Using the Browser

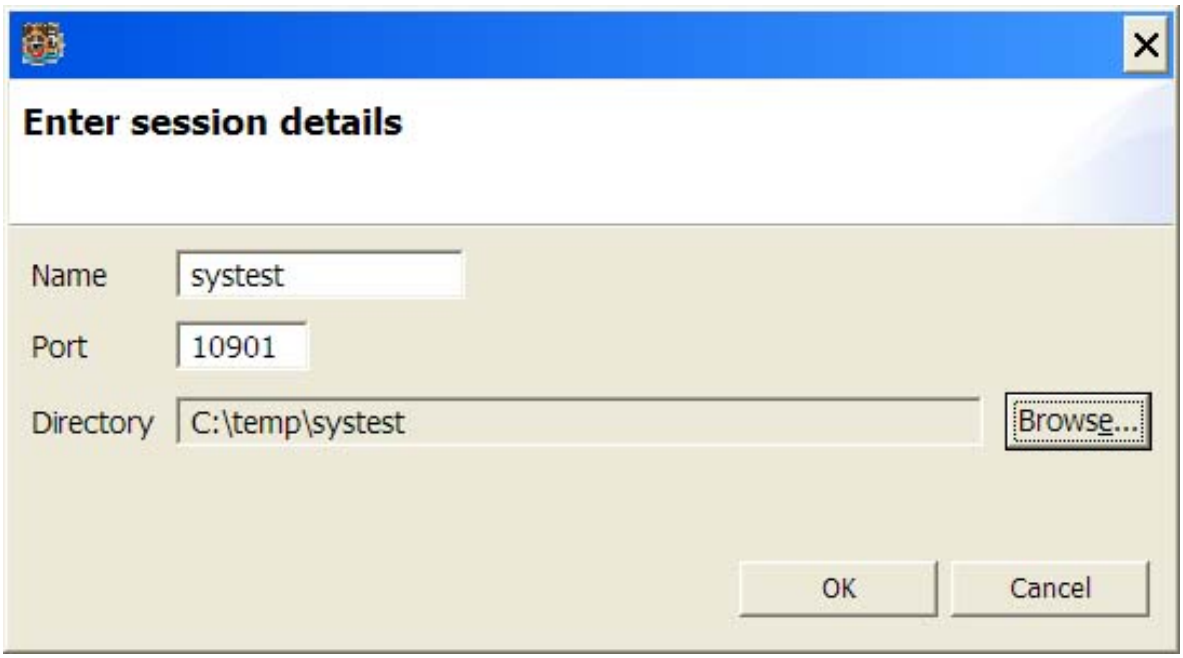
by

Table of contents

1. Adding and Removing Sessions

To start a session (that is, monitoring a UDP port), you typically would use File>Add Session and enter the details as shown:

Adding a Session



Since this is a common requirement, the browser can be invoked with command line arguments to automatically set up a single session:

- -n for the name
- -p for the port
- -d for the directory

Sessions can be removed using File>Remove Session.

2. Selecting Session Output

The easiest way to open up session outputs (that is, the XML files) is to double click on the appropriate session.

Alternatively, use File > Select Session Output to explicitly open a specific directory. All subdirectories are automatically shown in a tree view.

3. Opening an Interaction (Callstack)

To open an interaction in the editor, either double click on an XML file or use File > Open Selected File.

