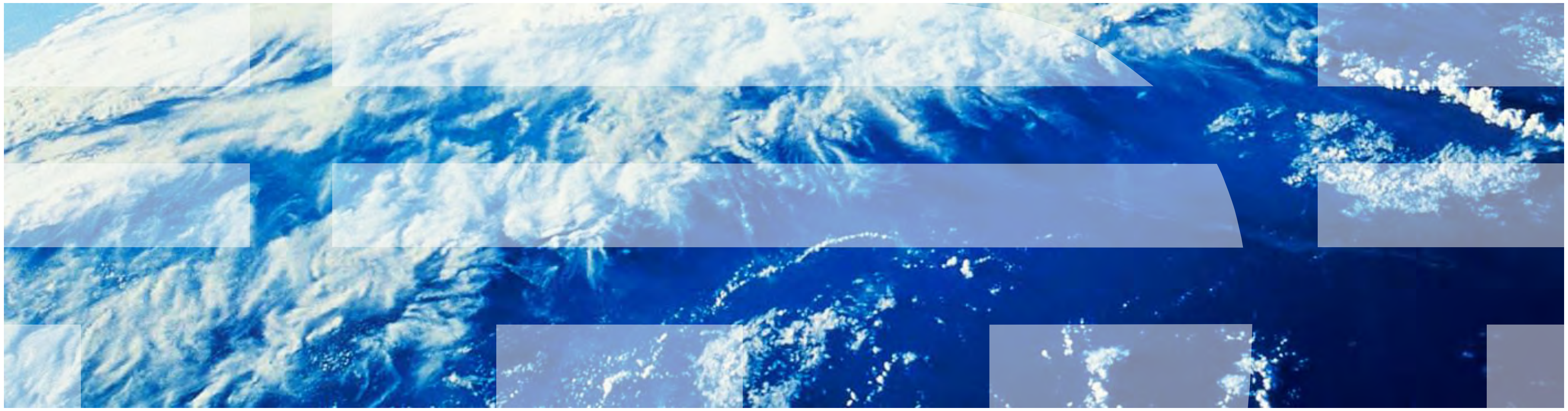
Eclipse MicroProfile: Accelerating the adoption of Java Microservices

Emily Jiang – twitter @emilyfhjiang

10th October 2017



What is Eclipse MicroProfile?

- Eclipse MicroProfile is an open-source community specification for Cloud Native Java microservices
- A community of individuals, organizations, and vendors collaborating within an open source (Eclipse) project to bring microservices to the Enterprise Java community

Community - individuals, organizations, vendors

payara FUJITSU Tomitribe join the tribe

IBM redhat

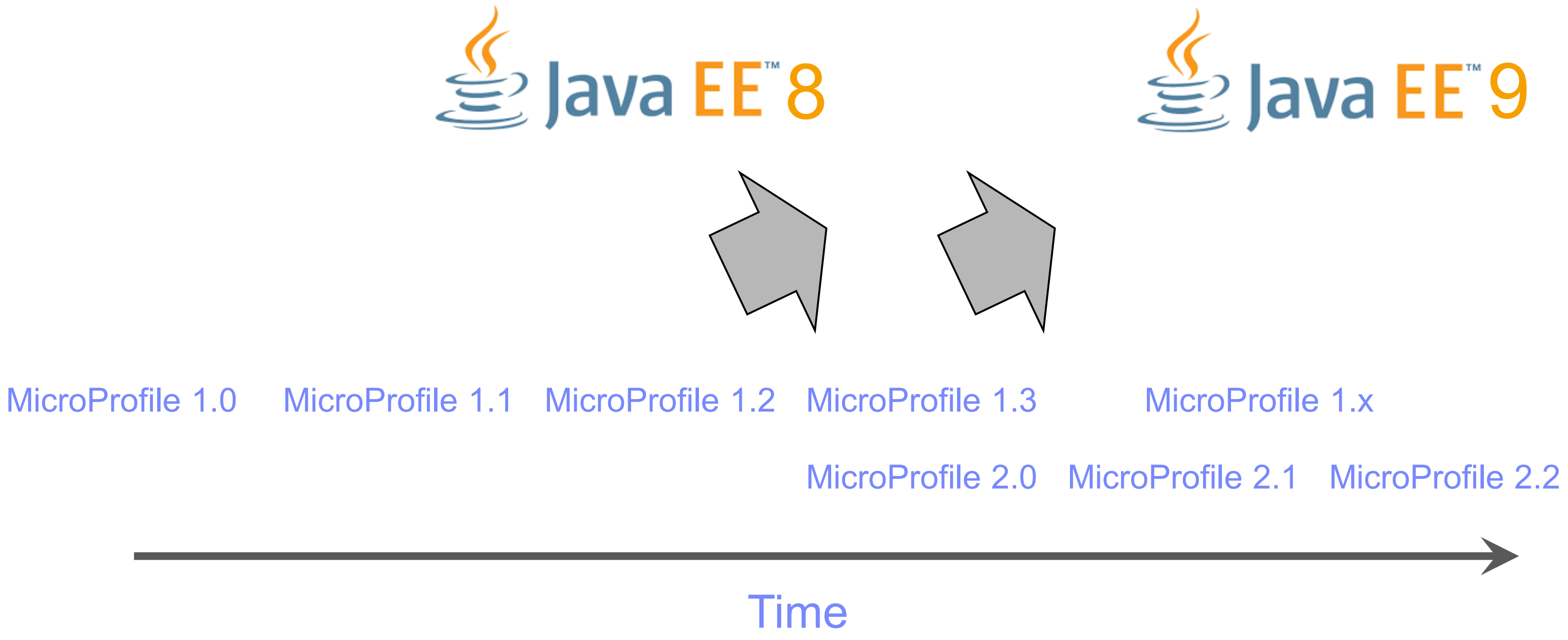
LJC HAMMOCK SOUJava sociedade de usuários java

