

© 2009 Adobe Systems Incorporated. All rights reserved.

Adobe<sup>®</sup> LiveCycle<sup>\*</sup> ES2 (9.0) Leveraging Solutions in LiveCycle<sup>\*</sup> ES2 for Microsoft<sup>\*</sup> Windows<sup>\*</sup> Edition 1.0, November 2009

This upgrading document is licensed for use under the terms of the Creative Commons Attribution Non-Commercial 3.0 License. This License allows users to copy, distribute, and transmit the document for noncommercial purposes only so long as (1) proper attribution to Adobe is given as the owner of the document; and (2) any reuse or distribution of the document contains a notice that use of the document is governed by these terms. The best way to provide notice is to include the following link. To view a copy of this license, visit <u>http://creative-commons.org/licenses/by-nc-sa/3.0/</u>.

Adobe, the Adobe logo, Adobe Reader, Acrobat, and LiveCycle are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are the property of their respective owners.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

# Contents

## Leveraging legacy solutions in LiveCycle ES2

About the upgraded environment
Strategies for leveraging legacy solutions
Continued execution
Maintenance of legacy items
Progressive development
Importing resources, processes, and events to applications
Importing from the run-time environment
Importing from the file system
Importing from LiveCycle ES archive files
Maintaining legacy run-time instances
Changing existing run-time instances
Updating legacy solutions in other environments
Upgrading legacy solutions to LiveCycle ES2
Breaking the link to existing run-time instances
Creating endpoints for processes
Replacing legacy custom event types    9
Replacing legacy subprocesses
Configuring security settings
Using LiveCycle 7.x QPACs
Using deprecated operations
Exporting embedded documents from operation properties10
Changing the reliance on null value results from XPath expressions
Upgrading human-centric processes
Replacing form-specific variables
Using legacy render and submit processes12
Configuring Workspace start points
Using the User 2.0 service
Updating references to assets

# Leveraging legacy solutions in LiveCycle ES2

LiveCycle ES2 and Workbench ES2 enable you to continue to use the solutions that you created using LiveCycle ES. After the LiveCycle ES environment has been upgraded to LiveCycle ES2, you can access legacy solution components for maintenance and further development:

- "About the upgraded environment" on page 1
- "Strategies for leveraging legacy solutions" on page 2
- "Importing resources, processes, and events to applications" on page 3
- "Maintaining legacy run-time instances" on page 5
- "Upgrading legacy solutions to LiveCycle ES2" on page 8

For an overview of the changes that have been implemented in Workbench ES2, see What's new in LiveCycle Workbench ES2 in *Application Development Using LiveCycle Workbench ES2*.

## About the upgraded environment

When the LiveCycle ES environment is upgraded to LiveCycle ES2, the legacy processes, resources, and events are unchanged:

Processes and process versions: Remain in the Processes view. Lock, activation, and recording properties remain unchanged.

Resources: Remain in the Resources view. Resource locks remain unchanged.

**Event types:** Remain in the Events view. Locks and activation properties remain unchanged.

The run-time instances of legacy processes, resources, and event types continue to execute in the LiveCycle ES2 environment. References between processes, resources, and events are persisted:

- Processes continue to execute correctly.
- · Forms continue to reference form fragments and other resources correctly.

Additionally, service endpoints remain unchanged. Processes can be invoked using the same endpoints that existed in the LiveCycle ES environment.

In Workbench ES2, interaction with items in the Processes, Resources, and Events view is limited. Many features and commands have moved from these views to the new Applications view. For example, you cannot open resources for editing or viewing from the Resources view.

#### Review the LiveCycle ES2 run-time environment:

- Open the following perspectives and views:
  - LiveCycle Runtime View perspective (includes the Applications, Resources, Services, and Events views)
  - Processes view

#### See also

What's new in LiveCycle Workbench ES2 in Application Development Using LiveCycle Workbench ES2

## Strategies for leveraging legacy solutions

Determine whether you will continue to use legacy solutions as-is, or use them as the starting point for developing LiveCycle ES2 assets. Consider following factors when you make a decision:

- Your satisfaction with current solutions.
- · The resources required to maintain current solutions.
- The resources required to upgrade legacy solutions so that you can use new LiveCycle ES2 features.
- The benefits of using new LiveCycle ES2 features. (See What's new in LiveCycle Workbench ES2 in Application Development Using LiveCycle Workbench ES2.)

## **Continued execution**

After upgrading the environment to LiveCycle ES2, leave legacy items unchanged in the Processes, Resources, and Events views. Processes, resources, and events that were used in the LiveCycle ES environment are fully supported in LiveCycle ES2. Your solutions continue to function as they did in the legacy environment. This strategy is suitable for mature solutions that require no maintenance.

## Maintenance of legacy items

Change legacy processes, resources, and events using Workbench ES2 while preserving their link to legacy runtime instances. After importing into LiveCycle ES2 applications, your processes, resources, and custom events continue to execute as LiveCycle ES solutions. However, when you preserve the link to legacy runtime instances, you cannot take advantage of new LiveCycle ES2 features.

