**JAX-RPC**

# JAX-RPC

Java API for XML-based RPC (JAX-RPC) 规范使您能够开发基于 SOAP 的可互操作并且可移植的 Web Service 和 Web Service 客户机。JAX-RPC 1.1 为在 Java 平台上开发和部署 Web Service 提供了核心 API，并且它是 Java Platform, Enterprise Edition (Java EE) 平台的 Web Service 的部件。Java EE 平台允许您开发可移植的 Web Service。

WebSphere® Application Server 实现了 JAX-RPC 1.1 标准。

JAX-RPC 标准包括用于对 Java 语言的 Web Service 使用 Web Service 描述语言 (WSDL) 的编程模型和绑定。JAX-RPC 通过屏蔽 SOAP 通信的底层复杂性来简化 Web Service 的开发。

表面上，JAX-RPC 看上去像是另一个实例化的远程方法调用 (RMI)。本质上，JAX-RPC 允许客户机访问 Web Service，就像 Web Service 是映射至客户机地址空间的本地对象，即使 Web Service 提供程序位于世界的另一角落。JAX-RPC 是使用基于 XML 的协议 SOAP（它通常驻留在 HTTP 顶部）执行的。

JAX-RPC 定义 WSDL 端口类型与 Java 接口之间以及 Java 语言与可扩展标记语言 (XML) 模式类型之间的映射。

可以从 JavaBeans 或者企业 Bean 实现来创建 JAX-RPC Web Service。可以通过在 Java 接口中定义远程方法来指定远程过程。只需要编码实现方法的一个或多个类。剩余的类和其他工件由 Web Service 供应商的工具生成。以下是 Web Service 接口的示例：

package com.ibm.mybank.ejb;

import java.rmi.RemoteException;

import com.ibm.mybank.exception.InsufficientFundsException;

/\*\*

\* Remote interface for Enterprise Bean: Transfer

\*/

public interface Transfer\_SEI extends java.rmi.Remote {

public void transferFunds(int fromAcctId, int toAcctId, float amount)

throws java.rmi.RemoteException;

}

JAX-RPC 中的接口定义必须符合特定规则：

* 接口必须扩展 java.rmi.Remote，就像 RMI。
* 方法必须创建 java.rmi.RemoteException。
* 方法参数不能为远程引用。
* 方法参数必须是 JAX-RPC 规范支持的某个参数。下表是受支持的方法参数的示例。有关方法参数的完整列表，请参阅 JAX-RPC 规范。
  + 基本类型：boolean、byte、double、float、short、int 和 long
  + 基本类型的对象包装器：java.lang.Boolean、java.lang.Byte、java.lang.Double、java.lang.Float、java.lang.Integer、java.lang.Long、java.lang.Short
  + java.lang.String
  + java.lang.BigDecimal
  + java.lang.BigInteger
  + java.lang.Calendar
  + java.lang.Date
* 除了聚集值对象外，方法还可以采用组成先前列出类型的组合的值对象。

客户机创建存根并调用它上面的方法。存根象 Web Service 的代理一样运作。从客户机代码的角度来看，它看上去象本地方法调用。但是，每种方法调用编组到远程服务器。编组包括按 SOAP 协议规定编码 XML 中的方法调用。

以下是编写 Web Service 和 Web Service 客户机所需的键类和接口。

* 服务接口：存根或动态调用的工厂以及用来调用方法的代理对象
* ServiceFactory 类：用于服务的工厂。
* loadService

WebSphere Application Server V6.0 中提供了 loadService 方法以生成 JAX-RPC 实现所需的服务定位器。如果重新调用，那么先前版本中没有特定方式以获取生成的服务定位器。对于受管客户机，您使用 JNDI 方法获取服务定位器，而对于非受管客户机，需要您实例化 IBM 的特定服务定位器 ServiceLocator service=new ServiceLocator(...);，该定位器不提供可移植性。loadService 参数包括：

* + wsdlDocumentLocation：表示服务的 WSDL 文档位置的 URL，或者为 null。
  + serviceName：服务的限定名
  + 属性：一组特定于实现的属性，用于帮助查找生成的服务实现类。
* isUserInRole

isUserInRole 方法返回一个布尔值以表明端点实例上当前方法调用的已认证用户是否包含在特定的逻辑角色中。

* + 角色：角色参数是指定角色名的字符串。
* Service
* 调用接口：用于动态调用
* 存根接口：存根的基本接口

如果您正在使用存根来访问 Web Service 提供程序，那么将对您隐藏大多数 JAX-RPC API 详细信息。客户机创建 ServiceFactory (java.xml.rpc.ServiceFactory)。客户机实例化 ServiceFactory 的服务 (java.xml.rpc.Service)。该服务是创建端口的工厂对象。该端口是 Web Service 的远程服务端点接口。在 DII 的情况下，服务对象用于创建调用对象，您可配置该调用对象以调用 Web Service 端口上的方法。

**使用 JAX-RPC 的 RMI-IIOP**

可以将基于因特网 ORB 间协议的远程方法调用 (RMI-IIOP) 与 JAX-RPC 配合使用以支持非 SOAP 绑定。

基于 XML 的远程过程调用的 Java API (JAX-RPC) 是 Java 标准 API，它通过远程过程调用来调用 Web Service。编程语言使用传输来在因特网上进行通信。可以将协议与传输配合使用，例如，SOAP 和远程方法调用 (RMI)。您可以使用具有 JAX-RPC 远程方法调用/因特网 ORB 间协议 (RMI-IIOP) 来支持非 SOAP 绑定。

使用带有 JAX-RPC 的 RMI-IIOP，使 WebSphere® Java 客户机可以使用 WSDL 文件和 JAX-RPC 编程模型（而不是使用针对 Java Platform，Enterprise Edition (Java EE) 的标准 Web Service）来调用企业 Bean。使用 Enterprise JavaBeans 实现来调用 Web Service 时，多协议 JAX-RPC 允许针对 WebSphere Java 客户机优化 Web Service 调用路径。要了解有关此优化的更多信息，请阅读“使用企业 Bean 绑定从 Web Service 客户机调用 EJB”。

使用 RMI/IIOP 协议代替基于 SOAP 的协议的好处如下：

* 将不需要使用 XML 处理来发送和接收消息，转而使用 Java 序列化。
* 客户机 JAX-RPC 调用可以在不使用 SOAP 时参与用户事务。