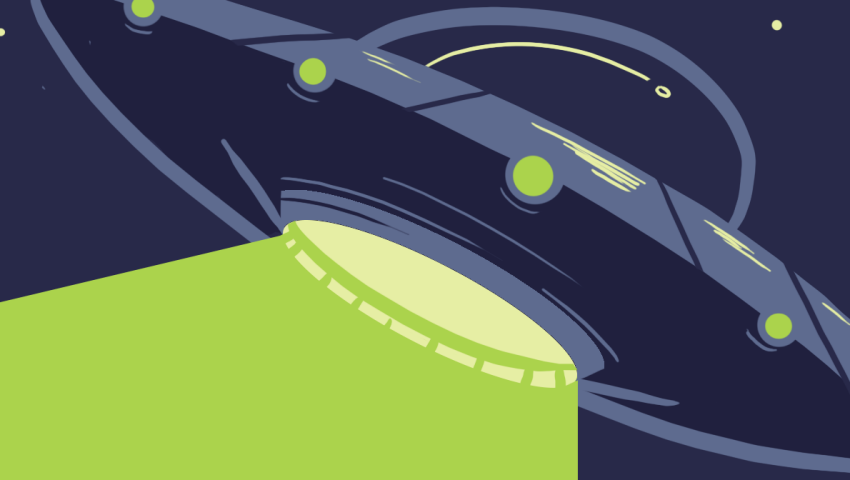
Open Liberty



Masterclass: Hands-on Liberty for Developers

Graham Charters, STSM, Dev Advocacy Lead

@gcharters

Alasdair Nottingham, STSM, Liberty Lead Architect

@nottycode

Mastersclass hands-on content

- The Application
- Module 1: Build
- Module 2: Feature-based Build
- Module 3: Application APIs
- Module 4: Server Configuration
- Module 5: Externalizing Configuration
- Module 6: Integration Testing
- Module 7: Docker
  - Overriding Dev Server Configuration
- Module 8: Testing in Containers
- Module 9: Support Licensing
- Conclusion

Brief History

Open Liberty

Liberty Goals

Open Liberty

- Efficient
- Simple to use
- Consistency
- Just enough runtime
- End of migration
- Agile-ready

Liberty

- First shipped in WAS 8.5 in 2012
  - Servlet + JSP + JPA
- Web Profile 6 in 2014
- Java EE 7 in 2016 – first commercial product to certify
- Java EE 8 in 2018 – first to certify
- Jakarta EE 8 in 2019 – first to certify
- Eclipse MicroProfile – first to deliver 1.0-1.4, 2.0-2.1, 3.0

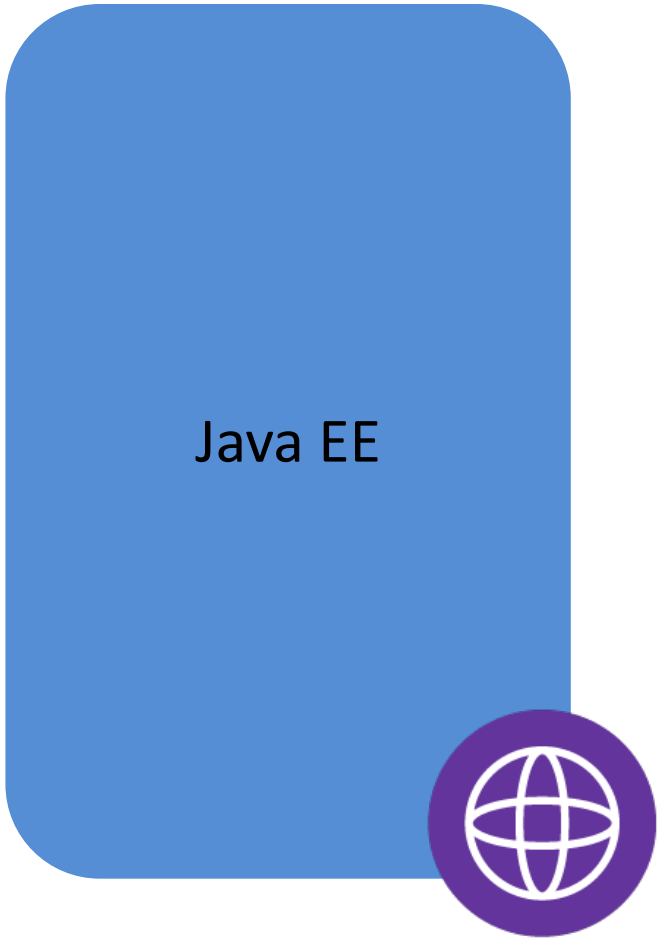

Docs
Get Started
Support
Fork the code
openliberty.io
Launched September 2017
Open Liberty
An IBM Open Source Project
A lightweight open framework for building fast and efficient cloud-native Java microservices.
Build cloud-native apps and microservices while running only what you need. Open Liberty is the most flexible server runtime available to Java™ developers in this solar system.
Get Open Liberty

