

Automation Engine 22

Workflows

12 - 2022

Contents

- 1. Workflow Basics..... 5**
- 2. Building Workflows using the Workflow Editor..... 6**
 - 2.1. The Workflow Editor Window..... 6
 - 2.2. The Workflow Editor Menus..... 7
 - 2.3. The Workflow Editor Tool Bar..... 9
 - 2.4. Building a Workflow from Scratch..... 10
 - 2.5. Building a Workflow using Custom Tickets..... 12
 - 2.6. Configuring Output Statuses..... 14
 - 2.7. Visual Optimizations of a Workflow..... 15
 - 2.8. Subworkflows..... 18
 - 2.8.1. Building a Subworkflow..... 20
 - 2.9. Inserting Wait Points..... 21
 - 2.9.1. The Wait Task..... 21
 - 2.9.2. The Wait for Files Task..... 22
 - 2.9.3. The Wait for Product Status Task..... 23
 - 2.9.4. The Wait for Action (Checkpoint) Task..... 23
 - 2.10. Routing in Workflows..... 40
 - 2.10.1. Simple Processing Routing..... 40
 - 2.10.2. Routing in Subworkflows..... 40
 - 2.10.3. Routing via To Do List Action Items..... 41
 - 2.10.4. Automatic Routing..... 43
 - 2.11. Exchanging Workflows..... 44
 - 2.12. Good Practices for Building Workflows..... 44
 - 2.12.1. Avoid Absolute Paths and Server Names..... 44
 - 2.12.2. Annotate..... 46
 - 2.12.3. Use Referenced Tickets and Workflows..... 47
 - 2.12.4. Avoid excessive amounts of Tokens..... 48
- 3. Launching and Diagnosing Workflows..... 51**
 - 3.1. The Parameter Inspector..... 51
 - 3.2. Launching a Workflow on a File..... 52
 - 3.3. Using Favorite Workflow Tickets..... 54
 - 3.3.1. Adding Favorite Workflow Tickets..... 55
 - 3.3.2. Adding a Favorite based on a SmartName..... 56
 - 3.3.3. Launching Favorite Workflow Tickets..... 59
 - 3.4. Building and Launching a Workflow On The Fly..... 59
 - 3.5. General Workflow Task Options..... 60
 - 3.6. Relaunching Your Workflow with Different Settings..... 61
 - 3.7. Diagnosing a Workflow in the Tasks Pane..... 62

3.8. Diagnosing a Workflow in the Workflow Editor.....	63
3.9. Understanding Tokens (Grouping of Output Files).....	67
3.10. Checking the Resources your Workflow contains.....	70
3.11. Pausing or Cancelling Your Workflow.....	72
4. Workflow Controls.....	73
4.1. Launch Workflow.....	74
4.2. Modify Workflow Parameter Values.....	75
4.2.1. JSONPath Expressions.....	78
4.3. Router.....	79
4.4. Select File.....	85
4.5. Select Job.....	87
4.6. Select Product.....	88
4.7. Select Referenced File.....	89
4.8. Set Priority.....	92
4.9. Sort.....	93
4.10. Mark File and Select Marked File.....	96
4.11. Data Collector.....	97
4.11.1. Examples of the Challenges in Merging Workflow Parameters.....	99
4.12. Data Splitter.....	103
4.13. Fork and Join.....	105
5. Using Public Parameters in Workflows.....	107
5.1. Concept.....	107
5.2. Creating Public Parameters in a Workflow.....	109
5.3. Conditional Availability of Public Parameters.....	110
5.4. Managing Public Parameters in a Workflow.....	116
5.5. Making Presets to Simplify the Users' Choices.....	117
5.6. Example of Further Customizing the Public Parameters Dialog.....	120
5.6.1. Step 1 - Example Workflow and its Initial Public Parameters Dialog.....	120
5.6.2. Step 2 - Customizing the Groups and the Text that the User will See.....	122
5.6.3. Step 3 - Adding a Preset and Combining Parameters.....	123
5.7. Launching a Workflow with Public Parameters.....	124
6. Using Workflow Parameters.....	129
6.1. Concept.....	129
6.2. Most Typical Use Cases for Using Workflow Parameters.....	131
6.3. Creating Workflow Parameters Manually.....	133
6.4. Loading Workflow Parameters from XML.....	134
6.5. Using Workflow Parameters.....	135
6.6. Using Workflow Parameter Values from XML.....	135
6.7. Changing the Value of a Workflow Parameter.....	137
6.8. Modifying Workflow Parameters During a Workflow.....	139
7. Automated Launching of Workflows.....	141

8. Migrating Old Task Chains.....142

1. Workflow Basics

Automation Engine enables you to create very powerful and automated workflows. You can customize them to do exactly what makes sense in your environment. You can make them extra smart by using SmartNames, subworkflows, routing, integration points with external systems and much more.

A step in a workflow is an Automation Engine **Task** performing an action on an input file to produce an output file. It can be also be a **Workflow Control**. Workflow controls can be about deciding which route the workflow will take or about file management (selecting, sorting etc.) or about other administrative decisions in the workflow.

Here is an example introducing some main tools:



1. A **Transition** is the green line that connects each step in the workflow. It is a graphical representation of an output file of a step being used as the input for the next step.
2. By default each step has an OK and Error **Output Pin**. When a step ends in the OK state, the output file will be sent to the step connected to the OK output pin. When a step errors out the workflow will continue with the step connected with the Error output pin. In the example workflow above the **Preflight with PitStop** task has 2 extra pins, matching the possible Preflight statuses.
3. This **subworkflow** handles the approval process. This subworkflow opens when you double click it.
4. Here we used the workflow control **Router**.
5. See how this Router shows 3 output pins. When hovering over them with your mouse you see their name, in this case the route to 'Off site printing'.



Workflows can be started

- manually, by an Automation Engine user,
 - using the Pilot or the [browser client](#)
 - using the Shuttle client tool or the Shuttle plug-in on board Esko editors or Adobe Illustrator.
- automatically
 - by an Access Point
 - by an external system that was integrated with Automation Engine. This could be WebCenter or any non-Esko system.



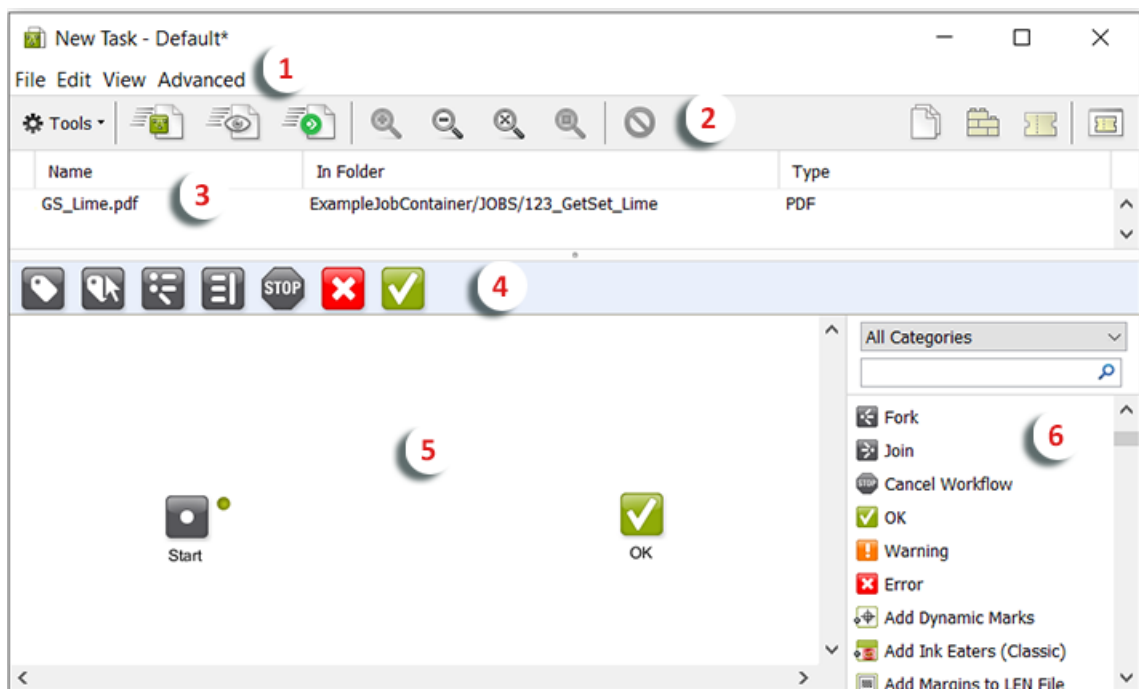
Note: Learn more about Access Points and integrations in [Integrating with External Systems](#).

2. Building Workflows using the Workflow Editor

You can open the workflow editor from the Pilot using  (from the **Tickets** view) or  (from the **Files** view).

2.1. The Workflow Editor Window

The workflow editor window consists of these main areas:



1. The workflow editor menus. Learn more in [The Workflow Editor Menus](#) on page 7.
2. The workflow editor tool bar. Learn more in [The Workflow Editor Tool Bar](#) on page 9.
3. The **Files** pane shows the files that you selected before opening this workflow (only when opened from the **Files** view) or it shows the files that you launched this workflow on. Enable or disable it via the menu **View > Hide / Show Files**.
4. The **Favorites** bar shows workflow steps that you often use when building workflows. You can then drag them from this bar down onto the canvas. Enable or disable this bar via the menu **View > Hide / Show Favorites**.

Add new favorites by drag and dropping new steps or custom tickets into this bar. Remove them via a right-click and click 'remove'.

5. The **canvas** shows your workflow. This is where you build it. You can monitor your workflow while it is processing files and you can diagnose a finished workflow. You can also print it (see [menu options](#)).
6. The **Tasks** pane lists all the types of steps that you can add to this workflow (steps can be task tickets and workflow controls). Enable or disable it via the menu **View > Hide / Show Tasks**.

To add a step to the workflow, simply drag and drop it onto the canvas.

You can easily find types of tasks by selecting a category and or using the filter and search field.

2.2. The Workflow Editor Menus

Some menu options are only available while creating a workflow, others are only available for a saved workflow ticket.



Note: Check the available menu shortcuts.








Use this menu...	to ...
File	<ul style="list-style-type: none"> • Launch: To launch this workflow on the selected input file(s)(available when you selected a file before building the workflow). The workflow editor window will close. • Launch And Monitor: To launch this workflow on the selected input file(s) and keep the workflow editor window open so you can monitor its progress. • Launch with Public Parameters...: Select this when you prefer to use a launch dialog that only shows the public parameters of this workflow (in stead of all steps and all their parameters). When working with public parameters, an administrator that set them up and created the custom dialog can so get a preview of how other users will see this simpler interface. Learn about public parameters in Using Public Parameters in Workflows on page 107. • Select File...: Select one or multiple files to launch in the workflow. • Open: Opens the selected step (alternative to double-clicking it). • Save Ticket: Saves your workflow and all its parameters in a (workflow) ticket. • Save Ticket as...: Saves your workflow under a different (ticket) name. • Export Ticket...: Saves your workflow on your client computer. Learn more about exporting workflows in Exchanging Workflows on page 44. • Show Info: Shows information about the workflow ticket: its Name, the Task Type, the Tags, whether or not the ticket is Public, a Description of the ticket. • Show Parameter Inspector: Use this tool to easily manage the ticket parameters of all the used tasks. Learn more in The Parameter Inspector on page 51. • Find: Puts your cursor in the search field of the tasks pane and clears any previously entered text string.

Use this menu...	to ...
	<ul style="list-style-type: none"> • Search: Puts your cursor in the search field of the tasks pane or highlights any previously entered text string. • Close: Closes this workflow editor window. • Print: Print your workflow canvas on a network printer. This function is only available when opening a saved workflow ticket from the Tickets view.
Edit	<ul style="list-style-type: none"> • Undo or Redo your latest changes. The menu option also mentions which type of change it can undo. For example: 'Undo Change Parameters' or 'Undo move' etc. • Cut, Copy or Paste the selected step(s). You can also copy-paste between multiple workflow editor windows. You can define where the copy will be placed by using the right-click function. • Select All workflow steps. • Delete the selected step.
View	<ul style="list-style-type: none"> • Change the zoom of the workflow to its Actual Size. Alternatively, you can also Zoom In, Zoom Out, or Zoom to Fit in the window. Check the available shortcuts. • Show / Hide Favorites, Files, Tasks and Sticky Notes. Learn more about sticky notes here. • Show / Hide Grid shows or hides the canvas grid. Activate Snap to Grid to more easily align your workflow steps on the canvas. • Expand or Collapse the Queue. When a workflow is running, a Queue is the vertical list of icons of workflow items that are being processed by a same step. These items represent (grouped) files or concepts like Job, Plate etc. If the amount of these icons is disturbing, choose to collapse that queue.
Advanced	<ul style="list-style-type: none"> • Add / Remove this workflow ticket to / from your Favorites bar. • See a processing Log (when the workflow is running). You can also send the Log contents as an E-mail. • Set up Task Options: Priority, Hold / Release, Launching a separate task per input file and Annotations. • Set up Notification Rules. Learn more in Creating a Notification Rule. Tip: check if using the separate Send E-mail task is a better alternative. • Manage Workflow Parameters... and Workflow Parameter Values.... Learn more about these in Using Workflow Parameters on page 129. • Manage Public Parameters for the selected step or check and Modify Public Parameter Values for the whole workflow. Learn more in Using Public Parameters in Workflows on page 107.

Use this menu...	to ...
	<ul style="list-style-type: none"> • Validate the current workflow (including all subworkflows). Any problems will be shown and explained in a separate dialog.

2.3. The Workflow Editor Tool Bar






The tool bar offers buttons to:

- Perform specific actions on the workflow ticket:  **Tools**. All available actions are explained in [The Workflow Editor Menus](#) on page 7.
-  **Launch** the workflow on the selected files and close the workflow editor (only available when you opened the workflow editor from the **Files** view).
-  **Launch and Monitor** the workflow (without closing the workflow editor).
-  **Launch using Public Parameters**: Select this when you prefer to use a launch dialog that only shows the public parameters of this workflow (in stead of all steps and all their parameters). When working with public parameters, an administrator that set them up and created the custom dialog can so get a preview of how other users will see this simpler interface. Learn about public parameters in [Using Public Parameters in Workflows](#) on page 107.
- Change the zoom in the workflow canvas (zoom in , zoom out , zoom to fit )



Tip:

You can also use these shortcuts:

- **Ctrl + Space** (on Windows) or **Space + Cmd** (on Mac) to activate the zoom in tool, then click or drag in the window.
- **Ctrl + Alt + Space** (on Windows) or **Cmd + Option + Space** (on Mac) to activate the zoom out tool, then click in the window.
- Scroll your mouse up or down to move the whole workflow up / down in the window. Add **Shift** to move left-right.
-  Delete the selected step(s) from the open workflow.
- Show / hide panes (areas within the window):
 - : The input **Files** (when opened from the **Files** or **Tasks** view).
 - : The [Resources this workflow contains](#).
 - : **Tasks**: the types of tasks and workflow controls.
 - : **Ticket Browser** shows available tickets in a separate dialog.



Tip: Use **Ctrl + click** or click and drag to select multiple steps.

2.4. Building a Workflow from Scratch

1. In the **Tickets** view, click  to open the workflow editor.



The canvas shows these two workflow controls by default: **Start** and **OK**.

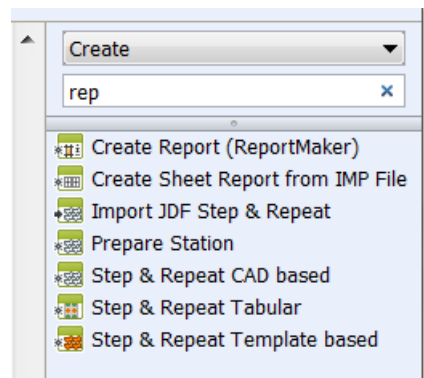
2. Drag and drop the desired steps from the tasks pane onto the canvas.



Note: If the tasks pane is not visible, click **View > Show Tasks**.

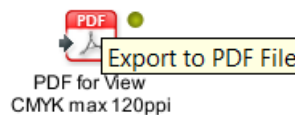


Note: To find a task quickly, you can use the filter-by-category and a search field.



3. If you want to give one or more steps a more descriptive name to make the workflow self documenting, double-click the name underneath the icon and change it.

You can still see the original name of the step by hovering over the step.



4. Connect your steps with one another. To do this, click on a step's green output **Pin** and drag it onto the next step.

This will link the two steps with a green arrow, called a **Transition**.

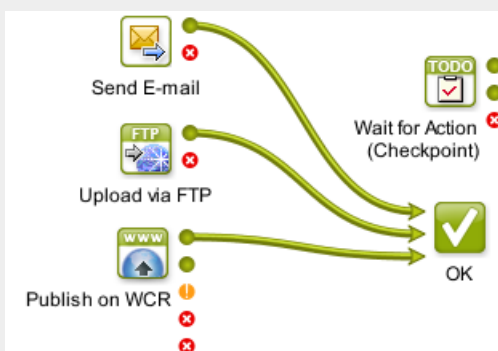


Note: Remove the transition by selecting the end part (arrow point) and dragging it away from the step that it was connected to.

Note: You can connect a step's output to multiple other steps. These steps will receive the same input file and will execute in parallel. An example:



Note: When you move your mouse to a point where multiple transitions come together, they will split up to make it easier for you to pick one of them (see below). When you want to move them together, hold the **Alt** button to select and move them together.



Tip: Learn more about keeping your workflow clean in [Visual Optimizations of a Workflow](#) on page 15.

5. Double-click each step and fill in its ticket parameters. Click **OK** to close the ticket. There is no need to save changes to each step ; their settings will be saved inside the workflow ticket.
6. Save your workflow using either:
 - **File > Save** or **Ctrl + S**,
 - **File > Save As...** or **Ctrl + Shift + S**.

Note: Until your workflow is saved, you can undo your changes. In **Edit > Undo** you can read which change can be undone.



Note: Until you save your changes, the name of the workflow will show an * behind it's name.





Tip: Before saving your workflow, it is good practice to check its validity. Use **Advanced > Validate Workflow** or **Ctrl + Alt + V**.

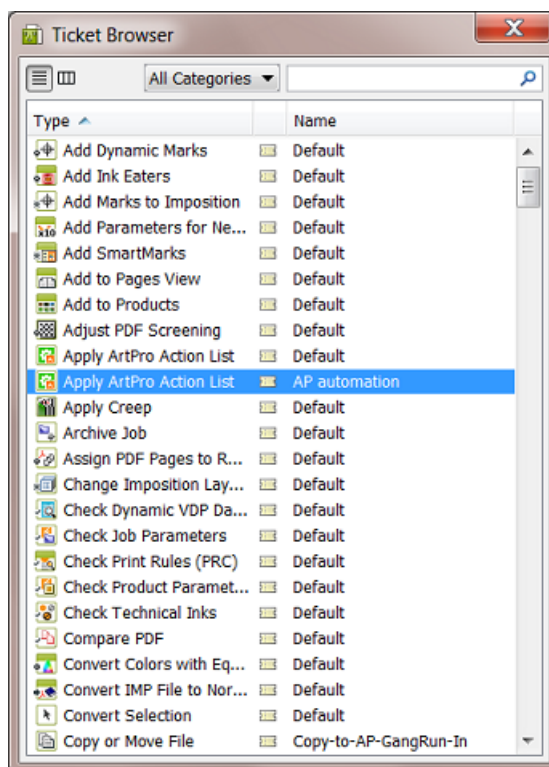
2.5. Building a Workflow using Custom Tickets

Compared to *building a workflow from zero*, using already made custom tickets as workflow steps adds these extra possibilities:

Using the Ticket Browser

In the tool bar of the workflow editor, click  to open the **Ticket Browser**, a pop-up dialog.

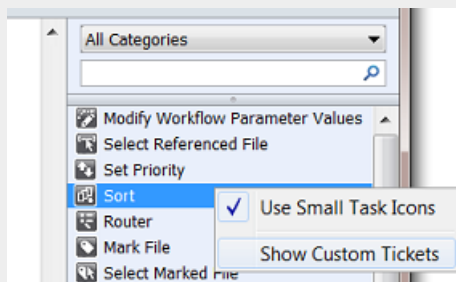
 Shows the complete list of tickets, sorted per task type.  Shows only the tickets of the selected task type.



1. Use the **Categories** filter and /or the search field to quickly find your custom ticket.
2. Select and drag the ticket onto your workflow canvas.



Note: Alternatively, you can choose to not use this **Ticket Browser**, but use the **Tasks** pane, right-click in the list and select **Show Custom Tickets**.



Custom Tickets in your Workflow can be Copies or References

- **Copy:** When you add a custom ticket onto your workflow canvas, it is by default a *copy* of that ticket. This means that any changes you make to that ticket will *not* be reflected in the original ticket and vice versa.
- **Reference:** When you hold **Ctrl + Shift** or **Cmd + Alt** (Mac only) *while* you drag a custom ticket onto your workflow canvas, you add *reference* to that ticket to your workflow. This means that any changes you make to the ticket will also be done in the original ticket and vice versa.



Note: This is equally valid when your ticket is a workflow ticket.

Referenced tickets also show their original name, with a small arrow in front. Below example shows 3 workflow steps: a copied ticket on the left, a referenced one in the middle and on the right a referenced one that was renamed in this workflow but whose original name is still shown in grey below, next to the small arrow.



Note: If the ticket that you refer to in your workflow uses any SmartName containing [... originating Input ...], be aware that this will reflect to the input of the ticket that you refer to, not of the workflow that refers to it.

Adding a Part of another Workflow to your Workflow

If you want to add parts of a workflow to another workflow, follow these steps:

1. Open both workflows in a workflow editor.
2. Select a part of one workflow and choose **Copy**.
3. Click on the canvas of the other workflow and choose **Paste**.
4. Add the desired transitions and save your new workflow.



Note: If you plan to reuse workflows or parts of a workflow several times, we advise to create subworkflows. Learn more in [Subworkflows](#) on page 18.

2.6. Configuring Output Statuses

For each workflow task step, you can customize what to do when the task results in warnings or errors. You can also choose to add extra output pins.

Right-click the workflow step and select **Configure Outputs...**

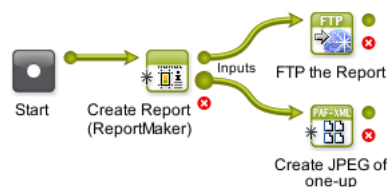


Note: The offered options and the precise words are different per type of task.

- **Warning handling**
 - Select **OK on Warning** to send the files to the **(Finished) OK** output pin.
 - Select **Error on Warning** to send them to the **(Finished with) Error(s)** output pin.
 - Select **Dedicated handling of Warning** to send them to the **(Finished with) Warning(s)** output pin.

- **Add an extra output pin where the task's inputs will be (when no errors occur).** This is useful when the output file of your task is different from the input file and when you (also) want to continue working on the input file. Typical examples are when you create a report PDF or a Step & Repeat and you (also) want to continue launching steps on the one-up.

See this example, where the 2nd green output pin 'Inputs' allows further steps on the one-up:



- **Add an extra output pin where the task's reports will be put.** Tasks that also output a separate report file (like the Preflight tasks) offer this extra choice. You can then connect that extra 'Reports' pin to the workflow step that will handle that report file.

See this example where the Preflight report goes to its own output pin 'Reports':

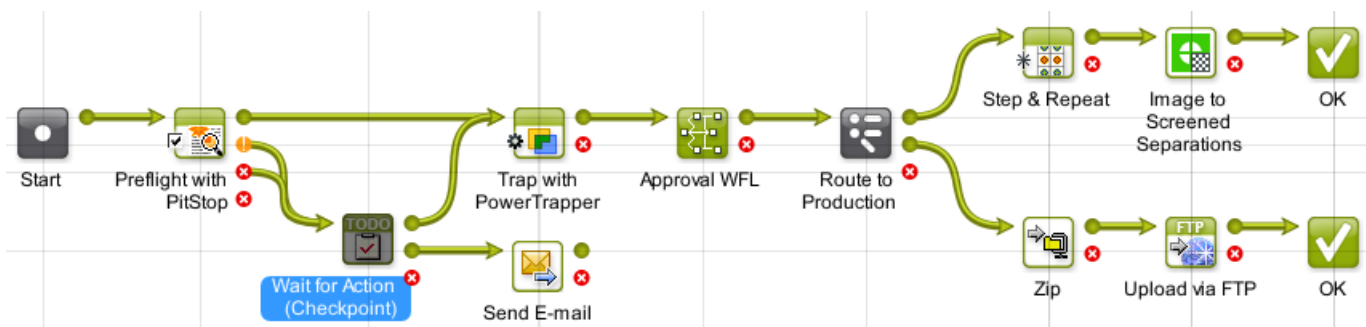


2.7. Visual Optimizations of a Workflow

Workflows can become very complex. The following tools help to visually optimize them, which will make it easier to understand them.

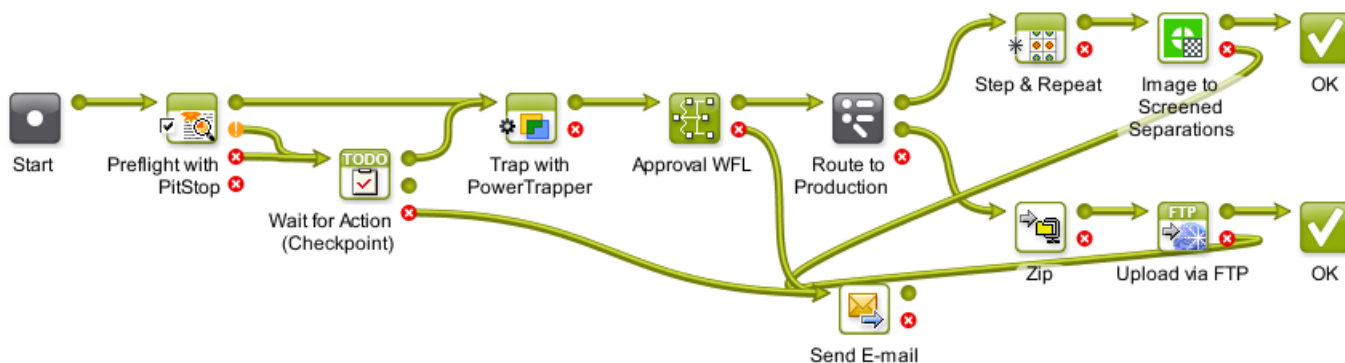
Aligning Workflow Steps on a Grid

You can align your workflow steps on a grid. To do this, hold the **Shift** key while dragging a step to the desired position. It will snap to a place on the grid.



Hiding and Showing Transitions

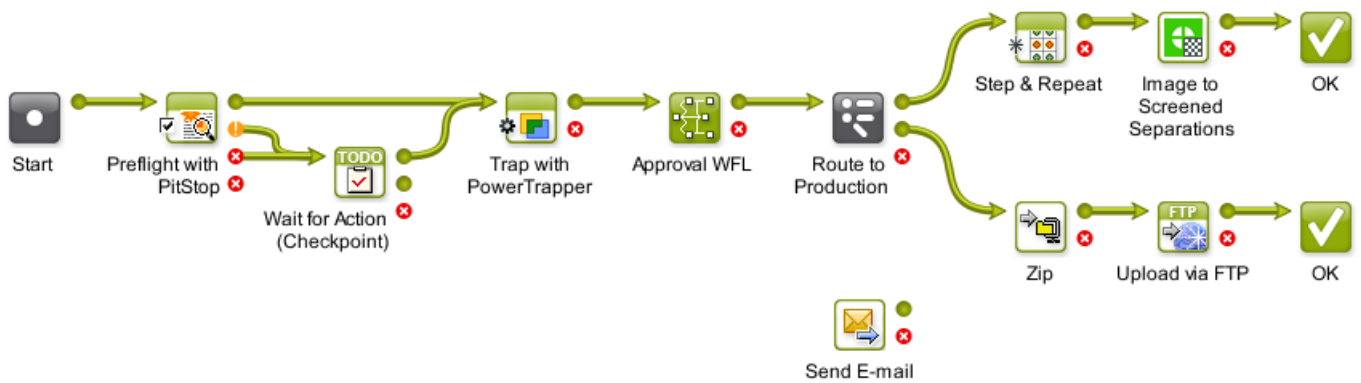
To optimize the overall look and readability of a workflow, you can hide specific transitions. See this example where the transitions to one **Send E-mail** step make the workflow look more complicated than it is:



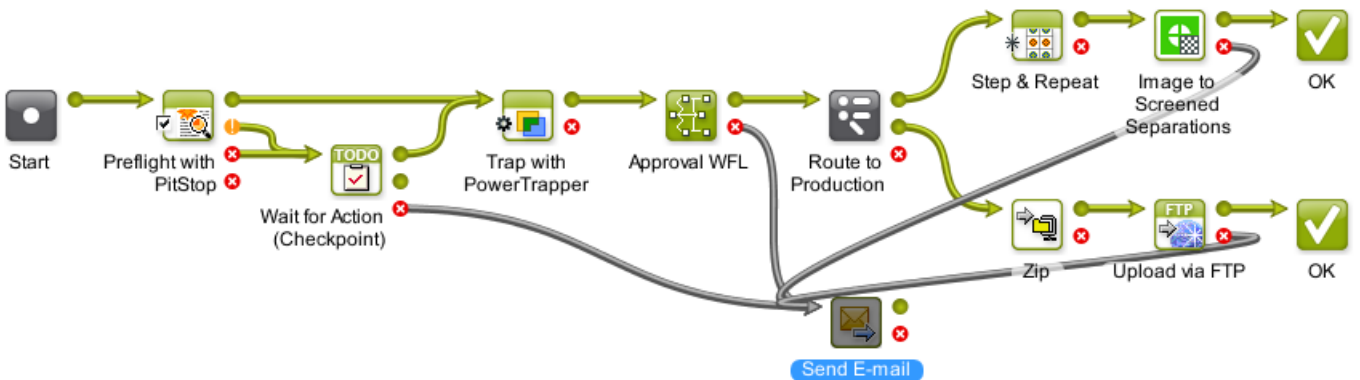
To hide the transitions to a (less important) step:

1. Right-click on the desired workflow step.
2. Click **Hide Transitions**. All transitions going to this workflow step will now be hidden. All other new transitions made to this step will automatically be hidden as well.

See this example where all transitions to the **Send E-mail** step are hidden:



If you want to see which transitions were hidden, click on the step (start or end). The hidden transitions will be revealed in a grey color:



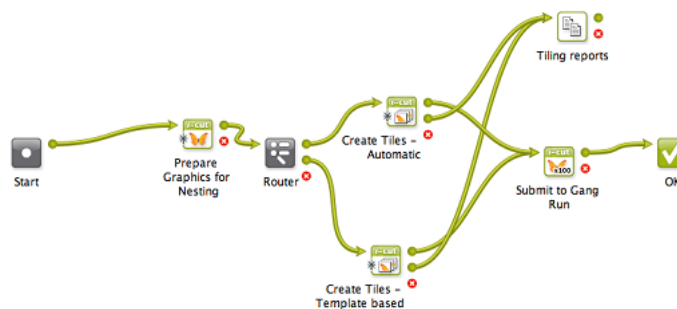
To reveal the transitions again, right-click the step and choose **Show Transitions**.

Distributing the Space between Workflow Steps

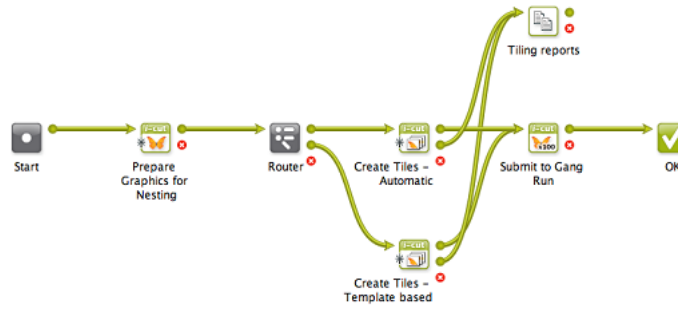
To distribute the space between all (or selected) steps evenly and also to align them on a grid, proceed as follows:

1. Right-click anywhere in the workflow canvas.
2. To align the steps horizontally, click **Horizontal Distribute Space** or use **Ctrl + Shift + H**.
3. To align the steps vertically, click **Vertical Distribute Space** or use **Ctrl + Shift + V**.

For example, this workflow needs some realignment:



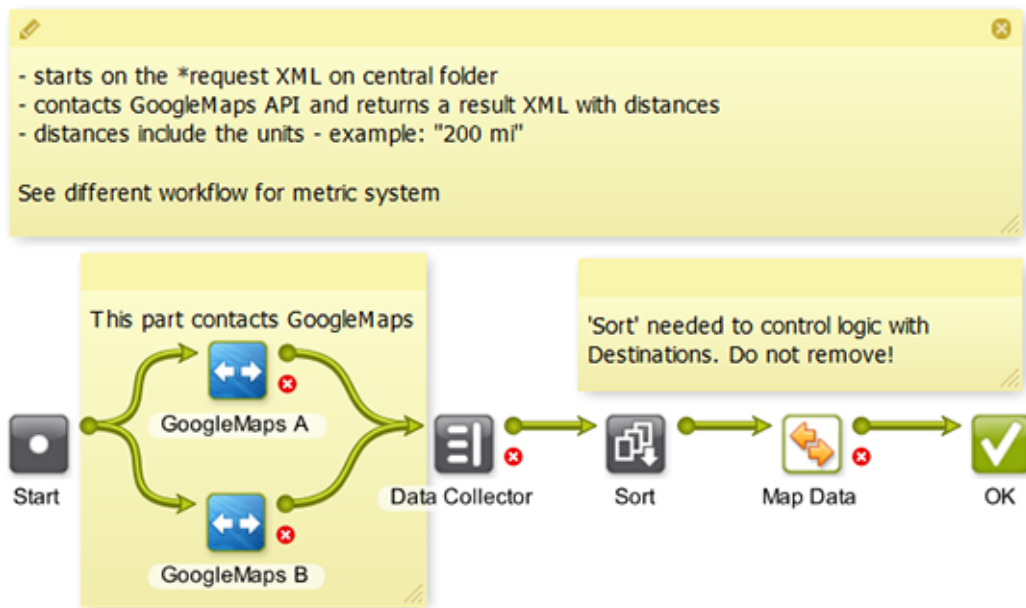
This is the result after distributing the space:



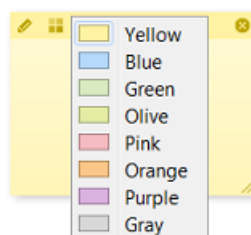
Adding Sticky Notes

Right-click anywhere on your canvas to add a **Sticky Note** to your workflow. Use these notes to add background information and so help users understand the workflow. They can serve as warnings or to remind users why some parts are constructed that way.

An example:



Click the icon on the top bar to choose a color:





Tip: To move the note, click the top bar and drag it to the new location.

Customizing the Canvas Background

You can choose a custom canvas background for all your workflows. Right-click on the canvas, choose **Change Canvas Background...** and browse to an image on your local computer.

Choose **Clear Canvas Background** to remove it again.

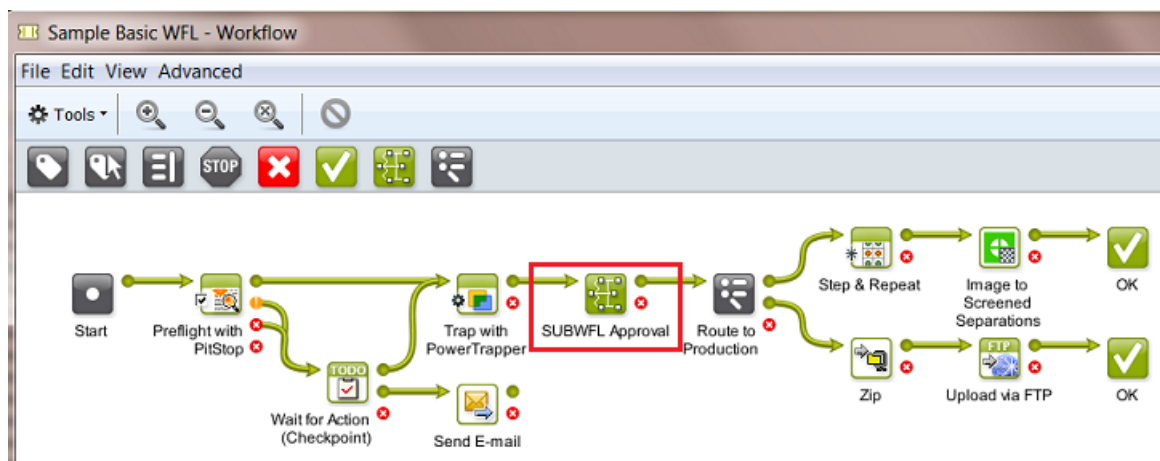
2.8. Subworkflows

A subworkflow is a workflow that is part of its master workflow. It is included in the workflow ticket of the master workflow. A subworkflow can also include another subworkflow. You can create workflows several levels deep.

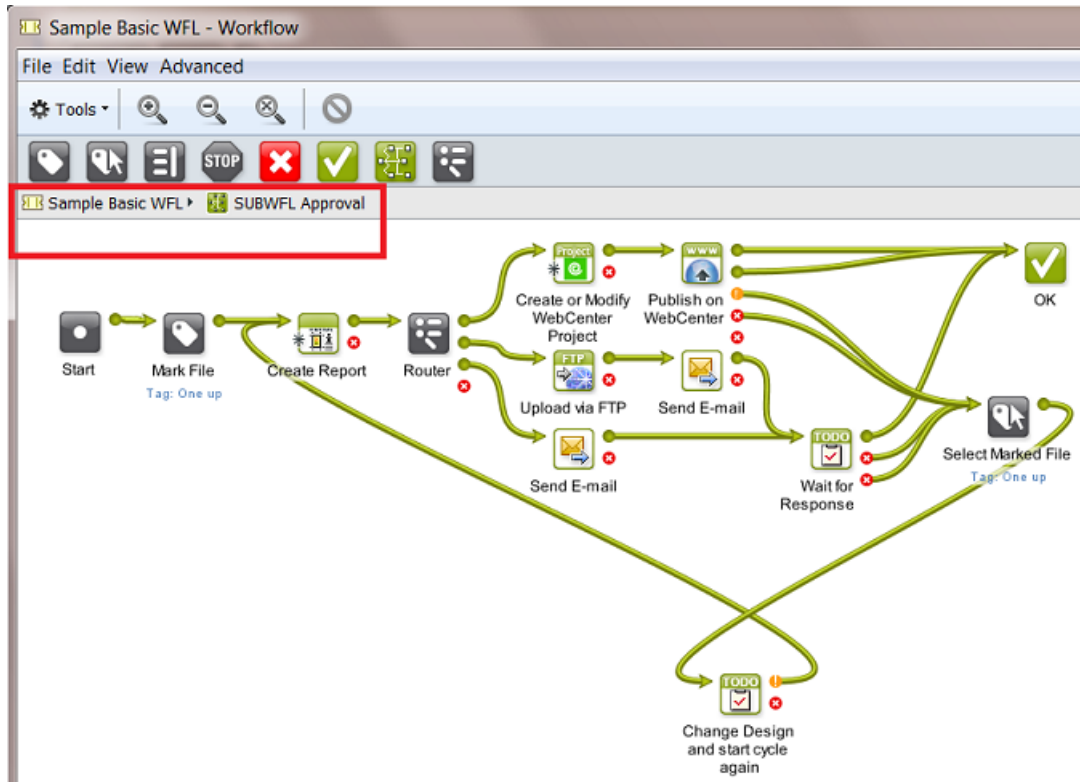
These are two main reasons to create a subworkflow:

- The master workflow will *look less complex* when a part of its flow is represented by only 1 workflow step.
- To stop creating the same (parts of) workflows again. If you notice that you keep creating the same or similar (parts of) workflows, then consider building that recurring part once and then copying it as a subworkflow into the other workflows.

In this example we see that the complete approval workflow is represented by a subworkflow named "SUBWFL Approval":



Double clicking the subworkflow step opens it in the same workflow window. Notice the indication of the path:



The Difference between a Subworkflow and a Referenced Workflow

A referenced workflow is not included in the master workflow, it is not copied on board. It is a link to a separate (workflow) ticket.

Note: Learn about adding referenced tickets or workflows in [Building a Workflow using Custom Tickets](#) on page 12.

The advantage of a referenced workflow is that by updating your referenced workflow you so automatically update all the master workflows that refer to it. See some examples in [Good Practices for Building Workflows](#) on page 44.


See this example where the approval part of the workflow is added as a reference. Notice the small arrow and extra name of the original workflow ticket:



When you double-click a referenced workflow step, it will open in a separate window.

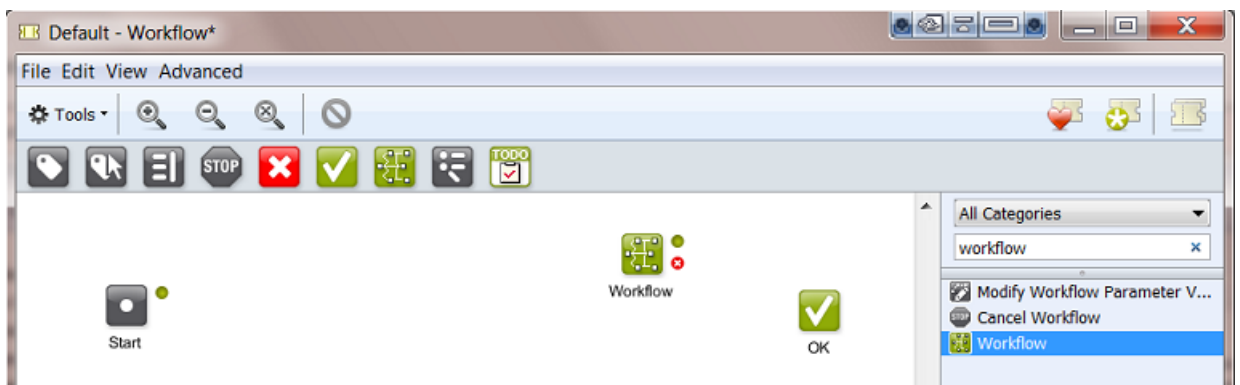
2.8.1. Building a Subworkflow

To add a subworkflow into your workflow:

1. In the workflow editor, drag a **Workflow**  step into your workflow. Or use an existing workflow ticket ; in that case skip step 3.

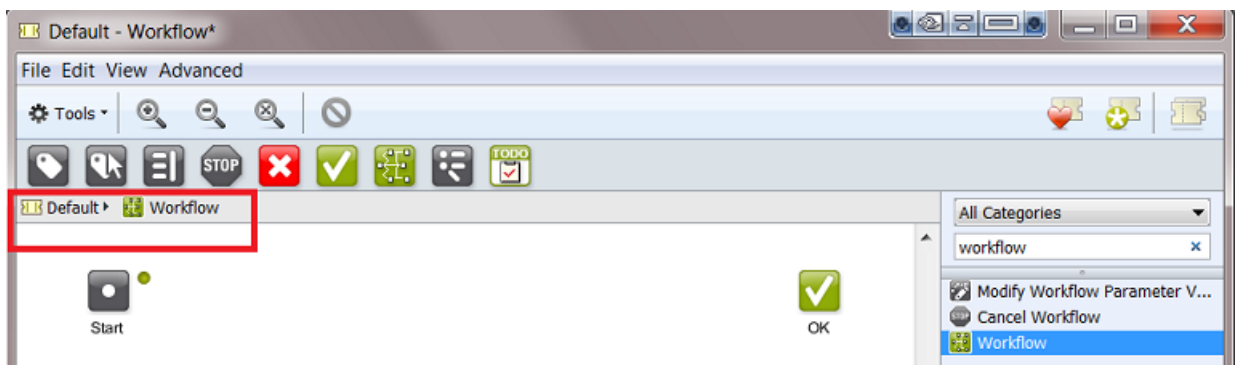





Tip: To find it quickly, select the category **Workflow** in the tasks pane or type 'workflow' in the search field.



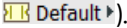
2. Double-click the workflow step.

The workflow step opens in a new canvas, where you can build your subworkflow. The extra level in the path indicates that this is a subworkflow:



Note: This is an example of a path indication of a workflow with 2 levels of subworkflows:  Flat Display WFL >  One Up Approval >  WebCenter approval

3. Build your subworkflow: Drag and drop steps into its canvas, add transitions, open each step to change its parameters. There is no need to save at this point.

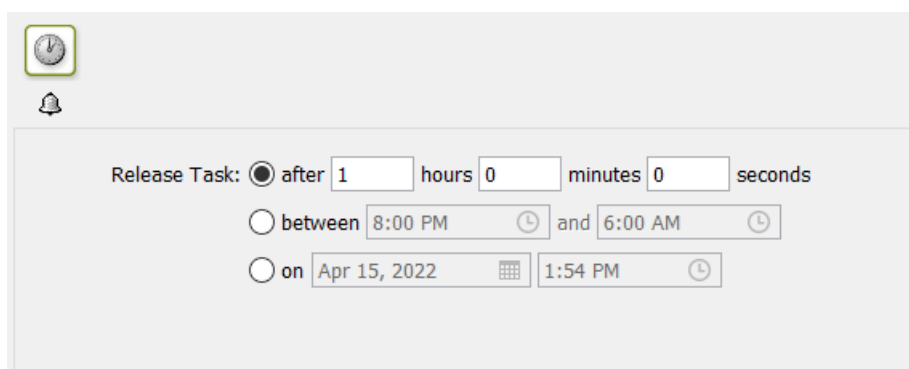
4. In the workflow path, click on your master workflow to go back to it (in above example: click on  Default ▾).
5. Save your master workflow. Your subworkflow(s) will be saved as a part of it.

