Welcome to NetObjects ScriptBuilder 3.0

NetObjects ScriptBuilder™ 3.0 is a script development environment for the World Wide Web, offering the fastest way to bring client- and server-side scripting to Web sites. It combines a powerful script editor and rich development tools with a visual, point-and-click environment that speeds script development.

ScriptBuilder bridges the gap between Netscape, Microsoft, IBM, and Sun Microsystems Web scripting environments. It includes full support for Allaire Cold Fusion Markup Language (CFML), Dynamic HTML, ECMAScript, HTML, IBM Java Server Pages (JSP), JavaSoft Java, Lotus LotusScript, Microsoft Active Server Pages (ASP), Microsoft Channel Definition Format (CDF), Microsoft Document Object Model (DOM), Microsoft JScript, Microsoft VBScript, Netscape JavaScript, Netscape LiveWire, and Perl.

With NetObjects ScriptBuilder, you can:

- Check your script syntax for errors
- Create reusable object-oriented script components
- Access local and Web-based language reference from within ScriptBuilder
- Customize the language reference using Extensible Markup Language (XML)
- Check your documents for browser compatibility
- Drag and drop language elements into your document
- Use Web-standard languages or languages other than Web-standard
- Quickly navigate to embedded functions and components
- Automate repetitive tasks
- Preview your documents internally or in the external browser of your choice

Overview

NetObjects ScriptBuilder provides an ideal script development environment for the Web, with templates for HTML, JavaScript, and other languages that you can edit and include in your HTML files. You can use the base templates provided with ScriptBuilder, develop your own custom templates, or write code from scratch.

With the introduction of script components, you can now apply object-oriented programming principles to scripting. A script component is a JavaScript (ECMA) object inside a JavaScript (.js) include file. To describe the component to the outside world, ScriptBuilder adds a public interface written in Extensible Markup Language (XML) to the include file, producing a Script Component Class (.**scc**) file you can drag onto your HTML page. Using ScriptBuilder's Component Inspector, you can assign properties and events to the script component.

By encapsulating your scripts into component wrappers, you make it possible for other developers to reuse your script components in Web pages and server-side applications, working with the script component's public interface instead of the underlying code.

ScriptBuilder provides a reference library to help you manage the complexity of multiple, incompatible scripting technologies. Use the reference library locally or online to identify functions and check usage in the different script languages. In addition, you can use XML to add language references for your own scripting tags and Document Type Definitions (DTDs).

ScriptBuilder includes several features to help you write clean, error-free code:

- Color syntax highlighting—to read and debug your code easily
- Automated tasks—to reduce repetitive keystrokes and errors
- Document/Components Map—to find embedded functions and scriptable objects visually
- Script Inspector—to learn whether your script contains elements that are unsupported by a particular browser and to find solutions for any compatibility issues you encounter
- Syntax checker—to identify syntax errors quickly in your code before previewing in the internal browser or an external browser

About the Help

The NetObjects Help addresses all aspects of using NetObjects ScriptBuilder to help you create code to build a Web site. It is organized around the tasks you perform while developing the Web site code.

The manual assumes that you are familiar with Windows 98, Windows 95, or Windows NT. If not, see the appropriate User Guide.

In NetObjects ScriptBuilder 3.0 User Guide, all file names, extensions, pathnames, and URLs appear in this typeface: **MyWebSite.html**.

Getting Other Types of Help

NetObjects ScriptBuilder offers many online help options. Help for language elements is available online and on the ScriptBuilder CD. See Using the InfoBrowser for Language Reference on page 30.

To obtain help for ScriptBuilder functions:

- Choose Help Overview from the Help menu.
- Visit **www.scriptbuilder.com**.
- Visit the newsgroup available at www.netobjects.com/support.

Menu bar

You can float the menu bar and place it in a separate window. You can also dock the menu bar in the ScriptBuilder window immediately below the title bar or on either side of the application window.

Toolbars

Toolbars provide rapid access to a large assortment of tools useful in writing and editing code, as well as to the Internet. ScriptBuilder has seven toolbars you can display all at once, individually, or in any combination you choose. ScriptBuilder initially displays a toolbar at the top of the window. You can then drag the toolbar anywhere you like in the Script Builder window. For more about toolbars, see the User Interface Reference.

Toolbox with InfoDesk Reference panel

The Toolbox, located in the left pane of the ScriptBuilder window has six panels that display reference and other information useful in creating documents.

The InfoDesk Reference panel gives you access to online language reference materials. You can also drag language elements from the InfoDesk Reference panel and drop them into your document.

Editor

The ScriptBuilder Editor is where you work on your source file. You can have several documents open at the same time in the Editor.

Status Bar

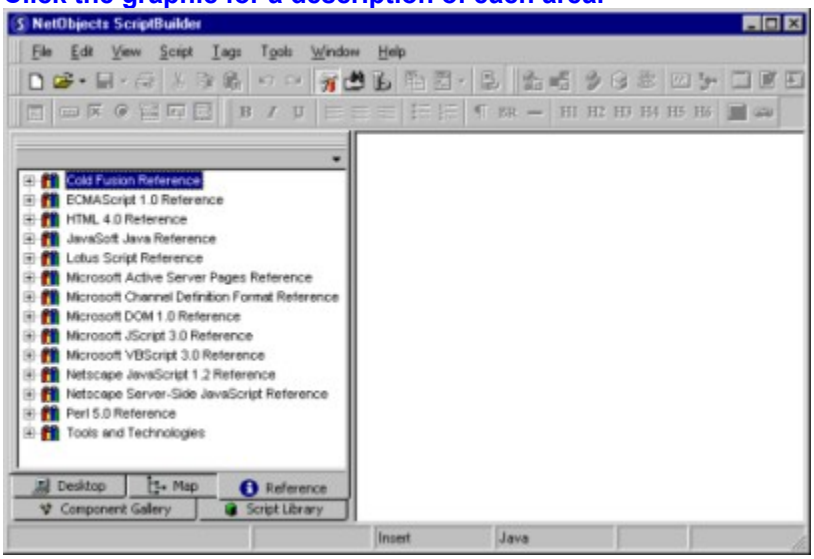
The status bar is located at the bottom of the ScriptBuilder window. By default, the status bar is active when you first open ScriptBuilder.

Working in the ScriptBuilder Environment

When you start ScriptBuilder, the screen is divided into three areas:

- Toolbars
- Toolbox
- ScriptBuilder Editor

Click the graphic for a description of each area.



As you write your Web document, you might want to change which toolbars are displayed, how they are arranged, whether the Editor takes up the full window, and so on. The ScriptBuilder environment is very flexible, and you can change its look to correspond to your various needs in the document development process.

Working with Toolbars

Toolbars provide rapid access to a large assortment of tools useful in writing and editing code, as well as to the Internet. ScriptBuilder has seven toolbars you can display all at once, individually, or in any combination you choose. ScriptBuilder initially displays a toolbar at the top of the window. You can then drag the toolbar anywhere you like in the Script Builder window.

When you first open ScriptBuilder, you see these toolbars at the top of the window:

- Standard

- Script

- Form Elements

- HTML Tags

You can display three additional toolbars:

- InfoBrowser

- Window List

- Find

The individual toolbar buttons are illustrated and described in The User Inferface Reference.

Floating and Docking Toolbars and the Menu Bar

You can float the toolbars and the menu bar—make them separate windows—or dock them in the ScriptBuilder window immediately below the title bar or on either side of the application window.

To float a toolbar or the menu bar:

- Drag it by the two lines at the end of the bar until it becomes a free-floating window.

To move a floating toolbar or the menu bar:

- Drag it by its title bar to the new location.

To resize a floating toolbar or the menu bar:

- Drag any of its sides.

Docking a Toolbar or the Menu Bar

To dock a toolbar or the menu bar:

1. Drag its outline toward the left or right side of the ScriptBuilder application window or up to the application window's title bar.

   If you move the toolbar or menu bar to either side of the window, the outline rotates 90 degrees when it touches the side.

   ScriptBuilder indicates that you can dock the toolbar or menu bar by changing from a shaded outline to a single outline.

2. Release the mouse button to complete the docking.

Hiding and Showing Toolbars

To hide a toolbar, do one of the following:

- From the View menu, choose Toolbars and then deselect the toolbar.
- Right-click the toolbar, and deselect it in the shortcut menu that appears.
- If the toolbar is floating, click its close box.

To display a toolbar, do either of the following:

- From the View menu, choose Toolbars and the name of the toolbar you want.
- Right-click the toolbar area under the menu bar, and select the toolbar you want to display from the shortcut menu.

Note:

If you hide all the toolbars, you must use the View menu to display the first toolbar. You can then right-click to display additional toolbars.

Using the Toolbox

The Toolbox, located in the left pane of the ScriptBuilder window (see Using the Toolbox for a more detailed description), has six panels that display reference and other information useful in creating documents.

- The InfoDesk Reference panel gives you access to online language reference materials. You can also drag language elements from the InfoDesk Reference panel and drop them into your document. Adding Your Own InfoDesk Reference Book describes how you add content to the InfoDesk Reference panel.

The Map panel shows separate maps of the components and language elements in your document.

The Desktop panel is like a small Windows Explorer. You can see the structure of your disks, and you can open files from the Desktop.

The InfoDesk Reference, Map, and Desktop panels are described in Using the Toolbox.

The Component Gallery contains script components, which are blocks of script code that are referenced from JavaScript include files. The Component Gallery is described in Working with Script Components.

The Script Library contains snippets of script code that you can insert into your document. The Script Library is described in Adding a New Script to the Script Library.

Opening and Closing the Toolbox

If you want your document or preview to have the entire application window, you can toggle the Toolbox closed (hidden) and then open it again.

To close or open the Toolbox, do either of the following:

-  Click the Toolbox button on the toolbar.
- From the View menu, choose Toolbox or press F6.

Using the ScriptBuilder Editor

The ScriptBuilder Editor, located on the right side of the ScriptBuilder window, is where you work on your source file. You can have several documents open at the same time in the Editor.

To change the size of the Editor relative to the Toolbox, drag the dividing line between them.

To maximize the size of the ScriptBuilder Editor, hide the Toolbox.

The Editor is described in detail in Setting Editor Options .

Checking Your Documents for Errors

ScriptBuilder includes powerful tools that help you find errors in your documents:

- The syntax checker lets you check your documents for syntax errors before you preview or publish your documents.
- The Script Inspector examines your document and points out any language elements that are specific to Netscape and Microsoft browsers.

The Script Inspector and syntax checker are described in Chapter 9.

Hiding and Displaying the Status Bar

The status bar is located at the bottom of the ScriptBuilder window.

The status bar provides this information:

- **Ln**—the insertion point's line number in the current document
- **Pos**—the insertion point's character position from the left margin
- **Total**—the total number of lines in the current document
- **Modified**—indicates that the current document has been modified since the last time it was saved
- **Insert/Overwrite**—indicates whether the Editor is in Insert mode or Overwrite mode
- **JavaScript/VBScript/Java/Perl**—identifies the current language setting for color syntax highlighting

By default, the status bar is active when you first open ScriptBuilder.

To show or hide the status bar:

- In the View menu, select or deselect Status Bar.

What's New in ScriptBuilder 3.0.

Version 3.0 of NetObjects ScriptBuilder includes several major enhancements:

- **Syntax Checking.** While in the Editor, you can locate and correct syntax errors in your JavaScript and VBScript code, before executing the code in a browser.
- **Script Components.** Script components revolutionize scripting through the application of object-oriented programming principles. You define a script component by creating a standard custom JavaScript (ECMAScript) object and placing it inside a **.js** include file. To describe the component to the outside world, you add a public interface written in XML (eXtensible Markup Language) to the include file. Following these conventions enables you to encapsulate the script into a component wrapper. Other developers can use your components in Web pages and server-side applications, working with the component's public interface instead of the underlying implementation.
- **Script Component Generator.** This assistant imports **.js** files and converts them to script components.
- **Extensible Reference Tab Panels.** The InfoDesk Language Reference is based on an XML architecture and is fully extensible. With this new architecture you can:
  - Add your own reference material to the InfoDesk Reference panel, including reference material describing your own scripting tags and/or DTD's that you want to use in ScriptBuilder.
  - Find tree-based information intuitively with SmartSearch.
  - Only view the reference material you want to see.
- **Editor.** The NetObjects ScriptBuilder Editor has several enhancements:
  - Improved, context-sensitive color syntax highlighting for HTML
  - Drag-and-drop text editing
  - Multi-level undo
  - Customizable AutoScripts
  - Insert and overwrite modes
  - Option to substitute spaces for tabs
- **Internal Preview**. Internal Preview now creates temporary files for your files, so you can preview your work without having to save your changes.
- **Script Development Environment**. You can now:
  - Access language reference in the Script Library and insert into your document.
  - Create folders, rename files, and launch programs from the Desktop panel.
  - Open recently used files easily via a drop-down menu on the toolbar.
  - Edit preferences for the Toolbox.
  - Create custom document templates.
  - Enjoy increased performance and a Windows 98 interface.
- Script Library. You can import and export scripts in the Script Library.

