Script the Web
An introduction to scripting environments and languages

NetObjects ScriptBuilder™ 2.0 White Paper
March 1998
Contents

Introduction .................................................. 1
About Scripting ............................................. 2
Problems Faced When Building a Dynamic Web Site .......... 3
The NetObjects ScriptBuilder Approach to Solving These Problems .......... 3
NetObjects ScriptBuilder: Summary of Key Features .......... 4
Quick Interface Tour .................................... 6
Appendix: Web Scripting Languages ....................... 8
Introduction

In the past three years, user experience on the Web has improved greatly. Most sites use some form of animation, interactivity, or dynamic content. All of these elements require both client- and server-side scripting. Many scripting languages, each with its own functionality and power, are at the disposal of Web developers.

Despite the popularity of scripting languages, scripting development tools have not been available. In early 1997, Acadia Infuse 1.0, the first JavaScript development tool, was released. Its unique features addressed the basic needs of Web scripters. NetObjects subsequently acquired Acadia Infuse, and in 1998 released a new version of Acadia Infuse as NetObjects ScriptBuilder 2.0.

NetObjects ScriptBuilder complements the first two NetObjects products, NetObjects Fusion and NetObjects TeamFusion, and extends the NetObjects family of Web development products for professional site builders. NetObjects ScriptBuilder can also be used in conjunction with HTML authoring products such as Allaire HomeSite and Microsoft FrontPage. NetObjects ScriptBuilder is best suited for Web developers and sophisticated HTML authors who want to create dynamic Web sites that include application functionality.

NetObjects ScriptBuilder is the perfect complement to HTML authoring tools. NetObjects ScriptBuilder complements Netscape Visual JavaScript for basic script editing and the creation of JavaScript components (JavaBeans). ScriptBuilder also complements more complex, traditional RAD (rapid application development) tools such as Microsoft Visual InterDev by enabling easy integration of scripts or pages written in ScriptBuilder with larger Web application projects. In addition, Java developers who want to integrate their applets, servlets, or JavaBeans within their Web sites (using tools such as Visual Age for Java or Symantec Café) will find NetObjects ScriptBuilder very complementary.
About Scripting

A script is a set of commands, either in a file or at the command line, that performs multiple operations on a file or files. Scripts (compared to traditional 2- and 3-GL languages) are not compiled to native machine-executable code but are interpreted at run time. Scripts provide higher levels of abstraction than 3-GL languages (such as C++ or Java™) do, and as a result, script and macro development is more accessible than 3-GL development.

Scripting is not a new concept. Scripting languages and tools have been available to developers since the 1970s. Early scripts were executed in PERL, Python, TCL/TK, REXX, AppleScript, and many other formats. However, the major scripting languages in use today on the Web are less than two years old. The only way to include interactivity or dynamic information in your HTML (Hypertext Markup Language) documents is through use of client or server-side scripts. These scripts could be used to deliver dynamic content such as the current date and time, a chart based on user criteria, a counter representing the number of visitors to a page, the results of a database search, an animation, or a user profile.

Originally, the Web community consisted of HTML designers and CGI coders. While many products and vendors have addressed HTML development, CGI coding has been an untapped market. By late 1997, CGI coders had evolved into Web scripters (estimated to number more than 1 million). These Web scripters’ skill sets range from that of page/site builders to that of programmers and application developers.

Scripting has become ubiquitous as a Web technology for three reasons:

1. As Web sites have become more dynamic, they have outgrown the functionality provided by HTML. HTML is a markup language limited to adding formatting attributes to a data stream.
2. Open Web scripting standards, including ECMAScript, JavaScript, and VBScript, have emerged.
3. Scripting is less technical than programming, and therefore is better suited to the majority of Web developers, many of whom have design and marketing backgrounds.

The NetObjects acquisition of Acadia’s market-leading scripting products gives NetObjects a technology edge, allowing it to extend its product offering and provide solutions for scripters as well as for site builders. NetObjects is becoming the one-stop shop for building Web sites.

