Introduction

We thought we had found the Holy Grail of software development when we started building web applications several years ago. Previously we had been developing thick client applications that required a lengthy installation process every time a new version of the company's application was released. The application was deployed to several hundred users scattered across the country, and much to our dismay we had to watch as the complex and error-prone installation process continually caused headaches and angst for developers and users alike.

Deploying an application through a browser seemed like a much more palatable option because it would eliminate the need to install software on the client computer. So, like many others, our organization moved swiftly to deploying applications to the web.

Despite the relative ease of deployment, web applications still had their share of issues. Most notable from a user's perspective was the significant loss of rich interactivity provided by the user interface. Web applications were constrained to the basic set of widgets provided by HTML. Worse yet, interacting with the server required a complete refresh of the page, which was disconcerting to users who were familiar with rich client/server applications.

We always considered this constant need to refresh the page a serious liability of web applications and often experimented with ways to avoid a page refresh whenever possible. We even at one point considered writing a Java applet that would handle the communication between the browser and the server. However, it soon became apparent that as more web applications were deployed, users simply got used to the constant page refreshes, and our zeal for finding alternatives slowly faded.

Fast-forward five years. Even before the term *Ajax* was coined, asynchronous communication between the browser and server using the `XMLHttpRequest` object was creating a buzz within the developer community thanks to applications such as Google Suggest and Gmail. The `XMLHttpRequest` object had been available in Internet Explorer for several years, but now that it was being supported by other browsers, it was poised for a breakthrough. We added Ajax functionality to an existing Java application we happened to be working on at the time, and we were so impressed with the results that we thought, "Hey, somebody should write a book about this." Thus, the seeds for this book were sown.

An Overview of This Book

Pro Ajax and Java is written to give you, the Java developer, all the tools you need to add Ajax techniques to your existing or future applications. Our motto while writing this book was, "Everything you need to know; nothing you don't." We assume that as a reader of this book, you are already an experienced web application developer. Because of this, we focus on the topics that are most likely new to you: Ajax and its associated tools and techniques.

Chapter 1 is a whirlwind tour of Ajax. We cover the basics of the `XMLHttpRequest` object along with some of the common gotchas of Ajax development. We also cover some of the common patterns you'll see in Ajax development.

Chapter 2 provides an overview of the tools that will make developing Ajax applications easier. We touch on JavaScript editors, formatters, and obfuscators. Debugging can be a real pain, which is why we've shown you a wealth of tools that treat JavaScript like the first-class citizen that it is. For starters, we'll show you how to log without using alerts. Of course you'll also want to test your code, which is why we review JsUnit and Selenium, proving that testing isn't just for server-side code.

Although Ajax certainly isn't rocket science, that doesn't mean you want to do all the heavy lifting yourself. In Chapter 3 we take a look at a variety of frameworks and toolkits that you'll want to leverage to make developing Ajax applications a snap. We'll review Dojo, Prototype, script.aculo.us, Taconite, and the Yahoo! library.

Chapter 4 continues our look at libraries, focusing on those that are of particular interest to the Java developer. We'll review DWR and AjaxTags, giving you an idea of the strengths of each library.

The second half of the book shows you how to integrate Ajax techniques into four of the leading Java frameworks. Chapter 5 discusses the wildly popular Struts framework. We'll show you how to take advantage of Struts validation, and we also discuss design considerations with Struts and Ajax.

Chapter 6 covers the component-based Tapestry framework. After an overview of Tapestry, we show you how to leverage the Ajaxified Tacos components to spiff up your applications.

You can hardly turn around these days without running into a Spring-based application, and in Chapter 7, we show you how to leverage Ajax in the Spring space. We show you how using DWR makes developing Ajaxified Spring applications a snap.

Of course no discussion of Java web frameworks would be complete without a look at JavaServer Faces. After showing you the basics of JSF, we show you how you can write your own Ajax components, and we also introduce you to prebuilt Ajax components.

Obtaining This Book's Source Code

All the examples in this book are freely available from the Source Code section of the Apress website. Point your browser to `www.apress.com`, click the Source Code link, and find *Pro Ajax and Java* in the list. From this book's home page you can download the source code as a ZIP file. The source code is organized by chapter.

Obtaining Updates for This Book

Despite our best efforts, you may find an occasional error or two scattered throughout the book—although we hope not! We apologize for any errors that may be present in the text or source code. A current errata list is available from this book's home page on the Apress website (`www.apress.com`) along with information about how to notify us of any errors you may find.

Contacting Us

We value your questions and comments regarding this book's content and source code examples. Please direct all questions and comments to `proajaxandjava@gmail.com`. We'll reply to your inquiries as soon as we can; please remember, we (like you!) may not be able to respond immediately.

Thank you for buying this book! We hope you find it a valuable resource and enjoy reading it as much as we enjoyed writing it.

Best regards,
Nathaniel T. Schutta and Ryan Asleson