Workflow 9.0: Consuming the Workflow Web Service

White paper

January 2013

Laserfiche®

Table of Contents

Introduction to the Workflow Web Service

The Workflow Web Service lets you transparently integrate Workflow with other programs, allowing those programs to pass information (such as the initiator, input/output parameters, and starting entry) to Workflow. This type of integration enhances flexibility and increases efficiency by allowing you to start workflows without opening a Laserfiche interface.

This paper provides information about consuming the Workflow Web Service. It is written for development users tasked with designing integrations.

Creating the Web Service Project

This example uses Visual Studio 2010 and creates a button that starts a workflow instance from a non-Laserfiche program.

To configure the service reference:

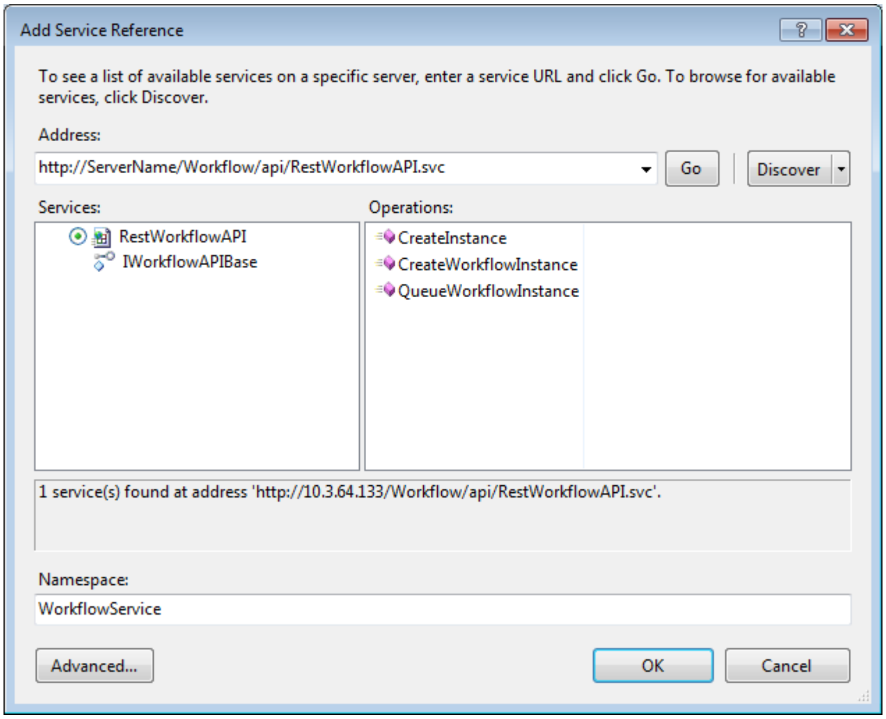
1. Open Visual Studio and create a new project by clicking **New Project.**
    a. Select **Visual C#** and **Windows Forms Application**. Name the project and, optionally, specify a location. Click **OK**.
2. Add a service reference to the project by right-clicking the project name in the **Solution Explorer Pane** and selecting **Add Service Reference.**
3. Add the Workflow's Web service by supplying the following default address: http://*WorkflowServerName*/workflow/api/RestWorkflowAPI.svc.

    Note: If the Workflow Server you specify is using a port other than the default port 80, input the port number after the Server name: http://*WorkflowServerName*:**81**/workflow/api/RestWorkflowAPI.svc

    Note: If the Workflow Server you specify uses a different virtual directory than workflow, input that virtual directory:

    http://*WorkflowServerName*/**LFWorkflow**/api/RestWorkflowAPI.svc

4. Click **Go**. Once the service downloads, select the RestWorkflowAPI service. Name the service: *WorkflowService*.



To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Discover.

