {0,n}  - n or less
// {n,m}  - range n to m

if(/ab?c/.test("ac"))      alert("OK"); // match: "ac", "abc"
if(/ab*c/.test("ac"))      alert("OK"); // match: "ac", "abc",
"abbc", "abbbc" etc.
if(/ab+c/.test("abc"))      alert("OK"); // match: "abc", "abbc",
"abbbc" etc.
if(/ab{3}c/.test("abbbc"))  alert("OK"); // match: "abbbc"
if(/ab{3,}c/.test("abbbc")) alert("OK"); // match: "abbbc", "abbbbc",
"abbbbbbc" etc.
if(/ab{1,3}c/.test("abc"))  alert("OK"); // match: "abc", "abbc",
"abbbc"
```

Anchors:

```
// ^      - string starts with
// $      - string ends with

if(/^My/.test("My name is Tom")) alert ("Hi!");
if(/Tom$/.test("My name is Tom")) alert ("Hi Tom!");
```

Subexpression

```
// ( )    - groups characters

if(/water(mark)?/.test("watermark")) alert("Here is water!"); //
match: "water", "watermark",
if(/(Tom)|(John)/.test("John"))      alert("Hi Tom or John!");
```

Flags:

```
// /g     - global
// /i     - ignore upper/lower case
// /m     -

alert("hi tom!".replace(/Tom/i,"John")); // = "hi John!"
alert("ratatam".replace(/ta/,"tu"));     // = "ratutam"
alert("ratatam".replace(/ta/g,"tu"));     // = "ratutum"
```

Advanced methods

```
my_array=my_string.split(my_delimiter);
// example
my_array="dog,cat,cow".split(","); //
my_array==("dog","cat","cow");
```

```
my_array=my_string.match(my_expression);
// example
my_array="We start at 11:30, 12:15 and 16:45".match(/\d\d:\d\d/); //
my_array=("11:30","12:15","16:45");
```

Capturing groups

```
var myRe = /(\d{4}-\d{2}-\d{2}) (\d{2}:\d{2}:\d{2})/;
var results = myRe.exec("The date and time are 2009-09-08 09:37:08.");
if(results) {
    alert("Matched: " + results[0]); // Entire match
    var my_date = results[1]; // Firstgroup = "2009-09-08"
    var my_time = results[2]; // Second group = "09:37:08"
    alert("It is " + my_time + " on " + my_date);
} else alert("Did not find a valid date!");
```

## Function

Every function in Javascript is an instance of the Function object:

```
var add=new Function('x','y','return x+y');//x,y is the
argument.'return x+y' is the function body, which is the last in the
argument list.
var t=add(1,2);
alert(t);//3
```

The add function above is often defined using the following pattern, as this pattern is faster and more intuitive.

```
function add(x,y)
{
    return x+y;
}
var t=add(1,2);
alert(t);//3
```

A function instance has properties and methods.

```
function subtract(x,y)
{
    return x-y;
}
alert(subtract.length);//2,expected amount of arguments.
alert(subtract.toString());
/*
function subtract(x,y)
{
    return x-y;
}
*/
```



## Operators

The '+' operator is overloaded; it is used for string concatenation and arithmetic addition and also to convert strings to numbers. It also has special meaning when used in a regular expression.

```
// Concatenate 2 strings
var a = 'This';
var b = ' and that';
alert(a + b); // displays 'This and that'

// Add two numbers
var x = 2;
var y = 6;
alert(x + y); // displays 8

// Adding a string and a number results in concatenation
alert(x + '2'); // displays 22

// Convert a string to a number
var z = '4'; // z is a string (the digit 4)
var x = '2'; // x is a string (the digit 2)
alert(z + x); // displays 42
alert(+z + +x); // displays 6
```

## Arithmetic

JavaScript supports the following **binary arithmetic operators**:

+	Addition
-	Subtraction
*	Multiplication
/	Division (returns a floating-point value)
%	Modulus (returns the integer remainder)

JavaScript supports the following **unary arithmetic operators**:

+	Unary conversion of string to number
-	Unary negation (reverses the sign)
++	Increment (can be prefix or postfix)
--	Decrement (can be prefix or postfix)

```
var x = 1;
alert( ++x ); // displays: 2
alert( x++ ); // displays: 2
alert( x );   // displays: 3
alert( x-- ); // displays: 3
alert( x );   // displays: 2
alert( --x ); // displays: 1
```

## Assignment

```
=      Assign
+=     Add and assign
-=     Subtract and assign
*=     Multiply and assign
/=     Divide and assign
%=     Modulus and assign
```

```
var x = 1;
x *= 3;
alert( x ); // displays: 3
x /= 3;
alert( x ); // displays: 1
x -= 1;
alert( x ); // displays: 0
```

## Comparison

```
==      Equal
!=      Not equal
>       Greater than
>=      Greater than or equal to
<       Less than
<=      Less than or equal to
===     Identical (equal and of the same type)
!==     Not identical
```

## Logical

JavaScript provides four logical operators:

- unary negation (NOT = !a)
- binary disjunction (OR = a || b) and conjunction (AND = a && b)
- ternary conditional (c ? t : f)

In the context of a logical operation, any expression evaluate to true except the following:

- Strings: "", "",
- Numbers: 0, NaN,
- Special: null, undefined,
- Boolean: false, !true.

The Boolean function can be used to explicitly convert to a primitive of type Boolean:

```
// Only empty strings return false
alert( Boolean( "" ) )      === false );
alert( Boolean( "false" ) ) === true );
alert( Boolean( "0" ) )     === true );

// Only zero and NaN return false
alert( Boolean( NaN ) )     === false );
alert( Boolean( 0 ) )       === false );
alert( Boolean( -0 ) )      === false ); // equivalent to -1*0
```

```

alert( Boolean( -2 )      === true );

// All objects return true
alert( Boolean( this )    === true );
alert( Boolean( {} )      === true );
alert( Boolean( [] )      === true );

// These types return false
alert( Boolean( null )    === false );
alert( Boolean( undefined ) === false ); // equivalent to Boolean()

```

The NOT operator evaluates its operand in as a Boolean, and returns the negation. Using the operator twice in a row, as a double negative, explicitly converts an expression to a primitive of type Boolean:

```

alert( !0 === Boolean( !0 ) === !!1 === Boolean( 1 ) === true );
alert( !!0 === Boolean( 0 ) === !1 === Boolean( !1 ) === false );
alert( !"" === Boolean( !"" ) === !!"s" === Boolean( "s" ) === true );
alert( !!"" === Boolean( "" ) === !"s" === Boolean( !"s" ) === false );

```

Expressions that use features such as post-incrementation, (i++), have an anticipated side effect. JavaScript provides short-circuit evaluation of expressions; the right operand is only executed if the left operand does not suffice to determine the value of the expression.

```

alert( a || b ); // When a is true, there is no reason to evaluate b.
alert( a && b ); // When a is false, there is no reason to evaluate b.
alert( c ? t : f ); // When c is true, there is no reason to evaluate f.

```

In early versions of JavaScript and JScript, the binary logical operators returned a Boolean value (like most C-derived programming languages). However, all contemporary implementations return one of their operands instead:

```

alert( a || b ); // if a is true, return a, otherwise return b
alert( a && b ); // if a is false, return a, otherwise return b

```

Programmers who are more familiar with the behavior in C might find this feature surprising but it allows for a more concise expression of patterns like null coalescing:

```

var s = t || "(default)"; // assigns t, or the default value if t is null, empty, etc.

```

## Bitwise

JavaScript supports the following **binary bitwise operators**:

```
&      And
|      Or
^      Xor

<<     Shift left  (zero fill)
>>     Shift right (sign-propagating); copies of the leftmost bit (sign bit) are shifted in from the left.
>>>    Shift right (zero fill)
```

For positive numbers, >> and >>> yield the same result.

JavaScript supports the following **unary bitwise operators**:

```
~      Not (inverts the bits)
```

## String

```
=      Assignment
+      Concatenation
+=     Concatenate and assign
```

Examples

```
str = "ab" + "cd";    // "abcd"
str += "e";           // "abcde"

str2 = "2"+2          // "22", not "4" or 4.
```

## Control structures

### Compound statements

A pair of curly brackets { } and an enclosed sequence of statements constitute a compound statement, which can be used wherever a statement can be used.

### If ... else

```
if (expr)
{
    //statements;
}
else if (expr2)
{
    //statements;
}
else
{
    //statements;
}
```

## Conditional operator

The conditional operator creates an expression that evaluates as one of two expressions depending on a condition. This is similar to the *if* statement that selects one of two statements to execute depending on a condition. I.e., the conditional operator is to expressions what *if* is to statements.

```
var result = (condition) ? expression : alternative;
```

is the same as:

```
if (condition)
{
    result = expression;
}
else
{
    result = alternative;
}
```

Unlike the *if* statement, the conditional operator cannot omit its "else-branch".

## Switch statement

The syntax of the JavaScript Switch statement is as follows:

```
switch (expr) {
    case SOMEVALUE :
        //statements;
        break;
    case ANOTHERVALUE :
        //statements;
        break;
    default :
        //statements;
        break;
}
```

- `break`; is optional; however, it is usually needed, since otherwise code execution will continue to the body of the next case block.
- Add a `break` statement to the end of the last case as a precautionary measure, in case additional cases are added later.
- Strings literal values can also be used for the case values.
- case default: is optional
- Braces are required.

## For loop

The syntax of the JavaScript for loop is as follows:

```
for (initial;condition;loop statement) {  
    /*  
        statements will be executed every time  
        the for{} loop cycles, while the  
        condition is satisfied  
    */  
}
```

or

```
for (initial;condition;loop statement) // one statement
```

## For ... in loop

The syntax of the JavaScript For ... in loop is as follows:

```
for (var property_name in some_object) {  
    //statements using some_object[property_name];  
}
```

- Iterates through all enumerable properties of an object.
- Sources differ on whether this is usable for arrays<sup>[5]</sup>.
- There are differences between the various web browsers with regard to which properties will be reflected with the for...in loop statement. In theory, this is controlled by an internal state property defined by the ECMAScript standard called "DontEnum", but in practice each browser returns a slightly different set of properties during introspection. It is useful to test given property using if (some\_object.hasOwnProperty(property\_name)) { ... }

## While loop

The syntax of the JavaScript while loop is as follows:

```
while (condition) {  
    statement1;  
    statement2;  
    statement3;  
    ...  
}
```

## Do ... while loop

The syntax of the JavaScript do ... while loop is as follows:

```
do {  
    statement1;  
    statement2;  
    statement3;  
    ...  
} while (condition);
```

## With

The with statement sets the default object for the set of statements that follow.

```
with(document) {
  var a = getElementById('a');
  var b = getElementById('b');
  var c = getElementById('c');
};
```

- Note the absence of document. before each getElementById() invocation.

The semantics are similar to the with statement of Pascal.

## Functions

A function is a block with a (possibly empty) parameter list that is normally given a name. A function may utilize local variables. If exit is not by a return statement, the value undefined is returned.

```
function gcd(segmentA, segmentB) {
  var diff = segmentA - segmentB
  if (diff == 0) return segmentA
  if (diff > 0)
    return gcd(segmentB, diff)
  else
    return gcd(segmentA, -diff)
}
alert(gcd(60, 40)); // 20
```

The number of arguments given when calling a function may not necessarily correspond to the number of arguments in the function definition; a named argument in the definition that does not have a matching argument in the call will have the value undefined. Within the function, the arguments may also be accessed through the arguments object; this provides access to all arguments using indices (e.g. arguments[0], arguments[1], ... arguments[n]), including those beyond the number of named arguments. Note that while the arguments list has a .length property, it is *not* an instance of Array; it does not have methods such as .slice(), .sort(), etc.

All parameters are passed by value (for objects it is the reference to the object that is passed).

```
var obj1 = {a:1}
var obj2 = {b:2}
function foo(p) {
  p = obj2; // ignores actual parameter
  p.b = arguments[1]
}
foo(obj1, 3) // Does not affect obj1 at all. 3 is additional parameter
alert(obj1.a + " " + obj2.b); // writes 1 3
```

Functions can be declared inside other functions, and access the outer function's local variables. Furthermore they implement closures by remembering the outer function's local variables even after the outer function has exited.

```
var v = "top"
var bar
function foo() {
  var v = "foo"
```

```
bar = function() {alert(v)}  
}  
foo()  
bar() // writes "foo", not "top" even though foo() has exited.
```

## Objects

For convenience, Types are normally subdivided into *primitives* and *objects*. Objects are entities that have an identity (they are only equal to themselves) and that map property names to values, ("slots" in prototype-based programming terminology). Objects may be thought of as associative arrays or hashes, and are often implemented using these data structures. However, objects have additional features, such as a prototype chain, which ordinary associative arrays do not have.

JavaScript has several kinds of built-in objects, namely Array, Boolean, Date, Function, Math, Number, Object, RegExp and String. Other objects are "host objects", defined not by the language but by the runtime environment. For example, in a browser, typical host objects belong to the DOM (window, form, links etc.).

## Creating objects

Objects can be created using a constructor or an object literal. The constructor can use either a built-in Object function or a custom function. It is a convention that constructor functions are given a name that starts with a capital letter:

```
// Constructor  
var anObject = new Object();  
  
// Object literal  
var objectA = {};  
var objectA2 = {}; // A != A2, {}s create new objects as copies.  
var objectB = {index1:'value 1', index2:'value 2'};  
  
// Custom constructor (see below)
```

Object literals and array literals allow one to easily create flexible data structures:

```
var myStructure = {  
  name: {  
    first: "Mel",  
    last: "Smith"  
  },  
  age: 33,  
  hobbies: [ "chess", "jogging" ]  
};
```

This is the basis for JSON, which is a simple notation that uses JavaScript-like syntax for data exchange.



## Methods

A method is simply a function that is assigned to the value of an object's slot. Unlike many object-oriented languages, there is no distinction between a function definition and a method definition. Rather, the distinction occurs during function calling; a function can be called as a method.

When called as a method, the standard local variable *this* is just automatically set to the object instance to the left of the ".". (There are also *call* and *apply* methods that can set *this* explicitly -- some packages such as jQuery do unusual things with *this*.)

In the example below, Foo is being used as a constructor. There is nothing special about a constructor, it is just a method that is invoked after the object is created. *this* is set to the newly created object.

Note that in the example below, Foo is simply assigning values to slots, some of which are functions. Thus it can assign different functions to different instances. There is no prototyping in this example.

```
function y2() {return this.xxx + "2 "};

function Foo(xz) {
  this.xxx = "yyy-";
  if (xz > 0)
    this.xx = function() {return this.xxx + "X "};
  else
    this.xx = function() {return this.xxx + "Z "};
  this.yy = y2;
}

var foo1 = new Foo(1);
var foo2 = new Foo(0);

foo1.y3 = y2; // Assigns the function itself, not its evaluated result,
              // i.e. not y2()
foo2.xxx = "aaa-";

alert("foo1/2 " + foo1.xx() + foo2.xx());
// foo1/2 yyy-X aaa-Z

var baz={ "xxx": "zzz-" }
baz.y4 = y2 // No need for a constructor to make an object.

alert("yy/y3/y4 " + foo1.yy() + foo1.y3() + baz.y4());
// yy/y3/y4 yyy-2 yyy-2 zzz-2

foo1.y2(); // Throws an exception, because foo1.y2 doesn't exist.
```

## Constructors

Constructor functions simply assign values to slots of a newly created object. The values may be data or other functions.

Example: Manipulating an object

```
function MyObject(attributeA, attributeB) {
  this.attributeA = attributeA;
  this.attributeB = attributeB;
}

MyObject.staticC = "blue"; // On MyObject Function, not obj
alert(MyObject.staticC); // blue

obj = new MyObject('red', 1000);

alert(obj.attributeA); // red
alert(obj["attributeB"]); // 1000

alert(obj.staticC); // undefined

obj.attributeC = new Date(); // add a new property

delete obj.attributeB; // remove a property of obj
alert(obj.attributeB); // undefined

delete obj; // remove the whole Object (rarely used)
alert(obj.attributeA); // throws an exception
```

The constructor itself is stored in the special slot *constructor*. So

```
x = new Foo()
// Above is almost equivalent to
y = {};
y.constructor = Foo;
y.constructor();
// except
x.constructor == y.constructor // true
x instanceof Foo // true
y instanceof Foo // false, surprisingly.
```

Functions are objects themselves, which can be used to produce an effect similar to "static properties" (using C++/Java terminology) as shown below. (The function object also has a special prototype property, as discussed in the Inheritance section below.)

Object deletion is rarely used as the scripting engine will garbage collect objects that are no longer being referenced.

## Inheritance

JavaScript supports inheritance hierarchies through prototyping in the manner of Self.

In the following example, the `Derive` class inherits from the `Base` class. When `d` is created as a `Derive`, the reference to the base instance of `Base` is copied to `d.base`.

`Derive` does not contain a value for `aBaseFunction`, so it is retrieved from `Base` when `aBaseFunction` is accessed. This is made clear by changing the value of `base.aBaseFunction`, which is reflected in the value of `d.aBaseFunction`. Some implementations allow the prototype to be accessed or set explicitly using the `__proto__` slot as shown below.

```
function Base() {
  this.anOverride = function() {alert("Base::anOverride()");};

  this.aBaseFunction = function() {alert("Base::aBaseFunction()");};
}

function Derive() {
  this.anOverride = function() {alert("Derive::anOverride()");};
}

base = new Base();
Derive.prototype = base; // Must be before new Derive()

d = new Derive(); // Copies Derive.prototype to d instance's hidden
prototype slot.

base.aBaseFunction = function() {alert("Base::aNEWBaseFunction()");}

d.anOverride(); // Derive::anOverride() d.aBaseFunction(); //
Base::aNEWBaseFunction() alert(d.aBaseFunction ==
Derive.prototype.aBaseFunction); // true

alert(d.__proto__ == base); // true in Mozilla-based implementations
but false in many other implementations
```

The following shows clearly how references to prototypes are *copied* on instance creation, but that changes to a prototype can affect all instances that refer to it.

```
function m1() {return "One ";}
function m2() {return "Two ";}
function m3() {return "Three ";}

function Base() {}

Base.prototype.yyy = m2;
bar = new Base();
alert("bar.yyy " + bar.yyy()); // bar.yyy Two

function Top(){this.yyy = m3}
ttt = new Top();
```

```

foo = new Base();
Base.prototype = ttt;
    // No effect on foo, the *reference* to ttt is copied.
alert("foo.yyy" + foo.yyy()); // foo.yyy Two

baz = new Base();
alert("baz.yyy" + baz.yyy()); // baz.yyy Three

ttt.yyy = m1; // Does affect baz, and any other derived classes.
alert("baz.yyy1" + baz.yyy()); // baz.yyy1 One

```

In practice many variations of these themes are used, and it can be both powerful and confusing.

## Exception handling

Newer versions of JavaScript (as used in Internet Explorer 5 and Netscape 6) include a try ... catch ... finally exception handling statement to handle run-time errors.

The try ... catch ... finally statement catches exceptions resulting from an error or a throw statement. Its syntax is as follows:

```

try {
    // Statements in which exceptions might be thrown
} catch(errorValue) {
    // Statements that execute in the event of an exception
} finally {
    // Statements that execute afterward either way
}

```

Initially, the statements within the try block execute. If an exception is thrown, the script's control flow immediately transfers to the statements in the catch block, with the exception available as the error argument. Otherwise the catch block is skipped. The Catch block can throw(errorValue) if it does not want to handle a specific error.

In any case the statements in the finally block are always executed. This can be used to free resources, although memory is automatically garbage collected.

Either the catch or the finally clause may be omitted. The catch argument is required.

The Mozilla implementation allows for multiple catch statements, as an extension to the ECMAScript standard. They follow a syntax similar to that used in Java:

```

try { statement; }
catch ( e if e == "InvalidNameException" ) { statement; }
catch ( e if e == "InvalidIdException" ) { statement; }
catch ( e if e == "InvalidEmailException" ) { statement; }
catch ( e ) { statement; }

```

In a browser, the onerror event is more commonly used to trap exceptions.

```

function handleErr(errorValue,url,lineNr){...; return true;}
onerror=handleErr;

```

## See also

- Comparison of Javascript-based source code editors
- ECMAScript
- JavaScript
- JScript

## Reference Material

- David Flanagan, Paula Ferguson: *JavaScript: The Definitive Guide*, O'Reilly & Associates, ISBN 0-596-10199-6
- Danny Goodman: *JavaScript Bible*, Wiley, John & Sons, ISBN 0-7645-3342-8
- Thomas A. Powell, Fritz Schneider: *JavaScript: The Complete Reference*, McGraw-Hill Companies, ISBN 0-07-219127-9
- Emily Vander Veer: *JavaScript For Dummies, 4th Edition*, Wiley, ISBN 0-7645-7659-3

## External links

- A re-introduction to JavaScript - Mozilla Developer Center <sup>[6]</sup>
- ECMAScript standard references: ECMA-262 <sup>[7]</sup>
- Interactive JavaScript Lessons - example-based <sup>[8]</sup>
- JavaScript on About.com: lessons and explanation <sup>[9]</sup>
- Mozilla Developer Center Core References for JavaScript versions 1.5 <sup>[10]</sup>, 1.4 <sup>[11]</sup>, 1.3 <sup>[12]</sup> and 1.2 <sup>[13]</sup>
- Mozilla JavaScript Language Documentation <sup>[14]</sup>

## References

[1] <http://hepunix.rl.ac.uk/~adye/jsspec11/intro.htm#1006028>

[2] Flanagan, David (2006). *JavaScript: The definitive Guide*. p. 16. ISBN 978-0-596-10199-2. "Omitting semicolons is not a good programming practice; you should get into the habit of inserting them."

[3] <http://www.jibbering.com/faq/#numbers>

[4] <http://www.jibbering.com/faq/#formatNumber>

[5] The issue is whether it would iterate not only through the array indices, but also through other visible properties.

An article in the Microsoft Developer Network website ([http://msdn.microsoft.com/en-us/library/kw1tezhk\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/kw1tezhk(VS.85).aspx)) specifically states that `For...In` can be used for "*stepping through ... all the elements of an array*". The MSDN article refers to JScript, which is, effectively, what is used by Internet Explorer and Windows Script Host for JavaScript script. An example in the W3Schools website ([http://www.w3schools.com/js/js\\_loop\\_for\\_in.asp](http://www.w3schools.com/js/js_loop_for_in.asp)) gives arrays as an example of the use of `For...In`.

An article in the Mozilla Developer Centre ([https://developer.mozilla.org/en/Core\\_JavaScript\\_1.5\\_Guide/Object\\_Manipulation\\_Statements#for...in\\_Statement](https://developer.mozilla.org/en/Core_JavaScript_1.5_Guide/Object_Manipulation_Statements#for...in_Statement)) explains more about the problem: "*Although it may be tempting to use this as a way to iterate over Array elements, because the `for...in` statement iterates over user-defined properties in addition to the array elements, if you modify the Array object, such as adding custom properties or methods, the `for...in` statement will return the name of your user-defined properties in addition to the numeric indexes. Thus it is better to use a traditional for loop with a numeric index when iterating over arrays.*"

[6] [https://developer.mozilla.org/en/docs/A\\_re-introduction\\_to\\_JavaScript](https://developer.mozilla.org/en/docs/A_re-introduction_to_JavaScript)

[7] <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

[8] <http://javalessons.com/cgi-bin/fun/java-tutorials-main.cgi?sub=javascript&code=script>

[9] <http://javascript.about.com/>

[10] [https://developer.mozilla.org/en/docs/Core\\_JavaScript\\_1.5\\_Reference](https://developer.mozilla.org/en/docs/Core_JavaScript_1.5_Reference)

[11] <http://research.nihonsoft.org/javascript/CoreReferenceJS14/>

[12] <http://research.nihonsoft.org/javascript/ClientReferenceJS13/>

[13] <http://research.nihonsoft.org/javascript/jsref/>

[14] <https://developer.mozilla.org/en/docs/JavaScript>

# JavaScript Style Sheets

---

<b>Filename extension</b>	.js
<b>Internet media type</b>	text/javascript
<b>Developed by</b>	Netscape Communications Corporation
<b>Type of format</b>	Style sheet language
<b>Standard(s)</b>	Netscape's JavaScript-Based Style Sheets submission to the W3C <sup>[1]</sup>

**JavaScript Style Sheets (JSSS)** was a stylesheet language technology proposed by Netscape Communications Corporation in 1996 to provide facilities for defining the presentation of webpages.<sup>[2]</sup> It was an alternative to the Cascading Style Sheets (CSS) technology.<sup>[2]</sup> Although Netscape submitted it to the World Wide Web Consortium (W3C), the technology was never accepted as a formal standard and it never gained much acceptance in the market. Only Netscape Communicator 4 supported JSSS, with the rival Internet Explorer web browser choosing not to implement the technology. Soon after Netscape Communicator's release in 1997, Netscape stopped promoting JSSS, instead focusing on the rival CSS standard, which was also supported by Internet Explorer and had much wider industry acceptance. The follow-up to Netscape Communicator, Netscape 6 (released in 2000), dropped support for JSSS. It now remains little more than a historical footnote, with many Web developers not even being aware of its existence. The proposed standard was not finished.

Using JavaScript code as a stylesheet, JSSS styles individual element by modifying properties of the `document.tags` object. For example, the CSS:

```
h1 { font-size: 20pt; }
```

is equivalent to the JSSS:

```
document.tags.H1.fontSize = "20pt";
```

Note the case significance in element names.

JSSS is in some ways more powerful and in some ways less powerful than CSS. It lacks the various CSS selector features, supporting only a simple tag name selector. On the other hand, since it is written using a complete programming language, stylesheets can include highly complex dynamic calculations and conditional processing. (In practice, however, this can be achieved with CSS by using JavaScript to modify the stylesheets applicable to the document at runtime.) Because of this JSSS was often used in the creation of DHTML.

Being written in JavaScript, JSSS may seem less friendly than CSS to users without a programming background.

## Example

The following example shows part of the sourcecode of an HTML-document:

```
<style type="text/javascript">
tags.H1.color = "blue";
tags.p.fontSize = "14pt";
with(tags.H3)
  color = "green";
with(tags.H2)
{
  color = "red";
```

```
fontSize = "16pt";
marginTop = "2cm";
}
</style>
```

Similar to Cascading Style Sheets JSSS can be used in a <style>-tag. This Example shows two different methods to select tags.

## Browser support

JavaScript Style Sheets were only supported by Netscape 4.x (4.0–4.8) but no later versions. No other web browser integrated JSSS.

## External links

- Netscape's JavaScript-Based Style Sheets submission to the W3C <sup>[1]</sup>
- The dynamic, powerful abilities of JavaScript Style Sheets <sup>[3]</sup>

## References

[1] <http://www.w3.org/Submission/1996/1/WD-jsss-960822>

[2] Håkon Wium Lie; Bert Bos. "Chapter 20 - The CSS saga" (<http://www.w3.org/Style/LieBos2e/history/>). World Wide Web Consortium. . Retrieved 23 June 2010.

[3] <http://sunsite.uakom.sk/sunworldonline/swol-04-1997/swol-04-webmaster.html>

# JavaScript engine

A **JavaScript engine** is specialized software which interprets and executes JavaScript. Although there are several uses for a JavaScript engine, the most common usage is for web browsers. <sup>[1]</sup> <sup>[2]</sup>

## History

Before the second browser wars in 2008-2009, the **JavaScript engine** (also known as **JavaScript interpreter** or **JavaScript implementation**) was known simply as an interpreter that reads JavaScript source code and executes the script accordingly.

The first ever JavaScript engine was created by Brendan Eich at Netscape Communications Corporation, for the Netscape Navigator web browser. The engine, code named SpiderMonkey, is implemented in C. It has since been updated (in JavaScript 1.5) to conform to ECMA-262 Edition 3. The Rhino engine, created primarily by Norris Boyd (also at Netscape) is a JavaScript implementation in Java. Like SpiderMonkey, Rhino is ECMA-262 Edition 3 compliant. Applications of the technology include Apple Safari 4's Nitro, Google Chrome's V8 and Mozilla Firefox 3.5's TraceMonkey.

By far, the most common host environment for JavaScript is a web browser. Web browsers typically use the public API to create "host objects" responsible for reflecting the DOM into JavaScript.

The web server is another common application of the engine. A JavaScript webserver would expose host objects representing a HTTP request and response objects, which a JavaScript program could then manipulate to dynamically generate web pages. Microsoft's ASP technology for IIS allows server-side code to be written in VB Script or JScript (Microsoft's implementation of JavaScript). Jaxer is a web server that runs entirely on JavaScript. This has the benefit of allowing the same code to be shared on the server and on the client.

## Performance evolution

"..previously behind-the-scenes programming technology called JavaScript is getting new visibility .."
—CNET [2]

A typical major browser has a graphical engine and an independent JavaScript engine, which allows for easier testing, reimplementation or usage in other projects. For example Carakan is used with Presto, Nitro with WebKit, SpiderMonkey with Gecko, KJS with KHTML, Rhino by default doesn't have any layout engine. Other combinations are sometimes possible, for example, V8 with WebKit in Google Chrome. The JavaScript engine gives developers access to functionality (networking, DOM handling, external events, HTML5 video, canvas and storage) needed to control the web browser.

Sunspider is a JavaScript benchmark utility for measuring the performance of JavaScript engines in more than a dozen of tests, each concentrating on different part of JavaScript language. Sunspider doesn't use for benchmarking any features beyond this needed to test pure computations (no HTML, no CSS, no networking).

### The JavaScript engine race: 2008 and 2009

There has since been a race by browser developers to develop even faster JavaScript engines. In 2008, Google Chrome was praised for its JavaScript performance, but other browsers soon received new JavaScript engines which were faster. Later, Google Chrome won in the races of better performance. Chrome's strength is its application performance and JavaScript processing speed, both of which were independently verified by multiple websites to be the fastest amongst the major browsers of its time.<sup>[3] [4] [5]</sup> With the advent of WebKit's Squirrelfish Extreme and Mozilla's TraceMonkey JavaScript virtual machines, Chrome's JavaScript execution performance has been found to be slower.<sup>[6] [7] [8] [9]</sup> Google responded with the Danish developed V8 (JavaScript engine) which boosted JS performance in Google Chrome 2.

On June 2, 2008 the WebKit development team announced SquirrelFish<sup>[10]</sup> — a then new JavaScript engine that vastly improves Safari's speed at interpreting scripts.<sup>[11]</sup> The engine was one of the new features in Safari 4, released for developers on June 11, 2008; the final JavaScript engine was called Nitro.

2009

In January 2009 the engine then known as SquirrelFish Extreme (SFX) was enabled for Mac OS X on x86-64 architectures as it passes all tests on that platform by Apple Corp..<sup>[12]</sup> Released June 30, 2009 Firefox 3.5 includes the optimization technique which offered "performance improvements ranging between 20 and 40 times faster" compared to Firefox 3 in some cases<sup>[13]</sup>

### The JavaScript engine race: 2010

In early 2010 the Norwegian Opera browser replaced the aging Futhark with the faster Carakan, which was 2.5 times faster in early testing<sup>[2]</sup> Others in the race at this time include Apple's Safari's Nitro (the engine formerly known as SquirrelFish), and Firefox's new JaegerMonkey (a "cross-child of Nitro with the older TraceMonkey Engine").<sup>[1]</sup> Microsoft lagged behind, lacking a dedicated JavaScript engine and being the slowest of the major browsers, although by mid-2010 they held out the carrot of Chakra in then unreleased Internet Explorer 9.<sup>[1]</sup> JaegerMonkey began testing in the publicly released Firefox 4.0 beta in the Summer of 2010.<sup>[14]</sup> Safari 5, also released in summer 2010, featured 30 percent faster JavaScript performance than Safari 4 (using the Nitro engine).<sup>[15]</sup>



## JavaScript engines

Major browser JS engines:

### Mozilla

- Rhino, managed by the Mozilla Foundation, open source, developed entirely in Java
- SpiderMonkey (code name), the first ever JavaScript engine, written by Brendan Eich at Netscape Communications
- TraceMonkey, the engine promoted with Firefox 3.5
- JägerMonkey, the engine in development for Mozilla Firefox 4. <sup>[16]</sup>
- Tamarin, by Adobe Labs

### Google

- V8 - open source, developed by Google in Denmark, part of Google Chrome

### Other

- KJS - KDE's ECMAScript/JavaScript engine originally developed by Harri Porten for the KDE project's Konqueror web browser
- Narcissus open source, written by Brendan Eich, who also wrote SpiderMonkey
- Chakra, for Internet Explorer 9. <sup>[17]</sup>
- Nitro, (formerly SquirrelFish) for Safari 4
- Carakan, by Opera Software, used since Opera 10.50
- Futhark, by Opera Software, replaced by Carakan in Opera 10.50 (released March 2010)

## Implementations

JavaScript is a dialect of ECMAScript, which is supported in many applications, especially web browsers. Dialects sometimes include extensions to the language, or to the standard library and related APIs such as the W3C-specified DOM. This means that an application written in one dialect may be incompatible with another, unless the applications are written to use only a common subset of supported features and APIs.

Note that there is a distinction between a dialect and an implementation. A dialect of a language is significant variation of the language, while an implementation of a language/dialect executes a program written in that language/dialect.

Application	Dialect and latest version	ECMAScript edition
Google Chrome, the V8 engine	JavaScript	ECMA-262, edition 3
Mozilla Firefox, the Gecko layout engine, SpiderMonkey, and Rhino	JavaScript 1.8.1	ECMA-262, edition 3
Opera	ECMAScript with some JavaScript 1.5 and JScripextensions <sup>[18]</sup>	ECMA-262, edition 3
KHTML layout engine, KDE's Konqueror, and Apple's Safari	JavaScript 1.5	ECMA-262, edition 3
Adobe Acrobat	JavaScript 1.5	ECMA-262, edition 3
OpenLaszlo Platform	JavaScript 1.4	ECMA-262, edition 3
Max/MSP	JavaScript 1.5	ECMA-262, edition 3
ANT Galio 3	JavaScript 1.5 with RMAI extensions	ECMA-262, edition 3

## See also

- Sunspider – Browser speed test

## External links

- Peacekeeper<sup>[19]</sup> – Browser speed test
- Speed-Battle<sup>[20]</sup> – Online JavaScript speed test

## References

- [1] [http://news.cnet.com/8301-30685\\_3-20000110-264.html](http://news.cnet.com/8301-30685_3-20000110-264.html) "Opera 10.5 brings new JavaScript engine" Stephen Shankland
- [2] [http://news.cnet.com/8301-17939\\_109-10157475-2.html](http://news.cnet.com/8301-17939_109-10157475-2.html)
- [3] Stephen Shankland (2008-09-02). "Speed test: Google Chrome beats Firefox, IE, Safari" ([http://news.cnet.com/8301-1001\\_3-10030888-92.html](http://news.cnet.com/8301-1001_3-10030888-92.html)). *cnet.com Business Tech*. CNET News. . Retrieved 2010-06-28.
- [4] "Big browser comparison test: Internet Explorer vs. Firefox, Opera, Safari and Chrome" (<http://www.pcgameshardware.com/aid,687738/Big-browser-comparison-test-Internet-Explorer-vs-Firefox-Opera-Safari-and-Chrome-Update-Firefox-35-Final/Practice/>). *PC Games Hardware*. Computec Media AG. . Retrieved 2010-06-28.
- [5] "Lifehacker Speed Tests: Safari 4, Chrome 2" (<http://lifehacker.com/5286869/lifehacker-speed-tests-safari-4-chrome-2-and-more>). Lifehacker. . Retrieved 2010-06-28.
- [6] Stephen Shankland (2008-09-02). "Third Chrome beta another notch faster" (<http://news.cnet.com/third-chrome-beta-another-notch-faster/>). *cnet.com*. CNET News. . Retrieved 2010-06-28.
- [7] Stephen Shankland (2008-09-19). "Step aside, Chrome, for SquirrelFish Extreme" ([http://news.cnet.com/8301-13579\\_3-10046637-37.html](http://news.cnet.com/8301-13579_3-10046637-37.html)). *cnet.com*. CNET News. . Retrieved 2010-06-29.
- [8] "SquirrelFish Extreme: Fastest JavaScript Engine Yet" (<http://www.satine.org/archives/2008/09/19/squirrelfish-extreme-fastest-javascript-engine-yet/>). *satine.org*. . Retrieved 2010-06-29.
- [9] Stephen Shankland (2008-09-03). "Firefox counters Google's browser speed test" ([http://news.cnet.com/8301-1001\\_3-10031278-92.html](http://news.cnet.com/8301-1001_3-10031278-92.html)). *cnet.com Business Tech*. CNET News. . Retrieved 2010-06-29.
- [10] Garen, Geoffrey (2008-06-02). "Announcing SquirrelFish" (<http://webkit.org/blog/189/announcing-squirrelfish/>). . Retrieved 2008-06-11.
- [11] Lipskas, Vyantas (2008-06-11). "Apple Safari 4" (<http://www.favbrowser.com/apple-safari-4/>). . Retrieved 2008-06-11.
- [12] <https://trac.webkit.org/changeset/40439>
- [13] Ryan Paul (2008-08-22). "Firefox to get massive JavaScript performance boost" (<http://arstechnica.com/news/ars/post/20080822-firefox-to-get-massive-javascript-performance-boost.html>). *arstechnica.com*. Ars Technica © 2010 Condé Nast Digital. . Retrieved 2010-06-28.
- [14] "Firefox 4 Vision: fast, powerful, and empowering" (<http://blog.mozilla.com/blog/2010/05/10/firefox-4-vision-fast-powerful-and-empowering/>). .
- [15] <http://www.pnnewswire.com/news-releases/apple-releases-safari-5-95817479.html> Safari 5 Released
- [16] "Firefox 4 Vision: fast, powerful, and empowering" (<http://blog.mozilla.com/blog/2010/05/10/firefox-4-vision-fast-powerful-and-empowering/>). .
- [17] Marius Oiaga (2010-03-20), "Internet Explorer 9 Beta Next – New IE9 Builds Every 8 Weeks" (<http://news.softpedia.com/news/Internet-Explorer-9-Beta-Next-New-IE9-Builds-Every-8-Weeks-138013.shtml>), *softpedia.com* (SoftNews NET SRL), , retrieved 2010-06-28
- [18] "Web specifications support in Opera Presto" (<http://www.opera.com/docs/specs/#ecmascript>). *Opera.com*. Opera Software ASA. . Retrieved 2010-06-28.
- [19] <http://service.futuremark.com/peacekeeper/index.action>
- [20] <http://www.speed-battle.com>

# Ajax (programming)

---

**Ajax** (pronounced English pronunciation: /ˈeɪ.dʒæks/) (shorthand for Asynchronous JavaScript and XML<sup>[1]</sup>) is a group of interrelated web development techniques used on the client-side to create interactive web applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. The use of Ajax techniques has led to an increase in interactive or dynamic interfaces on web pages. Data is usually retrieved using the *XMLHttpRequest* object. Despite the name, the use of XML is not actually required, nor do the requests need to be asynchronous.<sup>[2]</sup>

Like DHTML and LAMP, Ajax is not a technology in itself, but a group of technologies. Ajax uses a combination of HTML and CSS to mark up and style information. The DOM is accessed with JavaScript to dynamically display, and to allow the user to interact with the information presented. JavaScript and the *XMLHttpRequest* object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads.

## History

In the 1990s, most web sites were based on complete HTML pages; each user action required that the page be re-loaded from the server (or a new page loaded). This process is not efficient, as reflected by the user experience (all page content disappears then reappears, etc.). Each time a page is reloaded due to a partial change, all of the content must be re-sent instead of just the changed information. This can place additional load on the server and use excessive bandwidth.

Asynchronous loading of content first became practical when Java applets were introduced in the first version of the Java language in 1995. These allow compiled client-side code to load data asynchronously from the web server after a web page is loaded.<sup>[3]</sup> In 1996, Internet Explorer introduced the *IFrame* element to HTML, which also enabled asynchronous loading.<sup>[4]</sup> In 1999, Microsoft created the XMLHTTP ActiveX control in Internet Explorer 5, which was later adopted by Mozilla, Safari, Opera and other browsers as the native *XMLHttpRequest* object.<sup>[4]</sup><sup>[5]</sup> Microsoft has adopted the native *XMLHttpRequest* model as of Internet Explorer 7, though the ActiveX version is still supported. The utility of background HTTP requests to the server and asynchronous web technologies remained fairly obscure until it started appearing in full scale online applications such as Outlook Web Access,(2000)<sup>[6]</sup> Oddpost (2002), and later, notably Google made a wide deployment of Ajax with Gmail (2004) and Google Maps (2005).<sup>[7]</sup>

The term "Ajax" was coined in 2005 by Jesse James Garrett.<sup>[1]</sup> However, a patent application covering this type of user interface was filed on September 3, 2003, thus predating the term itself by two years. This application resulted in US Patent #7,523,401 being issued to Greg Aldridge of Kokomo, IN.<sup>[8]</sup>

On April 5, 2006 the World Wide Web Consortium (W3C) released the first draft specification for the object in an attempt to create an official web standard.<sup>[7]</sup>

## Technologies

The term *Ajax* has come to represent a broad group of web technologies that can be used to implement a web application that communicates with a server in the background, without interfering with the current state of the page. In the article that coined the term Ajax,<sup>[1]</sup> Jesse James Garrett explained that the following technologies are required:

- HTML or XHTML and CSS for presentation
  - the Document Object Model for dynamic display of and interaction with data
  - XML for the interchange of data, and XSLT for its manipulation
  - the *XMLHttpRequest* object for asynchronous communication
  - JavaScript to bring these technologies together
-

Since then, however, there have been a number of developments in the technologies used in an Ajax application, and the definition of the term Ajax. In particular, it has been noted that:

- JavaScript is not the only client-side scripting language that can be used for implementing an Ajax application. Other languages such as VBScript are also capable of the required functionality.<sup>[2] [9]</sup> However JavaScript is the most popular language for Ajax programming due to its inclusion in and compatibility with the majority of modern web browsers.
- XML is not required for data interchange and therefore XSLT is not required for the manipulation of data. JavaScript Object Notation (JSON) is often used as an alternative format for data interchange,<sup>[10]</sup> although other formats such as preformatted HTML or plain text can also be used.<sup>[11]</sup>

Classic Ajax involves writing ad hoc JavaScript on the client. A simpler if cruder alternative is to use standard JavaScript libraries that can partially update a page, such as ASP.Net's UpdatePanel. Tools such as Echo2 and ZK enable fine grained control of a page from the server, using only standard JavaScript libraries.

Introducing XMLHttpRequest and pseudomultithreading, it is possible to swap the role of client and server (web browser may start behaving as a server and web server may start behaving as a client) in Client-Server model.

## Drawbacks

- Owing to their dynamic nature, Ajax interfaces are often harder to develop when compared to static pages.
- Pages dynamically created using successive Ajax requests do not automatically register themselves with the browser's history engine, so clicking the browser's "back" button may not return the user to an earlier state of the Ajax-enabled page, but may instead return them to the last full page visited before it. Workarounds include the use of invisible IFrames to trigger changes in the browser's history and changing the URL fragment identifier (the portion of a URL after the '#') when Ajax is run and monitoring it for changes.<sup>[12] [13]</sup>
- Dynamic web page updates also make it difficult for a user to bookmark a particular state of the application. Solutions to this problem exist, many of which use the URL fragment identifier (the portion of a URL after the '#') to keep track of, and allow users to return to, the application in a given state.<sup>[12] [13]</sup>
- Depending on the nature of the Ajax application, dynamic page updates may interfere disruptively with user interactions, especially if working on an unstable internet connection. For instance, editing a search field may trigger a query to the server for search completions, but the user may not know that a search completion popup is forthcoming, and if the internet connection is slow, the popup list may show up at an inconvenient time, when the user has already proceeded to do something else.
- Because most web crawlers do not execute JavaScript code,<sup>[14]</sup> publicly indexable web applications should provide an alternative means of accessing the content that would normally be retrieved with Ajax, to allow search engines to index it.
- Any user whose browser does not support JavaScript or XMLHttpRequest, or simply has this functionality disabled, will not be able to properly use pages which depend on Ajax. Similarly, devices such as mobile phones, PDAs, and screen readers may not have support for the required technologies. Screen readers that are able to use Ajax may still not be able to properly read the dynamically generated content.<sup>[15]</sup> The only way to let the user carry out functionality is to fall back to non-JavaScript methods. This can be achieved by making sure links and forms can be resolved properly and do not rely solely on Ajax. In JavaScript, form submission could then be halted with "return false".<sup>[16]</sup>
- The same origin policy prevents some Ajax techniques from being used across domains,<sup>[7]</sup> although the W3C has a draft of the XMLHttpRequest object that would enable this functionality.<sup>[17]</sup> Techniques exist to sidestep this limitation by using a special Cross Domain Communications channel embedded as an iframe within a page.<sup>[18]</sup>
- Like other web technologies, Ajax has its own set of vulnerabilities that developers must address. Developers familiar with other web technologies may have to learn new testing and coding methods to write secure Ajax applications.<sup>[19] [20]</sup>

- Ajax-powered interfaces may dramatically increase the number of user-generated requests to web servers and their back-ends (databases, or other). This can lead to longer response times and/or additional hardware needs.

## See also

- Ajax framework
- ASP.NET AJAX
- Comet (programming)
- Reverse Ajax
- Rich Internet application
- XMLHttpRequest
- Google Web Toolkit
- jQuery Framework (Ajax)

## External links

- Ajax: A New Approach to Web Applications <sup>[21]</sup> Article that coined the term and Q&A.
- Ajax (programming) <sup>[22]</sup> at the Open Directory Project
- Ajax Tutorial <sup>[23]</sup> with get, post, text and XML examples.
- Mastering Ajax Tutorial <sup>[24]</sup>.

## References

- [1] Jesse James Garrett (2005-02-18). "Ajax: A New Approach to Web Applications" (<http://www.adaptivepath.com/ideas/essays/archives/000385.php>). AdaptivePath.com. . Retrieved 2008-06-19.
- [2] Ullman, Chris (March 2007). *Beginning Ajax* (<http://www.wrox.com/WileyCDA/Section/id-303217.html>). wrox. ISBN 978-0-470-10675-4. . Retrieved 2008-06-24.
- [3] "Code Samples and Apps: Applets" (<http://java.sun.com/applets/>). Sun Microsystems, Inc.. . Retrieved 2009-01-02.
- [4] Hinchcliffe, Dion (June 2006). *Real-World Ajax: Secrets of the Masters* (<http://ajaxdevelopersjournal.com/read/338113.htm>). SYS-CON Media. ISBN 9780977762200. .
- [5] "Dynamic HTML and XML: The XMLHttpRequest Object" (<http://developer.apple.com/internet/webcontent/xmlhttpreq.html>). Apple Inc. . Retrieved 2008-06-25.
- [6] Hopmann, Alex. "Story of XMLHttpRequest" (<http://www.alexhopmann.com/story-of-xmlhttp/>). *Alex Hopmann's Blog*. . Retrieved 17 May 2010.
- [7] "A Brief History of Ajax" (<http://www.aaronsw.com/weblog/ajaxhistory>). Aaron Swartz. 2005-12-22. . Retrieved 2009-08-04.
- [8] Gregory E. Aldridge (2003-03-09). "System and method for providing a browser-based user interface" (<http://www.uspto.gov/web/patents/patog/week16/OG/html/1341-3/US07523401-20090421.html>). United States Patent and Trademark Office. . Retrieved 2010-03-28.
- [9] But use of VBScript assumes the target browser supports it.
- [10] "JSON - JavaScript Object Notation" (<http://tapestry.apache.org/tapestry4.1/ajax/json.html>). Apache.org. . Retrieved 2008-07-04.
- [11] "Speed Up Your Ajax-based Apps with JSON" (<http://www.devx.com/webdev/Article/32651>). DevX.com. . Retrieved 2008-07-04.
- [12] "Why use Ajax?" ([http://www.interaktonline.com/support/articles/Details/Ajax:+Asynchronously+Moving+Forward-Why+use+Ajax?.html?id\\_art=36&id\\_asc=309](http://www.interaktonline.com/support/articles/Details/Ajax:+Asynchronously+Moving+Forward-Why+use+Ajax?.html?id_art=36&id_asc=309)). InterAKT. 2005-11-10. . Retrieved 2008-06-26.
- [13] "Deep Linking for AJAX" (<http://blog.onthewings.net/2009/04/08/deep-linking-for-ajax/>). .
- [14] Prokoph, Andreas (2007-05-08). "Help Web crawlers efficiently crawl your portal sites and Web sites" (<http://www.ibm.com/developerworks/library/x-sitemaps/index.html>). IBM. . Retrieved 2009-04-22.
- [15] Edwards, James (2006-05-05). "Ajax and Screenreaders: When Can it Work?" (<http://www.sitepoint.com/article/ajax-screenreaders-work>). sitepoint.com. . Retrieved 2008-06-27.
- [16] Quinsey, Peter. "User-Proofing Ajax" (<http://www.alistapart.com/articles/userproofingajax>). .
- [17] "Access Control for Cross-Site Requests" (<http://dev.w3.org/2006/waf/access-control/>). World Wide Web Consortium. . Retrieved 2008-06-27.
- [18] "Secure Cross-Domain Communication in the Browser" (<http://msdn.microsoft.com/en-us/library/bb735305.aspx>). The Architecture Journal (MSDN). . Retrieved 2010-04-27.
- [19] Sullivan, Bryan. "Testing for security in the age of Ajax Programming" (<http://www.developerfusion.com/article/6197/testing-for-security-in-the-age-of-ajax-programming/>). developerFusion. . Retrieved 2008-10-15.

- [20] Stamos, Alex; Lackey, Zane. "Attacking Ajax Web Applications" ([http://www.isecpartners.com/files/iSEC-Attacking\\_AJAX\\_Applications.BH2006.pdf](http://www.isecpartners.com/files/iSEC-Attacking_AJAX_Applications.BH2006.pdf)). iSEC Partners. . Retrieved 2009-04-02.
- [21] <http://www.adaptivepath.com/ideas/essays/archives/000385.php>
- [22] <http://www.dmoz.org/Computers/Programming/Languages/JavaScript/Ajax/>
- [23] <http://www.xul.fr/en-xml-ajax.html>
- [24] <http://www.ibm.com/developerworks/web/library/wa-ajaxintro1.html>

# AJAX.OOP

---

**AJAX.OOP** is an open source JavaScript framework distributed under MIT License. Providing with OOP-style coding engine and AJAX requests handling functionality to create web2.0 components. Due to OOP paradigm AJAX.OOP library can be easily extended with additional functionality or used as core for other projects.

## The JavaScript OOP Library

AJAX.OOP is a fast and scalable JavaScript Library for creating JavaScript/AJAX components in an object oriented way.

Main feature - strong OOP paradigm implementation. AJAX.OOP engine allows the programmer to:

- Create classes with object-like defining syntax
- Inherit classes (both AJAX.OOP-style defined and created with native JavaScript code)
- Aggregate classes (including aggregation of self and parent class with special operators `this._self` and `this._super`)
- Define constructors as they are (name = 'constructor')
- Override parent class properties and methods (but call parent if you need by using special access operator `this.$super`)
- Access/call any parent properties and/or methods from any method of child class (just use `this.$super` accessor whenever you need to access parent properties and methods)
- Use strict defined objects with workable 'instanceof' operator on them when instantiating an exemplar of AJAX.OOP classes

## External links

- [AJAX.OOP Official Website](#) <sup>[1]</sup>
- [AJAX.OOP - Download](#) <sup>[2]</sup>
- [AJAX.OOP - Documentation](#) <sup>[3]</sup>

## References

- [1] <http://ajaxoop.org/>
  - [2] <http://code.google.com/p/ajaxoop/downloads/list>
  - [3] <http://ajaxoop.org/documentation>
-

# ?:

In computer programming, **?:** is a ternary operator that is part of the syntax for a basic conditional expression in several programming languages. It is commonly referred to as the conditional operator.

It originally comes from BCPL, whose equivalent syntax for  $e1 \text{ ? } e2 : e3$  was  $e1 \rightarrow e2, e3$ <sup>[1]</sup>. Languages derived from BCPL tend to feature this operator.

## Conditional assignment

**?:** is used as follows:

```
condition ? value if true : value if false
```

The *condition* is evaluated *true* or *false* as a Boolean expression. On the basis of the evaluation of the Boolean condition, the entire expression returns *value if true* if *condition* is true, but *value if false* otherwise. Usually the two sub-expressions *value if true* and *value if false* must have the same type, which determines the type of the whole expression. The importance of this type-checking lies in the operator's most common use—in conditional assignment statements. In this usage it appears as an expression on the right side of an assignment statement, as follows:

```
variable = condition ? value if true : value if false
```

The **?:** operator is similar to the way conditional expressions (**if-then-else** constructs) work in functional programming languages, like Scheme, ML, and Haskell, since if-then-else forms an expression instead of a statement in those languages.

## Usage

This ternary operator's most common usage is to make a terse simple conditional assignment statement. For example, if we wish to implement some C code to change a shop's opening hours to 12 o'clock in weekends, and 9 o'clock on weekdays, we may use

```
int opening_time = (day == WEEKEND) ? 12 : 9;
```

instead of the more verbose

```
int opening_time;

if (day == WEEKEND)
    opening_time = 12;
else
    opening_time = 9;
```

The two forms are nearly equivalent. Keep in mind that the **?:** is an expression and if-then-else is a statement. Note that neither *value if true* nor *value if false* expressions can be omitted from the ternary operator without an error report upon parsing. This contrasts with if..else *statements*, where the else clause can be omitted.

## C Variants

A GNU extension to C allows the second operand to be omitted, and the first operand is implicitly used as the second as well:

```
a = x ? : y;
```

The expression is equivalent to

```
a = x ? x : y;
```

except that if *x* is an expression, it is evaluated only once. The difference is significant if evaluating the expression has side effects.

C# (and Perl) provide similar functionality with their coalescing operator

```
a = x ?? y;
```

(Unlike the above usage of "*x* ?: *y*", *??* will only test if *x* is non-null, as opposed to non-false.)

## C++

In C++ there are conditional assignment situations where use of the *if-else* statement is not possible, since this language explicitly distinguishes between initialization and assignment. In such case it is always possible to use a function call, but this can be cumbersome and inelegant. For example, if you want to pass conditionally different values as an argument for a constructor of a field or a base class, it is not possible to use a plain *if-else* statement; in this case we can use a conditional assignment expression, or a function call. Mind also that some types allow initialization, but do not allow assignment, or even the assignment operator does totally different things than the constructor. The latter is true for reference types, for example:

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main(int argc, char *argv[])
{
    string name;
    ofstream fout;
    if (argc > 1 && argv[1])
    {
        name = argv[1];
        fout.open(name.c_str(), ios::out | ios::app);
    }

    ostream &sout = name.empty() ? cout : fout;
}
```

In this case there's no possibility to replace the use of *?:* operator with *if-else* statement. (Although we can replace the use of *?:* with a function call, inside of which can be an *if-else* statement.)



Furthermore, the ternary operator can yield an lvalue, i.e. a value to which another value can be assigned. Consider the following example:

```
#include <iostream>
int main () {
    int a=0, b=0;

    const bool cond = ...;
    (cond ? a : b) = 1;
    std::cout << "a=" << a << ', '
               << "b=" << b << '\n';
}
```

In this example, if the boolean variable **cond** yields the value **true** in line 5, the value **1** is assigned to the variable **a**, otherwise, it is assigned to **b**.

## Python

The Python programming language uses a different syntax for this operator:

```
value_when_true if condition else value_when_false
```

This feature is not available for Python versions prior to 2.5, however. The Python programming FAQ<sup>[2]</sup> mentions several possible workarounds for these versions.

## Visual Basic .NET

Although it doesn't use **?:** per se, Microsoft Visual Basic .NET has a very similar implementation of this shorthand if...else statement. Using the first example provided in this article, you can do:

```
Dim opening_time As Integer = If((day = WEEKEND), 12, 9)
```

*'general syntax is If(condition, value\_if\_true, value\_if\_false)*

In the above example, **If** is a function, and not an actual ternary operator. As a function, the values of all three portions are evaluated before the function call occurs. This imposed limitations, and in Visual Basic .Net 9.0, released with Visual Studio 2008, an actual ternary operator was introduced, using the **If** keyword instead of **If**. This allows the following example code to work:

```
Dim name As String = If(person Is Nothing, "", person.Name)
```

Using **If**, **person.Name** would be evaluated even if **person** is null (**Nothing**), causing an exception. With a true short-circuiting ternary operator, **person.Name** is not evaluated unless **person** is not null.

## PHP

```
<?php
$arg = "T";
$vehicle = ( ( $arg == 'B' ) ? 'bus' :
              ( $arg == 'A' ) ? 'airplane' :
              ( $arg == 'T' ) ? 'train' :
              ( $arg == 'C' ) ? 'car' :
              ( $arg == 'H' ) ? 'horse' :
```

```

        'feet');
echo $vehicle;
?>

```

Due to an unfortunate error in the language grammar, the implementation of `?:` in PHP uses the incorrect associativity when compared to other languages, and given a value of **T** for **arg**, the PHP equivalent of the above example would yield the value **horse** instead of **train** as one would expect. To avoid this, nested parenthesis are needed, as in this example:

```

<?php
$arg = "T";
$vehicle = ($arg == 'B') ? 'bus' :
            (($arg == 'A') ? 'airplane' :
            (($arg == 'T') ? 'train' :
            (($arg == 'C') ? 'car' :
            (($arg == 'H') ? 'horse' :
            'feet'))));
echo $vehicle;
?>

```

This will produce the correct result of **train** being printed to the screen.

### CFML (Railo only)

```

<cfscript>
arg = "T";
vehicle = ( ( arg == 'B' ) ? 'bus' :
            ( arg == 'A' ) ? 'airplane' :
            ( arg == 'T' ) ? 'train' :
            ( arg == 'C' ) ? 'car' :
            ( arg == 'H' ) ? 'horse' :
            'feet' );
</cfscript>
<cfoutput>#vehicle#</cfoutput>

```

### Result type

Clearly the type of the result of the `?:` operator must be in some sense the type unification of the types of its second and third operands. In C this is accomplished for numeric types by arithmetic promotion; since C does not have a type hierarchy for pointer types, pointer operands may only be used if they are of the same type (ignoring type qualifiers) or one is void or NULL. It is undefined behaviour to mix pointer and integral or incompatible pointer types; thus

```
number = spell_out_numbers ? "forty-two" : 42;
```

will result in a compile-time error in most compilers.

## ?: in style guidelines

Some corporate programming guidelines list the use of the conditional operator as bad practice because it can harm readability and long-term maintainability. Conditional operators are widely used and can be useful in certain circumstances to avoid the use of an if statement, either because the extra verbiage would be too lengthy or because the syntactic context does not permit a statement. For example:

```
#define MAX(a, b) (((a)>(b)) ? (a) : (b))
```

or

```
for (i = 0; i < MAX_PATTERNS; i++)  
    c_patterns[i].ShowWindow(m_data.fOn[i] ? SW_SHOW : SW_HIDE);
```

(The latter example uses the Microsoft Foundation Classes Framework for Win32.)

When properly formatted, the conditional operator can be used to write simple and coherent case selectors. For example:

```
vehicle = arg == 'B' ? bus :  
          arg == 'A' ? airplane :  
          arg == 'T' ? train :  
          arg == 'C' ? car :  
          arg == 'H' ? horse :  
          feet;
```

Appropriate use of the conditional operator in a variable assignment context reduces the probability of a bug from a faulty assignment as the assigned variable is stated just once as opposed to multiple times.

## See also

- ?? Operator

## References

- [1] "BCPL Ternary operator (page 15)" (<http://cm.bell-labs.com/cm/cs/who/dmr/bcpl.pdf>). *BCPL Reference Manual*. .
- [2] <http://www.python.org/doc/faq/programming/#is-there-an-equivalent-of-c-s-ternary-operator>

# Appcelerator Titanium

---

<b>Developer(s)</b>	Appcelerator, Inc.
<b>Stable release</b>	1.3 / 14 May 2010
<b>Operating system</b>	iPhone, Android, Mac OS, Windows, Linux
<b>Type</b>	Application framework
<b>License</b>	Apache Public License v2
<b>Website</b>	<a href="http://www.appcelerator.com">www.appcelerator.com</a> <sup>[1]</sup>

**Appcelerator Titanium** is a platform for developing mobile and desktop applications using web technologies. Appcelerator Titanium is developed by Appcelerator Inc. and was introduced in December 2008<sup>[2]</sup>. Support for developing iPhone- and Android-based mobile applications was added in June 2009<sup>[3]</sup>. Support for developing iPad-based tablet apps was added in April 2010<sup>[4]</sup>.

Appcelerator Titanium is one of several phone web based application framework solutions allowing web developers to apply existing skills to create native applications for iPhone and Android. Appcelerator Titanium is frequently compared to Adobe Air for developing desktop applications for Windows, Mac and Linux.<sup>[5]</sup> Traditionally, proprietary tools and specialized skills are required to develop native software applications for each computing platform<sup>[6]</sup>.

Appcelerator Titanium includes a web-based, cross-compilation tool which requires internet access and a developer account. The tool can deploy standalone applications for Mac, Windows, and Linux from any of those platforms. It does this by submitting the source files to a proprietary server-side solution which then returns the binaries. An open source command-line compiler is also available which is not subject to the same network and account requirements, but it does not cross-compile. Mobile compilation is subject to additional requirements: iPhone builds require Mac OS X and the iPhone SDK, and Android builds require the Android SDK and Mac, Windows, or Linux. The latest version of Appcelerator Titanium's cross-compiler was built using itself.

In April 2010 Appcelerator expanded the Titanium product line with the Titanium Tablet SDK<sup>[4]</sup>. The Titanium Tablet SDK draws heavily from the existing support for iPhone but also includes native support for iPad-only user interface controls such as split views and popovers. Initially the Tablet SDK supports only Apple's iPad.

Appcelerator, Inc. also offers cloud-based services for packaging, testing and distributing software applications developed on the Titanium platform<sup>[7]</sup>.

## Features

The core features of Appcelerator Titanium include<sup>[6]</sup>:

- Support for standards-based web technologies: HTML, CSS and Javascript on all platforms along with PHP, Python and Ruby for desktop platforms
  - Integrated support for popular JavaScript and AJAX Frameworks including jQuery, YUI, MooTools, Scriptaculous and others.
  - A platform-independent API to access native UI components including navigation bars, menus, dialog boxes and alerts, and native device functionality including the file system, sound, network and local database
  - API access to native mobile functionality like geolocation, accelerometer and maps
  - Extensibility through open interfaces and licensing, allowing developers to introduce support for additional scripting languages, media codecs and device-specific functionality
  - Available under the Apache Public License v2.0 open source software license
-

## See also

- List of rich internet application frameworks

## External links

- Appcelerator Web site <sup>[1]</sup>
- Appcelerator Titanium tutorials <sup>[8]</sup>

## References

- [1] <http://www.appcelerator.com>
- [2] "Appcelerator Raises \$4.1 Million for Open Source RIA Platform" (<http://www.techcrunch.com/2008/12/09/appcelerator-raises-41-million-for-open-source-ria-platform/>). Techcrunch. 9 December 2008. . Retrieved 29 October 2009.
- [3] "Appcelerator enables iPhone, Android app dev" (<http://www.infoworld.com/d/developer-world/appcelerator-enables-iphone-android-app-dev-655>). InfoWorld. 8 June 2009. . Retrieved 29 October 2009.
- [4] "Appcelerator Simplifies iPad App Development" (<http://mashable.com/2010/04/05/titanium-tablet-sdk/>). 5 April 2010. . Retrieved 6 April 2010.
- [5] "Appcelerator Takes On Adobe AIR with Titanium" (<http://www.eweek.com/c/a/Application-Development/Appcelerator-Takes-on-Adobe-AIR-with-Titanium/>). eWeek. 9 December 2008. . Retrieved 29 October 2009.
- [6] "Titanium gets hardened with new beta" (<http://ajaxian.com/archives/titanium-gets-hardened-with-new-beta-that-features-mobile-and-more>). Ajaxian. 9 June 2009. . Retrieved 29 October 2009.
- [7] "Appcelerator Network Cloud Services" (<http://www.appcelerator.com/products/cloud-services/>). Appcelerator, Inc.. . Retrieved 29 October 2009.
- [8] <http://www.sergemeunier.com/blog/titanium-tutorials/>

# AppengineJS

---

**AppengineJS** is a framework for creating web applications using JavaScript and deploy to Google's scalable infrastructure.

AppengineJS is open source software licensed under the liberal MIT license.

## External links

Official website <sup>[1]</sup>

## References

- [1] <http://www.appenginejs.org>
-

# BSON

---

<b>Filename extension</b>	.bson
<b>Type of format</b>	Data interchange

**BSON** (pronounced /ˈbiːsɒn/) is a computer data interchange format. It is a binary form for representing simple data structures and associative arrays (called objects or documents). The name "BSON" is based on the term JSON and informally means "Binary JSON".

## Data types and syntax

BSON documents (objects) consist of a well ordered list of *elements*. Each element consists of a field name, a type, and a value. Field names are strings. Types include:

- string
- integer
- double
- date
- byte array (binary data)
- boolean (true and false)
- null
- BSON object

This is nominally a superset of JSON types (JSON does not have a byte array type, for example), but because of length limitations, some valid JSON values (such as very long strings) are not valid BSON values.

## Efficiency

Compared to JSON, BSON is efficient both in storage space and scan-speed. Large elements in a BSON document are prefixed with a length field to facilitate scanning.

## See also

- JSON
- Protocol Buffers
- Document-oriented database

## External links

- BSON Specification <sup>[1]</sup>

## References

[1] <http://bsonspec.org>

# Bookmarklet

---

A **bookmarklet** is an applet, a small computer application, stored as the URL of a bookmark in a web browser or as a hyperlink on a web page. The term is a portmanteau of the terms *bookmark* and *applet*. Whether bookmarklet utilities are stored as bookmarks or hyperlinks, they are designed to add one-click functionality to a browser or web page. When clicked, a bookmarklet performs some function, one of a wide variety such as a search query or data extraction. Usually the applet is a JavaScript program.

## Concept

Web browsers use URIs for the href attribute of the <a> tag and for bookmarks. The URI scheme, such as http:, file:, or ftp:, specifies the protocol and required form for the rest of the string. Browsers also implement a prefix javascript: that to a parser is just like any other URI. Internally, the browser sees that the protocol is *javascript*, treats the rest of the string as javascript code which is then executed, and uses the resulting string as the new page.

The executing script has access to the current page, which it may inspect and change. If the script returns an undefined type (rather than, say, a string), the browser will not load a new page, with the result that the script simply runs against the current page content. This permits in-place font size and color changes, for example, without a page reload.

An anonymous function can be used to force the script to return an undefined type:

```
javascript:(function(){  
  /* Statements returning a non-undefined type, e.g. assignments */  
})();
```

## Usage

Bookmarklets are saved and used as normal bookmarks. As such, they are simple "one-click" tools which add functionality to the browser. For example, they can:

- Modify the appearance of a web page within the browser (e.g., change font size, background color, etc.).
- Extract data from a web page (e.g., hyperlinks, images, text, etc.).
- Submit the current page to a blogging service such as Posterous, link-shortening service such as su.pr, or bookmarking service such as Delicious.
- Query a search engine, with search term(s) provided by previously selected text, or by a dialog box.
- Submit the current page to a link validation service, or translation service.
- Set commonly chosen configuration options when the page itself provides no way to do this.

"Installation of a bookmarklet" is performed by creating a new bookmark, and pasting the code into the URL destination field. Alternatively, if the bookmarklet is presented as a link, under some browsers it can be dragged and dropped onto the bookmark bar. The bookmarklet can then be run by loading the bookmark normally.

## History

Steve Kangas of [bookmarklets.com](http://bookmarklets.com) coined the term "bookmarklet",<sup>[1]</sup> which he started to create based on an idea suggested in the Netscape JavaScript Guide. The term **favelet** was used early on by Tantek Çelik on 6 September 2001 (personal email). Brendan Eich, who developed JavaScript at Netscape, gave this account of the origin of bookmarklets:

They were a deliberate feature in this sense: I invented the javascript: URL along with JavaScript in 1995, and intended that javascript: URLs could be used as any other kind of URL, including being bookmark-able.

In particular, I made it possible to generate a new document by loading, e.g. javascript:'hello, world', but also (key for bookmarklets) to run arbitrary script against the DOM of the current document, e.g. javascript:alert(document.links[0].href). The difference is that the latter kind of URL uses an expression that evaluates to the undefined type in JS. I added the void operator to JS before Netscape 2 shipped to make it easy to discard any non-undefined value in a javascript: URL.

—Brendan Eich, email to Simon Willison<sup>[2]</sup>

## Example

This example bookmarklet performs a Wikipedia search on any highlighted text in the web browser window. In normal use, the following Javascript would be installed to a bookmark in a browser<sup>[3]</sup> bookmarks toolbar. From then on, after selecting any text, clicking the bookmarklet performs the search.

```
UNIQ-source-4-0cf60905ada3d8e1-QINU
```

## See also

- Greasemonkey
- iMacros
- Ubiquity (Firefox)

## External links

- Marklets.com - Bookmarklet Directory<sup>[4]</sup> - The largest most up to date collection of bookmarklets.
- Jesse's Bookmarklets Site<sup>[5]</sup> - One of the earliest but now mostly outdated collections.
- Bookmarklets Boost Web Surfing<sup>[6]</sup>, *PC Magazine*<sup>[7]</sup>, Tara Calishain, 4 March 2004. Retrieved 31 August 2007.
- How to Use Is.Gd/Tiny Url/Bookmarklet-X in Chrome<sup>[8]</sup> - Effective method for bookmarklets in Google Chrome, 1 June 2009. Retrieved 18 July 2009
- WikiReader – bookmarklet for Wikipedia reading<sup>[9]</sup>
- Dottoro.com Client-Side JavaScript<sup>[10]</sup> – Good site for developers, with many cross-browser problem solutions.



## References

- [1] Domain bookmarklets.com (<http://www.bookmarklets.com>) registered 9 April 1998
- [2] Willison, Simon (April 10th, 2004). "Email from Brendan Eich" (<http://www.sitepoint.com/blogs/2004/04/09/bookmarklets/#comment-3424>). SitePoint. . Retrieved 22 April 2007.
- [3] Tested on Mozilla Firefox, Opera, Safari, and Chrome. Does not work in IE7 or IE8. Original source: Alex Boldt (<http://math-www.uni-paderborn.de/~axel/bookmarklet.html>)
- [4] <http://www.marklets.com/bookmarklets/>
- [5] <https://www.squarefree.com/bookmarklets/>
- [6] [http://www.pcmag.com/print\\_article2/0,1217,a=116252,00.asp](http://www.pcmag.com/print_article2/0,1217,a=116252,00.asp)
- [7] <http://pcmag.com>
- [8] <http://jonathan.dickinsons.co.za/blog/2009/06/how-to-use-isgdtiny-urlbookmarklet-x-in-chrome/>
- [9] <http://www.vcarrer.com/2010/02/wikireader-bookmarklet-for-wikipedia.html>
- [10] <http://help.dottoro.com/ljswgnnf.php>

## Client-side JavaScript

**Client-side JavaScript (CSJS)** is JavaScript that runs on the client-side. While JavaScript was originally created to run this way, the term was coined because the language is no longer limited to just client-side, for example, server-side JavaScript (SSJS) is now available.

### Environment

The most common Internet media type attribute for JavaScript source code is `text/javascript`, which follows HTML 4.01 and HTML 5 specifications and is supported by all major browsers. In 2006, `application/javascript` was also registered, though Internet Explorer versions 6 through 8 does not recognize scripts with this attribute. When no type attribute is specified in a script tag, the type value is by default `"text/javascript"` per HTML 5 specification.

To embed JavaScript code in an HTML document, it must be preceded with the `<script>` tag and followed with `</script>` (possible attribute options omitted).

Older browsers typically require JavaScript to begin with:

```
<script language="JavaScript" type="text/javascript">
<!--
```

and end with:

```
// -->
</script>
```

The `<!-- ... -->` comment markup is required in order to ensure that the code is not rendered as text by very old browsers which do not recognize the `<script>` tag in HTML documents (although `script`-tags contained within the head-tag will never be rendered, thus the comment markup is not always necessary), and the `LANGUAGE` attribute is a deprecated HTML attribute which may be required for old browsers. However, `<script>` tags in XHTML/XML documents will not work if commented out, as conformant XHTML/XML parsers ignore comments and also may encounter problems with `--`, `<` and `>` signs in scripts (for example, the integer decrement operator and the comparison operators). XHTML documents should therefore have scripts included as XML CDATA sections, by preceding them with

```
<script type="text/javascript">
//<![CDATA[
```

and following them with

```
//]]>  
</script>
```

(A double-slash // at the start of a line marks a JavaScript comment, which prevents the <![CDATA[ and ]]> from being parsed by the script.)

The easiest way to avoid this problem (and also as a best practice) is to use external script, e.g.:

```
<script type="text/javascript" src="hello.js"></script>
```

Historically, a non-standard (non-W3C) attribute language is used in the following context:

```
<script language="JavaScript" src="hello.js"></script>
```

HTML elements [1] may contain intrinsic events to which you can associate a script handler. To write valid HTML 4.01, the web server should return a 'Content-Script-Type' with value 'text/javascript'. If the web server cannot be so configured, the website author can optionally insert the following declaration for the default scripting language in the header section of the document.

```
<meta http-equiv="Content-Script-Type" content="text/javascript" />
```

## Hello World example

For an explanation of the tradition of programming "Hello World", as well as alternatives to this simplest example, see Hello world program.

This is the easiest method for a Hello world program that involves using popular browsers' support for the virtual 'javascript' protocol to execute JavaScript code. Enter the following as an Internet address (usually by pasting into the address box):

```
javascript:alert('Hello, world!');
```

## DOM binding

### User interaction

Most interaction with the user is done by using HTML forms which can be accessed through the HTML DOM. However there are also some very simple means of communicating with the user:

- Alert dialog box
- Confirm dialog box, prompt dialog box
- Status bar
- Console

### Events

Element nodes may be the source of various events which can cause an action if a JavaScript event handler is registered. These event handler functions are often defined as anonymous functions directly within the element node.

See also DOM Events and XML Events.

## Incompatibilities

**Note:** Most incompatibilities are not JavaScript issues but Document Object Model (DOM) specific. The JavaScript implementations of the most popular web browsers usually adhere to the ECMAScript standard, such that most incompatibilities are part of the DOM implementation. Some incompatibility issues that exist across JavaScript

implementations include the handling of certain primitive values like "undefined", and the availability of methods introduced in later versions of ECMAScript, such as the .pop(), .push(), .shift(), and .unshift() methods of arrays.

JavaScript, like HTML, is often not compliant to standards, instead being built to work with specific web browsers. The current ECMAScript standard should be the base for all JavaScript implementations in theory, but in practice the Mozilla family of browsers (Mozilla, Firefox and Netscape Navigator) use *JavaScript*, Microsoft Internet Explorer uses *JScript*, and other browsers such as Opera and Safari use other *ECMAScript* implementations, often with additional nonstandard properties to allow compatibility with JavaScript and JScript.

JavaScript and JScript contain several properties which are not part of the official ECMAScript standard, and may also miss several properties. As such, they are in points incompatible, which requires script authors to work around these bugs. JavaScript is more standards-compliant than Microsoft's JScript, which means that a script file written according to the ECMA standards is less likely to work on browsers based on Internet Explorer. However, since there are relatively few points of nonconformance, this is very unlikely.

This also means every browser may treat the same script differently, and what works for one browser may fail in another browser, or even in a different version of the same browser. As with HTML, it is thus advisable to write standards-compliant code.

## Combating incompatibilities

There are two primary techniques for handling incompatibilities: *browser sniffing* and *object detection*. When there were only two browsers that had scripting capabilities (Netscape and Internet Explorer), browser sniffing was the most popular technique. By testing a number of "client" properties, that returned information on computer platform, browser, and versions, it was possible for a scripter's code to discern exactly which browser the code was being executed in. Later, the techniques for *sniffing* became more difficult to implement, as Internet Explorer began to "spoof" its client information, that is, to provide browser information that was increasingly inaccurate (the reasons why Microsoft did this are often disputed). Later still, browser sniffing became something of a difficult art form, as other scriptable browsers came onto the market, each with its own platform, client, and version information.

Object detection relies on testing for the existence of a property of an object.

```
function set_image_source ( imageName, imageURL )
{
    if ( document.images ) // a test to discern if the 'document'
object has a property called 'images' which value type-converts to
boolean true (as object references do)
    {
        document.images[imageName].src = imageURL; // only executed if
there is an 'images' collection
    }
}
```

A more complex example relies on using joined boolean tests:

```
if ( document.body && document.body.style )
```

In the above, the statement "document.body.style" would ordinarily cause an error in a browser that does not have a "document.body" property, but using the boolean operator "&&" ensures that "document.body.style" is never called if "document.body" doesn't exist. This technique is called minimal evaluation.

Today, a combination of browser sniffing, object detection, and reliance on standards such as the ECMAScript specification and Cascading Style Sheets are all used to varying degrees to try to ensure that a user never sees a JavaScript error message.

## See also

- JavaScript syntax
- Bookmarklet
- DOM
- Comparison of JavaScript frameworks
- SWFObject, a JavaScript library used to embed Adobe Flash content into webpages.

## External links

### Resources

- Examples and demo of Javascript <sup>[2]</sup>
- QuirksMode - for all your browser quirks <sup>[3]</sup>
- comp.lang.javascript newsgroup <sup>[4]</sup>
- HTMLWorld - JavaScript Objects - Overview on all JavaScript objects <sup>[5]</sup> (German)
- seehowitruns.net - queryable database of browser JavaScript support <sup>[6]</sup>
- Examples of Modal Dialog boxes with Javascript <sup>[7]</sup>

### Libraries

- The Javascript Toolbox <sup>[8]</sup>: A collection of libraries solving common problems such as date pickers, color pickers, dynamic option lists, tree structures, date formatting and validation, form utils, etc.
- Free JavaScript tutorials, reference, code, and help <sup>[9]</sup>
- Small but useful JavaScript Collection <sup>[10]</sup>
- JavaScript Search Engine <sup>[11]</sup>
- JavaScript Code Generators <sup>[12]</sup>

### Tools

- Javascript whitespace stripper <sup>[13]</sup>
- c# to javascript (source generator) <sup>[14]</sup>
- Simple Javascript Tester iGoogle Gadget <sup>[15]</sup>

### Cooperation with

- PHP <sup>[16]</sup>
  - TCL <sup>[17]</sup>
  - Perl <sup>[18]</sup>
-

## References

- [1] <http://www.w3.org/TR/html4/interact/scripts.html#h-18.2.3>
- [2] <http://www.hotajax.org>
- [3] <http://www.quirksmode.org>
- [4] <http://groups-beta.google.com/group/comp.lang.javascript>
- [5] [http://www.html-world.de/program/js\\_obj.php](http://www.html-world.de/program/js_obj.php)
- [6] <http://www.seehowitruns.net>
- [7] <http://modp.com/release/jsmodaldialogs/>
- [8] <http://www.JavascriptToolbox.com/>
- [9] <http://javascript.internet.com/>
- [10] <http://www.js-vault.us>
- [11] <http://www.javascript-2.com/>
- [12] <http://www.scriptomizers.com/scripts/javascript>
- [13] <http://modp.com/release/jsstrip/>
- [14] <http://jsc.sourceforge.net/>
- [15] <http://seewah.blogspot.com/2009/11/javascript-tester-igoogle-gadget.html>
- [16] <http://pear.php.net/package-info.php?pacid=93>
- [17] <http://tcllib.sourceforge.net/doc/javascript.html>
- [18] <http://search.cpan.org/dist/Data-JavaScript/JavaScript.pm>

## CommonJS

**CommonJS** is a project with the goal of specifying an ecosystem for JavaScript outside the browser (e.g. on the server or for native desktop applications). The project was started by Kevin Dangoor in January 2009 and initially named *ServerJS*.<sup>[1]</sup>

“What I’m describing here is not a technical problem. It’s a matter of people getting together and making a decision to step forward and start building up something bigger and cooler together.”

