Accessing non Progress® Application via Web Service – An overview

Progress User Group India Let's do PROGRESS together!	SKIL evolving thoughts... changing mindsets
www.pugindia.org	www.skilglobal.com
pugadmin@pugindia.org	info@skilglobal.com
Author: Raghuraman Kadambi	

Contents

Introduction to Web Services and OpenEdge® Support for Web Service

A Web service is a method of communication between two electronic devices over a network (Machine-to-Machine).

The W3C defines a Web service generally as:
"a software system designed to support interoperable machine-to-machine interaction over a network". (Source: https://en.wikipedia.org/wiki/Web_service)

History

Web services evolved from previous technologies that served the similar purpose such as Remote Procedure Call (RPC), DCOM, CORBA and JAVA Remote Method Invocation (RMI).

Web Services were intended to solve three main problems:

1. Interoperability
2. Firewall traversal
3. Complexity

Interoperability

Earlier distributed systems suffered from interoperability issues because each software vendor implemented its own format for distributed object messaging.

For example, development of DCOM apps strictly bound to Windows Operating system. And even, development of RMI bound to Java programming language.

Firewall

Collaboration across companies was an issue because distributed systems such as CORBA and DCOM used non-standard ports.

However, Web Services use HTTP as a transport protocol and most of the firewalls allow access though port 80 (HTTP), leading to easier and dynamic collaboration.

Complexity

Web Services is a developer-friendly service system.

Most of the above-mentioned technologies such as RMI, COM, and CORBA involve a whole learning curve. New technologies and languages have to be learnt to implement these services.

Web services in OpenEdge®

In OpenEdge, a Web service is usually an AppServer® application that is accessible to a client application through a Web server.

In OpenEdge, you could:

- Create new Web services that you build as ABL (Advanced Business Language) applications and deploy on an AppServer®.
- Expose existing AppServer® applications as Web services.
- Create an interface to your Web services and deploy it on a Web server.
- Create the client-side applications that interact with your Web services.

OpenEdge includes support for Web services that are based either on SOAP (Simple Object Access Protocol), or on REST (Representation State Transfer).

Both SOAP and REST are industry standards, but there are many variations on how they are implemented.

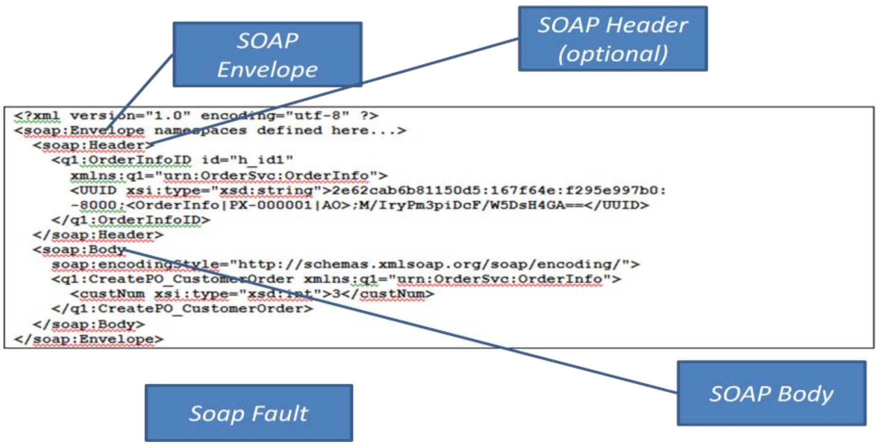
Loads of information available on web for these topics!

SOAP (Simple Object Access Protocol)

The Simple Object Access Protocol (SOAP) is the basis for enabling applications to communicate over the Internet, independent of how they are programmed or what platform they are deployed on.

The W3C standard for SOAP defines the format that a message must have when it is sent over the Internet for Web service requests and responses.

