

Impact2011 IBM Software Accelerated Value Program

AVP-2928 WebSphere JVM Flight Simulator

http://www-01.ibm.com/software/support/acceleratedvalue/



WebSphere JVM Flight Simulator

What this lab is about	2
Lab requirements	3
What you should be able to do	3
Part 1: Lab Set Up	4
Part 2: Setup Health Center to Monitor a running WebSphere JVM (5 minutes)	5
Part 3: Use Health Center to Investigate Application Errors (15 minutes)	12
Part 4: Trigger a Memory Leak and Generate Java Dumps (15 minutes)	26
Part 5: Optional - Analyzing a Java Core for Evidence of a Memory Issue (5 minutes)	
Part 6: Using the ISA and the Memory Analyzer to Analyze a Heapdump (20 minutes)	37
Part 7: Using the IBM Extensions to Memory Analyzer for Further Memory Analysis (10 minutes)	50
Appendix A: Optional - Using the ISA and GCMV to Analyze GC Data (10 minutes)	58
Reference Links	68

What this lab is about

This lab is provided **AS-IS**, with no formal IBM support.

In this lab you will use IBM tools to monitor and diagnose JVM issues experienced by an 'in flight' running WebSphere Application Server. A badly implemented web application will be used to simulate common problems such as **memory leaks**, unexpected garbage collection cycles triggered by **System.gc()**, **large application objects** and **large HTTP session** sizes.

It will also describe the data needed to debug these issues, and introduce the Java problem determination tools available as part of the IBM Support Assistant (ISA) including Health Center, Garbage Collection Memory Visualizer and Memory Analyzer.

Impact2011 IBM Software Accelerated Value Program

Lab requirements

List of system and software required for the attendee to complete the lab.

- WebSphere Application Server V7.0.0.13
- IBM Support Assistant V4.1 with the following tools installed:
 - ► IBM Monitoring and Diagnostic Tools for JavaTM Health Center
 - ► IBM Monitoring and Diagnostic Tools for JavaTM Memory Analyzer
 - ▶ IBM Monitoring and Diagnostic Tools for JavaTM Garbage Collection and Memory Visualizer
- IBM Extensions for Memory Analyzer

What you should be able to do

At the end of this lab you should be able to:-

- Install and configure Health Center to monitor a running WebSphere JVM
- Identify bugs in running code such as unnecessary calls to System.gc(), large object allocations and memory leaks
- Analyze a Javacore.txt file for evidence of a memory issue
- Analyze verbosegc logs to inspect memory usage and garbage collection performance
- Understand the basic techniques for debugging Java[™] memory issues with Memory Analyzer
- Analyze a heap dump to determine those objects consuming the most heap space
- Use IBM extensions to Memory Analyzer to perform product specific memory analysis of a system dump

Part 1: Lab Set Up

Impact2011 Changing the Way Business and IT Leaders Work

_ Login to the VMWare image with the username/password below:

Username : Administrator

Password : Happy2Be

NOTE:

Due to the physical memory on the VMware image being used for this lab please understand that certain operations may take time to perform – please be patient. The expected duration of this lab is **70 minutes** and each part has an estimated duration. Some lab sections and individual steps are marked **optional** so you can skip them to save time if necessary.

Part 2: Setup Health Center to Monitor a running WebSphere JVM (5 minutes)

NOTE:

Impact²⁰

ng the Way Business and IT Leaders Wor

Health Center is a free low-overhead diagnostic tool for monitoring applications running on an IBM Java Virtual Machine.

The Health Center tool is provided in two parts:

The **Health Center client** is a GUI-based diagnostics tool for monitoring the status of a running Java Virtual Machine (JVM). The Health Center client is installed into the IBM Support Assistant (ISA) workbench.

The **Health Center agent** provides the mechanism by which the Health Center client obtains information about your Java application.

The agent uses a small amount of processor time and memory on the server (less than 3%). It is installed by default in an IBM JVM at Java 5 SR8 or IBM Java 6 SR1 and above.

However, it is good practice to update the Health Center agent to benefit from the latest features in the client which is updated frequently. In addition, the agent must be manually enabled by setting a JVM property. Both these tasks are described later.

ISA 4.1 and the Health Center client and agent are already installed. Double click the "IBM Support Assistant 4.1" shortcut on the desktop.



__ Click "Launch Activity->Analyze Problem" and select the Health Center tool. Then click "Launch".

Impact2011 IBM Software Accelerated Value Program

Changing the Way Business and IT Leaders Work

ch Activity 🐌 🚹 Home 🗙 🗟 Analyze Problem 🗙		
👕 Tools 📒 Collect Data 🥳 Guided Troubleshooter		
Case/Incident		
default		
Tools Catalog	Find New Tool Add-ons	Description
Tool Name	Version	The IBM Monitoring and Diagnostic Tools for Java Health Center is a lightweight tool that monitors
IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Vi	is 2.4.0.20101007	active IBM Virtual Machines for Java with minimal
IBM Monitoring and Diagnostic Tools for Java™ - Health Center	1.3.0.20101104	performance overhead. The Health Center sugges live tuning recommendations for garbage collections
IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer [Tech Preview]	0.6.0.201101042253	profiles methods including call stacks, and highlig contended locks.
		Restrictions
		None
		Associations
		Tool is not associated with any products

_____ Click the "Enable an application for monitoring" link. This will display the help contents, including a link to the latest Health Center agent code.

Impact2011 Changing the Way Business and IT Leaders Work

Status - IBM Support Assistant Workber		
File Administration Update Data W Support Assistant		
*] 🖂 🖭 🛛 🖷 🖬 🔒	🚈 Health Center: Connection wizard	
🖹 Status 🛛	Enable an application for monitoring (Select Cancel to import an existing file.)	nt. See the help
G Classes	This wizard guides you through connecting the Health Center to a currently running Java application.	
Environment	NOTE: Your application must be enabled for monitoring before Health Center can be connected. For instructions on how to setup an application to enable live monitoring, click on the link below before continuing.	nt. See the help
🗊 Garbage Colle		nt. See the help
☑ I/O		nt. See the help
Locking		nt. See the help
📟 Native Memor	< Back Next > Finish Cancel	nt. See the help
🖻 Profiling	 No data available. Open saved Health Center data or connect to an application running the Health Center ag section "Troubleshooting". 	ent. See the help
Connection 🔀		- 8
8 No data		

_ Click the "Installing the Health Center agent" link.

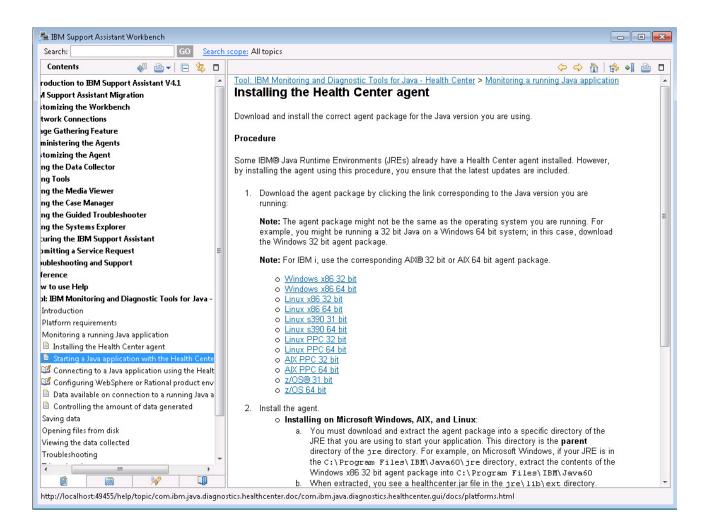


_____ Observe that the Health Center client ships the latest agent code for various Java versions (do not click any links, see note below).

Impact2011 IBM

Note:

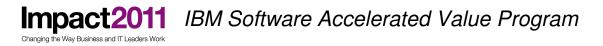
No action is required to update the agent code for this lab. To save time, the "Windows x86 32 bit" agent was already downloaded and installed. The installation process consists of extracting a small zip file over the existing WebSphere JRE. This updates the Health Center agent files originally shipped with the JVM to the latest version shipped with the Health Center client. **Only a few Health Center specific files are overwritten, not the entire JRE.**



Close the window showing help page "Installing the Health Center agent".

Note:

To save time, WebSphere Application Server has already been configured with "-**Xhealthcenter**" as a "Generic JVM Argument" via the WebSphere administration console. This enables the Health Center agent on the default port number (1972). If this port was in use, or if you needed to monitor multiple JVMs



on the same host, the port number can be customized with "-Xhealthcenter:port=<port_number>". No action is required for this lab.

Connect the Health Center client to the agent in the WebSphere JVM (which is already running in this lab). Return to the "Health Center Connection Wizard" window and click next to scan for available connections.

樳 Health Center: Connection wizard				
Enable an application for monitoring	l			
(Select Cancel to import an existing file.)				
This wizard guides you through connecting the	Health Center t	o a currently runn	ing Java applicatio	on.
NOTE: Your application must be enabled for mo	nitoring before	Health Center can	be connected.	
For instructions on how to setup an application	to enable live m	onitoring, click on	the link below bef	ore continuing.
Enabling an application for monitoring				
	< Back	Next >	Finish	Cancel

___ Connect on the default port number by clicking "Next", and "Next" again

Impact2011 IBM Software Accelerated Value Program

Changing the Way Business and IT Leaders Work

_

🖨 Health Center: Connection wizard		
JVM Connection Details Enter the details of the JVM you want to connect (Select Cancel to import an existing file)	: to.	
Hostname:	localhost	•
Port:	1972	
🔽 Scan next 100 ports for available connections		
Use basic authentication		
Username:		
Password:		
	< Back Next > Finish	Cancel

_____ Click "Finish" to start the data collection, the "Connection" panel will confirm the connection status.

Impact 2011 IBM Software Accelerated Value Program

樳 Health Center: Connection wizard			
Search for a JVM			
Select a detected JVM.			
Finished searching ports.			
localhost:1972			
	< Back Ne	ext > Finish	Cancel
■ Connection 🛛			
🔲 📗 localhost:1972			
4.3 MB received Last updated 06:21:02			



Part 3: Use Health Center to Investigate Application Errors (15 minutes)

Note:

WebSphere Application Server is running the Plants by WebSphere sample web application which has been modified with some deliberate programming errors.

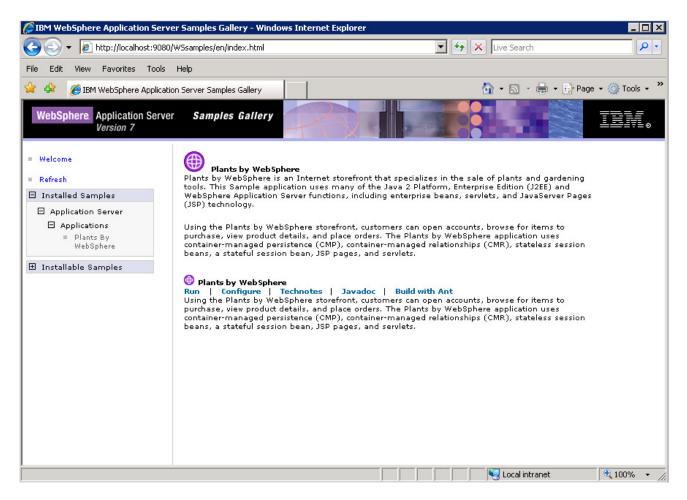
_____ First verify the Plants by WebSphere sample is running. Double click the "First Steps" desktop short cut. Note this launches a DOS window and the GUI may take a few moments to appear.



_____ Click the "Samples Gallery" link in the "First Steps" window to launch a browser. Navigate to the "Plants by WebSphere" Sample and click the "Run" link.

Impact2011 IBM Software Accelerated Value Program

Changing the Way Business and IT Leaders Work



_____ The Plants sample will launch in a new browser window. Feel free to have a look around, but to avoid some deliberate mistakes **do not** click any products on the "Accessories" tab.

Impact2011 IBM Software Accelerated Value Program

Changing the Way Business and IT Leaders Work



Next you will use the Jmeter load generating tool to simulate some user requests to the Plants web application. Some of these user requests will trigger deliberate errors which you will diagnose using Health Center.

Double click the Jmeter shortcut on the desktop.



_____ Click "File->Open" and navigate to "C:\labfiles\Jmeter script\Impact 2011 Plants.jmx". Click "Open".



📟 Apache JMeter (2.4 r961	953)
<u>File Edit Run Options</u>	Help
Test Plan	WorkBench Name: WorkBench Comments:
	Look In: Jmeter script
	Impact 2011 Plants.jmx
	File Name: Impact 2011 Plants.jmx
	Files of Type: JMeter [.jmx]
	Open Cancel

_____ Click "Run->Start". Wait a few moments until the number of threads has reached 2, as indicated in the right hand corner of the Jmeter window.

🕮 Impact 2011 Plants.jmx (C:\labfiles\Jmeter script\Impact 2011 Plants.jmx) - Apache JMeter (2.4 r961953)					
<u>File Edit Run Options H</u> elp					
		2/2 🗖			
P- ▲ Test Plan P- Ⅳ Scenario 1	Test Plan				
- Nv16/t/mh-submenu-marker-right.	Name: Test Plan				
HTTP Header Manager	Comments:				

_____ Switch to the Health Center window which should already be monitoring the WebSphere JVM, if not make a new connection with File->New Connection.

Observe the Health Center status panel. This summarizes the main categories of data that Health Center is monitoring, and also summarizes current recommendations. Note that the data categories to be collected can be customized from the Monitored JVM menu, but for now leave this at the default setting.

Changing the Way Business and IT Leaders Work

Impac

	oort Assistant Workbench		
File Administration	n Update Data Monitore I nt	ed J∿	M Window Help
*	⊖ • î ≥ \$ m №	200	
Launch Activity	🦺 🏦 Home 🙁 🗟 Ar	aly:	ze Problem 🗶 Status Summary 🔀
🚼 Status 🔀			- [
	G <u>Classes</u>	(i)	Some monitored data was dropped in the JVM being monitored because it was produced at a faster rate than the Health Center client was able to consume. This dropped data might have included information about classes loaded. Some loaded classes might not be shown.
	Environment	0	No configuration problems were detected.
	Garbage Collection	۵	The application seems to be using some quite large objects. The largest request which triggered an allocation failure was for 10000 KB.
	<u>■</u> <u>I/O</u>	(i)	Your application has opened 8,427 files and closed 8,204 files.
	Locking	9	No problems detected.
	📟 Native Memory	0	No native memory leak detected.
	10 Profiling	۵	The method ShoppingServlet.deliberateSlowMethod() is consuming approximately 83% of the CPU cycles. It may be a good candidate for optimization.
🖫 Connection 🖾			- C
📃 📫 💭 54 MB	ost:1972 I received: Last updated 06:08: lata was dropped because it was		uced faster than the client could consume it. Around 13% of the data was lost.
Connected to localh	ost:1972. 54 MB received.		

