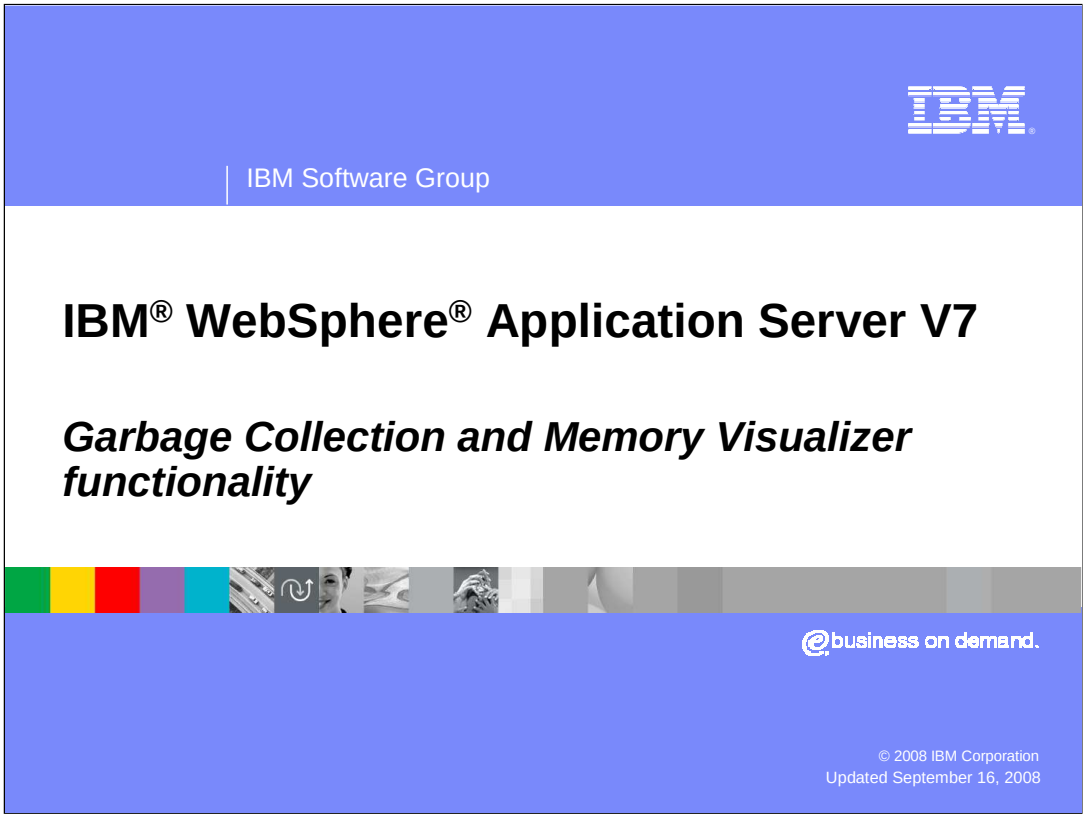


This presentation covers the functionality of the Garbage Collection and Memory Visualizer, a graphical user interface for analyzing garbage collection data.

Agenda

- Garbage Collection (GC) and Memory Visualizer functionality
  - Overview
  - Architecture
  - Features

First, this presentation will provide an overview of the GC and Memory Visualizer and its architecture. You will also learn about the plotting and reporting features in the tool.

Section

Visualizer functionality

This portion of the presentation will briefly walk you through the GC and Memory Visualizer's functionality.

