

# Liberty Quarterly Update

## 20.0.0.7-20.0.0.9

Alasdair Nottingham – Liberty Lead Architect

 @nottycode

# Agenda



Part 1: 10 Minute Liberty overview

Part 2: Liberty Performance Tuning – Vijay Sundaresan

Part 3: What is new this quarter

Part 4: Q&A

# 10 minute overview

# Why Choose Liberty



- Cloud Ready
  - Container optimized
  - Designed for dev/ops
  - Small disk footprint
  - Efficient memory usage
  - Fast startup
  - High throughput
  - Self-Tuned Thread Pool
- Developer friendly
  - Just enough application server
  - Fast inner loop with dev mode
  - Support for industry standard dev tools
  - Jakarta EE, Java EE, MicroProfile APIs
  - Zero Migration

# What people are saying

- "Light operation and efficient operation (restart time can be reduced compared to ND)"
- "Ease of use small footprint ability to be built into a container"
- "It's the best tool for the job."
- "Easy to manage, Cloud Ready a Smart Product with cool feature. Modular flexible and fast."
- "Simple, well supported, cloud and container friendly."
- "Liberty is a fast-to-configure application server with some advantages over other similar products."
- "Simplicity of use."
- "Now that we have experienced Liberty, we are spoiled and want to stick to those easy, non-impacting migrations."

Open Liberty



**Alex Lewis**  
@seekingbinary

Follow

Starting to use this in my own projects. Thoroughly recommended. Keep up the great work @OpenLibertyIO team.

**openlibertyio** @OpenLibertyIO

Dev mode support and more developer experience enhancements to the Liberty Maven plug-in: [openliberty.io/blog/2019/11/2...](https://openliberty.io/blog/2019/11/2...)

11:36 PM - 28 Nov 2019

5 Retweets 13 Likes



**Tim Zöller**   
@javahippie

Follow

The @OpenLibertyIO dev mode is one of the best hot-reload features I have ever worked with, I am seriously impressed!

11:56 PM - 25 Oct 2019

9 Retweets 25 Likes



**Philip Riecks**  
@rieckpil

Follow

This Maven plugin is really a gamechanger for the development experience when it comes to developing @JakartaEE and @MicroProfileIO applications on @OpenLibertyIO. Read my review here: [rieckpil.de/joyful-open-li ...](https://rieckpil.de/joyful-open-li...)

# Just Enough Application Server



You control which features are loaded into each server instance

Java EE



```
<feature>jsf-2.3</feature>
```

jsp-2.3

jsf-2.3

servlet-4.0

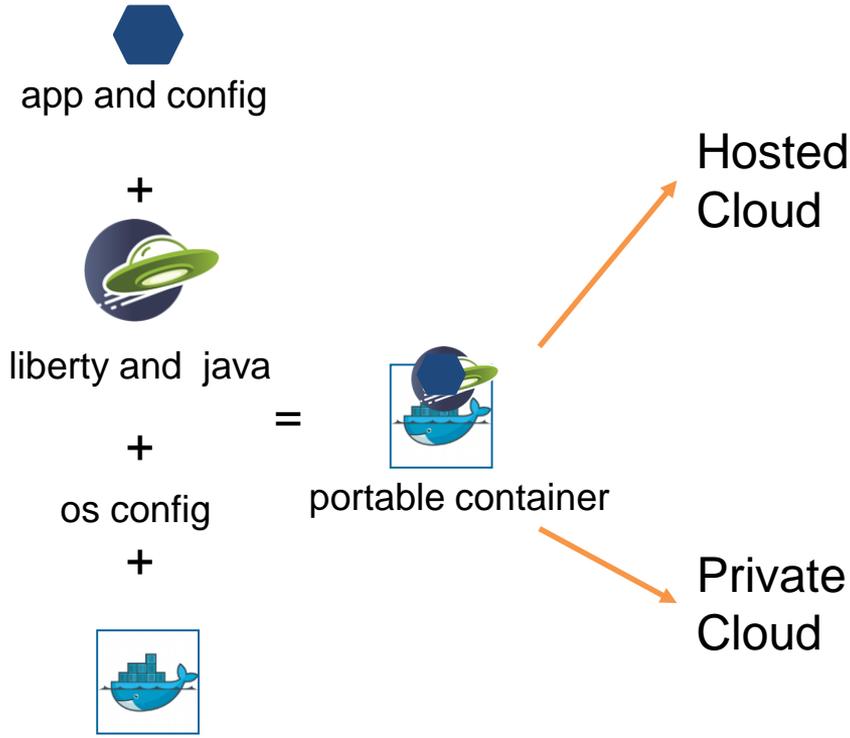
http-2.0

appmgr

Kernel



# Liberty in Containers



- IBM Cloud Kubernetes Service
- Azure Kubernetes Service
- Google Kubernetes Engine
- Amazon Elastic Kubernetes Service
- Jelastic

- Red Hat Open Shift Container Platform
- Pivotal Kubernetes Service
- Pivotal Cloud Foundry

FROM open-liberty  
ADD myapp.war /config/dropins/myapp.war

# Liberty Zero Migration

Open Liberty

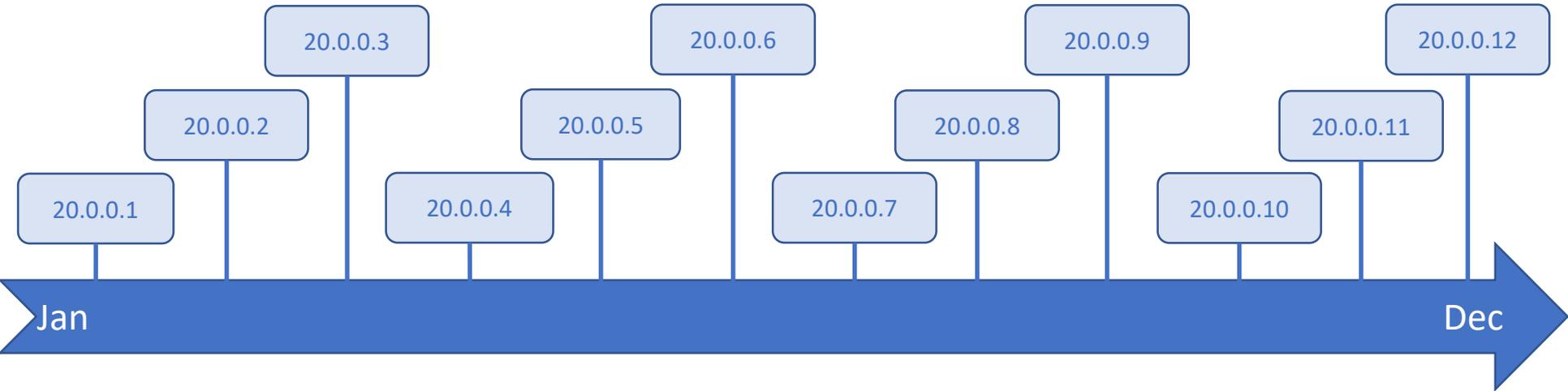


- Zero config migration
  - Write once, run forever
- Zero app migration
  - No behavior changes in existing features
  - New behaviors in new features
- Choose your Java
  - Java 14, 11, 8
  - AdoptOpenJDK
  - IBM
  - OpenJDK
  - Oracle