SMARTBEAR hazelcast **And Growing …**

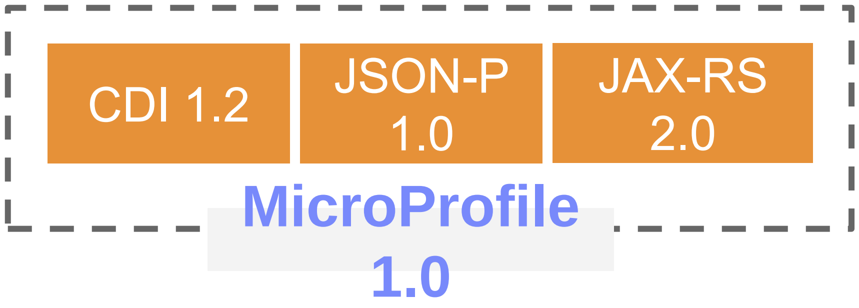
Innovation vs. Standardization

Eclipse MicroProfile	Java Community Process
(Open Source) Project	Standards organization
Incremental feature release	Large multi-feature releases
Community controls pace	Spec Lead controls pace

Accelerating Adoption of Microservices



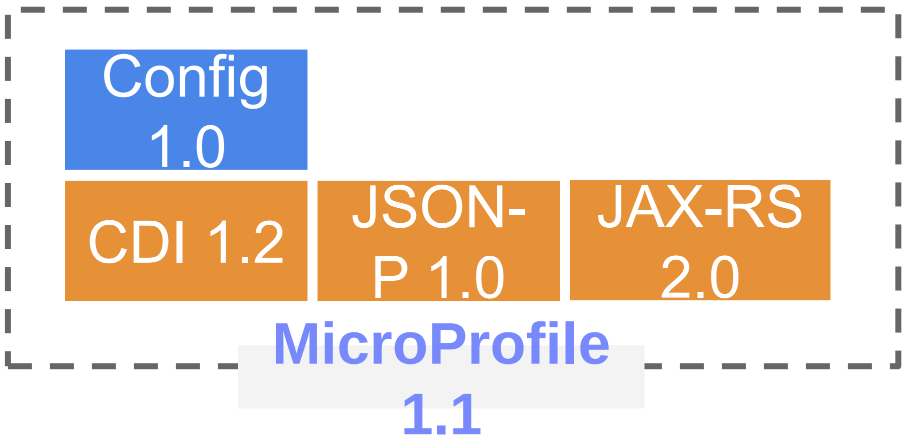
MicroProfile 1.0 (Sep, 2016)



Announced on
8th August

MICROPROFILE™ OPTIMIZING ENTERPRISE JAVA 1.1

Eclipse MicroProfile 1.1 (August, 2017)



🟦 = New

🟧 = No change from last release

Config 1.0

- Why?
  - – Configure Microservice without repacking the application
- How?
  - – Specify the configuration in configure sources
  - – Access configuration via
    - Programmatically lookup

```
Config config = ConfigProvider.getConfig();
config.getValue("myProp", String.class);
```

    - Via CDI Injection

```
@Inject @ConfigProperty(name="my.string.property") String myPropV;
```

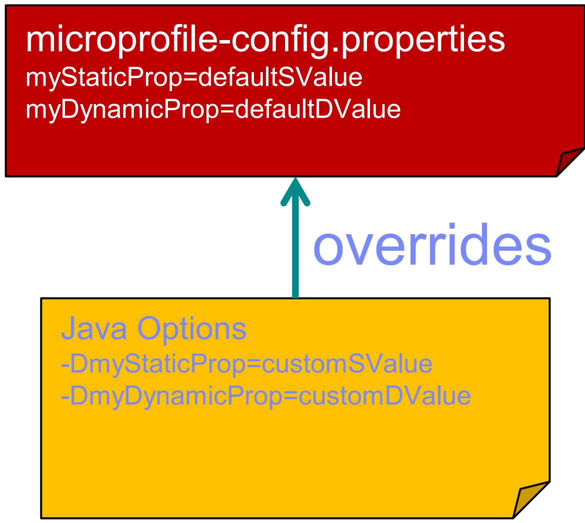
MicroProfile Config

- Static Config

```
@Inject
@ConfigProperty(name="myStaticProp")
private String staticProp;
```

- Dynamic Config

```
@Inject
@ConfigProperty(name="myDynamicProp")
private Provider<String> dynamicProp;
```