Start by analyzing where the WebSphere JVM is spending most of its time and see if any optimizations can be made.

_____ Click the "Profiling" link.

Health Center will show the results of its "sampling based" method profiler. This means it takes a periodic sample of the methods running and reports which are consuming the most time in the JVM.

_____ Sort the table of data by "Tree %" by clicking the "Tree %" column heading.

Within the Health Center, collections of methods are organized into structures called trees. You should see that in this case, a "**ThreadPool\$Worker.run()**" method represents the **top of a tree** which is consuming a very high percentage of the JVM's time.

However, also note the value in the "Self (%)" column, which indicates that the method

"ThreadPool\$Worker.run()" is actually using a low percentage of the JVM's time. Therefore the problem must be in some code called by the "ThreadPool\$Worker.run()" method, i.e. further down the tree / method call stack.

As incoming HTTP requests are handled by WebSphere using the "ThreadPool\$Worker" class, this gives a clue that there could be something wrong in a running web application.



🖕 Method profile - IBM Support Assistant Workbencl					
File Administration Update Data Monitored JVM Wind	ow Help				
Support Assistant		•			₽ ₽ ₩33 3 *
🕴 🖂 🔝 🕒 👄 🗊 ڬ 🏦 📖 🔥 🏣					
		-			
🔝 Status 🛛 🗖	🔏 Method profile 🛛	3			- 8
G <u>Classes</u>	Filter methods:				Apply Clear
Environment 😣	Samples	Self (%) Self	Tree (%) 🔻	Tree	Method
	6	0.087	82.2		com.ibm.ws.util.ThreadPool\$Worker.run()
Garbage Collection 😣	3	0.044	56.6		com.ibm.io.async.ResultHandler.runEventProcessingLoop(bc
	0	0.0	56.6		com.ibm.io.async.ResultHandler\$2.run()
I/O (1)	0	0.0	56.5		com.ibm.io.async.ResultHandler.complete(com.ibm.io.async
	0	0.0	56.5		com.ibm.io.async.AsyncFuture.completed(long)
🔒 Lockina 🛛 🔗	0	0.0	56.5		com.ibm.io.async.AsyncChannelFuture.fireCompletionAction
	0	0.0	56.5		com.ibm.io.async.AbstractAsyncFuture.invokeCallback(com.
I Native Memory	0	0.0	56.5		com.ibm.ws.tcp.channel.impl.AioReadCompletionListener.ful
	0	0.0	56.5		com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.pro
Refiling ④	0	0.0	56.3		com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.har
I O Profiling (1)	0	0.0			com.ibm.ws.webcontainer.channel.WCChannelLink.ready(cc
	0	0.0	56.3		com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.har
Connection 🛛 🗖 🗖	1	0.015	56.2		com.ibm.ws.webcontainer.WebContainer.handleReguest(co
	0	0.0			com.ibm.ws.webcontainer.WSWebContainer.handleReguest
	0	0.0	54.8		com.ibm.ws.http.channel.inbound.impl.HttpICLReadCallback
	0	0.0	53.7		com.ibm.ws.webcontainer.servlet.ServletWrapper.handleRe
	0	0.0	53.7		com.ibm.ws.webcontainer.servlet.ServletWrapperImpl.hanc
localhost:1972	0	0.0	53.7		com.ibm.ws.webcontainer.servlet.ServletWrapper.handleRe
📃 📗 45 MB received: Last updated 08:02:33	0	0.0	53.6		com.ibm.ws.webcontainer.servlet.ServletWrapper.service(i
Some data was dropped because it was produced	0	0.0		-	javax.servlet.http.HttpServlet.service(javax.servlet.http.H
Faster than the client could consume it. Around 5%	0	0.0	53.6		javax.servlet.http.HttpServlet.service(javax.servlet.Servlel
of the data was lost.	0	0.0	52.2		com.ibm.websphere.samples.plantsbywebspherewar.Shopp
	0	0.0			com.ibm.websphere.samples.plantsbywebspherewar.Shopp
		7.000			Ŀ
Analysis and Recommendations 🖾 📃 🗖	S Invocation paths	🔀 🔏 Called method	ds 💊 Timeline		- 6
(1) The method		10			
ShoppingServlet.deliberateSlowMethod() is consuming					
approximately 44% of the CPU cycles. It may be a good					
candidate for optimization.			2 2 22 22		
		Sele	ct a method in th	e table to s	see how it was invoked

Reorder the table to see results for individual methods by clicking "Self %".

Now you can see the individual method "**deliberateSlowMethod**" in the **ShoppingServlet** class is using a high percentage of the JVM's time. Note, the "Self" and "Tree" columns (without the % symbol) are a graphical indication that the method is very expensive, and is part of an expensive tree.

Select the expensive method in the table by clicking it once.

🗞 Method profile				
Filter methods:				Apply Clear
Samples	Self (%) 🔻	Self	Tree (%)	Method
4327	51.3		51.4	com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet.deliberateSlowMethod() 📃 🚽

Impact 2011 IBM Software Accelerated Value Program

The "**Invocation paths**" tab shows what is calling the "deliberateSlowMethod". The "**timeline**" tab shows when the "deliberateSlowMethod" was invoked.

Also notice that Health Center has automatically identified the erroneous method and has highlighted this fact in the "Analysis and Recommendations" section.

http://www.commonscience.com/support Assistant Workbench				
File Administration Update Data Monitored JVM Window Help				
Support Assistant	•		€ ©	
*] 🖂 🗄 😑 💼 🖬 🔓 📖 🗞 🚴				
Launch Activity 🦆 🏦 Home 🛪 🗟 Analyze Problem 🛪 🤶 S	tatus Summary 🛛 🐞 Pr	ofiling ×		
🔝 Status 🔀 🗖 🗖	Nethod profile 🛛			- 8
O <u>Classes</u>	Filter methods:			Apply Glear
Environment 😣	Samples Self (%)	r Self Tree (%) Tree	Method
The second			8.7	com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet.deliberateSlowMethod()
Garbage Collection 8			.58 1	com.ibm.ws.util.BoundedBuffer.take()
<u>□</u> I/O ①			.41	java.lang.ClassLoader.defineClassImpl(java.lang.String, byte[], int, int, java.lang.Object)
<u>■ 10</u> 0			.54 1	com.ibm.ejs.util.am.AlarmManagerThreadCSLM.run()
🔒 Locking 🛛 🛇			.22 1	java.util.Timer\$TimerImpl.run()
			.97	com.ibm.io.async.Timer.run()
I Native Memory 😣			.81 .07	java.lang.Object.wait(long, int) com.ibm.oti.vm.BootstrapClassLoader.loadClass(java.lang.String)
			.18	com.ibm.rmi.iiop.CDRReader.availableData(int, int, int)
R Profiling			.97	com.ibm.ws.util.ThreadPool.execute(java.lang.Runnable, int, long)
			.78	java.util.concurrent.ConcurrentSkipListMap.doRemoveFirstEntry()
Connection X			2.1	java.lang.ClassLoader.loadClass(java.lang.String, boolean)
	36 0	.34 1	.93 1	java.lang.J9VMInternals.verify(java.lang.Class)
			.32	sun.io.ByteToCharSingleByte.convert(byte[], int, int, char[], int, int)
			.33	java.io.DataOutputStream.writeUTF(java.lang.String, java.io.DataOutput)
			.74	com.ibm.ws.util.BoundedBuffer.waitGet_(long)
localhost:1972			.45	com.ibm.oti.vm.VM.findClassOrNull(java.lang.String, java.lang.ClassLoader)
50 MB received: Last updated 08:04:59			0.4	java.util.Properties.load(java.io.Reader)
50me data was dropped because it was produced faster than the client			0.3	com.ibm.rmi.iiop.ColocatedInputStream.mark() com.ibm.rmi.iiop.CDRInputStream.read_octet_array(byte[], int, int)
could consume it. Around 4% oF the data was lost.			.24	iava.lang.StringBuilder.append(iava.lang.String)
			0.4	iava,io.Win32FileSvstem.normalize(iava.lang.String)
	4	.21	0.4	
the second secon			1	
🗖 Analysis and Recommendations 🛛 🗧 🗖		Called methods 🛛 🗞 Timeline		
	Methods that call ShoppingSer	/let.deliberateSlowMethod()		
A The method ShoppingServlet.deliberateSlowMethod() is consuming	🖃 🔞 ShoppingServlet.delib			
approximately 59% of the CPU cycles. It may be a good candidate for	E-O ShoppingServlet.			
optimization.	E O ShoppingServec.			
	⊞–100 HttpServi	et.service (100%)	1	
			-	
×	1			

_____ As the Plants sample is clearly suffering with at least one slow method, type "com.ibm.websphere.samples" in the "Filter Methods" box and click "Apply".

You can see only the "**ShoppingServlet.deliberateSlowMethod**" in the Plants sample has a high value for the "Self (%)".

	com.iom.webspr	ere.samples			Apply
Samples	Self (%) 🔻	Self	Tree (%)	Tree	Method
9285	65.7		65.8		com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet.deliberateSlowMethod()
26	0.18		0.9		com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet.deliberateLargeObjectAllocation()
26	0.18		0.18		com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet.deliberateSystemGC()
2	0.014		0.014		com.ibm.websphere.samples.plantsbywebsphereejb.Inventory.pcReplaceField(int)
1	0.0071		0.0071		com.ibm.websphere.samples.plantsbywebsphereejb.Inventory.pcGetGenericContext()
1	0.0071		0.19		com.ibm.websphere.samples.plantsbywebsphereejb.EJSRemote0SLCatalog_59598632.getItemInventory(java.lang.String)
1	0.0071		0.48		com.ibm.websphere.samples.plantsbywebsphereejb.CatalogBean.getInv(java.lang.String)
1	0.0071		0.49		com.ibm.websphere.samples.plantsbywebsphereejb.CatalogBean.getItemImageBytes(java.lang.String)
0	0.0		1.22		com.ibm.websphere.samples.plantsbywebsphereejb.Inventory.writeObject(java.io.ObjectOutputStream)
0	0.0		0.17		com.ibm.websphere.samplesgallery.gallerymenu.SamplesPropFile.loadFile(java.lang.String)
0	0.0		0.64		com.ibm.websphere.samples.plantsbywebsphereejbCatalog_Stub.getItemImageBytes(java.lang.String)
0	0.0		0.021		com.ibm.websphere.samplesgallery.gallerymenu.SamplesPropFile.getCategoryPathForSampleEar(org.w3c.dom.Node)
0	0.0		0.33		com.ibm.websphere.samples.plantsbywebsphereejbCatalog_Stub.getItemInventory(java.lang.String)
0	0.0		0.81		com.ibm.websphere.samples.plantsbywebspherewar.ImageServlet.doGet(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServlet
0	0.0		0.12		com.ibm.websphere.samples.plantsbywebsphereejb.CatalogBean.getItemInventory(java.lang.String)
0	0.0		0.0071		com.ibm.websphere.samples.plantsbywebsphereejb.OrderItem.pcNewObjectIdInstance()
0	0.0		0.81		com.ibm.websphere.samples.plantsbywebspherewar.ImageServlet.performTask(javax.servlet.http.HttpServletRequest, javax.servlet.http.Http
0	0.0		5.15	1	com.ibm.websphere.samples.plantsbywebsphereejbCatalog_Stub.getItemsByCategory(int)
0	0.0		3.21	1	com.ibm.websphere.samples.plantsbywebsphereejb.CatalogBean.getItemsByCategory(int)
0	0.0		0.62		com.ibm.websphere.samples.plantsbywebsphereejb.EJSRemote05LCatalog_59598632.getItemImageBytes(java.lang.String)
0	0.0		0.028		com.ibm.websphere.samples.plantsbywebsphereejb.OrderItem. <clinit>()</clinit>
0	0.0		0.071		com.ibm.websphere.samples.plantsbywebsphereeib.Order. <clinit>()</clinit>



Optional Steps:

Double click the desktop shortcut to ShoppingServlet.java to inspect the programming error.



_____ Click "Edit->Find" and search for "deliberateSlowMethod". Click "Find Next" to find the second occurrence of the search string.

The "deliberateSlowMethod" is invoked from the servlet's "doGet" processing every time the user clicks on the tulips. The "deliberateSlowMethod" executes a tight loop which does not end until a 3.5 wait time has passed. You have found the first deliberate mistake.

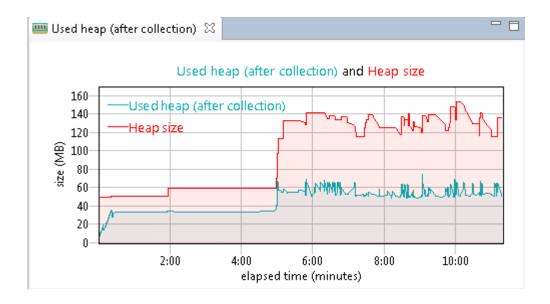
```
System.out.println("==> STARTING SLOW METHOD");
long timestamp = System.currentTimeMillis();
long target = timestamp + 3500;
System.out.println("timestamp="+timestamp);
System.out.println("resume at="+target);
while(timestamp < target) {
    timestamp = System.currentTimeMillis();
}
System.out.println("==> ENDING SLOW METHOD");
```

_____ Return to the Health Center window and click the "Garbage Collection" link to monitor the performance of garbage collection and memory usage

🚼 Status 🔀	
G <u>Classes</u>	(i)
Environment	8
Garbage Collection	0
☑ <u>I/O</u>	(i)
🔒 Locking	S
INATIVE Memory	8
🔥 Profiling	۵



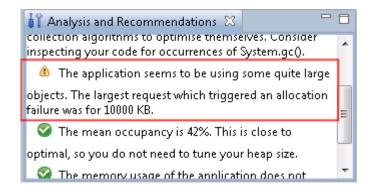
Observe the current JVM heap size and used heap size after collection. After starting the load generator, you will notice the heap size and heap usage has increased but by now should have leveled out. There is currently no evidence of a memory leak.