# Liberty Release Cadence

Open Liberty



	All CD releases	CD releases ending .3 .6 .9 .12
Support Provided	5 years	5 years
iFixes	24 weeks	2 years
Proactive Security iFixes	Most recent	Most recent 2

# Liberty Release Cadence Example

Open Liberty



	All CD releases	CD releases ending .3 .6 .9 .12
Support Provided	5 years	5 years
iFixes	24 weeks	2 years
Proactive Security iFixes	Most recent	Most recent 2

# How to get Support



## WebSphere



z/OS  
ND  
Base  
Core



## IBM Cloud Pack for Apps

### IBM Integrated Application Runtimes

Java:

- WebSphere
- Liberty
- MicroProfile
- Jakarta EE
- OpenJ9
- HotSpot

Node.js:

- Loopback
- Express



**Red Hat Runtimes**

**Integrated DevOps**



**Red Hat OpenShift**

## Red Hat Runtimes



Open Liberty

JBoss EAP

OpenJDK

Spring

Thorntail

Node.js

Red Hat AMQ

Red Hat Data Grid



# Key Performance and Troubleshooting tips for Liberty

Vijay Sundaresan

## Summary

- Top Open Liberty Performance Tips
  1. Tune maximum Java heap and nursery sizes using verbosegc
  2. Use IBMJCEPlus as the JCE provider if using the IBM JDK
  3. Tune thread pools and connection pools
  4. Configure smallest possible feature set needed by your application
  5. Ensure CPU, RAM, network, and disk aren't saturated
  6. Monitor arrival rate, response times, utilization, error rate and throughput at each layer and investigate with thread dumps and/or a sampling profiler
  7. Aggressively use caching where possible

## Summary

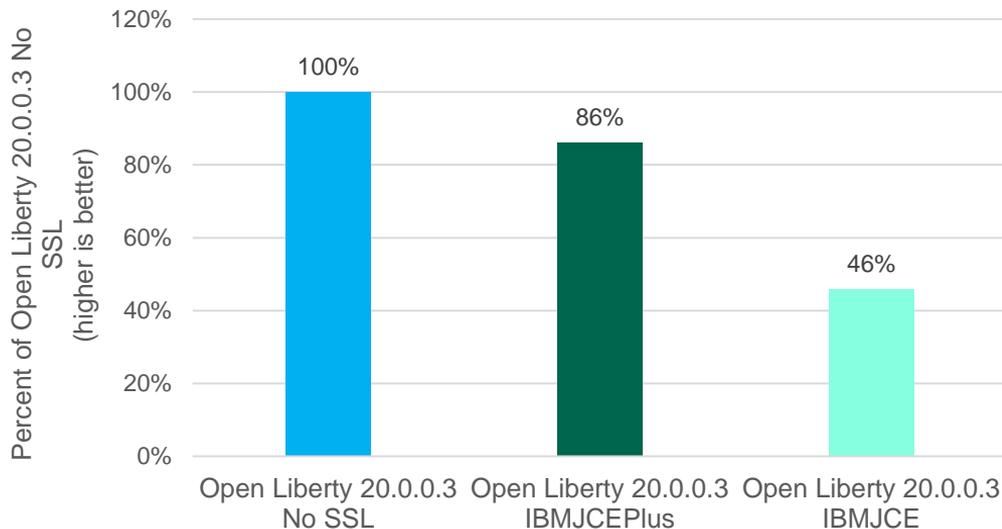
- Top Open Liberty Troubleshooting Tips
  1. Gather multiple thread dumps and collect relevant logs
  2. Use the requestTiming Liberty feature for long responses
  3. Use a sampling profiler
  4. For J9 JVMs, use OS core dumps instead of PHDs for OutOfMemoryErrors
  5. Consider using the NCSA Access Log

## Performance Tip #1: Java heap sizes

- The proportion of time in garbage collection should generally be less than 5%
- Always run with [verbose garbage collection](#), including in production
  - Overhead < ~0.2%
  - J9: `-Xverbosegclog:logs/verbosegc.%seq.log,20,50000`
  - HotSpot: `-Xlog:gc:logs/verbosegc.log`
- Load verbosegc into a tool like GCMV, click Report, and review [“Proportion of time in GC”](#)
- Tune maximum Java heap size based on performance testing and experimentation
  - `-XmxZm`: Fixed maximum heap size of Z megabytes, e.g. `-Xmx2048m`
  - `-XX:MaxRAMPercentage=N`: Maximum heap size based on physical RAM
  - `-XmnZm`: Nursery for generational collectors; try > 25% of max heap size

## Performance Tip #2: Use IBMJCEPlus provider with IBM JDK

Open Liberty SSL overhead comparison on X86



Procedure to use IBMJCEPlus :

Add the IBMJCEPlus provider to the provider list by editing the `java.security` file in the `JAVA_HOME/jre/lib/security/` directory :

```
security.provider.1=com.ibm.security.jgss.IBMJGSSProvider
security.provider.2=sun.security.provider.Sun
```

```
security.provider.3=com.ibm.crypto.plus.provider.IBMJCEPlus
security.provider.4=com.ibm.crypto.provider.IBMJCE
```

- TLSv1.2 with `https.cipherSuites=TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`
- IBM JDK 8 SR6 FP7 IBMJCEPlus provider used for these results
- **Significant improvement in SSL overhead with IBMJCEPlus provider (only 14% over no SSL)**

### System Configuration:

- **SUT:** LinTel – SLES 12.3, Intel(R) Xeon(R) Platinum 8180 CPU @ 2.50GHz, 2 physical cores, 64GB RAM.
- IBM JDK 8 SR6 FP7 is used for Open Liberty

## Performance Tip #3: Thread and connection pools

- On Liberty, there is a single executor thread pool for different kinds of work and the default autonomics work very well in most cases
- **On Liberty, the best tip for the thread pool is to avoid tuning it explicitly** but if you still suspect the thread pool, try `<executor maxThreads="X" />` where  $X = \$cpus * 2$ ; if it or another value is better, open a support case

