

Introduction

Welcome to *ASP.NET AJAX Programmer's Reference with ASP.NET 2.0 or ASP.NET 3.5*. The ASP.NET AJAX framework consists of two frameworks: the ASP.NET AJAX client-side framework and the ASP.NET AJAX server-side framework.

It's a well-known fact that client-side programming is very different from server-side programming. The main difference lies in the fact that client-side programming lacks a feature-rich programming framework like the ASP.NET/.NET framework. Wouldn't be great if you could write your client-side code in a framework with programming styles and capabilities like those of the ASP.NET/.NET framework? Enter the ASP.NET AJAX client-side framework. It simulates the rich programming features of the ASP.NET/.NET framework on the client side as much as possible. The capabilities of these simulations are only limited by the fundamental limitations of client-side technologies such as JavaScript.

The ASP.NET AJAX server-side framework extends the ASP.NET Framework to provide server-side support for Ajax-enabled Web applications. The ASP.NET AJAX client-side and server-side frameworks work hand in hand to meet both the client-side and server-side needs of your Ajax-enabled applications. That said, the ASP.NET AJAX client-side framework can also work alongside server technologies other than the ASP.NET Framework.

This book uses a unique approach characterized by the following attributes to provide you with an in-depth coverage of both the ASP.NET AJAX client-side and server-side frameworks:

- ❑ **Practical real-world examples:** The discussions in this book are presented in the context of numerous practical real-world examples that you can use in your own ASP.NET AJAX applications.
- ❑ **Under-the-hood looks:** This book takes you under the hood of both ASP.NET AJAX client-side and server-side frameworks, where you can see for yourself how they work from the inside out and how you can extend them to meet your application requirements.
- ❑ **Code walkthroughs:** I'll use numerous code walkthroughs to help you gain the skills, experience, and knowledge you need to implement similar features in your own ASP.NET AJAX applications.

Who This Book Is For

This book is aimed at the ASP.NET developer who wants to learn ASP.NET AJAX for the first time. No knowledge of ASP.NET AJAX is assumed.

What This Book Covers

This book is divided into 24 chapters and six appendices, as follows:

- ❑ **Chapter 1, “Ajax Technologies,”** provides an overview of the main technologies used in Ajax-enabled Web applications, such as `XMLHttpRequest`, `XML`, and `JSON`, in the context of examples.
- ❑ **Chapter 2, “JavaScript Base Type Extensions,”** explains the JavaScript base type extensions. You’ll learn how these extensions enhance the JavaScript base types such as `Array`, `Boolean`, `Date`, `Error`, `Object`, and `String` to enable you to experience these types — as much as possible — as you would their .NET counterparts.
- ❑ **Chapter 3, “Built-In and Custom Exception Types,”** first covers the ASP.NET AJAX built-in exception types, including `ArgumentException`, `ArgumentNullException`, `ArgumentOutOfRangeException`, `ArgumentTypeException`, `ArgumentUndefinedException`, `InvalidOperationException`, `NotImplementedException`, and `ParameterCountException`, in depth. Then it provides you with a recipe for developing custom exception types, uses this recipe to implement a custom exception type named `DuplicateItemException`, and implements a page that uses this custom exception type.
- ❑ **Chapter 4, “JavaScript Object-Oriented Programming and Type Reflection Extensions,”** first examines those JavaScript technologies that the ASP.NET AJAX object-oriented programming (OOP) and type reflection extensions use under the hood to extend JavaScript to add OOP and type reflection support. Then it provides a comprehensive coverage of the `Type` and its methods, where you’ll learn through numerous examples how to define namespaces, interfaces, classes, and enumeration types, how to implement classes that implement one or more interfaces, and how to implement classes that derives from other classes.
- ❑ **Chapter 5, “Event Programming Extensions,”** provides you with a detailed step-by-step recipe for implementing and adding events to your custom ASP.NET AJAX client classes to enable the clients of your classes to extend their functionality to execute application-specific logic. It then presents and discusses a practical example that uses this recipe. This chapter also describes the `EventArgs`, `CancelEventArgs`, and `EventHandlerList` classes and their methods and properties in detail.
- ❑ **Chapter 6, “DOM Extensions,”** shows you how the ASP.NET AJAX DOM extensions extend traditional DOM programming to add support for .NET-like methods and properties, and how to use these extensions in your own DOM programming. It covers the ASP.NET AJAX delegates and the `DomElement` and `DomEvent` client classes and their methods and properties.
- ❑ **Chapter 7, “Component Development Infrastructure,”** covers the ASP.NET AJAX component development infrastructure and its main constituent interfaces, including `IDisposable`, `INotifyDisposing`, `INotifyPropertyChanged`, and `IContainer` and its main constituent classes, including `Component` and `Application`. You’ll also learn through numerous examples how to implement these interfaces and how to implement a custom component that derives from the `Component` base class. This chapter also covers the application and component life cycles and application level events in detail.
- ❑ **Chapter 8, “Developing Client Controls,”** describes the `Control`, `Label`, `Image`, and `HyperLink` client controls and their methods and properties, and presents examples that use these client controls. This chapter also presents and discusses the implementation of a custom `Image` client control that extends the functionality of the ASP.NET AJAX `Image` client control.