OpenEdge support for Web services includes the functionality of a SOAP processor, which is used to manage SOAP messages on behalf of your application.



```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope namespaces defined here...>
  <soap:Header>
    <q1:OrderInfoID id="h_id1"
      xmlns:q1="urn:OrderSvc:OrderInfo">
      <UUID xsi:type="xsd:string">2e62cab6b81150d5:167f64e:f295e997b0:
      -8000;<OrderInfo|PX-000001|AO>;M/IryPm3piDcF/W5DsH4GA==</UUID>
    </q1:OrderInfoID>
  </soap:Header>
  <soap:Body
    soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <q1:CreatePO_CustomerOrder xmlns:q1="urn:OrderSvc:OrderInfo">
      <custNum xsi:type="xsd:int">3</custNum>
    </q1:CreatePO_CustomerOrder>
  </soap:Body>
</soap:Envelope>
```

© Progress Software Corporation, USA

WSDL (Web Services Definition Language)

WSDL, the Web Services Definition Language, is another W3C standard.

A WSDL document is a body of metadata in the form of an XML-based description of how to communicate with a Web service. It describes the SOAP messages that a Web service accepts and generates.

The WSDL document contains all the information a Web services client needs to make a request of, or consume, a Web Service. A WSDL document also contains the information used to locate the Web service on the Internet.

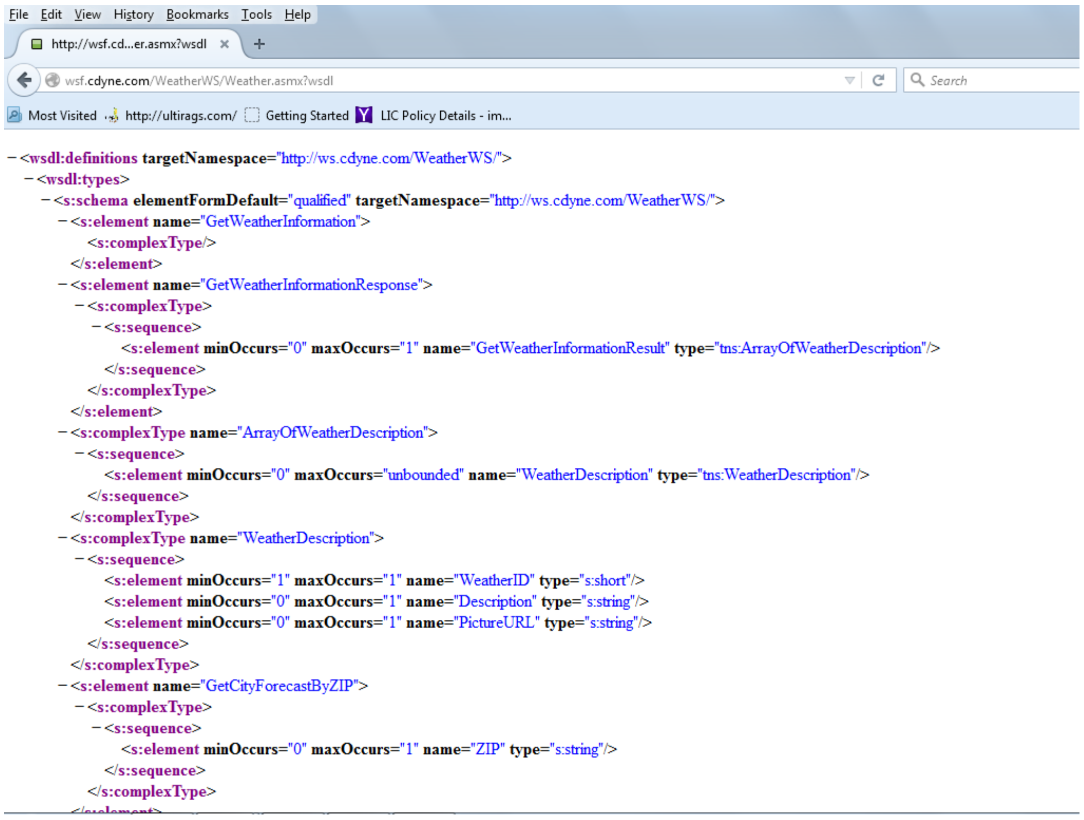
If an organization wishes to allow access to its Web services, it must create and host WSDL documents that describe them.

To exposing ABL procedures as Web services, you have to use the OpenEdge ProxyGen tool. (Not in the scope of this document)

But if you are consuming someone else's Web service, you need only know the location of the WSDL that describes it. (In the scope of this document)

For the learning purpose, CDYNE Weather Web Service is used in this document. This is a free SOAP Web Service provides you with up to date weather information in the United States.