Liberty Architecture

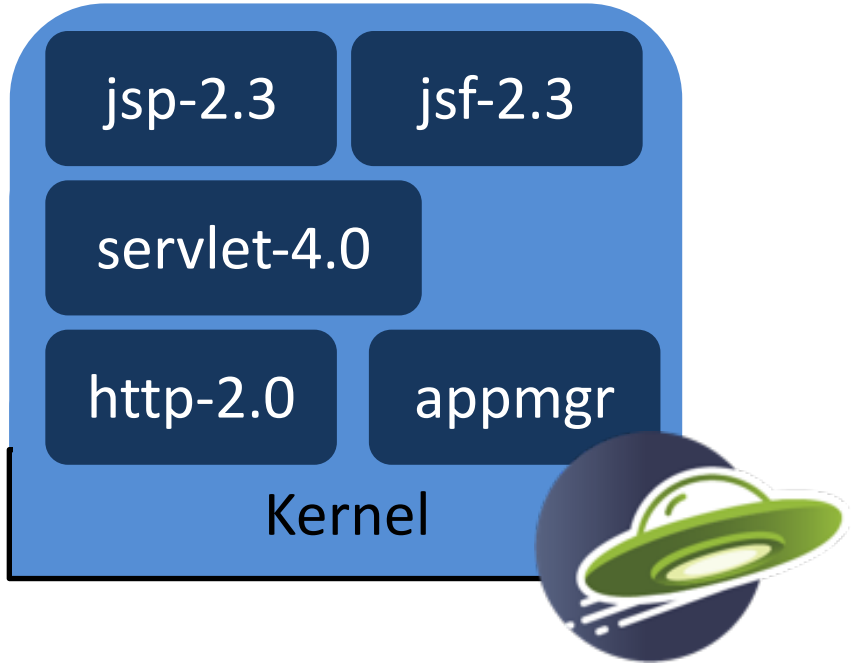
Fit for purpose

Open Liberty

- You control which features are loaded into each server instance




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



Liberty Zero Migration

Open Liberty

- Zero config migration
  - Write once, run forever
- Zero migration for apps
  - No behavior changes in existing features
  - New behaviors in new features
- Choose your Java
  - Java 12, 11, 8
  - AdoptOpenJDK
  - IBM
  - OpenJDK
  - Oracle

Features

Open Liberty

Periodic Table of Liberty (19.0.0.8)

Open Liberty

zOS

ND

Base

Core

Open Liberty

New in 3Q19

New in 2Q19

New in 1Q19

New in 4Q18

batchSMFLogging-1.0
zosLocalAdapters-1.0
zosRequestLogging-1.0
zosTransaction-1.0
zosWlm-1.0
zosSecurity-1.0

collectiveController-1.0
healthAnalyzer-1.0
scalingController-1.0

clusterMember-1.0
dynamicRouting-1.0
healthManager-1.0
scalingMember-1.0

Security

cloudant-1.0
javaee-7.0
javaee-8.0
sipServlet-1.0
batchManagement-1.0
wsAtomicTransaction-1.2

Operations

passwordUtilities-1.0
wsSecurity-1.1
wsSecuritySaml-1.0

bells-1.0
concurrent-1.0
javaMail-1.6
jaxb-2.2
jdbc-4.3
jpaContainer-2.2
jsfContainer-2.3
json-1.0
jsonbContainer-1.0
jsonpContainer-1.1
microProfile-3.0

mpContextPropagation-1.0
mpReactiveStreams-1.0
opentracing-1.3
osgiConsole-1.0
springBoot-2.0
webProfile-7.0
webProfile-8.0

APIs

adminCenter-1.0
collectiveMember-1.0
distributedMap-1.0
eventLogging-1.0
logstashCollector-1.0
monitor-1.0
openapi-3.1
requestTiming-1.0
usageMetering-1.0
restConnector-2.0
sessionCache-1.0

constrainedDelegation-1.0
federatedRepository-1.0
jwt-1.0
jwtSso-1.0
sessionDatabase-1.0
webCache-1.0

audit-1.0
ldapRegistry-3.0
oauth-2.0
openid-2.0
openidConnectClient-1.0
openidConnectServer-1.0
samlWeb-2.0
scim-1.0
socialLogin-1.0
spnego-1.0
transportSecurity-1.0

Periodic Table of Liberty (19.0.0.8)

Open Liberty

zOS

ND

Base

Core

Open Liberty

New in 3Q19

New in 2Q19

New in 1Q19

New in 4Q18

batchSMFLogging-1.0

collectiveController-1.0

clusterMember-1.0

dynamicRouting-1.0

cloudant-1.0

javaee-7.0

javaee-8.0

sipServlet-1.0

bells-1.0

concurrent-1.0

javaMail-1.6

jaxb-2.2

jdbc-4.3

jpaContainer-2.2

jsfContainer-2.3

json-1.0

jsonbContainer-1.0

jsonpContainer-1.1

microProfile-3.0

mpContextPropagation-

mpReactiveStreams-1.0

opentracing-1.3

osgiConsole-1.0

springBoot-2.0

webProfile-7.0

webProfile-8.0

APIs

appClientSupport-1.0
appSecurityClient-1.0
batch-1.0
concurrent-1.0
ejb-3.2
ejbHome-3.2
ejbPersistentTimer-3.2
ejbRemote-3.2
j2eeManagement-1.1
javaMail-1.6
jacc-1.5
jaxb-2.2
jaxws-2.2
jca-1.7
jms-2.0
managedBeans-1.0
mdb-3.2
wasJmsClient-2.0
webProfile-8.0
wmqJmsClient-2.0

appSecurity-3.0
beanValidation-2.0
cdi-2.0
ejbLite-3.2
el-3.0
jaspic-1.1
jaxrs-2.1
jaxrsClient-2.1
jdbc-4.2
jndi-1.0
jpa-2.2
jsonb-1.0
jsonp-1.1
jsf-2.3
jsp-2.3
managedBeans-1.0
servlet-4.0
ssl-1.0
websocket-1.1

cdi-2.0
jaxrs-2.1
jsonp-1.1
jsonb-1.0
mpConfig-1.3
mpFaultTolerance-2.0
mpMetrics-2.0
mpJwt-1.1
mpOpenAPI-1.1
mpOpenTracing-1.3
mpRestClient-1.3
mpHealth-2.0

