

WEB SERVICES BOOTCAMP

Bill Buchan - HADSL

AGENDA

- Introduction
- Web Services Overview
- Using Domino to provide Web Services
- Using Notes to consume Web Services
- Summary, Q+A



INTRODUCTION

- I am Bill Buchan. I am:
 - A blogger - <http://www.billbuchan.com>
 - I was blogging when it was called 'Talking rubbish on the internet'.
 - Some things never change.
 - A principal of HADSL - <http://www.hadsl.com>

WE DO COMPLICATED STUFF

- Our product deals with enterprise user and resource/service management
- Continually pushing the Notes platform
- Currently coding around the OOO and ID Vault API's
- All my presentations are based on the work we do in FirM

LOTS OF CONTENT HERE

- This takes you from a low level in a subject to a good level of expertise
- Little existing knowledge is assumed
 - But we assume you are developers, and are familiar with LotusScript
- No xPages, Java
- This is a 2 hour presentation, squeezed into 45 minutes
 - This will be fast

I HAD THE PLEASURE

- Of working in Holland for many years
- Be gentle!
- Questions at the end...





A QUICK OVERVIEW

- Web services are:
 - A standard application to application protocol for exchanging structured data
 - Usually (but not always) using the http protocol
 - Usually (but not always) using XML
 - Usually we provide meta-information on what our service will do, using a language called WSDL (Web Service Description Language) – formatted in XML.
- Web services are Language independent, Platform independent, Data format representation independent
- But hopefully you all knew this already...

WEB SERVICES ARCHITECTURE

- The players:
 - The Web Service server
 - Can answer questions on what it can do
 - Can accept requests from Web Service clients
 - Can respond to requests from Web service clients

WEB SERVICES PROTOCOL

- A typical web service session looks like:
 - ▶ Client to Web Service Server
 - “Tell me the answer to this question”
 - ▶ Web Service Server
 - “Here it is”
- Conclusion:
 - ▶ The clients drive the conversation
 - ▶ The Server cannot initiate a conversation with the client.

WEB SERVICES PROTOCOL

- The Web Services Protocol:
 - ▶ The client decides it needs to ask the server for some information
It constructs the relevant XML based web services query (either dynamically or in a static fashion depending on the complexity of the application)
 - ▶ It then connects to the Web Service Server
 - Pushing the web request up as an HTTP 'Post' request
 - It waits for a response
 - ▶ The Web service server then 'Posts' back an XML payload representing the answer
 - ▶ The client then unpacks and interprets the data.

THIS SOUNDS HARD...

- Not really.
 - Its a combination of web protocol and XML construction and unpacking
 - Most of the hard work is done for you by the various Web Service servers and Consumers
 - Its fairly easy to debug using the correct tools

A SAMPLE APPLICATION

- Fairly simple Domino Contact Database
 - A single 'Contact' form with some information
 - Not meant to be a 'real-world' example - Its here to demonstrate the techniques
- All example code is in the database
 - Simple to figure out and 'research'
 - Probably isn't best practice!

EXAMPLE DATABASE:

Lotusphere 2008 LS2008 Web Services Bootcamp
[Bill Buchan](#)
hads!

New.. Goto.. Edit

Name	Contact
Barney Rubble	eMail: Barney.Rubble@bedrock.com Personal: 1 712 313 5555
Fred Flintstone	Mobile: 1 711 711 5555 eMail: fred@bedrock.com Personal: 1 721 211 5555

*LS2008 Web Services Bootcamp hosted on this computer in location: customers\LS2008\Contacts.nsf.
You are logged in as: Bill S Buchan.*

Disconnected | idm-demo1

EXAMPLE DATABASE: CONTACTS.NSF

Contact: Barney Rubble

Contact:	Contact Details
First Name: [Barney]	Tel: [1 712 313 5545]
Middle Initials: []	Mobile Tel: []
Last Name: [Rubble]	eMail: [Barney.Rubble@bedrock.com]
Full Name: Barney Rubble	Home Tel: [1 712 313 5555]
Notes: [Small but perfectly formed]	

HOW DO WE TEST A WEB SERVICE?

