About the WebService Behavior

The WebService behavior enables client-side script to invoke remote methods exposed by Web Services, or other Web servers, that support the SOAP and Web Services Description Language (WSDL) 1.1. This behavior provides developers the opportunity to use and leverage SOAP, without requiring expert knowledge of its implementation. The **WebService** behavior supports the use of a wide variety of data types, including intrinsic SOAP data types, arrays, objects, and XML data. The **WebService** behavior is implemented with an HTML Component (HTC) file as an attached behavior, so it can be used in Microsoft Internet Explorer 5 and later versions.

This article provides a general overview of the **WebService** behavior and examines the improvements and alternatives it offers to traditional database-driven Web page design. Once the **WebService** behavior is attached to a Web page, Internet Explorer 5 can invoke methods from Web Services and use the results directly in client-side script. The Using the WebService Behavior article complements this overview by providing detailed code samples and by discussing the specific functionality of the behavior.

The following topics are covered in this document.

Prerequisites

To use the information in this document most effectively, you should have a good understanding of scripting and Dynamic HTML (DHTML) behaviors. Specifically, you should be able to script using an object that is implemented in an HTC file.

The **WebService** behavior uses the SOAP protocol to communicate with Web Services, yet its purpose is to provide a simple way to take advantage of this protocol without requiring expert knowledge of SOAP. Although no knowledge of these protocols is necessary to use the **WebService** behavior, some familiarity with these topics will help you understand how the behavior works behind the scenes. For more information, see the section on SOAP.

It is not necessary to implement a Web Service to use the **WebService** behavior—all that is required is an existing compatible Web Service. Also, the Web Service must be accessible from the Web server that is hosting the Web page. To initiate remote method calls using the **WebService** behavior, it is necessary to know some details about the classes that are implemented on the Web Service, such as the methods and the required parameters. For more information, see the Web Services section.

Terminology

This section defines certain terms that are used throughout the article.

Web Service	Programmable application logic that is accessible using standard Internet protocols. Web Services provide well-defined interfaces, or contracts, that describe the services provided. The **WebService** behavior uses Web Services.

WebService behavior	The primary subject of this article. A reusable DHTML component that uses Web Services by communicating over HTTP using SOAP.
callService method	One of the methods exposed by the **WebService** behavior.
method call	The invocation of a method.
synchronous	A form of processing that blocks interaction with the application until execution has completed.
asynchronous	A form of processing that permits interaction with the application during the execution of a task.

Note The **WebService** behavior enables both synchronous and asynchronous remote procedure calls to Web Services. By default the asynchronous mode of method invocation is used. The async property of the call object is used to control the mode of invocation.

WebService HTC File

The **WebService** behavior is encapsulated in an HTC file. Therefore, before you begin using the behavior in your own Web pages, download the WebService HTC File and copy it to a folder in your Web project.

For more information on referencing the HTC file in a Web page, see Using the WebService Behavior, which shows you how to set up the **WebService** behavior and use it to invoke remote methods. Also, see the WebService reference pages.

Benefits

The primary benefit of the **WebService** behavior is that it provides a simple way for you to call methods that are exposed by Web Services using the SOAP protocol. The **WebService** behavior enables you to call a remote method using a few, straightforward, client-side scripting methods exposed by the behavior. Navigating to another URL is unnecessary when using the **WebService** behavior to deliver data to a page because DHTML is used to update the page's content. This approach enhances the browsing experience significantly, compared to traditional browsing approaches that require a full page refresh.

The **WebService** behavior is implemented as an attached behavior, as opposed to an element behavior, and, therefore, can be used in Internet Explorer 5 and later versions. The **WebService** behavior is a reusable component, so its encapsulated capabilities help reduce the need to duplicate code, thus improving the overall organization of the client-side script and the Web application. All protocol-specific code, and most of the code required to handle the data transactions, is encapsulated from the client-side script in the main Web page, which is a general benefit of using DHTML behaviors. You only need to attach the **WebService** behavior once in order to invoke methods from one or more different Web Services.

This behavior enables Internet Explorer 5 users to take advantage of some of the latest cross-platform programming techniques. Web Services can reside anywhere on the Internet and encapsulate building blocks of capability, which can be assembled, packaged, or presented in various ways in a Web page. The Windows Communication Foundation (WCF) Developer Center site provides access to a variety of tools and resources for designing and using Web Services. Using such a distributed architecture offers improved scalability because data- or CPU-intensive tasks can be organized into dedicated Web Services, freeing the client from unnecessary burden. Therefore, the **WebService** behavior can help enhance the client browser experience and improve the overall organization of the Web application.

The **WebService** behavior provides a more streamlined approach to the problem of delivering information from the Web server to Internet Explorer 5 and later. Using the **WebService** behavior to access Web Services simplifies things on the client side, making the use of Web Services more appealing. The behavior can be updated and adapted as the SOAP standard evolves, without requiring major changes to client-side script in the main Web page.

