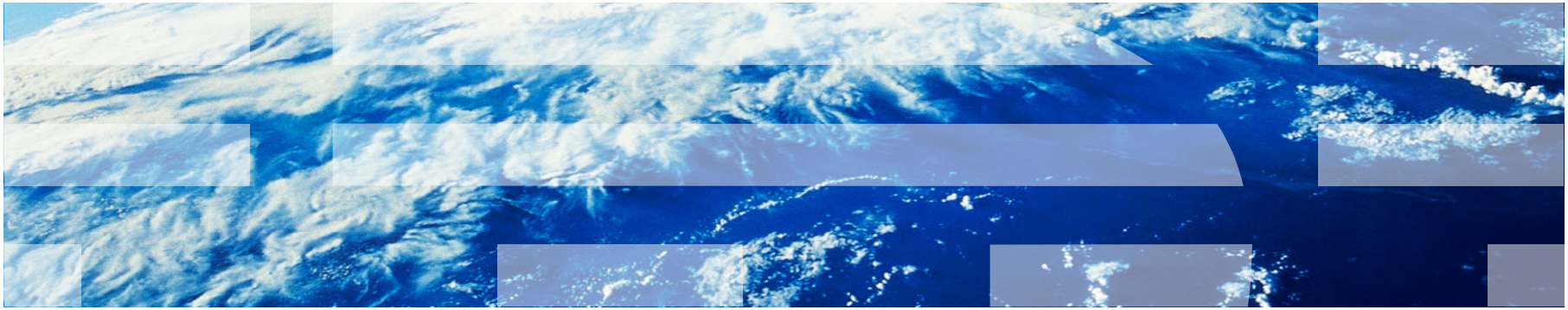
WebSphere Application Server

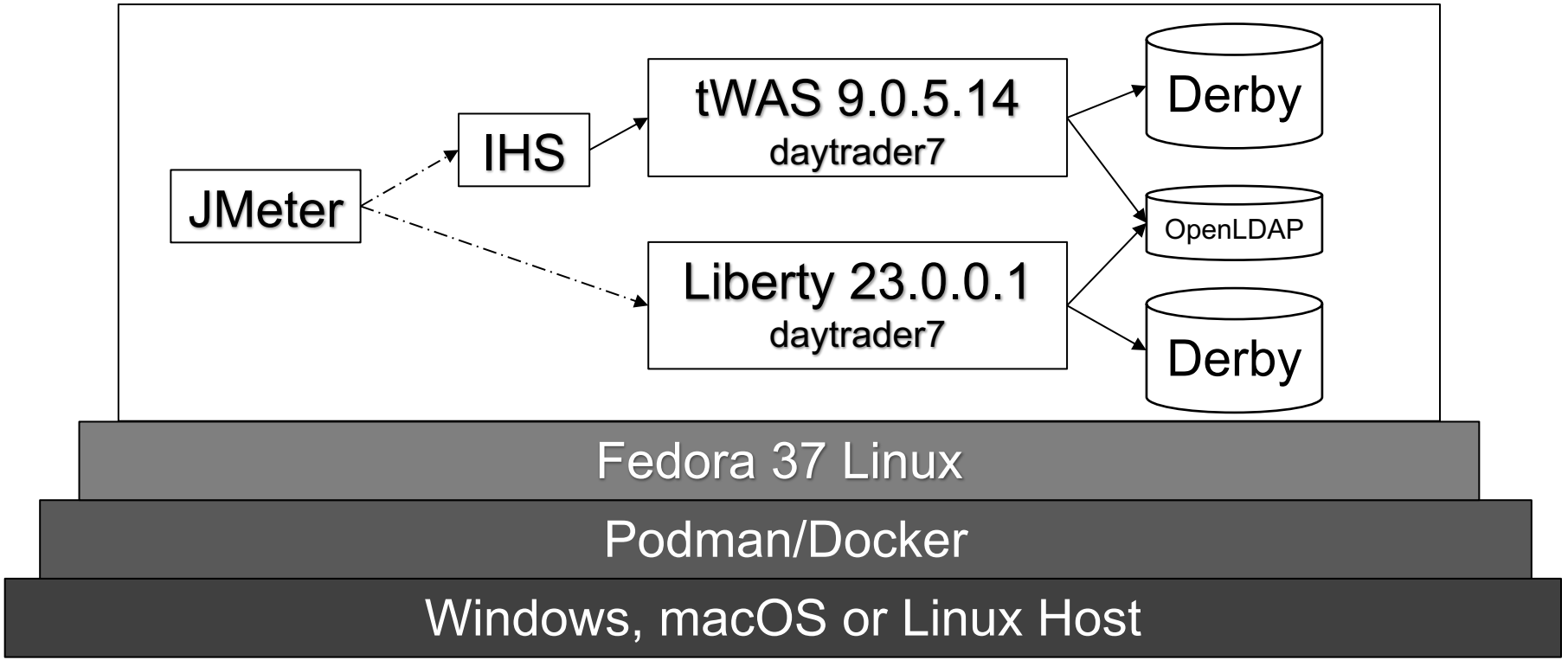
WebSphere Troubleshooting and Performance Lab