- I recommend SoapUI
 - Its free!
 - **<http://www.soapui.org>**
- It allows you to interrogate a web service
 - And build test case with real data

SOAPUI: EXAMPLE

The screenshot displays the SoapUI 1.7.1 application window. The main area shows a SOAP request and response for the endpoint `http://bes1.jsector.net/servlet/Portal`. The request is a SOAP envelope with a header and a body containing a `urn:GETAPPLICATIONTITLE` element. The response is a SOAP envelope with a body containing a `ns1:GETAPPLICATIONTITLEResponse` element, which includes a `GETAPPLICATIONTITLEReturn` element.

SOAP Request:

```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:GETAPPLICATIONTITLE soapenv:encoding="UTF-8" xmlns:urn="urn:GETAPPLICATIONTITLE"/>
  </soapenv:Body>
</soapenv:Envelope>
```

SOAP Response:

```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:GETAPPLICATIONTITLEResponse soapenv:encoding="UTF-8" xmlns:ns1="urn:GETAPPLICATIONTITLE">
      <GETAPPLICATIONTITLEReturn xsi:type="xsd:string">
        </GETAPPLICATIONTITLEReturn>
    </ns1:GETAPPLICATIONTITLEResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

The interface also shows a log window at the bottom with the following messages:

```
Mon Dec 10 07:59:14 GMT 2007:INFO:Missing folder [C:\Program Files\eviware\soapUI-1.7.1\bin\ext] for external libraries
Mon Dec 10 07:59:14 GMT 2007:INFO:Loading workspace from [\\.\PSF\Home\default-soapui-workspace.xml]
Mon Dec 10 07:59:14 GMT 2007:INFO:Loaded project from [\\.\PSF\Home\Documents\test-soapui-project.xml]
Mon Dec 10 07:59:15 GMT 2007:INFO:Loaded project from [\\.\PSF\Home\Documents\BBPortal-coaxis-project.xml]
```



DOMINO WEB SERVICE SERVER

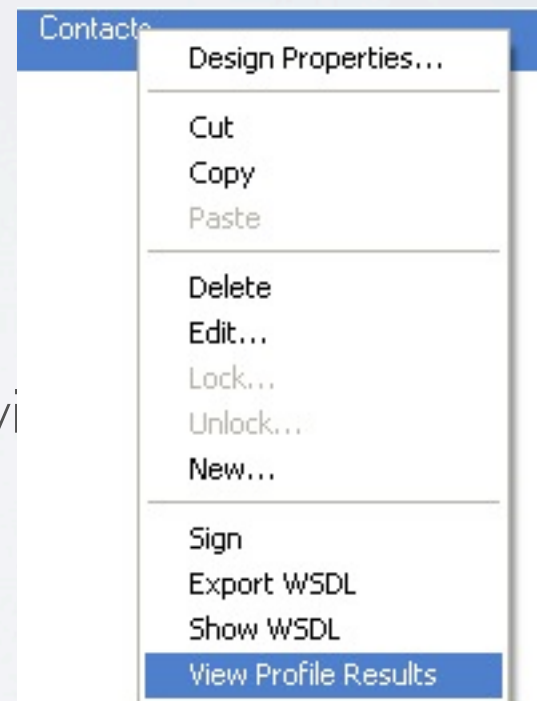
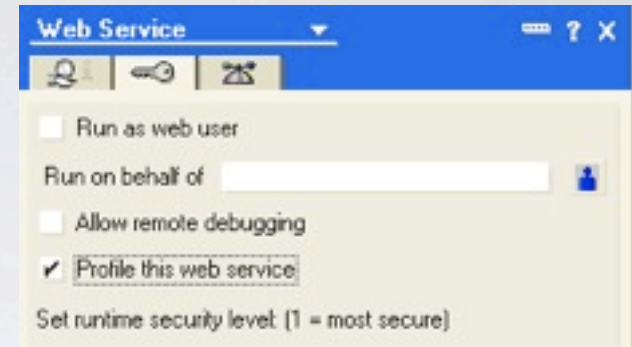
- By far the simplest is to use a LotusScript Web Service
 - The majority of the work is done for you!
- Servlets should be avoided unless
 - You are running on Domino v6.5.x or older
 - You absolutely require persistence

LOTUSSCRIPT WEB SERVICES

- Introduced in Lotus Domino 7
- Robust
- Code the web service using LotusScript
- Fairly high performance but
 - Remember: Lack of persistence between calls, so
 - Be aware of what your code is doing!
 - Use 'Agent Profiling' to see timings...

PROFILING A WEB SERVICE

- LotusScript web services are very similar to LotusScript agents
 - So you can profile them.
- Enable Profiling on the Web Services Properties tab