JavaScript has become the most popular client-side scripting language on the Web by far. As reported by HotBot.com, Web developers rapidly adopted JavaScript in the year following its 1995 introduction. In 1996, there were more than 300,000 JavaScript- and 7,000 VBScript-enabled pages on the Internet. In 1997, there were 3.1 million JavaScript- and 60,000 VBScript-enabled pages on the Internet. It is worth noting that these statistics reflect only client-side JavaScript; scripts executed on the server side cannot be tracked, since only their resulting HTML is accessible to traditional Web crawlers such as HotBot.com.
Problems Faced When Building a Dynamic Web Site

Since 1995, the Web has evolved dramatically, from providing static content to providing dynamic, interactive, and personalized content. Web site creators and managers must have fresh, dynamic content in order to retain visitors' interest. Scripting is a must-do task that helps achieve this goal.

Challenges that are unique to organizations adding sophisticated scripts to their Web sites include:

- Taking advantage of new Document Object Models and DHTML capabilities for richer, more seamless user experiences.
- In moving from static to dynamic Web sites, avoiding low-level Java or C++ programming while taking advantage of universal server-side scripting platforms such as Netscape LiveWire, Allaire Cold Fusion, or Microsoft Active Server Pages.
- Addressing Web browser incompatibilities. Each browser behaves differently when interpreting scripts, causing infinite problems for Web site developers.
- Determining which scripting languages to support. Each language has different uses, strengths, and weaknesses. In addition, staff members typically are not familiar with all these languages.
- Providing information and applications on the Internet, where competition drives technology adoption faster than ever before. The number of JavaScript pages on the Web, for example, grew by more than 930 percent between 1996 and 1997.
- Meeting the need for Web sites that are interactive and constantly fresh in content, which increases the burden and stress on the organizations providing these sites.

The NetObjects ScriptBuilder Approach to Solving These Problems

To build dynamic Web sites, professional site builders need to work in an open environment that offers total flexibility. Site builders also need to use existing HTML editors such as Allaire HomeSite or BBEdit, to add scripts easily, and to publish documents that are viewable by different browsers.

To meet the growing and diverse needs of professional site builders, NetObjects ScriptBuilder 2.0 offers a visual and open environment, supports multiple browsers, and is independent of the Web server platform.

NetObjects ScriptBuilder is the best companion for HTML authors (such as HomeSite) because it is the first editor designed for scripting. NetObjects ScriptBuilder bridges the gap between the Netscape and Microsoft Web scripting environments, with its full support for JavaScript, JScript, HTML, DHTML, Netscape LiveWire, VBScript, Allaire Cold Fusion, and Microsoft Active Server Pages. It combines a powerful script editor and rich development tools with a point-and-click environment that speeds script development. The Script Inspector evaluates which browser versions are needed to run scripts and makes it easy to find flagged language elements.
NetObjects ScriptBuilder: Summary of Key Features

**Visual Integrated Development Environment**

NetObjects ScriptBuilder 2.0 includes an advanced script editor that lets you create and update Web scripts and documents in a visual IDE.

The editor provides a familiar environment that instantly complements your existing development tools. User-definable color syntax highlighting makes your code easier to read and debug. Auto Indent reduces keystrokes, and AutoScripting ensures proper syntax by automatically inserting the appropriate script elements.

Standard HTML support is available through the HTML tag tree, as well as through tag shortcuts available on the HTML toolbar and related menu commands.

**InfoDesk and Scripting Language Reference**

The InfoDesk allows you to navigate easily among various language reference guides, document maps, and desktop folders.

The InfoDesk consists of three tabs:

1. **Info** integrates language references for the Netscape and Microsoft Browser Object Models, Netscape JavaScript, Microsoft JScript, Microsoft VBScript, Lotus Script, and HTML. The InfoBrowser displays reference information on language elements, and the language tree lets you drag and drop elements into your source document. You can also access online documentation and related Web pages from this visual tree.