- ❑ **Chapter 9, “Event Bubbling and Button Client Control,”** first covers the `CommandEventArgs` event data class and the `Button` client control. Then it discusses ASP.NET AJAX event bubbling and shows you how to implement custom controls that bubble their events up to their parents, and how to implement custom controls that capture events bubbled up by their children. This chapter implements a client control named `GridView`, which uses ASP.NET AJAX event bubbling.
- ❑ **Chapter 10, “Type Description Extensions,”** provides comprehensive coverage of the `TypeDescriptor` class and `ICustomTypeDescriptor` interface, from which you’ll learn how to take advantage of the ASP.NET type description capabilities in your own applications in order to isolate your client code from the specifics of the types of the objects that your client code deals with. This will allow your code to interact with different types of objects without code change. This chapter implements three Web pages that you can use to generically inspect the properties, events, and methods of any ASP.NET AJAX type. This chapter also implements a custom client control named `CustomTable` that uses the ASP.NET AJAX type description capabilities to display any type of data records. Finally, this chapter shows you how to dynamically inject metadata information.
- ❑ **Chapter 11, “Data Classes,”** first discusses the `IData` interface and then dives into the ASP.NET AJAX `DataColumn`, `DataRow`, and `DataTable` data classes. It also implements a custom client control that can display data from any data source, such as `DataTable`, and that implements the `IData` interface.
- ❑ **Chapter 12, “Client-Server Communications,”** covers the client-server communications layer of the ASP.NET AJAX framework and its constituent classes, including detailed discussions of `WebRequest`, `WebRequestExecutor`, `WebRequestManager`, `NetworkRequestEventArgs`, and `XMLHttpRequestExecutor`, and presents several examples that show you how to use these classes in your own ASP.NET AJAX applications.
- ❑ **Chapter 13, “Consuming Web Services Via SOAP Messages,”** first discusses WSDL documents and SOAP messages in detail and then presents an example that uses the classes in the client-server communications layer of the ASP.NET AJAX framework to exchange SOAP messages with a Web service.
- ❑ **Chapter 14, “Consuming Web Services Via JSON Messages,”** provides in-depth coverage of the `WebServiceProxy` and `WebServiceError` classes and teaches you three different ways to invoke server-side methods from your client code: calling page methods, Web service methods, and Web services bridges. It also covers `.aspx` files in detail. This chapter then presents and implements fully functional replicas of the main components of the ASP.NET AJAX REST method call-request-processing infrastructure, including the `ScriptHandlerFactory`, `RestHandlerFactory`, `RestHandler`, `HandlerWrapper`, and `ScriptModule` classes, and implements an example in which these replicas are used. This chapter also uses these replicas to demystify page method calls and Web services bridges.
- ❑ **Chapter 15, “Proxy Classes,”** first covers proxy classes associated with page methods, Web services bridges, and Web services methods in detail. Next, it discusses `ScriptManager` and `ScriptManagerProxy` server controls and the role of `ScriptManagerProxy` server controls in parent/child page scenarios. This chapter then implements fully functional replicas of the main components of the ASP.NET AJAX automatic proxy-class-generation infrastructure, such as `ScriptManager`, `ServiceReferenceCollection`, `ServiceReference`, `ClientProxyGenerator`, and `RestClientProxyHandler`, and you’ll see for yourself how this infrastructure generates the proxy classes associated with page methods, Web services bridges, and Web services methods. This chapter then implements an example that uses these replicas.