Introduction

The Internet has already evolved well beyond basic connectivity and presentation services to offer rich interactive pages, which are commonly enhanced by client-side or server-side script and other components. To further extend the performance and versatility of the Internet, Web Services are being developed that provide new capabilities which Internet Explorer 5 and later can use to deliver a new level of performance.

The primary reason to use the **WebService** behavior is that it provides you with a way to use Web Services in Internet Explorer 5 and later without having expert knowledge of SOAP. With the **WebService** behavior, you can implement a method call on a Web Service with just a few lines of code and you can apply the **WebService** behavior in any Web page or from within another HTC file. For practical advice on using the **WebService** behavior, see Using the WebService Behavior.

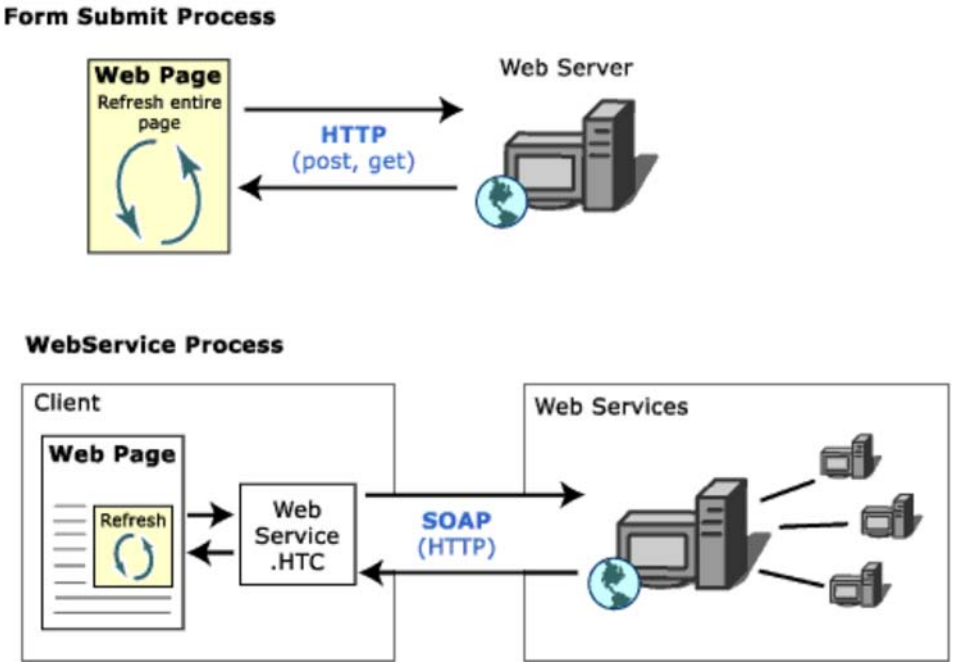
The **WebService** behavior simplifies the use of Web Services by handling communication of the SOAP data packets between the browser and the Web Service. All the SOAP-specific handling code is encapsulated inside the behavior, which helps to simplify client-side script in the main Web Page—this is a general benefit that comes from using DHTML behaviors.

The **WebService** behavior parses the XML data returned from method calls and exposes objects and properties to client-side script. The objects that are exposed by the behavior enable error handling and easy access to the returned data. You only need to attach the **WebService** behavior to the Web Page once, even if the client script references methods from several different Web Services.

WSDL is an XML-based language that describes the features and methods exposed by a Web Service. The **WebService** behavior supports WSDL 1.1 and can be used with other products that support this version. Microsoft .Net Frameworks SDK and Microsoft Visual Studio .NET generate the appropriate WSDL for Web Services automatically.

Comparing the WebService Approach to Forms

To help explain why the **WebService** behavior is so useful, it's worth comparing the **WebService** behavior technique with the approach commonly used to deliver data-driven Web sites today. The following diagram shows the basic process used by each technique.



The first part of the illustration shows how a Web page containing a form commonly uses either the `get` or `post` method through HTTP to update a Web page. Each time a **form** is submitted, the client navigates to a new URL, after which the browser downloads and renders the entire page. This method is widely used today but is inefficient because the Web page refreshes and re-renders an entire page of content, even if only a small percentage of the page has actually changed. Web surfers commonly encounter this behavior when browsing e-commerce and data-driven pages. When there's a need to browse numerous items and pages, such as when searching a catalog or search engine, the delays and waste of resources can be significant.

The second part of the diagram illustrates how a Web page can use the **WebService** behavior to avoid the drawbacks associated with the traditional **form** submit approach. The **WebService** behavior receives

method calls from the client-side script and sends a request to the Web Service. The results are returned to the client script, and processing continues. The Web page can then use the information in whatever context is required, such as updating some portion of page rendering using DHTML.

A key feature of the **WebService** behavior is that it enables client-side script to access a Web Service without requiring navigation to another URL. Using the **WebService** behavior approach, the portions of the page that are indicated by the user's inputs can be dynamically updated using DHTML, providing a significant improvement in the browsing experience.

