Expose Your Business Services With JAX-RS

Martin Kouba

Here comes JAX-RS...

- Java API for RESTful Web Services
  - 1.0 (2008) - JSR 311
  - 1.1 (2009) - JSR 311 / **part of Java EE 6**
  - 2.0 (? 2012) - JSR 339
  - implemenations: Jersey (RI), RESTEasy, Apache CXF, Restlet
  - annotation-based: `@Path`, `@GET`, `@POST`, `@Produces`, `@Consumes`, `@PathParam`, `@QueryParam`, ...

REpresentational State Transfer

- Roy Fielding (2000)
- **architecture style** for distributed systems
- key goals:
  - scalability, loose coupling, intermediary components, ...
- constraints:
  - client/server, stateless (no client session context on the server), cacheable, …
- RESTful means conforming to the constraints

RESTful Web Service

- web service implemented using HTTP and conforming to the REST constraints
- alternative to “heavyweight” SOAP-based WS

RESTful WS vs SOAP-based WS

- SOAP-based WS
  - protocol/W3C standard
  - tied to XML (message format)
  - envelope format (tunneling over transport protocols)
  - various transport protocols (HTTP, SMTP, JMS, ...)
- RESTful WS
  - not standardized
  - any representational (message) format: XML, JSON, YAML, ...
  - tied to HTTP

RESTful WS use-cases

from the enterprise developer point of view

- replacement for SOAP-based WS
  - simpler to develop, test and use
  - not tied to XML
- and endpoints for HTML5 and mobile clients
  - unified API

RESTful WS downsides

from the enterprise developer point of view

- security - there are no standards
  - e.g. authentication & authorization
    - session state violates REST constraints and complicates scalability
    - HTTP authentication is useless
    - OAuth is web-centric and also quite complicated

RESTful WS...

- application is a collection of resources
- each resource has its:
  - identifier (URI)
  - representation (HTML, JSON, XML, …)
  - set of allowed HTTP operations (GET, PUT, POST, DELETE, ...)

In JAX-RS...

- resource is POJO
  - new instance is created for each request by default
    - an implementation may offer other lifecycles
    - Java EE - resource may be EJB (stateless or singleton session bean) or CDI bean (in CDI app)
  - has resource methods annotated with a request method designator (`@GET`, `@POST`, …) to handle the request

JAX-RS request matching

- identify and obtain the resource object method that will handle the request
  - URI (`@Path`)
  - HTTP method (e.g. `@GET`)
  - media types
    - `@Consumes`
      - media types resource method consumes
      - matching HTTP request header `Content-type`
    - `@Produces`
      - media types resource method produces
      - matching HTTP request header `Accept`

JAX-RS providers

- way of extending JAX-RS runtime
  - Entity Providers
    - supply mapping between representations and associated Java types (e.g. JSON provider)
  - Context Providers
    - supply context to resource classes and other providers
  - Exception Mapping Providers
    - map an exception to an instance of Response

RESTEasy extras

- client framework
- caching features
- interceptors
- rich set of providers: XML, JSON, YAML, Fastinfoset, Multipart, XOP, Atom, ...
- GZIP compression/decompression
- Seam 2, Guice, Spring integration
- ...

Time for example...

Some useful links

- JSR 311: JAX-RS
- RESTEasy project
- RESTful Web services: The basics
- REST Anti-Patterns
- JBoss TicketMonster Tutorial - Business Logic
- REST with Java using Jersey - Tutorial

That's all...

Thanks for listening