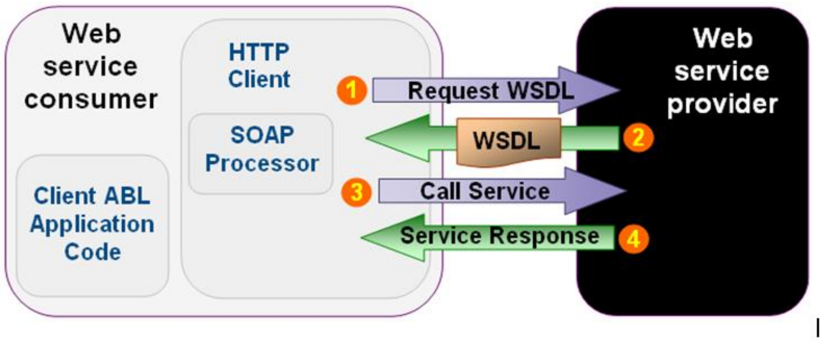
(Source: http://wiki.cdyne.com/?title=CDYNE_Weather)


File Edit View History Bookmarks Tools Help
http://wsf.cd...er.asmx?wsdl
wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Search
Most Visited
http://ultirags.com/
Getting Started
LIC Policy Details - im...

```
- <wsdl:definitions targetNamespace="http://ws.cdyne.com/WeatherWS/">
  - <wsdl:types>
    - <s:schema elementFormDefault="qualified" targetNamespace="http://ws.cdyne.com/WeatherWS/">
      - <s:element name="GetWeatherInformation">
          <s:complexType/>
        </s:element>
      - <s:element name="GetWeatherInformationResponse">
        - <s:complexType>
          - <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="GetWeatherInformationResult" type="tns:ArrayOfWeatherDescription"/>
            </s:sequence>
          </s:complexType>
        </s:element>
      - <s:complexType name="ArrayOfWeatherDescription">
        - <s:sequence>
            <s:element minOccurs="0" maxOccurs="unbounded" name="WeatherDescription" type="tns:WeatherDescription"/>
          </s:sequence>
        </s:complexType>
      - <s:complexType name="WeatherDescription">
        - <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="WeatherID" type="s:short"/>
            <s:element minOccurs="0" maxOccurs="1" name="Description" type="s:string"/>
            <s:element minOccurs="0" maxOccurs="1" name="PictureURL" type="s:string"/>
          </s:sequence>
        </s:complexType>
      - <s:element name="GetCityForecastByZIP">
        - <s:complexType>
          - <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="ZIP" type="s:string"/>
            </s:sequence>
          </s:complexType>
```

(Source: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl)