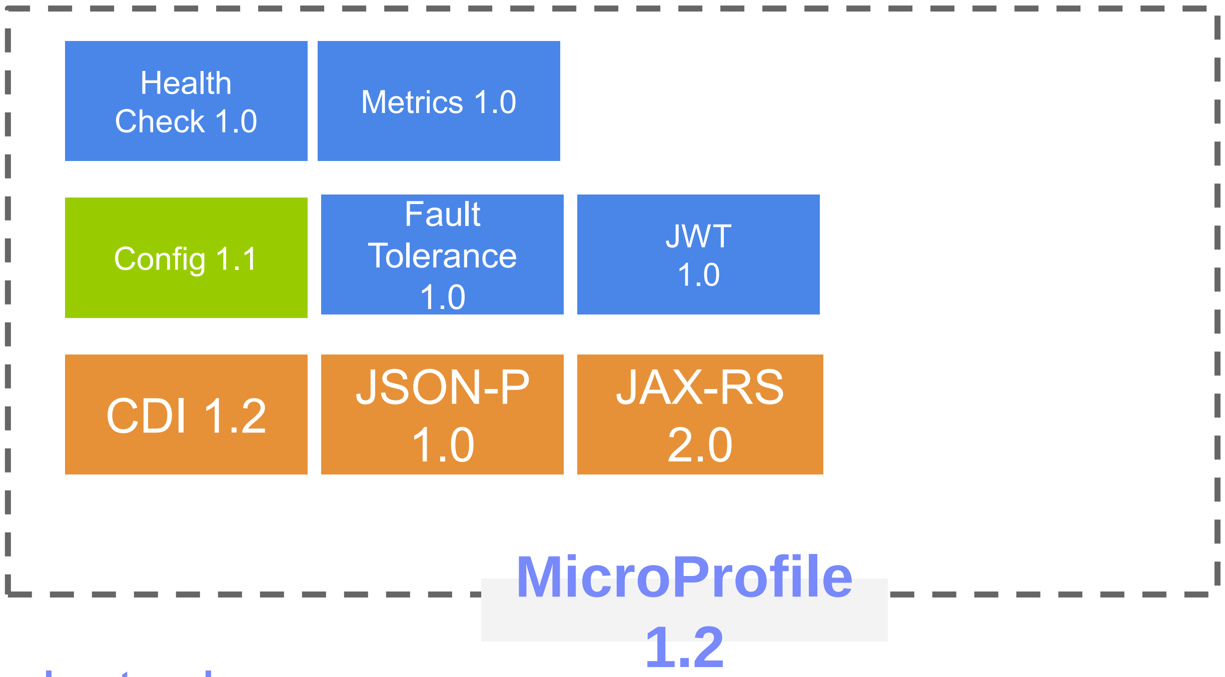

microprofile-config.properties

```
myStaticProp=defaultSValue
myDynamicProp=defaultDValue
```

Java Options

```
-DmyStaticProp=customSValue
-DmyDynamicProp=customDValue
```

Proposed Eclipse MicroProfile 1.2 (Q3 CY2017)



🟦 = New

🟩 = Update from last release

🟧 = No change from last release

Robust Microservices

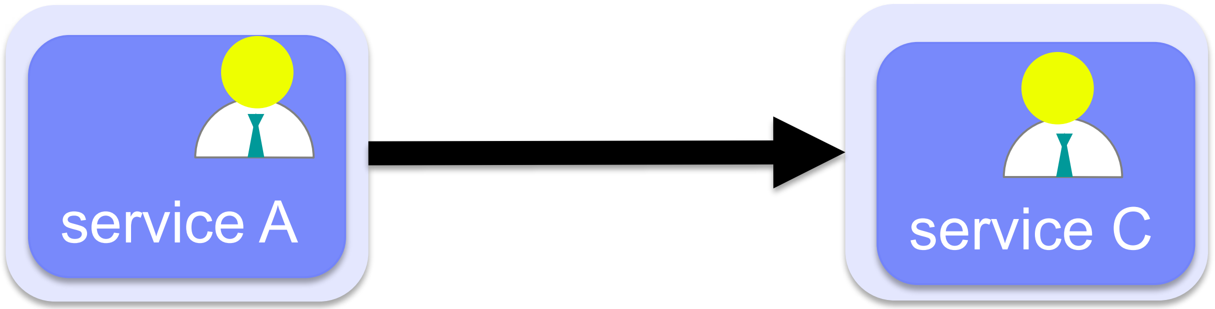
http://microprofile.io/

MicroProfile adds new enterprise Java capabilities for microservices

Config	Fault Tolerance	Health Check	Health Metrics	JWT
externalize configuration to improve portability	build robust behavior to cope with unexpected failures	ensure services are running	understand the interactions between services while running	resolve problems in complex distributed systems

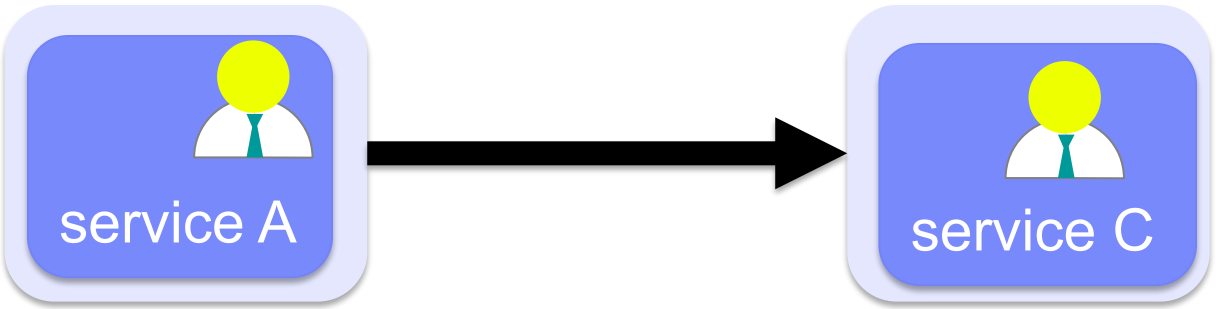
New in Eclipse MicroProfile Release 1.2: https://projects.eclipse.org/projects/technology.microprofile/releases/1.2

Transient Failure



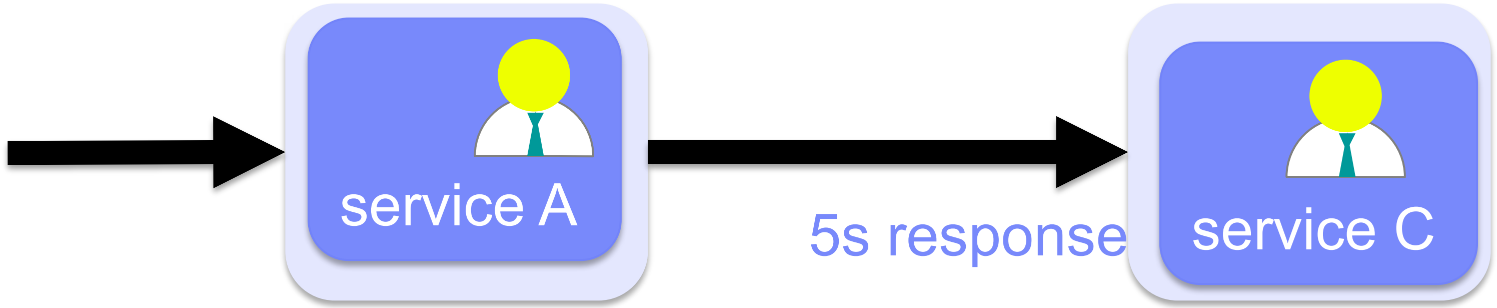
```
for (int i = 0; i < 5; i++) {
  try {
    callServiceC();
    break;
  } catch (IOException e) {
  }
}
```

