

Introduction to JavaScript

Learn how to program your computer!

0

***You can and must
understand computers
NOW***

“Everything is deeply intertwined. In an important sense there are no “subjects” at all; there is only all knowledge, since the cross-connections among the myriad topics of this world simply cannot be divided up neatly.”

—Ted Nelson, *Computer Lib/Dream Machines*

“When human beings acquired language, we learned not just how to listen but how to speak. When we gained literacy, we learned not just how to read but how to write. And as we move into an increasingly digital reality, **we must learn not just how to use programs but how to make them.**”

—Douglas Rushkoff, *Program or Be Programmed*

“The single most significant change in the politics of cyberspace is the coming of age of this simple idea: **The code is law. The architectures of cyberspace are as important as the law in defining and defeating the liberties of the Net.**”

—Lawrence Lessig, *The Code Is the Law*

1

Learning a new language

Code is `text`

Programming is
typing

Programming is **very**
careful typing

Programming is
fast typing

Programming is
figuring out why it
broke

Programming in general

- A series of **text files** that get compiled and executed
- Code is “digested,” going from human-readable to a hardware-ready form
- Ultimately programs run as assembly, low-level **instructions for your CPU**

JavaScript in particular

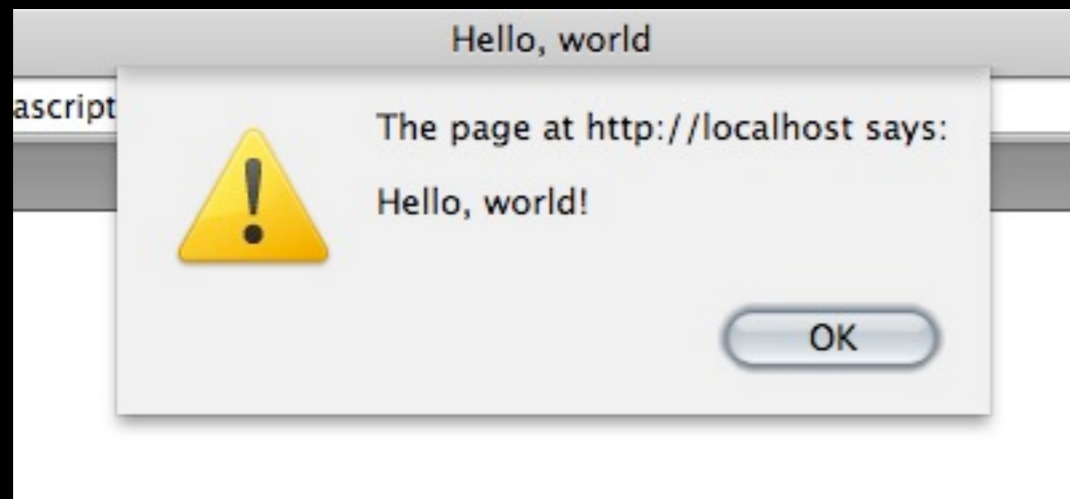
- Increasingly *the* **web page scripting** language
- Most likely the widest deployed runtime
- JavaScript has nothing to do with **Java**, except some syntax similarities

Lines of code

- A line of code is a basic unit of programming
- Tells the computer to do something
- Sometimes a “line” of code can span more than one line

A simple line of code

```
alert("Hello, world!");
```



Let's try this using
Firebug

2

Writing code

Compilers are unforgiving

- The computer cuts you no slack
- All code is subject to **bugs**
- The error console is your friend
- **Debugging** is about identifying, characterizing, and resolving problems

The image shows a browser's developer console with the following elements:

- Toolbar:** Includes icons for a bug, a mouse cursor, navigation arrows, a menu, and a dropdown menu currently set to "Console". Other tabs include "HTML", "CSS", "Script", "DOM", and "Net".
- Filtering:** A row of icons and buttons for "Clear", "Persist", "Profile", and a selected "All" filter. Other filter options are "Errors", "Warnings", "Info", and "Debug Info".
- Code:** The command `>>> alert("Hello, world"` is entered in the console.
- Error:** A red error message is displayed: `SyntaxError: missing) after argument list`.
- Input:** A prompt `>>>` is visible at the bottom left of the console area.

The image shows a browser's developer console with the following elements:

- Toolbar:** Includes icons for a bug, a mouse cursor, navigation arrows, a menu icon, and a dropdown menu currently set to "Console". Other tabs include "HTML", "CSS", "Script", "DOM", and "Net".
- Filtering:** A filter icon (three vertical bars) is followed by buttons for "Clear", "Persist", "Profile", and a selected "All" button. Other filter options are "Errors", "Warnings", "Info", and "Debug Info".
- Log Entries:**
 - Input: `>>> alert("Hello, world"`
 - Error: `SyntaxError: missing) after argument list`
 - Input: `>>> alert("Hello, world)`
 - Error: `SyntaxError: unterminated string literal`
- Input Field:** At the bottom, there is a prompt `>>>` followed by a vertical cursor line.