For example, edit a process to fix a bug. The changes affect existing process instances and new instances as the process is invoked. After you implement the changes, you can deploy the application. To propagate the changes to another environment, create a LiveCycle ES archive or update an existing LiveCycle ES archive.

This strategy is suitable for solutions that completely satisfy your business needs, but require occasional maintenance.

## To maintain legacy items:

- 1 Import legacy items to an application. (See "Importing resources, processes, and events to applications" on page 3.)
- 2 Edit the items and deploy them or add to LiveCycle ES archives for deployment to production. (See "Maintaining legacy run-time instances" on page 5.)

## **Progressive development**

Break the link between legacy processes, resources, and events and their legacy runtime instances and create full-fledged LiveCycle ES2 application assets:

- Take advantage of new features.
- Develop assets independent of their legacy runtime counterparts.
- · Legacy processes, resources, and events remain intact.

#### To create LiveCycle ES2 assets from legacy items:

- 1 Import legacy items to an application. (See "Importing resources, processes, and events to applications" on page 3.)
- 2 Break the link to the legacy runtime instance. (See "Upgrading legacy solutions to LiveCycle ES2" on page 8.)

## Importing resources, processes, and events to applications

Import legacy processes, resources, and events to add them to applications in the Applications view. For each legacy item, a corresponding asset is created in a target application.

The following rules apply when importing legacy items to applications:

- You can import only one version of a process to an application version. Create multiple applications (or application versions) to import multiple versions of a process.
- You can import a process, resource, or event to multiple applications to create multiple copies.
- Applications can include either legacy items or assets created using LiveCycle ES2. Do not place both legacy items and LiveCycle ES2 assets in the same application.

Application names and folder names have the following restrictions:

- Names can have a maximum length of 40 characters. •
- The total number of characters in the path of an asset cannot exceed 256.
- Names cannot include control characters (characters that have an ASCII value that is less than 32). •
- Names cannot include any of the following characters: "/", "\", "+", "\$", "?", "\*", ":", "|", "<", ">", ",", and the tab key.

If your legacy resources use a folder structure that conflicts with these restrictions, modify the folder structure before importing. Not adhering to these restrictions can cause resources to be inaccessible.

You can import legacy items from the run-time environment, the file system, and LiveCycle ES archives:

- "Importing from the run-time environment" on page 3
- "Importing from the file system" on page 4
- "Importing from LiveCycle ES archive files" on page 4

Check in imported items to save them on the LiveCycle ES2 server. (See Developing applications in a multi-developer environment in Application Development Using LiveCycle Workbench ES2.)

## Importing from the run-time environment

Import legacy items from the run-time environment to create representations of them in the Applications view. The resultant application assets are directly linked to the instances in the run-time environment.

Typically, you import legacy items from the run-time environment to edit them. For example, to edit a process version that exists in the run-time environment, import it to an application and then open it. (See Maintaining legacy run-time instances.)

After importing, the locations in the run-time environment do not change. Consequently, references between imported items are persisted regardless of which application they belong to. Therefore, legacy applications continue to execute successfully.

Note: Create applications before importing legacy items. (See Adding and removing applications in Application Development Using LiveCycle Workbench ES2.)

#### Import legacy items from the run-time environment:

- 1 Click File > Import.
- Under LiveCycle Runtime Content, select Event, Process, or Resource, depending on the type of item you are importing, and then click 2 Next.
- 3 Select the event types, resources, or process versions to import:
  - Events: Select one or more event types. To select multiple event types, expand the event categories and Ctrl+click.

- **Processes:** Select one or more process versions. To select multiple process versions, expand the process groups and processes and then Ctrl+click.
- **Resources:** Select resources or resource folders. If you select a folder, the entire contents, including subfolders and resources, are imported. Ctrl+click to select multiple folders or resources.
- 4 In the Enter Or Select Import Location box, specify the target location for the selected items, and then click Finish.

## Importing from the file system

Import LiveCycle ES items from the file system when you want to use them in LiveCycle ES2 applications. For example, import processes that you exported to XML files from the LiveCycle ES environment.

You can import individual files or entire folders. You can also duplicate the file structure on the file system in the application.

If the path on the file structure is long, the path in the application can exceed the 256-character limit after importing. Exceeding this limit can cause the resource to be inaccessible.

*Note:* Create applications before importing legacy items. (See *Adding and removing applications in Application Development Using LiveCycle Workbench ES2.*)

## Import from the file system:

- 1 Click File > Import.
- 2 Click General > File System and then click Next.
- **3** Browse for the folder that contains the file to import.

*Note:* If you import a file that is in a subfolder of the folder you select, the subfolder is also created in the application.

4 Select entire folders or individual files to import.

You can select files from multiple folders.

5 Click the Browse button next to the Into Folder box and select the application folder to contain the files.

O To see only the file types that you are interested in, click Filter Types and select those file types.

- 6 To replace assets in the application that have the same name and location as the items that you are importing, select Overwrite Existing Resources Without Warning. If you do not select this option, you decide whether to replace same-named files when they are being imported.
- 7 Duplicate the folder structure of the file you are importing, including all folders to the root of the file system. Select Create Complete Folder Structure, and then click Finish.