2.9. Inserting Wait Points

Sometimes you want the workflow to wait a while before continuing with the next task. There are several ways to do this:

2.9.1. The Wait Task

Use the **Wait** task in a workflow to postpone the next task to a relative or absolute time.



A few examples of workflows where this task is useful:

- Your workflow sends a proof to a proofing device. You want to wait 30 minutes and then send the user a *To-Do* to go and pick up the proof.



- A **Folder Access Point** launches a workflow on incoming design files. These design files have been backed up already somewhere else. That is why you can delete them from the Automation Engine Container when you are sure that the workflow no longer needs them (for example after 4 hours).



- Every night you want to clean up a specific type of files that you consider temporary. You have built this into your workflows, where a **Wait** task is set to release between 3 and 5 AM, and is followed by a **Delete** task.

2.9.2. The Wait for Files Task

Use this task when you want to process files that are not present yet.



Note: Instead of using this task as a step in a workflow, consider if using an [Access Point](#) is not a better way to wait for files (and then start a workflow on those files).

This task looks for files at a regular interval, and when it detects them, it passes them to the next workflow step.

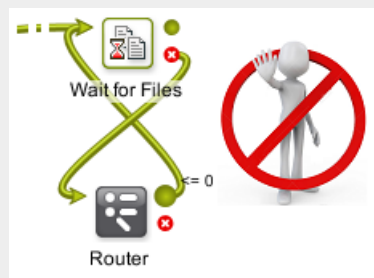
You can define which files it should look for, how long the task should wait for them to appear and what to do if the files are not found after that time period.



Warning: In a workflow, this kind of type of task can put some strain on the workflow engine, especially when used a lot. So if your case can be solved by using an [Access Point](#), than we strongly advise to do so.

In any case, do not create workflows that use this task in a loop! For example: A Wait for Files task that waits for any specified period and, when it doesn't detect the files by then, the workflow then loops back to that same Wait for Files step, which then waits some more etc. Such constructions create a massive amount of [tokens](#) and this blows up the memory consumed by the workflow engine.

An example of what not to do: When the Wait for Files step was unsuccessful, a Router step does a check of the number of files in that output pin and if (for example) it is 0 or less, the workflow goes back to the same Wait for Files step. Such a setup is a guarantee for major problems with the workflow engine of your Automation Engine!




Important: In many cases, a better solution is to let the workflow fail and use other workflow steps or tools to continue your logic (To-Do action, notification, etc.).

1. Select the **File Type** of the file(s) you want to wait for.

You can either:

- choose to wait for **Any** file type,
- select a specific file type from the list,
- select several file types (see below).

To select one or more file types:

- a) Choose **Select File Types...** in the **File Type** list.
This opens the **Select File Types** dialog.
- b) In this dialog, select the files types you want from the **Available File Types** column.
- c) Click  to send them to the **Selected File Types** column.
- d) Click **OK** when you are done. You will see the selected file types listed in parentheses.

2. Use the **File Name Matches** parameter to search on file names.

You can use wildcards, SmartNames or a [Regular Expression](#). You can also browse to a file with a name close to what you need, and add wildcards and/or SmartNames to that example name.

3. Define where to look for the files in **Look in Folder**.

Choose to **Look in subfolders** or not.

4. Define when the task may **Stop waiting** for the specified files:

1. When a certain number of **Files** or **Pages** are collected,
2. After a certain time if the files haven't been found (when **time exceeds ...**). To set a custom time, select **other...** from the dropdown list and define a custom date and time.

When you selected this option, this extra option becomes available:

- **Task ends in error when not all files were found.** Select it if you want the task to finish with an **Error** state when it has not yet detected all files at the time that you set in **Stop waiting when time exceeds**

Depending on what you choose, the task will finish either after it has found all the files/pages, or after a specified time if it can not detect all the files/pages.



Note: If you do not give the task a time limit, and it can not detect all the files, it will run indefinitely.

2.9.3. The Wait for Product Status Task

This task allows to hold a workflow until one or more **Products** have reached a certain status.

This task is documented in the chapter on [Products](#). Find a direct link [here](#).

2.9.4. The Wait for Action (Checkpoint) Task

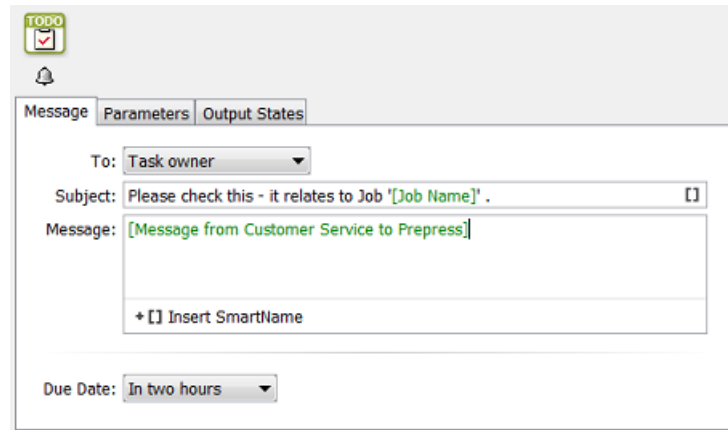
Concept

Use this task to hold your automated workflow for any form of user intervention.

This task adds a **'Wait for Action'** item in the [To Do List](#).

When a user checks and completes that required action, he 'releases' that item back to the workflow. He can choose to let the workflow continue, to abort it or to have it take another route in the rest of the workflow.

Tab 'Message'



You here define what you want to communicate to which user.

- **To:** From the drop-down list, choose a specific user or the **Task owner** (the user who launched the task) or choose **any user** (all users then get the to-do item in their list).

If you use a SmartName to specify the user, make sure it resolves to the method "name@domain" (for example "franky@aeserver01" or "franky@fruitco-domain").



Note: Also the choices **any user** and **Task owner** can also be accessed through a SmartName. This SmartName needs to resolve to their internal **Parameter Value** "Task Owner...@" and "Any User...@". Learn more about this in [SmartNames of Parameter Values](#).

- **Subject:** This subject line is shown in the To Do List overview.
- **Message:** Optionally, add extra information.
- **Due Date:** Setting a due date becomes useful when you want to sort of create filters in the To Do List overview (learn more in [The To Do List View](#)). Choose from the drop-down list to set a relative or an absolute time ('On').

Tab 'Parameters'

Concept

A to-do item that was created by the **Wait for Action** task can be extended with additional parameters.

There are 3 types of parameters that you can add:

- **Workflow Parameters** make it possible to change the value of a workflow parameter in a to-do item. Learn more [here](#).
- **Info** parameters make it possible to show extra information in a structured way (name, value). Learn more [here](#).
- **Link** parameters allow to add hyperlinks to the to-do item. These can be links to an external web site, an E-mail address or even a task in the Tasks view. Learn more [here](#).

You can add several parameters of several types.

The order in which you add them in this tab is the order in which they will appear in the to-do item. To change this order, simply 'drag-and-drop' the parameter up or down.

Adding a Workflow Parameter to a To-Do

Settings



Note: Find some examples below.

- Click **Add**. The **Edit Parameter** dialog opens.
- **Type:** Select **Workflow Parameter**. Learn all about workflow parameters [here](#).
- **Workflow Parameter:** The name of the existing workflow parameter that you allow to be changed in this to-do.

Type in the name or click  to pick it from a list.

If the workflow parameter does not exist yet, you will be informed about this when closing this 'Edit Parameter' dialog. That message will allow you to create it or not (yet).

- **Style:** How should this workflow parameter be presented in the to-do item?
 - As a **Text Field:** Fill in any text as the value for this workflow parameter.
 - **Placeholder Text:** This text is shown in the text field when the value of the workflow parameter is still empty.
 - As a **Combo Box:** The parameter is presented in the to-do as a combo box. The user, after clicking Edit, can pick a value from a list of predefined values.
 - **Placeholder Text:** This text is shown and selected in the combo box when the value of the workflow parameter is still empty.
 - **Values:** A comma separated list of values. Do not use spaces. These are the values from which the user can choose.



Note: A running workflow can decide another value than those in this list. When the value of the workflow parameter is not present in the list of values, and the value is not empty, the value will be added to the combo box and will be shown and selected in the to-do item.

- As a **Checkbox:** The parameter is presented in the to-do item as a checkbox. The user, after clicking Edit, can either select or deselect the checkbox.
 - **Checkbox Text:** The text that will be displayed to the right of the checkbox.
 - **Unselected Value:** The value that will be given to the workflow parameter when the checkbox is not selected.
 - **Selected Value:** The value that will be given to the workflow parameter when the checkbox is selected.

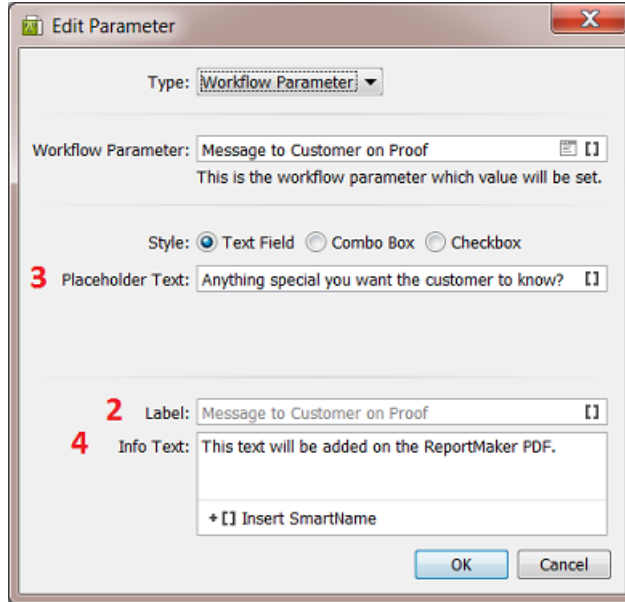


Note: In case the value of the workflow parameter is not equal to the unselected value nor to the selected value, then the value will be set to the unselected value.

- **Label:** In the to-do item, this text will appear in front of the parameter.
- **Info Text:** In the dialog **Edit Workflow Parameter**, this extra text can help to clarify this parameter or its effect.

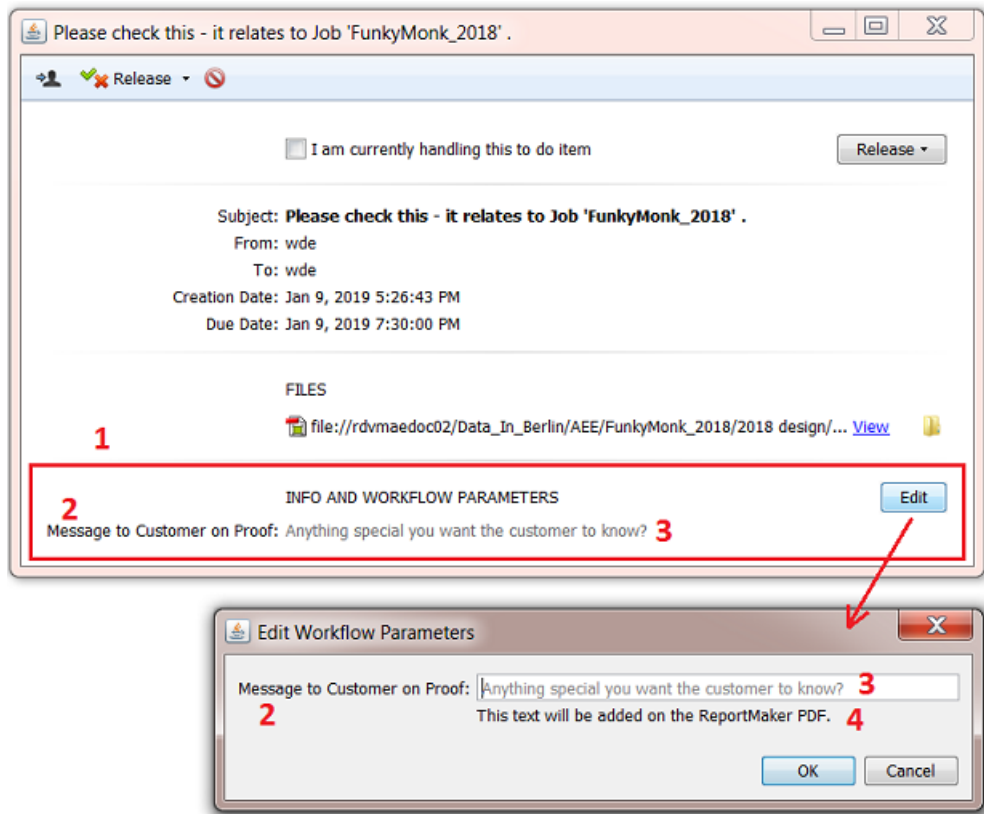
Some Examples

- Using a **Text Field**:



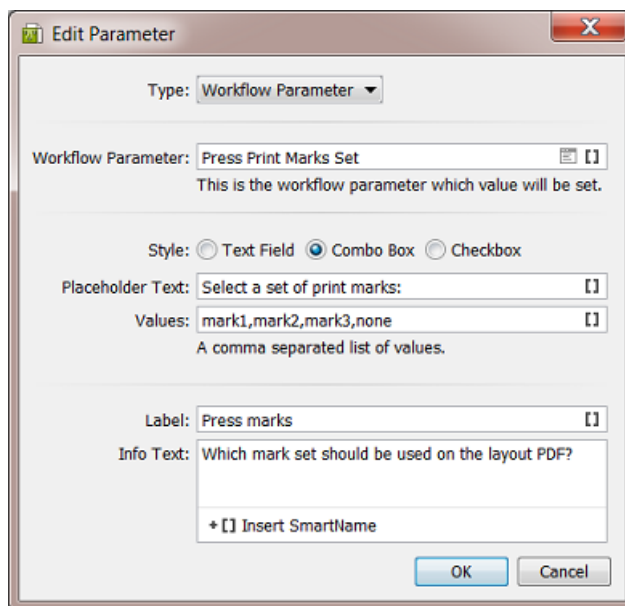
Resulting to-do item:

- 1. = **Info and Workflow Parameters**: An extra area in the to-do dialog where these extra parameters appear.
- 2. = **Label**
- 3. = **Placeholder Text**
- 4. = **Info Text**

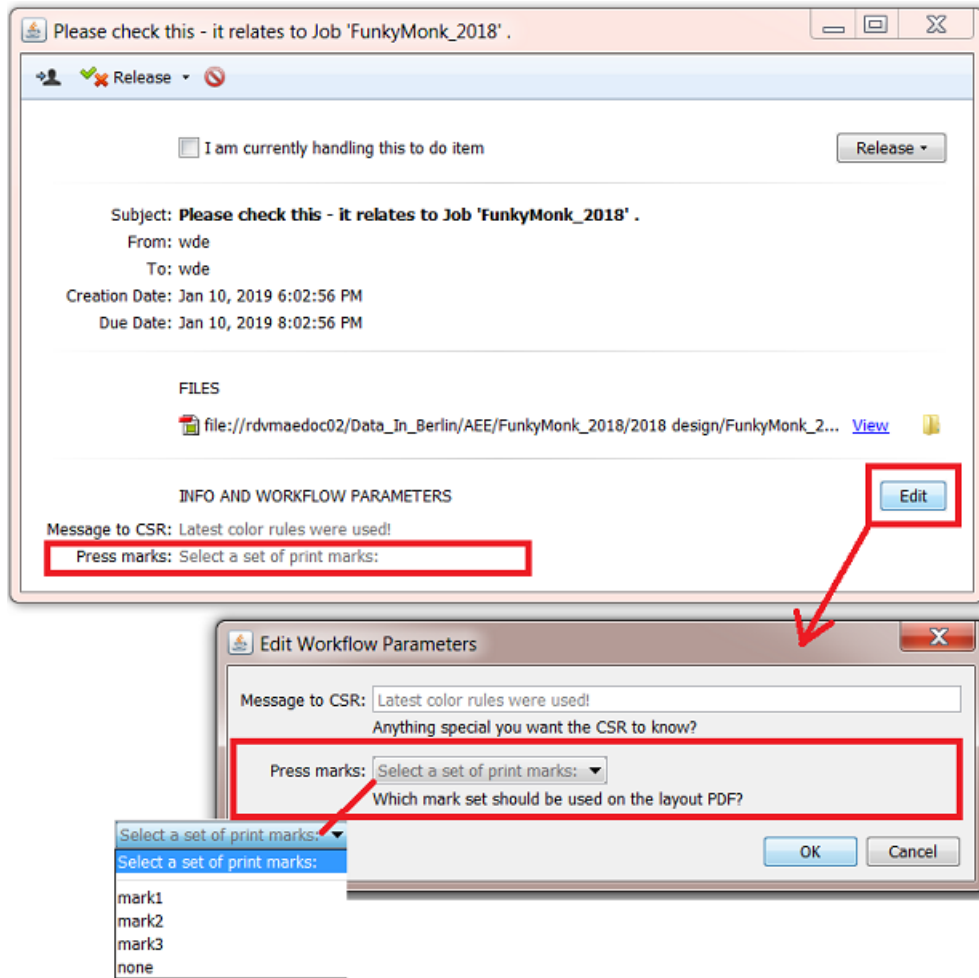


Attention: Notice how the user needs to click **Edit** to change a value.

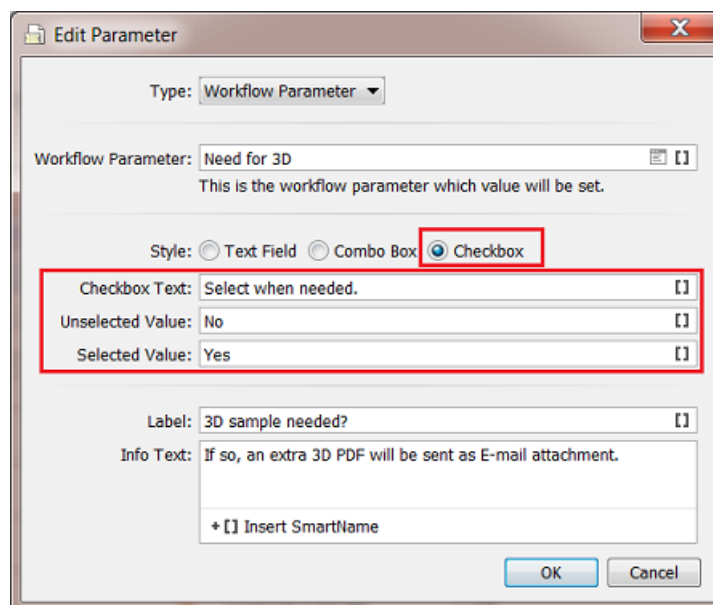
- Using a **Combo Box**:



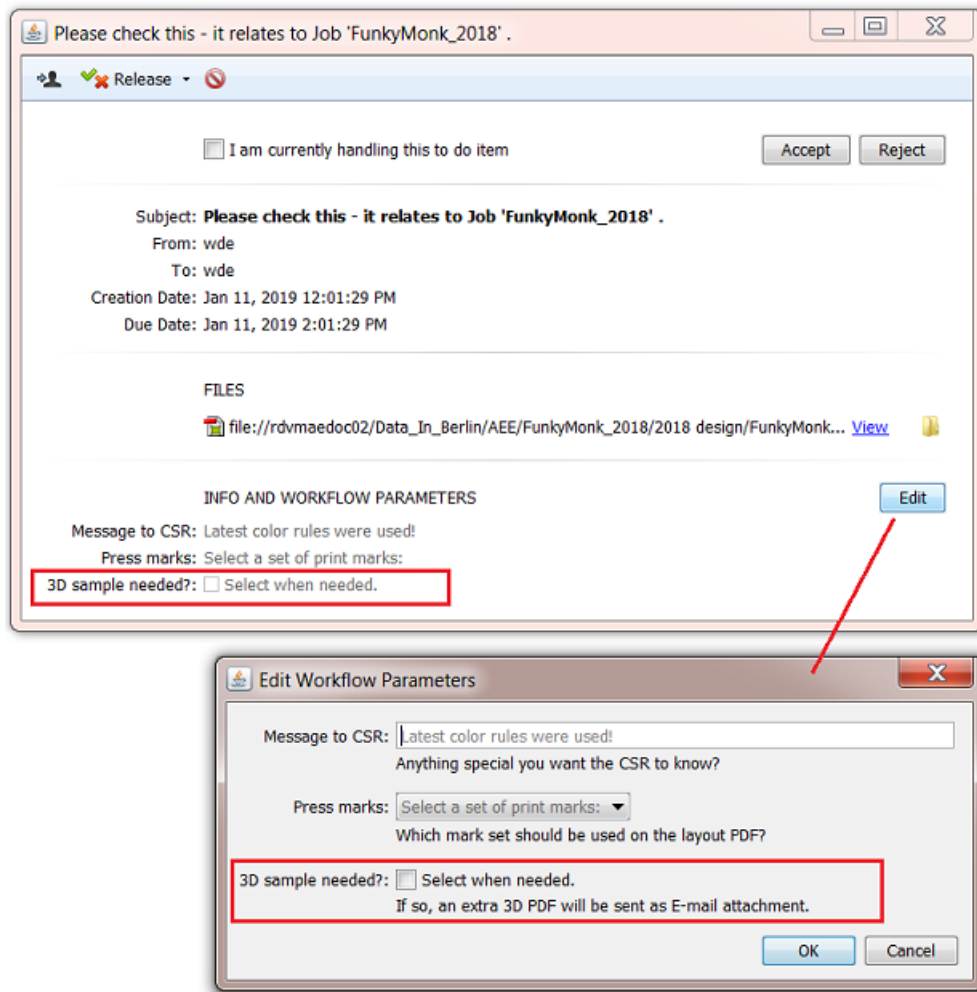
Resulting to-do:



- Using a **Checkbox**:



Resulting to-do:



Adding extra Info to a To-Do

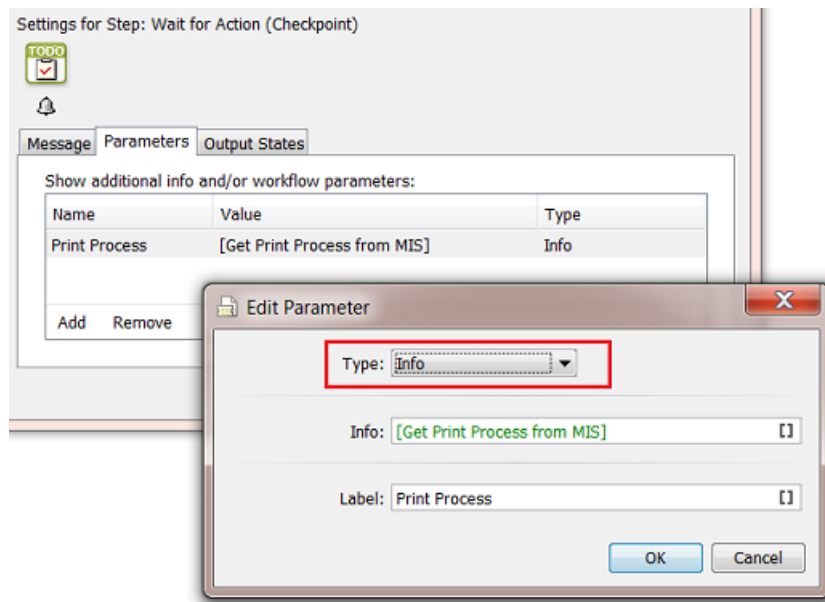
Settings

- **Info:** The value of the to-do parameter.
- **Label:** The text that will appear in front of the to-do parameter.



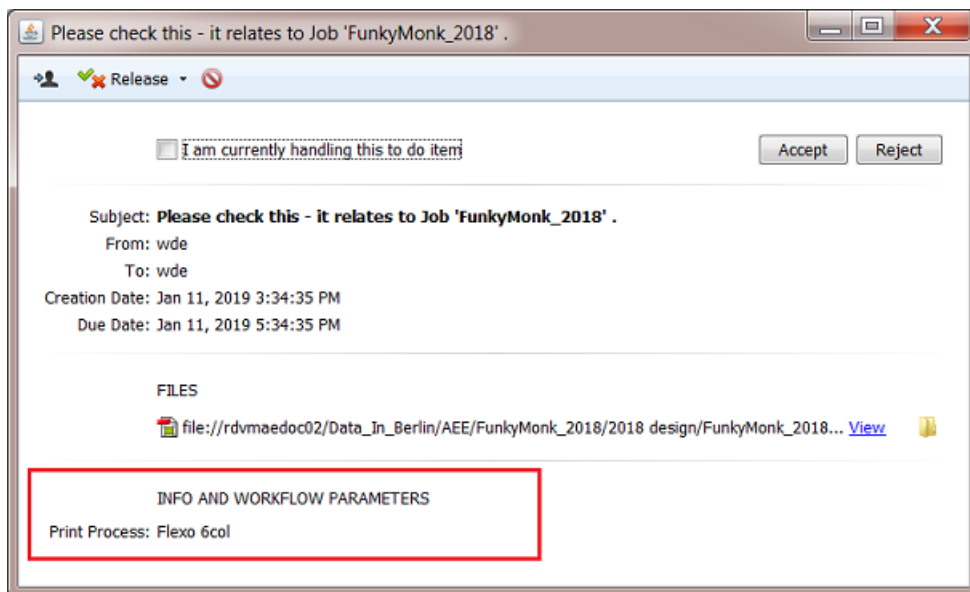
Note: Contrary to workflow parameters, this information is read-only ; the value can not be edited in the to-do item.

Example



Resulting to-do:

When the value that the SmartName retrieved from the MIS is 'Flexo 6col':



Adding a Link to a To-Do

Settings

- Click **Add**. The **Edit Parameter** dialog opens.
- **Link Type**: The type of hyperlink.
 - **External Link**: Fill in the URL in the **URL** field. This can be a hyperlink to an external website (example: <https://www.esko.com>) or to an *E-mail link* (example: <mailto:someone@example.com>).



Note: The scheme `http://` or `https://` is mandatory for the URL to be correct. `www.esko.com` is not a valid URL. The URL also needs to be *URL encoded*.

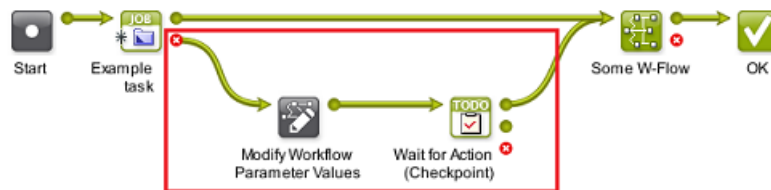
- **Task Link:** This is a link to a (finished) Automation Engine task. The value needs to be the **Task ID (internal)** of that task (learn below how to retrieve that value). In the to-do item, when you then click on such a link, the task with that internal ID will be selected in the Tasks view. See an example below.
- **Link Text:** It is possible to specify a text that will be shown in stead of the actual hyperlink. When you leave this empty, the URL will be shown in the hyperlink.
- **Label:** This is the text that, in the to-do item, will appear in front of the link.

Example of a Task Link

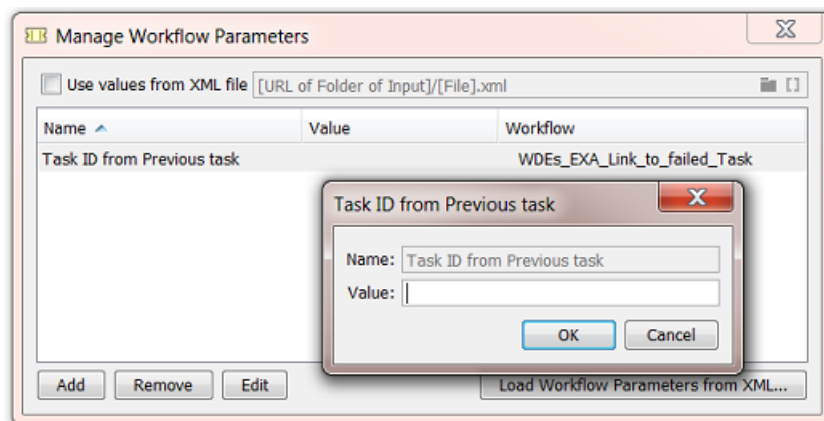
Some automated tasks are not monitored by users, and nobody notices when they go wrong. To alert the users and make them analyse why this specific task had a problem, you can add the following settings to that automated workflow:

In this example, we want to make sure we are aware when something goes wrong with the 'Create Job' step in our workflow. This workflow could for example be triggered by a business system or be started by Esko Cloud.

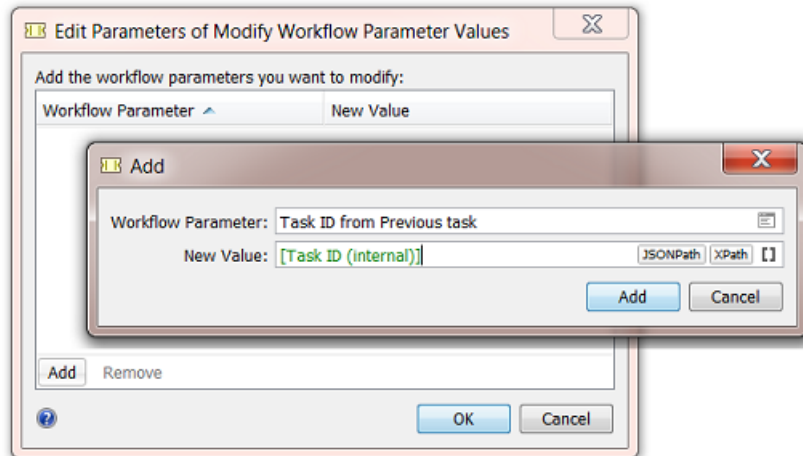
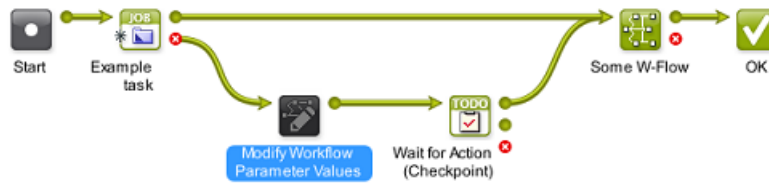
1. When that step ends in error (or warning), redirect the workflow to these 2 extra steps:



2. In the workflow editor, go to **Advanced > Manage Workflow Parameter**, **Add** a workflow parameter and **Name** it "Task ID from Previous task". Leave the **Value** empty and click **OK** and close the dialog.

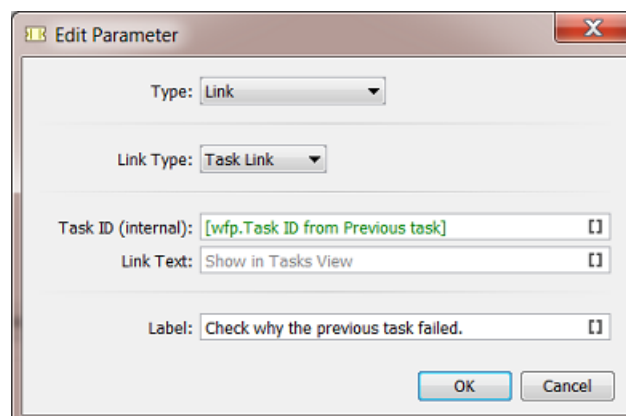


3. Open the ticket of the step **Manage Workflow Parameter Values**. Click **Add** and click to select the workflow parameter you just created. Set the **New Value** of this parameter to the SmartName **Task ID (Internal)**. This will pick up the internal ID of the previous task in the workflow, in our example the 'Create Job' task. Click **Add** and **OK**.



4. Open the ticket of the step **Wait for Action (Checkpoint)**.

- In the tab **Message**, add the **subject** "Check why the previous task failed".
- In the tab **Parameters**, click **Add** and select
 - **Type: Link** and **Link Type: Task Link**.
 - **Task ID (internal)**: Open the list of SmartNames. Select the category 'workflow parameters', select the one you made earlier and click **Insert**:



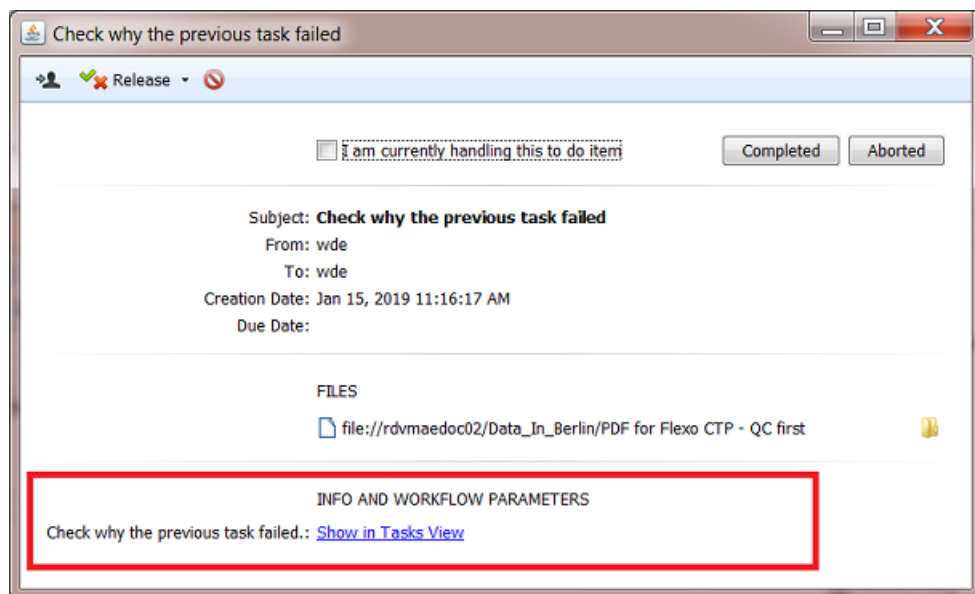
- **Link Text**: In our example, we leave the default text that will appear as the text that the user will need to click on ("**Show in Tasks View**").
- **Label**: Type 'Check why the previous task failed'.

- Click **OK**, **OK** and select **File > Save** to save the workflow ticket (or choose **Save As...** and save it as **Favorite**).

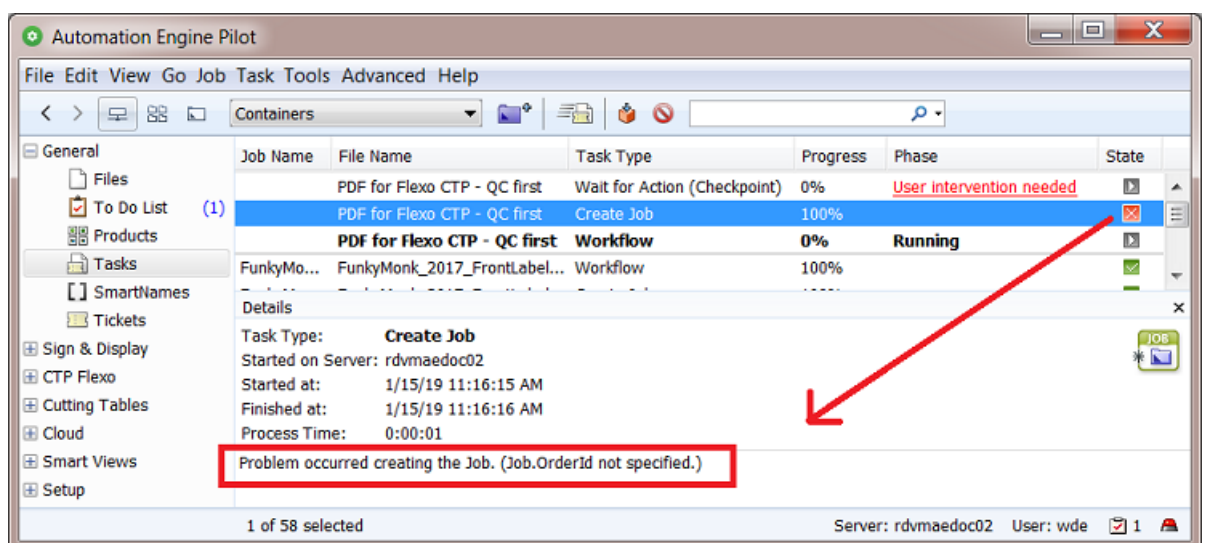


Note: In this workflow, we did not edit the step 'Create Job'. This is fine because we want to have that step end with an error anyway.

- In the Pilot, in **Files** view, choose the **Containers** mode, select any input file or folder and launch this workflow.
- You should quickly see that the step 'Create Job' failed and that a to-do item was created.
- Go to the **To Do List** view, select or open the item that was just created:



- Click on the blue hyperlink. It should open the **Tasks view** and have the task selected that failed. The task's **Details** or **Log** file will mention what the problem was.





Tip: When you right-click a [token](#), you can ask to see the value of the workflow parameters at that point in the workflow. In our example, this visualises the internal ID that we used to find that task.

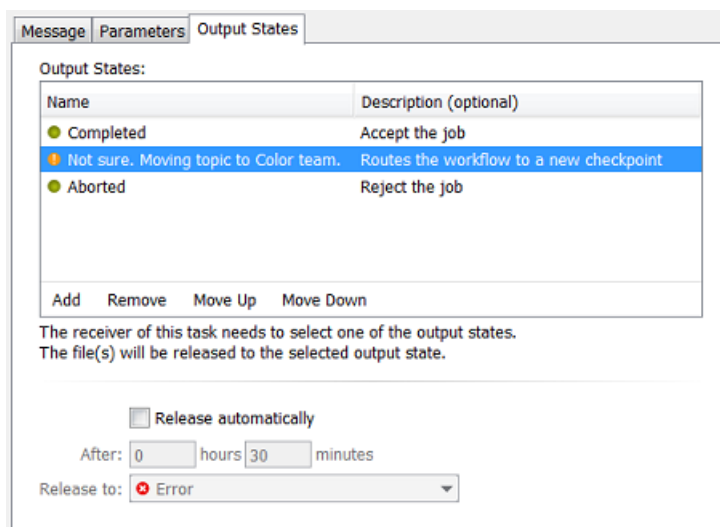
The screenshot shows a workflow diagram with several tasks: Start, Example task, Modify Workflow Parameter Values, Wait for Action (Checkpoint), User intervention needed, Some W-Flow, and OK. A context menu is open over the 'Wait for Action (Checkpoint)' task. The menu items are: Open..., Open With, View..., Relaunch, Show in Files View, Show in Tasks View, Open Task..., Open Log..., Open Details..., and Workflow Parameter Values... (highlighted with a red box and a red arrow pointing to it).



Tip: In the to-do item, hover your mouse over the hyperlink to see what it resolves to:

The screenshot shows a tooltip titled 'INFO AND WORKFLOW PARAMETERS'. The text inside reads: 'Check why the previous task failed.: [Show in Tasks View](#)'. Below the text, the URL 'ae.task?id=15c3ff1d-b11e-4e35-821e-287c436a109d' is displayed in a text box.

Tab 'Output States'



- **Output States:** You here define in what state this to-do item can be released. In a workflow, these output states are shown as output 'pins' with that (custom) name.
- Click **Add** to add a new output state. Enter a **Name** and **Description** (optional) for the output state.
- Choose the **Type** from the list of symbols.



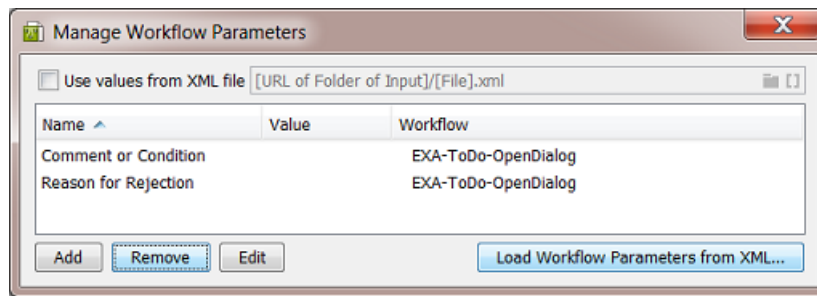
Note: When this task is used as a step in a workflow, the **Type** only defines the small icon that will be used for the output pin. The state of the workflow only depends on which next step in the workflow that that output pin links to.

For example: When choosing a (red) 'Error' symbol for a state that you named 'Not OK', then this does not yet mean that the workflow ends in error.

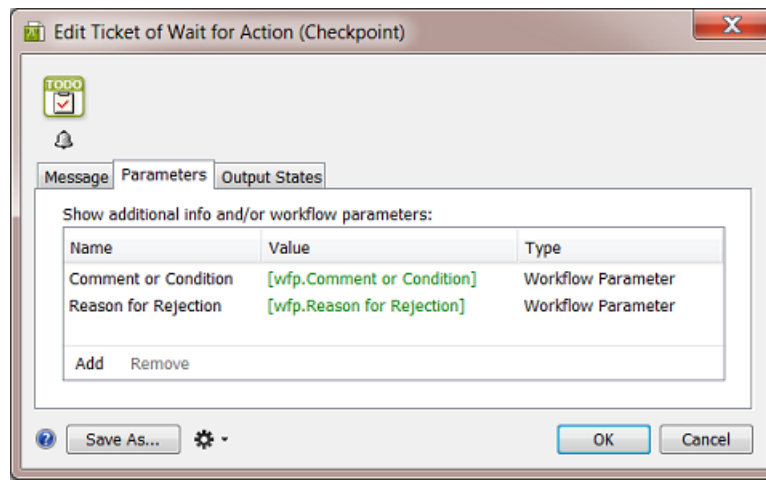
- Whatever the type of output pin (green, orange, red), it can be linked to any next step in the workflow, possibly a specific branch for that state.
 - When there is no next step, then indeed the workflow will end with the same state as the one you choose here.
- **Show workflow parameters dialog.** When selected, clicking the button of an output status in the to-do message will first open this dialog. This allows you to add comments or make changes to these parameters before that chosen output state is confirmed. See an example below.
 - **Release automatically:** Select this if you want the task to end after a specific time to a specific state.

Example of Output States that first Ask for More Info

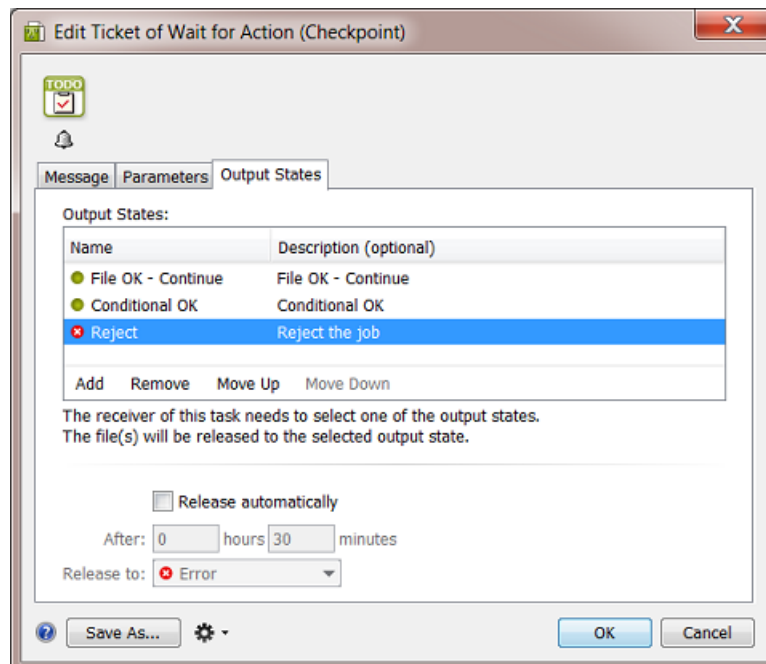
1. Add the workflow parameters to the workflow ticket that contains the 'Wait for Action' step.



2. In the **Wait for Action (Checkpoint)** step, they appear in the tab **Parameters**:

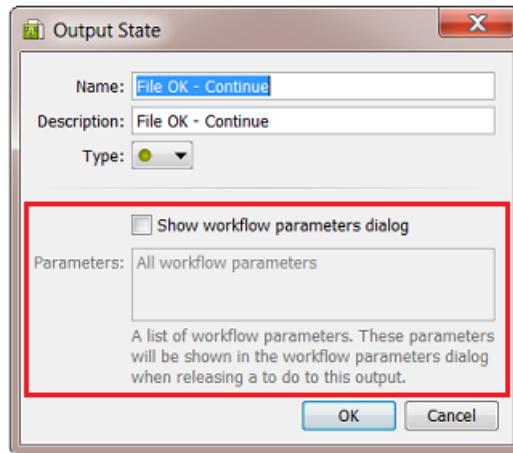


3. The next tab **Output States** is where we define how a user can decide how to end the resulting to-do action. Here are the 3 states we added and their settings:



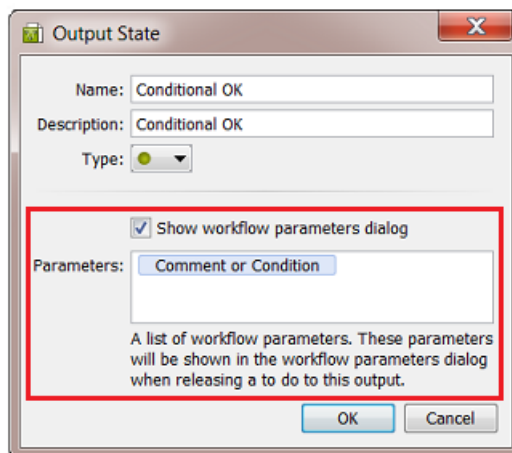
- a. The output state 'File OK - Continue':

When a user clicks this button, the to-do action will end and close.



