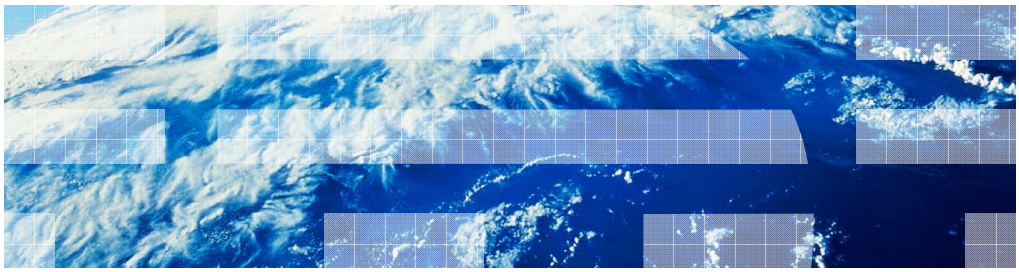
Michael Casile – Software Architect, WebSphere Serviceability
10 June 2010

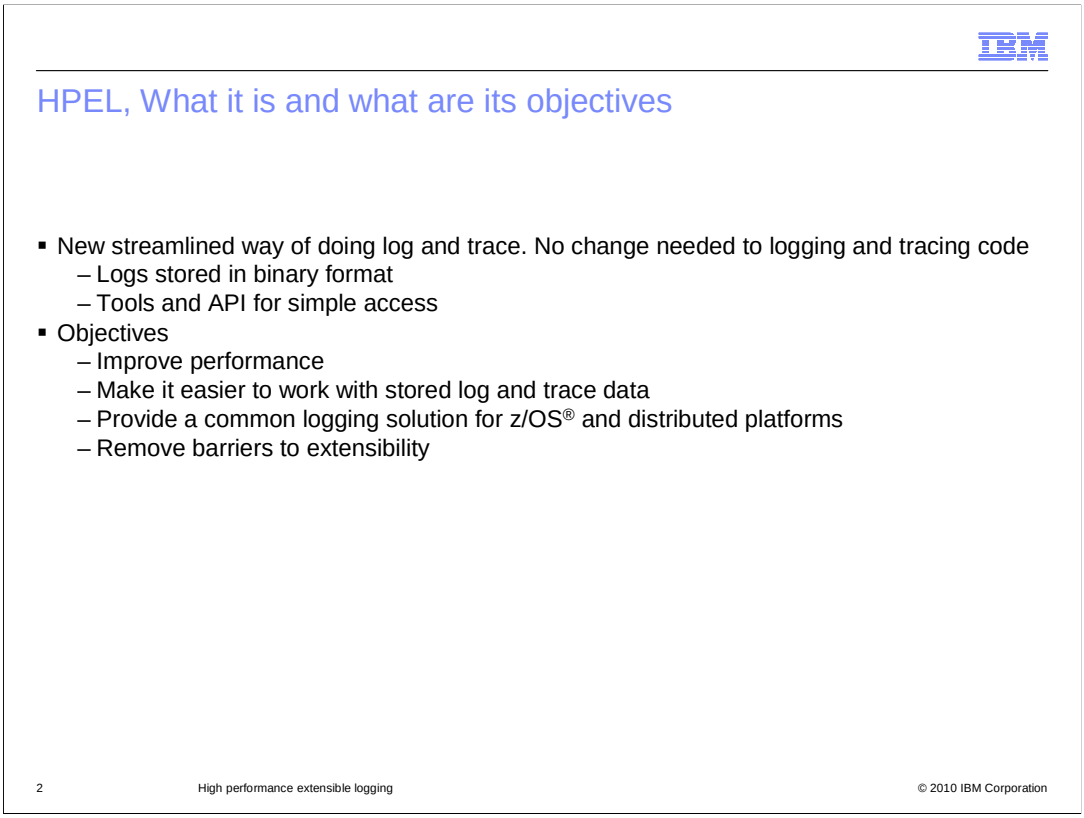
IBM WebSphere Application Server Version 8