## Importing from LiveCycle ES archive files

To import items from LiveCycle ES archive files (that were created using LiveCycle ES), first import the archive files into LiveCycle ES2 using LiveCycle Administration Console. After importing, the items in the LiveCycle ES archive file appear in the Processes, Resources, and Events views as they did in the original LiveCycle ES environment. For example, process versions appear in the same process group in the Processes view.

**Note:** If the root of the path of imported resources is Applications, the imported resources do not appear in the Resources view. For example, an LiveCycle ES archive file contains the resources from the Application/LoanApplication folder. When this LiveCycle ES archive file is imported, the contents do not appear in the Resources view.

Create applications before importing legacy items. (See Adding and removing applications in *Application Development Using LiveCycle Workbench ES2*.)

#### Legacy LCA files that include service configurations

In LiveCycle ES, when processes used services that were configured using the Components view, LiveCycle ES archive files included the service configuration. These service configurations cause problems when they are imported into LiveCycle ES2 environments. When you preview archive contents when importing into LiveCycle ES2, deselect any service configurations for service components.

For example, a LiveCycle ES process uses the LDAP service. The connection properties for the LDAP service were configured using the Components view. When the process is packaged in a LiveCycle ES archive, the archive includes the service configuration. When imported, the service configuration is appears as ServiceConfiguration\_LDAPService in the preview. This item should be deselected before importing into LiveCycle ES2.

LC AI	ODE* LIVECYCLE* ES2					Welcome administrator
		🟠 Home 🛱 Services 📰 Se	ttings	Health	Monitor	(1) About 🚯 Logou
✓	ServiceConfiguration_GetEmailAddress	Complete I	Low	Add	No	Configuration, Endpoints, Security Profiles
	ServiceConfiguration_JdbcService	Complete 1	Medium	Overwrite	Yes	Service Configuration, Endpoints, Security Profiles
✓	ServiceConfiguration_LDAPService	Complete I	Low	Add	No	Service Configuration, Endpoints, Security Profiles
<b>&gt;</b>	ServiceConfiguration_Render WFRE001WithPrefill	Complete I	Low	Add	No	Service Configuration, Endpoints, Security

Note: Do not deselect service configurations for process services.

#### Import legacy items from LiveCycle ES archive files:

- 1 Use LiveCycle Administration Console to import the LiveCycle ES archive files. (See Import and manage LiveCycle 8.x archives in *LiveCycle ES2 Administration Help.*)
- 2 In Workbench ES2, open the Processes view and refresh the view. Refreshing causes any processes that were imported from the LCA file to appear.
- 3 Open the Events view and refresh the view. Refreshing causes any event types that were imported from the LCA file to appear.
- **4** Import the items that the LiveCycle ES archive file contained from their run-time location. (See Importing from the run-time environment.)

Note: Refresh the Applications view to see the imported items.

## Maintaining legacy run-time instances

After you import from the run-time environment, perform the following tasks to continue the development of your legacy processes, custom event types, and resources:

- Modify the imported items. (See Changing existing run-time instances.)
- Propagate changes to other environments. (See Updating legacy solutions in other environments.)

## **Changing existing run-time instances**

By default, items that are imported to applications from the run-time environment are linked to the existing run-time instances. Consequently, changes to the imported items affect the run-time behavior of legacy solutions.

For example, a LiveCycle ES process is imported to an application. The form it uses is also imported. The form is checked out and opened using Designer ES2. After changes are made to the form, it is checked in and the application is deployed. The changes affect all new form instances that are created when users open their tasks to participate in the process.

#### Change legacy processes, events, and resources:

- 1 Open the process, event type, or other asset from the Applications view, edit and save the changes. (See Editing and viewing assets in *Application Development Using LiveCycle Workbench ES2*.)
- 2 Check in the process, event type, or other asset. (See Editing and viewing assets in *Application Development Using LiveCycle Workbench ES2*.)
- 3 Deploy the application. (See Checking in applications or assets in Application Development Using LiveCycle Workbench ES2.)

## Updating legacy solutions in other environments

After changing legacy items in the development environment, propagate the changes to other environments, such as staging or production environments. Create or update LiveCycle ES 8.x archives to update legacy applications that were deployed using LiveCycle ES.

The Processes view and the Resources view enable you to create and update LiveCycle ES archive files. After you create or update the LiveCycle ES archive file, use LiveCycle Administration Console to import it into the other environment. (See Import and manage LiveCycle 8.x archives in *LiveCycle ES2 Administration Help.*) When importing, matching processes, resources, and event types that exist on the target environment are updated with the copies in the archive.

#### Note: LiveCycle ES2 archives are based on LiveCycle ES2 applications and are not compatible with LiveCycle ES 8.x archives.

The following user permissions are required for creating or updating archive files:

- To include processes in the archive, the user must have Service Read and Service Invoke permissions.
- To include resources in the archive, the user must have Repository Read and Repository Traverse permissions for the resource being exported. The user must have Repository Traverse permission for the parent folders.

