Initiating Workflows from Other Programs with a Custom Start Form

This guide provides information about how to invoke Nintex workflows from other programs, such as Microsoft Dynamics CRM and Salesforce, by using custom start forms created in Nintex Forms 2013 for SharePoint to pass information from other programs to Nintex Workflow 2013 workflows created in Nintex Workflow 2013 for SharePoint.

The included integration example demonstrates how to create a workflow and custom start form that can be initiated from the user interface of Microsoft Dynamics CRM.

Initiating workflows

There are a few ways by which other programs can programmatically initiate workflows, including Nintex workflows, in SharePoint 2013. However, these methods require a familiarity with .NET Framework development, the development of custom code, and an environment that allows and supports such custom development.

For example, SharePoint 2013 provides a couple of .NET Framework methods that you can use, depending on the workflow type:

- Use the StartWorkflow method, included in the Microsoft.SharePoint.WorkflowServices.WorkflowInstanceService class, to initiate a workflow instance for a specified site workflow, or;
- Use the StartWorkflowOnListItem method, included in the same class, to initiate a workflow on a specified list item for a specified list workflow.

Both of these methods start a Workflow Manager Client 1.0 instance for a specified workflow subscription, optionally including a payload of input parameters.

Nintex Workflow 2013 and Nintex Forms 2013, used together, can provide an easier option than custom development in .NET Framework. This is done by invoking the InitiateWorkflow.aspx page, included with Nintex Workflow, with a URL that identifies the workflow to be initiated.

Initiating a workflow only solves half of the puzzle, in that you can't easily pass information from the other program to the workflow itself. The other half of the puzzle can be solved by adding a custom start form, using Nintex Forms 2013, to the workflow. The custom start form contains controls connected to workflow variables, with default values populated by query string parameters passed as part of the InitiateWorkflow.aspx URL. The controls can use a Nintex inline function to set their default values to the values from the query string parameters. This, in turn, sets the values of the connected workflow variables, allowing the workflow to use information passed from other programs to perform actions.

Prerequisites

This guide assumes that you are proficient with the concepts and operation of SharePoint 2013, Nintex Workflow 2013, Nintex Forms 2013, and Microsoft Dynamics CRM.

Before you can create the integration sample in SharePoint 2013 2013 and Microsoft Dynamics CRM, the following prerequisites must be satisfied:

- A SharePoint 2013 instance that includes Nintex Workflow 2013 and Nintex Forms 2013.
- A Microsoft Dynamics CRM instance.

Integration example

Nintex Workflow 2013 and Nintex Forms 2013, used together, provide a simple mechanism by which workflows can be initiated from other programs, such as Microsoft Dynamics CRM, without the need for complex programming.

To illustrate how this mechanism works, a short integration example has been provided. The following scenario defines the integration example:

An account representative receives a request from a sales account to extend their credit limit from $200,000 to $250,000. The account representative wants to initiate the process for approval of this request. The representative opens Microsoft Dynamics CRM, displays the sales account, and clicks a button named Credit Extension. This action displays a dialog box, in which the account representative can specify a new credit limit. The account representative sets it to $250,000, clicks Submit, and a credit extension workflow is started. The workflow uses the Nintex Connector for Microsoft Dynamics CRM to connect to the Microsoft Dynamics CRM instance, loads the account record for the sales account, and then uses a FlexiTask workflow action to have one or more assignees determine whether the credit extension request is approved, denied, or if the sales account is to be placed on credit hold pending further investigation. Once a response is received, the workflow updates the sales account record in Microsoft Dynamics CRM accordingly.

In this scenario, the process flow is as follows:

1. Microsoft Dynamics CRM specifies the unique identifier of the sales account record as a query string parameter, which can be retrieved and stored in a hidden control on a custom start form for a workflow in Nintex Workflow 2013.
2. The hidden control is used to set the value of a connected workflow variable for the workflow.
3. The workflow can then access and update the sales account record by using workflow actions included with the Nintex Connector for Microsoft Dynamics CRM.

