Auszug aus JAX-WS Folien

Dieses Dokument ist ein Auszug aus unserem Skript zur Java Web Services Schulung. Es dient lediglich als Beispiel für unsere Kursunterlagen.

Mehr zum Kurs finden Sie unter:
http://www.thomas-bayer.com/java-webservices-schulung.htm

Thomas Bayer

Hauptstraße 33
75050 Gemmingen

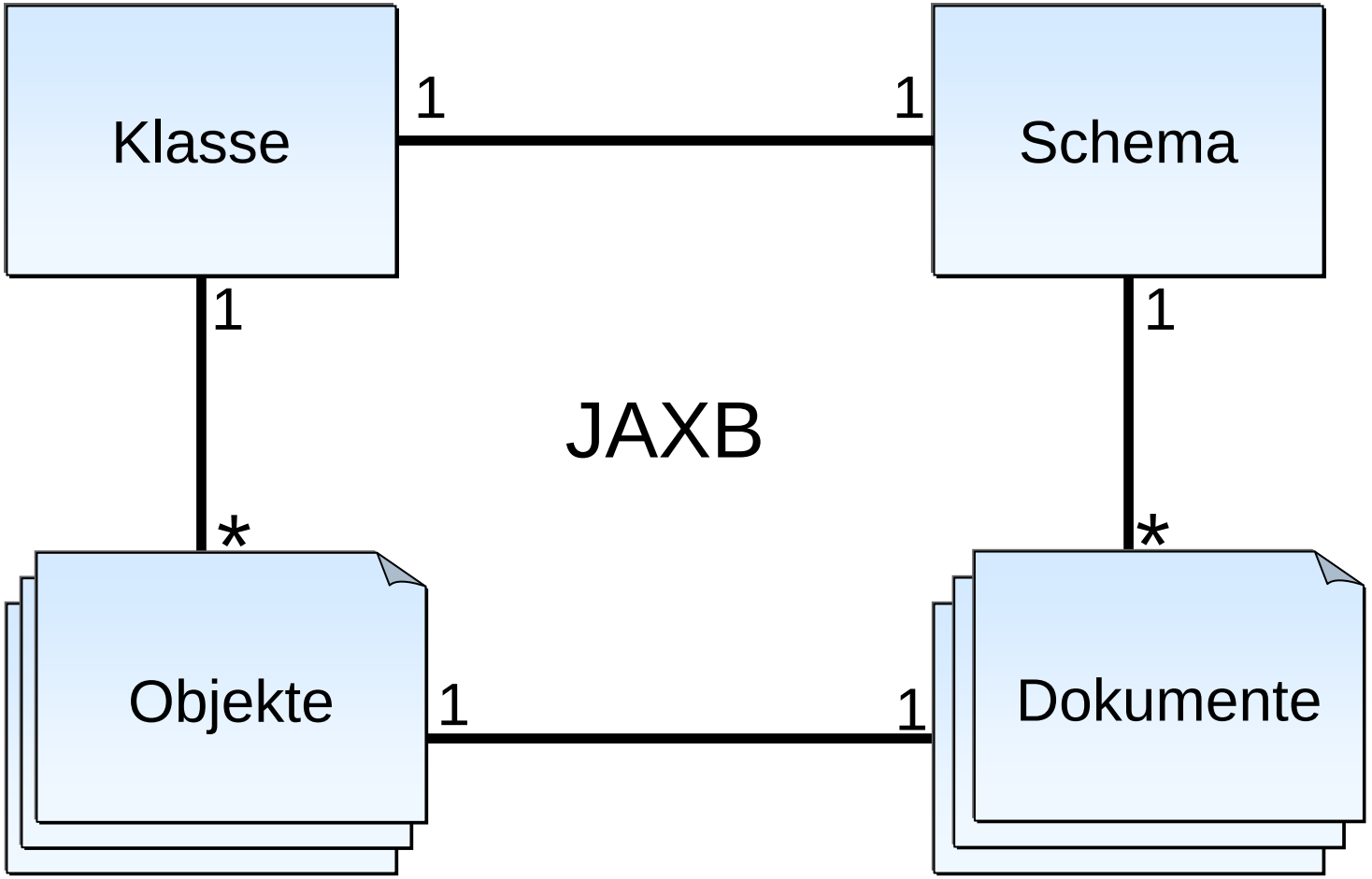
www.thomas-bayer.de
info@thomas-bayer.de