- ❑ **Chapter 16, “Behaviors,”** begins by providing in-depth coverage of the `Behavior` base class and its methods and properties, and shows you how to derive from this base class to implement your own custom behaviors. It then discusses the ASP.NET AJAX control toolkit behavior base class named `BehaviorBase`, and shows you how to derive from the `BehaviorBase` class to implement your own custom toolkit behavior.
- ❑ **Chapter 17, “Script and Extender Server Controls,”** implements fully functional replicas of those components of the ASP.NET AJAX server-side framework that are deeply involved in the internal functioning of two important types of server controls, known as script controls and extender controls, to help you gain a solid understanding of these server controls, how they interact with their associated client-side components, how they differ from one another, and how to implement your own custom script controls and extender controls. The components of the ASP.NET AJAX server-side framework whose replicas these chapter implements include `IExtenderControl`, `ExtenderControl`, `IScriptControl`, `ScriptControl`, `ScriptDescriptor`, `ScriptComponentDescriptor`, `ScriptBehaviorDescriptor`, `ScriptControlDescriptor`, `ScriptReference`, `ResolveScriptReference`, `ScriptReferenceCollection`, and `ScriptManager`. This chapter then implements custom script and extender server controls, and you’ll gain the skills you need to develop your own custom script and extender server controls.
- ❑ **Chapter 18, “Web Services Bridges and Transformers,”** first walks you through the implementation of a Web services bridge-enabled script server control that uses the Amazon Web services. Then it discusses ASP.NET AJAX transformers in detail, including `XmlBridgeTransformer` and `XsltBridgeTransformer`, and uses them to enhance the Web services bridge-enabled script server control. This chapter also shows you how to implement your own custom transformers.
- ❑ **Chapter 19, “UpdatePanel and ScriptManager,”** uses numerous examples in which you learn how to enable asynchronous partial page rendering, how to use triggers, and several different ways to conditionally update an `UpdatePanel` server control, including by setting its `ChildrenAsTrigger` property, by directly adding it to another `UpdatePanel` server control, by indirectly adding it to another `UpdatePanel` server control via a user control, by indirectly adding it to another `UpdatePanel` server control via a content page, and by explicitly calling its `Update` method from your code. This chapter then implements two base custom partial-page-rendering-enabled server controls named `BaseMasterDetailControl` and `BaseMasterDetailControl2`.
- ❑ **Chapter 20, “Using UpdatePanel in User Controls and Custom Controls,”** implements three custom partial-page-rendering-enabled server controls named `MasterDetailControl`, `MasterDetailControl2`, and `MasterDetailControl3`, a custom partial-page-rendering-enabled data control field named `MasterDetailField`, and a partial-page-rendering-enabled threaded discussion forum user control. This chapter also implements pages that use these partial-page-rendering-enabled custom server controls, data control field, and user control.
- ❑ **Chapter 21, “Page Life Cycle and Asynchronous Partial Page Rendering,”** follows the `Page` object through its life cycle phases to process the first request to a partial-page-rendering-enabled Web page to help you gain a solid understanding of the ASP.NET AJAX asynchronous partial-page-rendering infrastructure and its main components, such as the `ScriptManager` and server-side `PageRequestManager`, `UpdatePanel`, `UpdatePanelTrigger`, `UpdatePanelControlTrigger`, and `AsyncPostBackTrigger` classes. This chapter also implements a custom `UpdatePanel` server control named `CustomUpdatePanel` and a custom trigger named `AsyncMultiPostBackTrigger`.

