JavaAgent

Fetch SiteFinder Mobile Documents

WebServices

System Requirements:
Download Domino Designer **8.5.3** Environment (DDE)
http://www.ibm.com/developerworks/downloads/ls/dominodesigner/

Introduction:
Use JavaAgent code samples to create WebService provider and consumer using two different NSF files.   You will be creating the WebService provider on sitefindermobile.nsf and the consumer will connect to the WSDL on sitefindermobile.nsf using filebin.nsf. Build a connection to these Lotus Notes Domino Database(s) located on C Drive to get started...

Disclaimer:
Information contained in the following is presented as is. This tutorial assumes you have basic programming knowledge. All tutorials are based on an Eclipse/Eclipse-based software. Should you need to familiarize yourself with a certain Eclipse environment, prior to continuing this tutorial, please stop now and see our Tutorials page...

Replace Generated WebService Code

At this point we assume Domino Designer 8.5.3 is downloaded/installed. Create a Java WebService provider, which should generate an Untitled Java class, as shown below.

Untitled.java

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

public class Untitled{

  // This is a template implementation class for your web service. It
  // becomes extraneous if you import a WSDL document. Consumers of this
  // web service can call any public method in the implementation class.
  //
  // To obtain a Session object use this code:
  // Session s = WebServiceBase.getCurrentSession();
}
```

Copy and Paste WebService JavaAgent Code

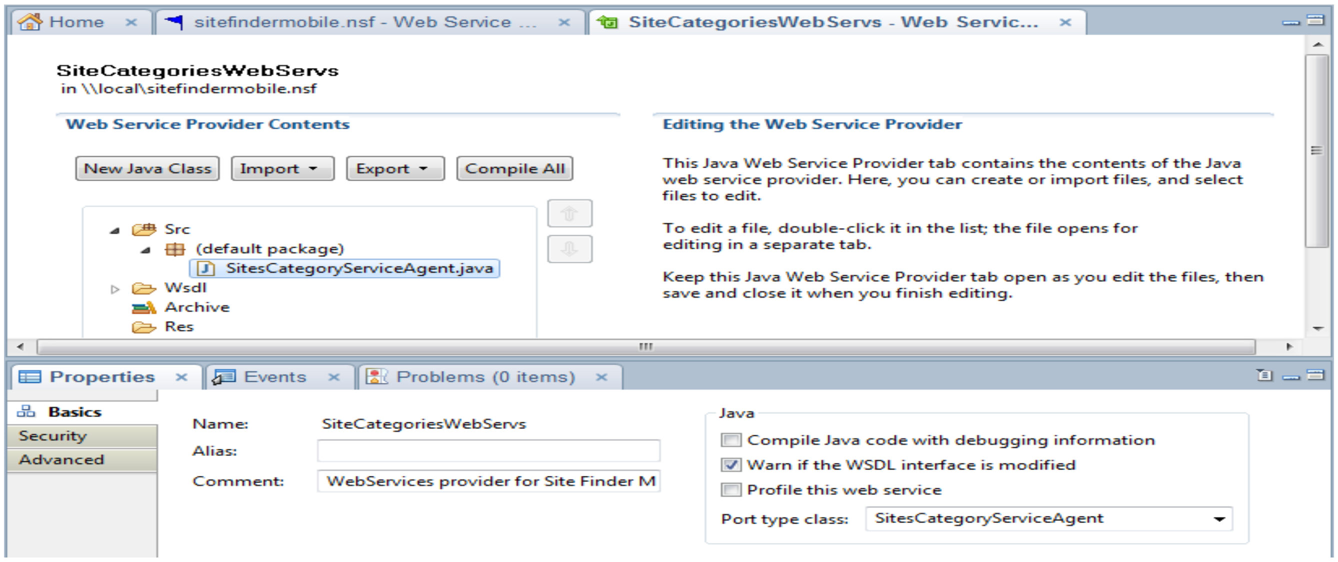
Copy and paste below code sample to your environment over the above generated 'Untitled' class. You might also want to make sure your code is making the proper connection to an existing view, and that your search criteria is solid and bound to the right key; it will save you a ton of work and time.

SiteCategoriesWebServs;

```
/**
 * Created: 2013.09.22.5.40.AM
 * SiteCategoriesWebServs | SitesCategoryServiceAgent.java
 * WebServices data for Xpages
 */
import lotus.domino.Database;
import lotus.domino.Document;
import lotus.domino.NotesException;
import lotus.domino.Session;
import lotus.domino.View;
import lotus.domino.WebServiceBase;
/**
 * @author Dököll Solutions, Inc.
 * @version 2013.09.22.5.40.AM
 *
 */
public class SitesCategoryServiceAgent {
      private Session session;
      private Database database;
      private View view;
      private Document document;
      public SitesCategoryServiceAgent() throws NotesException {
            session = WebServiceBase.getCurrentSession();
            // grab database from session obtained

            database = session.getDatabase("", "sitefindermobile.nsf");
            System.out.println("database loaded from session..." + database);
            // grab view from database

            view = database.getView("SiteList");
            System.out.println("view obtained from database..." + view);
      }
      //perform search
      public String getSiteCategory(String SiteName) {
            System.out.println("declare variable to load search results..");
            String SiteCategory = "";
            try {
                  //search view based on first column (Key)
                  System.out.println("view column search from database...");
                  System.out.println("using getDocumentByKey...");

                  document = view.getDocumentByKey(SiteName);
                  if (document == null) {
                        //alert user where no site are found
                        SiteCategory = "Search results for specific site type
returned null";
                        System.out.println("view obtained from database, but no
sites..." + SiteCategory);
                  } else {
                        //load site to user where site are found

                        SiteCategory = document.getItemValueString("Type");
                        System.out.println("view obtained, site Found..." +
SiteCategory);

