Eclipse Microprofile

Optimizing Enterprise Java
for a Microservices Architecture

Alexis Lopez
Consultor Independiente,
@aa_lopez/aalopez@gmail.com

Acerca de Alexis López

ORACLE Certified Professional Java SE 8 Programmer

ORACLE Certified Specialist

Java Champions

Oracle Groundbreaker Ambassador

CLOJUG

@acelopezco

www.acelopez.com

payara®

radiate RUBY

Enterprise World

Java EE / Jakarta EE

Cloud Computing - Microservices

Cloud

- Reduce time to market
- Address unpredictable loads
- Pay as you go
- Containerization

Microservices

- Deliver new features more quickly
- Smaller, more agile teams
- Deliver business features as discrete services
- Scale services independently

Eclipse MicroProfile

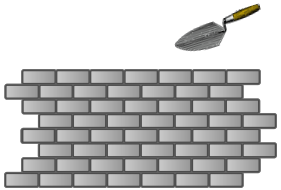
MicroProfile Release Philosophy

Release 1.0

JAX-RS
CDI
JSON-P

Rapidly iterate and innovate

Build consensus



Standardize

Sept 2016

Java EE/Jakarta EE Based

MicroProfile 1.4

Based on Java EE 7

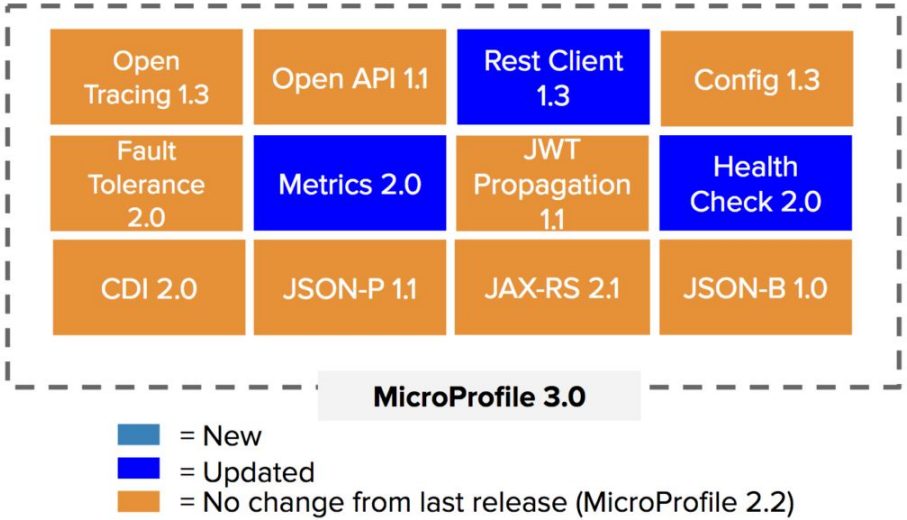
MicroProfile 3.x

Based on Java EE 8

Microprofile evolution

Septiembre 2016

Junio 2019



microprofile-metrics

https://microprofile.io/project/eclipse/microprofile-metrics

Desde MP-1.2

REST Endpoints:

- **/metrics/base**
- **/metrics/vendor**
- **/metrics/application**
- **/metrics**

GET -> Valor
OPTIONS -> Metadatos

Application Metrics:

- @Gauge
- @Counted
- @Timed
- @Metered
- @ConcurrentGauge
- Histogram

microprofile-metrics

https://microprofile.io/project/eclipse/microprofile-metrics

Desde MP-1.2

```
@Gauge(unit = "USD", name = "price", absolute = true)
public long getPrice() { return 4; }

GET /metrics/application/price
```

```
@Timed (name = "getAll", absolute = true,
        unit = MetricUnits.MILLISECONDS
        description = "Monitor the time getAll Method takes")
public Response getAll() { ... }

GET /metrics/application/getAll
```


```
@Counted(unit = MetricUnits.NONE, name = "login",
        absolute = true,
        description = "Shows how many times login method
        was called.")
@POST
@Path("/login")
public Response login() {
  //...
  return Response.ok().build();
}

GET /metrics/application/login
```

microprofile-health

https://microprofile.io/project/eclipse/microprofile-health

Desde MP-1.2

Tipos de verificaciones:

- **@Readiness**
- **@Liveness**
- **@Health**

```
@FunctionalInterface
public interface HealthCheck { HealthCheckResponse call(); }
```

REST Endpoints:

- **/health/ready**
- **/health/live**
- **/health**

```
public abstract class HealthCheckResponse {
    public enum State { UP, DOWN }
    public abstract String getName();
    public abstract State getState();
    public abstract Optional<Map<String, Object>> getData();
    [...]
}
```

microprofile-health

https://microprofile.io/project/eclipse/microprofile-health

Desde MP-1.2

```
@Readiness
@ApplicationScoped
public class SysReadinessCheck implements HealthCheck {
  @Override
  public HealthCheckResponse call() {
    if (/*VALIDATIONS*/) {
      return HealthCheckResponse.named("SystemReadiness")
                                .withData("server", "not available")
                                .down()
                                .build();
    }
    return HealthCheckResponse.named("SystemReadiness")
                                .withData("server", "available")
                                .up()
                                .build();
  }
}
```

```
@Liveness
@ApplicationScoped
public class SysLivenessCheck implements HealthCheck {
  @Override
  public HealthCheckResponse call() {
    return HealthCheckResponse.named("SystemLiveness")
            .withData("memory", Runtime.getRuntime().freeMemory())
            .state(true)
            .build();
  }
}
```

Microprofile open api

https://microprofile.io/project/eclipse/microprofile-open-api

Microprofile open api

https://microprofile.io/project/eclipse/microprofile-open-api

REST Endpoints:

- **/openapi**

Tipos de verificaciones:

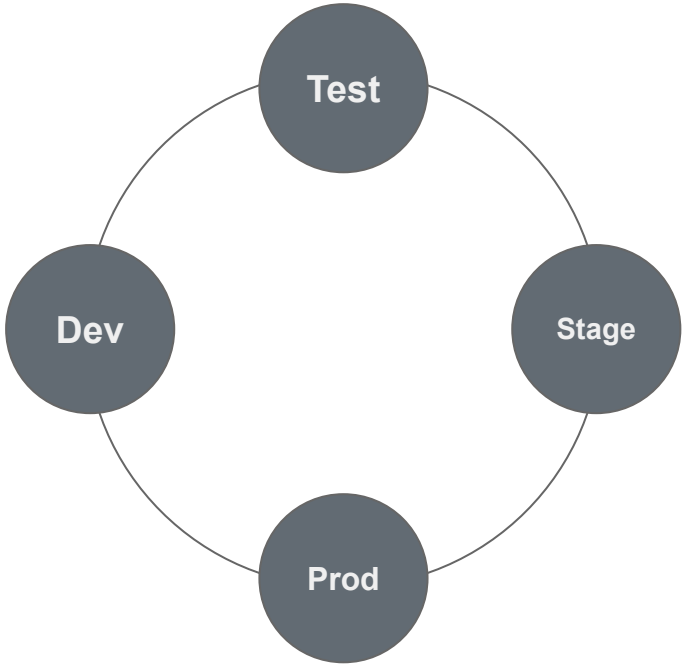
- **@APIResponses**
- **@APIResponse**
- **@Content**
- **@Schema**
- **@Operation**
- **@Parameter**

```
@GET
@Path("/endpoint")
@Produces(MediaType.APPLICATION_JSON)
@APIResponses( value = {
                    @APIResponse(
                        responseCode = "404",
                        description = "No encontrado",
                        content = @Content(mediaType = "text/plain"))
})
@Operation(summary = "Texto descriptivo del endpoint ",
            description = "Más descripción")
public Response getProps( @Parameter(description = "Param desc.",
                                        required = true,
                                        example = "foo",
                                        schema = @Schema(type =
                                                    SchemaType.STRING))
                        @PathParam("hostname") String hostname
{
//...
}
```

MicroProfile Configuration Feature

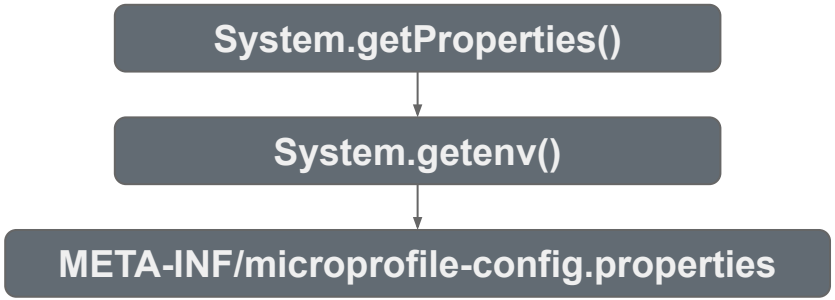
https://microprofile.io/project/eclipse/microprofile-config

