

Workflow (Workflow type definition)

Introduction

Workflow outlines the stages and requirements needed to complete a business task from start to finish. Workflows are based on the operational flows of the organisation to which they belong.

Use a Workflow when you want people to enter data consistently, move through the same stages, and follow the same steps in a specific business process.

For example, use a Workflow if you want everyone to handle account address change requests in the same way.

Workflow type definition outlines the stages and necessary actions within those stages, guiding users towards achieving a desired business outcome. It's an Extension built on a Custom entity unique to that Workflow. (not applicable for external use, such as EOL Extensions).

Getting Started

[Exact Online Premium - Extensions | Getting Started with Extensions](#)

Overview of Workflow Definition Elements and Attributes

Attributes and details are explained below:

`<workflowdefinition>` - Definition of the Workflow (Workflow type definition).

`name` - Name of the Workflow. It should be prefixed with an Extension code.

`description` - Description of the Workflow.

`descriptionplural` - Description of the Workflow (e.g., the plural form of the Workflow that will be used as a caption in the List page).

`translationid` - unique id referring to the translation of the Workflow definition

`description` as defined in `<translationextensions>` element in the XML. It should be prefixed with an Extension code. Refer to [Extensions - Guidelines](#) for more details on custom translations.

`translationpluralid` - unique id referring to the translation of the Workflow definition `descriptionplural` as defined in

<translationextensions> element in the XML. It should be prefixed with the Extension code. Refer to [Extensions - Guidelines](#) for more details on custom translations.

Custom entity

<customentity> - Refer to [Custom entities](#) for more details.

- For Workflow Custom entities, the attribute `table` can be used to define a custom table name for your table; however, it must be prefixed with the Extension code. By default, the Workflow definition `name` is the table name.

Other attributes like a `name`, `description`, `descriptionplural`, `translationid`, `translationpluralid` are redundant for Workflow Custom entities, as these values are inherited from the level of <workflowdefinition> .

- isdescription** - Exactly one property must have `isdescription` set to true.

The following is an example of how you can define your Workflow Custom entity with a custom table name:

```
1 <workflowdefinitions>
2   <!--Address workflow-->
3   <workflowdefinition name="SDK_Wfl_AddressChange" description="Address change"
4     descriptionplural="Address changes">
5     <customentity table="SDK_Wfl_Address">
6       <property caption="Address description" name="Description" type="string"
7         length="255" allowempty="false" isdescription="true" />
8       <property caption="Postcode" name="Postcode" type="string" length="60" />
9       <property caption="Address" name="AddressLine1" type="string" length="60" />
10      <property caption="Workflow remarks" name="WorkflowRemarks" type="string"
11        length="1000" allowempty="true" />
12      <property caption="Workflow attachments" name="WorkflowAttachment"
13        type="attachment"/>
14    </customentity>
15  </workflowdefinition>
16</workflowdefinitions>
```

- Specifically for Workflow, the property attribute is expanded with type: `attachment` - creates a GUID property in the business component, which is related to the internal storage. This supports all the file types for documents.

In combination, the attribute `uploadmultiple` is optional for `attachment` `uploadmultiple` - specifies if multiple files can be applied to the attachment property.

It may contain either *true* or *false*. It is set to *true* by default, and the maximum uploads per attachment is 15. By specifying it to *false*, it will limit the uploads to 1 file.

```
1   <applicationextension pagetype="Maintenance" entity="SDK_WF_AddressChange">
2     <cardsection id="csAttachments" existing="false" caption="Attachments">
3       <field id="Attachment" datafield="WorkflowAttachment" type="attachment"
4         uploadmultiple="false" caption="Workflow attachments" datasource="bc" existing="false" />
5       </cardsection>
6     </applicationextension>
7   </applicationextensions>
```

Stages

<stages> - Element holds a collection of various stages within the Workflow.