Transient Failure



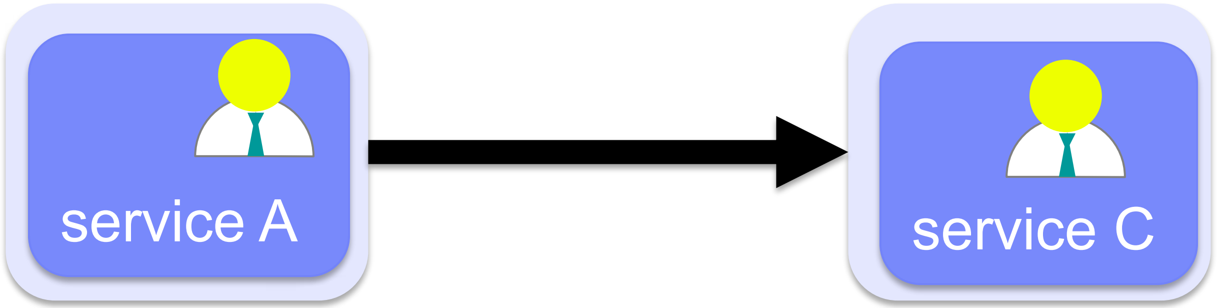
```
@Retry(retryOn=IOException.class,
       delay = 500,
       maxRetries=5
public void callServiceC() {
   // call the service
}
```

Dealing with slow services



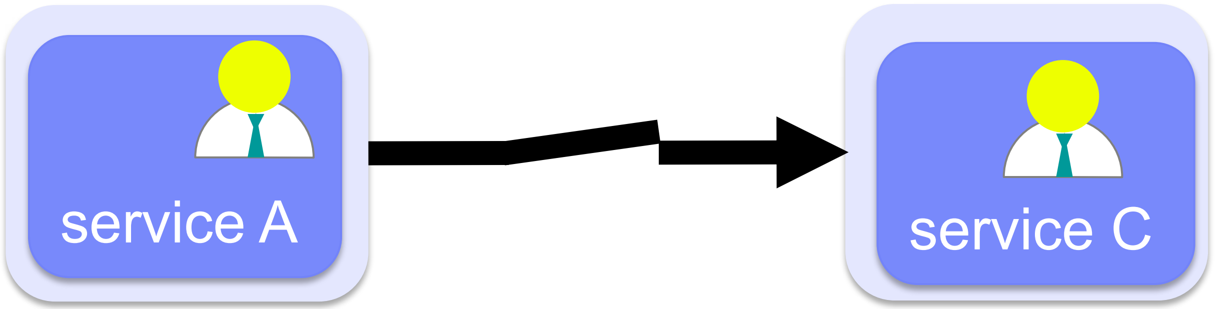
```
@Timeout(2000)
public void callServiceC() {
    // call the service
}
```

Don’t overload serviceC

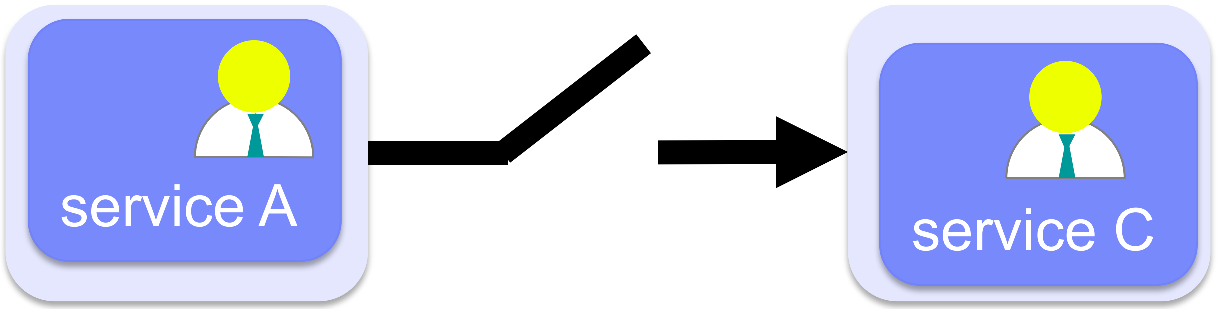


```
@Bulkhead
public void callServiceC() {
    // call the service
}
```