- ❑ **Chapter 22, “ASP.NET AJAX Client-Side PageRequestManager,”** first provides a comprehensive coverage of the instantiation and initialization process of the current client-side `PageRequestManager` instance, where you also learn about this instance’s `pageLoaded` event and its associated `PageLoadedEventArgs` event data class. It also shows an example in which this event is used. This chapter then dives into the process through which the current client-side `PageRequestManager` instance makes an asynchronous page postback request to the server, and you also learn about this instance’s `initializeRequest` and `beginRequest` events. It also shows examples in which you’ll learn how to use these events in your own ASP.NET AJAX applications.
- ❑ **Chapter 23, “Asynchronous Partial Page Rendering: Server-Side Processing,”** follows the `Page` object through its life cycle phases to process an asynchronous page postback request where you’ll learn about the role of the server-side `PageRequestManager`, `RetrievePostData`, `ScriptManager`, `UpdatePanel`, `ScriptRegistrationManager`, and triggers in generating the final response text. This chapter also implements a page that enables you to inspect the server response text.
- ❑ **Chapter 24, “Asynchronous Partial Page Rendering: Client-Side Processing,”** follows the client-side `PageRequestManager` instance through its life cycle phases to process the server response to an asynchronous page postback request where you’ll see for yourself how the current client-side `PageRequestManager` manages to parse the server response text, download the required scripts, and update the required `UpdatePanel` server controls on the client side. You’ll also learn about the `pageLoading` and `endRequest` events of the current client-side `PageRequestManager` instance and their associated `PageLoadingEventArgs` and `EndRequestEventArgs` event data classes. This chapter shows examples in which these events and their associated event data classes are used. It also covers the `PageRequestManagerTimeoutException`, `PageRequestManagerServerErrorException`, `PageRequestManagerParserErrorException`, and `InvalidOperationException` exceptions that the current client-side `PageRequestManager` instance raises. Finally, it implements a custom error handler and a page that uses this error handler.
- ❑ **Appendix A, “XML Script,”** provides comprehensive coverage of the ASP.NET AJAX `xml-script`, which enables you to program declaratively with little or no imperative or procedural JavaScript code. This appendix covers the main components of the ASP.NET AJAX `xml-script` parsing infrastructure, such as `MarkupContext` and `MarkupParser`, and you’ll learn through numerous examples how to enable the clients of your client classes to declaratively instantiate and initialize instances of your classes in `xml-script` without writing any procedural JavaScript code. You’ll also learn how to extend the ASP.NET AJAX `xml-script` parsing infrastructure to add support for custom parsing of your own client classes.
- ❑ **Appendix B, “Binding,”** covers ASP.NET AJAX binding in detail. The `BindingBase` client class, built-in and custom transformers, and the `Binding` client class are discussed in depth.
- ❑ **Appendix C, “Actions,”** discusses the ASP.NET AJAX actions including the `IAction` client interface, the `Action` base class, actions in `xml-script`, and built-in actions such as `InvokeMethodAction`, `SetPropertyAction`, and `PostBackAction` in detail.
- ❑ **Appendix D, “Data Control,”** first provides a comprehensive coverage of the ASP.NET AJAX `DataControl` base class and its methods, properties, and events. Then it implements a custom data control named `CustomTable` that derives from the `DataControl` base class, and uses the ASP.NET AJAX type description capabilities to display any type of data records.

- ❑ **Appendix E, “Templated Controls,”** first covers the `ITemplate` client interface, `TemplateInstance` client class, and `Template` client class in detail. Then it develops a custom template named `TemplateField` that derives from the `Template` class and supports its own `parseFromMarkup` static method, which tells the ASP.NET AJAX xml-script parsing infrastructure how to parse an instance of the `TemplateField` class declared in xml-script. Finally, it develops a custom templated data control that enables its clients to use `TemplateField` instances in xml-script to specify different types of HTML markup texts for rendering different types of database fields.
- ❑ **Appendix F, “ListView,”** begins by providing an overview of the ASP.NET AJAX `ListView` client control and its methods, properties, and events, and goes on to present examples in which this client control is used to display data records downloaded from a backend Web service. Then it dives into the internals of the `ListView` client control and its methods, properties, events, and surrounding classes and interfaces such as `ITask`, `_TaskManager`, and `ListViewRenderTask`. You’ll learn the skills you need to develop a custom templated data control as complex as the `ListView` client control.