<stage> - Definition of a stage, its property settings, actions and permissions.

name - Name of the stage.

caption - Caption for the stage.

translationid - unique id referring to the translation of the stage **caption** as defined in **<translationextensions>** element in the XML. It should be prefixed with an Extension code. Refer to [Extensions - Guidelines](#) for more details on custom translations.

defaultaction - name of the action that is defined in the current stage's actions, indicating which action should be treated as the default. It is optional, and if not defined, then the first available action is considered the default action.

stagetype - Specifies the type of the stage. Stage types are described in [Stage types](#).

<propertysettings> - This contains a collection of Workflow Custom entity properties.

<propertysetting> - Custom entity property used in the stage.

property - Name of the Custom entity property.

mandatory - Checks whether the property is mandatory or not.

visible - Specifies if the property is visible for the stage.

enabled - Specifies if the property is enabled for the stage.

i If not specified, the values are defaulted as: **mandatory = false, visible = true, enabled = true**.

Actions

<actions> - Defines actions available for the stage.

<action> - Defines the action available for the stage.

name - Specifies the name of the action from this stage. The action names must be unique within a Workflow definition.

caption - Caption of the action.

translationid - unique id referring to the translation of the action **caption** as defined in **<translationextensions>** element in the xml. It should be prefixed with an Extension code. Refer [Extensions - Guidelines](#) for more details on custom translations.

tostage - Specifies the name of the stage to execute the actions from this stage.

<permissions> - Specifies the users and their access for the action. Refer section [Workflow \(Workflow type definition\) | Stage and action permission](#) to define who can perform the action.

Automations

<automations> - Contains Automations definitions for the stage action.

<automation> - Definition of the Automation.

Description - Describes the Automation.

<conditionexpression> - Specifies the condition to execute the Automation actions.

<automationaction> - Definition of Automation action.

type - Specifies the Type of Automation action, e.g. Create, Update or Delete.

businesscomponent - Business component of the Automation action.

description - Description of the Automation action.

<properties> - Contains a list of Properties which are used while executing the Automation action.

<property> - Definition of the Property.

name - Name of the Business component property, which is specified for the Automation action.

valueexpression - Value expression assigned for the property

literalvalue - Literal value assigned for the property.

Skip

Skip element declares a conditional redirect from an action's default tostage to an alternative stage. At runtime, skips are evaluated in the order they appear. The first skip whose condition evaluates to true determines the stage transition. If no skip condition is met, the action transitions to its own **tostage**. First, the permissions are defined, followed by the skip element if desired, and concluded with applicable Automations.

<skip> - User can define multiple skip elements

tostage - The stage the Workflow will process to if the condition is met.

<condition> - The logical expression that determines whether the Workflow skips to **tostage**.

```
1 <action name="CreateProposal" tostage="ProposalCreated" caption="Create proposal"
2   translationid="SDK_EMP_CreateProposalActionTranslation">
3     <permissions>
4       <user involvementtype="Creator" />
5     </permissions>
6     <automations>....</automations> <!-- Specify automations if desired -->
7     <skip tostage="ProposalAccepted">
8       <condition>Amount < 20</condition>
9     </skip>
10    <skip tostage="IDCheckDone">
11      <condition>City = 'Delft'</condition>
12    </skip>
13  </action>
```

- i** In the above examples for the condition, if the Amount is lower than 20, the Workflow will process to stage “ProposalAccepted”, if the Amount is equal to or higher than 20, but the City is ‘Delft’, the Workflow will process to stage “IDCheckDone”. If the Amount is higher than 20, the City is ‘Eindhoven’, and the Workflow will process to stage “ProposalCreated”.

Stage and action permission

A user can perform actions like editing/deleting a Workflow instance in a given stage or move from one stage to another by defining any or all of the following elements, such as **user**, **actor**, **team**, **existingrole**, **extensionrole**, within the elements **<permissions>**, **<editpermissions>** and **<deletepermissions>**.

`<user>` - Defines allowed users.

`involvementtype` - Involvement types are defined as described in the section [Involvement types](#).

