Fallando con estilo
Fault Tolerance & Metrics

22 de Julio, 2023

Eclipse MicroProfile Community

¿Quién?

Jorge Cajas
@cajasmota

- JUG co-leader en GuateJUG
- Desarrollador Java certificado con +9 años de experiencia.
- Ganador del Duke Choice Award en 2016 con GuateJUG
- Parte del comité organizador del JConf Guatemala
- Conferencista desde 2012 en eventos nacionales e internacionales.
- Consultor en MangoChango S.A.

GUATE-JUG

Duke's Choice Award
2016 Winner
Java

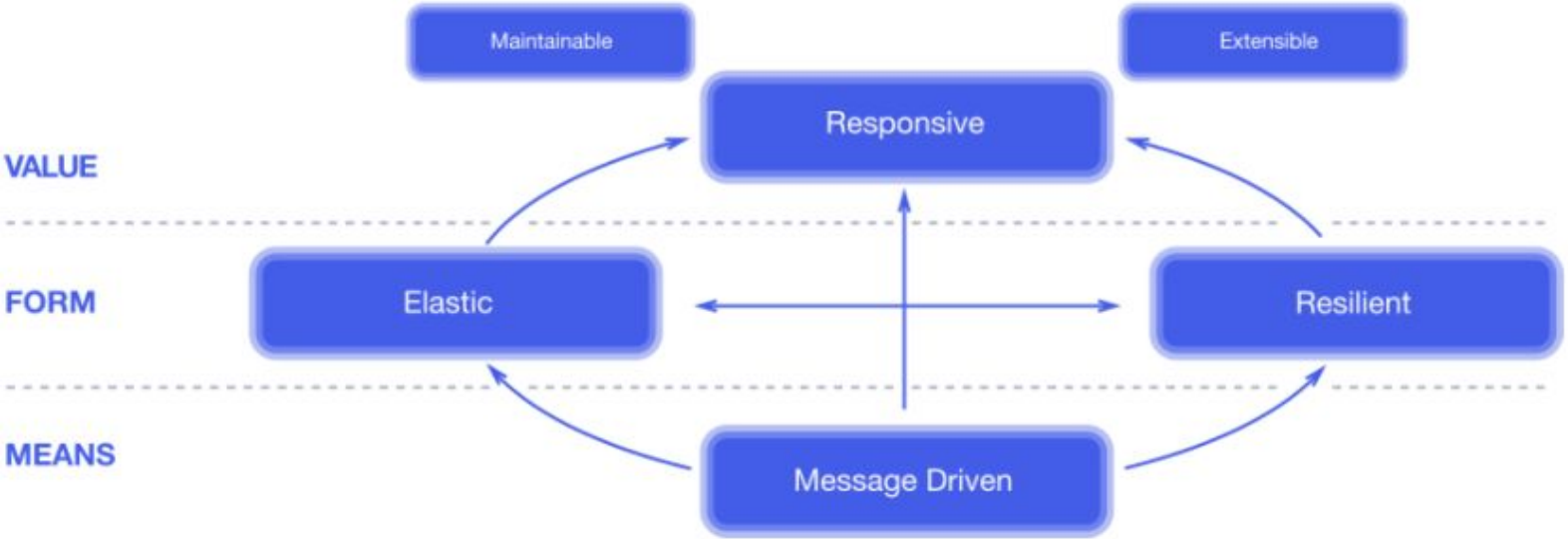
ORACLE Certified Associate

Contenido

- **JakartaEE & Eclipse MicroProfile**
- **Como empezar**
- **Métricas**
  - API
  - Ejemplos
- **Tolerancia a Fallos**
  - API
  - Ejemplos

¿Porqué?

Aplicaciones modernas



Imágenes: @tuxtor

La comunidad de JavaEE

JAKARTA EE

JAVA EE GUARDIANS

MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA

Eclipse MicroProfile

¿Porqué MicroProfile?