- View the last run profile:
 - Right click on the Web Service
 - View Profile Results



A SAMPLE PROFILE

- This shows how long the agent took to run, and how long the agent spent inside the Notes Object Interface
 - In our case, 761ms total, and 0ms within NOI
 - One call to CurrentDatabase, and one call to Title.

Contacts Profile

15/12/2007 23:42:43 GMT
Elapsed time: 761 msec
Methods profiled: 2
Total measured time: 0 msec

Class	Method	Operation	Calls	Time
Database	Title	Get	1	0
Session	CurrentDatabase	Get	1	0

DESIGNING A WEB SERVICE

- Spend time thinking about how your consumer will use this web service.
 - More time thinking, less time coding
 - You do NOT want to change this web service interface once its published
 - Instead - roll out a new web service
 - Think through client use cases

CONTACTS.NSF

- In our case we wish to:
 - Be able to list all users by Name, by Department
 - Be able to get details on a particular user
 - Be able to search for users
- Do we
 - Pass all fields back as web service items?
 - Pass back an array of fields?
- In this case, we pass back Web Service Items
 - Assume data mapping knowledge and responsibility.

WEB SERVICE DESIGN

- We design our web service interface within a 'Class' in LotusScript.
 - At this point, you probably wished that you studied the Object Orientated LotusScript sessions I used to give...
- Our web service will expose all 'public' methods and properties
 - We do not have to put all the code in one 'class'
 - Beware of extending existing classes - you might expose more than you want!
 - Beware over-exposure...

WEB SERVICE DESIGN

- LotusScript does NOT allow functions to return Arrays.
 - This is a language limitation, not a Web Service Limitation
 - It can return simple types - but not variants (web services don't support 'em)
 - Instances of other classes...
- We need to use arrays to return results
 - For instance, we need to return an array of Zero or more entries to list our users. How do we do that?
- We can define a class and return an instance of that class...

RETURNING COMPLEX TYPES

- We can define a class called 'ReturnArray' which has a single array as a public member.
 - The constructor initialises the array so that it has one blank entry.
- We can then populate this array with one or more entries.
- Nd7 supports arrays with one element or more - nd8 supports 'empty' arrays.

