

VI-32 Vision Xtend Web Services Validation

Alexander Ng

Principal Consultant

Deltek, Inc.

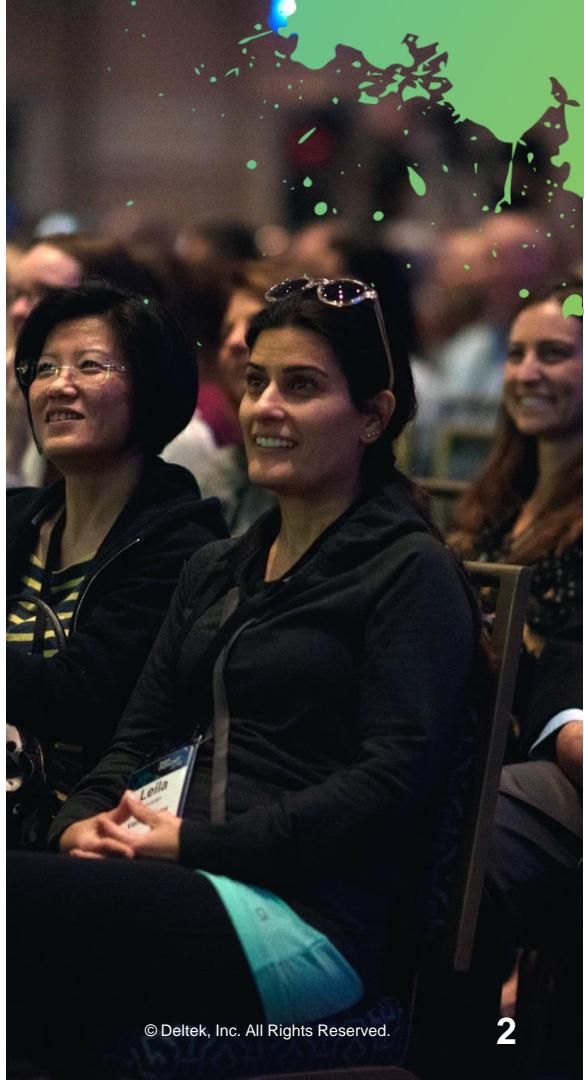
A photograph of the Nashville skyline at dusk or night. The city is reflected in the Cumberland River in the foreground. A large bridge spans the river on the left. The buildings, including the iconic AT&T Building, are illuminated against a dark sky.

Deltek INSIGHT > 2017

October 23–26 » Nashville, TN

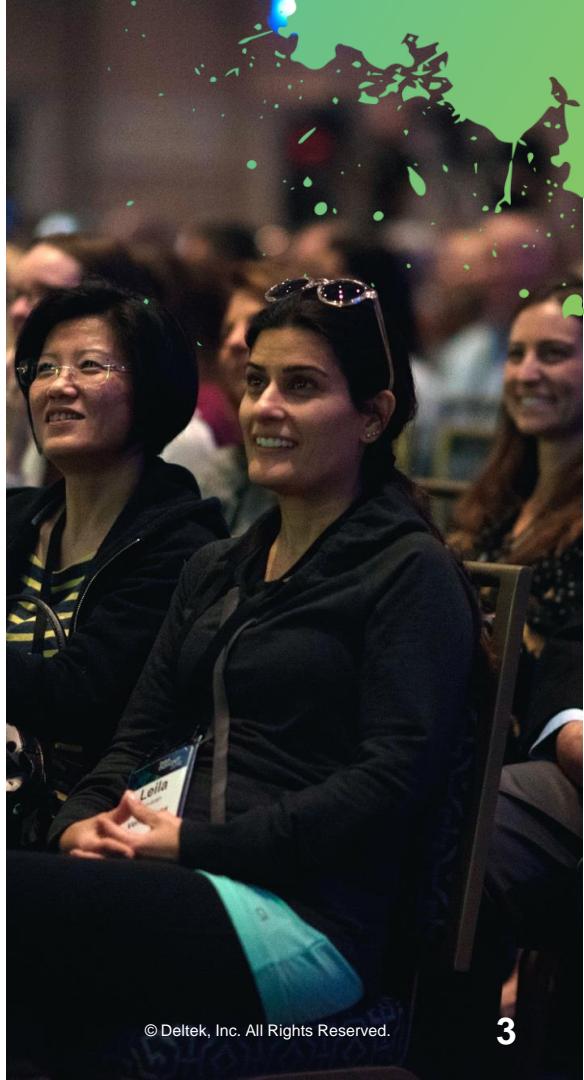
Agenda

1. What is a Web Service?
2. What kind of Vision Xtend Web Services does Vision support?
3. Why Have A Vision Xtend Web Service?
4. Building Your First Web Service
5. Configuring The Web Service
6. How To Make Vision Call The Web Service?



Agenda

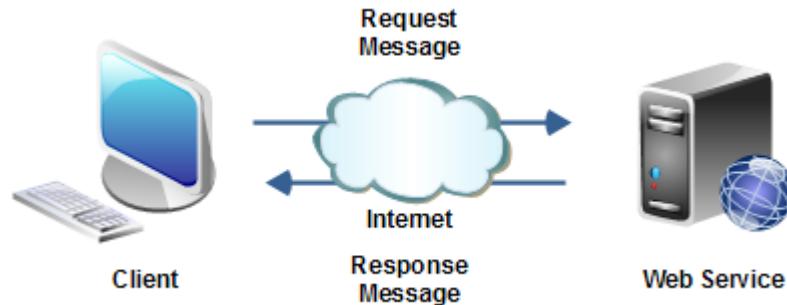
7. What Is Passed Into The Web Service?
8. How To Parse The XML?
9. What Should The Web Service Return?
10. Tips and Tricks
11. Questions & Contact



What is a Web Service?

What is a Web Service?

- Application programming interface (API) which can be accessed over a network and executed on a remote system hosting the requested services.
- Prominent web services are Google Maps API, Weather Channel API, FaceBook API, Twitter API, etc.



What is a Web Service?

- In the context of Vision, web services are used to extend the business logic of the Vision application without recoding Vision's source files.

What Vision applications can be Xtended via a Web Service?

What Vision applications can be Xtended via a Web Service?

- Vision supports web services extension for Expense Reports, Item Requests, Project Plans, Purchase Orders, Purchase Requisitions, Request for Price Quotes, Timesheets, Transaction Entry, and Transaction Posting.

What Vision applications can be Xtended via a Web Service?

- Vision's Web Services Module is located in 'Configuration\Workflow\Web Services'.

The screenshot shows the Deltek Vision software interface. The top navigation bar includes 'Hide Navigation', 'Back', 'Forward', 'Dashboard', 'iAccess', 'Kona', 'Search', and 'Options'. The left sidebar, titled 'Navigation', contains several categories: Reporting, Utilities, Configuration (which is selected and highlighted in blue), Workflow (selected and highlighted in red), Web Services (selected and highlighted in red), Security, Organization, Planning, Billing, Accounting, Payroll, and General. The main content area is titled 'Web Services' and displays a table with columns for Application, Description, and Web Service. The table lists nine items: Expense Reports, Item Requests, Project Plans, Purchase Orders, Purchase Requisitions, RequestForPrice Quotes, Timesheets, Transaction Entry, and Transaction Posting. The 'Description' column for the first item, 'Expense Reports', is currently selected.

Application	Description	Web Service
	1	
Expense Reports		
Item Requests		
Project Plans		
Purchase Orders		
Purchase Requisitions		
RequestForPrice Quotes		
Timesheets		
Transaction Entry		
Transaction Posting		

Why Have A Vision Xtend Web Service?

Why Have A Vision Xtend Web Service?

- The main purpose of a Vision Xtend Web Service is to establish a set of custom validation rules before the application being extended is allowed to be saved.

Example Timesheet Rules:

- Employee must work 8 regular hours before entering overtime.
- An Employee cannot charge overtime until 40 regular hours have been worked.
- An Employee cannot charge time to both a Vacation project and a Regular project on the same day.

Why Have A Vision Xtend Web Service?

- Messages are returned as errors or warnings, and those messages are displayed in Vision dialog box.
- All of the custom data validation rules can be encapsulated in one single web service.
- You can do some data update whether within Vision or outside of Vision when the application being extended is submitted. For example, maybe you want to update some outside database you have to track a timesheet audit when the timesheet is saved.

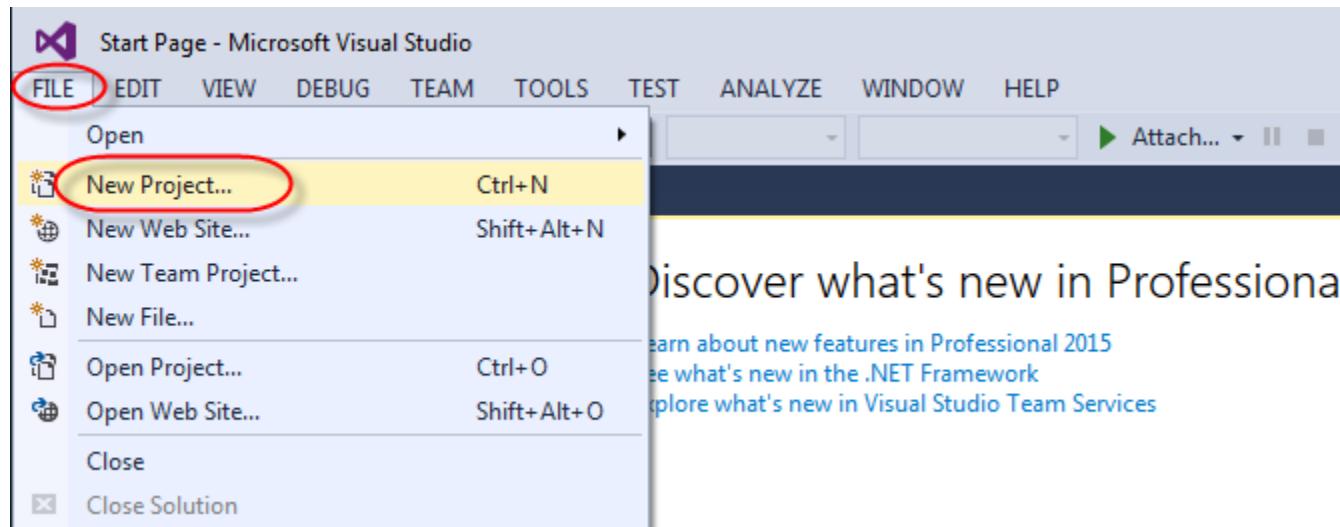
Why Have A Vision Xtend Web Service?

- You can send some emails out when the application being extended is submitted. For example, maybe you want to email the Requisitioner's supervisor when a Purchase Requisition is submitted.
- The web service can be used to launch other events and services that are outside and independent of Vision. For example, it could be possible to call another web service and pass the Vision Timesheet data to it.

