There’s no compiler to catch errors! There’s no good IDE with productivity tools such as code completion! There’s no good debugging environment!"

These are just a sampling of the myriad of excuses people have for avoiding JavaScript development. Ever since the browser wars of the late 1990s, JavaScript has had a reputation as being an error-prone, difficult development language. This reputation is based largely on the early, bugged-riddled versions of JavaScript that ran in the early, bugged-riddled editions of Netscape Navigator and Internet Explorer. To make matters worse, JavaScript was often written by nonprogrammers who got JavaScript to work by trying everything until something seemed to work. Freely available scripts were also available on the Web, and many of these scripts were rather poor quality, further perpetuating JavaScript’s reputation as a substandard programming environment.

Today’s JavaScript implementations are vastly superior to those found just a few years ago. Thanks to JavaScript syntax and behavior being standardized by the ECMA, a stable JavaScript version, and better implementation by Web browsers, programming in JavaScript is a much more enjoyable endeavor. In addition, tools such as full-fledged debugging environments, Ajax-specific debugging tools, and JavaScript error consoles provide developers with many more development tool options.

Some of the issues still remain; for example, at the time of this writing, no useful JavaScript IDE, which offers productivity tools such as code completion, exists. In addition, JavaScript is still often written by those who have no background in programming. However, more tools are available today that greatly reduce the amount of pain associated with JavaScript development and, by extension, Ajax development.

In this chapter, you’ll explore some of the tools you can use when things go wrong and don’t work as expected, also known as debugging. Having tools available to help you diagnose problems is half the battle. Once you’ve mastered these tools, debugging is actually kind of fun!

**Debugging Ajax Requests with Greasemonkey**

Ajax requests greatly enhance the user experience because they occur in the background without freezing the browser or otherwise indicating that a request to the server is taking place. The bad side effect of this behavior is that the page is more difficult to debug. Assuming that a problem with the Ajax request or response exists, it’s difficult for the developer to know