   For more information about using query string parameters in Nintex Forms 2013, see "Working with query string parameters" on page 14.

The following steps are performed to create the integration sample:

1. "Create the workflow" below
2. "Create the custom start form" on page 9
3. "Publish the workflow and obtain the URL" on page 17
4. "Initiate the workflow from Microsoft Dynamics CRM" on page 17

Create the workflow

The workflow handles a credit extension request by retrieving the specified account record for the request from Microsoft Dynamics CRM, using a FlexiTask workflow action to determine whether the request is approved, denied, or placed on credit hold, and then updating the account record in Microsoft Dynamics CRM based on the response. The account record is identified by the value specified for the Record ID workflow variable, which will be connected to a control on a custom start form (to be created after the workflow is created.)

To create the workflow

1. Start the Nintex Workflow designer and create a new site workflow.
2. Create the following workflow variables, using the specified settings:

Name	Type	Default value or date	Show on start form
Record ID	Single line of text		Yes
Last Name	Single line of text		No
Fields	Collection		No
counter	Number	0	No
collcount	Number	0	No
CollectionCompleted	Yes/No	No	No
Field Value	Single line of text		No
Values	Collection		No
Date Check	Date and Time	Blank	No
XML Result	Single line of text		No
Credit Limit	Number	0	No
UserName	Single line of text	The user name of the Microsoft Dynamics CRM credentials used by the workflow.	No
Password	Single line of text	The password of the Microsoft Dynamics CRM credentials used by the workflow.	No
URL	Single line of text	The URL of the Microsoft Dynamics CRM instance used by the workflow.	No
URL2	Single line of text	The URL of the Microsoft Dynamics CRM instance used by the workflow.	No

3. Add and configure a new **Dynamics CRM retrieve record** workflow action.

   The **Dynamics CRM retrieve record** workflow action retrieves an existing account record, using the record identifier specified in the Record ID workflow variable, and returns an XML representation of the account record in the XML Result workflow variable.

   All of the workflow actions for Dynamics CRM Connector are included in the Nintex Live group of the toolbar in the Nintex Workflow designer.

   A. Add the workflow action to the design canvas.
   B. Click the dropdown arrow for the workflow action, and then click **Configure**.
   C. In the Configure Action - Dynamics CRM retrieve record dialog box, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Query	account
where	AccountId
equals	Reference to the Record ID workflow variable
Server URL	Reference to the URL2 workflow variable
Username	Reference to the UserName workflow variable
Password	Reference to the Password workflow variable
Record URL	URL
Fields	Fields
Values	Values
Result as XML	XML Result

4. Add and configure a new **Query XML** workflow action.

   The **Query XML** workflow action queries the contents of the XML Result workflow variable, using an XPath query to retrieve the name of the account, and stores the result in the Field Value workflow variable.

   The **Query XML** workflow action is included in the Integration group of the toolbar in the Nintex Workflow designer.

   A. Add the workflow action to the design canvas, after the **Dynamics CRM retrieve record** workflow action added in the previous step.

   B. Click the dropdown arrow for the workflow action, and then click **Configure**.

   C. In the Configure Action - Query XML dialog box, set the following properties to the specified values, and then click **Save**.

Property name	Property value
XML source	XML
XML	Reference to the XML Result workflow variable
Process using	XPath
XPath query	//name
Return results as	Text
Store result in	Field Value

5. Add and configure a new **Assign Flexi task** workflow action.

   The **Assign Flexi task** workflow action sends an email to specified assignees, to determine whether the assignee should approve the credit request, deny the credit request, or place the credit line for the account on hold.

   The **Assign Flexi task** workflow action is included in the User interaction group of the toolbar in the Nintex Workflow designer.

