Lotus Notes/Domino 7 Web Services

Robert Perron

September 27, 2004

Web Services is new in Lotus Notes/Domino 7. This article introduces you to the new Web Services design element, showing how to create one in Domino Designer, and describes how to implement Web Services with both LotusScript and Java code examples.

[Editor's note: This article describes the Lotus Notes/Domino 7 Beta 2 implementation of Web services. It may not accurately reflect the features or functionality of the Gold version of Lotus Notes/Domino 7.]

A Web service is an archive of remote operations that can be called by sending messages over the Internet. A Web service provider publishes a Web service for query and use, and a Web service consumer calls operations from the service. A Web service provider makes available a WSDL (Web Services Description Language) document that defines the service interface. The WSDL document is in XML format. What happens behind the interface is up to the provider, but most providers map the interface to procedure calls in a supported programming language. Incoming requests from a consumer are passed through to the underlying code, and results are passed back to the consumer.

Lotus Domino maps the WSDL interface to an agent-like Web service design element that can be coded in LotusScript or Java. To be used, the Web service must be on a Domino server with HTTP enabled. (We can test the Web service through an HTTP session in the Notes client preview.) Access is through one of the following Domino URL commands:

- ?OpenWebService invokes the Web service in response to a SOAP-encoded message sent through an HTTP POST. An HTTP GET (for example, a browser query) returns the name of the service and its operations.
- ?WSDL returns the WSDL document in response to an HTTP GET.

This article describes the Web services design element in Lotus Notes/Domino 7 and provides LotusScript and Java examples of the design element. This article assumes that you are an experienced Notes application developer with knowledge of LotusScript or Java.

An example

Let's take a simple example. Given a database name, a view name, and a document number, our operation returns the content of a Subject item. We'll call our operation getNthSubject.
To make the operation available to the outside world, we publish it in a Web service called GetSubject. GetSubject can contain any number of operations. For example, we might find getFirstSubject and getLastSubject useful. But for now let’s just deal with our example operation, getNthSubject. Here’s an excerpt from a WSDL document describing a Web service that contains such an operation. Look at it in conjunction with the annotations that follow.

```xml
<wSDL:message name="getNthSubjectRequest"> (4)
  <wsdl:part name="dbname" type="xsd:string"/> (5)
  <wsdl:part name="viewname" type="xsd:string"/> (5)
  <wsdl:part name="n" type="xsd:int"/> (5)
</wsdl:message>

<wSDL:message name="getNthSubjectResponse"> (4)
  <wsdl:part name="getNthSubjectReturn" type="xsd:string"/> (6)
</wsdl:message>

<wSDL:portType name="GetSubjectPortType"> (1)
  <wsdl:operation name="getNthSubject" parameterOrder="dbname viewname n"> (2)
    <wsdl:input message="impl:getnthSubjectRequest" name="GetNthSubjectRequest"/> (3)
    <wsdl:output message="impl:getNthSubjectResponse" name="GetNthSubjectResponse"/> (3)
  </wsdl:operation>
</wsdl:portType>
```

Look first at the portType element (1), which defines a set of operations for the service. Our service has just one portType, which has just one operation, getNthSubject (2). The operation has two "messages" (3): one for input and one for output. The messages are defined in message elements (4). We see that the input message has three parts (5): two strings named dbname and viewname and an int named n. The output message has a single part (6) named getNthSubjectReturn which is a string.

So we have one operation with three input parts and one output part, which maps very neatly to a procedure with three read-only parameters and one return value. In LotusScript, such a procedure would be defined by the following function:

```lss
Public Function getNthSubject(dbname As String, viewname As String, n As Long) As String
```

And in Java by the following method:

---

**Figure 1. getNthSubject diagram**

![getNthSubject diagram](image-url)
public String getNthSubject(String dbname, String viewname, int n)

**Web service design element**

Several approaches are possible for creating a Web service design element in Domino Designer. We can code it entirely in LotusScript or Java. In these cases, saving the design element generates a WSDL document that reflects the LotusScript or Java code. Or we can import an existing WSDL document. In this case, LotusScript or Java is generated that reflects the operations in the imported WSDL. The Web service design element saves the WSDL document as well as the code. If the public interface has not changed, the WSDL document stays as is. If, in our coding, we change anything that affects the public interface, a new WSDL is generated.

In Domino Designer, the Web service design element resides below Agents under Shared code. The Web service design window looks a lot like the agent design window. Click the New Web Service button to create a new Web service. Double-click an existing Web service to edit it.

**Figure 2. New Web Service**

![New Web Service](image)

The Web Services Property box has three tabs just like agents. Here's the Basics tab:

**Figure 3. Web Services Property box**

![Web Services Property box](image)

A name is required. An alias and comment can be supplied or not. You can elect to be warned if a coding change causes generation of a new WSDL.

The PortType class is the name of the class that defines the procedures that map to the WSDL operations. These procedures must be public functions or subs in LotusScript and public methods.
in Java. Private functions, subs, and methods are not exposed through the Web services interface. We cannot enter the PortType class in the properties box until we have created the class through coding or importing a WSDL. We will look more closely at the code shortly.

The Security tab is almost exactly the same as the agent Security tab. We will discuss Security in more detail later.

The Advanced tab has additional information for defining the Web service and generating the WSDL. We will discuss this in more detail later.

The editor pane is similar to an agent's. In the right drop-down box, we can select LotusScript or Java. Below a selection of Java is shown. On the left are Objects and Reference panes.

**Figure 4. Web service (Java)**

Use the Import WSDL button to create a new Web service based on an existing WSDL document. The Show WSDL button compiles any changes to the Web service and displays the WSDL document that defines its public interface. The Export WSDL button compiles any changes to the Web service and exports the WSDL document that defines its public interface. We can also compile by saving or closing the Web service. The WSDL is regenerated only if the public interface changes.