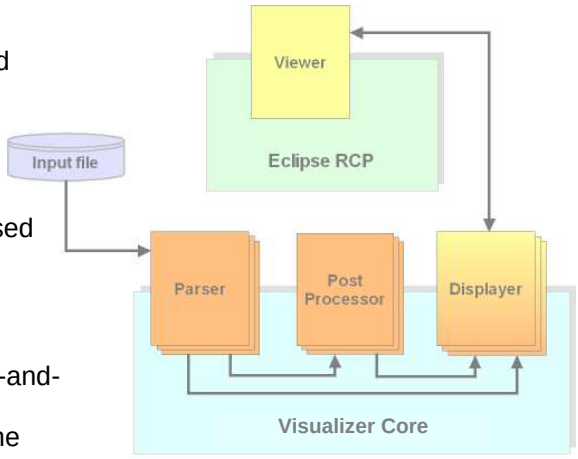
GC and Memory Visualizer overview

- The GC and Memory Visualizer is a graphical tool for visualizing for verbose garbage collection data
  - The tool parses and plots verbose GC logs and garbage collection traces
  - Processes data from all IBM SDKs, version 1.4.2 or higher
- The toolkit provides
  - Raw view of data
  - Line plots to visualize a variety of GC data characteristics
  - Tabulated reports with heap occupancy recommendations
  - View of multiple datasets on a single set of axes
  - Ability to save data as an image (jpeg) or comma separated file (CSV)

The GC and Memory Visualizer allows you to visualize your garbage collection data, as raw datasets, line plots, reports, and images. It can parse and plot data from all IBM SDKs, version 1.4.2 or higher. The “Line plot” portion of the tool displays graphs of over forty different garbage collection data characteristics – including used heap, pause times, and the reason for garbage collection being triggered. You can read in multiple sets of garbage collection logs and display them together on a single set of axes, which allows you to easily compare garbage collection behavior across multiple test runs of an application. The “Report” section of the tool contains a summary of the line plot data, along with information on general garbage collection behavior and heap sizing recommendations based on overall heap occupancy.

GC and Memory Visualizer architecture

- Parsers
  - Takes input file and builds common data model
  - Parsers available for VGC and TGC input
  - See available parsers in the Parser menu
- Post processors
  - Transforms and enriches parsed data
  - Provides recommendations, statistics
  - Updates internal data representation to support pan-and-zoom display
  - See available processors in the Postprocessor menu

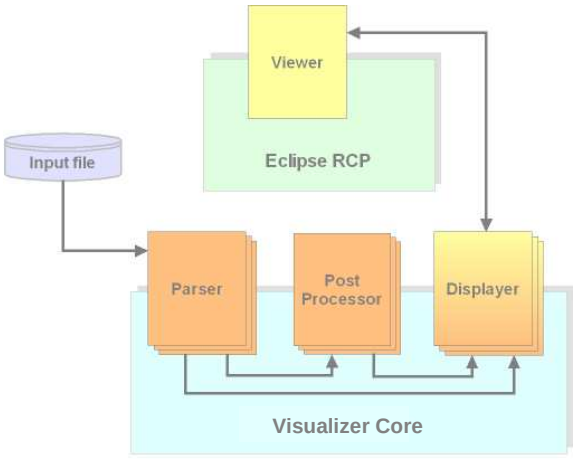


The GC and Memory Visualizer has been designed to be extensible. The core of the visualizer is componentized, and communicates with the viewer, which is where you are able to see and interact with your data. The core consists of three types of components: parsers, post processors, and displayers. Parsers take input files – like verbose GC logs – and convert them into a form that the GC and Memory Visualizer can understand. The toolkit has parsers for trace data and verbose garbage collection data. Version 1.0.2 has built-in parsers for verbose GC 5.0 and later, verbose GC 1.4.2 and earlier, realtime verbose GC, Solaris only verbose GC, and traces. You can see all of the available parsers in the Parsers menu in the GC and Memory Visualizer workbench. If there are some parsers that you will not need to use, you can turn them off in the Parsers menu.

Post processors take parsed data and manipulate it. For example, they produce statistics on the data, truncate data sets to a more manageable size, or provide extra analysis and insight into the data. Disabling a post processor can reduce the number of menu items available or alter the data being displayed. The GC and Memory Visualizer has several built-in post processors, including the limit adjuster and verbose GC derived data and summary. These processors can be configured in the tool's Postprocessor menu.