Building Your First Web Service

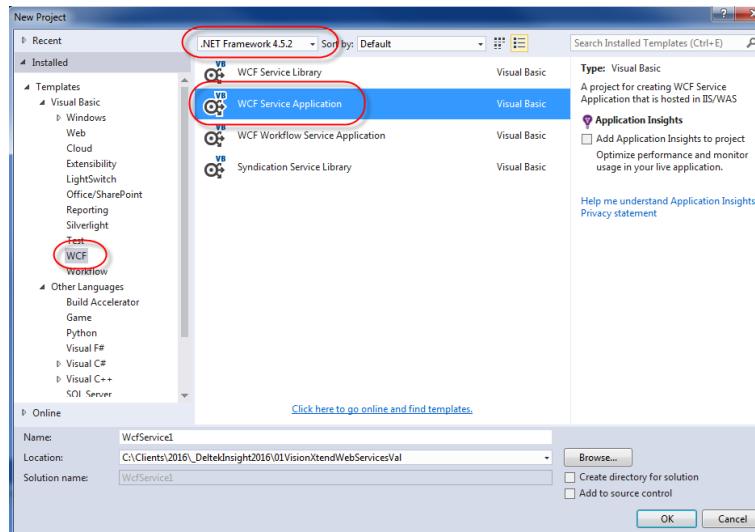
Building Your First Web Service

- Using Visual Studio 2015 click on ‘File / New Project...’



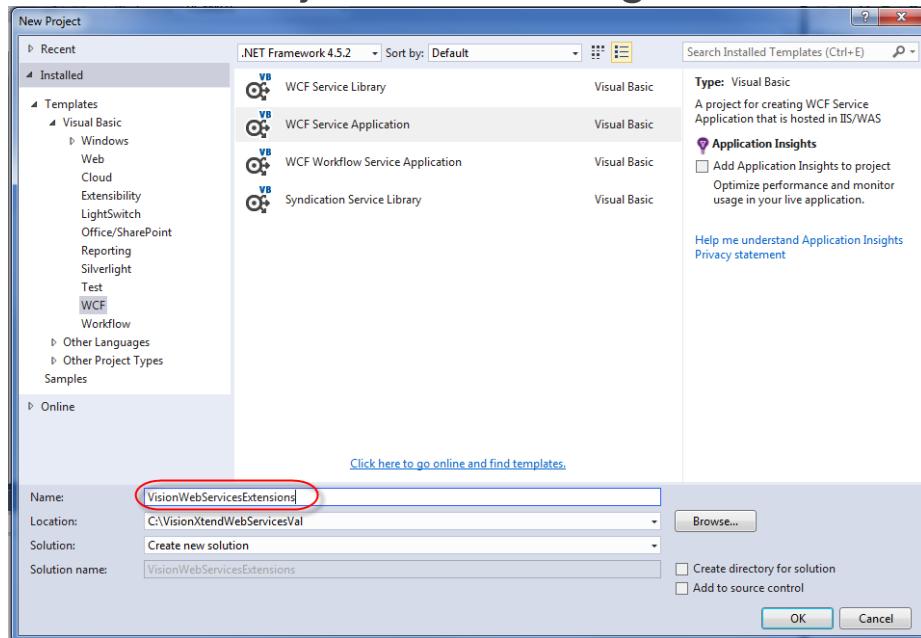
Building Your First Web Service

- Make sure the targeted .NET Framework version is 4.5.2 (for Vision 7.6)
- Then under your preferred programming language select WCF / WCF Service Application



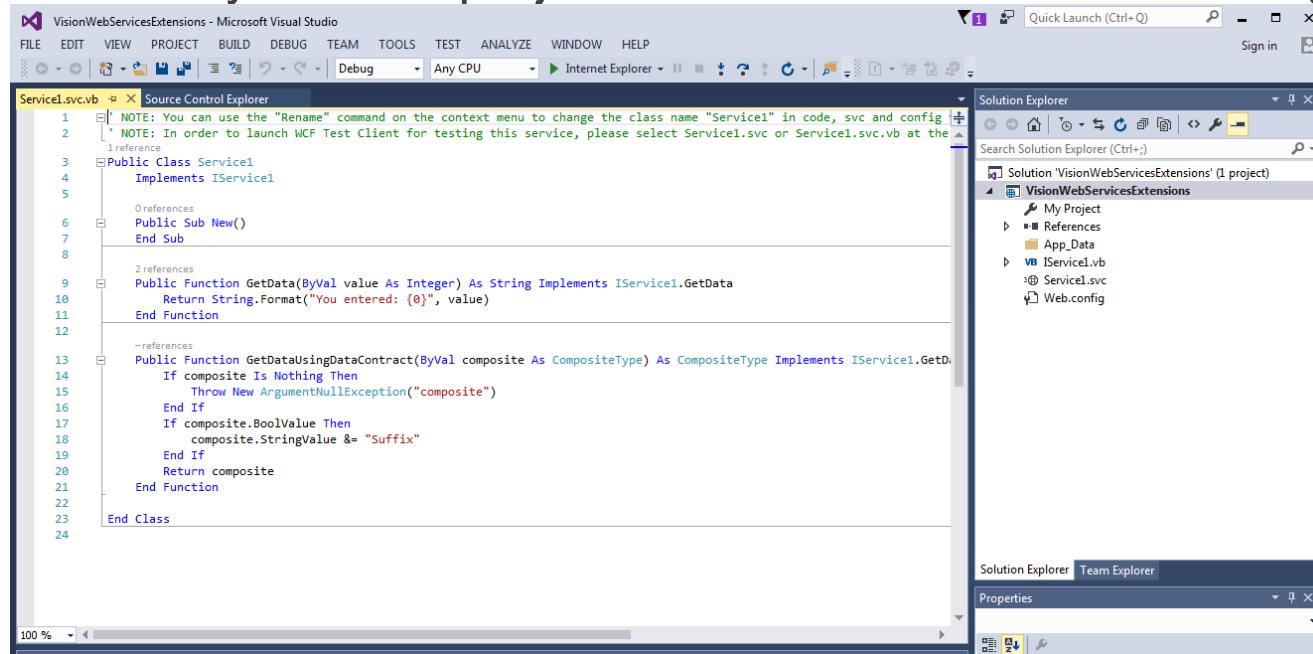
Building Your First Web Service

- Give the Project a meaningful name: VisionWebServicesExtensions



Building Your First Web Service

- The newly created project should resemble the following:

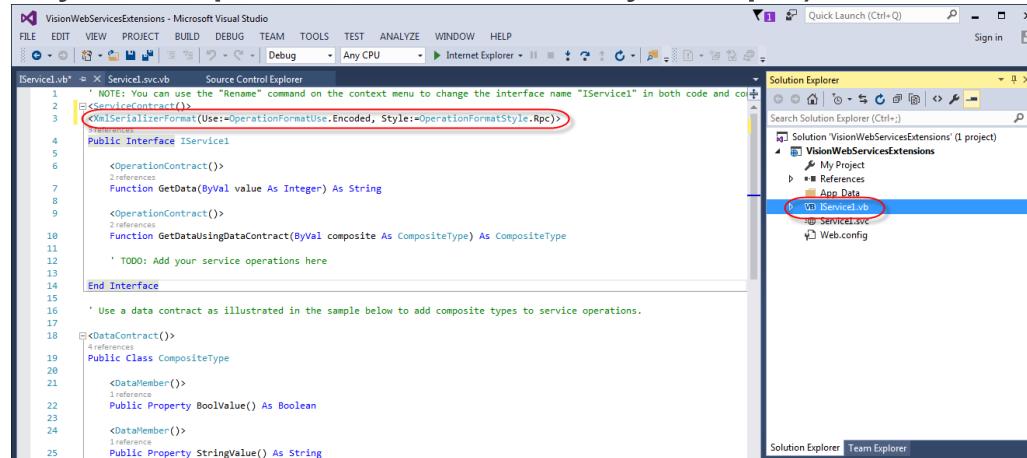


A screenshot of Microsoft Visual Studio showing a WCF service project named "VisionWebServicesExtensions". The main window displays the code editor for "Service1.svc.vb". The code defines a class "Service1" that implements the interface "IService1". It contains two functions: "GetData" and "GetDataUsingDataContract". The "GetData" function returns a string with the entered value. The "GetDataUsingDataContract" function takes a composite type and returns a modified version of it if its boolean value is true. The Solution Explorer on the right shows the project structure with files like "IService1.vb", "Service1.svc", and "Web.config".

```
Service1.svc.vb  Source Control Explorer
1  ' NOTE: You can use the "Rename" command on the context menu to change the class name "Service1" in code, svc and config file simultaneously.
2  ' NOTE: In order to launch WCF Test Client for testing this service, please select Service1.svc or Service1.svc.vb at the top of this page, or press F12 key to switch to WCF Test Client.
3  Public Class Service1
4      Implements IService1
5
6      ' References
7      Public Sub New()
8          ' End Sub
9
10     ' References
11     Public Function GetData(ByVal value As Integer) As String Implements IService1.GetData
12         Return String.Format("You entered: {0}", value)
13     End Function
14
15     ' References
16     Public Function GetDataUsingDataContract(ByVal composite As CompositeType) As CompositeType Implements IService1.GetDataUsingDataContract
17         If composite Is Nothing Then
18             Throw New ArgumentNullException("composite")
19         End If
20         If composite.BoolValue Then
21             composite.StringValue &= "Suffix"
22         End If
23         Return composite
24     End Function
25
26 End Class
```

Building Your First Web Service

- Open IService1.vb and add the following XMLSerializerFormat line to the Public Interface:
`<XmlSerializerFormat(Use:=OperationFormatUse.Encoded,
Style:=OperationFormatStyle.Rpc)>`



The screenshot shows the Microsoft Visual Studio interface. The top menu bar includes FILE, EDIT, VIEW, PROJECT, BUILD, DEBUG, TEAM, TOOLS, TEST, ANALYZE, WINDOW, and HELP. The toolbar below has icons for file operations like New, Open, Save, and Print, along with Debug and Build buttons. The status bar at the bottom shows 'Quick Launch (Ctrl+Q)' and 'Sign in'. The main window contains two tabs: 'IService1.vb' and 'Service1.vb'. The 'IService1.vb' tab displays VB.NET code for a service interface. A specific line of code, '`<XmlSerializerFormat(Use:=OperationFormatUse.Encoded, Style:=OperationFormatStyle.Rpc)>`', is highlighted with a red oval. The 'Service1.vb' tab shows a similar interface definition. The Solution Explorer on the right lists the project 'VisionWebServicesExtensions' (1 project) containing 'My Project', 'References', 'App Data', and 'VB\IService1.vb' (which is also highlighted with a red oval). Other files listed are 'SERVICE1.WCF' and 'Web.config'. The bottom navigation bar includes 'Solution Explorer' and 'Team Explorer'.

```
1  ' NOTE: You can use the "Rename" command on the context menu to change the interface name "IService1" in both code and co-
2  <ServiceContract()
3      <XmlSerializerFormat(Use:=OperationFormatUse.Encoded, Style:=OperationFormatStyle.Rpc)>
4  Public Interface IService1
5
6      <OperationContract()
7          <Description>
8              Function GetData(ByVal value As Integer) As String
9
10     <OperationContract()
11         <Description>
12             Function GetDataUsingDataContract(ByVal composite As CompositeType) As CompositeType
13
14     ' TODO: Add your service operations here
15
16     ' Use a data contract as illustrated in the sample below to add composite types to service operations.
17
18     <DataContract()
19     <DataMember()
20         <Reference>
21             Public Property BoolValue() As Boolean
22
23         <DataMember()
24             <Reference>
25             Public Property StringValue() As String
End Interface
```

Building Your First Web Service

- Comment out (or delete) the sample code headers and enter the header(s) for your own code

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** VisionWebServicesExtensions - Microsoft Visual Studio
- Menu Bar:** FILE, EDIT, VIEW, PROJECT, BUILD, DEBUG, TEAM, TOOLS, TEST, ANALYZE, WINDOW, HELP
- Toolbars:** Standard and Debug
- Code Editor:** The file IServiceProvider.vb is open. The code defines an interface IServiceProvider with three operations: GetData, GetDataUsingDataContract, and timesheetVal. A red box highlights the first two operations. A second red box highlights the implementation of timesheetVal.

```
2 <ServiceContract()
3     <XmlSerializerFormat(Use:=OperationFormatUse.Encoded, Style:=OperationFormatStyle.Rpc)>
4     2 references
5     Public Interface IServiceProvider
6         <OperationContract()
7             'Function GetData(ByVal value As Integer) As String
8
9         <OperationContract()
10            'Function GetDataUsingDataContract(ByVal composite As CompositeType) As CompositeType
11
12            ' TODO: Add your service operations here
13            <OperationContract()
14                'References
15                Function timesheetVal(ByVal inData As String) As String
16
17            End Interface
18
19            ' Use a data contract as illustrated in the sample below to add composite types to service operations.
20            <DataContract()
21                0 references
22                Public Class CompositeType
23
24                    <DataMember()>
```

- Solution Explorer:** Shows the solution 'VisionWebServicesExtensions' with one project, 'My Project', containing files like App_Data, IServiceProvider.vb, Servicel.svc, and Web.config.