Migrating Your Script Library from Previous Versions

If you have a previous version of NetObjects ScriptBuilder, you probably created entries in the Script Library. With this version of ScriptBuilder you can merge scripts from one Script Library to another.

To merge Script Libraries:

1. In the Toolbox, open the Script Library. See Opening the Script Library.
2. Right-click the Script Library and choose Merge from the shortcut menu.
3. In the Merge Script Libraries dialog, make the following selections:

- **Current file:** Typically this is the library to which you want to merge— your NetObjects ScriptBuilder 3.0 library. The file name is **scriptlib3.sbl**.

  You can change your current library in the Files tab of the Options dialog (Tools menu).

- **Reference file:** Typically this is the library from which you want to merge—your NetObjects ScriptBuilder 2.0 library. The file name is **scriptlib.sbl**.

  After you select the two files, their contents appear in the two Scripts available lists below the file names.

4. To copy a script from one library file to the other, select the script name in the Scripts available list and click the appropriate Copy button.
5. Click Close to close the Merge Script Libraries dialog.

Note:

You can copy in either direction, from the Reference file to the Current file, or from the Current file to the Reference file.

Using the Toolbox

ScriptBuilder's Toolbox, located in the left pane of the ScriptBuilder window, contains six panels of information and tools useful in creating documents.

- The InfoDesk Reference panel is a visual language tree with reference information on ECMAScript, HTML, JavaScript, VBScript, Perl, and others. You obtain language reference information via the InfoBrowser, either locally or from the Internet. You can also use the InfoDesk Reference panel to locate and insert language elements into your documents.
- The Map panel contains a visual tree of the script components and language elements in the current document. You can use it to go directly to components and code elements within the document.
- The Desktop panel is functionally similar to the Windows Explorer. You can search for and manipulate files in the Desktop and open them in the ScriptBuilder Editor.
- The Script Library contains ready-made code snippets that you can use as you need them. You can edit the provided snippets, and you can add your own.
The Script Library is described in Adding a New Script to the Script Library.
- The Component Gallery contains reusable code that is referenced from JavaScript **.js** include files. Components make it possible for you to add scripts to your Web page without having to insert large blocks of code into the file.

  Components and the Component Gallery are described in Chapter 11.

Each panel is identified in the Toolbox by a tab bearing its name.

Floating and Docking Toolbox Panels

Just as you can float and dock the toolbars (see Floating and Docking Toolbars and the Menu Bar), you can float and dock the toolbox panels.

To float a Toolbox panel:

- Drag the top of the panel to the right, toward the Editor, until it becomes a free-floating window.

To dock a floating Toolbox window:

- Drag the window by its title bar until the pointer touches the left side of the ScriptBuilder window. The panel automatically docks to the Toolbox.

Closing and Opening Toolbox Panels

To close a Toolbox panel, do any of the following:

- From the View menu, choose Toolbox Windows and then the name of the panel you want to close.
- If the panel is floating, click its Close box.
- Right-click the tab of a docked Toolbox panel and on the shortcut menu, click the name of the panel you want to close.

To open a Toolbox panel:

- From the View menu, choose Toolbox Windows and then the name of the panel you want to open.
  If the Toolbox is closed (hidden), you must use the Toolbox Windows command to open a panel.
- Right-click the tab of a docked Toolbox panel and on the shortcut menu, click the name of the panel you want to open.

Changing the Order of Toolbox Panels

To change the order in which the panels appear in the Toolbox:

1. Right-click the tab of any Toolbox panel and choose Customize Toolbox from the shortcut menu.
2. In the Customize Toolbox dialog, drag the panels into the order you want.

Using the InfoDesk Reference Panel

The InfoDesk Reference panel is both a language reference and a drag-and-drop source for inserting language objects into your documents.

Using the InfoDesk Reference panel is an excellent way to learn scripting languages. The tree format illustrates the hierarchy for the objects it contains, making it easy to find specific properties or methods by traversing through the branches.

For a discussion of the InfoDesk Reference panel as a language reference source, see Using the InfoBrowser for Language Reference. For a discussion of using the InfoDesk Reference panel to add language elements to your documents, see Inserting Language Elements from the InfoDesk Reference Panel.

Selecting What Appears in the InfoDesk Reference Panel

NetObjects ScriptBuilder comes with language documentation that is installed onto your hard drive, as well as access to Web-based language reference. However, although all this language reference is accessible, you have the final choice of what language reference appears in the InfoDesk Reference panel. By customizing which reference materials appear, you optimize performance of the InfoDesk Reference files.

1. Right-click the InfoDesk Reference panel.
2. Choose Settings from the shortcut menu.
3. In the Settings dialog, check the language reference you want to appear in the InfoDesk Reference panel. Uncheck any language reference you don't want to appear.
4. Click OK.

Using the InfoBrowser for Language Reference

The InfoBrowser is a Web browser inside ScriptBuilder that, in conjunction with the InfoDesk Reference panel, accesses local and Web-based language documentation.

Note:

You must have Microsoft Internet Explorer 3.02 or greater installed to enable internal browser language reference.

To use the InfoBrowser as a scripting language reference:

1. Open the InfoDesk Reference panel in the Toolbox.
2. Click the language tree node for which you want information.

   ScriptBuilder loads the language reference.
3. Expand the InfoDesk Reference tree (see Expanding and Collapsing Nodes in the Reference and Map Panels) until you find the language element you want.
4. Double-click the language element to open the documentation in the InfoBrowser.

Note:

The InfoBrowser can also run as a Web browser from within the ScriptBuilder environment. See Using the InfoBrowser as a Web Browser.

Configuring the Language Reference

To configure your language reference:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the InfoDesk Reference tab.

   By default, ScriptBuilder is configured to view language reference in the internal browser using local sources where available.
3. To view documentation in your selected external browser, select View in default external browser. See Configuring External Browsers.
4. To view language reference on the Web, even if local language reference is available, select Always use Web as source.
5. Click OK

Viewing Component and Document Structure with the Map Panel

The Map panel gives you a visual tree view of the following elements of your current document:

- Script Components
- Functions
- Subroutines
- Tags

To see the properties, methods, and events in a script component, right-click the Components pane and choose the appropriate command from the shortcut menu.

You can use the Map panel to jump quickly to specific locations in your file. Especially with large files, using the Map panel is quicker and more efficient than scrolling the document.

To go directly to a function, object, or tag:

1. Locate the function or object in the Map panel.
2. Double-click the node, or right-click the node and choose Go to Line from the shortcut menu.

   To go to the line for a node that has children, you must right-click and choose Go to Line. Double-clicking expands and collapses nodes with children.

   ScriptBuilder positions the insertion point in the function's or object's location in the current document.

Note:

If you define a custom object, its constructor is represented as a function in the Map panel, not as an object. Custom objects and built-in language objects are not represented in Map view.

Expanding and Collapsing Nodes in the Reference and Map Panels

The Reference panel's language tree and the Map panel both work much like the Windows Explorer, and you can expand or collapse individual nodes of the trees.

To expand a node:

- Double-click the node, or click the plus sign to the left of the node's name.

To collapse a node and its children, do either of the following:

- Double-click the node, or click the minus sign to the left of the node's name.
- Right-click the panel and choose Collapse, Node and Children from the shortcut menu.

To collapse the entire tree:

- Right-click the panel and choose Collapse, All Nodes from the shortcut menu.

Managing Files with the Desktop Panel

You can use the Desktop panel in the Toolbox to open documents in the ScriptBuilder Editor. You can set the Desktop panel to open to a default folder and to filter out (by extension) all but the types of files you want to use while you work with ScriptBuilder.

Opening a File from the Desktop Panel

You can use the Desktop panel instead of the File menu's Open command to locate and open files in the ScriptBuilder Editor.

To open a file from the Desktop panel, do either of the following:

- In the Desktop panel, double-click the file name, or right-click and select Open from the shortcut menu.
- Drag the file from the Desktop panel and drop it into the Editor pane.

Changing the Desktop Panel View

Functionally, the Desktop view is like the Windows Explorer. You can view the folders and files as large icons, small icons, a simple list, or as a list with size and type details.

To change the Desktop view:

1. Right-click the Desktop panel, and choose View from the shortcut menu.
2. Select the way you want to view folders and files from the shortcut menu.

Specifying a Default Folder from the Desktop

You can customize your ScriptBuilder Desktop by specifying a default folder, which is the one that will be opened in the Desktop panel when you start ScriptBuilder. Its name appears in the top field of the Desktop panel, with its contents listed in the window below.

To specify the default folder, do one of the following:

- Set the Desktop view to the folder you want as the default. Right-click the Desktop panel and choose Set as Default Folder from the shortcut menu.
- From the Tools menu, choose Options to open the Options dialog. Click the Files tab. Enter the default folder's path in the Default Folder field or click Browse to locate the folder you want to set as the default.

Customizing File Filters for the Desktop

You can further customize the Desktop view by defining which types of files are listed. By filtering out unwanted files, you can easily locate the files you want to work on in ScriptBuilder.

To customize file filters:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Files tab.
3. In the File types to be displayed field, enter the extensions for the files you want to see on the desktop.

   For example, to see only HTML files, enter ***.html**. Separate multiple file extensions with a semi-colon followed by no space, as in this example:

   .html;.js;*.asp
4. Click OK.

Working with Documents

Document management in ScriptBuilder is similar to working with documents in a word processor or code editor. ScriptBuilder also includes templates for common code formats. Creating, opening, saving, and printing documents in ScriptBuilder uses a familiar interface design that makes your work easier and faster.

Creating Documents

ScriptBuilder provides six templates for new documents:

- **Default HTML File.html**
- **JavaScript Include File.js**
- **Microsoft Active Server Page File.asp**
- **Netscape LiveWire File.html**
- **Script Component Class File.scc**
- **Text File.txt**

To create a new document from one of the templates:

1. From the File menu, choose New or press Ctrl+N.
2. In the New dialog, double-click the template icon, or highlight the template icon and click OK.

 You can also create a new document by clicking the New button on the toolbar. The next section describes how to configure the New button for one of the document templates.

Setting the Default New Document Type

You can set one of the provided document types—or a custom document type you create (see Creating and Using Custom Document Templates)—as the default type for the New button on the toolbar.

To configure the New button:

1. From the File menu, choose New or press Ctrl+N.
2. In the New dialog, select one of the document templates.
3. Click the Set Default button near the bottom of the dialog.
4. Click OK in the confirmation dialog.
5. Click OK in the New dialog.

Clicking the New button now automatically starts a new document of the selected type.

Opening Documents

You can open documents on your local system or network.

To open a document, do one of the following:

-  Click the Open button on the Standard toolbar, choose Open from the File menu, or press Ctrl+O. Locate the file in the Open dialog and double-click it.

To see files of other types, display the drop-down list in the Files of type field.

- Display a list of recently opened documents by clicking the arrow on the Open button or by choosing Recent Files from the File menu. Select the file you want to open.
- Locate the file in the Desktop panel of the Toolbox and double-click it, or drag it from the Desktop panel and drop it in the Editor.

Saving Documents

You can save documents to your local system or network.

To save a document to its current location or a new document you haven't yet named and saved:

-  Click the Save button on the Standard toolbar, choose Save from the File menu, or press Ctrl+S.