                        if (SiteCategory.equalsIgnoreCase("")) {
```

```
                                                  //alert user where key retrieve null for Type
                                                  SiteCategory = "check view column, Type is empty...";
                                                  System.out.println("check view, column empty..." +
SiteCategory);
                                            }
                                    }
                        } catch (NotesException e) {
                                e.printStackTrace();
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                        return SiteCategory;
                }
}
```

Configure WebService Agent/Create WSDL:
We hope you are connecting to an existing view and your search criteria is up to snuff at this point. You want to check 'Warn if the WSDL Interface is modified' and select the Port Type class from the dropdown. Please note, once you hit save, the WSDL will be created, thus no turning back.


Home
sitefindermobile.nsf - Web Service ...
SiteCategoriesWebServs - Web Servic...

SiteCategoriesWebServs
in \\local\sitefindermobile.nsf

Web Service Provider Contents

New Java Class
Import
Export
Compile All
Src
(default package)
SitesCategoryServiceAgent.java
Wsdl
Archive
Res

Editing the Web Service Provider

This Java Web Service Provider tab contains the contents of the Java web service provider. Here, you can create or import files, and select files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

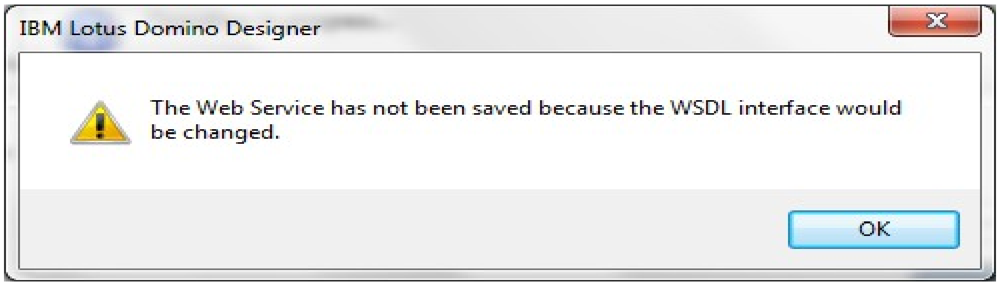
Keep this Java Web Service Provider tab open as you edit the files, then save and close it when you finish editing.

Name: SiteCategoriesWebServs
Alias:
Comment: WebServices provider for Site Finder M

Java
- [ ] Compile Java code with debugging information
- [x] Warn if the WSDL interface is modified
- [ ] Profile this web service

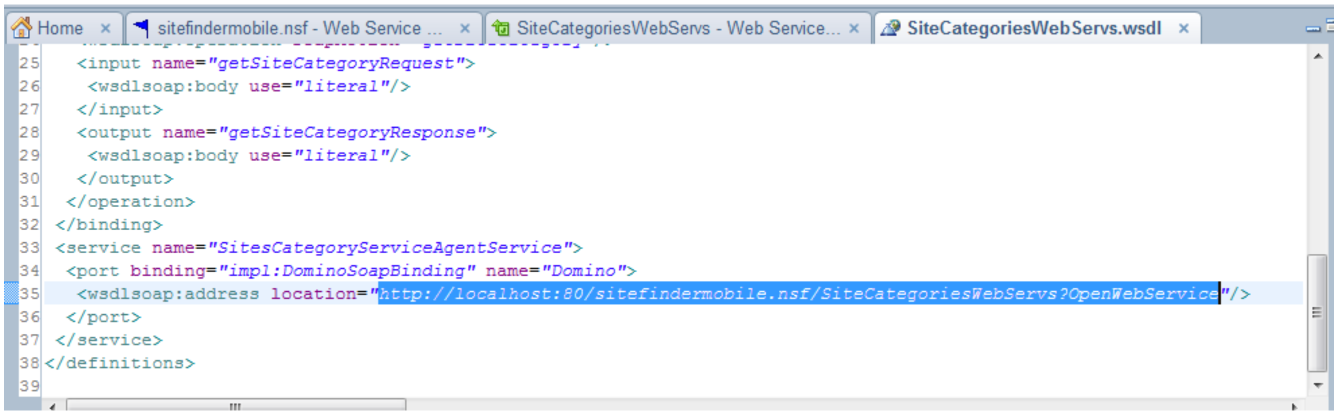
Port type class: SitesCategoryServiceAgent

Possible WebService Provider Error:
Make sure your WebService JavaAgent is completely written before generating the WebService file (WSDL). If you want to change code, you will get an error that the service cannot be saved. At which point, you may need to delete the code entirely and start over.


IBM Lotus Domino Designer

The Web Service has not been saved because the WSDL interface would be changed.

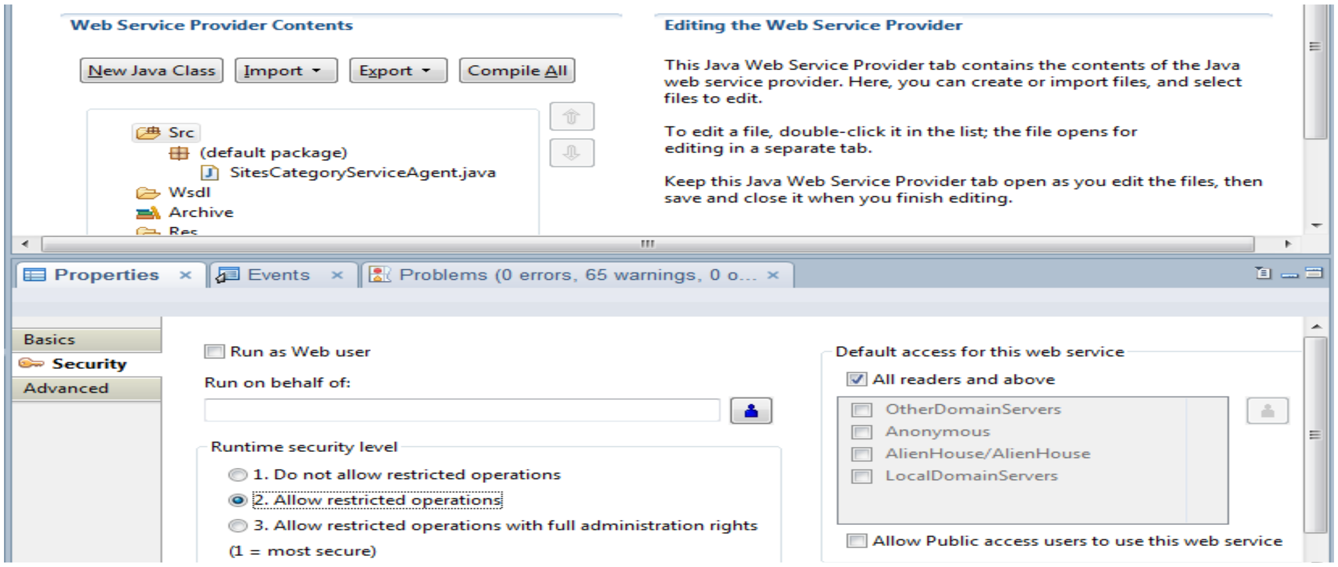
Check the generated WSDL:
It is a good idea to double-check the WSDL and modify the URL at the bottom of the XML, which can be defaulted to http://localhost, you want the full URL of the location of the WSDL reflected in this file. Go ahead and save the WSDL, you may be asked to regenerate the WSDL, it is okay to do this.



Home | sitefindermobile.nsf - Web Service ... | SiteCategoriesWebServs - Web Service... | SiteCategoriesWebServs.wsdl