—Kevin Dangoor<sup>[1]</sup>

In August 2009, the project was renamed *CommonJS* to show the broader applicability of the specified APIs<sup>[2]</sup>. Specifications are created in an open proposal process and voted on a mailinglist. A specification is only considered *final* after it has been implemented by several CommonJS implementations<sup>[3]</sup>. The CommonJS project is not affiliated with the Ecma International group TC39 working on ECMAScript, but some members of TC39 participate in the project.<sup>[4]</sup>

## Specifications

[5]

### Current

- Modules/1.0 (Superseded by Modules/1.1)
- Modules/1.1
- Modules/1.1.1
- Packages/1.0
- Promises/B
- Promises/C
- System/1.0

## Proposals

- Binary/B
- Binary/F
- Console
- Encodings/A
- Filesystem/A
- Filesystem/A/0
- Modules/Async/A
- Modules/Transport/B
- Packages/1.1
- Packages/Mappings
- Unit Testing/1.0

## CommonJS implementations

- CouchDB<sup>[6]</sup>
- Flusspferd<sup>[7]</sup>
- GPSEE<sup>[8]</sup>
- Joyent Smart Platform<sup>[9]</sup>
- Narwhal (JavaScript platform)<sup>[10]</sup>
- node.js<sup>[11]</sup>
- Persevere<sup>[12]</sup>
- RingoJS<sup>[13]</sup>
- SproutCore<sup>[14]</sup>
- v8cgi<sup>[15]</sup>

## External links

- CommonJS website <sup>[16]</sup>
- CommonJS effort sets JavaScript on path for world domination <sup>[17]</sup> (article on Ars Technica)

## References

- [1] <http://www.blueskyonmars.com/2009/01/29/what-server-side-javascript-needs/>
  - [2] <http://commonjs.org/history/>
  - [3] <http://wiki.commonjs.org/wiki/ProposalProcess>
  - [4] <http://www.blueskyonmars.com/2010/01/29/commonjs-the-first-year/>
  - [5] <http://commonjs.org/specs/>
  - [6] <http://wiki.commonjs.org/wiki/Implementations/CouchDB>
  - [7] <http://wiki.commonjs.org/wiki/Implementations/Flusspferd>
  - [8] <http://wiki.commonjs.org/wiki/Implementations/GPSEE>
  - [9] <http://wiki.commonjs.org/wiki/Implementations/Smart>
  - [10] <http://wiki.commonjs.org/wiki/Implementations/Narwhal>
  - [11] <http://wiki.commonjs.org/wiki/Implementations/node.js>
  - [12] <http://wiki.commonjs.org/wiki/Implementations/Persevere>
  - [13] <http://wiki.commonjs.org/wiki/Implementations/RingoJS>
  - [14] <http://wiki.commonjs.org/wiki/Implementations/SproutCore>
  - [15] <http://wiki.commonjs.org/wiki/Implementations/v8cgi>
  - [16] <http://commonjs.org/>
  - [17] <http://arstechnica.com/web/news/2009/12/commonjs-effort-sets-javascript-on-path-for-world-domination.ars>
-

# Comparison of JavaScript frameworks

	<div>dhtmlx</div> <div>[1]</div>	Dojo	Echo3	Ext JS	Google Web Toolkit	jQuery	midori	MochiKit	MooTools	Prototype & script.aculo.us	[2]	Pyjamas	qooxdoo	Rialto Toolkit	Rico	SmartClient & SmartGWT	SweetDEV RIA	[3]	YUI															
compared	2.5 11 Nov 2009	1.5.0 15 Jul 2010	3.0.beta8 August 6, 2009	3.1.1 17 December 2009	1.7.1 July 2009	<div>[4]</div> 1.4.2 13 Feb 2010	2010.05 10 May 2010	1.4.2 17 Nov 2008	1.2.4 19 Oct 2009	1.6.1/1.8.3 14 Nov 2009	0.5 Mar 2009	1.1 28 Apr 2010	1.0 30 May 2008	2.0	SmartClient 7.0 August 2009 SmartGWT 1.2 August 2009	3.1 9 Jun 2008	3.0 29 Sept 2009	5.0 29 Sept 2009																
	Variable	Variable; Base size: 65 KB minified, 28 KB minified and gzipped, 123 KB uncompressed	[5]	84–502 KB	Variable	72 KiB minified, 24 KiB minified and gzipped, 155 KiB uncompressed	50 KB uncompressed, 9 KB minified and gzipped	32–200 KB	Variable 7.3-65 KiB (YUI Compressor) 101 KiB uncompressed	[6] [7]	46–278 KB	Variable	Variable, starting at 6 KB gzipped	520 KB		100-500kb gzipped.	550 KB	Variable; library core is 31KB	V															
	GPL & Commercial	[8]	BSD & AFL	MPL, LGPL or GPL	Depends Commercial & GPL 3.0	[9]	Apache License	MIT & GPL	MIT License	MIT & AFL	MIT License	MIT License	Apache 2 License & GPL	LGPL & EPL	Apache License	Apache License	LGPL & Commercial	[10]	Apache 2 License	BSD License	L													
	Samples Explorer Demo Apps	[11] [12]	Feature Explorer Demos Documentation and 300+ Examples API Cheatsheet API Docs	[13] [14] [15] [16] [17]	Client-Side JavaScript Demo	[18]	Samples & Demos	[19]	GWT Examples	[20]	UI demo Documentation	[21] [22]	midori Documentation	[23]	Effects Demos	[24]	Demos	[25]	Effects Demos and Example Game	[26]	pyjs.org examples	[27]	qooxdoo demo	[28]	Demos	[29]	Demos	[30]	SmartClient Showcase SmartGWT Showcase SmartGWT EE Showcase	[31] [32] [33]	Getting Started	[34]	300 examples including adv. app example	[35] [36]
Features																																		
[38]	No	[39] No	No	No	[40] No	[41] Yes	[42] No	[43] No	[44] No	[45] No			No	[46] No	No	Partial	[47]		[48] No	N														
[50]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes															
TPRequest level	Yes	[51] Yes		Yes	Yes	Yes	Yes	Yes	[52] Yes	Yes	Yes	Yes	Yes	No	[53] Yes	Yes	Yes	Yes	Yes															
ta	Yes	[54] Yes		Yes	Yes	Yes	Yes	Yes	[55] Yes	Yes	Yes	[56] Yes	Yes	No	Yes	Yes	Yes	Yes	Yes															
ush data		[57] Yes			No											[58] Yes				Y														

ata	Yes: XML, CSV	Yes: XML, HTML, CSV, ATOM, [60] etc.		Yes: XML	Yes: RPC	Yes: XML, HTML			Yes: XML, HTML						Yes: XML, WSDL, RSS, and Java-based SQL, Hibernate, POJO adapters		
drop	Yes	[61] Yes	Yes	Yes	With plugin [62]	Yes	Yes	Yes	[63] Yes	Yes	Depends	Yes	Yes	Yes	Yes	Depends [64]	Yes
visual	No	[65] Yes		Yes	Yes	Yes	Yes	Yes	[66] Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes
on / d visual	No	[67] Yes	Yes	Yes	Yes	Yes	Yes	[68] Yes	[69] Yes	Yes		Yes	No	Yes	Yes	No	Yes
ndling	Yes	[70] Yes		Yes	Yes	Yes	Yes	Yes	[71] Yes	Yes	[72] Yes	Yes	Yes	Yes	Yes	[73] Depends	Yes
ton / ment	No	[74] Yes		[75] Yes	Yes	With plugins [76]	Yes		With plugin [77]	Yes	[78] Yes	Yes	No	Yes	Yes	No [79]	Yes
m widgets tion	[80] Yes	[81] Yes	Yes	Yes	Yes, Validation requires plugin [82]	With plugins [83]	Yes		Yes	Yes	[78] Yes	Yes	Yes	Yes	Yes	Yes	Yes
	dhtmlmix [1]	Dojo	Echo3	Ext	Google Web Toolkit	jQuery	midori	MochiKit	MooTools	Prototype & script.aculo.us [2]	Pyjamas	qooxdoo	Rialto Toolkit	Rico	SmartClient and SmartGWT	SweetDEV RIA [3]	YUI
	[84] Yes	[85] Yes	Yes	Yes	Yes	With plugins [86]	No		With plugin [87]		Yes	Yes	Yes	[88] Yes	Yes	Yes	[89] Yes
ical Tree	[90] Yes	[91] Yes		[92] Yes	Yes	With plugins [93]	No		With plugins [94]		[95] Yes	[96] Yes	Yes		Yes	Yes	[97] Yes
t editor	[98] Yes	[99] Yes	[100] Yes	[101] Yes	Yes	With plugins [102]	No		[103] Yes		Yes	Yes	No	No	Yes	No	Yes
pletion	[104] Yes	[105] Yes		Yes	Yes	With plugins [106]	Yes		With plugin [107]	Yes	Yes		Yes	Yes	Yes	Yes	Yes
eneration	No	[108] Yes		Yes	Yes	Yes	No		[109] Yes	Yes	Yes		Yes	Yes	Yes	No	
themeable le	[110] Yes	[111] Yes		[112] Yes	[113] Yes	[114] Yes			Yes		Yes	[115] Yes			Yes		[116] Yes
izable nd modal	[117] Yes	Yes		Yes	Yes	With plugins			[118] Yes			Yes			Yes		Yes
ge layout	[119] Yes	Yes		Yes	Yes	With plugin [120]			[118] Yes			Yes			Yes		
support									[121] Yes		Yes	Yes			Yes		



ility / ion	No [123]	Yes		No [124]	Yes [125]	Yes						No [126]			Degradation: No Accessibility: Yes	No [127]	Yes
ompliant		Yes [128]				In development [129] [130]						No			Yes	No	Yes
er tools		Yes [131]	In development [132]	Yes [133]	Yes	Yes [134] [135]		Yes [136] [137]		No	Yes [138]				Yes	No	Yes [139]
orage	No	No [141]		Via Google Gears or Adobe Air	Via Google Gears [142]	No	No			Via Pyjamas-Desktop [143]	Via Google Gears	No			Via Google Gears or Adobe Air	No	
rowser 2d [144]		Yes [145]						Yes [146]		Yes					Yes		
& rd	Yes [148]	Yes [149]				With plugin [150]									Yes		Yes [151]
	dhtmlx [1]	Dojo	Echo3	Ext	Google Web Toolkit	jQuery	midori	MochiKit	MooTools	Prototype & script. aculo.us [2]	Pyjamas	qooxdoo	Rialto Toolkit	Rico	SmartClient and SmartGWT	SweetDEV RIA [3]	YUI

Browser Support

Explorer	6+	6-8	6+	6+	6+	[152] 6+	6+	6	6+	6+	6+	6+	6+	5.5+	6+	6+	6+	6
Firefox	1+	[153] 3+	1.5+	1.5+	1+	[154] 2+	1.5+	1.0.7, 1.5b2	1.5+	1.5+	1+	2+	1.5+	1+	1+		3+ [155]	2
	2.0+	[156] 4	3+	3+	3+	[157] 3+	2+	2.0.2	2+	2.0.4+	2+	3+		2.0.3 [158]	3+	3+	4.0	3
	9+	[159] 10	9+	9+	9+	[160] 9+	9+	8.5	9+	9.25+	9+	9+	9+		9+	9.21+, possibly earlier as well	10.0+	9
	1+	[161] 3	1+			[162] 1+			1+	1+ (starting with 1.6.1RC3)		2+			1+			2

## See also

List of JavaScript libraries

## External links

- Performance tests for different JavaScript Frameworks (Dojo, jQuery, Mootools, qooxdoo, Prototype, YUI), Jan 8, 2010 <sup>[163]</sup>
- jQuery vs Mootools by Aaron Newton, May, 2009 <sup>[164]</sup>
- Slashdot: jQuery compares favorably to MooTools <sup>[165]</sup>
- StackOverflow: jQuery vs. Prototype+Script.aculo.us and MooTools <sup>[166]</sup>
- Dojo vs JQuery vs MooTools vs Prototype Performance Comparison (not up-to-date, outdated versions of YUI, jQuery and also Dojo Toolkit used)| Peter Velichkov's Blog - Jan 19, 2009 <sup>[167]</sup>

- Compare JavaScript frameworks - Ext JS, JQuery, MooTools <sup>[168]</sup>
- JavaScript frameworks survey results | Kyle Hayes' Blog - Mar 29, 2009 <sup>[169]</sup>
- Which JavaScript libraries are most used on the most popular websites? <sup>[170]</sup>

## References

- [1] <http://dhtmlx.com/>
- [2] [script.aculo.us](http://script.aculo.us) is an add-on to Prototype.
- [3] <http://sweetdev-ria.sourceforge.net/>
- [4] <http://blog.jquery.com/2010/02/19/jquery-142-released/>
- [5] Dojo Base (<http://www.dojotoolkit.org/downloads>)
- [6] <http://mootools.net/get>
- [7] <http://mootools.net/download>
- [8] <http://dhtmlx.com/docs/products/licenses.shtml>
- [9] <http://extjs.com/store/extjs>
- [10] <http://www.smartclient.com/product/index.jsp>
- [11] <http://www.dhtmlx.com/docs/products/docsExplorer/samples.shtml>
- [12] <http://www.dhtmlx.com/docs/products/demoApps/index.shtml>
- [13] <http://dojocampus.org/explorer>
- [14] <http://dojotoolkit.org/demos>
- [15] <http://docs.dojocampus.org/>
- [16] <http://download.dojotoolkit.org/release-1.3.0/cheat.html>
- [17] <http://api.dojotoolkit.org/>
- [18] <http://demo.nextapp.com/echo3csjs>
- [19] <http://extjs.com/deploy/dev/examples/samples.html>
- [20] <http://code.google.com/webtoolkit/examples/>
- [21] <http://ui.jquery.com/demos>
- [22] <http://docs.jquery.com>
- [23] <http://www.midorijs.com/docs.html>
- [24] <http://www.mochikit.com/demos.html>
- [25] <http://demos.mootools.net/>
- [26] <http://github.com/madrobby/scriptaculous/wikis/demos>
- [27] <http://pyjs.org/examples/>
- [28] <http://demo.qooxdoo.org>
- [29] <http://rialto.improve-technologies.com/rialtoDemo/demo2/demoRialto.html>
- [30] <http://demos.openrico.org>
- [31] <http://www.smartclient.com/#Welcome>
- [32] <http://www.smartclient.com/smartgwt/showcase/>
- [33] <http://www.smartclient.com/smartgwt/tee/showcase/>
- [34] <http://demo.sweetdev-ria.com>
- [35] <http://developer.yahoo.com/yui/examples/>
- [36] [http://developer.yahoo.com/yui/examples/layout/adv\\_layout\\_source.html](http://developer.yahoo.com/yui/examples/layout/adv_layout_source.html)
- [37] <http://www.zkoss.org/demo/>
- [38] Feature detection is preferred by many to browser sniffing to support future browsers: Browser Detecting (and what to do Instead) ([http://www.jibbering.com/faq/faq\\_notes/not\\_browser\\_detect.html](http://www.jibbering.com/faq/faq_notes/not_browser_detect.html)), Feature Detection: State of the Art Browser Scripting (<http://peter.michaux.ca/articles/feature-detection-state-of-the-art-browser-scripting>), Browser Feature Detection ([https://developer.mozilla.org/En/Browser\\_Feature\\_Detection](https://developer.mozilla.org/En/Browser_Feature_Detection))
- [39] Integrate of Feature Detection Code (<http://trac.dojotoolkit.org/ticket/8947>)
- [40] GWT implementations for every browser (<http://code.google.com/p/google-web-toolkit/source/browse/#svn/trunk/user/src/com/google/gwt/user/client/impl>)
- [41] jQuery 1.3 ([http://docs.jquery.com/Release:jQuery\\_1.3](http://docs.jquery.com/Release:jQuery_1.3))
- [42] <http://www.midorijs.com/midori.js> midori.js
- [43] <http://www.mochikit.com/MochiKit/Style.js> MochiKit/Style.js
- [44] <http://mootools.net/download/get/mootools-1.2.3-core-nc.js> mootools-1.2.3-core-nc.js
- [45] Feature detection all the way (<http://thinkweb2.com/projects/prototype/feature-detection-all-the-way/>)
- [46] <https://svn.improve.fr/rialto/Rialto-coreV1/rialtoEngine/javascript/rialto.js>
- [47] Author disagrees that feature detection alone is sufficient
- [48] <http://yui.yahooapis.com/3.0.0/build/yui/yui.js> yui.js

- [49] <http://zssdemo.zkoss.org/zkau/web/js/zk/zk.js.dsp> zk.js
- [50] kangax (5 April 2010). "What's wrong with extending the DOM" (<http://perfectionkills.com/whats-wrong-with-extending-the-dom/>). . Retrieved 6 April 2010.
- [51] AJAX and Dojo (<http://docs.dojocampus.org/quickstart/ajax>) [52]
- MooTools Request (<http://mootools.net/docs/Request/Request>) [53]
- Inner HTML demo ([http://demos.openrico.org/inner\\_ajax\\_HTML](http://demos.openrico.org/inner_ajax_HTML)) [54]
- <http://docs.dojocampus.org/dojo/data>
- [55] MooTools JSON (<http://docs.mootools.net/Utilities/JSON>)
- [56] Pyjamas JSON-RPC Example (<http://pyjs.org/examples/jsonrpc/output/JSONRPCExample.html>)
- [57] <http://cometd.org/>
- [58] Real-time Messaging Module ([http://www.smartclient.com/#\\_Applications\\_IRC.Chat.\(real-time.messaging\)](http://www.smartclient.com/#_Applications_IRC.Chat.(real-time.messaging)))
- [59] [http://docs.zkoss.org/wiki/Server\\_Push\\_with\\_a\\_Stock\\_Chart\\_Example#Long\\_polling](http://docs.zkoss.org/wiki/Server_Push_with_a_Stock_Chart_Example#Long_polling)
- [60] <http://docs.dojocampus.org/dojox/data>
- [61] <http://docs.dojocampus.org/dojo/dnd>
- [62] <http://code.google.com/p/gwt-dnd/>
- [63] MooTools Drag.Move Plugin (<http://mootools.net/docs/Plugins/Drag.Move>)
- [64] For nodes in treeview only
- [65] <http://docs.dojocampus.org/dojo/fx/>
- [66] MooTools Fx.Tween (Simple CSS Transitions) (<http://mootools.net/docs/Fx/Fx.Tween#Fx-Tween>)
- [67] <http://docs.dojocampus.org/dojox/fx/>
- [68] Mochikit.Visual (<http://www.mochikit.com/doc/html/MochiKit/Visual.html>)
- [69] MooTools Fx.Morph (Advanced CSS Transitions) (<http://docs.mootools.net/Fx/Fx.Morph#Fx-Morph>)
- [70] <http://docs.dojocampus.org/quickstart/events>
- [71] <http://docs.mootools.net/Native/Event#Event> MooTools Events
- [72] Pyjamas demo of onclick event handling (<http://pyjs.org/examples/onclicktest/output/OnClickTest.html>)
- [73] No event management between components on client side, server side only for some basic event
- [74] <http://docs.dojocampus.org/dojo/back>
- [75] <http://extjs.com/products/extjs/roadmap.php> Since v2.2
- [76] History (<http://plugins.jquery.com/project/history>), jquery History ([http://www.mikage.to/jquery/jquery\\_history.html](http://www.mikage.to/jquery/jquery_history.html)), History/Remote (<http://www.stilbuero.de/jquery/history/>)
- [77] Digitalald's HistoryManager Plugin (<http://digitalald.de/project/history-manager/>)
- [78] Pyjamas Kitchen Sink demo (<http://pyjs.org/examples/kitchensink/output/KitchenSink.html>)
- [79] Back button support not in the demo, and no documentation about it, or about browser history support (<http://www.google.com/search?q=site:http://sweetdev-ria.sourceforge.net+back+history>)
- [80] dhtmlxForm (<http://www.dhtmlx.com/blog/?p=258>)
- [81] <http://docs.dojocampus.org/dijit/form>
- [82] <http://techblog.maydu.eu/?p=7>
- [83] <http://docs.jquery.com/Plugins>
- [84] dhtmlxGrid (<http://www.dhtmlx.com/docs/products/dhtmlxGrid/>)
- [85] <http://docs.dojocampus.org/dojox/grid>
- [86] jqGrid (<http://www.trirand.com/blog/>), Ingrid (<http://rekonstrukt.com/ingrid/>), Flexigrid (<http://www.flexigrid.info/>)
- [87] JxLib Library (<http://jxlib.org/>) or phatfusion sortable table plugin (<http://www.phatfusion.net/sortabletable/index.htm>) or DrasticGrid (<http://www.drasticdata.nl/DDHome.php?m=3>)
- [88] Live grid (<http://demos.openrico.org/livegrid>)
- [89] YUI DataTable (<http://developer.yahoo.com/yui/datatable>)
- [90] dhtmlxTree (<http://www.dhtmlx.com/docs/products/dhtmlxTree/>)
- [91] <http://docs.dojocampus.org/dijit/Tree>
- [92] Tree demos (<http://extjs.com/depoy/dev/examples/samples.html#sample-4>)
- [93] treeview (<http://jquery.bassistance.de/treeview/demo/>), file\_tree\_viewer (<http://abeautifulsite.net/notebook/58>)
- [94] MooTree 2 (<http://sites.google.com/a/mindplay.dk/mootree/>) or JxLib Library (<http://jxlib.org/>)
- [95] <http://pyjs.org/examples/kitchensink/output/KitchenSink.html#Trees> Tree
- [96] [http://demo.qooxdoo.org/current/demobrowser/Demo\\_browser](http://demo.qooxdoo.org/current/demobrowser/Demo_browser)
- [97] [http://developer.yahoo.com/yui/treeview/YUI\\_TreeView](http://developer.yahoo.com/yui/treeview/YUI_TreeView)
- [98] dhtmlxEditor (<http://www.dhtmlx.com/docs/products/dhtmlxEditor/>)
- [99] <http://docs.dojocampus.org/dijit/Editor>
- [100] Echo 3 new features (<http://wiki.nextapp.com/echo3NewFeatures>)
- [101] problems with the HTML5Editor creating XHTML vs the HTML it does now (<http://extjs.com/forum/showthread.php?p=129748#post129748>)

- [102] markitup (<http://markitup.jaysalvat.com/home/>), jwysiwyg (<http://code.google.com/p/jwysiwyg/>), htmlbox (<http://remiya.com/cms/projects/jquery-plugins/htmlbox/>), WYMeditor (<http://www.wymeditor.org/en/>)
- [103] MooEditable (<http://cheeaun.github.com/moeditable/>)
- [104] dhtmlxCombo (<http://www.dhtmlx.com/docs/products/dhtmlxCombo/>)
- [105] <http://docs.dojocampus.org/dijit/form/ComboBox>
- [106] Autocomplete (<http://plugins.jquery.com/project/autocomplete>), Dylans Autocomplete (<http://plugins.jquery.com/project/ac>), Yet Another Autocomplete ([http://plugins.jquery.com/project/YA\\_AutoComplete](http://plugins.jquery.com/project/YA_AutoComplete)), jQuery plugin for Autocomplete (<http://plugins.jquery.com/project/jqac>), JQuery Autocomplete (<http://plugins.jquery.com/project/jq-autocomplete>), jquery.suggest (<http://plugins.jquery.com/project/suggest>), Interface Autocompleter (<http://interface.eyecon.ro/demos/autocomplete.html>)
- [107] Digitalald's Autocompleter Plugin (<http://digitalald.de/project/autocomplete/>)
- [108] As of Dojo 1.3, the `NodeList.addContent()` api can be used to create content for example: `dojo.query(".note").addContent("h4 NOTE: h4", "before");`
- [109] MooTools Elements (<http://docs.mootools.net/Element/Element#Element>)
- [110] SkinBuilder (<http://www.dhtmlx.com/docs/products/skinBuilder/index.shtml>)
- [111] <http://docs.dojocampus.org/dijit-themes>
- [112] Community-driven skins (<http://extjs.com/blog/2008/03/03/can-ext-be-skinned-of-course/>), Theme Builder (<http://extjs.com/blog/2008/04/07/spket-ide-1611-released-includes-new-ext-theme-builder>)
- [113] Skinning GWT controls with style sheets (<http://code.google.com/webtoolkit/documentation/com.google.gwt.doc.DeveloperGuide.UserInterface.html#StyleSheets>)
- [114] ThemeRoller (<http://ui.jquery.com/themeroller>)
- [115] [http://qooxdoo.org/documentation/ui\\_theming](http://qooxdoo.org/documentation/ui_theming)
- [116] Skinning YUI (<http://developer.yahoo.com/yui/articles/skinning>)
- [117] dhtmlxWindows (<http://www.dhtmlx.com/docs/products/dhtmlxWindows/>)
- [118] Using MochaUI Library (<http://mochaui.com/demo/>) or JxLib (<http://jxlib.org/>)
- [119] dhtmlxLayout (<http://www.dhtmlx.com/docs/products/dhtmlxLayout/>)
- [120] (<http://code.google.com/p/css-template-layout/>)
- [121] MochaUI Library (<http://ajaxian.com/archives/mocha-ui-mootools-canvas-ui-class>)
- [122] All JavaScript frameworks can be written in an accessible way with graceful degradation, frameworks seen here which imply out-of-the-box accessibility have made a special effort to document best practices for their particular framework.
- [123] <http://docs.dojocampus.org/quickstart/writingWidgets/a11y>
- [124] Section 508 accessibility improvements – v3.0 roadmap (<http://extjs.com/products/extjs/roadmap.php>)
- [125] Built-in Accessibility in GWT 1.5 Applications (<http://googlewebtoolkit.blogspot.com/2008/09/built-in-accessibility-in-gwt-15.html>)
- [126] Qooxdoo's "Extreme JavaScript" paradigm (<http://codecorps.wordpress.com/2007/04/11/qooxdoo-funny-name-interesting-ajax-toolkit/>)
- [127] Yahoo! UI Graded Browser Support (<http://developer.yahoo.com/yui/articles/gbs>)
- [128] <http://docs.dojocampus.org/quickstart/writingWidgets/a11y>
- [129] Ticket #2514 Add ARIA semantics to UI (<http://dev.jquery.com/ticket/2514>)
- [130] Fluid Project's Michelle D'Souza is working on ARIA support (<http://jqueryui.com/about>)
- [131] Dojo Development Tools (<http://www.dojotoolkit.org/book/dojo-book-0-9/part-4-meta-dojo/development-tools>)
- [132] EchoStudio 3 (<http://www.nextapp.com/products/echostudio>)
- [133] GUI builder (<http://www.projectspace.nl>), Theme Builder (<http://extjs.com/blog/2008/04/07/spket-ide-1611-released-includes-new-ext-theme-builder>), Ext on Air (<http://code.google.com/p/exteditor/>), Custom build tool (<http://extjs.com/products/extjs/build/>)
- [134] Netbeans has jQuery support (<http://netbeans.org/kb/docs/web/js-toolkits-jquery.html>)
- [135] jQuery API (<http://api.jquery.com/>)
- [136] MooTools-core Documentation (<http://mootools.net/docs/core>)
- [137] (<http://mootools.net/docs/more/MooTools-more/Document>)
- [138] Custom build, JS Linker, API generation, Unit test framework, etc. (<http://qooxdoo.org/about/framework>)
- [139] CSS Grid Builder (<http://developer.yahoo.com/yui/grids/builder/>), YUI Test Utility (<http://developer.yahoo.com/yui/yuitest>), Profiler (<http://developer.yahoo.com/yui/profiler>), Logger Control (<http://developer.yahoo.com/yui/logger>)
- [140] JavaScript frameworks currently only have the ability to support offline storage by taking advantage of pre-installed browser extensions such as Google Gears. If a user does not have one of these supported extensions installed in their browser already then offline support will be unavailable to the framework. JavaScript framework developers interested in implementing Gears may want to start with the Gears Getting Started Guide (<http://code.google.com/apis/gears/design.html>).
- [141] o.DojoToolkit.Org/offline (<http://o.dojotoolkit.org/offline>) says "Dojo Offline is no longer supported after Dojo 1.3"
- [142] Google API Libraries for Google Web Toolkit (<http://code.google.com/p/gwt-google-apis/>)
- [143] Pyjamas Desktop (<http://pyjd.org>)
- [144] Some JavaScript libraries provide 2d graphics primitives that can be used for cross-browser vector graphics. These libraries rely on underlying technologies in the browser or plugins such as Canvas, SVG, VML, Flash, and Silverlight to do the actual rendering, but help

- isolate application and widget code from the rendering engine api differences
- [145] <http://docs.dojocampus.org/dojox/gfx>
  - [146] MooTools ART (<http://github.com/kamicane/art>)
  - [147] Some JavaScript libraries include widgets for Charts, Gauges, and other data visualizations.
  - [148] dhtmlxChart (<http://www.dhtmlx.com/blog/?p=412>)
  - [149] <http://docs.dojocampus.org/dojox/charting>
  - [150] jQuery Visualize Plugin ([http://www.filamentgroup.com/lab/jquery\\_visualize\\_plugin\\_accessible\\_charts\\_graphs\\_from\\_tables\\_html5\\_canvas/](http://www.filamentgroup.com/lab/jquery_visualize_plugin_accessible_charts_graphs_from_tables_html5_canvas/))
  - [151] (<http://developer.yahoo.com/yui/charts/>)
  - [152] [http://docs.jquery.com/Browser\\_Compatibility](http://docs.jquery.com/Browser_Compatibility)
  - [153] <http://docs.dojocampus.org/releasenotes/1.4>
  - [154] [http://docs.jquery.com/Browser\\_Compatibility](http://docs.jquery.com/Browser_Compatibility)
  - [155] <http://developer.yahoo.com/yui/articles/gbs/>
  - [156] <http://docs.dojocampus.org/releasenotes/1.4>
  - [157] [http://docs.jquery.com/Browser\\_Compatibility](http://docs.jquery.com/Browser_Compatibility)
  - [158] <http://openrico.org/resources>
  - [159] <http://docs.dojocampus.org/releasenotes/1.4>
  - [160] [http://docs.jquery.com/Browser\\_Compatibility](http://docs.jquery.com/Browser_Compatibility)
  - [161] <http://docs.dojocampus.org/releasenotes/1.4>
  - [162] [http://docs.jquery.com/Browser\\_Compatibility](http://docs.jquery.com/Browser_Compatibility)
  - [163] <http://dante.dojotoolkit.org/taskspeed/report/charts.html?exclude=defaultbrowser0>
  - [164] <http://jqueryvsmootools.com/>
  - [165] <http://books.slashdot.org/article.pl?sid=08/11/26/1414236>
  - [166] <http://stackoverflow.com/questions/176324/why-does-everyone-like-jquery-more-than-prototypescriptaculous-or-mootools-or-wh>
  - [167] <http://blog.creonfx.com/javascript/mootools-vs-jquery-vs-prototype-vs-yui-vs-dojo-comparison-revised>:
  - [168] <http://www.ibm.com/developerworks/web/library/wa-jsframeworks/index.html>
  - [169] <http://www.kylehayes.info/2009/03/29/survey-results-javascript-frameworks>:
  - [170] <http://trends.builtwith.com/javascript>

## Comparison of JavaScript-based source code editors

This article provides basic feature comparison between some of the JavaScript-based source code editors available today. This article is not all-inclusive or necessarily up-to-date.

### Overview

List of source code editors

	Site	Latest version	Style / Clone of	Cost (US\$)	Software license	Open source	Browser compatibility
<b>CodeMirror</b>	Home <sup>[1]</sup> , Demo <sup>[2]</sup>	0.7, 2010-06-21	Emacs / regular textarea	Free	BSD-like license	Yes <sup>[3]</sup>	Firefox 1.5+, Internet Explorer 6+, Safari 3+, Opera 9.52+, <sup>[4]</sup> Chrome
<b>CodePress</b>	dead home page <sup>[5]</sup> , Demo <sup>[6]</sup> , SourceForge <sup>[7]</sup>	0.9.6, 2007-09-26	Microsoft Visual Studio	Free	LGPL	Yes	?
<b>CodeTextArea</b>	Home page <sup>[8]</sup> , Demo <sup>[9]</sup> , Experimental demo <sup>[10]</sup>	Jun. 2009.	Microsoft Visual Studio	Free	BSD license	Yes	?

<b>EditArea</b>	Home <sup>[11]</sup> , Demo <sup>[12]</sup>	0.8.2, 2010-01-14	Microsoft Visual Studio	Free	LGPL	Yes	IE 6+, Firefox 1.5+, Safari 3+, Opera 9+ and Chrome <sup>[13]</sup>
<b>Helene</b>	Home <sup>[14]</sup> , Demo <sup>[15]</sup>	0.9, unknown release date	Microsoft Visual Studio	Free	GPL	Yes	
<b>Markitup</b>	Home <sup>[16]</sup> , Demo <sup>[17]</sup>	1.1.6, 2010-01-12	markup editor, no syntax highlight	Free	MIT, GPL	Yes	IE 6 & 7, Firefox 2 & 3, Safari 3.1, Opera 9+ <sup>[18]</sup>
<b>9ne</b>	Home Page <sup>[19]</sup>	?	emacs	Free	GPL	Yes	
<b>jsvi</b>	Home Page <sup>[20]</sup>	?	vi	Free	GPL	Yes	
<b>Ymacs</b>	Home <sup>[21]</sup> , Demo <sup>[22]</sup>	0.4, 2010-Jan-17	emacs	Free	BSD-like license	Yes <sup>[23]</sup>	Firefox, Chrome, Safari
<b>MDK-Editor</b>	Home Page <sup>[24]</sup>	2.10, 2008	Microsoft Visual Studio	Depends on the use	Dual license	Code is readable	tested to work on: IE 6, 7 - Firefox 2, 3 - Chrome
<b>Bespin</b>	Home <sup>[25]</sup> , Demo <sup>[26]</sup>	0.9a1, 2010-06-29	Emacs / regular textarea	Free	MPL	Yes <sup>[27]</sup>	Firefox 3.5+, Safari 4+, Chrome

## List of features

Feature testing was performed with Firefox 3.0.6 against the current demo version, and results may not match those in other browsers or downloadable versions.

### List of source code editor features

	<b>CodeMirror</b>	<b>CodePress</b>	<b>CodeTextArea</b>	<b>EditArea</b>	<b>Helene</b>	<b>markItUp!</b>	<b>MDK-Editor</b>	<b>Bespin</b>
<b>Implementation</b>	nestable full parsers	pattern-based parser					parsers	
<b>Syntax highlight</b>	JS, CSS, XML, PHP <sup>[28]</sup> , SPARQL <sup>[29]</sup> , community-extended with new parsers: Python <sup>[30]</sup> , Lua <sup>[31]</sup> , Ruby <sup>[32]</sup>	limited mixed + JavaScript (no CSS), PHP + HTML (no JavaScript or CSS), Java, Perl, SQL	only keywords	only one language at a time <sup>[33]</sup> Perl, : PHP, CSS, Javascript, Python, HTML, XML, VB, C, CPP, SQL, Pascal, Basic, Brainf*ck	PHP	No	mixed mode: PHP + HTML + JavaScript + CSS, single-mode: PHP, Javascript, CSS, XML; extensible	mixed & single mode: PHP, HTML, JavaScript, CSS, Python, Arduino, C, C#, Ruby; extensible
<b>Syntax checking</b>	basic; more comprehensive for PHP <sup>[34]</sup>	No			No		HTML, JavaScript (using JSLint)	HTML, JavaScript (using JSLint)
<b>Indent, new line keeps level</b>	Yes	very limited	No	Yes	N/A (can't press Enter)	No	Yes	optional setting that is off by default
<b>Indent, syntax</b>	Yes			No			No	Yes

<b>Indent, selected block</b>	either automatically, or block-level indent/unindent	No		yes, including Shift+Tab			yes, including Shift+Tab and using context menu	yes, including Shift+Tab and using context menu
<b>Bracket matching</b>	Ctrl+[ after the bracket; no angle bracket matching	an implementation exists with mouse-hover bracket matching	Ctrl+B; no angle bracket matching			No	matching bracket ([{<>}]) always highlighted	
<b>XML matching tag highlight</b>	No	No					Yes	
<b>Code folding</b>	No	No	No	No	No	No	No	No
<b>Code snippets</b>	example using API [35]	type 'for' or 'if' then Tab	No	Yes	No	Yes	JavaScript	No
<b>Code suggestion</b>	No	No		yes		No	CSS, HTML, JavaScript)	No
<b>Toggle syntax highlight on/off</b>	No			last example in demo [12]		N/A	textmode	No
<b>Keyboard shortcuts</b>	basic [35]			some common used: Ctrl+f, Ctrl+g, Ctrl+z, Ctrl+y		Yes	All key combos (except F1 in IE7) can be bound to shortcuts	All Common Shortcuts [36] & Custom Keybindings [37]
<b>Line numbers</b>	Yes	Yes	Yes	Yes	Yes	No	supports mouse selection	Yes
<b>Search &amp; replace</b>	via API [35]		No	toolbar button			has API for the studio	regex supported
<b>Spell checking</b>	browser-based	browser-based	none	browser-based	none	browser-based	No	No
<b>Toolbar</b>	example using API [38], demo [35]		No	Yes	No	Yes	No	But command line console
<b>Visual styling</b>			font-type and font-size				5 styles to choose from, having 2 font-sizes	Fully theme-able [39]
<b>Undo/Redo</b>	Yes	Yes		Yes			Infinite diff-based, with visual update	Yes

<b>Usability</b>	Initial parse is slow, further performance is independent of document size	new text sometimes syntax highlighted only after going out of the viewable area	No PageDown/PageUp	text ghosting if syntax highlight enabled	can't type Enter		no documentation	Editor is fast and responsive even on 100k+ lines of code in a single file. Only one on a webpage.
------------------	--	---	--------------------	---	------------------	--	------------------	--

## TODO: Other aspects

- Add Amy Editor (see also embedded version <sup>[40]</sup>) <http://www.amyeditor.com/>
- What size files do these editors handle best
- Dependencies: Certain JavaScript library? How many files does it require? When compressed, what is the minimum download size in kilobytes?
- Tabbed editing?
- Javascript architecture (prototype, functional, or closure-based)
- How responsive is each library (with and without a large file loaded)? (Not sure how to measure this yet.)
- How long is the delay before syntax-highlighting occurs on new text?

## Extensibility features

- Feature plugins
- Interface languages (English, French, etc.)
- Syntax highlighting plugins
- Bracket completion plugins
- Language snippet plugins
- Code suggestion plugins

## Offspring projects

Below is a list of projects based on each engine.

### CodeMirror powered

- ShiftEdit Web-based IDE <sup>[41]</sup>
- Google's API playground <sup>[42]</sup>
- r3 <sup>[43]</sup>, a template management engine developed by Yahoo!
- FireRainbow <sup>[44]</sup> - JavaScript syntax highlighting for Firebug
- Freebase's Acre IDE <sup>[45]</sup>
- Google Earth KML sampler <sup>[46]</sup>
- Eloquent JavaScript's console <sup>[47]</sup>
- A cool tutorial about the <canvas> element <sup>[48]</sup>
- An online IDE for the Orc programming language <sup>[49]</sup>
- Kodingen - A Cloud Development Environment <sup>[50]</sup>



## CodePress powered

- Ultimate CMS Tool <sup>[51]</sup> by Martin Kirk - [Currently only Firefox is supported]
- Extplorer <sup>[52]</sup> by Soeren Eberhardt
- ModX CMS plugin <sup>[53]</sup>
- WordPress plugin <sup>[54]</sup>
- Typo 3 plugin <sup>[55]</sup>
- CMS Drupal integration <sup>[56]</sup>
- Codepress plugin for jQuery <sup>[57]</sup>
- Ext plugin <sup>[58]</sup>

## EditArea powered

- F->IT <sup>[59]</sup>, browser-based online JavaScript FTP client
- WebDevStudio <sup>[60]</sup> - web IDE.
- Django-EditArea <sup>[61]</sup> - EditArea integration into Django.