Building Your First Web Service

- Open Service1.svc (Service1.svc.vb) and comment out (or delete) the sample code for the sample code headers from IService1.vb

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Solution Explorer:** Shows the project structure for "VisionWebServicesExtensions". It includes a reference folder, an App_Data folder, a VB folder containing "IService1.vb", and a service file "Service1.svc" which is currently selected.
- Code Editor:** The active file is "Service1.svc.vb". The code implements the `IService1` interface. Two specific functions are highlighted with a red box:
 - `Public Function GetData(ByVal value As Integer) As String Implements IService1.GetData`
 - `Public Function GetDataUsingDataContract(ByVal composite As CompositeType) As CompositeType Implements IService1.GetData`

Building Your First Web Service

- Add in your own code and save. Note: for Visual Basic you need to explicitly specify which header in IService1.vb your are implementing.

The screenshot shows the Microsoft Visual Studio interface with the following details:

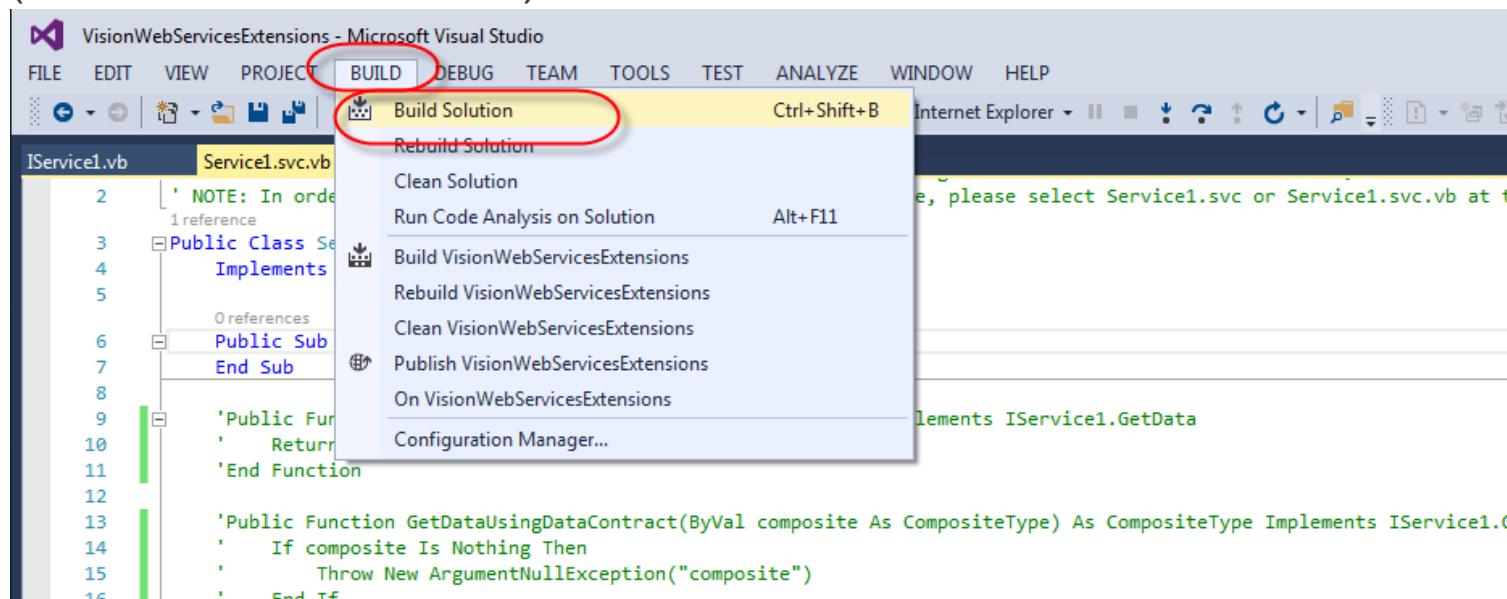
- Title Bar:** VisionWebServicesExtensions - Microsoft Visual Studio
- Menu Bar:** FILE, EDIT, VIEW, PROJECT, BUILD, DEBUG, TEAM, TOOLS, TEST, ANALYZE, WINDOW, HELP
- Toolbars:** Standard, Debug, Task List, Solution Explorer, Properties, Status Bar
- Code Editor:** The file IServiceProvider1.vb is open. A red oval highlights the line 'Implements IService1.timesheetVal' in the code. The code is as follows:

```
2  ' NOTE: In order to launch WCF Test Client for testing this service, please select Servicel.svc or Servicel.svc.vb at the Solution+ References
3  ' 1 reference
4  ' Public Class Servicel
5  '     Implements IService1
6  '
7  '     Public Sub New()
8  '     End Sub
9  '
10 '     Public Function GetData(ByVal value As Integer) As String Implements IService1.GetData
11 '         Return String.Format("You entered: {0}", value)
12 '     End Function
13 '
14 '     Public Function GetDataUsingDataContract(ByVal composite As CompositeType) As CompositeType Implements IService1.GetDataUsingDataContract
15 '         If composite Is Nothing Then
16 '             Throw New ArgumentNullException("composite")
17 '         End If
18 '         If composite.BoolValue Then
19 '             composite.StringValue &= "Suffix"
20 '         End If
21 '         Return composite
22 '     End Function
23 '
24 '     Public Function timesheetVal(ByVal inData As String) As String Implements IService1.timesheetVal
25 '         timesheetVal = "<errors warning=""y""><error>This is my first timesheet error</error></errors>"
```

- Solution Explorer:** Shows the project structure:
 - Solution VisionWebServicesExtensions' (1 project)
 - My Project
 - References
 - App_Data
 - IService1.vb
 - Service1.svc
 - Web.config
- Status Bar:** Solution Explorer, Team Explorer

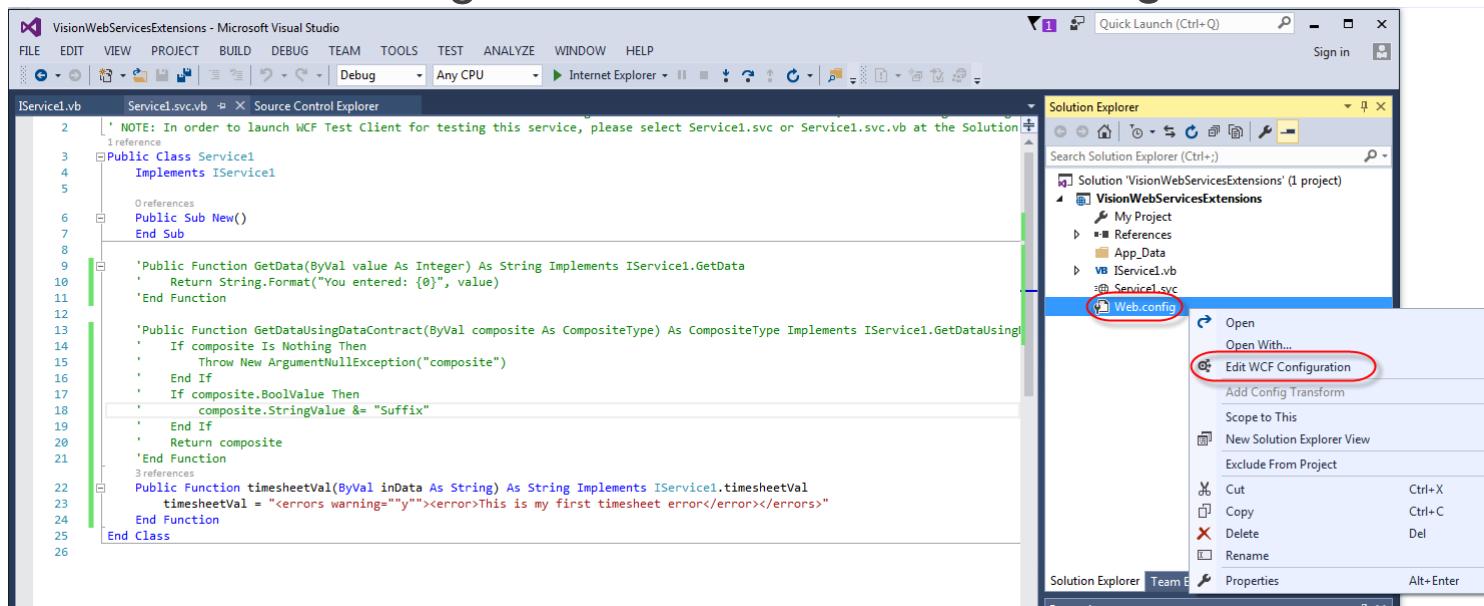
Building Your First Web Service

- Once your code is in place, build your solution to compile the project (BUILD/Build Solution)



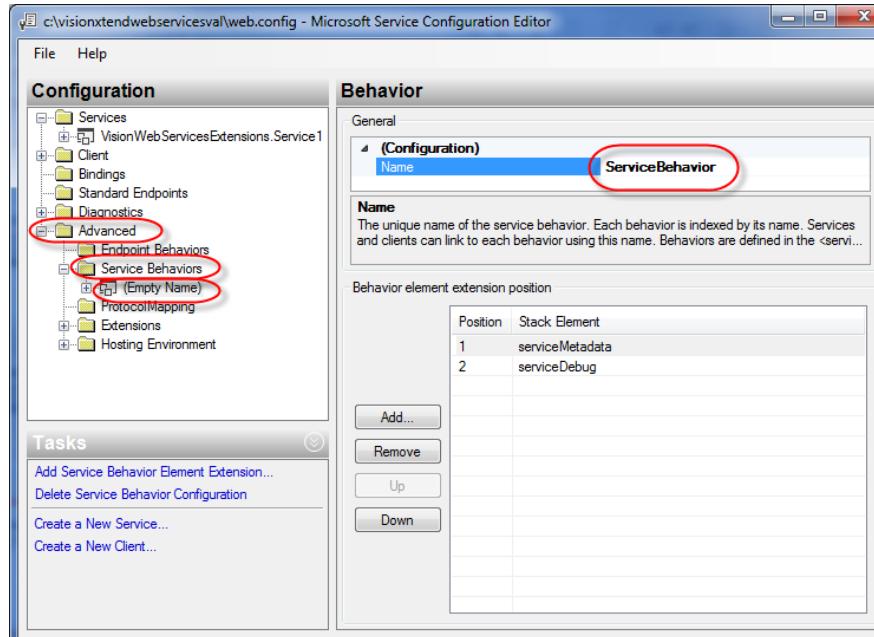
Building Your First Web Service

- Configure the WCF Web Service so that it is Vision compatible. Right click on Web.Config and select “Edit WCF Configuration”