```
   <input name="getSiteCategoryRequest">
    <wsdlsoap:body use="literal"/>
   </input>
   <output name="getSiteCategoryResponse">
    <wsdlsoap:body use="literal"/>
   </output>
  </operation>
 </binding>
 <service name="SitesCategoryServiceAgentService">
  <port binding="impl:DominoSoapBinding" name="Domino">
   <wsdlsoap:address location="http://localhost:80/sitefindermobile.nsf/SiteCategoriesWebServs?OpenWebService"/>
  </port>
 </service>
</definitions>
```

WebService Accessibility:
You can further check the Security of the WebService provider to be certain it can be reached by other programs/systems. Normally the second option in the security tab should do just fine.



Web Service Provider Contents

New Java Class | Import | Export | Compile All

Src
(default package)
SitesCategoryServiceAgent.java
Wsdl
Archive
Res

Editing the Web Service Provider

This Java Web Service Provider tab contains the contents of the Java web service provider. Here, you can create or import files, and select files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Web Service Provider tab open as you edit the files, then save and close it when you finish editing.

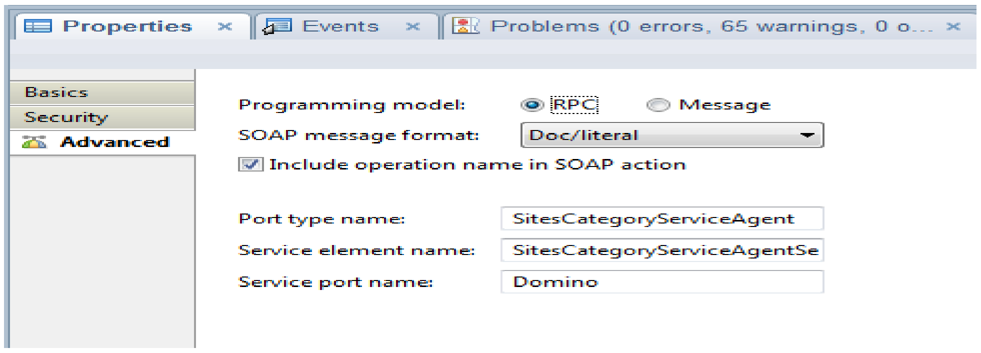
Properties | Events | Problems (0 errors, 65 warnings, 0 o...

Basics
Security
Advanced

☐ Run as Web user
Run on behalf of:

Runtime security level
○ 1. Do not allow restricted operations
● 2. Allow restricted operations
○ 3. Allow restricted operations with full administration rights
(1 = most secure)

Default access for this web service
☑ All readers and above
☐ OtherDomainServers
☐ Anonymous
☐ AlienHouse/AlienHouse
☐ LocalDomainServers
☐ Allow Public access users to use this web service

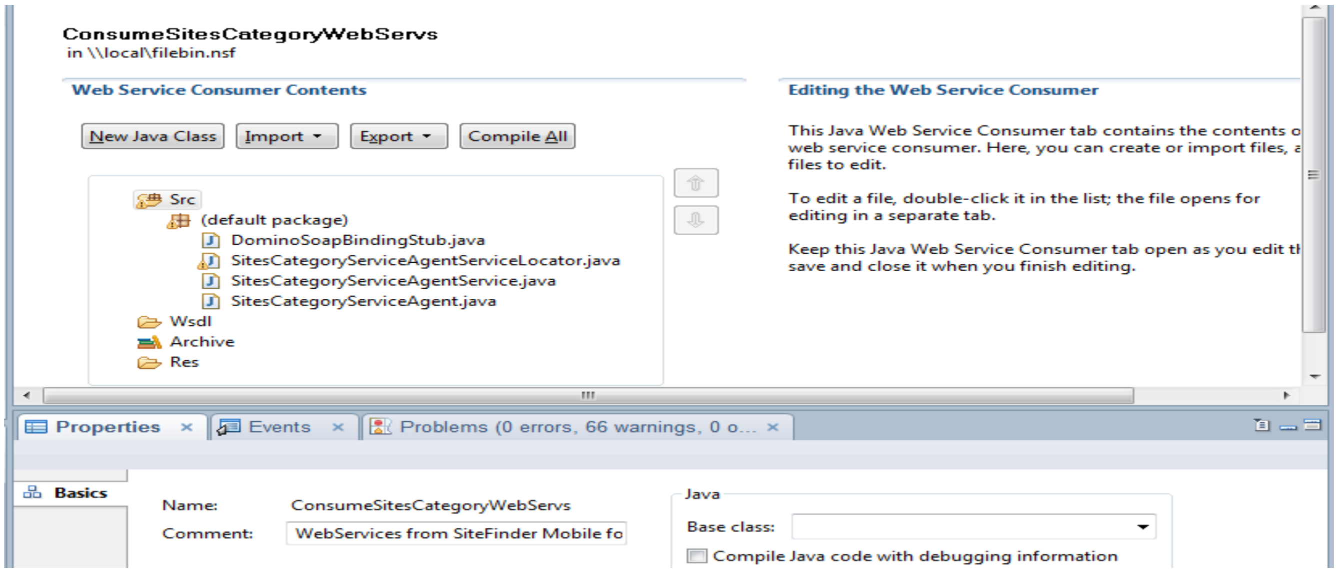


Programming model: [✓ RPC] Message
SOAP message format: Doc/literal
[✓] Include operation name in SOAP action

Port type name: SitesCategoryServiceAgent
Service element name: SitesCategoryServiceAgentSe
Service port name: Domino