```
Class returnArray
    Public S() As String
    Sub new
        Redim S(0)
        S(0) = ""
    End Sub
End Class
```

COMPLEX TYPES...

- We shall return our Person contact record using a class.
 - We DON'T have a constructor
 - All elements are Public - so it'll be exposed to the Web Service as data.
 - Our property names will be used by the web service as field names
 - Since LotusScript is case insensitive, the web service field names will be UPPERCASE.

Class Person

```
Public FirstName As String  
Public MiddleInitials As String  
Public LastName As String  
Public FullName As String
```

```
Public Telephone As String  
Public Cellphone As String  
Public eMail As String  
Public HomePhone As String  
Public Department as String
```

```
Public Notes As String
```

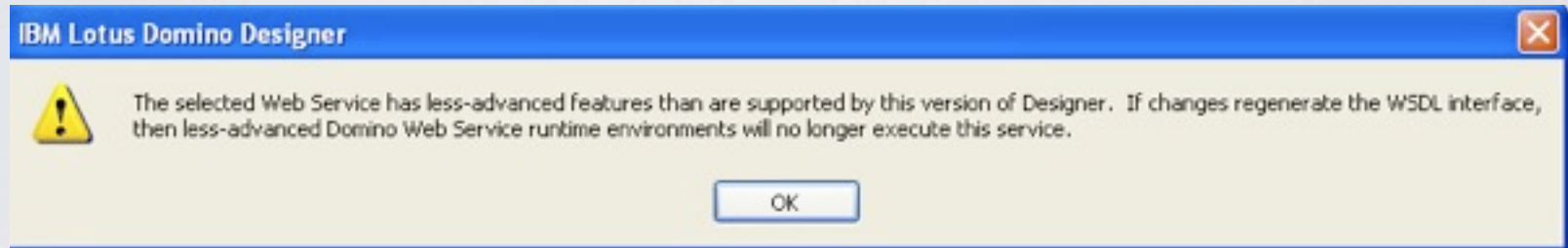
End Class

LETS MAKE IT BETTER

- We need to add:
 - Logging. We need to know when it works, and importantly, when it does not
 - Error Handling. We need to pass errors back to the consumer
 - Security. We might want to harden this service somewhat
- We committed the following sins:
 - We put all the code in the web service itself, making it hard to test. Best to consider embedding most of the code in script libraries
 - We've tightly bound the data structures in our data to our web service