High performance extensible logging (HPEL)



WebSphere. software

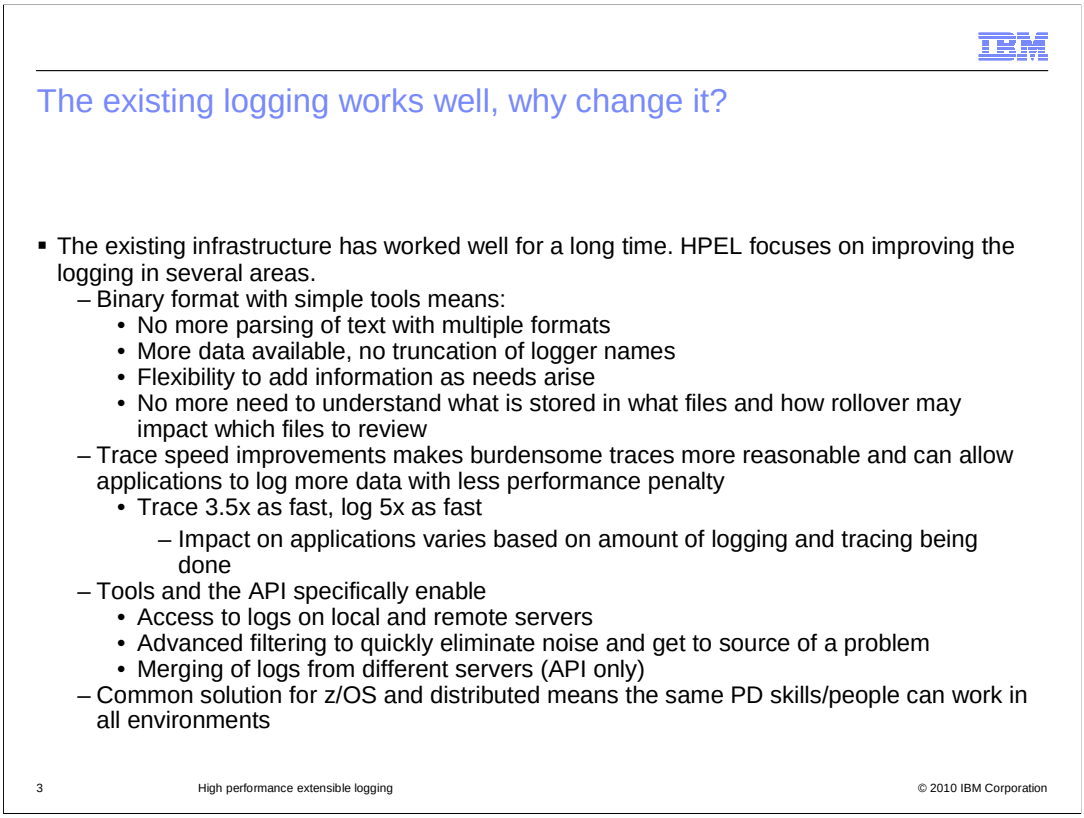
The purpose of this class is to educate the student on the new high performance extensible logging infrastructure available in WebSphere® Version 8.



HPEL is a new logging infrastructure that stores logs in a binary format that is easily consumed by the tools provided, and programmatically consumable with a powerful API that is available.

The objectives of HPEL are to improve log and trace performance, to make it easier to analyze log and trace data, to provide a common solution across z/OS and distributed platforms, and to remove the barriers to extensibility caused by a fixed text format.

By storing the data in a binary form (with a simple command to re-create the old format if needed), the restrictions caused by the existing format have been lifted.