Note: When locked resources are added to a LiveCycle ES archive, lock information is not included in the archive.

## Selecting contents

The following rules apply when selecting processes and resources:

- To make a selection, click the box beside a component that you want to select. A check mark appears in the box. To deselect a component, click the box again.
- A shaded box marks the components selected implicitly.
- Selecting a process category automatically selects each process type within the category.
- Selecting a process automatically selects the latest version of the process. You can manually select earlier versions of the process.
- Selecting a resource folder automatically selects all the files and folders within it.
- Processes and resources that are selected automatically cannot be deselected individually.
- To select several items, click the name of the first item, and then press Shift and click the name of the last item to select. You can also press Ctrl and click individual names. Then click the box beside one of the marked process names.

#### Create an archive file:

- 1 In the Processes or Resources view, click the Create A LiveCycle Archive From Selected Components button.
- 2 Select Create An Archive File and click Next.

#### Maintaining legacy run-time instances

- **3** (Optional) In the Archive Name box, type the name for the archive. The box is prepopulated with the words "New Archive" followed by a date and time string to ensure that it is unique. This name is displayed in LiveCycle Administration Console when you import the archive.
- **4** (Optional) In the Archive Description box, type a description for the archive so that others know what its contents are. A default description specifies the date and time of the archive creation and the name of the user who created the archive.
- **5** In the Local File System Destination box, specify the file name and location of the archive:
  - If you specify only the file name, the archive is saved in the *[install directory]*/Adobe/Adobe LiveCycle Workbench ES2/ workbench directory. *[install directory]* is the location where you installed Workbench ES2.
  - To save the archive in a specific location on the local or network system, either type the fully qualified path and the name. You can also click the ellipsis button \_\_\_\_\_\_ to browse to the location and specify the name of the archive.

Note: You do not need to specify the file name extension. It is added automatically to the file name.

6 Click Next.

If you clicked the Create A LiveCycle Archive From Selected Components button on the Resources view, continue with step 10.

7 (Processes only) Select processes to include in the archive file.

*Note:* Only the processes that are activated are displayed in the list.

8 (Processes only) Click Next to preview the list of processes to include in the archive file or click Finish to save the archive file immediately.

If you did not select any processes, clicking Next opens the Resource Selection panel.

**9** (Processes only) In the Archive Preview panel, review the list of processes that are included in the archive file.

The list indicates the process version and whether you explicitly selected the process or it is implicitly included. Processes are implicitly included when a selected process depends on them. A check mark in the Implicit column indicates that the process was implicitly included.

- **10** Click Next to add resources to the archive file or click Finish to save the archive.
- **11** (Optional) Select resources to include in the archive file.

Resources required by the processes that are selected in the Processes Selection panel are automatically included in the archive and cannot be deselected.

- 12 Click Next to preview the list of the selected components to include in the archive file or click Finish to save the archive file immediately.
- **13** After reviewing the contents of the archive file, click Finish to save the file.

## Update an archive file:

- 1 In the Processes or Resources view, click the Create A LiveCycle Archive From Selected Components button.
- 2 Select Update Existing Archive File and click Next.
- 3 In the Existing Archive box, type the name and location of the LiveCycle ES archive file to update. Click the ellipsis button 🔜 to browse to the location of the file.
- **4** Click Next to modify archive properties.
- 5 (Optional) The Archive Name box displays the current name of the archive. Change it as required (for example, to indicate the date the archive was updated).

The archive name must be unique on the system where the archive is imported. This name is displayed in LiveCycle Administration Console when you import the archive.

- 6 (Optional) In the Archive Description box, type a description for the archive so that other users know what its contents are. The default description includes information about the archive creation, previous updates, and the current update.
- 7 (Optional) In the Local File System Destination box, change the name of the archive file to create an archive based on the one selected. To update the content of the archive, do not change the file name.

8 Click Next to add or remove processes from the archive file or click Finish to save the archive file. If you are using the same file name, a message prompts you to confirm whether you want to overwrite the existing file. Click OK.

When you click Finish, LiveCycle ES2 automatically updates the versions for processes and resources that were explicitly selected when the archive was originally created.

- 9 (Optional) Select processes to add to the archive file or deselect processes to remove from the archive file.
- **10** Click Next to preview the list of processes to include in the archive file or click Finish to save the archive file immediately. If you did not select any processes, clicking Next displays the Resource Selection panel.
- 11 In the Archive Preview panel, review the list of processes that are included in the archive file. Click Next to add or remove resources or click Finish to save the archive.
- 12 (Optional) Select resources to add to the archive file or deselect resources to remove from the archive file.
- 13 Click Next to preview the list of the selected components to include in the archive file or select Finish to save the archive file.
- 14 After reviewing the contents of the archive file, click Finish to save the file.

## Upgrading legacy solutions to LiveCycle ES2

Use legacy processes, resources, and events as starting points for developing LiveCycle ES2 solutions that take advantage of new features. After importing legacy processes, resources, and custom events into applications, make them independent from their legacy runtime instances. Then, modify the assets as required.