Periodic Table of Liberty (19.0.0.8)

Open Liberty

zOS

ND

Base

Core

Open Liberty

New in 3Q19

New in 2Q19

New in 1Q19

New in 4Q18

batchSMFLogging-1.0

collectiveController-1.0

clusterMember-1.0

dynamicRouting-1.0

cloudant-1.0

javaee-7.0

javaee-8.0

sipServlet-1.0

bells-1.0

concurrent-1.0

javaMail-1.6

jaxb-2.2

jdbc-4.3

jpaContainer-2.2

jsfContainer-2.3

json-1.0

jsonbContainer-1.0

jsonpContainer-1.1

microProfile-3.0

mpContextPropagation-

mpReactiveStreams-1.0

opentracing-1.3

osgiConsole-1.0

springBoot-2.0

webProfile-7.0

webProfile-8.0

APIs

appClientSupport-1.0 ejbPersistentTimer-3.2 jaspic-1.1 managedBeans-1.0
batch-1.0 ejbRemote-3.2 jaxb-2.2 mdb-3.2
concurrent-1.0 j2eeManagement-1.1 jaxws-2.2 wasJmsClient-2.0
ejb-3.2 javaMail-1.5 jca-1.7 webProfile-7.0
ejbHome-3.2 jacc-1.5 jms-2.0 wmqJmsClient-2.0

appSecurity-2.0 jaxrsClient-2.0 jsp-2.3
beanValidation-1.1 jdbc-4.2 managedBeans-1.0
cdi-1.2 jndi-1.0 servlet-3.1
ejbLite-3.2 jpa-2.1 ssl-1.0
el-3.0 jsonp-1.0 websocket-1.1
jaxrs-2.0 jsf-2.2

cdi-2.0 jsonb-1.0 mpMetrics-2.0 mpOpenTracing-1.3
jaxrs-2.1 mpConfig-1.3 mpJwt-1.1 mpRestClient-1.3
jsonp-1.1 mpFaultTolerance-2.0 mpOpenAPI-1.1 mpHealth-2.0

Build

Module 1

Support for Maven and Gradle

- Manage full server and application lifecycle through the two most popular build technologies.
- Maven
  - liberty-maven-plugin
  - liberty-maven-app-parent
  - liberty-archtetype-*
  - Docs & Source: https://github.com/OpenLiberty/ci.maven
- Gradle
  - liberty-gradle-plugin
  - Docs & Source: https://github.com/OpenLiberty/ci.gradle
- All the artefacts you need in Maven Central
  - https://search.maven.org/search?q=io.openliberty
  - https://search.maven.org/search?q=com.ibm.websphere

Feature-based Build

Module 2

Open Liberty

Maven Composed Liberty

```
project
xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchem
a-instance"
xsi:schemaLocation="http://maven.apache.org/
POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>test</groupId>
    <artifactId>serviceA</artifactId>
    <packaging>war</packaging>
    <version>1.0-SNAPSHOT</version>
```