Work	Traditional WAS	Liberty
HTTP	WebContainer	Default Executor
JMS (SIB)	SIBJMSRAThreadPool	Default Executor
JMS (MQ)	WMQJCAResourceAdapter or MessageListenerThreadPool	Default Executor
EJB	ORB.thread.pool	Default Executor
z/OS	WebSphere WLM Dispatch Thread	Default Executor

## Performance Tip #4: Configure smallest possible set of Liberty features

- Open Liberty allows user to configure the server with subset of (JakartaEE/JavaEE/MicroProfile) features as needed by the application
- Convenience features (such as Full Profile or Web Profile) are great to get an application up and running but the features unused by the application can have overheads
  - Startup time (up to 50% for some simple applications)
  - Memory footprint
  - Throughput
- In Open Liberty's features can be configured using the feature manager in server.xml

```
<server>  
  <featureManager>  
    <feature>servlet-4.0</feature>  
    <feature>jdbc-4.3</feature>  
  </featureManager>  
</server>
```

## Performance Tip #5: Ensure operating system not saturated

- CPU: Check (100 - idle%) over ~30 second intervals
- RAM: Check that there is no paging
- Network: Check ingress/egress rates and compare to known maximums
- Disk: Check read/write rates and compare to known maximums

On Linux the tools to use are :

CPU utilization : top, vmstat, nmon, collectl, sar

RAM utilization/Paging rates : free -m, top, vmstat, nmon, sar

Disk and NIC utilization/response times : iostat, nmon, atop, sar

Tune TCP : tcpdump

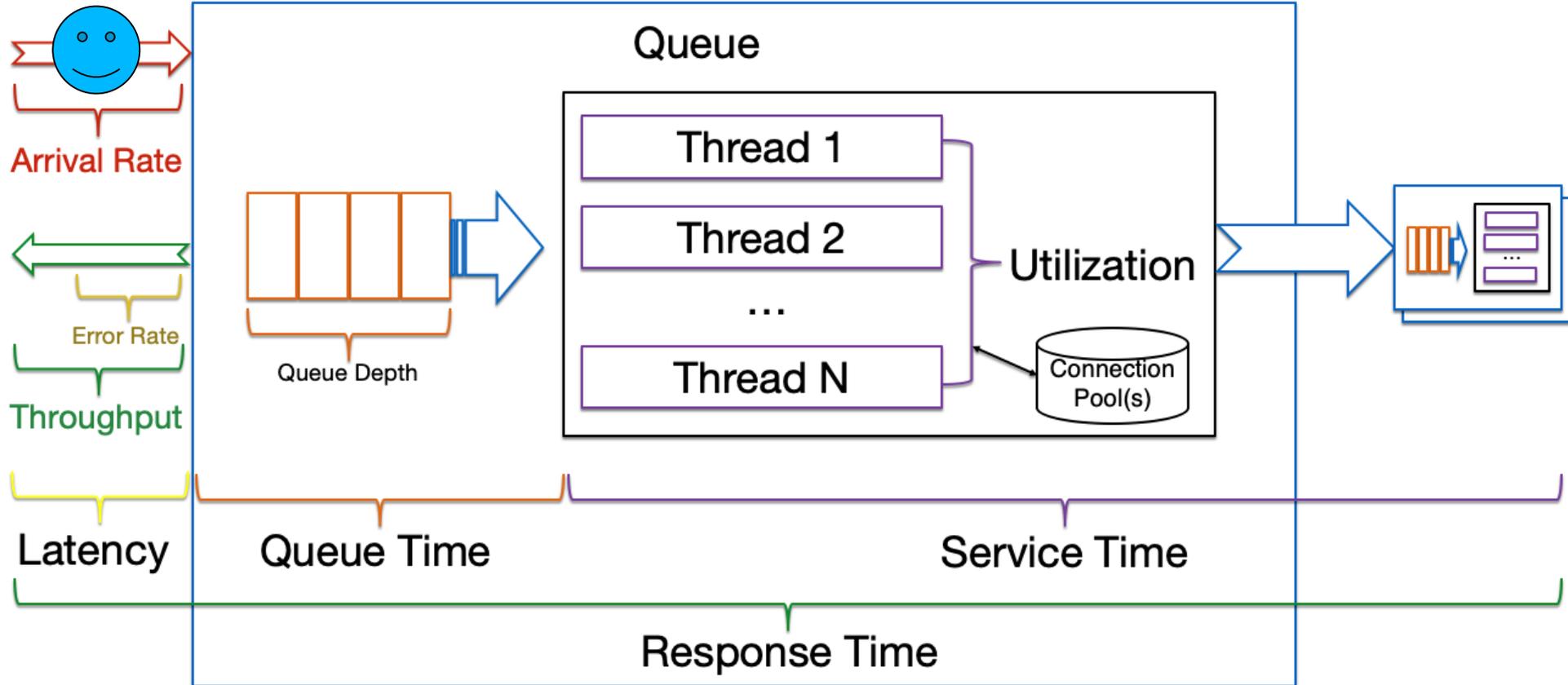
Historical data : sar, nmon, atop

Review OS logs : /var/log

Per process/thread activity : atop, top -H

Virtualization config/stealing/ballooning : vmstat st

# Performance Tip #6: Performance Monitoring



## Performance Tip #6: Performance Monitoring

- Use a 3<sup>rd</sup> party monitoring product or built-in capabilities
- Liberty built-in:
  - [mpMetrics-2.3](#): Expose statistics through a REST endpoint
    - `threadpool.activeThreads`, `connectionpool.(freeConnections,inUseTime.total)`, `servlet.(request.total, responseTime.total)`
  - [monitor-1.0](#): Expose statistics through Java MXBeans
  - [JAX-RS tracing](#) for applications with JAX-RS web services to Jaeger/Zipkin

## Performance Tip #7: Caching

- Many possible places to cache
- The best is on the client since that doesn't use your shared resources
  - HTTP response caching: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching>
- Content Delivery Network (CDN)
- Liberty: [webCache-1.0](#)
- Caching products (e.g. Hazelcast, WebSphere eXtreme Scale, etc.)