Export or Read WebService via URL:
From here you can either export the WebService WSDL to a location to be picked up by other programs or you can simply call the WSDL via URL to create a consuming WebService. For this example we are going to call the WebService provider. We are using a different NSF file to call the WebService provider. Create a new WebService consumer in Java, select the second option 'URL that points to a WSDL file'. You will type the full URL to grab the WebService provider:
http://localhost/sitefindermobile.nsf/SiteCategoriesWebServs?WSDL

You should get a message stating WebService is being created, at which point you should see the following code


ConsumeSitesCategoryWebServs
in \\local\filebin.nsf
Web Service Consumer Contents
New Java Class
Import
Export
Compile All
Src
(default package)
DominoSoapBindingStub.java
SitesCategoryServiceAgentServiceLocator.java
SitesCategoryServiceAgentService.java
SitesCategoryServiceAgent.java
Wsdl
Archive
Res
Editing the Web Service Consumer
Properties
Events
Problems (0 errors, 66 warnings, 0 o...
Basics
Java

This Java Web Service Consumer tab contains the contents o web service consumer. Here, you can create or import files, a files to edit.

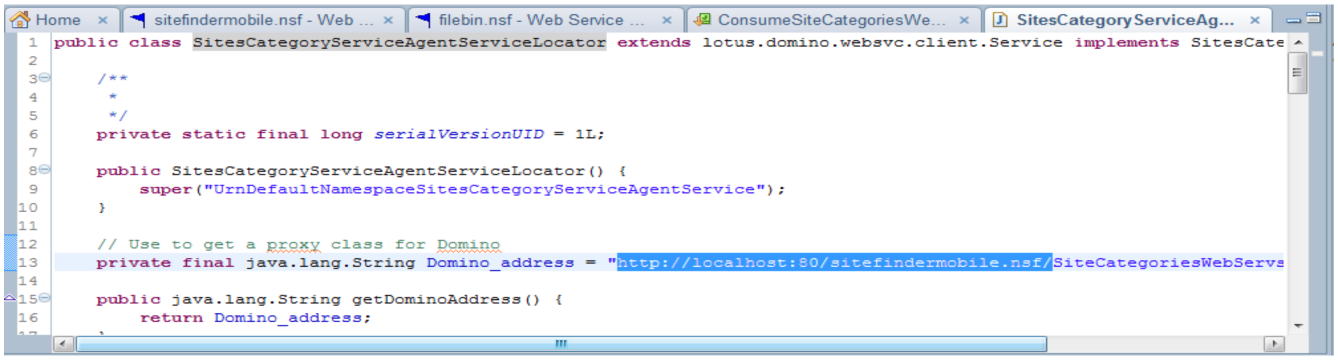
To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Web Service Consumer tab open as you edit tl save and close it when you finish editing.

Name: ConsumeSitesCategoryWebServs
Comment: WebServices from SiteFinder Mobile fo
Base class:
[ ] Compile Java code with debugging information

Examine WebService Consumer Code:
You will need to check the WebService service locator Java class, one of the files generated above, to be certain the URL points to sitefindermobile.nsf WSDL file:

http://localhost:80/sitefindermobile.nsf/SiteCategoriesWebServs?OpenWebService

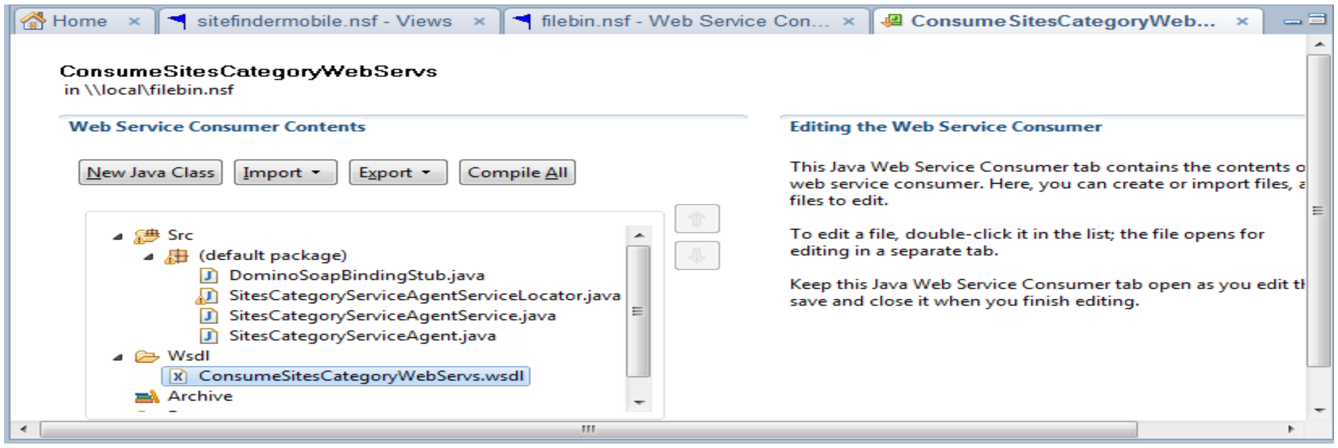

Home
sitefindermobile.nsf - Web ...
filebin.nsf - Web Service ...
ConsumeSiteCategoriesWe...
SitesCategoryServiceAg...

