Document Distributor

Version 5.4

SDK Guide

Contents

Getting started

Lexmark^{TM} Document Distributor (LDD) is a system for capturing paper documents and converting them to digital format, or capturing existing digital documents and then processing and routing these documents according to your business processes. Users can submit paper documents from multifunction printers, or they can submit digital documents from individual computers. For more information about Lexmark Document Distributor, see the *Lexmark Document Distributor Administrator's Guide*, available at **http://support.lexmark.com**.

This guide covers the LDD *software development kit* (SDK), which is used to build the custom solutions that model business processes in the Lexmark Intelligent Capture system and LDD system, as well as their supporting components.

Prerequisite skills for developing workflow solutions

The following skills are essential for developing LDD workflow solutions with the LDD SDK:

- Understanding of LDD 5.x. For more information, see the *Lexmark Document Distributor Administrator's Guide*.
- Basic knowledge of usage and network configuration of MFPs.
- Object-oriented programming basics:
  – Variables
  – Arrays
  – Conditional statements
  – Loops
  – Logic operators
  – Functions
  – Objects
- JavaScript 1.2.
- Eclipse development platform.

Prerequisite skills for developing LDD system components

The following skills are essential for developing LDD system components with the LDD SDK:

- Code development within Java Platform, Standard Edition
- XML
- Management of Apache Software Foundation Apache Tomcat Servlet containers software
- Web services with Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP)
- Spring Framework
- Development for the Lexmark Embedded Solutions Framework (eSF)

  Note: For more information on developing for the Lexmark eSF, contact Lexmark Professional Services.

Understanding workflow solutions

In LDD, a workflow solution determines the specifics of each stage of a job to meet a business need.

The stages of a job are the following:

1 Document capture

2 Document processing

3 Document routing

4 Confirmation

For more information on each stage, see the *Lexmark Document Distributor Administrator's Guide*.

A workflow solution provides a new profile or job type that a user can initiate on a printer or software client to which it is deployed. Some solutions can be scheduled to run at particular times.
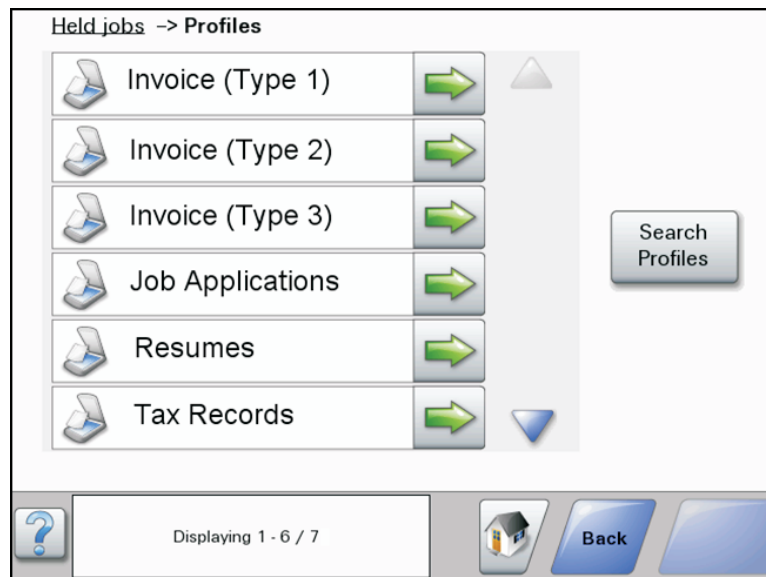
A workflow solution contains one or more of the following elements:

- **Script**—A list of actions that are executed when a solution is initiated. Scripts can be static, or may require user input for processing and routing documents. For example, a script that processes bank loans may require several user inputs, such as a branch name, account number, or Social Security number.
- **Policy**—The settings for the solution and the printers to which it is deployed.
  - **Device settings**—The required printer configuration to support the jobs. This configuration usually includes profiles but can include almost any device setting.
  - **Solution settings**—The settings that let you adjust jobs or printer configurations. Some settings are applicable to the solution across all printers, while others vary from one group of printers to another.
- **eSF application**—An application that is installed on printers to provide functionality needed by the solution, such as delayed sending of scanned documents.
- **Components**—JAR files that provide services not available in the base LDD installation, such as the interaction with a custom Enterprise Content Management (ECM) system.
- **Formsets**—The custom form files that are used for merging data with standard forms.
- **Custom reports**—The reports containing custom data that a solution developer designs.

Workflow solution scripts are written using JavaScript. For more information on JavaScript programming, see a JavaScript reference.

Understanding solution types

The types of solutions that can be developed are the following:

- A device solution lets users submit documents and respond to prompts on a printer.
- A client software solution lets users or other applications submit documents electronically.
- A hybrid solution uses a supporting eSF application on printers to deliver advanced functionality, such as cached or scheduled job submissions, to the LDD system.

You can also include functionality from more than one of these types in a single solution by including a separate script for each type.