`property` - Name of the Custom entity property within the workflow to which `involvementtype` relate. e.g. `<user involvementtype="Employee" property="Employee" />`

`customentityproperty` - Name of the property of the Custom entity which is referred (`referstocustomentity`) in the property definition of the mentioned “`property`”. This is an optional attribute.

e.g. `<permissions>`

```
    <user involvementtype="user"  
    property="Product" customentityproperty="Manager" />  
  </permissions>
```

“`Product`” is the property of the Workflow Custom entity with “`referstocustomentity`” as the name of the referred Custom entity.

“`Manager`” is the property of the mentioned “`referstocustomentity`” which refers to “`User`” .

`<actor>` - Defines allowed users who were the actors of executing the previous action

`involvementtype` - Involvement types are defined as described in section [Actor Involvement types](#).

`action` - Name of the action that was performed in an earlier stage, e.g. `<actor involvementtype="User" action="Approve"/>`

`<team>` - Defines which team is allowed to perform operations on this stage.

`code` - Code of the team that has been created before. This is an optional attribute which can be used instead of the `property` attribute. e.g. `<team code="HRTeam" />`

`property` - Name of the Custom entity property within Workflow. This is an optional attribute which can be used instead of the `code` attribute, e.g. `<team property="TeamId" />`

`role` - Code of the team role that has been created before. This is an optional attribute if you want to specify a certain team role. e.g. `<team code="HRTeam" role="AdministratorRole" />`

`<existingrole>` - Defines which existing role is allowed to perform the operations on this stage.

`id` - Id of a standard Exact Online role. e.g. `<existingrole id="100" />`

`name` (optional) - Name of an existing role. e.g. `<existingrole id="100" Name="General right/">`. This is an optional attribute.

`<extensionrole>` - Defines which Extension role is allowed to perform the operations on this stage.

`name` - name of a Premium Extension role. e.g. `<extensionrole name="SDK_CarPark_CarParkManager" />`

Involvement types

- **Creator:** Defines that the creator of the Workflow instance is allowed to execute the actions. e.g. `<user involvementtype="Creator"/>`
- **AccountManager:** Defines that the account manager of the account is allowed to execute the actions. The property must be a valid account property of the current Workflow Custom entity. e.g. `<user involvementtype="AccountManager" property="Account"/>`
- **ProjectManager:** Defines that the project manager of the project is allowed to execute the actions. The property must be a valid project property of the current Workflow Custom entity. e.g. `<user involvementtype="ProjectManager" property="Project"/>`
- **Employee:** Defines that the employee is allowed to execute the actions. The property must be a valid Employee property of the current Workflow Custom entity. e.g. `<user involvementtype="Employee" property="Employee" />`
- **EmployeeManager:** Defines that the manager of the employee is allowed to execute the actions. The property must be a valid Employee property of the current Workflow Custom entity. e.g. `<user involvementtype="EmployeeManager" property="Employee"/>`
- **User:** Defines that the user is allowed to execute the actions. The property must be a valid User property of the current Workflow Custom entity. e.g. `<user`

```
involvementtype="User" property="UserId" />
```

- **UserManager**: Defines the manager of the user is allowed to execute the actions. The property must be a valid User property of the current Workflow Custom entity. e.g. e.g.

```
<user involvementtype="UserManager" property="UserId" />
```

Actor Involvement types

- **User**: Defines the User who performed the action.

e.g. `<actor involvementtype="User" action="Approve"/>` User who performed the action “**Approve**” is allowed to execute further action.

- **UserManager**: Defines the manager of the User who performed the action.

e.g. `<actor involvementtype="UserManager" action="Approve"/>` Manager of the User who performed action “**Approve**” is allowed to execute further action.

Editing a Workflow instance

To edit a Workflow instance in a given stage, the roles and permissions as defined in the section `<roles>` for the Custom entity are used to resolve the edit permissions. Anyone with update permissions on the Custom entity can edit a Workflow instance.