ND7 AND ND8 DIFFERENCES

- Web services built or compiled in Notes 8 Designer WONT run on ND7 anymore!

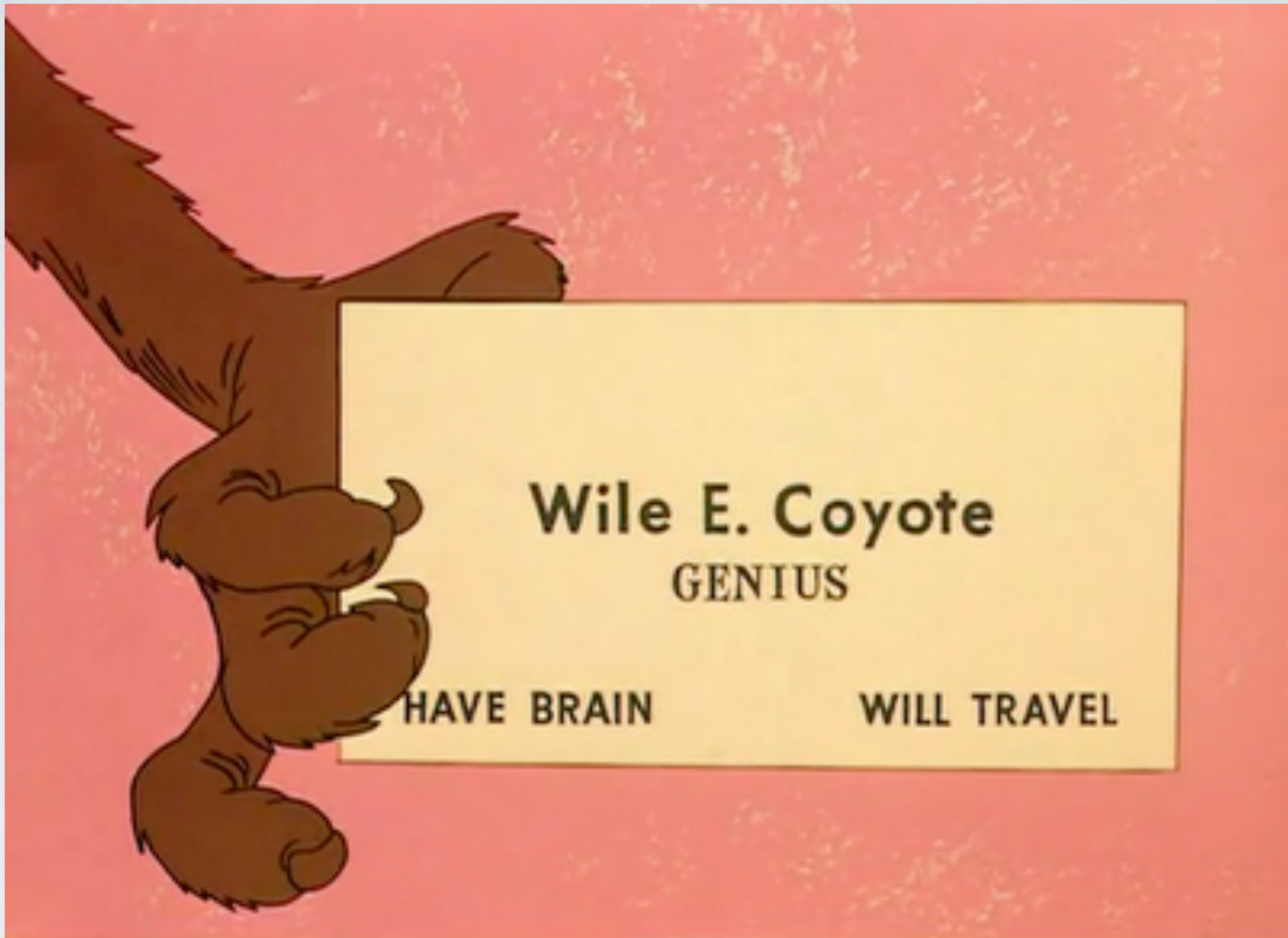


So be careful about what web services you host on which servers, and which version of designer you use.

- Just bear this version difference in the same manner as all other version differences.

ND7 AND ND8 DIFFERENCES

- In ND7, the class used for the web service has to reside in the web service design element. In ND8, this can be in a script library.
 - This means that for complex services, you can place all the business code in a script library, allowing simple construction of a test harness.
- ND8 now supports code in Java Libraries
- ND8 supports more error handling SOAP elements
- ND8 supports 'empty' arrays - nd7 does not



Wile E. Coyote

GENIUS

HAVE BRAIN

WILL TRAVEL

CONSUMING A WEB SERVICE

- Why consume web services?
 - To gain access to information in other systems
 - To prevent the synchronisation and replication of external data sources to Notes
 - Real time, up to date data lookup
- Where?
 - We could use a scheduled Java agent in Domino to interrogate external services - no UI means you have to monitor and check
 - We could use client-driven code on the client workstation - be aware of network issues between the clients and the remote service