## MDK Editor powered

- ShiftEdit Web-based IDE <sup>[41]</sup>
- Mobile Development IDE <sup>[62]</sup>

## External links

- CodeMirror vs. CodePress <sup>[63]</sup>
- EditArea vs. CodePress <sup>[64]</sup>


## References

- [1] <http://marijn.haverbeke.nl/codemirror/>
  - [2] <http://marijn.haverbeke.nl/codemirror/mixedtest.html>
  - [3] <http://github.com/marijnh/CodeMirror>
  - [4] CodeMirror supported browsers (<http://marijn.haverbeke.nl/codemirror/index.html#supported>)
  - [5] <http://codepress.org>
  - [6] <http://codepress.sourceforge.net/>
  - [7] <http://sourceforge.net/projects/codepress/>
  - [8] <http://code.google.com/p/codetextarea/> [9] <http://www.notepad.org>
  - [10] <http://www.nicolarizzo.com/gamesroom/experimental/CodeEditor.html>
  - [11] <http://www.cdolivet.com/index.php?page=editArea>
  - [12] [http://www.cdolivet.com/editarea/editarea/exemples/exemple\\_full.html](http://www.cdolivet.com/editarea/editarea/exemples/exemple_full.html)
  - [13] EditArea compatibility chart (<http://www.cdolivet.com/editarea/editarea/docs/compatibility.html>)
  - [14] <http://helene.muze.nl/>
  - [15] <http://helene.muze.nl/ariadne/loader.php/helene/demo/>
  - [16] <http://markitup.jaysalvat.com/home/>
  - [17] <http://markitup.jaysalvat.com/examples/>
  - [18] (<http://markitup.jaysalvat.com/home/>)
  - [19] <http://rob-rohan.com/projects/9ne/>
  - [20] <http://gpl.internetconnection.net/vi/>
  - [21] <http://www.ymacs.org/>
  - [22] <http://www.ymacs.org/demo/>
  - [23] <http://code.ymacs.org/hgwebdir.cgi/ymacs/>
  - [24] <http://www.mdk-photo.com/editor/>
  - [25] <https://mozillalabs.com/bespin/>
  - [26] <https://bespin.mozilla.com/>
  - [27] <http://hg.mozilla.org/labs/bespinclient/>
-

- [28] <http://marijn.haverbeke.nl/codemirror/contrib/php/>
  - [29] <http://marijn.haverbeke.nl/codemirror/sparqltest.html>
  - [30] <http://marijn.haverbeke.nl/codemirror/contrib/python/>
  - [31] <http://marijn.haverbeke.nl/codemirror/contrib/lua/index.html>
  - [32] <http://github.com/hakunin/ruby-in-codemirror>
  - [33] <http://www.cdolivet.com/index.php?page%3DeditArea>
  - [34] [http://groups.google.com/group/codemirror/browse\\_thread/thread/20cd864c74753db8/11a228d498da9261](http://groups.google.com/group/codemirror/browse_thread/thread/20cd864c74753db8/11a228d498da9261)
  - [35] <http://marijn.haverbeke.nl/codemirror/jstest.html>
  - [36] [https://wiki.mozilla.org/Labs/Bespin/UserGuide#Keyboard\\_Shortcuts](https://wiki.mozilla.org/Labs/Bespin/UserGuide#Keyboard_Shortcuts)
  - [37] [https://wiki.mozilla.org/Labs/Bespin/Tips#Want\\_custom\\_keybindings.3F](https://wiki.mozilla.org/Labs/Bespin/Tips#Want_custom_keybindings.3F)
  - [38] <http://marijn.haverbeke.nl/codemirror/js/mirrorframe.js>
  - [39] <https://wiki.mozilla.org/Labs/Bespin/UserGuide#Themes>
  - [40] [http://www.amyeditor.com/api/embed/test\\_php.html](http://www.amyeditor.com/api/embed/test_php.html)
  - [41] <http://edit.shiftcreate.com>
  - [42] <http://code.google.com/apis/ajax/playground>
  - [43] <http://rthree.wiki.sourceforge.net/>
  - [44] <http://github.com/darwin/firerainbow>
  - [45] <http://dev.freebaseapps.com/>
  - [46] <http://kml-samples.googlecode.com/svn/trunk/interactive/index.html>
  - [47] <http://eloquentjavascript.net/chapter1.html>
  - [48] <http://billmill.org/static/canvastutorial/index.html>
  - [49] <http://orc.csres.utexas.edu/tryorc.shtml>
  - [50] <http://kodingen.com>
  - [51] <http://www.mdk-photo.com/Trial/>
  - [52] <http://explorer.sourceforge.net/>
  - [53] <http://modxcms.com/forums/index.php/topic,8029.0.html>
  - [54] <http://www.naden.de/blog/wordpress-code-editor>
  - [55] <http://www.typo3-unleashed.net/nc/singleentry/date/2007/05/13/typoscript-editor-beta.html>
  - [56] <http://drupal.org/node/145663>
  - [57] <http://www.fyneworks.com/jquery/Codepress/>
  - [58] <http://extjs.com/learn/Extension:CodePress>
  - [59] <https://fit.jupiterit.com/>
  - [60] <http://www.webdevstudio.org>
  - [61] <http://www.wefoundland.com/project/Django-EditArea>
  - [62] <http://www.Siruna.com>
  - [63] [http://groups.google.com/group/codemirror/browse\\_thread/thread/2039eaa4f01e3f1f](http://groups.google.com/group/codemirror/browse_thread/thread/2039eaa4f01e3f1f)
  - [64] <http://www.peterbe.com/plog/EditArea-vs-CodePress>
-

# Douglas Crockford

---

Douglas Crockford	
	
Douglas Crockford at the "Browser Wars: Episode II Attack of the DOMs" event on 2007-02-28	
<b>Born</b>	Minnesota
<b><i>Alma mater</i></b>	San Francisco State University
<b>Known for</b>	JavaScript Object Notation
<b>Website</b> crockford.com <sup>[1]</sup>	

**Douglas Crockford** is an American computer programmer and entrepreneur, best known for his ongoing involvement in the development of the JavaScript language, and for having popularized the data format JSON (JavaScript Object Notation). He is currently a senior JavaScript architect at Yahoo!, and is also a writer and speaker on JavaScript, JSON, and related web technologies.

## Early years

He earned a degree in Radio and Television from San Francisco State University.<sup>[2]</sup>

## Career

Crockford worked on the computerization of media at Atari, Lucasfilm, and Paramount. He became something of a cult figure on videogame oriented listservs in the early 1990s after he posted his memoir "The Expurgation of Maniac Mansion" to a videogaming bulletin board; the memoir documented his efforts to censor the computer game *Maniac Mansion* to Nintendo's satisfaction so that they could release it as a cartridge, and Crockford's mounting frustrations as Nintendo's demands became more obscure and confusing.<sup>[3]</sup>

Together with Randy Farmer and Chip Morningstar, Crockford founded Electric Communities and was its CEO from 1993 to 2001. He was involved in the development of the programming language E.

Crockford was also the founder of State Software (also known as Veil Networks) and its CTO from 2001 to 2002.

During his time at State Software, Crockford popularized the JSON data format, based upon existing JavaScript language constructs, as a lightweight alternative to XML. He obtained the domain name json.org in 2002, and put up his description of the format there.<sup>[4]</sup> In July 2006 he specified the format officially, as RFC 4627.<sup>[5]</sup>

Peter Seibel interviewed Crockford for his 2009 book *Coders at Work*, discussing the evolution of JavaScript standards.<sup>[6]</sup>

## Works

- *JavaScript: The Good Parts* ISBN 978-0596517748.

## External links

- Douglas Crockford's homepage<sup>[1]</sup>
- Douglas Crockford's LinkedIn page<sup>[7]</sup>
- Douglas Crockford's Lectures on Javascript<sup>[72]</sup>

## References

- [1] <http://crockford.com/>
  - [2] Douglas Crockford speaker biography ([http://www.almaden.ibm.com/cs/new\\_paradigms/crockfor.html](http://www.almaden.ibm.com/cs/new_paradigms/crockfor.html)), New Paradigms for Using Computers conference, IBM Almaden Research Center, August 22, 1996
  - [3] The Expurgation of Maniac Mansion: A Memoir by Douglas Crockford (<http://www.crockford.com/wrrrld/maniac.html>)
  - [4] JSON: The Fat-Free Alternative to XML (<http://www.json.org/fatfree.html>), Douglas Crockford, December 6, 2006
  - [5] RFC 4627: The application/json Media Type for JavaScript Object Notation (JSON) (<http://tools.ietf.org/html/rfc4627>)
  - [6] Coders at Work: Douglas Crockford (<http://www.codersatwork.com/douglas-crockford.html>), Peter Seibel's *Coders at Work* website
  - [7] <http://www.linkedin.com/in/douglascrockford>
-

# DWR (Java)

---

<b>Developer(s)</b>	David Marginian / Joe Walker / Dojo Foundation
<b>Stable release</b>	2.0.6 / February 12, 2009
<b>Preview release</b>	3.rc2 / February 12, 2008
<b>Development status</b>	Active
<b>Written in</b>	Java and JavaScript
<b>Operating system</b>	Cross-platform
<b>Size</b>	1.08 MB (archived)
<b>Type</b>	Ajax technology
<b>License</b>	Apache 2.0 Licence
<b>Website</b>	<a href="http://directwebremoting.org/">http://directwebremoting.org/</a>

**DWR**, or **Direct Web Remoting**, is a Java open source library that helps developers write web sites that include Ajax technology. It allows code in a web browser to use Java functions running on a web server as if those functions were within the browser.

It consists of two main parts:

- Code to allow JavaScript to retrieve data from a servlet-based web server using Ajax principles.
- A JavaScript library that makes it easier for the web site developer to dynamically update the web page with the retrieved data.

DWR takes a novel approach to Ajax by dynamically generating JavaScript code based on Java classes.<sup>[1]</sup> Thus the web developer can use Java code from JavaScript as if it were local to the web browser; whereas in reality the Java code runs in the web server and has full access to web server resources. For security reasons the web developer must configure exactly which Java classes are safe to export (which is often called *web.xml* or *dwr.xml*).

This method of remoting functions from Java to JavaScript gives DWR users a feel much like conventional RPC mechanisms like RMI or SOAP, with the benefit that it runs over the web without requiring web browser plug-ins.

DWR does not consider the web browser / web server protocol to be important, and prefers to ensure that the programmer's interface is natural. The greatest challenge to this is to marry the asynchronous nature of Ajax with the synchronous nature of normal Java method calls.

In the asynchronous model, result data is only available some time after the initial call is made. DWR solves this problem by allowing the web developer to specify a function to be called when the data is returned using an extra method parameter. This extra method is called **Callback Method**.

Here is a sample Callback:

```
MJavaClassOnJs.getListProducts(selectedCategory,{
    callback:function(returnedList){

dwr.util.addOptions(myCombold,returnedList,"productId","productName")
    }
})
```

The callback is that function inside the Json object passed as an additional parameter to the remoted function.

With version 2.0 DWR supports Reverse Ajax<sup>[1]</sup> where Java code running on the server can deliberately send dedicated JavaScript to a browser.

The DWR project was started by Joe Walker in 2004.

## Bibliography

- Salkosuo, Sami (October 29, 2008), *DWR Java AJAX Applications* <sup>[2]</sup> (1st ed.), Packt Publishing, pp. 210, ISBN 1847192939
- Zammetti, Frank (January 25, 2008), *Practical DWR 2 Projects* <sup>[3]</sup> (1st ed.), Apress, pp. 540, ISBN 1590599411

## External links

- DWR project homepage <sup>[4]</sup>

## References

- [1] *Overview of DWR* (<http://directwebremoting.org/dwr/introduction/index.html>), , retrieved 2008-06-24
- [2] <http://www.packtpub.com/direct-web-remoting-java-ajax-applications/book>
- [3] <http://www.apress.com/book/view/9781590599419>
- [4] <http://getahead.org/dwr/>

# EMVC

---

<b>Developer(s)</b>	Ed Hertzog
<b>Written in</b>	JavaScript
<b>Operating system</b>	Cross-platform
<b>License</b>	MIT License
<b>Website</b>	<a href="http://www.OpenSourceMVC.com/">http://www.OpenSourceMVC.com/</a>

**e|MVC** is a JavaScript based Open Source Model-View-Controller (MVC) framework, which includes an extensive JavaScript object library and a collection of widgets in the free download. The core e|MVC object bootstraps a model/controller combination for corresponding configured view objects. Utilizing a factory pattern, models and controllers are dynamically instantiated with the help of the `dojo.query()` method utilizing a lazy load approach in a compact file. The framework relies heavily on Dojo 1.4+ toolkit, and is tightly integrated with the e|Objects JavaScript library. The simple object interface allows for the easy incorporation of other useful third-party libraries, such as jQuery, sIFR, and MooTools. e|MVC is licensed under the MIT (X11) license and is subject to license terms of numerous incorporated Open Source components.

## History

The first release of e|MVC was used commercially in 2008. e|MVC 1.0 became stable in August 2009 and was officially released to the public on July 16, 2010.

## External links

- [e|MVC homepage](#) <sup>[1]</sup>
- [e|MVC is a product of Electronic Ink, a Philadelphia based digital design firm](#) <sup>[2]</sup>

## References

[1] <http://www.opensourcemvc.com/>


[2] <http://www.electronicink.com/>

---

# Brendan Eich

---

Brendan Eich



Born	1961 (age 48–49) Pittsburgh, Pennsylvania
Alma mater	University of Illinois at Urbana–Champaign
Occupation	CTO, Mozilla Corporation
Known for	JavaScript

**Brendan Eich** (pronounced /ˈaɪk/) (born 1961<sup>[1]</sup> ) is a computer programmer and creator of the JavaScript scripting language. He is the chief technology officer at the Mozilla Corporation.

## Education

Brendan Eich received his bachelor's degree in math and computer science at Santa Clara University.<sup>[1]</sup> He received his master's degree in 1986 from the University of Illinois at Urbana-Champaign.

## Career

Eich started his career at Silicon Graphics, working for seven years on operating system and network code. He then worked for three years at MicroUnity Systems Engineering writing microkernel and DSP code, and doing the first MIPS R4000 port of gcc.

Eich is best known for his work on Netscape and Mozilla. He started work at Netscape Communications Corporation in April 1995, working on JavaScript (originally called Mocha, then called LiveScript) for the Netscape Navigator web browser. He then helped found mozilla.org in early 1998, serving as chief architect. When AOL shut down the Netscape browser unit in July 2003, Eich helped spin out the Mozilla Foundation.

In August 2005, after serving as Lead Technologist and as a member of the Board of Directors of the Mozilla Foundation, Brendan became CTO of the newly founded Mozilla Corporation.

---



## References

- [1] Steve Lohr (1996-09-09). "Part Artist, Part Hacker And Full-Time Programmer" (<http://www.nytimes.com/1996/09/09/business/part-artist-part-hacker-and-full-time-programmer.html?pagewanted=1>). New York Times. .
- Mozilla Futures: Analysis and Proposals (<http://www-archive.mozilla.org/events/dev-day-feb-2004/mozilla-futures/title.html>) (Slides presented at Mozilla Developer Day on February 27, 2004; more detailed than the recent slides cited in roadmap blog)
  - Brendan Eich and JavaScript ([http://inventors.about.com/library/inventors/bl\\_javascript.htm](http://inventors.about.com/library/inventors/bl_javascript.htm)) (about.com)

## External links

- Brendan's Roadmap Updates (<http://weblogs.mozillazine.org/roadmap/>) (Mozilla roadmap weblog)
- Brendan Eich on the Gillmor Gang July 2004 (<http://www.itconversations.com/shows/detail156.html>) and December 2005 (<http://gillmorgang.podshow.com/?p=25>)
- Brendan's Netscape Joke Homepage (<http://web.archive.org/web/20000815055653/people.netscape.com/brendan/>)
- Computerworld Interview with Brendan Eich on JavaScript (<http://www.computerworld.com.au/index.php/id;243672124;fp;4194304;fpid;1>)
- CNET Interview with Brendan Eich (<http://video.zdnet.com/CIOSessions/?p=347>)
- Tech Luminaries interview with Brendan Eich (<http://techluminaries.com/2008/12/15/episode-1-brendan-eich/>)

# JS/UIX

---

**JS/UIX** is a Unix clone by mass:werk that runs in a Web browser and is written entirely in Javascript. The terminal is implemented in DHTML, and most of the basic Unix tools are implemented. According to the website, POSIX compatibility is not a priority.

## External links

- JS/UIX <sup>[1]</sup>

## References

- [1] <http://www.masswerk.at/jsuix/>
-

# JSAN

---

**JSAN**, also known as the **JavaScript Archive Network**, is an on-line collaborative resource to develop Open Source libraries and software for JavaScript. JSAN's aim is to create the JavaScript equivalent of Perl's CPAN. For example, the module Joose is an equivalent to the CPAN module Moose. In addition there is Test.Simple and other modules that are very similar to their Perl counterparts.

## Background

JSAN was created by Casey West in 2005. Due to social reasons Casey West left the project. With the master website lost, people could not upload modules, and JSAN had several other limitations in its development, and it was considered "dead". On May 2009, the site was revived and new modules started being added.

## Design

JSAN works as a CPAN module, using a nearly identical testing system and directory structure. Documentation is written in POD which is used within JavaScript comments. The documentation is then converted to HTML and other formats with Perl POD modules. Some of the newer techniques used in JSAN are considered to be "test runs" for new features to be added to the design of the "6PAN", the CPAN upgrade intended to match the Perl 6 language.

## See also

- JSAN Wiki <sup>[1]</sup> - Wiki used in the JSAN project
- CPAN - the Perl equivalent of JSAN

## External links

- JSAN <sup>[2]</sup> - the JavaScript Archive Network

## References

[1] <http://wiki.github.com/openjsan/openjsan>

[2] <http://www.openjsan.org>

---

# JSDoc

**JSDoc** is a syntax for adding inline API documentation to JavaScript source code. This is distinct from the various tools that parse and manipulate code that follows the JSDoc **syntax**.

The JSDoc syntax is similar to the Javadoc syntax, used for documenting Java code, but is specialized to work with JavaScript's more dynamic syntax and therefore unique, as it is not completely compatible with Javadoc. However, like Javadoc, JSDoc allows the programmer to create doclets and tags which can then be translated into published output, like HTML or RTF.

## JSDoc tags

Although this list is incomplete, the following annotations are commonly used in modern JSDoc.

Tag	Description
@author	Developer's name
@constructor	Marks a function as a constructor
@deprecated	Marks a method as deprecated
@exception	Synonym for @throws
@param	Documents a method parameter; a datatype indicator can be added between curly braces
@private	Signifies that a method is private
@return	Documents a return value
@see	Documents an association to another object
@this	Specifies the type of the object to which the keyword "this" refers within a function.
@throws	Documents an exception thrown by a method
@version	Provides the version number of a library

## Example

An example of JSDoc usage.

```
/**
 * Creates an instance of Circle.
 *
 * @constructor
 * @this {Circle}
 * @param {number} r The desired radius of the circle.
 */
function Circle(r) {
  /** @private */ this.radius = r;
  /** @private */ this.circumference = 2 * Math.PI * r;
}

/**
 * Creates a new Circle from a diameter.
 *
 * @param {number} d The desired diameter of the circle.
```

```
* @return {Circle} The new Circle object.
*/
Circle.prototype.fromDiameter = function (d) {
    return new Circle(d / 2);
};

/**
 * Calculates the circumference of the Circle.
 *
 * @deprecated
 * @this {Circle}
 * @return {number} The circumference of the circle.
 */
Circle.prototype.calculateCircumference = function () {
    return 2 * Math.PI * this.radius;
};

/**
 * Returns the pre-computed circumference of the Circle.
 *
 * @this {Circle}
 * @return {number} The circumference of the circle.
 */
Circle.prototype.getCircumference = function () {
    return this.circumference;
};

/**
 * Find a String representation of the Circle.
 *
 * @override
 * @this {Circle}
 * @return {string} Human-readable representation of this Circle.
 */
Circle.prototype.toString = function () {
    return "A Circle object with radius of " + this.radius + ".";
};
```

## History

The earliest example of using a Javadoc-like syntax to document JavaScript was released in 1999 with the Netscape/Mozilla project named Rhino <sup>[1]</sup>.

## JSDoc In Use

- The JSDoc syntax has been described at length in the Apress book *Foundations of Ajax* ISBN 1-59059-582-3.
- IntelliJ and RubyMine understand JSDoc syntax.
- The Eclipse IDE <sup>[2]</sup> has extensions that understand the JSDoc syntax. Eclipse-based Aptana Studio supports ScriptDoc, and the included JavaScript files are commented in ScriptDoc.
- Mozile <sup>[3]</sup>, the Mozilla Inline Editor uses JSDoc.
- The Helma <sup>[4]</sup> application framework uses JSDoc.

## See also

- Comparison of documentation generators

## External links

- Annotating JavaScript for the Closure Compiler <sup>[5]</sup>
- Create useful relevant JavaScript documentation with JSDoc <sup>[6]</sup>
- Javadoc Tool: How to Write Doc Comments for the Javadoc Tool <sup>[7]</sup>

## Documentation generators

- JSDoc <sup>[8]</sup> - written in Perl
- JsDoc Toolkit <sup>[9]</sup> - written in JavaScript

## References

- [1] <http://lxr.mozilla.org/mozilla/source/js/rhino/examples/jsdoc.js>
  - [2] [http://www.interaktonline.com/Products/Eclipse/JSEclipse/Features/Details/Use+of+JSDoc+and+inline+parameter+comments+to+detect+parameter+type.html?id\\_ftr=639](http://www.interaktonline.com/Products/Eclipse/JSEclipse/Features/Details/Use+of+JSDoc+and+inline+parameter+comments+to+detect+parameter+type.html?id_ftr=639)
  - [3] <http://mozile.mozdev.org/0.8/doc/jsdoc/index.html>
  - [4] <http://helma.zumbrunn.net/reference/core/>
  - [5] <http://code.google.com/closure/compiler/docs/js-for-compiler.html>
  - [6] <http://blogs.techrepublic.com.com/programming-and-development/?p=451>
  - [7] <http://java.sun.com/j2se/javadoc/writingdoccomments/>
  - [8] <http://jsdoc.sourceforge.net/>
  - [9] <http://code.google.com/p/jsdoc-toolkit/>
-

# JSLint

---

<b>Original author(s)</b>	Douglas Crockford
<b>Initial release</b>	2002
<b>Stable release</b>	2010-04-06 / April 6, 2010
<b>Development status</b>	Active
<b>Written in</b>	JavaScript
<b>Operating system</b>	Cross-platform
<b>Available in</b>	English
<b>Type</b>	Static code analysis
<b>License</b>	MIT License
<b>Website</b>	<a href="http://www.jslint.com">www.jslint.com</a> <sup>[1]</sup>

**JSLint** is a static code analysis tool used in software development for checking if JavaScript source code complies with coding rules. It is developed by Douglas Crockford.

## References

- Doernhoefer, Mark (2006). "JavaScript"<sup>[2]</sup>. *SIGSOFT Softw. Eng. Notes* **31** (4): 16–24. doi:10.1145/1142958.1142972. Retrieved 2010-03-12.
- Appendix C of Crockford, Douglas (2008-05). *JavaScript: The Good Parts* (1 ed.). O'Reilly Media. ISBN 0596517742.
- Section 'Performing JavaScript Syntax Checking with JSLint', Pages 143-145 of Asleson, Ryan; Nathaniel T. Schutta (2005-10-14). *Foundations of Ajax* (1 ed.). Apress. ISBN 1590595823.

## External links

- JSLint site<sup>[3]</sup>

## References

[1] <http://www.jslint.com/>

[2] <http://portal.acm.org/citation.cfm?id=1142958.1142972&coll=Portal&dl=ACM&CFID=81676408&CFTOKEN=16664136>

[3] <http://www.jslint.com/lint.html>

---

# JSSP

---

**JSSP**, or **JavaScript Server Pages**, is an open source project that implements Javascript directly on a web server, as source code embedded in HTML and parsed without being sent to the user, much the way VBScript is in ASP.

## External links

- JSSP at Sourceforge.net <sup>[1]</sup>

## References

- [1] <http://jssp.sourceforge.net/>

# JScript

---

<b>Appeared in</b>	1996
<b>Developer</b>	Microsoft
<b>Stable release</b>	5.8 (March 2009)
<b>Typing discipline</b>	dynamic, weak, duck
<b>Major implementations</b>	Windows Script, JScript .NET
<b>Influenced by</b>	JavaScript
<b>OS</b>	Microsoft Windows
<b>Usual file extensions</b>	.js, .jse, .wsf, .wsc (.htm, .html, .asp)
<b>Website</b>	JScript <sup>[1]</sup>

**JScript** is a scripting language based on the ECMAScript standard that is used in Microsoft's Internet Explorer. JScript is implemented as a Windows Script engine. This means that it can be "plugged in" to any application that supports Windows Script, such as Internet Explorer, Active Server Pages, and Windows Script Host. It also means that any application supporting Windows Script can use multiple languages — JScript, VBScript, Perl, and others. JScript was first supported in the Internet Explorer 3.0 browser released in August 1996. Its most recent version is JScript 5.8, included in Internet Explorer 8.

Another dialect of JScript (as of 2004 - 2006) was JScript .NET, which was based on the yet-unfinished edition 4 of the ECMAScript standard, and can be compiled for the Microsoft .NET platform. JScript.NET, which added several new features to ECMAScript ed. 3, such as optional static type annotations, appears no longer to be actively developed.

## Comparison to JavaScript

As explained by JavaScript guru Douglas Crockford in his talk entitled *The JavaScript Programming Language* on YUI Theater, "[Microsoft] did not want to deal with Sun about the trademark issue, and so they called their implementation JScript. A lot of people think that JScript and JavaScript are different but similar languages. That's not the case. They are just different names for the same language, and the reason the names are different was to get around trademark issues."

## Differences from JavaScript

Unlike JavaScript, JScript supports conditional compilation. This allows a programmer to selectively execute code within block comments.

## Versions

### JScript

The original JScript is an Active Scripting engine. Like other Active Scripting languages, it is built on the COM/OLE Automation platform and provides scripting capabilities to host applications.

This is the version used when hosting JScript inside a Web page displayed by Internet Explorer, in an HTML application, in classic ASP, in Windows Script Host scripts and several other Automation environments.

---



JScript is sometimes referred to as "classic JScript" or "Active Scripting JScript" to differentiate it from newer .NET-based versions.

Some versions of JScript are available for multiple versions of Internet Explorer and Windows. For example, JScript 5.7 was introduced with Internet Explorer 7.0 and is also installed for Internet Explorer 6.0 with Windows XP Service Pack 3, while JScript 5.8 was introduced with Internet Explorer 8.0 and is also installed with Internet Explorer 6.0 on Windows Mobile 6.5.

Version	Date	Introduced with <sup>[2]</sup>	Based on <sup>[3]</sup>	Similar JavaScript version
1.0	Aug 1996	Internet Explorer 3.0	Netscape JavaScript	1.0
2.0	Jan 1997	Windows IIS 3.0	Netscape JavaScript	1.1
3.0	Oct 1997	Internet Explorer 4.0	ECMA-262 1 <sup>st</sup> edition <sup>[4]</sup>	1.3
4.0		Visual Studio 6.0 (as part of Visual InterDev)	ECMA-262 1 <sup>st</sup> edition	1.3
5.0	Mar 1999	Internet Explorer 5.0	ECMA-262 2 <sup>nd</sup> edition	1.4
5.1		Internet Explorer 5.01	ECMA-262 2 <sup>nd</sup> edition	1.4
5.5	Jul 2000	Internet Explorer 5.5 & Windows CE 4.2	ECMA-262 3 <sup>rd</sup> edition	1.5
5.6	Oct 2001	Internet Explorer 6.0 & Windows CE 5.0	ECMA-262 3 <sup>rd</sup> edition	1.5
5.7	Nov 2006	Internet Explorer 7.0	ECMA-262 3 <sup>rd</sup> edition + ECMA-327 (ES-CP) <sup>[5]</sup>	1.5
5.8	Mar 2009	Internet Explorer 8.0 & Internet Explorer Mobile 6.0	ECMA-262 3 <sup>rd</sup> edition + ECMA-327 (ES-CP) + JSON (RFC 4627) <sup>3</sup>	1.5

JScript is also available on Windows CE (included in Windows Mobile, optional in Windows Embedded CE). The Windows CE version lacks Active Debugging.

## JScript .NET

JScript .NET is a Microsoft .NET implementation of JScript, it is a CLS language and thus inherits very powerful features, but lacks many features of the original JScript language, making it inappropriate for many scripting scenarios. JScript .NET can be used for ASP.NET pages and for complete .NET applications, but the lack of support for this language in Microsoft Visual Studio place it more as an upgrade path for classic ASP using classic JScript than as a new first-class language.

Version	Platform	Date	Introduced with	Based on
7.0	Desktop CLR 1.0	2002-01-05	Microsoft .NET Framework 1.0	ECMA-262 3 <sup>rd</sup> edition <sup>[6]</sup>
7.1	Desktop CLR 1.1	2003-04-01	Microsoft .NET Framework 1.1	ECMA-262 3 <sup>rd</sup> edition <sup>[6]</sup>
8.0	Desktop CLR 2.0	2005-11-07	Microsoft .NET Framework 2.0	ECMA-262 3 <sup>rd</sup> edition <sup>[6]</sup>

JScript .NET is not supported in the .NET Compact Framework.

Note: JScript .NET versions are not related to classic JScript versions. JScript .NET is a separate product. Even though JScript .NET is not supported within the Visual Studio IDE, its versions are in sync with other .NET languages versions (C#, VB.NET, VC++) that follows their corresponding Visual Studio versions.

.NET Framework 3.0 and 3.5 are built on top of 2.0 and do not include newer releases of JScript .NET.

(Source: file version of Microsoft.JScript.dll in each framework install)

## See also

- Active Scripting
- Chakra (JScript engine)
- ECMAScript, the ECMA International language definition standard which all implementations of this language family must at a minimum follow
- JScript.NET
- JavaScript, Originally LiveScript, it was the first implementation of this language family
- Windows Script File
- Windows Script Host

## External links

- JScript documentation in the MSDN Library<sup>[7]</sup>
- JScript 5.7 Release Notes<sup>[8]</sup>
- JScript .NET documentation in the MSDN Library<sup>[9]</sup>
- JScript blog<sup>[10]</sup>
- JavaScript - JScript - ECMAScript version history<sup>[11]</sup>

## References

- [1] [http://msdn.microsoft.com/en-us/library/hbxc2t98\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/hbxc2t98(VS.85).aspx)
- [2] *Version Information (Windows Scripting - JScript)* (<http://msdn.microsoft.com/en-us/library/s4esdbwz.aspx>), Microsoft, , retrieved 2010-05-31
- [3] JScript supports various features not specified in the ECMA standard, *Microsoft JScript Features - Non-ECMA (Windows Scripting - JScript)* (<http://msdn.microsoft.com/en-us/library/4tc5a343.aspx>), Microsoft, , retrieved 2010-05-31 as does JavaScript.
- [4] Microsoft said JScript 3.0 was "the first scripting language to fully conform to the ECMA-262 standard". *Microsoft Embraces ECMA Internet Scripting Standard; Delivers Industry's First ECMA-Compliant Scripting Language, JScript 3.0, In Key Microsoft Products* (<http://www.microsoft.com/presspass/press/1997/Jun97/jecmapr.msp>), Microsoft, 1997-06-30,
- [5] JScript 5.7 includes an implementation of the ECMAScript Compact Profile (ECMA-327) which turns off features not required by the ES-CP when using the "JScript.Compact" ProgID.
- [6] JScript .NET is "being developed in conjunction with ECMAScript Edition 4". *What Is JScript .NET?* ([http://msdn.microsoft.com/en-us/library/xkx7dfw1\(VS.71\).aspx](http://msdn.microsoft.com/en-us/library/xkx7dfw1(VS.71).aspx)), Microsoft,
- [7] <http://msdn2.microsoft.com/en-us/library/hbxc2t98.aspx>
- [8] <http://download.microsoft.com/download/f/f/e/fea3abf-b55f-4924-b5a5-bde0805ad67c/Windows%20Script%20Release%20Notes.rtf>

[9] [http://msdn2.microsoft.com/en-us/library/72bd815a\(vs.71\).aspx](http://msdn2.microsoft.com/en-us/library/72bd815a(vs.71).aspx)

[10] <http://blogs.msdn.com/jscript>

[11] <http://www.webmasterworld.com/forum91/68.htm>

# JavascriptMVC

---

JavaScript <b>MVC</b>	
<b>Developer(s)</b>	Justin B. Meyer, Brian Moschel
<b>Written in</b>	JavaScript
<b>Operating system</b>	Cross-platform
<b>License</b>	MIT License
<b>Website</b>	<a href="http://www.javascriptmvc.com/">http://www.javascriptmvc.com/</a>

**JavaScriptMVC** is an open-source Rich Internet Application framework based on jQuery and OpenAjax. It extends those libraries with a Model-View-Controller architecture and tools for testing and deployment. As it does not depend on server components, it can be combined with any Web service interface and server-side language like PHP, Perl, ASP.NET, Python or Java.

## History

The first release of JavaScriptMVC was published in May 2008. JavaScriptMVC 2.0 became stable in June 2009 and is now based directly on jQuery, mainly to keep the code size small and to focus on its unique features.

## Controller

A Controller is a list of functions that get called back when the appropriate event happens. The name of the function provides a description of when the function should be called. By naming functions in the correct way, Controller recognizes them as Actions and hook them up in the correct way, for example:

```
$.Controller.extend('TodosController',{
  ".todo mouseover": function(el, ev){
    el.css("backgroundColor","red")
  },
  ".todo mouseout": function(el, ev){
    el.css("backgroundColor","")
  },
  "#create_todo_click": function(){
    this.find("ol").append("New Todo");
  }
});
```

In a controller you can also handle OpenAjax events, for example:

```
$.Controller.extend('TodosController',{
  "main.test subscribe": function(ev, publisherData){
    // TODO: do something
  },
```

```
"other.event subscribe": function(ev, publisherData){  
    // TODO: do something  
}  
});
```

## View

JavaScriptMVC uses EJS templates to render HTML data in controllers and inject them into the DOM. The syntax was inspired by ERuby and is similar to PHP or other server-side template engines.

For example file "test.ejs" ( data = [ "Hello", "World" ] ):

```
<ul>  
<% for( var i=0; i<data.length; i++ ) { %>  
  <li><%= data[i] %></li>  
<% } %>  
</ul>
```

produces the following "output":

```
<ul>  
  <li>Hello</li>  
  <li>World</li>  
</ul>
```

## Model

The Model class provides basic functionality to organize the application's data layer.

## Tests

JavaScriptMVC also comes with a comprehensive Test plug-in that supports classic unit tests for models, as well as functional test, that are required to deal with event driven architectures. Tests can be run on the command line with Rhino, using Selenium and during development with the integrated test console pop-up window.

## External links

- JavaScriptMVC homepage<sup>[1]</sup>
- Project page on Google Code<sup>[2]</sup>

## References

[1] <http://www.javascriptmvc.com/>

[2] <http://code.google.com/p/javascriptmvc/>

# JSON

---

<b>Filename extension</b>	.json
<b>Internet media type</b>	application/json
<b>Type of format</b>	Data interchange
<b>Extended from</b>	JavaScript
<b>Standard(s)</b>	RFC 4627
<b>Website</b>	<a href="http://json.org">http://json.org</a>

**JSON** (an acronym for **JavaScript Object Notation**) is a lightweight text-based open standard designed for human-readable data interchange. It is derived from the JavaScript programming language for representing simple data structures and associative arrays, called objects. Despite its relationship to JavaScript, it is language-independent, with parsers available for virtually every programming language.

The JSON format was originally specified by Douglas Crockford, and is described in RFC 4627. The official Internet media type for JSON is application/json. The JSON filename extension is .json.

The JSON format is often used for serializing and transmitting structured data over a network connection. It is primarily used to transmit data between a server and web application, serving as an alternative to XML.

## History

Douglas Crockford was the first to specify and popularize the JSON format. <sup>[1]</sup>

JSON was used at State Software, a company co-founded by Crockford, starting around 2001. The JSON.org website was launched in 2002. In December 2005, Yahoo! began offering some of its web services in JSON. <sup>[2]</sup> Google started offering JSON feeds for its GData web protocol in December 2006. <sup>[3]</sup>

Although JSON was based on a subset of the JavaScript programming language (specifically, Standard ECMA-262 3rd Edition—December 1999<sup>[4]</sup>) and is commonly used with that language, it is considered to be a language-independent data format. Code for parsing and generating JSON data is readily available for a large variety of programming languages. json.org <sup>[5]</sup> provides a comprehensive listing of existing JSON libraries, organized by language.

## Data types, syntax and example

JSON's basic types are:

- Number (integer or real)
- String (double-quoted Unicode with backslash escaping)
- Boolean (true or false)
- Array (an ordered sequence of values, comma-separated and enclosed in square brackets)
- Object (a collection of key:value pairs, comma-separated and enclosed in curly braces)
- null

The following example shows the JSON representation of an object that describes a person. The object has string fields for first name and last name, contains an object representing the person's address, and contains a list (an array) of phone number objects.

---

```
{
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021"
  },
  "phoneNumber": [
    { "type": "home", "number": "212 555-1234" },
    { "type": "fax", "number": "646 555-4567" }
  ]
}
```

A possible equivalent for the above in XML could be:

```
<Person firstName="John" lastName="Smith" age="25">
  <address streetAddress="21 2nd Street" city="New York" state="NY" postalCode="10021"/>
  <phoneNumber type="home">212 555-1234</phoneNumber>
  <phoneNumber type="fax">646 555-4567</phoneNumber>
</Person>
```

Since JSON is a subset of JavaScript it is possible (but not recommended) to parse the JSON text into an object by invoking JavaScript's `eval()` function. For example, if the above JSON data is contained within a JavaScript string variable `contact`, one could use it to create the JavaScript object `p` like so:

```
var p = eval("(" + contact + ")");
```

The `contact` variable must be wrapped in parentheses to avoid an ambiguity in JavaScript's syntax.<sup>[6]</sup>

The recommended way, however, is to use a JSON parser. Unless a client absolutely trusts the source of the text, or must parse and accept text which is not strictly JSON-compliant, one should avoid `eval()`. A correctly implemented JSON parser will accept only valid JSON, preventing potentially malicious code from running.

Modern browsers, such as Firefox 3.5 and Internet Explorer 8, include special features for parsing JSON. As native browser support is more efficient and secure than `eval()`, it is expected that native JSON support will be included in the next ECMAScript standard.<sup>[7]</sup>

## JSON schema

There are several ways to verify the structure and data types inside a JSON object, much like an XML schema.

JSON Schema<sup>[8]</sup> is a specification for a JSON-based format for defining the structure of JSON data. JSON Schema provides a contract for what JSON data is required for a given application and how it can be modified, much like what XML Schema provides for XML. JSON Schema is intended to provide validation, documentation, and interaction control of JSON data. JSON Schema is based on the concepts from XML Schema, RelaxNG, and Kwalify, but is intended to be JSON-based, so that JSON data in the form of a schema can be used to validate JSON data, the same serialization/deserialization tools can be used for the schema and data, and it can be self descriptive.

## Using JSON in Ajax

The following JavaScript code shows how the client can use an XMLHttpRequest to request an object in JSON format from the server. (The server-side programming is omitted; it has to be set up to respond to requests at url with a JSON-formatted string.)

```
var the_object = {};  
var http_request = new XMLHttpRequest();  
http_request.open( "GET", url, true );  
http_request.onreadystatechange = function () {  
    if (http_request.readyState == 4 && http_request.status == 200){  
        the_object = JSON.parse( http_request.responseText );  
    }  
};  
http_request.send(null);
```

Note that the use of XMLHttpRequest in this example is not cross-browser compatible; syntactic variations are available for Internet Explorer, Opera, Safari, and Mozilla-based browsers. The usefulness of XMLHttpRequest is limited by the same origin policy: the URL replying to the request must reside within the same DNS domain as the server that hosts the page containing the request. Alternatively, the JSONP approach incorporates the use of an encoded callback function passed between the client and server to allow the client to load JSON-encoded data from third-party domains and to notify the caller function upon completion, although this imposes some security risks and additional requirements upon the server.

Browsers can also use <iframe> elements to asynchronously request JSON data in a cross-browser fashion, or use simple <form action="url\_to\_cgi\_script" target="name\_of\_hidden\_iframe"> submissions. These approaches were prevalent prior to the advent of widespread support for XMLHttpRequest.

Dynamic <script> tags can also be used to transport JSON data. With this technique it is possible to get around the same origin policy but it is insecure. JSONRequest<sup>[9]</sup> has been proposed as a safer alternative.