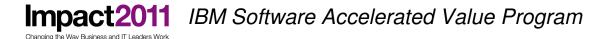


Note:

Garbage Collection (GC) affects the entire application and tuning GC correctly can potentially deliver significant performance gains. Health Center identifies where garbage collection is causing performance problems and suggests more appropriate command line options.

Observe the Analysis and Recommendations window. It warns of large object allocations which of course are likely to trigger frequent garbage collections and may indicate the application code can be optimized.





_ Click the "Object Allocations" tab to investigate this further.

	🖽 Summary 📟 Object Allocations 🕱 📃 🗖								
Enable collection of object allocation events Maximum stack depth 5									
	Low threshold (b)	/tes)		High threshold (bytes)	Undo Apply				
	Object size	Time of allocation	Request site						

Health Center allows you to view the size, time and code location of an object allocation request that meets specific threshold criteria.

Enable the "Enable collection of object allocation events" checkbox.

Set a threshold to focus on the biggest objects. The threshold values can be entered in bytes, kilobytes or megabytes. Enter **2m** for the low threshold and **10m** for the high threshold and click "Apply". Wait a few moments until large object allocation data is parsed by the Health Center client – **this could take up to 1 minute.**

🔲 Summary 📖	Object Allocations 🛛					
✓ Enable collect	tion of object allocation	events		N	1aximum stack depth 5	* *
Low threshold (b	ytes) 2m		High threshold (bytes)	10m	Undo	Apply
Object size	Time of allocation	Request site				

Observe the large object allocations meeting the defined threshold. They are associated with creating a very large String.

_ Click the rows in the table to see the stack trace leading to this large String allocation.

Once again you will see the ShoppingServlet class seems to be responsible, specifically a method named "deliberateLargeObjectAllocation". You have identified another deliberate error in the plants sample.

IBM Software Accelerated Value Program

w threshold (b	ytes) 2m	High threshold (bytes) 10m	Undo App
bject size 🔻	Time of allocation	Request site	
0000 KB	47:43 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
0000 KB	47:58 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
0000 KB	48:08 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
0000 KB	48:25 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
0000 KB	48:35 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
0000 KB	48:51 minutes	java.lang.StringBuilder.ensureCapacityImpl (StringBuilder.java:339) (Compiled Code)	
000 KB	47:43 minutes	java.lang.String. <init>(String.java:298) (Compiled Code)</init>	
000 KB	47:58 minutes	java.lang.String. <init> (String.java:298) (Compiled Code)</init>	
va.lang.StringBi va.lang.StringBi m.ibm.websphe	uilder.append (StringBuild uilder.append (StringBuild re.samples.plantsbyweb	ol (StringBuilder.java:339) (Compiled Code) Ier.java:205) (Compiled Code) Ier.java:180) (Compiled Code) spherewar.ShoppingServlet.deliberateLargeObjectAllocation (ShoppingServlet.java:618) spherewar.ShoppingServlet.performTask (ShoppingServlet.java:211) (Compiled Code)	

Optional Steps:

_ Double click the desktop shortcut for ShoppingServlet.java to inspect the programming error.



_____ Click "Edit->Find" and search for "deliberateLargeObjectAllocation". Click "Find Next" to find the second occurrence of the search string.

The "deliberateLargeObjectAllocation" is invoked from the servlet's "doGet" processing every time the user clicks on the grapes. The "deliberateLargeObjectAllocation" creates a large Array and fills it with a String of characters.

The variables used are local to the method so once the request has finished; the large objects are eligible for garbage collection. Therefore this is not a memory leak, but the creation of this large object makes unnecessary work for the JVM's garbage collector.

```
System.out.println("==> STARTING LARGE OBJECT ALLOCATION");
// Handle to a large object. Not a memory leak, just a LOA that will be GC'd
HashSet largeObject = null;
largeObject = new HashSet();
long timestamp = System.currentTimeMillis();
byte[] array = new byte[2560[000];
Arrays.fill(array, (byte) 66);
largeObject.add(new String(array) + (timestamp));
System.out.println("==> ENDING LARGE OBJECT ALLOCATION");
```

Return to the Heath Center window and click the "Garbage Collection" summary tab. This shows that System (forced) garbage collection is being called by some application code running in the JVM. There may also be a warning in the Analysis and Recommendations window, depending on how many times System (forced) garbage collection has been called.

🛅 Summary 🔀 📟 Object Allocations							
^							
Allocation failure count	1100						
GC Mode	Default (optthruput)						
Largest memory request	10000 KB						
Mean garbage collection pause	127 ms						
Mean heap unusable due to fragmentation	18.1 MB						
Mean interval between collections	1571 ms						
Proportion of time spent unpaused	91.9%						
Rate of garbage collection	1263 MB/minute						
System (forced) garbage collection count	539						
Time spent in garbage collection pauses	8.12%						

Note:

Impact

iging the Way Business and IT Leaders Wor

The Java code "System.gc()" forces a full garbage collection cycle. This is generally not recommended as the garbage collector should manage its own schedule of garbage collection, and does not always need to execute the compaction phase of GC which is the most CPU intensive. An application calling System.gc() will always trigger the most expensive compaction phase. Health Center can be used to track down the source of the System.gc() events.

The easiest way to determine what is calling System.gc() is using **JVM trace** – a facility that is provided in all IBM supplied JVMs that has a minimal affect on performance. Some types of JVM trace can be conveniently configured via Health Center. However in this case we need to use "**Method Trace**" which must be configured as a JVM command line option. To save time, the following Java method trace has already been configured as a "Generic JVM Argument" via the WebSphere administration console.

-Xtrace:print=mt,methods={java/lang/System.gc},trigger=method{java/lang/System.gc,jstacktrace}

This prints a **stack trace** when the **System.gc()** method is executed. It would also be possible to trigger other diagnostic information such as a Java dump or Java core (this will be explained in more detail in part 4 of this lab).

_ Double click the WinTail shortcut on the desktop.



Impact₂₀

17

v Business and IT Leaders Work

___ A "File Open" dialog box will appear. Navigate to:

"C:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1\native_stderr.log"

The file may be scrolling quite quickly, but every so often you will see a stack trace showing what is calling System.gc(). Alternatively use the scroll bar to find a previous stack trace.

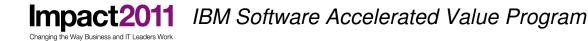
🕑 native_stderr (4.0 MB) - WinTail	×
<pre>ctemured freebytes="50411632" totalbytes="107329536" percent="46" ></pre>	^
<pre>2102 2102:51.175°0X76649e00 mt.3 > jara/lang/System.gc()Y Bytecode static method 2102:51.175°0X76649e00 g)trc_mux.1 - jstatic method arguments: () 2102:51.175°0X76649e00 j9trc_mux.1 - [] jom.lmm.websphere.smaples.jlantebywebspherewar.ShoppingServlet.deliberateSystemSC (ShoppingServlet.java:596) 2102:51.175°0X76649e00 j9trc_mux.1 - [] com.lmm.websphere.smaples.jlantebywebspherewar.ShoppingServlet.performTask (ShoppingServlet.java:596) 2102:51.175°0X76649e00 j9trc_mux.1 - [] com.lmm.websphere.smaples.jlantebywebspherewar.ShoppingServlet.performTask (ShoppingServlet.java:201) 2102:51.175°0X76649e00 j9trc_mux.1 - [] com.lmm.websphere.smaples.jlantebywebspherewar.ShoppingServlet.dece (ShoppingServlet.java:211)(Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] jom.lm.websphere.smaples.jlantebywebspherewar.ShoppingServlet.dece (ShoppingServlet.java:111) (Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] jom.websphere.smaples.jlantebywebspherewar.ShoppingServlet.java:111)(Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] java.servlet.http:/ttp:servlet.servlet.java:120](Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] java.servlet.http:/ttp:servlet.servlet.java:110)(Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] java.servlet.http:servlet.servlet.java:110](Compiled Code) 212:02:51.175°0X76649e00 j9trc_mux.1 - [] java.servlet.http:servlet.servlet.java:110](Compiled Code) 212:02:51.02007/50X76649900199ttes="10023530" thetadem="1" lasthreadtid="0x76E2000" /> </pre> <pre> </pre> <pre> <pre> </pre> <pre> <pre> </pre> <pre> </pre></pre></pre>	
<pre><refs dynamicsoftreferencethreshold="19" maxsoftreferencethreshold="32" phantom="215" soft="1377" weak="11285"></refs> </pre>	
12:02:51.378 0x7c6d9e00 mt.9 < java/lang/System.gc()V Bytecode static method	H 4

Once again the source of the problem is ShoppingServlet which calls a method "**deliberateSystemGC**". You have found another deliberate mistake in the plants sample.

Optional Steps:

_ Double click the desktop shortcut for ShoppingServlet.java to inspect the programming error.





_____ Click "Edit->Find" and search for "deliberateSystemGC". Click "Find Next" to find the second occurrence of the search string.

This "deliberateSystemGC" method is invoked from the servlet's "doGet" processing every time the user clicks on the gloves. The "deliberateSystemGC" method calls System.gc().

```
System.out.println("==> STARTING SYSTEM.GC");
System.gc();
System.out.println("==> ENDING SYSTEM.GC");
```

You will manually trigger the final deliberate error in the plants sample as it will cause a memory leak that you will diagnose using heap dumps in the remaining parts of this lab.

Part 4: Trigger a Memory Leak and Generate Java Dumps (15 minutes)

Note:

Impact²⁰

ng the Way Business and IT Leaders Wor

To save time, WebSphere Application Server has already been configured to generate both an IBM heap dump and a system dump on an out of memory exception. The following dump option was configured as a "Generic JVM Argument" using the WebSphere administration console:

-Xdump:heap:none

-Xdump:java+heap+system:events=user+throw,filter=java/lang/OutOfMem*,range=1..1

This overrides the default dump settings and specifies that exactly one heap dump (IBM PHD format) and one system core should be generated on an out of memory exception, or user signal to the process.

Java dumps (Javacore.txt files) are also configured. These are human readable text files containing summary information about the JVM, its memory and the running threads.

Verify the Jmeter application is still running as some of the remaining parts of the lab require there to be active sessions when the heap dump is triggered (in the next few steps).

📼 Impact 2011 Plants.jmx (C:\labfiles\Jmeter script\Impact 2011 Plants.jmx) - Apache JMeter (2.4 r961953)	
Eile Edit Run Options Help	
	2/2 🗖
Image: Scenario 1 Image: Scenario 1	
P / Mv16//mh-submenu-marker-right I Test Plan	

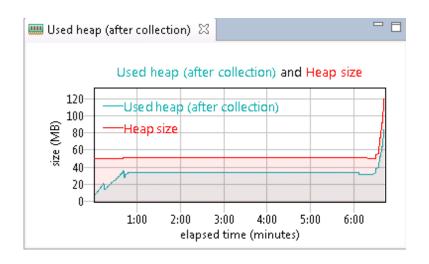
Use the browser to click the Wheelbarrow product on the Accessories page of the Plants by WebSphere sample. Return to the Accessories page and click the Wheelbarrow again.

Changing the Way Business and IT Leaders Work

Impact₂

CPlants by We	bSphere - Windows Int	ernet Explorer				
<u> </u>	http://localhost:9080/Pla	ntsByWebSphere/			Live Search	₽ •
File Edit View		elp				
🚖 🏟 🏉 РА	ants by WebSphere				🟠 • 🗟 • 🖶 • 🗟	Page 👻 🎯 Tools 👻 🦄
PLANTS BY	WEBSPHERE					
Flowers	Fruits & Vegetables	Trees Accessories			HOME SHOPPING CA	RT : LOGIN : HELP
Home						-
Accesso	ories			Page 1 of 1		
Bulb Digge	er <u>Birdfeeder</u>	Birdhouse	Finch Food	Grass Rake		
Leaf Rake	s Shovel	Gloves	Hand Rake	Large Pot		
Wheelbarro						
Done					Second Second Second	€ 100% ·

Return to the Health Center window and ensure it has refreshed its data a couple of times (there is a 10 second pause between each refresh). Take a look at the Garbage Collection statistics, you should see notice the memory usage has increased. It seems there is a memory leak.



Note:

Impact₂

aing the Way Business and IT Leaders Work

Health Center can be used to trigger a heap dump which is useful for analyzing memory leaks. However, to make the problem as easy to diagnose as possible, it is often beneficial to let the memory leak grow as large as possible.

_____ In the browser, continue clicking the Wheelbarrow until such time that the application no longer responds (**approximately 5-10 times**). This suggests the JVM has finally run out of memory and crashed. The JVM should begin to create some dumps.

Verify that some heap dumps have been generated in the JVM's working directory, "C:\IBM\WebSphere\AppServer\profiles\AppSrv01".

Use F5 to refresh the directory until the size of the files has stopped growing. **This may take up to 5 minutes so please be patient.** The system core (.dmp extension) will grow to around **400Mb to 500Mb** in size. Do not proceed until both the system core (.dmp) and IBM heap dump (.phd) files have been completely written to disk.

You will notice the system core (.dmp extension) is much larger than the IBM heap dump (.phd extension).



