**Apache CXF: An Open-Source Services Framework**

**Overview** Apache CXF is an open source services framework. CXF helps you build and develop services using frontend programming APIs, like JAX-WS and JAX-RS. These services can speak a variety of protocols such as SOAP, XML/HTTP, RESTful HTTP, or CORBA and work over a variety of transports such as HTTP, JMS or JBI.

Features : It has a broad feature set, but it is primarily focused on the following areas:

* **Web Services Standards Support:** CXF supports a variety of web service standards including SOAP, the WS-I Basic Profile, WSDL, WS-Addressing, WS-Policy, WS-ReliableMessaging, WS-Security, WS-SecurityPolicy, WS-SecureConverstation, and WS-Trust (partial).
* **Frontends:** CXF supports a variety of "frontend" programming models.

CXF implements the JAX-WS APIs. CXF JAX-WS support includes some extensions to the standard that make it significantly easier to use, compared to the reference implementation: It will automatically generate code for request and response bean classes, and does not require a WSDL for simple cases.

It also includes a "simple frontend" which allows creation of clients and endpoints without annotations. CXF supports both contract first development with WSDL and code first development starting from Java.

For REST, CXF also supports a JAX-RS frontend.

* **Ease of use:** CXF is designed to be intuitive and easy to use. There are simple APIs to quickly build code-first services, Maven plug-ins to make tooling integration easy, JAX-WS API support, Spring 2.x XML support to make configuration a snap, and much more.
* **Binary and Legacy Protocol Support:** CXF has been designed to provide a pluggable architecture that supports not only XML but also non-XML type bindings, such as JSON and CORBA, in combination with any type of transport.

To get started using CXF, check out the [downloads](https://cxf.apache.org/download.html), the [user's guide](http://cxf.apache.org/docs/index.html), or the [mailing lists](https://cxf.apache.org/mailing-lists.html) to get more information!

**Goals**

**General**

* High Performance
* Extensible
* Intuitive & Easy to Use

**Support for Standards**

**Flexible Deployment**

* Lightweight containers: deploy services in Jetty, Tomcat or Spring-based containers
* JBI integration: deploy as a service engine in a JBI container such as ServiceMix, OpenESB or Petals
* Java EE integration: deploy services in Java EE application servers such as Apache Geronimo, JOnAS, Redhat JBoss, OC4J, Oracle WebLogic, and IBM WebSphere
* Standalone Java client/server