The image shows a browser's developer console with the following elements:

- Toolbar:** Includes icons for a bug, mouse, back, forward, and menu. A dropdown menu is set to "Console". Other tabs include "HTML", "CSS", "Script", "DOM", and "Net".
- Filtering:** A filter icon is followed by buttons for "Clear", "Persist", "Profile", and a selected "All" button. Other filter options are "Errors", "Warnings", "Info", and "Debug Info".
- Code Input:** Three lines of JavaScript code are entered:

```
>>> alert("Hello, world"  
>>> alert("Hello, world)  
>>> alert("Hello, world"))
```
- Error Messages:** Three red error messages are displayed:
 - "SyntaxError: missing) after argument list" (corresponds to the first line of code)
 - "SyntaxError: unterminated string literal" (corresponds to the second line of code)
 - "SyntaxError: missing ; before statement" (corresponds to the third line of code)
- Input Line:** At the bottom, there is a prompt character ">>>" followed by a vertical cursor bar.

A simple line of code

```
alert("Hello, world!");
```



Function name

A simple line of code

```
alert("Hello, world!");
```



Function name

Parentheses call the function

A simple line of code

```
alert("Hello, world!");
```



Function name

Parentheses call the function

Function argument (a string)

A simple line of code

```
alert("Hello, world!");
```



Function name



Parentheses call the function



Function argument (a string)



Designates the end of the line

3

Variables

The variable metaphor

*“Variables are like a box
you can put data into.”*

The variable metaphor

The variable metaphor



Variables

- Variables store data for future use
- `var x = y;` is how you assign a new variable in JavaScript
- We can now refer to `x` in future lines of code, and know it means `y`

Variables (boolean type)

- Variables store data for future use
- `var x = true;` is how you assign a new variable in JavaScript
- We can now refer to `x` in future lines of code, and know it means `true`

Variables (boolean type)

- Variables store data for future use
- `var x = false;` is how you assign a new variable in JavaScript
- We can now refer to `x` in future lines of code, and know it means `false`

Variables (numeric type)

- Variables store data for future use
- `var x = 47;` is how you assign a new variable in JavaScript
- We can now refer to `x` in future lines of code, and know it means `47`

Variables (string type)

- Variables store data for future use
- `var x = "pony";` is how you assign a new variable in JavaScript
- We can now refer to `x` in future lines of code, and know it means `pony`.

Variable logic

```
// What is the value of z?  
var x = 3;  
var y = x + 1;  
var z = y;
```

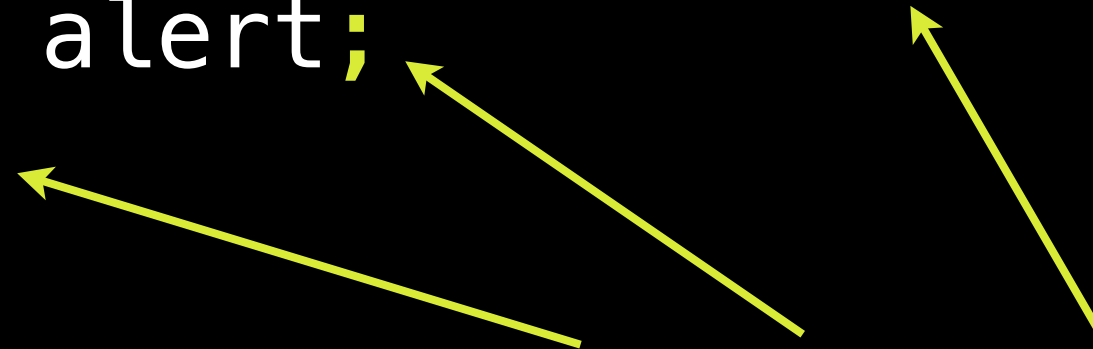
4

Functions

Multiple lines of code

```
var msg = "Hello, world!";  
var func = alert;  
func(msg);
```

Designate the ends of the lines



Multiple lines of code

```
var msg = "Hello, world!";  
var func = alert;  
func(msg);
```

The first line stores a string

Multiple lines of code

```
var msg = "Hello, world!";  
var func = alert;  
func(msg);
```

The second line stores a function

Multiple lines of code

```
var msg = "Hello, world!";  
var func = alert;  
func(msg);
```

The third line executes the stored function with the string

Commenting code

```
// First we store the message  
var msg = "Hello, world!";
```

```
// Next, we choose a function to call  
var func = alert;
```

```
// Finally, we combine the two  
func(msg);
```