To save a document under a new name or extension, or to save it to a different location on your local system or network:

1. Choose Save As from the File menu.
2. In the Save As dialog, locate the folder where you want to save the file.
3. Enter a name in the File name field.
4. To change the file type, click the Files of type drop-down list, and select a file type from the list.
5. Click Save.

Selecting File Format

You can save your documents in any of three file formats: PC (Windows), UNIX, or Macintosh.

To select a default file format:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Files tab.
3. Select the file format in the Default file format panel.
4. Click OK.

Now all files you open and save will be saved with this file format.

Saving All Open Documents

If you have several documents open in the ScriptBuilder Editor, you can save them all at the same time.

To save all open documents:

- From the File menu, choose Save All.

Printing Documents

To print a document, do either of the following:

-  Click the Print button on the Standard toolbar.
- From the File menu, choose Print, or press Ctrl+P.

Deploying Documents Containing Script Components

If your document includes script components (see Working with Script Components), you must use the ScriptBuilder Deploy command to place the document in another location on your local system or network. Deploying ensures that the document, regardless of its location, can access its associated script components.

To deploy your document to another location:

1.  From the Tools menu, choose Deploy, or click the Deploy button on the Standard toolbar.

   ScriptBuilder opens the Component Deploy dialog.

   The list box under Files to be published lists the names and paths of the document and any components that are included in it.
2. Choose a destination for the deployed files.

   By default, the files will be deployed to the current folder, which would then place the document and component files in the same location. If you choose Current, you can click OK, and the deployment is finished.

   You can also deploy files to Local, which is any folder on your system or on a network to which you are connected.
3. If you choose Local, ScriptBuilder opens the Browse for Folder dialog. Use its Windows Explorer-style tree to find the destination folder.
4. Click OK in the Browse for Folder dialog to return to the Component Deploy dialog.
5. Verify that the correct folder path appears in the Directory field.
6. Click OK in the Component Deploy dialog to complete the deployment.

Note:

Deploy works only on the document file and associated script components. If the document contains images, you must maintain the relative path between the document and images, whether on a local system, network, or remote server. See Images.

Creating and Using Custom Document Templates

Using the ScriptBuilder Editor, you can create your own custom document templates. A template can be any ASCII-based text file such as HTML, JavaScript, ASP, Script Component Class, or text.

To create a new document template:

1. Create a new document, as described under Creating Documents.
2. Add tags and language elements to create the template you want.
3. From the File menu, choose Save As, and save the template to the custom templates folder. The default path for this folder is:

   C:\Program Files\NetObjects ScriptBuilder\Templates\Custom

To use a custom template:

1. From the File menu, choose New.
2. In the New dialog, click the Custom tab and select the custom template.

Setting Editor Options

The Editor tab of the Options dialog contains these options:

- Default editor font. You can specify the font that appears in the Editor. The font the ScriptBuilder Editor uses can be any font available on your system.
- Alternate editor font. You can highlight text in the Editor by setting it to another font. For example, you might use Arial as the default font and Times New Roman as the alternate font, applied to text such as comments. See Setting Highlighting Colors and Font Attributes.
- Screen font. See Setting Screen Font.
- Auto indent. When you indent a line with auto indent enabled, the insertion point automatically returns to the same indent at the start of the next line.
- Word wrap. The Editor automatically wraps your text to fit the current window size.
- Drag and drop text editing. See Moving Text with Drag and Drop.
- Use syntax highlighting. See Highlighting Language Elements.
- Enable AutoScripting. See Entering Language Elements Automatically.
- Maximize document on open. You can specify that documents will open in a window or will occupy the entire Editor area.
- Tab stops. You can set the size of a tab stop and whether it is entered as a Tab character or as spaces. When you're writing code, it helps to have fairly small tab stops—for example, every three characters.

To set Editor options:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Editor tab.
3. In the Fonts section, select fonts and point size. In the Editor Options section, check the options you want to enable. In the Tabs section, set a tab size and specify whether tabs are entered as Tab characters or as spaces.
4. Click OK.

Setting Screen Font

You can set a font for the following ScriptBuilder screen elements:

- Drop-down lists
- Toolbox
- Status bar
- Dialogs

To set the screen font:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Editor tab.
3. Select the font from the Screen font drop-down list.
4. Click OK.

Highlighting Language Elements

To make your documents easier to read and work with, you can ''highlight different language elements with a combination of color syntax highlighting and font attributes. You can specify colors for the text and its background, in addition to an alternate font and attributes such as bold and italic text.

Selecting the Highlighted Language

To apply color syntax highlighting to language elements in JavaScript, VBScript, Java, or Perl:

- From the View menu, choose Highlighted Language and choose a language from the submenu.

Color syntax highlighting of scripts only occurs within Script blocks in those languages.

HTML language elements are automatically highlighted, regardless of the selected language.

Note:

When you develop with LotusScript, use VBScript as the highlighted language.

Setting Highlighting Colors and Font Attributes

You can set off elements in the highlighted languages with color and text attributes such as Bold or Italic.

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Editor tab, and make sure Use syntax highlighting is selected.
3. Click the Colors tab.
4. From the Text elements list box, select a text element to which you want to apply color syntax highlighting or a font attribute.
5. In the Text color list, select the text element's color.
6. In the Background color list, select a background color for the text element.
7. Check the font attributes—Bold, Italic, Underline, Strikeout, or the Alternate editor font—you want to apply to the text.
8. Click OK.

Working with AutoScripts

ScriptBuilder's AutoScripting makes it quicker and easier to type frequently used statements. With AutoScripting enabled, you need only enter a keyword as you type, press the spacebar, and ScriptBuilder enters the rest of the statement for you.

AutoScripting is language-sensitive. Typing for when JavaScript is the current language produces a different statement from what you get when VBScript is the current language. The keywords, extended expressions, and languages appear in the AutoScripting dialog.

To use AutoScripting:

1. If you are using a scripting language such as JavaScript, place the insertion point inside a Script block. Language AutoScripting works within Script blocks, while HTML AutoScripting works outside Script blocks.
2. Type one of the AutoScripting keywords and press the spacebar.

ScriptBuilder inserts the AutoScript represented by the keyword into your document.

Editing an Existing AutoScript

To edit an existing AutoScript:

1. From the Tools menu, choose AutoScripting.
2. In the AutoScripting dialog, select an AutoScript and click Edit.
3. In the Edit AutoScripting Keyword dialog, edit the AutoScript and click OK.
4. To specify that text will be selected when AutoScripting completes the insertion, select the text in the Edit AutoScripting Keyword dialog and click Set Selection.

   This is typically variable text, which you will replace as soon as the AutoScript is inserted into your file.
5. Click OK.

Adding a New AutoScript

To add a new AutoScript:

1. From the Tools menu, choose AutoScripting.
2. In the AutoScripting dialog, click Add.
3. In the Add AutoScripting Keyword dialog, type an AutoScripting keyword in the Keyword field.
4. Select the language from the drop-down list.
5. Type the text in the Text to Insert box.
6. To specify what text will be selected when AutoScripting completes the insertion, select the text in the Add AutoScripting Keyword dialog and click Set Selection.

   This is typically variable text, which you will replace as soon as the AutoScript is inserted into your file.
7. Click OK.

Deleting an AutoScript

To delete an AutoScript:

- In the AutoScripting dialog, select the AutoScript you want to delete, and click Delete.

Moving Text with Drag and Drop

1. Use the mouse to select a block of text.
2. Move the pointer into the selected block and hold down the mouse button.

   A rectangle appears under the tail of the pointer, indicating that the text can be moved.
3. Without releasing the mouse button, move the pointer to the text block's new location, indicated by the blinking text insertion point.
4. To scroll the Editor window, move the pointer into the first toolbar immediately above the window or the status bar below the window.

   You can vary the scrolling speed. The further you move the pointer into the first toolbar or into the status bar, the faster the window scrolls.
5. Release the mouse button to complete moving the text block.

Inserting Language Elements from the InfoDesk Reference Panel

The Toolbox's InfoDesk Reference panel is a ready source for language elements that you can drag and drop into your documents.

To add a language element from the InfoDesk Reference panel:

1. Open the Toolbox, if necessary, and click the Reference tab.
2. Click the name of the type of language element you want to add—for example, HTML 4.0 Reference.

   ScriptBuilder loads the language reference, if it's not already loaded.
3. Expand the language tree nodes until you see the language element you want to add. For example, to add an <HTML> tag, expand the Tags for Document Structure node.
4. Drag the language element from the InfoDesk Reference panel to your document.

   If you drag a language element with start and end tags—such as BLOCKQUOTE—to selected text in the Editor, ScriptBuilder puts the start and end tags on either side of the selected text.

If you're not sure where the language element is, you can search the InfoDesk Reference panel.

1. To start your search from a specific node, select that node first.
2. Click the triangular Show search box button in the upper right corner of the InfoDesk Reference panel to display the Enter Text field and the Find text in tree button.
3. In the Find field, enter text to search for.
4. Click the Find text in tree button, or press Enter.

   ScriptBuilder starts from the currently selected node and highlights the first occurrence of the element you selected. During this process it opens the nodes and children along the path to the element.

   You can now search for language elements based on their object hierarchy, such as window.open and document.write.

   The search finds the first node that satisfies the window criterion and then searches its child nodes for open.
5. If the first element is not the one you were looking for, click the Find text in tree button to find the next instance.

Formatting Text from the Toolbar

Using the HTML Tags toolbar, you can insert the following text formatting tags into your documents:

- Text style: Bold, Italic, and Underline
- Text alignment: Left, Center, and Right
- Lists: Unordered and Ordered
- Paragraphs and line breaks
- Horizontal lines
- Headings, level 1 through 6

To insert style and alignment tags:

1. Select the text you want to tag.
2. Click the appropriate button on the toolbar.

   ScriptBuilder inserts the start and end tags—for example, <B> and </B>—on either side of the selected text.

   You can set selected text to Bold, Italic, and Underline from the keyboard—using the keystrokes Ctrl+B, Ctrl+I, and Ctrl+U, respectively.

You can also insert the tags first, and then add the text to be formatted between the open and close tags.

To add lists:

1. Enter the list text, with each item on a separate line.
2. Put a list tag (<LI>) at the beginning of every line.
3. Select the list text, including all tags.
4.  For a bulleted list, click the Unordered List (<UL>) button.

   For a numbered list, click the Ordered List (<OL>) button.

To insert paragraph breaks, line breaks, and horizontal lines:

- Place the insertion point where you want the tag and click the appropriate button.

To format text as a heading:

- Select the heading text and click the appropriate button.

  ScriptBuilder inserts the start and end tags—for example, <H1> and </H1>—on either side of the selected text.

Working with Multiple Editor Windows

When you open multiple documents in the ScriptBuilder Editor, you can display the next document by pressing Ctrl+Tab and the previous document by pressing Ctrl+Shift+Tab, by choosing Next Document and Previous Document from the Window menu, or by clicking the Next Document and Previous Document buttons on the Window List toolbar.

You can also select a window from the drop-down list in the Window List toolbar or from the Window menu.

Finding and Replacing Text

In ScriptBuilder, as in a word processor, you can search for text in the current document. You can also replace the search text with new text, either globally or one instance at a time.

The Find and Replace dialog contains two tabs, one for finding text and the other for replacing text.

To find or replace text:

1. To have ScriptBuilder search only a portion of your document, select it now.
2. To find text, choose Find from the Edit menu (Ctrl+F) or right-click the Editor window and choose Find from the shortcut menu.

   ScriptBuilder displays the Find tab of the Find and Replace dialog.
3. To replace text, choose Replace from the Edit menu (Ctrl+H) or, if the Find and Replace dialog is already open, click the Replace tab.

   ScriptBuilder displays the Replace tab of the Find and Replace dialog.
4. In the Find what field, enter the text you want to find or replace.
5. If you're replacing text, enter the new text in the Replace with field.
6. In the Options section:

- Select the Match whole words only check box to search only for the instances where the text you entered occurs as a whole word.
- Select the Match case check box to search for text in the same case you entered in the Find what field. If you do not select this check box, ScriptBuilder finds the text regardless of the case of the letters.