Web Service request and service response cycle in OpenEdge®

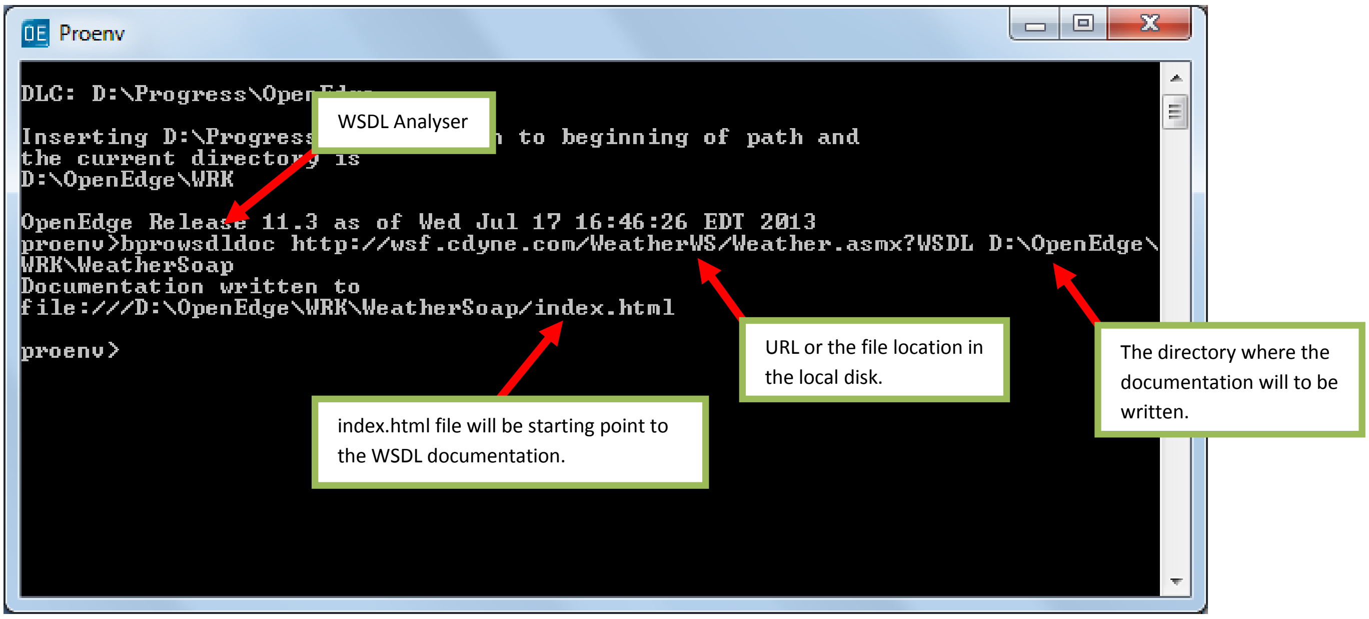


© Progress Software Corporation, USA

Generate the ABL WSDL documentation and use in developing your ABL code

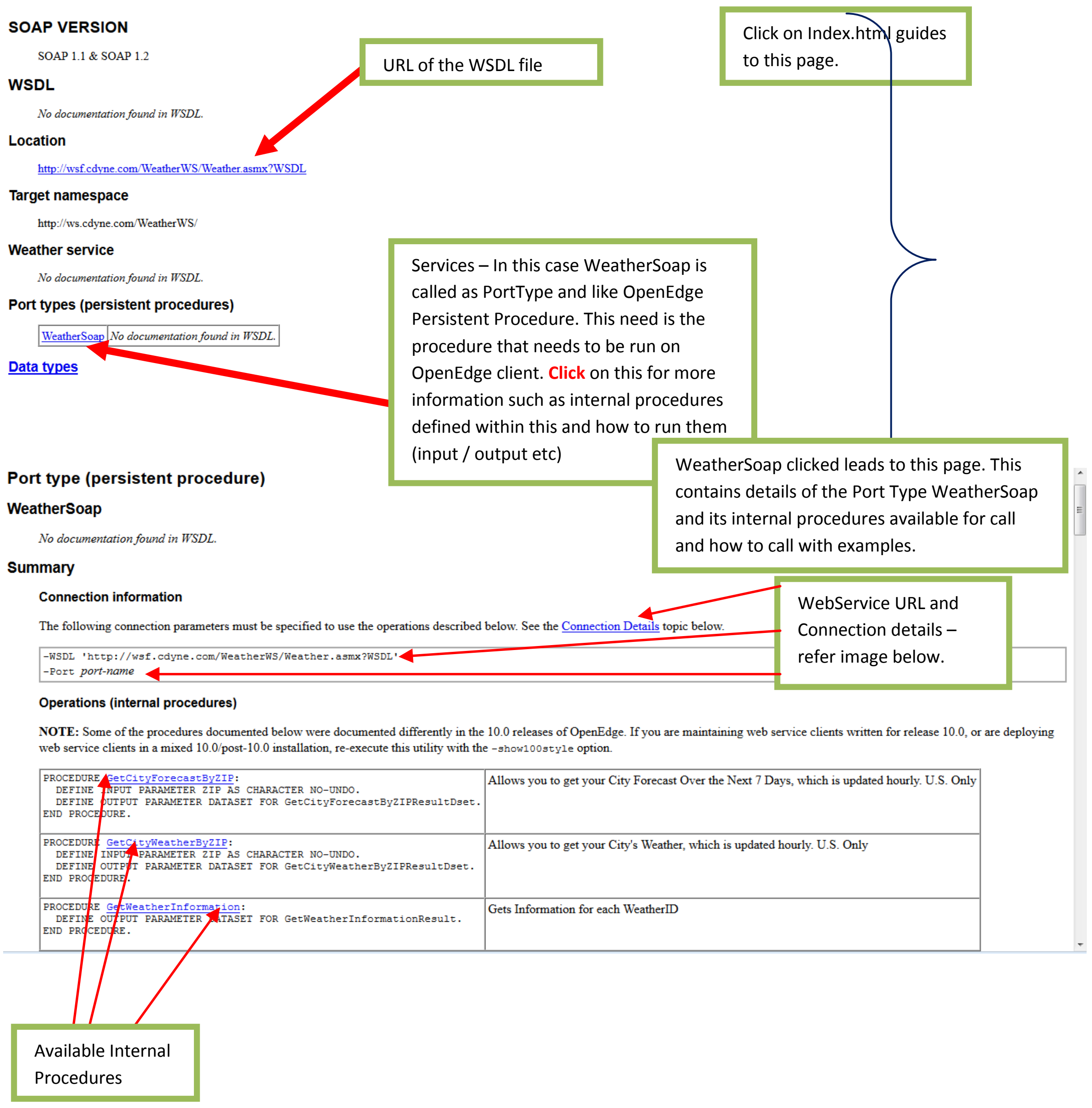
Step 1 - Generate documentation of the .WSDL from the URL