## Security issues

Although JSON is intended as a data serialization format, its design as a subset of the JavaScript programming language poses several security concerns. These concerns center on the use of a JavaScript interpreter to dynamically execute JSON text as JavaScript, thus exposing a program to errant or malicious script contained therein—often a chief concern when dealing with data retrieved from the internet. While not the only way to process JSON, it is an easy and popular technique, stemming from JSON's compatibility with JavaScript's eval() function, and illustrated by the following code examples.

### JavaScript eval()

Because all JSON-formatted text is also syntactically legal JavaScript code, an easy way for a JavaScript program to parse JSON-formatted data is to use the built-in JavaScript eval() function, which was designed to evaluate JavaScript expressions. Rather than using a JSON-specific parser, the JavaScript interpreter itself is used to execute the JSON data to produce native JavaScript objects.

Unless precautions are taken to validate the data first, the eval technique is subject to security vulnerabilities if the data and the entire JavaScript environment is not within the control of a single trusted source. If the data is itself not trusted, for example, it may be subject to malicious JavaScript code injection attacks. Also, such breaches of trust may create vulnerabilities for data theft, authentication forgery, and other potential misuse of data and resources. Regular expressions can be used to validate the data prior to invoking eval. For example, the RFC that defines JSON





```

state='NY'
postalCode='10021'
/>
<phoneNumbers home='212 555-1234' fax='646 555-4567' />
</Person>

```

Not counting the irrelevant white-space, and ignoring the required XML header, this XML encoding only requires 200 characters, which may be compared with 209 characters for the equivalent JSON encoding:

```

{
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021"
  },
  "phoneNumbers": [
    { "home": "212 555-1234" },
    { "fax": "646 555-4567" }
  ]
}

```

The XML encoding can therefore be shorter than the equivalent JSON encoding.

Beyond size, both XML and JSON lack an explicit mechanism for representing large binary data types such as image data (although binary data can be serialized in either case by applying a general-purpose binary-to-text encoding scheme). JSON lacks references (something XML has via extensions like XLink and XPointer) and has no standard path notation comparable to XPath.

## YAML

Both functionally and syntactically, YAML is effectively a superset of JSON.<sup>[21]</sup> The common YAML library (Syck) also parses JSON.<sup>[22]</sup> Prior to YAML version 1.2, YAML was not quite a perfect superset of JSON, primarily because it lacked native handling of UTF-32 and required comma separators to be followed by a space.

The most distinguishing point of comparison is that YAML offers the following syntax enrichments which have no corresponding expression in JSON:

Relational:

YAML offers syntax for relational data: rather than repeating identical data later in a document, a YAML document can refer to an anchor earlier in the file/stream. Recursive structures (for example, an array containing itself) can be expressed this way. For example, a film data base might list actors (and their attributes) under a Movie's cast, and also list Movies (and their attributes) under an Actor's portfolio.

Extensible:

YAML also offers extensible data types beyond primitives (i.e., strings, floats, ints, bools) which can include class-type declarations.

Blocks:

YAML uses a block-indent syntax to allow formatting of structured data without use of additional characters (ie: braces, brackets, quotation marks, etc.). Besides giving YAML a different appearance than JSON, this block-indent device permits the encapsulation of text from other markup languages or even JSON in the other languages native literal style and without escaping of colliding sigils.

## Efficiency

JSON is primarily used for communicating data over the Internet, but has certain inherent characteristics that may limit its efficiency for this purpose. Most of the limitations are general limitations of textual data formats and also apply to XML and YAML. For example, despite typically being generated by an algorithm (by machine), parsing must ironically be accomplished on a character-by-character basis. Additionally, the standard has no provision for data compression, interning of strings, or object references. Compression can, of course, be applied to the JSON formatted data (but the decompressed output typically still requires further full parsing by the browser for recognizable keywords, tags and delimiters).

In practice performance can be comparable to that of similar binary data formats <sup>[23]</sup> and often depends more on implementation quality than on the theoretical limitations of formats.

## JSONP

**JSONP** or "JSON with padding" is a complement to the base JSON data format, a *usage pattern* that allows a page to request and more meaningfully use JSON from a server other than the primary server. JSONP is an alternative to a more recent method called Cross-Origin Resource Sharing.

Under the same origin policy, a web page served from `server1.example.com` cannot normally connect to or communicate with a server other than `server1.example.com`. An exception is HTML `<script>` tags. Taking advantage of the open policy for `<script>` tags, some pages use them to retrieve JSON from other origins.

To see how that works, let's consider a URL that, when requested, returns a JSON statement. In other words, a browser requesting the URL would receive something like:

```
{"Name": "Cheeso", "Rank": 7}
```

### The Basic Idea: Retrieving JSON via Script Tags

It's possible to specify any URL, including a URL that returns JSON, as the `src` attribute for a `<script>` tag.

Specifying a URL that returns plain JSON as the `src`-attribute for a script tag, would embed a data statement into a browser page. It's just data, and when evaluated within the browser's javascript execution context, it has no externally detectable effect.

One way to make that script *have an effect* is to use it as the argument to a function. `invoke( {"Name": "Cheeso", "Rank": 7})` actually does something, if `invoke()` is a function in Javascript.

And that is how JSONP works. With JSONP, the browser provides a JavaScript "prefix" to the server in the `src` URL for the script tag; by convention, the browser provides the prefix as a named query string argument in its request to the server, e.g.,

```
<script type="text/javascript"
      src="http://server2.example.com/getjson?jsonp=parseResponse">
</script>
```

The server then wraps its JSON response with this prefix, or "padding", before sending it to the browser. When the browser receives the wrapped response from the server it is now a script, rather than simply a data declaration. In this example, what is received is

```
parseResponse({"Name": "Cheeso", "Rank": 7})
```

...which *can* cause a change of state within the browser's execution context, because it invokes a method.

## The Padding

While the padding (prefix) is *typically* the name of a callback function that is defined within the execution context of the browser, it may also be a variable assignment, an if statement, or any other Javascript statement prefix.

## Script Tag Injection

But to make a JSONP call, you need a script tag. Therefore, for each new JSONP request, the browser must add a new `<script>` tag -- in other words, *inject* the tag -- into the HTML DOM, with the desired value for the src attribute. This element is then evaluated, the src URL is retrieved, and the response JSON is evaluated.

In that way, the use of JSONP can be said to *allow browser pages to work around the same origin policy via script tag injection*.

## Basic Security concerns

Because JSONP makes use of script tags, calls are essentially open to the world. For that reason, JSONP may be inappropriate for carrying sensitive data.<sup>[24]</sup>

Including script tags from remote sites allows the remote sites to inject **any** content into a website. If the remote sites have vulnerabilities that allow JavaScript injection, the original site can also be affected.

## Cross-site request forgery

Naïve deployments of JSONP are subject to cross-site request forgery attacks (CSRF or XSRF).<sup>[25]</sup> Because the HTML `<script>` tag does not respect the same origin policy in web browser implementations, a malicious page can request and obtain JSON data belonging to another site. This will allow the JSON-encoded data to be evaluated in the context of the malicious page, possibly divulging passwords or other sensitive data if the user is currently logged into the other site.

This is only a problem if the JSON-encoded data contains sensitive information that should not be disclosed to a third party, and the server depends on the browser's Same Origin Policy to block the delivery of the data in the case of an improper request. There is no problem if the server determines the propriety of the request itself, only putting the data on the wire if the request is proper. Cookies are not by themselves adequate for determining if a request was authorized. Exclusive use of cookies is subject to cross-site request forgery.

## History

The original proposal for JSONP appears to have been made by Bob Ippolito in 2005<sup>[26]</sup> and is now used by many Web 2.0 applications such as Dojo Toolkit Applications, Google Web Toolkit Applications<sup>[27]</sup> and Web Services. Further extensions of this protocol have been proposed by considering additional input arguments as, for example, is the case of JSONPP<sup>[28]</sup> supported by S3DB web services.

## Object references

The JSON standard does not support object references, but the Dojo Toolkit illustrates how conventions can be adopted to support such references using standard JSON. Specifically, the `dojox.json.ref`<sup>[29]</sup> module provides support for several forms of referencing including circular, multiple, inter-message, and lazy referencing.<sup>[30]</sup>

---

## See also

- GeoJSON
- JSON-RPC
- SOAPjr — a hybrid of SOAP and JR (JSON-RPC)
- JsonML
- S-expressions
- YAML ("YAML Ain't a Markup Language")
- Comparison of data serialization formats

## External links

- Format home page <sup>[31]</sup>
- JSON-Introduction By Microsoft <sup>[32]</sup>
- JSON Schema Proposal <sup>[33]</sup>
- Limitations of JSON <sup>[34]</sup>
- Mastering JSON <sup>[35]</sup>
- Online JSON Viewer <sup>[36]</sup>
- Relationship between JSON and YAML <sup>[37]</sup>
- RFC 4627, current formal JSON specification.
- wxJSON - The wxWidgets implementation of JSON <sup>[38]</sup>

## References

- [1] Video: Douglas Crockford — The JSON Saga (<http://developer.yahoo.com/yui/theater/video.php?v=crockford-json>), on Yahoo! Developer Network. In the video Crockford states: "I do not claim to have invented JSON ... What I did was I found it, I named it, I described how it was useful. [and a few sentences later ...] So the idea's been around there for awhile. What I did was I gave it a specification, and a little website."
- [2] Yahoo!. "Using JSON with Yahoo! Web services" (<http://developer.yahoo.com/common/json.html>). . Retrieved July 3, 2009.
- [3] Google. "Using JSON with Google Data APIs" (<http://code.google.com/apis/gdata/json.html>). . Retrieved July 3, 2009.
- [4] Crockford, Douglas (May 28, 2009). "Introducing JSON" (<http://json.org>). json.org. . Retrieved July 3, 2009.
- [5] <http://json.org/>
- [6] Crockford, Douglas (July 9, 2008). "JSON in JavaScript" (<http://www.json.org/js.html>). json.org. . Retrieved September 8, 2008.
- [7] <http://www.json.org/js.html>
- [8] <http://json-schema.org>
- [9] <http://json.org/JSONRequest.html>
- [10] Douglas Crockford (July 2006). "IANA Considerations" (<http://tools.ietf.org/html/rfc4627&#035;section-6>). *The application/json Media Type for JavaScript Object Notation (JSON)* (<http://tools.ietf.org/html/rfc4627>). IETF. sec. 6. RFC 4627. . Retrieved October 21, 2009.
- [11] Crockford, Douglas (December 6, 2006). "JSON: The Fat-Free Alternative to XML" (<http://www.json.org/fatfree.html>). . Retrieved July 3, 2009.
- [12] "Using Native JSON" ([https://developer.mozilla.org/en/Using\\_JSON\\_in\\_Firefox](https://developer.mozilla.org/en/Using_JSON_in_Firefox)). June 30, 2009. . Retrieved July 3, 2009.
- [13] Barsan, Corneliu (September 10, 2008). "Native JSON in IE8" (<http://blogs.msdn.com/ie/archive/2008/09/10/native-json-in-ie8.aspx>). . Retrieved July 3, 2009.
- [14] "Web specifications supported in Opera Presto 2.5" (<http://www.opera.com/docs/specs/presto25/#ecmascript>). March 10, 2010. . Retrieved March 29, 2010.
- [15] Hunt, Oliver (June 22, 2009). "Implement ES 3.1 JSON object" ([https://bugs.webkit.org/show\\_bug.cgi?id=20031](https://bugs.webkit.org/show_bug.cgi?id=20031)). . Retrieved July 3, 2009.
- [16] "YUI 2: JSON utility" (<http://developer.yahoo.com/yui/json/#native>). September 1, 2009. . Retrieved October 22, 2009.
- [17] "Learn JSON" (<http://www.prototypejs.org/learn/json>). April 7, 2010. . Retrieved April 7, 2010.
- [18] "Ticket #4429" (<http://dev.jquery.com/ticket/4429>). May 22, 2009. . Retrieved July 3, 2009.
- [19] "Ticket #8111" (<http://trac.dojotoolkit.org/ticket/8111>). June 15, 2009. . Retrieved July 3, 2009.
- [20] "Ticket 419" (<https://mootools.lighthouseapp.com/projects/2706/tickets/419-use-the-native-json-object-if-available>). October 11, 2008. . Retrieved July 3, 2009.

- [21] Ben-Kiki, Oren; Evans, Clark; döt Net, Ingy (May 13, 2008). "YAML Ain't Markup Language (YAML) Version 1.2" (<http://yaml.org/spec/1.2/#id2560236>). . Retrieved July 3, 2009. "YAML can therefore be viewed as a natural superset of JSON, offering improved human readability and a more complete information model. This is also the case in practice; every JSON file is also a valid YAML file. This makes it easy to migrate from JSON to YAML if/when the additional features are required."
  - [22] RedHanded (April 7, 2005). "YAML is JSON" (<http://redhanded.hobix.com/inspect/yamllsJson.html>). . Retrieved July 3, 2009.
  - [23] <http://code.google.com/p/thrift-protobuf-compare/wiki/Benchmarking>
  - [24] RIAspot. "JSON P for Cross Site XHR" (<http://www.riaspot.com/blogs/entry/JSONP-for-Cross-Site-XHR>). .
  - [25] Grossman, Jeremiah (January 27, 2006). "Advanced Web Attack Techniques using Gmail" (<http://jeremiahgrossman.blogspot.com/2006/01/advanced-web-attack-techniques-using.html>). . Retrieved July 3, 2009.
  - [26] "Remote JSON - JSONP" (<http://bob.pythonmac.org/archives/2005/12/05/remote-json-jsonp/>). *from \_\_future\_\_ import \**. Bob.pythonmac.org. December 5, 2005. . Retrieved September 8, 2008.
  - [27] "GWT Tutorial: How to Read Web Services Client-Side with JSONP" (<http://www.gwtapps.com/?p=42>). *Google Web Toolkit Applications*. February 6, 2008. . Retrieved July 3, 2009.
  - [28] Almeida, Jonas (June 11, 2008). "JSON, JSONP, JSONPP?" (<http://sites.google.com/a/s3db.org/s3db/documentation/mis/json-jsonp-jsonpp>). S3DB. . Retrieved April 26, 2009.
  - [29] <http://api.dojotoolkit.org/jsdoc/1.2/dojox.json.ref>
  - [30] Zyp, Kris (June 17, 2008). "JSON referencing in Dojo" (<http://www.sitepen.com/blog/2008/06/17/json-referencing-in-dojo>). . Retrieved July 3, 2009.
  - [31] <http://www.json.org/>
  - [32] <http://msdn.microsoft.com/en-us/library/bb299886.aspx>
  - [33] <http://groups.google.com/group/json-schema/web/json-schema-proposal-working-draft>
  - [34] [http://blogs.sun.com/bblfish/entry/the\\_limitations\\_of\\_json](http://blogs.sun.com/bblfish/entry/the_limitations_of_json)
  - [35] [http://www.hunlock.com/blogs/Mastering\\_JSON\\_\(JavaScript\\_Object\\_Notation\\_\)](http://www.hunlock.com/blogs/Mastering_JSON_(JavaScript_Object_Notation))
  - [36] <http://jsonviewer.stack.hu/>
  - [37] <http://redhanded.hobix.com/inspect/yamllsJson.html>
  - [38] <http://wxcode.sourceforge.net/docs/wxjson/>
-

# JsonML

---

<b>Internet media type</b>	application/jsonml+json (unofficial)
<b>Type of format</b>	Markup language and Web template system
<b>Extended from</b>	XML, JSON and JavaScript

The **JSON Markup Language (JsonML)** is a lightweight markup language used to map between XML and JSON (JavaScript Object Notation). It converts an XML document or fragment into a JSON data structure for ease of use within JavaScript environments such as a web browser, allowing manipulation of XML data without the overhead of an XML parser.

JsonML has greatest applicability in Ajax web applications. It is used to transport XHTML down to the client where it can be deterministically reconstructed into DOM elements. Progressive enhancement strategy can be employed during construction to bind dynamic behaviors to otherwise static elements.<sup>[1]</sup>

JsonML can also be used as the underlying structure for creating intricate client-side templates: JBST (JsonML+Browser-Side Templates)<sup>[2]</sup>. Syntactically JBST looks like JSP or ASP.NET UserControls. See [3] for an interactive example.

## Syntax

There are two forms of JsonML<sup>[4]</sup>:

### Array Form

The Array Form is the original form, and allows any XML document to be represented uniquely as a JSON string

### Object Form

The Object Form assumes that there are no XML tags named *tagName* or *childNodes*.

Conversion from XML to JsonML is **fully reversible**. XML Namespaces are handled by prepending the element name with the namespace prefix (e.g. <myns:myElement/> becomes ["myns:myElement"]).

## JsonML Array Form

JsonML Array Form uses:

- JSON Arrays to represent XML elements
- JSON Objects to represent attributes, and
- JSON Strings to represent Text nodes.

JsonML (Array Form)	Original XML
---------------------	--------------

<pre>&lt;font size="8.10"&gt; ["person",  {   "created": "2006-11-11T19:23",   "modified": "2006-12-31T23:59",   ["firstName", "Robert"],   ["lastName", "Smith"],   ["address", {"type": "home"},    ["street", "12345 Sixth Ave"],    ["city", "Anytown"],    ["state", "CA"],    ["postalCode", "98765-4321"]   ]  } ] &lt;/font&gt;</pre>	<pre>&lt;font size="8.10"&gt; &lt;!-- XML representation of a person record --&gt; &lt;person created="2006-11-11T19:23" modified="2006-12-31T23:59"&gt;   &lt;firstName&gt;Robert&lt;/firstName&gt;   &lt;lastName&gt;Smith&lt;/lastName&gt;   &lt;address type="home"&gt;     &lt;street&gt;12345 Sixth Ave&lt;/street&gt;     &lt;city&gt;Anytown&lt;/city&gt;     &lt;state&gt;CA&lt;/state&gt;     &lt;postalCode&gt;98765-4321&lt;/postalCode&gt;   &lt;/address&gt; &lt;/person&gt; &lt;/font&gt;</pre>
---	--

JsonML (Object Form)

JsonML Object Form uses:

- JSON Objects to represent XML elements and their attributes, and
- JSON Arrays to represent child node lists

The JsonML Object Form representation of the person record is:

JsonML (Object Form)	Original XML
<pre>1 "&gt; rson", 06-11-11T19:23", 006-12-31T23:59", [ "firstName", "childNodes" : ["Robert"]}, "lastName", "childNodes" : ["Smith"]}, "address", "type": "home", "childNodes" : [ me": "street", "childNodes" : ["12345 Sixth Ave"]}, me": "city", "childNodes" : ["Anytown"]}, me": "state", "childNodes" : ["CA"]}, me": "postalCode", "childNodes" : ["98765-4321"]},</pre>	<pre>&lt;font size="6.11"&gt; &lt;!-- XML representation of a person record --&gt; &lt;person created="2006-11-11T19:23" modified="200   &lt;firstName&gt;Robert&lt;/firstName&gt;   &lt;lastName&gt;Smith&lt;/lastName&gt;   &lt;address type="home"&gt;     &lt;street&gt;12345 Sixth Ave&lt;/street&gt;     &lt;city&gt;Anytown&lt;/city&gt;     &lt;state&gt;CA&lt;/state&gt;     &lt;postalCode&gt;98765-4321&lt;/postalCode&gt;   &lt;/address&gt; &lt;/person&gt; &lt;/font&gt;</pre>

A “regular” JSON transformation produces a more compact representation, but loses some of the document structural information:

```
{
  "person": {
    "address": {
      "city": "Anytown",
      "postalCode": "98765-4321",
      "state": "CA",
      "street": "12345 Sixth Ave",
      "type": "home"
    },
    "created": "2006-11-11T19:23",
    "firstName": "Robert",
    "lastName": "Smith",
  }
}
```

```
"modified" : "2006-12-31T23:59"
}}
```

## Comparison To Similar Technologies

### XML/XSLT

XSLT and XML can produce client-side templating as well, and both allow caching of the template separate from the data. For many, however, the syntax of JBST<sup>[2]</sup> is much easier to manage, due to its familiarity. JBST uses JavaScript natively in the template rather than requiring a mixing of a different type of control language.

### innerHTML

While seemingly used to perform similar tasks, JsonML and innerHTML are quite different.

innerHTML requires you to have all the markup exactly as you want it ready to go, meaning that either the server is rendering the markup, or you are performing expensive string concatenations in JavaScript.

JsonML uses client-side templating through JBST, which means that HTML is converted into a JavaScript template at build time. At runtime, the data is supplied and DOM elements are the result. The resulting DOM elements can be inserted or replace an existing element—something innerHTML cannot easily do without creating excess DOM elements. Rebinding only requires requesting additional data, which is smaller than fully-expanded markup. As a result, large performance gains are often made, since the markup is requested (or cached) separately from the data.

## HTML Message Pattern / Browser-Side Templating

For simplicity, innerHTML has been the preferred method for the HTML-Message pattern<sup>[5]</sup> style of Ajax. However, tools like JsonFx<sup>[6]</sup> aim to simplify JsonML and JBST implementation while still providing a full browser-side templating Ajax pattern.<sup>[7]</sup>

### External links

- [JsonML.org](http://jsonml.org)<sup>[8]</sup>
- [IBM developerWorks Article](#)<sup>[9]</sup>
- [Java JSONML implementation](#)<sup>[10]</sup> - written by Douglas Crockford
- [JsonFx.NET](#)<sup>[11]</sup> - C#/.NET JBST Framework

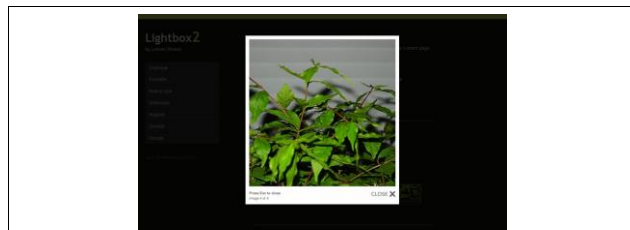
### References

- [1] <http://jsonml.org/Ajax-UI/Binding/>
- [2] <http://jsonml.org/bst>
- [3] <http://jsonml.org/BST/Example>
- [4] <http://tech.groups.yahoo.com/group/json/message/1115>
- [5] [http://ajaxpatterns.org/HTML\\_Message](http://ajaxpatterns.org/HTML_Message)
- [6] <http://jsonfx.net>
- [7] [http://ajaxpatterns.org/Browser-Side\\_Templating](http://ajaxpatterns.org/Browser-Side_Templating)
- [8] <http://jsonml.org>
- [9] <http://www.ibm.com/developerworks/library/x-jsonml/>
- [10] <http://json.org/javadoc/org/json/JSONML.html>
- [11] <http://jsonfx.googlecode.com>



# Lightbox (JavaScript)

---



A webpage showing a Lightbox 2 window.

<b>Developer(s)</b>	Lokesh Dhakar
<b>Stable release</b>	2.04 / March 9, 2008
<b>Development status</b>	Stable
<b>Operating system</b>	Cross-platform
<b>Available in</b>	English
<b>License</b>	Creative Commons Attribution 2.5 License
<b>Website</b>	[1]

**Lightbox**, and the newer **Lightbox 2**, is a JavaScript application used to display large images using modal dialogs. The script has gained widespread popularity due to its simple yet elegant style and easy implementation. While it was initially developed from scratch, Lightbox has since been modified to use a number of JavaScript libraries (such as the Prototype Javascript Framework<sup>[2]</sup> and script.aculo.us<sup>[3]</sup> for its animations and positioning), in order to reduce the size of the code.<sup>[4]</sup> The release of Lightbox encouraged other developers to work on similar projects, resulting in products such as the later Thickbox and lighter Slimbox.<sup>[4]</sup>

## How it works

On a Lightbox-enabled page, a user can click an image to have it magnified in a Lightbox window, which resizes itself according to the size of the image using a gliding animation. Lightbox determines which images will be shown in the modal window through the XHTML "rel" attribute, which is used on an <a> element wrapped around the <img> element. Lightbox also provides a way to attach captions to images and to run a slide show, which can be navigated using the arrow keys.

## Functionality

*Lightbox* permits users to view larger versions of images without having to leave the current page,<sup>[5]</sup> and is also able to display simple slideshows. The use of the dark background, which dims the page over which the image has been overlaid, also serves to highlight the image being viewed.

While *Lightbox* is dependent upon a browser's compatibility with Prototype to function,<sup>[3]</sup> Lightbox is triggered through a standard link tag. Thus browsers that do not support JavaScript simply load the image as a separate file, losing the Lightbox effect but still retaining the ability to display the full-sized image.<sup>[5]</sup> Even so, some compatibility problems have been identified with versions of Lightbox, in particular when displaying larger images on Firefox or Opera.<sup>[6]</sup>

Many users have also noted that there is a lack of compatibility across the Internet Explorer range of browsers. However, many times the compatibility issues can be attributed to users not reading the documentation on how to make lightbox function correctly in these browsers.

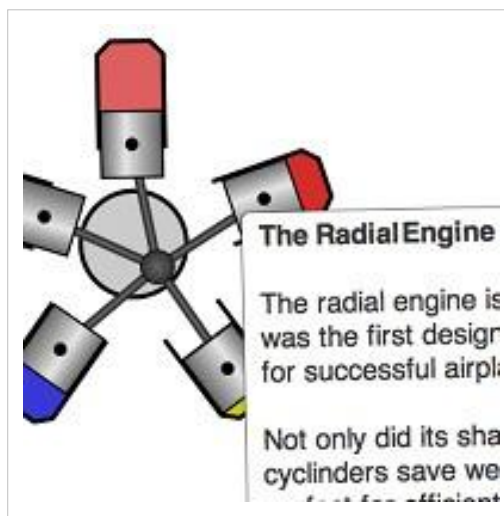
## External links

- Official Lightbox website <sup>[7]</sup>
- The Lightbox Clones Matrix <sup>[8]</sup> provides a listing of Lightbox alternatives (clones) in a JavaScript-enhanced tabular format

## References

- [1] <http://www.lokeshdhakar.com/projects/lightbox2/>
  - [2] Herrington, Jack D. "Ajax and XML: Ajax for lightboxes" (<http://www.ibm.com/developerworks/library/x-ajaxxml6/>). *IBM DeveloperWorks*. . Retrieved 2008-05-21.
  - [3] Schmitt, Christopher (2006). *CSS Cookbook*. O'Reilly. pp. p. 204. ISBN 0596527411.
  - [4] Resig, John (2006). *Pro JavaScript Techniques*. Apress. ISBN 1590597273.
  - [5] Zervaas, Quentin (2007). *Practical Web 2.0 Applications with PHP*. Springer. pp. p. 423. ISBN 1590599063.
  - [6] Campbell, Debbie (February 15, 2007). "Thickbox - For Image Display and Slideshows" (<http://www.webpronews.com/blogtalk/2007/02/15/thickbox-for-image-display-and-slideshows>). *WebProNews*. . Retrieved 2008-05-21.
  - [7] <http://lokeshdhakar.com/projects/lightbox2/>
  - [8] <http://planetozh.com/projects/lightbox-clones/>
-

# Lively Kernel



Lively Kernel example running in Safari

<b>Developer(s)</b>	Sun Microsystems Laboratories, Hasso Plattner Institut
<b>Stable release</b>	0.8.5 / April 6, 2009
<b>Operating system</b>	Most popular web browsers yet currently best experienced using: Safari 4 (Mac), Google Chrome (PC). Internet Explorer (PC) will require Google Chrome Frame.
<b>Type</b>	Web development
<b>License</b>	MIT
<b>Website</b>	<a href="http://lively-kernel.org/">http://lively-kernel.org/</a>

The **Lively Kernel** is a web programming environment originally developed at Sun Microsystems Laboratories. The Lively Kernel supports desktop-style applications with rich graphics and direct manipulation capabilities, but without the installation or upgrade hassles that conventional desktop applications have<sup>[1] [2]</sup>.

## Overview

The Lively Kernel is a graphical composition and integrated programming environment written entirely in the JavaScript programming language using standard browser graphics (W3C Canvas or SVG). It is thus accessible to any browser as a web page, and it begins to operate as soon as the web page is loaded. It is able to edit its own graphics, edit its own code and, through its built-in WebDAV support, it can save its results or even clone itself onto new web pages. In addition to its application development capabilities, Lively can also function as its own integrated development environment (IDE), making the whole system self-sufficient with no tools other than a browser.

## Shapes, Widgets, Windows and an IDE all on a Web Page

The Lively Kernel uses a Morphic graphics model to add behavior to a scene graph built from browser graphics. Simple graphics are thus assembled into such standard widgets as sliders, scroll bars, text views, lists and clipping frames. A simple window system built from these widgets offers object inspectors, file browsers and code browsers. Even the rudimentary demo pages thus have the ability to edit and test new code in a simple code browser while the system is running.

Lively has been used to build simple web sites, including its own tutorial, and also a client-side Wiki system that stores its pages in an SVN repository. Lively content varies from relatively static pages of text to fully dynamic models that look and behave like flash simulations. The Lively Kernel achieves complex dynamic behavior without any specific animation support (it does not use the animation features of SVG), but by simple scheduling of multiple green-thread processes in JavaScript.

## Lively Text and Transformations

The Lively Kernel includes its own multifont text editor written in JavaScript. It includes support for centering, justification and similar rudimentary text composition capabilities. Working in Lively thus has much the same feel as working in a web page design program, except that the on-the-fly text layout is not being done in an offline composition program, but it is in fact the built-in dynamic behavior of text in the Lively Kernel.

The liveliness of Lively graphics becomes even more apparent when you manipulate the scale and rotation handles for objects and text. The entire code browser can be used when tilted 20 degrees on its side. Because the text editor is made up entirely of lively graphics, it works perfectly well when rotated or scaled, just as do the scroll bars, clipping frames, and the rest of the entire lively user interface.

## Cross-Browser Compatibility

The Lively Kernel depends on browser support for JavaScript and SVG or Canvas graphics, all now part of the W3C standards. At the time of writing, this means that it runs in Safari, Firefox, Chrome and Opera browsers.

While this requirement might seem less compatible than HTML, Lively is actually more compatible across the browsers on which it runs than is HTML. This is because there is greater uniformity among JavaScript, SVG and Canvas implementations than there is from one HTML implementation to another. Except for one small initial file, the Lively Kernel code base is entirely free of tests for which client browser is being used.

## The Lively IDE

Lively includes an integrated development environment of considerable power, designed to work via WebDAV with a local set of a dozen or so source JavaScript files synchronizable with an SVN repository. If the user opens a SystemBrowser, all the JavaScript source files are listed in the file pane of the browser. If the user clicks on one of these files, it will be read, parsed (by an OMeta JavaScript parser) and displayed similar to a Smalltalk browser with functions or class definitions listed and, for each class, all the method names are shown. The user can click on a method name, edit its code in the bottom pane, and then save the new definition. The new definition will be checked for syntax and, if correct, it will be stored back in the .JS file. Moreover, if in "eval mode" (the usual case), the method will be redefined in the system that is running. This allows non-critical changes to be made without any need to restart Lively or any of the applications being developed. When all the source files are loaded, a rapid scan will find every reference to a selected text, and present all the code bodies that match in a separate sub-browser.

The Lively IDE includes object inspectors and morph style panels for controlling fills, borders, and text styles. There are also facilities for debugging at error points and profiling for performance tuning, but these have not been pushed, owing to the improving quality of such run-time support in all the major browsers.

## The Lively Wiki

The IDE operates on its source code files in an SVN repository to manage evolution of the Lively code base. The same approach has been used to empower users with control over active web content in the form of a client-side Wiki. Because Lively includes facilities to edit the content in its worlds (scene graphs and other content), and to store that content as web pages, a WebDAV connection to an SVN repository allows Lively to store new versions of its page content while it is being incrementally developed. As each new version is saved, it is viewable from the rest of the Internet and prior versions can be retrieved in cases of error or change of mind. It is notable that this Wiki style of evolving web content extends to all the textual, graphical, and scripting content in Lively, as well as to any new forms that may be defined, yet without any need for a Wiki server of any kind<sup>[3]</sup>. All that is required is a normal server, and a normal SVN-style repository. Increasing amounts of the Lively Project content is now maintained in this manner.

## Lively for Qt

Lively for Qt is a new implementation of the Lively Kernel in which the "kernel" parts of Lively have been replaced by functionality offered by the Qt framework<sup>[4]</sup>. Lively for Qt inherits most of the basic functionality (such as the implementation of widgets, layout management, core event handling and core JavaScript support) from Qt. Lively for Qt retains the exceptionally interactive nature (the "liveliness") of the Lively Kernel, e.g., by allowing the user interface and application source code to be edited on the fly. However, in Lively for Qt the development work is performed using the familiar, well-documented APIs of the Qt platform.

## Example code

```
// ClockMorph: A simple analog clock

Morph.subclass("ClockMorph", {

  defaultBorderWidth: 2,
  type: "ClockMorph",

  // Constructor
  initialize: function($super, position, radius) {

    $super(position.asRectangle().expandBy(radius), "ellipse");
    this.openForDragAndDrop = false; // Do not handle drag-and-drop
    requests
      this.makeNewFace(); // Construct the clock face
      return this;
  },

  // Construct a new clock face
  makeNewFace: function() {

    var bnds = this.shape.bounds();
    var radius = bnds.width/2;
    var labels = [];
    var fontSize = Math.max(Math.floor(0.04 * (bnds.width +
bnds.height)),2);
```

```

    var labelSize = fontSize; // room to center with default inset

    // Add Roman numerals to the clock
    for (var i = 0; i < 12; i++) {
        var labelPosition =
bnds.center().addPt(Point.polar(radius*0.85,
                                ((i-3)/12)*Math.PI*2)).addXY(labelSize,
0);

        var label = new
TextMorph(pt(0,0).extent(pt(labelSize*3,labelSize)),

['XII','I','II','III','IV','V','VI','VII','VIII','IX','X','XI'][i]);
        label.setWrapStyle(WrapStyle.SHRINK);
        label.setFontSize(fontSize);
        label.setInset(pt(0,0));
        label.setBorderWidth(0);
        label.setFill(null);

label.align(label.bounds().center(),labelPosition.addXY(-2,1));
        this.addMorph(label);
    }

    // Add clock hands
    this.addMorph(this.hourHand =
Morph.makeLine([pt(0,0),pt(0,-radius*0.5)],4,Color.blue));
    this.addMorph(this.minuteHand =
Morph.makeLine([pt(0,0),pt(0,-radius*0.7)],3,Color.blue));
    this.addMorph(this.secondHand =
Morph.makeLine([pt(0,0),pt(0,-radius*0.75)],2,Color.red));
    this.setHands();
    this.changed();
},

// Set clock hand angles based on current time
setHands: function() {

    var now = new Date();
    var second = now.getSeconds();
    var minute = now.getMinutes() + second/60; var
hour = now.getHours() + minute/60;
    this.hourHand.setRotation(hour/12*2*Math.PI);
    this.minuteHand.setRotation(minute/60*2*Math.PI);
    this.secondHand.setRotation(second/60*2*Math.PI);
},

// Will be called when the ClockMorph is placed in a world
startSteppingScripts: function() {

```

```

        this.startStepping(1000, "setHands"); // once per second
    }
});

```

## External links

The Lively Kernel is currently best experienced using Safari 4 (Mac) or Google Chrome (PC). Google Chrome Frame will be required to run the Lively Kernel in Internet Explorer (PC). Testing in all popular browsers is targeted for the next release.

- Lively Kernel Project Page <sup>[5]</sup>
- Lively Kernel Technical Overview <sup>[6]</sup>
- Lively Kernel Abstract <sup>[7]</sup>
- Lively Kernel Mailing List <sup>[8]</sup>
- Lively Kernel Wiki <sup>[9]</sup>
- Lively Kernel Wiki User Pages <sup>[10]</sup>
- Lively Kernel Example <sup>[11]</sup>
- Lively for Qt <sup>[12]</sup>

## References

- [1] The Lively Kernel A Self-supporting System on a Web Page (<http://www.springerlink.com/content/c48378251j2vj07/>)
- [2] The Lively Kernel Application Framework ([http://www.svgopen.org/2008/papers/93-The\\_Lively\\_Kernel\\_Web\\_Application\\_Framework/](http://www.svgopen.org/2008/papers/93-The_Lively_Kernel_Web_Application_Framework/))
- [3] Lively Wiki A Development Environment for Creating and Sharing Active Web Content ([http://www.hpi.uni-potsdam.de/hirschfeld/publications/media/KrahnIngallsHirschfeldLinckePalacz\\_2009\\_LivelyWikiADevelopmentEnvironmentForCreatingAndSharingActiveWebContent\\_AcmDL.pdf](http://www.hpi.uni-potsdam.de/hirschfeld/publications/media/KrahnIngallsHirschfeldLinckePalacz_2009_LivelyWikiADevelopmentEnvironmentForCreatingAndSharingActiveWebContent_AcmDL.pdf)), Krahn Ingalls Hirschfeld Lincke Palacz, WikiSym '09, October 25–27, 2009
- [4] Lively for Qt: A Platform for Mobile Web Applications (<http://lively.cs.tut.fi/qt/documents/LivelyForQt-Mobility2009.pdf>), Mikkonen Taivalsaari Terho, Mobility 2009, Sep 2-4
- [5] <http://lively-kernel.org/>
- [6] <http://lively-kernel.org/development/media/LivelyKernel-TechnicalOverview.pdf>
- [7] [http://www.svgopen.org/2008/papers/93-The\\_Lively\\_Kernel\\_Web\\_Application\\_Framework/](http://www.svgopen.org/2008/papers/93-The_Lively_Kernel_Web_Application_Framework/)
- [8] <http://lively-kernel.org/list/index.html>
- [9] <http://lively-kernel.org/repository/lively-wiki/index.xhtml>
- [10] <http://lively-kernel.org/repository/lively-wiki/users/>
- [11] <http://lively-kernel.org/repository/lively-wiki/Engine.xhtml>
- [12] <http://lively.cs.tut.fi/qt/>

# Log4javascript

---

**log4javascript** is a JavaScript logging framework based on the Java logging framework `log4j`. The latest version is 1.4.1, released in March 2009.

## See also

Log4js

## External links

- [log4javascript Homepage](http://log4javascript.org/)<sup>[1]</sup>
- [log4javascript Homepage on Tigris](http://log4javascript.tigris.org/)<sup>[2]</sup>

## References

[1] <http://log4javascript.org/>

[2] <http://log4javascript.tigris.org/>

# Medireview

---

**Medireview** was an erroneous correction of the word *medieval* in Yahoo! Mail in 2001.<sup>[1]</sup> <sup>[2]</sup> <sup>[3]</sup>

In 2001, Yahoo! introduced an email filter which automatically replaced Javascript-related strings with alternate versions, to prevent the possibility of JavaScript viruses in HTML email. The filter would hyphenate the terms "Javascript", "Jscript", "Vbscript" and "Livescript", and replaced "eval", "mocha" and "expression" with the similar but not quite synonymous terms "review", "espresso" and "statement". Assumptions were involved in the writing of the filters: no attempts were made to limit these string replacements to script sections and attributes, or to respect word boundaries, in case this would leave some loophole open.

One unintended effect of this was that the word *medieval* became *medireview*, in the body of Yahoo! email messages. Not all email recipients recognised this as an error or may have even thought it was intentional. As of 2009 there are still web pages seen which use *medireview* instead of *medieval*.<sup>[4]</sup> <sup>[5]</sup> <sup>[6]</sup> Among other effects were the replacement of *evaluation* with *reviewuation*, *expressionist* with *statementist*, *prevalent* with *prreviewent* and the replacement of the French word *cheval* (horse) with *chreview*, rendering them into near-gibberish.