OVERVIEW

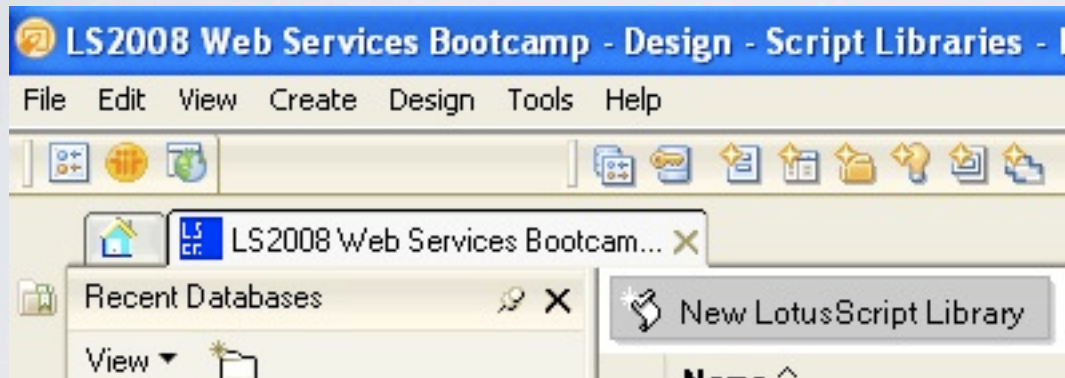
- Security
 - Remote web services may require
 - Username/password login, Encrypted username/password pairs in data, SSL.
- Things change
 - You may have to change your web service consumer to accommodate this
 - The remote web service may move
 - When designing, don't hard-code...

CONSUMING WEB SERVICES

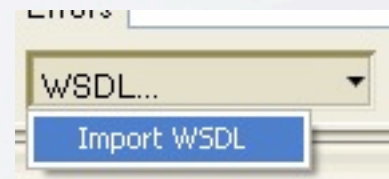
- Summary:
 - Notes 8 - Its very simple - It works in LotusScript and in Java
 - It does a lot of work for you
- How do I construct a Web Service Consumer in LotusScript?
 - Create a new, empty script library
 - Click on the Import WSDL button
 - It creates a Class definition
 - You can then “use” the script library and class.

CONSUMING WEB SERVICES

- On the nd8 basic client: create a new LotusScript Script Library:

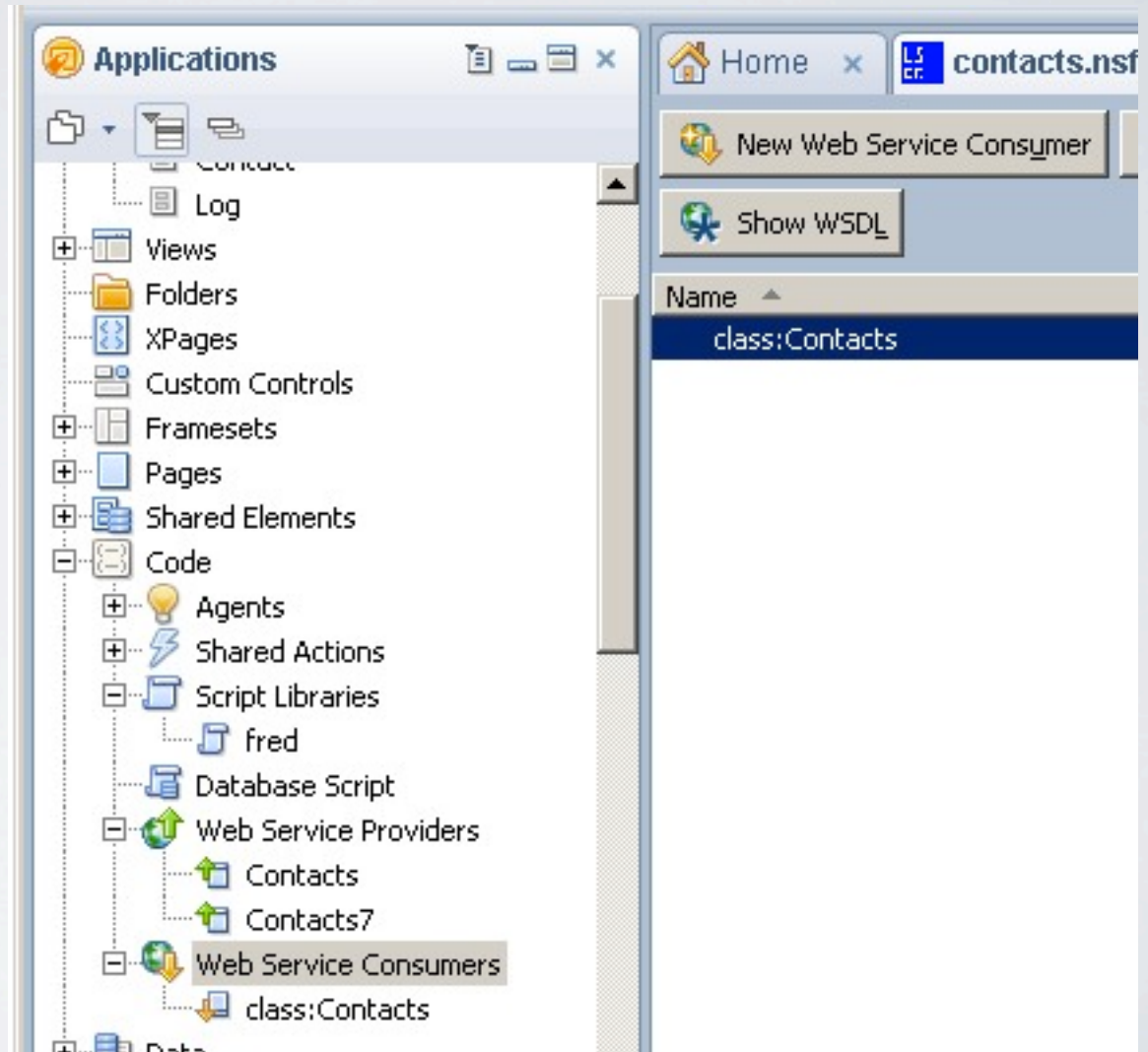


- Click on the WSDL button...



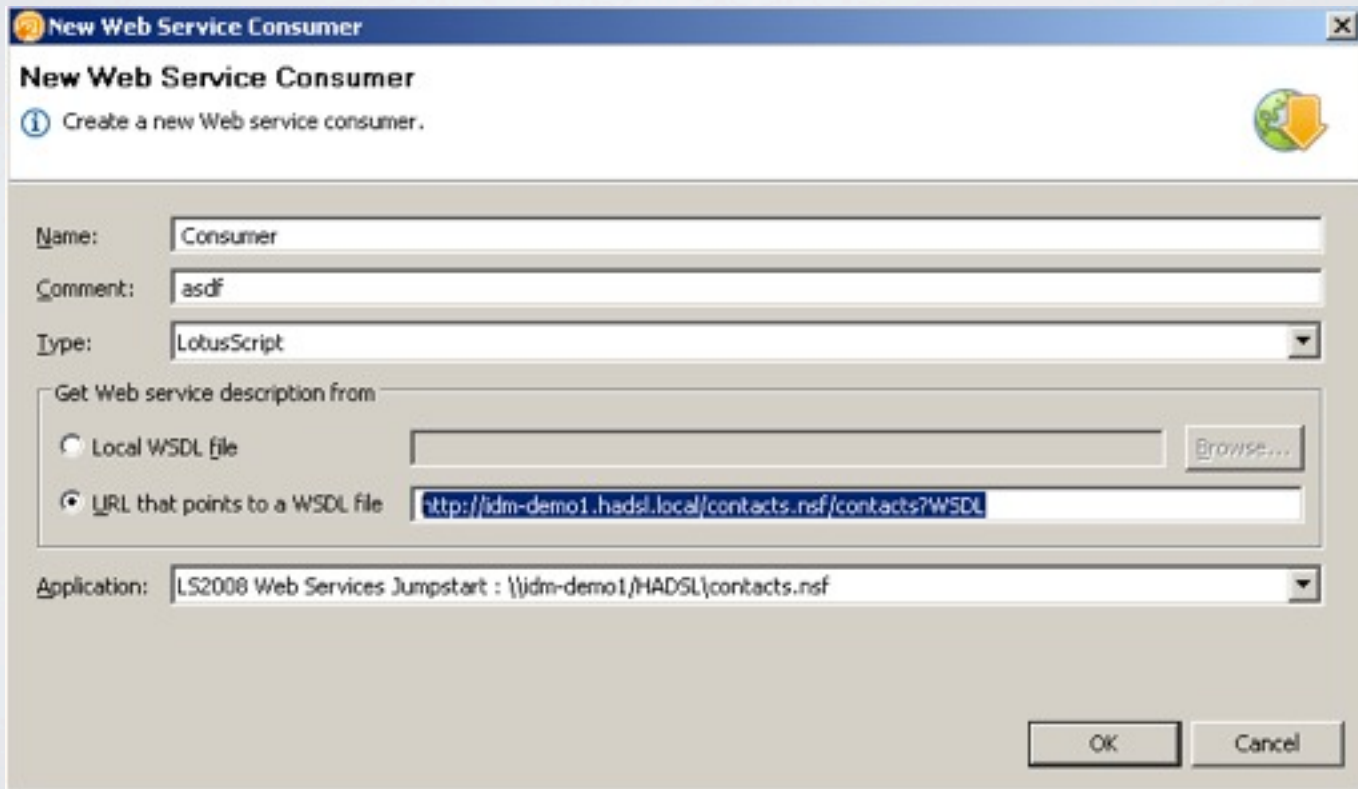
CONSUMING WEB SERVICES

- On nd8 standard designer, go to Code, Web services consumer



CONSUMING WEB SERVICES

- Click on 'New Web service consumer' and fill in the dialog:



New Web Service Consumer

Create a new Web service consumer.

Name:

Comment:

Type:

Get Web service description from