Generate documentation from the URL http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL using OpenEdge WSDL Analyser. The OpenEdge tool **bprowsdldoc** is WSDL Analyser and resides in the *ProgressInstallDirectory/bin* directory. This tool generates a set of HTML files that describe the **service** (typically like a persistent procedure – Port Type) and all **its operations** (typically like an internal procedure within the persistent procedure). Refer to the screen below:


Proenv
WSDL Analyser
URL or the file location in the local disk.
The directory where the documentation will to be written.
index.html file will be starting point to the WSDL documentation.

```
DLC: D:\Progress\OpenEdge

Inserting D:\Progress...n to beginning of path and
the current directory is
D:\OpenEdge\WRK

OpenEdge Release 11.3 as of Wed Jul 17 16:46:26 EDT 2013
proenv>bprowsdldoc http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL D:\OpenEdge\
WRK\WeatherSoap
Documentation written to
file:///D:\OpenEdge\WRK\WeatherSoap/index.html

proenv>
```

Step 2 – Review the documentation for Service, Operation & Client Connection Details



SOAP VERSION

SOAP 1.1 & SOAP 1.2

WSDL

No documentation found in WSDL.

Location

http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL

Target namespace

http://ws.cdyne.com/WeatherWS/

Weather service

No documentation found in WSDL.

Port types (persistent procedures)

WeatherSoap	*No documentation found in WSDL.*

Data types

URL of the WSDL file

Click on Index.html guides to this page.

Services – In this case WeatherSoap is called as PortType and like OpenEdge Persistent Procedure. This need is the procedure that needs to be run on OpenEdge client. Click on this for more information such as internal procedures defined within this and how to run them (input / output etc)

Port type (persistent procedure)

WeatherSoap

No documentation found in WSDL.

WeatherSoap clicked leads to this page. This contains details of the Port Type WeatherSoap and its internal procedures available for call and how to call with examples.

Summary

Connection information

The following connection parameters must be specified to use the operations described below. See the Connection Details topic below.

```
-WSDL 'http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL'
-Port port-name
```

WebService URL and Connection details – refer image below.

Operations (internal procedures)

NOTE: Some of the procedures documented below were documented differently in the 10.0 releases of OpenEdge. If you are maintaining web service clients written for release 10.0, or are deploying web service clients in a mixed 10.0/post-10.0 installation, re-execute this utility with the `-show100style` option.

