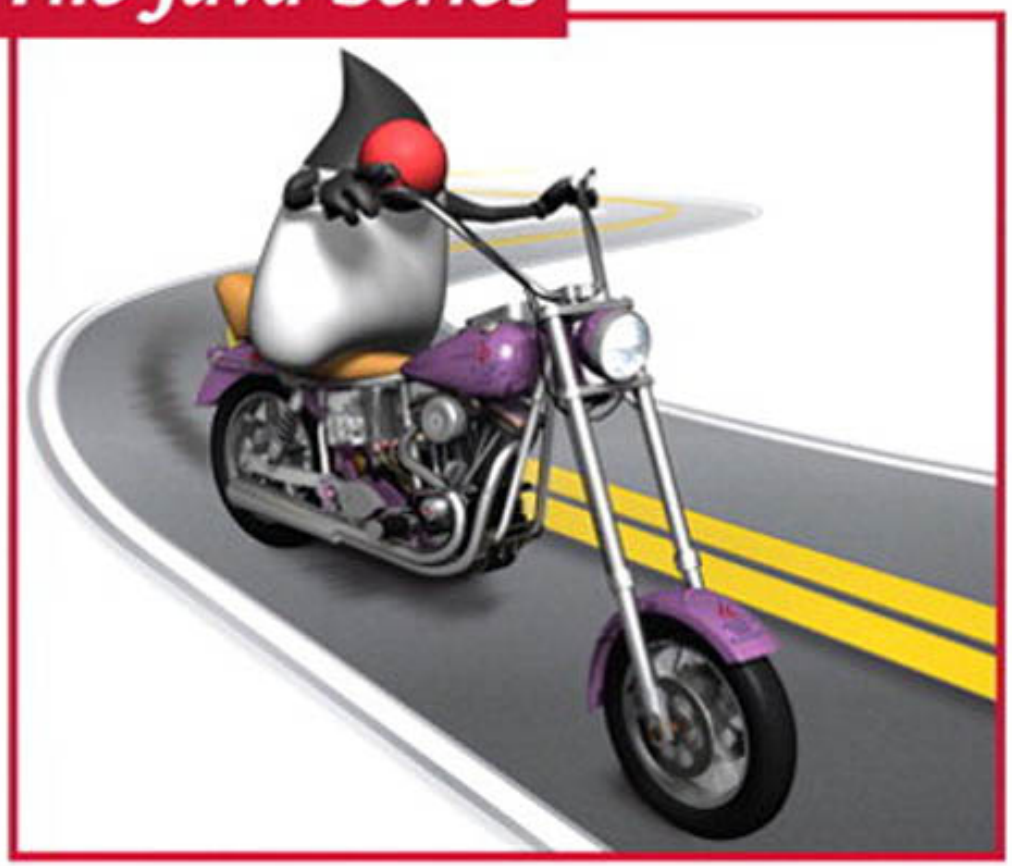
Charlie Hunt · Binu John
Forewords by James Gosling and Steve Wilson