`<editpermissions>` - can be defined to limit performing an edit operation by only certain users on top of the roles as defined in the section `<roles>`. Defining this element is optional, and consider defining when you want to limit the users who can edit a Workflow instance.

`denyall` - used to restrict certain users or all the users from editing a Workflow instance, regardless of their roles. It is an optional attribute and set to false by default.

- When `denyall` is set to true, then no one is allowed to edit regardless of their granted roles. e.g. `<editpermissions denyall="true" />`
- When `denyall` is set to false (default), and the elements as mentioned in section [Workflow \(Workflow type definition\) | Stage and action permission](#) are defined, then the specified involvements in addition to the roles are considered to resolve the edit permissions.

The following code is an example of using `editpermissions` and `denyall` to restrict only certain users to edit a Workflow instance in a given stage.

```
1 <!--Only the creator can edit the Workflow instance-->
2 <editpermissions denyall="false">
3   <user involvementtype="Creator" />
4 </editpermissions>
```

Deleting a Workflow instance

To delete a Workflow instance in a given stage, the roles and permissions as defined in the section `<roles>` for the Custom entity are used to resolve the delete permissions. Anyone with delete permissions on the Custom entity can delete a Workflow instance.

`<deletepermissions>` - can be defined to limit performing a delete operation by only certain users on top of the roles as defined in the section `<roles>`. Defining this element is optional, and consider defining when you want to limit the users who can delete a Workflow instance.

`denyall` - used to restrict certain users or all the users from deleting a Workflow instance, regardless of their roles. It is an optional attribute and set to false by default.

- When `denyall` is set to true, then no one is allowed to delete, regardless of their granted roles. e.g. `<deletepermissions denyall="true" />`
- When `denyall` is set to false (default), and the elements as mentioned in the section [Workflow \(Workflow type definition\) | Stage and action permission](#) are defined, then the specified involvements, in addition to the roles, are considered to resolve the delete permissions.

The following code is an example of using `deletepermissions` and `denyall` to restrict only certain users from deleting a Workflow instance in a given stage.

```
1 <!--Only the creator can delete the Workflow instance-->
2 <deletepermissions denyall="false">
3   <user involvementtype="Creator" />
4 </deletepermissions>
```

Roles

Roles allow customers to define their own permissions for Custom entities using existing roles or define their own Extension roles

More details regarding roles can be found at [Extension roles and standard roles for Custom entities](#).

Stage Types

Stage types help to identify the type of stage. This attribute is **not mandatory**. Currently, we support 3 stage types as follows:

1. **New:** The stage with stage type “New“ indicates the start of the Workflow. As the stage type is not mandatory, if there is no stage with stagetype “New“, then the system creates a stage with stagetype “New“ internally, and the action of this stage (tostage) directs to the

first defined stage in the Workflow. In the example below, the stage with the name “New“ is marked as a stagetype “New“. This means the Workflow starts from this stage.

```
1 <stage name="New" caption="New" stagetype="New">
2   <propertysettings>
3     <propertysetting property="Description" mandatory="true"/>
4     <propertysetting property="Account" mandatory="true"/>
5     <propertysetting property="City" mandatory="true"/>
6     <propertysetting property="AddressLine1" mandatory="true"/>
7   </propertysettings>
8   <actions>
9     <action name="Create" tostage="Requested" caption="Create"
10    translationid="SDK_PF_ActionCreate">
11      <!--Permission to perform action: Create-->
12      <permissions>
13        <user involvementtype="Creator"/>
14        <team code="HRTeam"/>
15      </permissions>
16    </action>
17  </actions>
18  <editpermissions>
19    <user involvementtype="Creator"/>
20    <team code="HRTeam"/>
21  </editpermissions>
22  <deletepermissions>
23    <user involvementtype="Creator"/>
24  </deletepermissions>
25 </stage>
```