Desde MP-1.1



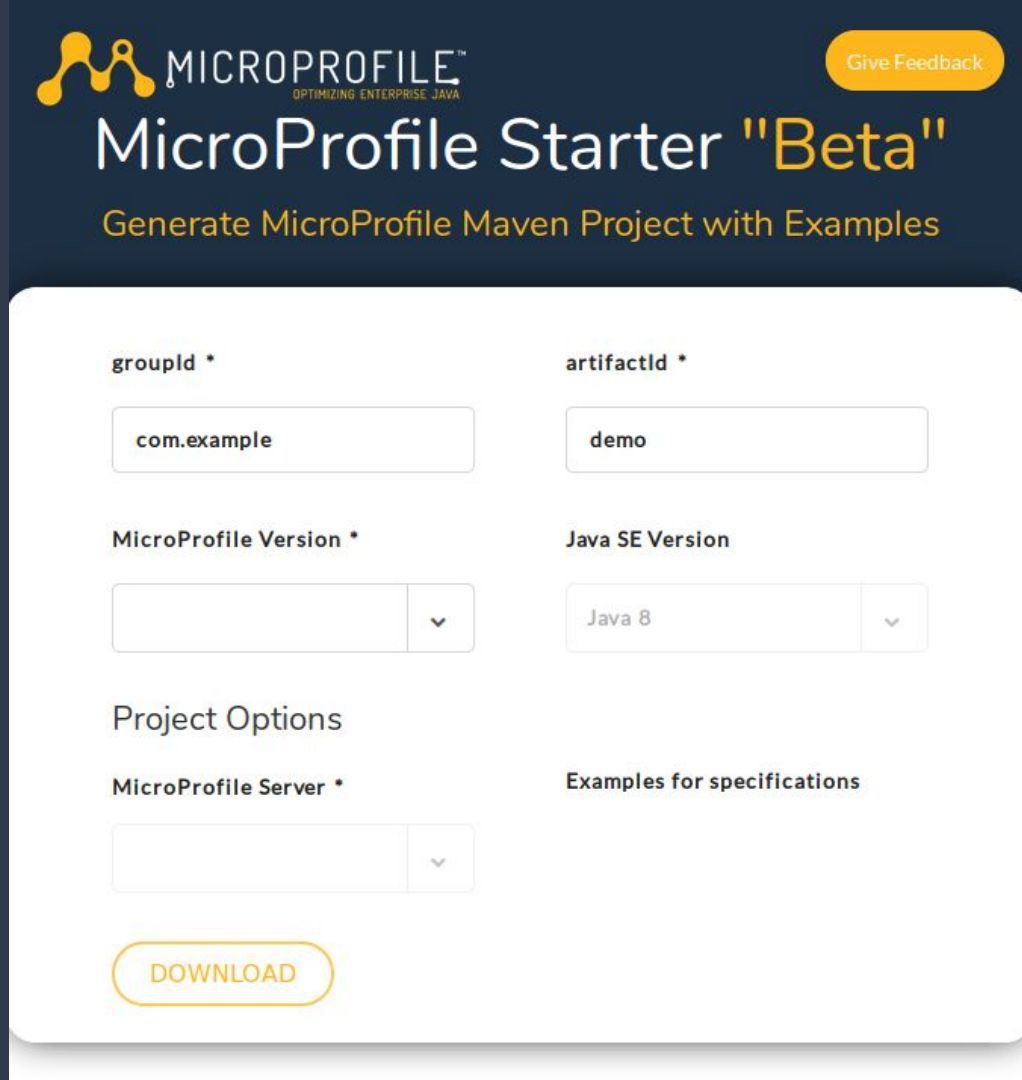

```
@Inject
@ConfigProperty(name = "host", defaultValue = "localhost")
private String host;

@Inject
private Config config;
```