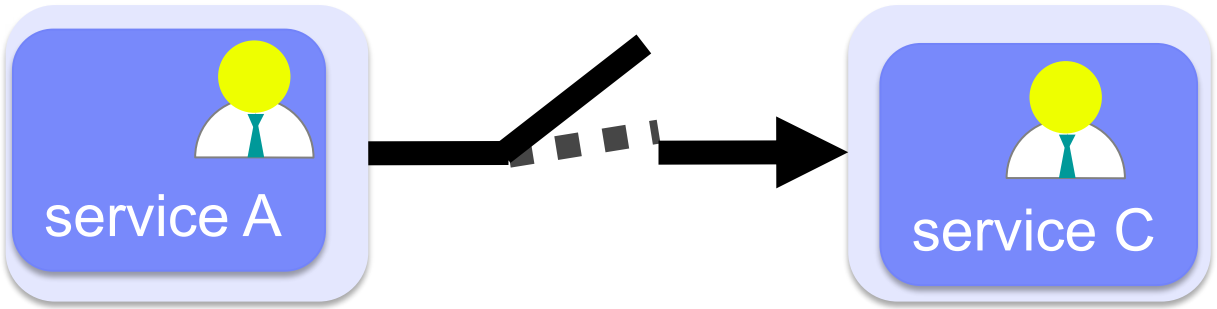
Circuit Breaker Closed



Circuit Breaker Open

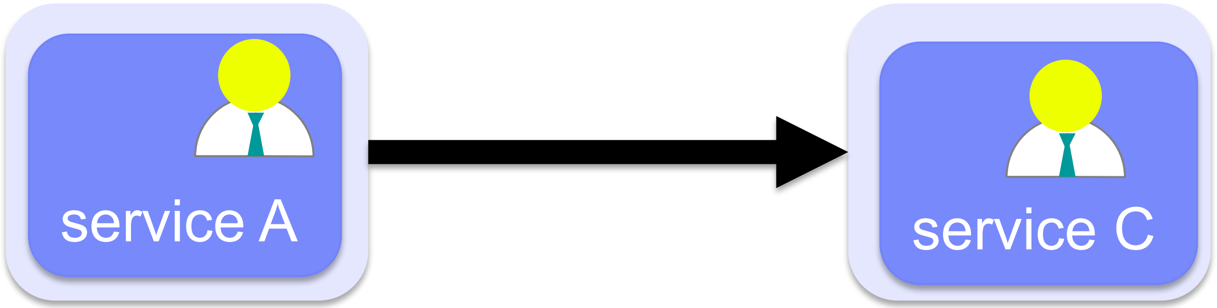


Circuit Breaker Half-Open



```
@CircuitBreaker(failOn=IOException.class,
                delay = 500)
public void callServiceC() {
    // call the service
```

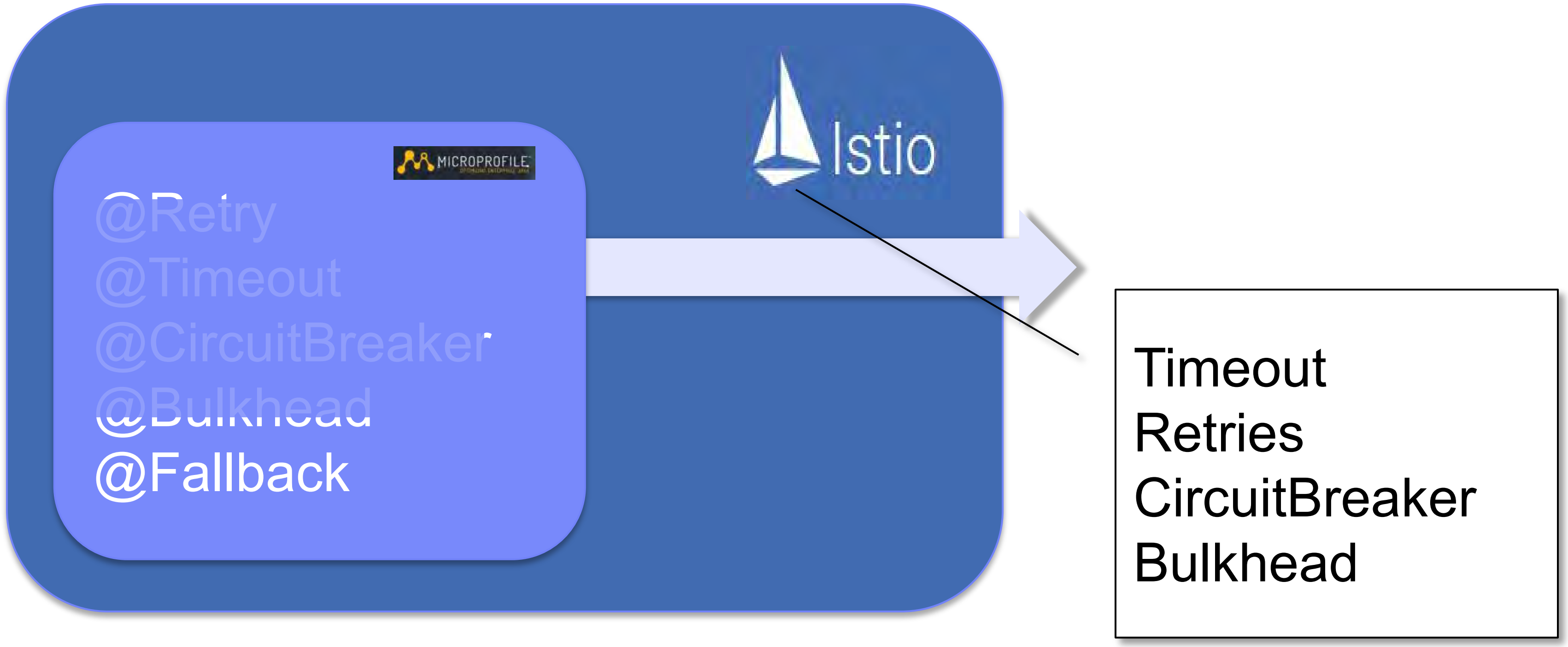
Fallbacks



```
@Fallback(MyFallback.class)
public void callServiceC()
{
  // call the service
}
```

```
private class MyFallback
        implements FallbackHandler {
    public void handle(ExecutionContext c)
{
        // perform fallback action
    }
}
```

MicroProfile Fault Tolerance with Istio



MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA

Config	Fault Tolerance	Health Check	Health Metrics	Security (JWT)
externalize configuration to improve portability	build robust behavior to cope with unexpected failures	ensure services are running and meeting SLAs	understand the interactions between services while running	provides role based access control (RBAC) for microservice endpoints
		mpHealth-1.0	mpMetrics-1.0	

- Exposes **/health** default endpoint for the server/container if feature enabled
  - Standard API for optional application-specific implementation
  - Can be used with Kubernetes liveness check yaml
- Exposes **/metrics** endpoint for the server/container if feature enabled
  - Exposes system, vendor and app-specific metrics
    - OOB metrics include stats about:
      - JVM memory
      - Garbage Collection
      - JVM uptime
      - Threads
      - Thread Pools (stretch goal)
      - ClassLoading
      - CPU usage and availability
  - Response in JSON (for collection from collectd or other JSON-friendly tools) and Prometheus text formats
  - App metrics can be provided using Dropwizard-based API or new CDI-enabled annotations

```
@Timed(name="thinkTime", absolute=true)
void someImportantThing() {
   // method logic here...
}

@Gauge
(name="myGauge", absolute=true)
double somethingToTrack() {
   return myValue;
}
```
Microservice