Version: 1.0

JAX-RPC versus JAX-WS

JAX-RPC	JAX-WS
Eigenes Data-Binding	JAXB
-	Support für Annotations
Limitiert auf RPC	RPC und Messaging
Fokus auf Interface	Fokus auf Message

XML / Java Binding



WSDL -> Java Mapping

definitions/@targetNamespace -> package (JAXB)

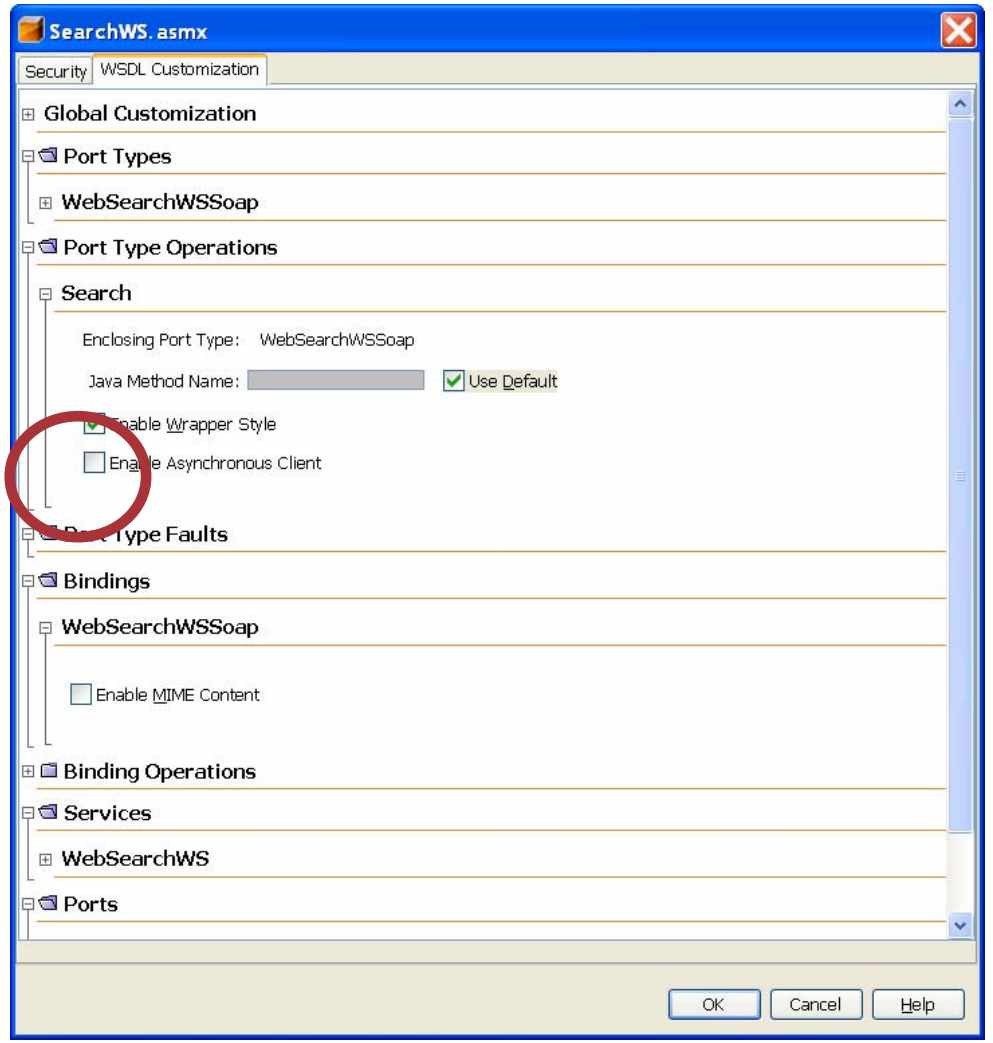
portType -> Interface (SEI) mit @WebService Annotations