**Basic coding**

The code for a Web service has the following elements:

- A class definition for the implementation code. This class must become the PortType class named in the Basics tab of the properties box and must be public.
- Within the class, a procedure (function, sub, or method) definition for each operation in the Web service. These procedures must be public. Supporting procedures that we don't want in the interface must be private.
- Inclusion of lxsdl.lss for LotusScript and import of lotus.domino.types.* for Java.
- Initialization of a NotesSession (LotusScript) or Session (Java) object if Domino Objects are accessed. This is best done in a new block for LotusScript or a no-parameter constructor for Java. For Java, we used WebServiceBase.getCurrentSession() to get a Session.
object. We may also want to get an AgentContext object with Session.getAgentContext().
WebServiceBase is the equivalent of JavaAgent, but the Web service does not have access
to the object. The only useful method is the static getCurrentSession().

Here is a template for LotusScript code where the Web service contains one operation. The
operation is the example described above with three input parameters and one return value.

Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession
Class GetSubject
  Sub NEW
    Set s = New NotesSession
  End Sub
  Function getNthSubject(dbname As String, viewname As String, n As Long) As String
    ! Code for doing the operation goes here
  End Function
End Class

And here is a template for Java code where the Web service contains one operation. The
constructor must be the default constructor (have no parameters). Other constructors are ignored.

import lotus.domino.*;
import lotus.domino.types.*;
public class GetSubject {
  Session s;
  public GetSubject() {
    s = WebServiceBase.getCurrentSession();
  }
  public String getNthSubject(String dbname, String viewname, int n) {
    // Code for doing operation goes here
  }
}

Now we’ll expand the examples to include the working code. Here’s the LotusScript:

Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession
Class GetSubject
  Sub NEW
    Set s = New NotesSession
  End Sub
  Function getNthSubject(dbname As String, viewname As String, n As Long) As String
    Dim db As NotesDatabase
    Dim view As NotesView
    Dim doc As NotesDocument
    Set db = s.GetDatabase("", dbname)
    If Not(db.IsOpen) Then
      getNthSubject = "Cannot open database " & dbname
      Exit Function
    End If
    Set view = db.getView(viewname)
    If view Is Nothing Then
      getNthSubject = "Cannot open view " & viewname
      Exit Function
    End If
    ! Code for doing operation goes here
  End Function
End Class
Here's the Java:

```java
import lotus.domino.*;
import lotus.domino.types.*;

public class GetSubject {
    Session s;

    public GetSubject() {
        s = WebServiceBase.getCurrentSession();
    }

    public String getNthSubject(String dbname, String viewname, int n) {
        String subject = null;
        try {
            Database db = s.getDatabase(null, dbname);
            if (!db.isOpen()) subject = "Cannot open database " + dbname;
            else {
                View view = db.getView(viewname);
                if (view == null) subject = "Cannot open view " + viewname;
                else {
                    Document doc = view.getNthDocument(n);
                    if (doc == null) subject = "Cannot get document " + n;
                    else {
                        if (doc.hasItem("Subject"))
                            subject = doc.getItemValueString("Subject");
                        else subject = "Document does not have Subject";
                    }
                }
            }
            catch(Exception e) {
                subject = e.toString();
                e.printStackTrace();
            }
        }
        return subject;
    }
}
```

**Invoking and testing Web services**

Ultimately, the Web service design element must reside on a Domino 7 server with HTTP running. We can test a Web service design element residing on Domino Designer. We must first select Design - Preview in Web Browser on anything (a form, for example) which starts HTTP. Use 127.0.0.1 for the computer address if the consumer is on the same machine as the Notes client. To act as a consumer of a Web service, we have to send a SOAP message in an HTTP POST request to the URL for the Domino Web service. The URL looks something like this:

The SOAP message looks something like this:

```xml
<SOAP-ENV:Envelope
  ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ns0="urn:DefaultNamespace">
  <ns0:getNthSubject (1)
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:ns0="urn:DefaultNamespace">
    <dbname xsi:type="xsd:string">Webservices2</dbname> (2)
    <viewname xsi:type="xsd:string">Main View</viewname> (2)
    <n xsi:type="xsd:int">2</n> (2)
  </ns0:getNthSubject>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In this example, the SOAP message (1) identifies the operation and (2) provides values for the input parts. The specific elements appearing in the SOAP-ENV:body are determined by the WSDL binding characteristics, especially the SOAP message format (see "Advanced Properties" below for more detail).

The WebSphere SDK for Web Services provides a tool for invoking Web services and viewing results. This SDK runs under Eclipse. The following must be installed:

- **Eclipse 2.1**
- **IBM WebSphere SDK for Web Services Version 5.1**

The WebSpere SDK (as of this writing) does not work with Eclipse 3.0. You can also use [WebSphere Studio Application Developer](http://www.ibm.com/software/webservers/appdev/ws) to invoke Web services. The latest version is v5.1.2.

To use the WebSphere SDK tool, open Eclipse and choose Run - Launch the Web Services Explorer. After the Web Services Explorer loads:

1. Click the WSDL Page icon. This is the third icon after the right arrow on the right top. A WSDL Main link appears in the Navigator pane on the left.
2. Click the WSDL Main link. An Open WSDL box appears in the right pane.
3. Enter the URL of the Web service with the command ?WSDL, for example, `http://rperron300pl.notesdev.ibm.com/Webservices2.nsf/GetSubject?WSDL`, and click Go. We want ?WSDL (not ?OpenWebService) because Web Services Explorer reads the WSDL document at this point.
4. A WSDL Binding Details box appears in the right pane. It contains links to the operations defined by the Web service.
5. Click the name of the operation, for example, getNthSubject. An Invoke a WSDL Operation box appears.
6. Enter values for the input parts (parameters) and click Go.

The response comes back in the bottom (Status) pane. Here's what Web Services Explorer might look like after invoking the example Web service.
Figure 5. Web Services Explorer

The Actions box has a Source link in the upper right. Clicking Source shows the actual SOAP message. We can modify the SOAP message, and then transmit it by clicking Go. Click Form in the upper right to go back to the original display. The Status box also has a Source link which allows us to see the SOAP response. If the status says there is nothing to display (and a response is expected), the code probably failed.

After running a Web service, check the server console or log.nsf for error messages. We can log or debug by inserting MessageBox statements, which print to the server console or log.nsf. (Do not use Print statements in Beta 2. They go to the HTTP stream as for an agent and corrupt the SOAP response.)

Advanced properties

The Advanced tab of the properties box affects the Web service definition as reflected in the WSDL document.

Figure 6. Advanced tab of the Web Services Property box

We can provide what names we want for port type, service element, and service port. For example, we could use GetSubject for all the names. For clarity, we use suffixes that reflect the element type. When we provide names in the properties box, these names are plugged into the generated WSDL document. If we import a WSDL document, the names in the WSDL document are automatically plugged into the properties box.
Below is the complete WSDL document for the GetSubject example. Annotated are the portions reflected in the Advanced tab of the Web Services Property box.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
  targetNamespace="urn:DefaultNamespace"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:apachesoap="http://xml.apache.org/xml-soap"
  xmlns:impl="urn:DefaultNamespace"
  xmlns:intf="urn:DefaultNamespace"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/
  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:message name="getNthSubjectResponse">
    <wsdl:part name="getNthSubjectReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="getNthSubjectRequest">
    <wsdl:part name="dbname" type="xsd:string"/>
    <wsdl:part name="viewname" type="xsd:string"/>
    <wsdl:part name="n" type="xsd:int"/>
  </wsdl:message>
  <wsdl:portType name="GetSubjectPortType">
    <wsdl:operation name="getNthSubject" parameterOrder="dbname viewname n">
      <wsdl:input message="impl:getNthSubjectRequest" name="getNthSubjectRequest"/>
      <wsdl:output message="impl:getNthSubjectResponse" name="getNthSubjectResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="GetSubjectPortSoapBinding" type="impl:GetSubjectPortType">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="getNthSubject">
      <wsdlsoap:operation soapAction="" name="getNthSubjectRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:DefaultNamespace" use="encoded"/>
      </wsdlsoap:operation>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="GetSubjectElement">
    <wsdl:port binding="impl:GetSubjectPortSoapBinding" name="GetSubjectPort">
      <wsdlsoap:address location="http://localhost"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```
(1) The port type defines a set of operations. The WSDL document contains a name attribute for \texttt{wsdl:portType} which corresponds to the Port type name advanced property. (2) The service identifies the supported ports. The WSDL document contains a name attribute for \texttt{wsdl:service} which corresponds to the Service element name advanced property. The location attribute of \texttt{wsdlsoap:address} is not correct if the WSDL was obtained through Export WSDL, Show WSDL, or preview in a browser in Domino Designer; it is correct if obtained from the server through the \texttt{?WSDL URL} command. (3) A port identifies a binding which in turn identifies a port type and provides additional information. The WSDL document contains a name attribute for \texttt{wsdl:port} under \texttt{wsdl:service}. Domino allows one service and one port per service. (4) Two programming models and four SOAP message formats are available. The RPC programming model allows four SOAP message formats: RPC/encoded, RPC/literal, Doc/literal, and Wrapped (the utility of Doc/encoded, the fifth possible format, is not well understood, so is not supported here). The Message programming model forces Doc/literal message format, but as a hint only; the actual SOAP message format that gets passed in a Message-based Web service is not published, but rather is by private contract between consumer and provider. The style attribute of \texttt{wsdlsoap:binding} is set as follows:

- \texttt{wsdlsoap:binding style="rpc" for RPC/encoded and RPC/literal}
- \texttt{wsdlsoap:binding style="document" for Doc/literal and Wrapped}

(5) In the input and output elements under \texttt{wsdl:binding}, the use attribute of \texttt{wsdlsoap:body} is set as follows:

- \texttt{wsdlsoap:body use="encoded" for RPC/encoded. In this case, an encodingStyle attribute is present.}
- \texttt{wsdlsoap:body use="literal" for RPC/literal, Doc/literal, and Wrapped. In these cases, there is no encodingStyle attribute.}

(6) For RPC/encoded and RPC/literal, each message part defines the data type by a direct reference to the XMLSchema namespace (for example, type="xsd:string" or type="xsd:int") or a complex type defined in the WSDL "types" section (not shown in this example).

For Doc/literal, each message part refers to a previously defined data element. Below is an excerpt from the sample WSDL with Doc/literal specified. Under \texttt{wsdl:types}, each input part is defined in an element named after the corresponding parameter in the procedure code, and the output part is defined in an element named after the procedure plus "Return."
For Wrapped, each message has one part which refers to a previously defined element of type complexType named for the operation using it and having no attributes. Below is the WSDL excerpt with Wrapped specified.

For an excellent discussion of the SOAP formats, see the developerWorks article, "Which style of WSDL should I use?" by Russell Butek.

(7) soapAction="" if "Include operation name in SOAP action" remains unchecked. If this option is checked, the soapAction specifies the name of the operation, for example:

<wsdl:soap:operation soapAction="getNthSubject"/>
Security

Web services security is similar to security for a server agent invoked from the Web. Below is an example of the Security tab in the Web Services Property box.

**Figure 7. Security tab in Web Services Property box**

![Security tab in Web Services Property box](image)

The first two lines determine who is running the Web service, that is, the effective user: If neither line is used, the effective user is the owner of the Web service (the last user who edited or signed the design element).

- If the "Run as Web user" option is selected, the effective user is the user who negotiates network access to the database containing the Web service: Anonymous if the database allows anonymous access or the name supplied to the authentication process.
- If the "Run on behalf of" field is filled in, the effective user is that user.

The consumer of the Web service must be able to negotiate access to the server. Access is automatic if anonymous access is allowed on the HTTP port. Otherwise, the consumer must authenticate with a valid name and Internet password. The database ACL must give the effective user at least Depositor access with Read public documents checked.

"Compile Java code with debugging information" allows connection to a running Web service from a Java debugger such as Eclipse that supports the JPDA (Java Platform Debugger Architecture). Java debugging is new in release 7 and works only on a Notes client. For debugging, then, the Web service must reside on a Notes client. Start an HTTP task on the client by choosing Design - Preview in Web Browser on anything. Invoke the Web service. The Web service should contain debug-only code to pause it for awhile. Then connect the debugger to the running Web service.

For LotusScript Web services, "Allow remote debugging" takes the place of the Java debugging line. Remote debugging for a Web service is the same as for an agent. In this case, the Web service must reside on a server.
"Profile this Web service" allows the collection of elapsed times taken by Domino Objects. To report the results on a selected Web service, choose Design - View Profile Results. Profiling is new in release 7 and works for agents coded in LotusScript and Java as well as Web services.

The "Set runtime security level" box allows three levels of security. The higher-numbered security levels allow potentially damaging operations such as writing to the file system, manipulating environment variables, and so on.

For "Default access to this Web service," we can allow all readers and above, or we can enumerate those who have access.

Web services use the same framework as agents. In the back end, most but not all of agent context applies to Web services. We have already seen how to obtain Session and AgentContext objects in Java and a NotesSession object in LotusScript. Here are the other main contextual elements associated with Web services.

### Web service context

<table>
<thead>
<tr>
<th>Contextual element</th>
<th>Java</th>
<th>LotusScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Web service</td>
<td>AgentContext.getCurrentAgent();</td>
<td>NotesSession.CurrentAgent</td>
</tr>
<tr>
<td>Current database</td>
<td>AgentContext.getCurrentDatabase();</td>
<td>NotesSession.CurrentDatabase</td>
</tr>
<tr>
<td>Print to server console and log.nsf</td>
<td>System.out</td>
<td>MessageBox</td>
</tr>
<tr>
<td>Web service name and alias</td>
<td>Agent.getName();</td>
<td>NotesAgent.Name</td>
</tr>
<tr>
<td>Web service owner (full name)</td>
<td>Agent.getOwner();</td>
<td>NotesAgent.Owner</td>
</tr>
<tr>
<td>Web service owner (common name)</td>
<td>Agent.getCommonOwner();</td>
<td>NotesAgent.CommonOwner</td>
</tr>
<tr>
<td>Web service run-time owner</td>
<td>Agent.getOnBehalfOf();</td>
<td>NotesAgent.OnBehalfOf</td>
</tr>
<tr>
<td>Web service comment</td>
<td>Agent.getComment();</td>
<td>NotesAgent.Comment</td>
</tr>
<tr>
<td>Web service HTTP URL</td>
<td>Agent.getHttpURL();</td>
<td>NotesAgent.HttpURL</td>
</tr>
<tr>
<td>Web service Notes URL</td>
<td>Agent.getNotesURL();</td>
<td>NotesAgent.NotesURL</td>
</tr>
<tr>
<td>Web service parent database</td>
<td>Agent.getParent();</td>
<td>NotesAgent.Parent</td>
</tr>
<tr>
<td>Web service lock holders</td>
<td>Agent.getLockHolders();</td>
<td>NotesAgent.LockHolders</td>
</tr>
</tbody>
</table>

Web service design elements can be locked and unlocked the same as agents.

Here is a LotusScript example that demonstrates getting properties associated with the Web service context. The Web service has three operations.