   A. Add the workflow action to the design canvas, after the **Query XML** workflow action added in the previous step..

   B. Click the dropdown arrow for the workflow action, and then click **Configure**.

   C. In the Configure Action - Assign Flexi task dialog box, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**	
Assignees	References to the email addresses of the assignees.	
Task description	The following text, replacing *<Credit Limit>* with a reference to the Credit Limit workflow variable: A credit limit of *<Credit Limit>* has been requested.	
Outcomes	Add the following outcomes:	
	Name	**Comments**
	Approve	Optional
	Reject	Optional
	Credit Hold	Required
Behaviour	First response applies	
Task name	Credit Limit Increase Request	
Task content type	Nintex Workflow Multi Outcome Task using Nintex Forms	
Priority	(2) Normal	
Form type	Nintex Forms	

6. Add and configure a new **Dynamics CRM update record** workflow action.

   The **Dynamics CRM update record** workflow action updates the existing account record, setting the Credit Limit field for the account record to the value of the Credit Limit workflow variable. This workflow action occurs only if an assignee approves the credit limit increase request.

   A. Add the workflow action to the design canvas, under the Approve branch of the **Assign Flexi task** workflow action added in the previous step.

   B. Click the dropdown arrow for the workflow action, and then click **Configure**.

   C. In the Configure Action - Dynamics CRM update record dialog box, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Update	account
where Record ID equals	Reference to the Record ID workflow variable
Fields	Add the following fields, with the specified values: **Field** — **Value** Credit Limit — Reference to the Credit Limit workflow variable
Server URL	Reference to the URL2 workflow variable
Username	Reference to the UserName workflow variable
Password	Reference to the Password workflow variable

7. Add and configure a new **Dynamics CRM update record** workflow action.

   The **Dynamics CRM update record** workflow action updates the existing account record, setting the Credit Hold and Credit Limit fields to "Yes" and 0, respectively, for the account record. This workflow action occurs only if an assignee places the credit line for the account on hold.

   A. Add the workflow action to the design canvas, under the Credit Hold branch of the **Assign Flexi task** workflow action added in the previous step.

   B. Click the dropdown arrow for the workflow action, and then click **Configure**.

   C. In the Configure Action - Dynamics CRM update record dialog box, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Update	account
where Record ID equals	Reference to the Record ID workflow variable
Fields	Add the following fields, with the specified values: **Field** / **Value** Credit Limit / 0 Credit Hold / Yes
Server URL	Reference to the URL2 workflow variable
Username	Reference to the UserName workflow variable
Password	Reference to the Password workflow variable

8. Save the workflow, setting the title of the workflow to Credit Extension.

   You do not need to publish the workflow or close the Nintex Workflow designer at this time. If you do close the Nintex Workflow designer, ensure that the workflow is open in Nintex Workflow designer before performing the next procedure.

Create the custom start form

After the workflow is created, the custom start form for the workflow must be created. In addition to storing information provided by other programs, the custom start form provides a dialog box that passes information from both the other program and the user to the workflow prior to initiation. The other program provides information, in the form of query string parameters, when calling the custom start form. Once the custom start form is displayed, the user provides information, in the form of data entry, and then submits the form. Submitting the custom start form initiates the workflow.

The following procedure assumes that the Nintex Workflow designer is open, and that the Credit Extension workflow created in the previous procedure is open.

To create the custom start form

1. In the Nintex Forms tab of the ribbon bar for the Workflow designer, click **Workflow Settings**, and then click **Workflow Settings**.
2. In the Workflow Settings dialog box, perform the following actions:
   A. Set the value of **Title** to Credit Extension.
   B. Ensure that **Start manually** is selected.
   C. Ensure that **Enable custom history messages** is selected.
3. In the ribbon bar for the Workflow Settings dialog box, click **Edit Start Form**, and then click **Edit with Nintex Forms**.