As a Web developer, you no longer need to maintain a bookmark collection pointing to various scripting sites and resources. NetObjects ScriptBuilder provides an integrated, comprehensive list of URLs in a single reference point for language elements. It even includes a list of popular scripting sites that can be of help to users.

2. **Map** provides an interactive outline view of the current document. Not only can you view your source document from a hierarchical point of view, but you can also work directly with the document elements from within this view.

3. **Desktop** provides a Windows Explorer look and feel, allowing you to drag local files into the NetObjects ScriptBuilder workspace.

![Figure 1. Map of script functions and HTML elements in a document](image)
Script Inspector

One of the key challenges faced by Web scripters and developers is the incompatibility of Netscape Navigator and Microsoft Internet Explorer. Each browser implements different levels of JavaScript and Dynamic HTML. The same snippet of code may be interpreted differently by each browser. This situation forces scripters to test their code in multiple browsers to make sure it works properly.

NetObjects ScriptBuilder 2.0 features an intelligent Script Inspector that provides a fast and easy way to inspect the Web pages being developed. The Script Inspector reports any incompatibilities with specific versions of either Netscape or Microsoft browsers, which dramatically speeds the development of error-free scripts.

![Figure 2. Script Inspector results, showing supported browser versions](image)

The Script Inspector runs behind the scenes and evaluates the open document. It informs you of which versions of Netscape and Microsoft browsers are needed to successfully run the code residing in your document. You can also step through the document and locate the specific language elements that were flagged as “challenging” to each browser.

Script Library

NetObjects ScriptBuilder 2.0 offers a rich library of frequently used scripts. Users can customize and expand this library.

Users can also update their Script Library easily by adding scripts from NetObjects and other companies. This library saves enormous amounts of development time by allowing easy reuse of code.
Internal/External Preview
NetObjects ScriptBuilder 2.0 lets you preview your code while in design mode by using an internal browser to toggle between edit and preview modes. Alternatively, you can launch Netscape Navigator, Microsoft Internet Explorer, or any other browser externally. NetObjects ScriptBuilder doesn’t restrict you to the latest browser version. You can configure the external preview to use any version of any browser.

Quick Interface Tour
The following descriptions provide a quick overview of the NetObjects ScriptBuilder environment.

User Interface Overview
NetObjects ScriptBuilder 2.0 has an innovative, intuitive user interface that incorporates both drag-and-drop functionality and color syntax highlighting. The visual IDE (Integrated Development Environment), with its ease of navigation, rich features, and power, rivals those of traditional development tools.

Desktop structure:
- InfoDesk
- Script Inspector
- Script Library, which offers an easy way to manage frequently used scripts and code, providing scripters with greater code management capabilities and support for JavaScript include files
- Language references
- Auto Indent, unindent, block indent, and bookmarking
- AutoScripting, an optional feature that completes text as you type. For example, if you type “for,” the block of a “for loop” is inserted and the first variable text is highlighted
- Color syntax highlighting, which makes your code easier to read by coloring various elements such as keywords, functions, comments, HTML tags, and much more
- File drag-and-drop from Windows Explorer onto the NetObjects ScriptBuilder workspace
**Scripting Overview**
NetObjects ScriptBuilder 2.0 supports the latest scripting technologies, including JavaScript 1.2, JScript 3.0, Dynamic HTML, and VBScript language extensions for both Netscape Navigator and Microsoft Internet Explorer 4.0. The InfoDesk also includes the latest server-side technologies for both Netscape and Microsoft, with reference information on both Netscape LiveWire and Microsoft Active Server Pages.

NetObjects ScriptBuilder features several ready-to-use code templates for inserting code quickly. For example, you can insert a function block (complete with description header) from a menu command or a shortcut key, or you can turn the selected text into a script comment block with a shortcut key.

**NetObjects ScriptBuilder Strengths**
NetObjects ScriptBuilder is designed expressly for the Web developer. The purpose of NetObjects ScriptBuilder is to facilitate the overall development and debugging of script-intensive Web pages. The product is focused on:

- Writing both client- and server-side scripts for complex Web pages
- Ensuring that Web pages are accessible by all popular browsers
- Previewing scripted pages instantly
- Referencing and adding the latest language elements seamlessly
- Navigating to embedded functions and objects quickly
- Accessing frequently used scripts and code snippets with one click
Appendix: Web Scripting Languages

What Is JavaScript?