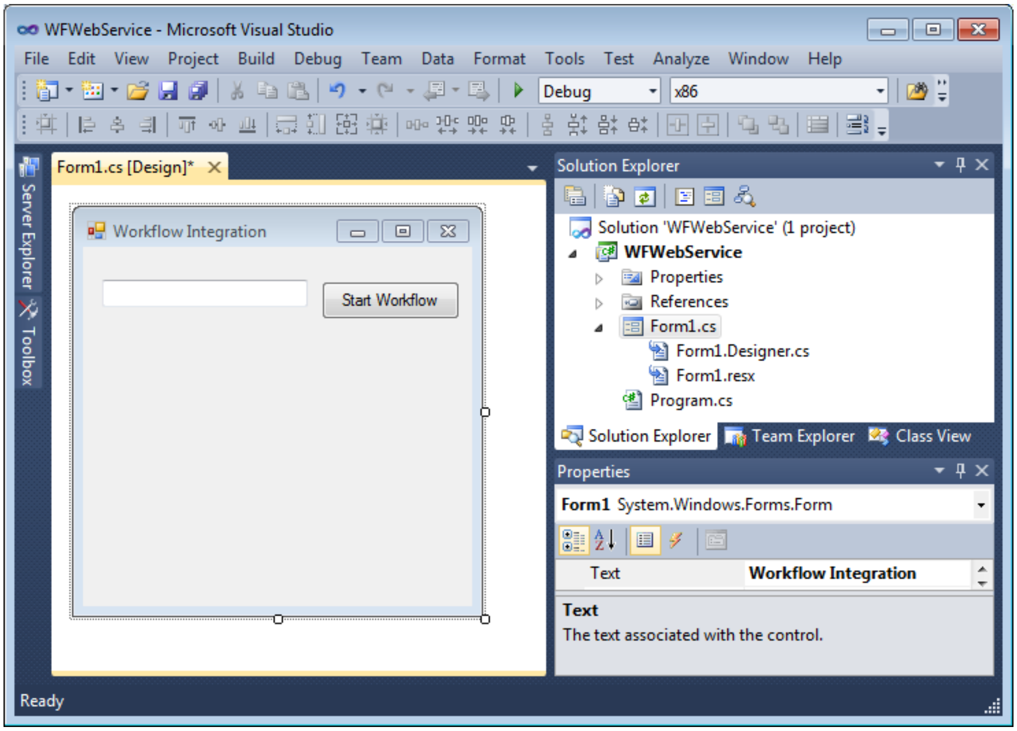
Address: http://ServerName/Workflow/api/RestWorkflowAPI.svc

1 service(s) found at address 'http://10.3.64.133/Workflow/api/RestWorkflowAPI.svc'.

Namespace: WorkflowService

5. Click **OK**.

6. Create your user interface.

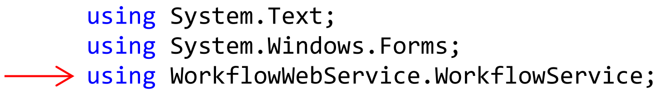


Form1 System.Windows.Forms.Form

Text: Workflow Integration

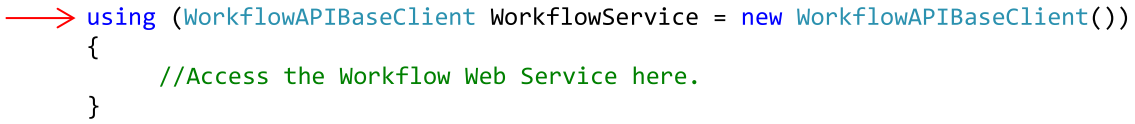
Text
The text associated with the control.

7. Double-click on the button object in the **Design Pane** to automatically populate its code.
8. Configure the code: (See the sample code.)
   a. Add the project name and the namespace you gave the service to the using statements.



```
using System.Text;
using System.Windows.Forms;
using WorkflowWebService.WorkflowService;
```

   b. Inside ***ButtonName*****Button_Click**, add:



```
using (WorkflowAPIBaseClient WorkflowService = new WorkflowAPIBaseClient())
{
     //Access the Workflow Web Service here.
}
```

   c. Inside **WorkflowService**, create an InstanceCreationData object, and configure it. Then, use an operation, like CreateWorkflowInstance, to start a workflow. See the sample code and available objects for more details.

Note: The InstanceCreationData object cannot be null.

Sample Code

The following example assumes you followed the basic steps outlined in the first section of this paper. The example code applies to the example interface.

See a list of available instance creation data statements in the Available Objects section.

Key: The project name is “WorkflowWebService,” the service reference is “WorkflowService,” and the button name is “WorkflowButton.”