GET /metrics

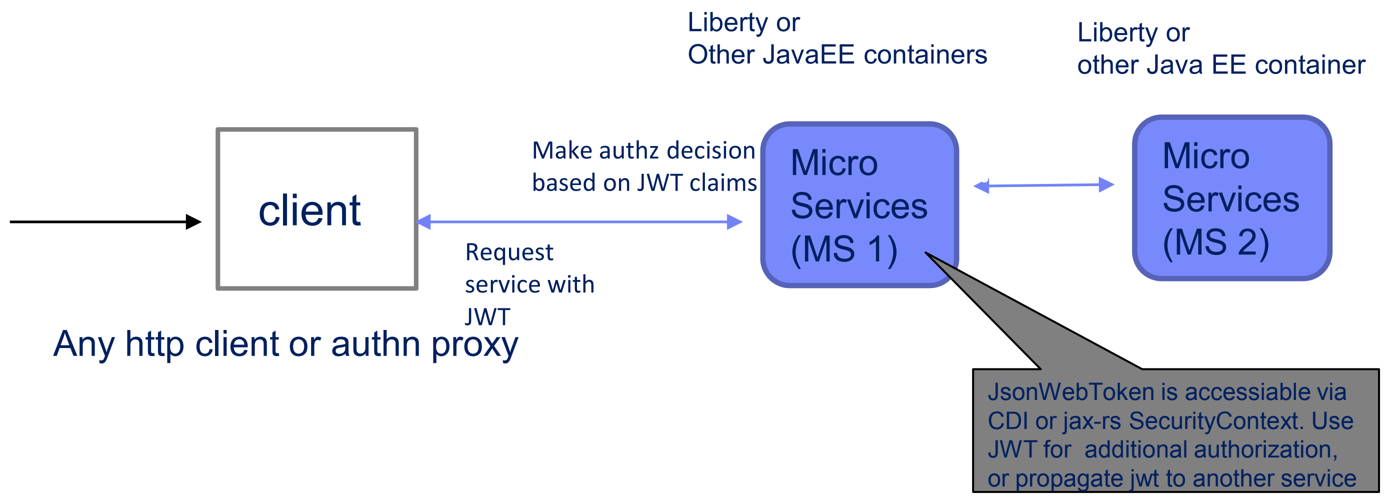
```
# HELP base:cpu_availableProcessors number of processors available to the Java virtual machine
# TYPE base:cpu_availableProcessors gauge
base:cpu_availableProcessors 8

# HELP base:memory_commitedHeap amount of memory in bytes that is committed for the JVM to use
# TYPE base:memory_committedHeap gauge
base:memory_committedHeapMemory 41287680

# TYPE application:thinkTime_count counter
application:thinkTime_count 944534

# TYPE application:myGauge gauge
application:myGauge 0.1834432
```

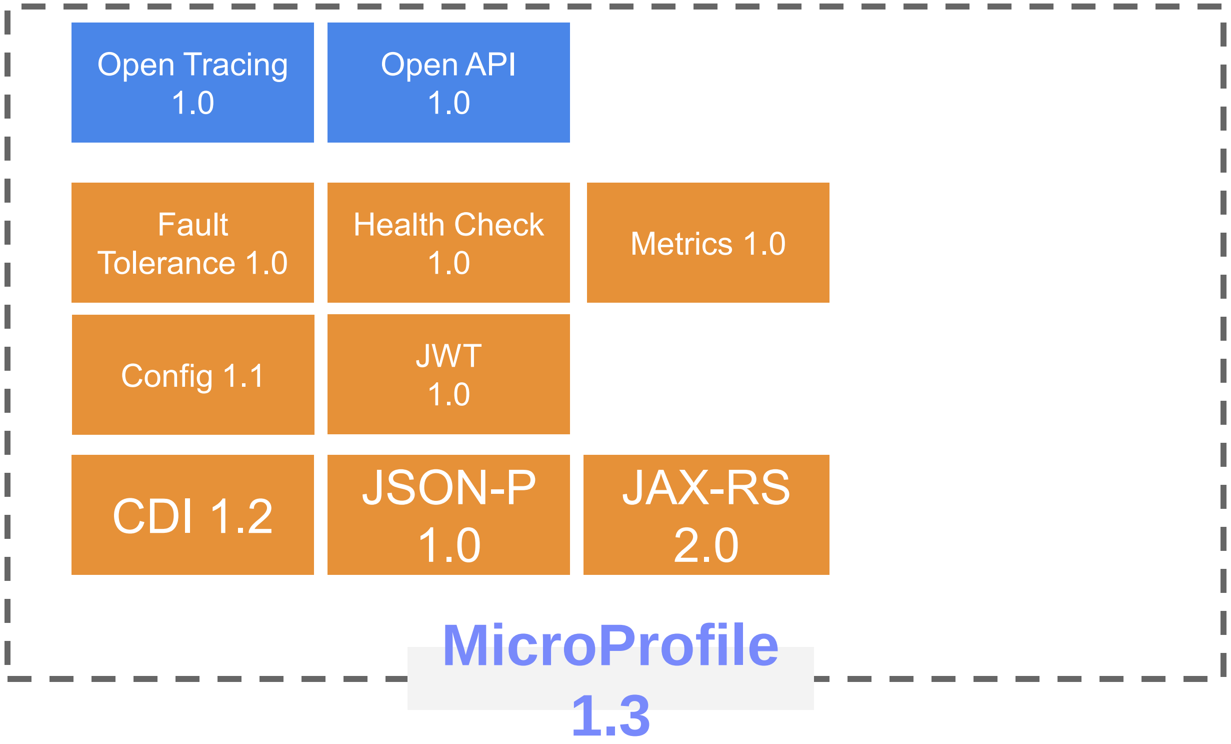
Config	Fault Tolerance	Health Check	Health Metrics	Security (JWT)
externalize configuration to improve portability	build robust behavior to cope with unexpected failures	ensure services are running and meeting SLAs	understand the interactions between services while running	provides role based access control (RBAC) for microservice endpoints mpJwt-1.0



Liberty or
Other JavaEE containers
Liberty or
other Java EE container
client
Make authz decision based on JWT claims
Request service with JWT
Micro Services (MS 1)
Micro Services (MS 2)
Any http client or authn proxy
JsonWebToken is accessiable via CDI or jax-rs SecurityContext. Use JWT for additional authorization, or propagate jwt to another service

1. Client has JWT, and use it to request service over http header
2. Service verifies JWT& create JsonWebToken.& subject
3. Service authorizes request with JsonWebToken.