## Troubleshooting Tip #1: Thread dumps

- When you have **any** problem, gather multiple thread dumps and basic OS statistics during the issue
- Automated example scripts: “[MustGather: Performance, hang, or high CPU issues](#)”
- After the issue, gather and package all the logs for post-mortem analysis:
  - Liberty: messages\*, console\*, ffdc/\*, verbosegc\*, javacore\*, \*phd, \*dmp, trace\*, \*perf\_RESULTS\*, server.xml, or use server dump
- Automate as much as possible using scripts
- Analyze thread dumps [right on the node](#) or use the free [TMDA tool](#)

# Troubleshooting Tip #1: Thread dumps

- TMDA

**Thread Dump List**

Name	Timestamp	Runnable/
javacore1781806...	...	...
javacore1781806...	...	...
javacore1781806...	...	...

- File name: ...
- Cause of thread dump : signal 3 received

**Monitor Detail : javacore1781806.1281102854.txt**

[TotalSize/Size] ThreadName (ObjectName) 1  
 [1/1] Signal dispatcher  
 Thread-2 (Thread queue lock (0x302710F8))

<b>Thread Name</b>	Thread-2
<b>State</b>	Waiting on condition
<b>Monitor</b>	Waiting for Monitor Lock on Thread queue lock (0x302710F8)
<b>Java Stack</b>	No Java stack trace available
<b>Native Stack</b>	No Native stack trace available

**Compare Threads : javacore1781806.1281102854.txt javacore1781806.1281103199.txt javacore1781806.1281103326.txt**

Thread	javacore17...	javacore17...	javacore17...
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE
WebConta...	X* IDLE	X* IDLE	X* IDLE

<b>Thread Name</b>	WebContainer : 20
<b>State</b>	Waiting on condition
<b>Java Stack</b>	at java.lang.Object.wait(Native Method) at java.lang.Object.wait(Object.java(Compiled Code)) at com.ibm.ws.util.BoundedBuffer.take(BoundedBuffer.java(Compiled Code)) at com.ibm.ws.util.ThreadPool.getTask(ThreadPool.java(Compiled Code))

## Troubleshooting Tip #2: Slow Request Detection

- Liberty: Not enabled by default. Add the feature and configure it:
  - ```
<featureManager>  
  <feature>requestTiming-1.0</feature>  
</featureManager>  
  
<requestTiming  
  slowRequestThreshold="10s"  
  hungRequestThreshold="600s"  
  sampleRate="1"  
</>
```
- More expensive than Traditional WAS but also more powerful. For high volume traffic, increase the sampleRate.
- The hungRequestThreshold automatically takes multiple thread dumps.

## Troubleshooting Tip #2: Slow Request Detection

- Liberty example output:

TRAS0112W: Request AAC9KLwFFXT\_AAAAAAAAAAAN has been running on thread 0000006b for 1549.460ms. The following stack trace shows what this thread is currently doing.

```

${THREAD DUMP}

```

The following table shows the events that have run during this request.

| Duration     | Operation                                                                                                        |
|--------------|------------------------------------------------------------------------------------------------------------------|
| 1552.012ms + | websphere.servlet.service   DayTrader Web   TradeScenarioServlet                                                 |
| 0.014ms      | websphere.session.getAttribute   R-ObCtcDfr8Zd9riQEMCh6R   uidBean                                               |
| 30.714ms     | websphere.servlet.service   DayTrader Web   TradeAppServlet                                                      |
| 0.010ms      | websphere.session.getAttribute   R-ObCtcDfr8Zd9riQEMCh6R   uidBean                                               |
| 30.456ms     | websphere.servlet.service   DayTrader Web   /quote.jsp                                                           |
| 28.903ms     | websphere.servlet.service   DayTrader Web   /displayQuote.jsp                                                    |
| 0.194ms      | websphere.datasource.psExecuteQuery   jdbc/TradeDataSource   SELECT t0.CHANGE1, t0.COMPANYNAME, t0.HIGH,         |
| 1520.695ms + | websphere.servlet.service   DayTrader Web   TradeAppServlet                                                      |
| 0.013ms      | websphere.session.getAttribute   R-ObCtcDfr8Zd9riQEMCh6R   uidBean                                               |
| 522.450ms    | websphere.datasource.psExecuteUpdate   jdbc/TradeDataSource   INSERT INTO orderejb (ORDERID, COMPLETIONDATE, OPE |
| 230.333ms    | websphere.datasource.psExecuteUpdate   jdbc/TradeDataSource   INSERT INTO holdingejb (HOLDINGID, PURCHASEDATE, P |
| 0.936ms      | websphere.datasource.psExecuteUpdate   jdbc/TradeDataSource   UPDATE accountejb SET BALANCE = ? WHERE ACCOUNTID  |

## Troubleshooting Tip #3: Sampling Profiler

- Low overhead sampling profilers give deep insight into what's happening in the JVM
- IBM Java/OpenJ9: Health Center (shipped with the JVM) or other 3<sup>rd</sup> party profilers
  - For example, Health Center shows CPU usage by method, verbosegc, thread stacks, file I/O, socket I/O, etc.
  - `-Xhealthcenter:level=headless`
  - Stop the JVM and load the `.hcd`
- HotSpot: Mission Control (shipped with the JVM) or other 3<sup>rd</sup> party profilers
  - `-XX:+FlightRecorder -XX:StartFlightRecording=name=jfr,filename=recording.jfr,settings=profile -XX:FlightRecorderOptions=defaultrecording=true,dumponexit=true,dumponexitpath=path`
  - Stop the JVM and load the `.jfr`

## Troubleshooting Tip #4: Use OS Core dumps for OOMs

- For IBM Java/OpenJ9, there are two ways to investigate the memory for an `OutOfMemoryError` using the [Memory Analyzer Tool](#): PHDs and OS core dumps.
  - OS core dumps are a superset of PHDs.
- By default, an OS core dump will be produced if ulimits allow. Check core and file ulimits are unlimited.
- For example, core dumps show detailed thread locals:
  - Note that there is potentially sensitive data in a core dump so treat securely.