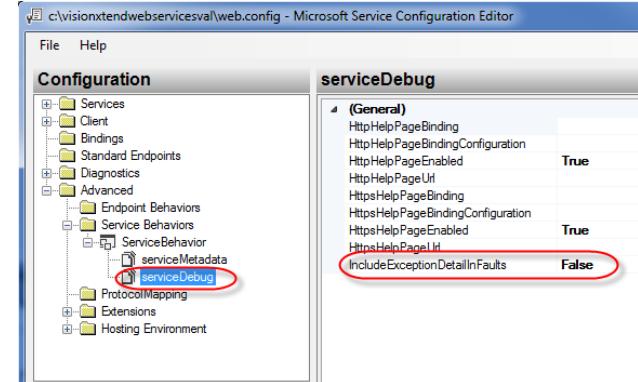
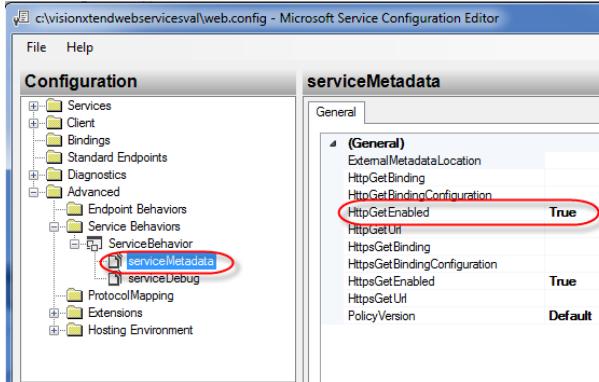
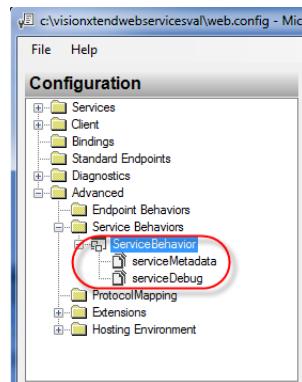
Building Your First Web Service

- Expand/drill down to the “Empty Name” service behavior and give it a name



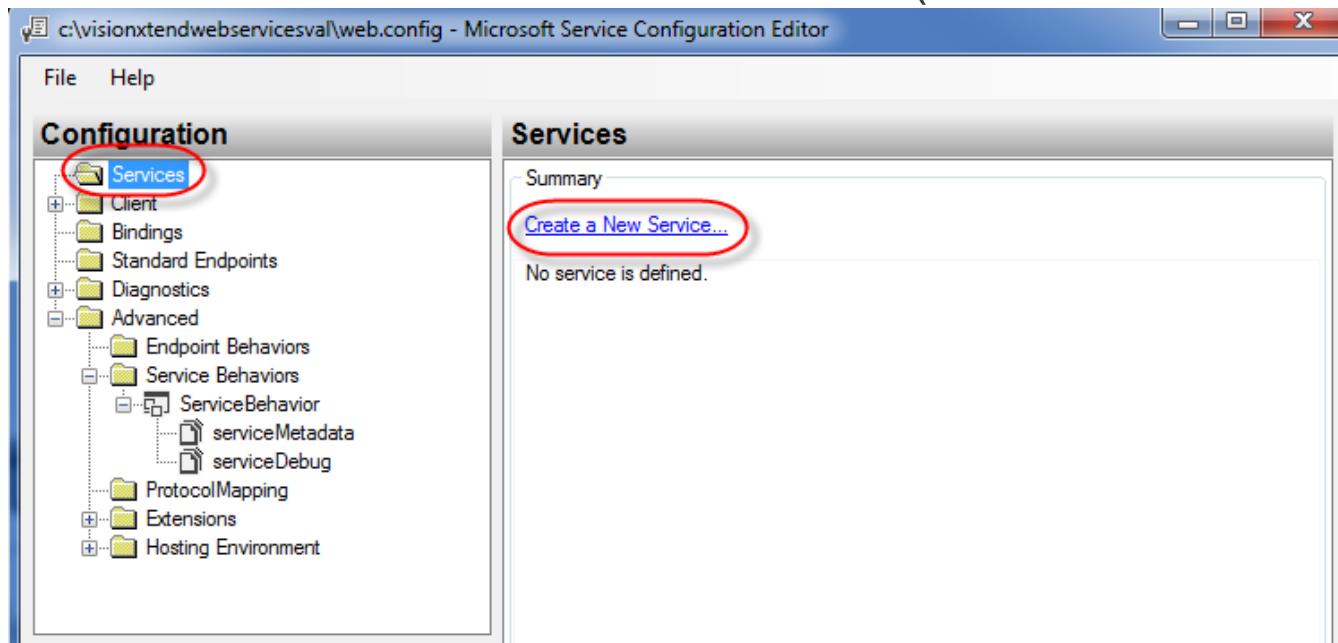
Building Your First Web Service

- Expand ServiceBehavior (or whatever name was given to the Empty Name service behavior) and check that
 - serviceMetadata/HttpGetEnabled = True
 - serviceDebug/includeExceptionDetailInFaults = False



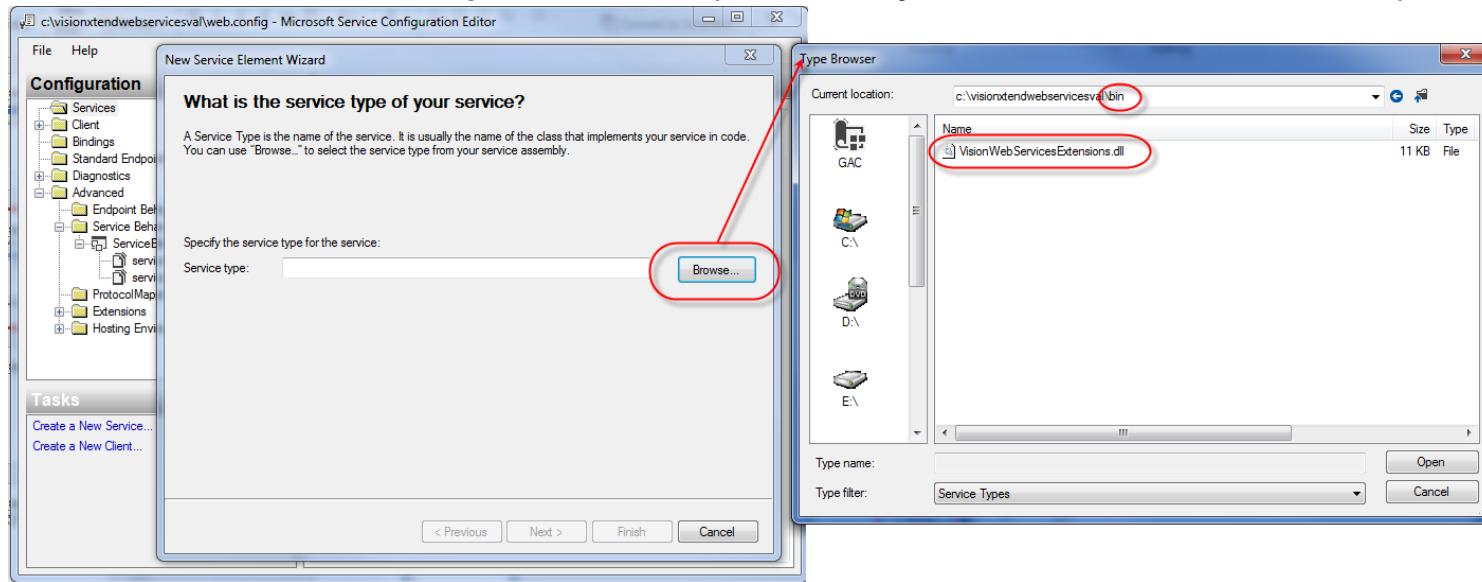
Building Your First Web Service

- Create a New Service under Services (Services/Create a New Service...)



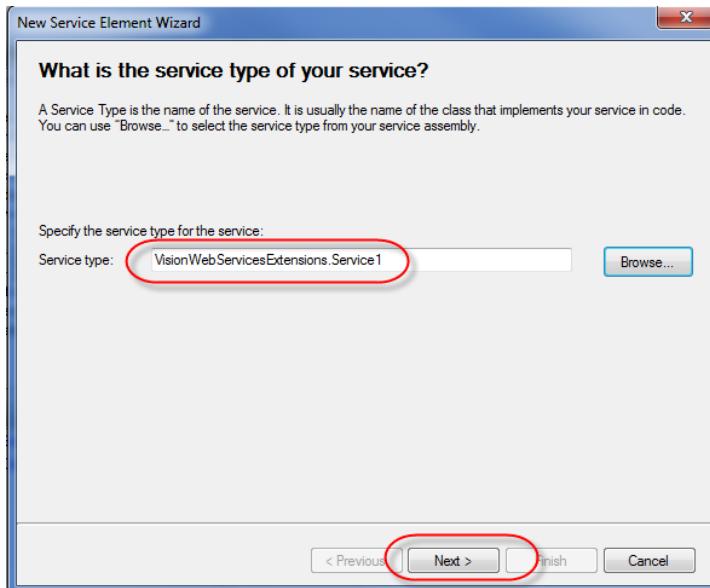
Building Your First Web Service

- Browse for the compiled DLL (normally under the bin folder)