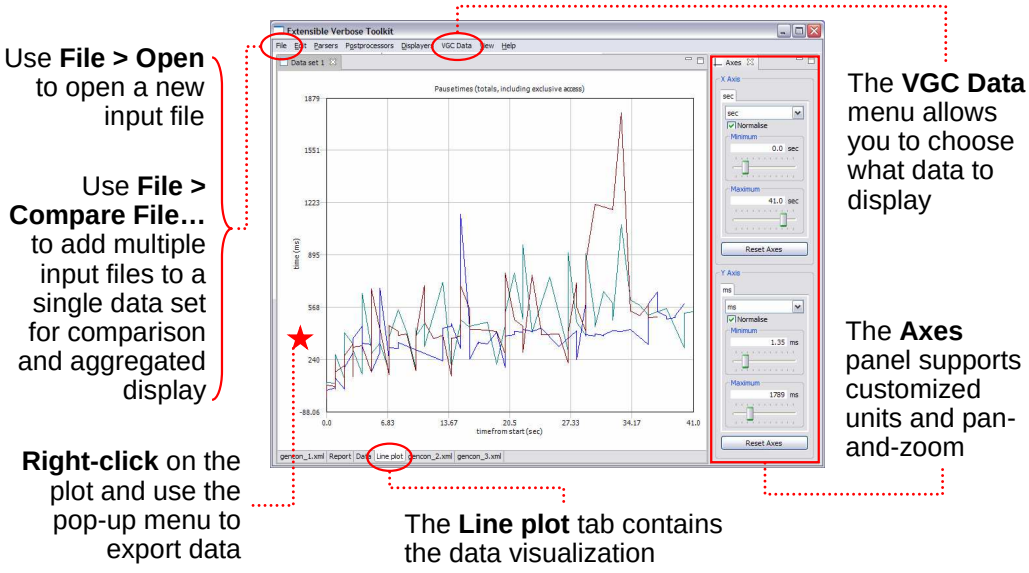
GC and Memory Visualizer architecture

- Displayers
  - Renders parsed data in a useful format
  - HTML report, text data, line plot
  - See available displayers in the Displayers menu
- Viewer
  - Requires a displayer
  - Presents text and graphical data in the Eclipse Rich Client Platform
  - Provides interactive features and display options



Displayers take parsed data and represent it in a useful form. For example, they turn data into tables, reports, and charts. The GC and Memory Visualizer has several built-in displayers that can be configured in the Displayers menu in the workbench. These include line plots, HTML reports, text data, and comma separated data. The Viewer component, which is where you can see and interact with your garbage collection data, is built on the Eclipse Rich Client Platform. It communicates with the displayer components in the core of the toolkit to be able to display graphs, pan and zoom in your data set, show reports, and more.

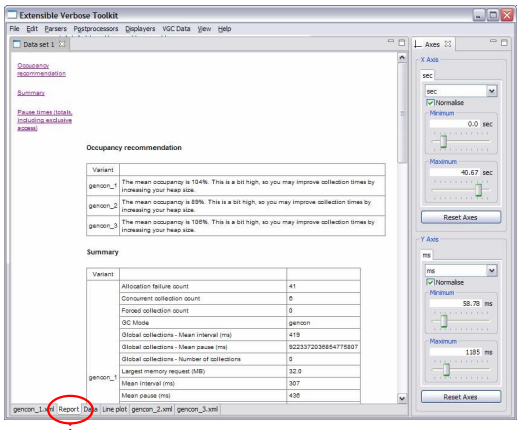
Plotting data with the visualizer



This is an example of what the GC and Memory Visualizer looks like in action. This graph shows the GC cycle pause times for three test runs of the same application, all using the generational concurrent garbage collection policy. All of the plots for your dataset are displayed in the Line plot tab. To read in a new file, use the File > Open menu in the upper right of the workbench. If you want to load in multiple input files to compare on a single set of axes, use the File > Compare File… option to add more files to your data set. There are over forty data characteristics that you can display in your line plot; these can be controlled in the VGC Data menu. If you want to save a copy of the line plot, right-click anywhere on the plot and you will see a pop-up menu that allows you to save the plot as an image file.

