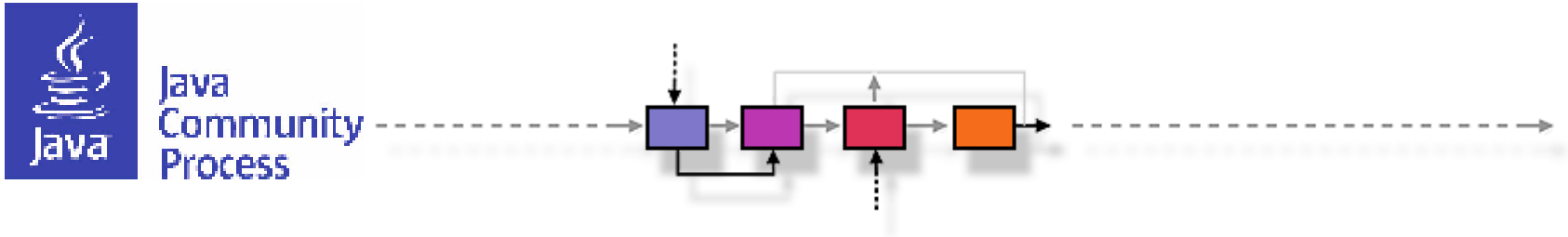
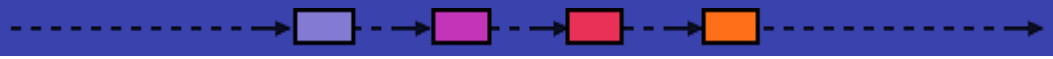



# JAX-RS Review

May 9 2017

Santiago Pericas-Geertsen

# Agenda



- About JAX-RS 2.1 (JSR 370)
- Introduction and History
- New Technical Features
- Community
- Conclusions and Next Steps

Java Community Process

# About

- 10 years in the making
  - JAX-RS 1.0 started August, 2007
- Primary goal:

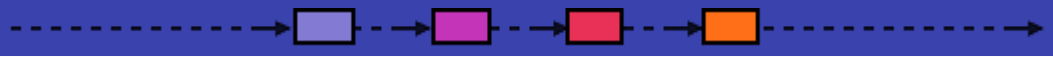
  *Define a set of Java APIs for the development of Web services according to the REST architectural style*
- Sub-goals:
  - POJO based
  - HTTP centric
  - Format independence
  - Container independence
  - Part of Java EE

Java Community Process

# History

- Spanned 3 JSRs: 311, 339, 370 (current)
- Versions:
  - JAX-RS 1.0: October 2008
  - JAX-RS 1.1: November 2009
  - JAX-RS 2.0: May 2013
  - JAX-RS 2.1: *Planned for July 2017*

# The Expert Group



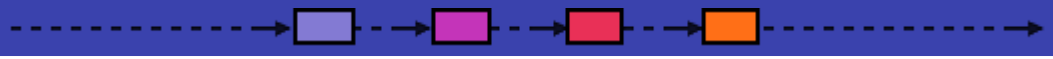
- Members:
  - Pavel Bucek, Santiago Pericas-Geertsen (Leads)
  - Adam Bien, Sebastian Dashner, Andy McCright, Markus Karg, Marcos Luna, Alessio Soldano, Julian Reschke, Sergey Beryozkin, Casey Lee
- Companies represented:
  - IBM, Red Hat, Talend, VSP and Oracle
- Communication: expert mailing list only
- Tools: Wikis, GIT forks

# Proposed Features in 2.1

- Server Sent Events
- Reactive clients
- ~~Non-Blocking I/O~~
- Alignment with JSON-B and Servlet 4.0
- Better integration with CDI

Java Community Process

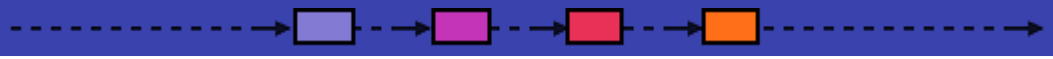
# Server Sent Events (1 of 2)



- Server API: Injectable Sse and SseEventSink types
- Declare SSE media type in @Produces
- Example:

```
@GET  @Produces(“text/event-stream”)  public void eventStream(
    @Context SseEventSink eventSink,      @Context Sse sse) {
    executor.execute(() -> {
        try (SseEventSink sink = eventSink) {
eventSink.send(sse.newEvent("event1"));
            eventSink.send(sse.newEvent("event2"));
        } catch (IOException e) {                    // handle exception            }
});}
```

# Server Sent Events (2 of 2)



- Client API: SseEventSource as dual of server sink
- Multiple life-cycle events: onEvent, OnError, OnComplete
- Example with onEvent callback:

```
try (SseEventSource source =
        SseEventSource.target("http://…").build()) {
   source.register(System.out::println);
   source.open();     Thread.sleep(500);    } catch (InterruptedException e)
{     // falls through   }
```

# Reactive Clients (1 of 2)

- Support for rx() modifier and CompletionStage
- Leverage async computation library in Java SE 8
- Example:

```
CompletionStage<String> cs1 =
    target1.request().rx().get(String.class);

CompletionStage<String> cs2 =
    cs1.thenCompose(user ->
      target2.request().header("user", user)
        .rx().get(String.class));

cs2.thenAccept(quote -> System.out.println(quote));
```