Make sure that you have imported all processes, resources, and custom events that are referenced. For example, if you are upgrading a process, import the process, forms, and other resources that it uses, custom events, custom render and submit processes, and subprocesses. (See "Importing resources, processes, and events to applications" on page 3)

To upgrade legacy solutions to LiveCycle ES2, follow the procedures in the following topics:

- "Breaking the link to existing run-time instances" on page 8
- "Creating endpoints for processes" on page 9
- "Replacing legacy custom event types" on page 9
- "Replacing legacy subprocesses" on page 10
- "Configuring security settings" on page 10
- "Using LiveCycle 7.x QPACs" on page 10
- "Using deprecated operations" on page 10
- "Exporting embedded documents from operation properties" on page 10
- "Upgrading human-centric processes" on page 11

#### See also

"Updating references to assets" on page 16

## Breaking the link to existing run-time instances

Break links between imported assets and their legacy run-time counterparts to change assets without affecting existing run-time instances.

For example, you want to use a legacy form in an application in the LiveCycle ES2 environment. The new application requires you to change the form. However, the changes cause errors with a legacy process that still uses the form. So, you add the form to the application, and then break the link between the form and the run-time counterpart. Now the changes do not affect the legacy process.

All items that are imported from the run-time environment have a value for their Deployment ID property. This property identifies the location of the item in the run-time environment. The value also persists the link between the imported asset and the run-time counterpart. To break the link, remove the value for this property.

You can still change the run-time instance of a legacy asset after you break the link. Import the process version, resource, or event type again from the run-time environment.

#### Remove the value of Deployment ID:

- 1 In the Applications view, right-click the asset and click Properties.
- 2 Next to the Deployment ID box, click the Reset button and then click OK.

Note: If you no longer want legacy processes to be available for invocation, remove the endpoints of the process.

## **Creating endpoints for processes**

Endpoints that were created in the LiveCycle ES environment are no longer associated with imported processes that have a reset Deployment ID property. To invoke processes using the same type of endpoints, create them.

In LiveCycle ES2, you create endpoints by adding start points to process diagrams. When the application is deployed, the start points cause the automatic creation of endpoints. You can add the following types of start points:

Workspace: Users invoke processes from Workspace ES2.

Email: Users send an email to the LiveCycle ES2 server to invoke the process.

Watched Folder: Users copy a file or folder to a watched folder to invoke the process.

Programmatic: Java and web service applications, REST, and applications built with Flex, can invoke processes.

Before you add start points, use LiveCycle Administration Console to review the end points that were used in the LiveCycle ES environment. Note the end point properties so that you can duplicate them in the start point properties. (See Managing Endpoints in *LiveCycle ES2 Administration Help.*)

**Note:** Workspace start points generate Task Manager end points. Before you can configure Workspace start points, configure assets and xml variables. (See "Upgrading human-centric processes" on page 11.)

#### To add a start point:

- 1 Drag the start point icon ( ) onto the process diagram.
- 2 Select the type of start point to add and click OK.
- 3 Click the start point and provide values for properties in the Process Properties view.

#### See also

Starting processes using start points in Application Development Using LiveCycle Workbench ES2

## **Replacing legacy custom event types**

If your process uses legacy custom event types, replace them in your process. Use the custom event types that you have imported into an application and reset the Deployment ID.

Before you replace a legacy custom event type, note the name and property values.

## Replace legacy custom event types:

1 Delete the legacy event start point, receive, throw, or catch from the process.

- **2** Drag the Event Picker icon **()** to the process diagram.
- 3 In the Find box of the Define Event dialog box, type the name of the event type.
- 4 Select the imported event type (located in an application) and click OK.
- 5 Click the button that corresponds with what you want to do with the event type.
- 6 Configure the event type using the same property values that were used for the legacy event type. (See Controlling process flow using events in *Application Development Using LiveCycle Workbench ES2.*)

## **Replacing legacy subprocesses**

If your process invokes legacy processes (subprocesses), import the legacy subprocesses into an application and reset the Deployment ID. The application that contains the subprocess to invoke must be deployed. When deployed, the LiveCycle ES2 service for the process is created.

Before you replace a legacy subprocess, note the name and property values.

## Replace legacy subprocesses:

- 1 Delete the icon for the subprocess from the process diagram.
- **2** Drag the Subprocess icon 🔂 to the process diagram.
- 3 In the Find box of the Define Subprocess Activity dialog box, type the name of the process to invoke.
- 4 Select the process that exists in the application and click OK.
- 5 Select the subprocess icon and configure the properties in the Process Properties view.

## **Configuring security settings**

Security settings of legacy processes are not persisted when the Deployment ID property is reset. Configure security settings manually using LiveCycle Administration Console. (See Modifying security settings for a service in *LiveCycle ES2 Administration Help.*)

## Using LiveCycle 7.x QPACs

Processes that use LiveCycle 7.x QPACs continue to function in LiveCycle ES2. Consider upgrading your processes using the Process Upgrade Tool to take advantage of new features and continued support. (See Upgrading processes in *Application Development Using LiveCycle Workbench ES2*.)