portType/operation -> Methode im SEI mit @WebMethod

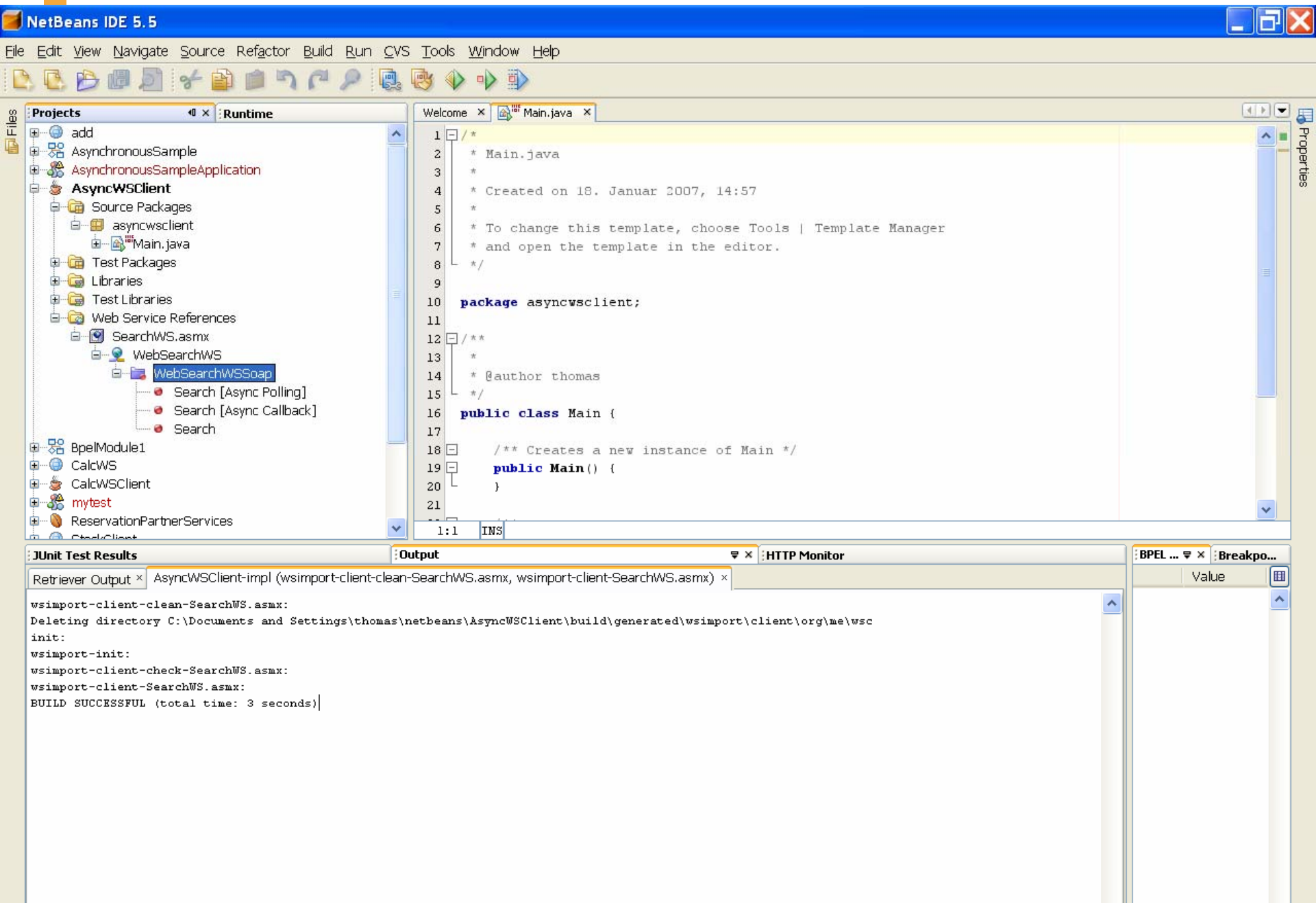
fault -> Exception @WebFault

Asynchrone Clients

- Polling oder Callback



Asynchrone Clients in NetBeans



Client mit Callback Handler

```
WebSearchWS service = new WebSearchWS();
WebSearchWSSoap port = service.getWebSearchWSSoap();
String keyWord = "bpel";

AsyncHandler<SearchResponse> handler = new AsyncHandler<SearchResponse>() {
    public void handleResponse(Response<SearchResponse> response) {
        System.out.println("Result = "+ response.get().getSearchResult());
    }
}

Future<? extends Object> result = port.searchAsync(keyWord, asyncHandler);
while(!result.isDone()) {
    Thread.sleep(100);
}
```

Asynchroner Client mit Polling

```
javax.xml.ws.Response<AddResponse> resp;
resp = port.addAsync(i, j);

while(!resp.isDone()) {
    Thread.sleep(100);
}
```

Provider

- Alternative zu SEI
- Arbeitet mit Payload oder der Nachricht selbst
- Interface javax.xml.ws.Provider

```
@WebServiceProvider
@ServiceMode(value=Service.Mode.MESSAGE)
public class MyService implements Provider<SOAPMessage>
{

    public SOAPMessage invoke(SOAPMessage request) {
        return request;
    }
}
```

WebServiceContext

- Zugriff auf WebServiceContext von einer Serviceimplementierung über DI

```
@WebService()
public class CalculatorWS {

    @Resource
    private WebServiceContext ctx;

    @WebMethod
    public int add(@WebParam(name = "i") int i,
                   @WebParam(name = "j") int j) {
        System.out.println("Ctx:" + ctx);
        return i + j;
    }

}
```

JAX-WS Handler Typen

- Logical
  - Zugriff auf Message Context Properties und Payload
  - Implementieren javax.xml.ws.handler.LogicalHandler
- Protocol
  - Zugriff auf MessageContext Properties und Protocol spezifische Nachrichten
- Sind Protokoll spezifisch z.B. für HTTP
- Implementieren javax.xml.wshandler.Handler

@javax.xml.ws.RequestWrapper

Target: Methoden eines SEI

Beeinflußt das von JAXB generierte Wrapper Bean.

Property	Beschreibung	Default
localName targetNamespace className	NameXMLElement NS des Elementes Name der Wrapper Klasse	-