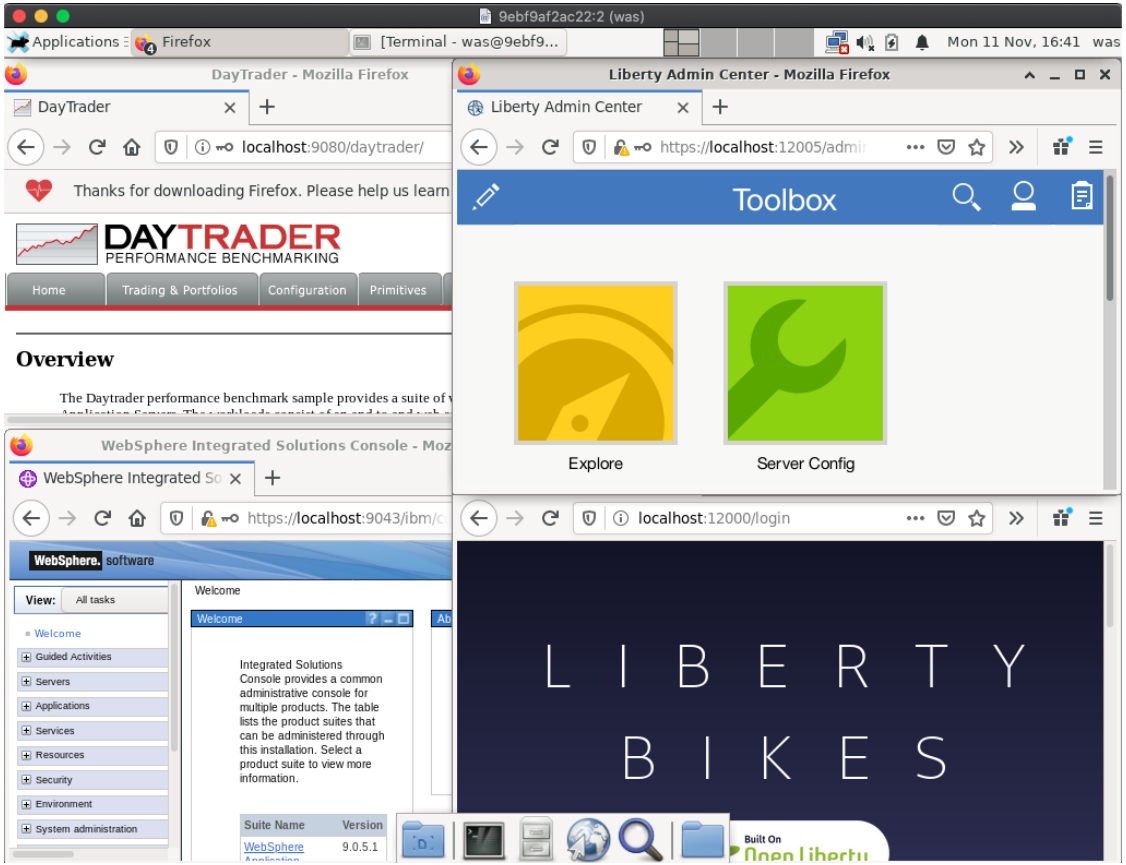
What is it?