## Using deprecated operations

Some operations have been deprecated since the previous release. Typically, deprecated members are replaced with new ones. Replace deprecated operations in your legacy processes to take advantage of improvements they provide. Also, replacement operations are supported longer than deprecated ones.

In lists of services and operations, such as on the Services view, the names of deprecated services and operations have the suffix "(deprecated)".

## **Exporting embedded documents from operation properties**

In LiveCycle ES, some service operations allowed you to select documents from the file system to use as property values. After selecting the document, it became embedded in the process properties. For example, you could select DDX documents from the file system to configure the properties of invokeDDX and invokeOneDocument operations of the Assembler service.

LiveCycle ES2 now enables you to select documents from applications so that you can more easily update them. Consequently, after importing a process that includes embedded documents, you cannot use the Process Properties view to view or edit the documents. To view or edit embedded documents, export them to an application. After exporting, open the asset to view or edit it.

You can export embedded documents only after you import the process to an application. The process must also be checked out when you export. When you export the document, the operation property is automatically configured to use the new asset as a value.

#### **Export embedded documents:**

- 1 On the process diagram, select the operation that uses an embedded document as a property value.
- 2 In the Process Properties view, locate the property that uses the embedded document as a value.
- Click the Export button .The Export button appears only when an embedded document value exists.
- 4 In the File Name box, type a name for the file. Use the correct file name extension.
- **5** Select the application version and folder in which to save the file, and then click OK.

## Changing the reliance on null value results from XPath expressions

Review any logic in your processes that rely on the following situation:

- Null values are returned from XPath expressions.
- The XPath expressions retrieve values from data types that are defined by service components or custom Java classes in custom components (service container files).

In LiveCycle ES2, this situation causes XPath expressions to return an empty-value instance of the data type instead of null. If your business logic relies on null values from XPath expressions, make sure the XPath expressions still return null. Otherwise, change the implementation of the business logic.

Note: XPath expressions that return values from xml variables continue to support null values.

## Upgrading human-centric processes

Human-centric processes that are imported into applications and have reset Deployment ID properties require several changes due to the following changes in LiveCycle ES2:

- Forms and form data are represented differently.
- Render and submit services are configured differently.
- Task Manager end points are created using Workspace start points.
- The User service is deprecated, and replaced with the User 2.0 service.

*Note:* Although the User service is supported, product changes have complicated its use in LiveCycle ES2 solutions. Adobe strongly recommends that you use the User 2.0 service.

Processes that use the User service or Task Manager end point require the following changes:

- "Replacing form-specific variables" on page 12
- "Configuring Workspace start points" on page 13
- "Using the User 2.0 service" on page 14

You must have already imported your processes (including custom render and submit processes and subprocesses), forms, and other resources into one or more applications. (See "Importing resources, processes, and events to applications" on page 3.)

## **Replacing form-specific variables**

In LiveCycle ES2, xfaForm, Document Form, and Form variables are not used to represent forms and form data. Instead, form assets are referenced directly and data is saved in xml variables. When forms submit PDF documents, the document is saved in document variables.

**Note:** Render and submit services are no longer configured in form variable properties. Instead, they are associated with the action profiles of form asset properties. Action profiles control which render and submit services are used, and other properties that involve presenting assets and data to users.

Before you replace an xfaForm, Document Form, or Form variable, note its property values. Perform the following procedure to create an xml variable for each xfaForm, Document Form, and Form variable in your process. You must have already imported the forms that the variables represented into applications.

#### Create an xml variable:

- In the Variables view, click Create New Variable 
   I .
- 2 In the Name box, type a name for the variable, following these naming rules:
  - Must be a valid XML element name that contains only valid XML characters.
  - Must not start with a digit.
  - Must be less than 100 characters long.
  - Must be unique and therefore cannot be id, create\_time, update\_time, creator\_id, or status, which are reserved variables always in the process data model.
- **3** In the Type list, select xml.
- 4 If the variable is used to store process data, select Process Variable in the Purpose area.
  - If the variable stores input data that is provided when the process is initiated, select Input.
  - If the variable stores data that is returned to the process initiator when the process is complete, select Output.
  - If the variable stores input data that is mandatory to initiate the process, select Required.
- 5 To import a schema to make the xml data structure appear in XPath Builder, click Import Asset.
- 6 Select an XSD file or an XDP file that has an embedded schema from an application and click OK.
- 7 Click OK.

## Using legacy render and submit processes

Action profiles provide more flexibility for prepopulating, rendering, and submitting presentation assets such as forms. The features of action profiles can negate the need for custom render and submit services. However, you can still use your custom render and submit processes from LiveCycle ES in your LiveCycle ES2 solutions.

# **Note:** To determine whether your custom render and submit processes are still needed in LiveCycle ES2, see Designing data capture and presentation in Application Development Using LiveCycle Workbench ES2.

In LiveCycle ES2, render and submit processes must include an input variable of type Task Context. To invoke your legacy render and submit processes directly, redesign them so that they use a TaskContext value as input. To avoid redesigning, create a process that invokes your legacy render or submit process:

- · Legacy render and submit processes require no changes.
- Input values for the legacy process are configured on the service's invoke operation.
- The TaskContext value that is automatically passed to the new process contains useful data for configuring the invoke operation.