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using WorkflowWebService.WorkflowService;

…
private void WorkflowButton_Click(object sender, EventArgs e)
{
 using(WorkflowAPIBaseClient WorkflowService = new WorkflowAPIBaseClient())
           {

          //Define the workflow’s starting information (i.e., starting entry and initiator)
          InstanceCreationData creationData = new InstanceCreationData();
          creationData.StartingEntry = new InstanceEntryData();
          creationData.Initiator = new InstanceUserData();
          creationData.Initiator.InitiatorDisplayName = "DisplayName";
          creationData.Initiator.InitiatorName = @"DOMAIN\UserName";
               creationData.Initiator.IntiatorSID =
                 System.Security.Principal.WindowsIdentity.GetCurrent().Owner.Value;
          creationData.InstanceCreationFlags = InstanceCreationFlags.FromUser;

          //Specify the parameters you want to pass to Workflow
          InstanceParameterData parameter1 = new InstanceParameterData();
          parameter1.Name = "ParameterName";
          parameter1.Value = "ParameterValue";
          creationData.ParameterCollection = new InstanceParameterCollection();
          creationData.ParameterCollection.Add(parameter1);

            creationData.RuleName = "StartingRule";

            //Start a workflow by typing its name in the text box
            InstanceCreationResultData results =
            WorkflowService.CreateWorkflowInstance(this.uxWorkflowName.Text, creationData);

            //The following is optional and used for information gathering
            string instanceId = results.instanceId;
            string fault = results.fault!= null ? results.fault.Detail : string.Empty;
            }
}
…
```

Available Operations

CreateWorkflowInstance: This operation will start a workflow based on the information specified in the InstanceCreationData object, however, the WorkflowName parameter will be ignored. This operation takes the workflow name as a separate parameter.

QueueWorkflowInstance: Queues the workflow instance instead of creating it synchronously during the call to the Workflow Web Service. The QueueWorkflowInstance operation is similar to the Workflow Subscriber.

> **Note:** Since the workflow instance will not be started immediately, some information about the instance, such as the instance ID, will not be returned.

Available Objects

InstanceCreationData:

- **Initiator:** The user set as having initiated the workflow.
- **WorkflowName:** This object is reserved for future versions.
- **Origin:** The machine responsible for starting the workflow. Used in diagnostic messages.
- **Originator:** The source of the workflow starting command. Used in diagnostic messages.
- **InstanceCreationFlag:** This object is reserved for future versions.
- **ParamaterCollection:** A collection of parameters (InstanceParameterData objects) passed to the workflow.
- **formData:** Used by Laserfiche Forms.
- **ExtensionData:** This object is reserved for future versions.
- **RuleName:** The name of the rule that Workflow will report as starting the workflow. This does not need to be the name of a real starting rule.
- **StartingEntry:** The starting entry the workflow instance will use.
- **TimeStamp:** This object is reserved for future versions.

InstanceParameterData: The following types can be passed as parameters: bool, char, byte, short, ushort, int, uint, long, ulong, float, double, decimal, string, Guid, DateTime, and TimeSpan.

> **Note:** Arrays (and most built-in collections) of these types can also be passed.

- **Name:** Name of the parameter being passed to Workflow.
- **Value:** Value for the parameter specified by InstanceParameterData.Name.

InstanceCreationResultData: This object defines information that can be returned by the workflow instance, such as error messages.

- **fault:** Failure message, if any.
- **instanceId:** The workflow instance ID.

InstanceUserData:

- **InitiatorDisplayName:** The workflow initiator’s display name.
- **InitiatorName:** The workflow initiator’s account name.
- **IntiatorSID:** The security identifier of the workflow initiator.
- **Email:** The initiator’s e-mail address.

InstanceEntryData:

- **EntryGUID:** The Laserfiche entry GUID.
- **EntryID:** The Laserfiche entry identification number.
- **FullPath:** The full path of the entry.
- **Repository:** The name of the Laserfiche repository being interacted with.
- **Server:** The Laserfiche Server name.

Laserfiche®

Workflow 9.0: Consuming the Workflow Web Services
January 2013

Author: Sierra Jahoda
Editor: Sarah Seene
Technical Editor: Nathaniel Rook, Ed Heaney

Description:
This paper outlines how to use the Workflow Web Service to integrate Workflow with other programs.

Laserfiche
3545 Long Beach Blvd.
Long Beach, CA 90807
U.S.A

Phone: +1.562.988.1688
www.laserfiche.com

Laserfiche makes every effort to ensure the accuracy of these contents at the time of publication. They are for information purposes only and Laserfiche makes no warranties, express or implied, as to the information herein.