```lotuscript
Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession
Dim agent As NotesAgent

Class GetAgentContext
    Sub NEW
        Set s = New NotesSession
        Set agent = s.CurrentAgent
    End Sub
End Class
```
Function getAgentName() As String
getAgentName = agent.Name
End Function

Function getEffectiveUserName() As String
getEffectiveUserName = s.EffectiveUserName
End Function

Function getDatabaseFileName() As String
Dim db As NotesDatabase
Set db = s.CurrentDatabase
getDatabaseFileName = db.FileName
End Function
End Class

Below is the Java code.

import lotus.domino.*;
import lotus.domino.types.*;

public class GetAgentContext {
    Session s;
    AgentContext ac;

    public GetAgentContext() {
        s = WebServiceBase.getCurrentSession();
        try {
            ac = s.getAgentContext();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }

    public java.lang.String getAgentName() {
        String agentName = null;
        try {
            Agent agent = ac.getCurrentAgent();
            agentName = agent.getName();
        } catch(Exception e) {
            e.printStackTrace();
        }
        return agentName;
    }

    public java.lang.String getEffectiveUserName() {
        String userName = null;
        try {
            userName = ac.getEffectiveUserName();
        } catch(Exception e) {
            e.printStackTrace();
        }
        return userName;
    }

    public java.lang.String getCurrentDatabase() {
        String dbFileName = null;
        try {
            Database db = ac.getCurrentDatabase();
            dbFileName = db.getFileName();
        } catch(Exception e) {
            e.printStackTrace();
        }
        return dbFileName;
    }
}
Complex data types

Operations that use the following model do not require complex data types:

- A single scalar output value (or no output value)
- Scalar input values (or no input values)

Operations returning more than one scalar output value or taking more than scalar input values require complex data types. The use of complex data types allows the movement of large and varied data structures.

The following sections discuss complex data types:

- Arrays
- Classes
- Inout and output parameters

Arrays

Arrays map to a complexType WSDL element named ArrayOf suffixed by the data type. The WSDL defines the complexType element as an array.

For example, the following operation, implemented as a Java method, returns a String array.

```java
public java.lang.String[] getAll() {
    String[] info = new String[3];
    try {
        info[0] = ac.getEffectiveUserName();
        info[1] = s.getPlatform();
        info[2] = s.getNotesVersion(); }
    catch(Exception e) {
        e.printStackTrace(); }
    return info;
}
```

The Java String array maps to a WSDL complexType element named ArrayOf_xsd_string which is defined as an array of type string. The message that defines the return value for the getAll operation (getAllResponse) has one part whose type is ArrayOf_xsd_string.

