# Desenvolvendo Web Services utilizando JAX-WS

## Aprenda neste artigo a desenvolver web services com JAX-WS.

---

Java API para Serviços Web XML (JAX-WS) 2.0, JSR 224, é uma parte importante da plataforma Java EE 5. Uma continuação liberada de Java API para XML-based RPC 1.1 (JAX-RPC), JAX-WS simplifica a tarefa de desenvolver serviços web utilizando tecnologia Java. Encaminha alguns dos resultados em JAX-RCP 1.1 providenciando suporte para protocolos múltiplos assim como SOAP 1.1, SOAP 1.2, XML e suprindo uma facilidade de manutenção para protocolos adicionais junto com HTTP. JAX-WS utiliza JAXB 2.0 para ligação de dados e suportes habituais para controlar serviços gerados em interfaces endpoint. Com seu apoio para anotações, JAX-WS simplifica o desenvolvimento de serviços web e diminui o tamanho do tempo de execução dos arquivos Jar.

Um exemplo acompanha este artigo. Este exemplo demonstra um serviço web simples que é acessado utilizando JAX-WS 2.0 através de uma aplicação Java externa. O exemplo é baseado em um recurso aberto de implementação do Java EE 5 SDK chamado GlassFish. O pacote do exemplo inclui o código fonte para o exemplo, scripts de configuração e um arquivo de configuração ant.

Comecemos por fazer algumas instalações iniciais. Se você ainda não fez, baixe o GlassFish da página de Downloads GlassFish. (Este artigo foi testada com o Build 29 do GlassFish.) Depois prepare as seguintes variáveis de ambiente:

· GLASSFISH_HOME. Deve indicar onde você instalou o GlassFish (por exemplo C:\Sun\AppServer)

· ANT_HOME. Isto indicara onde foi instalado o ant. Ant está incluído no pacote do GlassFish que você baixou. (No Windows, está no subdiretório lib\ant.) Você também pode baixar o Ant da página do Projeto Apache Ant. O exemplo requer do Apache Ant 1.6.5.

· JAVA_HOME. Isto indicará a localização do JDK 5.0 no seu sistema.

Adicione também a localização do Ant na sua variável do ambiente PATH.

Em seguida baixe o pacote do exemplo e extraia o conteúdo dele. O diretório principal para está dica é jaxws-techtip.

**Configurando o Web Service**

Com as configurações iniciais feitas, podemos começar a configurar o serviço web. Neste exemplo, o web service é desenvolvido a partir de uma classe Java. Para configurar o web service:

· Escreva uma classe de implementação endpoint.

· Compile a classe de implementação endpoint.

· Opcionalmente gere artefatos portáteis necessários para execução do web service.

· Empacote o web service num arquivo WAR e intale-o.

Escreva uma classe de implementação endpoint.

Se você navegar através da estrutura do diretório \techtip, você encontrará um diretório endpoint. Neste diretório, você achara uma classe chamada Calculator. A classe é uma implementação endpoint de um serviço simples que adiciona dois inteiros:

```
package endpoint;


  import javax.jws.WebService;
  import javax.jws.WebMethod;

  @WebService(
     name="Calculator",
     serviceName="CalculatorService",
     targetNamespace="http://techtip.com/jaxws/sample"
  )
  public class Calculator {
        public Calculator() {}

        @WebMethod(operationName="add", action="urn:Add")
        public int add(int i, int j) {
              int k = i +j ;
              System.out.println(i + "+" + j +" = " + k);

              return k;
        }
  }
```

JAX-WS 2.0 confia fortemente no uso de anotações como especificado na A Metadata Facility for the Java Programming Language (JSR 175) e Web Services Metadata for the Java Platform (JSR 181), assim como anotações adicionais definidas por especificações do JAX-WS 2.0.

Observe as duas anotações no tipo Calculator: @WebService e @WebMethod. Um tipo valido de implementação endpoint deve incluir uma anotação @WebService. À anotação marca a classe como serviço web.

O valor de propriedade do nome na anotação @WebService identifica o Web Service Description Language (WSDL) portType (neste caso, “Calculator”). O serviceName (“CalculatorService”) é um serviço WSDL. O TargetNamespace especifica o namespace XML utilizado para o WSDL. Todas as propriedades são opcionais. Para detalhes dos valores padrões destas propriedades, veja a seção 4.1 das especificações Web Services Metadata for the Java Platform, JSR 181.