Solution type features

Feature	Device solution	Client software solution	Hybrid solution
Workflow initiation	Printer	Client computer or external application, such as part of an ECM system	Printer
Document submission	Scanner glass or automatic document feeder (ADF)	Client software on a computer **Note:** Each solution determines the acceptable file types.	Scanner glass or automatic document feeder
Workflow control	LDD system	LDD system	Printer
Main processing load	LDD system	LDD system	Shared **Note:** The printer manages prompts and user interaction, and the LDD system manages other document processing and routing.
User prompt support	Yes	Lexmark Document Server Printer Port	Yes
Cached or scheduled job submission support	No	Conditional **Note:** An external application must manage caching and scheduling.	Yes
eSF application integration support	Yes	No	Required
Custom home screen support	Yes	No	Yes
Job confirmation customization	Contained in solution script	Contained in solution script	Contained in eSF application

Feature	Device solution	Client software solution	Hybrid solution
Example included as LDD SDK template	TestMFP	TestSNS	TestHybrid **Note:** The source code included in the eSF application file can be compiled only with the eSF SDK.

Version compatibility of workflow solutions

Solutions developed for LDD 4.5 and 4.4 are compatible with LDD 5.0, 4.9, 4.8, 4.7, and 4.6.

Solutions developed for LDD 5.0, 4.9, 4.8, 4.7, and 4.6 are not compatible with LDD 4.5 or earlier.

Understanding components

A component is a Java application built on the Spring Framework that is used to provide extended functionality to workflow solutions.

What is the LDD SDK?

The LDD SDK is a group of plug-ins that are included with LDD for the Eclipse IDE software. For more information, see the documentation for Eclipse IDE.

The SDK provides tools for creating solutions for LDD and lets you do the following:
- Write scripts.
- Edit solution settings and device policies.
- Develop components.
- Package finished solutions.

System requirements

Category	Requirement
Operating system	• Windows Server 2012 Standard or Enterprise Edition • Windows Server 2008 Standard or Enterprise Edition • Windows 8.x • Windows 7.x
Processor	2GHz
RAM	1GB
Hard disk drive	400MB, including space for the following: • Java Development Kit (JDK) • Eclipse IDE software • LDD SDK

Category	Requirement
Video	Capable of showing a resolution of 1024 x 768 or higher

Supported printers

e-Task 5 printers[1]	e-Task 4 printers	e-Task 3 printers	e-Task 2+ printers	e-Task 2 printers	e-Task printers[2]	X642 printers
MFPs • CX522 • CX622 • CX625 • CX725 • CX820 • CX825 • CX860 • CX920 • CX921 • CX922 • CX923 • CX924 • MX421 • MX521 • MX522 • MX622 • MX721 • MX722 • MX725 • MX822 • MX824 • MX826 **SFPs** • CS622 • CS720 • CS725 • CS820 • CS921 • CS923 • MS622 • MS822 • MS824 • MS826	**7- or 10-inch-screen MFPs** • CX510 • MX610, MX611 • MX6500e • MX710, MX711 • MX810, MX811, MX812 • MX910, MX911, MX912 **4.3-inch-screen MFPs** • CX410 • MX410, MX510, MX511 **7-inch-screen SFP** • MS812de **4.3-inch-screen SFPs** • CS510 • MS610de • MS810de • MS911	**MFPs** • 6500e • X548 • X746, X748 • X792 • X925 • X950, X952, X954 **SFPs** • C748 • C792 • C925 • C950	**MFPs** • X463, X464, X466 • X651, X652, X654, X656, X658 • X734, X736, X738 • X860, X862, X864 **SFP** • T656[4]	**MFPs** • X644, X646 • X772 • X782 • X850, X852, X854 • X940, X945 **SFPs** • T654[4] • W850[4]	**MFPs** • X620[3] • X632, X634[3] • X752[3] • X820[3] • X830, X832[3] • X912[3] **SFPs** • C520, C522, C524[4] • C530, C532, C534[4] • C734, C736[4] • C780, C782[4] • C920[4] • C935[4] • E450[4] • E460, E462[4] • T640, T642, T644[4] • T650, T652[4] • W840[4]	X642

[1] Only printers with firmware level 2 or later are supported.

[2] These printers do not support eSF applications used in hybrid solutions.

[3] These printers may be identified as "C," "T," or "W" models in LMC.

[4] These SFPs do not support all prompts that MFPs support. For a list of supported prompts, see .

Notes:

- For more information on the latest device and firmware level support, see the *Readme* file.
- Some printer models do not support double-byte characters. For more information, see "Double-byte character support" on page 11.

Prompts supported by SFPs

Some prompts may not be supported in some printer models.

	Touch-screen printers	Non-touch-screen printers
Supported prompts	• ArrayPrompt • AuthenticationPrompt[1] • BooleanPrompt • CustomVlmlPrompt • ImageBooleanPrompt • ImageListPrompt • ImageMessagePrompt • IntegerPrompt • ListPrompt • MessagePrompt • NumericPrompt • PasswordPrompt • StringPrompt • NullPrompt	• ArrayPrompt • AuthenticationPrompt[1, 2] • BooleanPrompt • IntegerPrompt • ListPrompt • MessagePrompt • NumericPrompt • PasswordPrompt[3] • StringPrompt[2] • NullPrompt[4]

[1] Requires a supported card reader.