```xml
<wSDL:types>
    <wsdl:types>
        <wsdl:types>
            <wsdl:types>
                <wsdl:types>
                    <wsdl:types>
                        <wsdl:types>
                            <wsdl:types>
                                <wsdl:types>
                                    <wsdl:types>
                                        <wsdl:types>
                                            <wsdl:types>
                                                <wsdl:types>
                                                    <wsdl:types>
                                                        <wsdl:types>
                                                            <wsdl:types>
                                                                <wsdl:types>
                                                                    <wsdl:types>
                                                                        <wsdl:types>
                                                                            <wsdl:types>
                                                                                <wsdl:types>
                                                                                    <wsdl:types>
                                                                                        <wsdl:types>
                                                                                            <wsdl:types>
                                                                                                <wsdl:types>
                                                                                                    <wsdl:types>
                                                                                                        <wsdl:types>
                                                                                                            <wsdl:types>
                                                                                                                <wsdl:types>
                                                                                                                    <wsdl:types>
                                                                                                                        <wsdl:types>
                                                                                                                            <wsdl:types>
                                                                                                                                <wsdl:types>
                                                                                                                                    <wsdl:types>
                                                                                                                                        <wsdl:types>
                                                                                                                                             <wsdl:types>
                                                                                                                                                <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                    <wsdl:types>
                                                                                                                                                   <wsdl:types>
                                                                                                                                                        <wsdl:types>
                                                                                            </wsdl:types>
                                                                                        </wsdl:types>
                                                                                    </wsdl:types>
                                                                                </wsdl:types>
                                                                            </wsdl:types>
                                                                        </wsdl:types>
                                                                    </wsdl:types>
                                                                </wsdl:types>
                                                            </wsdl:types>
                                                        </wsdl:types>
                                                    </wsdl:types>
                                                </wsdl:types>
                                            </wsdl:types>
                                        </wsdl:types>
                                    </wsdl:types>
                                </wsdl:types>
                            </wsdl:types>
                        </wsdl:types>
                    </wsdl:types>
                </wsdl:types>
            </wsdl:types>
        </wsdl:types>
    </wsdl:types>
</wSDL:types>
```
In LotusScript, we cannot return an array to a Web service consumer. The language rules require that an array return value be defined as a Variant which does not provide enough information to interpret the type when the WSDL is generated. The work-around is to put the array in a class as shown below.

```
Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession

Class infoArray
    Public info() As String
End Class

Class GetSessionInfo
    Sub NEW
        Set s = New NotesSession
    End Sub

    Function getItAll() As infoArray
        Set getItAll = New infoArray
        Redim getItAll.info(1 To 3)
        getItAll.info(1) = s.EffectiveUserName
        getItAll.info(2) = s.Platform
        getItAll.info(3) = s.NotesVersion
    End Function
End Class
```

**Classes**

Classes map to a complexType WSDL element named after the class. Here's a Java example that provides the same data as the Arrays example, but instead of returning an array, we return an object.

```
public InfoClass getAll2() {
    InfoClass info = new InfoClass();
    try {
        info.effectiveUserName = ac.getEffectiveUserName();
        info.platform = s.getPlatform();
        info.notesVersion = s.getNotesVersion();
    } catch(Exception e) {
        e.printStackTrace();
    }
    return info;
}

public class InfoClass {
    public String effectiveUserName;
    public String platform;
    public String notesVersion;
}
```

The Java class InfoClass maps to a complexType of the same name. The complexType has three elements, each of type xsd:string, named after the public data elements in the Java InfoClass class.
- <wsdl:types>
  - <schema targetNamespace="urn:DefaultNamespace"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
  - <complexType name="InfoClass">
    - <sequence>
      <element name="notesVersion"
        nillable="true" type="xsd:string" />
      <element name="platform"
        nillable="true" type="xsd:string" />
      <element name="effectiveUserName"
        nillable="true" type="xsd:string" />
    </sequence>
  </complexType>
</schema>
</wsdl:types>
- <wsdl:message name="getAll2Response">
  <wsdl:part name="getAll2Return" type="impl:InfoClass" />
</wsdl:message>

Here is the LotusScript equivalent.

Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession

Class InfoClass
  Public EffectiveUserName As String
  Public Platform As String
  Public NotesVersion As String
End Class

Class GetSessionInfo
  Sub NEW
    Set s = New NotesSession
  End Sub

  Function getItAll2() As InfoClass
    Set getItAll2 = New InfoClass
    getItAll2.EffectiveUserName = s.EffectiveUserName
    getItAll2.Platform = s.Platform
    getItAll2.NotesVersion = s.NotesVersion
  End Function
End Class

Inout and output parameters

Where an output message has one part, whether it be a simple or complex type, the output maps to the return value of a function or method. If an output message has more than one part, the output maps to parameters as well as or instead of a return value. The exact mapping depends on the input parts and how they combine with the output parts. If the first output part does not match any input part and the remaining output parts match the input parts, then the first output part maps to a function or method return value and the remaining parts map to inout parameters.

Otherwise, matching input and output parts map to inout parameters, non-matching input parts map to input parameters, and non-matching output parts map to output parameters. In this case, there is no return value and for LotusScript, subs are used instead of functions.
This WSDL excerpt is a variation of our first example that sends back the input values in the response:

```xml
<wss:message name="getNthSubjectResponse">
  <wss:part name="getNthSubjectReturn" type="xsd:string" /> (1)
  <wss:part name="dbname" type="xsd:string" /> (2)
  <wss:part name="viewname" type="xsd:string" /> (3)
  <wss:part name="n" type="xsd:int" /> (4)
</wss:message>
<wss:message name="getNthSubjectRequest">
  <wss:part name="dbname" type="xsd:string" /> (2)
  <wss:part name="viewname" type="xsd:string" /> (3)
  <wss:part name="n" type="xsd:int" /> (4)
</wss:message>
```

(1) One output part does not match an input part -- getNthSubjectReturn. This part maps to a function or method return value. (2)(3)(4) The three remaining output parts -- dbname, viewname, and n -- are the same as the three input parts. These parts map to three inout parameters.

Inout and output parameters cannot be primitive data types. Standard Java provides a package javax.xml.rpc.holders with methods for holding inout and output parameters of various types. Lotus Domino maps inout and output parameters to these classes, which are shown below:

- BigDecimalHolder
- BigIntegerHolder
- BooleanHolder
- BooleanWrapperHolder
- ByteArrayHolder
- ByteHolder
- ByteWrapperHolder
- CalendarHolder
- DoubleHolder
- DoubleWrapperHolder
- FloatHolder
- FloatWrapperHolder
- IntegerWrapperHolder
- IntHolder
- LongHolder
- LongWrapperHolder
- ObjectHolder
- QNameHolder
- ShortHolder
- ShortWrapperHolder
- StringHolder

These classes have a public variable named "value" that the application code can get and set. The following example is a variation of getNthSubject that returns a String as before, but makes the three parameters inout through the use of StringHolder and IntHolder classes. The values of the parameters are passed back to the consumer in the SOAP response.
import lotus.domino.*;
import lotus.domino.types.*;

public class GetSubject {

    Session s;

    public GetSubject() {
        s = WebServiceBase.getCurrentSession();
    }

    public String getNthSubject(javax.xml.rpc.holders.StringHolder dbname,
        javax.xml.rpc.holders.StringHolder viewname,
        javax.xml.rpc.holders.IntHolder n) {
        String subject = null;
        try {
            Database db = s.getDatabase(null, dbname.value);
            if (!db.isOpen()) subject = "Cannot open database " + dbname.value;
            else {
                View view = db.getView(viewname.value);
                if (view == null) subject = "Cannot open view " + viewname.value;
                else {
                    Document doc = view.getNthDocument(n.value);
                    if (doc == null) subject = "Cannot get document " + n.value;
                    else {
                        if (doc.hasItem("Subject"))
                            subject = doc.getItemValueString("Subject");
                        else subject = "Document does not have Subject";
                    }
                }
            }
        } catch(Exception e) {
            e.printStackTrace();
        }
        return subject;
    }
}

For LotusScript, the include file lsxsd.lss defines the following holder classes for inout and output parameters.

- BOOLEAN_HOLDER
- BOOLEANARRAY_HOLDER
- BYTE_HOLDER
- BYTEARRAY_HOLDER
- DOUBLE_HOLDER
- DOUBLEARRAY_HOLDER
- INTEGER_HOLDER
- INTEGERARRAY_HOLDER
- LONG_HOLDER
- LONGARRAY_HOLDER
- SINGLE_HOLDER
- SINGLEARRAY_HOLDER
- STRING_HOLDER
- STRINGARRAY_HOLDER
- VARIANT_HOLDER
• VARIANTARRAY_HOLDER

These classes have a public variable named "Value" that the application code can get and set. The following LotusScript example is the same as the preceding in Java. The holder classes are used for the three inout parameters.

```lotusscript
Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession
Class GetSubject
    Sub NEW
        Set s = New NotesSession
    End Sub

    Function getNthSubject(dbname As String_Holder, _
                          viewname As String_Holder, _
                          n As Long_Holder) As String
        Dim db As NotesDatabase
        Dim view As NotesView
        Dim doc As NotesDocument
        Set db = s.GetDatabase("", dbname.Value)
        If Not(db.IsOpen) Then
            getNthSubject = "Cannot open database " & _
                            dbname.Value
            Exit Function
        End If
        Set view = db.GetView(viewname.Value)
        If view Is Nothing Then
            getNthSubject = "Cannot open view " & _
                            viewname.Value
            Exit Function
        End If
        Set doc = view.GetNthDocument(n.Value)
        If doc Is Nothing Then
            getNthSubject = "Cannot get document " & _
                            n.Value
            Exit Function
        End If
        If doc.HasItem("Subject") Then
            getNthSubject = doc.GetItemValue("Subject")(0)
        Else
            getNthSubject = "Document does not have Subject"
        End If
    End Function
End Class
```

The primitive data types and their XSD counterparts generally map back and forth. The exception is that an imported SOAPENC data type maps to an object. However, the object maps to an XSD data type on output to a generated WSDL.

**Data type mappings**

<table>
<thead>
<tr>
<th>Imported WSDL</th>
<th>Java data type</th>
<th>Generated WSDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>xsd:boolean</td>
<td>boolean</td>
<td>xsd:boolean</td>
</tr>
</tbody>
</table>
Java uses the wrapper classes defined in java.lang: java.lang.Boolean, java.lang.Byte, and so on. LotusScript uses the XSD_ classes defined in lsxsd.lss: XSD_BOOLEAN, XSD_BYTE, and so on. The LotusScript classes inherit the following methods:

Function GetValueAsString() As String
Sub SetValueAsString(value As String)

Note: In an upcoming Beta release, the name SetValueAsString will change to SetValueFromString.

Here is a Java example of an operation returning a java.lang.Boolean type:

```java
import lotus.domino.*;
import lotus.domino.types.*;

public class GetDatabaseInfo {
    Session s;
    AgentContext ac;
    ```
Database db;

public GetDatabaseInfo() {
    s = WebServiceBase.getCurrentSession();
    try {
        ac = s.getAgentContext();
        db = ac.getCurrentDatabase();
    } catch(Exception e) {
        e.printStackTrace();
    }
}

public Boolean doesViewExist(String viewName) {
    Boolean b = null;
    try {
        if (db.getView(viewName) == null)
            b = new Boolean(false);
        else
            b = new Boolean(true);
    } catch(Exception e) {
        e.printStackTrace();
    }
    return b;
}

The corresponding operation in LotusScript returns an XSD_BOOLEAN type:

Option Public
%INCLUDE "lsxsd.lss"
Dim s As NotesSession
Dim db As NotesDatabase
Class GetDatabaseInfo
    Sub NEW
        Set s = New NotesSession
        Set db = s.CurrentDatabase
    End Sub

    Function DoesViewExist(viewName As String) As XSD_BOOLEAN
        Set b = New XSD_BOOLEAN
        If db.GetView(viewName) Is Nothing Then
            Call b.SetValueAsString("False")
        Else
            Call b.SetValueAsString("True")
        End If
        Set DoesViewExist = b
    End Function
End Class

(2) LotusScript does not use a primitive for xsd:byte. It always maps to XSD_BYTE (the LotusScript primitive maps to xsd:unsignedByte). (3) LotusScript does not use a primitive for xsd:long. It always maps to XSD_LONG (the LotusScript primitive maps to xsd:int). (4) Java has no primitive for xsd:string. It always maps to java.lang.String.