Existing logging has worked well for years in WebSphere. HPEL is not here because logging does not work; it is an improved implementation to provide a better logging infrastructure

Using a binary format has numerous benefits:

1. Complex and error-prone parsing of the text log is eliminated

2. More data is available and no more truncation takes place

3. Adding information in the future can be done without disrupting anyone's parsing code

4. People doing analysis no longer need to understand the intricacies of the file system and rollover behaviors

When trace runs 3.5 times faster, many traces that used to be impractical on a production server may now be practical. Faster logging may mean more information is logged over time and this can help in problem determination

The simplicity and power of the tools used to access the logs locally and remotely, filter and analyze information, and even merge logs using the API means that log analysis can be faster. The ability to eliminate noise and to reQuery repeatedly can also mean that that the needle may not get lost in the haystack.

And now, the skills developed doing problem determination on distributed platforms can be reused on z/OS systems.

Enabling and configuring HPEL in WebSphere V8

- The default configuration for HPEL is a good starting point, and the configuration details are generally not needed in order to analysis or PD on the logs
- The HPEL repository includes three types of information to configure
  - Log data – binary files representing logged information
  - Trace data – binary files representing trace
  - Text log – optional text file similar to SystemOut.log. You should disable this in optimized production environments
- For a short tutorial on enabling and configuring HPEL from the WebSphere V8 administrative console, pause the presentation and click this icon:

To use HPEL, the first steps are enablement and configuration. To avoid change in WebSphere externals when migrating to Version 8, legacy logging is the default implementation. Enablement and configuration of HPEL, however, is quite simple. The default configuration is also a good starting point.

The HPEL repository contains three types of information:

1) Log data stores all log information. This is information from SEVERE to INFO and some custom levels below INFO. It also captures SystemOut and SystemErr stream calls

2) Trace data stores all information of lower severity than log data.

3) Text log is an optional text file in the format of the legacy SystemOut.log. This is ideal for development scenarios, but you can turn it off to optimize performance in production environments.

Pause this presentation and click the icon here to see a sample of enabling HPEL and doing some simple configuration actions.

Analyzing log data with the command line log viewer

- Once HPEL is enabled and configured, it is no longer required to understand the details of the individual files to do analysis. The command line log viewer is one tool which can be used to perform problem determination with the log and trace in the repository.
- For a short tutorial on the command line log viewer, pause the presentation and click this icon:

The first tool for analyzing HPEL logs is the command line log viewer. This is a simple, intuitive, and fast tool for doing analysis on the logs in problem determination efforts.

Pause this presentation and click the icon here to see a brief tutorial on the command line log viewer.

Administrative console log viewer tool

- With legacy logging, the administrative console can be used to dump ranges of lines from local and remote logs into the administrative console.
- The HPEL administrative console log viewer enables powerful filtering and formatting features to do analysis on the logs. In addition to the features seen in the command line log viewer, the administrative console log viewer can also do analysis on remote logs.
- For a short tutorial on the administrative console log viewer, pause the presentation and click this icon:

The administrative console log viewer is a powerful graphical way to do problem determination with log files. It provides all of the functionality of the command line log viewer, and enables the analysis of local and remote log files.

Pause this presentation and click the icon shown here to go through a short tutorial of the administrative console log viewer.

Exercises

- There are four sets of exercises in this section of the class. If you have a WebSphere V8 server, try the exercises. Exercises 1 - 3 should look familiar and can reinforce a working knowledge of HPEL in roughly an hour. Exercise 4 goes deeper into the API and may take a bit longer.

This next section includes four sets of exercises that allow you to get hands-on experience with HPEL. If you have access an HPEL V8 server, you should try these; it is where the instruction becomes concrete experience.