# Reactive Clients (2 of 2)

- Other Reactive libraries supported as extensions
- RxJava Example:

```
Client client =
  client.register(ObservableRxInvokerProvider.class);

Observable<String> of =
    client.target("forecast/{destination}")
          .resolveTemplate("destination", "mars")
          .request()
          .rx(ObservableRxInvoker.class)
          .get(String.class);

of.subscribe(System.out::println);
```

Register a Provider

Override default Invoker

Java Community Process

# Non-Blocking I/O: Feature Dropped

- Basic idea:
  - Support for streaming collections in entities
  - Proposal based on Java SE 9 Flows
- Problems:
  - Hard to support on Java SE 8 with forward compatibility
  - Unproven approach needs more experimentation
  - Time constraints

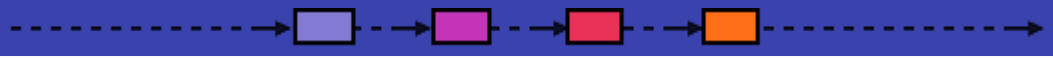
# Other deliverables

- RI: Jersey (Oracle)
- TCK (Oracle)
- Jersey's user guide and samples
  - https://github.com/jersey/jersey

# Liasions with other JSRs

- JSON-B
  - Support for JSON-B objects
- BV
  - Parameter and entity validation
- Servlet
  - Integration with new features
- CDI
  - Dependency injection in JAX-RS classes

# Implementations



- Other implementations besides RI:
  - RESTeasy: resteasy.jboss.org (Red Hat)
  - Apache CXF: cxf.apache.org (Apache)

# Conference Talks

- Devoxx US:
  - JAX-RS 2.1 Reloaded (March 2017)
  - http://cfp.devoxx.us/2017/talk/GFN-6657
    - Includes: Reactive Clients, SSE and NIO

# Schedule

- Renew Ballot 2: December 12, 2016
- Early Draft Review: April 12, 2017 (Completed)
- Public Review: May 20, 2017 (Completed)
- Public Review Ballot: June 5, 2017 (*In progress*)
- Proposed Final Draft: *Late June, 2017*
- Proposed Final Draft Ballot: *Late July 2017*

# Participation and transparency

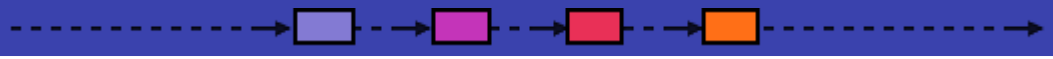
- JSR page on JCP.org:
  - https://www.jcp.org/en/jsr/detail?id=370
- Other Links:
  - https://jax-rs-spec.java.net/ (Old)
  - https://github.com/jax-rs (New)
    - Work in progress
  - https://github.com/jersey/jersey (RI)

Java Community Process

# Mailing lists

- Mailing lists:
  - jsr370-experts@jax-rs-spec.java.net (Old)
    - # of messages 2017: 360
    - Mirrored on User's alias
  - users@jax-rs-spec.java.net (Old)
    - # of messages 2017: 411

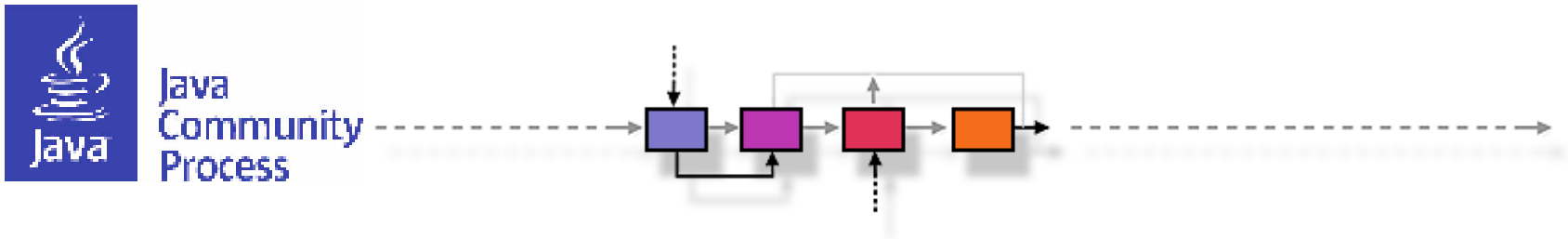
# Issue tracker



- 164 Open and 385 Closed
- https://github.com/jax-rs/api/issues (New)
- Discussions on mailing lists
  - Most actions carried out by Spec Leads

# Conclusions and Next Steps

- Summary:
  - Work on SSE completed and stable
  - Work on Reactive Clients completed and stable
  - Initial integration with JSON-B in progress
  - Non-blocking I/O left as ***future work***
    - Initial proposal based on Java SE 9 Flows
- Next Steps:
  - Review high-priority issues
  - Some BV and CDI issues left




Thank you!

https://www.jcp.org/en/jsr/detail?id=370