7. In the Search section:

- Choose Down to search from the insertion point's current location to the end of the document.
- Choose Up to search from the insertion point's current location to the beginning of the document.
- Choose Selected Text to search only the selected text.
- Choose All to search the entire document.

8. Click Find Next to begin searching the selected text or document for the text you entered.

   When ScriptBuilder finds an instance of the search text, it highlights the text.
9. If you're on the Replace tab, click Replace to replace the text one occurrence at a time. ScriptBuilder replaces the highlighted text and moves to the next occurrence of the search text.

   To replace all occurrences of the search text at once, click Replace All.
10. To move to the next occurrence of the text, click Find Next again.
11. When ScriptBuilder notifies you the search is complete, click OK to confirm and then click Close in the Find and Replace dialog.

Using the Find Toolbar

The Find toolbar provides you with a quick way to search for text in the current document. ScriptBuilder searches for occurrences of the text regardless of case.

1. In the Find toolbar text box, enter the text you want ScriptBuilder to search for.
2.  To search from the insertion point's current position toward the end of the document, click the Find Next button.

 To search from the insertion point's current position toward the beginning of the document, click the Find Previous button.
When ScriptBuilder finds an instance of the text, it highlights the text in the document.

3. To find the next occurrence of the text, click the Find Next or Find Previous button again.

 You can also use the Find and Replace buttons on the Find toolbar to open the Find and Replace dialog.

Finding Text in Files in Your Computer

You can search for text in unopened files using ScriptBuilder. ScriptBuilder displays the results of your search in Results View. See Displaying Results Lists in Results View.

1.  Display the Find in Files dialog by doing one of the following:

- Click the Find in Files button on the Find toolbar.
- From the Edit menu, choose Find in Files.

2. In the Find what field, enter the text you want to find in the files, or select from prior entries.
3. In the In file types field, enter the file extensions of the types of files you want to search, or select from prior entries.
4. In the In folder field, enter the path to the folder you want to search (the target folder), click the Browse button to browse for the folder, or select from prior entries.
5. In the Options section:

- Select the Match case check box to search for text in the same case you entered in the Find what field. If you do not select this check box, ScriptBuilder finds the text regardless of the case of the letters.
- Select the Look in subfolders check box to search for the text in the target folder's subfolders.

6. Click Find.

ScriptBuilder searches the folders you indicated, and displays the results of the search in Results View.

Displaying Results Lists in Results View

ScriptBuilder displays the results of Find in File searches in Results View, a pane that appears at the bottom of the ScriptBuilder desktop.

Each search has its own numbered tab. For example, the Find in Files 2 tab contains the results of the second Find in File search of the current ScriptBuilder session.

You can have up to three search list tabs at a time. To move between different searches, click the corresponding tabs.

Results View displays:

- The path to the folder you searched, and the text you searched for
- The path to each file where ScriptBuilder found at least one match
- The total number of matches found in the folder

Opening a File from Results View

To open a file in the Editor from Results View, do either of the following:

- Double-click the file's path.
- Right-click Results View, and choose Go to Line from the shortcut menu.

Clearing Find in File Results Lists

To clear or delete the results of a Find in File search:

1. Right-click Results View.
2. Select Clear All from the shortcut menu.
3. Click Yes to confirm.

ScriptBuilder clears all searches from the results list.

Closing Results View

To close the Results View, do one of the following:

-  Click the Results View button on the Standard toolbar.
- Click the close button in the upper right corner of Results View.
- From the View menu, deselect Results View, or press F7.

Adding Language Elements to Documents

ScriptBuilder makes it easy for you to insert code and language elements into your document.

The InfoDesk Reference panel of the Toolbox contains a visual language tree of ECMAScript, Java, JavaScript, JScript, VBScript, Perl, ASP, and HTML language elements, among others. You can locate elements, methods, or properties you want to use in your document and insert them directly into your file. See Inserting Language Elements from the InfoDesk Reference Panel.

Inserting from the Toolbars and Menus

In addition to the InfoDesk Reference, you can use the buttons and commands in the ScriptBuilder toolbars and menus to insert language elements—even entire files—to your documents. You will learn about the language elements contained in the Form Elements and Script toolbars and the Script and Tags menus. Where a menu command has a hotkey equivalent, it appears in parentheses immediately following the command.

Script Blocks

To insert scripting language code into your HTML document, you must first use the HTML <SCRIPT> tag to define which language or languages you are going to use. <SCRIPT> statements are normally used in the <HEAD> section of the document, although they can be placed in the <BODY> section.

The <SCRIPT> tag has two attributes—LANGUAGE and SOURCE. Use the LANGUAGE attribute if you want to write script statements directly into your HTML document; use the SOURCE attribute to name a separate file where you are writing your script code. This file must have a .js extension. The advantage of the separate file is that you can modify your script code without risking unwanted changes to your HTML file.

Note:

JScript 1.0 and JavaScript 1.0 do not support **.js** files

You can use more than one <SCRIPT> tag in a document. Therefore, you can use multiple scripting languages or provide alternatives to site visitors whose browsers may not support a particular scripting language.

To insert a Script block:

1. Position the insertion point where you want to place the <SCRIPT> block.
2.  Click the Script Block button on the Script toolbar or choose Script Block from the Script menu (Ctrl+F7).

No Script Blocks

Netscape Navigator and Microsoft Internet Explorer now support the use of the <NOSCRIPT>...</NOSCRIPT> container in HTML, which lets you write content for site visitors who disable the scripting capabilities of their browsers, or who have non-scriptable browsers. Use the No Script block in the body section of your HTML document to deliver a message to these visitors.

To insert a No Script block:

1. Position the insertion point in the document where you want to place the No Script block.
2. From the Script menu, choose No Script Block.

Server Blocks

To add a Server block:

1. Position the insertion point where you want to place the Server block.
2.  Click the Server Block button on the Script toolbar or choose Server Block from the Script menu (Ctrl+F10).

The Server block tags can take two forms, <% %> and <SERVER> </SERVER>. To specify the form of the Server block tags:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Code tab.
3. In the Code tab, make your selection from the Tag Types drop-down list.
4. Click OK.

If you are writing ASP server-side scripts, you must specify the language you're using before you insert the Server block.

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Code tab.
3. Under Syntax checker, select VBScript or JavaScript from the ASP Language drop-down list.
4. Click OK.

Function Blocks

Functions are the building blocks of a script and can be called on to execute at any time before or after a site visitor views an HTML document. Functions are essential in HTML documents as they perform the tasks of reading or taking in data, performing operations on data, or displaying and sending out data. You can avoid writing the same code repeatedly by creating and naming a function in the beginning of your HTML document. Then, every time you need to perform the function's operation, all you need to do is name (or call) the function.

Note:

In HTML documents, be sure to enclose all function declarations within <SCRIPT> tags.

In ScriptBuilder, a function block includes both the beginning and ending code for the function itself, as well as a comment block naming the function and providing a description of its purpose.

To insert a function block:

1. Position the insertion point in the document where you want to place the function block.
2.  Click the Function Block button on the Script toolbar or choose Function Block from the Script menu (Ctrl+F9).
3. In the Function Name field of the Function Block dialog, enter a name for the function.
4. In the Description field, enter a description of the function.
5. Click OK.

ScriptBuilder inserts the function block, including a commented description, into your document.

To include your user name and creation date in your function description:

1. From the Edit menu, choose Options.
2. In the Options dialog, click the Code tab.
3. In the Function blocks section of the Code tab, check Include username in function description and Include creation date in function description. Enter the user name into the Username field.
4. Click OK.

Note:

The appearance of the function block depends on the highlighted language. JavaScript and VBScript function blocks are different.

Comment Blocks

Comments are an essential form of communication between a programmer and anyone who must read or maintain the program code. Here are some guidelines for writing good comments:

- Include author, date of last update, purpose of the function, and last changes.
- Write comments about code that the source code itself can't easily explain.
- Write comments in the appropriate location—for example, right before or after a line of code—to explain the purpose of the code, as well as how it does its processing.
- Comment on global variables—what they contain and where they get updated.
- Comment on unusual or hard-to-follow pieces of code—what they do and why you used them.

Comments are useful in identifying and fixing errors in code. You can enclose lines of code in comments to block them out. Then reload the page in the browser, note the result, and modify the code.

Note:

The elements of comment blocks—delimiters and the like—are specific to the current highlighted language (JavaScript, VBScript, Java, or Perl), even if the comment is contained within HTML.

To insert a comment block:

1. Position the insertion point in the document where you want to place the comment block.
2.  Click the Comment Block button on the Script toolbar or choose Comment Block from the Script menu (Ctrl+F8).

To enclose lines of code in a comment:

1. Highlight the code you want to enclose.
2. Click the Comment Block button on the Script toolbar.

ScriptBuilder encloses the highlighted code in comment tags.

Note:

The appearance of the comment block depends on the highlighted language. JavaScript and VBScript comment blocks are different.

Message Boxes

A message box is a scripting method for communicating with site visitors. There are three types of message boxes:

- The alert message box pops up a short message to the visitor.
- The confirm message box is similar to the alert box, but it provides an OK and Cancel response to the visitor.
- The prompt message box asks the visitor for a response, in addition to providing the OK and Cancel response.

To insert a message box:

1. Position the insertion point in the document where you want to place the message box code.
2.  Click the Message Box button on the Script toolbar or choose Message Box from the Script menu.
3. In the Message Text field of the Message Box dialog, enter the text of the message you want to display in the message box.
4. In the Message Box section, select the button corresponding to the type of message box you want to create.
5. If the message box is of the **prompt** type, enter text in the Default Reply field. It appears in the message box's prompt field.
6. Click OK.

The Message Box dialog inserts code that is JavaScript-specific.

You can preview the message box by switching to Preview Mode. See Previewing Documents.

Custom Objects

Custom objects store both data (similar to arrays) and behaviors. For example, you could create a custom object to keep track of a CD collection. You could then track the CD's title, artist, type of music, and release date using this object.

A custom object is composed of properties (data) and methods (behaviors or responses to messages). To use a custom object, you first need to create a constructor. A constructor is a function (or template) describing what the object looks like and how it acts. To use the CD example, the constructor would consist of a function defining the variables for each CD in the catalog—the CD's title, artist, type of music, and release date. Next, you would create a function to display the CD catalog information to the site visitor.

ScriptBuilder creates an object constructor when it inserts a custom object.

To insert a custom object:

1. Position the insertion point in the document where you want to place the custom object.
2.  Click the Custom Object button on the Script toolbar or choose Custom Object from the Script menu.
3. In the Object Name field of the Custom Object dialog, enter a name for the object.

   Avoid names that create hard-to-read code. For example, if you're creating an object to represent an invoice, call it **invoice**, as opposed to **createInvoice**, to avoid confusion once you start defining properties and methods.
4. In the Object Elements section, build the custom object by defining its properties and methods.
5. To add a property:
   a. Click the Property button.
   b. Enter the name of the property or data item.
   c. Click Add.

   ScriptBuilder adds the property to the custom object and to the tree in the Object Elements section.

   Continue to add properties.
6. To add a method:
   a. Click the Method button.
   b. Enter the name of the method or behavior.
   c. Click Add.

   ScriptBuilder adds the method to the custom object and to the tree in the Object Elements section.

   Continue to add methods.
7. Click OK.

ScriptBuilder creates the code for the custom object constructor in the form of a function, and defines the properties and methods you specified. ScriptBuilder also creates a comment block defining the function as an object constructor, and naming it. To proceed, you must add the code for the methods.

Window.open() Method

The window.open() method defines the characteristics of a new window in an HTML document. You create new windows when you want the browser to display a secondary window that could display text to the site visitor or lead to additional information.

To insert a window.open() method:

1. Position the insertion point in the document where you want to place the window.open() method.
2.  Click the Window.open() button on the Script toolbar or choose Window.open() from the Script menu.
3. To display a Web page in the window, add its URL to the Window.open() dialog.
4. In the Window Title field, enter a name for the window.
5. In the Options section, select the window's display attributes:

Attribute	Description
Toolbar	Shows/hides the toolbar
Menubar	Shows/hides the menubar
Scrollbars	Shows/hides the horizontal and vertical scrollbars
Resizable	Allows/disallows resizing of the window
Status bar	Shows/hides the status bar
Location	Show/hides the URL location box
Directories	Shows/hides a secondary toolbar (Netscape)
Copy History	Copies the current window's Go history for new window

6. In the Height field, specify the height of the new window in pixels.
7. In the Width field, specify the width of the new window in pixels.
8. Click OK.