depends on

Maven Repository

Open Liberty

commons

javaee-api

wlp

serviceA.war

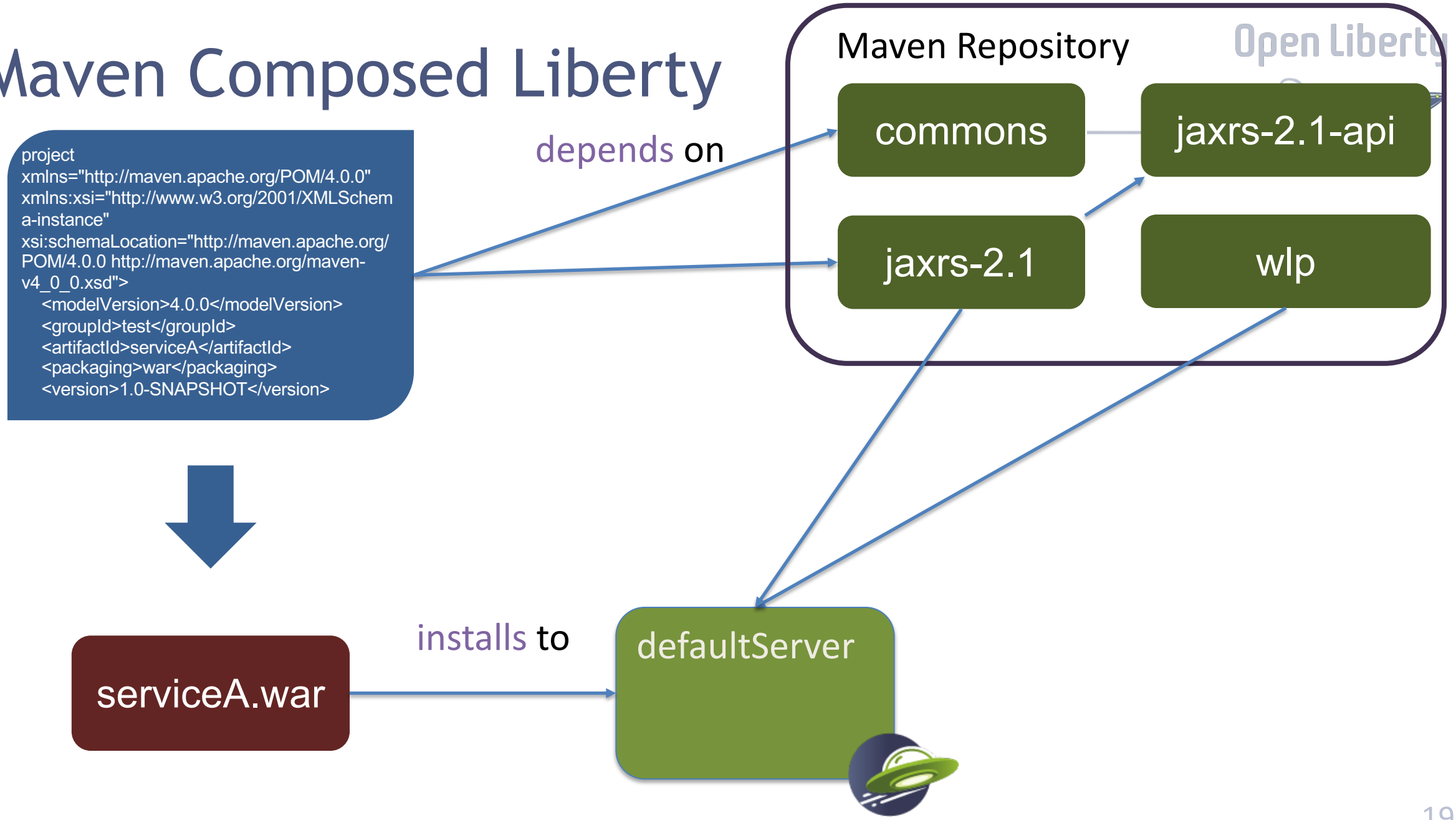
installs to

defaultServer

Liberty Repository

jaxrs-2.1

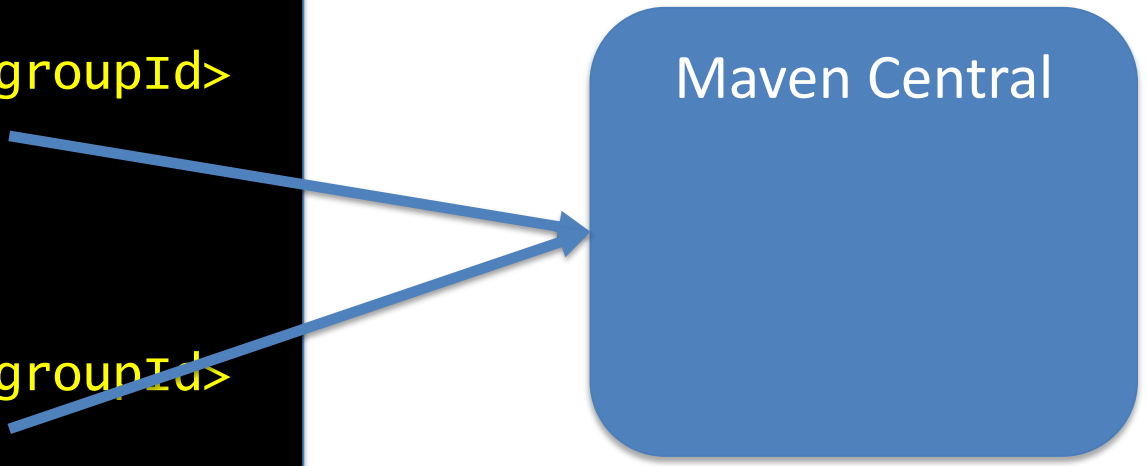
Maven Composed Liberty



Maven/Gradle

Open Liberty

```
<dependencies>
  <dependency>
    <groupId>io.openliberty.features</groupId>
    <artifactId>jaxrs-2.1</artifactId>
    <type>esa</type>
  </dependency>
  <dependency>
    <groupId>io.openliberty.features</groupId>
    <artifactId>jsonb-1.0</artifactId>
    <type>esa</type>
  </dependency>
</dependencies>
```



Application APIs

Module 3

Open Liberty

Java EE 7

appClientSupport-1.0
batch-1.0
concurrent-1.0
ejb-3.2
ejbHome-3.2
ejbPersistentTimer-3.2
ejbRemote-3.2
j2eeManagement-1.1
javaMail-1.5
jacc-1.5
jaspic-1.1
jaxb-2.2
jaxws-2.2
jca-1.7
jms-2.0
managedBeans-1.0
mdb-3.2
wasJmsClient-2.0
webProfile-7.0
wmqJmsClient-2.0

appSecurity-2.0
beanValidation-1.1
cdi-1.2
ejbLite-3.2
el-3.0
jaxrs-2.0
jaxrsClient-2.0
jdbc-4.2
jndi-1.0
jpa-2.1
jsonp-1.0
jsf-2.2
jsp-2.3
managedBeans-1.0
servlet-3.1
ssl-1.0
websocket-1.1