Perform the following procedures for each form asset to use custom render and submit processes. You must have already imported the custom render and submit processes into an application. (See "Importing resources, processes, and events to applications" on page 3.)

#### **Reset Deployment ID properties:**

- 1 Reset the Deployment ID property of the legacy render and submit processes that you have imported. Resetting enables you to invoke them as subprocesses. (See "Breaking the link to existing run-time instances" on page 8.)
- 2 Right-click the application that contains the render and submit process and click Deploy.

#### Create new render and submit processes:

- 1 In the Applications view, right-click the asset and click Manage Action Profiles.
- 2 Select the default action profile, or create an action profile to preserve the default one.
- 3 To create a render process, under Render Process click the Create A Render Process button 🖬.
- **4** Type a name for the process and click OK.
- 5 To create a submit process, under Submit Process click the Create A Submit Process button 🗟.
- **6** Type a name for the process and click OK.
- 7 Click OK to close the Manage Action Profiles dialog box.

#### Configure render and submit processes:

- 1 In the Applications view, right-click the new render or submit process and select Open.
- 2 Define the operation that was automatically added to the process:
  - For render processes, right-click the Render operation and click Define Operation.
  - For submit processes, right-click the Submit operation and click Define Operation.
- 3 Select the invoke operation of the legacy render or submit process and click OK.
- 4 On the Confirm dialog box, click Yes.
- 5 In the Process Properties view, configure the Input and Output properties of the invoke operation.

#### **Configuring Workspace start points**

Configure the Workspace start point that you added using the procedure in "Creating endpoints for processes" on page 9.

Most of the values for the Workspace start point are the same that were used for the LiveCycle ES Task Manager endpoint. The following table shows how to use the values from the LiveCycle ES Task Manger endpoint properties to configure the Workspace start point.

LiveCycle ES Task Manager endpoint property	Corresponding Workspace start point property	
Name	General/Name	
Description	General/Description	
User Can Forward Task	Options/User Can Forward Task	
Show Attachment Window	No equivalent property. If Permit Adding Attachments is selected, the attachment window appears.	
Allow Attachment Adding	Attachment Options/Permit Adding Attachments	
Task Initially Locked	Options/Task Initially Locked	
Add ACLs for Shared Queues	Options/Add ACLs For Shared Queues	
Task Instructions	General/Task Instructions	

LiveCycle ES Task Manager endpoint property	Corresponding Workspace start point property
Process Owner	General/Contact Info
Categorization	General/Category
Operation Name	Not needed. The current process is invoked.

Other Workspace start point properties configure the form and variable to use to store form data:

**Asset:** The presentation asset to display to the user. Click the ellipsis button and select a form or Guide file. You can select a file from any application.

Action Profile: The action profile to use with the asset. The action profiles that appear are already created for the asset that you selected.

Variable: The variable in which to store the submitted data. Select either an xml, document, or Task Result variable:

- Select an xml variable if the asset submits field data that is en XML or XDP format.
- Select a document variable if the asset is a PDF form that submits a PDF document.
- Select a Task Result variable to save the field data as well as other information about the task.

**Reader Submit:** Properties for enabling the submission of XDP or PDF forms that do not include a submit button. LiveCycle ES did not support this situation so you do not need to configure this property.

Before you configure the start point, configure your form assets and xml variables. (See "Replacing form-specific variables" on page 12.)

#### See also

Workspace start point properties in Application Development Using LiveCycle Workbench ES2

#### Configure the Workspace start point:

- 1 In the process diagram, select the Workspace start point.
- 2 In the Process Properties view, configure the properties using equivalent values from the LiveCycle ES Task Manager endpoint according to the previous table.
- **3** Expand the Presentation & Data property group and provide the following values:
  - Asset: Select the form or Flex application to present to users.
  - Action Profile: Select the action profile of the asset that you want to use.
  - Start Point Output: Select the xml variable for saving the submitted data.

#### Using the User 2.0 service

Replace each Assign Task operation (User service) on your process diagram with an Assign Task operation from the User 2.0 service.

Most of the values for the new Assign Task operation are the same that were used for the deprecated Assign Task operation. The following table shows how to use the values from the deprecated operation to configure new Assign Task operations.