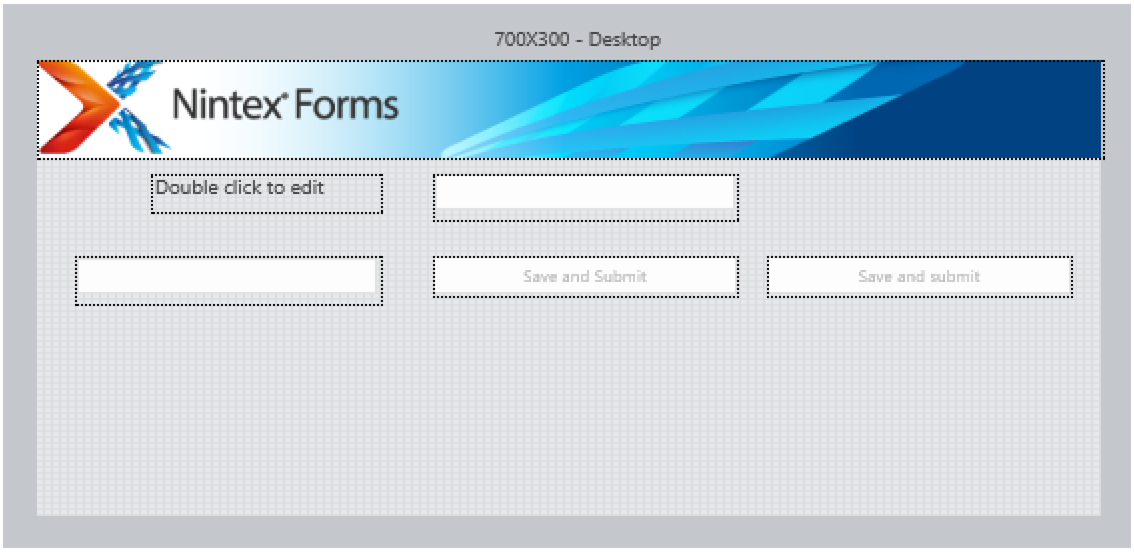
4. From the Form Controls pane of the Controls Toolbox for the Forms designer, drag the following controls to the design surface:
   - 1 Label control

     Position this control near the upper left corner of the design surface.
   - 2 Single Line Textbox controls

     Position the first control on the right side of the Label control, and then position the second control under the Label control.
   - 2 Button controls

     Position the first control under the first Single Line Textbox control, and then position the second control on the right side of the first control.

   The resulting form should resemble the following diagram:



5. From the design surface of the form, perform the following actions:
   A. Double-click the Label control, set the value of the control to Requested Credit Limit, and then click **Save**.
   B. Double-click the first Single Line Textbox control, on the right side of the Label control, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Name	Requested credit limit
Connected to	Credit Limit
Default value source	Use connected field's default value

   C. Double-click the second Single Line Textbox control, under the Label control, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Name	RecordID
Connected to	Record ID
Default value source	Expression
Default value	fn-GetQueryString(RecordID)
Visible	No

   This hidden control is used to retrieve the value of the specified query string parameter, RecordID, and store the value in the Record ID workflow variable. The query string parameter is specified by the other program at the time the custom start form is invoked.