🚞 AppSrv01 - 🗆 × File Edit View Favorites Tools Help 🤇 Back 👻 a), 國國 X 9 Search Folders 💌 🔁 Go Address C:\IBM\WebSphere\AppServer\profiles\AppSrv01 Size Type Name Date Modified bin 2/16/2011 3:11 PM File Folder 🚞 config File Folder 2/16/2011 3:11 PM Configuration File Folder 2/17/2011 11:26 AM Consolepreferences File Folder 2/17/2011 11:30 AM netc 📄 File Folder 2/16/2011 3:14 PM 🚞 firststeps File Folder 2/17/2011 11:25 AM 🚞 installableApps File Folder 2/16/2011 3:11 PM 🚞 installedApps File Folder 2/16/2011 3:12 PM installedConnectors File Folder 2/16/2011 3:11 PM 🚞 installedFilters File Folder 2/17/2011 8:49 AM 🚞 logs File Folder 2/25/2011 6:10 AM properties File Folder 2/16/2011 3:14 PM 🚞 samples File Folder 2/16/2011 3:14 PM 🚞 temp File Folder 2/25/2011 6:03 AM File Folder 🚞 tranlog 2/17/2011 8:50 AM wstemp File Folder 2/17/2011 11:30 AM 466,239 KB DMP File d core.20110225.062351.2360.0073.dmp 2/25/2011 6:26 AM d heapdump.20110225.062351.2360.0074.phd 12,217 KB PHD File 2/25/2011 6:27 AM javacore.20110225.062332.2360.0072 1,922 KB Text Document 2/25/2011 6:23 AM javacore.20110225.062351.2360.0075 1,935 KB Text Document 2/25/2011 6:27 AM javacore.20110225.062711.2360.0078 1,912 KB Text Document 2/25/2011 6:27 AM iavacore.20110225.062711.2360.0079 1.912 KB Text Document 2/25/2011 6:27 AM javacore.20110225.062711.2360.0081 1,912 KB Text Document 2/25/2011 6:27 AM 🗐 orbtrc.25022011.0559.52 10 KB Text Document 2/25/2011 5:59 AM 🔟 Snap.20110225.062332.2360.0071.trc 181 KB TRC File 2/25/2011 6:23 AM 🖬 Snap.20110225.062711.2360.0076.trc 157 KB TRC File 2/25/2011 6:27 AM 🔄 Snap.20110225.062711.2360.0077.trc 29 KB TRC File 2/25/2011 6:27 AM 🖬 Snap.20110225.062711.2360.0080.trc 37 KB TRC File 2/25/2011 6:27 AM

____ Right click on the Windows taskbar and select "Task Manager"

_____ Select the "Process Tab". Order the processes by name by clicking the "Image Name" column heading.

e Opti	ons vi	ew Hel	• •					
Applications Processes Performance Networking								
Imag	e Name		User Name		Mem Usage			
alg.e			LOCAL SERVICE	00	152 K			
	p.exe		Administrator	00	352 K			
CCEV	Mgr.ex	е	SYSTEM	00	84 K			
ccSel	:Mgr.ex	е	SYSTEM	00	112 K			
cmd.	exe		Administrator	00	80 K			
cmd.	exe		Administrator	00	72 K			
csrss	.exe		SYSTEM	00	980 K			
ctfmo	on.exe		Administrator	00	324 K			
DefW	/atch.e>	œ	SYSTEM	00	88 K			
eclips	e.exe		Administrator	16	55,344 K			
explo	orer.exe		Administrator	02	6,364 K			
iexpl	ore.exe		Administrator	03	3,976 K			
java.	exe		Administrator	00	1,712 K			
java.	exe		Administrator	56	457,784 K			
javav	v.exe		Administrator	00	692 K			
jqs.e	xe		SYSTEM	00	1,380 K			
Isass	.exe		SYSTEM	00	944 K			
msms	gs.exe		Administrator	00	308 K			
note	oad.exe		Administrator	00	192 K	-		
Show processes from all users End Process								

Locate one or more Java processes using a large amount of memory (e.g. 200Mb or more). Click the "End Process" button for each one.

_____ Click the "Yes" button.

Impact2011

Task Manager Warning 🛛 🔀								
1	WARNING: Terminating a process can cause undesired results including loss of data and system instability. The process will not be given the chance to save its state or data before it is terminated. Are you sure you want to terminate the process?							
	Yes No							

_ Close the task manager window.

If for any reason the dump files are not present or complete, you can use some pre-prepared heap dumps in directory "C:\Users\Administrator\Documents\Lab Files\Dumps".

Note:

Impact

ng the Way Business and IT Leaders Work

When a system dump (.dmp extension) is produced by the JVM, it is generated in a machine specific format and the internal structure of the information is specific to the VM that created it. To make the dump readable on other systems (e.g. heap dump analysis tools) it is necessary to run the **jextract** program on the dump.

_____ Click "Start->Run" and type "cmd" in the search box, open a cmd window.

_____ In the cmd window, type "cd C:\IBM\WebSphere\AppServer\java\jre\bin".

_____ Type command "jextract C:\IBM\WebSphere\AppServer\profiles\AppSrv01\"

(do not press enter yet)

Use windows explorer to copy the filename of the system dump (click "rename" to highlight the filename, then "copy").

i uniog wstemp			File Folde
acore.20110224.094239.3072.194		468,115 KB	DMP File
튇 javacore.20110224.094230.3072	Undo	1,903 KB	Text Doci
📃 javacore.20110224.094230.3072	Cut	1,902 KB	Text Doci
📃 orbtrc.24022011.0728.33	Сору	12 KB	Text Doci
🔤 Snap.20110224.094230.3072.19	Paste	173 KB	TRC File
🔤 Snap.20110224.094230.3072.19	Delete	21 KB	TRC File
🔤 Snap.20110224.094230.3072.19		— 21 KB	TRC File
🔤 Snap.20110224.094230.3072.19	Select All	21 KB	TRC File
🔋 javacore.20110224.094230.3072	.1942	1,906 KB	Text Doci

_____ Right click and paste filename into the DOS command window. Press enter to execute the **jextract** command.

Impact 2011 IBM Software Accelerated Value Program

C:\IBM\WebSphere\AppServer\java\jre\bin>jextra iles\AppSrv01\core.20110224.094239.3072.1944.d	mp_	rw ens pne re	спррзег оег (рг

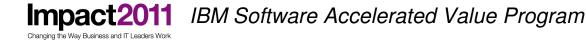
_____ The jextract process will take a few minutes to complete. However, you won't need this file for a while so continue with the next steps.

Note:

When the jextract process has completed, a new compressed version of the .dmp file will exist in the JVM working directory.

wstemp			File Folder	
🔜 🖬 core.20110224.094239.3072.1	.944.dmp 46	58,115 KB	DMP File	
ere.20110224.094239.3072.1	.944.dmp 10	01,884 KB	Compressed (zippe	
		0.011.00	oup et	

Now that you have the heap dumps to investigate the memory leak, close the open windows (except the DOS window running extract).



Part 5: *Optional* - Analyzing a Java Core for Evidence of a Memory Issue (5 minutes)

Note:

As the JVM has created Java cores, a system dump and a heap dump file, the first assumption should be that an out of memory issue has occurred. Java cores can be produced by a number of triggers, but the creation of the combination of a Java core and heap dump is the default JVM action on an out of memory error. Therefore, reviewing the Java core for its trigger is the logical first step. **This part of the lab is optional.**

Use a text editor to open one of the Java core files (named javacore < datetimestamp>.txt) from the JVM's working directory "C:\IBM\WebSphere\AppServer\profiles\AppSrv01"

Impact 2011 IBM Software Accelerated Value Program

_____ Determine the trigger that caused the Java core to be created. In the "**SIGINFO**" section you will see that a "**java/lang/OutOfMemoryError**" was indeed the signal that triggered the Java core. This confirms an out of memory condition and not a non-memory related crash.

📕 javacore.2011022	24.094230.3072.1940 - Notepad
File Edit Format Vi	iew Help
NULL OSECTION	TITLE subcomponent dump routine
NULL 1TISIGINFO 1TIDATETIME	Dump Event "systhrow" (00040000) Detail "java/lang/outOfMemoryError" received Date: <u>2011/02/24 at 09:42:31</u>
1TIFILENAME 1TIREQFLAGS 1TIPREPSTATE NULL	Javacore filename: C:\IBM\WebSphere\AppServer\profiles\AppSrv01\javacore.20110224.094230.3072.194 Request Flags: 0x81 (exclusive+preempt) Prep State: 0x4 (exclusive_vm_access)
0SECTION NULL	GPINFO subcomponent dump routine
2×HOSLEVEL 2×HCPUS 3×HCPUARCH	OS Level : windows XP 5.1 build 2600 Service Pack 3 Processors - Architecture : <u>x</u> 86
3×HNUMCPUS 3×HNUMASUP NULL	How Many : 2 NUMA is either not supported or has been disabled by user
1×HERROR2 NULL NULL	Register dump section only produced for SIGSEGV, SIGILL or SIGFPE.
0SECTION	ENVINFO subcomponent dump routine
1CIJAVAVERSION 1CIVMVERSION	JRE 1.6.0 IBM J9 2.4 Windows XP x86-32 build jvmwi3260sr8ifx-20100923_65174 VM build 20100923_065174 JIT enabled, AOT enabled - r9_20100401_15339ifx6 GC - 20100308_AA Running as a standalone JVM
ebug=off -Xtra 1CIJAVAHOMEDIR	C:\IBM\webSphere\AppServer/java/bin/java -Declipse.security -Dwas.status.socket=1087 -Dosgi.install. =allow -verbose:gc -xms50m -xmx256m -Dws.ext.dirs=C:\IBM\webSphere\AppServer/java/lib;C:\IBM\webSpher ce:print=mt,methods={java/lang/System.gc},trigger=method{java/lang/System.gc,jstacktrace} -Xdump:heag Java Home Dir: C:\IBM\webSphere\AppServer\java\ire\bin
1CISYSCP	sys Classpath: C:\IBM\WebSphere\AppServer/java/jre/lib/ext/ibmorb.jar;C:\IBM\WebSphere\AppServer/ re\AppServer\java\jre\lib\ibmjssefw.jar;C:\IBM\WebSphere\AppServer\java\jre\lib\ibmsaslfw.jar;C:\IBM UserArqs:
2CIUSERARG	✓ -xjcl:jclscar_24

Impact2011 IBM Software Accelerated Value Program

_____ The Java core also contains important Java heap configuration and status information. Observe the maximum heap size (-Xmx256m) in the "**CIUSERARG**" section.

ng the Way Business and IT Leaders Work

📕 javacore.20110	1224.094230.3072.1940 - Notepad
File Edit Format	View Help
	were\AppServer\java\jre\lib\ibmjssefw.jar;C:\IBM\webSphere\AppServer\java\jre\lib\ibmsaslfw.jar;C:\IBM\a
1CIUSERARGS	UserArgs:
2CIUSERARG	-xjcl:jclscar_24
2CIUSERARG	-Dcom.ibm.oti.vm.bootstrap.library.path=C:\IBM\WebSphere\AppServer\java\jre\bin
2CIUSERARG	-Dsun.boot.library.path=C:\IBM\websphere\Appserver\java\jre\bin
2CIUSERARG	-Djava.library.path=C:\IBM\webSphere\AppServer\java\jre\bin;.;C:\IBM\webSphere\AppServer\t
2CIUSERARG	-Djava.home=C:\IBM\WebSphere\AppServer\java\jre
2CIUSERARG	-D]ava.ext.dirs=C:\IBM\webSphere\AppServer\java\jre\lib\ext
2CIUSERARG	-Duser.dir=C:\IBM\webSphere\AppServer\profiles\AppSrv01
2CIUSERARG	_j2se_j9=71168 0x7FBE7290
2CIUSERARG	-Xdump
2CIUSERARG	-pconsole.encoding=Cp850
2CIUSERARG 2CIUSERARG	-Djava.class.path=C:\IBM\webSphere\AppServer\profiles\AppSrv01\properties;C:\IBM\webSpher@ -Declipse.security
2CIUSERARG	-bechipse.security
2CIUSERARG	-Dwas.status.sucket=100/ -Dosgi.install.area=C:\lBM\webSphere\AppServer
2CIUSERARG	-Dosg1.configuration.area=C:\IBM\websphere\AppServer\profiles\AppSrv01/configuration
2CIUSERARG	-bosgi.framework.extensions=com.ibm.cds,com.ibm.we.clipse.adaptors
2CIUSERARG	->osgrinamework.extensions=com.iom.com,com.iom.ws.ecripse.adaptors
2CIUSERARG	-xsrarx50M
2CIUSERARG	-DSUM.reflect.inflationThreshold=250
2CIUSERARG	
2CIUSERARG	-Xbootclasspath/p:C:\IBM\WebSphere\AppServer/java/jre/lib/ext/ibmorb.jar;C:\IBM\WebSphere\ -Djava.class.path=C:\IBM\WebSphere\AppServer\profiles\AppSrv01/properties;C:\IBM\WebSphere
2CIUSERARG	-Dibm.websphere.internalClassAccessMode=allow
2CTUSERARG	-verbosetac
2CIUSERARG	-xms 50m
2CIUSERARG	-×mx256m
ZCIUSERARG	-Dws.ext.dirs=C:\IBM\webSphere\AppServer/java/lib;C:\IBM\webSphere\AppServer\profiles\AppS
2CIUSERARG	-Dderby.system.home=C:\IBM\webSphere\AppServer/derby
2CIUSERARG	-Dcom.ibm.itp.location=C:\IBM\webSphere\AppServer/bin
2CIUSERARG	-Djava.util.logging.configureByServer=true
2CIUSERARG	-Düser.install.root=C:\IBM\webSphere\AppServer\profiles\AppSrv01
2CIUSERARG	-Djavax.management.builder.initial=com.ibm.ws.management.PlatformMBeanServerBuilder
2CIUSERARG	-Dwas.install.root=C:\IBM\webSphere\AppServer
2CIUSERARG	-Dpython.cachedir=C:\IBM\webSphere\AppServer\profiles\AppSrv01/temp/cachedir
2CIUSERARG	-Djava.util.logging.manager=com.ibm.ws.bootstrap.wsLogManager
•	
<u>, </u>	

Determine the free heap space and the total heap allocation from the "**MEMINFO**" section. In this case, there are 0 bytes free in the available heap space. Note that the values are in Hex, and 10000000 is 256 in decimal (the configured maximum heap size). If you wish to calculate the free heap for your Java core file, you can use the Windows calculator in scientific mode to convert from Hex to decimal.