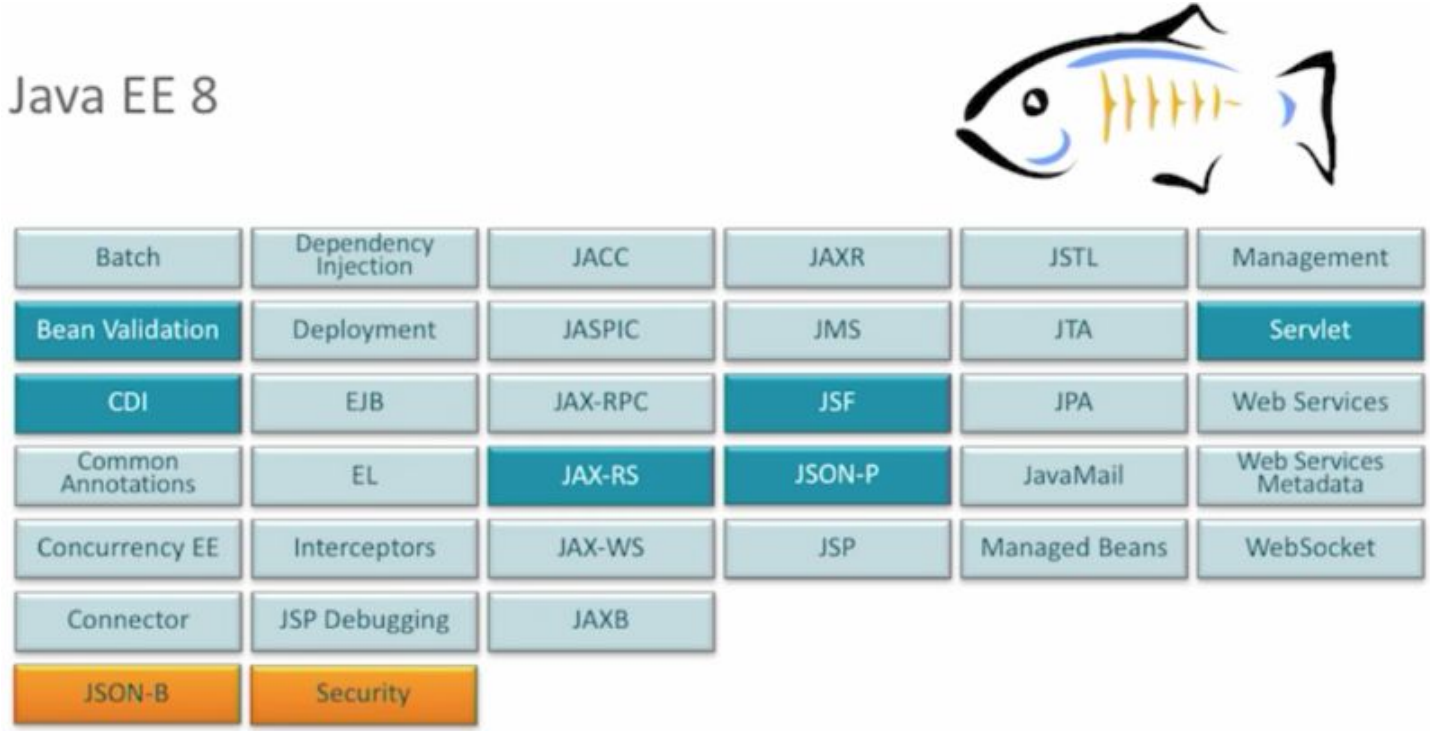
- Java se estaba quedando atrás en la nueva era de microservicios, por lo cual nacieron nuevos frameworks como vert.x, spring boot, entre otros, que ayudan al desarrollo de microservicios con Java.

- A pesar de esos esfuerzos, no existía una forma estandarizada de desarrollar microservicios, contrario a lo que diferencia el desarrollo de aplicaciones empresariales bajo los estándares de Java EE.