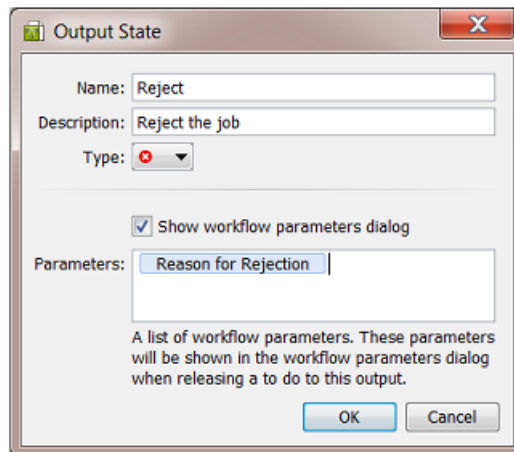
- b. The output state 'Conditional OK':

- **Show workflow parameters dialog** is selected.
- When the field **Parameters** is left empty, all workflow parameters will be presented. When you select one, as was done here, only that one will be shown.

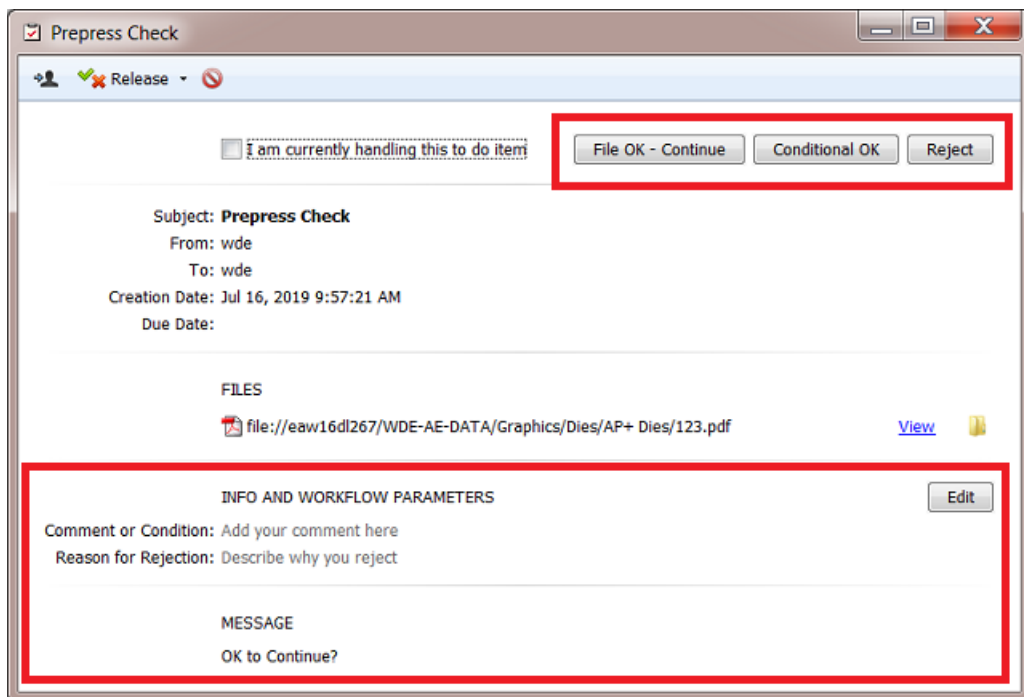


- c. The output state 'Reject':

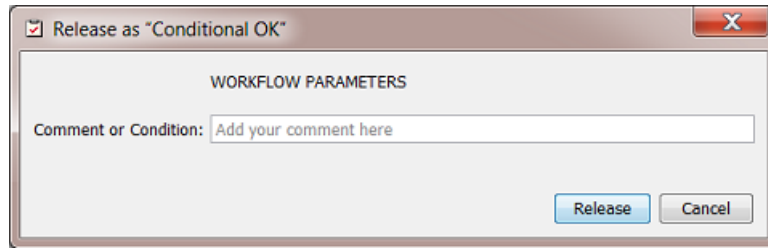
- **Show workflow parameters dialog** is selected.
- Again, only one parameter was selected: again only the one that is relevant to the choice of rejecting the file.



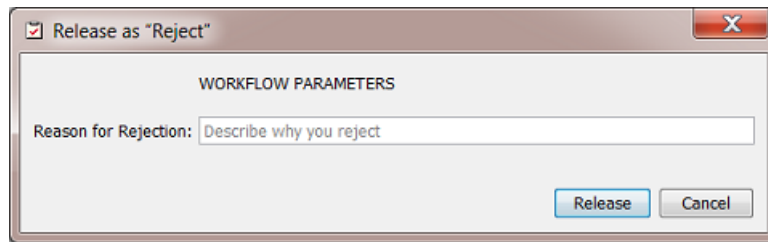
- As a result, when the workflow runs the step 'Wait for Action', this is how the to-do action will present itself:



- You can at any point click the button **Edit** and see or edit the provided parameters. But in this case we configured the decision buttons on top so that we don't need to click **Edit** any more.
- When the user clicks ...
 - 'File OK - Continue', the to-do action is released with this state and the dialog closes.
 - 'Conditional OK', the dialog opens that allows us to edit the workflow parameters. We only see the parameter that was chosen. Add a comment and click **Release** this to-do action to this output state.



- 'Reject', the same dialog opens, now with the other parameter that is relevant to this output state.



Using the To Do List to Distribute Work

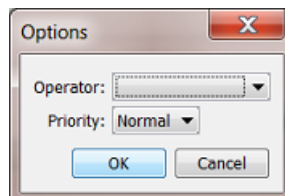
Example of a Workflow combining a To-Do for "Any user" with one for "Task Owner"

It is possible to use the To Do List to distribute tasks. The first user who picks up a to-do action can become the owner of the rest of the workflow.

See this example of a workflow that is launched by a **Folder Access Point**: the first Checkpoint task sends the to-do to **Any User**. The second Checkpoint task sends it to the **Task Owner**:



The Folder Access Point launches this workflow with a non-specified user:



Learn more about Access Points in [Access Points](#).

This is how the result looks like in the Tasks view:



Tasks						
File Name	Task Type	Progress	Phase	State	Launched	Operator
banana.pdf	Wait for Action (Checkpoint)	0%	User intervention needed		10/30/14 4:24 PM	wde@rdvmaedemo
banana.pdf	Wait	100%			10/30/14 4:23 PM	wde@rdvmaedemo
banana.pdf	Wait for Action (Checkpoint)	100%			10/30/14 4:20 PM	UPLC@RDVMAEDEMO
banana.pdf	Workflow	0%	Running		10/30/14 4:20 PM	UPLC@RDVMAEDEMO

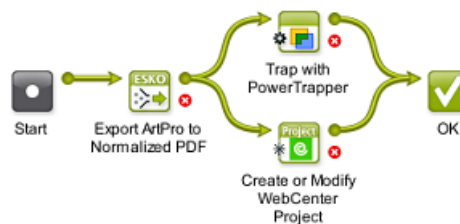
- At first, the user was "UPLC" (the system user when no user is specified in an Access Point).
- The first to-do became visible in the list of All users.
- This first to-do was handled by the user "wde", which made him/her the Task Owner from that point on in the workflow.
- The second to-do goes to the Task Owner, which is still "wde".

2.10. Routing in Workflows

2.10.1. Simple Processing Routing

As mentioned in *Building a Workflow from Scratch*, transitions can route your files to different workflow steps. Automation Engine will automatically check the format of the files going through this step, and send them to the appropriate step(s) for that file format.

See this example where the PDF will go to the trapping step:



However, to keep an easier visual overview of your workflow, we advise to use a Router workflow control instead (learn more in *Automatic Routing* on page 43).

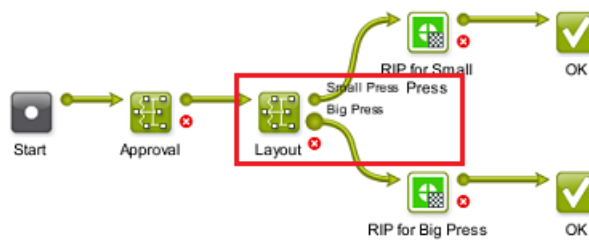
2.10.2. Routing in Subworkflows

When a subworkflow has multiple end steps, an extra type of routing becomes possible. The end steps of your subworkflow correspond with the outputs of their workflow step in your master workflow.

For example, this subworkflow ends with two 'OK' end steps that were renamed to "Small Press" and "Big Press".



In the master workflow, you can see that they correspond to the two green output pins of the subworkflow step "Layout".



This means that:

- files sent to the "Small Press" end of your subworkflow will go through the "Small Press" output pin in your master workflow,
- files sent to the "Big Press" end of your subworkflow will go through the "Big Press" output pin in your master workflow.

2.10.3. Routing via To Do List Action Items

Users can decide the route of the workflow (files) through **Action** items in their **To Do List**. They can also choose to forward the to-do action item to another Automation Engine user.

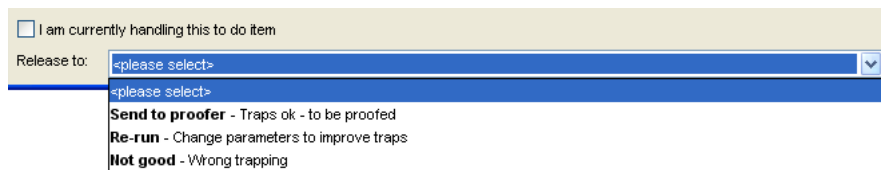
Learn more about Workflows creating Actions in a user's To Do List in [The Wait for Action \(Checkpoint\) Task](#) on page 23. Learn more about using the To Do List in [The To Do List View](#).

In the Pilot Tasks View

In the Tasks view, the **Wait for Action (Checkpoint)** task shows the **Phase "User intervention needed"**. Below right, the icon 1 shows that one (extra) item is added to this user's To Do List.

Job Name	File Name	Task Type	Progress	Phase	State	Launched
ta_shu_Plugin_Set...		Wait for Action (Checkp...	0%	User intervention needed		11/02/10 10:43
ta_shu_Plugin_Set...		Trap - Prepare and Cre...	100%			11/02/10 10:43
ta_shu_Plugin_Set...		Normalize PostScript / P...	100%			11/02/10 10:42
ta_shu_Plugin_S...		Workflow	0%	Running		11/02/10 10:42


You can click either of these elements to **Release** the item to one of the outputs that were defined in the **Wait for Action (Checkpoint)** task. This will close the To Do item.



In the Workflow Editor

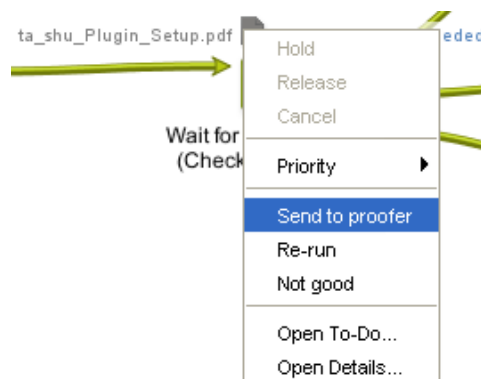
In the workflow editor, the user will see **User intervention needed** as the file status and the **Checkpoint** task step will jump up and down:




The workflow editor will also show one to-do  (for this workflow).

To route (and close) the to-do item, you can

- right-click the file icon and select one of the outputs defined in the **Checkpoint** task.



- right-click the file icon and select **Open To-Do...** and there select one of the outputs offered in the **Release** button
- click the To Do List icon  to open the item in the To Do List and there select one of the outputs.



Note: In the To Do List, make sure the **Actions Filter** shows the **Wait for Action** items.

In Shuttle

In the client application **Shuttle**, Checkpoint tasks and To Dos can be handled in the same way as you do [in the Pilot Tasks View](#).



Attention: The user name you used in the **Checkpoint** task and the user name used in Shuttle to connect to Automation Engine must match.



Note: Learn more about Shuttle in [Working with Shuttle](#).

2.10.4. Automatic Routing



The **Router** workflow control enables a workflow to automatically decide the route that your workflow (files) should follow.

The criteria that you define to route to a specific output Pin can be based on many different types: the file type, the number of separations, the value of a Job Parameter, etc.

For example: routing files based on their **File Type**:

Route based on: File Type

File Type	Route to
is ArtPro File	ArtPro workflow
is PDF File	PDF workflow
is Normalized PDF File	PDF workflow
no match	Error

Buttons: Add, Remove, Move Up, Move Down

... and so send each file type to an appropriate next task in the workflow:



In the above example, ArtPro files are routed to their own normalization task, while Native or Normalized PDF files are sent directly to the Trap task.

Learn more about the Router in [Router](#) on page 79.

2.11. Exchanging Workflows

The two main reasons to export and import workflows are

- to exchange them with other plants of your company that also use Automation Engine.
- to exchange them with Esko Customer Service.

Exchanging workflow tickets is done *in the same way as for single tickets*:

1. **Export** the workflow to a ZIP file (in Tickets view or in the Workflow Editor).
2. Send the ZIP file to the other Automation Engine (onto a client computer of that server).
3. At the receiving site, in Pilot Tickets View, select **Import**, browse to the ZIP file, and click **OK**.



Attention: The Automation Engine software version on the server that imports the ZIP may not be on an older version than the Automation Engine server that created the ZIP.

What about references?

As mentioned in *Functionality in the Tickets view*, references will also be copied into the ZIP file. These references are not just files like templates but also SmartNames, profiles, scripts, referenced workflow tickets, marks etc.



Attention: Workflows that include referenced (blue) Job Tickets can not be exported. An error message will appear.



Note: Exchanging workflows is often done just to exchange the SmartNames used in those workflows. Learn more in *Exporting and Importing SmartNames*.

2.12. Good Practices for Building Workflows

Below tips help you build a workflow that is sustainable and easy to maintain. They also avoid extra work and cost when you upgrade or migrate server computers.

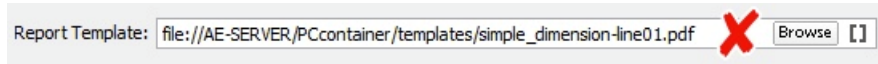


Note: Some of these tips are also present as an Esko Knowledge Base article. Find these and more on <http://help.esko.com/knowledgebase>.

2.12.1. Avoid Absolute Paths and Server Names

Do you know the name of your file server in five years from now? Probably not. That is why we recommend to avoid using that computer name in your Tickets and Workflows.

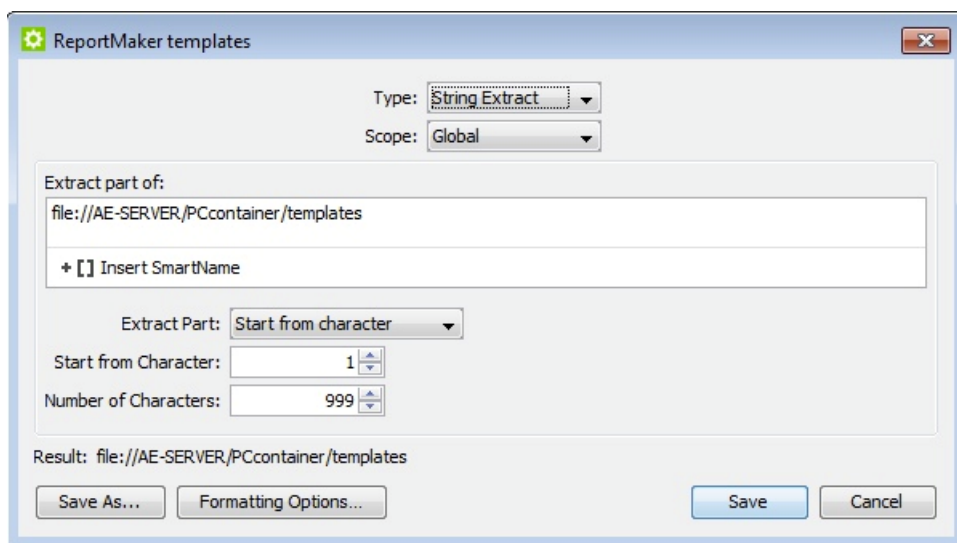
For example when you define a template for the ReportMaker task. See below example of a Ticket with an absolute path to one specific template. If you have many such tickets, and the location of the templates changes, you then have to adjust all your ReportMaker tickets.



This can be avoided by using a SmartName in the path. When the path changes, you then only have to update the SmartName.



The SmartName can be a **String Extract SmartName** that contains the path or even just the server name:



The same principle applies to other tasks with references to files or folders like **Compare PDF**, **Create CAD Sheet** and **Select File**. If you copy a PDF file to an external system, a CIP3 file to a press, a TIFF file to an imaging device, a JPEG preview to an MIS, a ZIP file to an archiving system etc., always avoid using server names and fixed locations.



Note: You can keep it even more generic. For example if your templates are saved on the Automation Engine server, you can use the system SmartName [Server] to make your path smart. No need to create an extra SmartName in this case, but you will have to keep the templates on the Automation Engine server when you migrate to new hardware.

Some typical situations where you better use SmartNames in paths:

- ReportMaker templates
- CAD files repository
- Image Mark files in SmartMark sets
- Files loaded in ArtPro Action Lists
- PDF, XML, JPEG communication with MIS/ERP
- TIFF, LEN, PS data sent to output devices
- Defining the file location of a Product.

For some situations, there is a default solution:

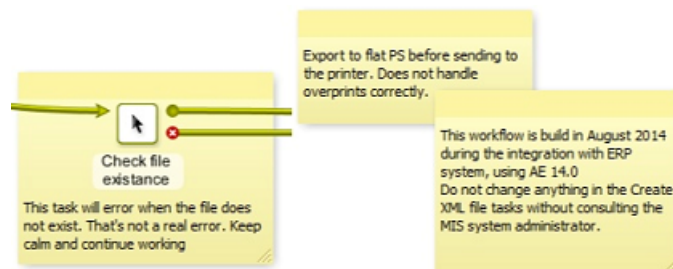
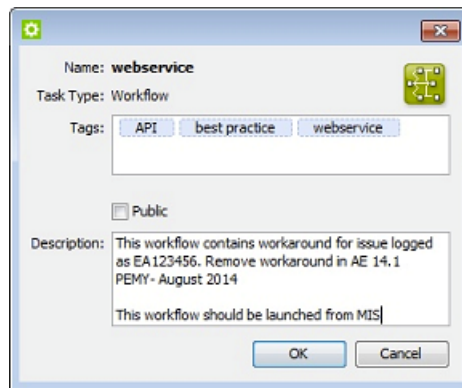
- In the Create Job ticket, use the Smart Job Location instead.
- Save PitStop Profiles in the default "PitStop Root" container (only visible in the PitStop ticket).
- FastImpose templates have a default location on the Automation Engine server.

2.12.2. Annotate

If you are implementing a temporary workaround or even misusing a task to achieve your goals, make sure that you later remember why you did this. There might be a better solution or less strict requirements in a next version of Automation Engine. An exotic part of the workflow for which you don't remember why or how you did it, is a part that nobody later dares to change or remove. It will cause problems sooner or later.

The best way to manage this is to document on the spot. Automation Engine provides plenty of possibilities to do so:

- Description field for tickets and workflows
- Tags for tickets and workflows (also helps filtering)
- Renaming the workflow steps
- Description field for SmartNames
- Tags for SmartNames
- Sticky Notes in workflows



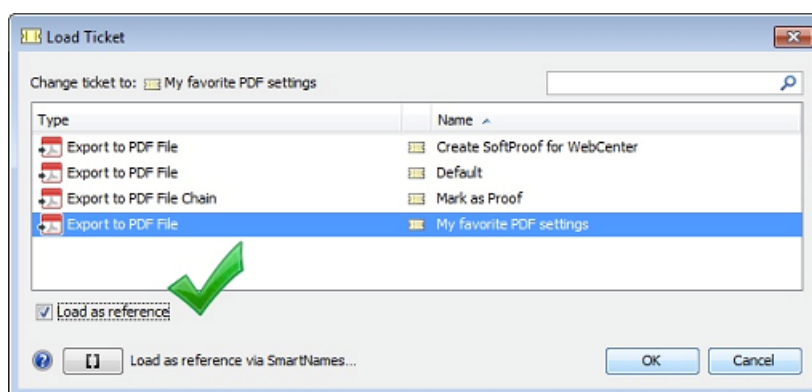


Note: Documenting your experiences while building a workflow is not only a great help for yourself and your colleagues but also for anybody at Esko Customer Service.

2.12.3. Use Referenced Tickets and Workflows

Referenced Tickets

You probably use several Export to PDF tickets. We advise to not copy-paste such a ticket in your workflows but to add a reference to this ticket. Then, when you decide to change a setting (for example "outline fonts"), you only have to change this single ticket and all the workflows that use it will so adapt and use these latest settings.

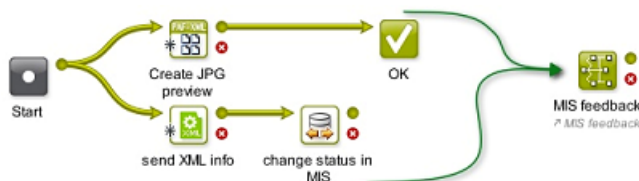


Learn more about referenced tickets in [Building a Workflow using Custom Tickets](#) on page 12.

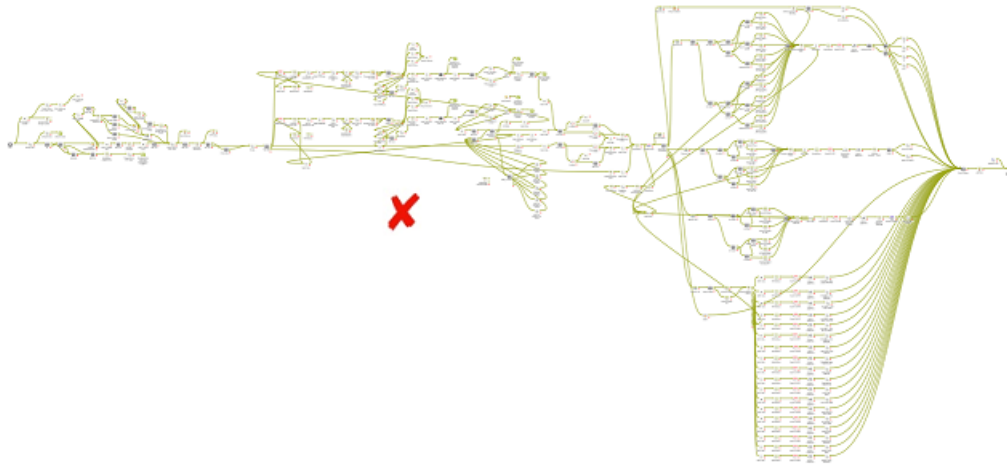
Reoccurring Workflows Parts

When part of a workflow is identical in other workflows, we advise to create it as a separate workflow and then refer to it from the master workflow(s).

For example, a part of your workflow sends feedback about a file to the business system. You probably use this part in several workflows. We advise to create it once and use it many times as a workflow step in various master workflows. We again advise to not copy it but to add it to the master workflow as a reference. This way, when settings change, you will only have to adapt it once.

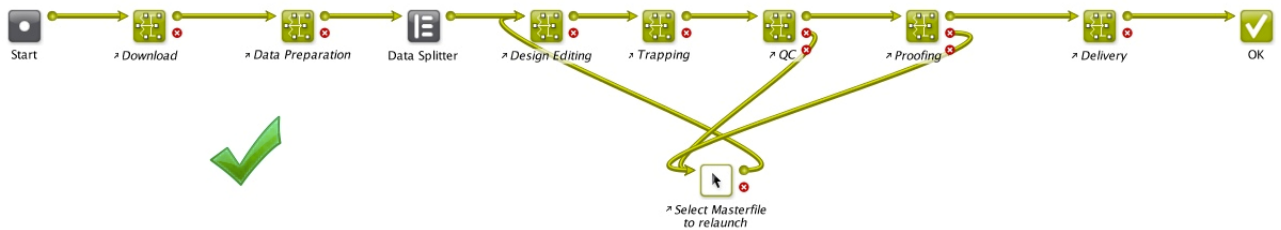


Workflows can get gigantic. When launching such a workflow, the Pilot will read all the settings of all the tasks in this workflow, even in the subworkflows. This is why launching bigger workflows seems slow. See an example of such huge workflows that you should try to avoid creating:



However, at the moment of launching, the Pilot will not immediately read the settings of a *referenced* workflow. This adds an extra reason to divide such huge workflows into several referenced workflows. The master workflow will not only react quicker, but will be much more structured and easier to understand.

Here is a good example of a master workflow that quickly visualises the logic workflow process steps:



Learn more about copied and referenced subworkflows in [Subworkflows](#) on page 18.

2.12.4. Avoid excessive amounts of Tokens

Learn about tokens in [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

Problem

Workflows that create an excessive amount of tokens will affect overall performance and stability.

Some constructions in a workflow ticket create a massive amount of tokens. Most typical constructs that can lead to such problems are

- Splitting without re-collecting later: For example a 'Split XML' task followed by a 'Data Splitter'. This way, one large XML file (one token) can – depending on how you split it – explode into hundreds of tokens that the workflow needs to manage in its next steps.

- Multiple inputs: Selecting a large amount of inputs files as initial input for a workflow.
- and of course a combination of the above.

Solution

Check the setup of that workflow. Analyse one that was launched or has finished ; you can then see where the 'explosion' of tokens starts.

Check how these tools can help creating less tokens:

- The workflow control [Data Collector](#). It is OK to split, but always consider to collect again later.
- The ['Mark File' and 'Select Marked File'](#). See an example below.
- Build checks into your workflow to detect and prevent an overload of tokens. See an example below.

Below examples contain some tips to avoid unnecessary build-up of tokens.

Example 1 - Allow the workflow to NOT track and provide all details

- An XML file lists the WebCenter projects where you need to launch a task for. The 'Split XML' and the 'Data Splitter' steps makes sure a task is launched for each project. This workflow will then track and trace all details of all 244 tasks that were launched in that subworkflow. The Tasks overview will show 491 entries for this workflow.



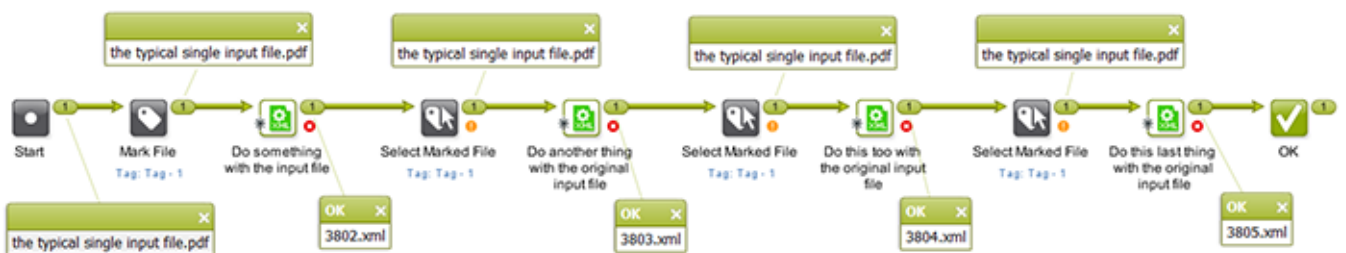
- An alternative is to use the 'Launch Workflow' step: This allows the workflow to not track and trace all details of these 244 sub tasks. The Tasks overview will show 3 entries for this workflow.



Example 2 - Enforce sequential processing and use checks

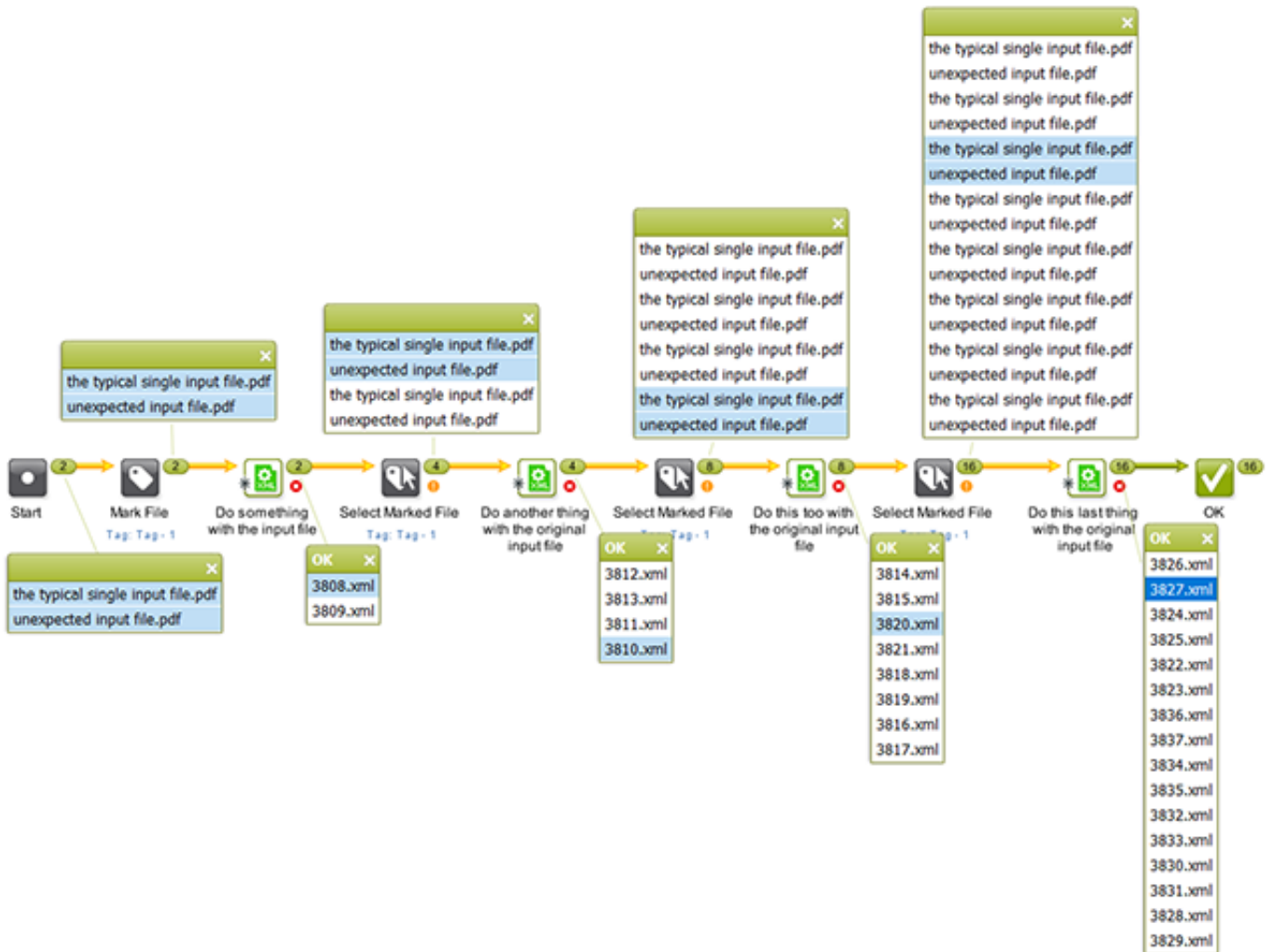
- This workflow wants to launch several actions on the original input file. And these 4 actions do not need to work with output files from another (previous) action.

This is an example case to use the steps ['Mark File' and 'Select Marked File'](#). Important effect is that workflow will not launch the 4 actions at the same time on that input file. They will happen **sequentially**, as you designed here in the workflow. This will be much better for the load on your workflow server than starting them all at the same time from the beginning.



- In several cases, it can happen that your workflow gets an unexpected input file (a wrong manual selection, a wrong decision by an external system, a bad timing on some Access Point, etc.).

See how, even in the above improved workflow, 1 extra unexpected input file can lead to many more tokens:



- A way to prevent this is to start with a checkpoint. In this case a [Router](#) that will only allow the usual workflow to continue when the amount of input files is the expected '1':



3. Launching and Diagnosing Workflows

3.1. The Parameter Inspector

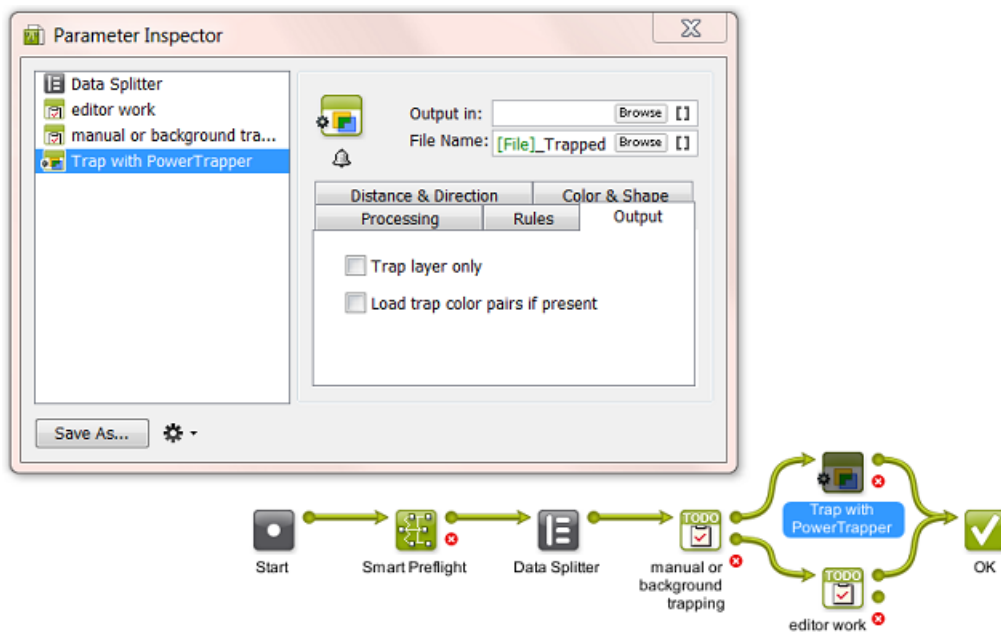
The **Parameter Inspector** enables you to quickly view or change the ticket parameters of your workflow steps, without having to open and close each workflow step separately.

Open it by choosing **Show Parameter Inspector...** from the workflow editor **File** menu or by choosing the same option after a right-click on the canvas.



Note: The parameter inspector is not available for workflows that were already launched (running or finished).


The window stays open while you navigate through the steps. The steps are shown in alphabetical order. You can navigate through the steps by selecting them in the parameter inspector or by clicking on steps in the workflow editor canvas.



The parameter inspector does not show

- steps that have no parameters (Start - Stop - Warning - Error - Cancel)
- subworkflow steps. Double click the subworkflow step on the canvas to see its own steps appear in the same parameter inspector window.
- referenced steps.

When available, you can

- choose **Save As...** to save the selected ticket as a separate task ticket.
- choose  to set extra options that you can also set when you open the ticket separately: [Notification Rules](#), [Manage Public parameters](#) and [Map Output Parameters](#).

3.2. Launching a Workflow on a File

There are several ways to launch a workflow on a file: from the Pilot or browser client, from within the Workflow Editor, from [Shuttle](#), via Access Points or via JDF/JMF.



Note: Learn more about launching workflows from external systems in [Integrating with External Systems](#).

Launching a workflow from the Pilot's **Files** View is very similar to launching a single Ticket. Follow these steps:

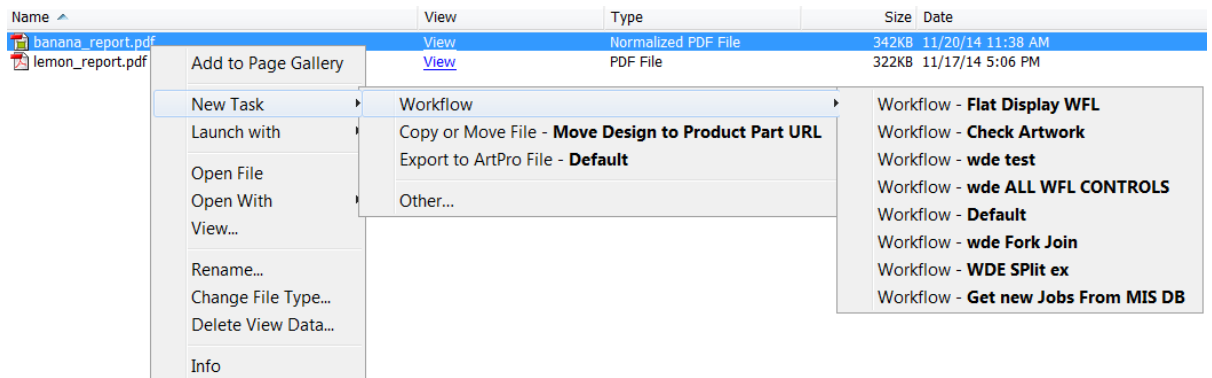
1. In the **Files** view, right-click the file(s) you want to process and select:
 - **New Task.** Choose this to select a workflow and view or change its settings before launching it.
 - **Launch with.** Choose this to select a workflow and launch it without changing its settings.



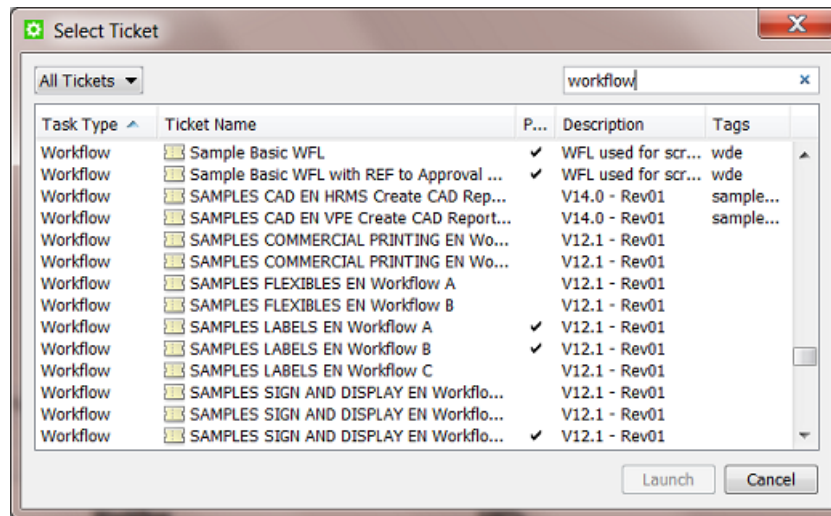
Note:

If you don't have the access right **Tickets: Show All Tickets and their Parameters (Public and other)**, you will only see **Launch with**. You will also only be able to select public workflow tickets. Learn more in [Access Rights](#).

2. Select the workflow ticket to use.
 - If you have launched workflows previously, the most recent workflow tickets you used are listed under **Workflow**.



- If you don't need a recently used workflow, select **Other....** In the **Select Ticket** dialog, select the workflow ticket you want to use and click **OK**.





Tip: Type `workflow` in the search field to quickly filter to the task type **Workflow**.

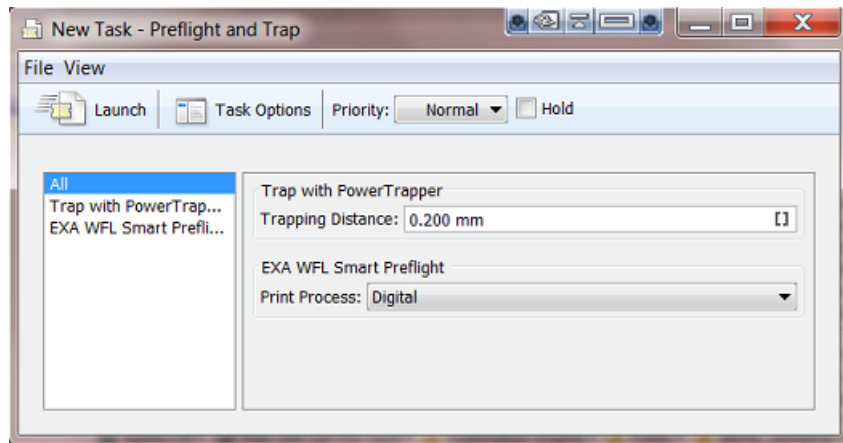


Tip: If you hold the **Alt** key, the **OK** button changes into **Launch**.

3. Some variations are possible:

- To keep the workflow editor window open and monitor the workflow while it is executing, hold **Alt** while clicking Launch to **Launch and Monitor** (the Launch button changes to ).
- If you choose **New Task**, the workflow editor opens. Fill in the settings you want to use in each workflow step and click the Launch  button.
- If you choose **Launch with** and you don't have the access right **Tickets: Show All Tickets and their Parameters (Public and other)**, then
 - if the ticket does not contain public parameters, the workflow is launched on the file as soon as you select the workflow ticket.
 - if the ticket contains public parameters, it will open for you to fill them in. You will only see the public parameters. Click the **Launch** button when you are done.

In this example the user is asked to decide 2 public parameters:



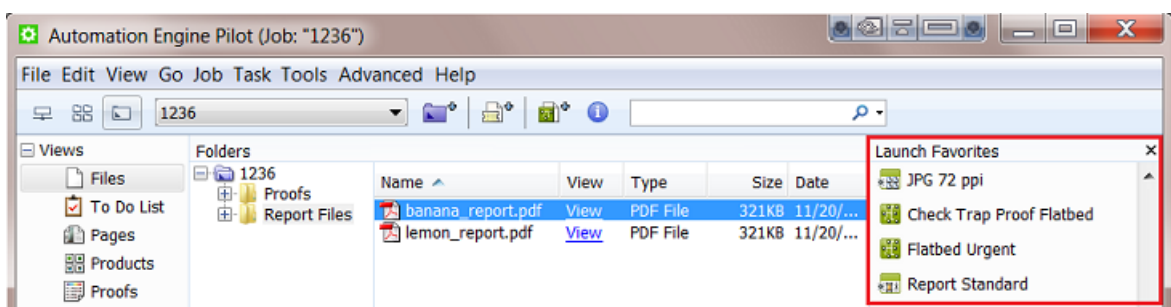
Note: If a workflow is launched on files that are linked to several Jobs, a separate workflow will be launched for each Job context. This can happen for example when you use the **Files** view or Shuttle or a Folder Access Point.

3.3. Using Favorite Workflow Tickets



Note: In [Creating Favorite Tickets](#), you can read how to build and launch **Favorite** tickets for single task tickets. You can do the same for workflows tickets, including [sorting and grouping them](#). We here repeat some information and add specifics about working with favorite workflows.

Favorite tickets are links to tickets that you use a lot. They show up in the **Favorites** pane, both in the Files and in the Tickets View. You can use **Pilot > View > Favorites** to show or hide the **Favorites** pane.



Favorites tickets show up depending on

- the file type you selected first.
- the user that is logged in. They are your personal favorites.

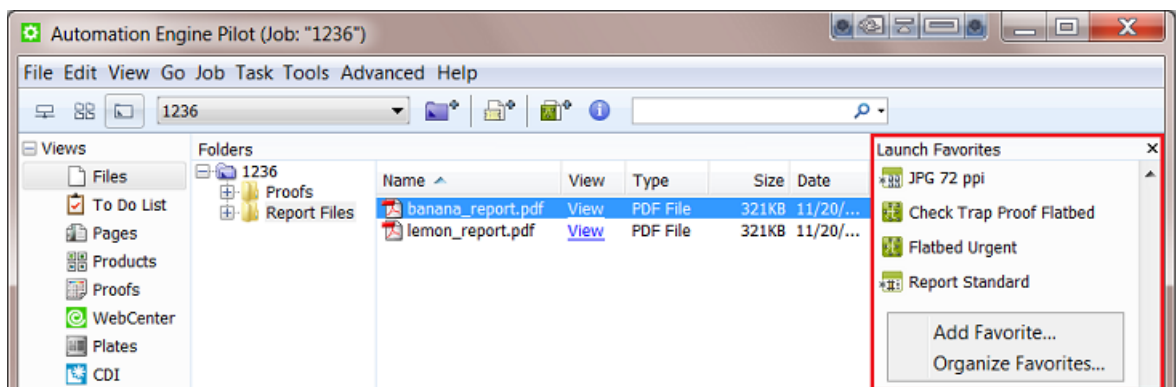


Tip: You can copy and paste favorite tickets from one user to another. In **Pilot > Tools > Users**, right-click a user name and choose **Copy** or **Paste Favorites**.