Java™ Performance

The Java Series



Java™ Performance

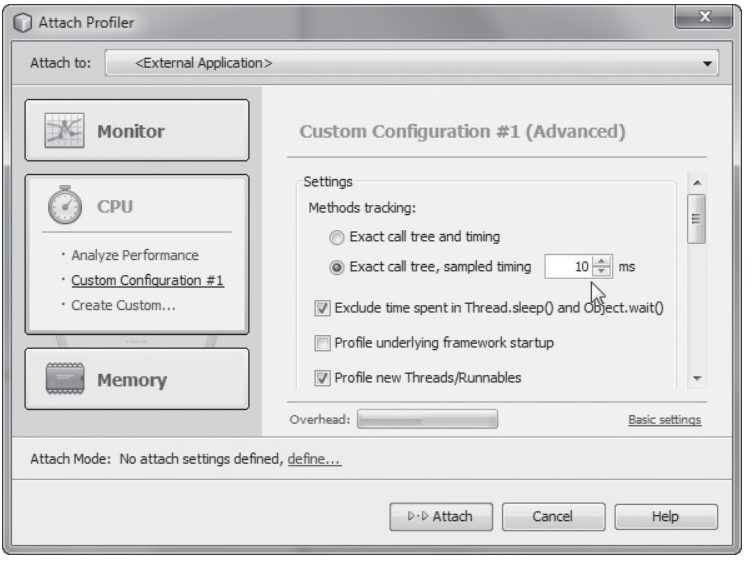


Figure 5-12 Reducing method profiling sampling interval

the profiler has attached to it. Dynamic attach allows you to attach, detach, and re-attach to the running application at any time. However, dynamic attach is not available for remote profiling or for profiling applications running on Java 5 or older JVMs.

For illustration purposes, as mentioned earlier, the example illustrated in this section assumes the target application is running remotely on a system called halas as a standalone application. Hence, on the Attach Wizard the following options are selected, also shown in Figure 5-13, Target Type is Application, Attach method is Remote, and Attach invocation is Direct.

Once the attach type is specified, you can press the Next button to proceed through the Attach Wizard.

On the next form of the Attach Wizard, you specify the hostname where the remote target application will be executed and the operating system along with specifying whether a 32-bit JVM or 64-bit JVM is being used on the target system as shown in Figure 5-14.

5. Generate the remote profiling pack.
Notice in Figure 5-13 there is a reminder that a Profiler Remote Pack is required to profile a remote application. If you have not profiled an application remotely on the target system where the application resides, you need to generate a Profiler Remote Pack. The Profiler Remote Pack makes the configuration and setup needed for the profiler to attach remotely much simpler