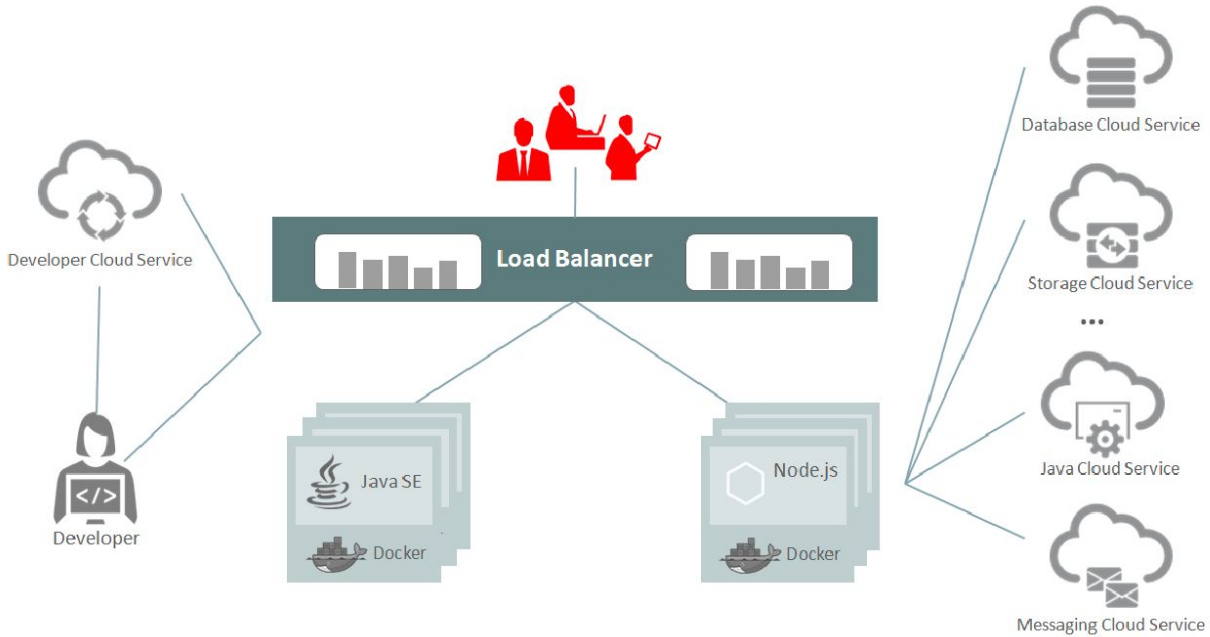


MicroProfile Starter "Beta"

https://start.microprofile.io/



Oracle Application Container Cloud

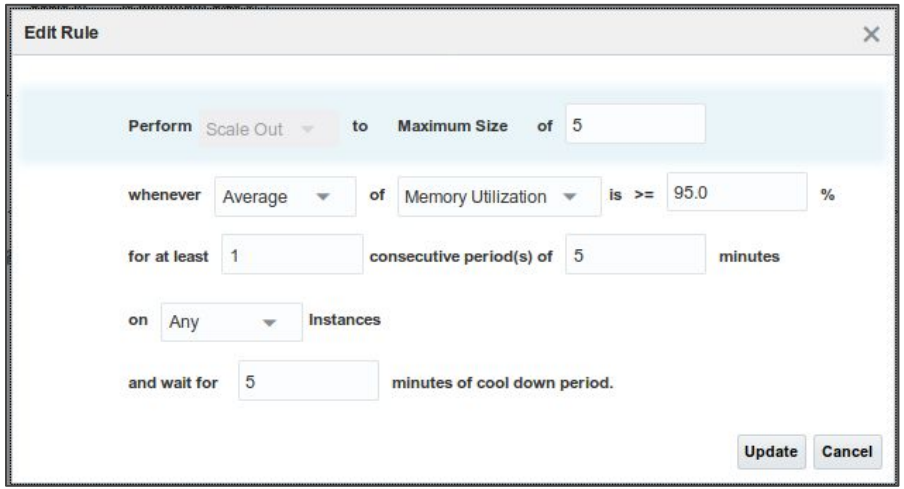


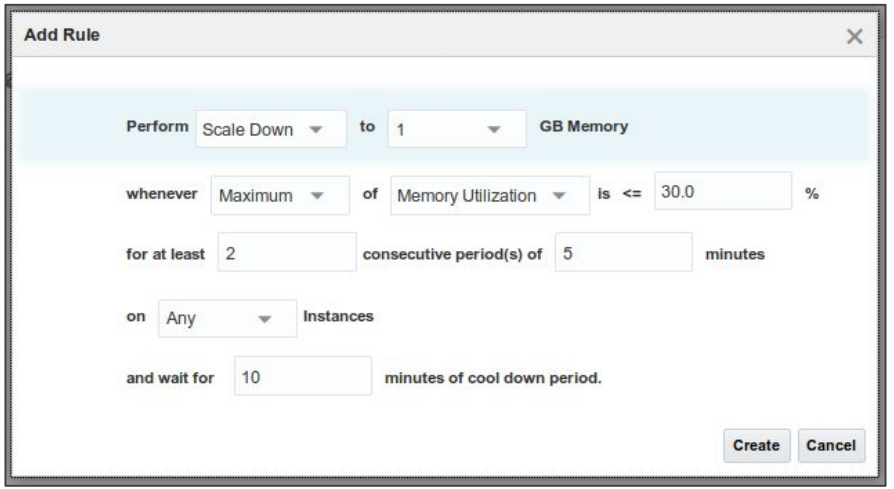
Oracle Application Container Cloud

Tecnologías Soportadas

Oracle Application Container Cloud

Escalamiento Automático





Join the Community!

Demo

Thank you!