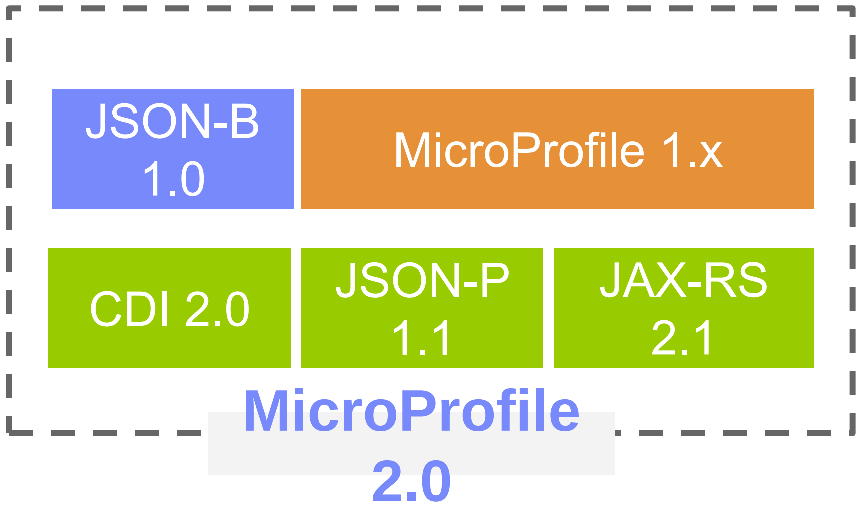
Proposed Eclipse MicroProfile 1.3 (Q4 CY2017?)



🟦 = New

🟧 = No change from last release

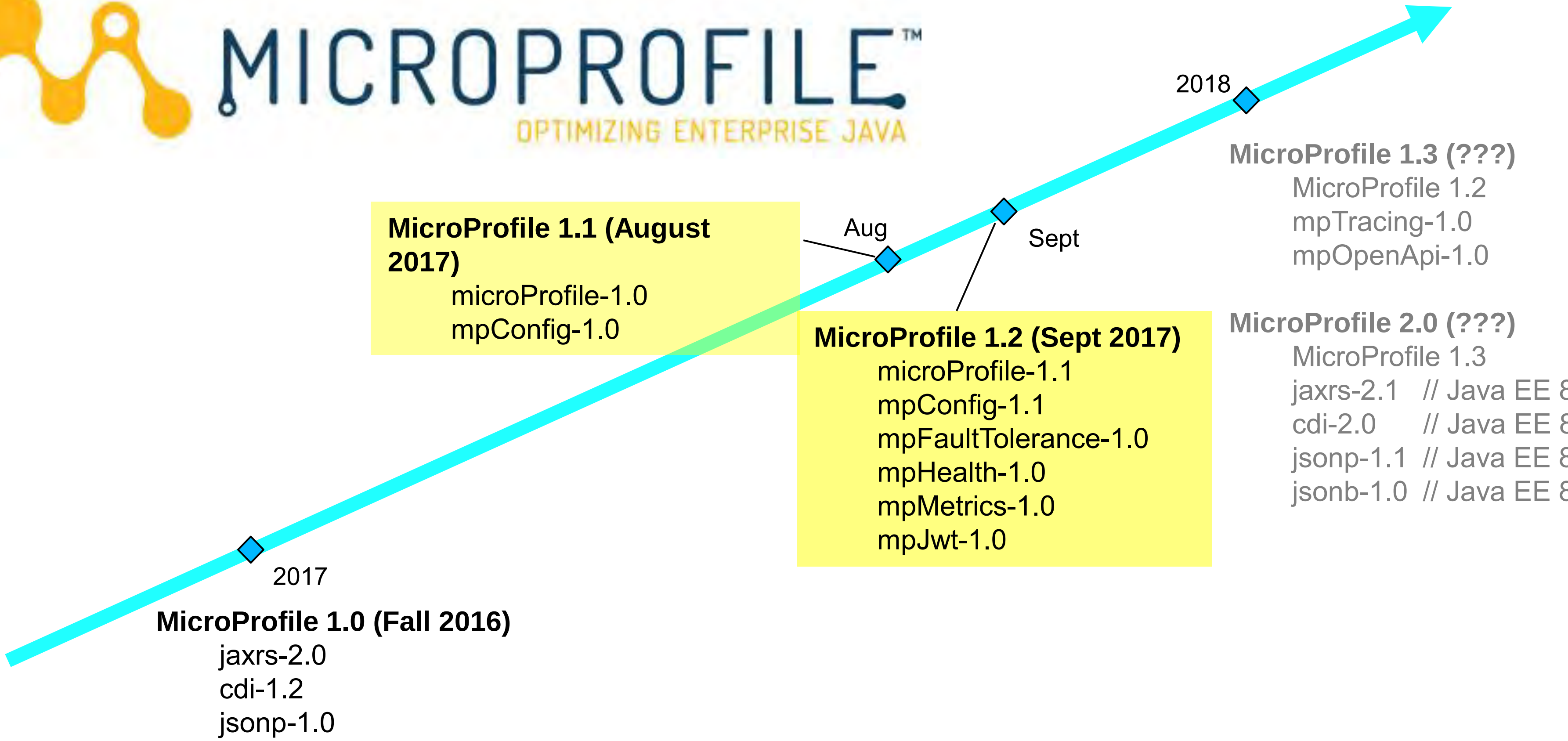
Proposed Eclipse MicroProfile 2.0 (Q4 CY2017?)