3.3.1. Adding Favorite Workflow Tickets

There are 3 ways to create a favorite (workflow) ticket:

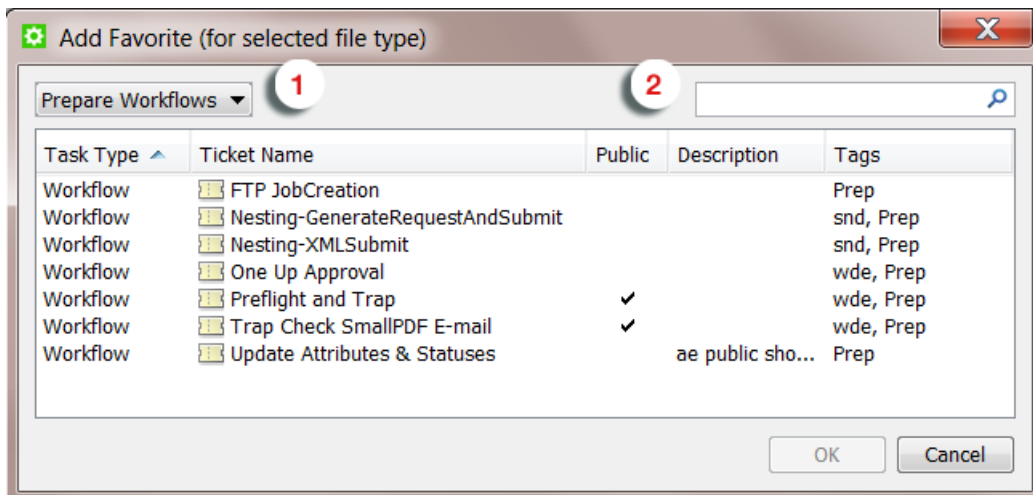
- when you first save a workflow or use 'save as', then check **Add to Favorites**.
- in the Tickets view, right-click a ticket and select **Add to Favorites**. Use **Shift** to select multiple tickets.
- both in the Files and in the Tickets View, in the **Favorites** pane, right-click and choose **Add Favorite...**



The Add Favorite Dialog

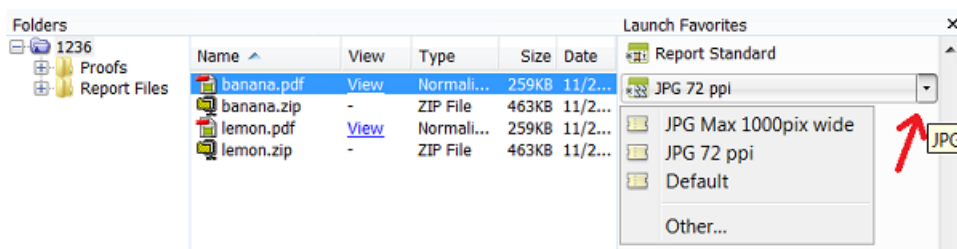
When selecting a (workflow) ticket to add as a favorite, these 2 tools help you filter your list of tickets:

1. Instead of seeing **All Tickets**, use the drop-down list and select one of the tickets filters that you created earlier and saved into your **Smart Views**. Learn more about Smart Views [here](#).
2. Type in a filter in the search field. What you type will be used to filter in all the columns.



There is No Need to add Many Favorites of Same Task Type

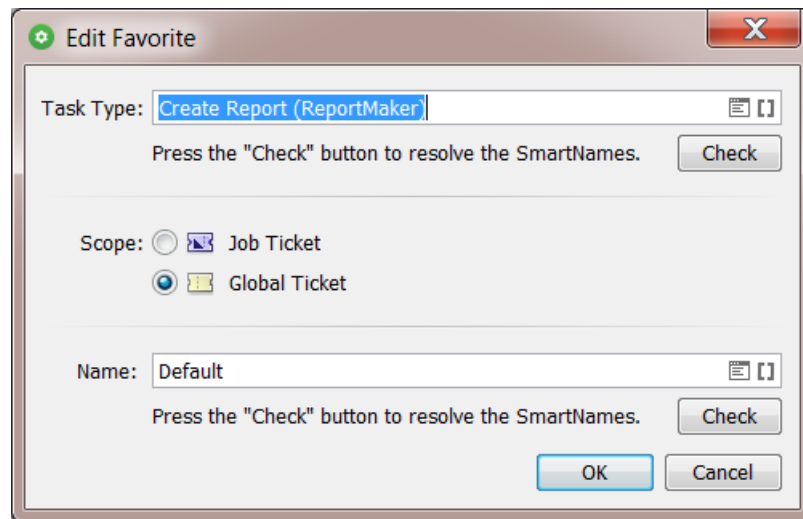
When you click on the right side of the favorite, a drop-down list will show you all tickets of the same task type (with a maximum of 10). This easy access to similar tickets means that you do not have to create a favorite of those other similar ones (when you also use them often).



When the ticket that you look for is not in that list (remember that maximum 10 are shown), you can choose **Other...** and select it there.

3.3.2. Adding a Favorite based on a SmartName

Right-click a favorite and choose **Edit Favorite**.

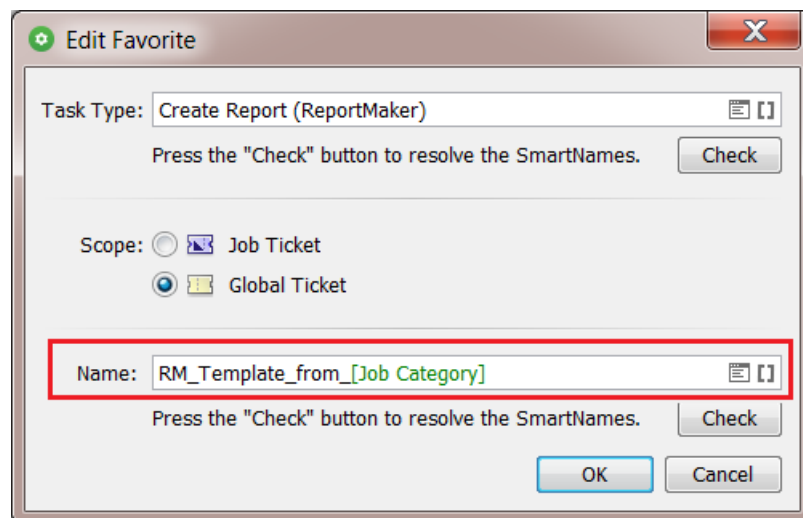


Click to pick a **Task Type** and or **Name** of the favorite ticket from a list.

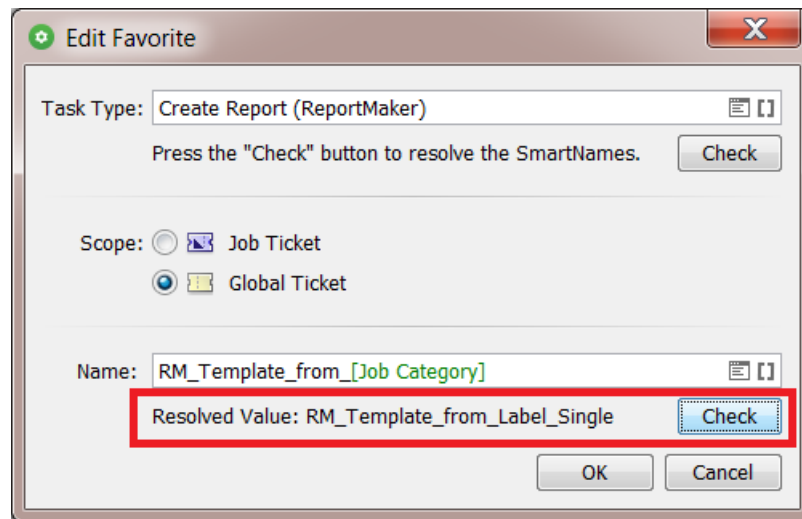
Click to define the Task Type and or the Name by a *SmartName*.

An example:

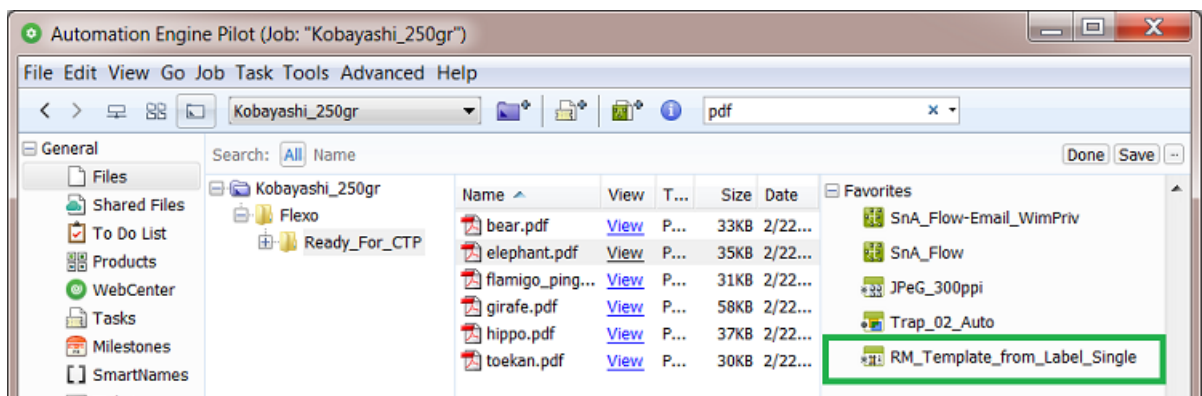
This favorite ticket will create a report file that uses a template chosen according to the category of job it is. It will pick a ticket with a name that mentions the **Job Category**:



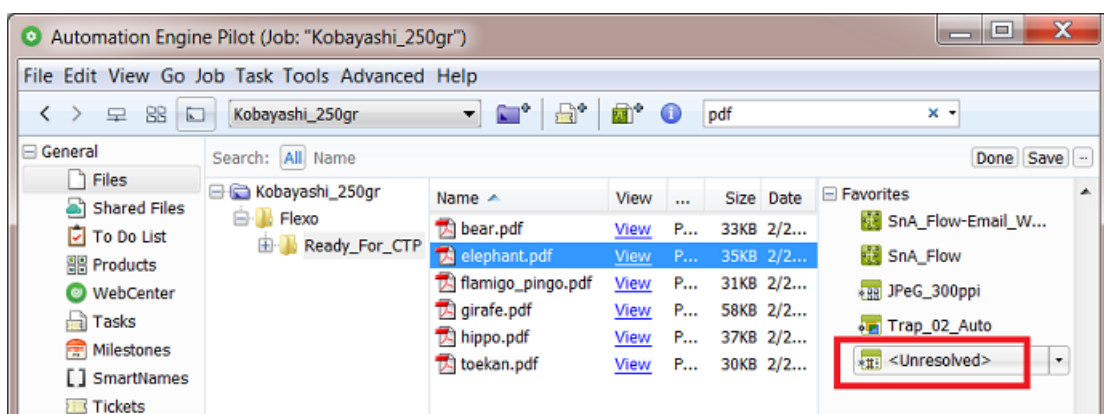
Click **Check** to test what that Name (or Type) field will resolve to. That is the name of the ticket that will be picked up. So make sure a ticket with that exact name (and of that type) does exist.



After you click **OK** to close this dialog, the pane of Favorites in your Pilot will show what the actual name is for this ticket, in this case based on the job it belongs to.



When the SmartName's **Resolved Value** does not match an existing ticket name, then that favorite will be shown as **<Unresolved>**. It is unusable, not in this specific context as required by this SmartName.



3.3.3. Launching Favorite Workflow Tickets

There are various ways to launch a favorite (workflow) ticket:

Launch Immediately

1. Select file(s).
2. Hold the **Alt** button.
3. Click on your favorite. The task is launched.

Choose a Ticket of the Same Task Type and Launch Immediately

1. Select file(s).
2. Hold the **Alt** button.
3. Click on the right-side of a favorite to open the drop-down list of tickets of the same task type.
4. Move your mouse down to the chosen ticket.
5. Click on the chosen ticket. The task is launched.
6. Release the **Alt** button.

Open the Favorite Ticket in the Workflow Editor

1. Select file(s).
2. Click on your favorite.
3. In the workflow editor, check or change the settings and then launch or 'launch and monitor'.

Choose a Ticket of the Same Task Type and Open it in the Workflow Editor

1. Select file(s).
2. Click on the right-side of a favorite to open the drop-down list of tickets of the same task type.
3. Move your mouse down to chosen ticket.
4. Click on the chosen ticket.
5. In the workflow editor, check or change the settings and then launch or 'launch and monitor'.

3.4. Building and Launching a Workflow On The Fly

If you build a workflow from the **Tickets** view, you cannot launch it until you finished building it. Your Workflow Editor will not have a **Files** pane to select files and then launch it. Working this way is typically something done by Automation Engine administrators while building the workflows. Once the workflow is ready for testing, users can then go to the **Files** view, make a file selection, select the workflow ticket and launch it.

However, you do not always have to prepare a workflow ticket before you can launch it. It is possible to build the workflow and launch it on a file without first saving the workflow ticket. Follow these steps to do this:

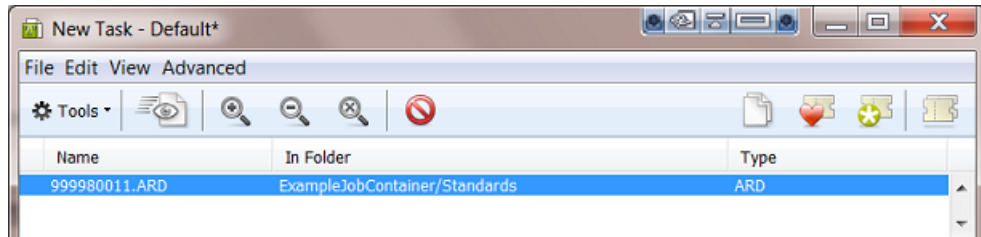
1. In the **Files** view, select the file(s) you want to launch your workflow on.



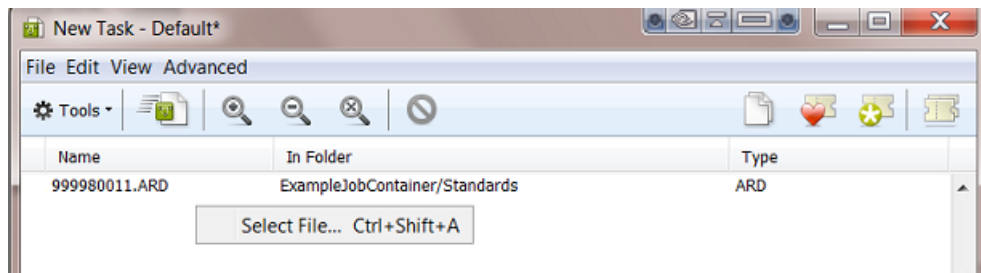
Note: If you need to select files from different (Job) folders, then first select those in one same folder, then choose **New Task** or **New Workflow** to open a Workflow Editor and there use the **Select File** tool to select the files from any other folders (see step 4).

2. Click to open the workflow editor.

You can see the file(s) you selected in the **Files** pane of the Workflow Editor.



3. Build your workflow and define the settings of every step.
4. If you want, select more files. Right-click in the **Files** pane and choose **Select File...**



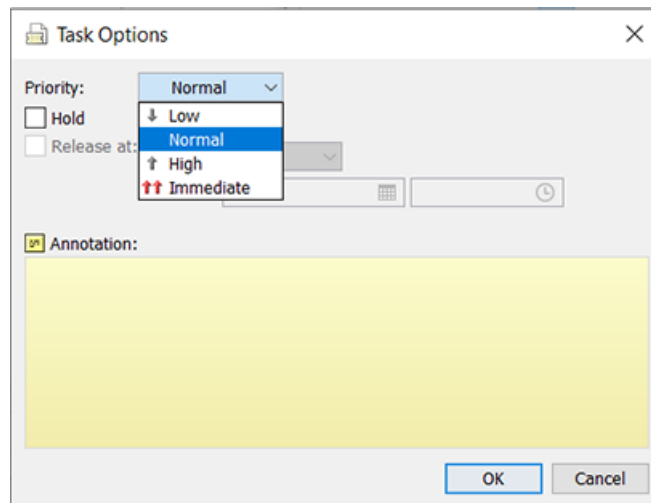
In the dialog that appears, select files from folders in **Jobs** or **Containers**.

5. Click the Launch button. Hold **Alt** to **Launch and Monitor** and keep the workflow editor open.
6. If you kept the workflow editor open, continue by changing the workflow settings, *relaunching* or launching on different files. When ready, save your workflow ticket.

3.5. General Workflow Task Options

Task Options are options that will be used when you push the **Launch** button (both here in the workflow editor and also in a single task ticket). These settings are not saved into the (workflow) ticket.

Open **Task Options** from the menu **Advanced** in the workflow editor:



- **Priority:** Choose the general priority with which you want to launch this workflow. The default is **Normal**.



Note: To choose **High** or **Immediate**, you need a special [User Access Right](#).



Note: You can also choose to set the priority of only a part of the workflow. Learn more in [Set Priority](#) on page 92.

- **Hold:** Check this when you want to hold the workflow when you launch it. The workflow will start but immediately go into 'pause' mode. You can then later decide when to use the **Release** button to start it for real.
- **Release:** When you decided to use **Hold**, you can set a time when the workflow should be released automatically. Choose a relative or absolute time.
- **Launch a separate task per input file:** Choose this option when you selected multiple input files but when you still want to launch a separate workflow task per input file.
- **Annotation:** When testing many workflow tasks or many versions of a workflow, it can be useful afterwards to read some notes that you added before launching it. Once the task was launched, you can read these annotations from this same menu or you can read them in the Tasks view when you right-click the main workflow task line (in bold).

For example: "This task was only launched to create the temporary files that I need to test the next one."

3.6. Relaunching Your Workflow with Different Settings

You can relaunch a workflow from the **Tasks** view and from the **Workflow Editor** (skip step 1):

1. In the Tasks view, double-click the main **Workflow** line to open your workflow in the workflow editor.

1236	banana.pdf,banana_report.pdf	Zip	100%	✓	11/20/14 4:20 PM
1236	lemon.pdf,lemon_report.pdf	Zip	100%	✓	11/20/14 4:20 PM
1236	lemon.pdf	Create Report (ReportMaker)	100%	✓	11/20/14 4:20 PM
1236	banana.pdf	Create Report (ReportMaker)	100%	✓	11/20/14 4:20 PM
1236	lemon.pdf	Optimize and Clean (PSFix)	100%	✓	11/20/14 4:20 PM
1236	banana.pdf	Optimize and Clean (PSFix)	100%	✓	11/20/14 4:20 PM
1236	banana.pdf,lemon.pdf	Workflow	100%	✓	11/20/14 4:20 PM



Tip: The workflow can be opened and relaunched even when it is still running.





Tip: If you double-click any sub-step of the workflow, only that ticket will open. You can also relaunch just that separate single step (if the input files are still available).

- In the workflow editor window, click the **Edit Workflow** button  if you want to make any changes.

When you are finished making changes, and if you want to keep the new settings before you try them out, save your workflow now.





Note: As soon as you make one change, the name of the workflow ticket in the top bar will show an asterisk * behind the name to indicate that it has been changed.

- Click Launch  or press **Alt** while clicking Launch to **Launch and Monitor** .

3.7. Diagnosing a Workflow in the Tasks Pane

You can check the processing status of a workflow in the Pilot's **Tasks** pane. The **Tasks** pane appears in the **Files, Products** and **Tasks** View.






Job Name	File Name	Task Type	Progress	Phase	State	Launched
1236	lemon.pdf	Create Report (ReportMaker)	0%	Reading \\...		11/20/14 4:30 PM
1236	banana.pdf	Create Report (ReportMaker)	0%	Reading \\...		11/20/14 4:30 PM
1236	lemon.pdf	Optimize and Clean (PSFix)	100%		✓	11/20/14 4:29 PM
1236	banana.pdf	Optimize and Clean (PSFix)	100%		✓	11/20/14 4:29 PM
1236	banana.pdf,lemon.pdf	Workflow	0%	Running		11/20/14 4:29 PM



Attention: The **Tasks** pane does not show the **Workflow Controls** that may be part of your workflow. To diagnose them, use the Workflow Editor.

Workflow Status

The workflow status can be

- : running, when one or more steps are waiting to be handled.
- : success, when all steps finished successfully.
- : warning, when one or more steps finished with a warning.
- : failed, when one or more steps finished with an error.
- : cancelled, when a user cancelled the processing.

Right-Click and Diagnose Further

Right-click or Cmd-click any task line to see many functions available for that task or its related files.

The availability of the functions depends on

- if the task is already finished or not
- if the task is the initial 'workflow' task type or any sub-step of that workflow.

Job Name	File Name	Task Type	Progress	Phase	State	Launched
1236	lemon.pdf,lemon_report.pdf	Zip	100%		✘	11/20/14 4:30 PM
1236	banana.pdf,banana_report.pdf	Zip	100%		✘	11/20/14 4:30 PM
1236	lemon.pdf	ReportMaker	100%		✔	11/20/14 4:30 PM
1236	banana.pdf	ReportMaker	100%		✔	11/20/14 4:30 PM
1236	lemon.pdf	an (PSFix)	100%		✔	11/20/14 4:29 PM
1236	banana.pdf	an (PSFix)	100%		✔	11/20/14 4:29 PM
1236	banana.pdf,lemon.pdf		100%		✘	11/20/14 4:29 PM
1236	banana.pdf,banana_report.pdf		100%		✔	11/20/14 4:20 PM
1236	lemon.pdf,lemon_report.pdf		100%		✔	11/20/14 4:20 PM
1236	lemon.pdf	ReportMaker	100%		✔	11/20/14 4:20 PM
1236	banana.pdf	ReportMaker	100%		✔	11/20/14 4:20 PM
1236	lemon.pdf	an (PSFix)	100%		✔	11/20/14 4:20 PM
1236	banana.pdf	an (PSFix)	100%		✔	11/20/14 4:20 PM
1236	banana.pdf,lemon.pdf		100%		✔	11/20/14 4:20 PM
WDETESTS	78999_ABC_report.pdf		100%		✔	11/20/14 4:08 PM
WDETESTS	999980011.ARD		100%		!	11/20/14 3:13 PM
WDETESTS	78999_ABC_report.pdf		100%		!	11/20/14 2:25 PM
1237	South.pdf	Checkpoint)	100%		✔	11/20/14 1:48 PM
1237	South.pdf	Checkpoint)	100%		✔	11/20/14 1:48 PM
1237	South.pdf		100%		✔	11/20/14 1:48 PM
WDETESTS	78999_ABC.pdf	Checkpoint)	100%		✔	11/20/14 1:48 PM
WDETESTS	78999_ABC.pdf	Checkpoint)	100%		✔	11/20/14 1:48 PM
WDETESTS	78999_ABC.pdf		100%		✔	11/20/14 1:48 PM
WDETESTS	banana_report.pdf	Checkpoint)	100%		✔	11/20/14 1:28 PM
WDETESTS	banana_report.pdf	Checkpoint)	100%		✔	11/20/14 1:28 PM
WDETESTS	banana_report.pdf		100%		✔	11/20/14 1:28 PM
WDETESTS	banana_report.pdf	Checkpoint)	100%		✔	11/20/14 1:27 PM
WDETESTS	banana_report.pdf	Checkpoint)	100%		✔	11/20/14 1:27 PM
WDETESTS	banana_report.pdf		100%		✔	11/20/14 1:27 PM
WDETESTS	banana_report.pdf	base	100%		✘	11/20/14 12:14 PM


These functions are documented in [Functionality on items in Tasks View](#).

Some specific notes:


- **Delete.** If you decide to manually delete tasks, then you best delete the main workflow task step so that all related task steps are also cleaned up. In general we advise to use the automated version based on the standard **Task Cleanup Rules** that you set up in the Configure tool.
- **Move to History.** Same comments as above for the **Delete** function. Learn more about storing task details in [Task History](#).

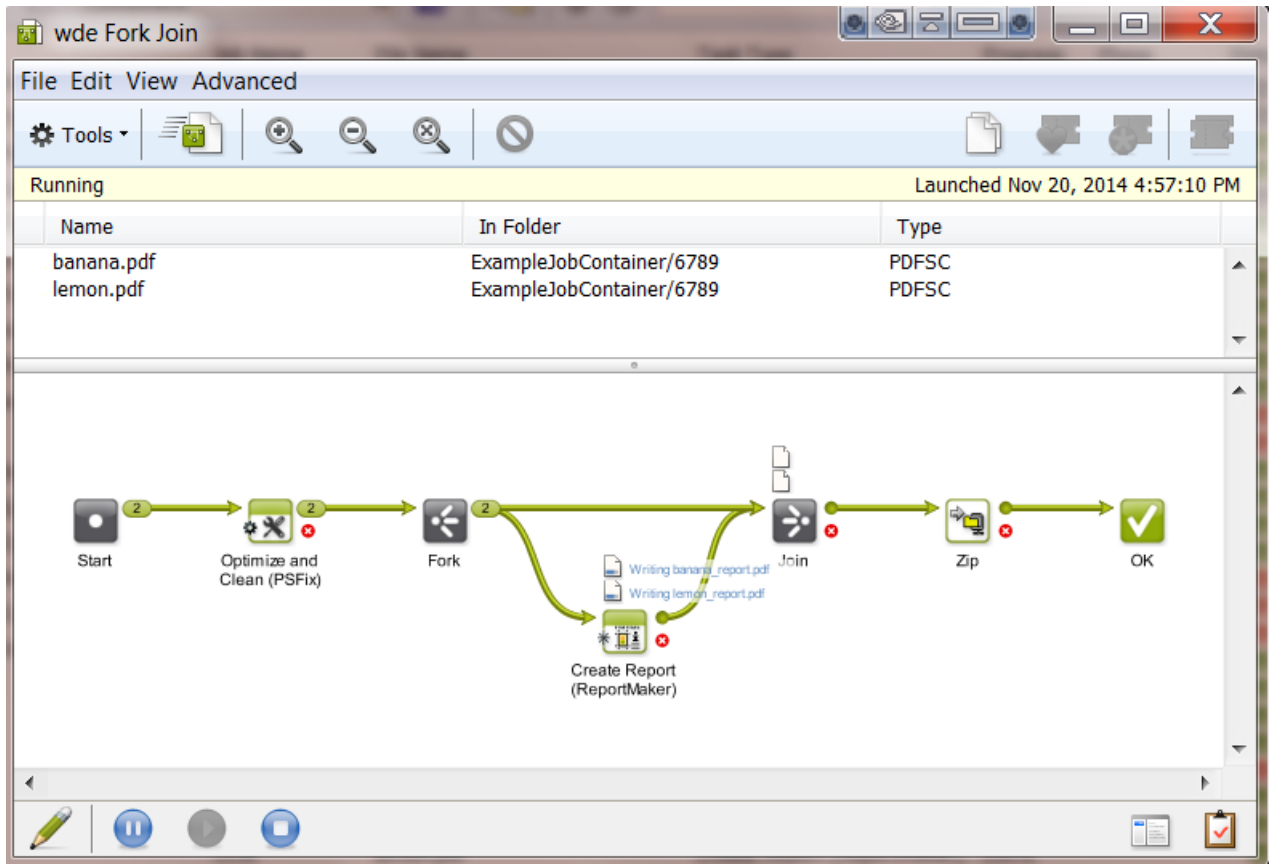
3.8. Diagnosing a Workflow in the Workflow Editor


General View

In the workflow editor, you can see your file(s)  going from step to step, with their processing **Phase** displayed in blue text.




Note: This icon  can represent a single file or a collection of related files.



- Click  to see the **Details** of the main workflow task (names of output files, stranded files, reading of workflow parameters...).



Tip: If you want the details of a specific workflow step, then first select one of the output files of that step and then right-click and choose **Details**.

- Click  to see the **To-Do** action items associated with this workflow.

Output Pins and Widgets

A number in the output pin of a step indicates the number of files that ran through it.

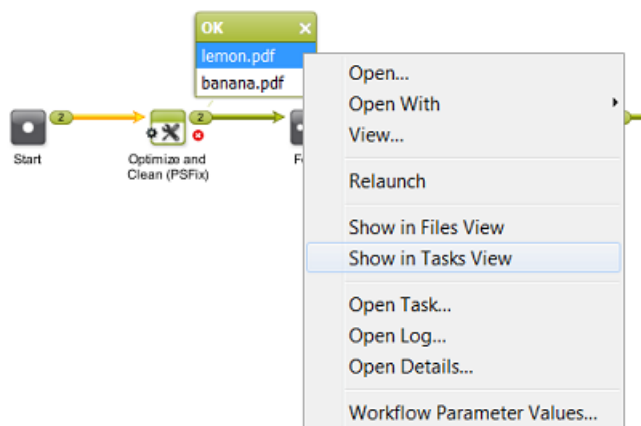


Click on the number to open a widget showing the file names.



Note: The items in these output widgets are actually more than files, technically we name them tokens. A token could consist of one file, or many files or even be empty. Learn more technical detail in [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

Inside the widget, right-click the name of a file (token) to see more options. Many of these options were explained in [Diagnosing a Workflow in the Tasks Pane](#) on page 62.



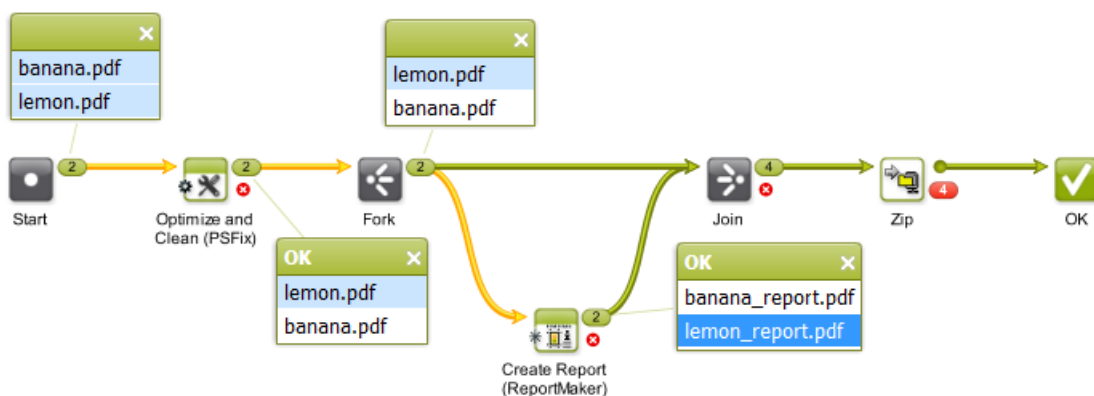
Note: You can also right-click the icons of files that are still being processed and get more options there.



Note: To check the **Workflow Parameter Values** of a (collection of) file(s): learn more in [Using Workflow Parameters](#) on page 129.

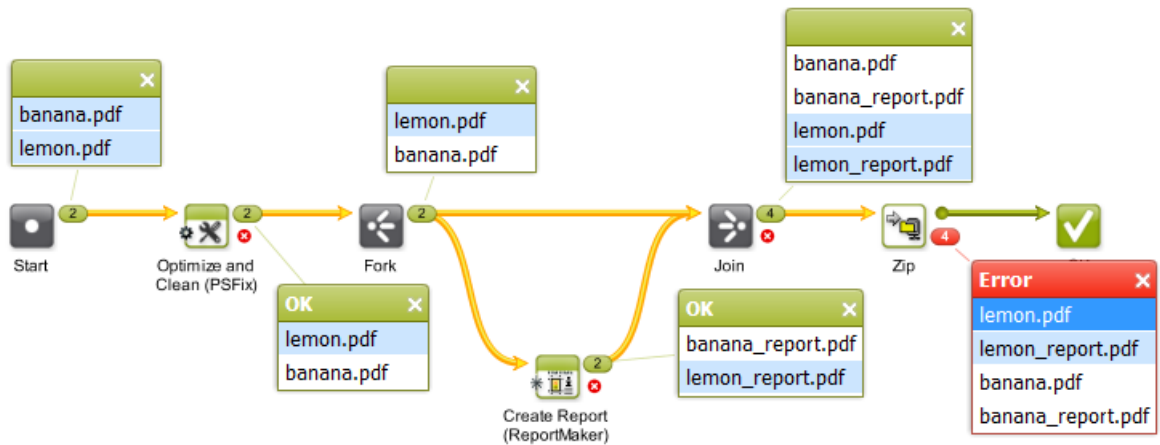
Seeing the Followed Path


In an output widget, click on a file name to see which path it followed. In this example we clicked on `lemon_report.pdf`. The name is highlighted in blue and the path that this file followed so far is shown in yellow.



Seeing the Related Files

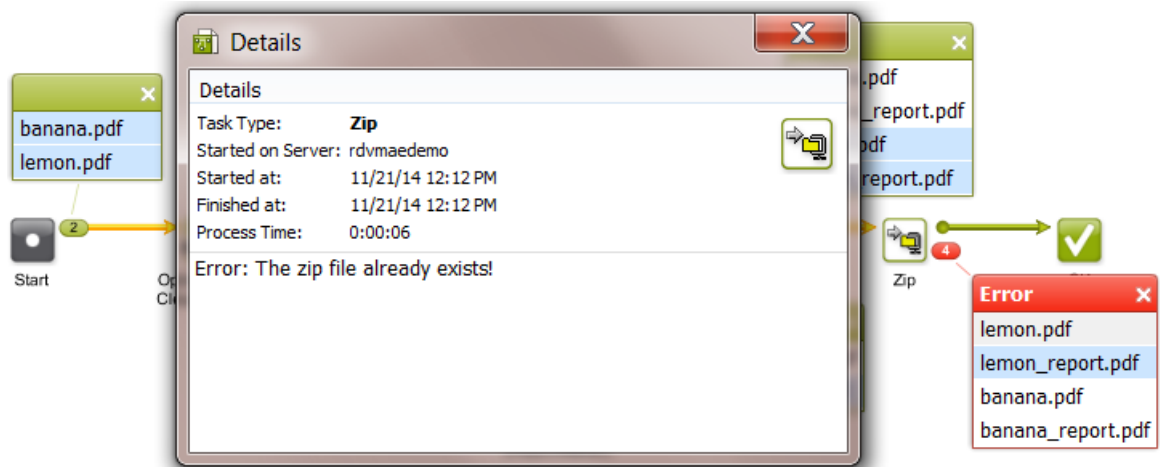
Clicking on a file name also helps diagnosing the relations between files. In below example we clicked on `lemon.pdf` in the 'Error' widget. This highlights all related files in a light blue color. These relations were created during their processing.



 **Note:** Files that are related to each other in a workflow will also share the same workflow parameters at that specific step of the workflow. Learn more in [Using Workflow Parameters](#) on page 129.


Diagnosing an Error



In below example, we right-clicked on a file in the 'Error' widget and asked the **Open Details ...**. This explains the error: the zip was not created because it already exists (and the ticket had the setting to fail in that case). Sometimes, you will need to **Open Log ...** to find what you are looking for.



Details is meant for the user and is translated. **Log** info is mainly for Esko service people, it is more complex and also not translated.

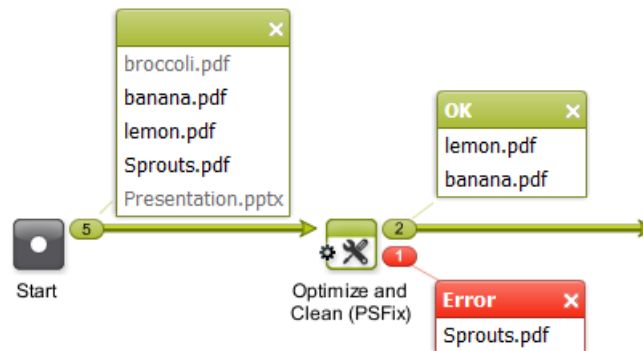
In this example, you could now continue with these steps:

1. Click on  to edit the workflow.
2. Double click the **Zip** task, change the setting that generated the error and click **OK** to confirm and close the ticket.

3. Launch this changed workflow: click  or press **Alt** and click .
4. If you prefer this new setting, save your workflow.

Error Files or Invalid Files

See the open widgets of this example workflow:



- In the widget of the **Start** step, `broccoli.pdf` and `Presentation.pptx` are shown in grey because they are not valid for the next step in the workflow.
 - `broccoli.pdf` is invalid because it has a size of 0 Kb.
 - `Presentation.pptx` is a PowerPoint file and that is not a valid input file type for the next step.



Note: Both file names would be shown in black when the next step would for example be a **Copy or Move File** task.

- `banana.pdf` and `lemon.pdf` are valid PDFs.
- `Sprouts.pdf`: Based on its extension '.pdf', the file was valid for the next step but it still resulted in an error (the task detected that the file was not really a PDF file).

3.9. Understanding Tokens (Grouping of Output Files)

Concept

Automation Engine internally decides how files are grouped as they progress through a workflow. When a workflow step has more than one input file, it then depends on what workflow step it is whether those files are handed over to the next step as separate files or as a group.

Automation Engine groups output files by attributing them a same **token** (an internal ID number). This token is not shown in the user interface. Files will seem grouped because they were attributed a same token. The next step in the workflow receives one or multiple such groups from the previous one. The next workflow step(s) typically starts a task per such group, technically 'per token'.

This internal behavior of the software is visualised in the [widgets](#) that list the output files of a workflow step. You can also see how the system works with groups or tokens in the list of **Tasks**.

Example

An example:

File Name	Task Type	Progress	Current Step	State	Launched
600_banana juice.pdf	Optimize and Clean (PSFix)	31%	Saving file \\leaw16d1267\WDE-AE-DATA\Graphics\JuiceCo\600_banana juice_Clean.pdf	Running	8/31/17 4:47 PM
601_carrot juice.pdf	Optimize and Clean (PSFix)	31%	Saving file \\leaw16d1267\WDE-AE-DATA\Graphics\JuiceCo\601_carrot juice_Clean.pdf	Running	8/31/17 4:47 PM
607_mango juice.pdf	Optimize and Clean (PSFix)	0%	Reading \\leaw16d1267\WDE-AE-DATA\Graphics\JuiceCo\607_mango juice.pdf	Running	8/31/17 4:47 PM
607_mango juice.pdf, 601_carrot juice.pdf, 600_banana juice.pdf	Optimize PDF Separations	61%	Saving Document	Running	8/31/17 4:47 PM
607_mango juice.pdf, 601_carrot juice.pdf, 600_banana ju...	Workflow	0%		Running	8/31/17 4:47 PM

- Each task type or workflow control has its own way of attributing tokens:
 - This workflow is launched on 3 input files. The step **Start** always attributes them the same token and so hands them over to the next step(s) as one group containing of 3 files. See how selecting one of them in the widget (dark blue) also highlights those that have the same token (light blue).
 - The workflow splits off into 2 steps that process this group differently: The step **Optimize PDF Separations** only shows one icon of a token, while the step **Optimize and Clean (PSFix)** shows 3 such tokens. This different behavior is decided in the software ; it is not a task setting you can change.
- The list of **Tasks** shows how a task is started for each token.
- In the widget that shows the output of the **Optimize PDF Separations** task, selecting one of its items shows that the 3 output files are still grouped. They still have the same token. This type of task did not split the group: it keeps them grouped for a next step.
- In the widget that shows the output of the **Optimize and Clean (PSFix)** task, selecting one of its items shows that the 3 files are no longer grouped. This is because this type of task always attributes its output files an individual token: they will be handed over to a next step as separate items.



Note: An item in a widget is not always a file. It can also be a folder or even only some internal number.

Controlling this grouping - Controlling how tokens are attributed

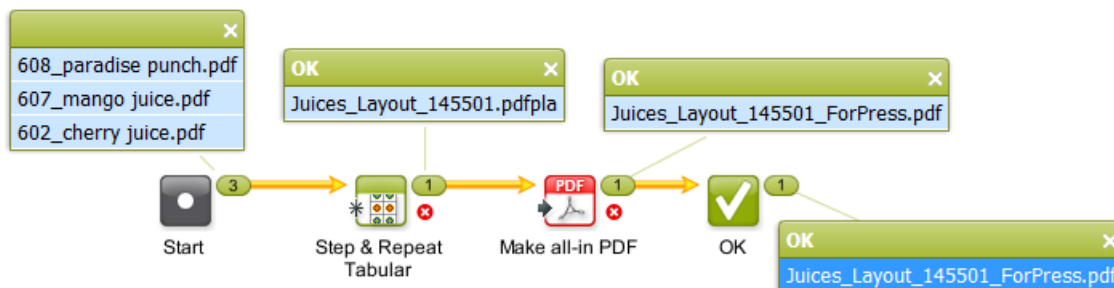
Sometimes you want control over these groups. You may want to change the built-in behavior.

Some examples:

- When you want separate files in stead of a group.

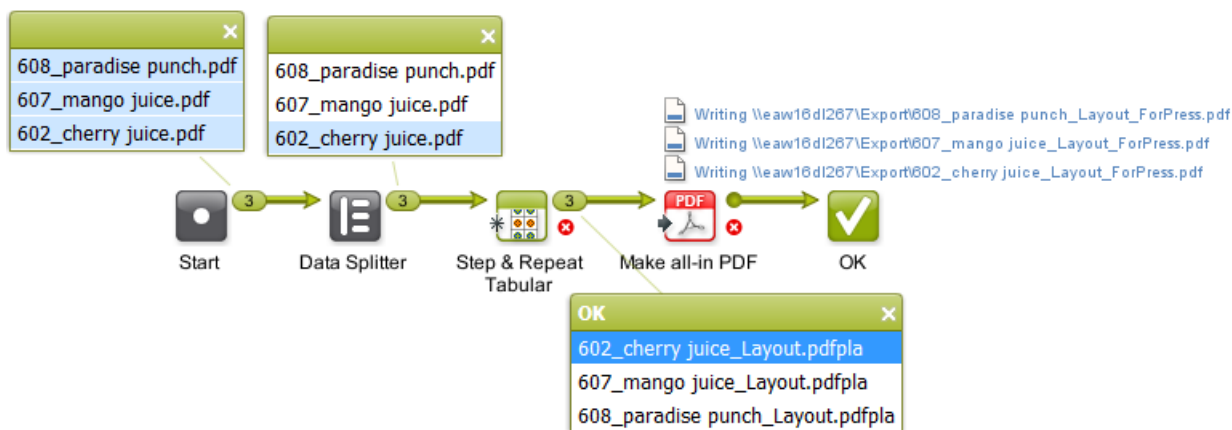
For example: When your workflow starts with a Step & Repeat task.

- With this type of task, you will get this result by default:



The S&R task here receives them as a group and so combines them on a same layout.

- If you do not want this (if you want a layout per input file), you can solve this by adding a **Data Splitter** in front:



- When you want to overrule the built-in grouping logic.

Some examples:

- When you want specific grouping of RIP output files before they become input files for the task **'Prepare for Viewer'**. This grouping decides how the Viewer will open them.

Learn more in [Data Splitter](#).

- When you want to control which files should be ZIP'ped or FTP'd together.

Learn more in [Data Collector](#).

- When you want to collect files that were created from a source file in that workflow.

Learn more in [Fork and Join](#).

About Empty Tokens

Automation Engine v18.1 started an important change in the behavior of AE workflows:

In case an output pin of a task or workflow control has no output file, a workflow can now continue with a so called 'empty token', where in previous versions, the workflow would have stopped.




This change was made to avoid that a workflow stops ('stranding') where it did not make sense to stop. Workflow steps that do not really need an input file, should not be blocked from starting when there is no such file.



Attention: This means that after an upgrade to 18.1, some workflows can now continue where before they used to stop. So check your workflows after this upgrade ; you might have to adapt some!

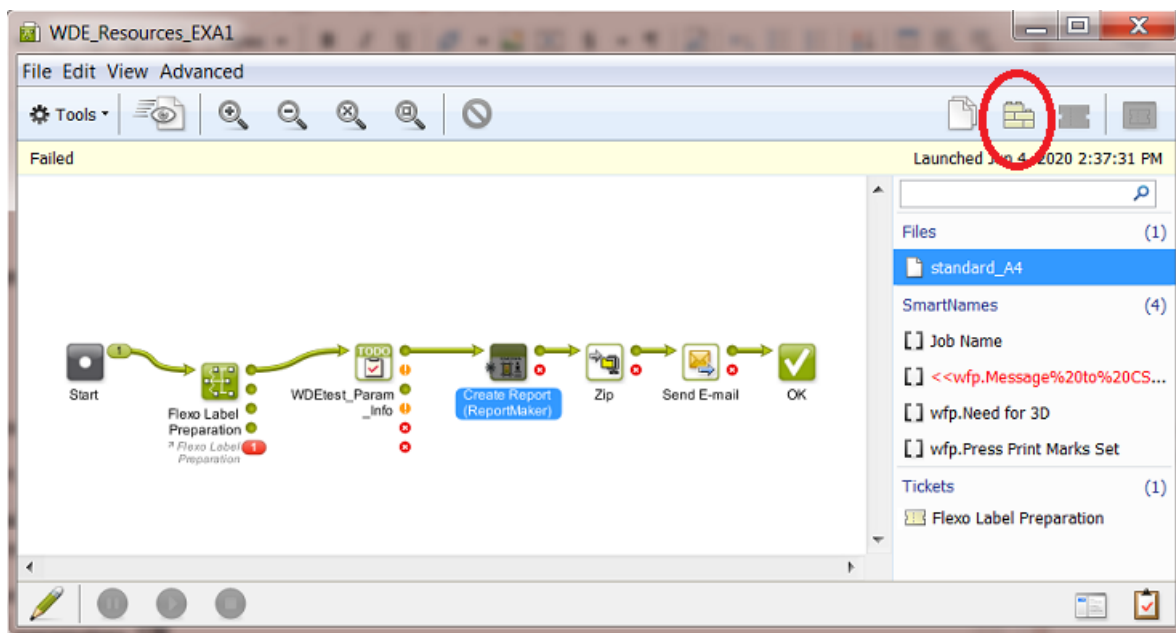
3.10. Checking the Resources your Workflow contains

Concept

In the workflow editor, the **Resources** pane allows to show and search the resources in your workflow. Click  to show or hide this pane.