`PROCEDURE GetCityForecastByZIP:`
`  DEFINE INPUT PARAMETER ZIP AS CHARACTER NO-UNDO.`
`  DEFINE OUTPUT PARAMETER DATASET FOR GetCityForecastByZIPResultDset.`
`END PROCEDURE.`
	Allows you to get your City Forecast Over the Next 7 Days, which is updated hourly. U.S. Only

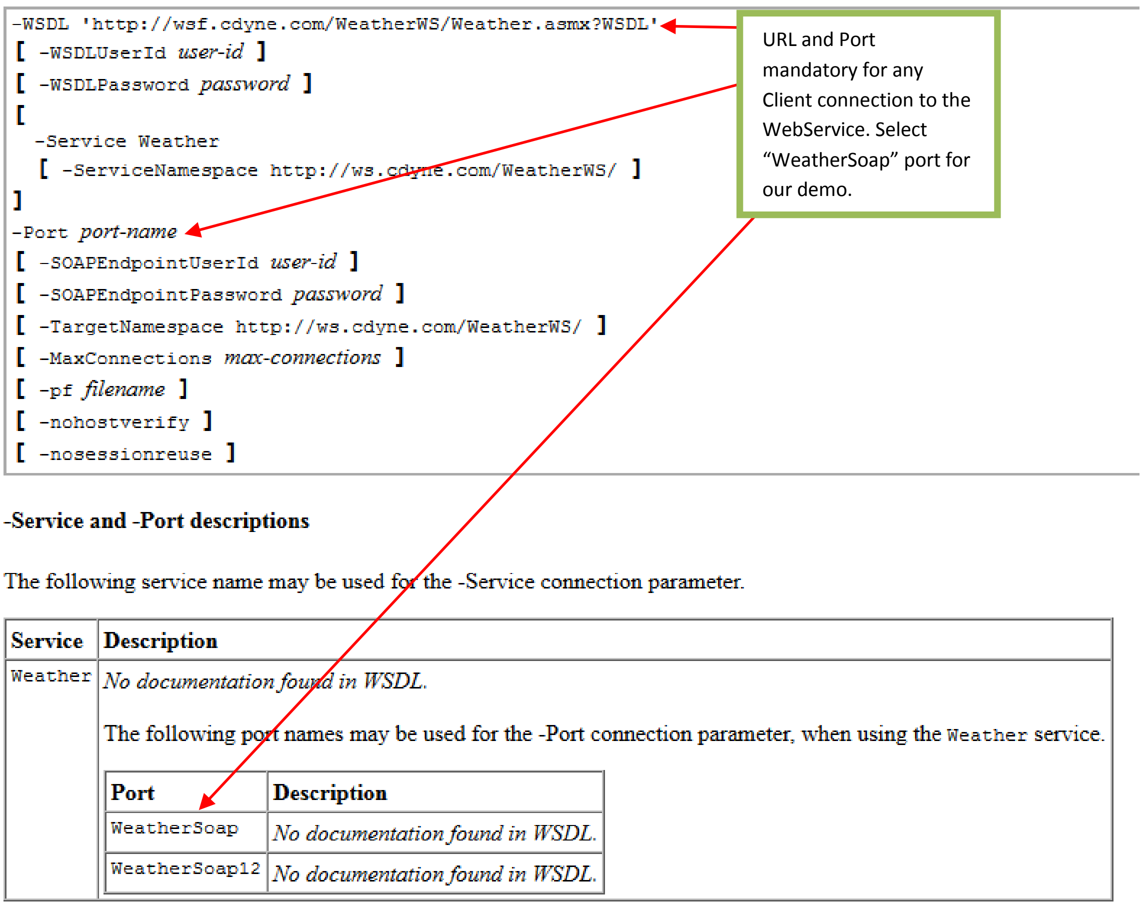
`PROCEDURE GetCityWeatherByZIP:`
`  DEFINE INPUT PARAMETER ZIP AS CHARACTER NO-UNDO.`
`  DEFINE OUTPUT PARAMETER DATASET FOR GetCityWeatherByZIPResultDset.`
`END PROCEDURE.`
	Allows you to get your City's Weather, which is updated hourly. U.S. Only

`PROCEDURE GetWeatherInformation:`
`  DEFINE OUTPUT PARAMETER DATASET FOR GetWeatherInformationResult.`
`END PROCEDURE.`
	Gets Information for each WeatherID