```
public class SitesCategoryServiceAgentServiceLocator extends lotus.domino.websvc.client.Service implements SitesCate

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    public SitesCategoryServiceAgentServiceLocator() {
        super("UrnDefaultNamespaceSitesCategoryServiceAgentService");
    }

    // Use to get a proxy class for Domino
    private final java.lang.String Domino_address = "http://localhost:80/sitefindermobile.nsf/SiteCategoriesWebServs

    public java.lang.String getDominoAddress() {
        return Domino_address;
```

Locate generated snippet code to consume WebService:
You will be using a JavaAgent to invoke the WebService consumer and grab documents from your view. To do this, open the Java class named 'SitesCategoryServiceAgentService':

```
/*
 * To instantiate a callable instance for invoking this service, copy and paste
from the following:

    import *;

    // Refer to SitesCategoryServiceAgent.java for service interface methods:
    SitesCategoryServiceAgent stub = new
SitesCategoryServiceAgentServiceLocator().getDomino();
*/
```

Double-check generated consumer WebService WSDL:
Here again, the WebService consumer WSDL may need to be updated to ensure the URL that points to the provider is proper.


Home
sitefindermobile.nsf - Views
filebin.nsf - Web Service Con...
ConsumeSitesCategoryWeb...
New Java Class
Import
Export
Compile All
Src
(default package)
DominoSoapBindingStub.java
SitesCategoryServiceAgentServiceLocator.java
SitesCategoryServiceAgentService.java
SitesCategoryServiceAgent.java
Wsdl
ConsumeSitesCategoryWebServs.wsdl
Archive

ConsumeSitesCategoryWebServs
in \\local\filebin.nsf

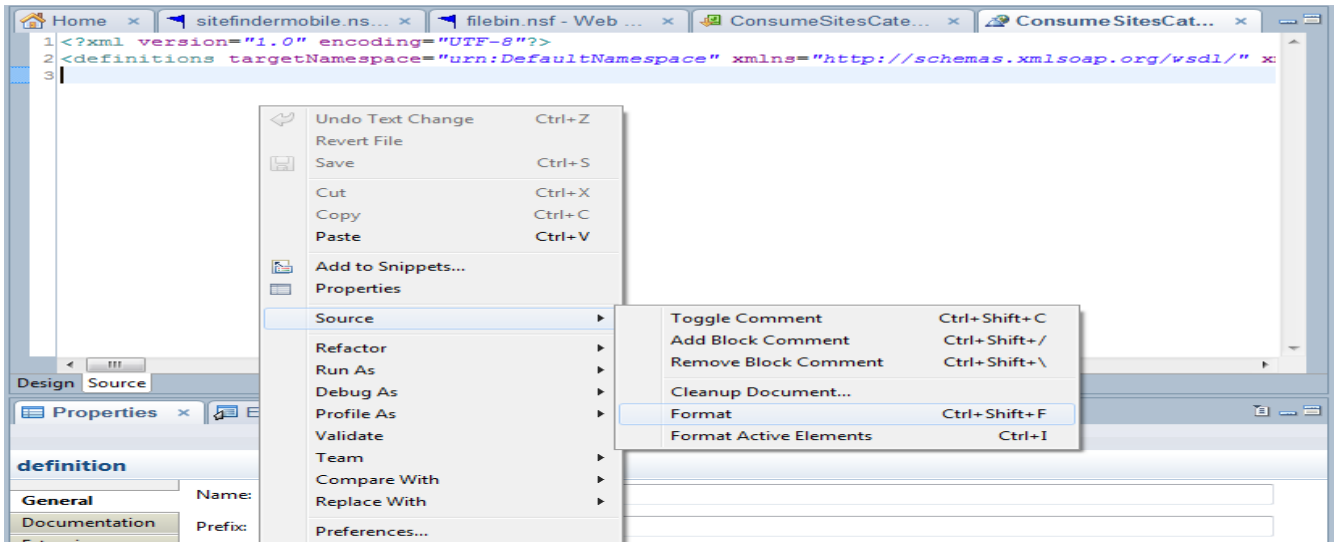
Web Service Consumer Contents

Editing the Web Service Consumer

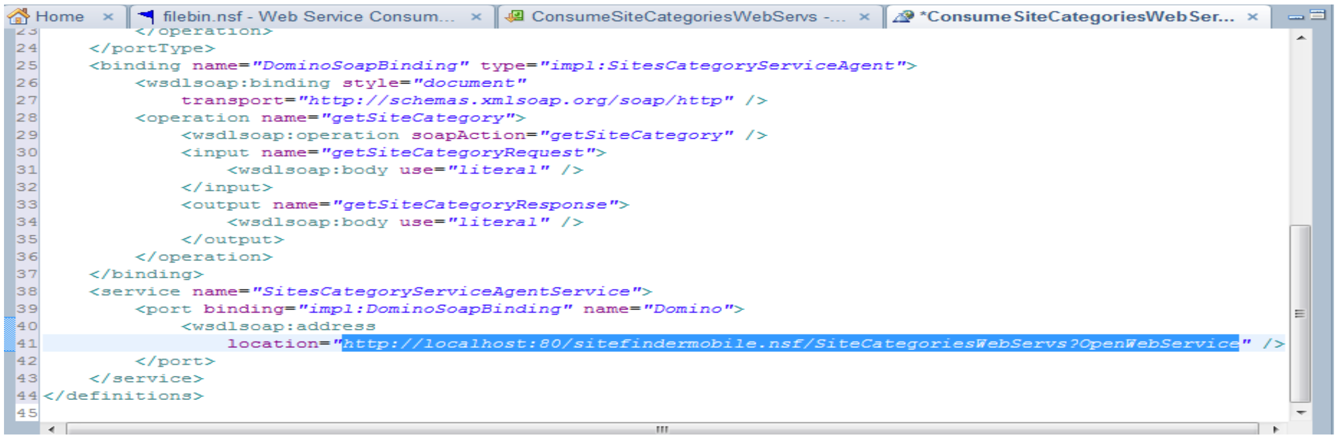
This Java Web Service Consumer tab contains the contents o web service consumer. Here, you can create or import files, a files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Web Service Consumer tab open as you edit th save and close it when you finish editing.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions targetNamespace="urn:DefaultNamespace" xmlns="http://schemas.xmlsoap.org/wsdl/" x
```