## See also

- Scunthorpe problem, a similar problem in automated filters designed to block obscenity

## References

[1] BBC story (<http://news.bbc.co.uk/2/low/science/nature/2138014.stm>)

[2] *Need To Know* story ([http://www.ntk.net/2002/07/12/#HARD\\_NEWS](http://www.ntk.net/2002/07/12/#HARD_NEWS))

[3] *New Scientist* story (<http://www.newscientist.com/news/news.jsp?id=ns99992546>)

[4] Example 1 (<http://www.hinduonnet.com/mag/2002/05/26/stories/2002052600100200.htm>)

[5] Example 2 ([http://www.irfi.org/articles/articles\\_1\\_50/women\\_in\\_islam.htm](http://www.irfi.org/articles/articles_1_50/women_in_islam.htm))

[6] Example 3 ([http://www.st-francis-lutheran.org/nyt\\_11Mar01.html](http://www.st-francis-lutheran.org/nyt_11Mar01.html))



# Minification (programming)

---

**Minification** (very often just **minify**, and sometimes also **minimisation** or **minimization**), in computer programming languages and especially JavaScript, is the process of removing all unnecessary characters from source code, without changing its functionality. These unnecessary characters usually include white space characters, new line characters, comments and sometimes block delimiters; which are used to add readability to the code, but are not required for it to execute.

Minified source code is especially useful for interpreted languages deployed and transmitted on the Internet (such as JavaScript), because it reduces the amount of data that needs to be transferred. Minified source code may also be used as a kind of obfuscation. In Perl culture, aiming at extremely minified source code is the purpose of Perl golf game.

Minified source code is also very useful for HTML code. All white space in HTML is reduced to only one space on the surface of a web page, so it's often quite possible to halve the size of a web page, by removing all excessive white space.

## Types

Reputed JavaScript optimizers such as JSMIn <sup>[1]</sup> and Packer <sup>[2]</sup>, themselves programmed in JavaScript, are specially designed for modern web programming techniques, and are able to understand and preserve conditional comments, and similar. Packer, for instance can optionally Base64 compress the given source code in a manner that can be decompressed by regular web browsers, shrink variable names that are typically 5-10 characters to single letters, reducing the file size of the script, and therefore making it download faster.<sup>[3]</sup> Google has released their Closure Compiler <sup>[4]</sup> which also provides minification as well as the ability to introduce more aggressive renaming, removing dead code, and providing function inlining. Certain online tools, such as the Yahoo! UI Compressor <sup>[5]</sup> or Pretty Diff <sup>[6]</sup> also compress CSS files.

## Web Development

Listed below are certain Web Development components that can be built into websites. These optimize and quicken page load times by reducing the size of various files. Content encoding is an approach taken by compatible web servers and modern web browsers to compress HTML and related textual content, often in the gzip format.

- JavaScript Optimizer <sup>[7]</sup> - easy management of JavaScript and CSS resources and to reduce the amount of data transferred between the server and the client.
- Pretty Diff tool <sup>[6]</sup> - A JavaScript based data differencing engine with integrated minify and beautify components.
- pack:tag <sup>[8]</sup> (Documentation <sup>[9]</sup>) - A JSP Taglib for minifying JavaScript and CSS on the fly, with caching, combination and compressing.
- JAWR <sup>[10]</sup> - A library for Java web applications that joins javascript files and then minifies and compresses them, fostering a modular approach to development. Its main benefit is that developers can switch from the joined, compressed form to separate and uncompressed versions of the scripts, without the need to change the pages.
- Minify <sup>[11]</sup> - A PHP library that combines, minifies, and caches JavaScript and CSS files on the fly.
- ShrinkSafe <sup>[12]</sup>, from the Dojo toolkit.
- Lightweight Minify <sup>[13]</sup>.

## Compressing HTTP proxy

Ziproxy, a forwarding, non-caching, compressing HTTP proxy targeted for traffic optimization, supports functionalities of code compression by means of optimization of code, named *HTMLOpt*, *CSSopt* and *JSopt* (HTML/CSS/JS) which are analogous with Minification. Ziproxy works by recompressing pictures, zipping text and data optimization.

## See also

- Copy protection
- Reverse engineering
- Obfuscated code

## References

- [1] <http://www.crockford.com/javascript/jsmin.html>
- [2] <http://dean.edwards.name/packer/>
- [3] Packer version 3.0 feature list (<http://dean.edwards.name/weblog/2007/04/packer3/>)
- [4] [http://code.google.com/closure/compiler/docs/gettingstarted\\_ui.html](http://code.google.com/closure/compiler/docs/gettingstarted_ui.html)
- [5] <http://developer.yahoo.com/yui/compressor/>
- [6] <http://prettydiff.com/>
- [7] <http://js-optimizer.sourceforge.net>
- [8] <http://sf.net/projects/packtag>
- [9] <http://www.galan.de/projects/packtag>
- [10] <https://jawr.dev.java.net>
- [11] <http://code.google.com/p/minify/>
- [12] <http://dojotoolkit.org/docs/shrinksafe>
- [13] <http://razorsharpcode.blogspot.com/2010/02/lightweight-javascript-and-css.html>

# Objective-J

---

<b>Paradigm</b>	Multi-paradigm: reflective, object oriented, functional, imperative, scripting
<b>Appeared in</b>	2008
<b>Developer</b>	280 North, Inc.
<b>Typing discipline</b>	dynamic, weak, duck
<b>Influenced by</b>	Objective-C, JavaScript
<b>License</b>	LGPL
<b>Website</b>	<a href="http://cappuccino.org/">http://cappuccino.org/</a>

**Objective-J** is a programming language developed as part of the Cappuccino web development framework. Its syntax is nearly identical to the Objective-C syntax and it shares with JavaScript the same relationship that Objective-C has with the C programming language: that of being a strict, but small, superset; adding traditional inheritance and Smalltalk/Objective-C style dynamic dispatch. Pure JavaScript, being a prototype-based language, already has a notion of object orientation and inheritance, but Objective-J adds the use of class-based programming to JavaScript.

Programs written in Objective-J need to be preprocessed before being run by a web browser's JavaScript virtual machine. This step can occur in the web browser at runtime or by a compiler which translates Objective-J programs into pure JavaScript code. The Objective-J compiler is written in JavaScript; consequently, deploying Objective-J programs does not require a web browser plug-in.

## Application

The first widely known use of Objective-J was in the Cappuccino-based web application 280 Slides. Even though Objective-J can be used (and has been designed) independently from the Cappuccino framework, Objective-J has primarily been invented to support web development in Cappuccino.

## Syntax

Objective-J is a superset of JavaScript, which implies that any valid JavaScript code is also valid Objective-J code. The following example shows the definition in Objective-J of a class named `Address`; this class extends the root object `CPObject`, which plays a role similar to the Objective-C's `NSObject`. This example differs from traditional Objective-C in that the root object reflects the underlying Cappuccino framework as opposed to Cocoa, Objective-J does not use pointers and, as such, type definitions do not contain asterisk characters (in Objective-C, all objects must be dynamically allocated). In addition, instance variable definitions are never done in the `@implementation` file.

```
@implementation Address : CPObject
{
    CPString name;
    CPString city;
}
- (id)initWithName:(CPString)aName city:(CPString)aCity
{
    self = [super init];
```

```
    name = aName;
    city = aCity;

    return self;
}
-(void)setName:(CPString)aName
{
    name = aName;
}
-(CPString)name
{
    return name;
}
+(id)newAddressWithName:(CPString)aName city:(CPString)aCity
{
    return [[self alloc] initWithName:aname city:aCity];
}

@end
```

As with Objective-C, class method definitions and instance method definitions start with '+' (plus) and '-' (dash), respectively.

## Memory management

Like Objective-C 2.0's garbage-collected mode, objects in Objective-J do not need to be manually released because they are automatically freed by JavaScript's garbage collector.

## See also

- Cappuccino (Application Development Framework)

## External links


- "Learning Objective-J"<sup>[1]</sup>. Cappuccino Web Framework.

## References

- [1] <http://cappuccino.org/learn/tutorials/objective-j-tutorial.php>

# John Resig

---

John Resig	
	
<b>Born</b>	May 8, 1984
<b>Residence</b>	Boston, USA
<b>Alma mater</b>	Rochester Institute of Technology
<b>Occupation</b>	JavaScript Evangelist
<b>Employer</b>	Mozilla Corporation
<b>Known for</b>	jQuery
<b>Website</b> <a href="http://ejohn.org">ejohn.org</a> <sup>[1]</sup> <a href="http://ejohn.org/blog">ejohn.org/blog</a> <sup>[2]</sup>	

**John Resig** is a JavaScript Evangelist for the Mozilla Corporation and the creator and lead developer of the jQuery JavaScript library. This library's goal is to simplify the process of writing JavaScript code that is compatible with all web browsers.<sup>[3]</sup> For his work on jQuery he was inducted into the Rochester Institute of Technology's Innovation Hall of Fame on April 30, 2010.<sup>[4]</sup>

Resig has also been involved in the creation of several other JavaScript libraries including FUEL (for Firefox plugin development),<sup>[5]</sup> Processing.js (for visual effects) and Sizzle (for CSS selectors).<sup>[6]</sup>

"The Google Address Translation, was developed by John Resig and gives Google Maps the ability to convert any US Postal Address into Latitude/Longitude coordinates," according to WebProNews.<sup>[7]</sup>

Resig is the author of the book *Pro JavaScript Techniques* and the forthcoming *Secrets of the JavaScript Ninja*.

He is also the creator of processing.js, which is a port of the processing language for JavaScript.

He is also author of a widely read blog, [ejohn.org/blog](http://ejohn.org/blog)<sup>[8]</sup>, which has more than 13,000 subscribers.<sup>[9]</sup>

## External links

- John Resig - About Me <sup>[10]</sup>
- John Resig: JavaScript's Chuck Norris <sup>[11]</sup>

## References

- [1] <http://ejohn.org>
- [2] <http://ejohn.org/blog/>
- [3] [http://www.infoworld.com/article/07/10/04/Code-library-aims-to-fuel-easier-JavaScript-handling\\_1.html](http://www.infoworld.com/article/07/10/04/Code-library-aims-to-fuel-easier-JavaScript-handling_1.html)
- [4] <http://www.rit.edu/alumni/ihf/inductee.php?inductee=10>
- [5] <http://ejohn.org/blog/tags/extensions/>
- [6] <http://ejohn.org/blog/new-processingjs-and-sizzlejs-sites/>
- [7] <http://www.webpronews.com/topnews/2005/07/11/google-maps-address-hack>
- [8] <http://www.ejohn.org/blog/>
- [9] <http://twitter.com/jeresig/statuses/1090013257>
- [10] <http://ejohn.org/about/>
- [11] <http://benalman.com/news/2009/12/john-resig-javascripts-chuck-norris/>
- Presentation on JavaScript & jQuery @ NEU ACM (<http://video.google.com/videoplay?docid=-7485992465859932389>)

## Reverse Ajax

---

**Reverse Ajax** refers to an Ajax design pattern that uses long-lived HTTP connections to enable low-latency communication between a web server and a browser. Basically it is a way of sending data from client to server and a mechanism for pushing server data back to the browser.<sup>[1] [2]</sup>

This server–client communication takes one of two forms:

- Client polling: the client repeatedly queries (polls) the server and waits for an answer.
- Server pushing: a connection between a server and client is kept open and the server sends data when available.

Reverse Ajax describes the implementation of either of these models, or a combination of both. The design pattern is also known as *Ajax Push*, *Full Duplex Ajax* and *Streaming Ajax*.

## Examples

The following is a simple example. Imagine we have 2 clients and 1 server, and client1 wants to send the message "hello" to every other client.

**With traditional Ajax** (polling):

- **client1** sends the message "hello"
- **server** receives the message "hello"
- **client2** polls the server
- **client2** receives the message "hello"
- **client1** polls the server
- **client1** receives the message "hello"

**With reverse Ajax** (pushing):

- **client1** sends the message "hello"
- **server** receives the message "hello"
- **server** sends the message "hello" to all clients

Less traffic is generated with Reverse Ajax and messages are transferred with less delay (low-latency).

---

## External links

- The Slow Load Technique/Reverse AJAX - Simulating Server Push in a Standard Web Browser <sup>[3]</sup>
- Exploring Reverse Ajax <sup>[4]</sup>
- Reverse Ajax with DWR (an Java Ajax framework) <sup>[5]</sup>
- Changing the Web Paradigm - Moving from traditional Web applications to Streaming-AJAX <sup>[6]</sup>

## References

- [1] Crane, Dave; McCarthy, Phil (July 2008) (in English). *Comet and Reverse Ajax: The Next Generation Ajax 2.0*. Apress. ISBN 1590599985.
- [2] Martin, Katherine (2007-03-22). "Developing Applications using Reverse Ajax" (<http://today.java.net/pub/a/today/2007/03/22/developing-applications-using-reverse-ajax.html>). java.net, O'Reilly and CollabNet. .
- [3] [http://www.obviously.com/tech\\_tips/slow\\_load\\_technique](http://www.obviously.com/tech_tips/slow_load_technique)
- [4] <http://gmapsdotnetcontrol.blogspot.com/2006/08/exploring-reverse-ajax-ajax.html>
- [5] <http://ajaxian.com/archives/reverse-ajax-with-dwr>
- [6] [http://www.lightstreamer.com/Lightstreamer\\_Paradigm.pdf](http://www.lightstreamer.com/Lightstreamer_Paradigm.pdf)

# Rico (Ajax)

---

**Rico** is an open source JavaScript library for developing rich Internet applications (RIAs) that use Ajax.<sup>[1]</sup>

Rico uses the Prototype Javascript Framework and JSON libraries.

## Features

- **LiveGrid** - One of Rico's best known features is its LiveGrid.
  - **Animation Effects** - Rico 2.0 provides responsive animation for smooth effects and transitions that can communicate change in richer ways than traditional web applications have explored before. Unlike most effects, Rico 2.0 animation can be interrupted, paused, resumed, or have other effects applied to it to enable responsive interaction that the user does not have to wait on.
  - **Behaviors** - Rico can be used to create components that behave similar to those found in Adobe Flex and OpenLaszlo.
  - **Styling** - Rico provides several cinematic effects as well as some simple visual style effects in a very simple interface.
  - **Drag and Drop** - Rico provides a simple interface for enabling web application to support drag and drop.
  - **Ajax Support** - Because Rico uses the Prototype library, which contains classes for using Ajax, Rico provides a very simple interface for registering Ajax request handlers as well as HTML elements or JavaScript objects as Ajax response objects. Multiple elements and/or objects may be updated as the result of one Ajax request.
-

## See also

- JSON
- Prototype JavaScript Framework
- Rich Internet application
- Ajax (programming)

## External links

- Rico <sup>[2]</sup> - Project on SourceForge
- Rico <sup>[3]</sup> - Original Homepage

## References

- [1] Asleson, Ryan & Schutta, Nathaniel T. (2006). Foundations of Ajax. Apress. ISBN 1-59059-582-3
- [2] <http://sourceforge.net/projects/openrico/>
- [3] <http://openrico.org>
-



# Seed (programming)

---

<b>Developer(s)</b>	Robert Carr, Matt Arsenault and Tim Horton
<b>Initial release</b>	November 8, 2008
<b>Discontinued</b>	2.30 "Piano Man" / March 29, 2010
<b>Preview release</b>	2.31.1 "The Black Album" / March 29, 2010
<b>Development status</b>	Active
<b>Written in</b>	C
<b>Platform</b>	Cross-platform
<b>Available in</b>	English
<b>Type</b>	Interpreter, library
<b>License</b>	GNU LGPL
<b>Website</b>	<a href="http://live.gnome.org/seed">live.gnome.org/seed</a> <sup>[1]</sup>

**Seed** is a JavaScript interpreter and a library of the GNOME project to create standalone applications in JavaScript. It uses the JavaScript engine JavaScriptCore of the WebKit project. It is possible to easily create modules in C.

Seed is integrated in GNOME since the 2.28 version<sup>[2]</sup>.

Seed is used by two games of the GNOME Games package and by Epiphany for the design of its extensions.

## Hello world in Seed

```
#!/usr/bin/env seed

print("Hello, world!");
```

## A program using GTK+

This code shows an empty window named "Example".

```
#!/usr/bin/env seed

Gtk = imports.gi.Gtk;
Gtk.init(null, null);

var window = new Gtk.Window({title: "Example"});

window.signal.hide.connect(Gtk.main_quit);

window.show_all();
Gtk.main();
```

## Modules

To use a module, just instantiate a class having for name ***imports***. followed by the name of the module respecting the case sensitivity.

- The modules using GObject Introspection <sup>[3]</sup>, who starts by ***imports.gi***. <sup>[4]</sup> :
  - Gtk
  - Gst
  - GObject
  - Gio
  - Clutter
  - GLib
  - Gdk
  - WebKit
  - GdkPixbuf <sup>[5]</sup>
- Libxml
- Cairo
- DBus
- MPFR
- Os (system library)
- Canvas (using Cairo)
- multiprocessing
- readline <sup>[6]</sup>
- ffi
- sqlite
- sandbox <sup>[7]</sup>

## List of the Seed versions

The names of the versions of Seed are albums of famous rock bands.

Version	Nom de code	Date de publication
0.1		8 November 2008
0.3	Wednesday Morning 3AM	2 January 2009
0.5	Transformer	16 April 2009
0.6	Beatles for Sale	29 April 2009
0.7	Another Side of Bob Dylan	13 May 2009
0.8	Bringing It All Back Home	29 May 2009
0.8.5	Self Portrait	10 July 2009
2.27.90	London Calling	10 August 2009
2.27.91	Yellow Submarine	21 August 2009
2.27.92	Metal Machine Music	7 September 2009
2.28.0	The Rise and Fall of Ziggy Stardust and the Spiders	21 September 2009
2.29.2	Never Mind the Bollocks	16 November 2009
2.29.3		30 November 2009
2.29.4		17 December 2009

2.29.5	Icky Thump	1 January 2010
2.29.5.1	Achtung Baby	1 January 2010
2.29.5.2	Third Stage	7 January 2010
2.29.5.3	Twist and Shout	11 January 2010
2.29.90	Fort Nightly	8 February 2010
2.29.91	Greatest Hits	23 February 2010
2.30.0	Piano Man	29 March 2010
2.31.1	The Black Album	29 March 2010

## See also

- GNOME
- JavaScript
- Server-side JavaScript
- JavaScriptCore

## References

*This article incorporates information from (informatique) this version <sup>[8]</sup> of the equivalent article on the French Wikipedia.*

- [1] <http://live.gnome.org/Seed>
- [2] Seed, the module! (<http://www.hortont.com/blog/2009/07/seed-the-module/>)
- [3] <http://live.gnome.org/GObjectIntrospection>
- [4] <http://devel.akbkhomes.com/seed/index.shtml>
- [5] <http://devel.akbkhomes.com/seed/GdkPixbuf.shtml>
- [6] <http://library.gnome.org/devel/seed/stable/readline-module.html>
- [7] <http://library.gnome.org/devel/seed/stable/Sandbox-module.html>
- [8] <http://fr.wikipedia.org/wiki/Seed>

## External links

- Seed on the GNOME wiki (<http://live.gnome.org/Seed>)
- Seed documentation (<http://library.gnome.org/devel/seed/stable/>)
- A auto-generated documentation of the Seed modules (<http://devel.akbkhomes.com/seed/>)
- Official tutorial of Seed (<http://live.gnome.org/Seed/Tutorial>)
- A short tutorial (<http://people.gnome.org/~racarr/seed/tutorial-standalone/tutorial.html>) showing how to create a basic web browser using WebKitGTK+.
- Blog of Robert Carr (<http://blogs.gnome.org/racarr/>)

## Relative external links

- gjs (<http://live.gnome.org/Gjs>)
- kjscmd (<http://xmelegance.org/kjseembed/>)
- GLUEScript (<http://gluescript.sf.net/>)
- XULRunner (<https://developer.mozilla.org/en/XULRunner>) (XUL, XULRunner)
- Rhino JavaScript Shell (<http://www.mozilla.org/rhino/shell.html>) (Rhino Shell)
- mjs (<http://manpages.ubuntu.com/manpages/karmic/en/man1/mjs.1.html>)

# Server-side JavaScript

---

**Server-side JavaScript (SSJS)** refers to JavaScript that runs on the server-side. This term was coined because the language is predominantly used on the client-side, i.e. client-side JavaScript (CSJS).

The first implementation of SSJS was Netscape's LiveWire, included in their Enterprise Server 2.0 product, released in 1996. CommonJS is a project to provide common specifications for SSJS development<sup>[1]</sup>.

## Specifications

- Reference for Server-Side JavaScript 1.2 <sup>[2]</sup>
- Guide for Server-Side JavaScript 1.2 <sup>[3]</sup>
- CommonJS Specifications to unify SSJS APIs <sup>[4]</sup>

## See also

- JavaScript
- Comparison of Server-side JavaScript solutions

## External links

- The Server-Side JavaScript Google Group <sup>[5]</sup> dedicated to creating cross-platform SSJS standard APIs.
- Mozilla JavaScript shells <sup>[6]</sup> especially section "Standalone JavaScript shells"

## References

- [1] <http://commonjs.org>
  - [2] <http://research.nihonsoft.org/javascript/ServerReferenceJS12/index.htm>
  - [3] <http://research.nihonsoft.org/javascript/ServerGuideJS12/index.htm>
  - [4] <http://wiki.commonjs.org/>
  - [5] <http://groups.google.com/group/commonjs>
  - [6] [https://developer.mozilla.org/en/JavaScript\\_shells](https://developer.mozilla.org/en/JavaScript_shells)
-

# Comparison of Server-side JavaScript solutions

This is a list of Server-side JavaScript solutions.

## Server-side JavaScript use

### Examples of current uses of JavaScript on the server side

Project/product name	JavaScript Engine	Server Platform(s)	Comments	Website
10gen	Rhino	10gen application server	Uses the Rhino parser. Compiles Javascript to Java.	[1]
Acre	Rhino	Jetty HTTP Server	Integrated with Freebase to power Freebaseapps.com <sup>[2]</sup> , a collaborative hosting environment.	
Apache Sling	Rhino	Any Java servlet container and standalone.	Sling is a generic Java web application framework that allows to use any script language via the standard JavaScript Engine interface. Sling is RESTful by design and sits on top of a Java Content Repository, giving scripts full access to the JCR.	[3]
APE	SpiderMonkey	Standalone HTTP	Used to write custom server modules (manage message queue, users, channels, sockets, http, ...). [4]	[5]
AppengineJS	Rhino	Google App Engine	AppengineJS is a port of the Google Appengine Python SDK to JavaScript.	[1]
AppJet	Rhino(modified)		Also provides hosting in a virtual machine	
Aptana Jaxer	Mozilla + SpiderMonkey	Apache HTTP Server	A community open source ajax-server based on the Mozilla browser (DOM + JavaScript engine). HTML, JavaScript, and CSS are native to Jaxer, as are XMLHttpRequests, JSON, DOM scripting, etc. It offers access to databases, files, and networking, as well as logging, process management, scalability, security, integration APIs, and extensibility.	[6]
ASP	JScript	IIS		[7] [8]js.web [1] [9]js.net [10]
Axiom Stack	Rhino	Jetty HTTP Server	Actively developed open source SSJS server. JSON, E4X, List Comprehensions. Complete access to Java APIs. Built-in security. Lucene data store by default with JDBC access to relational databases.	[11]
Cocoon Flowscript	Rhino	Apache Cocoon		[12]
CouchDB	SpiderMonkey	Standalone HTTP	Used in MapReduce and update validation functions as well as to transform JSON documents and view results into HTML or other content-types.	
DovetailDb	Rhino	Jetty HTTP Server	DovetailDB is a schemaless, JSON-based database with an Apache license. You can use the hosted database or install your own. You can override the handlers to supplement with access control, use map/reduce, install JavaScript stored procedures to call from the client, and more.	[13]

Eclipse e4	Rhino	Equinox OSGi, bundled with Jetty, any servlet container (using the servlet bridge)	Extensions can be written in JS, not just Java, especially servlets using the OSGi HTTP Service. Frontends can be developed with Eclipse RAP using the SWT and JFace APIs, or any other UI framework. Focus is on modularity (plug-ins), extensibility, scalability.	[14]
EditMe	Rhino	Hosted (Tomcat)	Wiki with embedded JavaScript engine and API for application development within the wiki	[15]
Ejscript	Ejscript	Appweb HTTP Server, Apache HTTP Server	Enhanced ECMA-262 Language and (Ruby on Rails like) web framework including Model/View/Controller paradigm, SQLite database connectivity, Ajax libraries and a suite of view controls. Includes command line generators and tools.	[16] [17]
ESXX	Rhino	FastCGI, stand-alone HTTP, any Java servlet container, Google App Engine	Focus is on ease of use, web security and XML/XSLT (via Saxon). Has SQL, LDAP, HTTP/REST, SOAP and Java support. Can also execute JS scripts from the command line.	[18]
GLUEScript	SpiderMonkey	Apache HTTP Server, FastCGI and Stand-alone	Glueing Libraries Using EcmaScript (GLUE) ports sqlite, mysql, memcached, wxWidgets, POCO, ... to JavaScript. GLUEScript is the successor of wxJavaScript	[19] [20] [21]
Google Apps Script			Automate simple tasks across Google Products	[22]
GromJS	SpiderMonkey	Bauk, FastCGI and Stand-alone using gromjscli	GromJS Server-Side JavaScript interpreter includes support for files (open, read, write, lock, seek, truncate, etc.), MySQL, PostgreSQL and SQLite databases, session variables, hash arrays, pipes, HTTP file-upload, cookies and more.	[23]
Helma	Rhino	Jetty HTTP Server	Complete package with web server and framework.	[24]
im-jssp	Rhino	Resin and any other Java servlet container.	"im-jssp" is a template engine that used Server Side JavaScript and HTML. And "im-jssp" has custom tag "jsspRpc" that is Server Side JavaScript can be called from Client Side JavaScript seamlessly.	[25] (Japanese)
Jack	Multiple	Multiple	A JavaScript engine and server-agnostic interface, much like Rack for Ruby and WSGI for Python.	[26]
jsex	SpiderMonkey	Apache HTTP Server, lighttpd and Stand-alone	Has modules for fastCGI, CGI, MySQL, SQLite, FTP, SMTP, HTTP, AJAX, JSON, SOAP, WSDL and more. Automatic inclusion of C libraries.	[27]
jslibs	SpiderMonkey	FastCGI and Stand-alone (TCP)	Native libraries bindings: NSPR, SQLite, FastCGI, libTomCrypt, libsvg, libpng, libjpeg, FreeType, libffi, ...	[28]
JSSP	Rhino	Any Java servlet container.	Contains a modified Rhino version for embedded SQL support	[29]
Juice	Flusspferd (SpiderMonkey)	Stand-alone	Built around the CommonJS specs.	[30]
Junction	Rhino	Apache HTTP Server	Ruby on Rails port to javascript	[31]
jsdb	SpiderMonkey	Stand-alone		[32] [33]
Livelink WCM Presentation Server	SpiderMonkey	Apache HTTP Server, IIS and Sun Java System Web Server	A commercial content management system used in a number of European public-sector and financial organizations. It uses SSJS for dynamic features and to customize the behavior of built-in objects. Though still developed and supported for existing users, this SSJS product is no longer available for sale to new customers. <sup>[34]</sup>	

mod_js	SpiderMonkey	Apache HTTP Server	mod_js is a stand-alone module for Apache that is able to execute JavaScript on the server. Now discontinued.	[35]
Myna Javascript Application Server	Rhino	Any Java servlet container, packaged with winstone	Myna is a general purpose, 100% open source, web platform modeled after Cold Fusion with centralized datasources, Web-based Administration, runtime Object Relational Mapping, templating, LDAP access, strong cryptography, SOAP and XML-RPC publishing and more.	[36]
Narwhal	Rhino, JavaScriptCore, XULRunner, V8, others	Multiple	An implementation of the CommonJS standard for multiple JavaScript interpreters.	[37]
Node.js	V8	Standalone	Javascript asynchronous, Event-based I/O. Influenced by systems like Ruby's Event Machine or Python's Twisted. Few modules available (yet).	[38]
NOTEX	SpiderMonkey	CGI/FastCGI script on any web server	NOTEX is a CGI script to run JavaScript files hosted anywhere on the network to process XML data with E4X	[39]
OpenMocha	Helma	Stand-alone		[40]
Opera	Futhark	Opera Unite	Javascript is the server-side language used to develop services for the Opera Unite feature of the Opera browser. This is a server built in to the browser. The javascript API includes local file access to a virtual sandboxed file-system and persistent storage via persistent global variables.	[41]
Persevere	Rhino	Jetty HTTP Server	JSON database integrated with JavaScript environment with a HTTP/REST, JSON-RPC, and Comet interfaces.	[42]
Phobos	Rhino	Glassfish and any other Java servlet container.		
POW	SpiderMonkey	Mozilla Firefox	A Mozilla Firefox Extension which adds a server to your browser. Templates can use SSJS.	[43]
PyV8	V8	Standalone	Python Wrapper for Google V8 Javascript Engine which could be used in the web framework or script	[44]
Rhino in Spring	Rhino	Spring		[45]
Rhinola	Rhino	Apache HTTP Server	Uses gcj	[46]
RingoJS	Rhino	Jetty HTTP Server	CommonJS-compliant JavaScript platform formerly known as Helma NG.	[47]
Server Side Execution (SSX)	Rhino	Devwex	JSP/ASP-like scripting; supports threads, object (de)serialization, access to local and remote filesystems, database connection via jdbc; released under the GPL2 license	[48]
Server Side Javascript	Rhino	Jetty 6	For writing servlets.	[49]
SSJS	SpiderMonkey	Synchronet BBS Integrated Http Server	Current developer trunk (3.15a) supports E4X	[50]
Torino	Rhino	Tomcat and any other Java servlet container.	Server-side JavaScript environment for developing web applications. Provides a rich server-side programming environment using Java APIs. Released under the GPL3 license.	[51]
v8cgi	V8	Apache, cgi, fcgi and Stand-alone	Libraries for GD, HTTP, MySQL, Sessions, Sockets and templating.	[52]

Whitebeam	SpiderMonkey	Apache HTTP Server	Server-side JavaScript environment for developing web applications. Includes integration with graphics libraries and the PostgreSQL database. Actively developed and in use supporting commercial sites. Released under the BSD license. Latest release supports E4X.	[53]
wxJavaScript	SpiderMonkey	Apache HTTP Server and Stand-alone	With E4X and wxWidgets ported classes. GLUEScript is the successor of wxJavaScript	[54]
Xpages	IBM JS	Lotus Domino Server	Compiles Javascript to Java, which then runs in Domino context	[55]

Other common server-side programming languages are JSP, ASP, Perl, PHP, Python, Ruby and ColdFusion amongst others.

## See also

- JavaScript
- Server-side JavaScript
- CommonJS implementations

## External links

- The Server-Side JavaScript Google Group <sup>[5]</sup> dedicated to creating cross-platform SSJS standard APIs.
- Mozilla JavaScript shells <sup>[6]</sup> especially section "Standalone JavaScript shells"
- Mozilla Rhino JavaScript Shell <sup>[56]</sup>
- Gnome JavaScript Programing <sup>[57]</sup> (Seed, Gjs <sup>[59]</sup>) /\*ref\*/ <sup>[4]</sup>
- KDE JavaScript <sup>[58]</sup> Programing <sup>[59]</sup> (kjscmd/\*kjsemlbed\*/ <sup>[60]</sup>) /\*ref\*/ <sup>[60]</sup>

## References

- [1] <http://www.10gen.com/>
- [2] <http://freebaseapps.com>
- [3] <http://sling.apache.org/>
- [4] [http://www.ape-project.org/wiki/index.php/How\\_to\\_build\\_a\\_serverside\\_JS\\_module](http://www.ape-project.org/wiki/index.php/How_to_build_a_serverside_JS_module)
- [5] <http://www.ape-project.org/>
- [6] <http://www.aplana.com/jaxer/>
- [7] [http://msdn.microsoft.com/en-us/library/ee532514\(VS.90\).aspx](http://msdn.microsoft.com/en-us/library/ee532514(VS.90).aspx)
- [8] [http://msdn.microsoft.com/en-us/library/ms525070\(VS.90\).aspx](http://msdn.microsoft.com/en-us/library/ms525070(VS.90).aspx)
- [9] <http://msdn.microsoft.com/en-us/library/system.web.aspx>
- [10] [http://msdn.microsoft.com/en-us/library/x85xxsf4\(VS.71\).aspx](http://msdn.microsoft.com/en-us/library/x85xxsf4(VS.71).aspx)
- [11] <http://www.axiomstack.com/>
- [12] <http://cocoon.apache.org/2.1/userdocs/flow/api.html>
- [13] <http://www.millstonecw.com/dovetaildb/>
- [14] <http://wiki.eclipse.org/E4/JavaScript>
- [15] <http://www.editme.com/>
- [16] <http://www.ejscrip.org/>
- [17] <http://ejscrip.org/products/ejs/doc/api/ejscrip/index.html>
- [18] <http://esxx.org/>
- [19] <http://gluescript.sf.net/>
- [20] <http://gluescript.sourceforge.net/?q=node/31>
- [21] [http://docs.wxwidgets.org/stable/wx\\_classref.html#classref](http://docs.wxwidgets.org/stable/wx_classref.html#classref)
- [22] <http://www.google.com/google-d-s/scripts/scripts.html>
- [23] <http://bauk.ws/gromjs.jsx>
- [24] <http://dev.helma.org/>
- [25] <http://oss.intra-mart.org/projects/im-jssp/>
- [26] <http://jackjs.org/>
- [27] <http://jsxt.sourceforge.net/>



- [28] <http://jslibs.googlecode.com/>
  - [29] <http://jssp.de/>
  - [30] <http://juicejs.org/>
  - [31] <http://code.google.com/p/trimpath/>
  - [32] <http://www.jsdb.org/>
  - [33] <http://www.jsdb.org/reference.html>
  - [34] "A letter to Livelink WCM customers" (<http://www.opentext.com/2/global/sol-products/pro-wcm/letter-wcm-customers.htm>). . Retrieved 2009-01-24.
  - [35] <http://modjs.org>
  - [36] <http://www.mynajs.org>
  - [37] <http://narwhaljs.org>
  - [38] <http://nodejs.org>
  - [39] <http://www.notex.info/>
  - [40] <http://openmocha.org/openmocha/>
  - [41] <http://unite.opera.com/>
  - [42] <http://www.persvr.org/>
  - [43] <https://addons.mozilla.org/firefox/3002/>
  - [44] <http://pyv8.org/>
  - [45] <http://rhinoinspring.sourceforge.net>
  - [46] <http://mod-gcj.sf.net/rhinola.html>
  - [47] <http://ringojs.org/>
  - [48] <http://www.seanox.de/projects.ssx.php>
  - [49] <http://www.bluishcoder.co.nz/2006/05/server-side-javascript.html>
  - [50] <http://www.synchro.net/>
  - [51] <http://torino.sourceforge.net/>
  - [52] <http://code.google.com/p/v8cgi/>
  - [53] <http://www.whitebeam.org/>
  - [54] <http://www.wxjavascript.net/>
  - [55] <http://www-10.lotus.com/ldd/ddwiki.nsf>
  - [56] <http://www.mozilla.org/rhino/shell.html>
  - [57] <http://live.gnome.org/JavaScript>
  - [58] <http://api.kde.org/4.x-api/kdelibs-apidocs/kjs/html/>
  - [59] <http://techbase.kde.org/Development/Languages/JavaScript>
  - [60] <http://xmelegance.org/kjsembed/jsref/index.html>
-

# SproutCore

	
	
A demo application of SproutCore	
<b>Developer(s)</b>	Sproutit, Apple Inc. and community.
<b>Initial release</b>	2010
<b>Stable release</b>	1.0.1046 / March 19, 2010
<b>Development status</b>	Active
<b>Written in</b>	Ruby/JavaScript
<b>Operating system</b>	Cross-platform
<b>License</b>	MIT License
<b>Website</b>	<a href="http://www.sproutcore.com/">http://www.sproutcore.com/</a>

**SproutCore** is an open-source JavaScript framework. Its goal is to allow developers to create web applications with advanced capabilities and a user experience comparable to that of desktop applications. When developing a SproutCore application, all code is written in JavaScript (including the view layer in 1.0). SproutCore, initially created in 2007 by Sproutit as the basis for their Mailroom application, is available under the MIT License.

Apple announced MobileMe at WWDC in 2008, noting that much of it was built using SproutCore. Apple has contributed greatly to the project as part of a Web 2.0 initiative. SproutCore is also used at iWork.com, the online extension of the iWork productivity software by Apple.

Latest stable SproutCore release is 1.0, a major milestone with significant improvements, and was released on March 19th in 2010<sup>[1]</sup>. SproutCore implements some CommonJS specifications in the upcoming 1.1 release<sup>[2]</sup>.

## Notes

- "Apple adopting SproutCore for web applications" <sup>[3]</sup>. Macrumors. 2008-06-16.
- "Apple's open secret: SproutCore is Cocoa for the web" <sup>[4]</sup>. Appleinsider. 2008-06-16.
- "Want to try out Apple's MobileMe? Check out SproutCore" <sup>[5]</sup>. Techcrunch. 2008-06-09.
- "Cocoa for Windows+Flash killer=SproutCore" <sup>[6]</sup>. Roughly Drafted. 2008-06-14.

## External links

- [Sproutcore.com](http://sproutcore.com) homepage <sup>[7]</sup>
- [SproutCore Wiki](#) <sup>[8]</sup>
- [SproutCore source code](#) <sup>[9]</sup>
- an Introduction to SproutCore <sup>[10]</sup>, discussing desktop vs. browser development, key/value observation, data bindings, demos, SproutCore features and API, and example SproutCore apps. 2009-12-30
- ListenApp. Web application done with SproutCore 1.0 <sup>[11]</sup>

- Tasks. Tasks management app done with SproutCore 1.0 (login as 'guest' no password) <sup>[12]</sup>
- Lebowski Framework - A test automation framework for SproutCore <sup>[13]</sup>
- itsgotwhatplantscrave.com - A SproutCore focused blog <sup>[14]</sup>
- frozencanuck.wordpress.com - Another SproutCore focused blog <sup>[15]</sup>

## References

- [1] Blog post about 1.0 release (<http://blog.sproutcore.com/post/459714079/sproutcore-1-0-is-done>)
- [2] <http://wiki.commonjs.org/wiki/Implementations/SproutCore>
- [3] <http://www.macrumors.com/2008/06/16/apple-adopting-sproutcore-for-web-applications>
- [4] [http://www.appleinsider.com/articles/08/06/16/apples\\_open\\_secret\\_sproutcore\\_is\\_cocoa\\_for\\_the\\_web.html](http://www.appleinsider.com/articles/08/06/16/apples_open_secret_sproutcore_is_cocoa_for_the_web.html)
- [5] <http://www.techcrunch.com/2008/06/09/want-to-try-out-mobileme-check-out-sproutcore/>
- [6] <http://www.roughlydrafted.com/2008/06/14/cocoa-for-windows-flash-killer-sproutcore/>
- [7] <http://www.sproutcore.com/>
- [8] <http://wiki.sproutcore.com/>
- [9] <http://www.sproutcore.com/download/>
- [10] <http://www.infoq.com/presentations/subelsky-sproutcore-intro/>
- [11] <http://listenapp.com/>
- [12] <http://tasks-demo.appspot.com/>
- [13] <http://github.com/FrozenCanuck/Lebowski>
- [14] <http://www.itsgotwhatplantscrave.com/>
- [15] <http://frozencanuck.wordpress.com/>

# Unobtrusive JavaScript

---

"**Unobtrusive JavaScript**" is an emerging technique in the JavaScript programming language, as used on the World Wide Web. Though the term is not formally defined, its basic principles are generally understood to include:

- Separation of functionality (the "behavior layer") from a Web page's structure/content and presentation<sup>[1]</sup>
- Best practices to avoid the problems of traditional JavaScript programming (such as browser inconsistencies and lack of scalability)
- Progressive enhancement to support user agents that may not support advanced JavaScript functionality<sup>[2]</sup>

## The need for a new paradigm

JavaScript historically has had a reputation for being a clumsy, hackish language unsuitable for serious application development.<sup>[3]</sup> <sup>[4]</sup> This has been largely due to inconsistent implementations of the language itself and the Document Object Model<sup>[5]</sup> in various browsers, and the widespread use of buggy cut-and-paste code. Runtime errors were so common (and so difficult to debug) that few programmers even tried to fix them, as long as the script behaved more or less the way it was supposed to;<sup>[6]</sup> scripts often failed completely in some browsers.

The recent emergence of standardized browsers, JavaScript frameworks and high-quality debugging tools have made organized, scalable JavaScript code possible, and the emergence of AJAX interfaces has made it essential.