Note: This resources pane is also available when you open a workflow in the **Tickets** or the **Tasks** view.



A workflow can use many different kind of resources. This tool can detect these resources:

- Files
 - Create Report (ReportMaker) template file. In above example, this is the template file used in the ReportMaker step.
 - Create PDF Report template file.
- SmartNames
 - All types, including public parameters, workflow parameters, ...
 - Workflow parameters appear in red when they are requested in a ticket but missing in the list of workflow parameters of this workflow.
- *Referenced* workflow tickets
- Marks sets
 - Dynamic Marks
 - SmartMarks
- Action list files
 - PDF Action List
 - ArtPro Action List
 - PitStop Profile
 - PitStop Action List
- CAD resource files
 - CAD Style
 - CAD DXF Tune
 - CAD Tune
- GlobalVision inspection tasks resource files
 - Compare Barcode Profile

- Braille Language
- Spelling Language
- Compare Artwork Profile
- Compare Text Profile

- Other resource files
 - Notification Template
 - Script files (.bat, etc.)
 - Tiling Template
 - Cutting Flow
 - Print Rule Set




Using the Resources pane

- When a same resource is used in several workflow steps, for example a SmartName, it will also be listed several times.
- When you select a resource, the workflow step where it is used is also selected and scrolled into view.
- When you click on an already selected resource, an animation draws extra attention to the workflow step where it is used.
- When you double-click a resource in the list, the workflow step where it is used is opened and ready to be edited.
- SmartName parameters that are missing within this workflow appear in red, typically <<pp . or <<wfp . parameters.

- Use the filter in the resources pane to filter the list of resources. The entered text will be looked for in the resource name, resource category. For SmartNames it also takes the ID of the SmartName into account.
 This helps to quickly answer the FAQ "Where is SmartName <<xyz>> used in this workflow?"
- When changes are made to the workflow, the list of resources is immediately updated to reflect the new situation.
- You can edit custom SmartNames directly from within the resources pane. Right-click the SmartName resource and select **Edit SmartName...** You can edit and save the SmartName in that dialog.

3.11. Pausing or Cancelling Your Workflow




While monitoring your file's progress in the workflow editor, you can:

- click  to pause your workflow (at the end of the task that is currently running),
- click  to make your paused workflow run again,
- click  to cancel your workflow's processing.

4. Workflow Controls

Workflow Controls are steps in a workflow. Instead of executing a task on an input file, they make workflow decisions on routing, file selection, file order etc.

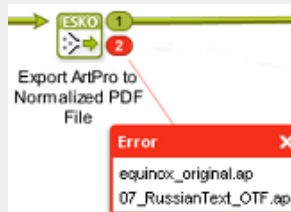
-  **Start:** Every workflow needs to start with this step. By default it is present on every new workflow canvas.
-  **Modify Workflow Parameter Values:** Use this to modify workflow parameter values at some point during the workflow. Learn more in [Modify Workflow Parameter Values](#) on page 75.
-  **Select Referenced File:** Use this to easily select files that are referred to in an XML file. Learn more in [Select Referenced File](#) on page 89.
-  **Set Priority:** Use this to assign a different priority to parts of your workflow. Learn more in [Set Priority](#) on page 92.
-  **Sort:** When the order of the input files is important in the next workflow step, use this one to manage that order. Learn more in [Sort](#) on page 93.
-  **Router:** This step enables automatic routing of files based on many possible criteria. Learn more in [Router](#) on page 79.
-  **Mark File:** Use this step to mark files that you will select later in the workflow using the **Select Marked File** workflow control. Learn more in [Mark File and Select Marked File](#) on page 96.
-  **Select Marked File:** Use this step to select the files that you marked earlier in the workflow using the **Mark File** workflow control. Learn more in [Mark File and Select Marked File](#) on page 96.
-  **Data Splitter:** Use this step when you want to send the files one by one to the next task instead of grouped. Learn more in [Data Splitter](#).
-  **Data Collector:** Use this step to collect all incoming files before sending them to the next task as a group. Learn more in [Data Collector](#).
-  **Fork:** Use this to tag incoming and related files with a group tag. This is useful in special cases where you want detailed control over how Automation Engine groups files in a workflow. Learn more in [Fork and Join](#).
-  **Join:** Use this to select the files tagged as a group by the **Fork** workflow control. Learn more in [Fork and Join](#).
-  **Cancel Workflow:** When a file is sent to it, this step cancels the processing of the entire workflow. This is useful when you also want to stop not just the branch of the workflow where this step is used, but all branches. For example: when the previous step **Manage Product Status** results in 'not approved' then the next step is **Stop**.

-  **OK:** By default, there is already one **OK** step in the canvas. You can add more and route files processed successfully to the appropriate **OK** step. When a subworkflow ends with multiple OK's, they will show up in the master workflow as multiple green output pins.
-  **Warning:** Add this step at the end of your workflow to route files that generated a warning during processing. For example preflight tasks or approval processes typically also offer a warning as end type.
-  **Error:** Add this step at the end of your workflow to route files that generated an error during processing.



Note: A task that has one or more files ending with an error automatically triggers an error status for both that task and the whole workflow. That is why it is not necessary to add this **Error** workflow control and connect to it from the error output pin of a task.

To see which files caused this error, you can also click on the error output pin and open the output widget:



Learn more in [Diagnosing a Workflow in the Workflow Editor](#) on page 63.

4.1. Launch Workflow

Concept

Using this workflow control to launch workflows is only needed or advised in special circumstances, maybe also on the advise of Esko support.

This workflow control is not to be compared with adding a sub-workflow in your main workflow (either as a copy or as a reference). The difference is that this workflow control launches the selected workflow in a 'fire-and-forget' mode (aka 'asynchronous').

This principle of 'fire-and-forget' means that the main workflow (the one that contains this workflow control as a workflow step) will

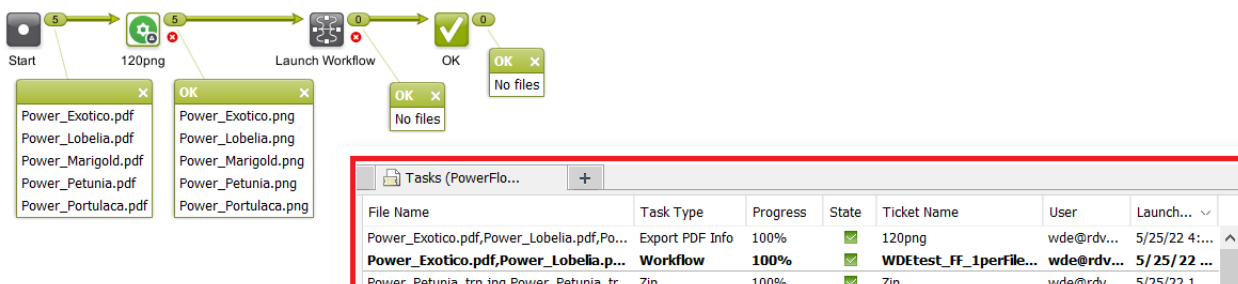
- not wait for nor collect any outputs of the workflow that this workflow control launches. This workflow control will also not have any output files, only an empty token.
- not check any execution statuses of the workflow launched with this workflow control.

This allows the main workflow to immediately continue with any other workflow steps.

Launching workflows with this tool is a good idea if you struggle with large workflows that result in too much unnecessary load on your system. Using this tool will then have a positive effect on your system stability.

The disadvantage is of course the 'forget' part: there is no tracking nor trace of what was launched with this tool.

For example, in the Tasks view, this step (just like all workflow controls) is not shown as a task step of the main workflow:



Settings

- **Workflow:** Select the workflow ticket that you want to launch in this mode.
- **Launch a separate workflow per input file:** When selected, a separate workflow will be launched per input file.



Note: This effect of this option can not be seen visually, neither in the Workflow Editor nor in the Tasks view.

Technical Details

- As soon as the selected workflow was launched successfully, an empty token will *instantly* appear on the **OK** output pin.
- When the selected workflow encountered errors while being launched (for example a SmartName that can't be resolved), the input files will be placed on the **Error** output pin.
- When using a SmartName for the name of the workflow to launch, be aware that this field is only calculated once. So if it is needed to calculate this SmartName per input file, you need to use the Splitter workflow control as the workflow step before this one.
- The selected workflow is always run without any Job context, even when the main workflow was run in a Job context. This means that you can't use Job-related SmartNames inside that workflow.
- When the here launched workflow has a value for a workflow parameter, it will be overruled by the value of that same workflow parameter that is set by the main workflow at the moment this workflow control starts.
- The selected workflow is always launched with priority 'Normal'.
- This workflow control requires some kind of input file (a non-empty token).

4.2. Modify Workflow Parameter Values

This workflow control allows to modify values of workflow parameters during the workflow. Learn more about workflow parameters in [Using Workflow Parameters](#) on page 129.

Concept

The typical reason to modify a workflow parameter during a workflow is that there is new or changed information that was not there earlier or at the start of that workflow. Some example cases:

- The new or changed information is based on a user's decision. For example the decision how to release a to-do item. See an example below.
- The new or changed information is coming from an external system (and was not available earlier).




Note: Mind these exceptions:

- New information coming from WebCenter does not need this separate workflow control. The tasks that communicate with WebCenter offer their own built-in tool to map their output parameters to workflow parameters. Learn more in [Modifying Workflow Parameters During a Workflow](#) on page 139.
- When, during a workflow, an external system sends an updated XML with new or modified parameters, check if you can use the feature [Using Workflow Parameter Values from XML](#) on page 135. This feature loads that XML at every step of the workflow, not just at the start. This means that the incoming updated XML can trigger an updated workflow parameter without the use of this separate workflow control.

Using the workflow control 'Modify Workflow Parameter Values'

To use this workflow control, proceed as follows:

1. Select and drag the **Modify Workflow Parameter Values** workflow control onto your workflow canvas. Double-click it to edit.
2. Click **Add** to add workflow parameters or double-click existing ones to modify.
3. Use  to select an existing parameter or type the desired workflow parameter name and provide a new value. This can be a fixed value, a SmartName or an XPath expression that retrieves the new value from the workflow control's XML input. Learn more on Xpath in [The XPath Builder](#).

When the input file is a JSON file, you can here also enter a **JSONPath** expression. Learn more in [JSONPath Expressions](#) on page 78 .



Attention: As mentioned in [The XPath Builder](#) , adding SmartNames or JSONPath expressions to your XPath expression is not supported when you are creating this kind of an 'inline' XPath expression.

4. Click **Add** to confirm.



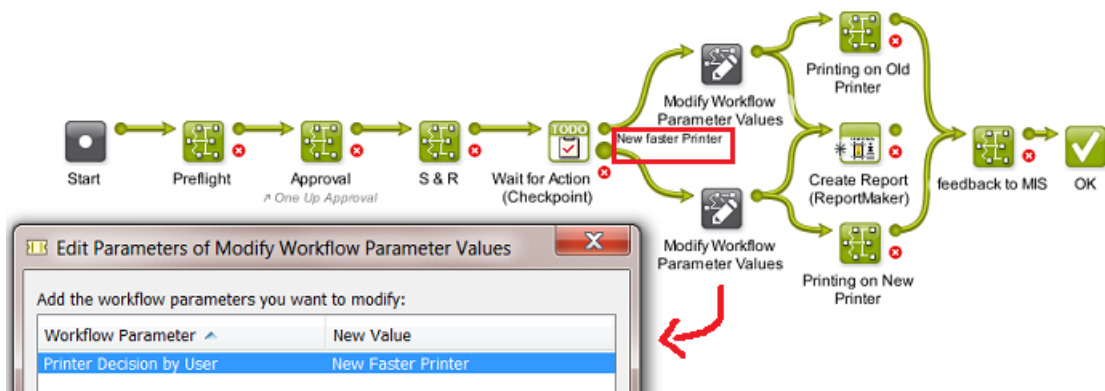
Note: When the parameters list contains a new one, you will be asked to **Confirm**.

5. Click **OK** to close. Save your workflow.

Example: Capturing the To-Do decision as a Workflow Parameter

Our workflow asks a user to decide if the output will be done on the old or on the new printer. This decision should also be mentioned on the Report PDF.

- See how the Checkpoint task output Pin 'New faster Printer' routes to the workflow control that sets the value of the workflow parameter 'Printer Decision by User' to 'New Faster Printer'.




- The ReportMaker template needs to pick up that workflow parameter. Those templates are created in PackEdge. Because PackEdge SmartMarks can not pick up workflow parameters, you first need to create a SmartName of that workflow parameter and then use that SmartName in the template. Follow these steps to do this:
 1. In Pilot, in the SmartNames View, in **Include SmartNames From**, check the option **A workflow** and select the workflow that contains the desired workflow parameter.
 2. Click **[+]** to create a new SmartName.
 3. Set the Scope to **Global** and the SmartName Type to **String Extract**.
 4. Click **+ [i]** **Insert SmartName** and select the category **Workflow Parameters**.
 5. Select the desired workflow parameter [wfp.] and click **Insert**.
 6. Save the SmartName.
 7. Open the ReportMaker template in PackEdge and open the **SmartMark** palette.
 8. Select **Text Mark** from the **Add SmartName** palette.
 9. Click the **Smart Text...** button, select the category **SmartNames (Global)** and select the SmartName you just created.
 10. Save the ReportMaker template
 11. Use this template in the ReportMaker task. The resulting PDF can now mention what the user decided in his to-do item. Here is an example of how this can look like:

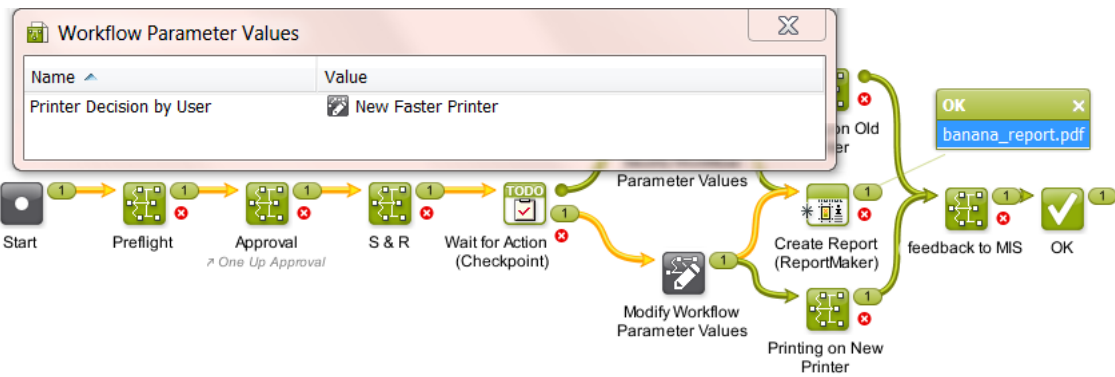



Visualizing the Workflow Parameter Values of a Step Output

To visualize the Workflow Parameter Values of a selected file, you can:

- choose the menu item **Advanced > Workflow Parameter Values**. This dialog stays opens while you click on output files.
- right-click on the desired step output and select **Workflow Parameter Values**.

Using above mentioned example, see the workflow parameter values of the Report PDF. The small icon  indicates that the value was modified during the workflow:



 **Note:** The set of workflow parameters can be different for each step output. For more info, see [Using Workflow Parameters](#) on page 129.

4.2.1. JSONPath Expressions

Concept

When the input file is an XML, you can directly type in an XPath expression in some fields of the task's options. Those fields also offer to click on **XPath** to open a panel to help you build that XPath expression with the correct syntax.

In the exact same way, when the input file is a JSON file, you can there type in a JSONPath **Expression**. And when you click on **JSONPath**, the same panel opens to help you build that JSONPath Expression.



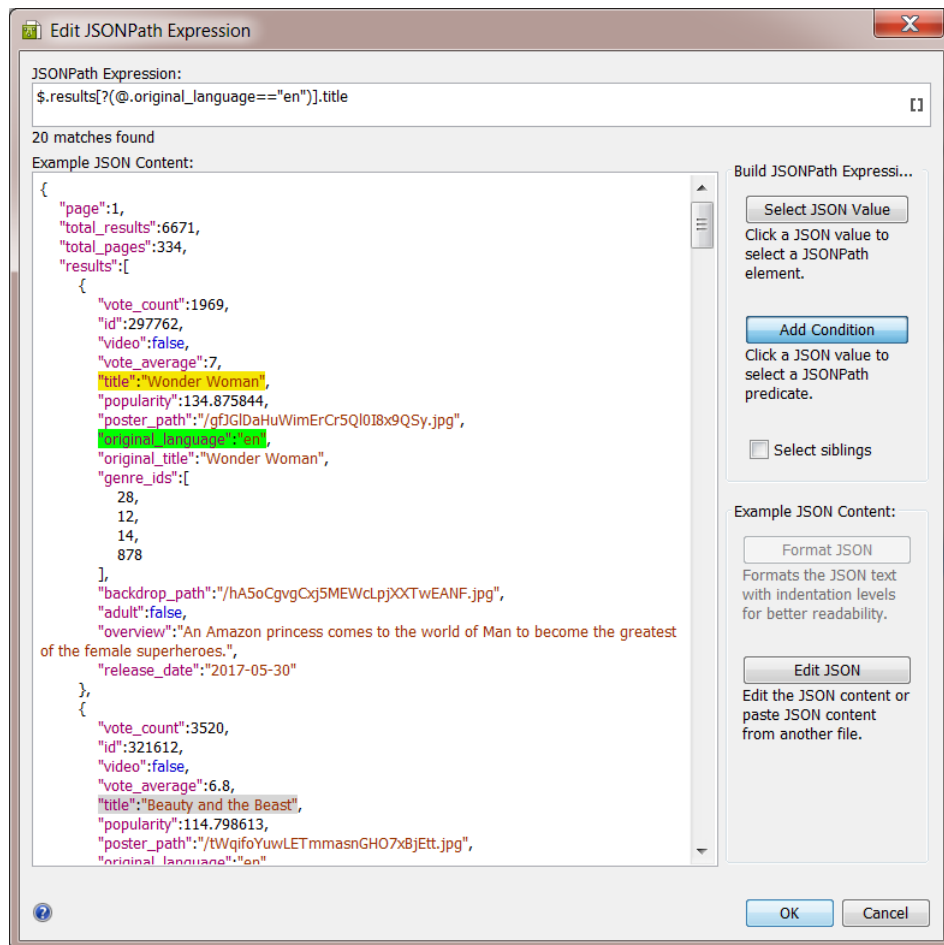
Attention: This button is not available in all tasks. It is only available in these workflow controls: [Modify Workflow Parameter Values](#), [Select Job](#) and [Select Referenced File](#).

Learn all about this panel that helps you build JSONPath expressions in [The XPath Builder](#). Its features are equally valid for JSON data.




Note: Unlike XPath, it is not possible to create JSONPath SmartNames.

Example

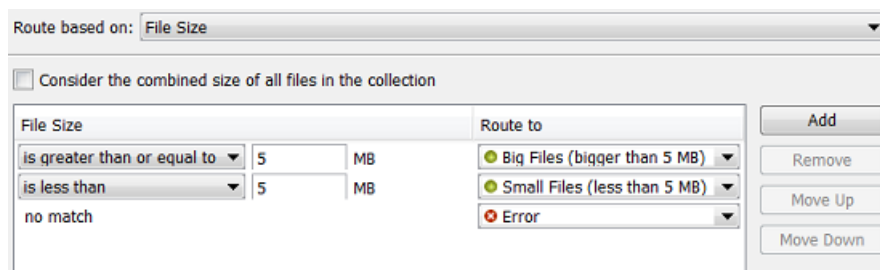


4.3. Router

The **Router** workflow control  serves to automatically decide the route that your workflow (files) should follow.

Let's take the example where a Router decides the workflow track based on the size of the incoming file. When the file is smaller than 5 MB, the file will be sent out via E-mail, otherwise via FTP.

- These are the settings in the Router:



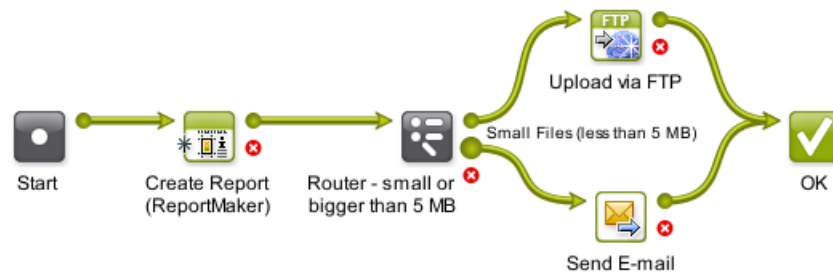
Two rules were defined and the **Route to** names were customized.



Note: The rules are executed from top to bottom. Their order can be changed with **Move Up / Move Down**.

- This is the workflow:

Notice how (when the mouse hovers over the Router step) the name of the middle output pin appears and how it matches the name that was defined in the **Route to** column of the Router settings.



Routing can be based on many different criteria (see the list below). Several of them offer these generic options:

- **Treat as numeric value:** When you select this, the options in the **value** column offer numeric possibilities (is less than, is greater than, etc.). This option is not there when it is obvious that the value is numeric.
- **Case sensitive:** Select this if you want to match the exact case of the value.



Note: If a value has to **match**, wildcards and *Regular Expressions* can be used in the value field.

Routing Based on Job Parameter Value

Choose this to route your file based on a Job Parameter. The **Parameter Name** drop-down list offers the parameters of the Job setup tab **Parameters** and also those of the tab **Step&Repeat**.



Note: You can also type in the parameter. This is not advised to avoid typing errors. However this can be useful when you build the workflow and router settings outside of a Job context in which you will be using it. In that case the parameters will not show up in the drop-down list yet.

An example:

The screenshot shows the Router configuration interface. At the top, 'Route based on:' is set to 'Job Parameter Value'. Below it, 'Parameter Name:' is 'Trapping needed'. There are two checkboxes: 'Treat as numeric value' (unchecked) and 'Case sensitive' (unchecked). The main configuration area has two columns: 'Job Parameter Value' and 'Route to'. The first row shows 'contains' with 'yes' leading to 'Trapping needed'. The second row shows 'contains' with 'no' leading to 'No Trapping needed'. A third row shows 'no match' leading to 'No Trapping needed'. On the right, there are buttons for 'Add', 'Remove', 'Move Up', and 'Move Down'.

Notice how you do not always need an Error route and how same routes can be chosen for more than 1 value.

Routing Based on Job Field Value

Choose this to route your file based on one of the Job setup fields listed in the **Field Name** drop-down list. The list also includes the custom **Job Categories** you may have defined.

The field **Job Operation Context** can only have these 2 values: `update` or `creation`. This value was useful in cases where we now advise to use the **Create Job** task as a workflow step. Learn more in the [Advanced](#) tab of that task.

Routing Based on Number of Files

Choose this to route collections of input files based on the number of files in that collection.

Some examples:

- When you get not 1 but 2 files, you want to route to the recto/verso (front/back) workflow.
- You created various **Tabular Step & Repeat** tickets, one for each amount of **Grids** that they offer. You use the Router to route to the Step & Repeat ticket with the same number of grids as the amount of input files.

Routing Based on Milestone

Choose this to route based on the fact if a certain Milestone is **Set** or **Not Set**.

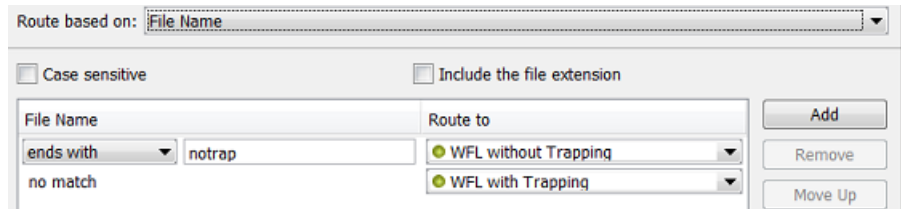
For example: you have a workflow where someone from the administration department uses the Pilot to select a Job and set the Milestone 'OK from Admin department to Print'. The Router checks if this OK was already given before continuing the workflow to the press.

The screenshot shows the Router configuration interface. At the top, 'Route based on:' is set to 'Milestone'. Below it, 'Milestone:' is 'OK from Admin Department to Print'. The main configuration area has two columns: 'Milestone' and 'Route to'. The first row shows 'is' with 'Set' leading to 'Print workflow'. The second row shows 'is' with 'Not Set' leading to 'Wait for 1 hour and check again'. A third row shows 'no match' leading to 'Error'. On the right, there are buttons for 'Add', 'Remove', 'Move Up', and 'Move Down'.

Routing Based on File Name

Choose this to route files based on their name. Choose to **Include the file extension** or not.

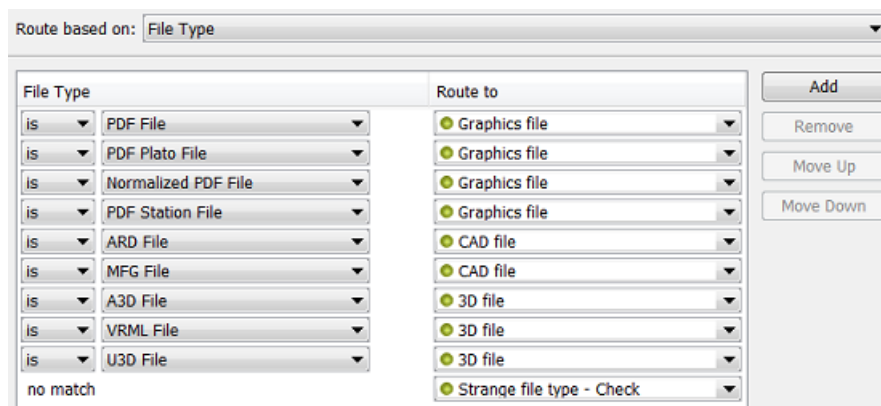
For example: You agreed with the person sending you design files that no trapping will be done if he ends the file name with 'notrap'. You created an Access Point that launches a workflow that starts with this Router:



Routing Based on File Type

Choose this to route files based on their type.

For example: this Router categorizes many file types into specific routes:



Routing Based on File Size

Choose this to route files based on their size.

Select **Consider the combined size of all files in the collection** if you want to match the size of the whole file collection and not of its individual files.

See an example in the above introduction.

Routing Based on Number of Pages

Choose this to route files based on their number of pages.

Select **Consider the combined page count of all files in the collection** if you want to match the page count of the whole file collection and not of its individual files.

For example: if the file has 2 pages, then you know it is a recto/verso (front/back) file.

Routing Based on Page Orientation

Choose this to route files based on their orientation (**Portrait**, **Landscape** or **Square**).

Some examples:

- your RIP does not have an auto-rotate function. Your workflow will then need to make sure that you send the data in the wanted orientation.
- you want to know if a file fits on an output sheet. Before you compare the sizes, you will want to know which side is the longest. If your output sheet is 100 horizontal by 60 vertical, you best only compare sizes when your input file is also in landscape mode.

Routing Based on Page Size

Choose this to route files based on their page size.

In **Pagebox to use**, you can choose either **Trimbox** or **Mediabox**. For images, both settings Trimbox and Mediabox will have the same result. The image size will be used for the comparison.

This is typically used to route files to the output device with the right size. For example to send small files to the small proofer, and bigger ones to the big proofer.

An example: When you want to route a file to a right printer:

- When a file fits on A4 paper=`"is less than or equal to 210x297 mm"`
- When a file fits on A3 paper=`"is less than or equal to 297x420 mm"`

Note that router entries have an order. In above example, a 100x100 mm file both fits A4 and A3 paper, but A4 is chosen because it comes first in the list of criteria.



Attention: Both page orientations of the input file are always checked! The page size that you here define is the size of a surface that the file can fit in or not, no matter how it is orientated.

`"Is less than or equal to"` means "Does the file fully fit in the specified surface (in portrait **or** landscape)?"

`"Is greater than"` means "Does the file fully cover the specified surface (in portrait **or** landscape)?"

For example: `'Page size'is equal to'210 x 297 mm'` will therefore match A4 files both in portrait and landscape mode.

Routing Based on Number of Separations

Choose this to route files based on their number of separations.

For example: when a flexo job has more than 6 inks, you use a different SmartMarks set. It then uses one with smaller rectangles for the color strips. You use a Router that checks the number of separations. If it detects more than 6 inks, the next step is the Step & Repeat ticket that uses the set of SmartMarks with the smaller rectangles.

Select **Consider the combined separation count of all files in the collection** if you want to match the number of separation of the whole file collection and not of its individual files.



Note: This counts the number of **unique** separations. For example, if you have a CMYK+Red and a CMYK+Green file, the collection's combined separation count will be 6.

Routing Based on Separation Name

Choose this to route files based on the name of their separation(s).

You can check if **any** (one or more), **all** or **none** of the separation names match or contain parts of a specified string.

For example:

- when 'any' separation names 'contain' the text string 'pantone', then route to the part of the workflow where you process the files that are not CMYK-only.
- when 'all' separation names 'does not contain' the text string 'Die', then route to the **Checkpoint** task that will add an Action to the To Do List. The Action will ask the user to add that separation.

Routing Based on Layer Name

Choose this to route files based on the name of their layer(s).

For example: you sometimes receive PDFs with several layers of black text in different languages. The Router checks if the file has any layers with these names: FR, SP, EN, DE, IT, or PT. If so, it routes them to the workflow where you first use **Version PDF** to create PDFs with only 1 language in black (k): name_cmy_FR_k.pdf, name_cmy_SP_k.pdf, etc.

You can check if **any**, **all** or **none** of the layer names match. When using **any**, only one layer name needs to match for the file to be considered matching.

Routing Based on SmartName Value

Choose this to route files based on a SmartName value.

For example: the Router checks the workflow parameters 'Print Process' or 'Preflight Yes or No'. The values of these parameters decide what route the file needs to follow.

Find a good example in [Launching a workflow with parameters from an XML](#). Step 5 of that example illustrates how the Routers use SmartNames.



Note:

Routing on SmartNames requires more resources from Automation Engine. If you have an alternative, we advise you to use another routing method instead.

If your SmartName only contains **Job** related parameters, and you have many input files, we advise to add a [Data Collector](#) in front of your **Router**. This will group the files into a collection and allow the Router to perform the SmartNames check only once.

Routing based on XPath

Use this when you have an XML file that contains a value that you want to route a workflow on.

When routing multiple files, the expression is calculated per input file.

To get help to specify the correct expression, click **Edit** to open the editor.

Learn about XPath expressions in [The XPath Builder](#).



Note: Using SmartNames is not supported with this type of routing.

Routing based on XMP XPath

Use this when you want to route a workflow on a value that is part of XMP metadata inside the (PDF) input file.

Learn about XMP XPath expressions in [XMP XPath Query SmartNames](#).

Learn about XPath expressions in [The XPath Builder](#).

To get help to specify the correct expression, click **Edit** to open the editor.



Note: Using SmartNames is not supported with this type of routing.

Routing based on JSONPath

Use this when you have a JSON file that contains a value that you want to route a workflow on.

This avoids having to create extra JSONPath SmartNames or to create many workflow parameters.

Learn about JSONPath expressions in [JSONPath Expressions](#) on page 78.

To get help to specify the correct expression, click **Edit** to open the editor.

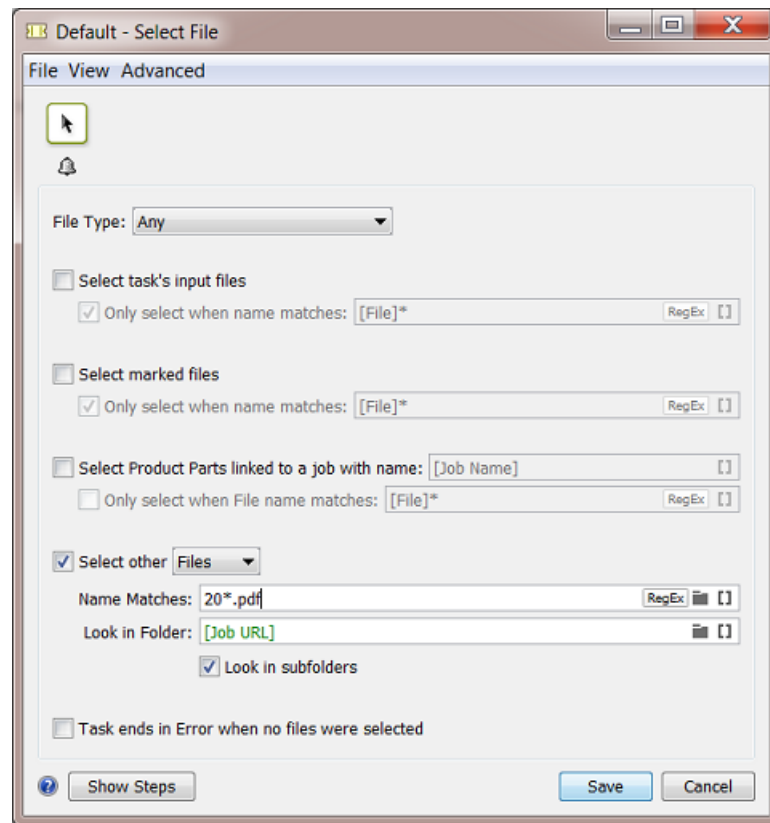
The [XPath Builder](#) then also functions as a JSONPath builder.



Note: Using SmartNames is not supported with this type of routing.

4.4. Select File

This task allows you to select one or a group of files on which you launch the next workflow step.



Note: In an input field, when the matching icon is present, you can use SmartNames or [Regular Expressions](#) or browse to define a folder. You can also use * as wild card character.

1. Define what file types you want to select in the **File Type** list. You can:
 - choose to select **Any** file type,
 - pick one of the default file types,
 - choose **Select File Types** and pick them in the **Select File Types** dialog.
2. Define which files you want to select:
 - **Select task's input files** will select all input files (of the selected **File Type** if you specified one).
Use the **Only select when name matches** option if you want to filter the input files to select based on their name.
 - **Select marked files** will select files you have marked earlier in the workflow (using the **Mark File** task).
Use the **Only select when name matches** option if you want to filter the marked files based on their name.
 - **Select Product Parts linked to a Job with name** will select Product Parts linked to a job of your choice. Specify the job name in this option.
Use the **Only select when File name matches** option if you want to filter the Product Parts based on the name of the file they contain.
See the chapter [Products](#) more information about Products and Product Parts.

- **Select other** will select other files or folders (that have not been processed by the workflow).
 1. Choose to select **Files** or **Folders**.
 2. Use the **Name Matches** option to select those files or folders based on their name. This is not case sensitive.
 3. Define where the files or folders to select are located in **Look in Folder**. Choose to **Look in subfolders** or not.
- 3. If you want, you can choose to make the **Task end in Error when no files were selected**.

4.5. Select Job

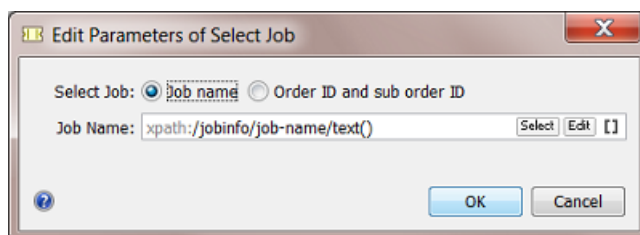
The **Select Job** workflow control serves to select a job in a workflow. It actually selects the Job Folder. The next task in the workflow will use that job folder as input.

This workflow control is typically followed by a [Remove Job](#) or [Archive Job](#) task.



Note: The job context and any job related SmartNames in the following workflow will still be those from the job that the workflow was started in originally.

You can choose to select a job using the **Job name** or by using the combination of the **Order ID** and **sub order ID**.



The **Job name**, **Order ID** and **sub order ID** can be specified by:

- Entering a value.
- Inserting SmartNames.
- Selecting a job using the **Select** button. The **Job name**, **Order ID** and **sub order ID** will then be filled in automatically.
- Entering an XPath expression. You can enter one manually or click **XPath** to use the **Xpath Builder** where you create an XPath expression that gets a value from an XML file. Learn more about creating Xpath expressions [here](#).
- When the input file is a JSON file, you can here also enter a **JSONPath** expression. Learn more in [JSONPath Expressions](#) on page 78 .



Attention: As mentioned in [The XPath Builder](#) , adding SmartNames or JSONPath expressions to your XPath expression is not supported when you are creating this kind of an 'inline' XPath expression.



Attention: The **Select Job** workflow control only handles one input file at the time. When you want to select the jobs of multiple input files, you need to use a **Data Splitter** step in front of it.

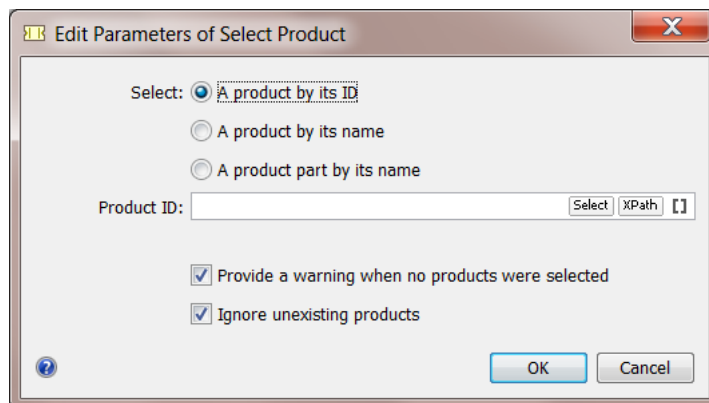
4.6. Select Product

Concept

The **Select Product** workflow control is typically used in a workflow where the next step is to change the status of the selected product (part) or to remove the product.

It can select one or multiple products or product parts. Its outputs are the URLs of all parts of the selected products.

You can select a product by its **ID** or **Name** or select a **product part by its name**.

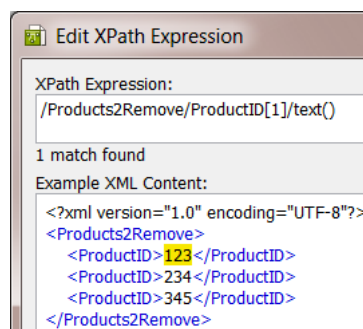


Ways to Identify the Product(s)

In the input field, you can type in values, **Select** a product from the list or use SmartNames.

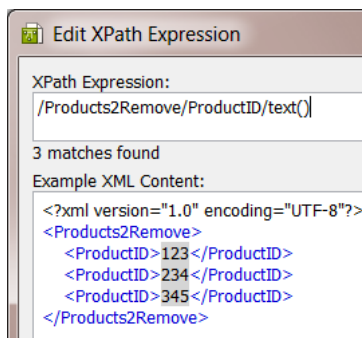
You can also add an **XPath** expression, in which case the input file of this workflow control has to be an XML file. The XML file can be one with or without XML namespaces. Click on **XPath** to open the XPath Builder and select the XML value matching the product (part).

This is an example where the expression results in one value, the first one of that XML element. Notice the [1] in the expression:



In this next example, the user manually removed the [1] in the expression and confirmed with **Enter** (or click **OK** and re-open the XPath Builder). You can see that the expression now selects **all** the values.

In such a case, where the XPath expression resolves to a list of values, each individual entry in the list is used to select that list of products.



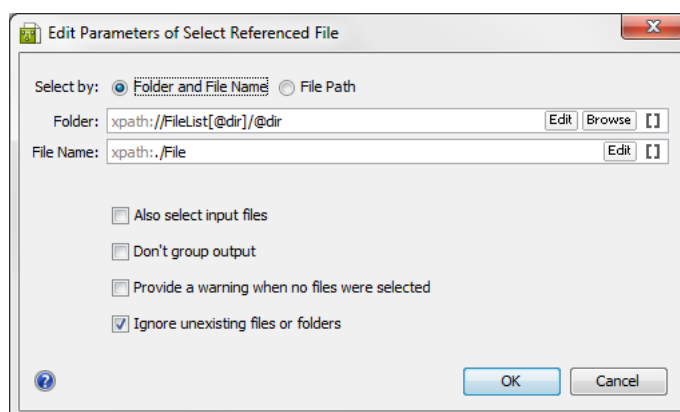
Learn more about the XPath Builder in [Creating XPath Expressions](#).

Options

- **Provide a warning when no products were selected.** When no products were selected and when this option is not checked, there will be no warning status and the **OK** output pin will show zero outputs.
- **Ignore unexisting products.** When this option is enabled, products that do not exist in the database will not be selected. When this option is disabled, selection of an unexisting product will lead to an error.

4.7. Select Referenced File

Use the workflow control **Select Referenced File** to easily select files that are referred to in an XML file.





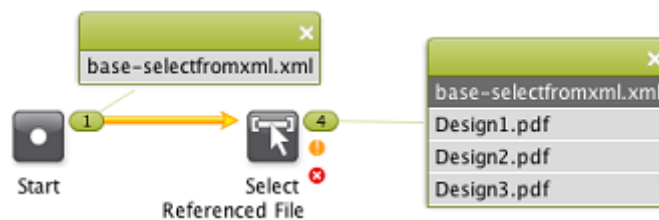
Note: The **XPath** button offers to use the **Xpath Builder**. Learn more about creating Xpath Expressions [here](#).

When the input file is a JSON file, you can here also enter a **JSONPath** expression. Learn more in [JSONPath Expressions](#) on page 78 .



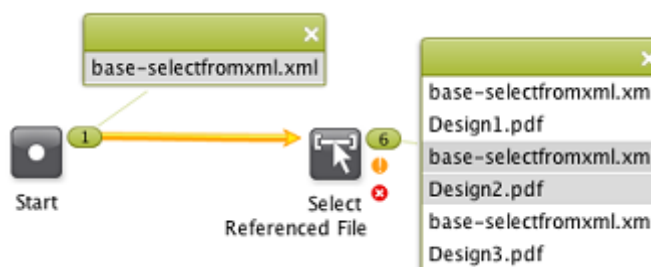
Attention: As mentioned in [The XPath Builder](#) , adding SmartNames or JSONPath expressions to your XPath expression is not supported when you are creating this kind of an 'inline' XPath expression.

- **Select by:** You can indicate (via SmartNames or XPath) which XML content to take to build up the corresponding file path(s) and name(s). You can choose between:
 - **Folder and File Name:** Choose this option if the source XML contains entries that define the file path and file name separately. If this option is chosen, specify both the **Folder** and the **File Name**.
 - **File Path:** Use this when each file in the source XML has a fully specified absolute file path.
- **Also select input files:** Select this option to also add the step's input file(s) to the group of output files.



- **Don't group output:** When this option is selected, the selected files will be offered separately to the next step in the workflow instead of as a group.

When **Don't group output** is not selected (so output is grouped) and **Also Select Input Files** is selected, the task will generate a group for each output:



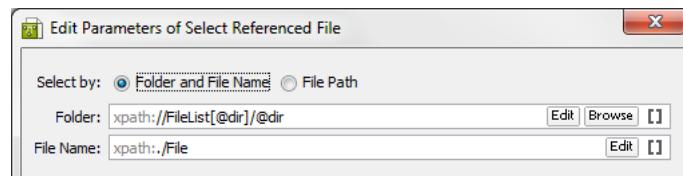
- **Provide a warning when no files were selected:** Select this option to get a warning when no files are selected.
- **Ignore unexisting file or folder:** Select this option to prevent workflow errors when one or more file paths refer to non-existing files. When selected, the non-existing files will be ignored.

Example 1: Folder and File Name

Our example XML refers to one file in a first Job and then to 3 files in another Job:

```
<?xml version="1.0" encoding="UTF-8"?>
- <doc>
  - <FileList dir="file:///loppem/ExampleJobContainer/Orders/Order-2014-100024">
    <File>PDF/Design1.pdf</File>
  </FileList>
  - <FileList dir="file:///loppem/ExampleJobContainer/Orders/Order-2014-100025">
    <File>PDF/Design1.pdf</File>
    <File>PDF/Design2.pdf</File>
    <File>PDF/Design3.pdf</File>
  </FileList>
</doc>
```

We used the **Edit** button to get to these settings:



After the **Select Referenced File** step ran, you will notice these 4 files on the OK output pin of the workflow control (assuming the files exist).