Other XSD data types map to java.lang, java.math, java.util, and lotus.domino.types (new with Lotus Notes/Domino 7) objects in Java, and XSD_ objects in LotusScript.
<table>
<thead>
<tr>
<th>WSDL</th>
<th>Java data type</th>
<th>LotusScript data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>xsd:anyType</td>
<td>java.lang.Object</td>
<td>Variant (1)</td>
</tr>
<tr>
<td>xsd:anyURI</td>
<td>lotus.domino.types.URI</td>
<td>XSD_ANYURI</td>
</tr>
<tr>
<td>xsd:base64Binary soapenc:base64 (2)</td>
<td>byte[]</td>
<td></td>
</tr>
<tr>
<td>xsd:date</td>
<td>java.util.Date</td>
<td>XSD_DATE</td>
</tr>
<tr>
<td>xsd:dateTime</td>
<td>java.util.Calendar</td>
<td>XSD_DATETIME</td>
</tr>
<tr>
<td>xsd:decimal</td>
<td>java.math.BigDecimal</td>
<td>XSD_DECIMAL</td>
</tr>
<tr>
<td>xsd:duration</td>
<td>lotus.domino.types.Duration</td>
<td>XSD_DURATION</td>
</tr>
<tr>
<td>xsd:ENTITY</td>
<td>lotus.domino.types.Entity</td>
<td>XSD_ENTITY</td>
</tr>
<tr>
<td>xsd:ENTITIES</td>
<td>lotus.domino.types.Entities</td>
<td>XSD_ENTITIES</td>
</tr>
<tr>
<td>xsd:gDay</td>
<td>lotus.domino.types.GDay</td>
<td>XSD_GDAY</td>
</tr>
<tr>
<td>xsd:gMonth</td>
<td>lotus.domino.types.GMonth</td>
<td>XSD_GMONTH</td>
</tr>
<tr>
<td>xsd:gMonthDay</td>
<td>lotus.domino.types.GMonthDay</td>
<td>XSD_GMONTHDAY</td>
</tr>
<tr>
<td>xsd:gYear</td>
<td>lotus.domino.types.GYear</td>
<td>XSD_GYEAR</td>
</tr>
<tr>
<td>xsd:gYearMonth</td>
<td>lotus.domino.types.GYearMonth</td>
<td>XSD_GYEARMONTH</td>
</tr>
<tr>
<td>xsd:hexBinary</td>
<td>lotus.domino.types.HexBinary</td>
<td>XSD_HEXBINARY</td>
</tr>
<tr>
<td>xsd:ID</td>
<td>lotus.domino.types.Id</td>
<td>XSD_ID</td>
</tr>
<tr>
<td>xsd:IDREF</td>
<td>lotus.domino.types.IDRef</td>
<td>XSD_IDREF</td>
</tr>
<tr>
<td>xsd:IDREFS</td>
<td>lotus.domino.types.IDRefs</td>
<td>XSD_IDREFS</td>
</tr>
<tr>
<td>xsd:integer</td>
<td>java.math.BigInteger</td>
<td>XSD_INTEGER</td>
</tr>
<tr>
<td>soapenc:integer (3)</td>
<td>lotus.domino.types.Language</td>
<td>XSD_LANGUAGE</td>
</tr>
<tr>
<td>xsd:language</td>
<td>lotus.domino.types.Name</td>
<td>XSD_NAME</td>
</tr>
<tr>
<td>xsd:NCName</td>
<td>lotus.domino.types.NCName</td>
<td>XSD_NCNAME</td>
</tr>
<tr>
<td>xsd:negativeInteger</td>
<td>lotus.domino.types.NegativeInteger</td>
<td>XSD_NEGATIVEINTEGER</td>
</tr>
<tr>
<td>xsd:NMTOKEN</td>
<td>lotus.domino.types.NMToken</td>
<td></td>
</tr>
<tr>
<td>xsd:NMTOKENS</td>
<td>lotus.domino.types.NMTokens</td>
<td></td>
</tr>
<tr>
<td>xsd:nonNegativeInteger</td>
<td>lotus.domino.types.NonNegativeInteger</td>
<td></td>
</tr>
<tr>
<td>xsd:nonPositiveInteger</td>
<td>lotus.domino.types.NonPositiveInteger</td>
<td></td>
</tr>
<tr>
<td>xsd:NOTATION</td>
<td>lotus.domino.types.NotAtion</td>
<td></td>
</tr>
<tr>
<td>xsd:normalizedString</td>
<td>lotus.domino.types.NormalizedString</td>
<td></td>
</tr>
<tr>
<td>xsd:positiveInteger</td>
<td>lotus.domino.types.PositiveInteger</td>
<td></td>
</tr>
<tr>
<td>xsd:QName</td>
<td>javax.xml.namespace.QName</td>
<td></td>
</tr>
<tr>
<td>xsd:time</td>
<td>lotus.domino.types.Time</td>
<td></td>
</tr>
<tr>
<td>xsd:token</td>
<td>lotus.domino.types.Token</td>
<td></td>
</tr>
<tr>
<td>xsd:unsignedByte</td>
<td>lotus.domino.types.UnsignedByte</td>
<td></td>
</tr>
<tr>
<td>xsd:unsignedInt</td>
<td>lotus.domino.types.UnsignedInt</td>
<td></td>
</tr>
<tr>
<td>xsd:unsignedLong</td>
<td>lotus.domino.types.UnsignedLong</td>
<td></td>
</tr>
<tr>
<td>xsd:unsignedShort</td>
<td>lotus.domino.types.UnsignedShort</td>
<td></td>
</tr>
</tbody>
</table>

(1) A Variant maps to xsd:anyType on output to a generated WSDL. (2) soapenc:base64 maps to byte[] and Byte when imported from a WSDL. A generated WSDL always maps to xsd:base64Binary. (3) soapenc:decimal and soapenc:integer map to XSD:DECIMAL and XSD:INTEGER when imported from a WSDL. A generated WSDL always maps to xsd:decimal and xsd:integer. (4) xsd:unsignedByte maps to Byte when imported from a WSDL. Byte and XSD_UNSIGNEDBYTE both map to xsd:unsignedByte in a generated WSDL.