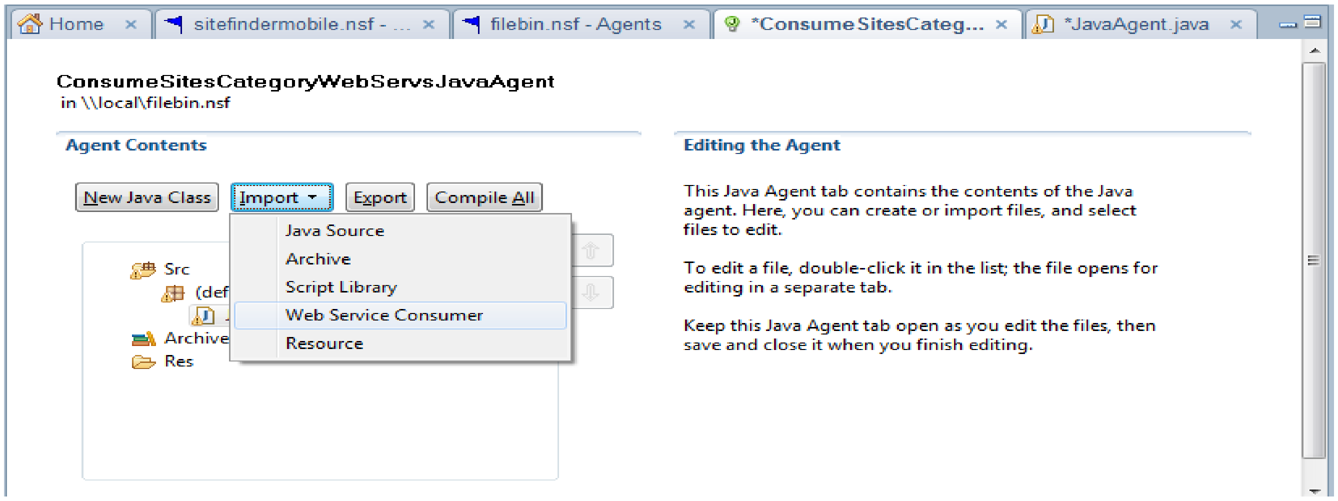

Home
filebin.nsf - Web Service Consum...
ConsumeSiteCategoriesWebServs -...
*ConsumeSiteCategoriesWebSer...

```xml
        </operation>
    </portType>
    <binding name="DominoSoapBinding" type="impl:SitesCategoryServiceAgent">
        <wsdlsoap:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http" />
        <operation name="getSiteCategory">
            <wsdlsoap:operation soapAction="getSiteCategory" />
            <input name="getSiteCategoryRequest">
                <wsdlsoap:body use="literal" />
            </input>
            <output name="getSiteCategoryResponse">
                <wsdlsoap:body use="literal" />
            </output>
        </operation>
    </binding>
    <service name="SitesCategoryServiceAgentService">
        <port binding="impl:DominoSoapBinding" name="Domino">
            <wsdlsoap:address
                location="http://localhost:80/sitefindermobile.nsf/SiteCategoriesWebServs?OpenWebService" />
        </port>
    </service>
</definitions>
```

Copy and Paste JavaAgent to invoke WebService:
Create a new JavaAgent, copy and paste the following over the generated code. You will be using this JavaAgent to invoke the WebService consumer and grab documents from your view. You will also add the snippet you checked earlier from the SitesCategoryServiceAgentService.java class.

```java
/**
 * Created: 2013.09.23.5.54.AM
 * ConsumeSitesCategoryWebServsJavaAgent | JavaAgent.java
 * WebServices data for Xpages
 */
import lotus.domino.*;

/**
 * @author Dököll Solutions, Inc.
 * @version 2013.09.23.5.54.AM
 *
 */
public class JavaAgent extends AgentBase {
    public void NotesMain() {
        try {
            //Locate the WebService code to perform the search, aided by
sitefindermobile.nsf WSDL
        //Note: below stub may be different than your version if you named your
WebService Agent differently
        //grab below stub from the YourWebServiceAgentNameService (for this
exercise ours is named 'SitesCategoryServiceAgentService'
        //code that can be found among the four Java classes generated when
creating the WebService consumer
        SitesCategoryServiceAgent stub = new
SitesCategoryServiceAgentServiceLocator().getDomino();
          //do something with the result of your search
        //example show the value of getSiteCategory method in an Xpages form
        System.out.println(stub.getSiteCategory("Dököll Solutions, Inc."));
      } catch(Exception e) {
          e.printStackTrace();
       }
   }
}
```

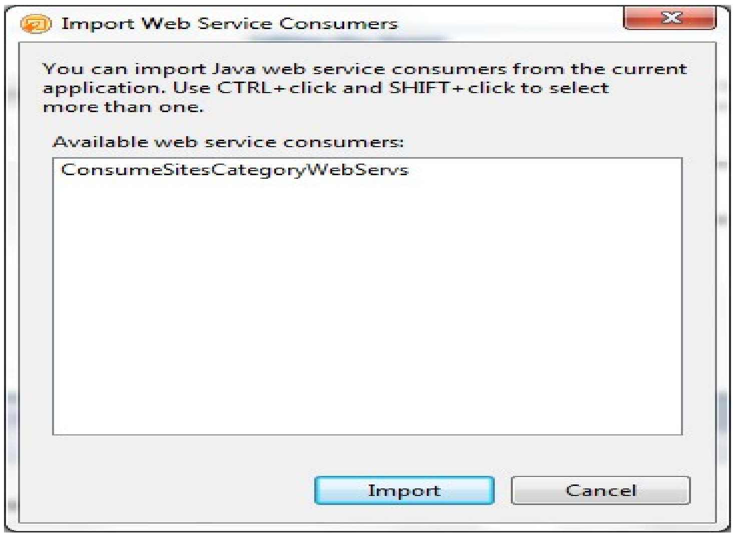
Pull in the WebService Consumer Code:
Your JavaAgent has been created, but wait! What's the deal with the error... Well, the Agent cannot locate the WebService consumer. Pull it in by importing all code as a WebService Consumer.



This Java Agent tab contains the contents of the Java agent. Here, you can create or import files, and select files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Agent tab open as you edit the files, then save and close it when you finish editing.

