Apache CXF and Spring

anova r&d bvba

Planning

- Introduction
- JAX-WS
- JAX-RS

Introduction

- About CXF
- CXF Terminology
- Setting up Spring and CXF

About CXF

- Apache License
  - business friendly open-source license
- History
  - ObjectWeb's Celtix
  - Codehaus' XFire

About CXF

- Features and highlights:
  - web service standards (SOAP, WSDL, WS-*, ...)
  - java standards: JAX-WS, JAX-RS interoperates with SAAJ, JAXB, JBI, SCA, ...
  - support for transports and bindings
  - support for POJO and Spring
  - tooling: WSDL Validator, code generator
  - ease of use and flexible deployment

CXF Terminology

- Bus
- Transport
- Data binding
- Interceptor
- Feature

Bus

- 'Glue' to hold everything together:
  - extensions
  - features
  - interceptor providers
- Configuration done here is active for all services

Transport

- Multiple transport options
  - HTTP
    - Asynchronous HTTP
    - Client HTTP
    - Jetty
    - Servlet
  - JMS
  - Local
  - UDP
  - Websocket

Data binding

- Default is JAXB
- Support for:
  - Aegis
  - SDO
  - XmlBeans
- MTOM

Interceptor

- Fundamental processing unit in CXF
  - Lot of functionality is configured through interceptors
  - Add your own when necessary
- Collected in `InterceptorChain`s
- Phases
  - `RECEIVE`, `READ`, `UNMARSHAL`, `INVOKE`, ...
  - `LOGICAL`, `WRITE`, `MARSHAL`, `STREAM`, ...

Feature

- Add capabilities to bus/server/client
- Examples:
  - addressing
  - failover
  - logging
  - policies
  - reliableMessaging

```
<cxf:bus>
    <cxf:features>
        <cxf:logging/>
    </cxf:features>
</cxf:bus>
```

Setting up Spring and CXF

- beans.xml : add namespace and import

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xmlns:jaxrs="http://cxf.apache.org/jaxrs"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
         http://cxf.apache.org/jaxws
           http://cxf.apache.org/schemas/jaxws.xsd
         http://cxf.apache.org/jaxrs
           http://cxf.apache.org/schemas/jaxrs.xsd">

    <!-- add beans and CXF endpoints here -->

</beans>
```

Setting up Spring and CXF

- web.xml : Spring configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         version="2.5"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="
               http://java.sun.com/xml/ns/javaee
                 http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>WEB-INF/beans.xml</param-value>
    </context-param>

    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>

    <!-- next slide bits go here -->

</web-app>
```

Setting up Spring and CXF

- web.xml : CXF configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         version="2.5" ... >

    <!-- previous slide bits go here -->

    <servlet>
        <servlet-name>CXFServlet</servlet-name>
        <display-name>CXF Servlet</display-name>
        <servlet-class>
            org.apache.cxf.transport.servlet.CXFServlet
        </servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>CXFServlet</servlet-name>
        <url-pattern>/*</url-pattern>
    </servlet-mapping>

</web-app>
```

JAX-WS

- About JAX-WS
- Service implementation
- Configuring service
- Configuring client

About JAX-WS

- Java API for XML Web Services
  - API and annotations
  - developing web service endpoints and clients
  - reference implementation: Glassfish's Metro project
  - Apache CXF is TCK compliant

Service implementation

- Annotations
  - Web Services Metadata
  - JAX-WS
  - JAXB
- Service Endpoint Interface
- Service Endpoint Implementation

Service implementation

- Web Services Metadata Annotations (`javax.jws`)
  - `@WebService`
  - `@WebMethod`
  - `@OneWay`
  - `@WebParam`
  - `@WebResult`
  - `@HandlerChain`
  - `@SOAPBinding`

Service implementation

- JAX-WS annotation (`javax.xml.ws`)
  - `@BindingType`
  - `@RequestWrapper` and `@ResponseWrapper`
  - `@ServiceMode`
  - `@WebEndpoint`
  - `@WebFault`
  - `@WebServiceClient`
  - `@WebServiceProvider`
  - `@WebServiceRef`

Service implementation

- JAXB annotations (`javax.xml.bind.annotations`)
  - `@XmlRootElement`
  - `@XmlAccessorOrder`
  - `@XmlAccessorType`
  - `@XmlType`
  - `@XmlElement`
  - `@XmlElements`
  - `@XmlList`
  - `@XmlAttribute`
  - ... (a lot more here)

Service implementation

- Service Endpoint Interface
  - Define a POJO interface
  - Annotate with `@WebService`