À anotação @WebMethod expõe um método como método de web service. O valor de propriedade do operationName na anotação do tipo Calculator identifica uma operação WSDL (neste caso, adiciona), e a ação do valor de propriedade ("urn:Add") especifica um namespace XML para o WSDL dos elementos gerados a partir da operação do web service. Ambas operações são opcionais. Se você não os especifica, o valor da operação do WSDL muda para methodname, e o valor da ação muda para targetNamespace do serviço.

Compile a classe de implementação.

Após codificar a classe de implementação, você precisa compilá-lo. Inicie o GlassFish inserindo o seguinte comando:

```
\bin\asadmin start-domain domain1
```

onde diz é o diretório onde você instalou o GlassFish. Depois navegue até a pasta jaxws-techtip na estrutura do diretório \jaxws, e execute o seguinte comando.

```
ant compile
```

Executando o comando é o equivalente a executar o seguinte comando Java (em uma linha):

```
javac -classpath $GLASSFISH_HOME/lib/javaee.jar -d
./build/classes/service/ endpoint/Calculator.java
```

Gera artefatos portáteis para à execução do web service

Este passo é opcional. O ferramenta expansível gera automaticamente estes artefatos se eles não são empacotados com a unidade de serviço expansível durante o serviço web. Entretanto se você quer gerar estes artefatos manualmente, execute o seguinte comando:

```
ant generate-runtime-artifacts
```

Isto cria um diretório build/generated abaixo de jaxws-techtip, e executa o seguinte comando wsgen (em uma linha):

```
$GLASSFISH_HOME/bin/wsgen -cp ./build/classes/service
-keep -d ./build/classes/service
-r ./build/generated -wsdl endpoint.Calculator
```

Um arquivo WSDL (CalculatorService.wsdl) é criado no diretório build/generated, junto com um arquivo schema (CalculatorService_schema1.xsd), que define o diagrama para CalculatorService.wsdl.

Os componentes da tecnologia JavaBean (JavaBeans) ajudam a oficializar os métodos de invocação, respostas e exceções especificas de serviço. Estas classes são utilizadas durante a execução do serviço web em uma aplicação no servidor. As classes de JavaBean são geradas no diretório /build/classes/service/endpoint/jaxws abaixo de jaxws-techtip. As classes são:

```
Add.java
Add.class
AddResponse.java
AddResponse.class
```

**Empacotando e instalando o arquivo WAR**

Em seguida você precisa empacotar e instalar o serviço. Para fazer isso, você precisa especificar os detalhes sobre o serviço na instalação dos descritores(deployment descriptors). Os serviços web podem ser empacotados como servlet ou como uma seção stateless bean. Serviços web empacotados como servlets são agrupados em formato de Arquivos Web (WAR). Neste artigo, o serviço é empacotado como um servlet.

Para empacotar o serviço como um arquivo WAR, navegue até a pasta jaxws-techtip e execute o seguinte comando:

```
ant pkg-war
```

Para a estrutura do arquivo war, você pode dar uma olhada em pkg-war no arquivo build.xml.

Você pode expandir o arquivo war gerado executando o seguinte comando:

```
ant deploy-app
```

Isto é o equivalente a executar o comando de instalação asadmin (em uma linha):

```
bash$GLASSFISH_HOME/bin/asadmin deploy --user admin
--passwordfile passwd --host localhost --port 4848
--contextroot jaxws-webservice --upload=true --target server
```

**Configurando o Cliente**

Após ter expandido o serviço web, você pode acessar desde um programa cliente. A seguir os passos para configurar o cliente:

1. Escreva o cliente.
2. Gerar artefatos portáteis requeridos para compilar o cliente.
3. Compile o cliente.
4. Execute o cliente.

**Escreva o cliente**

O seguinte programa, JAXWSCliente, é um programa de cliente autônomo providenciado com o código exemplo para este artigo. Isto invoca uma operação de adição no serviço instalado 10 vezes, adicionando 10 números de 0 a 9.

```
package client;

import javax.xml.ws.WebServiceRef;
import com.techtip.jaxws.sample.CalculatorService;
import com.techtip.jaxws.sample.Calculator;

public class JAXWSClient {
   @WebServiceRef(wsdlLocation=
    "http://localhost:8080/jaxws-webservice/CalculatorService?WSDL")

    static CalculatorService service;

   public static void main(String[] args) {
      try {
       JAXWSClient client = new JAXWSClient();
       client.doTest(args);
      } catch(Exception e) {
       e.printStackTrace();
      }
   }

   public void doTest(String[] args) {
     try {
       System.out.println(
          " Retrieving port from the service " + service);
       Calculator port = service.getCalculatorPort();
       System.out.println(
          " Invoking add operation on the calculator port");
       for (int i=0;i>10;i++) {
```

```
        int ret = port.add(i, 10);
        if(ret != (i + 10)) {
          System.out.println("Unexpected greeting " + ret);
          return;
        }
        System.out.println(
          " Adding : " + i + " + 10 = "  + ret);
        }
      } catch(Exception e) {
        e.printStackTrace();
      }
    }
  }
```