- Self-paced, free, publicly downloadable WebSphere Liberty and WAS traditional troubleshooting and performance lab
- Learn: CPU analysis, Hang determination, Performance tuning, Memory analysis, etc.
- Over 100 pages of exercises which can be done in sequence or a la carte
- Hosted on Quay.io
    1. Install podman or Docker Desktop for Windows and macOS
    2. podman/docker run ... quay.io/ibm/webspherelab (see instructions for details)
    3. VNC/Remote Desktop into the lab

What's in it?


JMeter
IHS
tWAS 9.0.5.14
daytrader7
Liberty 23.0.0.1
daytrader7
Derby
OpenLDAP
Derby
Fedora 37 Linux
Podman/Docker
Windows, macOS or Linux Host

Screenshot


9ebf9af2ac22:2 (was)
Applications
Firefox
[Terminal - was@9ebf9...
Mon 11 Nov, 16:41
DayTrader - Mozilla Firefox
DayTrader
localhost:9080/daytrader/
Thanks for downloading Firefox. Please help us learn
DAYTRADER
PERFORMANCE BENCHMARKING
Home
Trading & Portfolios
Configuration
Primitives
Overview
The Daytrader performance benchmark sample provides a suite of
Liberty Admin Center - Mozilla Firefox
Liberty Admin Center
https://localhost:12005/admi
Toolbox
Explore
Server Config
WebSphere Integrated Solutions Console - Moz
WebSphere Integrated So
https://localhost:9043/ibm/c
WebSphere. software
View:
All tasks
Welcome
Guided Activities
Servers
Applications
Services
Resources
Security
Environment
System administration
Welcome
Integrated Solutions Console provides a common administrative console for multiple products. The table lists the product suites that can be administered through this installation. Select a product suite to view more information.
Suite Name
Version
WebSphere
9.0.5.1
localhost:12000/login
LIBERTY
BIKES
Built On

How to run it?

1. Windows/macOS: Configure Docker Desktop or podman:
   - Memory >= 4GB
   - Disk >= 30GB
   - Currently requires x86_64/amd64 CPU (others in the works)
2. Run the container from Command Prompt or Terminal:
   - podman run --cap-add sys_chroot --rm -p 5901:5901 -p 5902:5902 -p 3390:3389 -p 9080:9080 -p 9443:9443 -it quay.io/ibm/webspherelab
   - docker run --rm -p 5901:5901 -p 5902:5902 -p 3390:3389 -p 9080:9080 -p 9443:9443 -it quay.io/ibm/webspherelab
3. Wait 5 minutes until you see:

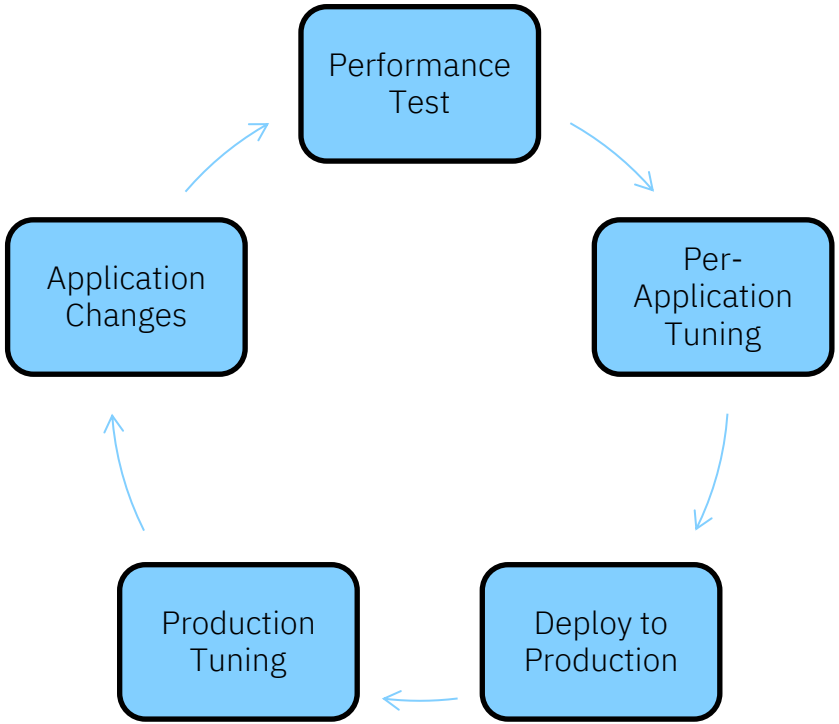
```
=========
= READY =
=========
```

How to run it?

4. Remote into the container:
    1. VNC to localhost:5902
        1. From the Terminal in macOS: open vnc://localhost:5902
        2. Linux Terminal: vncviewer localhost:5902
        3. Windows 3rd party VNC viewers
    2. Windows Remote Desktop: Requires configuration; see lab appendix
5. Perform the step-by-step lab:
   https://github.com/IBM/websphere lab/blob/main/Liberty_Perf_Lab.md

   Also available on the desktop inside the lab

Auto-tuning Thread Pool



Traditional Tuning

- Application thread pools
- Connection Pools (e.g. database, JMS, etc.)
- Java Virtual Machine
- Operating System
- Network

Liberty Tuning

- ~~Application thread pools~~ (Auto-tuning)
- Connection Pools (e.g. database, JMS, etc.)
- Java Virtual Machine
- Operating System
- Network

Major Recommended Tools

Tool	Analyze	Purpose
top -H or equivalent	CPU by Thread	Performance/Hang
monitor-1.0 / AdminCenter	Various statistics	Performance/Hang
requestTiming-1.0 & Access Log	Response times	Performance/Hang
Thread and Monitor Dump Analyzer (TMDA)	Thread dumps	Performance/Hang
Garbage Collection and Memory Visualizer (GCMV)	Verbose garbage collection	Performance/Hang
Eclipse Memory Analyzer Tool (MAT)	Heapdumps	OutOfMemoryError
Java Health Center	IBM Java	CPU Deep Dive

Liberty Tuning Top 10 Tips

1. Ensure your operating system CPU, RAM, disk, and network aren't saturated
2. Ensure time in Java garbage collection is less than ~5%; tune max heap size and nursery heap size
3. Liberty's main thread pool auto-tunes for throughput and generally should not be tuned
4. Monitor arrival rate, response times, utilization, error rate and throughput at each layer and investigate with thread dumps and/or a sampling profiler
5. If using databases, tune the maximum connection pool size (and make sure the DB supports it)
6. If using JMS MDBs, tune the maxConcurrency
7. If using security, tune the authentication cache size
8. Consider enabling request timing to watch for slow HTTP requests
9. Consider enabling the HTTP access log to understand and tune HTTP activity
10. Aggressively use caching where possible

For more, see the Liberty Performance Tips and the WebSphere Performance Cookbook