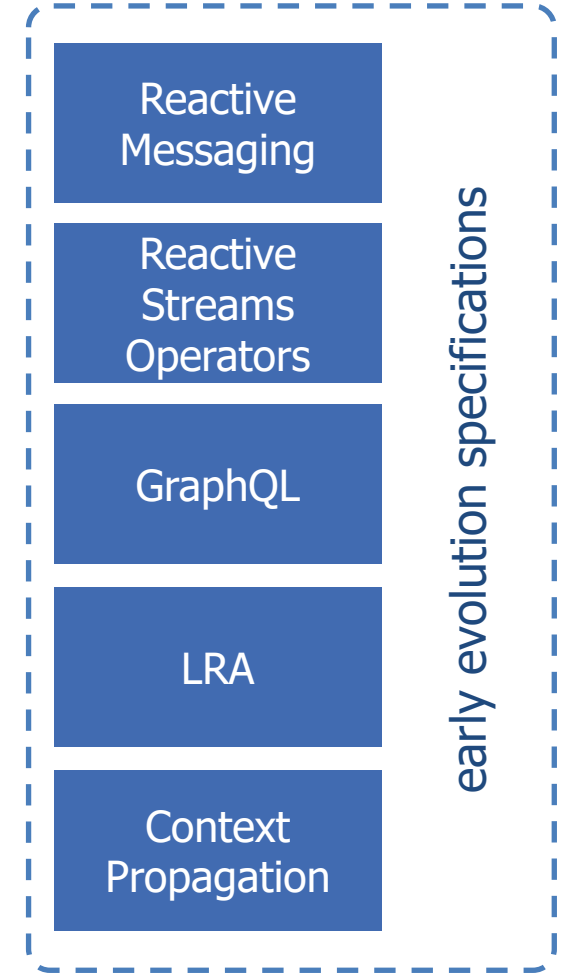
Java EE 8

appClientSupport-1.0
appSecurityClient-1.0
batch-1.0
concurrent-1.0
ejb-3.2
ejbHome-3.2
ejbPersistentTimer-3.2
ejbRemote-3.2
j2eeManagement-1.1
javaMail-1.6
jacc-1.5
jaxb-2.2
jaxws-2.2
jca-1.7
jms-2.0
managedBeans-1.0
mdb-3.2
wasJmsClient-2.0
webProfile-8.0
wmqJmsClient-2.0

appSecurity-3.0
beanValidation-2.0
cdi-2.0
ejbLite-3.2
el-3.0
jaspic-1.1
jaxrs-2.1
jaxrsClient-2.1
jdbc-4.2
jndi-1.0
jpa-2.2
jsonb-1.0
jsonp-1.1
jsf-2.3
jsp-2.3
managedBeans-1.0
servlet-4.0
ssl-1.0
websocket-1.1

Eclipse MicroProfile

Open Liberty



Eclipse MicroProfile

- Builds on Java EE
- By the Java EE community
- Open Source at Eclipse
- Multiple Implementations

```
@Path(“/”)
public class RestEE {

  @GET
  @Counter
  @Traced
  public String hello() {
    return "Hello MicroProfile!!";
  }
}
```

Server Configuration

Module 4

Open Liberty

Simple Config

Open Liberty

```
<server>
  <featureManager>
    <feature>jsp-2.3</feature>
  </featureManager>

  <webApplication location="myweb.war" contextRoot="/" />

  <applicationManager autoExpand="true"/>
</server>
```

server.xml

```
-Xmx1g
-Dsystem.prop=value
```

jvm.options

```
WLP_OUTPUT_DIR=/usr/wlp-out/
```

server.env

27

Composing Config


```
<server>
  <httpEndpoint id="defaultHttpEndpoint" host="${host}"
                httpPort="${http}"
                httpsPort="${https}"/>
</server>
```

configDropins/defaults/common-http.xml

```
<server>
  <include location="https://myHost/ports.xml"/>
  <variable name="host" value="${my.host}"/>
  <variable name="http" value="${my.host.http}"/>
  <variable name="https" value="${my.host.https}"/>
</server>
```

configDropins/overrides/ports.xml

28

Externalizing Configuration

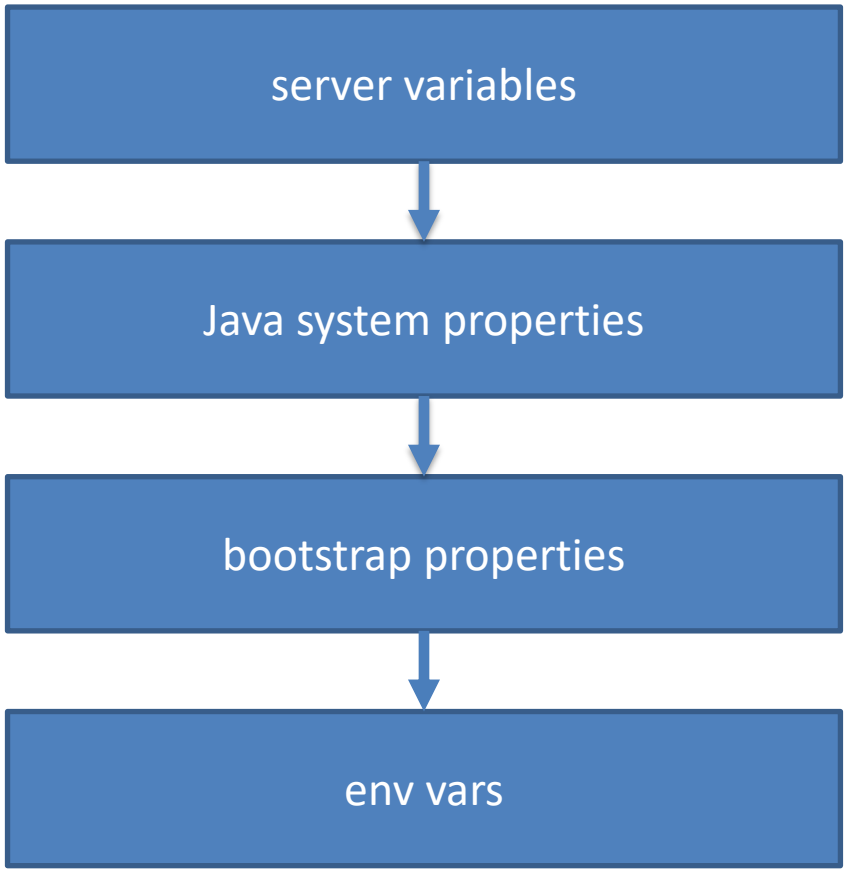
Module 5

Open Liberty

Variable Resolution

Open Liberty

- Liberty variable resolution happens top to bottom



Externalizing Config


```
<server>
  <httpEndpoint id="defaultHttpEndpoint" host="*"
                httpPort="${env.SERVER_HTTP_PORT}"
                httpsPort="${env.SERVER_HTTPS_PORT}"/>
</server>
```

configDropins/defaults/common-http.xml

31

Eclipse MicroProfile

- Inject Configuration
- Sourced from
  - Properties file
  - Java system properties
  - Environment

```
@Path(“/”)
public class RestEE {

  @Inject
  @ConfigProperty(“name”)
  private String name;

  @GET
  @Counter
  @Traced
  public String hello() {
    return "Hello " + name;
  }
}
```