What You Need To Use This Book

You’ll need the following items to run the code samples in this book:

- ❑ ASP.NET AJAX Extensions 1.0
- ❑ ASP.NET Futures
- ❑ Windows Server 2003, Windows 2000, Windows XP, or Windows Vista
- ❑ Visual Studio 2005, Visual Studio 2005 Express Edition, Visual Studio 2008, or Visual Studio 2008 Express Edition
- ❑ SQL Server 2005 or SQL Server 2005 Express Edition

You can download free copies of Visual Studio 2005 Express Edition or Visual Studio 2008 Express Edition and SQL Server 2005 Express Edition from <http://msdn.microsoft.com/vstudio/express/> and ASP.NET AJAX Extensions 1.0 and ASP.NET Futures from <http://ajax.asp.net/downloads/>.

Conventions

To help you get the most from the text and keep track of what’s happening, we’ve used a number of conventions throughout the book.

Boxes like this one hold important, not-to-be forgotten information that is directly relevant to the surrounding text.

Tips, hints, tricks, and asides to the current discussion are offset and placed in italics like this.

As for styles in the text:

- ❑ We *highlight* new terms and important words when we introduce them.
- ❑ We show keyboard strokes like this: Ctrl+A.
- ❑ We show file names, URLs, and code within the text like so: `persistence.properties`.
- ❑ We present code in two different ways:

```
In code examples we highlight new and important code with a gray background.
```

The gray highlighting is not used for code that's less important in the present context, or that has been shown before.

Source Code

As you work through the examples in this book, you may choose either to type in all the code manually or to use the source code files that accompany the book. All the source code used in this book is available for download at <http://www.wrox.com>. Once at the site, simply locate the book's title (either by using the Search box or by using one of the title lists) and click the Download Code link on the book's detail page to obtain all the source code for the book.

Because many books have similar titles, you may find it easiest to search by ISBN; this book's ISBN is 978-0-470-10998-4.

Once you download the code, just decompress it with your favorite compression tool. Alternately, you can go to the main Wrox code download page at <http://www.wrox.com/dynamic/books/download.aspx> to see the code available for this book and all other Wrox books.

Errata

We make every effort to ensure that there are no errors in the text or in the code. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, like a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata you may save another reader hours of frustration and at the same time you will be helping us provide even higher-quality information.

To find the errata page for this book, go to <http://www.wrox.com> and locate the title using the Search box or one of the title lists. Then, on the book details page, click the Book Errata link. On this page you can view all errata that have been submitted for this book and posted by Wrox editors. A complete book list including links to each's book's errata is also available at www.wrox.com/misc-pages/booklist.shtml.

If you don't spot "your" error on the Book Errata page, go to www.wrox.com/contact/techsupport.shtml and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page and fix the problem in subsequent editions of the book.

P2P.WROX.COM

For author and peer discussion, join the P2P forums at p2p.wrox.com. The forums are a Web-based system for you to post messages relating to Wrox books and related technologies and interact with other readers and technology users. The forums offer a subscription feature to e-mail you topics of interest of your choosing when new posts are made to the forums. Wrox authors, editors, other industry experts, and your fellow readers are present on these forums.

At <http://p2p.wrox.com> you will find a number of different forums that will help you not only as you read this book, but also as you develop your own applications. To join the forums, just follow these steps:

1. Go to p2p.wrox.com and click the Register link.
2. Read the terms of use and click Agree.
3. Complete the required information to join, as well as any optional information you wish to provide, and click Submit.
4. You will receive an e-mail with information describing how to verify your account and complete the joining process.

You can read messages in the forums without joining P2P, but in order to post your own messages you must join.

Once you join, you can post new messages and respond to messages other users post. You can read messages at any time on the Web. If you would like to have new messages from a particular forum e-mailed to you, click the Subscribe to this Forum icon by the forum name in the forum listing.

For more information about how to use the Wrox P2P, be sure to read the P2P FAQs for answers to questions about how the forum software works as well as many common questions specific to P2P and Wrox books. To read the FAQs, click the FAQ link on any P2P page.