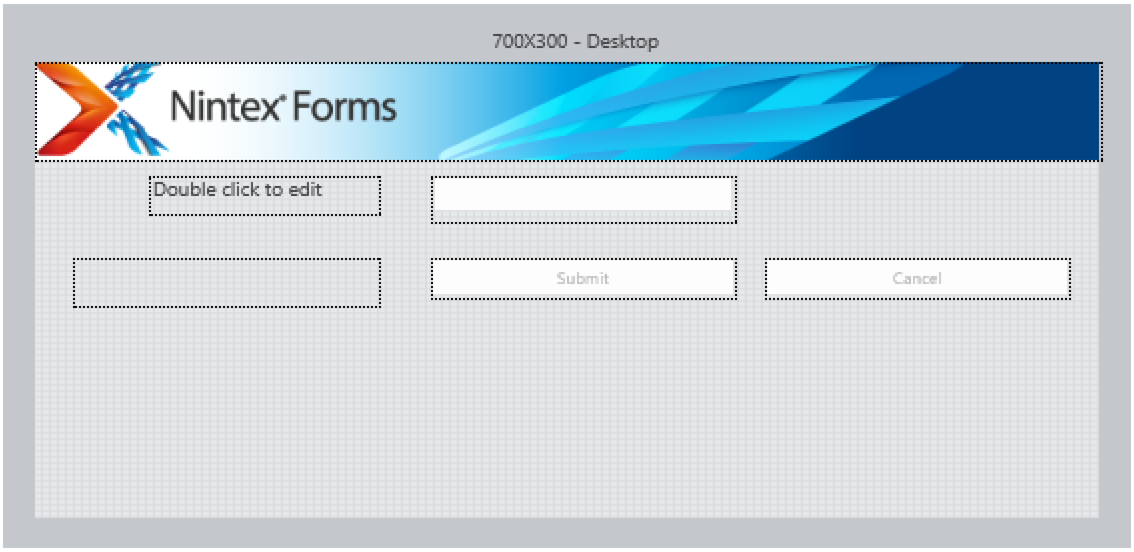
   D. Double-click the first Button control, under the first Single Line Textbox control, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Button action	Save and Submit
Button label	Submit

E. Double-click the second Button control, to the right side of the first Button control, set the following properties to the specified values, and then click **Save**.

Property name	**Property value**
Button action	Cancel
Button label	Cancel

The resulting form should resemble the following diagram:



6. From the Nintex Forms tab of the ribbon bar for the Forms designer, click **Settings**.

7. In Settings - Form, set the value of **Custom JavaScript** to the following script, and then click **Save**.

   ```
   NWF$(document).ready(
       function () {
           NWF$("#s4-ribbonrow").hide();
       }
   );
   ```

   The script hides the `<div>` element, named s4-ribbonrow, that contains the Nintex Forms ribbon for the form, allowing the custom start form to more closely resemble a dialog box.
8. Save the form, and then save the workflow settings.

   Do not publish the workflow or close the Nintex Workflow designer at this time.

Additional options for custom start forms

The custom start form defined for the integration example is basic in both function and appearance. You can easily modify the start form to provide a more complex experience for the user. For example, instead of using the default redirection URL, you could set the **Redirect URL** setting for the custom start form to a more sophisticated workflow completion form, specific to that workflow.

Working with query string parameters

The recommended method to pass information from other programs to a workflow in Nintex Workflow with a custom start form is to specify a query string parameter, which can be set when invoking the custom start form from another program, that the custom start form can then use to set the default value of a hidden control on the form itself. The value of the hidden control is then used to set the value of a connected workflow variable, which can be accessed by the workflow when performing actions.

To retrieve the value of a query string parameter, use the following Nintex inline function, replacing *ParameterName* with the name of the query string parameter:

```
fn-GetQueryString(ParameterName)
```

If the query string parameter specified in *ParameterName* is present in the query string for the URL, the value of the specified parameter is returned by the function; otherwise, a null value is returned. As with other inline functions, this function can be used wherever a reference to an inline function can be used. For example, the function can be used as the default value expression of a hidden control on a custom start form. When the form is invoked, the value of the hidden control is set to the value of the query string parameter, if the parameter is specified in the URL for the form.

Recommendations for using query string parameters

- Ensure that correct casing is used for query string parameters

  Query string parameters are case-sensitive. Ensure that you use the same casing for a query string parameter in both the custom start form and the URL used to invoke the form.

  This is especially important when using SharePoint and WSS query string parameters. For example, the custom start form looks and behaves differently if you specify `IsDlg=1` instead of `isdlg=1`, due to how SharePoint processes both query string parameters.

- Do not send sensitive data in custom query string parameters

  The value of a custom query string parameter is appended to the InitiateWorkflow.aspx URL, and sent in clear text as part of the URL. Even if the URL uses SSL to secure the request, there is no guarantee that the URL has not been preserved in some way - web server logs or browser history, for example.