| Object / Stack Frame                                                         | Retained Heap | Name             |
|------------------------------------------------------------------------------|---------------|------------------|
| <Regex>                                                                      | <Numeric>     | <Regex>          |
| com.ibm.ws.util.ThreadPool\$Worker @ 0x4202f20                               | 5,474,256     | WebContainer : 2 |
| at com.ibm.ws.rsadapter.jdbc.WSJdbcObject.close()V (WSJdbcObject)            |               |                  |
| > <local> com.ibm.ws.rsadapter.jdbc.WSJdbcResultSet @ 0xff397f0              | 80            |                  |
| > <local> java.lang.String @ 0xebbeb48 select * from test.table1             | 24            |                  |
| > <local> com.ibm.DatabaseTest @ 0xf1716f8                                   | 96            |                  |
| > <local> com.mysql.jdbc.ResultSetMetaData @ 0x107d2948                      | 24            |                  |
| > <local> java.lang.OutOfMemoryError @ 0x4258d30 Thread                      | 288           |                  |
| > <local> java.lang.String @ 0x107d2970 test                                 | 40            |                  |
| > <local> com.ibm.ws.rsadapter.jdbc.WSJdbcResultSet @ 0xff397f0              | 80            |                  |
| > <local> com.ibm.ws.rsadapter.jdbc.WSJdbcStatement @ 0xf175e2f              | 112           |                  |
| > <local> java.util.ArrayList @ 0xf175b28                                    | 5,378,120     |                  |
| > <local> com.ibm.ws.rsadapter.jdbc.WSJdbcConnection @ 0xf175b28             | 232           |                  |
| Σ Total: 10 entries                                                          |               |                  |
| > at com.ibm.jsp._dbtest._jspService(Ljavax/servlet/http/HttpServletRequest) |               |                  |
| > at com.ibm.ws.jsp.runtime.HttpJspBase.service(Ljavax/servlet/http/HttpS    |               |                  |

## Troubleshooting Tip #5: Consider using the NCSA Access Log

- If using HTTP, the NCSA access log writes a line to a file for each HTTP response including details such as the response time
- Test the performance overhead as it may be up to 2-3%
- [Enable on tWAS](#): \$SERVER > Web Container Settings > Web container transport chains > WCInbound\* > HTTP inbound channel
- Enable on Liberty:
  - ```
<httpEndpoint host="*" httpPort="9080" httpsPort="9443">  
  <accessLogging filepath="{server.output.dir}/logs/access.log" maxFileSize="250"  
  maxFiles="4" logFormat="%h %i %u %t &quot;%r&quot; %s %b %D %{R}W" />  
</httpEndpoint>
```

## Key online resources for Liberty performance tuning

- Links:

- [WebSphere Application Server Performance Cookbook](#)
- [Troubleshoot WebSphere Application Server \(The Basics\)](#)
- [Self-paced WebSphere Application Server Troubleshooting and Performance Lab](#)

# Recent Updates



# Periodic Table of Liberty (20.0.0.9)

Open Liberty



zOS

ND

Base

Core

Open Liberty

New in 4Q19

New in 3Q20

New in 2Q20

New in 1Q20

batchSMFLogging-1.0		zosLocalAdapters-1.0		zosTransaction-1.0		zosSecurity-1.0	
collectiveController-1.0		dynamicRouting-1.0		healthManager-1.0		scalingController-1.0	
clusterMember-1.0		healthAnalyzer-1.0		scalingMember-1.0		Security	
cloudant-1.0	heritageAPIs-1.0	batchManagement-1.0		Operations		passwordUtilities-1.0	
javaee-7.0	sipServlet-1.0	wsAtomicTransaction-1.2				wsSecurity-1.1	
javaee-8.0						wsSecuritySaml-1.0	
jakartaee-8.0							
bells-1.0	mpContextPropagation-1.0	adminCenter-1.0		constrainedDelegation-1.0		audit-1.0	
concurrent-1.0	mpGraphQL-1.0	collectiveMember-1.0		federatedRepository-1.0		ldapRegistry-3.0	
javaMail-1.6	mpReactiveMessaging-1.0	distributedMap-1.0		jwt-1.0		oauth-2.0	
jaxb-2.2	mpReactiveStreams-1.0	eventLogging-1.0		jwtSso-1.0		openid-2.0	
jdbc-4.3	opentracing-1.3	logstashCollector-1.0		sessionDatabase-1.0		openidConnectClient-1.0	
jpaContainer-2.2	osgiConsole-1.0	monitor-1.0		webCache-1.0		openidConnectServer-1.0	
jsfContainer-2.3	springBoot-2.0	openapi-3.1				samlWeb-2.0	
json-1.0	webProfile-7.0	requestTiming-1.0				scim-1.0	
jsonbContainer-1.0	webProfile-8.0	usageMetering-1.0				socialLogin-1.0	
jsonpContainer-1.1		restConnector-2.0				spnego-1.0	
microProfile-3.3	APIs	sessionCache-1.0				transportSecurity-1.0	

# Focus areas



Developer Experience

APIs

Foundation

Orchestration

Security

# Liberty Last Quarter Review



## Security

- Disable LTPA generation when using alternative SSO

## Dev Exp

- Eclipse MicroProfile Tools support for VS Code
- Open Liberty tools for IntelliJ
- Guide - Using a REST service from ReactJS
- Guide – Distributed trace with Jaeger
- Guide – Authentication using social media login

## Foundation

- JAX-RS 2.1 clients now support dynamic SSL config changes
- HTTP/2 performance improvements
- featureUtility improvements

## API

- SmallRye GraphQL client

## Orchestration

- HTTP access logs data in json format logs
- JSON written to logs no longer wrapped

# Disable LTPA Cookie



- When using TAI or SPNEGO Liberty issues LTPA token
  - This may be unnecessary

```
<server>  
  <trustAssociation disableLtpaCookie="true" />  
</server>
```

```
<server>  
  <spnego disableLtpaCookie="true" />  
</server>
```

# HTTP Access fields in JSON logs



```
<httpEndpoint id="defaultHttpEndpoint" httpPort="9080"
              httpsPort="9443" host="*">
  <accessLogging logFormat='%R{W} %u %{my_cookie}C %s' />
</httpEndpoint>
<logging messageFormat="json" messageSource="message,accessLog"
          jsonAccessLogFields="logFormat" />
```

```
{
  "type": "liberty_accesslog",
  "host": "192.168.1.15",
  "ibm_userDir": "/opt/wlp/usr/",
  "ibm_serverName": "defaultServer",
  "ibm_cookie_my_cookie": "example_cookie",
  "ibm_responseCode": 200,
  "ibm_datetime": "2020-06-18T09:30:47.693-0400",
  "ibm_sequence": "1592487047653_00000000000001"
}
```



# MicroProfile GraphQL UI



- Write and execute GraphQL queries
- Syntax highlighting
- Query completion
- Real time error highlighting

<http://localhost:9080/myGraphQLApp/graphql-ui>

The screenshot shows the GraphQL UI interface. On the left, a query is written with syntax highlighting: `query listMovies { allMovies { title director { name } releaseDate } }`. On the right, the JSON response is displayed, showing a list of movies with their titles, directors, and release dates. The response is: `{ "data": { "allMovies": [ { "title": "Independence Day", "director": { "name": "Roland Emmerich" }, "releaseDate": "1996-07-03" }, { "title": "Men In Black", "director": { "name": "Barry Sonnenfeld" }, "releaseDate": "1997-07-02" }, { "title": "Earth Girls Are Easy", "director": { "name": "Julien Temple" }, "releaseDate": "1989-05-12" }, { "title": "Spaceballs", "director": { "name": "Mel Brooks" }, "releaseDate": "1987-06-24" } ] } }`

```
<variable name="io.openliberty.enableGraphQLUI" value="true" />
```