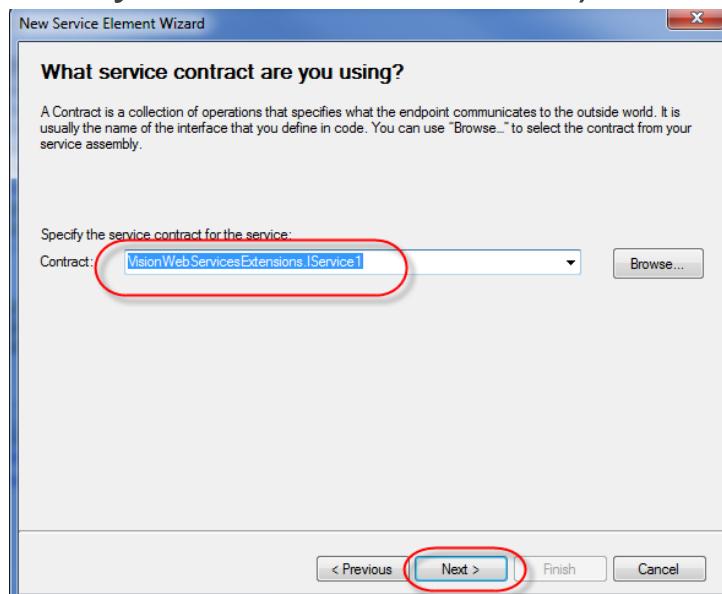
Building Your First Web Service

- Open it down to the class level (.Service1)
- Then click next



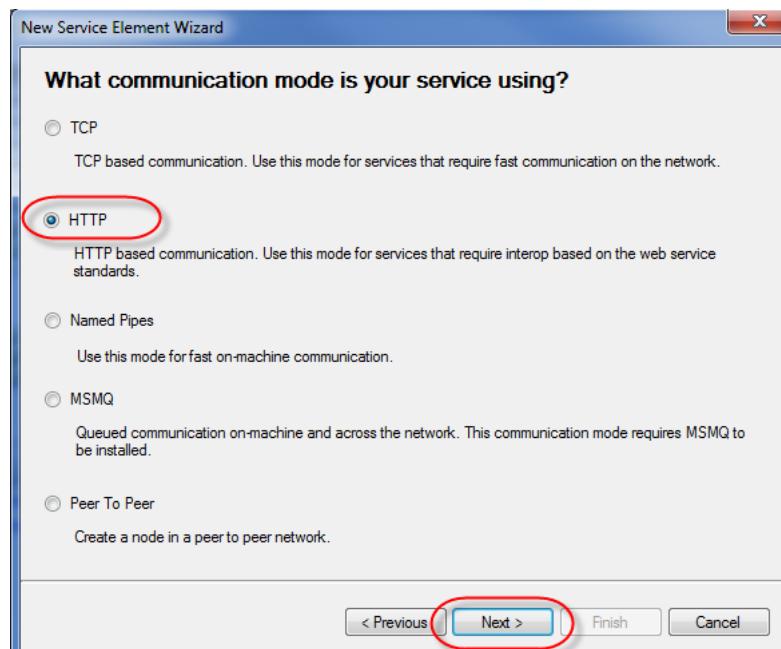
Building Your First Web Service

- Specify the contract for the service (it is usually the name of the interface that you define the code) then click next



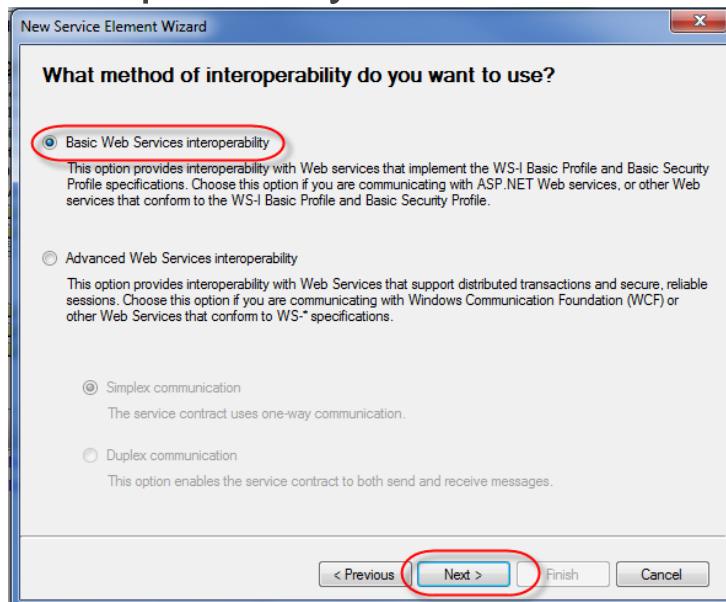
Building Your First Web Service

- Select HTTP as the communication mode and click next



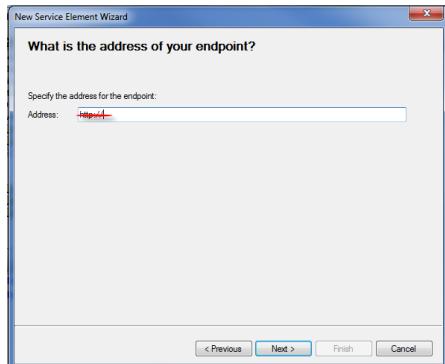
Building Your First Web Service

- Select “Basic Web Services interoperability” as the method of interoperability and click next

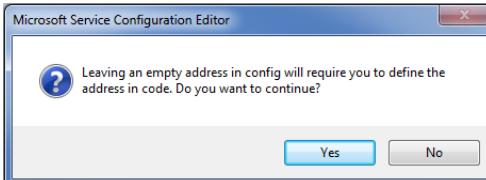


Building Your First Web Service

- Blank out the address for the endpoint and click next

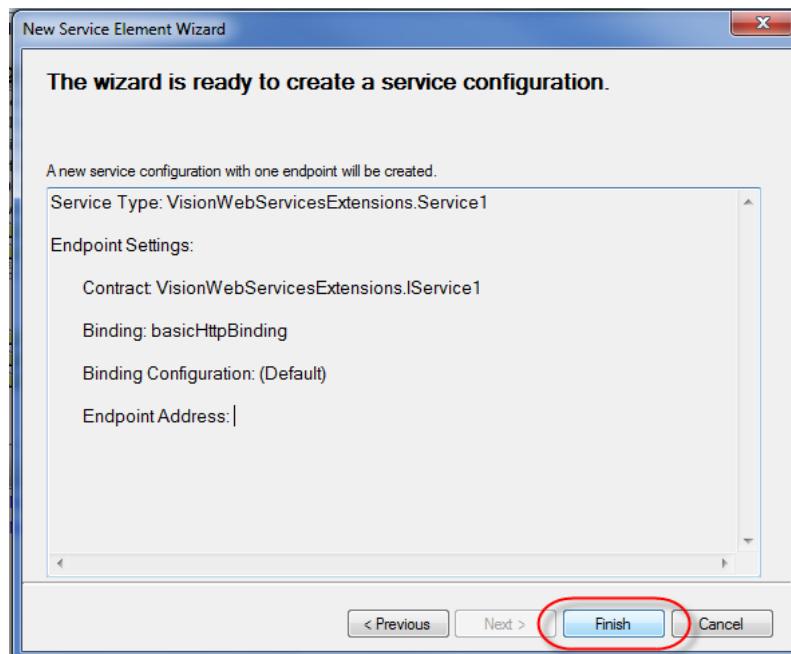


- You will get a warning that an empty address in config will require you to define the address in code. Click Yes to continue



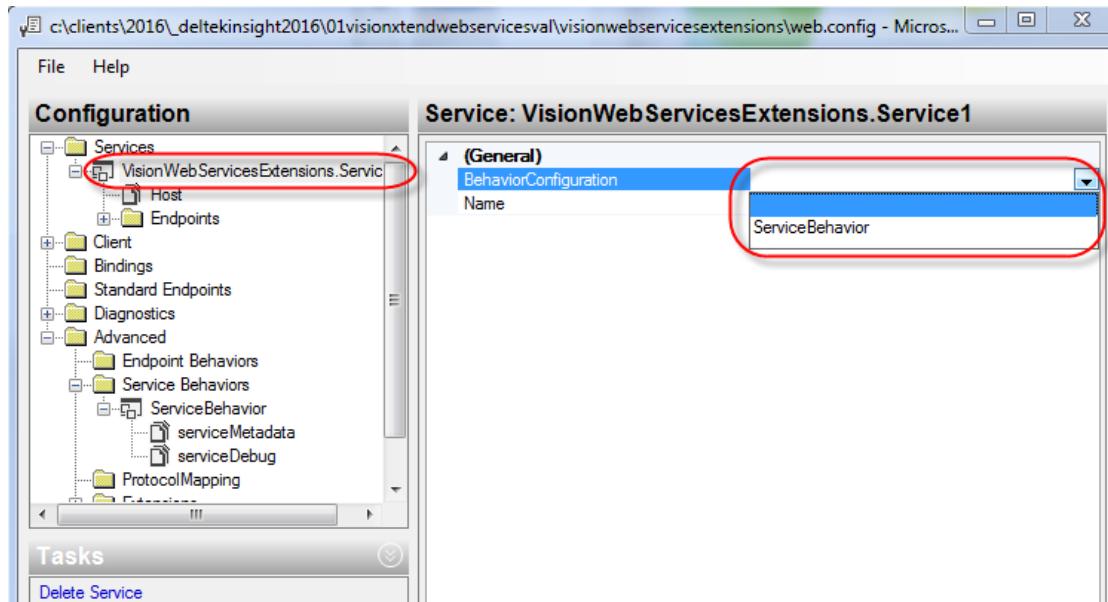
Building Your First Web Service

- Click Finish to create the service configuration



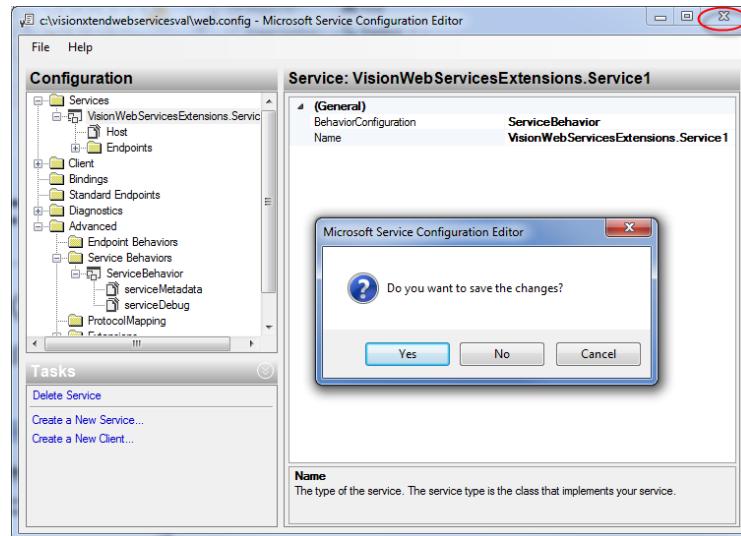
Building Your First Web Service

- Select the service configuration that was just created and attached the service behavior we defined earlier to the BehaviorConfiguration property



Building Your First Web Service

- Click on the dialog's X button (upper right corner) and you will be prompted to save changes. Click on Yes and your Web Service is ready for deployment.