Exercise set 1, enable and configure HPEL

- Goals:
  1 Turn on HPEL
  2 Disable the Text Log
  3 Shrink the log repository down to 20 Mb
  4 Keep items in the trace repository for 12 hours (no concern for size)
- Verification:
  1 When you are done, you should have a server with HPEL turned on
  2 You should NOT have text log* files in your server log directory
  3 You should have log data and trace data directories in your log directory
  4 If you try running logViewer.sh (UNIX®) or logViewer.bat (Windows®) with no parameters, you should get lots of records scrolling on your screen
- Optional extensions for the bold:
  1 Use wsadmin to see if HPEL is enabled
  2 Use wsadmin to change the log repository maximum back to 50Mb (the default)
- To see a tutorial of how to do this, pause the presentation and click this icon:

The goal of these exercises is to walk you through the process of enabling and configuring HPEL. Some of this was displayed earlier, but here you will be doing it, and the exercises go deeper into the configuration options.

Pause the class at this point and try the exercises on your own server. When you have completed - or if you are not in position to try the exercises at this time - click the Show Me icon to watch how they are done. Note that after turning on HPEL, you can do the configuration work before exiting the administrative console and restarting the server. A restart of the server is necessary for all of this to be put into affect.

For the optional exercises, in a few slides, you will see some links to the information center. The information center has some links that include wsadmin samples.

Exercise set 2: Command line log viewer - Accessing the data

- Goals:
  - Use the command line to get help on the log viewer including an explanation of all options
  - By going to the <profileHome> ... create an old style log named oldLogsForOldDogs.log containing just log records (INFO, WARNING, and SEVERE for this exercise)
  - List just the records for thread 0
  - Look for just WARNING messages
- Verification:
  - Using the oldLogsForOldDogs.log, see if you got the right number of thread 0 records from the second query (second query may have additional records of Thread 0 was doing activity between the run of exercise 1 and exercise 2)
  - Count the warning records (type w) in oldLogsForOldLogs and compare it to the number of WARNINGs you got from the third exercise
- Optional extensions for the bold:
  - Retrieve all log records NOT from loggers beginning with com.ibm
  - Extract a repository of just WARNING and SEVERE messages and put it in a new directory
  - Use the appropriate parameter to point log viewer at the new repository, and dump it into a separate file
  - (z/OS only ...) After getting the list of processes from which you can extract, select the most recent servant and capture its logs
  - z/OS only ....) zip up your entire log directory and bring it down to a client workstation. There, extract it and run the local log viewer against it.
- To see a tutorial of how to do this, pause the presentation and click this icon:

The goal of these exercises is to familiarize you with the simplicity and power of the command line log viewer tool. It can be used to re-create the legacy log files, but it can also do so much more in the way of filtering and analysis.

Pause the class at this point and try the exercises on your own server. When you have completed - or if you are not in position to try the exercises at this time - click the Show Me icon to watch how they are done.