# Liberty Plugin for IntelliJ

Open Liberty



ServletSample - HelloServlet.java

```
22
23
24 @WebServlet(urlPatterns="/servlet")
25 public class HelloServlet extends HttpServlet {
26     private static final long serialVersionUID = 1L;
27
28     // tag::javadoo1[]
29     /**
30      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
31      */
32     // end::javadoo1[]
33     protected void doGet(HttpServletRequest request, HttpServletResponse response)
34         throws ServletException, IOException {
35         // tag::responseString[]
36         response.getWriter().append("Hello! How are you today?\n");
37         // end::responseString[]
38     }
39 }
```

Liberty Dev Dashboard

- Start
- Start...
- Stop
- Run Tests
- View integration test report
- View unit test report

Terminal: Local x ServletSample x +

```
[INFO] [AUDIT ] CWWK0016I: Web application available (default_host): http://192.168.1.223:9080/ServletSample/
[INFO] [AUDIT ] CWWK2001I: Application ServletSample started in 0.609 seconds.
[INFO] [AUDIT ] CWWK0012I: The server installed the following features: [servlet-4.0].
[INFO] [AUDIT ] CWWK0011I: The guideServer server is ready to run a smarter planet. The guideServer server started in 3.902 seconds.
[INFO] CWWK0015I: Match number: 1 is [9/29/20 15:47:52:209 EDT] 0000037 com.ibm.ws.kernel.feature.internal.FeatureManager A CWWK0011I: The guideServer server is ready to r
un a smarter planet. The guideServer server started in 3.902 seconds.
[INFO] Press the Enter key to run tests on demand. To stop the server and quit dev mode, use Ctrl-C or type 'q' and press the Enter key.
[INFO] Source compilation was successful.
[INFO] Tests compilation was successful.
[INFO] [AUDIT ] CWWK0017I: Web application removed (default_host): http://192.168.1.223:9080/ServletSample/
[INFO] [AUDIT ] CWWK2009I: The application ServletSample has stopped successfully.
[INFO] [AUDIT ] CWWK0016I: Web application available (default_host): http://192.168.1.223:9080/ServletSample/
[INFO] [AUDIT ] CWWK2003I: The application ServletSample updated in 0.070 seconds.
```

# Liberty Plugin for VS Code



```
src > main > java > io > openliberty > sample > system > SystemResource.java > SystemResource.java
9  import javax.ws.rs.core.MediaType;
10
11  import org.eclipse.microprofile.metrics.annotation.Counted;
12  import org.eclipse.microprofile.metrics.annotation.Timed;
13
14  @RequestScoped
15  @Path("/properties")
16  public class SystemResource {
17
18      @GET
19      @Produces(MediaType.APPLICATION_JSON)
20      @Timed(name = "getPropertiesTime", description = "Time needed to get the
21      @Counted(absolute = true, description = "Number of times the JVM system p
```

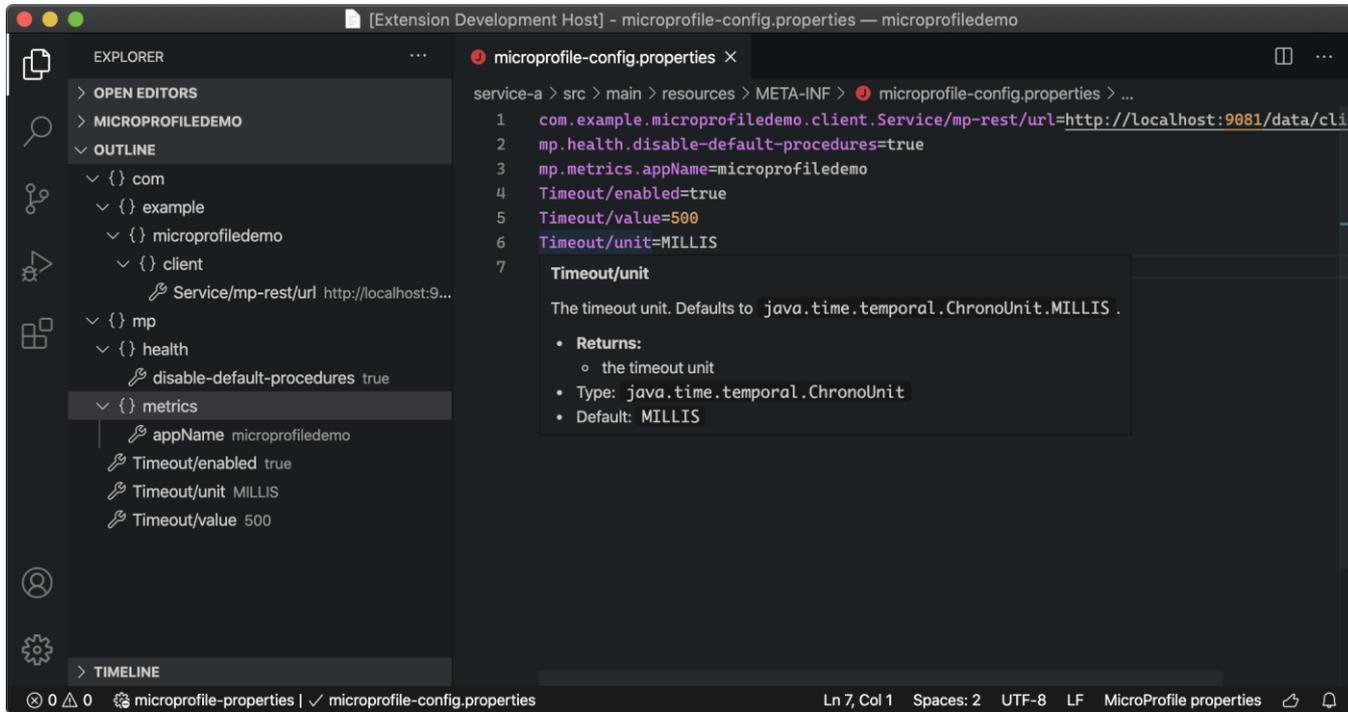
demo-devmode-maven ( 2: demo-devmode-maven ( + [ ] [ ] ^ x