Testing

Module 6

Open Liberty

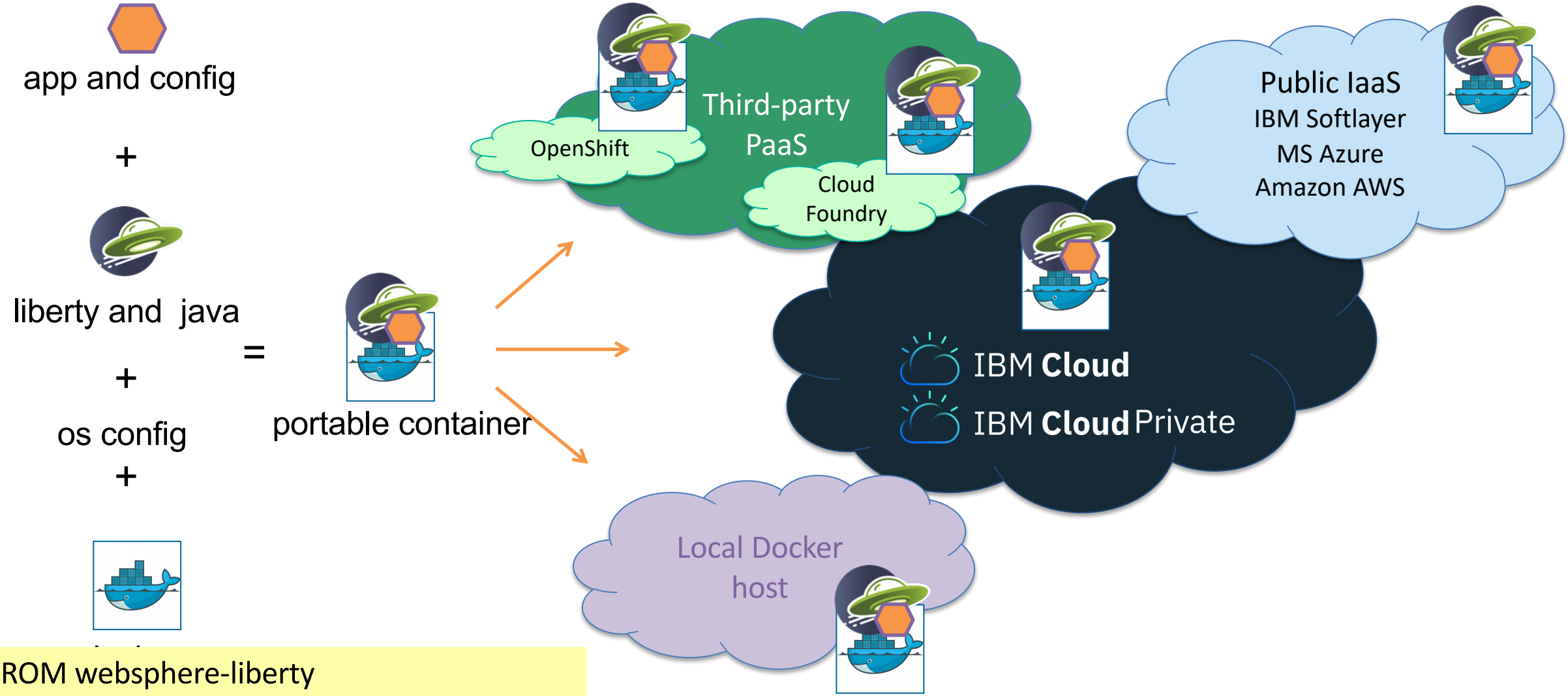
Testing

- Use maven-surefire-plugin to run unit test
  - Use CDI dependency injection to make your code is easy to mock & unit test
- Use maven-failsafe-plugin to run integration test
  - liberty-maven-app-parent predefined executions help
- Use Liberty Arquillian integration to run tests on a Liberty server
- Use MicroShed Testing for system testing in Containers (Module 8)

Docker

Module 7

Liberty in Docker




```
FROM websphere-liberty
ADD myapp.war /config/dropins/myapp.war
```

Making the most of Docker

Testing in Containers

Module 8

Open Liberty

MicroShed Testing

Open Liberty

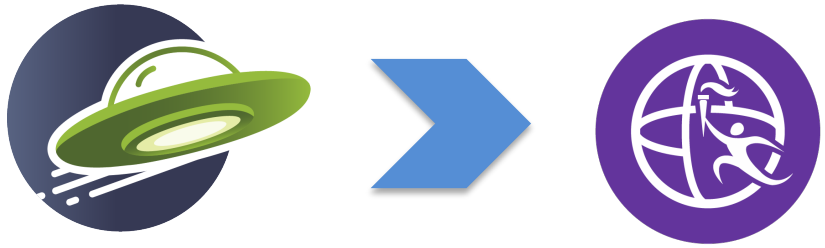
- Integration tests that are easy to setup, write, and run
- Test your apps the same way they run in production...in Containers
- Can run multiple containers on same network (e.g. test DB integration)
- https://microshed.github.io/

```
@MicroShedTest
public class MyTest {

    // Search for Dockerfile.
    // Start app in Container.
    // Wait for Container before running tests.
    @Container
    public static MicroProfileApplication app
        = new MicroProfileApplication()
                .withAppContextRoot("/myservice");

    // Inject JAX-RS REST Client
    @Inject
    public static MyService mySvc;

    // A test method like any other
    @Test
    public void testMyService() {
        ...
    }
}
```