JavaScript is a platform-independent, event-driven, interpreted programming language that was developed by Netscape. JavaScript enables Web developers to exercise control over the client's browser, primarily to enhance the user experience through event handling and client-side execution. Originally, JavaScript was called LiveScript. Netscape also released a precompiled, server-side language named LiveWire,™ which is now known as server-side JavaScript.

Before the introduction of JavaScript, creating interactive Web pages was very difficult. Collecting and processing user data required a common gateway interface (CGI) script that ran on the Web server. After designing the HTML data entry form, the developer would create a CGI script in C or PERL to process the form's elements on the Web server. Heavy traffic combined with CGI scripting would cause a major bottleneck on the server.

With this system, unfortunately, each time a user submits a form, data is sent to the Web server, where an instance of the CGI script is executed. The CGI script processes the data and returns the results to the user as a new HTML page. This happens every time that a user makes changes to the Web form and sends information across the Internet connection, requiring several server CPU cycles and affecting bandwidth. This process causes bottlenecks and delays for screen updates, an unexpected inconvenience from a graphical user environment such as the Web, and a bad user experience.

Managing multiple processes became easier with the introduction of NSAPI (Netscape Server Application Programming Interface) and ISAPI (Information Server Application Programming Interface). However, these technologies require advanced programming expertise. Client-side JavaScript provides an alternative solution by collecting and processing data within the browser. (Data validation is one example of this.) You may still need to send browser-processed data to a CGI script.

Contrary to popular perception, JavaScript is not based on Java (Sun's object-oriented programming language), nor is it a subset of Java. JavaScript and Java share an object-based foundation and interoperate well, but they are different both functionally and technically.

JavaScript is very popular for adding interactivity to Web pages, because JavaScript scripts can be embedded within HTML documents easily by enclosing code within a <SCRIPT> </SCRIPT> tag pair. Most popular browsers currently support a version of JavaScript.

Client-side scripting is ideal for validating user input, such as ensuring that required fields within a form have the appropriate values or format. Other uses of client-side scripting are presenting users with error messages, prompting users with confirmation dialogs, performing certain calculations on values within a form, and otherwise manipulating data. Client-side scripting is also convenient when performing conditional calculations, functions, or actions that depend on the state (such as browser type or operating system) of the client. Using client-side JavaScript dramatically lessens the execution load on Web servers.

Netscape Navigator 2.0 was the first browser to introduce JavaScript 1.0. It was followed by Netscape Navigator 3.0, which supported JavaScript 1.1. In 1997, Netscape Navigator 4.0 was released, with support for JavaScript 1.2 and DHTML (Dynamic Hypertext Markup Language).
What Is Server-Side JavaScript?
Server-side JavaScript is the JavaScript code (usually an HTML document) that resides and executes on the Web server. Depending on how the JavaScript code is tagged, it will execute either on the server or on the client that is connecting to the application or HTML page. Unlike `<SCRIPT>`-tagged client-side JavaScript (which executes within the browser), `<SERVER>`-tagged server-side JavaScript (recognized only by Netscape Servers) executes on the server, interprets the script, and sends results to the Web browser in the form of either HTML or client-side script.

Determining whether your JavaScript code will run on the client or on the server is usually dictated by the task that you'd like to achieve. Server-side scripting is ideal for maintaining data shared among several applications or browsers, tracking the state of the client session, calling external libraries on the Web server, performing database functions on the server, and generating customized code (that might run on the client or the server) dynamically.

What Is JScript?
Microsoft reverse-engineered Netscape JavaScript to enable Internet Explorer to support client-side JavaScript. JScript is a scripting language designed for the Internet. It is implemented as an interpreter for Web browsers and other applications. JScript is comparable to JavaScript, but it is supported only in Microsoft Internet Explorer. Like JavaScript and VBScript, JScript is an interpreted language that processes code embedded in HTML documents. JScript code (like VBScript code) does not produce standalone applets, but is used to add interactivity to Web sites.