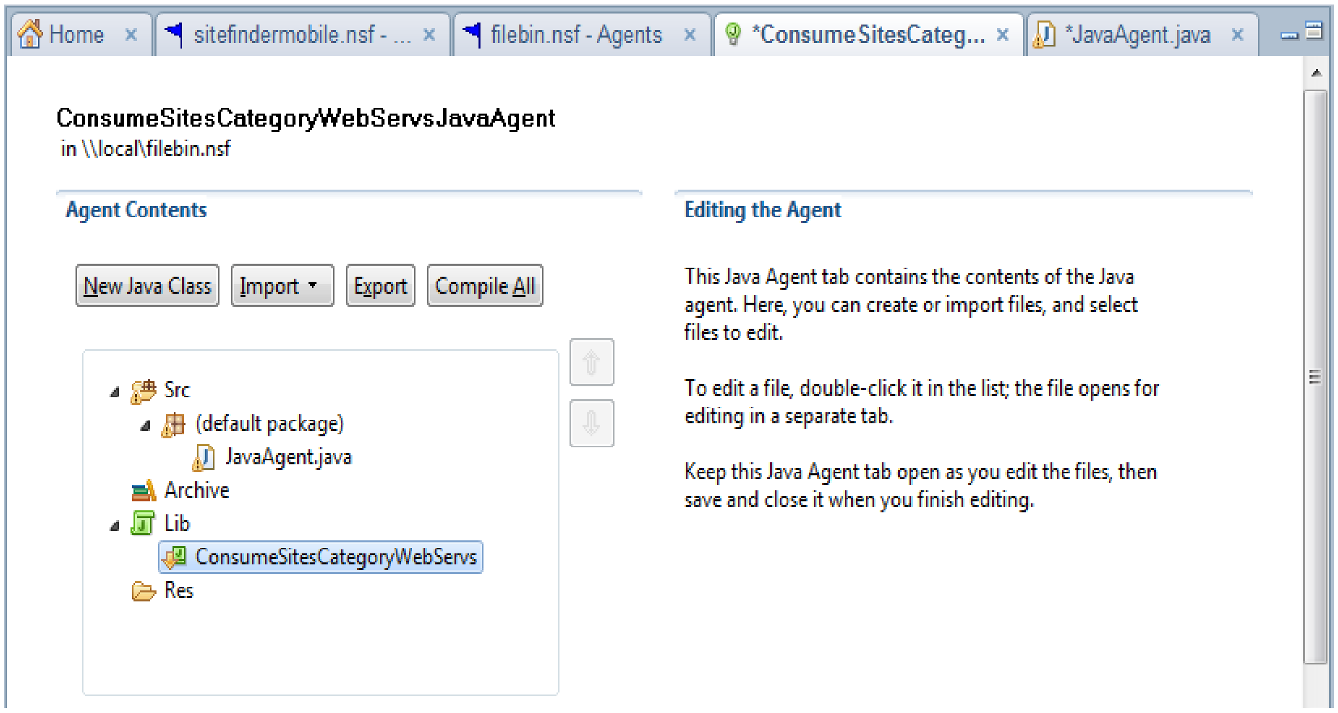

Import Web Service Consumers

You can import Java web service consumers from the current application. Use CTRL+click and SHIFT+click to select more than one.

Available web service consumers:

ConsumeSitesCategoryWebServs

Run JavaAgent /run WebService Consumer:
Be sure the security settings are dynamite, load the NSF file being used to consume the WebService to the Client to run the JavaAgent from menu. To do this, you must also load the App to your browser and make available 80 port, this allows your JavaAgent to see the WebService.



ConsumeSitesCategoryWebServsJavaAgent
in \\local\filebin.nsf

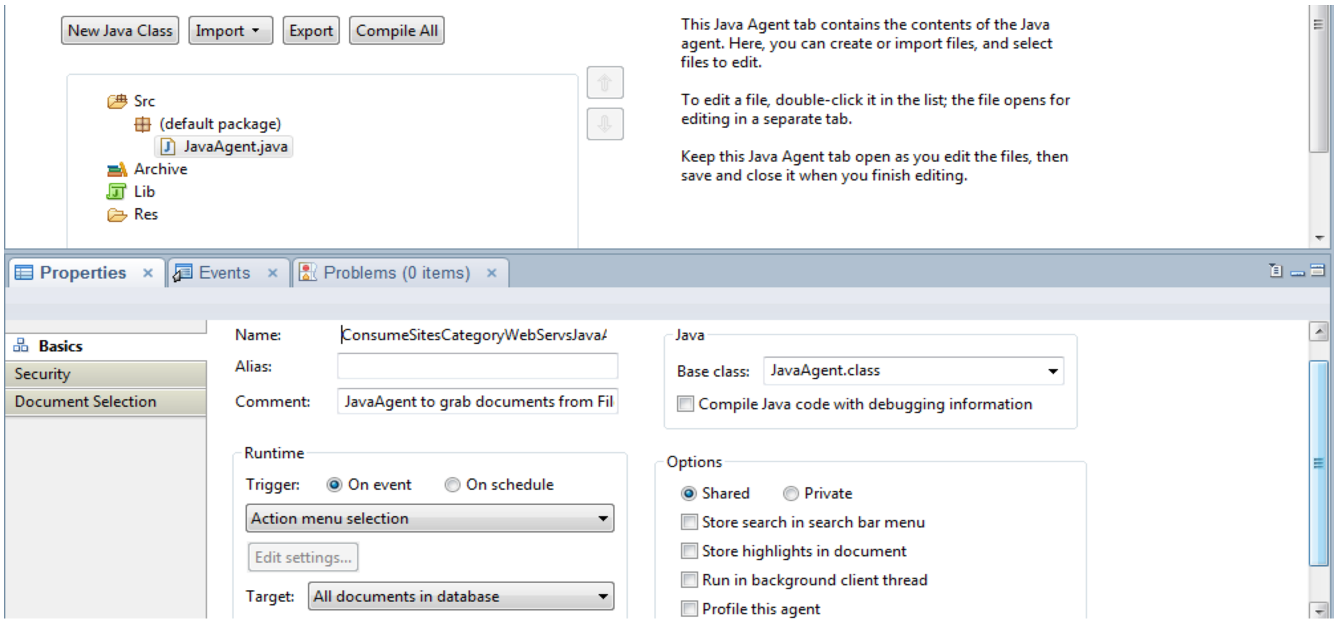
Agent Contents

Editing the Agent

This Java Agent tab contains the contents of the Java agent. Here, you can create or import files, and select files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Agent tab open as you edit the files, then save and close it when you finish editing.



This Java Agent tab contains the contents of the Java agent. Here, you can create or import files, and select files to edit.

To edit a file, double-click it in the list; the file opens for editing in a separate tab.

Keep this Java Agent tab open as you edit the files, then save and close it when you finish editing.

