Eclipse MicroProfile

•••

What's next?

MICROPROFILE™

OPTIMIZING ENTERPRISE JAVA

- Defines **open source** Java **microservices** specifications
- Industry Collaboration - Red Hat, IBM, Payara, Tomitribe, London Java Community, SouJava, Oracle, Hazelcast, Fujitsu, Lightbend, Microsoft...

Eclipse MicroProfile so far

- 1.0 - September 2016 (Java EE 7 - CDI, JAX-RS, JSON-P)
- 1.1 - July 2017 (1.0 + Config 1.0)
- 1.2 - September 2017 (1.1 + Config 1.1, Fault Tolerance 1.0, JWT Propagation 1.0, Metrics 1.0, Health 1.0)
- 1.3 - December 2017 (1.2 + Config 1.2, Metrics 1.1, Open API 1.0, Open Tracing 1.0, REST Client 1.0)

Eclipse MicroProfile so far

- 1.4 - June 2018 (1.3 + Config 1.3, Fault Tolerance 1.1, JWT Propagation 1.1, Open Tracing 1.1, REST Client 1.1)
- 2.0 - June 2018 (Java EE 8 - CDI, JAX-RS, JSON-P, JSON-B + Config 1.3, Fault Tolerance 1.1, JWT Propagation 1.1, Metrics 1.1, Health 1.0, Open API 1.0, Open Tracing 1.1, REST Client 1.1)
- 2.1 - October 2018 (2.0 + Open Tracing 1.2)

Future Releases

- Recently shifted to three releases a year:
  - February, June, October
- 2.2 - February 2019 (Proposed specifications)
  - Config 1.4
  - Fault Tolerance 2.0
  - Metrics 2.0
  - Health 1.1
  - REST Client 1.2
  - Reactive Streams Operators 1.0
  - Reactive Messaging 1.0

Jakarta EE

Eclipse MicroProfile and Jakarta EE

- General thoughts:
  - Jakarta EE still getting going
  - Likely different release cycles
  - Different processes
  - Desire to not duplicate effort
- MicroProfile keeps going, keep an eye on Jakarta EE

In Progress Specifications

Reactive Streams Operators

- Set of operators for manipulating Reactive Streams:
  - Creating
  - Manipulating
  - Consuming and Accumulating

Reactive Messaging

- CDI extension for developing an application with data streaming
- Based on `@Incoming` (consumed stream) and `@Outgoing` (feed streams)
  - Can be connected to any messaging transport (Kafka, AMQP, MQTT)
- Implicit or Explicit message acknowledgement
- Custom subtypes of `Message` such as `CloudEvent`, `KafkaMessage`, `MQTTMessage`

Long Running Actions (LRA)

- Coordination of activities between Services
- Ability to provide consistency guarantees
- No strong coupling between Services
- Compensating actions if an activity is cancelled

Concurrency

- `CompletableFuture` backed by managed threads
- Thread context propagation to subsequent `CompletableFuture` actions
- Precursor to the problem being addressed in Jakarta EE
- Ability to `@Inject ManagedExecutor` and `ThreadContext`
- Config annotations to customize settings of `ManagedExecutor` and `ThreadContext`

Service Mesh

- Testing and Verifying MP specifications in Service Mesh, such as Istio
- Documenting use cases
- https://github.com/eclipse/microprofile-service-mesh
- https://github.com/eclipse/microprofile-service-mesh-service-a
- https://github.com/eclipse/microprofile-service-mesh-service-b

MicroProfile Links

- http://microprofile.io
- https://projects.eclipse.org/projects/technology.microprofile
- https://wiki.eclipse.org/MicroProfile
- Forum: https://groups.google.com/forum/#!forum/microprofile
- Calendar: https://calendar.google.com/calendar/embed?src=gbnbc373ga40n0tvbl88nkc3r4%40group.calendar.google.com&ctz=GMT

General Discussion