JScript was first introduced in Microsoft Internet Explorer 3.0. Microsoft Internet Explorer 4.0 and Internet Information Server 4.0 include the scripting engine for JScript 3.0 that conforms to the ECMAScript ECMA-262 specification.

What Is VBScript?
VBScript (also known as Visual Basic Scripting Edition) is a subset of Microsoft Visual Basic. VBScript is implemented as an interpreter for Web browsers and other applications, so it uses variants instead of strict types. VBScript does not support file I/O or direct access to the underlying operating system.

VBScript is comparable to JavaScript and JScript, but not Java. Like JavaScript, VBScript is an interpreted language that processes source code embedded within an HTML page. VBScript (like JavaScript and JScript) does not produce standalone applets, and is used to add interactivity to static HTML documents. VBScript is used on the server currently, typically by Web developers who target the Microsoft Active Server Pages platform.

What Is ECMAScript?
ECMA (European Computer Manufacturers Association) is an international standards body responsible for information and communications systems, development languages, and other technologies. ECMA, with more than 30 years of standardizing experience, is a principal participant in the International Standards Organization (ISO) standards family and has a Class A ISO liaison, which facilitates acceptance of JavaScript as an ISO standard.

ECMAScript is a Web scripting language that can enrich and enliven HTML pages in a Web browser. ECMAScript is the only standards-based scripting language on the Web and is based on the ECMA-262 specification, which outlines an object-oriented programming language for performing computations and manipulating objects within a Web browser. The complete ECMA-262 specification can be found at http://www.ecma.ch/stand/ecma-262.htm.
In late 1996, Netscape submitted the JavaScript 1.1 language to the ECMA as the basis for the ECMA-262 standard. Microsoft then joined the task force and suggested enhancements of its own. When you're looking at JavaScript and JScript, bear in mind that they are implementations of the same standard scripting language (ECMA), but they are promoted and marketed by competing companies.

While JavaScript 1.1 served as the basis for ECMAScript, standardization has resulted in significant language improvements, particularly in IEEE math, date functions, and Unicode support.

Microsoft JScript 3.0 supports ECMAScript-262 and provides the same functionality that Microsoft submitted for consideration in the next ECMAScript specification. This functionality includes features such as conditional compilation, provision of variable code paths (depending on variable values at runtime), and control flow enhancements (such as support for SWITCH, LABEL, and DO... WHILE statements).

What Are Scriptlets and JS Beans?
DHTML scriptlets are snippets of code (supported only in Microsoft Internet Explorer 4.0 or later on the client) that let Web developers build user interface components via HTML and scripting. Microsoft also recently introduced server side scriptlets for the Microsoft Active Server Pages platform.

Larger Web teams tend to benefit from skill-set specialization and scriptlets can be a useful mechanism for describing and sharing code components and improving the Web production process. Scriptlets are based on Microsoft's Component Object Model (COM) and can only be used with other applications supporting COM technology.

Netscape does not support Microsoft scriptlets in the Navigator client, and has its own JavaScript Beans interface (that is exposed only in Visual JavaScript currently). The NetObjects JavaScript Bean Builder is a tool licensed by Netscape, and included in SuiteTools 2.0, that enables Visual JavaScript developers to build ready-to-use JavaScript components that can be dropped easily into the Visual JavaScript palette. NetObjects JavaScript Bean Builder includes:

- Visual environment for developing JavaScript components
- Visual component-interface assembly
- Robust JavaScript editor
- Built-in JavaScript 1.2 object model and language reference tree
- Drag-and-drop file functionality

In general, DHTML scriptlets are simply Web pages that have been authored using HTML and scripts. On the Win32 platform, a DHTML scriptlet is a component that uses COM to define the methods and properties of DHTML scriptlets. Once the DHTML scriptlet is downloaded to the client, it is cached for future use.