```
@WebService
public interface ForestService {

    public String getForestName();

    public Tree[] getTrees();

    public Forest getForest();

}
```

Service implementation

- Service Endpoint Implementation
  - Implements the SEI
  - Annotate with `@WebService`

```
@WebService(endpointInterface = "be.anova.course.cxf.service.ForestService")
public class ForestServiceImpl implements ForestService {

    public ForestServiceImpl() {
        super();
    }

    // implement the interface methods here

}
```

Configuring the service

- Use the `<jaxws:endpoint/>` element
  - Refer to implementation class or bean
  - Inner elements to fine-tune configuration

Configuring the service

- Example with class

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
         http://cxf.apache.org/jaxws
           http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:endpoint id="forestService"
                    implementor="some.service.impl.ForestServiceImpl"
                    address="/Forest" />

</beans>
```

Configuring the service

- Example with bean reference

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
         http://cxf.apache.org/jaxws
           http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:endpoint id="forestService"
                    implementor="#service"
                    address="/Forest"/>

    <bean id="service" class="some.service.impl.ForestServiceImpl">
        <property name="forest" ref="forest"/>
    </bean>

</beans>
```

Configuring the client

- Using the `<jaxws:client/>` element

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
         http://cxf.apache.org/jaxws
           http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:client id="client"
                  serviceClass="some.service.ForestService"
                  address="http://localhost:8080/Forest" />

</beans>
```

Configuring the client

- Using the `<jaxws:client/>` element

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
         http://cxf.apache.org/jaxws
           http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:client id="client"
                  serviceClass="some.service.ForestService"
                  address="http://localhost:8080/Forest" />

</beans>
```

Configuring the client

- Using the `JaxWsProxyFactoryBean`

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="client" class="some.service.ForestService"
          factory-bean="clientFactory" factory-method="create"/>

    <bean id="clientFactory"
          class="org.apache.cxf.jaxws.JaxWsProxyFactoryBean">
      <property name="serviceClass" value="some.service.ForestService"/>
      <property name="address" value="http://localhost:8080/Forest"/>
    </bean>

</beans>
```

What else is there?

- `Provider` and `Dispatch`
- Tooling to generate code from WSDLs
  - Command-line tool
  - `cxf-codegen-plugin`

JAX-RS

- About JAX-RS
- Annotations
- Service implementation

About JAX-RS

- Java API for RESTful Web Services
  - API and annotations
  - reference implementation: Jersey project
  - Apache CXF is TCK compliant

Annotations

- Annotations
  - HTTP methods and paths
  - Parameters

Annotations

- `@Path` to configure service path
- HTTP methods annotations
  - `@GET`
  - `@PUT`
  - `@POST`
  - `@DELETE`
  - `@HEAD`

Annotations

- Parameter annotations
  - `@PathParam`
  - `@QueryParam`
  - `@MatrixParam`
  - `@HeaderParam`
  - `@CookieParam`
  - `@FormParam`
  - `@DefaultValue`

Annotations

- Media types
  - `@Produces` and `@Consumes`
  - `application/xml` for XML
  - `application/json` for JSON

Service implementation

- Example class and configuration

```
@Path("/forest")
@Produces({"application/xml", "application/json"})
public class ForestService {

    @GET
    public Forest getForest() {
        // some code here
    }

    @GET
    @Path("/tree/{id}")
    public Tree getTree(@PathParam("id") int id) {
        // some code here
    }

    @POST
    @Consumes("application/xml")
    public Response update(Forest forest) {
        // some code here
        return Response.ok().build();
    }
}
```

Service implementation

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:jaxrs="http://cxf.apache.org/jaxrs"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
          http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-beans.xsd
          http://cxf.apache.org/jaxrs
            http://cxf.apache.org/schemas/jaxrs.xsd">

    <jaxrs:server id="forestService" address="/">
        <jaxrs:serviceBeans>
            <bean class="some.rs.ForestService" />
        </jaxrs:serviceBeans>
        <jaxrs:extensionMappings>
            <entry key="json" value="application/json"/>
            <entry key="xml" value="application/xml"/>
        </jaxrs:extensionMappings>
    </jaxrs:server>

</beans>
```

Exercise

- "Bring your own business model"
- Try building a JAX-RS and JAX-WS service for your own business model.
  Feel free to experiment and ask hard questions ;)