Reports and recommendations

- Report contents can be configured using VGC menu options
  - Occupancy recommendations tell you how to adjust heap size for better performance
  - Summary information is generated for each input in the dataset
  - Graphs included for all GC display data
- Can export as HTML by right-clicking and using the pop-up menu



The **Report** tab contains the report for the current dataset

The Report tab contains the report for your current dataset, including overall garbage collection summary statistics, graphs, and any other data that you have configured in the tool's menus. If you want to see tuning recommendations, be sure to enable the Tuning recommendation option in the VGC menu. By enabling this feature, your report describes your overall heap occupancy, provides recommendations on heap sizing and which garbage collection policy to use, and lets you know whether it looks like your application is leaking any memory. You can export an HTML version of your report by right-clicking in the report display and using the pop-up menu to save it as a file.

Tuning recommendation example

Tuning recommendation

The heap size was quite variable. This will be causing unnecessary compaction. If your application's workload is relatively steady, you should consider fixing the heap size. This should improve performance in two ways; changing the heap size is rather expensive for the garbage collector because it must compact first, and fewer collections will be required if your application is not running in a heap which is too small.

Heap usage seems to be growing over time. It increased by 51% in the last third of the log compared to the middle of the log. The heap size increased by 51% in response to the increased pressure on the heap. While this kept the change in the rate of collections to -100%, the heap growth is not sustainable. Unless the application stops growing its memory requirements, it is likely that an out memory error or severe performance degradation will eventually occur.If you don't know of a reason why the memory requirements of your application should be growing, your application may be leaking memory. Consider reviewing your application for references which are being held unnecessarily, large maps and sets, and large statically-held objects. Using weak references where appropriate may help.

A high proportion of the nursery is tenured each collection. (The average is approximately 40%.) This can lead to longer pause times for collections in the nursery, more frequent collections in the tenured area, and slower application access to these objects. Consider increasing the nursery size or the tenure age to see if this ratio can be lowered.

The application seems to be using some quite large objects. The largest request which triggered an allocation failure (and was recorded in the verbose gc log) was for 33554448 bytes.

The recommended command line is -Xminf0.1.

This is an example of the kind of tuning recommendations that the Garbage Collection and Memory Visualizer can provide. In this case, the sample application was using the generational concurrent garbage collection policy. The visualizer recognizes this and provides recommendations that are specific to tuning that garbage collection policy, like adjusting the size of the nursery or increasing the tenure age. The recommendations generally also include comments on overall heap size and usage and can contain specific command-line options to help you tune your application environment.