DHTML scriptlets can also be described as miniature Web pages, encapsulated within windows on the containing page. DHTML scriptlets can do anything possible with a regular Web page. Scripting tools such as NetObjects ScriptBuilder can be used to create the user interface for a DHTML scriptlet component, including any JavaScript behavior needed. It should be noted that the Netscape and Microsoft implementations of Dynamic HTML are different from each other. Netscape's Dynamic HTML technologies include Layers and Cascading Style Sheets, which are unsupported by Microsoft Internet Explorer.
What Is a JavaScript Include File?
Regardless of the size of your organization, you should seriously consider componentizing and packaging your scripts and HTML documents. One technique for handling such tasks is using JavaScript Include (JS Include) files. Developers, or groups of developers, can “include” many JavaScript functions in a single source file (.JS file) that can be referenced within multiple HTML pages. Changes to any given script are made in a single location, avoiding the need to modify every HTML document. (This functionality is analogous to the #include file in C programming.) JavaScript code in the external .JS file is handled as if it were located in the main document. JS Include files work quite well when developing libraries of JavaScript code for reuse in multiple pages.

Using JS Include files is as easy as adding an SRC property to the <SCRIPT> tag referencing the external include file (for example, <SCRIPT LANGUAGE="JavaScript" SRC="MyLib.js">).

One drawback with this approach becomes apparent anytime you need to change the “sniffing” code. Each page that references the include file will have to be updated individually. Also, since Internet Explorer caches URLs (including the include file code) on the client, server hits can be reduced. Although the first requested page that uses the include file requires two server hits (one for the page, one for the include), each subsequent HTML page that reuses the include file code can access it from the client cache, rather than making an additional request to the server.

JavaScript Include files are supported by both Netscape Navigator 3.0 or later and Microsoft Internet Explorer 3.02 or later, provided the Web server is configured for the .JS mime type=application/x-javascript exts=js.

What Is XML?
The eXtensible Markup Language (XML) is a simplified subset of the Standard Generalized Markup Language (SGML, ISO 8879), and is designed solely for Web applications. XML provides a standard format for describing different types of data, such as a personal contact record or an invoice, so that the information can be decoded, manipulated, and displayed consistently and correctly regardless of the type of client being used. XML provides a file format for representing data, a schema for describing data structure, and a mechanism for complementing and annotating HTML with semantic information. For instance, you might want to describe a financial electronic transaction in a form that any recipient can decode and process, or you might want to transmit an electronic message that a pager can interpret. These scenarios, as well as other vertical industry Document Type Definitions (DTDs), are possible using XML as a standard data format and HTML as the presentation format.

XML is becoming a universal format for data on the Web. It allows developers to easily describe and deliver rich, structured data from any application in a consistent way. It is inaccurate to assume that XML is a replacement for HTML. Rather, XML is a complementary format. On February 10, 1998, the W3C Consortium issued its recommendation for XML 1.0, which can be found at http://www.w3.org/TR/1998/REC-xml-19980210 and http://www.w3.org/XML/.

To construct the view of XML data, the Cascading Style Sheet (CSS) language must also be enhanced. The W3C is working on an eXtensible Style Language (XSL) specification, which is designed to deal with rich style content such as fonts, colors, spacing, and formulas for XML documents. CSS is declarative and not user-extensible; XSL is primarily declarative, but is user-extensible through use of ECMAScript. XSL uses an XML notation, while CSS uses its own.
XSL is compatible with CSS, and is derived from the Document Style Semantics and Specification Language (DSSSL), a language with roots in the SGML community. Although CSS can still be used for simply structured XML data, it cannot be used to present information in an order different from how it was received (for instance, it cannot present a purchase order number as a page footer). There are many other features of XSL, such as its scripting capability, that make it an exciting new technology. For more information on the XSL specification, refer to http://www.w3.org/Style/XSL/.

The anticipated future growth of XML/XSL will turn scripting into an even more pervasive computing activity throughout the Web. For the sake of simplicity, consider that XSL is to XML what CSS is to DHTML.