Configuring The Web Service

Configuring The Web Service

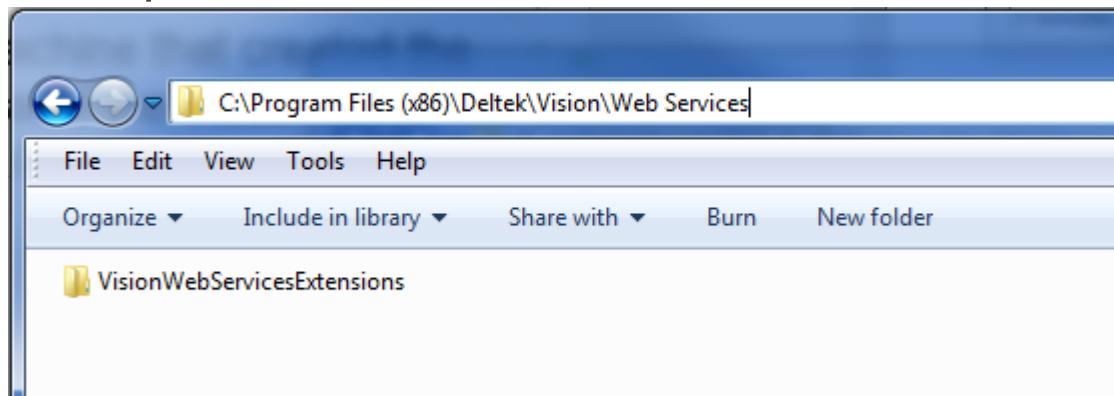
- The web service can reside anywhere on the Vision Web/App server or even a different server. However, it is suggested that the web service exists in the following path:

C:\Program Files (x86)\Deltek\Vision\

Web Services\<NameOfYourCustomWebService>

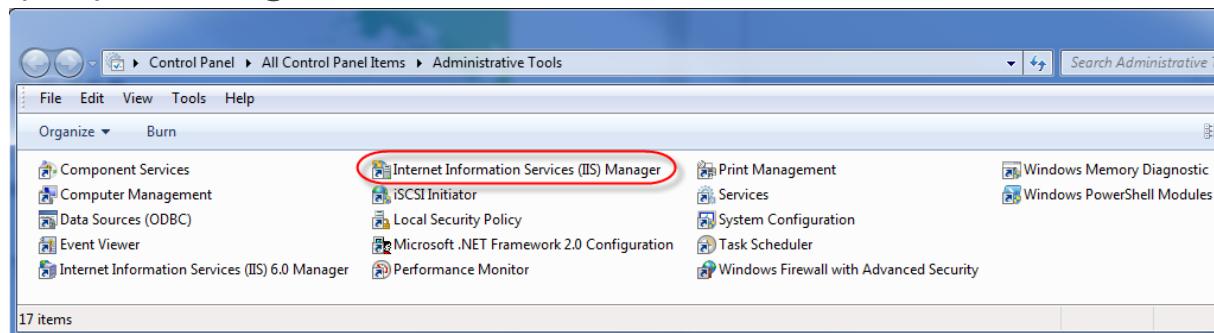
Configuring The Web Service

- If the server is different than the development machine that created the web service then copy the entire folder that contains the Visual Studio project and paste the folder onto the Vision Web/App Server. For example:



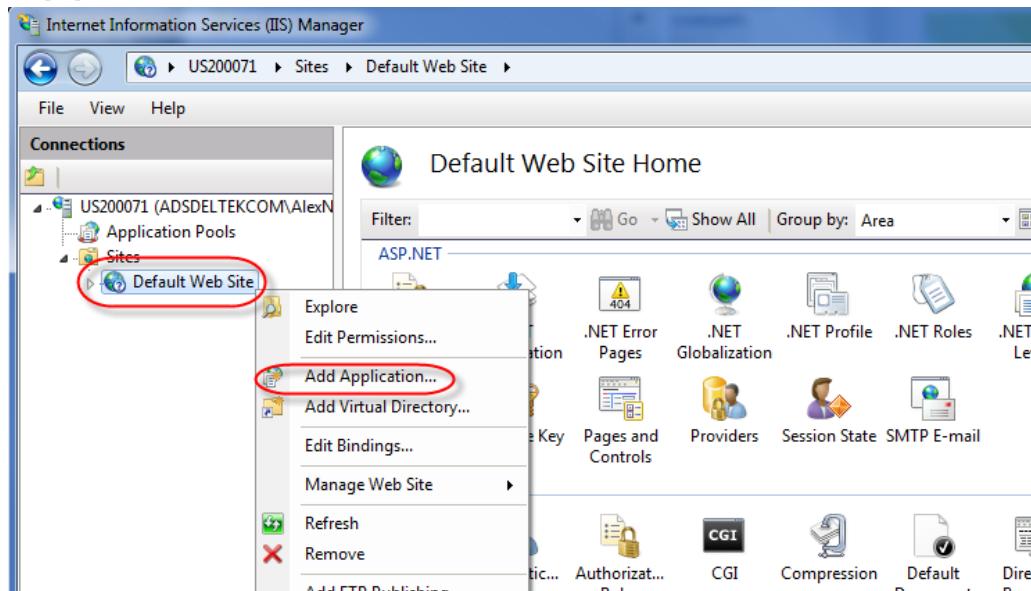
Configuring The Web Service

- After copying the web service folder onto the Vision Web/App server, an Application will need to be created within Internet Information Services (IIS).
- Open Control Panel\Administrative Tools\Internet Information Services (IIS) Manager.



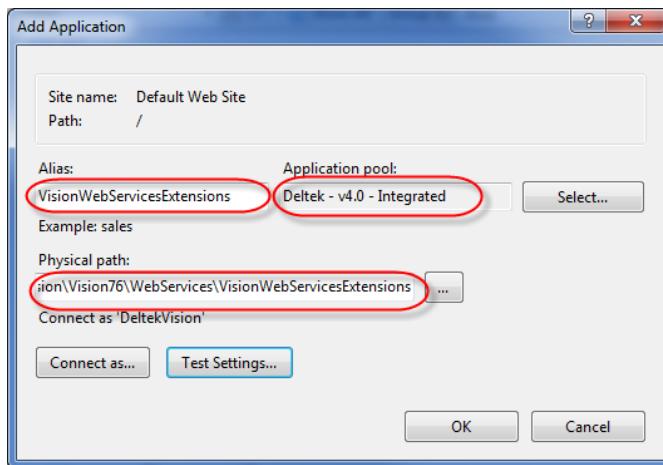
Configuring The Web Service

- Drill down and right click on “Default Web Site” and select “Add Application”:



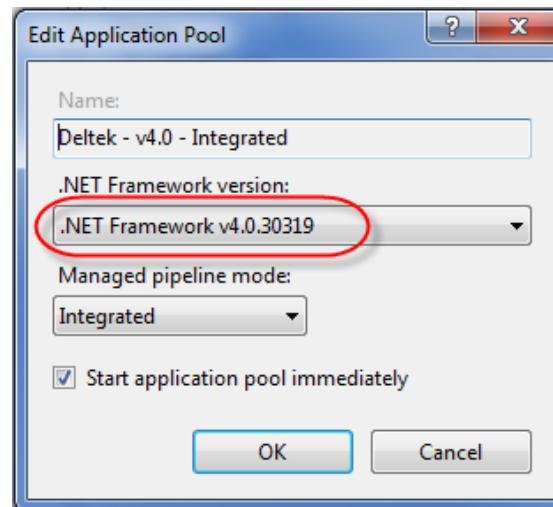
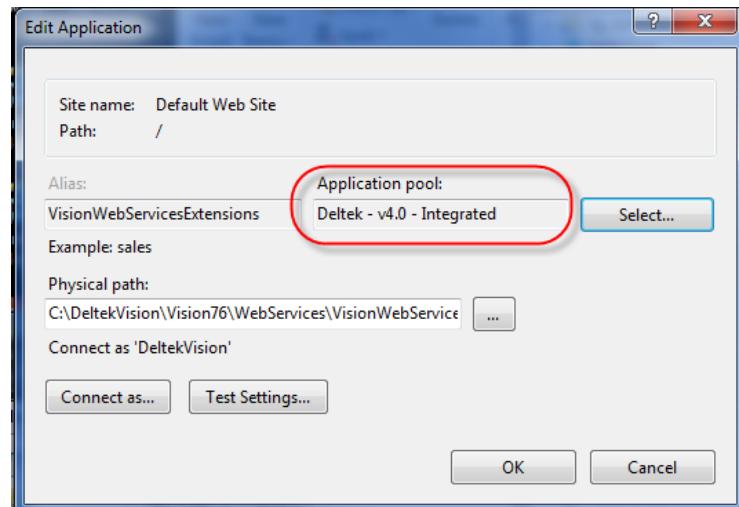
Configuring The Web Service

- Fill in the ‘Alias’, ‘Physical path’ and ‘Application pool’ with the appropriate values.



Configuring The Web Service

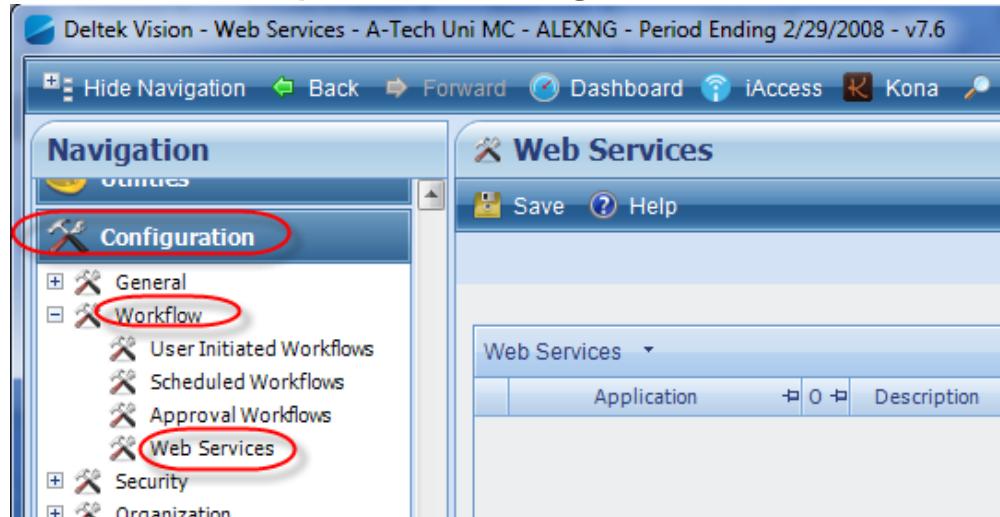
- For the application pool, you'd want to select the same one that Vision is using or if in a different server mimic the application pool that Vision is using.



How To Make Vision Call The Web Service?

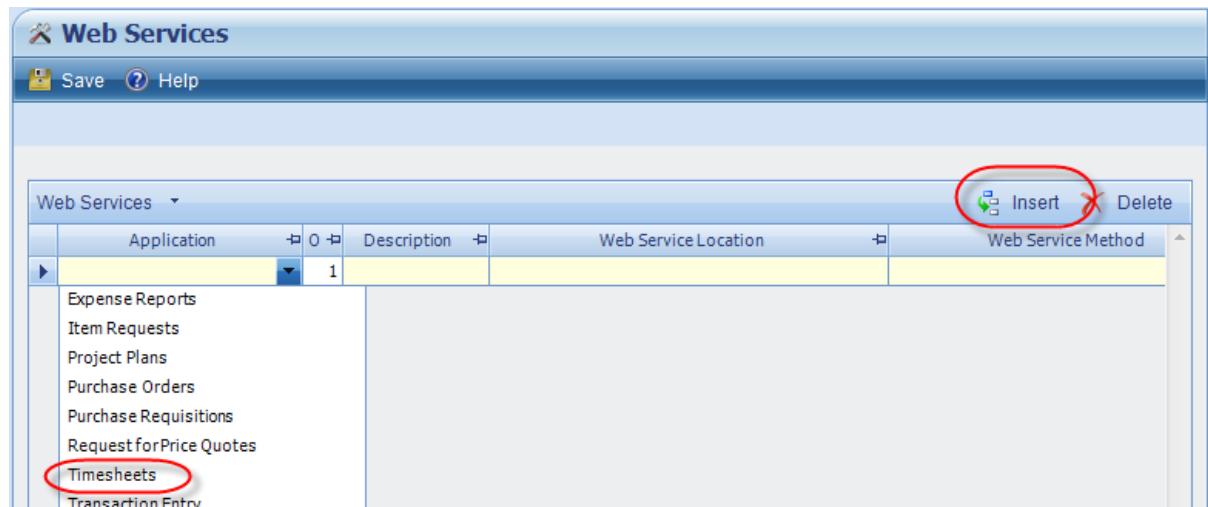
How To Make Vision Call The Web Service?

- In Vision, expand ‘Configuration\Workflow’ then click on “Web Services”



How To Make Vision Call The Web Service?