Document.write() Method

The document.write() method instructs the browser to display text in a window. To create a document.write() method:

1. Position the insertion point in the document where you want to place the document.write() method.
2.  Click the Document.write() button on the Script toolbar or choose Document.write() from the Script menu.

After ScriptBuilder inserts the document.write() method, you must add the text you want to display.

Basic HTML Tags

You can save time and increase accuracy in your documents by using ScriptBuilder to insert HTML formatting tags. ScriptBuilder inserts both the beginning and (if necessary) end tags.

To insert a basic HTML formatting tag:

1. Position the insertion point in the document where you want to place the HTML tag.
2. Insert the HTML tag by doing one of the following:

- Click the buttons in the HTML Tags toolbar.
- Drag the tag from the HTML section of the InfoDesk Reference in the Toolbox.
- From the Tags menu, choose the tag you want to insert, or choose Text Formatting to display this dialog:

  The options in this dialog are also available from the InfoDesk Reference in the Toolbox, under HTML, Text Characteristics.

Form Elements

Forms provide a way for Web pages to interact with site visitors. They prompt the visitor for information and carry out tasks based on the input. Forms contain one or more input elements such as radio buttons or text boxes. Once the visitor submits the data, the form collects it and sends it to the destination specified in the form element.

To insert an HTML form element:

1. Position the insertion point in the document where you want to place the form element.
2. Display by doing either of the following:
3.  Click the Form button on the Form Elements toolbar or choose Form Elements, Form from the Tags menu.
4. In the Name field of the Form dialog, enter the name of the form.
5. In the Target Window field, specify the window to which the server should send back information. If you don't select a window, the server displays results in the same window that submitted the form.
6. In the Action (Server URL) field, enter the URL to which you want to send the form's contents. This is usually a CGI program or a LiveWire application.
7. In the Encoding Type field, specify the MIME encoding of the form. MIME is an Internet standard defining how file formats (except plain text) can be passed in Internet mail messages.
8. In the Method section, select either Get or Post to specify how to send the form's data to the server. The default is Get.

- Get appends the arguments to the action URL and opens it as if it were an anchor.
- Post sends the data via an HTTP post transaction.

9. Click OK.

Buttons

Pushbuttons are probably the most common user interface components found in today's graphical operating environment. HTML has three types of buttons—button, submit, and reset. Each button type serves a different purpose:

- The button type has no predefined behavior built into it. For this element to do anything, you must add an onClick event handler.
- The submit button submits the form in which it is contained to the server, based on the parameters of the form. Because the button's behavior is built into the element, it doesn't require code to perform the action.
- The reset button clears the values in the fields of the current form, restoring default values. Like the submit button, it doesn't require code to perform the action.

To insert an HTML button:

1. Position the insertion point in the document where you want to place the button.
2.  Click the Button button on the Form Elements toolbar or choose Form Elements, Button from the Tags menu.
3. In the Name field of the Button dialog, enter a unique name for the button.
4. In the Value field, enter the label text you want to appear on the button.

   The text on a button determines its length. To increase the width of a label, add spaces around the text.
5. In the Button Type section, select the type of button (Submit, Reset, or Button) you are creating.
6. Click OK.

ScriptBuilder inserts the button into your document. If you selected the Button type, you must add methods and event handlers to make it functional.

Checkboxes

The checkbox element acts as a toggle switch that the site visitor (or your code) can either turn on or off.

To insert a checkbox:

1. Position the insertion point in the document where you want to place the checkbox element.
2.  Click the Checkbox button on the Form Elements toolbar or choose Form Elements, Checkbox from the Tags menu.
3. In the Name field of the Checkbox dialog, enter a unique name for the checkbox element.
4. In the Value field, enter a unique value or number for the checkbox element.
5. In the Text to Display field, enter the text to appear next to the checkbox.
6. Select Checked if you want the box to be checked when the form first loads.
7. Click OK.

ScriptBuilder inserts the checkbox element. You must add methods and event handlers to make the checkbox element functional.

Radio Buttons

Radio buttons let the site visitor select one option from a group of options. The visitor cannot select more than one option at the same time. The act of clicking a radio button automatically deselects any other previously selected button.

To insert a radio button:

1. Position the insertion point in the document where you want to place the radio button element.
2.  Click the Radio Button button on the Form Elements toolbar or choose Form Elements, Radio Button from the Tags menu.
3. In the Name field of the Radio Button dialog, enter the name of the set of radio buttons. Each radio button element within a group must use the same value in the Name field.
4. In the Value field, enter a unique value for the radio button (such as 1, 2, or 3) to distinguish it from others in the group.
5. In the Text to Display field, enter the text you want to appear next to the radio button.
6. Click Selected if you want the radio button to be selected when the form first loads.
7. Click OK.

ScriptBuilder inserts the radio button element into your document. You must add methods and event handlers to make the element functional.

Select Elements

The select element is one of the most useful and flexible of all the form elements. It performs the same function as radio buttons, but creates a more compact list of items for the site visitor to choose from. There are three types of select elements: a selection list, a scrolling list, and a multiselection scrolling list.

To insert a select:

1. Position the insertion point in the document where you want to place the select element.
2.  Click the Select button on the Form Elements toolbar or choose Form Elements, Select from the Tags menu.
3. In the Name field of the Select dialog, enter a name for the select element.
4. In the Items Visible field, select the number of items you want the site visitor to see at one time in the list.
5. Select the Allow multiple selections checkbox to enable the visitor to choose more than one item from the list at a time.
6. In the Options section, add the items the visitor can select.
   - Click Add to display the Add Option dialog.
   - In the Text to Display field, enter the text for the option.
   - In the Value field, enter a value for the option.
   - Click the Selected checkbox if you want this item to be selected when the form loads.
   - Click OK to add the option to the select element.
   - Repeat this process for each of the select element's options.
7. Click OK.

ScriptBuilder inserts the select element into the document. You must add methods and event handlers to make the element functional.

Text Elements

The text element is the element you use most often in collecting single-line, freeflowing information from the site visitor. To provide space to enter more than a single line of text, see Textarea Elements.

To insert a text element:

1. Position the insertion point in the document where you want to place the text element.
2.  Click the Text button on the Form Elements toolbar or choose Form Elements, Text from the Tags menu.
3. In the Name field, enter a unique name for the text element.
4. If you want text to appear in the text element when the form loads, enter it in the Value field.
5. In the Width field, specifiy the width of the text box in number of characters.
6. In the Max Width field, specify the maximum number of characters you want the text element to hold, up to a maximum of 256 characters.
7. Select the Password Field check box to hide the site visitor's entry in the text element.
8. Click OK.

ScriptBuilder inserts the text element into the document. You must add methods and event handlers to make the element functional.

Note:

To place a label above a text element, create an HTML paragraph tag (<P>) in the document above the element, and enter text.

Textarea Elements

Textarea elements capture information that does not lend itself to simple text fields, radio buttons, or selection lists. You can use the Textarea element to provide the site visitor with a means to enter free-form data spanning several lines.

Textarea elements display ASCII text, but basic formatting such as paragraphs is possible.

By default, the text in a Textarea element does not wrap. You can, however, set text wrapping options with the wrap parameter.

To insert a Textarea element:

1. Position the insertion point in the document where you want to place the Textarea element.
2.  Click the Textarea button on the Form Elements toolbar or choose Form Elements, Textarea from the Tags menu.
3. In the Name field of the Textarea dialog, enter a unique name for the Textarea element.
4. If you want text to appear in the Textarea element when the form loads, enter it in the Text to Display field.
5. In the Rows fields, set the number of rows you want the Textarea element to display.
6. In the Columns field, set the number of columns you want the Textarea element to display.

   One column equals the width of one text character.
7. In the Word Wrap section, select the radio button corresponding to the option you want for the Textarea element:

- Off is the default setting. There is no word wrapping.
- Virtual word wrapping means that lines wrap on the screen, but a new line begins only when the visitor actually presses the Enter key.
- Physical word wrapping means that lines wrap on the screen, but the browser automatically adds a carriage return at the end of each line when the text is sent to the server.

8. Click OK.

ScriptBuilder inserts the Textarea into the document. You must add methods and event handlers to make the Textarea functional.

Note:

To place a label above a Textarea element, insert an HTML paragraph tag (<P>) in the document above the Textarea element, and enter text.

Images

Images give a Web site visual appeal and impact. You can import images in **.jpeg**, **.gif**, and animated **.gif** formats.

To insert an image:

1. Position the insertion point in the document where you want to place the image.
2.  Click the Image button on the HTML Tags toolbar or choose Image from the Tags menu.
3. In the Source field of the Image dialog, enter the relative path from the document file to the image file.
4. Fill out the rest of the dialog:

- Alt Tag: Text that is displayed if the browser does not support the IMG tag and also when the site visitor's mouse passes over the image.
- Align: Aligns the image relative to the margins or adjacent text. If no alignment is specified, the default is Bottom.
- Border Size: The thickness in pixels of a border around the image. A border size of zero means the image has no border.
- Height: The height of the image in pixels or as a percentage.
- Width: The width of the image in pixels or as a percentage.
- Spacing-Horizontal: A margin in pixels between the sides of the image and surrounding text and images.
- Spacing-Vertical: A margin in pixels between the top and bottom of the image and surrounding text and images.

For a complete description of the options in this dialog, see the online reference.

5. Click OK.

HTML Links

HTML link elements are the core elements of any Web document, providing a means for site visitors to jump to another Web page with the click of a mouse.

To insert a link:

1. Position the insertion point in the document where you want to place the link element.
2.  Click the Link button on the HTML Tags toolbar or choose Link from the Tags menu.
3. In the URL field of the Link dialog, enter the URL of the Web page to which the text is linked.
4. In the Text to Display field, enter the text the visitor clicks to go to the Web page.
5. In the Name field, enter a unique name for the link element.
6. In the Target field, enter the name of a window or frame in the URL where you want the visitor to jump.
7. Click OK.

Inserting an Existing File

To insert an existing file into your document:

1. Position the insertion point in the document where you want to place the text file.
2. From the File menu, choose Insert.
3. In the Open dialog, locate the file you want to insert.
4. Double-click the file name, or select it and click Open.

ScriptBuilder inserts the contents of the file into your document.

Servlets

To insert a servlet into your page:

1.  Click the Servlet button on the Script toolbar or choose Servlet from the Script menu.
2. In the Text field of the Servlet dialog, enter a name for the servlet. You can use this name to refer to the servlet throughout the document.
3. In the Code field, enter the file name of the servlet.

   This name is provided by the creator of the servlet and typically takes the form **servletname.class**.
4. In the Codebase field, enter the URL of the base directory of the servlet file. This URL is provided by the creator of the servlet.
5. To add initialization parameters, click the first Add button.

   The Add Initialization Parameter dialog appears.

   The name is provided by the creator of the servlet. The initialization parameter is used when the servlet is first loaded at the server. You can specify more than one set of initialization parameters.
6. To add service parameters, click the second Add button.

   The Add Service Parameter dialog appears.

   The name is provided by the creator of the servlet. The service parameter is used each time the servlet is requested by a user. You can specify more than one set of service parameters.
7. When the Servlet dialog is filled out, click OK.

Note:

All the values in the Servlet dialog are optional, but you must supply enough information that a browser can access the servlet.

The following illustration shows an example of the code resulting from entries in all the fields of the Servlet dialog:

IBM WebSphere Support

ScriptBuilder supports the IBM WebSphere plug-in to Web servers. To insert WebSphere-specific tags from the Script menu, choose WebSphere and then the name of the WebSphere tag you want to insert into your document.