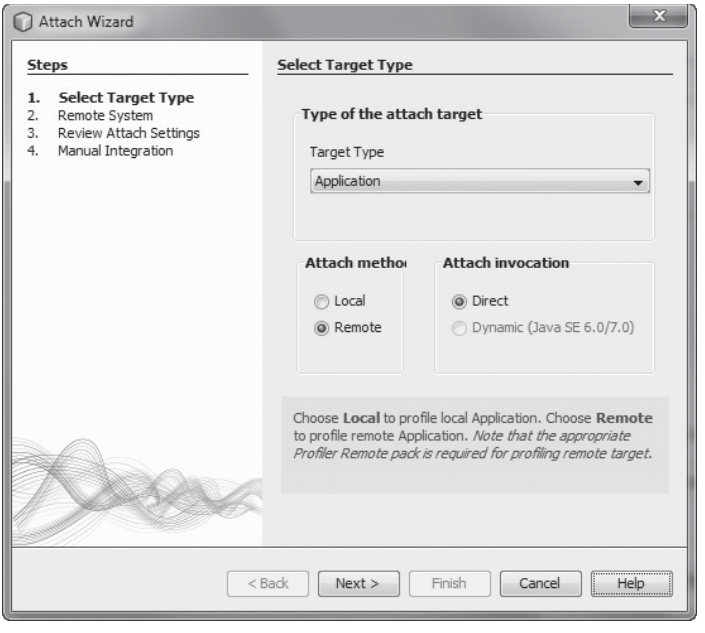


Figure 5-13 Specifying remote profiling

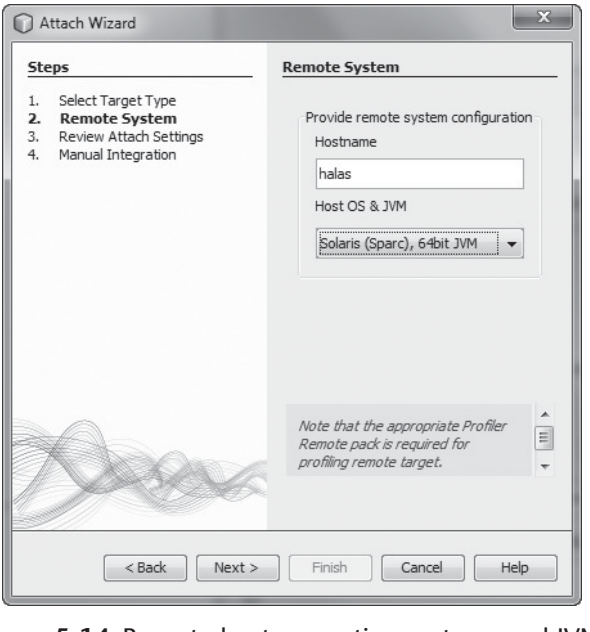


Figure 5-14 Remote host, operating system, and JVM

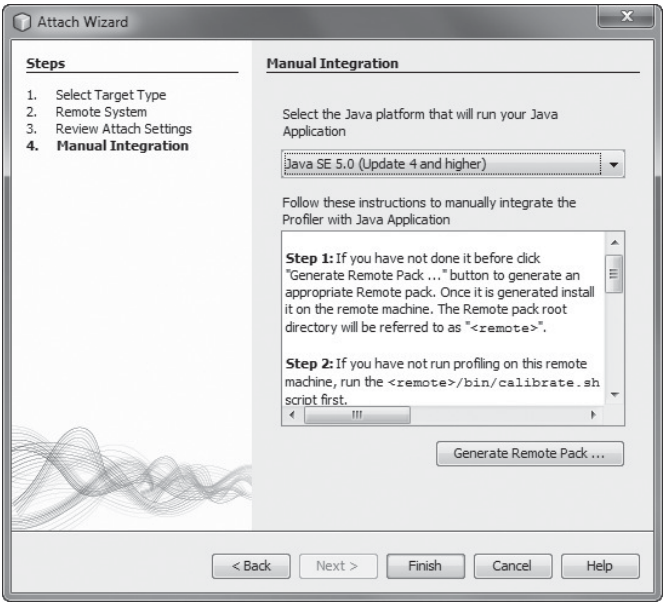


Figure 5-15 Generating Profiler Remote Pack

than doing it manually. The Profiler Remote Pack is generated by the NetBeans Profilers on the Manual Integration form of the Attach Wizard, which is one of the next forms in the Attach Wizard. Click the Next button until you reach the Manual Integration form. On the Manual Integration form, you specify the Java SE version the target application is running. In this example, the target application is running Java SE 6. Instructions on how to generate the Profiler Remote Pack are also listed on the Manual Integration form and also shown in Figure 5-15.

Review the instructions on the form and when you are ready to generate the Profiler Remote Pack, click the Generate Remote Pack button. You are prompted for a directory location in which to store the Profiler Remote Pack. Click the Finish button to complete the Attach Wizard.

6. Configure the remote system with the Remote Profiling Pack.
Now you must configure the remote system using the Remote Profiling Pack. In addition, if this is the first time you are profiling in the remote system with a target JVM, then the target JVM will also perform some calibration of the target JVM. The calibration can be performed by a script included in the Remote Profiling Pack called `calibrate.sh`.

The first task is to copy the Remote Profiling Pack to the remote target system and unzip its contents to directory on the remote system. In the instructions

given here, assume the directory that you have unzipped in the Remote Profiling Pack on the remote system is called "remote." The first task to execute on the remote system, if it has not been done previously, is to run the calibration script found in the `<remote>/bin` directory called `calibrate.sh`. Be sure to edit and update the `calibrate.sh` script with the appropriate `JAVA_HOME`, or set the `JAVA_HOME` environment variable externally to the `calibrate.sh` script prior to attempting to execute the `calibrate.sh` script. The `JAVA_HOME` environment variable must point to the base directory of the JVM you plan to use to run the remote application.

After executing the `calibrate.sh` script, you need to update the Java command line options you use to launch the target application to tell the JVM to block and wait until the Profiler has remotely connected to it. The Remote Profiler Pack has convenience scripts you can update to launch your Java application. These convenience scripts have the necessary HotSpot JVM command line option, `-agentpath`, needed for remote profiling. If you are using a Java 5 JVM you can update the `<remote>/bin/profile-15` command file or script file. If you are using a Java 6 JVM, you can update the `<remote>/bin/profile-16` command file or script file. Alternatively, you can add the appropriate `-agentpath` command line option for your platform. The command line option to specify for a Java 5 JVM or Java 6 JVM can be found in the Remote Profiling Pack's `<remote>/bin/profile-15` or `<remote>/bin/profile-16` command file or script file, respectively. When the `-agentpath` command line option is specified correctly, if you attempt to launch the target Java application, a message prints saying that the profiling agent is initializing and it is waiting for a connection from a remote profiler.

