Web service enhancements in ColdFusion

https://helpx.adobe.com/coldfusion/developing-applications/using-web-elements-and-external-objects/using-coldfusion-web-services/web-service-enhancements-in-coldfusion.html#

ColdFusion has Axis 2 Web service framework integrated. This enables your web services to use WSDL 2 specifications, SOAP 1.2 protocol, and document literal wrapped style. Also the enhancements resolve many interoperability issues that you might encounter while working with Web services in ColdFusion 9. The following table shows how the integration helps:

Consumption features	Publishing features
Supports WSDL 1.1 and WSDL 2.0 specifications.	Supports WSDL 1.1 and WSDL 2.0 specifications. To access WSDL2, use the URL as follows: {{ **http://localhost:8500/** <path of cfc>?wsdl2.}} To access WSDL1, use the URL as follows: {{ **http://localhost:8500/** <path of cfc>?wsdl.}}
Axis 2 support in ColdFusion lets you consume web services that publish WSDL in the	In ColdFusion 9, only

following styles: RPC encoded Document Literal Document Literal Wrapped To use Axis 2 for consumption purpose, specify wsversion as 2 in cfinvoke tag. You can also specify the version at the application level. For details, see Application level changes. If you have specified wsversion as 1 in cfinvoke, then Axis 1 is used. Axis 1 services published from ColdFusion are consumed only by Axis 1. If no value is specified, then If you publish WSDL in RPC encoded style, ColdFusion uses Axis 1 to consume the web service whereas, if you publish in either Document Literal or Document Literal Wrapped, the Axis 2 is used for consumption.	the following WSDL styles were supported: RPC Encoded Document Literal The style attribute that you specified in the cfcomponent tag decided the style used for publishing the WSDL. From ColdFusion 10, integration with Axis 2 lets you publish the WSDL in the new style Document Literal Wrapped. For publishing WSDL in this style, specify the style attribute as wrapped in cfcomponent. Note that you can specify style attribute as wrapped only if the wsversion specified is 2.

Supports SOAP 1.1 and SOAP 1.2 protocols.
While accessing the web service using cfinvoke, specify the serverport as shown in the following code:

```
<cfset str = structNew()>
<cfset str.serviceport =
"cfsuite.webservices.axis2.wscf.basic.cfcHttpSoap12Endpoint
">
<cfscript>
//invoke method
ws = createObject("webservice",
"http://localhost:8500/cfsuite/webservices/axis2/wscf/basic
.cfc?wsdl",str);
</cfscript>
<cfinvoke webservice="#ws#" method="Echo"
returnvariable="foo">
<cfoutput>#foo#</cfoutput>
```

Here, you have specified SOAP 1.2 endpoint and therefore, SOAP 1.2 protocol is used to send SOAP messages.|Supports SOAP 1.1 and SOAP 1.2. ColdFusion 9 supports only SOAP 1.1 endpoint. Integration with Axis 2 from ColdFusion 10 provides support for SOAP 1.1 and SOAP 1.2 protocols.That is, if you publish a Web service with WSVersion set to 2, the endpoints W3C SOAP 1.2 and W3C SOAP 1.1 are supported whereas if you specify 1 as the value, only W3C SOAP 1.1 is supported. For details on specifying WSVersion, see the configuration section.For details on SOAP, see www.w3.org/TR/SOAP/.|

Specifying the Axis settings

You can specify the Axis version at server level, application level, or component level.

Server level changes

Modify the following section in the neo-xmlrpc.xml available in the directory CFusion\lib.

```
<struct type='coldfusion.server.ConfigMap'>
<var name='version'>
<string> 2</string> </var>
</struct>
```

You could also specify web service version in the application.cfc by modifying the following tag:

```
<cfset this.wssettings.version.publish="1|2">
```

Application level changes

You can specify the Axis version that you want to use at the application level as follows:

- **For publication:** <cfset this.wssettings.version.publish="2">.

Note: The version you specify overrides the version specified at server level.

- **For consumption:** <cfset this.wssettings.version.consume="2">

Setting the attribute style at application level

Set as follows: <cfset this.wssettings.style="rpc/document/wrapped">

Handing ColdFusion's complex datatypes in Web services

Set the includeCFTypesInWSDL attribute at application level as follows: <cfset this.wssettings.includeCFTypesInWSDL="true">
The default value is true. If set to true, schemas are generated for complex datatypes and included in the WSDL. At the client-side, stubs are generated for complex datatypes. User can pass it as the argument.The following scenarios explain the need of setting the attribute includeCFTypesInWSDL:

```
<cffunction name="echoObject" access="remote" returnType="any">
<cfargument name="argObj" type="any">
<cfdump var="#argObj#" output="console">
<cfreturn argObj>
</cffunction>
```

Axis 2 generates WSDL by introspecting CFC's method signatures. In this function, both argument type and return type are any. Therefore, the schema generated with WSDL will not contain the schemas for ColdFusion's complex datatypes such as struct, query, or xml. If these schemas are not present in the WSDL, then at the client side, there will not be stub classes for the complex types, making the function useless.Similarly, in the following function, both argument type and return type are struct. In this case also, the WSDL will contain schema only for the struct. If you want to pass Document inside struct or Query inside struct, they are not available as schemas in the WSDL.

```
<cffunction name="echoComplexStruct" access="remote" returnType="struct">
<cfargument name="argStr" type="struct">
<cfdump var="#argStr#" output="console">
<cfreturn argStr>
</cffunction>
```

Component level changes

- New attribute wsversion has been added to cfcomponent.}}If you specify {{2, CFC is deployed using Axis 2 engine. The value you specify overrides the value you specify at application or server level.
- A new value wrapped can be specified for the attribute style. If you are setting wsversion as 2, the default value is wrapped and if it is 1, then the default value is rpc.

Note: If you have set style at the application level, in both the cases, instead of the default values, the values at the application level are used.

The syntax is as follows:

```
<cfcomponent
wsversion = 1|2>
style = "rpc|document|wrapped"
....
<cffunction ...>
...
</cffunction>

<cffunction ...>
...
</cffunction>
</cfcomponent>
```

Modifications to createObject and cfobject

- New property wsVersion has been added to createObject to specify the Axis version.
- New attribute wsVersion has been added to the tag cfobject to specify the Axis version.

Variations from ColdFusion 9

If the same display name is used in multiple CFCs, in ColdFusion 9, both the CFCs are published. But from ColdFusion 10, if wsVersion is set to 2, in the case of multiple CFCs with same display name, only the first CFC is published. Attempt to access the second CFC results in an error.

Web services are not automatically registered when you access the service using cfinvoke, cfobject, or createObject. You have to register the Web service in the ColdFusion Administrator (Data & Services > Web Services).

Limitations

Note: The following limitations will be addressed in the future releases of ColdFusion.

- Unlike in Axis 1, Axis 2 cannot support two CFCs having the same display name.
- The following attributes in the tag cfinvoke are not supported in Axis 2: Serviceaddress, portTypeName.
- If you are publishing JWS files, only Axis 1 is supported.