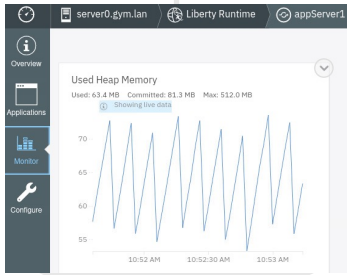
Monitoring statistics

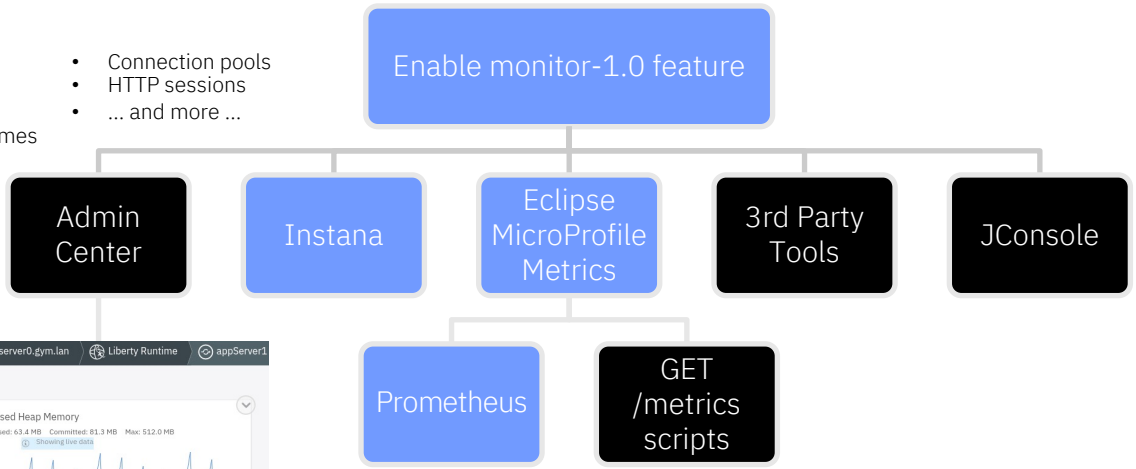
Java MXBeans are a standardized mechanism to expose statistics

Produce :

- Java heap
- Threads
- CPU
- Response Times
- Connection pools
- HTTP sessions
- ... and more ...

Consume





Slow and hung request detection

Enable

- Enable requestTiming-1.0 feature
- Configure slow and hung thresholds

```
<requestTiming slowRequestThreshold="2000ms"
hungRequestThreshold="600s" sampleRate="1" />
```

Monitor

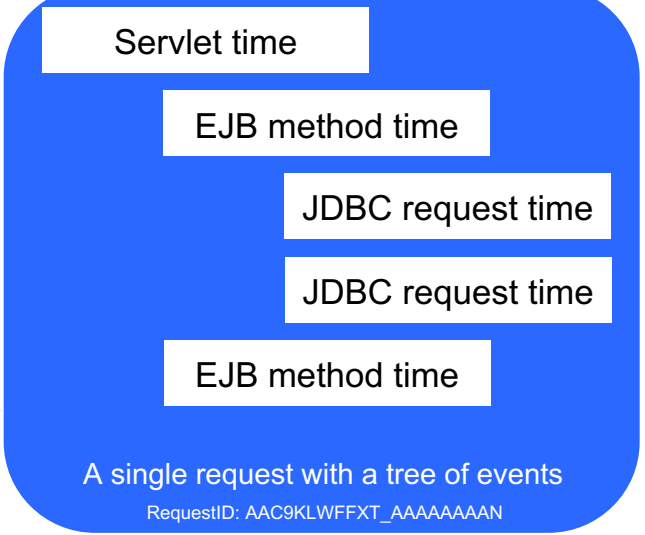
- Liberty watches all or a sampling of requests
- If a request exceeds a threshold, a warning (**TRAS0112W**) is printed to the logs

Analyze

- Time stamp of the breach
- Stack trace at the time threshold exceeded
- Tree of events leading up to the breach

```
[3/14/23 13:36:50:226 EDT] 0000004d
com.ibm.ws.request.timing.manager.SlowRequestManager         W TRAS0112W: Request
AAB3PkzwQZ9_AAAAAAAAADj has been running on thread 00000045 for at least
2001.972ms. The following stack trace shows what this thread is currently running.

        at java.lang.Thread.sleepImpl(Native Method)
        at java.lang.Thread.sleep(Thread.java:977)
        at java.lang.Thread.sleep(Thread.java:960)
        at
com.ibm.websphere.samples.pbw.war.ShoppingBean.getPriceInfo(ShoppingBean.java:221)
```



Demo

Thank you.
Questions?