- Bean creates an instance of a Bean that will be accessed elsewhere within a JSP file. Use JSP syntax and HTML template syntax to access the Bean.
  - **Bean Name**—The name used to look up the bean in the scope specified by the Scope attribute (case-sensitive)
  - **Class or Interface**—The name of the Bean class file, used to declare the Bean instance in the code (case-sensitive; the default value is type Object)
  - **Scope**—The lifetime of the Bean:
    Request: The Bean is set as a context in the request by a servlet that invokes the JSP file. If the Bean is not part of the request context, it is created and stored in the request context unless the 'Create bean if not found' option is unchecked.
    Session: If present in the current session, the Bean is reused; if not present, it is created and stored as part of the session if the 'Create bean if not found' option is checked.
    Userprofile: The user profile is retrieved from the servlet request object, cast to the specified type, and introspected.
  - **Create bean if not found in specified scope**—The JSP processor creates an instance of the Bean if the processor does not find the Bean within the specified scope
  - **Set bean properties from requested parameters**—The JSP processor examines all request properties and calls the set property methods that match the request properties
  - **Bean filename**—The name of the Bean .class file, Bean package name, or the serialized (.ser) file that contains the Bean; used only when the Bean is not present within the specified scope and the Create attribute is set to Yes (case-sensitive)
  - **Alias (VARNAME)**—The name used to refer to the bean elsewhere within the JSP file, by default the value of Bean Name (optional and case-sensitive)
- Insert is the base tag for specifying variable fields.
  - **Name**—The name of the Attribute, Request Parm, or Bean you are inserting
  - **Type**—The type of object identified by Name
    Attribute: The attribute to access within the request object, set using the setAttribute method (case-sensitive; cannot be used with the Bean and property attributes)
    Request Parm: The parameter to access within the request object (case-sensitive; cannot be used with the Bean and property attributes)
    Bean: The name of the JavaBean declared by a <BEAN> tag within the JSP file (case-sensitive)
  - **Property Name**—The property of the Bean to access for substitution (case-sensitive)
  - **Default Value**—A string to display when the value of the Bean property has not been set (optional and case-sensitive)
  - **Use Alternate Syntax**—Because the HTML standard does not permit embedding HTML tags within HTML tags, you can't embed the <INSERT> tag within another HTML tag. Check this box to use the HTML template alternate syntax.
- Repeat repeats HTML template syntax and the associated HTML formatting in a section of your Web

page.

- **Index Name**—Name identifying the index of this Repeat block (optional and case-sensitive)
- **Start**—Starting index value for this Repeat block (optional); the default value is zero
- **End**—Ending index value for this Repeat block (optional)

- Repeat Group creates repeating HTML blocks that display data already grouped in a database. Repeat and Repeat Group are especially useful for formatting tables and lists.
  - **Bean Name**—Name of the JavaBean that accesses the grouped data, declared by a <BEAN> tag within the JSP file (case-sensitive)
  - **Property Name**—Property of the Bean to access for grouping (case-sensitive)
  - **Start**—Starting index value for this RepeatGroup block (optional); the default value is zero
  - **End**—Ending index value for this RepeatGroup block (optional)

WebSphere is documented fully on the IBM Web site, www.ibm.com.

Previewing Documents

Previewing—testing documents in a browser—is a vital step in script development. If you have installed Microsoft Internet Explorer 3.02 or greater on your system, you can preview documents from within ScriptBuilder. In this case, internal previewing is automatically enabled. Otherwise, you can preview documents outside ScriptBuilder in Netscape Navigator, the full version of Microsoft Internet Explorer, or any other installed browser.

Note:

You cannot preview **.js** files. You must include or reference a **.js** file from a separate HTML file to test its functions. In addition, server-side scripts must be previewed from the server.

Configuring External Browsers

To preview documents or access InfoDesk Reference material in a browser other than Microsoft Internet Explorer 3.02 or greater, you must add the browser to the ScriptBuilder options.

To configure an external browser:

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Browsers tab.
3. Click Add to display the Add Browser dialog.
4. In the Browser Name field, enter the browser's name. To use multiple versions of a browser, include the release number.
5. In the location field, enter the path to the browser on your system, or click Browse to navigate to the location.
6. In the Type section, select the radio button corresponding to the type of browser you are configuring.
7. Click OK.

ScriptBuilder adds the browser to the list of browsers you can select when you choose to preview your document externally.

Specifying a Default External Browser

To specify one of the external browsers as the default:

1. From the Tools menu, choose Options.
2. On the Browsers tab of the Options dialog, select the browser you want as the default.

Previewing Documents Internally

To preview a document from Edit Mode, do one of the following:

-  Click the Toggle Edit/Preview button on the Standard toolbar.
- From the View menu, choose Toggle Edit/Preview, or press F8.
- Right-click the Editor and select Preview Mode from the shortcut menu.

Your HTML document opens in the internal browser inside the Editor pane.

If the Toolbox is open, you can close it by pressing F6 or by choosing Toolbox from the View menu. Closing the Toolbox gives you a full screen-width view of the preview.

Switching to Edit Mode

To switch back to Edit Mode from Preview Mode, do either of the following:

-  Click the Toggle Edit/Preview button on the Standard toolbar.
- From the View menu, choose Toggle Edit/Preview, or press F8.

Previewing Documents Externally

To start the external browser, do either of the following:

- From the View menu, choose External Preview and select the browser in which you want to preview your file.
-  Click the External Preview button. ScriptBuilder starts the default external browser (see Specifying a Default External Browser).

  If you have more than one external browser and don't want to use the default, click the arrow next to the External Preview button to open a menu of available external browsers.

Testing Script Syntax and Compatibility

Scripting languages greatly enhance Web sites by allowing for special effects, easier site navigation, animation buttons, and data validation. However, as with any complex language, it's easy to make errors. In addition, no scripting language is universally supported by all browsers. For example, you might use JavaScript in your code, but some site visitors won't be able to see the JavaScript features in their browsers.

NetObjects ScriptBuilder has three features designed to reduce or eliminate errors in your documents:

- A simple keystroke to find matching braces
- Syntax checking
- A Script Inspector to check for browser compatibility

Finding Matching Braces

To find matching braces:

1. Select the brace for which you want to find a matching brace.
2. Press F4.

If a matching brace exists, ScriptBuilder highlights it. If there is no matching brace, ScriptBuilder displays an error message.

Checking Script Syntax

ScriptBuilder includes a JavaScript and VBScript syntax checker, so you can check documents for errors before you preview them or post them to a Web site. The syntax checker verifies that keywords, object names, operators, delimiters, and the like are correctly placed in the document. The syntax checker does not check script semantics—for example, it doesn't check the validity of object references.

To run the syntax checker:

1. Position the insertion point at the beginning of your document.
2.  Click the Syntax Checker button on the Script toolbar or choose Syntax Checker from the Tools menu (Ctrl+F5).

The syntax checker reads through your document and displays descriptions of syntax errors in Results View. To go directly to the line containing the error, double-click the description in Results View.

In the illustration below, the syntax checker found a missing close-parenthesis in a function at line 57 of the document.

If you are writing ASP server-side scripts, you must specify the language you're using before you run the syntax checker.

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Code tab.
3. Under Syntax checker, select VBScript or JavaScript from the ASP Language drop-down list.
4. Click OK.

Checking Browser Compatibility

If you're creating an intranet application, you can probably assume that all site visitors are using the same Web browser, and no compatibility issues should arise. On the other hand, if you're creating an Internet application, you have to consider browser compatibility.

The two major browsers in use today are Netscape Navigator and Microsoft Internet Explorer. The following table shows which scripting languages these browsers support.

	Netscape Navigator	**Microsoft Internet Explorer**
JavaScript/JScript (Microsoft's implementation of JavaScript for Internet Explorer is called JScript)	2.0 and above	3.0 and above
VBScript	None	3.0 and above
Java	2.0 and above	3.0 and above
ActiveX	3.0 (plug-in) and above	4.0 and above
Netscape plug-ins	2.0 and above	3.0 and above

Not only are different scripting languages supported by different browsers, but also the scripting languages have different dialects (or versions), and not all versions are supported by all browsers.This table highlights the JavaScript dialects and their support.

JavaScript Version	**Browser Support**
JavaScript 1.2	Navigator 4.0, Internet Explorer 4.0 (partial)
JavaScript 1.1	Navigator 3.0 and above, Internet Explorer 4.0 (partial)
JavaScript 1.0	Navigator 2.0 and above, Internet Explorer 3.0 and above (partial)
JScript 3.0	Internet Explorer 4.0 and above, Navigator 4.0 and above (partial)
JScript 2.0	Internet Explorer 3.0 with JScript 2.0 patch, Internet Explorer 4.0, Navigator 3.0 (partial)
JScript 1.0	Internet Explorer 3.0 and above, Navigator 2.0 and above (partial)

Analyzing Documents with the Script Inspector

The Script Inspector analyzes the <SCRIPT> tags in the current open document and tells you which version of the Netscape and Microsoft browsers are needed to successfully run your script. In addition, if the Script Inspector finds something that is not compatible with a browser, it flags the problem. You can then step through the document and locate the specific language elements the Script Inspector flagged.

Note:

The Script Inspector only looks at <SCRIPT> tags. It does not analyze <SERVER> tags, which are analyzed by the server, not the browser.

The language attribute of the <SCRIPT> tag determines how the Script Inspector looks at its contents. The following table shows possible <SCRIPT> tag contents and what the Script Inspector analyzes to point out compatibility issues.

<SCRIPT> Tag	Browser Analysis
<SCRIPT LANGUAGE="JAVASCRIPT">	Navigator and Internet Explorer
<SCRIPT LANGUAGE="JAVASCRIPT1.1">	Navigator and Internet Explorer
<SCRIPT LANGUAGE="JAVASCRIPT1.2">	Navigator and Internet Explorer
<SCRIPT LANGUAGE="VBSCRIPT">	Internet Explorer only
<SCRIPT LANGUAGE="JSCRIPT">	Internet Explorer only

To analyze the current document with the Script Inspector:

1.  Click the Script Inspector button on the Script Toolbar, or choose Script Inspector from the Script menu or press Ctrl+F6.

   ScriptBuilder analyzes the documents and displays the results in Results View. This is a general message pointing out possible problems.

2. To display line-by line result details, do one of the following:

- Click the Show Details button (the down-arrow) at the right side of the Script Inspector pane.
- Right-click inside Results View, and choose Show Details from the shortcut menu.
- Double-click inside Results View.

   The bottom pane of Results View displays a more detailed description of the incompatibility.

3. Double-click a detail message to go directly to the instance of the incompatible element that the message describes.

Finding Workarounds for Browser Incompatibility

The Web contains a number of suggested workarounds for browser incompatibilities that can arise.

To see a workaround:

1. In Results View, right-click the detailed description of the browser incompatibility.
2. From the shortcut menu, choose View Workaround.

   ScriptBuilder launches the InfoBrowser and goes to the Web page with the appropriate workaround.
3. If a workaround is available—if none is available for your language version, the Web page will say so—select the workaround script and copy it to the Clipboard.
4. Back in your ScriptBuilder document, paste the workaround text into your code.

Saving and Reusing Scripts

ScriptBuilder's Script Library provides you with a way to manage and reuse code, a common problem with scripted pages. The Script Library comes with numerous prebuilt pieces of Perl, LotusScript, Active Server Pages, JavaScript, VBScript, Java Server Pages, and server-side JavaScript code. In addition to using these scripts, you can add your own. The Script Library has a sort feature that makes looking for scripts easy when the library becomes large.

Opening the Script Library

To open the Script Library, do either of the following:

- From the View menu, choose Toolbox Windows, Script Library.
  You must use the View menu to open the Script Library when the Toolbox is closed (hidden).
- Right-click the tab of any open Toolbox panel and choose Script Library from the shortcut menu.

Changing Your View of the Script Library

You can view the contents of the Script Library as icons—Large, Small, and List—and in a Detail format that itemizes the available scripts, their category, a description, and browser compatibility.

To change how you view the Script Library:

1. Right-click the Script Library.
2. From the shortcut menu, choose View and then choose the view you want.

Viewing Script Description and Code

1. Right-click the script's name in the Script Library.
2. Choose Properties from the shortcut menu.
3. To view a description of the script in the Script Library Properties window, click Description.
4. To view the script's code, click Code.

You can leave the Script Library Properties window open and select another script from the library. The window automatically updates itself to show the description or code of the newly selected script.