Deprecated Assign Task property	Corresponding Assign Task operation (User 2.0 service) property
General properties	General properties
Route Evaluation properties	Route evaluation properties
Initial User Selection properties	Initial User Selection properties
Task Instructions	Task Instructions

Deprecated Assign Task property	Corresponding Assign Task operation (User 2.0 service) property
Form Data Mappings	No corresponding properties. Forms and form data are represented differently in LiveCycle ES2.
Task Access Control List (ACL) properties	Task Access Control List (ACL) properties
Delegation and Consultation properties	Reassignment Restrictions properties
Attachments and Notes/Show Attachment Window For This Task	Attachments/Show Attachment Window For This Task
Attachments and Notes/Do Not Initialize This Action With Any Notes Or Attach- ments	Attachments/Input List (Do not specify an input list)
Attachments and Notes/Copy All Notes and Attachments From Previous Task	Attachments/Input List (Use the list variable that saved output attachments from the previous task)
Attachments and Notes/Copy All Notes and Attachments From A List Of Documents	Attachments/Input List
Attachments and Notes/Output Attachments	Attachments//Output Attachments
Routes And Priority/Initialize Task With Route Names	User Actions/Specify Custom Names For Actions. Add a user action for each route name.
Routes And Priority/User Must Select A Route To Complete The Task	No corresponding property. In LiveCycle ES2, users must always select a user action to complete the task.
Routes And Priority/Select Priority For This Task	Priority/Task Priority
Reminders properties	Reminders properties
Escalation properties	Escalation properties
Deadline properties	Deadline properties
Custom Email Template properties	Custom Email Template properties

Before you replace deprecated Assign Task operations, you must have already configured your form assets and xml variables. (See "Replacing form-specific variables" on page 12.) Use the following procedures to replace a deprecated Assign Task operation.

#### Add an Assign Task operation:

- **1** Drag the Assign Task operation **a** to the process diagram.
- 2 In the Process Properties view, configure the properties using equivalent values from the deprecated Assign Task operation that you are replacing according to the previous table.
- 3 Expand the Presentation & Data property group
- 4 Select Use An Application Asset, click The ellipsis button ... next to the Asset box, and select the form asset.
- 5 Make sure the action profile that you want to use is selected.
- 6 (Optional) To prepopulate the asset, in the Variable list select an existing xml variable that contains the data. For example, select the variable that was used as the Start Point Output property value. To create an xml variable to use as the property value, click the Create New Variable button ♀.
- 7 Expand the Output property group.
- 8 (Optional) To save submitted task data, in the Task Result list select a Task Result variable. To create a variable to use as the property value, click the New Variable button 🕂 and create a Task Result variable.

Task Result variables are new in LiveCycle ES2. They store all data that is submitted with tasks, such as asset data and information about the task.

#### Updating references to assets

9 (Optional) To save submitted asset data, in the Output Data list select an xml variable to save the data. Saving the asset data separately is useful for using the data to populate the asset of a subsequent task.

#### Draw routes for the new Assign Task operation:

- 1 If the routes that lead to or away from the deprecated Assign Task operation include conditions, note the expressions of the conditions.
- 2 Delete the routes that lead to and away from the deprecated Assign Task operation.
- 3 Draw routes to and from the new Assign Task operation that you added in the previous procedure.
- 4 (Optional) If the deprecated Assign Task operation used route names as task submission options, add a user action for each route. For example, if the route names were Approve and Deny, add user actions named Approve and Deny:
  - Expand the User Actions property group.
  - Click the Add A User Action button 4.
  - In the Action Name box, type the name of the action as you want it to appear to the user.
  - From the Destination list, select next operation to execute when the user action is clicked, and then click OK.
- 5 (Optional) If the routes to or from the deprecated Assign Task operation included conditions, add the conditions to the routes of the new Assign Task operation:
  - Select the route and in the Process Properties view, expand the Conditions property group.
  - Click the Add Route Condition 🛃 button.
  - In the Expression box on the left, type the first part of the expression. If the condition is complex, click the ellipsis button ... to display the XPath Builder.
  - In the Operation list, select an operation.
  - In the Expression box on the right, type the second part of the expression. If the condition is complex, click the ellipsis button to display the XPath Builder.
  - Click OK to close the Route Properties dialog box.
  - If you have more than one routing condition in the Conditions category, select the join condition to determine how the conditions are evaluated:
    - Use AND Join For Conditions means that the route is valid only if all the conditions evaluate to True.
    - Use OR Join For Conditions means that the route is valid when one or more of the conditions evaluate to True.

The default join condition is Use OR Join For Conditions.

#### Delete the deprecated Assign Task operation:

Right-click the deprecated Assign Task operation and click Delete Operation.

#### See also

Creating Tasks in Application Development Using LiveCycle Workbench ES2

## **Updating references to assets**

After importing legacy processes, resources, and event types into applications, the URL of the item is changed. Update the references to the items in your forms (if the relative location has changed) and custom client applications.

#### **Process services**

Change custom applications that you created with the LiveCycle ES SDK if they invoke services that are created for deployed processes. In LiveCycle ES2, the name of these services in your Java<sup>™</sup> code must be in the following format:

#### Updating references to assets

[Application Name]/[Service Name]

- [Application Name] is the name of the application that the process belongs to.
- [Service Name] is the name of the process service.

#### Resources

Change references to resources to use URLs in the following format:

repository:///Applications/[Application Name]/[version]/[legacy path]

- [Application Name] is the name of the application that the resource belongs to.
- [version] is the version of the application.
- [legacy path] is the path of the resource as it appeared in the Resources view.

For example, in the Resources view, the path of a form is Loan/Forms/request.xdp. The form is imported into version 1.0 of an application named *NewCustomers*. The new URL is repository:///Applications/NewCustomers/1.0/Loan/Forms/request.xdp.