Changing the Way Business and IT Leaders Work

Impact2

	224.094230.3072.1940 - Notepad
File Edit Format	
2CIENVVAR	USERPROFILE=C:\Documents and Settings\Administrator
2CIENVVAR	WAS_CELL=IMPACT2011Node01cell
2CIENVVAR	<pre>WAS_CLASSPATH=C:\IBM\webSphere\AppServer\profiles\AppSrv01\properties;C:\IBM\webSphere\AppServer\prof WAS_EXT_DIRS=C:\IBM\webSphere\AppServer\java\lib;C:\IBM\webSphere\AppServer\classes;C:\IBM\webSphere\ </pre>
2CIENVVAR 2CIENVVAR	WAS_HOME=C:\IBM\webSphere\AppServer WAS_HOME=C:\IBM\webSphere\AppServer
2CIENVVAR 2CIENVVAR	WAS_LOWE=C.\Low(websphere explose)ver WAS_LOGGING=-Djava.util.logging.manager=com.ibm.ws.bootstrap.wsLogManager -Djava.util.logging.confic
2CIENVVAR	WAS_NODE=IMPACT2011node01
2CIENVVAR	WAS_PATH=C:\IBM\webSphere\AppServer\bin;C:\IBM\webSphere\AppServer\iava\bin;C:\IBM\webSphere\AppServ
2CIENVVAR	WAS_USER_SCRIPT=C:\IBM\webSphere\AppServer\profiles\AppSrv01\bin\setupCmdLine.bat
2CIENVVAR	WAS_USER_SCRIPT_FILE_NOT_EXISTS=false
2CIENVVAR	windir=C:\windows
NULL	
1CIJVMMI	JVM Monitoring Interface (JVMMI)
NULL	
2CIJVMMIOFF	[not available]
NULL	
NULL	
0SECTION	MEMINFO subcomponent dump routine
NULL	
1STHEAPFREE 1STHEAPALLOC	Bytes of Heap Space Free: 0 Bytes of Heap Space Allocated: 10000000
NULL	Bytes of heap space Affocated. 1000000
1STSEGTYPE	Internal Memory
NULL	segment start alloc end type bytes
1STSEGMENT	176CD2D4 18744FF8 18744FF8 18754FF8 01000040 10000
1STSEGMENT	185A939C 18788068 18788068 187C8068 01000040 10000
1STSEGMENT	17DCF2E4 187E8018 187E8018 187F8018 01000040 10000
1STSEGMENT	17DCF1C4 18677018 18677018 18687018 01000040 10000
1STSEGMENT	1468DA64 1B7A8040 1B7A8040 1B7B8040 01000040 10000
1STSEGMENT	17DCF404 1868E018 1868E018 186CE018 01000040 10000
1STSEGMENT	18354834 18785028 18785028 18795028 01000040 10000
1STSEGMENT	18F0D274 1BAC93A0 1BAC93A0 1BAD93A0 01000040 10000
1STSEGMENT	1835A704 18808038 18808038 18818038 01000040 100000
1STSEGMENT	1848C874 18666FA8 18666FA8 18676FA8 01000040 10000 1468D8E4 18655728 18655728 18655728 01000040 10000
1STSEGMENT 1STSEGMENT	14080E44 18055728 18055728 18055728 10005728 01000040 10000 181A0394 188E780 188E780 188E780 01000040 10000
•	

Note:

At this point, it appears that the JVM is experiencing a severe shortage of heap space. If you needed more detailed information you could study the verbose GC to help identify if the cause of the out of memory condition is a memory footprint problem, or a memory leak.

A footprint problem would manifest itself as an increase in used heap space associated with an increase in workload for the JVM. A memory leak would manifest itself as an increase in used heap space either gradually over time, or rapidly at different times when associated with a particular application event.

IBM provides to tool to assist in this analysis - the **IBM Monitoring and Diagnostic Tools for Java™** - **Garbage Collection and Memory Visualizer (GCMV)** which is an add-on to the IBM Support Assistant (ISA). **Appendix A** contains an optional lab part where you can examine in more detail the garbage collection activity that led to the out of memory condition, using GCMV. It is suggested to complete Appendix A if you have time after the remaining parts of this lab.

In the next part of the lab, you will analyze the Java heap dump files and diagnose the code that caused the memory leak.

Part 6: Using the ISA and the Memory Analyzer to Analyze a Heapdump (20 minutes)

Note:

Impact₂

ping the Way Business and IT Leaders Wor

Memory Analyzer is a powerful and flexible tool for analyzing Java heap memory using system or heap dumps of the Java process. The maximum heap size for the tool has been increased in this lab to ensure the tool can handle larger heap dumps.

The next two parts of lab will direct you to open the IBM heap dump and system dump you generated previously. Opening these heap dumps for the first time can take up to **5 minutes**. If you prefer, use the ready made heap dump in **C:\labfiles\Dumps** – these files have previously been opened by Memory Analyzer which creates "index files". Using these files will slightly reduce the amount of time required to complete the lab – it's up to you.

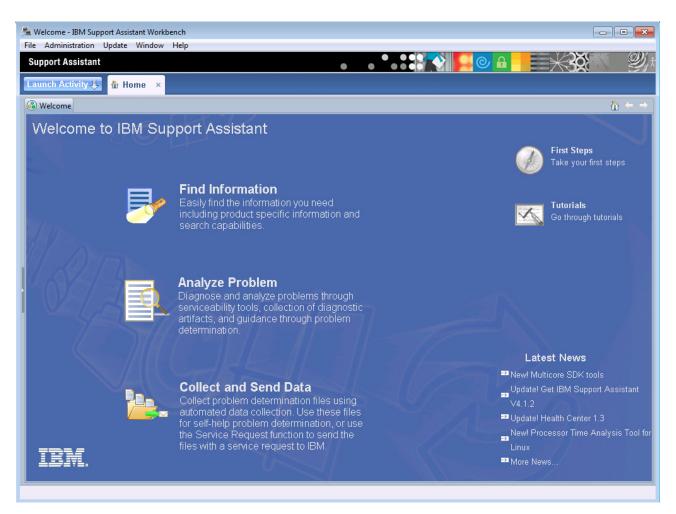
As every heap dump is different, you may see some slight variation from the screenshots in this lab document, e.g. exact number of bytes for the object size or number of objects in a data structure etc.

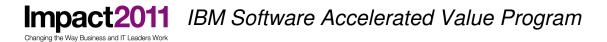
_ Double click the "IBM Support Assistant 4.1" shortcut on the desktop.



____ Click the blue "Launch Activity" button and select "Analyze Problem".

Changing the Way Business and IT Leaders Work





_____ Select "IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer", and then click the grey "Launch" button at the bottom of the screen.

🛸 Tools - IBM Support Assistant Workbench		
File Administration Update Window Help		
Support Assistant		
Launch Activity 🖡 🚹 Home 🛪 🗟 Analyze Problem 🗙		
👩 👔 Tools 📙 Collect Data 🥳 Guided Troubleshooter		
Case/Incident		
default Select		
Tools Catalog	Find New Tool Add-ons	Description
Tool Name	Version	IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer is a fast and feature-rich Java heap
IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory V	/is 2.4.0.20101007	analyzer that helps you find memory leaks and reduce
IBM Monitoring and Diagnostic Tools for Java™ - Health Center	1.3.0.20101104	memory consumption. The tool reads IBM Portable Heap Dump (.phd) files,
IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer [Tech Preview]	0.6.0.201101042253	system dumps generated by IBM VMs and processed by jextract, and Sun hprof binary heap dump files.
		Restrictions
		None A
		Associations
		Tool is not associated with any products
Launch Submit Feedback Help		

____ On the "Tool Input Parameters Values" box, click "Next". This will launch the tool immediately.

晕 Tool Input Parameter Values	×
	(* 1 * * * * *
□IBM Monitoring and Diagnostic Tools for Java™ - Memory Anal	yzer (Tech Preview)
Heap dump file	Browse
Next >	Cancel
T Wext >	

Click "File->Open heap dump" and **navigate** to the JVM's working directory "C:\IBM\WebSphere\AppServer\profiles\AppSrv01" (note the default directory is for the 'ready made' heap dumps, not the JVM's working directory).

_____ Select the IBM heap dump (extension .phd) and click the "Open" button. Please wait a few minutes for the heap dump to be processed.

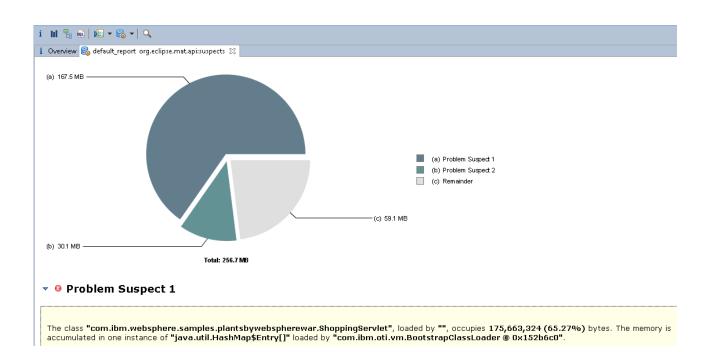
Open Snapshot				? ×
Look in:	C AppSrv01	•	G 🔌 📂 🖽 -	
	Name 🔺		Size	Туре 🔺
	C properties			File Folder
My Recent	🛅 samples			File Folder
Documents	i temp	Size	: 79.6 KB	
	🛅 tranlog		lers: version	
	🗀 wstemp	Files	s: client.policy, firsts	tepsport.props, ip
Desktop	🗖 core.20110224.094239.3072.1944.dmp		468,115 KB	DMP File
	Core.20110224.094239.3072.1944.dmp		101,884 KB	Compressec
	heapdump.20110224.094239.3072.1945.phd		12,244 KB	PHD File
Mu Desuments	javacore.20110224.094230.3072.1940		1,903 KB	Text Docum
My Documents	javacore.20110224.094230.3072.1941		1,902 KB	Text Docum
	javacore.20110224.094230.3072.1942		1,906 KB	Text Docum
	javacore.20110224.094230.3072.1943		1,906 KB	Text Docum
My Computer	javacore.20110224.094824.3072.1946		1,904 KB	Text Docum
my compater	🗐 orbtrc.24022011.0728.33		12 KB	Text Docum 🚽
- 🥡 - I	<u>د</u>			
My Network	File name: heapdump.20110224.094239.3	072.1	1945 💌	Open
Places	Files of type: All Known Formats		•	Cancel

_____ In the "Getting Started Wizard" box, click the "Finish" button to open the "Leak Suspects Report" and wait a few moments more.

Impact2011 Changing the Way Business and IT Leaders Work

🚰 Getting Started Wizard	
Getting Started	
Choose one of the common reports below. Press Escape to close this dialog.	
C Leak Suspects Report	
Automatically check the heap dump for leak suspects. Report what objects are kept alive and why they are not garbage collected.	
C Component Report	
Analyze a set of objects for suspected memory issues: duplicate strings, empty collections, finalizer, weak references, etc.	
C Re-open previously run reports	
Existing reports are stored in ZIP files next to the heap dump.	
Show this dialog when opening a heap dump.	
< Back Next > Finish	Cancel

This report provides basic heap statistics as well as a list of possible leaking objects. Your heap dump will indicate that ShoppingServlet is responsible for a large percentage of memory, and that a single HashMap is involved.



_ Click on the "Details" link to display the shortest paths to an accumulation point.

Note:

ng the Way Business and IT Leaders Work

An accumulation point is simply a reference that is suddenly responsible for keeping lots of heap space alive. In this case, the single HashMap at the top of the table has been identified as the single object that is responsible for the large accumulation of further objects.

The shortest path to this accumulation point shows what is responsible for keeping that accumulation object alive. In this case, a class loader, Thread and ShoppingServlet Class are referring to the HashMap.

Shortest Paths To the Accumulation Point

Class Name	Shallow Heap	Retained Heap
n java.util.HashMap\$Entry[131072] @ 0xa3218f8	524,304	175,662,656
🔤 🛅 java.util.HashMap @ 0x72455c8	48	175,662,720
🛄 🚹 java.util.HashSet @ 0x72455b8	16	175,662,736
느 🚡 class.com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet @ 0x7b79c28	76	175,663,324
— 🜔 <java local=""> <u>com.ibm.ws.util.ThreadPool\$Worker @ 0x2ca1318</u> Thread</java>	136	176,720
Class> com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet @ 0x72455f8 *	24	24
🔞 com.ibm.ws.classloader.CompoundClassLoader @ 0x7138118 >>	152	6,988
Σ Total: 3 entries		

____ Scroll down to view the accumulated objects.

This view shows the objects referred to by the HashMap accumulation point. You will see the HashMap contains entries that total approximately 175Mb or 65% of the total heap space. Only the first 20 entries from the HashMap are shown in this report.

Impact2011

IBM Software Accelerated Value Program

hanging the Way Business and IT Leaders Work

Accumulated Objects

Class Name	Shallow Heap	Retained Heap	Percentage
c) class com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet @ 0x7b79c28	76	175,663,324	65.27%
🛄 java.util.HashSet @ 0x72455b8	16	175,662,736	65.27%
🛄 java.util.HashMap @ 0x72455c8	48	175,662,720	65.27%
📖 👔 java.util.HashMap\$Entry[131072] @ 0xa3218f8	524,304	175,662,656	65.27%
🗋 java.util.HashMap\$Entry @ 0x6c90f78	32	211,888	0.08%
🗋 java.util.HashMap\$Entry @ 0x8de8070	32	211,888	0.08%
🗋 java.util.HashMap\$Entry @ 0x6f79bd0	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x718f4a8	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x75c46d0	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x75f3450	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x761e770	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x76f2870	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x786f5d8	32	209,304	0.08%
🗋 java.util.HashMap\$Entry @ 0x93ad788	32	209,304	0.089
🗋 java.util.HashMap\$Entry @ 0x60afbd0	32	206,720	0.089
🗋 java.util.HashMap\$Entry @ 0x68706e0	32	206,720	0.089
- Djava.util.HashMap\$Entry @ 0x6b967e0	32	206,720	0.089
🗋 java.util.HashMap\$Entry @ 0x6f7d430	32	206,720	0.089
🗋 java.util.HashMap\$Entry @ 0x6fba370	32	206,720	0.089
🗋 java.util.HashMap\$Entry @ 0x6fe61a8	32	206,720	0.089
🗋 java.util.HashMap\$Entry @ 0x7068188	32	206,720	0.089
Diava.util.HashMap\$Entry @ 0x725b4c8	32	206,720	0.08%
🗋 java.util.HashMap\$Entry @ 0x7478c30	32	206,720	0.089
🗽 Σ Total: 20 entries	640	4,165,408	0.01

Scroll down to view the "Accumulated Object by Class" table. This shows the total number of objects referred to by the HashMap accumulation point, in this case over 17,000 objects with a retained size of over 175Mb. Of course, the exact statistics in your heap dump will vary.

Accumulated Objects by Class

Label	Number Of Objects	Used Heap Size	Retained Heap Size
🔓 java.util.HashMap\$Entry	17,928	573,696	175,138,352

Note:

Shallow heap refers to the size of an individual object in isolation, and **retained heap** includes all the objects that are referenced (and kept alive) by that object.

____ Scroll to the "Accumulated Objects" table.

_____ To determine what is stored in each of these HashMap entries, left click the accumulation point,

i.e. the HashMapEntry with the array icon $\widehat{\mathbb{D}}$.

_____ Select "Java Basics->Open in Dominator Tree" to display the most significant references from the selected HashMap object.