**Private procedures**

Domino Web services expose public functions, subs, and methods in the implementation class. Private procedures are hidden. Here is a revision of the GetSubject example that uses public procedures to expose the operations getFirstSubject, getLastSubject, and getNthSubject. Common code is provided through the private procedures openDatabase, openView, and getSubject.

```vbnet
Dim s As NotesSession
Dim db As NotesDatabase
Dim view As NotesView
Dim doc As NotesDocument
Dim msg As String
```
Class GetSubject

Sub NEW
    Set s = New NotesSession
End Sub

Function getFirstSubject(dbname As String, viewname As String) As String
    If openDatabase(dbname) Then
        If openView(viewname) Then
            Set doc = view.GetFirstDocument
            If doc Is Nothing Then
                msg = "Cannot get first document"
            Else
                Call getSubject
            End If
        End If
    End If
    getFirstSubject = msg
End Function

Function getLastSubject(dbname As String, viewname As String) As String
    If openDatabase(dbname) Then
        If openView(viewname) Then
            Set doc = view.GetLastDocument
            If doc Is Nothing Then
                msg = "Cannot get last document"
            Else
                Call getSubject
            End If
        End If
    End If
    getLastSubject = msg
End Function

Function getNthSubject(dbname As String, viewname As String, n As Integer) As String
    If openDatabase(dbname) Then
        If openView(viewname) Then
            Set doc = view.GetNthDocument(n)
            If doc Is Nothing Then
                msg = "Cannot get document " & n
            Else
                Call getSubject
            End If
        End If
    End If
    getNthSubject = msg
End Function

Private Function openDatabase(dbname As String) As Boolean
    Set db = s.GetDatabase("", dbname)
    If db.IsOpen Then
        openDatabase = True
    Else
        openDatabase = False
        msg = "Cannot open database " & dbname
    End If
End Function

Private Function openView(viewname As String) As Boolean
    Set view = db.GetView(viewname)
    If view Is Nothing Then
        openView = False
        msg = "Cannot open view " & viewname
    Else
        openView = True
    End If
End Function
Private Sub getSubject
  If doc.HasItem("Subject") Then
    msg = doc.GetItemValue("Subject")(0)
  Else
    msg = "Document does not have Subject"
  End If
End Sub

End Class

Here is the example in Java.

```java
import lotus.domino.*;
import lotus.domino.types.*;

public class GetSubject {

  Session s;
  Database db;
  View view;
  Document doc;
  String msg;

  public GetSubject() {
    s = WebServiceBase.getCurrentSession();
  }

  public String getFirstSubject(String dbname, String viewname) {
    try {
      if (openDatabase(dbname)) {
        if (openView(viewname)) {
          doc = view.getFirstDocument();
          if (doc == null)
            msg = "Cannot get first document ";
          else
            getSubject();
        }
      }
    }
    catch(Exception e) {
      e.printStackTrace();
    }
    return msg;
  }

  public String getLastSubject(String dbname, String viewname) {
    try {
      if (openDatabase(dbname)) {
        if (openView(viewname)) {
          doc = view.getLastDocument();
          if (doc == null)
            msg = "Cannot get last document ";
          else
            getSubject();
        }
      }
    }
    catch(Exception e) {
      e.printStackTrace();
    }
    return msg;
  }

  public String getNthSubject(String dbname, String viewname, int n) {
    try {
```
if (openDatabase(dbname)) {
    if (openView(viewname)) {
        doc = view.getNthDocument(n);
        if (doc == null)
            msg = "Cannot get document " + n;
        else
            getSubject();
    }
}

private boolean openDatabase(String dbname) {
    boolean b = false;
    try {
        db = s.getDatabase(null, dbname);
        if (db.isOpen())
            b = true;
        else
            msg = "Cannot open database " + dbname;
    }
    catch(Exception e) {
        e.printStackTrace();
    }
    return b;
}

private boolean openView(String viewname) {
    boolean b = false;
    try {
        view = db.getView(viewname);
        if (view != null)
            b = true;
        else
            msg = "Cannot open view " + viewname;
    }
    catch(Exception e) {
        e.printStackTrace();
    }
    return b;
}

private void getSubject() {
    try {
        if (doc.hasItem("Subject"))
            msg = doc.getItemValueString("Subject");
        else
            msg = "Document does not have Subject";
    }
    catch(Exception e) {
        e.printStackTrace();
    }
}

Conclusion
Lotus Notes/Domino 7 supports the provider side of Web services through agent-like design elements coded in Java or LotusScript. The Web service must reside on a Domino 7 server with HTTP enabled, except that a Web service can be tested and debugged through a Web preview.
on a Notes client. Consumers access Domino Web services through SOAP-encoded HTTP POST requests.

Web service operations map to public Java methods and public LotusScript functions and subs. Web service data parts map to parameters and return values. Where possible, XSD data types map to Java and LotusScript primitives. Otherwise, complexType elements map to objects.

This article is based on the Beta 2 release of Lotus Notes/Domino 7. Enhancements may be made as development progresses. For example, a future release is expected to support placement of Web service code in script libraries.
Related topics

- Lotus Domino supports SOAP 1.1 and WSDL 1.1. For specifications and background information, see the following W3C documents:
  - Simple Object Access Protocol (SOAP) 1.1
  - Web Services Description Language (WSDL) 1.1
  - Web Services Architecture
  - Web Services Activity
- For more information about SOAP formats, read the developerWorks article, "Which style of WSDL should I use?" by Russell Butek.