Web Services Support in Java SE 6

Sang Shin
Michèle Garoche
www.javapassion.com
"Learning is fun!"

Agenda

- Web Services Support on Java SE 6 Platform
- API Support
- Server-Side Programming Model
- Client-Side Programming Model

Web Services Support on Java SE 6 Platform

Web Services Support on Java SE 6 Platform

- JAX-WS
- Data binding using JAXB 2.0
- Updates to the JAXP, which includes StaX
- Standards supported
  - SOAP 1.2
  - WS-I Basic Profile 1.1
  - XML-binary Optimized Packaging (XOP) and SOAP Message Transmission Optimization Mechanism (MTOM)
  - Representational State Transfer (REST)
  - Totally on XML schema

API Support

API Support

- Java SE 6 provides support for the JAX- WS web services stack.
  - For the client side: Service class for creating proxy
  - For the server side: Endpoint class for publication

Server-Side Programming Model

Server-Side Programming Model

1. Write a Plain Old Java Object (POJO) implementing the service.
2. Add **`@WebService`** to it.
3. Optionally, inject a **`WebServiceContext`**
4. Publish the Web service endpoint through **`Endpoint.publish()`** method
   - WSDL is automatically generated at runtime
5. Point your clients at the Web Services Description Language (WSDL), for example:
   - **`http://myserver/myapp/MyService?WSDL`**.

Server-Side Programming Model

1. Write a Plain Old Java Object (POJO) implementing the service.
2. Add **`@WebService`** to it.
3. Optionally, inject a **`WebServiceContext`**
4. Publish the Web service endpoint through **`Endpoint.publish()`** method
   - WSDL is automatically generated at runtime
5. Point your clients at the Web Services Description Language (WSDL), for example:
   - **`http://myserver/myapp/MyService?WSDL`**.

Publishing Endpoint

- The **`publish`** methods can be used to start publishing an endpoint, at which point it starts accepting incoming requests.
- The **`stop`** method can be used to stop accepting incoming requests and take the endpoint down
- Publish using the HTTP server embedded in Java SE 6.
- Supports reasonable defaults for threading.
- Creates WSDL and publishes it at runtime:

```
> http://localhost/calculator?WSDL
```

Publishing an Endpoint

```
@WebService
public class Calculator {
    @Resource
    WebServiceContext context;
    public int add(int a, int b) {
        return a+b;
    }
}
// Create and publish an endpoint
Calculator calculator = new Calculator();
Endpoint endpoint = Endpoint.publish
      ("http://localhost/calculator",calculator);
```

Client-Side Programming Model

Client-Side Programming Model

1. Point a tool at the WSDL for the service
2. Generate annotated classes and interfaces through a tool
3. Call new on the service class.
4. Get a proxy using a **`getxxxPort`** method.
5. Invoke any remote operations.

Example: Java SE-based Client

```
// Create a Service object
CalculatorService svc = new
    CalculatorService();

// Create a proxy from the Service object
Calculator proxy =
    svc.getCalculatorPort();

// Invoke a Web service operation
int answer = proxy.add(35, 7);
```

Thank you!

Sang Shin
Michèle Garoche
www.javapassion.com
“Learning is fun!”