2. **Restarted:** This stage type indicates the Workflow can be restarted from this stage. As per the example below, stage “Rejected“ is marked as a stage type “Restarted“. The “tostage” attribute under “Restarted“ stage type action should be similar to the “tostage” attribute under “New“ stage type action. In the below example, tostage is “Requested“ which should match with the tostage of the action under stage type “New“.

```
1 <stage name="Rejected" caption="Rejected" stagetype="Restarted">
2   <actions>
3     <action name="Resubmit" tostage="Requested" caption="Resubmit">
4       <permissions>
5         <user involvementtype="Creator" />
6         <team code="HRTeam" />
7       </permissions>
8     </action>
9   </actions>
10 </stage>
```

3. **Canceled:** This stage type indicates that the Workflow can be Canceled at a particular stage. No action can be performed once the Workflow is canceled, it means the Workflow is **stopped** at this stage. The example below defines a stage with a stage type “Canceled“.

```
1 <stage name="Canceled" caption="Canceled" stagetype="Canceled">
2   <editpermissions>
3     <user involvementtype="AccountManager" property="Account"/>
4   </editpermissions>
5 </stage>
```

- Known issue (Note): If the newly installed version of the Workflow Extension does not contain a stage with “Canceled” stage type or a fully processed stage (final stage), and currently there are Workflows with stages of “Canceled” stage type or fully processed stage, then those Workflows are not displayed on the Workflow overview page.

Workflow Pages

Since Workflow Custom entities support low-code solutions like Extensions, standard pages are automatically generated for each Workflow Custom entity definition.

You can access these pages by defining the `href` attribute of UI elements like - link(megamenu), button or monitoritem. The Custom entity page names and page URLs are tabulated as below.

Page type	Page name	Page URLs
List	<code>ExtWorkflows. aspx</code>	<code>ExtWorkflows.aspx?Entity=</code> <code><CustomEntityName> *</code>
Maintenance	New/Edit - <code>ExtWorkflow.a spx</code>	New - <code>ExtWorkflow.aspx?</code> <code>BCAction=0&Entity=</code> <code><CustomEntityName> *</code> Single row edit - <code>ExtWorkflow.aspx?</code> <code>Entity=<CustomEntityName>&ID=</code> <code><dID> *</code>

Reference to Workflow

Using attribute `referstocustomentity`, a property (a field, a column or a filter) can refer to a Workflow.

`referstocustomentity` - Name of the referencing Workflow that is defined in the current Extension file. Used with the GUID type.

You can also use the same attribute to refer to a Custom entity, as explained in [Custom Entities - Guidelines](#).

- Using `referstocustomentity`,
- EOL entities can refer to a Custom entity or a Workflow
 - Custom entities can refer to a Custom entity or a Workflow

- Workflows can refer to a Custom entity or a Workflow

Create Workflow as a Standard Generic Automation action

When an Extension with a Workflow definition is installed, an Automation action is created as a standard generic action to create a Workflow instance. These standard generic Automation actions on Workflow definitions can be used in other business entities too.

For instance, when an Extension with a workflow definition for Applicant onboarding is installed, then a corresponding Automation action (say **Create: Applicant onboarding**) is created as a standard generic action which can be used in any other business entity (say Account).

Workflow Pivot Reports

Pivot reports provide customers with powerful insights into their Workflows.

The Workflow pivot reports are built on the EOL pivot reporting framework.

Using this framework, customers can create their own pivot reports. It supports three report types:

- **List** – Aggregated reports on a single dimension, e.g. *number of bug reports per product*
- **Matrix** – Aggregated reports across two dimensions (rows and columns), e.g. *number of bug reports per product per creator*
- **Graph** – Visual charts such as column, bar, line, or pie charts, e.g. *number of bug reports per product*

By design, a Workflow pivot analysis report is based on a single Workflow type definition.

Samples for Workflow

Sample Extensions can be downloaded from:

Company name > Master data > Developer: Maintain Extensions (under section **Flex**) >
Developer resources (button) > **Download samples and XSD** (button) > **Download Workflow samples** (button)

API for Workflow

[API for Workflow](#)