Available Internal Procedures

Connection details

Connection parameters



```
-WSDL 'http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL'
[ -WSDLUserId user-id ]
[ -WSDLPassword password ]
[
  -Service Weather
  [ -ServiceNamespace http://ws.cdyne.com/WeatherWS/ ]
]
-Port port-name
[ -SOAPEndpointUserId user-id ]
[ -SOAPEndpointPassword password ]
[ -TargetNamespace http://ws.cdyne.com/WeatherWS/ ]
[ -MaxConnections max-connections ]
[ -pf filename ]
[ -nohostverify ]
[ -nosessionreuse ]
```

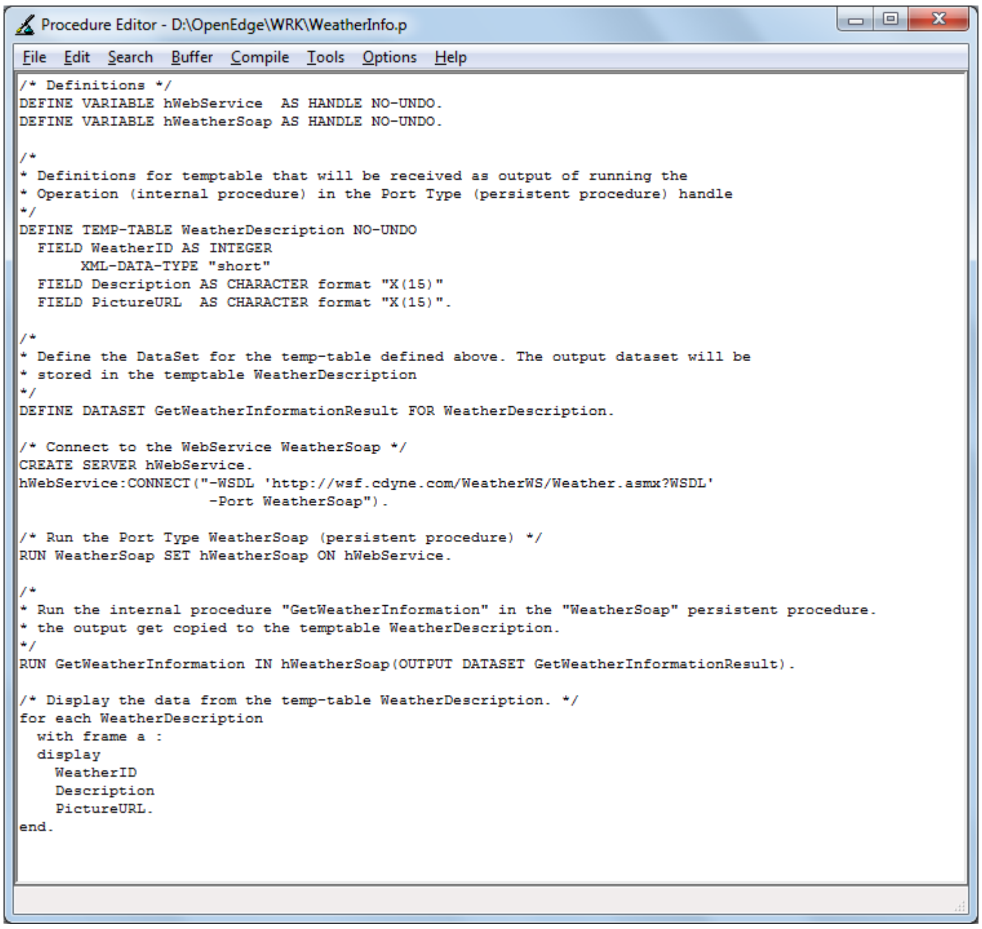
URL and Port mandatory for any Client connection to the WebService. Select "WeatherSoap" port for our demo.

-Service and -Port descriptions

The following service name may be used for the -Service connection parameter.