Support Licensing

Module 9

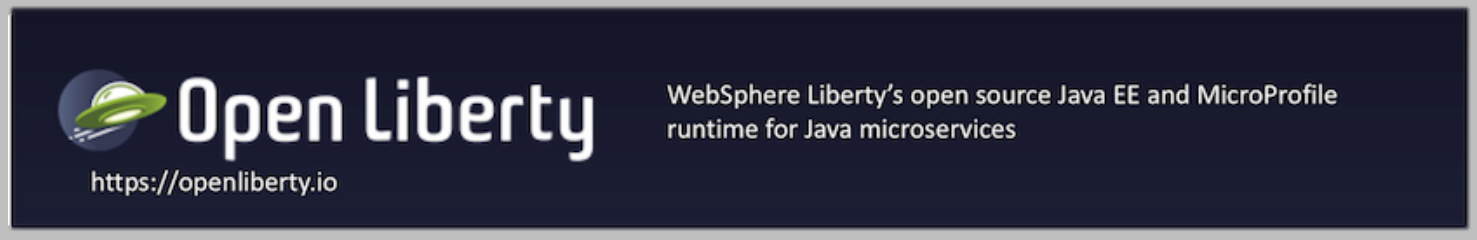
Buy WebSphere Liberty



	Monthly VPC (List Price)
WebSphere ND	US$331
WebSphere	US$88.50
Liberty Core	US$43.70

Open Liberty Support

Support for Open Liberty *and* OpenJDK on Eclipse OpenJ9 JVM



Buy support online on a per virtual-processor-core basis as your need arises.

Price: **$750.00** USD

Continue to Checkout

Support for Open Liberty includes:

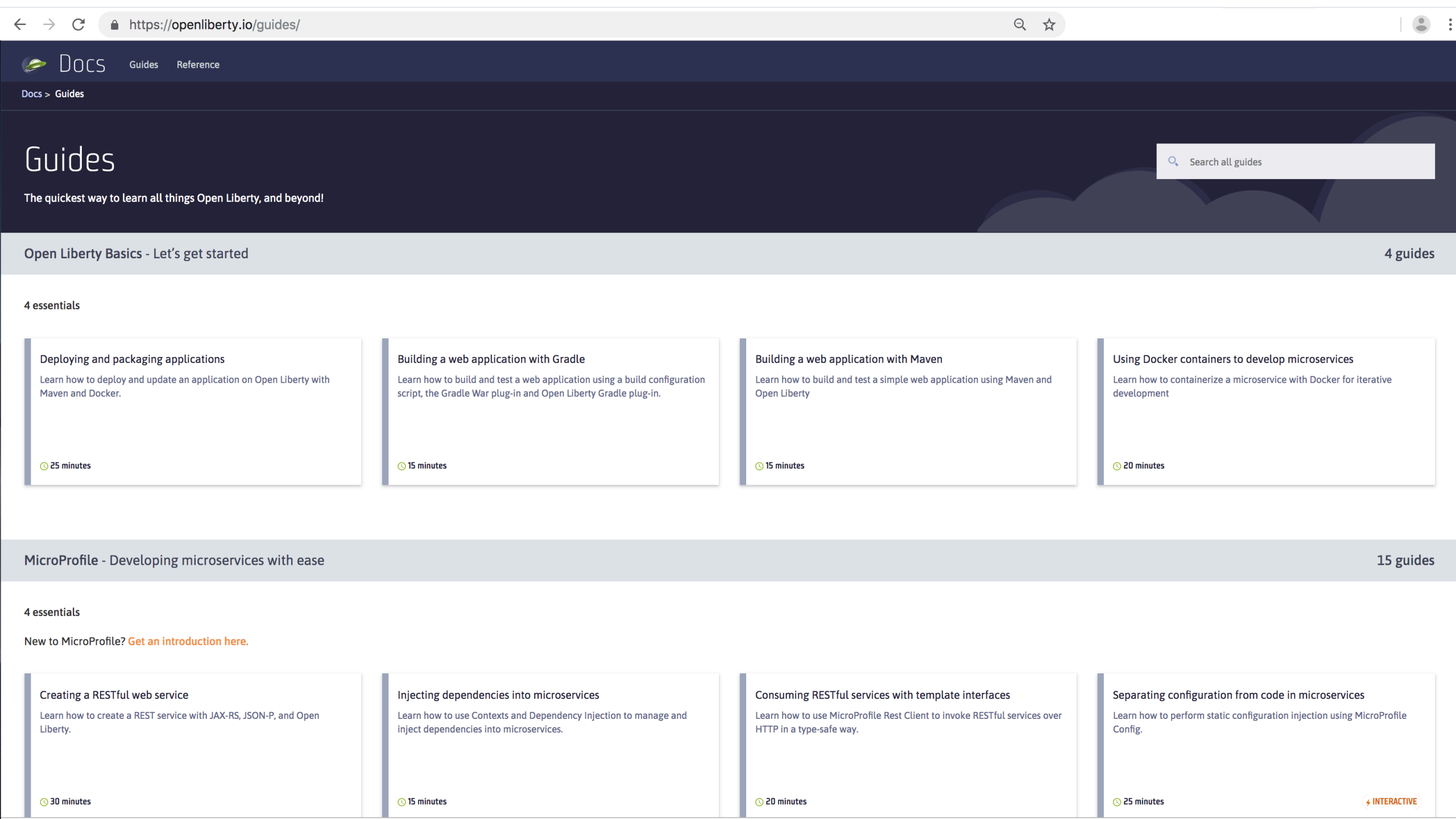
- Unlimited number of support tickets
  - Standard support hours are 8am – 5pm Mon – Fri, or 24 x 7 x 364 for Severity level 1 (everyday except Dec 25th)
- Immediate security fixes (PSIRT)
- Ticket prioritization with a 2-hour response time