Types of graphs

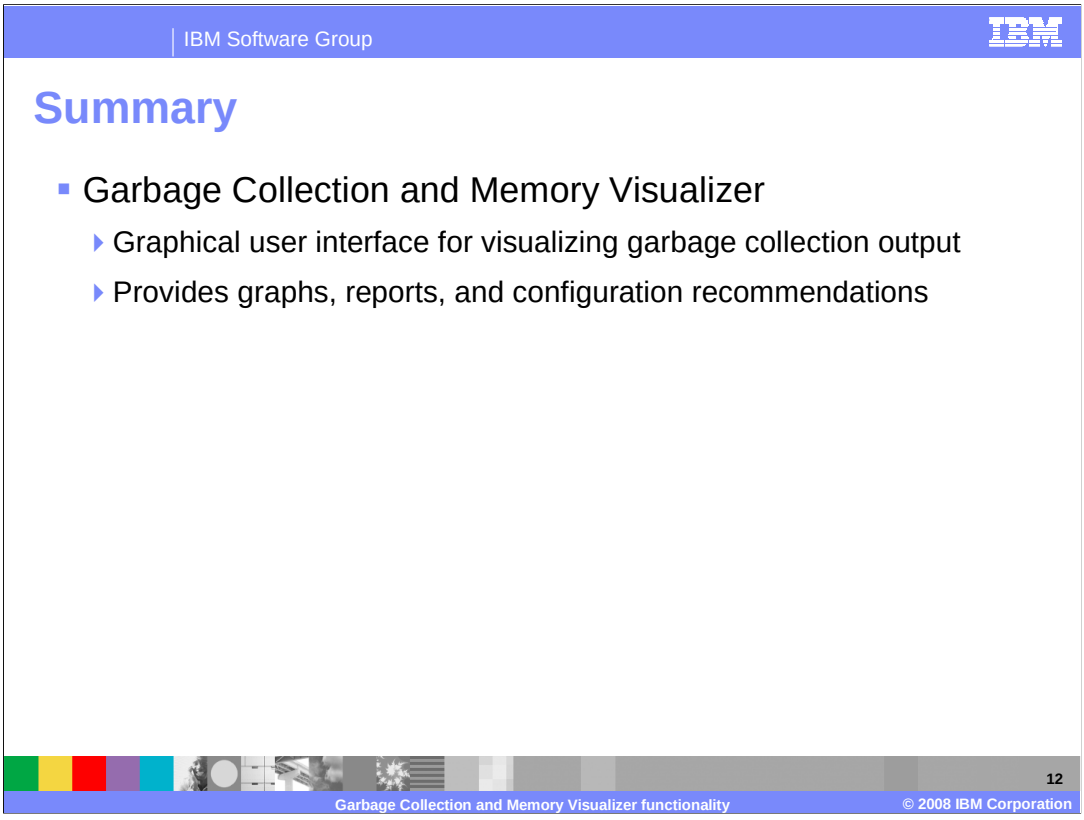
- The visualizer has built-in support for over forty different types of graphs
  - These are configured in the VGC Data menu
  - Options vary depending on the current dataset and the parsers and post-processors that are enabled
- Some of the VGC graph types are:
  - Used total heap
  - Pause times (mark-sweep-compact collections)
  - Pause times (totals, including exclusive access)
  - Compact times
  - Weak references cleared
  - Soft references cleared
  - Free tenured heap (after collection)
  - Tenured heap size
  - Tenure age
  - Free LOA (after collection)
  - Free SOA (after collection)
  - Total LOA
  - Total SOA
- *Note: Different graph types and a different menu are available for TGC output*

There are over forty data characteristics that you can display in your line plot, but not all of them are applicable to each file. For example, if you are running with the garbage collection policy that is optimized for average pause times, it does not make sense for you to look at nursery heap or tenured heap behavior, because those ideas only apply when using the generational concurrent garbage collection policy. Similarly, options in the VGC Data menu can vary depending on the processors and post-processors that you have enabled. If you do not have the VGC Summary postprocessor turned on, for instance, you are not able to see summary information or tuning recommendations for your dataset.

Section

Summary and references

This section contains a summary and links to references.



The GC and Memory Visualizer is a graphical tool for visualizing garbage collection data. It is built on top of a core architecture of parsers, postprocessors, and displayers, which communicate with an interactive viewer that runs on the Eclipse Rich Client Platform. The interface provides line plots for over forty GC data characteristics, summary reports, and general tuning recommendations.

References

- IBM Support Assistant
  http://www-306.ibm.com/software/support/isa/
- Diagnostics Guide
  http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp
- developerWorks® articles about garbage collection
  http://www-128.ibm.com/developerworks/java/library/j-ibmjava2/
  http://www-128.ibm.com/developerworks/java/library/j-ibmjava3/

This slide lists some references

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASv7_GCMVFunction.ppt

This module is also available in PDF format at: ../WASv7_GCMVFunction.pdf

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

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

developerWorks IBM WebSphere

A current list of other IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.