- Do not use reserved names for custom query string parameters

  SharePoint and WSS reserve certain query string parameter names for forms. Do not define custom query string parameters that use one of these reserved names; unpredictable results may occur. For more information about the SharePoint and WSS reserved query string parameter names, see "Querystring parameters you should not use in your SharePoint application", on Microsoft TechNet at http://blogs.technet.com/b/stefan_gossner/archive/2009/01/30/querystring-parameters-you-should-not-use-in-your-sharepoint-application.aspx.

  Nintex also reserves certain query string parameter names for the InitiateWorkflow.aspx page. Do not define custom query string parameters that use any of the following parameter names :

  - ID
  - List
  - Source
  - TemplateID
  - WorkflowName

Publish the workflow and obtain the URL

Once the site workflow, with its custom start form, has been created, the workflow must be published before the custom start form can be referenced by other programs. After the workflow is published, you can retrieve the InitiateWorkflow.aspx URL from SharePoint for later use in other programs.

The following procedure assumes that the Nintex Workflow designer is open, and that the Credit Extension workflow created in the previous procedures is open.

To publish the workflow and obtain the URL

1. In the Nintex Workflow designer, publish the workflow, and then close the Workflow designer.
2. In SharePoint, display the site workflows for the site in which the Credit Extension workflow was published.
3. In the Workflows page, click the link for the Credit Extension workflow to display the custom start form for that workflow.
4. Right-click any point on the Start Workflow Credit Extension page, and select **Properties**.
5. Copy the contents of the **Address (URL)** property to the clipboard.

You now have a fully resolved URL with which to initiate the workflow, to which Microsoft Dynamics CRM can add the necessary query string parameters. For more information about adding the query string parameters, see "Initiate the workflow from Microsoft Dynamics CRM" below.

Initiate the workflow from Microsoft Dynamics CRM

After publishing the workflow and obtaining the fully resolved InitiateWorkflow.aspx URL needed to initiate the workflow from other programs, some way of initiating the workflow must be added to the other program, such as Microsoft Dynamics CRM. Also, the necessary query string parameters must be added to the URL, to pass information from Microsoft Dynamics CRM to the workflow and to control the appearance of the custom start form itself.

For the purposes of the integration sample, a custom ribbon button is added to Microsoft Dynamics CRM, specifically for accounts. The custom ribbon button adds query string parameters to the InitiateWorkflow.aspx URL needed to initiate the workflow, and then invokes the URL.

Only general directions for creating a custom ribbon button in Microsoft Dynamics CRM are provided. The details of creating custom solutions for Microsoft Dynamics CRM are beyond the scope of this documentation. However, several tools are available, such as Ribbon Workshop, which can make the task easier.

When creating the custom ribbon button, ensure that the following configuration steps are performed:

- Ensure that a command definition is created for the custom ribbon button, containing a Url action for which the following attributes must be set to the specified values:

Attribute name	Attribute value
Address	The InitiateWorkflow.aspx URL obtained from the previous procedure.
PassParams	0
WinMode	0

- Ensure that the command definition for the custom ribbon button includes a BoolParameter child element, named IsDlg, set to 1.

  The IsDlg query string parameter, when set to 1, sets the custom start form to a fixed size, removes the scroll bars, and otherwise works to render the form as a dialog box.

- Ensure that the command definition for the custom ribbon button includes a CrmParameter child element, named RecordID, set to the FirstPrimaryItemId field.

  The RecordID query string parameter, when set to the FirstPrimaryItemId field, provides the unique identifier for the current record to the custom start form.

Next steps

When clicked, the custom ribbon button added to Microsoft Dynamics CRM appends the IsDlg and RecordID query string parameters to the InitiateWorkflow.aspx URL, and then invokes the URL. The custom start form appears, as a dialog box, into which the Microsoft Dynamics CRM user can specify a credit limit for the current account. After the user clicks the Submit button on the custom start form, the workflow is initiated, using the specified values for the unique identifier for the current record and the credit limit.