Commenting code

```
/*
```

This code demonstrates the standard Hello World program, over three lines instead of just one.

```
*/
```

```
var msg = "Hello, world!";  
var func = alert;  
func(msg);
```

Creating a new function

```
// Outputs a simple message  
function output_message() {  
    var msg = "Hello, world!";  
    var func = alert;  
    func(msg);  
}
```

Calling our function

```
// Outputs a simple message  
function output_message() {  
    var msg = "Hello, world!";  
    var func = alert;  
    func(msg);  
}
```

```
output_message();
```

Arguments

```
// Outputs a simple message
function output_message(msg) {
    var func = alert;
    func(msg);
}

output_message("Hello, world!");
output_message("¡Hola, mundo!");
```

5

Libraries

JavaScript on the web

```
<script>
```

```
// JavaScript code is typically embedded in HTML  
// <script> tags
```

```
</script>
```

HTML + JavaScript

```
<html>
  <head>
    <title>HTML + JavaScript</title>
  </head>
  <body>
    <p>Stuff *on* the page goes up here.</p>
    <script>

      // JavaScript code that modifies the page should
      // go below everything else in the <body>.

    </script>
  </body>
</html>
```

Hide content

```
<html>
  <head>
    <title>Hide content</title>
  </head>
  <body>
    <p id="hide">Click to hide me!</p>
    <script src="mootools.js"></script>
    <script>
      $('hide').addEvent('click', function() {
        $('hide').fade('out');
      });
    </script>
  </body>
</html>
```

HTML + CSS + JavaScript

```
<html>
  <head>
    <title>HTML + CSS + JavaScript</title>
    <style>
      #content {
        background: #000;
      }
    </style>
  </head>
  <body>
    <p id="content">Hello, world!</p>
    <script>
      var content = document.getElementById('content');
      content.style.color = '#fff';
    </script>
  </body>
</html>
```

HTML + CSS + JavaScript

```
<html>
  <head>
    <title>HTML + CSS + JavaScript</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <p>
      Separating code into .js and .css files is a
      good way to keep things tidy.
    </p>
    <script src="scripts.js"></script>
  </body>
</html>
```

6

Slide show



a compact javascript framework

IN PARTNERSHIP WITH

[Home](#)

[Download](#)

[Docs](#)

[Forge](#)

[Blog](#)

[Dem](#)

Core

[More](#)

Type and hit enter to search

MooTools Core v1.3

Core

[Core](#)

Types

[Array](#)

[String](#)

[Number](#)

[Function](#)

[Object](#)

[Event](#)

Browser

[Browser](#)

Class

[Class](#)

[Class.Extras](#)

Element

[Element](#)

[Element.Style](#)

[Element.Event](#)

MooTools API Documentation

Popular Pages

- [Element](#) - Interact with the DOM
- [Element.Event](#) - Add events to DOM Elements
- [Class](#) - Use MooTools with Class
- [Fx.Tween](#) - Create effects for single properties
- [Request](#) - An XMLHttpRequest Wrapper

Interesting Blogposts

- [Setting Up Elements](#)
- [A Magical Journey into the Base Fx Class](#)
- [Get friendly with the Natives](#)
- [A Better Way to use Elements](#)

Previous Versions Documentation

- [MooTools 1.2.5 Docs](#)
- [MooTools 1.1 Docs](#)

Slide show HTML

```
<html>
  <head>
    <title>Slide show</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <div id="slides">
      <div id="inner">
        
        
        
        
      </div>
    </div>
    <script src="mootools.js"></script>
    <script src="script.js"></script>
  </body>
</html>
```

Slide show CSS

```
#slides {  
  width: 991px;  
  height: 671px;  
  margin: 0 auto;  
  overflow: hidden;  
  position: relative;  
}
```

```
#inner {  
  position: absolute;  
  left: 0;  
  top: 0;  
}
```

```
#slides img {  
  float: left;  
}
```

Slide show JavaScript

```
var width = 991;
var n = 0;
var count = $$('#slides img').length;

$('#slides').addEvent('click', function() {
    n = (n + 1) % count; // Increment
    $('#inner').tween('left', n * -width);
});
```

7

What next?

*Come up with a
project*

*Try to build it
yourself*

*Take your time, it
won't come quickly*

Resources

- [Eloquent JavaScript](#)
- [MooTorial](#)
- [w3schools.com](#)
- [Mozilla devmo](#)
- [WebMonkey](#)
- [The Rhino Book](#)
- [why's poignant guide to Ruby](#)
- [Dive into Python](#)
- [Visual Quickstart Guide](#)
- [Lynda tutorials](#)

<http://www.vimeo.com/5047563>

ARTANDCODE

Hackety Hack
why the lucky stiff

Art && Code Conference
<http://artandcode.ning.com>
Carnegie Mellon University
8 March 2009

<http://www.vimeo.com/5047563>