Sorting the Script Library

To make it easier to find a script in the Script Library, you can sort the scripts by any of the column headings:

- Name
- Category
- Description
- Netscape [Navigator version compatibility]
- Microsoft [Internet Explorer version compatibility]

The default sort is by category.

To sort the Script Library:

- In any view, right-click the Script Library and choose Sort from the shortcut menu.
- In Details view, click the column header on which you want to sort the scripts.

Inserting a Script into a Document

To insert a script into a document, do either of the following:

- Drag the script from the Script Library and drop it into the document.
- Position the insertion point in the document where you want to insert the script, right-click the script you want to insert, and click Insert in the shortcut menu.

Note:

When you develop with LotusScript, use VBScript as the highlighted language.

Adding a New Script to the Script Library

1. Open the document containing the script you want to add to the Script Library.
2. Select the code you want to add.
3. Do either of the following:
   - Right-click the ScriptBuilder Editor and choose Add to Script Library from the shortcut menu.
   - Right-click the Script Library and click Add from the shortcut menu.
4. In the Description tab of the Script Library Properties dialog:
   - Enter a name for the script in the Name field.
   - In the Category field, enter the type of script you are adding. You can also select from the drop-down list of script categories.
   - In the Description field, enter a brief description of the purpose of the script.
   - From the drop-down lists in the Browser support section, select the versions of Netscape Navigator and Microsoft Internet Explorer that support the script. "Unsupported" is an option.
5. Click the Code tab of the Script Library Properties dialog and make sure you have included all the necessary code.
6. Click the close box.

In the same way, you can also write new code directly into the Script Library Properties dialog.

Editing a Script in the Library

1. In the Script Library, right-click the script you want to edit and choose Properties from the shortcut menu.
2. Make the necessary changes in the Script Library Properties dialog.
3. Click the close box.

Removing a Script from the Library

1. In the Script Library, right-click the script you want to remove.
2. In the shortcut menu, click Remove.
3. Click Yes to confirm that you want to remove the script.

Creating a New Script Library

Using your existing Script Library as a starting point, you can create a new Script Library.

1. Right-click the Script library and choose Merge from the shortcut menu.
2. In the Merge Script Libraries dialog, click the New button.
3. In the Create New Reference File dialog, enter a file name and choose a location for the new Script Library.

   If you're working as part of a team, you can place the new library on a network drive to be shared with your co-workers.
4. Click OK to return to the Merge Script Libraries dialog.
5. Select one script from the current library's Scripts available list, and copy it to the Scripts available list for the new library (Reference file).
6. Click Close to complete creating the new Script Library.

Specifying an Alternate Script Library

You can name your newly created Script Library or any other as the current library.

1. From the Tools menu, choose Options.
2. In the Options dialog, click the Files tab.
3. If you've changed Script Libraries before, you can select a previous library from the drop-down list under Script library location. Then click OK to close the Options dialog.

To select a library that's not in the drop-down list:

1. Click Browse.
2. Select the new Script Library from the Select Script Library File dialog and click Open.
3. Click OK to close the Options dialog.

Importing and Exporting Script Libraries

Using the Merge Script Libraries dialog, you can copy scripts from one library file to another. See Migrating Your Script Library from Previous Versions.

Working with Script Components

ScriptBuilder 3.0 introduces script components, which bring the principles of object-oriented programming to Web scripting. As an easy way to save and reuse complex blocks of code, script components make it possible for you to add commonly performed operations to your documents—such as data validation, setting cookies, or creating URL history lists—without having to rewrite the code from scratch each time. In addition, because they are kept in files that are only referenced from your document—the script component code is not actually inserted into the document—script components don't add significantly to the size of your documents, although they do add significantly to the power of your scripts.

A script component is a "wrapper" written in XML for pieces of code that have distinct functionality and that can reused in any number of situations. You insert a script component into your document, fill out a few script-specific attributes, and the script component executes at run-time. You can then insert the same script component into another document, fill out the attributes that apply to that component, and again the script component executes.

Note:

If your document includes script components, you must deploy the document to place it in another location on your local system or a network server. See Deploying Documents Containing Script Components.

Looking Inside Script Components

Each script component is made up of two files:

- A Script Component Class **.scc** description file, an XML file that includes the following:
  - Script component name
  - Name that appears in the Component Gallery
  - Name of the source **.js** file
  - Methods and properties
  - Names of any associated icon files in **.gif** format

- A JavaScript **.js** include file, which contains the actual code of the script component

When you insert a script component into your document, you don't see either file. Instead, ScriptBuilder inserts a script block containing links to the source files and a function that initializes the script component.

Using the Component Gallery

The Component Gallery, one of the five panels in the Toolbox, is where ScriptBuilder keeps its script components—both those provided by NetObjects and those you create.

The Gallery itself contains four pages:

- The Client page contains script components that mostly produce results the site visitor will see in a browser, such as a pop-up window.
- The Utility page contains script components that mostly produce results that are not visible in the browser, such as a string processing component that performs some utility on a string.
- The Forms page contains script components for working with and building forms.
- The Database page contains script components for working with databases.

To display the contents of a page, click its handle.

Inserting Script Components into Documents

When inserting a script component into your document, you drag and drop it from the Component Gallery and then fill out its attributes in the Component Inspector. As noted previously, you never touch the **.scc** description file or the **.js** include file.

To insert a script component into your document:

1. Open the Component Gallery, either by clicking its tab in the Toolbox or by choosing View, Toolbox Windows, Component Gallery.
2. Go to the page containing the script component you want to insert.
3. Drag the script component from the Component Gallery and drop it in your document.

   You can drop it anywhere in the Editor pane. ScriptBuilder automatically places the script component in the correct location.

   ScriptBuilder opens the Component Inspector.
4. Fill in the required information on the Properties and Events tabs of the Component Inspector.

   To enter data from the Component Inspector into the script, press Enter or click the next field on the Component Inspector.

   To cancel a data entry, press Esc.
5. To close the Component Inspector, click its close box.

Note:

Not all script components have properties or events.

Deleting a Script Component from a Document

1. Display the Toolbox Map panel.
2. In the Component pane, right-click the name of the script component you want to delete.
3. In the shortcut menu, choose Delete.

Modifying Script Component Attributes

You can modify the attributes of a script component in your document by displaying the Component Inspector and changing the entries on the Properties and Events tabs.

To modify script component attributes:

1.  Click the Component Inspector button in the Standard toolbar.
2. In the Component Inspector, select the name of the script component whose attributes you want to change from the drop-down list.
3. Make the changes you want and then close the Component Inspector.

You can also open the Component Inspector by:

- Choosing Component Inspector from the View menu.
- Displaying the Toolbox Map panel, right-clicking a script component name in the Component pane, and choosing View Component Inspector from the shortcut menu.

Viewing Script Component Attributes

To view the attributes of a script component:

1. Displaying the Toolbox Map panel.
2. In the Component pane, right-click the name of the script component whose attributes you want to view.
3. From the shortcut menu, choose the attributes you want to view.

- To view the script component's properties, choose Properties; to view its methods, choose Methods; to view events, choose Events.

  The text in the submenu—from example, the listing of properties—is purely informational. You can't edit a script component's properties from the menu.

- To view the script component's specification from its .scc file, choose About from the shortcut menu.

Importing JavaScript as a Script Component

You can create your own script components by writing the JavaScript include file and then running it through the Script Component Generator in ScriptBuilder.

To import JavaScript as a Script Component:

1. Write the JavaScript .js include file for your JavaScript object and save it to disk.
2. From the Tools menu, choose Import JavaScript Object.
   ScriptBuilder opens the Import JavaScript Object dialog.
3. Click the Browse button and select your JavaScript include file.
   ScriptBuilder lists the functions contained in your include file.
4. Select the script component's constructor function from the list of functions.
5. Click OK.

ScriptBuilder creates a Script Component Class **.scc** file with a file name matching the name of the constructor function. For example, if the constructor function is **com_netobjects_currencyComponent**, the Script Component Class file is **com_netobjects_currencyComponent.scc**.

Adding a Script Component to the Gallery

To add script component to the Gallery:

1. From the Tools menu, choose Add to Component Gallery.
2. ScriptBuilder opens the Add Component dialog.
3. Click the Browse button and select the .scc file for your script component.
4. From the Page Name drop-down list, select the Component Gallery page for your new script component.
5. Click OK.

Standard toolbar

- **New.** Creates a new document in the ScriptBuilder Editor using the default page type you have selected.
- **Open.** Opens an existing document in the ScriptBuilder Editor. Click the down-arrow to see a list of recently opened files.
- **Save.** Saves the current document in the ScriptBuilder Editor. Click the down-arrow to save all open files.
- **Print.**Prints the current document in the ScriptBuilder Editor.

 **Cut.** Cuts selected text and saves it to the Clipboard.

 **Copy.** Copies selected text and saves it to the Clipboard.

 **Paste.** Pastes cut or copied text from the Clipboard to the document.

 **Undo.** Cancels the latest action.

 **Redo.** Redoes the last canceled action.

- **Toolbox**. Shows/hides the Toolbox.
- Results View. Shows/hides the Results View.
- Component Inspector. Shows/hides the Component Inspector.
- **Toggle Edit/Preview Mode.** Toggles the ScriptBuilder Editor between Edit mode (default) and internal preview (if Microsoft Internet Explorer 3.02 or higher is installed).
- **External Preview.** Sets the ScriptBuilder Editor to preview the current document in a browser outside ScriptBuilder. Click the down-arrow to display a list of available external preview browsers.
- **Deploy.** Displays the Deploy Component dialog.

Script toolbar

- **Syntax Checker.** Starts the syntax checker and displays the results in Results view.
- **Script Inspector.** Starts the Script Inspector and displays the results in the Results View.
- **Script Block.** Inserts a <SCRIPT> block into the your document in the currently selected highlighted language.
- **Server Block.** Inserts a Server block into the your document.
- **Servlet.** Displays the Servlet dialog.
- **Comment Block.** Inserts a comment block into the current document.
- **Function Block.** Displays the Function Block dialog so you can insert a function block into the current document.
- **Window.open().** Displays the Window.open() dialog so you can insert a window.open() method into the current document.
- **Document.write ().** Inserts a document.write() method into the current document.
- **Message Box.** Displays the Message Box dialog so you can insert a message box into the current document.
- **Custom Object.** Displays the Custom Object dialog so you can insert a custom object into the current document.

Form Elements toolbar

- **Form.** Displays the Form dialog.
- **Button**. Displays the Button dialog.
- **Checkbox.** Displays the Checkbox dialog.
- **Radio Button.** Displays the Radio Button dialog.
- **Select.** Displays the Select dialog.
- **Text.** Displays the Text dialog.
- **Textarea.** Displays the Textarea dialog.

HTML Tags toolbar

 **Bold.** Inserts HTML bold tags into the current document.

 **Italic.** Inserts HTML italic tags into the current document.

 **Underline.** Inserts HTML underline tags into the current document.

 **Align Left.** Inserts HTML align left tags into the current document.

 **Center.** Inserts HTML center tags into the current document.

 **Align Right.** Inserts HTML align right tags into the current document.

 **Unordered List**. Inserts HTML unordered list tags into the current document.

 **Ordered List.** Inserts HTML ordered list tags into the current document.

 **Paragraph.** Inserts HTML paragraph tags into the current document.

 **Line Break.** Inserts an HTML line break tag into the current document.

 **Horizontal Line.** Inserts an HTML horizontal line tag into the current document.

 **Heading 1.** Inserts HTML H1 tags into the current document.

 **Heading 2.** Inserts HTML H2 tags into the current document.

 **Heading 3.** Inserts HTML H3 tags into the current document.

 **Heading 4.** Inserts HTML H4 tags into the current document.

 **Heading 5.** Inserts HTML H5 tags into the current document.

 **Heading 6.** Inserts HTML H6 tags into the current document.

- **Image.** Displays the Image dialog.
- **Link.** Displays the Link dialog.

InfoBrowser toolbar

 **Back.** Takes you back to the previous Web page.

 **Forward.** Takes you forward (after having moved back) to a previously viewed Web page.

 **Stop.** Cancels the current loading of a Web page.

 **Refresh.** Reloads the current Web page.

You can type a URL directly into the text box on the InfoBrowser toolbar. The toolbar includes a drop-down list of these URLs. URLs accessed through the InfoDesk do not appear in the list.