Once you buy support there is no update or change required to your app already running in production

Wrap-up

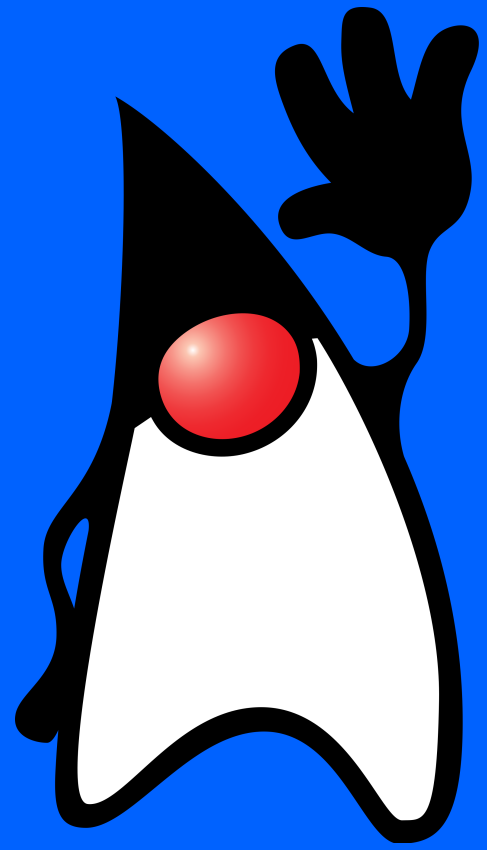
Mastersclass hands-on content

- Open Liberty Masterclass
  - Table of Contents
  - Before you begin
    - Install Pre-requisites
    - Prime Maven and Docker Caches
  - The Application
  - Module 1: Build
  - Module 2: Feature-based Build
  - Module 3: Application APIs
  - Module 4: Server Configuration
  - Module 5: Externalizing Configuration
  - Module 6: Integration Testing
  - Module 7: Docker
    - Overriding Dev Server Configuration
  - Module 8: Support Licensing
  - Conclusion


https://openliberty.io/guides/
Docs
Guides
Reference
Docs > Guides
Guides
The quickest way to learn all things Open Liberty, and beyond!
Search all guides
Open Liberty Basics - Let's get started
4 guides
4 essentials
Deploying and packaging applications
Learn how to deploy and update an application on Open Liberty with Maven and Docker.
25 minutes
Building a web application with Gradle
Learn how to build and test a web application using a build configuration script, the Gradle War plug-in and Open Liberty Gradle plug-in.
15 minutes
Building a web application with Maven
Learn how to build and test a simple web application using Maven and Open Liberty
15 minutes
Using Docker containers to develop microservices
Learn how to containerize a microservice with Docker for iterative development
20 minutes
MicroProfile - Developing microservices with ease
15 guides
4 essentials
New to MicroProfile? Get an introduction here.
Creating a RESTful web service
Learn how to create a REST service with JAX-RS, JSON-P, and Open Liberty.
30 minutes
Injecting dependencies into microservices
Learn how to use Contexts and Dependency Injection to manage and inject dependencies into microservices.
15 minutes
Consuming RESTful services with template interfaces
Learn how to use MicroProfile Rest Client to invoke RESTful services over HTTP in a type-safe way.
20 minutes
Separating configuration from code in microservices
Learn how to perform static configuration injection using MicroProfile Config.
25 minutes
INTERACTIVE

https://cloud.ibm.com/docs/java

IBM **Cloud**



http://ibm.biz/javanewsletter

WebSphere Customer Advisory Board – open invitation

email: claudiab@us.ibm.com

Sign up now at Think

Influence deliverables

Choose your engagement level:

1. **Stay ahead of the curve**: more time commitment
2. **Close the gap**: quarterly involvement
3. **At your own pace**: impact longer term goals

Get involved.
Be successful.

WebSphere Customer Advisory Board - Engagement levels

Engagement level	Comm frequency	User group studies and testing	Contribute Code	Design discussions	Prototype feedback, pre-release code	Create joint assets (webcasts, quotes), roadmaps	Meetings, conferences, Think 2020
Stay ahead of the curve	Monthly	✓	✓	✓	✓	✓	✓
Close the gap	Quarterly			✓	✓	✓	✓
At your own pace	2 times a year or more as requested				✓	✓	✓

* subject to change based on input

WebSphere Customer Advisory Board – Open invitation

Do you get timely and relevant information from WebSphere?

- What do you find most important? (Webcasts, Proof of Technology, Newsletters, conferences)

Join our WebSphere Customer Advisory Board – New for 2019

Share your modernization and Cloud Journey

- Share experience at private and public forums
- Join us for Webcast, meet ups
- Joint articles, blogs

Become our trusted advisors – help us prioritize on next-gen features

- Design Thinking - help improve user experience before we write the first line of code
- Get involved and be successful

email: claudiab@us.ibm.com