Note:

As an alternative to the Dominator Tree, you could show **all** references with List objects->with outgoing references.

<u>iava.util.HashMap\$Entry @ 0x6f</u> <u>iava.util.HashMap\$Entry @ 0x71</u>	0	List objects Show objects by class Path To GC Roots Merge Shortest Paths to GC Roots IBM Extensions	* * * *		524,304 32 32 32 32	175,662,656 211,888 211,888 209,304 209,304		65.27% 0.08% 0.08% 0.08% 0.08%
		Java Basics Java Collections Leak Identification Immediate Dominators Show Retained Set Copy Search Queries	* * *	چ کے ¹⁰	Find Strin Group By Open In D	ed Retained Set gs Value Dominator Tree Histogram etails	•	0.08% 0.08% 0.08% 0.08% 0.08% 0.08% 0.08%

____ Click "Finish" on the "Open In Dominator" dialog.

Changing the Way Business and IT Leaders Work

樳 Java Basics / C)pen In Dominator Tree		_ 🗆 🗡
Open In Domina Open Dominator Tr	ator Tree ree for selected objects.		
Argument	Value		
objects	[context]		
-groupby	No Grouping (objects)		-
1			
?		Finish	Cancel
Open Dominator Tree	for selected objects.		
Arguments:			
-groupby	dominator tree should be sho objects in the resulting tree t		or by package.

Expand the top level HashMap object to show the thousands of entries that compose this HashMap.

Impact2011 Changing the Way Business and IT Leaders Work

i III 🖳 💀 | 🔎 🕶 😂 🕶 | 🔍 | 🎦 🕶 🖼 🕶 | 🦣

ss Name	Shallow Heap	Retained Heap	Percentage
♣ <regex></regex>	<numeric></numeric>	<numeric></numeric>	<numeric></numeric>
🔟 java.util.HashMap\$Entry[131072] @ 0xa3218f8	524,304	175,662,656	65.27%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6c90f78	32	211,888	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x8de8070	32	211,888	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6f79bd0	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x718f4a8	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x75c46d0	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x75f3450	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x761e770	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x76f2870	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x786f5d8	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x93ad788	32	209,304	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x60afbd0	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x68706e0	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6b967e0	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6f307a0	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6f7d430	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6fba370	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x6fe61a8	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x7068188	32	206,720	0.08%
java.util.HashMap\$Entry @ 0x725b4c8	32	206,720	0.08%
java.util.HashMap\$Entry @ 0x7478c30	32	206,720	0.08%
java.util.HashMap\$Entry @ 0x7565828	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x7739498	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x773a8c8	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x773bcf8	32	206,720	0.08%
⊳ 🗋 java.util.HashMap\$Entry @ 0x773c710	32	206,720	0.08%
∑₄ Total: 25 of 17,928 entries			

_____ Display some further entries by **right clicking** the "Total" and selecting "Next 25". The "Expand All" option is likely to take some time so **avoid** clicking that.

⊳ 🗋 java.util.HashMap\$Entry @ 0x773c710		32		206,720	0.08%	
∑₊ Total: 25 of 17,	9-20 	Columns	×			
📝 Notes 🛛 🤤 Navigati	+	Next 25				
	++	Custom expand Expand All				

Expand one of the HashMap entries.

You will notice that the HashMap entry refers to a ShoppingContainer class in the plants sample.

🖃 📄 java.util.HashMap\$Entry @ 0xc4ea5e8	
표 📄 java.util.HashMap\$Entry @ 0xc4ab388 🗧 🧧	
🗉 📄 com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet	hoppingContainer @
∑ Total: 2 entries	

Note:

Impact

ng the Way Business and IT Leaders Work

The class name "ShoppingServlet\$ShoppingContainer" means the "ShoppingContainer" class is an inner class of "ShoppingServlet".

This ShoppingContainer object is responsible for keeping approximately 2,552 bytes alive in the heap. The HashMapEntry actually keeps 209,304 bytes alive in total, including the ShoppingContainer. All the other HashMapEntry objects in the list follow an identical pattern. Their retained heap varies, but they always contain a ShoppingContainer.

👔 Overview 📊 Histogram 😜 default_report_org.eclipse.mat.api:suspects 🖳 show_dominator_tree [context] 🛛					
Class Name	Shallow Heap	Retained Heap	Percentage		
🔆 <regex></regex>	<numeric></numeric>	<numeric></numeric>	<numeric></numeric>		
a 🔟 java.util.HashMap\$Entry[131072] @ 0xa3218f8	524,304	175,662,656	65.27%		
b 🗋 java.util.HashMap\$Entry @ 0x6c90f78	32	211,888	0.08%		
b 🗋 java.util.HashMap\$Entry @ 0x8de8070	32	211,888	0.08%		
b] java.util.HashMap\$Entry @ 0x6f79bd0	32	209,304	0.08%		
🔺 🗋 java.util.HashMap\$Entry @ 0x718f4a8	32	209,304	0.08%		
java.util.HashMap\$Entry @ 0x75dfc58	32	206,720	0.08%		
com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet\$ShoppingContainer	32	2,552	0.00%		
∑ Total: 2 entries					

You have established that objects from the plants sample seem to be involved with the memory leak - a **ShoppingServlet** is referring to a large **HashMap** that contains many instances of **ShoppingContainer**. Let's look at the overall footprint of the entire plants sample.

_____ Click on the 🛄 icon to open a histogram.

Type "**com.ibm.websphere.samples.plantsbywebspherewar.***" in the "Regex" filter box and press the Enter key. The Regex filter box is the first line of the table, starting with the icon Regex>.

Changing the Way Business and IT Leaders Work

impact

i III 🖳 💀 🔎 ▼ 🚱 ▼ 🔍 🍃 ▼ 🖾 ▼ 🖾 ▼ 😤			
i Overview 😂 default_report org.eclipse.mat.api:suspects 🔢 Histogram 🔀			
Class Name	Objects	Shallow Heap	Retained Heap
Com.ibm.websphere.samples.plantsbywebspherewar.*	<numeric></numeric>	<numeric></numeric>	<numeric></numeric>
Com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet\$ShoppingContainer	67,779	2,168,928	>= 172,969,564
Com.ibm.websphere.samples.plantsbywebspherewar.ImageServlet	1	24	>= 100
Com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet	1	24	>= 175,663,348
G com.ibm.websphere.samples.plantsbywebspherewar.AdminServlet	0	0	>= 76
Com.ibm.websphere.samples.plantsbywebspherewar.AccountServlet	0	0	>= 76
Σ Total: 5 entries (19,926 filtered)	67,781	2,168,976	

In this heap dump there are 67,779 instances of ShoppingContainer with a shallow heap of 2Mb and a retained heap of 172Mb. The cause of this memory leak is becoming increasingly clear, and it seems only the ShoppingContainer and ShoppingServlet classes are involved.

Optional Steps:

_____ Double click the desktop shortcut for ShoppingServlet.java to inspect the programming error.



_____ Click "Edit->Find" and search for "ShoppingContainer" to locate the definition of the inner class.

____ Click "Edit->Find" and search for method "deliberateMemoryLeak", click "Find Again".

You will find method "deliberateMemoryLeak" adds 10000 instances of the ShoppingContainer inner class to a **static** HashMap every time the wheelbarrow image is clicked. The constructor of the ShoppingContainer class fills a pointless array of size 2500 bytes. This is the source of the memory leak.

```
Impact 2011 IBM Software Accelerated Value Program
```

```
private void deliberateMemoryLeak() {
   // -----
   // User clicked on the wheelbarrow, let's fill it with a
   // memory leak.
   // -----
   System.out.println("==> STARTING MEMORY LEAK");
   int LEAK SIZE = 10000;
   for (int i = 0; i < LEAK SIZE; i++) {</pre>
      leakObject.add(new ShoppingContainer());
   }
   System.out
   .println("==> Added "
         + LEAK SIZE
         + " objects to memory leak. The leak now contains "
          + leakObject.size() + " objects.");
   System.out.println("==> ENDING MEMORY LEAK");
}
class ShoppingContainer {
     private byte[] array;
     private long timestamp;
     ShoppingContainer() {
        array = new byte[2500];
        Arrays.fill(array, (byte) 65);
        timestamp = System.currentTimeMillis();
     }
 }
```

Θ

Θ

Part 7: Using the IBM Extensions to Memory Analyzer for Further Memory Analysis (10 minutes)

Note:

Impaci

ng the Way Business and IT Leaders Wo

"**IBM Extensions for Memory Analyzer**" add-ons have been installed into ISA. These have recently been released by IBM via the **IBM Alphaworks** site. They are unsupported, but very useful extensions to the Memory Analyzer tool.

They provide IBM product specific analysis of heap dumps by applying knowledge of the internal data structures of the IBM products into useful reports. For example, there is a WebSphere Application Server report to view the size and other details of the WebSphere objects in the heap dump that hold the HTTP sessions. This enables the Memory Analyzer tool to be used for **more than just diagnosing memory leaks**.

Most of these extensions require the additional data only available in a full system dump which you will analyze in this final part of the lab.

_____ Return to the ISA window and close the Memory Analyzer tab which contains the analysis of the IBM heap dump.

heapdump.20110224.094239.3072.1945	ohd - IBM Support Assistant Workbench				_ 🗆 ×
File Administration Update Views Window	telp				
Support Assistant		ک 🔒 📕	×3×	. Sitt	
Launch Activity 🤳 🏠 Home 🗙 🛃 A	nalvze Problem 🗙 🚱 IBM Monitoring and Diagnostic Tools 🗴				
🖾 Inspector 🛛 😂 🗖 🗖	😫 heapdump.20110224.094239.3072.1945.phd 🛛 🛛				- 8
	<u>i III 및 @ № + Q + Q + D + D +</u> 23 + 23				
	i Overview 😂 default_report org.eclipse.mat.api:suspects 🖳 show_dominator_tree [con				
	Class Name	Objects	Shallow Heap 🔻	Retained Heap	
	🔆 com.ibm.websphere.samples.plantsbywebspherewar.*	<numeric></numeric>	<numeric></numeric>	<numeric></numeric>	
	© com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet\$ShoppingContainer	87,163	2,789,216	>= 222,440,052	
	G com.ibm.websphere.samples.plantsbywebspherewar.ImageServlet	1	24	>= 100	
	G com.ibm.websphere.samples.plantsbywebspherewar.ShoppingServlet	1	24	>= 225,751,604	
	G com.ibm.websphere.samples.plantsbywebspherewar.AccountServlet	0	0	>= 76	
Statics Attributes Class Hierarchy	G com.ibm.websphere.samples.plantsbywebspherewar.AdminServlet	0	0	>= 76	
Stades Attributes classifierarchy	Σ Total: 5 entries (17,167 filtered)	87,165	2,789,264		
Type Name Value	-				

____ Click "File->Open heap dump".

____ Navigate to the system dump you previously processed with jextract.

Select the compressed .dmp file and click "Open". Please be patient, parsing a system dump can take longer that a compressed IBM heap dump. Approximately 3-5 minutes may pass with little nothing reported by the progress bar.

Impact2011

Open Snapshot			? ×
Look in:	C AppSrv01	🕝 🦻 📂 🖽]-
	Name 🔺	Size	Туре 🔺
	Cologs		File Folder
My Recent	in properties		File Folder
Documents	🗀 samples		File Folder
	i i i i i i i i i i i i i i i i i i i		File Folder
	🛅 tranlog		File Folder
Desktop	🚞 wstemp		File Folder
	🗖 core.20110224.094239.3072.1944.dmp	468,115 KB	DMP File
	Ecore.20110224.094239.3072.1944.dmp	101,884 KB	Compressed (
	🗖 heapdump.20110224.094239.3072.1945.phd	12,244 KB	PHD File
My Documents	Leak	77 KB	Compressed (
	🗐 javacore.20110224.094230.3072.1940	1,903 KB	Text Documer
	🗐 javacore.20110224.094230.3072.1941	1,902 KB	Text Documer
Mu Computer	🗐 javacore.20110224.094230.3072.1942	1,906 KB	Text Documer
My Computer	🗐 javacore.20110224.094230.3072.1943	1,906 KB	Text Documer 📕
	E increase 20110224 004024 2072 104/	1.004 //D	T
My Network Places	File name: core.20110224.094239.3072.1944	.dmp 💌	Open
	Files of type: All Known Formats	•	Cancel

____ On the "Getting Started Wizard" box, click "**Cancel**" to avoid generating the standard reports.

Click the reports icon and select "IBM Extensions->WebSphere Application Server->Application Classloader Leaks".

This is a useful way of visualizing which class loaders, and therefore which WebSphere applications are exhibiting memory leaks. In this lab, you will find one has been identified as exhibiting signs of a memory leak.

Changing the Way Business and IT Leaders Work

i III 🖳 💀 🔎 ▼ 🚱 ▼ 🔍	
i Overview 😪 application_classloader_leaks 🛛	
0x0294f450 com/ibm/ws/classloader/CompoundClassLoader	
0x02988a48 com/ibm/ws/classloader/CompoundClassLoader	
0x02991978 com/ibm/ws/classloader/CompoundClassLoader	
0x02992380 com/ibm/ws/classloader/CompoundClassLoader	
0x029933b0 com/ibm/ws/classloader/CompoundClassLoader	
0x02995018 com/ibm/ws/classloader/CompoundClassLoader	
0x029a9d18 com/ibm/ws/classloader/CompoundClassLoader	
0x02a9bd80 com/ibm/ws/classloader/CompoundClassLoader	
0x02a9c778 com/ibm/ws/classloader/CompoundClassLoader	
0x02a9f150 com/ibm/ws/classloader/CompoundClassLoader	(O DelegatingClassLoader, 1 JSPExtensionClassLoader)
0x02afe788 com/ibm/ws/classloader/CompoundClassLoader	
0x02aff090 com/ibm/ws/classloader/CompoundClassLoader	(O DelegatingClassLoader, 3 JSPExtensionClassLoader)
0x02b00598 com/ibm/ws/classloader/CompoundClassLoader	
0x02c193d8 com/ibm/ws/classloader/CompoundClassLoader	
0x02c19da8 com/ibm/ws/classloader/CompoundClassLoader	
0x02fc0248 com/ibm/ws/classloader/CompoundClassLoader	
0x02fc0d20 com/ibm/ws/classloader/CompoundClassLoader	
0x03002fe8 com/ibm/ws/classloader/CompoundClassLoader	
0x03003d48 com/ibm/ws/classloader/CompoundClassLoader	
0x03049710 com/ibm/ws/classloader/CompoundClassLoader	
0x0304f128 com/ibm/ws/classloader/CompoundClassLoader	
0x030567d0 com/ibm/ws/classloader/CompoundClassLoader	
0x03058aa8 com/ibm/ws/classloader/CompoundClassLoader	
Stopped application class loaders	
0x0249d860 com/ibm/ws/classloader/CompoundClassLoader	
[unknown LEAK]	

_ Left click the class loader identified as leaking and select List Objects->with outgoing references

002 - 6-89		·r	- ····- ,- , , ,			
0x02a	List objects	•	with outgoing references			
0x02b(Show objects by class	►	with incoming references			
<u>0x02c:</u> 🏣	Path To GC Roots	•	undClassLoader			
0x02c1	Merge Shortest Paths to GC Roots	►	undClassLoader undClassLoader			
0x02f(** 0x02f(IBM Extensions	►	undClassLoader			
0x030(Java Basics	•	undClassLoader			
<u>0x030(</u>	Java Collections	•	undClassLoader			
0x030+ 0x030+	Leak Identification	►	undClassLoader undClassLoader			
0x030	Immediate Dominators		undClassLoader			
<u>0x030!</u>	Show Retained Set		undClassLoader			
Stoppe	Сору	►				
🗞	Search Queries					
0x0249d86	0 com/ibm/ws/classloader/C	omp	oundClassLoader			

[unknown LEAK]

_ Expand the class loader references.