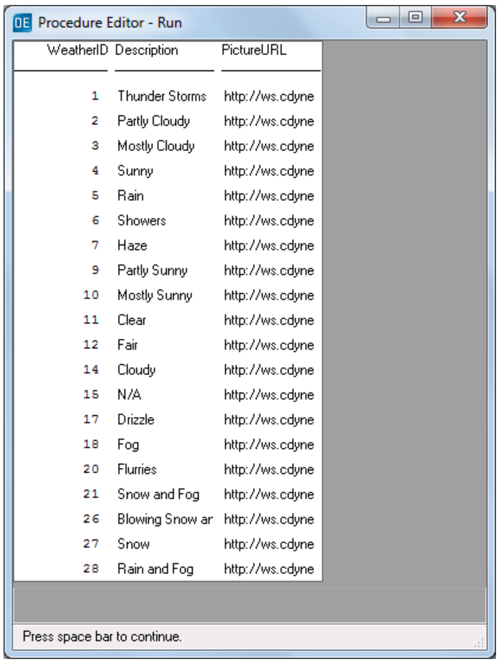
Service	Description
`Weather`	*No documentation found in WSDL.* The following port names may be used for the -Port connection parameter, when using the `Weather` service. **Port** | **Description** `WeatherSoap` | *No documentation found in WSDL.* `WeatherSoap12` | *No documentation found in WSDL.*

Step 3 –OpenEdge ABL Code to Connect to the WebService – WeatherInfo.p


Procedure Editor - D:\OpenEdge\WRK\WeatherInfo.p
File Edit Search Buffer Compile Tools Options Help

```
/* Definitions */
DEFINE VARIABLE hWebService  AS HANDLE NO-UNDO.
DEFINE VARIABLE hWeatherSoap AS HANDLE NO-UNDO.

/*
* Definitions for temptable that will be received as output of running the
* Operation (internal procedure) in the Port Type (persistent procedure) handle
*/
DEFINE TEMP-TABLE WeatherDescription NO-UNDO
  FIELD WeatherID AS INTEGER
       XML-DATA-TYPE "short"
  FIELD Description AS CHARACTER format "X(15)"
  FIELD PictureURL  AS CHARACTER format "X(15)".

/*
* Define the DataSet for the temp-table defined above. The output dataset will be
* stored in the temptable WeatherDescription
*/
DEFINE DATASET GetWeatherInformationResult FOR WeatherDescription.

/* Connect to the WebService WeatherSoap */
CREATE SERVER hWebService.
hWebService:CONNECT("-WSDL 'http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL'
                     -Port WeatherSoap").

/* Run the Port Type WeatherSoap (persistent procedure) */
RUN WeatherSoap SET hWeatherSoap ON hWebService.

/*
* Run the internal procedure "GetWeatherInformation" in the "WeatherSoap" persistent procedure.
* the output get copied to the temptable WeatherDescription.
*/
RUN GetWeatherInformation IN hWeatherSoap(OUTPUT DATASET GetWeatherInformationResult).

/* Display the data from the temp-table WeatherDescription. */
for each WeatherDescription
  with frame a :
  display
    WeatherID
    Description
    PictureURL.
end.
```

Step 4 – Output of above code



```
WeatherID Description     PictureURL
--------- --------------- ---------------
        1 Thunder Storms  http://ws.cdyne
        2 Partly Cloudy   http://ws.cdyne
        3 Mostly Cloudy   http://ws.cdyne
        4 Sunny           http://ws.cdyne
        5 Rain            http://ws.cdyne
        6 Showers         http://ws.cdyne
        7 Haze            http://ws.cdyne
        9 Partly Sunny    http://ws.cdyne
       10 Mostly Sunny    http://ws.cdyne
       11 Clear           http://ws.cdyne
       12 Fair            http://ws.cdyne
       14 Cloudy          http://ws.cdyne
       15 N/A             http://ws.cdyne
       17 Drizzle         http://ws.cdyne
       18 Fog             http://ws.cdyne
       20 Flurries        http://ws.cdyne
       21 Snow and Fog    http://ws.cdyne
       26 Blowing Snow ar http://ws.cdyne
       27 Snow            http://ws.cdyne
       28 Rain and Fog    http://ws.cdyne

Press space bar to continue.
```

References, Credits, Trademarks and Copyrights

1. Accessing Web Services From OpenEdge, by John Sadd (2007)
2. Progress Software Corporation, USA – OpenEdge Web Services documentation.
3. CDYNE Weather SOAP Web Services: http://wiki.cdyne.com/?title=CDYNE_Weather
4. Definition of Web Service: https://en.wikipedia.org/wiki/Web_service