A anotação @WebServiceRef em JAXWSCliente é utilizada para declarar uma referência a um serviço web. O valor do parâmetro wsdlLocation em @WebServiceRef é um apontador URL para a localização do arquivo WSDL para o referido serviço. As anotações @WebServiceRef suportam propriedades opcionais adicionais, como especificadas na seção 7.9 do JSR 224.  A variável estática do serviço nomeado será injetada pelo recipiente da aplicação do cliente.

Observe as indicações de importação em JAXWSClient: com.techtip.jaxws.sample.CalculatorService e com.techtip.jaxws.sample.Calculator.  Estas importações são para os artefatos portáteis que serão gerados no próximo passo. CalculatorService é o artefato portátil para à implementação do serviço. Calculator é a interface Java para o serviço endpoint gerado desde o WSDL identificado pela propriedade wsdlLocation do @WebServiceRef.

O cliente recupera o Calculator endpoint do ClaculatorService através do método getWebServiceRefNamePort, onde WebServiceRefName é o nome da propriedade do @WebServiceRef, ou o valor do porto WSDP no arquivo gerado WSDL. Após a recuperação do endpoint, o cliente invoca a operação de adição solicitada dez vezes.

**Gerando artefatos portáteis para o cliente**

Como mencionado anteriormente, CalculatorService e Calculator são artefatos portáveis. Para gerar todos os artefatos portáveis para o cliente, navegue até a pasta jaxws-techtip e edite o seguinte comando:

```
ant generate-client-artifacts
```

Isto é o equivalente a executar o seguinte comando wsimport (em uma linha):

```
$GLASSFISH_HOME/bin/wsimport -keep  -d ./build/classes/client
http://localhost:8080/jaxws-webservice/CalculatorService?WSDL
```

Isto gera o artefato no diretório build/classes/com/techtip/jaxws/sample abaixo de jaxws-techtip. Os artefatos são:

```
Add.java
Add.class
AddResponse.java
AddResponse.class
Calculator.java
Calculator.class
CalculatorService.java
CalculatorService.class
package-info.java
package-info.class
ObjectFactory.class
ObjectFactory.java
```

**Compile o cliente**

O seguinte passo é para compilar o tipo de cliente. Você pode fazer isso inserindo o seguinte comando:

```
ant compile-client
```

A tarefa de compilação do ant compila cliente/JAXWSClient e escreve o tipo de arquivo dopara configurar o subdiretório /classes/client. Isto é o equivalente a executar o seguinte comando (em uma linha):

```
javac -d ./build/classes/client
-classpath $GLASSFISH_HOME/lib/javaee.jar:
$GLASSFISH_HOME/lib/appserv-ws.jar:
./build/classes/client client/JAXWSClient.java
```

**Execute o cliente**

Para ver se o exemplo funciona, execute o seguinte comando:

```
ant runtest-jaxws
```

o que é o equivalente a executar o seguinte comando executado a partir da pasta build/classes/cliente:

```
$GLASSFISH_HOME/bin/appclient -mainclass client.JAXWSClient
```

Você deve ver uma saída similar à seguinte:

```
runtest-jaxws:
    [echo] Executing appclient with client class as
    client.JAXWSClient
    [exec]  Retrieving port from the service
    com.techtip.jaxws.sample.CalculatorService@162522b
    [exec]  Invoking add operation on the calculator port
    [exec]  Adding : 0 + 10 = 10
    [exec]  Adding : 1 + 10 = 11
    [exec]  Adding : 2 + 10 = 12
    [exec]  Adding : 3 + 10 = 13
    [exec]  Adding : 4 + 10 = 14
    [exec]  Adding : 5 + 10 = 15
    [exec]  Adding : 6 + 10 = 16
    [exec]  Adding : 7 + 10 = 17
    [exec]  Adding : 8 + 10 = 18
    [exec]  Adding : 9 + 10 = 19    all:
```

BUILD SUCCESSFUL

Total time: 6 secondsFonte:

Manisha Umbar - http://www.java-tips.org/java-ee-tips/java-api-for-xml-web-services/developing-web-services-using-j.html

por Eduardo Oliveira

Expert em Java e programação Web