While this is obviously a simple example, you can probably see how using custom parameters can be useful in running test suites or test pages under various circumstances. Of course, you’ll probably want to add those special query strings to your browser’s bookmark folder for ease of use later.

Another interesting standard parameter is debug. Don’t confuse this parameter with the debug trace level—they aren’t the same. The debug parameter is for those doing development work on JsUnit. You probably won’t do much with this query string, but it’s there should your curiosity be piqued (see Figure 6-24).

Two other standard parameters make sense only in relation to the JsUnit Server: submitResults and resultId. They’ll make more sense when we cover the JsUnit Server in just a minute, but adding submitResults=true to your test runner’s browser path will tell JsUnit to send the results of the test to JsUnit’s “acceptor” servlet. What does that buy you? Well, we’re spoiling the surprise, but using this parameter causes the results of the test run to create an XML representation (with the same structure as JUnit’s XML output) of the results that can be retrieved later.

The resultId query makes sense only when used with submitResults=true. Based on what little we’ve told you about how the JsUnit Server works, you might have guessed that each test run that is submitted to the JsUnit Server has some kind of unique identifier. You can let JsUnit come up with its own ID, but if you absolutely must use your favorite number, you can do so by passing it via the resultId parameter.