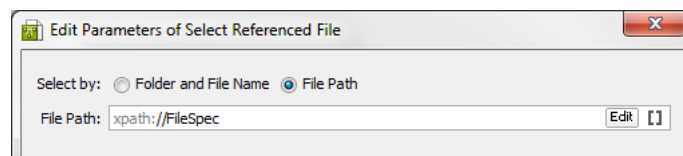
- file:///loppem/ExampleJobContainer/Orders/Order-2014-100024/PDF/Design1.pdf
- file:///loppem/ExampleJobContainer/Orders/Order-2014-100025/PDF/Design1.pdf
- file:///loppem/ExampleJobContainer/Orders/Order-2014-100025/PDF/Design2.pdf
- file:///loppem/ExampleJobContainer/Orders/Order-2014-100025/PDF/Design3.pdf

Example 2: Absolute File Path

This example XML contains several absolute file paths:

```
<?xml version="1.0" encoding="UTF-8"?>
...
<FileSpec>file:///AEServer01/Food/Orders/1234/PDF/design1.pdf</FileSpec>
<FileSpec>file:///AEServer01/Food/Orders/1234/PDF/design2.pdf</FileSpec>
<FileSpec>file:///AEServer01/Food/Orders/1234/PDF/design3.pdf</FileSpec>
...
```

We chose the option **File Path** and used the **Edit** button to get to this setting:



The output pin of the **Select Referenced File** step will show these 3 file paths:

- file:///AEServer01/Food/Orders/1234/PDF/design1.pdf

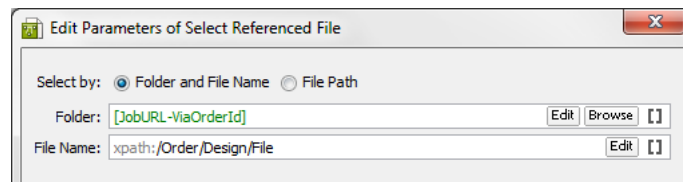
- file://AEServer01/Food/Jobs/1234/PDF/design2.pdf
- file://AEServer01/Food/Jobs/1234/PDF/design3.pdf