@javax.xml.ws.WebServiceClient

Wird bei generierten Client Klassen verwendet

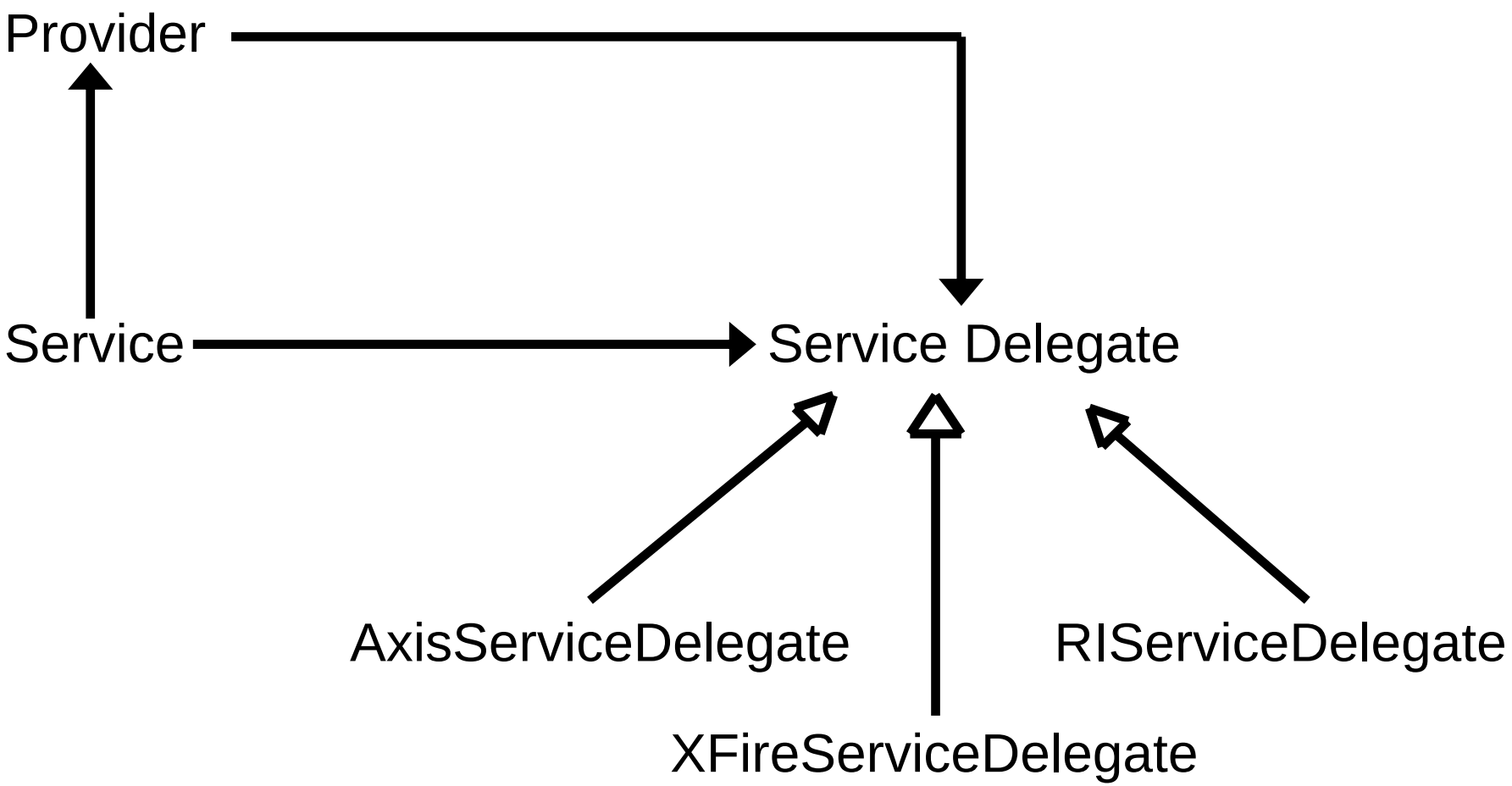
Property	Beschreibung	Default
name	Name des Services	-
targetNamespace	NS des Services	-
WsdlLocation	URL der WSDL Beschreibung	-

@javax.xml.ws.WebService Provider

Target: Klassen, die javax.xml.ws. Provider implementieren

Property	Beschreibung	Default
wsdlLocation	URL der WSDL	-
serviceName	Name des Services	-
portName	Name des Ports	-
targetNamespace	Targetnamespace des Services	-

SPI



Provider Lookup

1.) META-INF/Services/javax.xml.ws.spi.Provider
   - Name in erster Zeile
2.) Property javax.xml.ws.spi.Provider in $JAVA_HOME/lib/jaxws.properties
3.) System Property javax.xml.ws.spi.Provider
4.) Default Implementation