Developing JAX-RS Web Application Utilizing Server-sent Events and WebSocket

Arun Gupta, Java EE & GlassFish Guy
blogs.oracle.com/arungupta, @arungupta

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Program Agenda

- About This Lab
- Quick Intro to the Used Technologies
- Lab Exercises
- Getting Started
- Resources

About this lab

- Maven 3.0.x
- GlassFish 4.0 b57
- JDK 7u11
- NetBeans 7.2.x
  - Follow Appendix in lab-guide.pdf to register GlassFish in NetBeans

About This Lab

- DO
  - Follow the lab guide
  - Exercises are self-paced
  - Raise your hand if you get stuck – we are here to help
- DON’T
  - Just blindly copy-paste

Technologies Used in this Lab

Quick Intro

- Jersey/JAX-RS 2.0
  - Server-sent events
- Tyrus/Java API for WebSocket
- JSON Processing

JAX-RS 2.0/Jersey

Description

- Java API for RESTful Web Services
  - Annotation-based API for exposing RESTful web services
  - Maps HTTP requests to Java methods
- New in JAX-RS 2.0
  - Client API
  - Filters/ Entity Intereptors
  - Server-side content negotiation
  - Asynchronous processing

JAX-RS 2.0/Jersey

Client API

```
// Get instance of Client
Client client = ClientFactory.newClient();

// Get account balance
String bal = client.target("http://.../atm/{cardId}/balance")
    .pathParam("cardId", "111122223333")
    .queryParam("pin", "9876")
    .request("text/plain").get(String.class);
```

JAX-RS 2.0/Jersey

Where to get more info

- On the web:
  - Specification project: http://jax-rs-spec.java.net
  - Implementation project: http://jersey.java.net
  - Twitter: @gf_jersey

Java API for Web Socket

Description

- Annotation-based API for utilizing Web Socket protocol in Java web applications
- Allows defining Web Socket client and endpoints
  - Define lifecycle and message callbacks
  - Bi-directional communication between peers
- Support for encoders/decoders to map message content to/from Java objects

Java API for Web Socket

Example – Simple Endpoint

```
@WebSocketEndpoint("/echo")
public class EchoBean {
    @WebSocketMessage
    public String echo(String message) {
        System.out.println("Message received: " + message);
        return message + " (from your server)";
    }
}
```


Java API for Web Socket

Example – Decoder/Encoder

```
@WebSocketEndpoint(”/drawing/”,
    decoders = ShapeCoding.class, encoders = ShapeCoding.class,
)
public class DrawingWebSocket {
    @WebSocketMessage
    public void shapeCreated(Shape shape, Session session) { … }
}

public class ShapeCoding implements Decoder.Text<Shape>, Encoder.Text<Shape> {
    public Shape decode(String s) throws DecodeException { … }
    public boolean willDecode(String s) { … }
    public String encode(Shape object) throws EncodeException { … }
}
```

Java API for Web Socket

Where to get more information

- On The Web
  - Specification Project: http://websocket-spec.java.net
  - Implementation: http://tyrus.java.net

Standard JSON API

Contents

- Parsing/Processing JSON
- Data binding : JSON text <-> Java Objects
- Two JSRs (similar to JAXP and JAXB)
  - Processing/Parsing (JSON-P) – Java EE 7
  - Binding (JSON-B) – Java EE 8

Java API for Processing JSON

JSR-353

- Streaming API to produce/consume JSON
  - Similar to StAX API in XML world
- Object model API to represent JSON
  - Similar to DOM API in XML world

JSR-353: Java API for Processing JSON

JsonReader/JsonWriter

- JsonReader – reads JsonObject/JsonArray from i/o

```
try(JsonReader reader = new JsonReader(io)) {
    JsonObject jsonObj = reader.readObject();
}
```

- JsonWriter – writes JsonObject/JsonArray to i/o

```
try(JsonWriter writer = new JsonWriter(io)) {
    writer.writeObject(jsonObj);
}
```

Resources

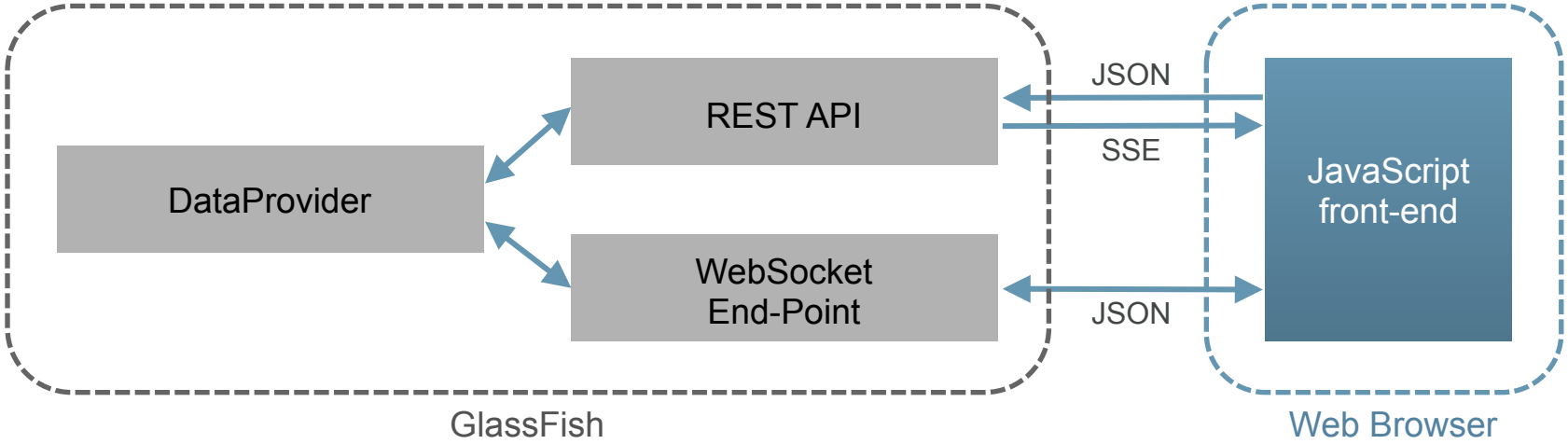
- Projects
  - Specification Project - http://json-processing-spec.java.net
  - RI Project - http://jsonp.java.net
- Latest Javadoc
  - http://json-processing-spec.java.net/nonav/releases/1.0/edr/javadocs/index.html

Lab Exercises

- Drawing Board web application:
  - Exercise 1: Exposing RESTful API
  - Exercise 2: Adding Server-Sent Events
  - Exercise 3: Adding Web Sockets
- Simple Drawing Board client:
  - Exercise 4: Implementing a Simple Java Client

Drawing Board Application

High-Level Overview



Getting Started

- Open lab-guide.pdf
- Follow the instructions
  - Follow Appendix in lab-guide.pdf to register GlassFish in NetBeans