Example 3: Using SmartNames

This example XML has a different structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<Order id="2014-102457">
  <Customer id="JuiceCo"/>
  <Design>
    <File>PDF/design1.pdf</File>
    <File>PDF/design2.pdf</File>
  </Design>
</Order>
```

In this case, we find the Folder via a SmartName, based on the Order ID, and the File Name via an Xpath:



This is the resulting selection ; the part of the path including the Job number was found via the SmartName, followed by what the XML defined in <File> :

- file://AEServer01/Food/Jobs/102457/PDF/design1.pdf
- file://AEServer01/Food/Jobs/102457/PDF/design2.pdf

4.8. Set Priority

Use the **Set Priority** workflow control to assign a different priority to a part or a specific branch of a workflow.

You can set the priority to:

- **Workflow** (to use the general priority assigned to the workflow)



Note: By default, the general workflow priority is set to **Normal**. You can change it by going to **Advanced > Task Options...** in the workflow editor window.

- **Low**
- **Normal**
- **High**
- **Extra High**



Attention: This level of priority is different than the **'immediate'** one that a user can select in **Task Options** when he launches a task from the Pilot or browser client. Here, inside this step of a workflow, this maximum level of priority 'extra high' will not result in an immediate launch. Mind that this workflow control could be asked to set a priority for hundreds of **tokens** at once. To protect the stability and continuity of the workflow engine, you can here not request to have them all start immediately.



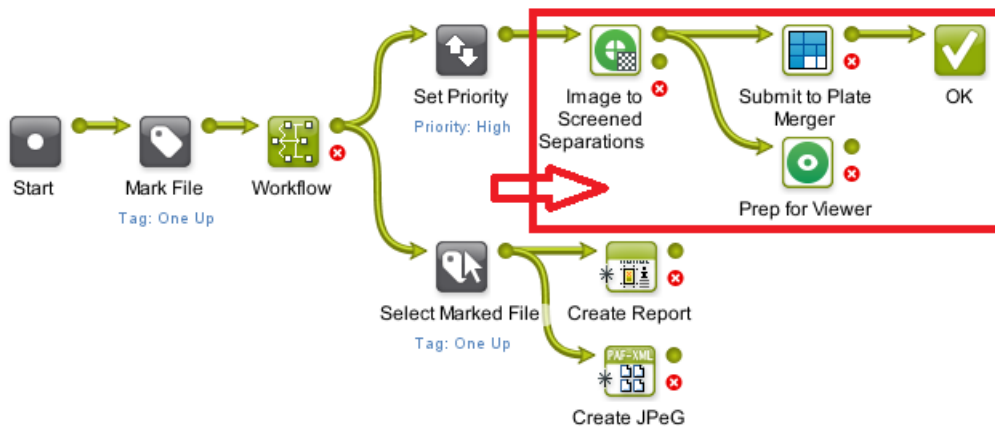
Set Priority

Priority: Low

The chosen priority is shown in blue right under the workflow control:

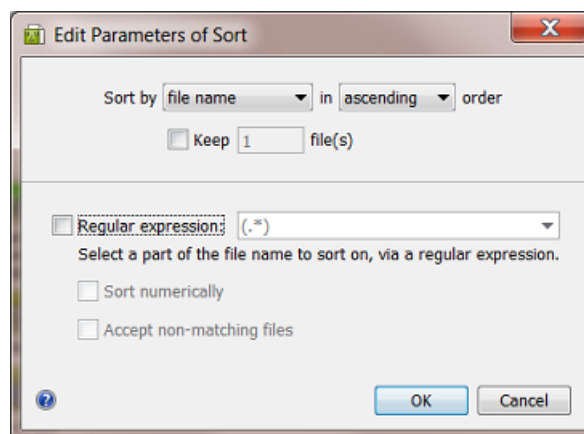
Example

In this example, the most urgent operation is RIP'ing the files and outputting them to the device. Creating a PDF report or a JPEG is less urgent. All the workflow steps following the "Set Priority" step will get this high priority, that whole workflow branch:



4.9. Sort

Use the **Sort** workflow control to define the order of the files in a file set. You do this because the order is important in the next workflow step.



You can sort the files by:

- **file name.**
- **workflow inputs.** Files that entered the workflow first will be at the top of the list.

- **modification date.** This helps you to select the oldest or newest data.
- **SmartName.** The value of a SmartName is calculated per file and that value determines the order.

For every of these methods, you can sort in **ascending** or **descending** order.

Keep ... file(s): Use this if you only want to send a specific number of sorted files to the next workflow step.

Sort by file name

Use a [Regular Expression](#) to define the part of the file name that you want to sort on. The dropdown list shows two example regular expressions:

- `(.*)` matches the entire file name
- `.*?([0-9]+)` matches the number in the file name (for example to sort files named `cover1.pdf`, `page_2.pdf`, etc.).

Learn more about regular expressions in [Using Regular Expressions](#).

- **Sort numerically**

Select this option if you are using a regular expression to extract a number from file names. The results will then be sorted numerically and not as text. For example, a value of 10 will be sorted after 9 when this option is selected. When you don't sort numerically, the values are considered as text: then 10 will come before 9 (as text, the first character 1 comes before 9).

- **Accept non-matching files**

Select this option if you also want to sort files that don't match the criteria. For example your regular expression is based on numbers but the name of one file name doesn't contain numbers.

Any non-matching file(s) will be put at the top of the sorted list.

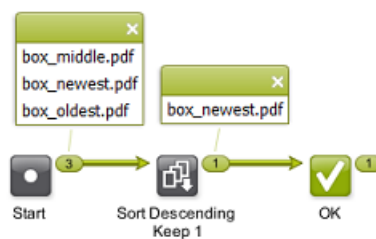
If you don't use this option and a file does not match, all your files will go to the error output pin.

Sort by workflow inputs

Files that entered the workflow first will be at the top of the list. You can sort in **ascending** or **descending** order and optionally define how many files you want to keep.

Sort by modification date

This method can be used to select the oldest or most recent data. See this example where we used the settings **descending** and **Keep 1 file** to sort and end up with only the most recent file:



Sort by SmartName

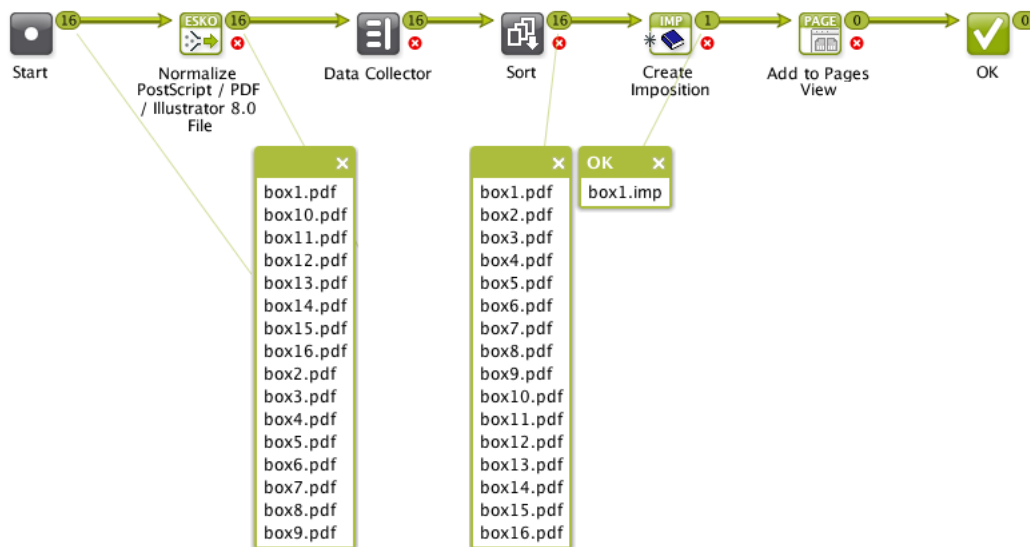
Select the SmartName you want to sort on. This list only offers custom SmartNames.

Numerical versus text based sorting works as described above for **Sort by file name**. The same is true for the handling of non matching files.

Example: Sorting by Page Number

In this workflow, the **Data Collector** collects the pages of a page list after Normalizing, and sends them to the **Sort** workflow control. Learn more about **Data Collector** [here](#).

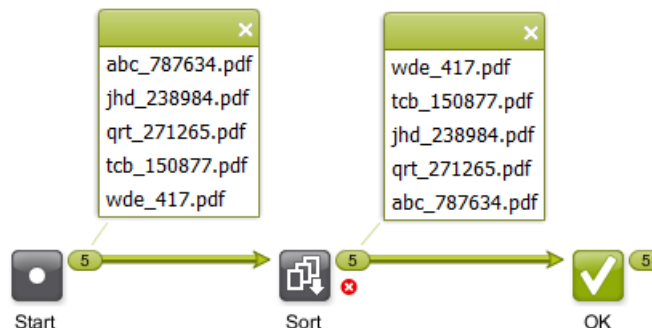
The **Sort** workflow control then sorts them in the right order, before sending them to the **Create Imposition** task. The files are sorted on **file name**, in **ascending order**, using a **Regular expression** to extract the number in the file name (and with the **Sort numerically** option):



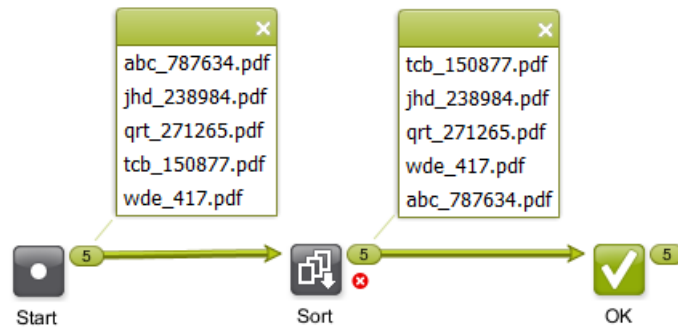
Example: Sorting by SmartName, Numerically and As Text

In this example, you want to sort your files by the number that they contain. The number is always mentioned after an underscore. You created a SmartName that extracts, from a file name, everything after the underscore.

This is the result when you **Sort numerically**:



This is the result when you don't **Sort numerically** ; the part of the name after the underscore is now considered as text:



4.10. Mark File and Select Marked File

These 2 workflow controls are used as a pair. First you use **Mark File** to tag files, then you use **Select Marked File** to select those files later in your workflow.

You can have several pairs of **Mark File** and **Select Marked File** in your workflow, as long as each pair has its own tag.

Mark File Settings

In **Mark all incoming items with tag**, enter the tag to set on all files passing through this workflow control.

Select Marked File Settings

In **Select all items with tag**, use the drop-down list to select a tag that you entered earlier in the corresponding **Mark File** workflow control.

You can filter the selection of tagged files. Select **Only select when name matches** and enter what name the file should match. You can use SmartName(s), *Regular Expressions* or an asterisk * as a wild card.

If you are working with subworkflows, you can choose to **Also search workflows containing this workflow** (to search 'up'). Learn more about subworkflows [here](#).



Note: When no files are found, an empty token will be put on the 'warning' output pin. This allows to continue a next step originating from that warning status.



Learn more about tokens in [Understanding Tokens](#).

Example

A typical example is launching tasks on the one-up PDF after your workflow was already working on the Step & Repeat version. In this case, you can tag the one-up file with **Mark File**, later select it with **Select Marked File** and then for example archive it:



Note: You need to use the exact same tag in the **Mark File** and the **Select Marked File** step. It is case-sensitive, so we advise to always use the drop-down list.

4.11. Data Collector

Use the **Data Collector** workflow control

- to collect incoming files from different routes in the workflow and then offer them to the next step as a group
- to wait for files till all expected ones have arrived, and then send them to the next step (without grouping them).

For a Data Collector to know when it has collected all necessary data can be an intensive process when workflows are rather complex, not just in amount of workflow steps and routes, but especially when many files and workflow parameters are involved.



Note: This workflow control is actually a 'Token Collector'. It collects tokens in a workflow. We strongly advise to first learn about tokens and file grouping in [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

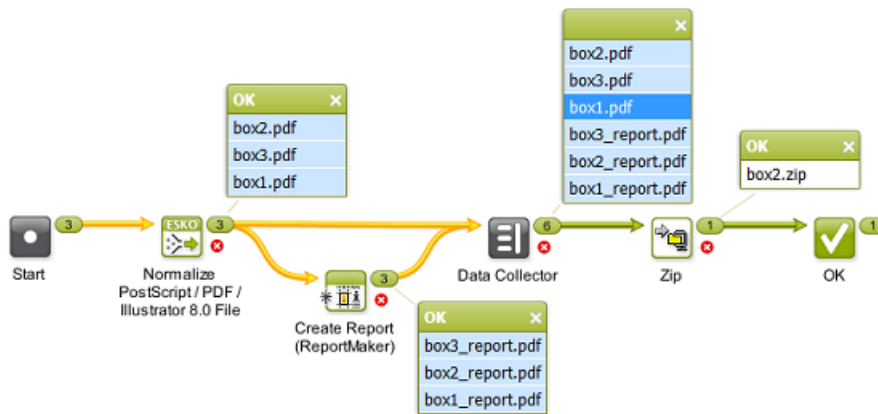


Warning: Do not make a workflow where you connect the output pin of a Data Collector with the input pin of another Data Collector. The behavior of such a loop is unspecified and so not supported.

Example

In this example, the Data Collector waits for all of the files to arrive (one-ups and their reports), then groups them and sends them as one token to the **Zip** task.

The result is one ZIP file containing all the one-ups and also their reports.



Settings

- **Error Policy**

Define what needs to happen when errors appeared earlier in the workflow, before this Data Collector step:

- **Fail when errors occur.** When a token arrives that came from a step with an error state, it will end up on the Error output pin. When the token did not come from an error state in this workflow, it will end up on the OK output pin.
- **Ignore errors (and continue):** All output token(s) always end up on the output pin OK.

- **Grouping**

- **Group all collected data:**

- When **selected**, all incoming data are collected into one group. The output pin will show that group as one token.

In the above example, see how, in the output widget of the Data Collector, a mouse-click on `box1.pdf` also highlights all the other members of the group in light-blue (and as usual also highlights the related files in previous workflow steps).

- **Deselect** this option when you want to use a Data Collector as a wait point in your workflow. It then serves to hold the workflow until all previous steps are finished. Once all expected tokens have arrived, the Data Collector allows the workflow to continue, using the same input items (tokens) as they entered the Data Collector (whether they were separate items or already grouped).

In this case, each incoming token will also appear as a separate token on the output pin.

- **Merge workflow parameters from all collected data:**

This option is disabled when you did not select **Group all collected data**. Any incoming *workflow parameters* then remain attached to their specific file set.

- Select this option to also merge workflow parameters from the incoming tokens (default setting). The 1 output token will hold 1 set of values for its defined workflow parameters.



Warning: Merging workflow parameters can create unwanted effects in your workflow!

Some steps in a workflow can set or change the value of workflow parameters. The token then continues with that new value.

It is not only the dedicated task [Modify Workflow Parameters Values](#) that can set these values, also the tasks [Checkpoint - To Do](#) and [Integrate with WebCenter](#) can do that.

When that token then arrives at the Data Collector, the values that its workflow parameters then have will be those in its output token(s).

However, when these tokens arrive from different routes, how should the workflow engine decide of which token it should choose that parameter and its value?

See the below page to learn more on this complex matter. It uses [some examples](#) to illustrate why the result of this 'merge' option is unpredictable.

- Deselect this option to keep any workflow parameter values only valid for their specific token.

See the below page to see [some examples](#).



Important: Having this option deselected can have a positive effect on the performance of the workflow. So if your workflow ticket does not require this merging of workflow parameters, we advise to deselect this option.



Note: When it not advised to use it, why is it the default setting in this task ticket? Because, before this option existed, this was the built-in behavior in the Data Collector. From v20 on, it is an option in this task's ticket and we advise to not use it.

4.11.1. Examples of the Challenges in Merging Workflow Parameters

The Problem with Merging Workflow Parameters

The Data Collector is typically used to collect data from several routes. And back-tracking information in very complex workflow is a very intensive action for a workflow. Of course it's a lot easier when a workflow is only 1 linear route: a same file can only be collected once and its workflow parameter's last set value is the only one arriving in the Data Collector.

In the Data Collector, the option to **Merge workflow parameters from all collected data** raises these 2 challenges:

- **Unpredictable result:**

Except in very simple workflows, it is not clear which parameter values will be part of the grouped token that the Data Collector outputs.

When different workflow routes end up in the Data Collector, and when, for a same workflow parameter, each route sets a different value, which value should then be taken? And also, does it still matter at that point in the workflow? Maybe the values were already 'used' by earlier workflow steps...

The workflow engine tends to keep the first value that arrived at the Data Collector but the problem is that timing *within* a workflow can not be controlled. You can not predict the moment of execution

of specific steps within a workflow. The time that a workflow step starts is always influenced by the load of the workflow engine at that specific time, by available memory of the server computer, etc. When asked to find a value or choose between values for a same parameter, the workflow engine tends to pick the first value arriving, but because we can not control the time they arrive, we can not be 100% sure which one that will be.

Conclusion: When, in different routes, different values were set for a same parameter, the resulting parameter value is **unpredictable**.

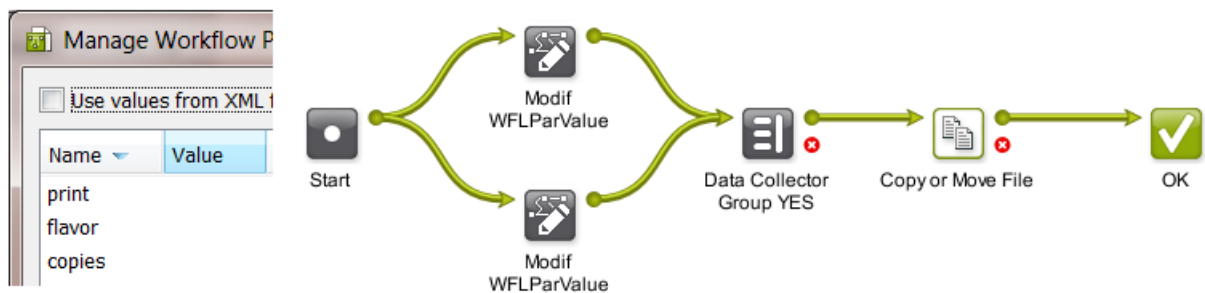
- **Load on the workflow engine:**

The back-tracking that the Data Collector needs to do to find all relevant data, can, in specific and complex workflows, lead to an extreme **load on the workflow engine**.

A Data Collector collects, i.e. it searches back in the workflow routes for the tokens it should collect. But it also collects the tokens' parameters and the value of those parameters that it should hand over to the next step in the workflow, as part of its grouped output token. When you have a large workflow, with several routes, where parameters were modified, where you may have added more than one data collector, this becomes a massive investigation to be done back in the workflow.

See some examples and resulting advice below. The examples are minimal, i.e. they only show what is relevant for this explanation.

Setup of our Example Workflow

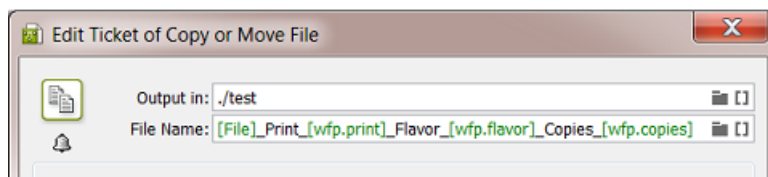


1. Our example workflow has 3 workflow parameters: `print`, `flavor` and `copies`. In our case, their initial value is empty.
2. Each input file is sent to 2 routes. In each route, the value of one or more workflow parameters is modified.



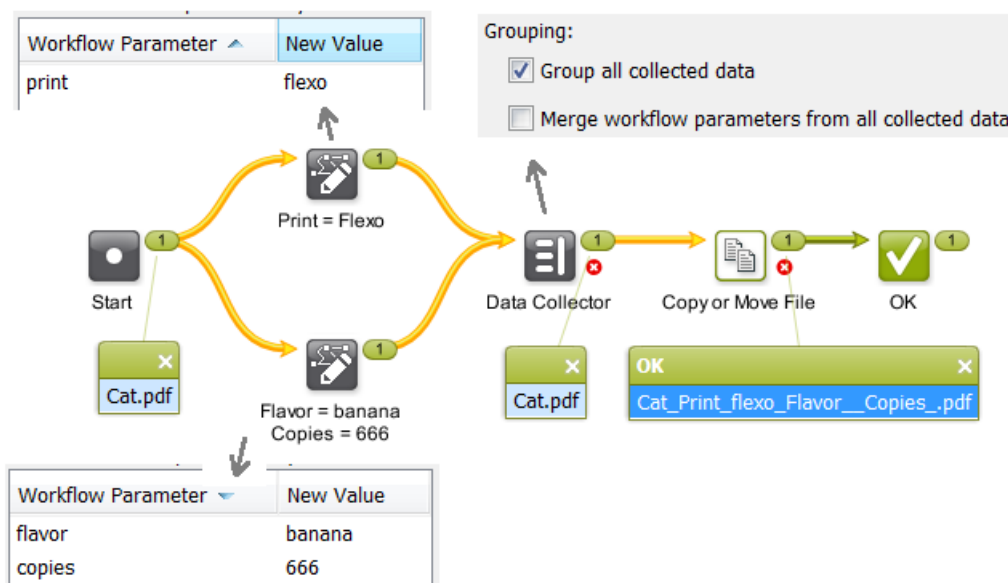
Note: We here use the step [Modify workflow parameter values](#), but, as mentioned [above](#), there are more task types that can set values of workflow parameters.

3. The Data Collector groups the files coming out of these routes. This is where the below examples show the different cases: What happened to the workflow parameters in these routes? Should the Data Collector look them all up and merge them for the output token? Are they really all still relevant to the next steps in the workflow? And what if a token received different values for a same parameter?
4. We use the Copy or Move File task as a little trick to quickly see the resulting parameter values. We have it create a file which name contains the value of the workflow parameters.



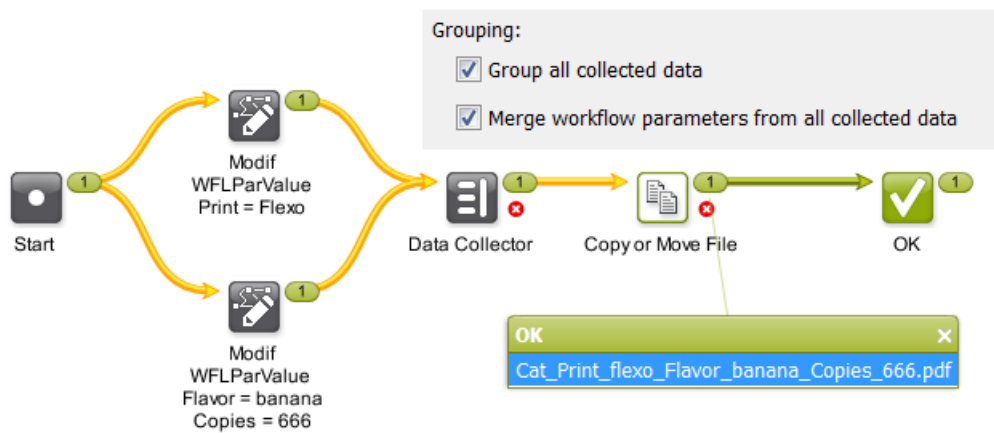
Restriction: Unfortunately, in the workflow editor, you can not use the tool **Workflow Parameter Values** to help analyse what is happening in below examples. This tool uses another algorithm that does not show the result of this option to merge parameters.

Example 1a - Non-overlapping Parameters - Merge NO



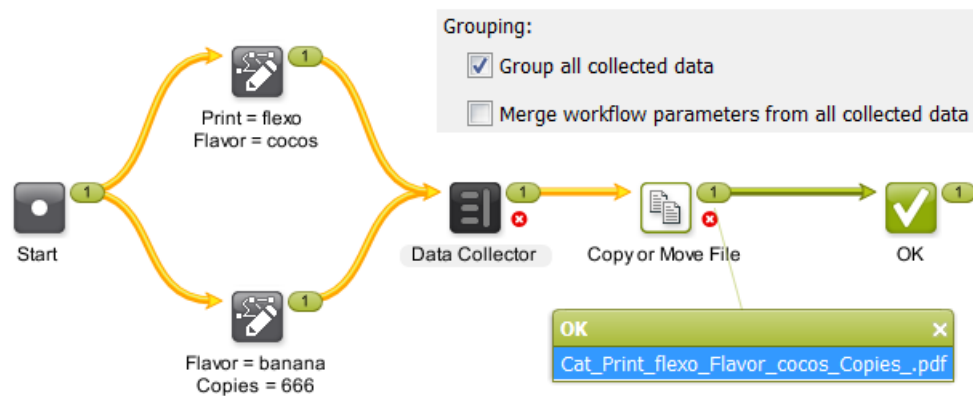
- For sake of simplicity, the workflow was started with 1 input file. The result would be the same with multiple input files (and then at least the Data Collector would indeed 'group' at least these 2).
- After splitting into routes, the top workflow control sets the value for the workflow parameter `print` to `flexo`. The bottom one sets a value for the 2 other parameters, `flavor` and `copies`. The parameters do not overlap once they arrive at the Data Collector.
- The Data Collector groups and is not asked to merge the incoming workflow parameters, which in this case would be easy as they do not overlap. So which ones will he keep then, in the group output token, those that came from the top route or those from the bottom route?
- The name of the output file shows us that the workflow did 'receive' the value `flexo` for the parameter `print` from the top route, but that it's not aware of the changes done to other 2 workflow parameters in the bottom route. It didn't even check them after it already received a value from, in this case, the top route.
- The result actually could have been be the reverse as well: that the parameters from the bottom route were used, and not those from the top one.
- **Conclusion:** A simple re-run of this workflow could have a different result. Any other activity in the workflow engine could slow down any workflow step or route and so change the sequence in which they arrive at the Data Collector. The result is unpredictable.

Example 1b - Non-overlapping Parameters - Merge YES



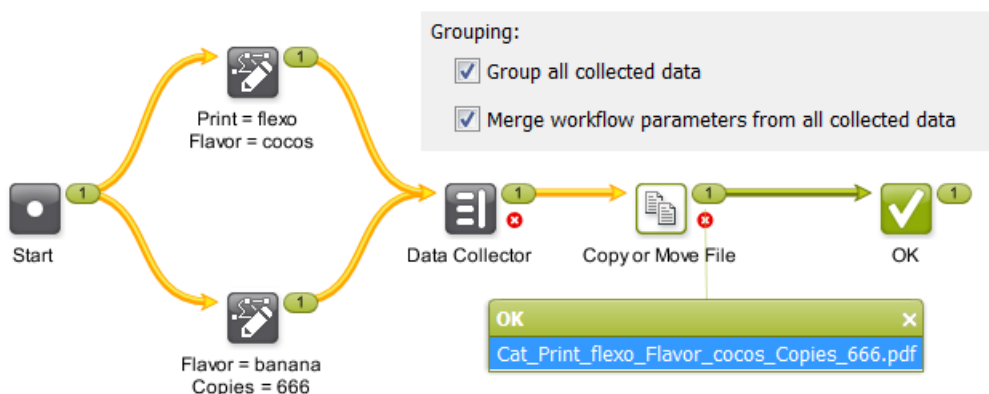
- Same workflow, same input file(s). The only difference is that we ask the Data Collector to also merge the workflow parameters.
- At the end of the workflow, the name of the output file now shows that all 3 workflow parameters were successfully merged into the output token.
- **Conclusion:** This may look good but we were lucky: It is not always realistic to keep the parameters unique over their route, and, most important, this workflow was very simple. A real workflow would have become noticeably slower.

Example 2a - Overlapping Parameter Values - Merge NO



- In this case, both routes set a value for the parameter `flavor`.
- The Data Collector is not asked to merge parameters. It can take either one of the values. It picks one route to check back on and as soon as it finds a value, that one will be taken.
- This time, we see that the value `cocos` was chosen to be forwarded with the output token.
- Relaunching this workflow in realistic circumstances can as well result in the value `banana`.
- **Conclusion:** It can not be predicted which value will be taken.

Example 2b - Overlapping Parameter Values - Merge YES



- Also in this case, both routes set a value for the parameter `flavor`.
- Now, the Data Collector is asked to merge parameters. It therefore will check back in the workflow and assemble all the values for all workflow parameters.
- Both routes were investigated and all parameters from both routes were merged.
- In the result, notice how now the parameter `copies` got a value as well (from the bottom route). And that the values for the other 2 parameters came from the top route.
- Relaunching this workflow on a busy server can have a different result.
- **Conclusion:** A request to merge means that the Data Collector needs to do a lot more investigation but still it can not be predicted which value will be taken.

Resulting Advice

The process of merging workflow parameters can be problematic to the workflow engine and its result can not even be predicted.

Therefore,

- we advise to build your workflows in a way that you do NOT need to ask to merge workflow parameters. As mentioned, this also benefits the processing of the workflow engine in general.
- we advise that, if in some simple workflow you do need this option, that you thoroughly test it. Be aware that the results are unpredictable and that it increases the processing load of the workflow engine.

4.12. Data Splitter

Use the **Data Splitter** if you want the next step in the workflow to process each file individually.



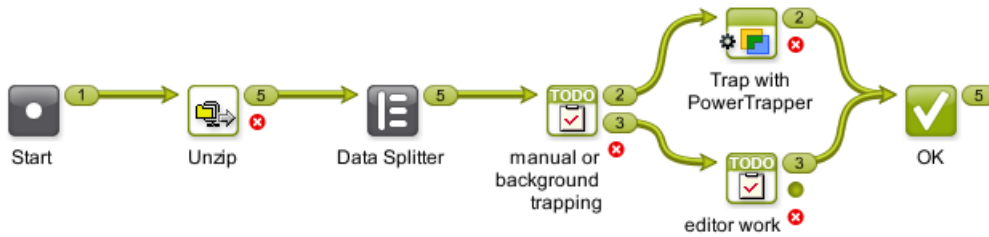
Attention: Only use this task when necessary ; when the default behavior of the workflow is not what you want. This kind of splitting can create a very large amount of extra 'tokens' and that significantly increases the complexity and load of the workflow engine.



Important: We advise to first learn about how a workflow can process files in groups in the page [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

Example

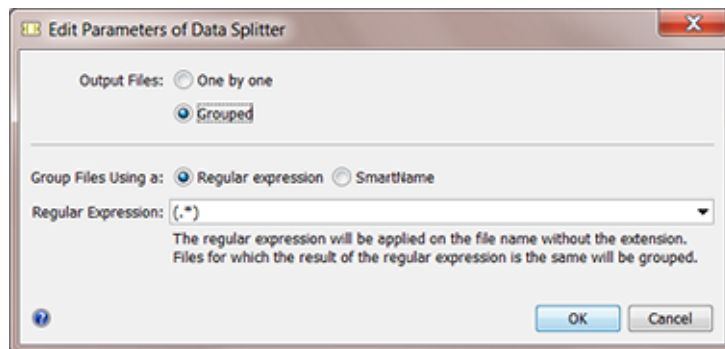
In this example, the Data Splitter (with the default setting **One by one**) sends the unzipped files to the **Checkpoint** task one by one. This way, 5 **To-Do Actions** are created. The user needs to decide for each file if it needs manual or background trapping:



Without the Data Splitter in this workflow, there would only be one **To-Do Action**. The Action item would provide a link to the 5 files, but they would need to share the same decision.

Split and Create a (new) Group

You can also use the Data Splitter to split the collected files and **group** them according to a criterion you define.



You can group the output by using a [Regular Expression](#) or a **SmartName**.

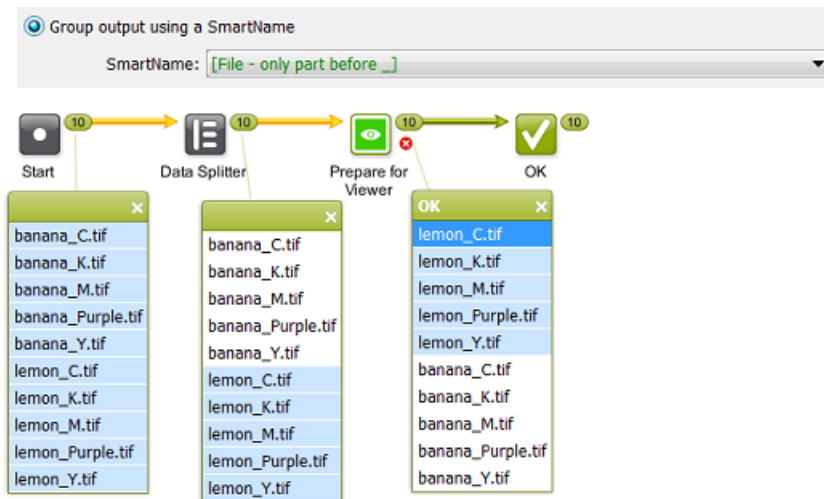
The dropdown list by default offers 2 regular expressions:

- (.*) matches the entire file name without the extension
- .*_pag([0-9]+).*? matches the number directly after _pag in the file name (use this for example for files named page_1.pdf, pagina_2.pdf, etc.).

Only the file name will be matched, without the extension. Learn more about regular expressions in [Using Regular Expressions](#).

Example: Grouping the RIP output to Prepare for Viewing

The Automation Engine **Viewer** enables viewing of RIP'ed data when they were **prepared for viewing** together, as a group. When you have a folder containing RIP output of several jobs, it is important that you send the right groups of separations to the **Prepare for Viewing** task. You can use the Data Splitter and group the separations by using a SmartName:



Note: You can use the same principle in a workflow where you use the **Create Wrapper file** task.

4.13. Fork and Join



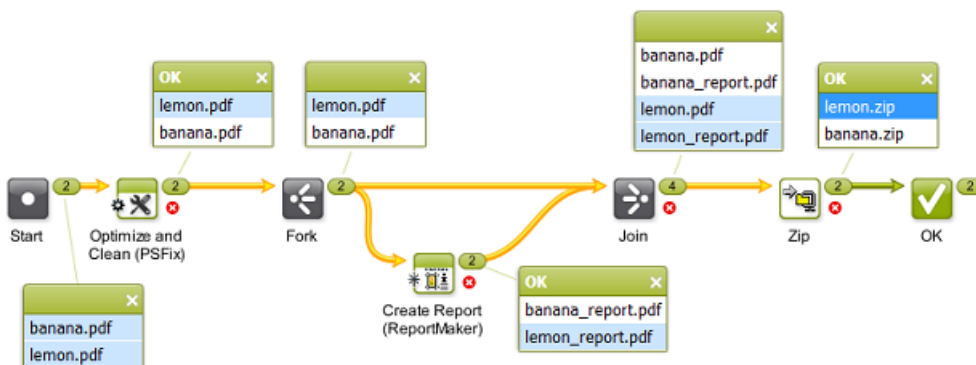
Note: We advise to first learn about file grouping in [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

The **Fork** workflow control automatically tags every input file with a same token as all the files that the workflow generates from those initial files in the next steps, this until you **Join** them again. This creates a **Group Tag**: These related files all get a same **token**.

The **Join** workflow control then groups all the files with the same tag.

Example of Group Tagging

Our example workflow creates a ZIP containing the one-up PDF and its report PDF. This is easy to set up when the workflow starts with only one file. This is different when your workflow starts with multiple files:



In above example,

- we have 2 input files that will travel through this workflow together. This example starts with an 'Optimize and Clean' step, just as an example of a task that outputs more than one file.
- the **Fork** step then starts tagging each input file and each file created from that file with a same token. In this example this is the input file and its report file that is created in the next step.
- the **Join** step creates 2 groups based on those tokens:
 - one group is the `lemon.pdf` and the `lemon_report.pdf`
 - another group is the `banana.pdf` and the `banana_report.pdf`
- these groups then become the inputs for the **Zip** task. You now have a Zip file per group: 2 Zip files.

Without the **Fork and Join** steps, you get 4 ZIP files: one for every file. If you use a **Data Collector** to group all of them, you get 1 ZIP of all 4 files.

With these **Fork and Join** steps, you can get the ZIP files of the 2 'related / grouped' files.



Note: This **Fork and Join** combination so enables special cases that you can not solve by using the more general [Data Collector](#) or [Data Splitter](#).

5. Using Public Parameters in Workflows



Note: Public Parameters are not only available for workflows tickets but also for single task tickets. This is documented in [Using Public Parameters in Automation Engine](#). We here only add the specifics about using public parameters in workflow tickets.

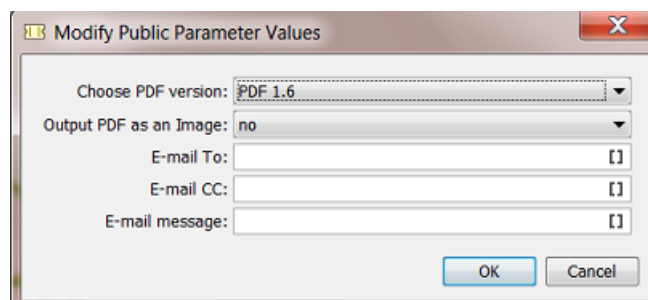
5.1. Concept

Most Task Tickets offer many settings ('parameters'). And Workflows can be quite complex combinations of many such Tickets. That is why, when an administrator creates a workflow, it can be useful to not show this whole complexity to every day users of those workflows.

This can be done by making only some parameters public. Thus, the daily user can only see and modify a limited set of parameters that he understands and is allowed to change.

For example:

A user wants to launch a workflow and needs to check or change only a few settings. Instead of having him open the workflow and open several steps to change only 5 parameters, you offer him this simple dialog, where he is asked to define the 5 parameters that you made public to him:



When the user presses **OK**, the workflow starts and uses the chosen settings.

Available from many Client Applications

This functionality is available for users launching workflows from these applications that have a connection with the Automation Engine server:

- Automation Engine clients Pilot, Shuttle and from the browser client workspaces
- Esko editors ArtPro+ and DeskPack (Adobe Illustrator)
- Esko legacy editors ArtPro, PackEdge, Plato, FastImpose.

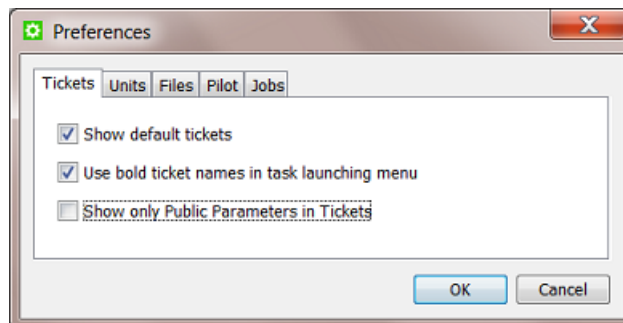


Important:

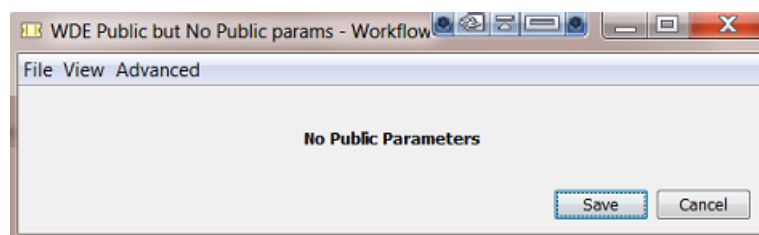
- Client applications only show **Public** tickets and workflows, no matter what user is logged in. In other words, they only show the same list of tickets that a user sees who does *not* have the User Access Right "**Tickets: Show all Tickets and their Parameters (Public and other)**".
- If those public tickets and workflows do not have any public parameters, the user will not be able to open them to change any parameters. They will be launched right after he selected the (workflow) ticket.

Preference to have the Pilot show Only Public Parameters

In the Pilot, in **Edit > Preferences** the option **Show only Public Parameters in Tickets** can help you prevent users from opening workflows in the workflow editor and so having access to all settings.



- When this option is not checked, users will be able to open workflow tickets and see all steps and settings. If they can make changes and save the ticket depends on their **User Access Rights**.
- When this option is checked, a users that opens a workflow ticket will only be shown a dialog with the public parameters of that workflow. When that workflow has no public parameters, the user will see this dialog:



About Public Parameters in a Workflow Ticket that is Not Public

It is possible to set public parameters in a workflow ticket that is not saved as public. This can be useful when the non public workflow ticket is referenced inside a master workflow that is public. The public parameters of the referenced subworkflow will show up in the list of public parameters of the master workflow.

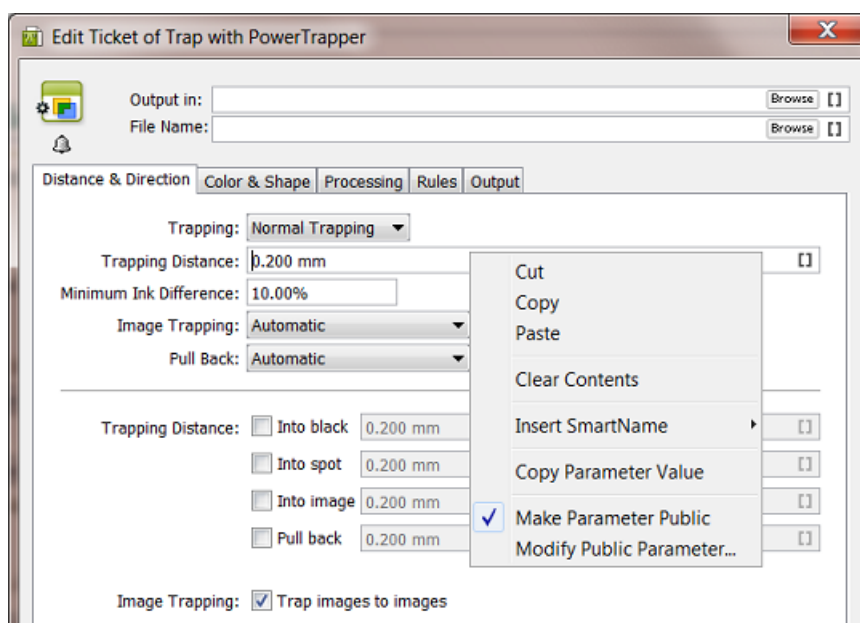
5.2. Creating Public Parameters in a Workflow

Creating public parameters in a workflow is very similar to creating them for a single task ticket, as explained in [Making a Parameter Public](#).

A summary:

1. In the workflow editor, select a step and open the ticket.
2. Right-click the parameter you want to make public and choose **Make Parameter Public**.

We here right-clicked in the field for **Trapping Distance**:



Note: Not all parameters of all tickets can be made public.

3. Right-click the same parameter again and choose **Modify Public Parameter....** Decide here if you want the parameter to be a free or predefined value.



Note: You can also skip this step and later manage this and all other public parameters in the [Manage Public Parameters](#) dialog. You can also use both dialogs.

4. Save your ticket to also save these changes to your public parameters.



Note: Public parameters belonging to steps that you later deleted from the workflow are not removed. You can remove them using the **Manage Public Parameters** dialog.

Next step is to manage the workflow parameters in a special dialog where you will define more precisely how they will present themselves to the user.

5.3. Conditional Availability of Public Parameters

Concept

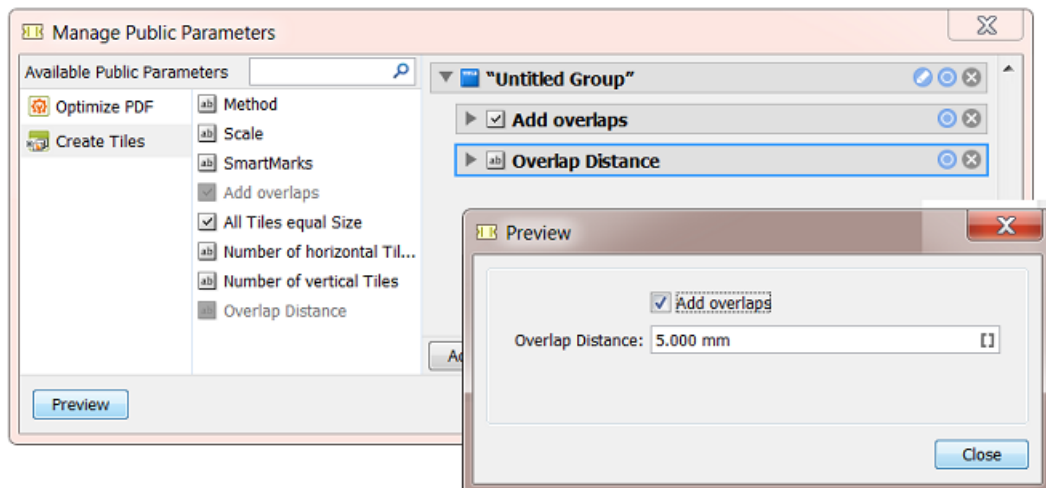
Some parameters depend on others. An extra tool offers to disable or even hide parameters depending on the state of others.

Below examples illustrate the functionality.


At the end of this page, we mention some limitations.

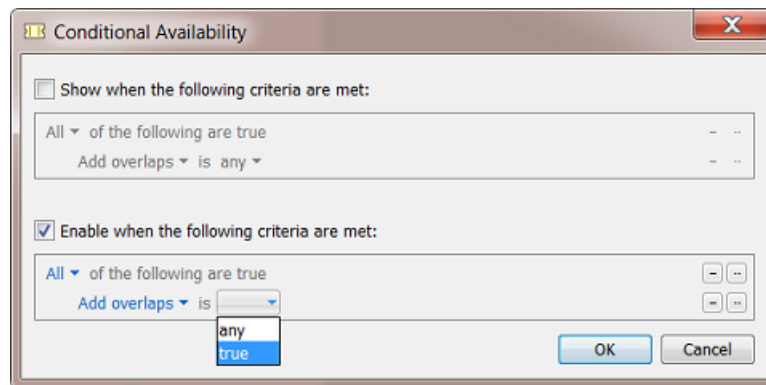
Example 1: Conditional Availability of one Parameter

This example shows the basic principle: When the parameter 'Add Overlaps' is not enabled, the parameter 'Overlap Distance' becomes irrelevant. However, the user still sees this option and can even fill it in.



We will use the blue circle button  to make this dialog more user friendly.

- Click the  of the parameter 'Overlap Distance'.
- The dialog **Conditional Availability** allows to **Show** / hide this parameter, or to just **Enable** / disable it.
- To only disable it, deselect the option **Show when the following criteria are met**, select the **Enable** option and set the criteria when this parameter should be enabled. In our example, this is when the user selected the parameter 'Add overlaps'.

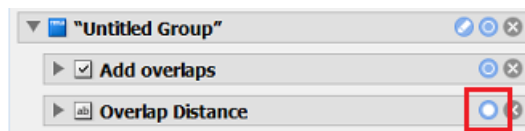


- In the list of criteria, the dialog automatically already lists the other parameters in that group of public parameters that can be used as a condition.

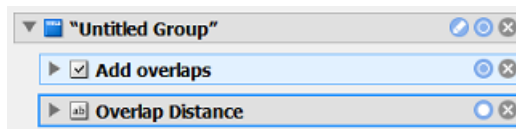


Note: Not all types of public parameters can be used to build a condition on. Learn more at the end of this page.

- Click **OK**. Notice how the blue circle button now changed to , indicating that a condition has been set for this parameter.

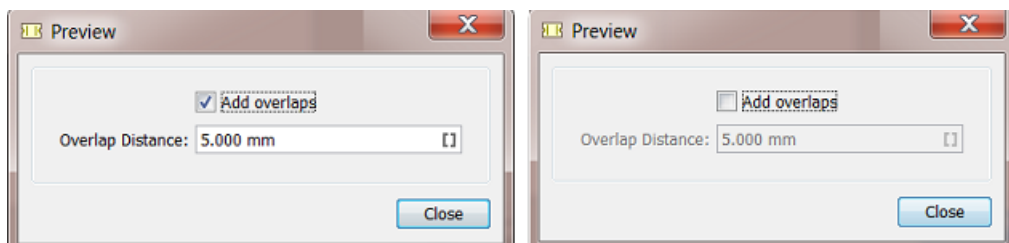


Also, when you select that parameter, the other parameter(s) that were used as criteria for the condition will be highlighted in light blue:

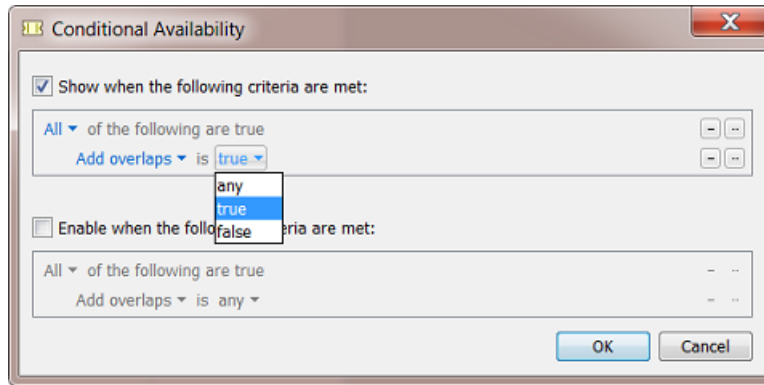


Note: When you remove a parameter that was used to set a condition, then those conditions will also be removed from those parameters: the blue circle button of the inflicted parameters will change back from to .

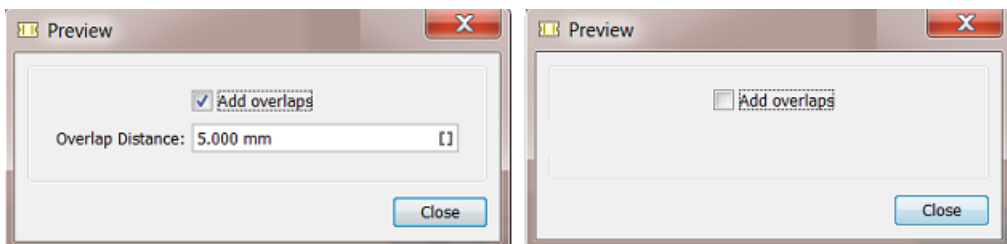
- Now click **Preview**. See how when no overlaps are asked for, the distance parameter is disabled (greyed out):



- If you prefer to not just disable an option that became redundant, but even hide it, then set one or more criteria for the **Show** option:

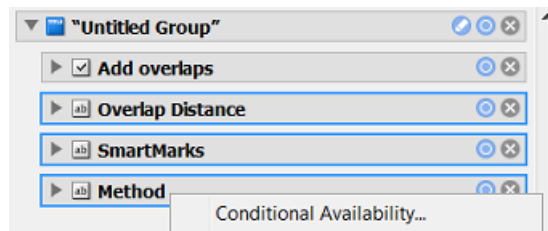


The preview will then show you this:

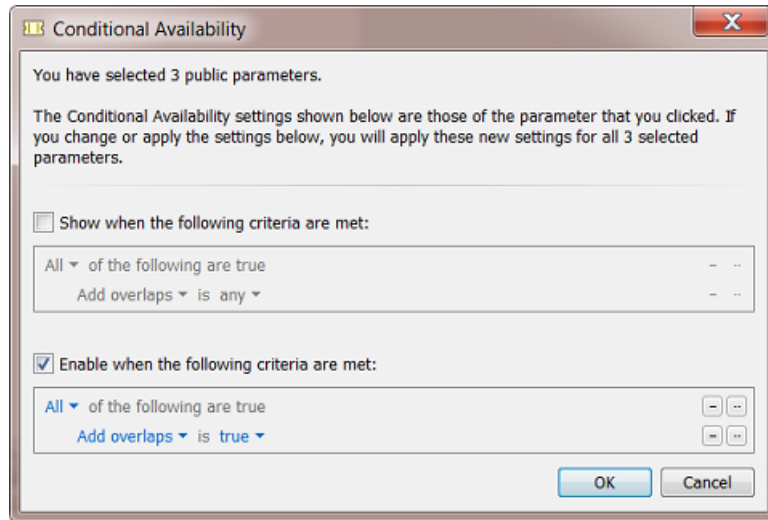


Example 2: Conditional Availability of Multiple Parameters

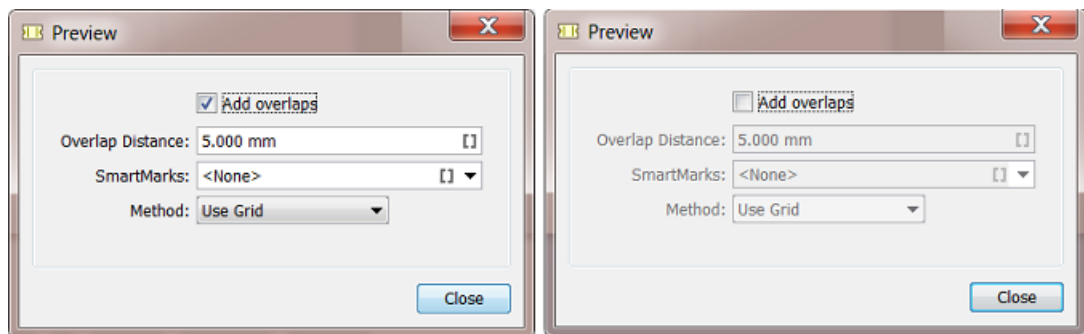
- You can also set a same condition for multiple parameters.
- Select multiple parameters, right-click any of them and choose **Conditional Availability**.



- The dialog shows message you that what you now apply will be valid for all the selected parameters.

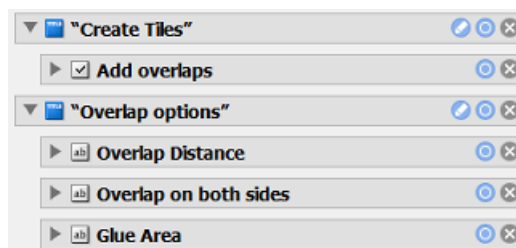


- Similar to the settings we showed above, using the **Show** or the **Enable** option, you can now get these example results:

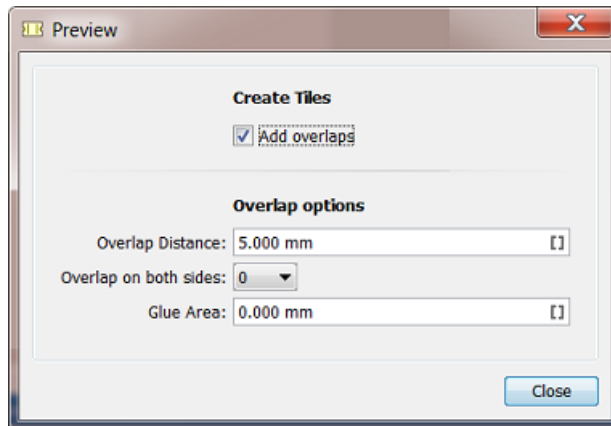


Example 3: Conditional Availability of a Group of Parameters


- This time we organized 4 parameters in 2 groups and gave these groups a good name:



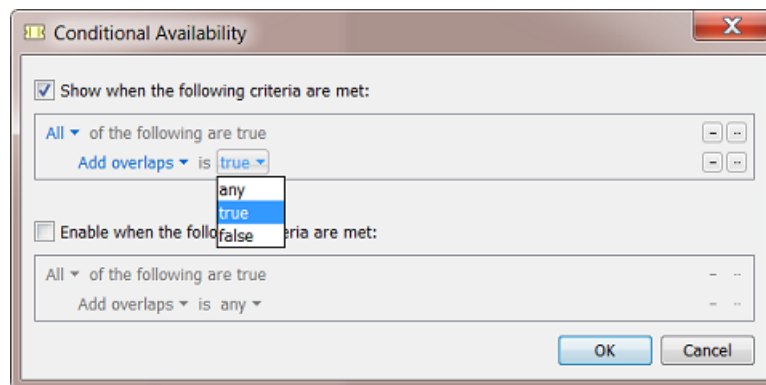
- This results in this preview:



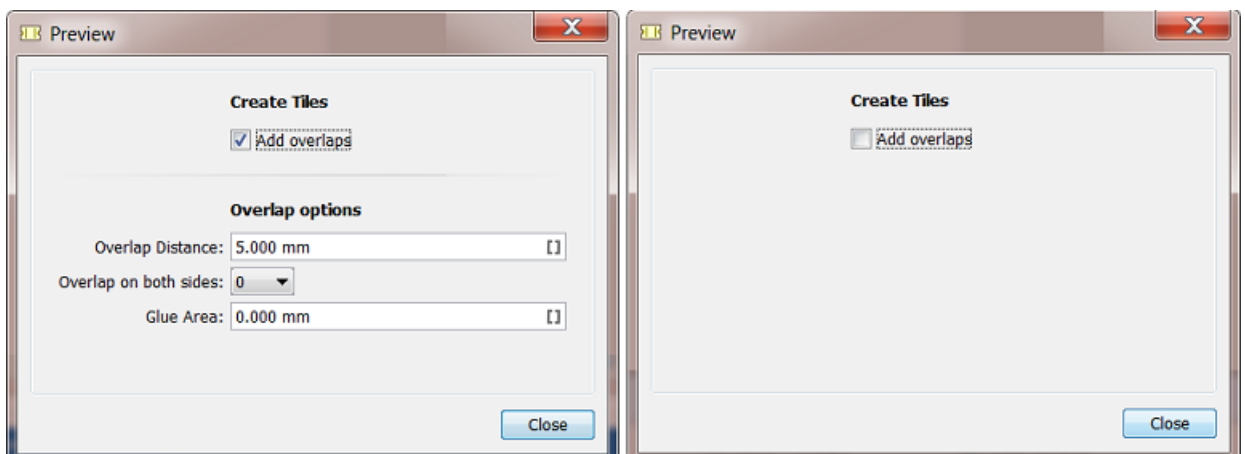
- Because the overlap options only make sense when overlaps are requested, we can now set this condition on group level, instead of setting the same condition for each of those option parameters.

We now click the  of the group 'Overlap options'.

- We choose to only show that group when 'Add overlaps' is selected:

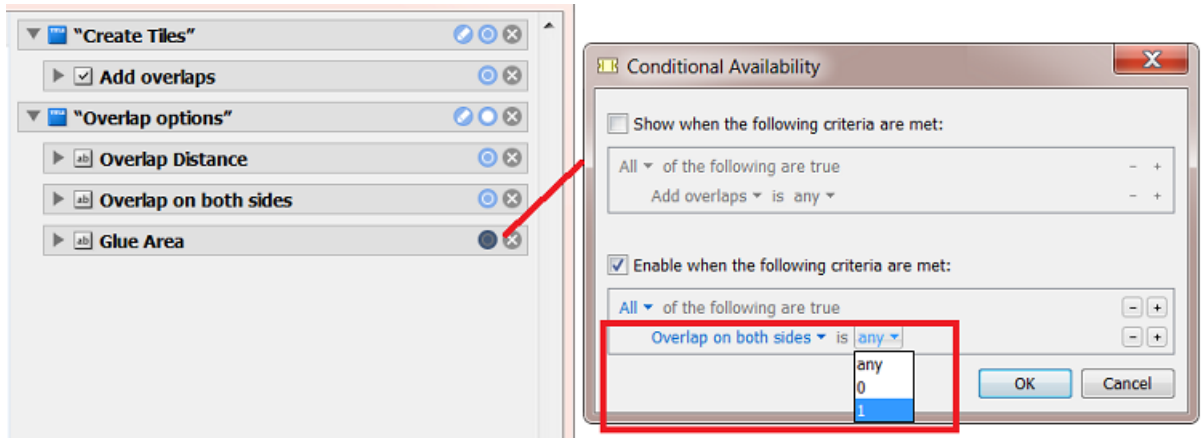


- See the result in both cases:

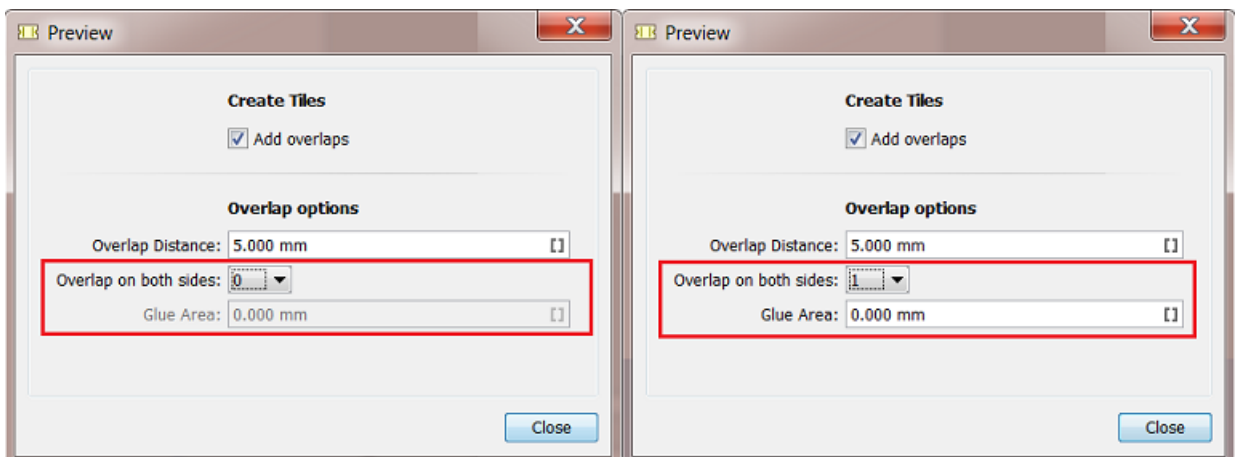


Example 4: Combination

- Building on the above example, we add an extra condition within that group:



- When the group 'Overlap options' is shown, now see the result with either value of the parameter 'Overlap on both sides':



Limitations

These are parameters that require a fixed value, for example yes-no, top-right-bottom-left, etc.) or whose value is another kind of clear unmistakable value, for example (a part of) a text or number, etc.

Some parameters however result in values that cannot be used to base a condition on, for example a numeric field where the value is also defined by its unit or a table, or for example a table of names of layers to be removed. Such parameters cannot be used in a condition ; they will automatically not present themselves in this dialog.



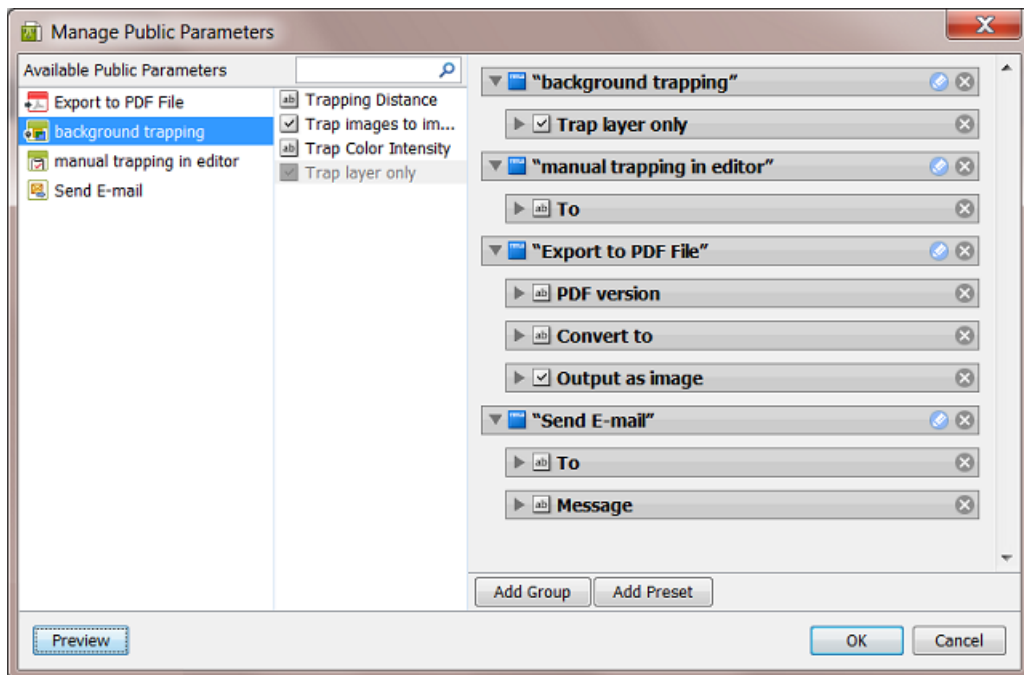
Note: You can use a parameter based on a table when you created predefined settings. for that table. This way, you can base a condition on whether 'Table 1' was selected or 'Table 2', etc.

5.4. Managing Public Parameters in a Workflow

Once you have made parameters public in your workflow steps, you can open the **Manage Public Parameters** dialog to get an overview and make changes.

1. In the workflow editor, go to **Advanced > Manage Public Parameters...**



The dialog shows all public parameters of this workflow, sorted by workflow step:

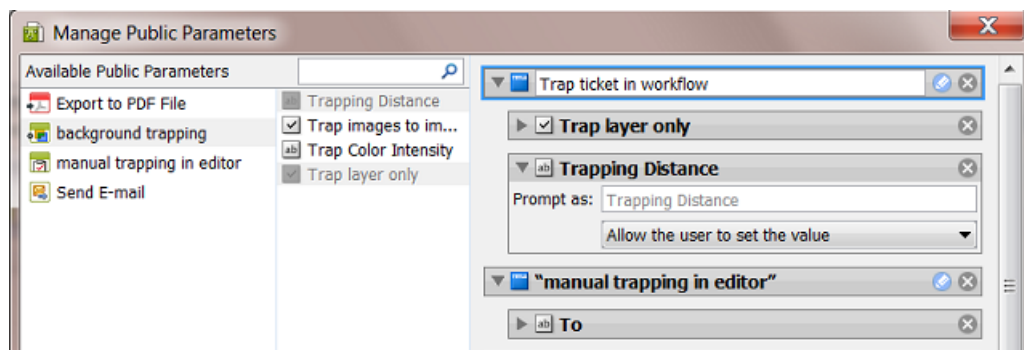


- **The left pane** shows the workflow steps for which you have made parameters public.
- **The middle pane** shows the public parameters of the workflow step that was selected on the left. They are greyed out when they are already used in the right pane. When parameters are not used in the right pane, they will not be public parameters.
- **The right pane** shows all the public parameters in detail (name, type of value, etc.). They are default organized in **Groups** matching the (custom) name of the step where they came from. This is where you build the dialog that the users will see and use to decide the values before launching the workflow.
- The **Preview** button shows how the end-user will see the panel you are building here.

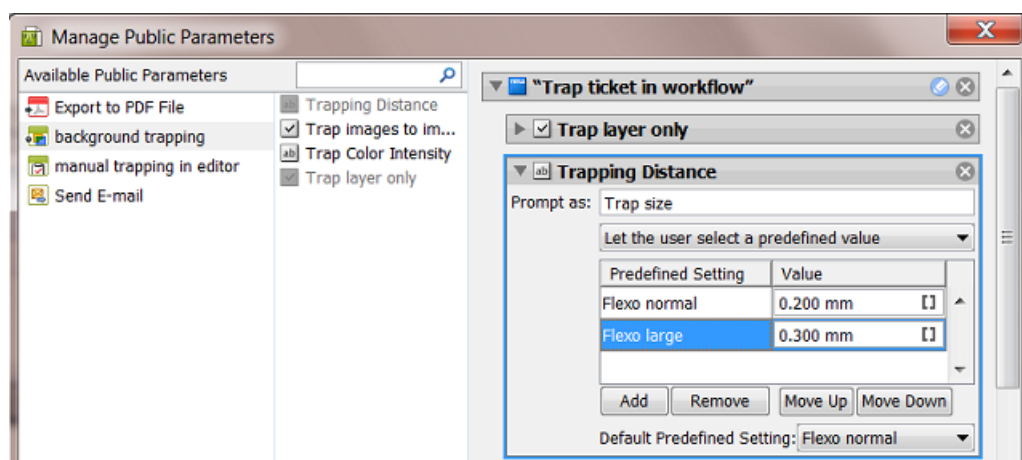
To help you trace their origin, selecting items in the panels also shows selections of related items in the other panels.

2. You can
 - drag parameters from the middle to the right pane to make them public. For example dragging the parameter **Trapping Distance** into the group **"background trapping"**.

- click on  to remove the parameter or group from the right pane. It will then not be public anymore.
- re-arrange the order of your public parameters or groups (drag them up or down).
- click on  to change the default name of the group.



- change the settings of the public parameters. Define in **'Prompt as'** is what the user will see. Instead of **Allowing the user to set a value**, you can offer the user a choice of predefined values. In below example the user will get a drop-down list to choose between **Flexo normal** and **Flexo large**. The user making this choice when launching the workflow will not see the actual values of 0,2 and 0,3 mm.



Use the buttons **Move Up / Down** to define the order of the **Predefined Settings** in the drop-down list that the end-user will see.

Learn more about editing and re-designing this pane in [Making Presets to Simplify the Users' Choices](#) on page 117 and [Example of Further Customizing the Public Parameters Dialog](#) on page 120.

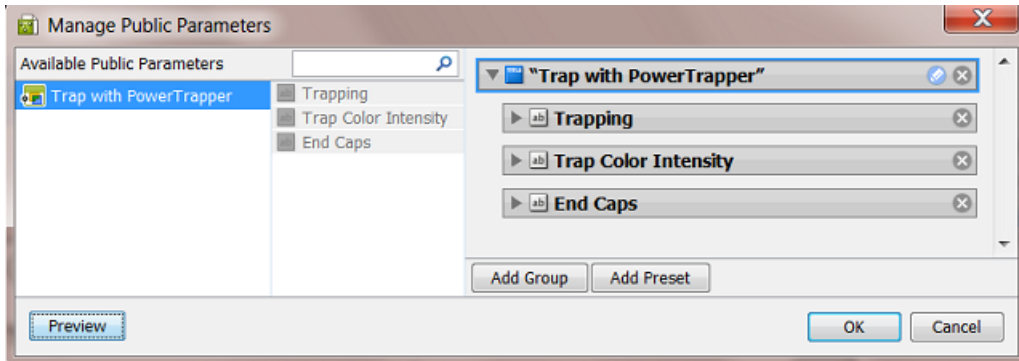
5.5. Making Presets to Simplify the Users' Choices

Presets are customized combinations of parameters. They limit the amount of decisions that users have to make.

Example

Let's assume you want users to use different trapping settings when printing offset versus dry offset. You can collect those trapping settings in a **Preset**, so that users will only have to choose between 'offset' and 'dry offset'.

We start from a dialog of a workflow ticket where the trapping step has 3 public parameters:



1. Click to remove the (default) **Group** named "Trap with PowerTrapper"

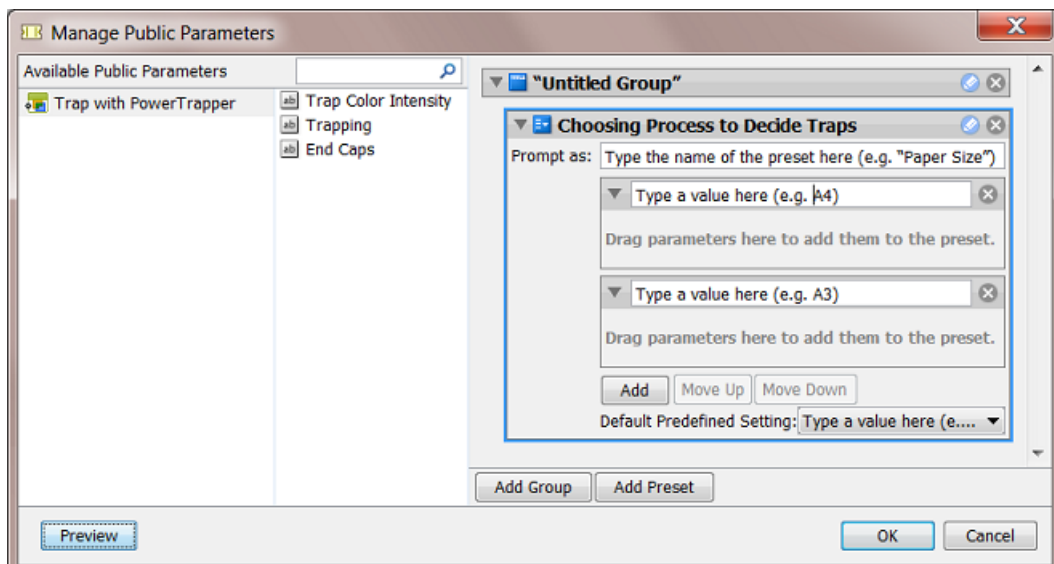
2. Click **Add Group**.

A new "Untitled Group" is created. Click to change its name.

3. Click **Add Preset**.

A new "Preset" is created. Click to change its name.

By default, two example values will appear. The preset was renamed to 'Choosing Process to Decide Traps':



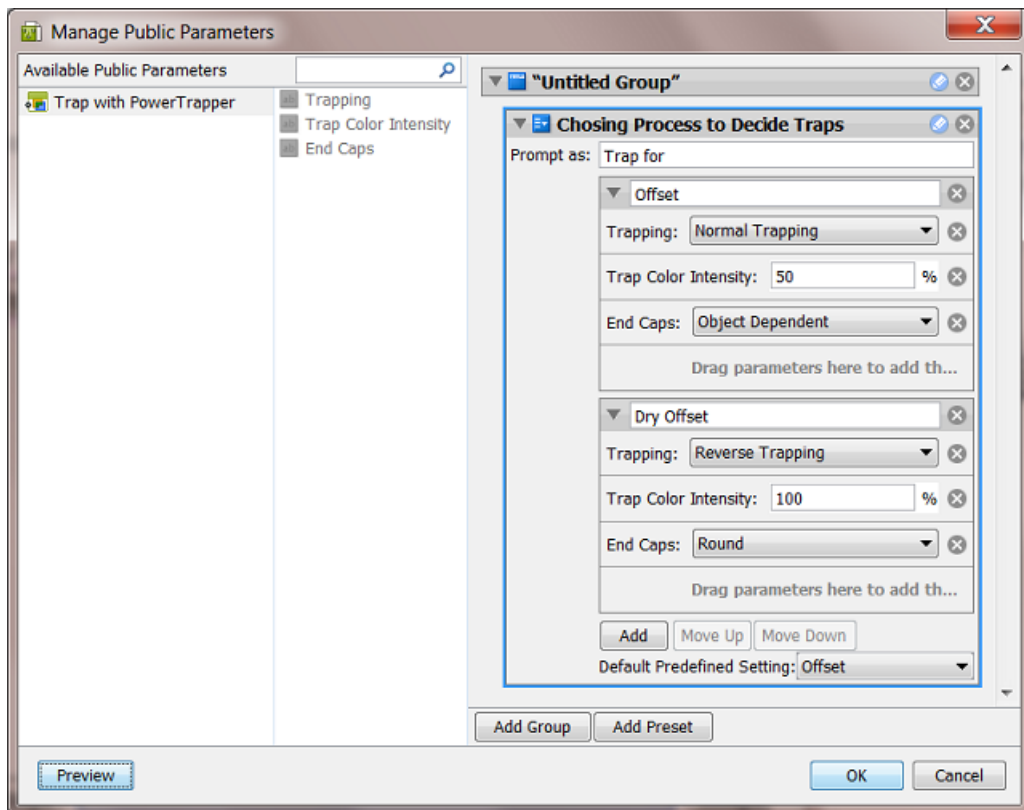
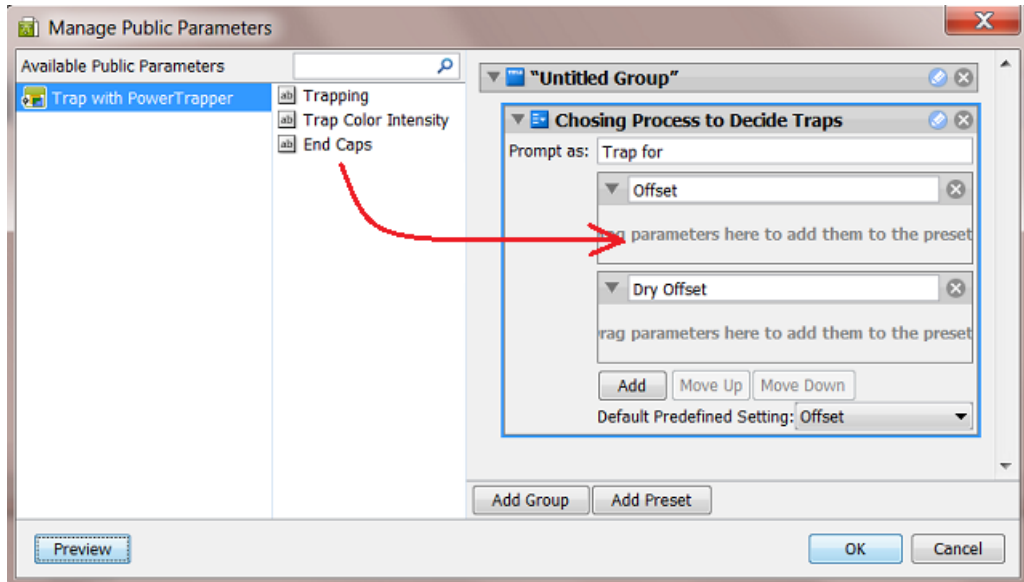
4. Define what the users will see:

a) in **Prompt as**, enter the text that users will see (replacing 'Type the name of the preset here (e.g. "Paper Size")')

b) enter the values that the users will choose from (replacing 'Type a value here (e.g. A4)'). In our example, we add 'Offset' and 'Dry Offset'.

Click **Add** if you want to add extra values (not the case in our example).

5. Drag the parameters from the dialog's middle pane onto one preset item. This will make them automatically appear under each preset item (here both under *Offset* and *Dry Offset*).



6. Define the parameters values for each preset item.
See the above screen shot for a good example.
7. Use the buttons **Move Up / Down** to define the order of the **Presets** in the panel that the end-user will see.
8. Click **OK** to confirm and close the dialog.
9. Save your ticket to also save these public parameters.

This is how the resulting dialog will look like to the users of the client applications:



5.6. Example of Further Customizing the Public Parameters Dialog

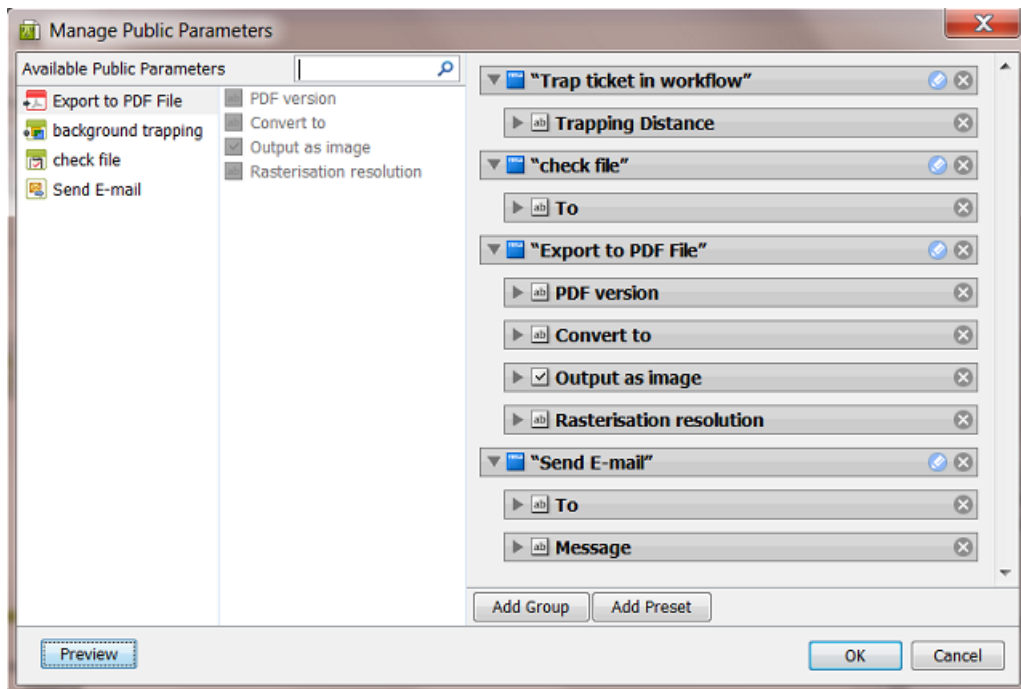
In this example, we further customize the public parameters dialog by reorganizing the **Groups** and adding a **Preset** to combine parameters.

5.6.1. Step 1 - Example Workflow and its Initial Public Parameters Dialog

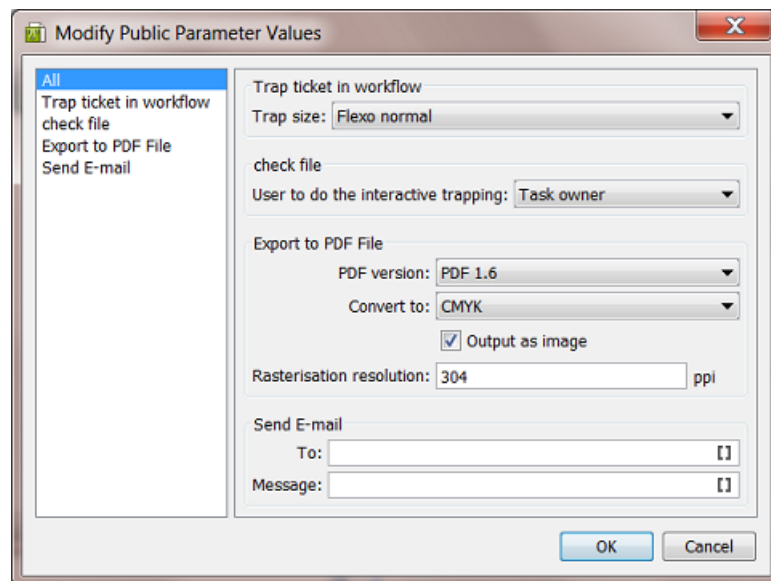
This is our example workflow:



We created public parameters in 4 steps. This is the initial **Manage Public Parameters** dialog, where the groups match the workflow steps:



If you don't customize this, the resulting dialog, that will be shown to the users launching a task, will look like this:



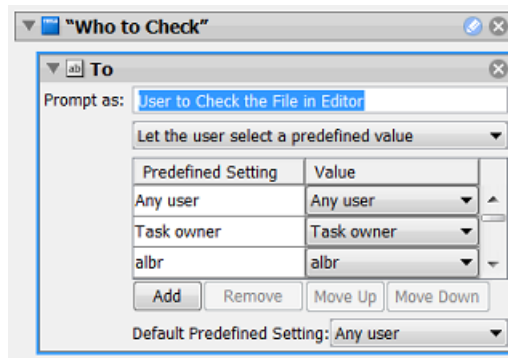
In this dialog, you can see

- that the items on the left pane of this dialog match the groups in the **Manage Public Parameters** dialog.
- that when there are multiple groups, an extra one named 'All' is automatically created. This group offers all parameters together.
- that the parameter 'Trap size' is offered as a choice of presets. This was illustrated [earlier](#).

In the next steps we will further optimize this dialog for the end user.

5.6.2. Step 2 - Customizing the Groups and the Text that the User will See

- In the group "check file",
 - rename the group to "Who to Check",
 - change **Prompt as** to 'User to Check the File in Editor',
 - set the **Default Predefined Setting** to 'Any User'.

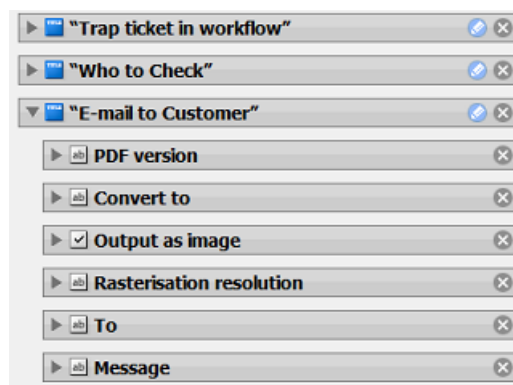


- Press **Add Group** to create a new group and name it "E-mail to Customer".
 - In the right pane, drag the parameters from the groups "Export to PDF File" and "Send E-mail" into this new group.



Tip: You can press **Shift** to select and drag multiple parameters.

- Remove these two groups that are now empty: "Export to PDF File" and "Send E-mail". This is the result:



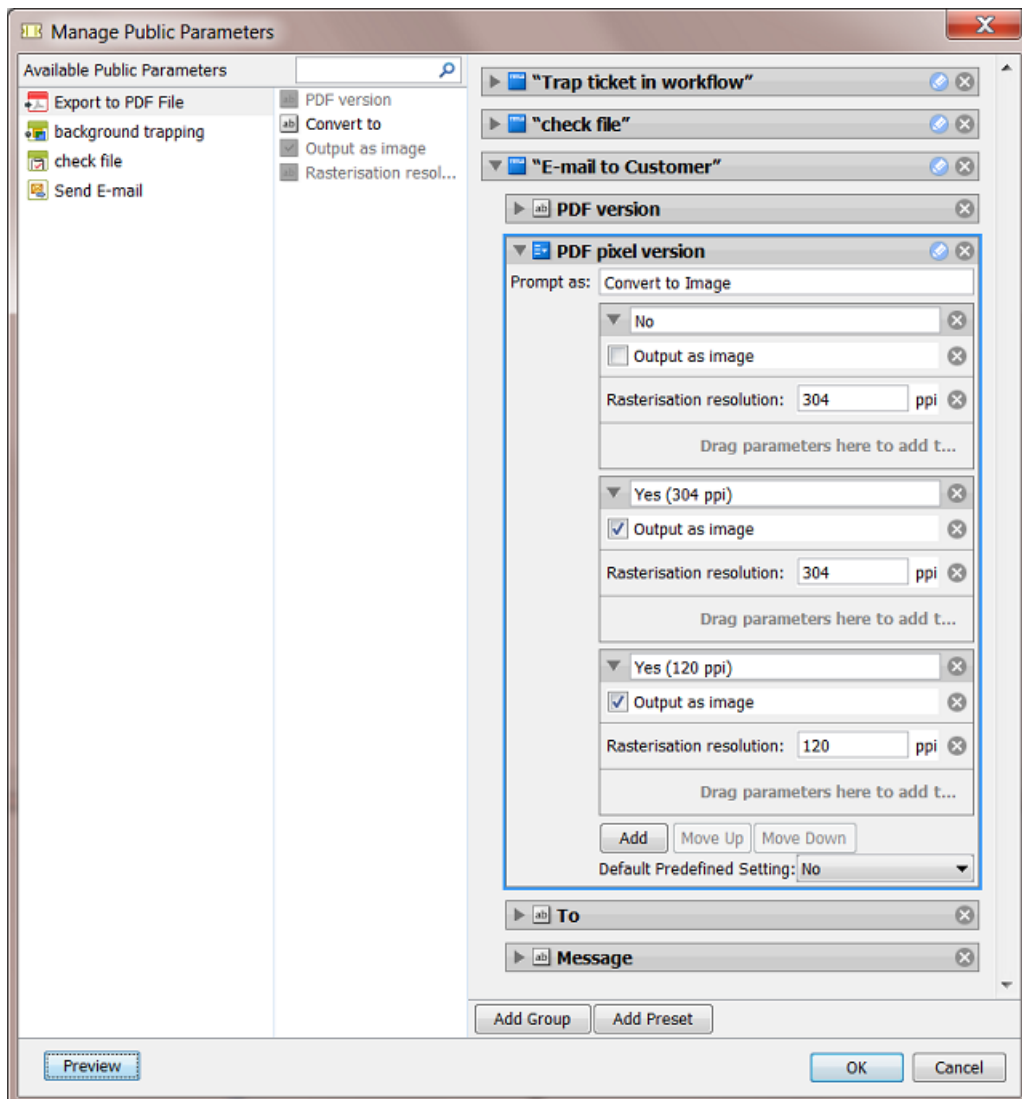


Note: An alternative way to do this, is to first delete the 2 groups, then create the new one and then drag the 5 parameters from the middle pane into the new group.

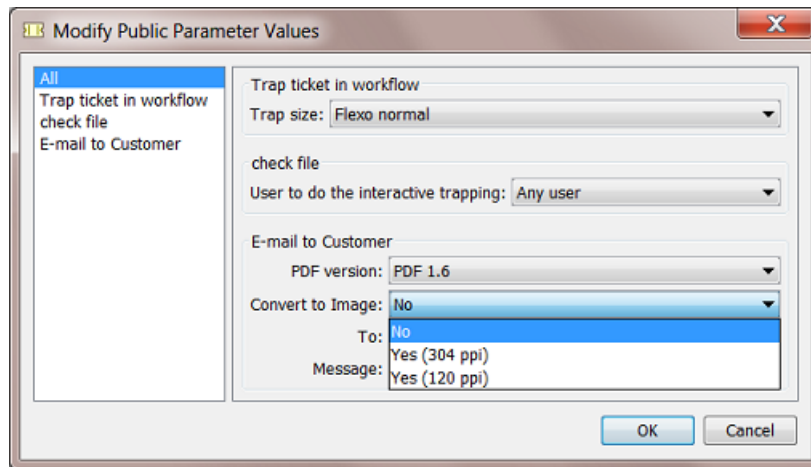
5.6.3. Step 3 - Adding a Preset and Combining Parameters

In the new **Group** "E-mail to Customer", we now combine 2 parameters into a new **Preset**:

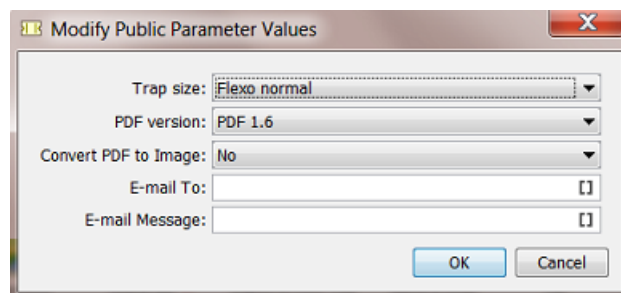
- Press **Add Preset**. A new preset appears at the bottom of the bottom group. Rename it to 'PDF pixel version'.
- Set the **Prompt As** to 'Convert to Image'
- Press **Add** to add a third parameter and rename the 3 parameters to 'No', 'Yes (304 ppi)' and 'Yes (120 ppi)'.
- Drag the parameters 'Output as Image' and 'Rasterization Resolution' onto the first value of this new preset.
- Adapt the parameter values as shown in this screen shot:



For the user launching the workflow, the resulting dialog already looks a bit simpler:



In this example, the list of parameters is not that long. That is why you could also decide to *only* offer the **All** view in this dialog. You can do this easily by moving all parameters into one group. The result can look like this:



Note: Many tasks have Tickets with a conditional user interface: where settings depend on another setting. For example settings that only become available after you first make another choice. Example: 'Apply color conversion?'. If 'Yes', then there is the choice 'Convert to CMYK' or 'Convert to RGB'.

These kind of conditions can not be created in the **Manage Public Parameters** dialog. If you wish to recreate such a user decision, you have to create a preset that offers a choice for each combination. In our example this would mean a drop-down list with these 3 choices: 'No color conversion', 'Convert to CMYK' and 'Convert to RGB'.

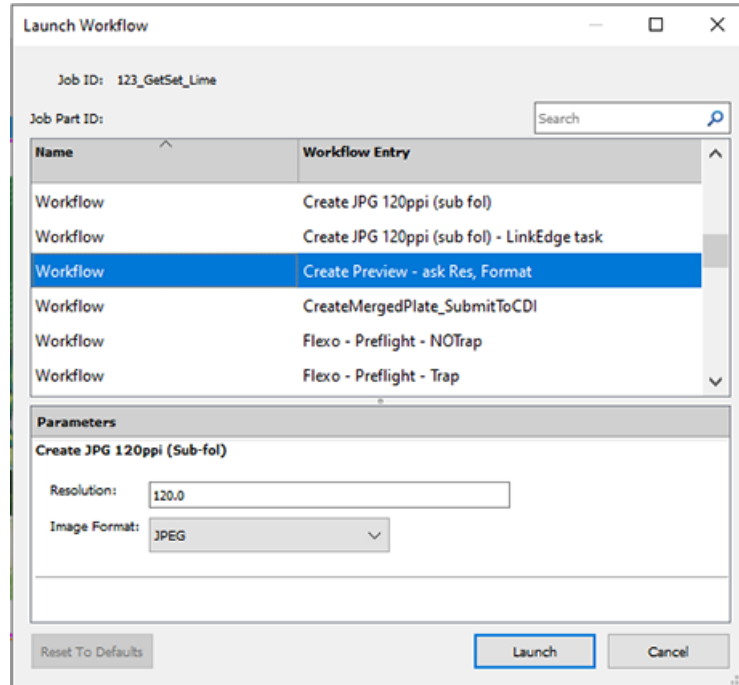
5.7. Launching a Workflow with Public Parameters

Launching a workflow with public parameters is very similar to how it is done with single task tickets.

Here are some examples:

Example when launching from ArtPro+

In the **Launch Workflow** dialog, ArtPro+ enables to launch a task or workflow on the open file. Only public tickets will be shown and from a selected ticket, only its public parameters.



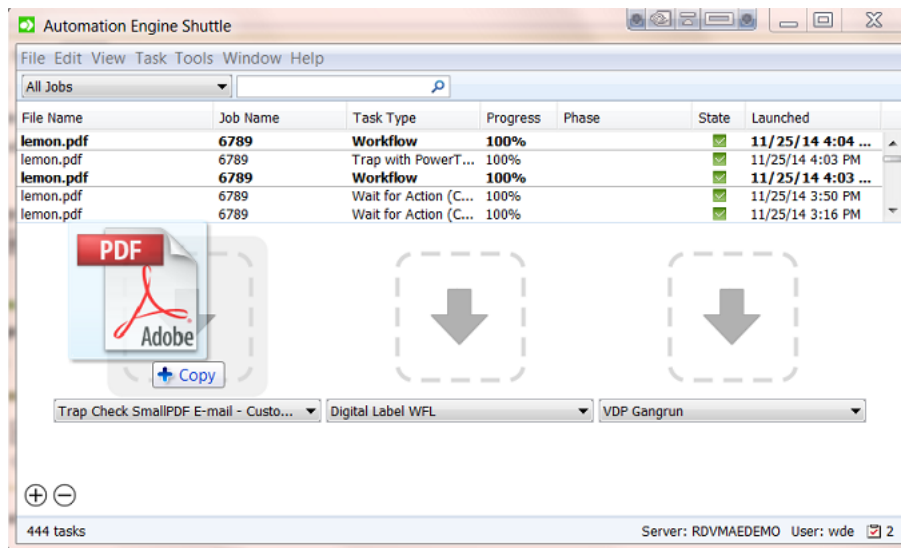
Learn more in the [ArtPro+ user guide: Working with Files > AE connection](#).

Launching public workflows from My Workspace

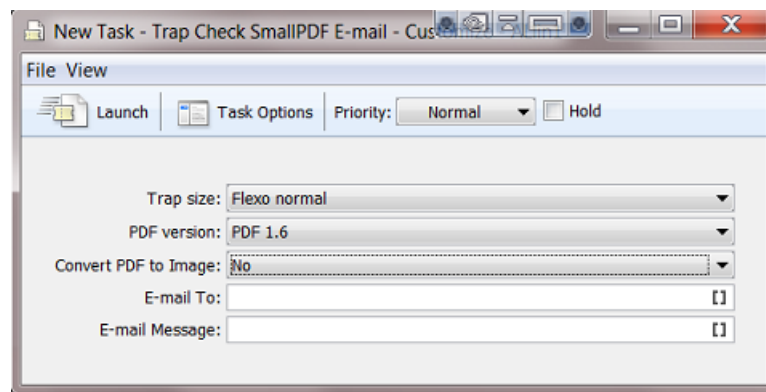
Users of the simplified interface that is offered in the browser client can also launch workflows. Learn more in the chapter [Browser Client Workspaces](#).

Example when launching from Shuttle

Drop the input file on the workflow in the **Launch panel**.



If you drop a file on the launch field of a workflow that has public parameters, a dialog will open asking you to decide these public parameters:

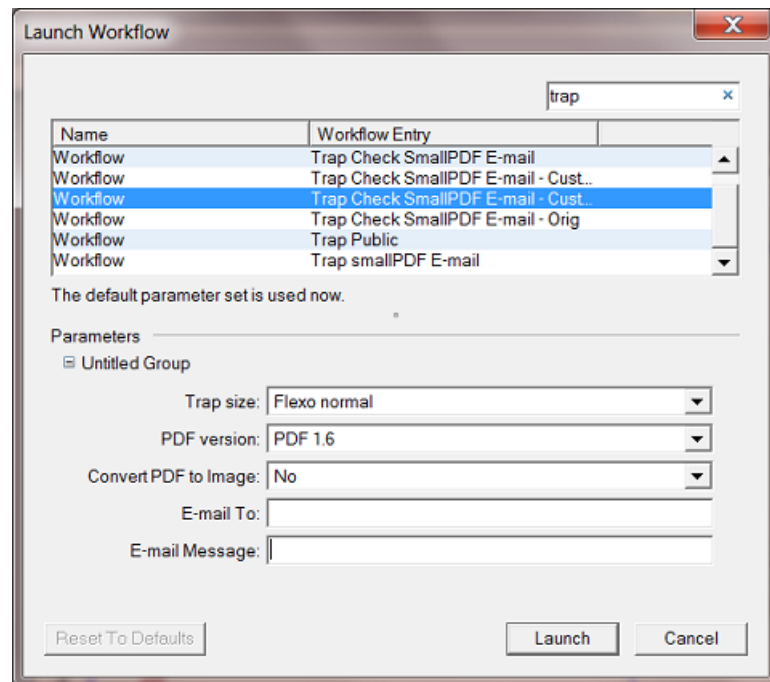


Click **Launch** when ready.

Learn more about all variants of launching workflows from Shuttle client applications in [Using Shuttle to Upload and Process Files](#).

Example when launching from PackEdge (legacy editor)

Choose **File > Launch Workflow** from the menu, select the workflow, decide the public parameters and press **Launch**.



Example when launching from ArtPro (legacy editor)

In the **Launch Workflow** dialog, ArtPro offers extra functionality to convert the ArtPro file before it enters the workflow and to set a Job or Product context (which is important when using SmartNames).

Launch Workflow

Launch as:

Job:

Product:

Job ID: Job Part ID:

Workflow

Name	Workflow Entry	Server
Workflow	Call Webservice Demo	rdalpha25
Workflow	Trap Check SmallPDF E-mail	rdvmaedemo
Workflow	Trap Check SmallPDF E-mail - Customize	rdvmaedemo
Workflow	Trap Check SmallPDF E-mail - Customize - ALLin1	rdvmaedemo
Workflow	Trap Check SmallPDF E-mail - Orig	rdvmaedemo
Workflow	Trap smallPDF E-mail	rdvmaedemo

The default parameter set is used now.

Parameters

▼ Untitled Group

Trap size:

PDF version:

Convert PDF to Image:

E-mail To:

E-mail Message:

6. Using Workflow Parameters

6.1. Concept

Workflow Parameters are parameters that are specific for a workflow. Their values are picked up by steps in the workflow.

For example: You print displays and you mostly use the same output workflow. Your workflow ticket has 3 workflow parameters: `substrate`, `sheet size` and `quantity`. The values of these parameters can be typed in by a user or can be picked up from an external source.

Workflow parameters show up as SmartNames starting with `[wfp.]`. Much like Job and Product parameters, you can use them by inserting them in the SmartName enabled field in your workflow steps.

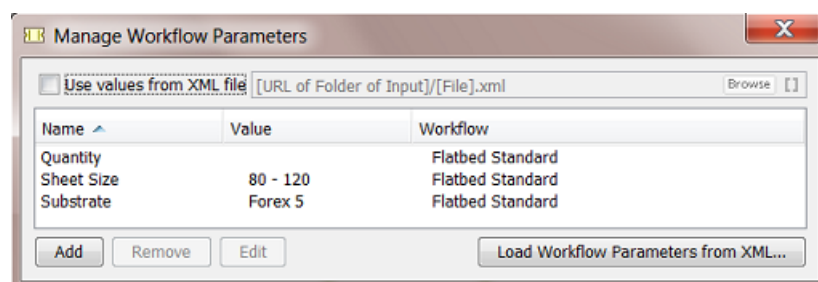
In many cases, you could accomplish the same by using Job or Product parameters, but some Automation Engine customers choose to not use the Jobs or Products concept.

Workflow Parameters are Loaded at Workflow Start-Up, but can Change Later

Workflow parameters and their values are loaded when the workflow starts up. However they can change during the workflow. This can happen via a user intervention or more automated steps. Learn more in [reading their value from an XML file](#) and in [modifying workflow parameters during a workflow](#).

Managing Workflow Parameters

In the workflow editor, go to **Advanced / Tools > Manage Workflow Parameters....**



- The **Manage Workflow Parameters** dialog is where you **Add**, **Remove** and **Edit** workflow parameters.
- This dialog lists all the workflow parameters defined in the current workflow and all of its subworkflows.



Note: If you are in a subworkflow, the workflow parameters defined in the master workflow will not be shown in this dialog.

- You can keep this dialog open while making changes in the current or subworkflow(s) and while navigating to and from subworkflows. The list of workflow parameters is instantly updated to reflect changes.

Workflow Parameters can be Different per Group of Files

Workflow parameters do not have to be valid for *all* files in that workflow ; they can be different per file or per group of files.

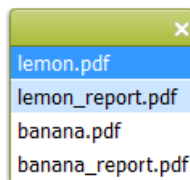
For example: The banana and lemon displays share the same `substrate` and `sheet_size`, but have a different value for the parameter `quantity`.

Workflow parameters are valid for one file or for a group of files. You can see such grouping of files when you open a Widget and select an item in its list. This will not only highlight the selected item (in blue) but also the other items belonging to that group (in light blue).



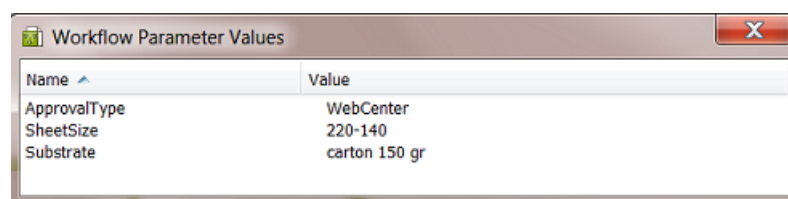
Note: Such a group is decided by the workflow system. It attributes "tokens" to each output file (an internal ID number). Files with a same "token" will in that part of the workflow seem to be grouped. Learn more about this concept in [Understanding Tokens \(Grouping of Output Files\)](#) on page 67.

For example: When you select `lemon.pdf`, `lemon_report.pdf` also gets highlighted. This means that they belong together in a group. This means that they share the same workflow parameters (at that point in the workflow).



As mentioned above, these groups are created automatically by the file management logic in the workflow steps. However, sometimes a setting lets you decide how these groups need to be created. A typical example is the [Data Collector](#) on page 97.

When you right-click on an item in a widget and select **Workflow Parameter Values**, you will see the values for that group of files at that point in the workflow. An example:

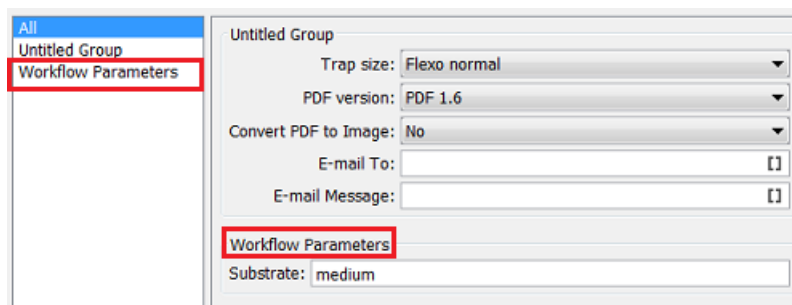


Workflow Parameters versus Public Parameters

Public Parameters serve to simplify the decisions users have to make when they launch workflows from client applications. **Workflow Parameters** serve to make your workflow more automated.

Workflow parameters are also SmartNames, public parameters are not.

In the **Manage Public parameters** dialog that is shown to the user launching the workflow, workflow parameters show up as a separate parameter group named **Workflow Parameters**:



You can easily remove workflow parameters from the list of public parameters using the **Manage Public Parameters dialog**. Learn more about public parameters in [Using Public Parameters in Workflows](#) on page 107.



Note: There is an exception: If you [use workflow parameters values from an XML file](#), the workflow parameters will not be public.

6.2. Most Typical Use Cases for Using Workflow Parameters

1. Automating workflows by Loading Workflow Parameters from an XML file

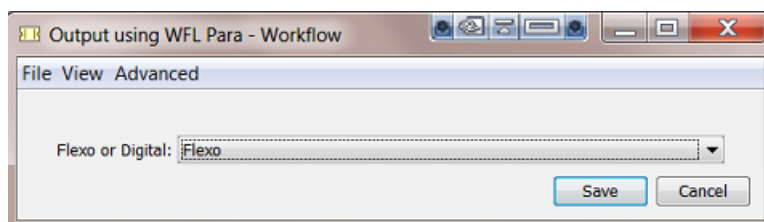
This is the case where the order management or production planning system sends Automation Engine an XML with the details of the output workflow. These production specifics are automatically picked up and serve as parameters for the workflow. Learn more in [Using Workflow Parameter Values from XML](#) on page 135.

2. Using Workflow Parameters as a Smart Public Parameter

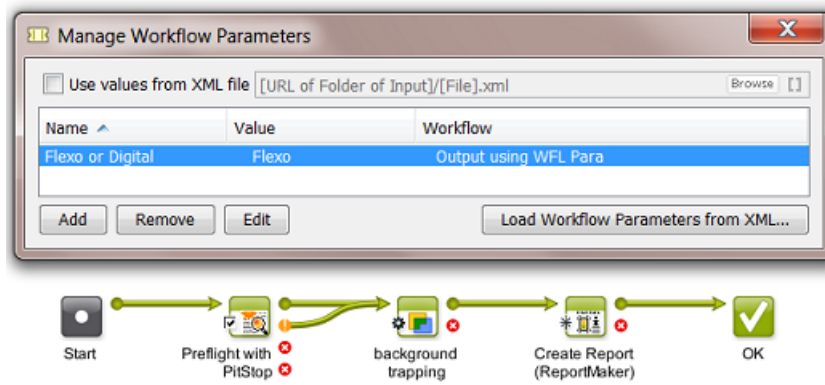
When users enter the value of workflow parameters manually, it usually resembles the case of a using public parameters, but then also taking advantage of the referencing power of a SmartName.

For example:

Your users need to decide if the workflow is flexo or digital. You want them to answer this one question as a public parameter:



This is only 1 question but we will use the answer for 3 different decisions in our workflow, once in each of below workflow steps:



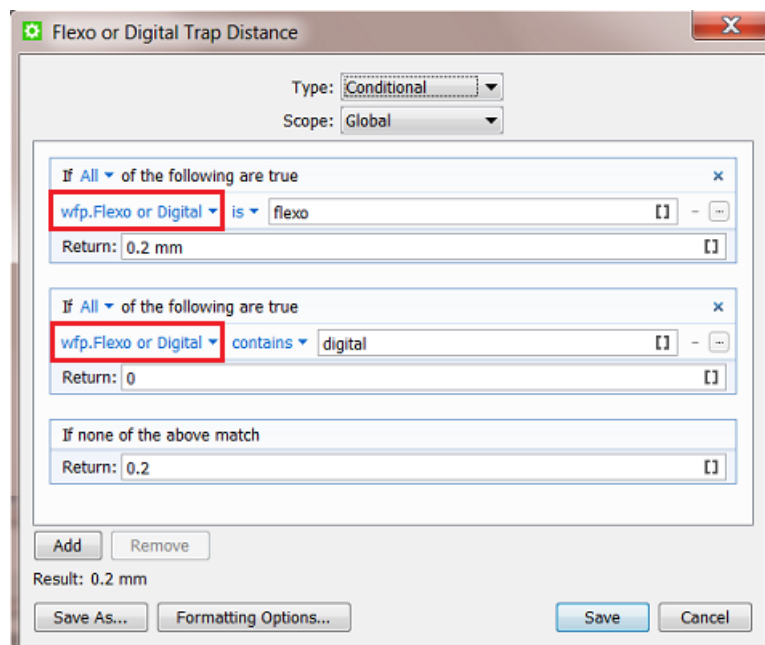
Here is an example how this can be done. We also illustrate two different levels of complexity:

- You **use the value of the workflow parameter as text**. For example the Preflight task, depending on the user's answer, uses the profile `flexo.ppp` or `digital.ppp`. In this case you pick up the workflow parameter as a SmartName:

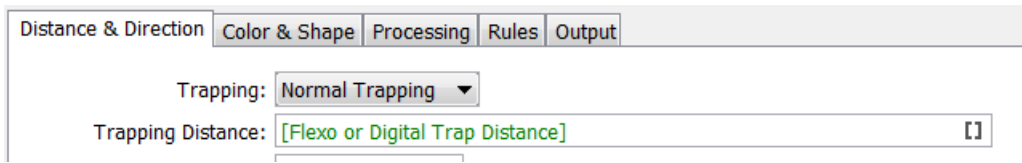
Profile: `naedemo/PitStop/PDF Profiles/Custom/[wfp.Flexo or Digital].ppp`

In the same way, the ReportMaker task uses `flexo.pdf` or `digital.pdf` as template. One decision from the user so already decided values in 2 tickets.

- You can also **use value of the workflow parameter inside another SmartName**. For example this conditional SmartName to define the trapping distance:



You then use that conditional SmartName in the trapping task:



In this example, when you do not use a workflow parameter, the user launching the task would have to decide 3 different public parameters. Now, the user is only asked 1 decision.

6.3. Creating Workflow Parameters Manually

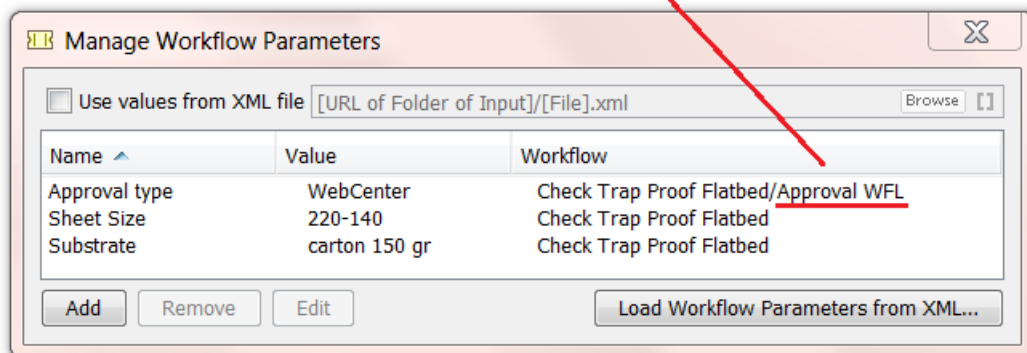
Follow these steps to create workflow parameters manually:

1. Open the (sub)workflow in the workflow editor.
2. Open **Advanced / Tools > Manage Workflow Parameters....**
3. In the **Manage Workflow Parameters**, click on **Add** and fill in the **Name** and **Value**.



Note: You can leave the value empty at this time. The value has to be defined at the latest by the time the workflow is launched (for example when a user who launches the workflow fills in the value in the dialog of public parameters).

In the dialog, the column **Workflow** shows if the workflow parameter is present in the current workflow or was created in a subworkflow.



Note: If you need to add many workflow parameters and if their names are already present in an XML file, you can load that list of names by using the button **Load Workflow Parameters from XML....** Learn more [here](#).

4. Save your workflow to also save your workflow parameters.

6.4. Loading Workflow Parameters from XML

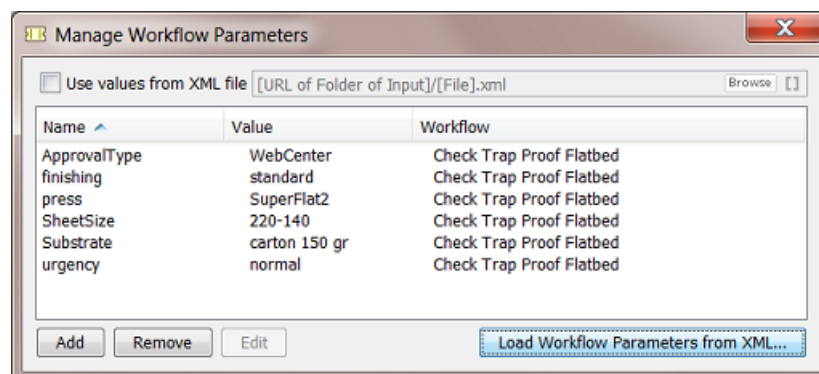
Instead of typing them in manually, you can load **Workflow Parameters** from a XML file. You can select all parameters from the XML file or only those of a specific XML Element. After loading them, you can keep the values that came from the XML file or change them.

This is an example of a valid XML file:

```
<?xml version="1.0"?>
- <job>
  - <production>
    <ApprovalType>WebCenter</ApprovalType>
    <Substrate>carton 150 gr</Substrate>
    <SheetSize>220-140</SheetSize>
    <press>SuperFlat2</press>
    <finishing>standard</finishing>
    <urgency>normal</urgency>
  </production>
  - <customer>
    <name>FruitCo</name>
    <number>2712</number>
    <currency>USD</currency>
  </customer>
</job>
```

Follow these steps to load only the items of the XML Element 'production' :

1. Open the (sub)workflow in the workflow editor.
2. Open **Advanced / Tools > Manage Workflow Parameters....**
3. Click **Load Workflow Parameters from XML....**
4. Browse to the XML file, select it and click **Open**.
A dialog pops up asking you to specify an XML **Element** or to leave the field blank to select all items in the Root Element of the XML file.
5. In that dialog, type in the element name `production` and click **OK** to confirm.
This will load these workflow parameters:




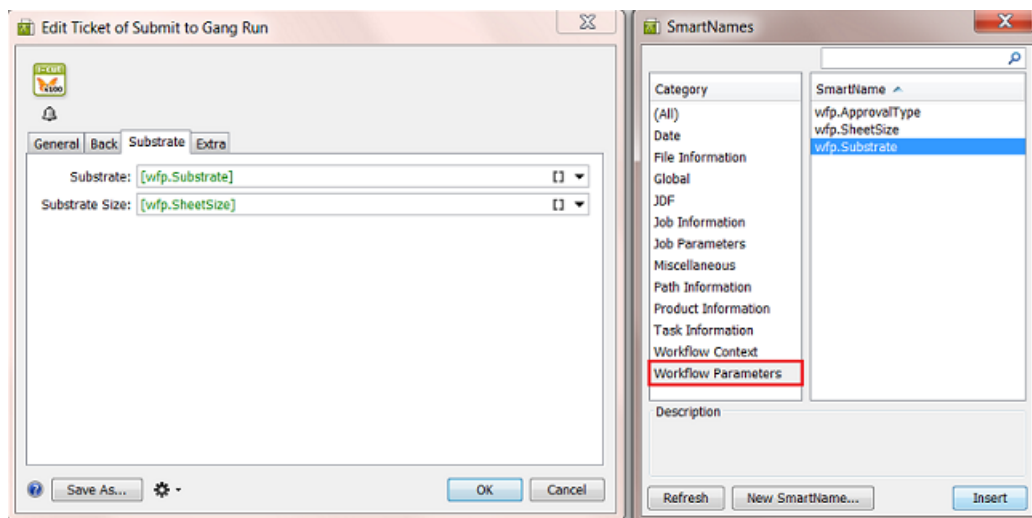
6. If you want you can make any modifications like changing a value or even removing a parameter. In our example, we don't need the parameter 'finishing' so we can remove it.

7. Save your workflow to also save your workflow parameters.

6.5. Using Workflow Parameters

Once you have created them, you use the workflow parameters by picking up their SmartName version in the workflow step ticket.

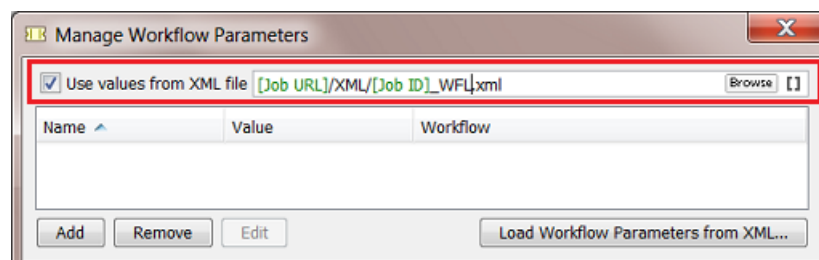
1. Open the (sub)workflow in the workflow editor.
2. Double-click the ticket in which you want to use a workflow parameter.
3. Click on  in the SmartName-enabled field where you want to add the workflow parameter. In the SmartNames Browser, select the **Category 'Workflow Parameters'**



4. Select the workflow parameter [wfp.] that you want and click **Insert** (or double-click) to add it to the ticket.
5. Click **OK** to confirm and close the ticket.
6. Save your workflow.

6.6. Using Workflow Parameter Values from XML

Workflow parameters and their values can be defined by an XML file.



This is an example of a valid XML file:

```
<?xml version="1.0"?>
- <job>
  - <production>
    <ApprovalType>WebCenter</ApprovalType>
    <Substrate>carton 150 gr</Substrate>
    <SheetSize>220-140</SheetSize>
    <press>SuperFlat2</press>
    <finishing>standard</finishing>
    <urgency>normal</urgency>
  </production>
  - <customer>
    <name>FruitCo</name>
    <number>2712</number>
    <currency>USD</currency>
  </customer>
</job>
```

In this concept, there are 2 differences versus [creating workflow parameters manually](#) or [loading them once from an XML](#):

- the workflow will load and re-read this XML file at every step of the workflow. This also means
 - that you will probably not use the step [Modify Workflow Parameter Values](#) in this workflow because its modifications would be overruled.
 - that, during the workflow, the XML file can be updated with new or changed values.
- the resulting workflow parameters will not be public parameters. The concept here is that it is not a user who decides the values, but an XML file.



Note: A subworkflow gets the workflow parameters from its parent workflow that loads an XML, unless it loads values from an XML file itself.

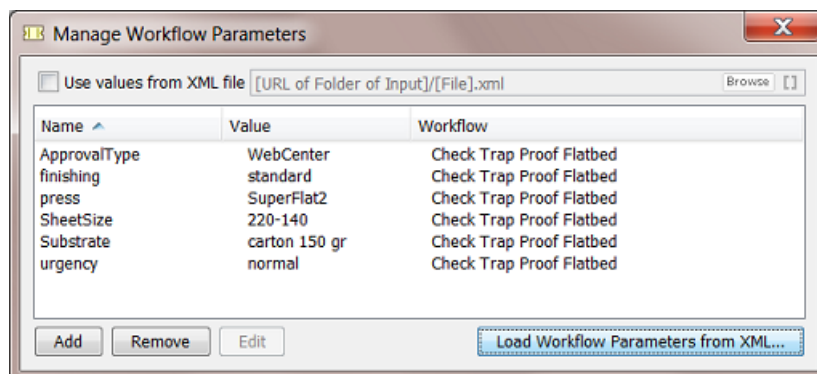
Example

Follow these steps to set up a workflow where workflow parameters values are used from an XML file:



Attention: Before we can ask a workflow to use the XML file, we first have to insert the workflow parameters as [wfp.] SmartNames into the workflow step tickets. To be able to do that, we will first load a sample of that XML and so create these required SmartNames first.

1. Open the (sub)workflow in the workflow editor.
2. Open **Advanced / Tools > Manage Workflow Parameters...**
3. Click **Load Workflow Parameters from XML....** Browse to your sample XML file, select it and click **Open**.
A dialog pops up asking you to specify an XML **Element** or to leave the field blank to select all items in the Root Element of the XML file.
4. Using the example XML mentioned in [Loading Workflow Parameters from XML](#), type in the element `production` in that dialog and click **OK** to confirm.
This will load these workflow parameters and create a [wfp.] SmartName of each of them:



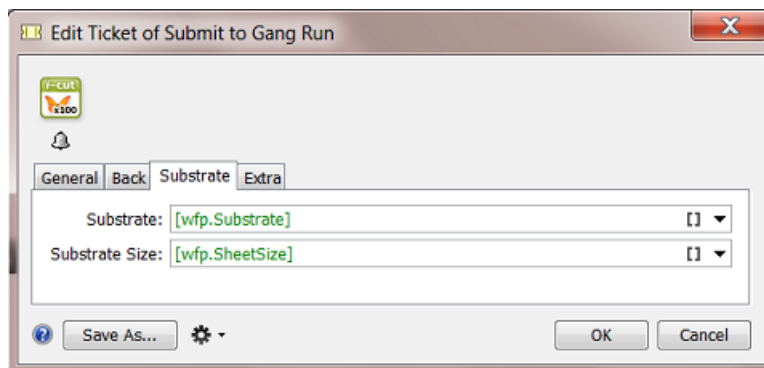
- From that list, remove any parameters that you will not be using.



Note:

- The **Values** in this list will not be used anyway, they will come from the XML file (see step 7).
- The **Names** in this list are the basis for the [wfp.] SmartNames. So do not delete them if you need the SmartName version.

- Use the resulting SmartNames in your workflow step tickets (as shown in [Using Workflow Parameters](#) on page 135).
An example:



- In the **Manage Workflow Parameters** dialog, check **Use values from XML file**. You will be asked to confirm this step. Click **OK**.
- Select the XML file using **Browse** or use SmartNames. This is the place and name that the workflow will look for at the start of every step of the workflow.
- Save your workflow to also save this setup of your workflow parameters.

6.7. Changing the Value of a Workflow Parameter

You can easily change the value of a workflow parameter by using the **Edit** or (Mac) button in the **Manage Workflow Parameters** dialog.

When would you change the value of a workflow parameter?

When you are testing a workflow while staying in the workflow editor. You edit the parameter value, relaunch the workflow (🔄) and monitor the result.

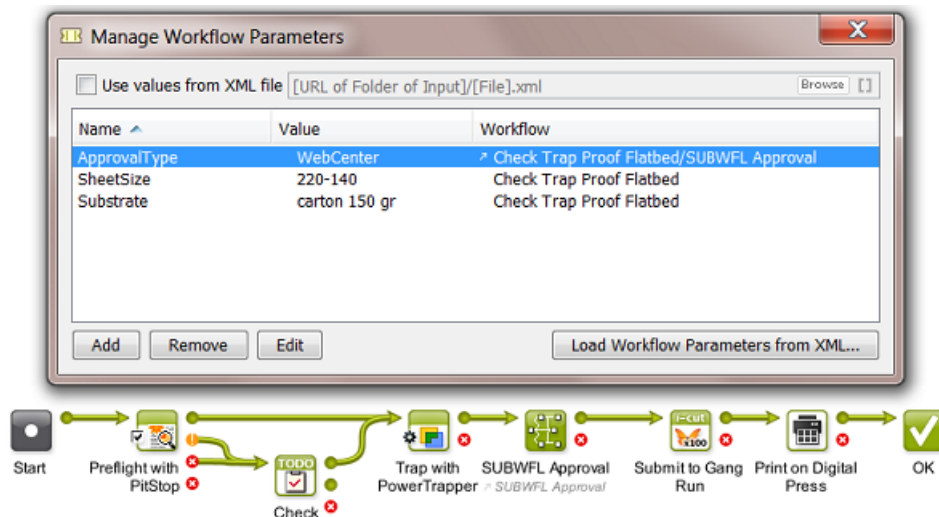


Tip: When *using values from an XML file*, some administrators use this "edit and test" method to learn how the XML file should look like.

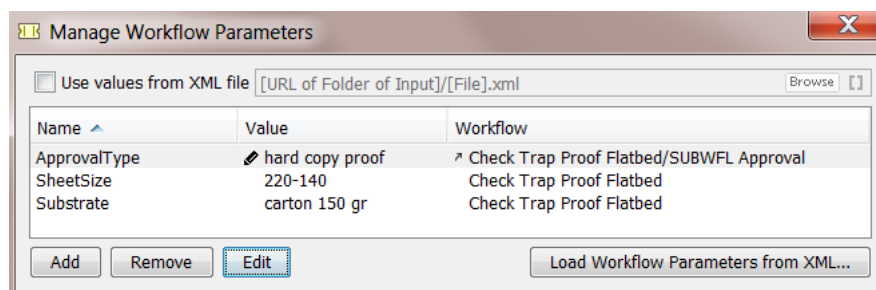
Changing the value of a workflow parameter from a referenced subworkflow

If you edit the value of a workflow parameter that is defined in a referenced workflow, its value will be overruled with the new value without changing that value in the referenced workflow. In the **Manage Workflow Parameters** dialog, a pencil icon ✎ will appear next to the workflow parameter that was overruled.

For example, our workflow named *Check Trap Proof Flatbed* has a referenced subworkflow named *SUBWFL Approval* in which a workflow parameter *ApprovalType* has the value *WebCenter*:



When you now edit this value, for example into *hard copy proof*, this icon ✎ will appear next to the value to indicate that this parameter was overruled:



6.8. Modifying Workflow Parameters During a Workflow

The **Value** of a workflow parameter can be modified while the workflow is running. This can be done because

- the value changed or was empty before and has now 'arrived'. This change can be decided by a user or by an XML file from an external system.
- the new value did not exist yet because it is based on an result of a previous workflow step.

These workflow steps can modify workflow parameters:

- The workflow control **Modify Workflow Parameter Values**. Learn more in [Modify Workflow Parameter Values](#) on page 75.
- The **Integrate with WebCenter** task.
- The **Create or Modify WebCenter Project** task.
- The **Publish on WebCenter** task.


These WebCenter related tasks are then part of workflows that use the result of their interaction with WebCenter as a workflow parameter value in one of the next steps of that workflow.

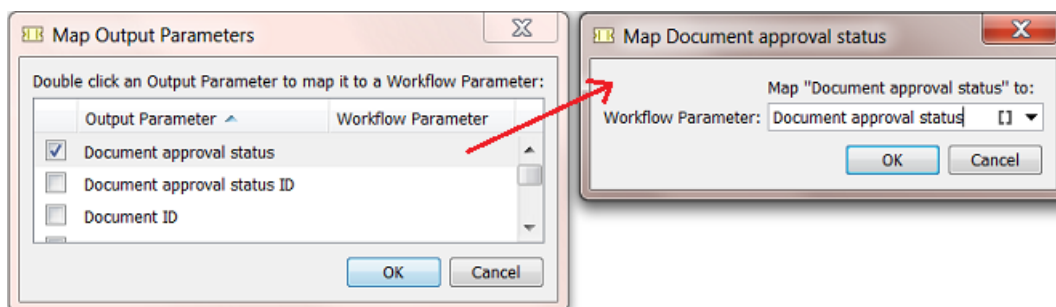
Using the Integrate with WebCenter task

Some **Actions** in the **Integrate with WebCenter** task allow you to use the output of the Action to modify the value of a workflow parameter.

To do this, you need to **Map** one of the task's **Output Parameters** to a workflow parameter.

For example, a workflow asks WebCenter what the approval status is of a document and then uses that status information in the next part of the workflow:

1. In the workflow editor, in the **Integrate with WebCenter** ticket, select an **Action**. For example '**Get Document History**'.
2. Click on the Tools icon  and choose **Map Output Parameters....** A dialog will open.
3. Double-click an **Output Parameter**. For example **Document approval status**.
4. Define to which workflow parameter this **Output Parameter** needs to be mapped to. Select it from the drop-down list or from the **SmartNames Browser** or type in a (new) name. If the workflow parameter does not exist yet, you will be asked to confirm its creation.



Learn more about the **Integrate with WebCenter** task [here](#).

Using the Create or Modify WebCenter Project Task

The **Create or Modify WebCenter Project** task allows you to modify the value of a workflow parameter with the value of the **Project ID** of that WebCenter Project.

To map the **Project ID** to a workflow parameter, use the same **Map Output Parameters** dialog as explained above (point 2).

For example, you did not create the WebCenter project but you want to know the ID of this project because you will publish data to that project later in your workflow.

Learn more about the **Create or Modify WebCenter Project** task [here](#).

Using the Publish on WebCenter Task

The **Publish on WebCenter** task allows you to modify the value of a specific workflow parameter with the value of the **Document Version ID** of the published document in WebCenter.

To map the **Document Version ID** to a specific workflow parameter, use the **Map Output Parameters** dialog, as explained as explained above (point 2).

For example:

You use WebCenter in your approval workflow. You have already published your production PDF twice to WebCenter and the customer still wants changes. You make more changes and as usual overwrite the PDF before publishing again to WebCenter.

Now the customer checks this 3rd version and changes his mind: he says that he liked version 2 better. This is a problem because nobody has version 2 any more. Remember that the document versions that WebCenter keeps are versions of only the View Data.

This can be solved when, after publishing your PDF, you ask WebCenter the document version ID and then make a copy of your production file and add that version number in its name. In this way, you also keep versions of your production PDFs.

Learn more about the **Publish on WebCenter** task [here](#).

7. Automated Launching of Workflows

These tools allow to launch workflows automatically:

- Access Points launch a workflow after they detected a signal from 'outside'. There are 5 types of Access Points. The Folder Access Point is the most classic example and is also often referred to as a 'hot folder'. Access Points have their own View in the Pilot.

Learn more in [Access Points](#).

- WebCenter can also launch workflows on Automation Engine (when they are in the same LAN). WebCenter can do this as a specific step in its own workflow, there called 'Submit Workflow'.

Learn more in the [WebCenter documentation](#). Here is a [direct link](#) to this specific topic (in v22.03).

- Some external systems can launch a workflow on Automation Engine via JDF or JMF commands. Although most of this functionality is also available through use of XML, some MIS vendors prefer to use JDF/JMF because of specific additional features.

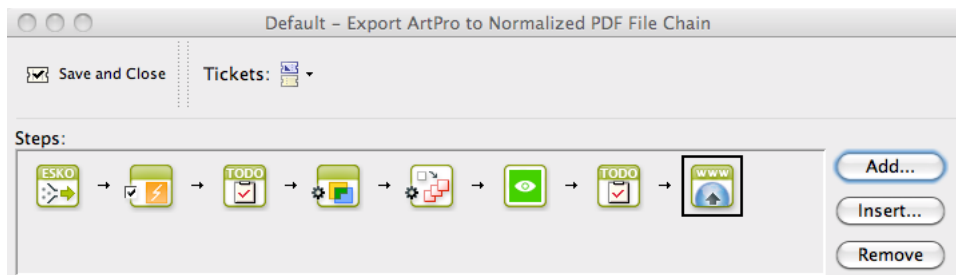
Learn more in [What about JDF?](#).

8. Migrating Old Task Chains

Old Workflow Technology

The Esko product **BackStage**, a predecessor of Automation Engine, had a limited workflow technology where workflows were only linear and were named **Task Chains**.

For example:



If you still have such task chains, they will still work but Esko strongly advises to migrate them to Automation Engine **Workflows**. See below how to do this.



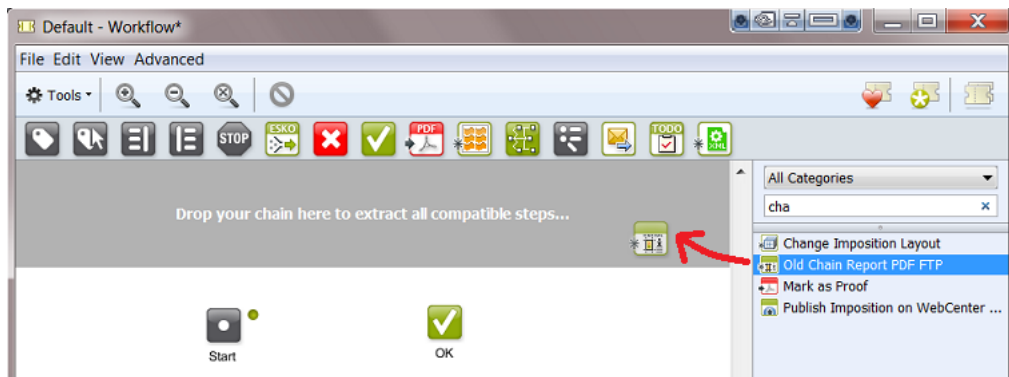
Note: Depending on your version and configuration, it is still possible that your Automation Engine offers some task chains by default. Page printing workflows still offer **Export to PDF File Chain (Mark as Proof)** and **Publish Imposition on WebCenter Chain** as chains of multiple steps. Also, a 'FlexRip plus InkPlanner Chain' ticket is still available (until you can add it to a workflow with an **Imaging Engine** step).

Migrating old Task Chains to new Workflows

You can not automatically convert task chain tickets into workflow tickets, but the workflow editor offers a tool that helps you migrating a task chain to a workflow that will keep all your settings.

Follow these steps to migrate a task chain to a workflow:

1. In the Pilot, click on **New Workflow**.
2. In the workflow editor, in the Tasks pane, select your task chain ticket.
3. Drag and drop your task chain ticket into the top part of the workflow editor window. A grey area will appear, showing you a message where to drop the ticket.



4. This starts a conversion tool that extracts all compatible steps:



- Each extracted step keeps all of its ticket settings.
- The step is named after the name of the task chain.
- The steps are not connected. You are invited to optimize your workflow construction. There may be better ways to build your workflow in the current version.





Note: When the chain contains "outdated" steps (FKA 'obsolete'), a warning appears:

Some chain steps will be excluded.
Only chain steps that are compatible with this workflow system are included.

Excluded chain steps:

#	Task Type
3	Mark File

OK

When you click **OK**, the chain is migrated but the outdated step will disappear after a few seconds. This means that you should replace that step with a corresponding step that is available in your current version. For example you should replace the old **Mark File**  by **Mark** .

5. Based on these extracted steps, build your new workflow.

In below example we removed the old **Export to PDF** step because this functionality is now available on board the **ReportMaker** ticket:



6. Save your new workflow.
7. Test it and, when ready, delete the old task chain from your list of Tickets.