Whereas JavaScript was once reserved for relatively simple and non-critical tasks such as form validation and decorative novelties, it is now being used to write large, complex codebases that are often part of a site's core functionality. Run time errors and unpredictable behavior are no longer minor annoyances; they are fatal flaws.

Advocates of Unobtrusive JavaScript see it as part of the larger Web standards movement; much as the demand for cross-browser compatibility has driven the increasing emphasis on standardized markup and style, the increasing demand for rich Internet applications is driving the movement toward better practices with the use of JavaScript. The concept of *Unobtrusiveness* in relation to JavaScript programming was coined in 2002 by Stuart Langridge<sup>[7]</sup> in the article "Unobtrusive DHTML, and the power of unordered lists <sup>[8]</sup>".<sup>[9]</sup> In the article Langridge argues for a way to

keep all JavaScript code, including event handlers, outside of the HTML. Stuart Langridge has since expanded upon this thought in book<sup>[10]</sup> and article format.<sup>[11]</sup>

## Separation of behavior from markup

Traditionally, JavaScript often was placed inline together with an HTML document's markup. For example, the following is a typical implementation of JavaScript form validation when written inline:

```
<input type="text" name="date" onchange="validateDate()" />
```

Adherents to "Unobtrusive Javascript" argue that the purpose of markup is to describe a document's structure, not its programmatic behavior and that combining the two negatively impacts a site's maintainability for similar reasons that combining content and presentation does. They also argue that inline event handlers are harder to use and maintain, when one needs to set several events on a single element or when one is using event delegation. Nor can they be used with custom events.

The unobtrusive solution is to register the necessary event handlers programmatically, rather than inline. That is to say, rather than using the event handlers as shown with the hyperlink above to directly fire the desired action, the following methods are used. These are commonly achieved by assigning a particular CSS selector to all elements which need to be acted upon by the script:

```
<input type="text" name="date" />
```

The script can then look for all input elements with the name date, and set them up accordingly:

Using native JavaScript, HTML5 events and DOM Level 2 event listeners:

```
document.addEventListener("domcontentloaded", function() {
    // domcontentloaded will fire when the DOM is loaded but unlike
    "load" it does not wait for images, etc.
    // It is being standardized in HTML5
    // It is assumed that there is only one element with name="date"
    document.getElementsByName("date")[0].addEventListener("change",
    validateDate, false);
}, false);

function validateDate(){
    // Do something when the content of the 'input' element with the
    name 'date' is changed.
}
```

The above example will not work in all browsers; some – particularly Internet Explorer (including version 8) – do not support DOM 2 event listeners. In practice developers often use libraries to get away from browser inconsistencies and deficiencies. The following script is specific to the MooTools JavaScript library, but accomplishes the same setup:

```
window.addEvent('domready', function() {
    $$('input[name=date]').addEvent('change', validateDate);
});
```

The following script is specific to the jQuery JavaScript library:

```
$(document).ready(function(){
    $('input[name=date]').bind('change', validateDate);
});
```

```
});
```

Because the intended purpose of the name attribute is to describe the semantic role of an element, this approach is consistent with modern standards-based markup practices.

## Best practices

Though the essence of unobtrusive JavaScript is the concept of an added separate behavior layer, advocates of the paradigm generally subscribe to a number of related principles, such as:

- DOM Scripting, i.e. adherence to the W3C DOM and event model, and avoidance of browser-specific extensions (except when really necessary, because of bugs or lacking implementations)
- Capability detection, i.e. testing for specific functionality before it is used<sup>[12]</sup>. In particular this is seen as the opposite of browser detection.
- More generally, JavaScript best practices often parallel those in other programming languages, such as encapsulation and abstraction layers, avoidance of global variables, meaningful naming conventions, use of appropriate design patterns, and systematic testing. Such principles are essential to large-scale software development, but have not been widely observed in JavaScript programming in the past; their adoption is seen as an essential component of JavaScript's transition from a "toy" language to a tool for serious development.

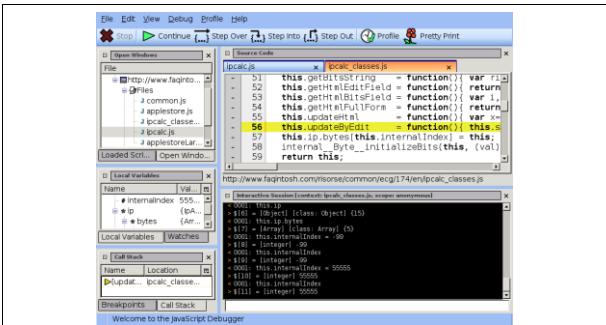
## See also

- Graceful degradation

## References

- [1] Keith, Jeremy (2006-06-20). "Behavioral Separation" (<http://www.alistapart.com/articles/behavioralseparation>). .
- [2] Olsson, Tommy (2007-02-06). "Graceful Degradation & Progressive Enhancement" (<http://accessites.org/site/2007/02/graceful-degradation-progressive-enhancement/>). .
- [3] Crockford, Douglas (2007-01-24). "The JavaScript Programming Language" (<http://yuiblog.com/blog/2007/01/24/video-crockford-tjpl/>). .
- [4] Crockford, Douglas (2006-11-27). "Advanced JavaScript" (<http://yuiblog.com/blog/2006/11/27/video-crockford-advjs/>). .
- [5] Crockford, Douglas (2006-10-20). "An Inconvenient API: The Theory of the Dom" (<http://yuiblog.com/blog/2006/10/20/video-crockford-domtheory/>). .
- [6] Crockford, Douglas (2007-06-08). "JavaScript: The Good Parts" (<http://yuiblog.com/blog/2007/06/08/video-crockford-goodstuff/>). .
- [7] "Building dynamic websites" (<http://www.netmag.co.uk/zine/dhtml-1/building-dynamic-websites>). 2006-08-09. . Retrieved 2010-05-18.
- [8] <http://www.kryogenix.org/code/browser/aqlists/>
- [9] Langridge, Stuart (2002-11). "Unobtrusive DHTML, and the power of unordered lists" (<http://www.kryogenix.org/code/browser/aqlists/>). . Retrieved 2008-08-07.
- [10] Langridge, Stuart (2005). *DHTML Utopia: Modern Web Design Using JavaScript & DOM*. SitePoint. ISBN 0-9579218-9-6. (Reference to the first edition, since it shows how the author pioneered the concept.)
- [11] Eg: Langridge, Stuart (2005-06-01). "DHTML Utopia: Modern Web Design Using JavaScript & DOM" (<http://articles.sitepoint.com/article/dhtml-utopia-modern-web-design>). . Retrieved 2000-11-03.
- [12] <http://dev.opera.com/articles/view/using-capability-detection/>

# Venkman



Screenshot of Venkman at work

Developer(s)	Mozilla Foundation / Mozilla Corporation
Stable release	0.9.87.4 / June 19, 2008
Written in	C++, XUL, XBL, JavaScript
Operating system	Cross-platform
Type	JavaScript debugger
License	MPL/GPL/LGPL tri-license
Website	[1]

**Venkman** is the JavaScript debugger component of the Mozilla Application Suite. It is also available as a Mozilla Firefox extension. Venkman is named after the character Dr. Peter Venkman played by Bill Murray in the movies Ghostbusters and Ghostbusters II.

## History

In 1998, John Bandhauer was in charge of creating the Netscape 4x JavaScript debugger. In order to keep things modular, he began by creating a mid-level JavaScript debugging application programming interface (API) known as js/jsd. This API augments the existing JavaScript API, providing clients with a useful set of debugger functionality implemented in C. John then went on to reflect that API into Java, and create his cross-platform front end, eventually producing Netscape JavaScript Debugger 1.0 and 1.1.

In April 2001, Robert Ginda began work on a JavaScript debugger for Mozilla, called Venkman. Venkman builds on the js/jsd portion of John's 1998 work, exposing it as an XPCOM component. This allows the user interface be written in JavaScript and XUL, which allows Venkman to be fully cross platform.

## External links

- Venkman Homepage<sup>[1]</sup>
- Venkman Developer homepage<sup>[2]</sup>
- Venkman for Firefox 1.5<sup>[3]</sup>
- Learning the JavaScript debugger Venkman<sup>[4]</sup>

## References

- [1] <http://www.mozilla.org/projects/venkman/>
- [2] <https://developer.mozilla.org/en/docs/Venkman>
- [3] <http://getahead.ltd.uk/dwr/ajax/venkman/>
- [4] [http://www.svendtofte.com/code/learning\\_venkman/](http://www.svendtofte.com/code/learning_venkman/)
-

# XMLHttpRequest

---

HTTP
Persistence · Compression · HTTP Secure
Header field
ETag · Cookie · Referrer · Location
Status codes
301 Moved permanently
302 Found
303 See Other
403 Forbidden
404 Not Found

**XMLHttpRequest (XHR)** is an API available in web browser scripting languages such as JavaScript. It is used to send HTTP or HTTPS requests directly to a web server and load the server response data directly back into the script.<sup>[1]</sup> The data might be received from the server as XML text<sup>[2]</sup> or as plain text.<sup>[3]</sup> Data from the response can be used directly to alter the DOM of the currently active document in the browser window without loading a new web page document. The response data can also be evaluated by the client-side scripting. For example, if it was formatted as JSON by the web server, it can easily be converted into a client-side data object for further use.

XMLHttpRequest has an important role in the Ajax web development technique. It is currently used by many websites to implement responsive and dynamic web applications. Examples of these web applications include Gmail, Google Maps, Facebook, and many others.

## History and support

The concept behind the *XMLHttpRequest* object was originally created by the developers of Outlook Web Access for Microsoft Exchange Server 2000.<sup>[4]</sup> An interface called *IXMLHTTPRequest* was developed and implemented into the second version of the MSXML library using this concept.<sup>[4]</sup> <sup>[5]</sup> The second version of the MSXML library was shipped with Internet Explorer 5.0 in March 1999, allowing access, via ActiveX, to the *IXMLHTTPRequest* interface using the **XMLHTTP** wrapper of the MSXML library.<sup>[6]</sup>

The Mozilla Foundation developed and implemented an interface called *nsIXMLHttpRequest* into the Gecko layout engine. This interface was modelled to work as closely to Microsoft's *IXMLHTTPRequest* interface as possible.<sup>[7]</sup> <sup>[8]</sup> Mozilla created a wrapper to use this interface through a JavaScript object which they called **XMLHttpRequest**.<sup>[9]</sup> The *XMLHttpRequest* object was accessible as early as Gecko version 0.6 released on December 6 of 2000,<sup>[10]</sup> <sup>[11]</sup> but it was not completely functional until as late as version 1.0 of Gecko released on June 5, 2002.<sup>[10]</sup> <sup>[11]</sup> The *XMLHttpRequest* object became a de facto standard amongst other major user agents, implemented in Safari 1.2 released in February 2004,<sup>[12]</sup> Konqueror, Opera 8.0 released in April 2005,<sup>[13]</sup> and iCab 3.0b352 released in September 2005.<sup>[14]</sup>

The World Wide Web Consortium published a *Working Draft* specification for the *XMLHttpRequest* object on April 5, 2006, edited by Anne van Kesteren of Opera Software and Dean Jackson of W3C.<sup>[15]</sup> Its goal is "to document a minimum set of interoperable features based on existing implementations, allowing Web developers to use these features without platform-specific code." The last revision to the XMLHttpRequest object specification was on November 19 of 2009, being a last call working draft.<sup>[16]</sup> <sup>[17]</sup>



Microsoft added the *XMLHttpRequest* object identifier to its scripting languages in Internet Explorer 7.0 released in October 2006.<sup>[6]</sup>

With the advent of cross-browser JavaScript libraries such as jQuery and the Prototype JavaScript Framework, developers can invoke XMLHttpRequest functionality without coding directly to the API. Prototype provides an asynchronous requester object called Ajax.Request that wraps the browser's underlying implementation and provides access to it.<sup>[18]</sup> jQuery objects represent or wrap elements from the current client-side DOM. They all have a .load() method that takes a URI parameter and makes an XMLHttpRequest to that URI, then by default places any returned HTML into the HTML element represented by the jQuery object.<sup>[19] [20]</sup>

The W3C has since published another *Working Draft* specification for the *XMLHttpRequest* object, "XMLHttpRequest Level 2", on February 25 of 2008.<sup>[21]</sup> Level 2 consists of extended functionality to the **XMLHttpRequest** object, including, but not currently limited to, progress events, support for cross-site requests, and the handling of byte streams. The latest revision of the XMLHttpRequest Level 2 specification is that of 20th August 2009, which is still a working draft.<sup>[22]</sup>

### Support in Internet Explorer versions 5, 5.5 and 6

Internet Explorer versions 5 and 6 did not define the XMLHttpRequest object identifier in their scripting languages as the XMLHttpRequest identifier itself was not standard at the time of their releases.<sup>[6]</sup> Backward compatibility can be achieved through object detection if the XMLHttpRequest identifier does not exist.

An example of how to instantiate an XMLHttpRequest object with support for Internet Explorer versions 5 and 6 using JScript method ActiveXObject is below.<sup>[23]</sup>

```
/*
  Provide the XMLHttpRequest constructor for IE 5.x-6.x:
  Other browsers (including IE 7.x-8.x) do not redefine
  XMLHttpRequest if it already exists.

  This example is based on findings at:

http://blogs.msdn.com/xmlteam/archive/2006/10/23/using-the-right-version-of-msxml-in-internet-explorer
*/
if (typeof XMLHttpRequest == "undefined")
  XMLHttpRequest = function () {
    try { return new ActiveXObject("Msxml2.XMLHTTP.6.0"); }
    catch (e) {}
    try { return new ActiveXObject("Msxml2.XMLHTTP.3.0"); }
    catch (e) {}
    try { return new ActiveXObject("Msxml2.XMLHTTP"); }
    catch (e) {}
    //Microsoft.XMLHTTP points to Msxml2.XMLHTTP.3.0 and is redundant
    throw new Error("This browser does not support XMLHttpRequest.");
  };
```

Web pages that use XMLHttpRequest or XMLHttpRequest can mitigate the current minor differences in the implementations either by encapsulating the XMLHttpRequest object in a JavaScript wrapper, or by using an existing framework that does so. In either case, the wrapper should detect the abilities of current implementation and work within its requirements.

## HTTP request

The following sections demonstrate how a request using the XMLHttpRequest object functions within a conforming user agent based on the W3C Working Draft. As the W3C standard for the XMLHttpRequest object is still a draft, user agents may not abide by all the functionings of the W3C definition and any of the following is subject to change. Extreme care should be taken into consideration when scripting with the XMLHttpRequest object across multiple user agents. This article will try to list the inconsistencies between the major user agents.

### The *open* method

The HTTP and HTTPS requests of the XMLHttpRequest object must be initialized through the **open** method. This method must be invoked prior to the actual sending of a request to validate and resolve the request method, URL, and URI user information to be used for the request. This method does not assure that the URL exists or the user information is correct. This method can accept up to five parameters, but requires only two, to initialize a request.

The first parameter of the method is a text string indicating the HTTP request method to use. The request methods that must be supported by a conforming user agent, defined by the W3C draft for the XMLHttpRequest object, are currently listed as the following.<sup>[24]</sup>

- GET (Supported by IE7+, Mozilla 1+)
- POST (Supported by IE7+, Mozilla 1+)
- HEAD (Supported by IE7+)
- PUT
- DELETE
- OPTIONS (Supported by IE7+)

However, request methods are not limited to the ones listed above. The W3C draft states that a browser may support additional request methods at their own discretion.

The second parameter of the method is another text string, this one indicating the URL of the HTTP request. The W3C recommends that browsers should raise an error and not allow the request of a URL with either a different port or *ihost* URI component from the current document.<sup>[25]</sup>

The third parameter, a boolean value indicating whether or not the request will be asynchronous, is not a required parameter by the W3C draft. The default value of this parameter should be assumed to be true by a W3C conforming user agent if it is not provided. An asynchronous request ("true") will not wait on a server response before continuing on with the execution of the current script. It will instead invoke the **onreadystatechange** event listener of the XMLHttpRequest object throughout the various stages of the request. A synchronous request ("false") however will block execution of the current script until the request has been completed, thus not invoking the **onreadystatechange** event listener.

The fourth and fifth parameters are the username and password, respectively. These parameters, or just the username, may be provided for authentication and authorization if required by the server for this request.

### The *setRequestHeader* method

Upon successful initialization of a request, the **setRequestHeader** method of the XMLHttpRequest object can be invoked to send HTTP headers with the request. The first parameter of this method is the text string name of the header. The second parameter is the text string value. This method must be invoked for each header that needs to be sent with the request. Any headers attached here will be removed the next time the *open* method is invoked in a W3C conforming user agent.

---

## The *send* method

To send an HTTP request, the **send** method of the XMLHttpRequest must be invoked. This method accepts a single parameter containing the content to be sent with the request. This parameter may be omitted if no content needs to be sent. The W3C draft states that this parameter may be any type available to the scripting language as long as it can be turned into a text string, with the exception of the DOM **document** object. If a user agent cannot stringify the parameter, then the parameter should be ignored.

If the parameter is a DOM *document* object, a user agent should assure the document is turned into well-formed XML using the encoding indicated by the *inputEncoding* property of the *document* object. If the **Content-Type** request header was not added through *setRequestHeader* yet, it should automatically be added by a conforming user agent as "application/xml; charset=*charset*," where *charset* is the encoding used to encode the document.

If the user agent is configured to use a proxy server, then the XMLHttpRequest object will modify the request appropriately so as to connect to the proxy instead of the origin server, and send Proxy-Authorization headers as configured.

## The *onreadystatechange* event listener

If the **open** method of the XMLHttpRequest object was invoked with the third parameter set to *true* for an asynchronous request, the **onreadystatechange** event listener will be automatically invoked for each of the following actions that change the **readyState** property of the XMLHttpRequest object.

- After the **open** method has been invoked successfully, the **readyState** property of the XMLHttpRequest object should be assigned a value of 1.
- After the **send** method has been invoked and the HTTP response headers have been received, the **readyState** property of the XMLHttpRequest object should be assigned a value of 2.
- Once the HTTP response content begins to load, the **readyState** property of the XMLHttpRequest object should be assigned a value of 3.
- Once the HTTP response content has finished loading, the **readyState** property of the XMLHttpRequest object should be assigned a value of 4.

The major user agents are inconsistent with the handling of the *onreadystatechange* event listener.

## The HTTP response

After a successful and completed call to the **send** method of the XMLHttpRequest, if the server response was valid XML and the **Content-Type** header sent by the server is understood by the user agent as an Internet media type for XML, the **responseXML** property of the XMLHttpRequest object will contain a DOM document object. Another property, **responseText** will contain the response of the server in plain text by a conforming user agent, regardless of whether or not it was understood as XML.

---

## See also

- Hypertext Transfer Protocol
- Representational State Transfer
- Ajax

## External links

- Level 1 specification of the XMLHttpRequest object from W3C <sup>[26]</sup>
- Level 2 specification of the XMLHttpRequest object from W3C <sup>[27]</sup>
- Specification of the XMLHttpRequest object for Apple developers <sup>[28]</sup>
- Specification of the XMLHttpRequest object for Microsoft developers <sup>[29]</sup>
- Specification of the XMLHttpRequest object for Mozilla developers <sup>[30]</sup>
- Specification of the XMLHttpRequest object for Opera developers <sup>[31]</sup>
- "Attacking AJAX Applications" <sup>[32]</sup>, a presentation given at the Black Hat security conference. Discusses several issues involving XHR and the future of cross-domain AJAX.

## References

- [1] "XMLHttpRequest object explained by the W3C Working Draft" (<http://www.w3.org/TR/XMLHttpRequest/>). W3.org. . Retrieved 2009-07-14.
- [2] "The responseXML attribute of the XMLHttpRequest object explained by the W3C Working Draft" (<http://www.w3.org/TR/XMLHttpRequest/#responsexml>). W3.org. . Retrieved 2009-07-14.
- [3] "The responseText attribute of the XMLHttpRequest object explained by the W3C Working Draft" (<http://www.w3.org/TR/XMLHttpRequest/#responsetext>). W3.org. . Retrieved 2009-07-14.
- [4] "Article on the history of XMLHttpRequest by an original developer" (<http://www.alexhopmann.com/xmlhttp.htm>). Alexhopmann.com. 2007-01-31. . Retrieved 2009-07-14.
- [5] "Specification of the XMLHttpRequest interface from the Microsoft Developer Network" ([http://msdn.microsoft.com/en-us/library/ms759148\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms759148(VS.85).aspx)). Msdn.microsoft.com. . Retrieved 2009-07-14.
- [6] Dutta, Sunava (2006-01-23). "Native XMLHttpRequest object" (<http://blogs.msdn.com/ie/archive/2006/01/23/516393.aspx>). IEBlog. Microsoft. . Retrieved 2006-11-30.
- [7] "Specification of the nsIXMLHttpRequest interface from the Mozilla Developer Center" (<https://developer.mozilla.org/en/nsIXMLHttpRequest>). Developer.mozilla.org. 2008-05-16. . Retrieved 2009-07-14.
- [8] "Specification of the nsIJSXMLHttpRequest interface from the Mozilla Developer Center" (<https://developer.mozilla.org/en/NsIJSXMLHttpRequest>). Developer.mozilla.org. 2009-05-03. . Retrieved 2009-07-14.
- [9] "Specification of the XMLHttpRequest object from the Mozilla Developer Center" (<https://developer.mozilla.org/en/XMLHttpRequest>). Developer.mozilla.org. 2009-05-03. . Retrieved 2009-07-14.
- [10] "Version history for the Mozilla Application Suite" (<http://www.mozilla.org/releases/history.html>). Mozilla.org. . Retrieved 2009-07-14.
- [11] "Downloadable, archived releases for the Mozilla browser" (<http://www-archive.mozilla.org/releases/>). Archive.mozilla.org. . Retrieved 2009-07-14.
- [12] "Archived news from Mozillazine stating the release date of Safari 1.2" ([http://weblogs.mozillazine.org/hyatt/archives/2004\\_02.html](http://weblogs.mozillazine.org/hyatt/archives/2004_02.html)). Weblogs.mozillazine.org. . Retrieved 2009-07-14.
- [13] "Press release stating the release date of Opera 8.0 from the Opera website" (<http://www.opera.com/press/releases/2005/06/16/>). Opera.com. 2005-04-19. . Retrieved 2009-07-14.
- [14] Soft-Info.org. "Detailed browser information stating the release date of iCab 3.0b352 from" (<http://www.soft-info.org/browsers/icab-10109.html>). Soft-Info.com. . Retrieved 2009-07-14.
- [15] "Specification of the XMLHttpRequest object from the Level 1 W3C Working Draft released on April 5th, 2006" (<http://www.w3.org/TR/2006/WD-XMLHttpRequest-20060405/>). W3.org. . Retrieved 2009-07-14.
- [16] "XMLHttpRequest W3C Working Draft 19 November 2009" (<http://www.w3.org/TR/2009/WD-XMLHttpRequest-20091119/>). W3.org. . Retrieved 2009-12-17.
- [17] "W3C Process Document, Section 7.4.2 Last Call Announcement" (<http://www.w3.org/2005/10/Process-20051014/tr#last-call>). W3.org. . Retrieved 2009-12-17.
- [18] Porteneuve, Christophe (2007). "9". in Daniel H Steinberg. Raleigh, North Carolina: Pragmatic Bookshelf. pp. 183. ISBN 1-934356-01-8.
- [19] Chaffer, Jonathan; Karl Swedberg (2007). *Learning jQuery*. Birmingham: Packt Publishing. pp. 107. ISBN 978-1-847192-50-9.
- [20] Chaffer, Jonathan; Karl Swedberg (2007). *jQuery Reference Guide*. Birmingham: Packt Publishing. pp. 156. ISBN 978-1-847193-81-0.

- [21] "Specification of the XMLHttpRequest object from the Level 2 W3C Working Draft released on February 25th, 2008" (<http://www.w3.org/TR/2008/WD-XMLHttpRequest2-20080225/>). W3.org. . Retrieved 2009-07-14.
  - [22] "XMLHttpRequest Level 2, W3C Working Draft 20 August 2009" (<http://www.w3.org/TR/XMLHttpRequest2/>). W3.org. . Retrieved 2010-04-08.
  - [23] "Ajax Reference (XMLHttpRequest object)" (<http://www.javascriptkit.com/jsref/ajax.shtml>). JavaScript Kit. 2008-07-22. . Retrieved 2009-07-14.
  - [24] "Dependencies of the XMLHttpRequest object explained by the W3C Working Draft" (<http://www.w3.org/TR/XMLHttpRequest/#dependencies>). W3.org. . Retrieved 2009-07-14.
  - [25] "The "open" method of the XMLHttpRequest object explained by the W3C Working Draft" (<http://www.w3.org/TR/XMLHttpRequest/#the-open-method>). W3.org. . Retrieved 2009-10-13.
  - [26] <http://www.w3.org/TR/XMLHttpRequest/>
  - [27] <http://www.w3.org/TR/XMLHttpRequest2/>
  - [28] <http://developer.apple.com/internet/webcontent/xmlhttpreq.html>
  - [29] [http://msdn.microsoft.com/en-us/library/ms535874\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms535874(VS.85).aspx)
  - [30] <https://developer.mozilla.org/en/XMLHttpRequest>
  - [31] <http://www.opera.com/docs/specs/opera9/xhr/>
  - [32] [http://www.isecpartners.com/files/iSEC-Attacking\\_AJAX\\_Applications.BH2006.pdf](http://www.isecpartners.com/files/iSEC-Attacking_AJAX_Applications.BH2006.pdf)
-

# Article Sources and Contributors

**JavaScript** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375564333> *Contributors:* 12 Noon, 16@r, 31stCenturyMatt, 80N, 9cep22202@sneakemail.com, A bit iffy, ABCD, ABF, ANGGUN, Apo Latineen, Aarabia, Acostin, Adam Hauner, AdmN, Aeons, Afrobdudy, AgentCDE, Ahda, Ahoerstemeier, AhY1, Aka042, Akronym, Alansohn, AlastairIrvine, AlistairMcMillan, Aman2008, AmitDeshwar, Ancheta Wis, Andre Engels, Andreas S., Andrew Levine, AndrewHowse, Andrewrost3241981, AndyZ, Anfernyjohnsun, Ankles, Anna512, AnnaFrance, Antimatter15, Antipastor, Antonio Lopez, Aoi, AquaGeneral, Ardonik, Artw, Arvindn, Asqueella, Astrobloby, Avijitguhary, AxelBoldt, B0at, BP7865, Bamkin, Beao, Belgarat, Ben-Zin, BenAveling, Bencherliffe, BernhardBauer, Betax, Bevo, BinaryWeapon, Bionicseraph, BirdieGalyan, Blobglob, Blonkm, Bobdc, Bogtha, Boing! said Zebedee, Bonadea, Boomshadow, Bpeps, Brainyiscool, Bratsche, Brendan Eich, Brettz9, Brian Kendig, Brighterorange, Brilliant trees, Brion VIBBER, Bryan Derksen, Bwickery, CForrester, CO, CQJ, Caltas, Calvin 1998, CambridgeBayWeather, CanadianLinuxUser, CanisRufus, Canonical Rush, Cap'n Refsmaat, Capricorn42, Carewolf, Carey Evans, CaseyPenk, Cash cab, Cat-five, Cenarium, Centrx, Chairboy, Charles Gaudette, Charles Iliya Krempaux, Charles2345, CharlotteWebb, Chazwatson, Chealer, Choronzon111, Chris the speller, Chrisdolan, Chrisman247, Chrishwiki, Chuq, CiudadadGlobal, Civilis, Clidui, Closedmouth, Cisdennis2007, Cmdrjameson, Coffee, ColinHelvensteijn, ColinJF, Collabi, Combustablejo, Corti, Cpu111, Crazycomputers, Cronium, Crystallina, Cst17, Cwllm2, Cybercobra, Cyfal, D.brodale, DO'Neil, DTR, Damian Yerrick, Damicatz, Danakil, Daniel.Cardenas, Dantheman531, Darkride, Dasch, DaveK@BTC, Daveydwieb, David Gerard, David Wahler, DavidCary, Davigoli, Dbabbitt, Dcoetzee, Delcnsltmd, Delfuego, Den fjättrade ankan, DennisWithem, Deor, Desiblast, Dhtml, DhtmlKitchen, Digita, Dicespinstier, Dlecx, Dodecagon12, DonToto, Donbert, Dragon Dave, Dreadstar, DreamGuy, Dreflymac, Drsqurizl, Dycedarg, Dysprosia, Eagleale, Earle Martin, Echo95, Ed g2s, EdC, EdJohnston, Edfrommars, Edgar181, Edknol, Edward Z. Yang, Elefectoborde, Eliashedberg, Eliot1785, Ellmist, Emperorbma, Engelec, Enkauston, Ennajm, Epochbb, Er Komandante, Erich gasboy, Erkan, Eskimospy, EvanProdromou, Everyking, Ex-dude329.4, Excirial, Execvator, Explicit, Extremecircuizt, Faced, Faisalakeel, Familygyu0108, FatalError, Fchoong, Femmina, Fetchcomms, Fistsboy, Flager, Flame200, Floreence4eva, Frau Holle, Frecklefoot, Fred Bradstadt, Fredrik, Func, Furrykef, Fwerdna, GTBacchus, Gail, Galwhaa, Gamaliel, Gary King, Geary, Geneb1955, Geoffrey, Georgeryp, Gerbrant, Gerweck, Ghettohlaster, Giftlite, Gogo Dodo, Goplat, GraemeL, Graham87, Grayczyk, Greenie2600, Greenminz, GregorB, Griba2010, Grika, GroovySandwich, Guaka, Guiltardemon666, Gunuag, Gurchzilla, Gwernol, HDrake, HairY Dude, Hannes Hirzel, Hbackman, Hblank, Herorev, Hervegirod, Hgmichna, Hirzel, Hmains, Hongguo, Hu, Hu12, HundoUno, HyperCapitalist, I Mystic I, I already forgot, IRP, Ian Bailey, Ida Shaw, Idiottromia, Igoldste, Illoeobnbu, IlliterateSage, ImperatorExercitus, Inseeisyou, Insouciance, Intgr, InvertRect, Indescent, Irish Souffle, Irlbloss, Itpastorn, IvanLanin, J.delanoy, JFG, JForget, JVz, JackyBrown, Jacobolus, Jagginess, Jake Nelson, Jake Wartenberg, James.abley, Jan1nad, JavaKid, Jay, Jec, Jeresig, Jerryobject, Jlemer, Jmartinson, Jobanjohn, Jodi.a.schneider, Joffeloff, JokaMan, JonathanAquino, Joncunn, Johnanson, Jon, Jpgordon, Juliancolton, Jumbuck, Justie1220, Jwadeo, Kenny TM--, Kenyon, Kesac, KeybladeSephi, Kiensvay, Killab14, King of Hearts, Kingpin13, Kinu, KnowledgeOfSelf, Koavf, Kroum, Kungming2, L Kensington, Lacqui, Lanxiashi, Largoplazo, LedgendGamer, Ledgerbobb, LeeU, Lego614, Lemon octopus, Leotohill, Lesgles, Lethe, Lhtown, Lianmei, Linuxerist, Lijiang, Lukasblakk, Lupin, MER-C, MFNickster, MIT Trekkie, MVelliste, Mabdul, Macdall, Magioladitis, Magister Mathematicae, Maian, Marc-André Aßbrog, MarekPro, Mariano.viola, Mark Renier, Marklbarrett, Martarius, Martin.kopta, Martinhenz, MatheoDJ, Matifibrahim, Matijap, Maury Markowitz, Maximus Rex, Mazin07, Mbarone911, McDutchie, McSly, McClurmc, Mckoss, Message From Xenu, Mfc, MichaelBillington, Miernik, Mifter, Mikehtf, Mikenolte, Mikething, Milefool, MilesAgain, Milly, Mindmatrix, Minesweeper, Minghong, Miranda, Mjb, Mkwiese, Mnemeson, Mnemonicof, Mob590, ModJesus, Moeron, Moonyfruit, Mortense, Mouchoir le Souris, Mr. Wheely Guy, Mr.Clown, Mr.Ollie, Mrowigg1es, Ms2ger, Muntfish, Muzzamo, Mxn, Mypallmall, Müitze, NYKevin, Nako16, Nanshu, Nashleyj, Nate Silva, Nate879, Nealcardwell, Nearfar, Neile, Neilshermer, NeonMerlin, Neonmario, Nezticle, Nick8325, NickW557, Nifky?, Nigeli, Nivix, Nixeagle, Njuuton, Nk, Nlu, No Guru, Nol888, OLENDJONE123, Odaravlaac, Onhoitsjamie, Oleg Alexandrov, Oli Filth, Oliphant, Oliver Pereira, Oneiros, Opticyclic, OrangeDog, Orderud, Orosio, Oxyromon83, OzFerd, Pacpercebal, Paradox7798, Patrick, Patrick Corcoran, Paul.Irish, Paul1337, Peak, Peter L, Peter S., Phantombantam, Pharaoh of the Wizards, PhilHibbs, Philcha, PhilpR, Piano non troppo, Piet Delpoit, Piotrus, Plest, PlusMinus, PointedEars, Poor Yorick, Prakshal, Presto8, Prince-of-Life, Prolog, Psalmtrist cheifprosperity, Psychcf, Pyrospiriti, Qatter, Quamaretto, Quantum00, QueenCake, Qutezuze, Qvwx, R3m0t, RG, RJaguar3, Radagast83, Ramesh Chandra, RandomStringOfCharacters, Razoxr, Rbirky, Rbucci, Rdsmith4, Recnligarc, RedRollerskate, RedWolf, Reinthal, RexNL, Rgodemote, Rgldl, Rhobite, Rich Farmbrough, Rich Janis, Rick Block, Ringbang, Rjwllmsi, Robbie098, Roberto Cruz, Robmw, Rohitjs, Rror, Rufous, Rufustiffley, Ruud Koot, Ryan Postlethwaite, S-n-ushakov, SJP, ST47, Salarmehr, Sam Hovevar, Sam Korn, Samfosteriam, Samliew, Samuel, Samwr1234, Sango123, Sayer55512, ScalarField, Sceek, Scuirinæ, Scark23, Sealican, Seav, Selket, Shalom Yechiel, Shanel, SharkD, Shenme, Shwaza, SineQ, Singhivender, SirLewk, Sjorman, Skizzik, SkyWalker, Slamb, Sleepyhead81, SlowByte, Smyth, Snownolf, Snoyes, SomeStranger, Someguyonthestreet, Spalding, SparrowsWing, Spe88, SpeedyGonsales, Spellmaster, Spiel, Spitfire, SpuriousQ, StaticGull, SteveSims, Steveholt, Stickguy, Stuffandthings, Super Rad!, Superm401, Superslacker87, Surfeited, Susan Davis, SusanLesch, TFOWR, TOReilly, Tabrez, TakuyaMurata, Tarmok, Tasc, Taw, Technome, Tedickey, Template namespace initialisation script, Thatguyflint, The MoUsY spell-checker, The Nut, The Parting Glass, The Rambling Man, The Thing That Should Not Be, The Wild Falcon, TheKMan, TheMuffinWalkers, Thefuzzballster, Thekaleb, TheMASTERed, Theone256, Thingg. Think outside the box, ThirdeSide, Thron7, Thumperward, Tigerwolf753, Timwi, Tizio, To Fight a Vandal, Tobias Bergemann, Todd Vierling, Toddc, Tokel, Tomjenkins52, Tophu, Toussaint, Tpk5010, Trigger hurt, Troels Arvin, Trustle, Tsemii, Tuntable, Tyomitch, Typhoonhurricane, Tysto, UU, Udittmer, Ultramandk, Uncle G, Ueights, Uetoursch, Versageek, Violetriga, Vocaro, VoluntarySlave, Vriullop, W00zy5No03r, Wadsworth, Waldir, Wapcaplet, Wasell, Wayward, Website-andrew, Weirdo134, Wellithy, Werner, Who, Why Not A Duck, WikiLaurent, Wikibob, Wikieat, Wikieditorftoday, Wisdom89, Wzww, XP1, Xedret, Xhienne, Yamakiri, Yamakiri on Firefox, Yaron K., Ybissop, Ysangkok, Yuki Konno, Yuval Madar, Zaak, ZakuSage, Zigger, Zin, ZmLa, Zoicon5, Zojji, Zootm, Zundark, Zven, Zvn, Zzuuzz, 1296 anonymous edits

**JavaScript syntax** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375162644> *Contributors:* (jarbarf), Aaaidan, Acasson, Albrecht andrzejewski, AnAj, Ancheta Wis, Arthur Rubin, Betax, Buzzard2501, Cactus.man, Chutzpan, Clayhalliwell, Clidui, Cmcfarlad, Comocomocomocomo, Crystallina, David-Sarah Hopwood, Davidstr, Dipset1991, DonToto, Dougluce, Dreflymac, Drknkn, Dto, Enkidu1947, Erich gasboy, Exert, Explinator, Extremecircuizt, Flash200, FrankSier, Fred Bradstadt, Furrykef, Gabrielsroka, GavinSharp, Gerbrant, Grahamzibar, Grshiplett, Imshardy, Inimino, Insouciance, Int9h, J.delanoy, JLaTondre, Jeffz1, Jeremysborne, Jeshan, Jessmerimran, Jiri Svoboda, John Vandenberg, Jorge Stolfi, KMeyer, Kgykipedican, Khaledziyaeeen, Kickboy, Kohtala, Kusunose, Lanxiashi, Lingwitt, Lucian900, Luna Santin, LuoShengli, Lycurgus, Machine Elf 1735, Maian, Matjball, Mikeblas, Mild Bill Hiccup, Misortie, Mmalessa, Nate879, Nigeli, OverfordQ, Piano non troppo, PleaseStand, Quamaretto, Quolikos, RIdaworth, Reazal, RobG-bne, Rockower, Rsjaffe, Rufous, Rursus, Ruud Koot, Sesembeki, Shanes, Sleeper220, Takarada, ThomasStrohmann, Thumperward, Tobias Bergemann, Tony Sidway, Triul, Tryforceful, Tuntable, Viebel, Voyagerfan5761, Wackywave, Walden, Wknight94, Woggae2, X42bn6, Ynhockey, Youngoat, Zzedar, 269 anonymous edits

**JavaScript Style Sheets** *Source:* <http://en.wikipedia.org/w/index.php?oldid=369732321> *Contributors:* Cybercobra, Den fjättrade ankan, Dlohrer2003, Func, Gudeldar, Hhelscher, Mabdul, Minghong, Quale, Ringbang, Robinh, Samboy, Volker E., 15 anonymous edits

**JavaScript engine** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374628624> *Contributors:* ARUNKUMAR P.R, Aavindraa, CRGreathouse, Chrismiceli, Digita, DonToto, Fantasticfears, Fchoong, FedericoMP, Foolishgrunt, Gott wisst, Ikawe.saeem, Jbshopp, Jeffw57, JonHarder, Jondel, Kesal, Mabdul, Minghong, Oddity-, Radagast83, Rehmt, William.Aleman, 28 anonymous edits