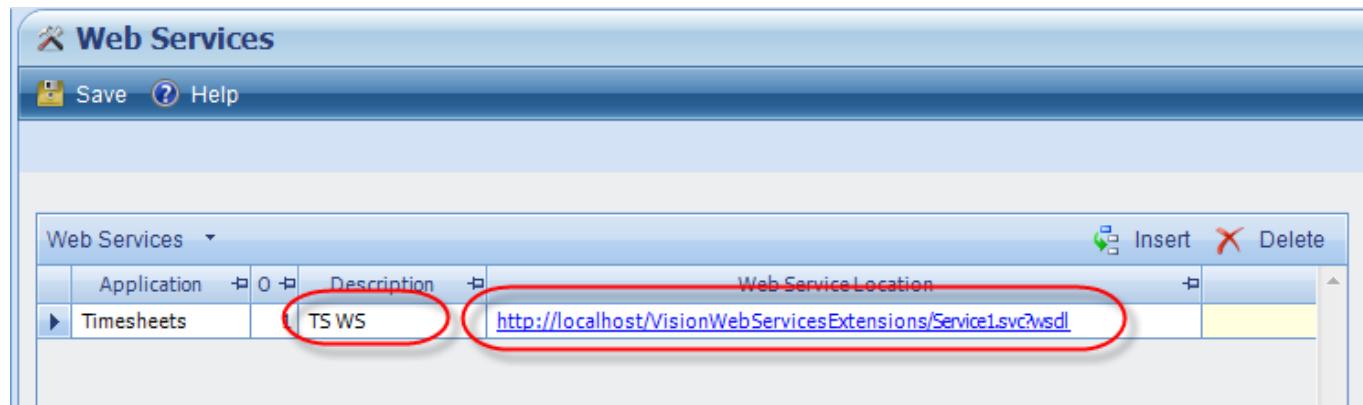
- Click “Insert” and choose “Timesheets”:



How To Make Vision Call The Web Service?

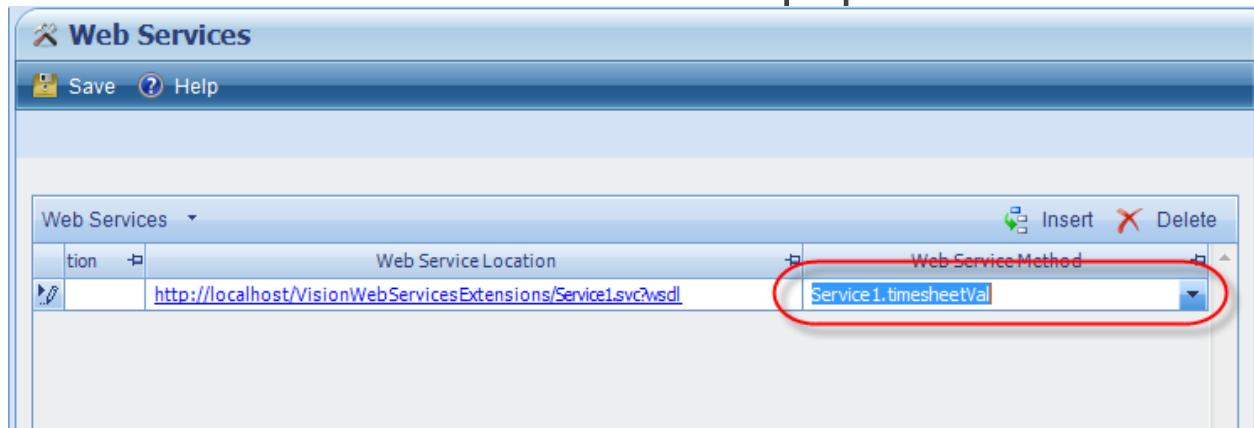
- Enter a Description and the Web Service Location. The web service location should be followed by '?wsdl'. WSDL means Web Service Definition Language. An example of a web service location is:

<http://localhost/VisionWebServicesExtensions/Service1.svc?wsdl>



How To Make Vision Call The Web Service?

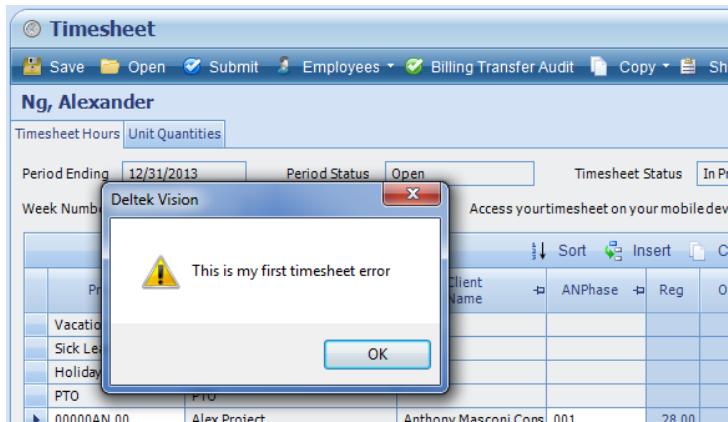
- Once the web service location has been entered then the drop down for 'Web Service Method' should be populated with a method:



- In case you have multiple methods in your web service, check to make sure it is the desired method you want to call and then click 'Save'.

How To Make Vision Call The Web Service?

- Verify that the web service is running correctly by opening the ‘Timesheet’ module, entering some data and saving the timesheet.
- After the save is called Vision should display the contents of the warning message:



What Is Passed Into The Web Service?

What Is Passed Into The Web Service?

- The web service should expect a string as its only parameter:

```
3 references
Public Function timesheetVal(ByVal inData As String) As String Implements IService1.timesheetVal
    timesheetVal = "<errors warning= "y" ><error>This is my first timesheet error</error></errors>" 
End Function
'-----
```

- That string is in XML format.
- Thus, it is possible to load that string into an XML Dom and start parsing it.

What Is Passed Into The Web Service?

- Using the Timesheet application as example, the following is the XML passed into the function “timesheetVal” (inData):

```
- <ROOT>
+ <tkMaster>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
</ROOT>
```

What Is Passed Into The Web Service?

- The ‘tkMaster’ node contains one child node called ‘ROW’ and that child node contains multiple children:

```
- <ROOT>
  - <tkMaster>
    - <ROW>
      <Employee>00001</Employee>
      <EndDate>2013-12-31T00:00:00.000</EndDate>
      <Submitted>N</Submitted>
      <Selected>N</Selected>
      <SubmittedBy/>
      <ApprovedBy/>
      <TKGroup> </TKGroup>
      <EmployeeCompany>00</EmployeeCompany>
    </ROW>
  </tkMaster>
+ <tkDetail>
+ <tkDetail>
```

- Each of the ‘ROW’ child nodes represents a column in the Vision “tkMaster” table.

What Is Passed Into The Web Service?

- The tkMaster table is the master table for Timesheets.
- The primary keys in the tkMaster table are Employee and EndDate.

What Is Passed Into The Web Service?

- The ‘tkDetail’ node consists of many child nodes. Each child node represents a column in the tkDetail table:

```
<ROOT>
+ <tkMaster>
- <tkDetail>
  <EndDate>2013-12-31T00:00:00-08:00</EndDate>
  <Employee>00001</Employee>
  <Seq>4</Seq>
  <TransDate>2013-12-16T00:00:00-08:00</TransDate>
  <Category>4</Category>
  <WBS1>0000002.00</WBS1>
  <WBS2>000</WBS2>
  <WBS3></WBS3>
  <LaborCode>00000</LaborCode>
  <RegHrs>1.0000</RegHrs>
  <OvtHrs>0.0000</OvtHrs>
  <SpecialOvtHrs>0.0000</SpecialOvtHrs>
  <BillCategory>0</BillCategory>
  <Locale>CT</Locale>
  <RegAmt>0.0000</RegAmt>
  <OvtAmt>0.0000</OvtAmt>
  <SpecialOvtAmt>0.0000</SpecialOvtAmt>
  <RegAmtProjectCurrency>0.0000</RegAmtProjectCurrency>
  <OvtAmtProjectCurrency>0.0000</OvtAmtProjectCurrency>
  <SpecialOvtAmtProjectCurrency>0.0000</SpecialOvtAmtProjectCurrency>
  <BillExt>0.0000</BillExt>
  <OvtPct>0.0000</OvtPct>
  <SpecialOvtPct>0.0000</SpecialOvtPct>
  <Rate>0.0000</Rate>
  <OvtRate>0.0000</OvtRate>
  <SpecialOvtRate>0.0000</SpecialOvtRate>
  <RateProjectCurrency>0.0000</RateProjectCurrency>
  <OvtRateProjectCurrency>0.0000</OvtRateProjectCurrency>
  <SpecialOvtRateProjectCurrency>0.0000</SpecialOvtRateProjectCurrency>
  <StartTime>2013-12-16T11:00:00-08:00</StartTime>
  <EndTime>2013-12-16T12:00:00-08:00</EndTime>
  <EmployeeCompany>00</EmployeeCompany>
</tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
```

What Is Passed Into The Web Service?

- The tkDetail table is the table that contains the Timesheets detail data to the tkMaster table.
- The primary keys in the tkDetail are EndDate, Employee, Seq and TransDate.

What Is Passed Into The Web Service?

- Given a timesheet with two rows of projects, each with two days of data. The XML that is sent to the web service is then built reading the Timesheet from the first row, reading each date in the first row that has value and then continues reading the second row from left to right. The following is the XML string that is then passed to the web service:

Project		Project Name		Reg	Ovt	Ovt-2	Sat 9/1	Sun 9/2	Mon 9/3	Tue 9/4
Vacation	Vacation									
Sick Leave	Sick Leave									
Holiday	Holiday									
00000AN.00	Alex Project	Anthon	22.00				6.00	5.00		
► 0000AN8.00	alex 8		26.00				6.00	5.00		

```
- <ROOT>
+ <tkMaster>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
</ROOT>
```

What Is Passed Into The Web Service?

Project	Project Name	Reg	Ovt	Ovt-2	Sat 9/1	Sun 9/2	Mon 9/3	Tue 9/4
Vacation	Vacation							
Sick Leave	Sick Leave							
Holiday	Holiday							
00000AN.00	Alex Project	Anthr	22.00		6.00	5.00		
► 00000AN8.00	alex 8		26.00		6.00	5.00		

```
- <ROOT>
+ <tkMaster>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
</ROOT>
```

- If we open up the first tkDetail node the value of the “WBS1” node will be 00000AN.00 and the value of the “TransDate” node will be “09/03/2012”.
- If we open up the second tkDetail node the value of the “WBS1” node will be 00000AN.00 and the value of the “TransDate” node will be “09/04/2012”.

What Is Passed Into The Web Service?

Project	Project Name	Reg	Ovt	Ovt-2	Sat 9/1	Sun 9/2	Mon 9/3	Tue 9/4
Vacation	Vacation							
Sick Leave	Sick Leave							
Holiday	Holiday							
00000AN.00	Alex Project	Anthr	22.00		6.00	5.00		
► 0000AN8.00	alex 8		26.00		6.00	5.00		

```
- <ROOT>
+ <tkMaster>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
+ <tkDetail>
</ROOT>
```

- If we open up the third tkDetail node the value of the “WBS1” node will be 0000AN8.00 and the value of the “TransDate” node will be “09/03/2012”.
- If we open up the fourth tkDetail node the value of the “WBS1” node will be 0000AN8.00 and the value of the “TransDate” node will be “09/04/2012”.