- Start
- Start...
- Stop
- Run tests
- View integration test report
- View unit test report

```
ng integration tests...
-----
S T S
-----
ng it.io.openliberty.sample.PropertiesEndpointIT
run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.647 s - in
it.io.openliberty.sample.PropertiesEndpointIT
[INFO]
[INFO] Results:
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] Integration tests finished.
```

# MicroProfile Plugin for VS Code

Open Liberty



EXPLORER

- OPEN EDITORS
- MICROPROFILEDEMO
- OUTLINE
  - com
    - example
      - microprofiledemo
        - client
          - Service/mp-rest/url http://localhost:9...
  - mp
    - health
      - disable-default-procedures true
    - metrics
      - appName microprofiledemo
      - Timeout/enabled true
      - Timeout/unit MILLIS
      - Timeout/value 500

TIMELINE

microprofile-properties | ✓ microprofile-config.properties

Ln 7, Col 1 Spaces: 2 UTF-8 LF MicroProfile properties

```
service-a > src > main > resources > META-INF > microprofile-config.properties > ...
1 com.example.microprofiledemo.client.Service/mp-rest/url=http://localhost:9081/data/cli
2 mp.health.disable-default-procedures=true
3 mp.metrics.appName=microprofiledemo
4 Timeout/enabled=true
5 Timeout/value=500
6 Timeout/unit=MILLIS
7
```

**Timeout/unit**

The timeout unit. Defaults to `java.time.temporal.ChronoUnit.MILLIS`.

- Returns:
  - the timeout unit
- Type: `java.time.temporal.ChronoUnit`
- Default: `MILLIS`

# Labs, Questions

# WebSphere Liberty Virtual POT



Download the operating system specific content zip file from either

<https://ibm.box.com/WASLibertyVPoT> (fast - about 10 minute download)

<https://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/wasdev/pot/> (slower but firewall friendly – about 1 hour download)

 Liberty Quarterly Update\_20.0.0.4-6.final.pdf

 LibertyPoT\_20.0.0.6\_WIN.zip V2

 LibertyPoT\_20.0.0.6\_MAC.zip V2

 LibertyPoT\_20.0.0.6\_LINUX.zip V2

 labs\_n\_presentations\_only.zip V2

 LibertyResourceList.pdf V2

charts only

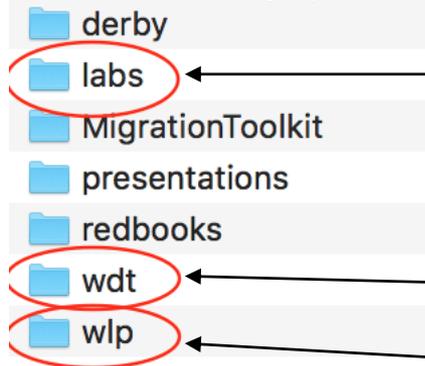
charts & lab instructions  
only

# WebSphere Liberty Virtual POT



Unzip to C:\wlp\_pot

- Note: You can unzip to anywhere you wish but the lab instructions assume the unzip location is C:\wlp\_pot



All labs in here

Eclipse IDE with Liberty tools

Liberty runtime and Java SDK

Follow [labs\gettingStarted\0\\_setup\\_20180105\setup.pdf](#)

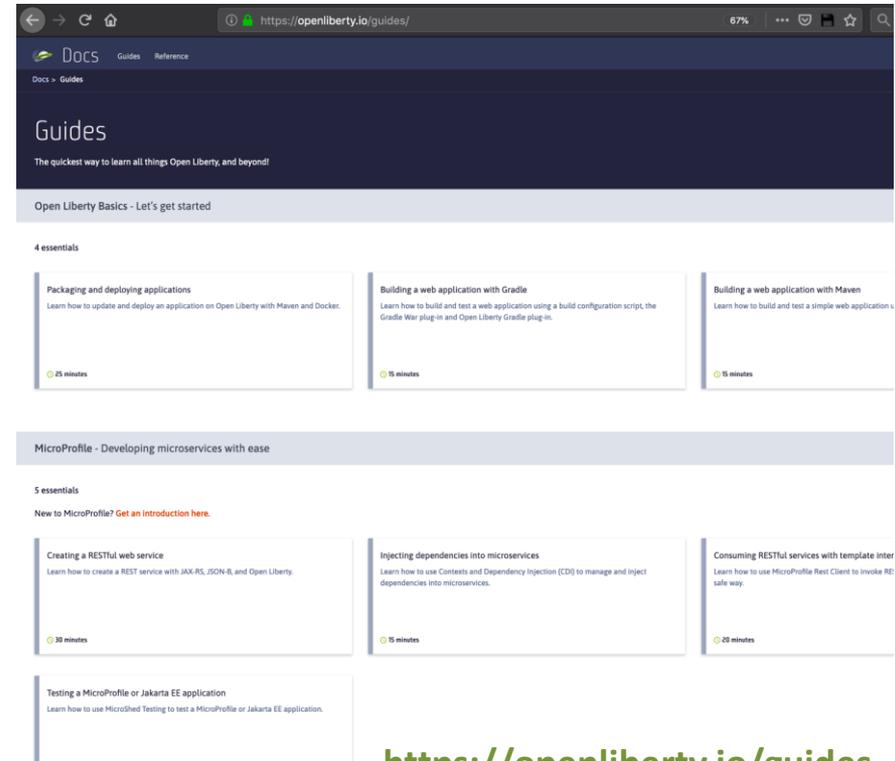
Then choose any labs you want to do

# Open Liberty Guides

Open Liberty



- Hands-on learning in ~20 minutes
- 52 guides
  - MicroProfile & Jakarta EE
  - Open Shift, Docker, Kubernetes Istio
- Latest Guides
  - *Consuming a RESTful web service with ReactJS*
  - *Authenticating users through social media providers*
  - *Enabling distributed tracing in microservices with Jaeger*