- Se buscaba una forma de reutilizar todo el conocimiento de JavaEE que ya tenemos y orientarlo a el desarrollo de microservicios.

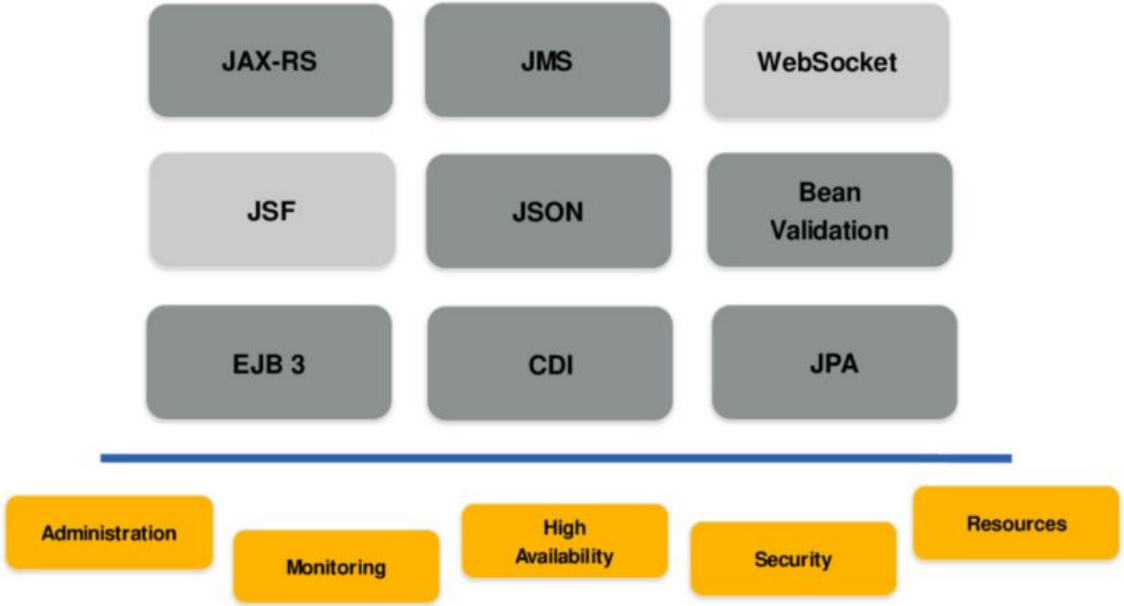
JavaEE / JakartaEE

MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA



MicroProfile - API's

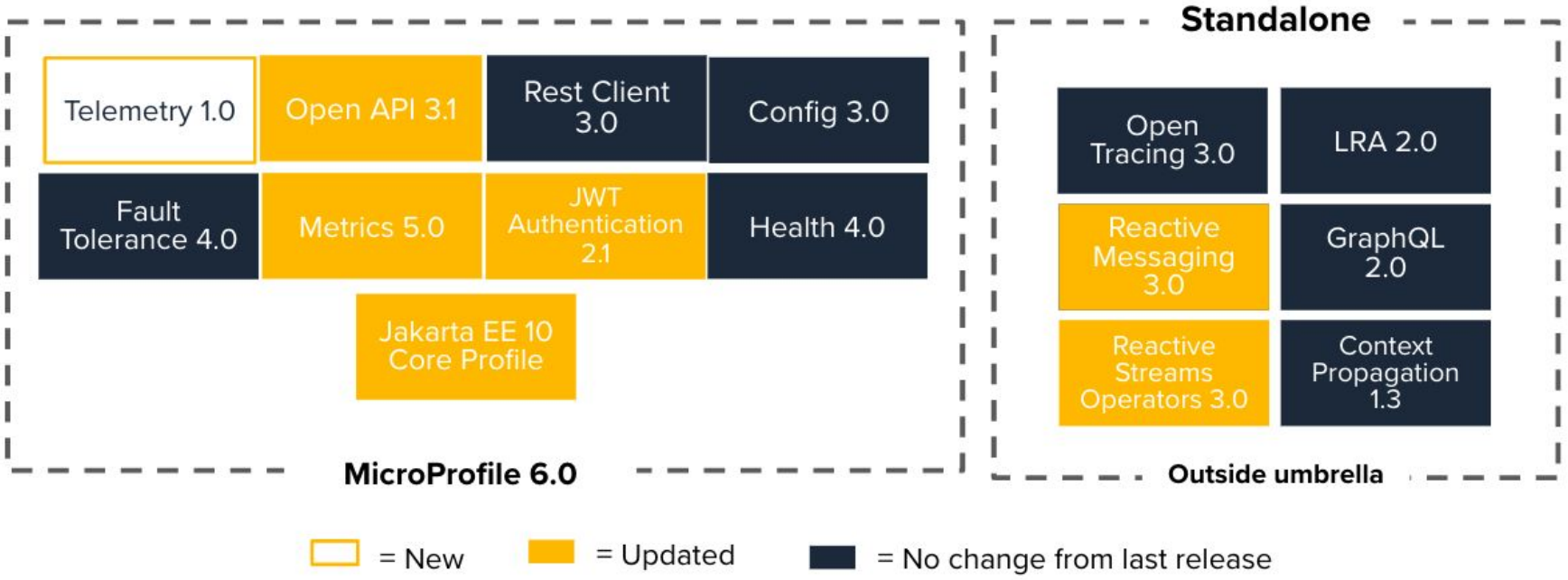
JavaEE / Jakarta EE no es suficiente



Imágenes: @reza_rahman

¿Quienes apoyan a MicroProfile?

MicroProfile - API's



Metricas

@Counted

@Gauge

@Timed

@Histogram

¿Cómo leer métricas?

Tolerancia a fallos

@Timeout

@Retry

@Fallback

@CircuitBreaker

@Bulkhead

Tolerancia a Fallos

MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA

```
@GET
@Path("/{id}")
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}
```

Tolerancia a Fallos

MICROPROFILE™
OPTIMIZING ENTERPRISE JAVA

```java
@GET
@Path("/{id}")
@Timeout(300)
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}
```

Tolerancia a Fallos

MICROPROFILE
OPTIMIZING ENTERPRISE JAVA

```
@GET
@Path("/{id}")
@Timeout(300)
@Retry(maxRetries = 5, delay = 60)
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}
```

Tolerancia a Fallos

MICROPROFILE
OPTIMIZING ENTERPRISE JAVA

```
@GET
@Path("/{id}")
@Timeout(300)
@Retry(maxRetries = 5, delay = 60)
@Fallback(fallbackMethod = "findByIdFallBack")
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}

public Response findByIdFallBack(@PathParam("id") final String imdbId) {
    //more magic from cache here
    return Response.ok().build();
}
```

Metricas

```
@GET
@Path("/{id}")
@Timeout(300)
@Counted(name = "queries")
@Retry(maxRetries = 5, delay = 60)
@Fallback(fallbackMethod = "findByIdFallBack")
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}

public Response findByIdFallBack(@PathParam("id") final String imdbId) {
    //more magic from cache here
    return Response.ok().build();
}
```

Metricas

```
@Inject
@Metric
Counter failures;

@GET
@Path("/{id}")
@Timeout(300)
@Counted(name = "queries")
@Retry(maxRetries = 5, delay = 60)
@Fallback(fallbackMethod = "findByIdFallBack")
public Response findById(@PathParam("id") final String id) {
    //Logic and magic
    return Response.ok().build();
}

public Response findByIdFallBack(@PathParam("id") final String imdbId) {
    //more magic from cache here
    failures.inc();
    return Response.ok().build();
}
```

Demo

- WS REST
- Métricas
- Tolerancia a fallos
- Cliente REST

Fallando con estilo
Fault Tolerance & Metrics

@cajasmota