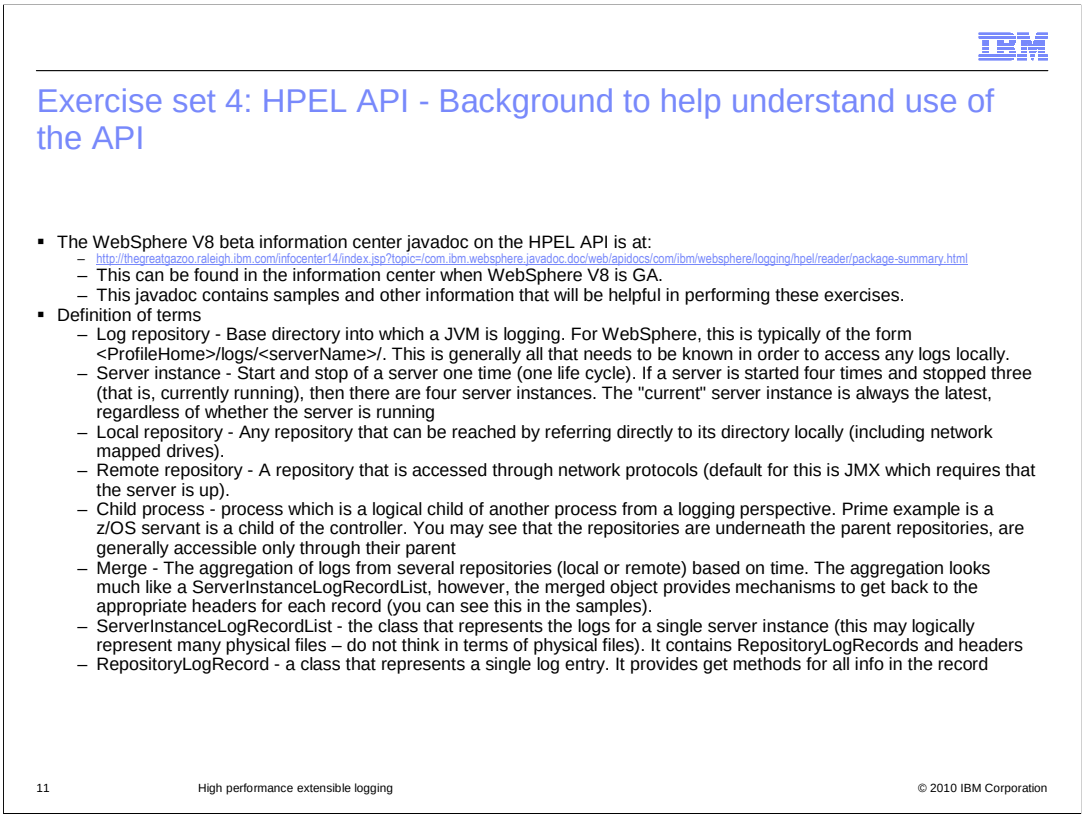
A hint on getting started is to open a command prompt (on Windows) or a terminal (or telnet session or SSH session) on UNIX or z/OS and change directory to the <profileHome> or <profileHome>/bin. Then when you have accomplished goal #1 (hint, -help), closely review all of the options, because they will provide you with what you need to accomplish the other goals.

Exercise set 3: Browser-based log viewer

- Goals:
  - 1. Select to view just the current server instance (currently running server)
  - 2. Identify the PID/Process ID for the server
  - 3. Find early message TRAS0017I and get the detailed information for what this message is about
  - 4. Find the first activity on a thread other than 00000000, then filter to include only this thread
  - 5. Show only WARNING and higher messages
- Verification:
  - 1. Records in the display should all have times associated with only the latest instance of the server
  - 2. Should see the process ID and many other process-scoped information
  - 3. Should see a pop-up that includes user action, explanation, and others
  - 4. Should see only rows for the thread selected. Select Show all Threads after to move on to goal #5
  - 5. Should see a much smaller # of rows and all should be WARNING, SEVERE, or FATAL
- Optional extensions for the bold:
  - 1. From the WARNING and SEVERE view at the end of the previous exercises, export a binary format log with your filter criteria applied.
  - 2. Use the command line log viewer to query just WARNING records from this new repository (on some browser/OS combinations, the export may not function properly)
- To see a tutorial of how to do this, pause the presentation and click this icon:

The goal of these exercises is to give you practical experience with the administrative console log viewer. Remember that, after selecting TroubleShooting -> Logs and Trace -> selecting your server -> View HPEL Logs and Trace, you will be in the administrative console log viewer panels.

Pause the class at this point and try the exercises on your own server. When you have completed - or if you are not in position to try the exercises at this time - click the Show Me icon to watch how they are done.



At this point, if you have done the first three sets of exercises, you really have a working knowledge of HPEL and its basic administration. This set of exercises focuses on the API which is ideal for tool developers and people who want to do more complex analysis and formatting.

Before you start into these exercises, it is important to cover some background information, starting with the definition of terms:

Log repository is the base directory where HPEL stores sub-directories and logs. It is typically the same directory uses by legacy logging to store logs

Server instance is a single life cycle of a server from server start to server stop. Using the API, you can access a single server instance or multiple server instances. It is at the server instance level that logs from different servers can be merged

Local repository is a repository in a directory that you can access directly from your system (either a local directory, a mapped network drive, or some other virtually local resource.