What the WebService Behavior Does

The **WebService** behavior enables a method call to be made to a Web Service using a simple scripted syntax, as shown in the following code example.

```
iCallID = myService.MyMath.callService("add",int1,int2);
```

To invoke a method on a Web Service, the author first attaches the **WebService**.HTC file to any element in the page. Once the behavior is attached, the **WebService** behavior enables either synchronous or asynchronous calls to be made to Web Services from client-side script. The asynchronous nature of a remote method invocation means that there is a delay between the execution of the method and the arrival of its returned result. Using the synchronous mode of processing means that the client script processing halts until the **callService** method has completed. The asynchronous mode of method invocation is the default mode of the **WebService** behavior.

Note Synchronous calls to remote services lock the user interface while the call is pending and, therefore, aren't practical for browser-based applications.

The **WebService** behavior handles the process of calling the method and receiving the raw XML data packets from the Web Service. The user has the option of using either an event handler or a callback handler function to process the results. If an event handler is used, the **WebService** behavior fires the onresult event, which occurs when the **WebService** receives the response from a method call. Alternatively, if a callback function is used to process results, a result object is passed directly as an input parameter to the callback function.

Any client script using the **WebService** behavior should always test the error property to determine if the method invocation was successful. When an error is encountered, an errorDetail object is also exposed; this object has properties that can be evaluated to help identify the source of the error. The different techniques for handling returned results from method calls are described in Using the WebService Behavior.

The **WebService** behavior cannot directly invoke a method on a Web Service that is hosted in a different domain from the machine hosting the Web page. Nevertheless, a Web Service running on the Web server hosting the Web page can be configured to act as a proxy for other remote Web Services. For more information, see Calling Methods on Remote Servers.

SOAP

SOAP is a standard that uses XML data to communicate between processes and devices. Because SOAP supports HTTP, it is becoming a popular mechanism for binding Internet-based cross-platform applications together. The **WebService** behavior enables developers to take advantage of the versatility of XML and SOAP without needing to develop SOAP-specific code. In most cases, it is much easier to use the **WebService** behavior with some client script than it is to write the code to generate and handle SOAP transactions.

SOAP Resources

A detailed discussion of SOAP lies outside the scope of this article. The following articles provide comprehensive discussions and technical information on SOAP.

- W3C: Simple Object Access Protocol (SOAP) 1.2
- SOAP Toolkit

- Migrating from the Microsoft SOAP Toolkit to the .NET Framework
- Messaging Specifications Index Page

Web Services

Traditional software applications provide APIs that enable the applications to be automated by other clients and servers. But, unlike most of these APIs, Web Services expose methods that can be called from other machines and devices across the Internet. For example, an authentication Web service can expose methods that are used by other applications for access control, and developers can invoke the methods on the Web Service to enhance their own Web applications.

In general, the **WebService** behavior should work well with Web Services and servers that support both SOAP and WSDL 1.1. The behavior has been tested successfully with the **SOAP Toolkit**. The range of supported data types depends on the server implementation. See WebService Behavior: Supported Data Types for more information.

Web Service Resources

Any server that exposes methods which can be invoked using SOAP and WSDL 1.1 can be used by the **WebService** behavior. All you need to begin using the **WebService** behavior is a live Web Service that can be accessed from your browser.

Many of you will want to write Web Services to complement your **WebService** behavior-enhanced Web pages. The **SOAP Toolkit** provides a powerful collection of tools to help you create Web Services, and it is compatible with the **WebService** behavior. For developers interested in the next generation of Internet development tools, the Microsoft .NET Frameworks SDK and **Visual Studio .NET** products provide extensive support for developing Web Services, SOAP and WSDL 1.1.

Here are some recommended tools and references.

- Microsoft .NET Home Page
- Visual Studio .NET
- Web Services Description Language (WSDL) 1.1
- Windows Communication Foundation (WCF) Developer Center
- XML and Web Services

WebService Behavior Documentation

The following documents cover the basics of using the **WebService** behavior and related issues.

- Using the WebService Behavior

  This practical article shows you how to set up the **WebService** behavior and use it to invoke remote methods.

- WebService Behavior

  This is the main reference page for the **WebService** behavior.

- WebService HTC File

  Click the link to download the **WebService** HTC file.

- WebService Behavior: Supported Data Types

  This page lists the XML and ASP.NET data types supported by the **WebService** behavior.

Related Topics

- Introduction to DHTML Behaviors
- HTC Reference
- Using HTML Components to Implement DHTML Behaviors in Script

- Using Custom Tags in Internet Explorer
- Using DHTML Behaviors

Community Content

My version of HTC does not encode objects properly

Hi,

I have webservice.htc file version 1.0.1.1120. This version does not encode the objects properly if they are of derived type. For e.g., let us say a method MethodX() takes an argument of type BaseClass1[] and we pass DerivedClass1 in the array elements. When the call is made it will just have fileds that correspond to BaseClass1 but not DerivedClass1. Can someone help on this?

Thanks.
Sai

7/2/2008
Saigiridhar K