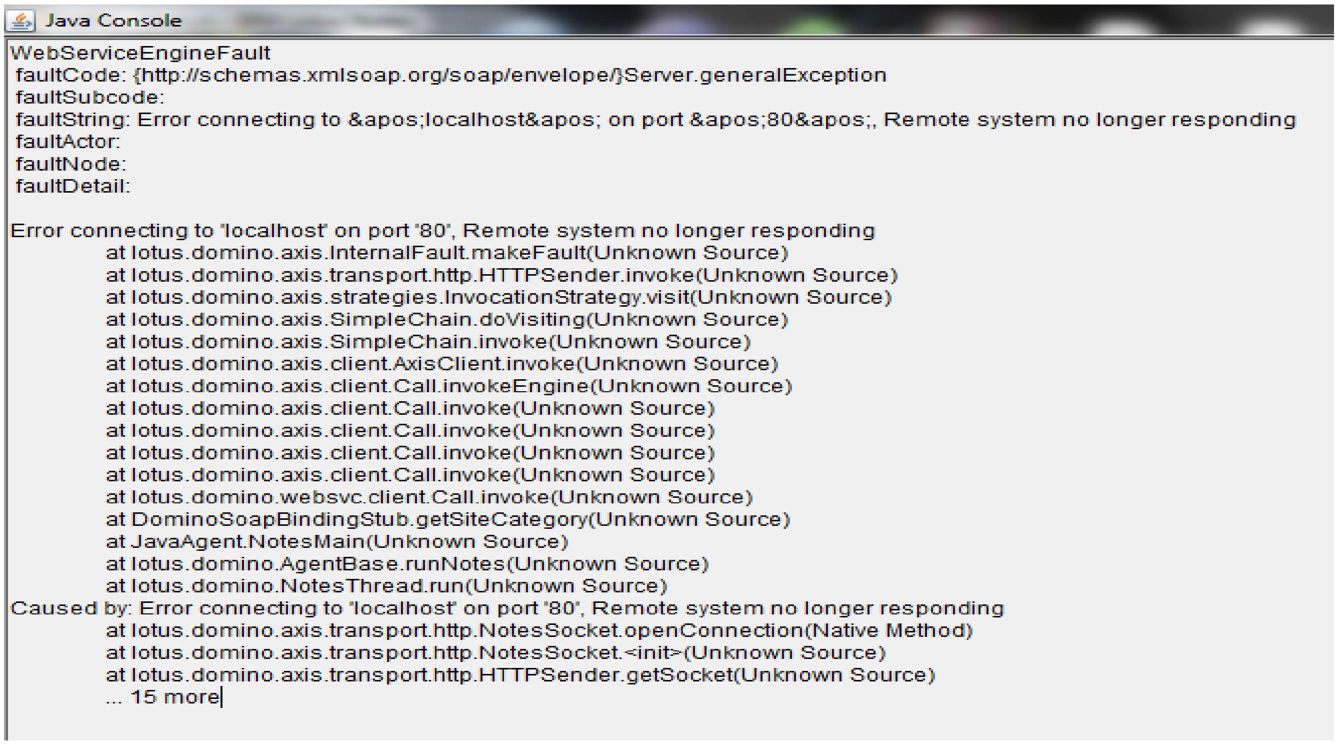
Name: ConsumeSitesCategoryWebServsJava/
Alias:
Comment: JavaAgent to grab documents from Fil

Runtime
Trigger: [✓ On event] On schedule
Action menu selection
Edit settings...
Target: All documents in database

Java
Base class: JavaAgent.class
☐ Compile Java code with debugging information

Options
[✓ Shared] Private
☐ Store search in search bar menu
☐ Store highlights in document
☐ Run in background client thread
☐ Profile this agent

Possible WebService Consumer Error:
What? More errors! Why yes, the JavaAgent needs to run the WebService consumer to grab view documents but must be able to find the provider file, which means that the 'sitefindermobile.nsf' server needs to be running, in this case http://localhost/sitefindermobile.nsf/ needs to be open for business.



```
WebServiceEngineFault
 faultCode: {http://schemas.xmlsoap.org/soap/envelope/}Server.generalException
 faultSubcode:
 faultString: Error connecting to &apos;localhost&apos; on port &apos;80&apos;, Remote system no longer responding
 faultActor:
 faultNode:
 faultDetail:

Error connecting to 'localhost' on port '80', Remote system no longer responding
	at lotus.domino.axis.InternalFault.makeFault(Unknown Source)
	at lotus.domino.axis.transport.http.HTTPSender.invoke(Unknown Source)
	at lotus.domino.axis.strategies.InvocationStrategy.visit(Unknown Source)
	at lotus.domino.axis.SimpleChain.doVisiting(Unknown Source)
	at lotus.domino.axis.SimpleChain.invoke(Unknown Source)
	at lotus.domino.axis.client.AxisClient.invoke(Unknown Source)
	at lotus.domino.axis.client.Call.invokeEngine(Unknown Source)
	at lotus.domino.axis.client.Call.invoke(Unknown Source)
	at lotus.domino.axis.client.Call.invoke(Unknown Source)
	at lotus.domino.axis.client.Call.invoke(Unknown Source)
	at lotus.domino.axis.client.Call.invoke(Unknown Source)
	at lotus.domino.websvc.client.Call.invoke(Unknown Source)
	at DominoSoapBindingStub.getSiteCategory(Unknown Source)
	at JavaAgent.NotesMain(Unknown Source)
	at lotus.domino.AgentBase.runNotes(Unknown Source)
	at lotus.domino.NotesThread.run(Unknown Source)
Caused by: Error connecting to 'localhost' on port '80', Remote system no longer responding
	at lotus.domino.axis.transport.http.NotesSocket.openConnection(Native Method)
	at lotus.domino.axis.transport.http.NotesSocket.<init>(Unknown Source)
	at lotus.domino.axis.transport.http.HTTPSender.getSocket(Unknown Source)
	... 15 more
```

Conclusion:
Provided all is well, and your site (sitefindermobile.nsf) is running in the browser, clear the above

errors from the Java console, then try running the JavaAgent in filebin.nsf again, you should now see a record in the console from the sitefindermobile.nsf view.

References:

http://blog.redturtle.it/2010/01/18/lotusphere-day-1-jmp105

Questions, comments, please post a brief message on our Contact form on the main site.
Follow Us: @DokollSolutions | https://twitter.com/DokollSolutions
or Like Us: Dököll Solutions, Inc. | https://www.facebook.com

Thank you for coming...