Remote repository is a repository that you will use remote protocols (JMX™ in this case) to access.

Child process is a z/OS-only term describing a process that is logically a child of another. Child processes do not have their own repository locations. They use directory structures under the parent directory. z/OS servants and adjunct processes are children of z/OS controllers.

Merge is the aggregation of logs from several repositories (local or remote).

ServerInstanceLogRecordList is a class that represents a query result set (matching your filter criteria) for a single server instance. Note that in some of these exercises, you will use code that creates an iterator of ServerInstanceLogRecordLists ... one for each life cycle of the server in the repository. For each, it will reference just the records that match your query.

RepositoryLogRecord is a class that contains the information from a single log record. It provides get methods for all data and it has a reference back to the header (from the containing ServerInstanceLogRecordList), which can be helpful in merge situations

Exercise 4: Background continued

- Building and running samples
  - Building a sample that uses only local repositories:
    - javac -cp <WasHome>/plugins/com.ibm.hpel.logging.jar:. com/ibm/sample/hpel/ReaderSamplesForExercises.java
  - Running a sample that uses only local repositories
    - java -cp <WasHomes>/plugins/com.ibm.hpel.logging.jar:. com/ibm/sample/hpel/ReaderSamplesForExercises <WasHome>/profiles/<profileDir>/logs/server1/ 1
  - Building a sample that involves using JMX to access remote repositories
    - javac -cp <WasHome>/plugins/com.ibm.ws.admin.client_8.0.0.jar:. com/ibm/sample/hpel/ReaderSamplesForExercises.java
  - Running a sample that involves using JMX to access remote repositories
    - java -cp <WasHome>/plugins/com.ibm.ws.admin.client_8.0.0.jar:. com/ibm/sample/hpel/ReaderSamplesForExercises <args>
- The .zip files in this flow contain:
  - Sample Java source for the exercises, plus some helper classes to make JMX communication a bit easier and provide some formatting options
  - Sample repositories including z/OS repositories and multi-server profiles. These allow you to learn the capabilities of high performance extensible logging without having to generate test data first.

To build and run samples accessing local repositories requires only your code and the HPEL jar. If you are using remote repositories, then use the administrative thin-client jar instead. This contains the classes from the HPEL jar and the classes needed for a JMX client accessing a WebSphere server.

Included with this material are two important .zip files. HpelJavaSampleSource.zip contains sample code that uses the APIs to accomplish common analysis functions using local and remote repositories. HpelRepositories.zip contains some sample repositories that you can use for doing your analysis. This allows you to get started right away without having to generate data first. There are z/OS and distributed platform repositories.