7. Start profiling; examine the data it displays and the data it collects.
Everything necessary for remote profiling is set up and configured in the previous steps. All that is left to do is launch the remote Java application and connect the Profiler to it. Launch the remote Java application with the command file or script file you updated in the `-agentpath` command line in step 6. As mentioned in step 6, when the remote Java application launches, it reports that it is waiting for the Profiler to attach. Go to your desktop system and tell the NetBeans Profiler to attach to the remote Java application. If you have forgotten how to get to the Attach Profiler panel, select the Profile > Attach Profiler option from the main menu in NetBeans IDE.

Once the NetBeans Profiler has successfully attached, the remote Java application unblocks and continues to execute. The NetBeans Profiler opens a Profiler Control Panel in NetBeans IDE with Controls, Status, Profiling

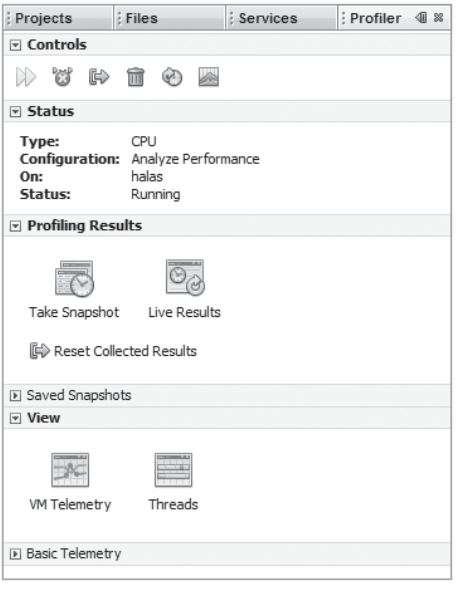


Figure 5-16 Profiler control panel

Results, Save Snapshots, View, and Basic Telemetry subpanels as shown in Figure 5-16.

Regardless of whether you are doing method profiling or memory profiling, the Profiler Control Panel looks the same. Each section of the Profiler Control Panel can be expanded or hidden by clicking the arrow icon next to the name of the section. Each of the Profiler Control Panel's subpanels is explained in more detail in the following subsections.

Controls

See Table 5-7 for an explanation of the buttons in the Profiler Control Panel Controls section.

Status

See Table 5-8 for an explanation of the entries in the Profiler Control Panel Status section.

Profiling Results

See Table 5-9 for an explanation of the entries in the Profiler Control Panel Profiling Results section.

Table 5-7 Profiler Control Panel Controls

Component	Description
	**ReRun Last Profiling** Run the last profiling command again.
	**Stop** Stops the current profiling command. Also stops the target application if the application was started by the profiler.
	**Reset Collected Results** Discards the already accumulated profiling results.
	**Run GC** Runs garbage collection.
	**Modify Profiling** Opens the Modify Profiling Task dialog box and allows you to run a new profiling command without stopping the target application.
	**VM Telemetry** Opens the VM Telemetry Overview in the Output window of the IDE, displaying smaller versions of the telemetry graphs.

Table 5-8 Profiler Control Panel Status

Component	Description
Type	The type of profiling: Monitor, CPU, or Memory
Configuration	Indicates whether the profiler was started with one of its preset configurations
On	An identifier indicating the name of the system where application is being profiled
Status	Running or Inactive

Table 5-9 Profiler Control Panel Profiling Results

Component	Description
	**Take Snapshot** Displays a static snapshot of the profiling results accumulated thus far
	**Live Results** Displays the current results of the profiling task
	**Reset Collected Results** Discards the already accumulated profiling results