<https://openliberty.io/guides>

# Resources



- Open Liberty: <https://www.openliberty.io/>
- Eclipse MicroProfile: <https://microprofile.io/>
- Jakarta EE: <https://jakarta.ee/>
- Liberty advantage: <https://www.ibm.com/downloads/cas/NVY3KY4E>
- Open Liberty Guides: <https://openliberty.io/guides>
- Why Liberty is the best Java runtime for the Cloud <https://developer.ibm.com/wasdev/docs/liberty-profile-best-java-runtime-cloud/>
- WebSphere Application Server V8.5 Administration and Configuration Guide for Liberty Profile (Redbook)  
<http://www.redbooks.ibm.com/abstracts/sg248170.html?Open>
- Liberty videos: [https://www.ibm.com/support/knowledgecenter/SSAW57\\_liberty/com.ibm.websphere.wlp.nd.multiplatform.doc/ae/covr\\_media.html](https://www.ibm.com/support/knowledgecenter/SSAW57_liberty/com.ibm.websphere.wlp.nd.multiplatform.doc/ae/covr_media.html)
- Java support dates <http://www.ibm.com/developerworks/java/jdk/lifecycle>
- Single Stream Continuous Delivery <https://www-01.ibm.com/support/docview.wss?uid=ibm10869798>
- WebSphere Migration Knowledge Collection: Planning and Resources <https://www-01.ibm.com/support/docview.wss?uid=swg27008724>
- IBM Transformation Advisor <http://ibm.biz/cloudta>
- WebSphere Binary Migration Toolkit: <http://ibm.biz/WAMT4AppBinaries>

# Next Quarterly Update

Open Liberty



## Liberty 20.0.0.7-9 Update

Session#1: ~~September 30, 2020 from 1-3pm ET - <http://ibm.biz/Liberty-Sep30>~~

Session#2: October 14, 2020 from 9-11am ET - <http://ibm.biz/Liberty-Oct14>

## Liberty 20.0.0.10-12 Update

Session#1: January 13, 2020 from 1-3pm ET - <http://ibm.biz/Liberty-Jan13>

Session#2: January 20, 2020 from 9-11am ET - <http://ibm.biz/Liberty-Jan20>

# WebSphere & Cloud Pak for Apps Customer Advisory Board

All Customers and Business  
Partners welcome

<http://ibm.biz/WebSphereAdvisoryBoard>

email: [claudiab@us.ibm.com](mailto:claudiab@us.ibm.com)

## OPEN invitation

Join 230+ other  
members

Recordings/charts:

[ibm.biz/WASCABCommunityResources](http://ibm.biz/WASCABCommunityResources)

**NEW**

Monthly  
sessions for  
Business  
Partners and in  
the IST timezone

### Possible engagement levels – no commitment needed:

1. Fly on the wall – NEW\*\*
2. Stay ahead of the curve: more time commitment
3. Close the gap: quarterly involvement
4. At your own pace: impact longer term goals

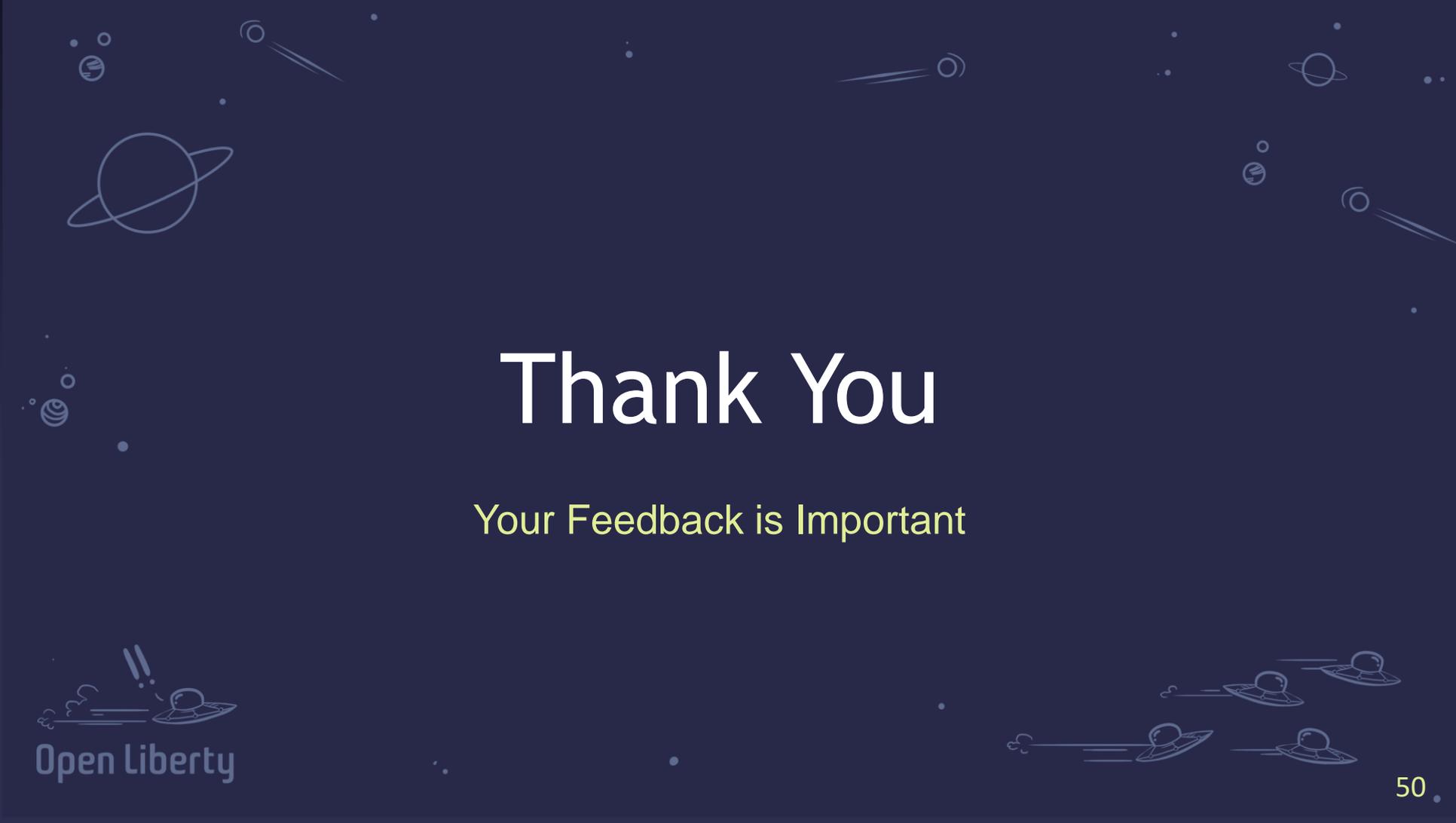
### What you get out of it:

- ✓ **Direct access** to architects, developers, team leads during sessions
- ✓ **Insight** into roadmaps, pre-announce insider tips (under Feedback agreement)
- ✓ Directly **influence** deliverables
- ✓ **Gain** insights from other customers
- ✓ **Bonus:** special sessions at conferences
- ✓ **Bonus:** Free Cloud assessment

# Questions?

<http://stackoverflow.com/questions/tagged/websphere-liberty>  
alasdair@ibm.com





# Thank You

Your Feedback is Important