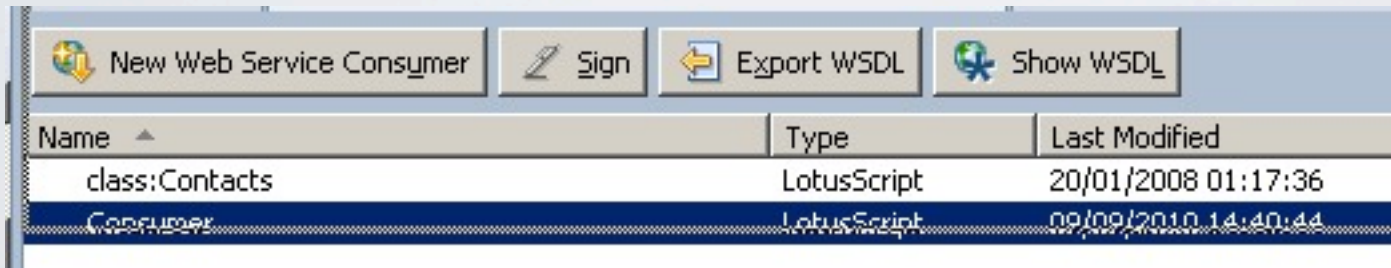
Local WSDL file

URL that points to a WSDL file

Application:

CONSUMING WEB SERVICES

- And then use the web service consumer library that has been created:

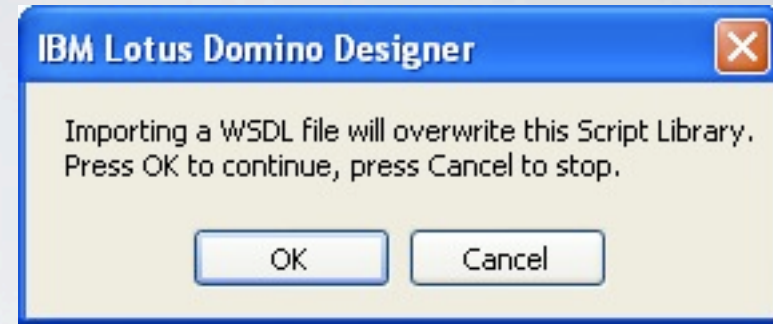


The screenshot shows a software interface for managing web service consumers. At the top, there are four buttons: 'New Web Service Consumer' (with a globe icon), 'Sign' (with a pencil icon), 'Export WSDL' (with a left-pointing arrow icon), and 'Show WSDL' (with a globe icon). Below the buttons is a table with three columns: 'Name', 'Type', and 'Last Modified'. The table contains two rows. The first row is 'class:Contacts' with type 'LotusScript' and last modified '20/01/2008 01:17:36'. The second row is 'Consumer' with type 'LotusScript' and last modified '09/09/2010 14:40:44'. The 'Consumer' row is highlighted with a blue background.

Name	Type	Last Modified
class:Contacts	LotusScript	20/01/2008 01:17:36
Consumer	LotusScript	09/09/2010 14:40:44

CONSUMING WEB SERVICES

- It needs a WSDL file. What's that?
 - You can open the WSDL definition for your web service in your browser
 - Use View Source, and save that as a WSDL file.
 - Select it...
- Designer will then construct helper classes and a main class which we can then call
- That's it!



```
Sub Initialize
```

```
Dim C As New Contacts
```

```
Dim V As Variant  
Set V = C.listUsers()
```

```
Forall thisUser In V.S  
    Print "Found User:" + thisUser  
End Forall
```

```
End Sub
```

GOTCHAS

- The entire LotusScript library is taken up with the computed Web service definition. Best not to make changes.
 - Next time its refreshed, you will lose those changes!
- If you change the web service, you must remember to update the Web Services consumer.
 - It doesn't take long - so don't forget it.
 - And when you change the web service consumer script library, you must recompile all the LotusScript that relies on it.
 - 'Tools, Recompile LotusScript' is your friend

GOTCHAS

- It requires the Notes 8 client to run on:
 - We as Application Developers deploy solutions on users' target notes clients
 - Notes 8 may not yet be adopted in your environment
 - This may help drive the upgrade process
- What if you need Web Service consumption now, and are not yet on Notes 8?
 - You can use MS COM objects.

SUMMARY

- Web Services
 - Are pretty straightforward
 - An important part of your armoury
 - Help break down barriers between Domino and other systems
 - Help prevent data duplication in your environment



QUESTIONS AND ANSWERS ?

- Please fill out your evaluations:
- Thank you for your time today