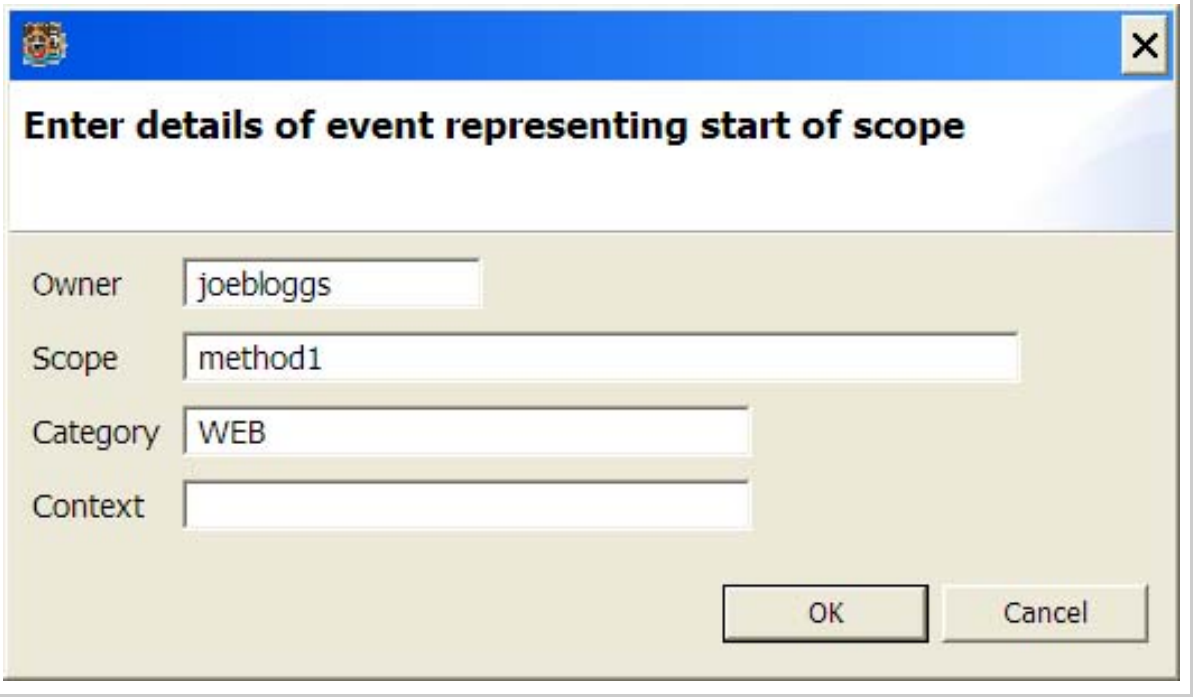
It is also possible to open an arbitrary XML file using File > Open Other File.

4. Testing

One way to see the browser in action is to write a program using the Lightweight appender (or alternatively run an appropriate ProfDog JUnit tests). An easier way though is to use its built-in test actions, under Session > Test ...

To use these actions, first select a session in the Sessions view. Then, to start a new scope (or subscope), use Session > Test > Start scope:

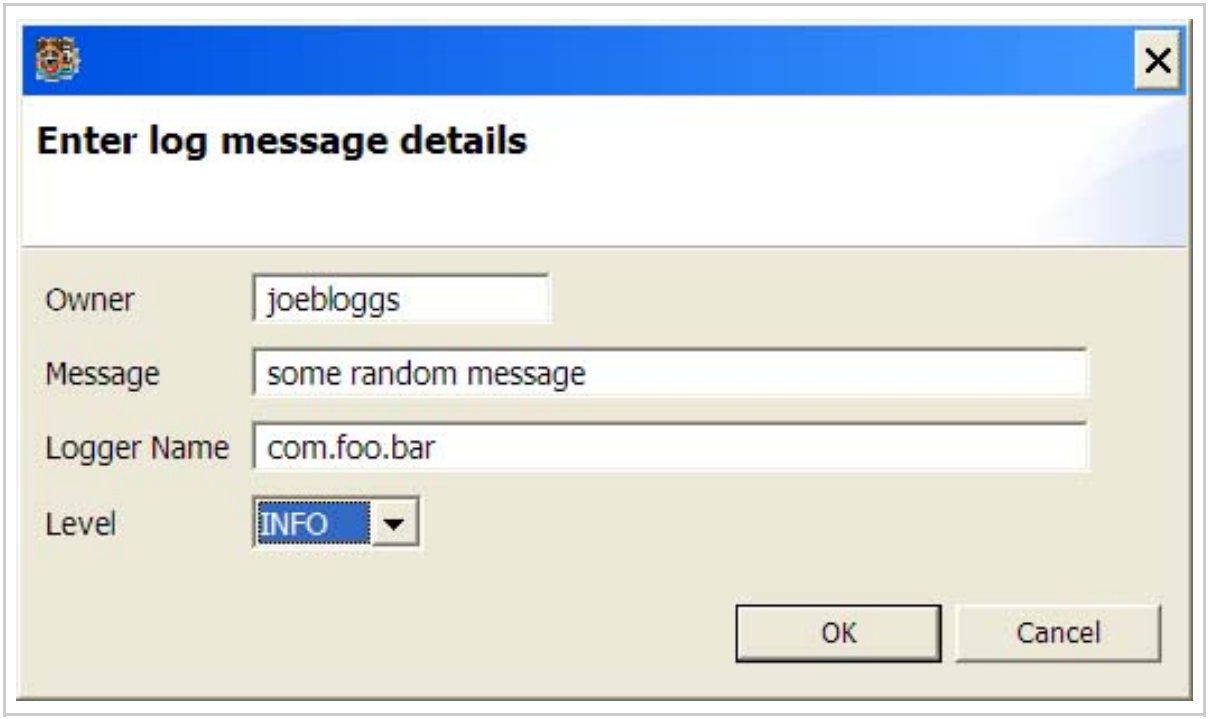
Starting a Scope



A scope should appear for the owner ("joebloggs") indicating the outermost and innermost scopes, and other details. No XML will be written until this scope completes.

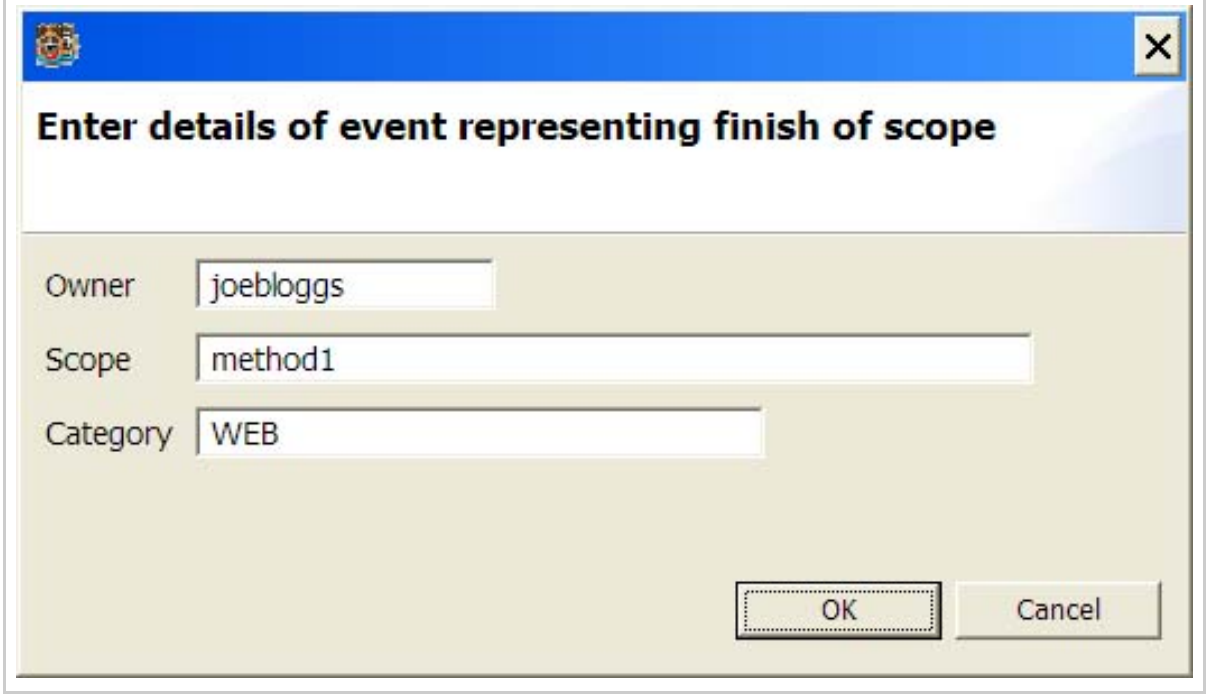
To add a log message within a scope, use Session > Test > Log message:

Logging a Message



To finish the scope, use Session > Test > Finish scope:

Finishing a Scope



Note that if the scope name does not match then the scope will not be completed.

5. Closing Scope (force close)

If (for whatever reason) the finish event does not arrive then the scope will remain unclosed and the XML will not be written out. This usually represents a programming error in the system performing the logging.

The developer can force the scope to close by selecting it and then using Session > Close Scope. We then recommend that you open up the generated XML file to determine why the scope was not closed. Then go and fix the problem so that it does not re-occur.