Exercise 4: API Exercises

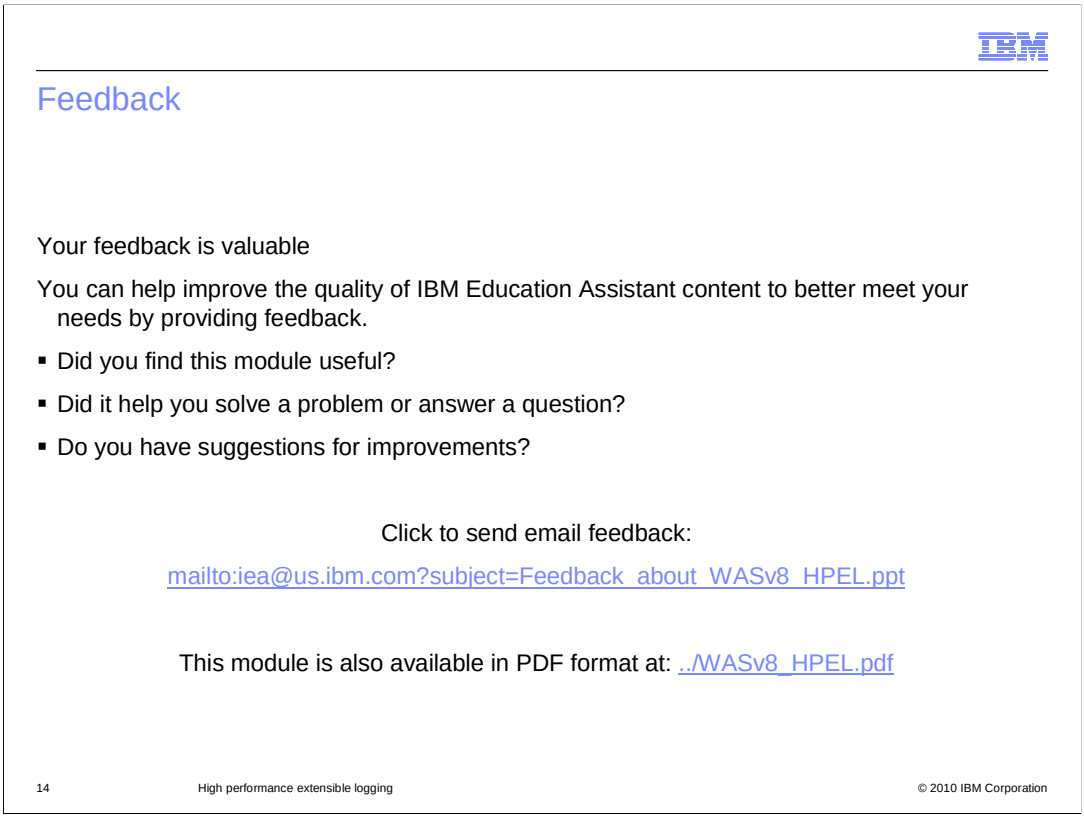
- Goals:
  1. Print date, thread, and formatted message for all WARNING and ERROR messages in a local repository
  2. Same as exercise 1 ... only this time, ONLY for the most recent server instance (life cycle of the server)
  3. Create a second server with HPEL enabled, generate workload on both servers at roughly the same time. Merge all INFO, WARNING, and ERROR messages from the most current server instance on each server. Refer back to headers to get the PID for each record and thus show PID, date, thread, and formatted message. To avoid some of the time of doing this, you can use the one of the repositories in HPEL repositories (info) for that. Note that there are multiple repository directories there, so you will need to specify one of the child directories.
  4. (z/OS only): Retrieve log for most recent controller, and get the label for all children of the controller (servants and adjuncts)
  5. (z/OS only): Retrieve and merge all records from the most recent controller and all of it's children (just WARNING and SEVERE)
- Verification:
  1. You should see a list of error and warning messages that includes all server life cycles for which HPEL was enabled
  2. This time you should see a subset of the list (or repeat if you had only one server instance out there) with just latest server life cycle
  3. Your results should see different PIDs on different records, and the rest should look similar to #2
  4. Should see just a list of controller and servants (no actual log records)
  5. Similar to #3, only printing job name and job ID would get you bonus points.
- To see a tutorial of how to do this, pause the presentation and click this icon:

13 High performance extensible logging

The goal of these exercises is to give you practical experience writing Java code that uses the HPEL API. The examples use the eclipse development environment, but any good Java development environment will work and make your job that much easier. Pause the presentation and try these exercises. The previous slide shows how to build and run these. You should start a new Java project for local samples and add the HPEL jar to your build path. You can add the remote samples to a separate Java project, or to this Java project. In either case, add the administrative thin client jar to your build path if you are using remote.

The sample source and sample repositories from the last slide can be a valuable resource as you work through these exercises.

When you have completed - or if you are not in position to try the exercises at this time - click the Show Me icon to watch how they are done.



You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, WebSphere, and z/OS are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. Java, JMX, JVM, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.