How To Parse The XML?

How To Parse The XML?

- The basic form of the XML string that is passed into the web service is:

```
<ROOT><tkMaster><ROW></ROW></tkMaster><tkDetail></tkDetail><tkDetail></tkDetail>...</ROOT>
```

- First, we can use an XmlDocument object to load the XML string:

```
Dim timesheetDom As New XmlDocument  
'Loading inData  
timesheetDom.LoadXml(inData)
```

- To read the values from the “tkMaster” node we can use the following code:

```
Dim tkMasterNode As XmlNode  
'Initializing tkMasterNode  
tkMasterNode = timesheetDom.DocumentElement.SelectSingleNode("descendant::ROW")
```

How To Parse The XML?

- Now we can read the values from the “tkMaster” node as follows:

```
Dim test As String  
  
test = tkMasterNode.Item("EndDate").InnerText  
test = tkMasterNode.Item("Employee").InnerText  
test = tkMasterNode.Item("Submitted").InnerText  
test = tkMasterNode.Item("Selected").InnerText  
test = tkMasterNode.Item("SubmittedBy").InnerText  
test = tkMasterNode.Item("ApprovedBy").InnerText  
test = tkMasterNode.Item("TKGroup").InnerText
```

How To Parse The XML?

- To read the values from the “tkDetail” nodes we can use similar code:

```
Dim tkDetailNode As XmlNode  
tkDetailNode = timesheetDom.DocumentElement.SelectSingleNode("descendant::tkDetail")  
  
Dim test As String  
While Not (tkDetailNode Is Nothing)  
    test = tkDetailNode.Item("EndDate").InnerText  
    test = tkDetailNode.Item("Employee").InnerText  
    test = tkDetailNode.Item("Seq").InnerText  
    test = tkDetailNode.Item("TransDate").InnerText  
    test = tkDetailNode.Item("Category").InnerText  
    test = tkDetailNode.Item("WBS1").InnerText  
    test = tkDetailNode.Item("WBS2").InnerText  
    test = tkDetailNode.Item("WBS3").InnerText  
    test = tkDetailNode.Item("LaborCode").InnerText  
    test = tkDetailNode.Item("RegHrs").InnerText  
    test = tkDetailNode.Item("OvtHrs").InnerText  
    test = tkDetailNode.Item("SpecialOvtHrs").InnerText  
    tkDetailNode = tkDetailNode.NextSibling  
End While
```

What Should The Web Service Return?

What Should The Web Service Return?

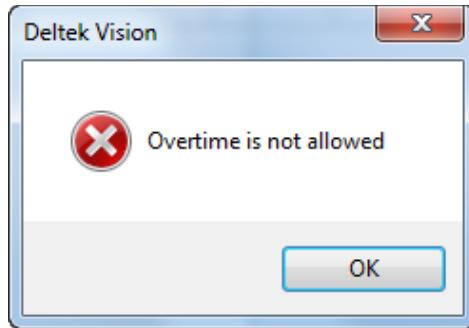
- In its most basic form the web service should either return an empty string ("") or "<errors></errors>" if no errors are found.
- Any error that is to be displayed to the user should be enclosed in an error node (<error></error>).

What Should The Web Service Return?

- For example, if an employee cannot enter overtime then the return error string should be as follow

```
<errors><error>Overtime is not allowed</error></errors>
```

and the user will be presented with this error message



What Should The Web Service Return?

- Passing back such an error string will prevent the Timesheet from being saved
- To only display a warning and allow the Timesheet to be saved add the “warning” attribute to the “errors” node.

```
<errors warning='Y'><error>Warning: Less than 40 hours  
submitted</error></errors>
```

What Should The Web Service Return?

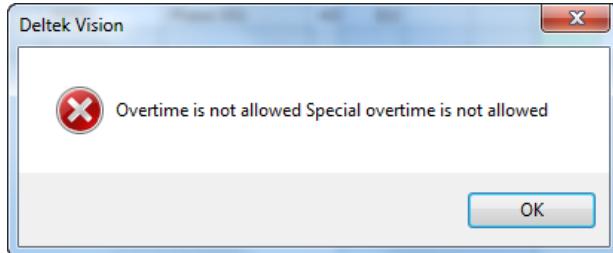
- If there are two errors then the error string would appear similar to the following:

```
<errors>  
  <error>Overtime is not allowed </error>  
  <error>Special overtime is not allowed</error>  
</errors>
```

- Each error is wrapped in its own “`<error>`” node within its parent “`<errors>`”.

What Should The Web Service Return?

- The dialog would appear as follows:



- You can add formatting like a carriage return and/or line feed to make your messages more readable:

```
<errors><error>Overtime is not allowed</error>
```

```
<error>"&Chr(13)&Chr(10)&"Special overtime is not allowed</error>
```

```
</errors>"
```

What Should The Web Service Return?

- It is also possible to have a global error string and add errors to it as they occur.
- Thus, instead of having multiple error nodes add one error node that has one or more errors separated by the carriage return and line feed characters.

```
Private errorString As String = ""  
  
| Private Sub AddErrorToErrorString(ByVal errorMessage As String)  
|   If errorString <> "" Then  
|     errorString = errorMessage  
|   Else  
|     errorString += Chr(13) & Chr(10) & errorMessage  
|   End If  
End Sub
```

What Should The Web Service Return?

- Similar to having a global XmlDocument that holds the Timesheet data it is possible to have a global XmlDocument that would be responsible for gathering all of the error nodes.
- Declare the global error XML Document:

```
Private errorXMLDom As New XmlDocument  
Private errorRoot, errorChild As XmlElement
```

- To add errors to the XmlDocument we would first instantiate it with root node and add the errors.

```
Private Sub initializeErrorRoot()  
    If errorRoot Is Nothing Then  
        errorRoot = errorXMLDom.CreateElement("errors")  
        errorXMLDom.AppendChild(errorRoot)  
    End If  
End Sub
```

What Should The Web Service Return?

- To make your life easier you can also create a method that would append an error that you pass in as follows:

```
Private Sub AddError(ByVal errorMsg)
    If errorRoot Is Nothing Then
        initializeErrorRoot()

        errorChild = errorXMLDom.CreateElement("error")
        errorChild.InnerText = errorMsg
    Else
        errorChild = errorXMLDom.CreateElement("error")
        errorChild.InnerText = Chr(13) & Chr(10) & errorMsg
    End If

    errorRoot.AppendChild(errorChild)
    errorChild = Nothing
End Sub
```

What Should The Web Service Return?

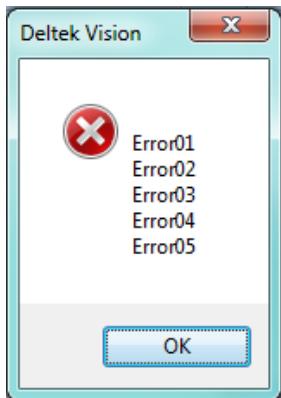
- Next we can just call those methods to add a couple of errors to be returned:

```
Public Function TimesheetValidationTest(ByVal inData As String) As String
    initializeErrorRoot()
    AddError("Error01")
    AddError("Error02")
    AddError("Error03")
    AddError("Error04")
    AddError("Error05")

    TimesheetValidationTest = errorXMLDom.OuterXml
End Function
```

What Should The Web Service Return?

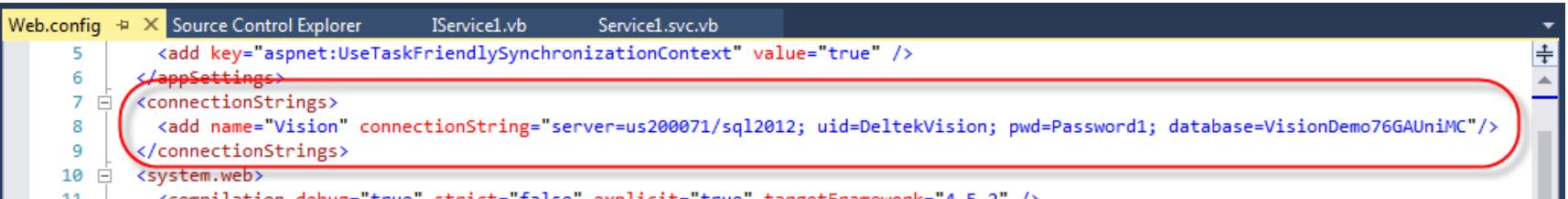
- The result would be following error dialog in Vision when a timesheet is saved:



Tips and Tricks

Tips and Tricks

Often times it will be required to obtain a connection back to the database. If this is the case then the database connection string can be stored in the Web.config file inside the node called “connectionStrings” for easy update:



```
Web.config Source Control Explorer IService1.vb Service1.svc.vb
5      <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />
6  </appSettings>
7  <connectionStrings>
8    <add name="Vision" connectionString="server=us200071/sql2012; uid=DeltekVision; pwd=Password1; database=VisionDemo76GAUniMC" />
9  </connectionStrings>
10 <system.web>
11   <compilation debug="true" strict="false" explicit="true" targetFramework="4.5.2" />
```

Tips and Tricks

Then you can refer to that connection string within code as follows:

```
Public Sub OpenVisionConnection()
    Dim connectionString As String
    Dim sqlConnection As SqlConnection

    connectionString = ConfigurationManager.ConnectionStrings("Vision").ConnectionString
    sqlConnection = New SqlConnection(connectionString)

    sqlConnection.Open()
End Sub
```

This way if you need to change the connection string down the road you only have to update Web.Config and don't have to recompile your code.

Tips and Tricks

Once the connection is opened, the database can be queried. For example, what if we wanted to know if a particular project was a vacation project. The following code could be applied:

```
Private Sub isVacationProject(ByVal currentWBS1 As String)
    Dim connectionString As String
    Dim tmpConnection As SqlConnection
    Dim tmpSqlCommand As SqlCommand
    Dim sqlStmt As String
    Dim tmpSqlDataReader As SqlDataReader

    connectionString = ConfigurationManager.ConnectionStrings("Vision").ConnectionString

    tmpConnection = New SqlConnection(connectionString)

    tmpConnection.Open()

    sqlStmt = "SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED; Select custRegularProject " + _
    "From ProjectCustomTabFields Where WBS1 = '" & currentWBS1 & "' And WBS2 = '' And " + _
    "WBS3 = '' And custVacationProject = 'Y'"

    tmpSqlCommand = New SqlCommand(sqlStmt, tmpConnection)

    tmpSqlDataReader = tmpSqlCommand.ExecuteReader

    If tmpSqlDataReader.HasRows() Then
        _vacationProjectFound = True
    End If

    tmpConnection.Close()
End Sub
```

Tips and Tricks

- Some important items to address in the code are the following:
 - The following within the SQL script ensures that when the database is queried the records are read only and thus will not create a record lock.

```
sqlStmt = "SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;"
```

- Always ensure that any database connection that is opened is also closed:

```
ttmpConnection.Close()
```

Questions & Contact

Alexander Ng

AlexNg@deltek.com

Thank you!

Top 5 Reasons to Team with Deltek Consulting on Your Implementation Project



One provider
for all things
Deltek



Certified
team of
consultants



A well-planned
and managed
implementation with
predictable success



Expertise
across all
Deltek
software



A deeper level
of project
governance



For more information visit www.deltek.com/globalconsulting

Remember to visit
the mobile app and
complete the
session survey. ➤

Your feedback is greatly appreciated!