Window List toolbar

- **Previous Document.** Makes the previous open document the active document in the ScriptBuilder Editor.
- **Next Document.** Makes the next open document the active document in the ScriptBuilder Editor.

The toolbar also includes a drop-down list of the current open windows.

Find toolbar

 **Find.** Displays the Find and Replace dialog open to the Find tab.

 **Replace.** Displays the Find and Replace dialog open to the Replace tab.

- **Find Previous.** Searches the current document for the first occurrence of the search text before the insertion point.
- **Find Next.** Searches the current document for the first occurrence of the search text after the insertion point.
- **Find in Files.** Displays the Find in Files dialog.

You can type a search string directly into the text box on the toolbar. The drop-down list shows your most recent searches. Searches made with the Find and Replace dialog do not appear in the list.

Using Hot Keys

NetObjects ScriptBuilder uses the following hot keys:

Hot Key	Format
F1	Help
F3	Edit | Find Next
F4	Script | Find Matching Brace
F6	View | Toolbox (toggle)
F7	View | Results View (toggle)
F8	View | Edit/Preview (toggle)
F11	View | Component Inspector (toggle)
Ctrl+F5	Tools | Syntax Checker
Ctrl+F6	Tools | Script Inspector
Ctrl+F7	Script | Script Block
Ctrl+F8	Script | Comment Block
Ctrl+F9	Script | Function Block
Ctrl+F10	Script | Server Block
Ctrl+A	Edit | Select All
Ctrl+B	Tags | Bold
Ctrl+C	Edit | Copy
Ctrl+F	Edit | Find
Ctrl+G	Edit | Go to Line
Ctrl+H	Edit | Replace
Ctrl+I	Tags | Italic
Ctrl+K	Delete current line
Ctrl+L	File | Save All
Ctrl+N	File | New
Ctrl+O	File | Open
Ctrl+P	File | Print
Ctrl+S	File | Save
Ctrl+T	Tools | Options
Ctrl+U	Tags | Underline
Ctrl+V	Edit | Paste
Ctrl+W	File | Close
Ctrl+X	Edit | Cut
Ctrl+Y	Edit | Redo
Ctrl+Z	Edit | Undo
Ctrl+Tab	Next open document
Shift+Ctrl+Tab	Previous open document

Using the InfoBrowser as a Web Browser

To use the InfoBrowser to connect to the Web:

- From the View menu, choose Toolbars, then InfoBrowser.

ScriptBuilder displays the InfoBrowser toolbar. Enter a URL into the URL box on the toolbar and press Enter.

The page you requested appears in the InfoBrowser.

CLIENT

The **ActiveImage** component responds to events triggered by the site visitor—such as onMouseOver and onClick—with image swapping and other actions.

The **AnimImage** component is a rotating banner image. Images are rotated in a continuous loop, with intervals set by a timer. You can associate a URL with the component.

The **Browser** component has several methods that detect the browser type. It can return the name of the browser or an integer index of the browser, or it can fire an event based on the browser type. You can use the Browser component in conjunction with other components to redirect a visitor automatically to a specific page dependent on the visitor's browser type.

The **DigitalClock** component is a simple digital clock that updates every minute, displaying in 12- or 24-hour time. By using images it appears "embedded" in the page, unlike other JavaScript clocks, which require the use of a text box.

The **Display Source** component contains a function that pops up a window showing the source code of the current page. Use this component if you want visitors to see your code as an example.

The **Filter** component makes sure return values are not null or undefined. For example, if a function requires a non-null or defined parameter but is passed an undefined or null variable, a JavaScript error occurs. To avoid this, you can pass the variable through the filter functions, and these functions will pass in a variable to the function that won't cause an error. You can also use this component for data validation.

The **historyNav** component makes a selection list containing recently visited sites and goes to them when selected. After being instantiated, the selection list tracks all Web pages visited by the visitor and lists them as options in the list. Choosing a selection transfers the visitor back to that page in the history list.

The **JSError** component lets you block JavaScript errors from popping up on the page.

The **Navigate URL** component sends the site visitor to a URL selected from a list. You create the selection list with URLs the visitor can select, and passes this selection list to the component, which will send the visitor to the selected site. This component is useful for pages with many resource links.

The **Radio** component returns information on radio buttons—for example, which radio button is currently selected. Useful for sites with forms to verify that certain elements are selected, for error checking and data validation, and for dynamic pages dependent on which radio button is selected.

The **Rotating Banner** component rotates different images in a banner, like many advertisements on the Web. If clicked, it sends the visitor to another URL; the target URL depends on the image clicked. The Rotating Banner component doesn't need any CGI. It is useful for pages with a lot of advertisers, which require an equal distribution of advertising time.

The **Select** component returns information on selected items in a select list. It can display the currently selected element or elements. Use this component for data validation and error checking, or for dynamic pages based on forms, without the use of CGI.

The **Show Status** component displays link information in the status bar. Instead of seeing the URL in the status bar when the mouse passes over a link, the visitor sees helpful information or a better description of the link. The information or description is provided by the site designer.

The **Show Visits** component uses cookies to display how many times a page has been visited.

The **Special Effects** component scrolls your message across the status bar so the visitor can learn more about the page.

The **Pop-up Color Chooser** component is a pop-up window containing color tiles of browser-safe" colors, such as the palette of colors from NetObjects Fusion. Visitors can pick a color and when picked, can specify that an action take place, such as changing the background color of another page.

The **Pop-up Window** component is a wrapper of the JavaScript window.open() command. It pops up a new browser window with configurable parameters, such as size, location, and toolbars.

The **Pop-up Dialog** component is a wrapper of the Alert, Prompt, and Confirm JavaScript dialogs.

The **Pop-up Calendar** component is a pop-up calendar that can display days of the week, navigate forward and backward one month at a time, and can return a date that a visitor clicks. It is also capable of displaying events that occur on certain days, as you define them.

UTILITY

The Array component stores a list of elements, such as a list of URLs in a random URL generator.

The Cookie component gets, sets, and deletes cookies, which is data held on the visitor's computer, placed there by the site. This information can be used for such things as recording preferences when a person visits a site frequently.

The Currency component formats strings as currency. For example, the string "1234" can be formatted as "$12.34" or "$1234.00."

The **Data Validation** component validates such data as phone numbers, zip codes, alphabetic and alphanumeric data, and so on. Use this component on sites with forms that need valid data before processing the fields, such as mail order forms or registration forms.

The **Date** component has various functions that return the day of the week, day of the month, current month, and so on. Use this component to display dynamic data, such as the current date.

The **Objects** component displays information about objects on the page. It has functions that verify if an object has a certain property and also lists all of an object's properties and methods. This component is useful if you create a lot of custom objects, and for debugging them.

The **Random** component returns random integers or floating point numbers. It is needed because the Math.random() function only gives a random number between 0 and 1. This component is useful in conjunction with the array component to display random quotes, send visitors to random links, and for games dependent on randomness.

The **Redirection** component sends the visitor to another URL. It is used in conjunction with the browser component to send visitors to different pages dependent on which browser they are using, or dependent on their age, or their preferences, or if the site has moved somewhere else.

The **String** component manipulates strings. For example, it takes away spaces or non-numeric characters, or finds and replaces substrings within strings. Use this component to receive information from the site visitor and handle it efficiently.

FORMS

The **Button** component is a wrapper of the Submit, Reset and Text Form buttons.

The **CheckBox** component is a wrapper of a checkbox element.

The **FormImage** component is a wrapper of the form image element component an image that acts like a submit button.

The **RadioButton** component is a wrapper of a radio button element.

The **SelectBox** component is a wrapper of a select element. It can be rendered as a drop-down list or a scrolling list. Because events fire on selection, elements such as SelectNav, which takes visitors to a different page after selecting an item, can easily be configured with this control.

The **TextArea** component is a wrapper of the TextArea field that can perform some basic validations, such as IsBlank and maxChars.

The **TextBox** component is a sophisticated wrapper of a text box that performs validation and formatting as you define them. For example, if you specify that the text entered must be a number in fixed-width format, the text box automatically validates and formats visitor input.

DATABASE

The **DBConnection** component establishes a connection to a database.

The **DBList** component displays a record set from a DBQuery in a formatted HTML table. You can control many formatting options, including table border, column width, and so on.

The **DBQuery** component defines a select query on a database that other elements can call upon to retrieve and display data.

The **DBPackager** component "listens" for filters sent via the URL that will narrow down a query in a DBQuery component. It is required for database operations.

Customizing the InfoDesk Reference Panel

With ScriptBuilder 3.0, you can access centralized reference material in the InfoDesk Reference panel, making your work more efficient. These materials can include reference materials you create or acquire, which you add to the InfoDesk Reference panel using its XML technology.

XML Architecture

You can customize the contents of the InfoDesk Reference panel by modifying existing information or adding your own reference books. The InfoDesk Reference uses XML to define the data structure for reference books that are housed inside an ASCII-based .idk file. This appendix details the XML tags used to define InfoDesk Reference books.

If you are not familiar with XML, you can also find basic information and updates on Microsoft's XML web site **http://www.microsoft.com/xml**.

InfoDesk Tags

The following table describes each ''XML tag and its attributes, and also includes sample values.

XML tag	Attribute	Description Sample Value
INFODESK		Indicates start of a new InfoDesk Reference XML section <INFODESK></INFODESK>
	NAME	Text displayed in the first node within the Reference panel indicating the start of the library <INFODESK NAME=“Tools and Technologies”>
	DESCRIPTION	Text displayed as a hint when the entire node text is not visible <INFODESK DESCRIPTION=“Reference Technology Websites”>
NODE		Indicates the start of a new Reference item <NODE></NODE>
	NAME	Text displayed as the node for the user to select (required) <NODE NAME=“Scripting”>
	ICON	Name of icon file that will be displayed within the reference treeÑavailable values are BOOK, TEXT, OBJECT, PROPERTY, METHOD, EVENT, FUNCTION, STATEMENT, and OPERATOR (required) <NODE ICON=“Book”>
	LOCALURL	Path from the \NetObjects ScriptBuilder\InfoDeskDocs\ folder to local reference material for a given node <NODE LOCALURL=“Vbscript.htm”> In a default ScriptBuilder installation, the sample value would open the following file: C:\Program Files\NetObjects ScriptBuilder\InfoDeskDocs\Microsoft VBScript3.0 Reference\Vbscript.htm
	REMOTEURL	URL for online reference material for a given node that is accessed by the InfoBrowser <NODE REMOTEURL=http://www.microsoft.com/xml/>
	TEXT	Text that will be inserted if this item is drag and dropped from reference window. If dropped onto a selection block, this text will be inserted before the selected block <NODE TEXT=“<H3>“>
	ENDTEXT	Text that will be inserted if this item is dragged and dropped from the reference window. If dropped onto a selection block, this text will be appended after the selected block. <NODE ENDTEXT=“</H3>“>
	STARTPOS	Starting position within the TEXT drag & dropped that will be selected upon a successful drag and drop <NODE STARTPOS=“3”>

LENGTH	Number of characters that will be selected <NODE LENGTH="2">

Adding Your Own InfoDesk Reference Book

1. Create a new text file. You will later save this as an InfoDesk (**.idk**) file.
2. Using the InfoDesk tags table, insert the INFODESK tags, with the NAME and DESCRIPTION attributes as desired.
3. Within the INFODESK tags, insert the NODE tags, including desired attributes.

   To maintain a hierarchy of nodes, embed NODE tags within other NODE tags.
4. Once you have completed adding nodes, save the file in XML format with an **.idk** file extension.

   The file name must be identical to the NAME attribute in the INFODESK tag.
5. Place the file in the Idk folder, which is in the NetObjects ScriptBuilder folder.

   ScriptBuilder automatically displays all the Idk folder's **.idk** files in the InfoDesk Reference panel's Settings dialog.
6. Right-click the InfoDesk Reference panel and choose Settings from the shortcut menu.
7. In the Settings dialog, select the check box next to your reference library book and click OK to load it.