**Ajax (programming)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374493911> *Contributors:* 16@r, A brisson, ARC Gritt, Aashish950, Abelson, AbsolutDan, Abulfazl, Academic Challenger, Acroterion, Adashiel, AdrianLozano, Af1218, Ahoerstemeier, Ajaxbee, Ajaxdevil, Ajaxtoday, Ajdecandis, Akadruid, Akhristov, Akronymn, Alaa.moustafa, Alansohn, Albanaco, Alcalazar, Alefn1, Aleksandar Šušnjar, Alekseenkokirill, Aleph-4, AlexMorozov, Alexa Foster, AlexaW, Alexei.white, Alexhop, Alhuth, AlistairMcMillan, Amalthea, AmDaniel, Amniarix, Amux, Anakin101, Anamanfan, Antichist42, Anapolpanom, Ancheta Wis, Anderiv, Andreas Kaufmann, Andrewkitchen, Angelpeream, Anguis, AniRaptor2001, AnnaFinotera, Anthonyfrey, AntiVan, Api, Apoltix, Archer3, Archfahwyl, Arminbachmann, Artw, Arvindn, ArwinJ, Asharism, Ashmoo, Athenasoft, Atif.mod, Atmos, Aude, AvengeX, B0at, BBilge, BCube, BRPXQZME, BTLizard, BankingBum, Barte, Bash, Bbatsell, Bbefty, Beetstra, Beland, Ben kenobi 00, Ben-oni, Ben@atmail.nl, Benawhite, Beradrian, Betamod, Bevo, Bewildebeast, Bgrayless, Bgupta55, Bhadani, Bigtrick, Bill37212, Billfromhk, BillyTFried, Bipin Jethwani, Bjhecht, BlanchardDiogenes, Blonkm, Bmiccom, Bobet, Bobo192, Bogtha, Bongwarung, Booch, Boos, Borkwe, Bornhj, Boulevardier, Bovineone, Brat32, Brclancy111, Breno, Brentashley, Brest, Brian Gunderson, Brightc, Bukharin, C++ Template, CKlunck, CPWinter, CQJ, Cabiria, Cactus.man, Cahtye@cruzio.com, Cakeman, Caldini, Camelltrader, Campustr, Camw, Can't sleep, clown will eat me, Canadian-Bacon, Canderson7, CapitalR, Capricorn42, Cascadiadude@gmail.com, Catrope, Chbuckley, Cdc, Cedars, Ceejayoz, Ceiton, Cfinazzo, Clust, Charles.kendrick, CharlesClarkson, CharlotteWebb, Chase me ladies, I'm the Cavalry, Chatmepres, Chbarts, Chelsel, Chineeab, Chris Kuehl, Chris the speller, ChrisHibbert, Clamum, Closedmouth, CodeAdams, Coder cotton, Color probe, Computerjo, Cookiecrook, Coolbeka, Courcelles, Crazycomputers, Crimsomphire, Crowdes, Csega, Ckteen, Cyberhitesh, Cyjpat, Cynical, CzarB, D'Artagnol, D.taveira, DHN, DMacks, DRogers, Damian Yerrick, Damicatz, DanMS, Dancter, Dandv, Danic, Danlevine, Danyymo, Danyuyou, Dar-Ape, DarkFalls, Darth Panda, Darwin226, Dasch, Dave Cohoe, David costanzo, DavidCary, Davidhorman, Davipo, Dawnseeker2000, Darcov, Dcylfer, DeadEyeArrow, Deathanatos, Debackeri, Debresser, Decrease789, Dedrick, Delfuego, Demi, DepartedUser2, Derivator, Destynova, Dhartung, Digitxpsp3, Dixtra, Dina, Diophantus, Dirkbike, Disavian, Discospinster, Dispenser, DKreatto, DmitryKotorov, Dmwtechnologies, DoSiDo, DocWatson42, Donald Albury, Donama, Doug Bell, Doubertram, Dougofborg, Dr. Zaret, Draicone, Drbreznjev, Dream out loud, Dreflymac, Dlugosz, E.au, Eagleale, Ebruchez, Ed Fitzgerald, Ed g2s, Edward, Edward301, Eenu, Egil, Eliz81, Eloi.sannarint, Eman502, Emre D., Encephalon, Enchanter, Ento, Epb123, Eric B. and Rakim, Elicross, EstebanF, Euryalus, EvanProdromou, Evil Monkey, EvocativeNtrigue, Execvator, Fadookie, Faelomx, Falling Cloud, FatalError, Faustnh, Fdp, Feezo, Felix.rivas, Fenin, Fennec, Fergalbrean, Ferkelparade, FRe, Filemon, FireWeed, Firstauthor, Fmccown, Fosnez, Fram, Framamax, Frangibility, Freakofnuture, Frecklefoot, Fred Bradstadt, Fred J, Fredrik, Frenchgib, Funnibunny, Furrygeek, Furrykef, Fusiondeveloper, Fuzzy510, GCarty, Galactor213, Galwhaa, Gamol, Garrett Albright, Gary King, Gatekiller, Gblaz, Gham367, Glane23, Glen, Gmauruthi, Gmcoley, Gmd588, Gnome of Fury, Gobik, Golbez, Gpatnude, GraemeL, Graham87, Greenie2600, Greggobridges, GreyCat, Guilherme Blanco, Gurch, Gurubrahma, Gustavb, GuyBehindtheGuy, Guyjohnston, Gwernol, Gypsodyctor, H3h, Hapab, Hallandnash, Hanberke, Harborsparrow, Harej, Harmil, Harshadoah, Hede2000, Heezy, Hegh, Henricks, Henrikb4, Hervegirod, Hgmichna, Hoolooovoo, Hoosierplew, Hu12, HumbertoDiogenes, Huwleslie, HybridBoy, Hypnoticcyst, IByte, ICTlogist, IGod, Ian Moody, Ianneub, Idearat, Imsoclver, Insanephantom, Insanity Incarnate, Intgr, Irish Souffle, Imavash, Ironfistofanarchy, Isnow, IvanLanin, Ixf64, J\$, J.delanoy, JFreeman, JLaTondre, JVz, Jabberwoch, Jacecole, Jacobolus, Jahleldaruis, Jainvineet, James Skarzinskas, Jancikotuc, Jarchitect, Jareha, Jasongaylord, Jaspsoft, Jenkins, Jay Gatsby, Jcw69, Jdthod, Jebba, Jeet020, Jeffmcfarland, Jeffg, Jeffrey O. Gustafson, Jemplymethoud, Jepc, Jessidicple, Jisset77, Jesusjonez, Jevon, Jibbegod, JimD, JimR, Jkln, Jlibeezer, Jmabel, Jmlk17, JoDiamonds, JoanneB, Jobanjohn, JoeWalker, Johayek, John Mark Williams, John Seward, JohnManuel, Johndci, Johndrinkwater, Jonathanrcxhead, Jonovision, Josenaves, Josh-Levin@ieee.org, Joshf, Jossi, Jouk pleiter, Jourmeyman, Jppaul, Jriffel, Julyscripser, Junky, Jupition, Justavo, Jutiphon, K.le, KHaskell, KJK::Hyperion, KTyson, Kafziel, Kamal006, Karlthegreat, Kartano, Katieh5584, Kaylanimis, Kel-nage, Kelly Martin, Kelner, Kenyon,

Keppx0r, Kevin, Kevin Baas, KevinLocker, Khaderv, Khadlock, Khakbaz, Khakman, Khalid hassani, KiaThr, Kiand, Killiondude, Kingpin13, Kirils, Kithplana, KjD, Kks krishna, Kku, Klimpong, Klutzy, Kmcocoy, Knut, Knutux, Komperre, Korg, Kovyrin, Kozuch, Kwsn, L Kensington, Laksono, Largoplazo, Latrippi, Leaf of Silver, Leafyplant, Leandro Cardoso, LeeU, Leif, LeonardoGregoriani, Lianmei, Lightdarkness, Lightmouse, Lindsay-mclennan, Ling.Nut, LinguistAtLarge, Lishugo, Logain2006, Loren.wilton, Luk, Luna Santin, LunaticBeatnik, M. B., Jr., M4rk, MC10, MER-C, MK8, MKoltnow, Mac, Macaldo, Mahemoff, Majid khonji, MajykHands, Makzy, Mandarax, Manop, Maoj-jb, Marc.GZR, MarkSweep, Markidj, Markmarucot, Marktmilligan, Marky1124, Marty Pauley, Master Of Ninja, Mattd, Matthewpun, Mauri1980, Maxmaxb, Mboverload, Mdchachi, Mehraj, MeirM, Melah Hashamaim, Mendaliv, Mg cristi, Michael Stone, MichaelBillington, MichaelMaggs, MightyWarrior, Mikebals, Milefool, Milnivri, Mindmatrix, Minghong, Mini-Geek, Minimac, Minimac's Clone, Mistsrider, Mixx941, Mknouse, Mimos, Mohanrl, Monotonehell, Moonside, Moralis, Moreschi, Mr Minchin, Mr buick, Mr datawolf, MrCalifornia, MrMoran, MrOllie, Mberryman, Mschel, Muthuveerappan, Mxl, Myanw, Mysid, NHRHS2010, NSR, Nakakapagabagabag, Nakkeeran, Nako16, NathanBeach, NawlinWiki, Neilrick, Nescio, NiceGuyAlberto, Nickmjohnson, Nickmurdoch, Nigeli, Nihonjoe, Nitinshah23, Nkatsaras, Norm mit, Notinasnoid, Nova77, Oazabir, Oben, Oblivious, Odinjobs, Ohnoitsjamie, Olego, Oli Filth, Omicronpersei8, One Random Fellow, Onkarshinde, Opelio, Orchid Righteous, Orioane, Ornil, Orrc, Osbus, OwenBlacker, OwenX, Oxymeron83, PB0305, Palfrey, Pathoschild, Patrick, Paul August, PaulHoadley, Pavel Vozenilek, Payrard, Pctopp, Pegasus1138, Pentapenguin, Perfecto, Peter Delmonte, Peter McGinley, Petr.adamek, Pgan002, Pkg, Pharos, Philip Trueman, PhilipO, Phuntism, Pickatutorial, Pieldesomere, Pikiwyn, Pilotguy, Pimlotc, Pingveno, Pioverto, Pip2andahalf, Pjdonnely, Pkchan, Pkrecker, Plyd, Pmsyzz, Pne, PointedEars, Porqin, Praveen bv, PrimeCupEevee, Printer222, PrometheeFeu, PseudoSudo, Pwt-wsu-mg, Pyrowolf, Qebafhzn, QubitOtaku, Quendus, Quill18, Quilokos, Quinsareth, Qviri, Qwayzer, Qxz, RHJesusFreak40, RMHED, Raano, Rabidpoobear, Radiier, Ramu50, RandalSchwartz, Raven4x4x, Rawling, Rayngwf, RazorICE, Rcronk, Reallyjoel, Red Director, RedWolf, Redvers, Reedbeta, Reinthal, Reisio, Remember the dot, Renmiri, Renwique, RexNL, Ricardo, Rgarcia09, Rhobite, RicardoAmador, Rich Janis, Rick Block, RidinHood25, Riki, Rklawton, Rob cowie, RobGonda, Robert Stephen Spiegel, Robferrer, Robmanson, Robomason, Robomaeyhem, Rocastelo, Rohijs, Roleplayer, Roman à clef, Ronchristie, Ronz, Rookstar, Rosdec, Rosswelton, Roux, Royboycrashfan, Rpwiltzek, Rpenner, Rrjanbiah, Rs564, Rsocol, Rubencepeda, Rufous, Ruud Koot, Rydel, Ryulong, SCPearson, SGBailey, Safety Cap, Salilkaul, SallyDawg, Sam Korn, SamJohnston, Samdeskin, Samyem, SanDiegoPolitico, Sarefo, Sathaeeseelan, Saturday, Saxifrage, Sayden, Sciarinæ, SeanR, Seanhan, SeattleJazzMan, Seba5618, Seefeld, Seriouswikifan, Shadov1, Shanes, Sheiko, Shenme, Shii, Shinmawa, Shon, Shwaza, Simetrical, Simishag, SimonP, Singularity, Sintaku, Sir Vicious, Siroxo, Skeejay, Skizzik, Skomorokh, Slashme, Sleepnomore, Sleepyhead81, Smpdawg, Snaxe920, Snezy, SnowFire, Soeren1611, Softguyus, Softtest123, Sophrosune, Soumyasch, SpaceFlight89, Spankman, Spayrard, Spookfish, Sprocketonline, SpuriousQ, SqueakBox, Ssrgi, Sshoberi, Stadler, Stajler, Starwiz, Stefanostaus, Stephen B, Thumperward, Thundربولt, ThylekShran, Tibbetts2c, TigerShark, Tiggerjay, Tigrisek, TimmmCam, Timmeu22, Tippler, Tricky, Tohd8BohaiithuGh1, Tomjenkins52, Tomkarlo, Tomyeh, Tooto, Tovelng, Trails, TravisCross, Tregoweth, TreyHarris, Triona, Trogera, Tudorol, Tutable, TuvoK77, Typofixer76, Tzarius, UU, Uberdude85, Ugictox, UkPaolo, Umapathy, Uncle Miltly, Unclejedd, Uniwalk, Unschool, Unsuared, Vary, Velociostrich, Versageek, VictorAnyakin, Vigna, Virexmachina, Viridian, Voice of All, Voorhies, Vriullop, Vunutus, Wadems, Wafuzl, Waggars, Wagnernm14, Waltercruz, Warren, Wayward, Wengier, Weregberil, Westonmr, Weylinp, Wikiajax, Wikibofo, Wikilinus, Willemo, Williamlund, Wimt, Windharp, Winhunter, WiniVidiVici, Wisden17, Wm, Wnorris, Woogee, Woohookitty, Wpbasti, Wwwolf, XGraham, Xamian, Xelgen, Xephero, Xmnemonic, Xmskrat, Xpclient, Yamamoto Ichiro, Ycherkashin, Ymendel, Ynod, Yooden, Yoric, Zanejin, Zazou, Zealotgi, Zelchenko, Zr40, Zscout370, Zundark, Zvika, Zvn, ZWilson, Zzedar, Zzyzx11, 4141 anonymous edits

**AJAX.OOP** *Source:* <http://en.wikipedia.org/w/index.php?oldid=372278225> *Contributors:* Frap, lxf64d, JaGa, Kathleen.wright5, Meloman, Mikhus, Tabletop, Tedickey, Victuallers, 7 anonymous edits

**?**: *Source:* <http://en.wikipedia.org/w/index.php?oldid=355828431> *Contributors:* ABCD, Aaron Rotenberg, Alerante, Alf, Anakin101, Ancheta Wis, Anonymous Dissident, ArielGold, Btx40, BunsenH, Cap601, Capi, Carychan, Charles Matthews, Corps, Cybercobra, Dan Pelleg, Daniel.Cardenas, DavidHalton, Decltype, Dpbsmith, Dysprosia, Exe, Faya, Fieldday-sunday, Geary, Gigs, Greend, IAlax, Jesin, Jkl, Jrudderman, KingJason, Labalius, Marudubshinki, Merovingian, Minghong, Miranda, Moggie2002, Mrwojo, NathanBeach, Noodle snacks, Ordered, Phresnel, Qwertyus, Salvar, Sevcsik, Shlakblock, Simeon, Spoon!, Sverdrup, The Anome, The Thing That Should Not Be, The Wild Falcon, Thumperward, Tobias Bergemann, Torc2, WOT, Warren, Whiner01, Woohookitty, Xiong Chiamiov, Zanimum, ZeroOne, 74 anonymous edits

**Appcelerator Titanium** *Source:* <http://en.wikipedia.org/w/index.php?oldid=364166756> *Contributors:* Adamstac, CaribDigita, Derek farn, Freedman1, Jamesdlow, Sergius42, WikiLaurent, 5 anonymous edits

**AppengineJS** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375517384> *Contributors:* Bearcat, Gmosv, TFOWR, Wuhwuzdat

**BSON** *Source:* <http://en.wikipedia.org/w/index.php?oldid=366266973> *Contributors:* Beaddy1238, Cybercobra, Dm, Ilion2, Kwamikagami, Mdirolf, Peak, 1 anonymous edits

**Bookmarklet** *Source:* <http://en.wikipedia.org/w/index.php?oldid=368134165> *Contributors:* 16@r, ABCD, AbsolutDan, Alexamies, AlistairMcMillan, Amalthea, Amtree@gmail.com, Andre Engels, Andy Dingley, AoV2, Badanedwa, Berkut, Blue Em, Brighterorange, Brockert, Chochopk, CliffC, Cmglee, Crossmr, Cy21, Danlev, Davecohen17, Desummoner, Doron, Dreamyshade, Drj, Dysprosia, Diugosz, Edward, ElbridgeGerry, Elonka, Eloquence, Feralfeline, Foolswisdom, Gregorothfuss, Grstain, HaoZlian, Isaac Dupree, Jamesmorrison, Japanese Searobin, Jaxl, Jeroen, JesseW, Jm34harvey, Joshiesurber, Jousyl, Kindspirit, KnightRider, Kuda, Lexein, Life, Liberty, Property, LinguistAtLarge, Longhair, Luna Santin, MarcelB612, Mav, Mckoss, Mditto, Menchi, Mendaliv, Michael Hardy, Milly, Minghong, Moituous, Mpcalan, NeilN, Nextsco, Nick, Nikola Smolenski, ObfuscatPenguin, Ortolan88, Osias, Ottava Rima, Pascalsvanhecke, Peak, PleaseStand, Primary key, Proitexpert, Proofreader77, Purple Paint, Pxma, RadioActive, Raven Ryan, RedWolf, Reddi, Remember the dot, Rgf, Rich Farmbrough, SarekOfVulcan, Shadowjams, Shaydon, Steeve, Stephen Turner, That Guy, From That Show!, The Thing That Should Not Be, Thumperward, Tillwe, Timmywimmy, Tkrones, Toftari, Toner, TonyW, Tregoweth, Twaring, V0rt3x, Vishal.doshi, Vladislav Pogorelov, Waltpohl, WulfTheSaxon, XRDoDRX, Xyzzpplugh, Yoneh, Zigger, Zpb52, 161 anonymous edits

**Client-side JavaScript** *Source:* <http://en.wikipedia.org/w/index.php?oldid=365894854> *Contributors:* Aapo Laitinen, Bamkin, Blr21560, Cmdrjameson, David spector, Doekman, DonToto, EdJohnston, FatalError, Gabrielsroka, Gaius Cornelius, Glyphobet, Gpvos, Hut 8.5, Ian Bailey, Ian Burnet, Jacobolus, Korath, Lvr, Mabdul, Maian, Mdd4696, Minghong, Molotron, Nako16, Nogray, Patrick, PointedEars, Radagast83, Retired username, Rjwilmsi, Rmathews, Shanes, Stijn Vermeeren, Summit677, Swmcd, Tb, Technion, Tokek, Tomjenkins52, Triddle, Tutable, UU, Ultimatemadness, Versageek, Xedret, 75 anonymous edits

**CommonJS** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374021739> *Contributors:* Matt5091, Rabbittkillrun, Wikieditorftoday, 3 anonymous edits

**Comparison of JavaScript frameworks** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375347984> *Contributors:* Abel406, Ainze, Alanbly, Assassin of Joy, Aycan Gulez, B.moyet, Bensmoif, Blr21560, Brillyfresh, Cemitchellusa, Choas, Cool Matty, DOOOMKULTUS, Dandy, Davedx, DonToto, EVula, Emiraglia, FirefoxRocks, Flaming Grunt, Fred Bradstadt, Frédéric Delanoy, Glane23, Glyphobet, Greggheth, GregorB, Gudeldar, HexaChord, Ironholds, J.delanoy, JLaTondre, JMjimmy, Jojalo, Juggernaut0102, Kylehayes, Lkcl, MBParker, Mabdul, Maian, Martijn faassen, Mindmatrix, Misucat, MySchizoBuddy, Nickfitz, NikoN, Oskar Krawczyk, Plazarew, Rcoup, Robbiecheng, Rsrikanth05, Shenme, Suit, Talyian, Techtoucian, Thron7, Tiibidii, Tmillsclare, Tobius.miller, Twinkle, UnloX775, Ullersch, Venk2911, VoluntarySlave, Winterheat, 166 anonymous edits

**Comparison of JavaScript-based source code editors** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374028254> *Contributors:* 16x9, Adamjimenez1, Betacommand, Chris the speller, Dandv, Darolew, Digitalxero, FatalError, Francis Irving, Hamster128, Hownet, Jim Cornell, KevinDangoor, MartinKirk, Mbazon, Michael.c.harris, NathanaelJones, Nicolas Grekas, Phillipashlock, Sevamaster, Spellcoder, Tartley, Waldir, 122 anonymous edits

**Douglas Crockford** *Source:* <http://en.wikipedia.org/w/index.php?oldid=371818506> *Contributors:* Amalas, Anirvan, Bobblehead, DonToto, Dreftymac, Finlay McWalter, I900q1, Koviannyo, MantisEars, Martarius, Nadirss, Naval dudhoria, Pmaccabe, Quercus basaseachicensis, Renku, Souphanousiphone, Toussaint, Waldir, Yaron K., Zvn, 16 anonymous edits

**DWR (Java)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=370155856> *Contributors:* Bwilkins, Delfuego, Deltabeignet, Elwikipedista, Fplay, Ian Moody, JBellis, Jabencarsey, Jacob Robertson, Javy tahu, JoeWalker, Jonik, KuwarOnline, MER-C, Mendaliv, Modify, Ms2ger, Paul August, Pink greyhound, Rayngwf, RedWolf, RussianSpy2, Satya.sid, SoLando, Thron7, Tobias Bergemann, 27 anonymous edits

**EMVC** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374310309> *Contributors:* Fts 2010

**Brendan Eich** *Source:* <http://en.wikipedia.org/w/index.php?oldid=372035453> *Contributors:* AVRS, AcidJazzed, Alik Kirillovich, AlistairMcMillan, Amire80, Artichoker, Brendan Eich, ChrisEich, David Gerard, Davidwoswell, DonToto, DoriSmith, Dyl, Eliz81, FerranJorba, FlyingToaster, Giulio, GregorB, Hailey C. Shannon, Hervegired, Hu12, J JMesserly, Jamelan, Jamesmorrison, Jj137, Jlin, Joy, Jpbown, Jschuur, Katieh5584, Kwamikagami, Lzur, Mabdul, Minghong, Mms, Neurolysis, Paul1337, Peak, Plesner, Renku, Richmeister, Robertvan1, SMC, Sealican, SirFozzie, Srbauer, Stemonitis, Surfingslovak, SusanLesch, Toussaint, Ugur Basak, Underthefreeway, V0rt3x, Vlad, Voice of All, Wookieepedian, Yubal, Zebra zebra01, Irate, 43 anonymous edits

**JS/UIX** *Source:* <http://en.wikipedia.org/w/index.php?oldid=364357813> *Contributors:* PiRSquared17, Rwww, SuperTails92, 1 anonymous edits

**JSAN** *Source:* <http://en.wikipedia.org/w/index.php?oldid=363371375> *Contributors:* Acmeacme, id1337x, LastBall, Reinthal, Scrool, Spayrard, 8 anonymous edits

**JSdoc** *Source:* <http://en.wikipedia.org/w/index.php?oldid=369220608> *Contributors:* Atlant, CDV, Fintler, KAatremer, Micmath, Mindmatrix, Servel333, 13 anonymous edits

**JSLint** *Source:* <http://en.wikipedia.org/w/index.php?oldid=369495537> *Contributors:* Dnozay, DonToto, HelloAnnyong, Jodi.a.schneider, King of Hearts, Tomchen1989, Tomtheeditor, Waldir, 2 anonymous edits

**JSSP** *Source:* <http://en.wikipedia.org/w/index.php?oldid=354518246> *Contributors:* JLaTondre, JamieS93, Kazvorpai, Malcolm

**JScript** *Source:* <http://en.wikipedia.org/w/index.php?oldid=370628805> *Contributors:* Abu badali, Aegicen, AlistairMcMillan, Alksentrs, Ancheta Wis, Arbruijn, Asqueella, Black Falcon, Chealer, Cminor9, Danakil, Digita, DonToto, Dysprosia, FatalError, Fivestrokes, FleetCommand, Frap, Frecklefoot, FrenchIsAwesome, General Wesc, GhettoBlaster, Gracyzyk, Grendelkhan, Gschizas, Guaka, Gyrobo, Hervegirod, Isnow, Jacob Poon, John Nevard, John Vandenberg, Joshua Issac, Koavf, Latka, LittleDan, Mabdul, Maian, Maury Markowitz, Mdsy, Meand, Mephiles602, Minghong, Mipadi, Mlb1992, MrFirewall, Ms2ger, Myrdred, Oleg Alexandrov, Paul1337, PhMajerus, Piet Delpoit, Porges, Razork, Renku, RexNL, Ricvelozo, Rjwilmsi, Robbrown, Rowaars13, S.Örvarr.S, Sass24, SkyWalker, Soumyasch, Tanitall, Thumperward, TimJackson, TimotheusMax, Tizio, Toussaint, Tyomitch, Vlad, Voidvector, WadeSimMiser, William Avery, Ysangkok, 86 anonymous edits

**JavascriptMVC** *Source:* <http://en.wikipedia.org/w/index.php?oldid=364084415> *Contributors:* Drbreznjev, JLaTondre, Ohms law, Pvanrompay, Smashedpumpkin, Tedickey, Volox, 2 anonymous edits

**JSON** *Source:* <http://en.wikipedia.org/w/index.php?oldid=373597953> *Contributors:* 16@r, 1ForTheMoney, A1kmm, AdeBarkah, Agoode, AndonicO, Andrés Santiago Pérez-Bergquist, Andyparkins, Anna Lincoln, AnonMoos, Ariel., Ashawley, Austin512, AxelBoldt, Azbarcea, Balrog-kun, Beefyt, Beland, Betacommand, Bináris, Booles, Bspahh, Bunnyhop11, CHForsyth, CWii, CapitalR, Cassivs, Charles Iliya Krempeaux, Chowbok, Chris Q, Christopherlin, Colinmcc, Cowtowncoder, Cst17, Cybercobra, Cyberhitesh, D6, Dah31, Dainis, Damian Yerrick, DanilaKutkevich, David spector, Deblopper, Delfuego, Demisone, Dlinch, Digita, Digitalme, Dispenser, Dmeranda, Doekman, DonToto, Doniogo, DouglasCrockford, Dreftymac, Dylnuge, Easyas12c, Ebruchez, EdC, Elharo, Falcon9x5, FatalError, Frenchwhale, Furrykef, Gamol, Gerbrant, GhettoBlaster, Gorgalore, Gosub, Gsliin, Habbie, Hammer1980, Hogstrom, Hu12, Hughcharlesparker, IMSoP, Ian Moody, IvanLanin, Izhaki, læfai, Jainvineet, JamesNK, Jameswiltshire, Jarekadam, Jason.grossman, Jefe2000, Jeffreymanus, Jeltz, Jleedev, John Bentley, John Millikin, Johnuniq, Jose lcaza, Joshsteiner, Julesd, Jvangorp, Jvenema, Kadnan, Kakurady, Kingboyk, Klondike, Kragen, Krellis, Kromped, Ksn, Kuteni, Kwamikagami, Larsnostdal, Lindsay-mclennan, Lino Mastrodomenico, Lockoom, MEFlora, MER-C, Maian, Marcoshz, Markpeak, Martin.sweeney, Maxime.Debosschere, Mclid, Mets501, Mga, Mindmatrix, Minghong, Mister pink2, Mjb, Mjohjoseph, Mjs, Mralokk, Nasa-verve, Natkeeran, Nealmcb, Nickj, Nigelj, Ohnoitsjamie, Oliver.hessling, Ootachi, Otterfan, OwenBlacker, Panzi, Paranomia, Pctopp, Peak, Pediddle, Penagate, Peterl, Peu, Pgan002, Pimlottc, Piperez, Pne, Polyethene, Potatoswatter, Prashanthjoshi, Psifi, Rafiahmad, RedWolf, RedWordSmith, Reedy, Renku, RicJohnsonIII, Rich Farmbrough, Richtaur, Rjwilmsi, Robmanson, RockMFR, Roesser, RolandYoung, RossPatterson, RuudVisser, SQL, SaltyDawg, Sanspeur, Saxifrage, Scientus, Sevela.p, Shinkolobwe, Simetrical, Skaffman, Sleepyhead81, Snori, Ssd, Steffan Cline, Superm401, Supersiefriesj, Svipping, Syberguru, Tagith, TakuyaMurata, Thomasfrank.se, Thumperward, Tjholowaychuk, Tlane, Todd Vierling, Tom W.M., Tommy2010, Tothwolf, Tschubring, Unfletch, Universals, Vishalmamania, Vladimir Lapacek, Vocaro, Washi, Wineaficionado, Wrs1864, Zearin, Zlotstoy, Zziffle, Ævar Arnfjörð Bjarmason, 521 anonymous edits

**JsonML** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375391733> *Contributors:* Addshore, Cometstyles, Elwikipedista, GhettoBlaster, Kraftlos, Peak, Smm1051, Tobias Bergemann, Yellowweasel, Zearin, 19 anonymous edits

**Lightbox (JavaScript)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375520058> *Contributors:* Astangelo, Bilby, Black Falcon, Blantaff, Bpmox, Coil007, Dan Fuhrj, Danlev, Digitxps3, Drewbug01, Drum guy, Edupedro, Excirial, Fre133084, Hm2k, Hullo exclamation mark, Igge, Janderk, Kirkburn, Kueda, Matti Koskimies, Minghong, Minnyhaha, Ozh, Reinis, SchuminWeb, Staaky, Suit, Summit677, Wikipedian, WikiLaurent, 67 anonymous edits

**Lively Kernel** *Source:* <http://en.wikipedia.org/w/index.php?oldid=374731235> *Contributors:* DenisKrivosheev, Editor142, John of Reading, Robertkahn, 16 anonymous edits **Log4javascript**

*Source:* <http://en.wikipedia.org/w/index.php?oldid=354518347> *Contributors:* Android.en, Chet Ubetcha, JLaTondre, Malcolm, Stritti, Timdown, 4 anonymous edits **Medireview** *Source:*

<http://en.wikipedia.org/w/index.php?oldid=369643364> *Contributors:* AnonMoos, Apokrif, Asenine, BillFlis, David Gerard, Kate, Ligulem, McGeddon, Medinoc, Mets501, Michael Hardy, Mimzy1990, Minghong, Officiallyover, Pjacobi, Pne, R00m c, RickK, SimonP, Sjordford, Swpb, TonyW, Ubernstrum, Woohookitty, 19 anonymous edits

**Minification (programming)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=375376114> *Contributors:* Alejandro.myc, Alexius08, Andreas Kaufmann, Bcosca, Blazar, Buweichiu, Chappy84, Chris the speller, Clamum, Corpx, Czaries, DigitalOverload, Dreftymac, Ekerazha, Globex, Jaehyunkim, Kirillosipov, Liangent, Markusbradley, Motine, NSK Nikolaos S. Karastathis, Nneonneo, Pictonpanther, Tanthalas39, Tobias Bergemann, Tomjenkins52, Welsh, Wonko, 51 anonymous edits

**Objective-J** *Source:* <http://en.wikipedia.org/w/index.php?oldid=369274306> *Contributors:* Adam McMaster, Ahdustin, Aquilosion, Bertport, Cheesy, CyberGhostface, Cybercobra, DonToto, Dratman, Duncan, Frap, GiCodeWarrior, Jordandanford, Kdniedger, Maian, Massic80, Narendra Sisodia, Pascal.Honore, Pascal.Tesson, Paul Erik, Phil Boswell, Tohd8BohaihuGh1, Toussaint, VX, 22 anonymous edits

**John Resig** *Source:* <http://en.wikipedia.org/w/index.php?oldid=368928060> *Contributors:* Aavindraa, Altintx, B.rudge, Babbage, Ceejayoz, Collect, DonToto, DoriSmith, Dreftymac, Dtrebbien, Elwood J blues, FlyingToaster, Hobbestigrou, Jclerns, Jcottrell, Jerph, Message From Xenu, Rjwilmsi, RoS, 12 anonymous edits

**Reverse Ajax** *Source:* <http://en.wikipedia.org/w/index.php?oldid=354518378> *Contributors:* Agentscott00, Anaraug, Brest, CarlManaster, CometGuru, Damiens.rf, Fadookie, FatalError, Furrykef, Gregdan, In side the pc, Inquisitus, Jacobolus, Jwoodger, Kalan, Kdknigga, MrOllie, MuffledThud, Pcap, Psilya, Sleepyhead81, Sprocketonline, Stefan Hintz, Ødipus sic, 52 anonymous edits

**Rico (Ajax)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=362930613> *Contributors:* Bushido Hacks, Mabdul, PamD, PaulColby, Psychcf, 8 anonymous edits

**Seed (programming)** *Source:* <http://en.wikipedia.org/w/index.php?oldid=372692377> *Contributors:* CBM, Frap, Hortont424, Lineplus, Rich Farmbrough, 7 anonymous edits

**Server-side JavaScript** *Source:* <http://en.wikipedia.org/w/index.php?oldid=371519520> *Contributors:* Aaroniba, Alexandre Martins, Alexandre.Morgaut, Alik Kirillovich, Aschobel, Calc0000, Cheesy, Cirne, Cnomiya, Cryogenius, Digilee, Dikrib, Dm, Docu, Download, Dratman, Dreaded Walrus, Emersoni, FatalError, Fchoong, Frankgerhard, Gmosx, Gokudo, Greggrestonva, Gudeldar, Hutchike, Ironmagma, JLaTondre, JOuyang, Jchrisa, Jerry.yur, Jlin, KenFehling, Khakman2, Klizza, Kuteni, Lars3loff, Lucideer, Mabdul, Macaldo, Magioladitis, Maian, Mconstable, Metazargo, Mike Linksvayer, Mindmatrix, Minghong, Mob590, Modulatum, Ncb000gt, NetBMC, Pcolton, Peak, Petermichaux, Peterw8102, Pharos, Phyzome, Pipedreambomb, Pmedema, RHaworth, Reedy, SamJohnston, Sendmoreinfo, Smb1001, Stw, Sveinb, ThreeBlindMice, Timneu22, Tokek, Torc2, Toussaint, Tyvole, Westonmr, Wikieditoroftoday, Will Beback, 117 anonymous edits

**Comparison of Server-side JavaScript solutions** *Source:* <http://en.wikipedia.org/w/index.php?oldid=373221254> *Contributors:* Mabdul, 2 anonymous edits

**SproutCore** *Source:* <http://en.wikipedia.org/w/index.php?oldid=366875661> *Contributors:* Chronicfathead, Coolpixar, CultureDrone, DoriSmith, Forlonturtle, Id1337x, JLaTondre, Juanpin, Kissaki0, Mabdul, MarkAlexandre, Max Terry, Mbferg, Miami33139, MySchizoBuddy, Nyco, Pmedema, Rxcfc, Salavat, Savajit gupta, Theunixgeek, Thinkingman, Toussaint, Wikieditoroftoday, Witty lama, Xpclient, Yamla, Yonkeltron, 13 anonymous edits

**Unobtrusive JavaScript** *Source:* <http://en.wikipedia.org/w/index.php?oldid=372881297> *Contributors:* Alansohn, Amitchaudhary, Artw, BRPXQZME, Backzlider, Bobince, Bootedbear, Cander0000, Chrisfarms, Functionalguay, Greenie2600, Hymek, Inonit, Itpastom, Jatkis, Jesdisciple, Jesperonn, John Seward, Jseifer, Kazvorpai, Khalid hassani, MacCool, Metik, Moogle10000, NeoChaosX, Pmw57, Psd, Robroyaus, RossPatterson, SidneySM, Smalljim, Theone256, Vicoar, WernerPopken, 46 anonymous edits

**Venkman** *Source:* <http://en.wikipedia.org/w/index.php?oldid=369882841> *Contributors:* Agentab, Asqueella, C. A. Russell, CatherineMunro, Danakil, Enigmaman, GTBacchus, Gronky, IngerAlHaosulul, Isilanes, John Seward, Kathleen.wright5, K14m-AWB, Mabdul, Marudubshinki, Maximus Rex, Minghong, MyOwnLittlWorld, Patrickjolliffe, Rich Farmbrough, Rjwilmsi, Stan Shebs, TheParanoidOne, Vietbio, Wernher, Widefox, 19 anonymous edits

**XMLHttpRequest** *Source:* <http://en.wikipedia.org/w/index.php?oldid=372664827> *Contributors:* .:Ajvol.:, A3r0, Aditsu, Ahoerstemeier, Alaa.moustafa, Alansohn, Alcalazar, Alex Smotrov, Alexandre Martins, Algae, Alphachimp, Anirvan, Apv, Arjun G. Menon, Artw, Bezenek, Blackdenimgumby, BobBagwill, Bobo192, Bovineone, CDV, Caged.danimal, CambridgeBayWeather, CanisRufus, CapitalR, Catamorphism, Chealer, Christopherlin, Cic, Coffeeflower, DJ Rubbie, Damicatz, Dantman, Darklama, Delfuego, Digita, Dionyziz, Dirus, Discospinster, Dikenzie, Downfromzero, Drano, Dsnell923, EatMyShortz, EJOc, Eloi.sanmartin, Enyo, Eric B. and Rakim, Eve Teschlemacher, Fabiob, FatalError, Filipvr, Fred Bradstadt, Fromz, Furrykef, Gabrielsroka, Gerbrant, Gilgamesh, Gilliam, Gimboid13, GraemeL, GregorB, Haza-w, Hondavice, Ignacio Javier Igjav, Isnow, J.delanoy, Jaray, Javalenok, Javawizard, Jaw959, Jdowland, Jeroldan, Jmabel, John Vandenberg, Jriffel, Keelypavan, Khalid hassani, Kozuch, Krellis, Kugland, Lee J Haywood, LemonairePaides, Liberatus, Lindsay-mclennan, Locos epraix, Lupin, MC10, Macaldo, Maian, Mamund, Manop, Marktmilligan, Marskind, Martin Hampl, Martnym, Masonbarge, Meand, Merc64, Metaeducation, Mindmatrix, Minghong, Mnot, Molly, Mrs, Nickshanks, Nigelj, Nightstallion, Niven, Nkour, Norm mit, Oeln, Ohgyun Ahn, Pcj, Pctopp, PhOtOpHobc, Phloopy, Pjakubo86, Pjdonnelly, Proton.mule, Quiokos, Ramu50, RedWolf, Reisio, Remember the dot, Renku, RidinHood25, Ringbang, Rjwilmsi, Robert plevy, Rohan Jayasekera, Rufous, SalM, Schmoof, Sega381, Shameppwns, Simon Lieschke, SineSwiper, Skeejay, Slant, Sleepyhead81, Spankman, Speight, Stephen Morley, Suruena, SvartMan, Taka, TakuyaMurata, Tamlyn, Teiladnam, The Anome, TheJosh, Thedangerouskitchen, Thumperward, Tmc, Timeroot, Timwi,



Tolmaison, Twxs, Urkle0, Vberger, VictorAnyakin, Vladogr, Wengier, White 720, WhiteHatLurker, Widgetguy, WikHead, Yoderj, Zippedmartin, Zoef1234, Zvn, Zzuuzz, ~K, 380 anonymous edits

# Image Sources, Licenses and Contributors

**File:Wikibooks-logo-en.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Wikibooks-logo-en.svg> *License:* logo *Contributors:* User:Bastique, User:Ramac

**Image:Crystal source.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Crystal\\_source.png](http://en.wikipedia.org/w/index.php?title=File:Crystal_source.png) *License:* GNU Lesser General Public License *Contributors:* Dake, Kakurady

**Image:Symbol note.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Symbol\\_note.svg](http://en.wikipedia.org/w/index.php?title=File:Symbol_note.svg) *License:* Public Domain *Contributors:* User:KyraVixen

**File:Douglas Crockford.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Douglas\\_Crockford.jpg](http://en.wikipedia.org/w/index.php?title=File:Douglas_Crockford.jpg) *License:* Creative Commons Attribution-Sharealike 2.0 *Contributors:* Franco Folini from San Francisco, USA

**File:BEich.jpg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:BEich.jpg> *License:* unknown *Contributors:* Original uploader was AcidJazzed at en.wikipedia

**Image:Javascriptmvc.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Javascriptmvc.png> *License:* Trademarked *Contributors:* Justin B. Meyer

**Image:lightbox2.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Lightbox2.png> *License:* Public Domain *Contributors:* User:Dandaman32

**Image:Lively Kernel-engine text 200x200.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Lively\\_Kernel-engine\\_text\\_200x200.png](http://en.wikipedia.org/w/index.php?title=File:Lively_Kernel-engine_text_200x200.png) *License:* unknown *Contributors:* Editor142

**File:Jresig.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Jresig.png> *License:* Creative Commons Attribution-Sharealike 2.0 *Contributors:* Martin Kliehm

**Image:NewSproutCoreLogo.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:NewSproutCoreLogo.png> *License:* unknown *Contributors:* Calmer Waters, Mbferg, Salavat, Sreejithk2000, Yamla

**Image:SproutCoreDemo.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:SproutCoreDemo.png> *License:* unknown *Contributors:* Id1337x

**Image:Venkman 2.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Venkman\\_2.png](http://en.wikipedia.org/w/index.php?title=File:Venkman_2.png) *License:* unknown *Contributors:* Original uploader was Unforgettableid at en.wikipedia

# License

---

Creative Commons Attribution-Share Alike 3.0 Unported  
<http://creativecommons.org/licenses/by-sa/3.0/>

---