Appendix

Performance Tuning in Containers

- Check overall cluster utilization and pod utilization and limits
- Scale number of pods: Horizontal pod autoscaler
- Java:
  - Use neither -XX:MaxRAMPercentage nor -Xmx, or the former if needed
  - Consider a mounted shared class cache
  - Enable rotating verbosegc: -Xverbosegclog:logs/verbosegc.%seq.log,20,50000
  - Consider using the JITServer
- Liberty:
  - Consider using InstantOn
  - Use JSON logging
  - Run configure.sh in the Containerfile

Other Day 2 Capabilities

- Ensure time spent in garbage collection is less than ~5-10% using the free GCMV tool.
- Review performance tuning guidance and the WebSphere Performance Cookbook.
- Consider integrating the WebSphere Automation product that helps with server inventory, security patching, and automatic memory leak detection and diagnosis.
- Use the free IBM Thread and Monitor Dump Analyzer (TMDA) to graphically review thread dumps.
- To review OutOfMemoryErrors, use the free Eclipse Memory Analyzer Tool (MAT) with DTFJ and IBM Extensions.

Other Day 2 Capabilities

- An optional servlet response cache that takes significant effort to configure but may have a dramatic return on investment in reduced server utilization and improved response times.
- Distributed tracing to monitor the flow of JAX-RS web service calls through a web of Liberty processes.
- Consider grouping Liberty servers running the same application into Liberty clusters for easier management.
- Consider the auto scaling features to dynamically scale the number of Liberty servers based on CPU and memory.
- Dynamic routing rules in a collective may be used to route requests to specific servers, redirect requests, or reject requests.

Other Day 2 Capabilities

- If running WebSphere Liberty on IBM Java or IBM Semeru Runtimes:
- A shared class cache to improve startup time.
- Various garbage collection policies for different workloads.
- Richer thread dumps to help diagnose many types of issues.
- Method tracing to take diagnostic actions on method entry or exit, or to time method calls.
- A dump engine to gather diagnostics on various events such as large object allocations, thrown or caught exceptions, etc.
- A JITServer to separate Just-In-Time compilation into a separate process and share compiled code across processes.
- The IBM Java Health Center sampling profiler available in IBM Java 8 and IBM Semeru Runtimes 11 on z/OS to investigate Java CPU usage and lock contention.
- For richer memory analysis, consider enabling and configuring core dumps (e.g. core and file ulimits, kernel.core_pattern truncation settings, etc.) after reviewing the security, disk and performance risks.

Other Day 2 Capabilities

- Consider using maintenance mode for clusters during diagnostics or maintenance operations.
- Consider using health policies to capture diagnostics and perform other operations based on different conditions.
- Flexible diagnostic trace capabilities with the option of using a binary output format for reduced overhead.
- Gather basic operating system metrics such as CPU, memory, disk, and network utilization, saturation, and errors.
- Gather operating system logs and watch for warnings and errors.
- Monitor for common network issues such as TCP retransmissions.
- Monitor the web server's access and error logs for warnings and errors.

Other Day 2 Capabilities

- Monitor the web server's utilization with tools such as mod_mpmstats and mod_status.
- For newer applications, advanced capabilities for fault tolerance such as automatic retries, circuit breakers, fallbacks, and bulkheads. In addition, health checks may be enabled using readiness and liveness probes.
- When running in a container environment such as OpenShift:
  - Consider deploying applications using the WebSphere Liberty Operator and use capabilities such as the WebSphereLibertyTrace and WebSphereLibertyDump custom resources.
  - Consider enabling application monitoring integrated with and Grafana.
  - If you have cluster-admin permissions, use the MustGather: Performance, hang, or high CPU issues with WebSphere Application Server on Linux on Containers during performance, hang, and high-CPU issues.