🟦 = New

🟩 = Update from last release

🟧 = No change from current MP release



MicroProfile 1.1 (August 2017)
- microProfile-1.0
- mpConfig-1.0

MicroProfile 1.2 (Sept 2017)
- microProfile-1.1
- mpConfig-1.1
- mpFaultTolerance-1.0
- mpHealth-1.0
- mpMetrics-1.0
- mpJwt-1.0

MicroProfile 1.3 (???)
- MicroProfile 1.2
- mpTracing-1.0
- mpOpenApi-1.0

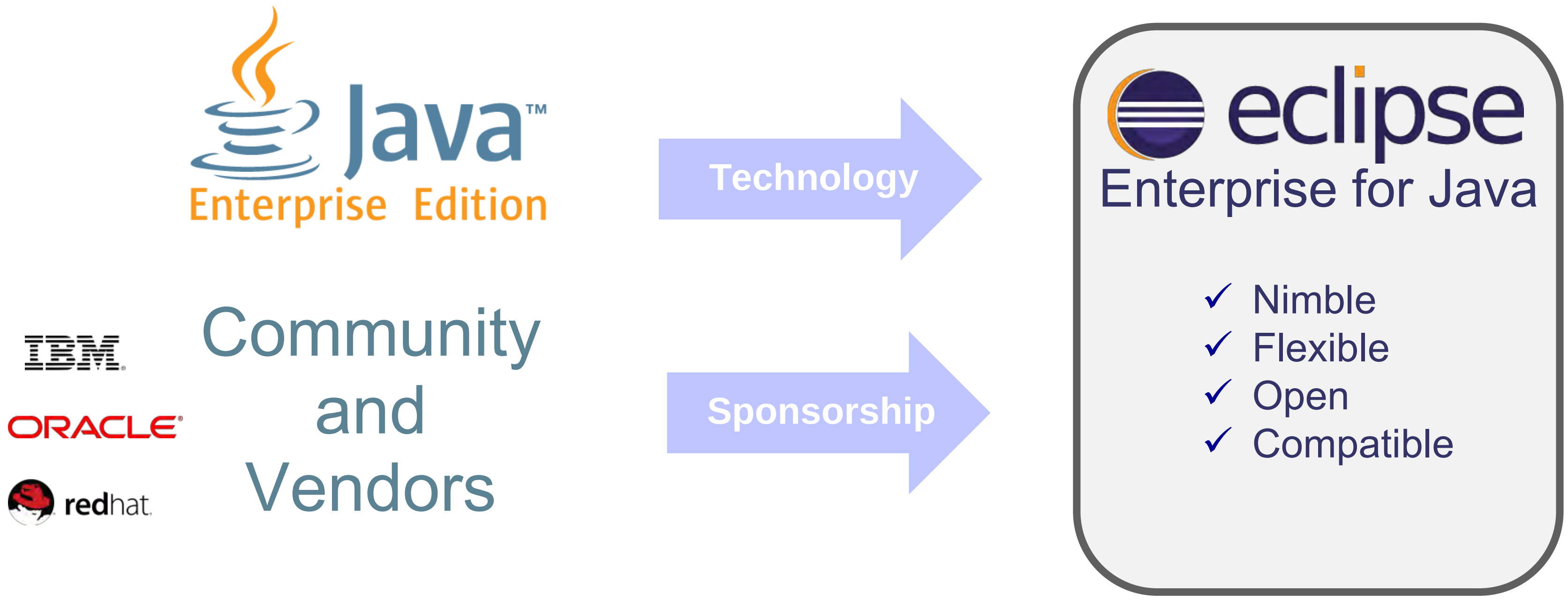
MicroProfile 2.0 (???)
- MicroProfile 1.3
- jaxrs-2.1 // Java EE
- cdi-2.0 // Java EE
- jsonp-1.1 // Java EE
- jsonb-1.0 // Java EE

MicroProfile 1.0 (Fall 2016)
- jaxrs-2.0
- cdi-1.2
- jsonp-1.0

- Resources
  - – http://microprofile.io/
  - – https://openliberty.io/
  - – https://www.eclipse.org/community/eclipse_newsletter/2017/september/

Backup

Eclipse Enterprise for Java (EE4J)
Moving Java EE to Eclipse Foundation



Join the discussion at ee4j-community@eclipse.org

MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA

IBM

Eclipse Enterprise for Java (EE4J) Project Overview

- Open process
- Collaboration: community, vendors, Eclipse
- Transition to EE4J in CY2018
  - GlassFish 5.0/Java EE 8 RIs, TCKs, product docs
  - Process for existing and new specs
  - Compatibility process
- Technology evolution, MicroProfile innovation
- Oracle Java EE Support through Java EE 8
  - Continuity for Java EE community

eclipse
Enterprise for Java
✓ Nimble
✓ Flexible
✓ Open
✓ Compatible

Benefits - A New, Open Direction Forward

- Nimble - more rapid evolution of the technology
- Flexible - modern open source process and licensing
- Open – transparent process, broader vendor, community participation
- Compatible - Transition from Java EE 8 to new offering
- Multiple vendors and established foundation supporting the initiative

Java EE 8 – Early 2017 Draft

JMS 2.1 (JSR 368)	Servlet 4.0 (JSR 369)	MVC 1.0 (JSR 371)
JAX-RS 2.1 (JSR 370)	JSF 2.3 (JSR 372)	JSON-B (JSR 367)
Security 1.0 (JSR 375)	CDI 2.0 (JSR 365)	Bean Validation 2.0 (JSR 380)
JPA 2.2 (JSR 338)	JSON-P 1.1 (JSR 374)	Java Mail 1.6 Common Annotations 1.3 Interceptors 1.2 rev A
Management 2.0 (JSR 373)	Configuration 1.0 (JSR ???)	Health Check 1.0 (JSR ???)

■ Java EE 8   No Change ▨   Drop from Java EE 8 ▢   Add to

Java EE 8 – Final Content

Servlet 4.0
(JSR 369)

JAX-RS 2.1
(JSR 370)

JSF 2.3
(JSR 372)

JSON-B
(JSR 367)

Security 1.0
(JSR 375)

CDI 2.0
(JSR 365)

Bean Validation
2.0
(JSR 380)

JPA 2.2
(JSR 338)

JSON-P 1.1
(JSR 374)

Java Mail 1.6
Common Annotations
1.3
Interceptors 1.2 rev A