The resulting objects loaded by this class loader relate to the Plants sample which gives a clear indication of the application which is leaking memory.

i III 🖫 💀 🔎 🕶 😂 🕶 🔍 🖂 🕶 🖾 🕶 🦣		
i Overview 😂 application_classloader_leaks 🎴 list_objects [context] 🔀		
Class Name	Shallow Heap	Retained Heap
a 🔯 com.ibm.ws.classloader.CompoundClassLoader @ 0x249d860	152	39,088
class> class com.ibm.ws.classloader.CompoundClassLoader @ 0x1b6bca0	9,496	10,680
b arent, parent com.ibm.ws.classloader.ProtectionClassLoader @ 0x2170cd0	96	648
assertionLock java.lang.ClassLoader\$AssertionLock @ 0x249d8f8	16	16
packages java.util.Hashtable @ 0x249daa8	40	136
IazyInitLock java.lang.ClassLoader\$LazyInitLock @ 0x249db10	16	16
packageSigners java.util.Hashtable @ 0x249db20	40	168
methodCache java.util.Hashtable @ 0x249db88	40	104
FieldCache java.util.Hashtable @ 0x249dbf0	40	18,610
ConstructorCache java.util.Hashtable @ 0x249dc58	40	104
pds java.util.HashMap @ 0x249dcc0	48	160
nativelibpaths java.lang.String[0] @ 0x249dd40	16	10
In libraryClassLoaders com.ibm.ws.classloader.CompoundClassLoader[0] @ 0x249	16	10
reloadableParents java.util.Vector @ 0x249dd60	32	8
Image:	24	9,130
providers com.ibm.ws.classloader.SinglePathClassProvider[1] @ 0x24a6d88	24	2,544
IocalClassPath java.lang.String @ 0x24a6da0 C:\IBM\WebSphere\AppServer\pro	32	290
preDefinePlugins java.util.ArrayList @ 0x24a6ec8	32	88
class com.ibm.websphere.samples.plantsbywebsphereejb.Supplier @ 0x3193f60	229	229
class com.ibm.websphere.samples.plantsbywebsphereejb.OrderItem @ 0x31ca1	391	391
class com.ibm.websphere.samples.plantsbywebsphereejb.OrderItem\$PK @ 0x31	265	26
class com.ibm.websphere.samples.plantsbywebsphereejb.IdGenerator @ 0x3279	154	154
class com.ibm.websphere.samples.plantsbywebsphereejb.Inventory @ 0x345122	644	644
annotationCache java.util.Hashtable @ 0x34918e8	40	7,432
class com.ibm.websphere.samples.plantsbywebsphereejb.BackOrder @ 0x34aa4	303	303
class com.ibm.websphere.samples.plantsbywebsphereejb.Order @ 0x37c9f28	666	666

____ Click the reports icon Cache Analysis".

- 😂

and select "IBM Extensions->WebSphere Application Server->WAS

This illustrates the contents of the WebSphere caches such as dynacache. These types of caches reside in memory. If the WAS cache report shows a high memory footprint, the size of the caches can be limited with WebSphere administration, or the cache contents can be automatically offloaded to disk. For this lab, there is no action required.

Cache Analysis	Analysi	s							
• WAS Caches I	Found								
Address	Cache Name			Cache Size	Cache Size Limit	Disk Offload	Shallow	Heap Retained	Неар
<u>0x2a9dde0</u>	com.ibm.workpla	ace/Extensio	onRegistryCache	14	5000	disabled	320	169024	
<u>0x2a2b610</u>	com.ibm.ws.wsse	ecurity.sctS	erviceCacheMap	0	2000	disabled	320	64184	
0x2a061e8	com.ibm.ws.wsse	ecurity.sctC	lientCacheMap	0	2000	disabled	320	64184	
∑ Total: 3 entries									
r EJS Caches Fo	ound Cache Name	Entries	Soft Limit	lard Limit E	viction Strategy	chal	ow Heap	Retained Heap	1

BackgroundLruEvictionStrategy 72

205416

.

0

2053

Click the reports icon and select "IBM Extensions->WebSphere Application Server->Web Container Analysis".

2258

This shows details of all the configured web applications. Observe that in this case only the Plants by WebSphere application has any active sessions. In the next step, we will check the memory size of these sessions.

1	Ⅲ 😤 🔍 陆 🔻 🏹 🔻		
i ()verview 음 was_overview	😪 web_container_analysis 🛛	
W	eb Application Analy	isis	

EJB Cache

Web Application Analysis

Web Application Details

Impact20

0x2b2ce08

∑ Total: 2 entries

av Business and IT Leaders Work

11

Address	Virtual Host Name	Web Group Name	Web App Name	Loader	Destroyed	Current Sessions
<u>0x3717a60</u>	default_host	/PlantsByWebSphere/docs/*	PlantsByWebSphere	war:PlantsByWebSphere/PlantsGallery.war	false	0
<u>0x27f9430</u>	default_host	/SamplesGallery/*	SamplesGallery	war:SamplesGallery/GalleryMenu.war	false	1
<u>0x2a79258</u>	default_host	/IBM_WS_SYS_RESPONSESERVLET/*	ibmasyncrsp	war:ibmasyncrsp/ibmasyncrsp.war	false	0
0x30200e8	default_host	/WSsamples/*	SamplesGallery	war:SamplesGallery/Gallery.war	false	1
0x1f48198	default_host	/*	DefaultApplication	war:DefaultApplication/DefaultWebApplication.war	false	0
0x2d815c8	default_host	/PlantsByWebSphere/*	PlantsByWebSphere	war:PlantsByWebSphere/PlantsByWebSphere.war	false	391
0x32321c8	admin_host	/ISCAdminPortlet/*	isclite	war:isclite/ISCAdminPortlet.war	false	0
0x262c598	admin_host	/FileTransfer/*	filetransferSecured	war:filetransferSecured.ear/filetransfer.war	false	0
<u>0x312e568</u>	admin_host	/ibm/help/*	isclite	war:isclite/iehs.war	false	0
<u>0x36e1de8</u>	admin_host	/wim/*	isclite	war:isclite/WIMPortlet.war	false	0
<u>0x2661028</u>	admin_host	/ibm/console/*	isclite	war:isclite/isclite.war	false	0
0x37fb5c8	admin_host	/wasportlet/*	isclite	war:isclite/wasportlet.war	false	0
∑ Total : 12 entri						



, select "IBM Extensions->WebSphere Application Server->HTTP Click the reports icon Sessions". This shows details of all HTTP sessions including the size, session attributes, timeout, user ID and session ID.

Ensure the table of sessions is ordered by "Retained Heap". Notice how some sessions for PlantsByWebSphere are around 10k while others are just 0.5k.

i III 🖳 🖳 🔎 🖶 🕶 🔍 🖂 🖛 🕗 🕶 🛃

Impact20

ing the Way Business and IT Leaders Work

Class Name	Shallow Heap	Retained Heap	AppName
₩ <regex></regex>	<numeric></numeric>	<numeric></numeric>	<regex></regex>
Com.ibm.ws.session.store.memory.MemorySession @ 0x391f368 for memory store Sar	112	181,696	default_host/SamplesGallery
Com.ibm.ws.session.store.memory.MemorySession @ 0x39a27f0 for memory store Pla	112	10,712	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0xda12f40 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0xd006840 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0xba6d460 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0x5599348 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0x51e6720 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0x3994dc8 for memory store Pla	112	10,672	default_host/PlantsByWebSpher
Com.ibm.ws.session.store.memory.MemorySession @ 0x3986da0 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x3945948 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x390d0f0 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38fc468 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38f8b70 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38f27f8 for memory store Plan	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38cf050 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38ce6e0 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0x38ca6e8 for memory store Pla	112	10,672	default_host/PlantsByWebSphe
com.ibm.ws.session.store.memory.MemorySession @ 0x10d1a990 for memory store P	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xf842d90 for memory store Pla	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xefe59d8 for memory store Pla	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xde8d398 for memory store Pla	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xd4165b0 for memory store Pla	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xd394188 for memory store Pla	112	472	default_host/PlantsByWebSphe
Com.ibm.ws.session.store.memory.MemorySession @ 0xd3400c8 for memory store Pla	112	472	default_host/PlantsByWebSpher

Highlight and then expand one of the **10k** session objects. The session attributes have been automatically extracted from the WebSphere objects and presented by the IBM Memory Analyzer extensions. Notice there is a key called "deliberateLargeSession" which contains a very long String of letter "C"'s. This looks like another deliberate mistake in the Plants application.

com.ibm.ws.session.store.memory.MemorySession @ 0x5599348 for memory store Pla	112	10,672
Key=deliberateLargeSession,Value=byte[10240] @ 0x559d110 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC	32	10,288
All Outgoing References of the Session Object	0	0
∑ Total: 2 entries		

Make a note of the "_sessionId" for the object. This can be seen on the "Attributes" tab, right click it and choose Copy->Value.

http://www-01.ibm.com/software/support/acceleratedvalue/



Changing the Way Business and IT Leaders Work

Impact₂₀

Statics /	Attributes Clas	s Hierarchy		ø	_	com.ibm.	ws.se	ession	n.store.memory.MemorySes
Туре	Name	Value	yWebSphere at /PlantsByWe m.ws.session.StoreCallback			Key=deliberateLargeSession,Value=byte[10240 All Outgoing References of the Session Ob Total: 2 entries com.ibm.ws.session.store.memory.MemorySe			
ref ref	_store _storeCall	-							
ref	_sessionId	8lcujxsV2q	חבדב	MISCOSICE		8			n.store.memory.MemorySes
boole	_isValid	true		Go Into					.store.memory.MemorySes
boole	_isNew	false		List objects			+		n.store.memory.MemorySes
long	_currentAc	129596157		Show objects by cla	cc			sion	.store.memory.MemorySes
long	_lastAcces	129596157	la-a	Path To GC Roots			ĺ.	sion	n.store.memory.MemorySes
long	_creationT	129596157	100					sion	n.store.memory.MemorySes
int	_version	0	6	Merge Shortest Path	ns to	GC Roots	1	sion	n.store.memory.MemorySes
ref	_userName	anonymo		IBM Extensions			•		n.store.memory.MemorySes
int	_maxInacti	120		Java Basics		•	sion.store.memory.MemorySes		
int	_refCount	0		Java Collections					n.store.memory.MemorySes
ref	_appName	default_ho		Leak Identification			*		n.store.memory.MemorySes
ref	_attributes	java.util.H	P 2	Immediate Dominat	tore				n.store.memory.MemorySes
ref	_attribute	java.util.H			.013				n.store.memory.MemorySes n.store.memory.MemorySes
ref	_adaptation	com.ibm.	-	Show Retained Set			-		
ref	_appAdapt	null		Сору			•		Address
boole	isOverflow	false	8	Search Queries					Class Name
boole	invalInPro	false							Value
ref	_gcCount	null						D	Save Value To File
ref	_smc	com.ibm.	ws.se	ssion.SessionManag					Selection
ref	appName			I					Servetion

The best way to relate this unusually large session to the application code is to search the log files for the session ID. If the application uses this in its logging, you may be able to determine what the user did to cause the large session.

Note:

If you are using the "pre-prepared" system core instead of an "in-flight" system core generated during this lab, you will not be able to relate the session ID to the log files. In which case, simply read the remaining steps for this part of the lab.

Launch Windows Explorer. Navigate to and open file: "C:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1\SystemOut.log"

_____ Click "Edit->Find" and search the file for the Session ID you identified in the heap dump. It should reveal a log statement that gives a clue about the application's actions.

SystemOut0 ==> STARTING DELIBERATE LARGE SESSION for ID=Qxo_5N8FZub0W38CoGqrYnoSystemOut0 ==> ENDING DELIBERATE LARGE SESSION



Optional Steps:

Double click the desktop shortcut for ShoppingServlet.java



_ Click "Edit->Find" and and search for "STARTING DELIBERATE LARGE SESSION".

The deliberate mistake is clear – for any user that clicks on the white poinsettia image, their session is loaded with an attribute containing a 10k string of "C"'s (ASCII code 67). Congratulations, you have successfully located the final deliberate mistake in the plants sample.

```
System.out.println("==> STARTING DELIBERATE LARGE SESSION for ID="+req.getSession().getId());
byte[] sessionAttr = new byte[10240];
Arrays.fill(sessionAttr, (byte) 67);
req.getSession().setAttribute("deliberateLargeSession", sessionAttr);
System.out.println("==> ENDING DELIBERATE LARGE SESSION");
```



Appendix A: Optional - Using the ISA and GCMV to Analyze GC Data (10 minutes)

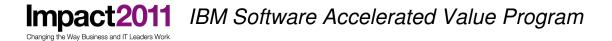
Double click the IBM Support Assistant 4.1 shortcut on the desktop.



Click the blue "Launch Activity" button and choose "Analyze Problem".

Select "Garbage Collection and Memory Visualizer" and click the grey "Launch" button at the bottom of the screen.

ich Activity 🦆 🏠 Home 🗙 🗟 Analyze Problem 🗴				
👕 Tools 📙 Collect Data 🥳 Guided Troubleshooter				
Case/Incident				
default Select				
Tools Catalog	Find New Tool Add-ons	Description		
Tool Name	Version	The IBM Monitoring and Diagnostic Tools for Jav Garbage Collection and Memory Visualizer is a		
IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Visua	lizer 2.4.0.20101007	verbose GC data visualizer. The GC and Memo		
IBM Monitoring and Diagnostic Tools for Java™ - Health Center	1.3.0.20101104	Visualizer parses and plots various log types inclu verbose GC logs, -Xtgc output, native memory lo		
IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer [Tech Preview]	0.6.0.201101042253	(output from ps, symon and perfmon).		
		It provides:		
		- a graphical display of a wide range of verbose GC data values		
		- tuning recommendations and detection of		
		problems such as memory leaks		
		- report, raw log, tabulated data and graph views		
		Restrictions		
		None		
		Associations		
		Tool is not associated with any products		



__ Click "Next" to go directly into the GCMV tool.

🚝 Tool Input Parameter Values	
☐IBM Monitoring and Diagnostic Tools for Java™ - Garb	age Collection and Memory Visualizer
Log name	Browse
	Next > Cancel

Click "File->Open File" and navigate to "C:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1\navtive_stderr.log"

_____ Click "OK" and wait a few moments for the log file to be parsed.

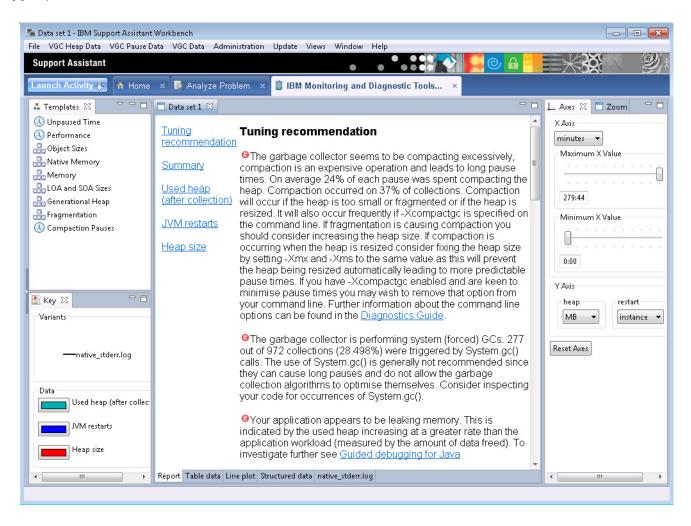
_____ Click the "Report" tab at the bottom of the screen.

Review the recommendations - they relate to the same garbage collection issues you identified with live monitoring using Health Center, i.e.:

- Excessive compaction caused by excessive calls to System.gc()
- Memory leakage leading to an out of memory condition
- Large object allocations

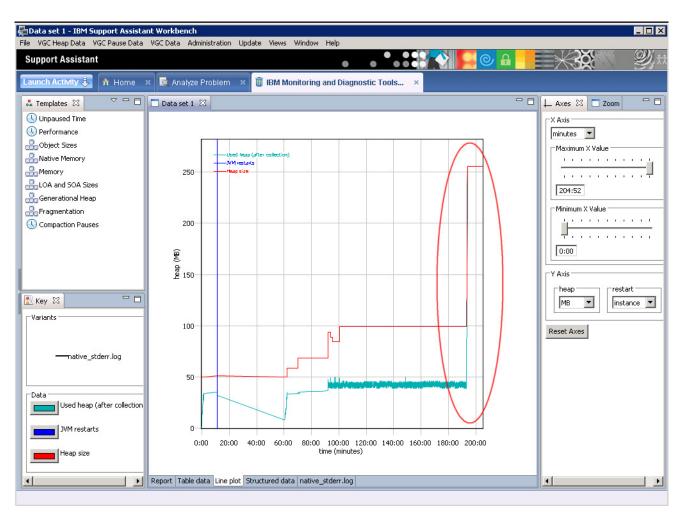
The report page also shows a summary table of garbage collection statistics.

Changing the Way Business and IT Leaders Work



Click the "Line Plot" tab at the bottom of the screen. The graph shows the heap size and used heap after garbage collection. You will see these were stable for a while until a sudden increase when the JVM experienced a rapid memory leak (highlighted in this document by the red oval)

Changing the Way Business and IT Leaders Work



The templates on the left hand side can be double clicked to show different garbage collection statistics. For example, double click the "**Object Sizes**" template graph which shows the size of objects that triggered an allocation failure. You will see very frequent requests for objects ranging from 5Mb to 10Mb (recall a large object allocation was one of the deliberate errors in the plants sample code)



💑 Templates 🖾	~						
🕔 Unpaused Time							
N Performance							
Object Sizes							
Native Memory							
Memory							
🔒 LOA and SOA Sizes							
🔒 Generational Heap	🔒 Generational Heap						
Compaction Pauses							

_ Double click "**Memory**" in the templates window.

You can customize any graph by choosing the data to plot. Click the menu "VGC Pause data" and enable (with a tick) menu item "Pause Time (mark sweep compact)".

👳 remplates - IBM S	Support Assista	nt Workbe	ench					
File VGC Heap Data	VGC Pause Data	VGC Data	Administration	Update	Views	Window	Help	
Support Assista	Select All Deselect All							
Launch Activity	Card cleaning t Compact times							
🕺 Templates 🛛	Exclusive access times							
Unpaused Time	Intervals between garbage collection triggers Intervals between garbage collections (mark-sweep, nursery, or global)							
Performance Mark times								
Object Sizes Pause time (mark, sweep compact)								
Native Memory	Sweep times							
Memory	Time spent unl	pading class	loaders					
LOA and SOA Siz	Time spent unle Total pause tim	-	es					

This will plot the total GC time which you see increasing dramatically as the JVM struggles to cope with the rapid demands of the **memory leak**.

Impact2011 Changing the Way Business and IT Leaders Work

VGC Heap Data VGC Pause Data	/GC Data Administration Update Views Window Help	
upport Assistant		◎
aunch Activity 퉞 🚹 Home 🔅	🗟 Analyze Problem 🛛 🗴 🗑 IBM Monitoring and Diagnostic Tools 🗡	
Templates 🛛 🔍 🗖 🗖	Data set 1 🕱	Axes 🔀 🗖 Zoom 🖓
🕓 Unpaused Time		_ X Axis
Derformance		minutes
Object Sizes		0.95
Native Memory		-0.90
Memory	250	0.85
LOA and SOA Sizes		0.80
Generational Heap		0.75
Fragmentation		Minimum X Value
Compaction Pauses	200	0.70
		0.65
		0.60
	(2) de 150	-0.55
	<u> </u>	-0.50 -
		-0.45 0 heap restart
Key 🛛 🗖 🗖		-0.40 g MB I instance
/ariants		-0.35
	100	
		-0.30 seconds
	WebSphere\AppServer\profiles\AppSrv01\logs\server1\native_stderr.log	-0.25
· · · · ·		0.20 Reset Axes
Data	50	0.15
Used heap (after collection		0.10
	the spendore domain of the second	-0.05
Heap size		
JVM restarts		J-0.00
	0:00 20:00 40:00 60:00 80:00 100:00 120:00 140:00 160:00 180:00 200:01 time (minutes)	°
Pause time (mark, sweep c	······································	
	Report Table data Line plot Structured data native stderr.log	

Another interesting statistic is "Intervals Between Garbage Collections". Click the menu "VGC Pause Data" and enable (with a tick) menu item "Intervals Between Garbage Collections".

In this case, while the jmeter load generator was running, there is a often very short interval because garbage collections as the plants sample application was constantly calling **System.gc()**.

Changing the Way Business and IT Leaders Work

VGC Heap Data VGC Pause Data	VGC Data Administration Update Vie	ews Window Help		
upport Assistant				×38 9
aunch Activity 🦆 🚹 Home 💠	< 🗟 Analyze Problem 🛛 🗑 IBN	M Monitoring and Diagnostic Tools ×		
Templates 🛛 🔍 🗖 🗖	🗖 Data set 1 🛛		- 8	🖵 Axes 🛛 🗖 Zoom 🗖
🕔 Unpaused Time				_ X Axis
N Performance				minutes 🔻
Object Sizes				Maximum X Value
Native Memory		ection) Bage collections (mark-sweep, nuisery, or global)		
Memory	250 Pause Gime (mark, sw			
LOA and SOA Sizes				
Generational Heap				204:52
Fragmentation				Minimum X Value
Compaction Pauses	200			
	~			0:00
	N, N		400	
	ନ୍ଥି ଜୁ ଅ 150			_ Y Axis
	_		time	heap
Key 🛛 🗖 🗖				MB Instance
			300 8	
Variants 🦳 🗖	100		<u>s</u>	
		R. I.		seconds 🔻
Data	50			Reset Axes
Used heap (after collect				
Heap size				
JVM restarts	0	W P DALBAR AND ALL ALL AND A DALBAR AND A DAL		
Pause time (mark, swee	0:00 20:00 40:00 6	50:00 80:00 100:00 120:00 140:00 160:	00 180:00 200:00	
Intervals between garb		time (minutes)		
	Report Table data Line plot Structured	d data native stderr.log		time (minutes)

_____ Try changing the color of the lines by clicking the colored line icon.

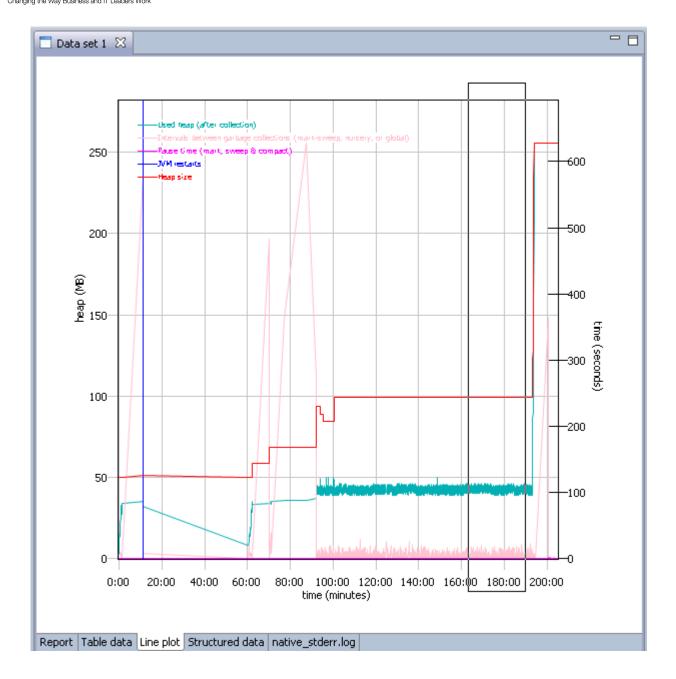


	Generationa	Color		-X
	品 Fragmentati	Basic colors:		
	🕔 Compactior			
I				
Ļ				
I	🛃 Key 🔀 📃			
l	-Variants			
I		Custom colors:		
I				
I	nac			
I		Define Cu	stom Colors >>	
I	Data JVM		Cancel	
I			Jancer	
I	Heap	size		
I		heap (after collec		0
		neup (arter conec		0:00
	Paus	e time (mark, swee		

Try dragging a box in the graph area to zoom on a particular time period. Hint – zooming works best if you include the axes in your zoom box. You can right click and select "Reset Zoom" at any time.

ess and IT Leaders Work

Impact2011 IBM Software Accelerated Value Program



Close the GCMV tab but leave IBM Support Assistant Workbench open.

Changing the Way Business and IT Leaders Work

upport Assistant			•			×33 3
unch Activity 🦺 🚹 Home 🔅	< 🗟 Analyze Prob	em 🗴 🗊 IBM Mo	onitoring and Diagnost	tic Tools ×		
Templates 🛛 🔍 🗖 🗖	🗖 Data set 1 🛛					Axes 🛛 🗖 Zoom 🗧
Unpaused Time						X Axis
Performance Object Sizes				1		minutes
Native Memory		-Used heap (after collection)				Maximum X Value
Memory	250	-Pause Gime (mark, sweep &	collections (mark-sweep, nuisery, o compact)	n global)		
LOA and SOA Sizes		—JVH restarts —Heap size			600	204:52
Generational Heap						
Fragmentation	200				500	Minimum X Value
Compaction Pauses	200					
	ŵ					0:00
	(원) 문 목 150				400	
	<u>p</u> 150				- F	TY Axis
Key 🛛 🗖 🗖						MB
Variants						
	100				~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	
					200	seconds 💌
Data	50		MARAMANALALA		and a statement	Reset Axes
Used heap (after collect	pmar.	unner anna 1944	MARTINE CONTRACTOR	unan una MAMA	100	
Heap size						
JVM restarts	0	mmmm	mmmmm	mmmm	mm	
Pause time (mark, swee	165	:00 170:00		80:00 185:00	190:00	
Intervals between garb			time (minutes)			

Reference Links

Impact2011

• IBM Support Assistant Information and Downloads:

http://www-01.ibm.com/software/support/isa/

• How to Install JVM Tools into ISA:

http://www-01.ibm.com/support/docview.wss?uid=swg27013279

• Memory Analyzer Tool for ISA:

http://www.ibm.com/developerworks/java/jdk/tools/memoryanalyzer/

Alphaworks IBM Memory Analyzer Extensions:
 http://www.alphaworks.ibm.com/tech/iema

• Eclipse Memory Analyzer:

http://www.eclipse.org/mat/

• Using the IBM DTFJ with the Eclipse Memory Analyzer Tool (i.e. 64-bit):

http://www.ibm.com/developerworks/java/jdk/tools/mat.html

• MustGather: Using the -Xdump Option:

http://www-01.ibm.com/support/docview.wss?uid=swg21242497#Limiting Dumps Using Filters

• Java Diagnostics Guide:

http://www.ibm.com/developerworks/java/jdk/diagnosis/

• Guided debugging for Java:

http://publib.boulder.ibm.com/infocenter/javasdk/tools/index.jsp?topic=/com.ibm.java.doc.igaa/ 1vg00011e 17d8ea-1163a087e6c-7ffe 1001.html



IBM Java Troubleshooting Blog: •

https://www.ibm.com/developerworks/mydeveloperworks/blogs/troubleshootingjava/?lang=en