[2] Supported by T654 and W850 models only.

[3] Supports numeric PINs only in e-Task printers. Supported e-Task printers include C520, C522, C524, C530, C532, C534, C732, C734, C736, C780, C782, C920, C935, E450, E460, E642, T640, T642, T644, T650, T652, W840.

[4] Supported by e-Task printers only.

Double-byte character support

Double-byte characters may not be supported in some printer models.

Printer model	Simplified Chinese	Traditional Chinese	Japanese	Korean
6500	✓	✓	✓	✓
C748	✓	✓	✓	✓
C792	✓	✓	✓	✓
C925	✓	✓	✓	✓
C950	✓	✓	✓	✓

Printer model	Simplified Chinese	Traditional Chinese	Japanese	Korean
CS510	✓	✓	✓	✓
CS720	✓	✓	✓	✓
CS725	✓	✓	✓	✓
CS820	✓	✓	✓	✓
CS921	✓	✓	✓	✓
CS923	✓	✓	✓	✓
CX410	✓	✓	✓	✓
CX510	✓	✓	✓	✓
CX725	✓	✓	✓	✓
CX820	✓	✓	✓	✓
CX825	✓	✓	✓	✓
CX860	✓	✓	✓	✓
CX920	✓	✓	✓	✓
CX921	✓	✓	✓	✓
CX922	✓	✓	✓	✓
CX923	✓	✓	✓	✓
CX924	✓	✓	✓	✓
MS610	✓	✓	✓	✓
MS810, MS812, MS911	✓	✓	✓	✓
MX410, MX510, MX511	✓	✓	✓	✓
MX610, MX611	✓	✓	✓	✓
MX710, MX711	✓	✓	✓	✓
MX810, MX811, MX812	✓	✓	✓	✓
MX910, MX911, MX912	✓	✓	✓	✓
X463, X464, X466	✓	✓	X	✓
X548	✓	✓	✓	✓
X642	✓	X	X	X
X644, X646	✓	X	X	✓

Printer model	Simplified Chinese	Traditional Chinese	Japanese	Korean
X651, X652, X654, X656, X658	✓	✓	X	✓
X734, X736, X738	✓	✓	X	✓
X746, X748	✓	✓	✓	✓
X782	✓	X	X	✓
X792	✓	✓	✓	✓
X850, X852, X854	✓	X	✓	✓
X860, X862, X864	✓	✓	X	✓
X925	✓	✓	✓	✓
X940, X945	✓	X	✓	✓
X950, X952, X954	✓	✓	✓	✓

Supported ECM software platforms

- ImageNow 6.7 and 6.6
- Microsoft SharePoint
  - Microsoft SharePoint Foundation 2010
  - Microsoft SharePoint 2010
  - Microsoft Office SharePoint Server 2007
  - Windows SharePoint Services 3.0
- Autonomy iManage WorkSite with WorkSite Server 8.5 or 8.2

Notes:

- Integration with ECM systems may require specific licenses.
- For ECM platforms without direct integration, the LDD system saves documents and metadata to a directory where an ECM system is configured to poll for files.

Getting support

When requesting technical support, provide the following information:
- Description of the problem
- Steps to reproduce the problem, if possible
- Operating system, version, and service pack level
- JDK and JRE version
- Eclipse software version
- LDD SDK plug-in versions

  Note: To find the SDK plug-in versions, click **Help** > **About Eclipse Platform** > **Plug-in Details**. LDD SDK plug-ins list the provider as "Lexmark International Inc."

- A ZIP file containing the source files in your solution project
- The current .solution file if the problem is with a built solution or installation
- The Eclipse error log

  Note: The Eclipse error log is the file ".log" in the .metadata folder within your workspace folder.

Preparing for development

Installing the development environment

Before you begin, do one of the following:

- For LDD 5.0 or later, install the Java SE 8 Development Kit (JDK 8) and Java Runtime Environment (JRE). These LDD versions are shipped with Java 1.8.0_91.
- For LDD 4.8.03 or later, install the JDK 7 and JRE. These LDD versions are shipped with Java 1.7.0_55-b13.
- For LDD 4.8.02 or earlier, install the JDK 6 and JRE. These LDD versions are shipped with Java 1.6.0_18-b07.

  Note: JDK and JRE are also available at the Oracle website.

The following instructions are for the Eclipse IDE software version 4.3.2 (Kepler). Instructions may vary for other versions of the software. The Eclipse IDE software for Java Developers is available at the Eclipse website.

1 From the Eclipse IDE software, change the workspace to a folder name with no spaces.

   Note: Run the Eclipse IDE software as an administrator.

2 If necessary, close the Welcome screen.

3 If you are using a proxy server, then set up a proxy connection.

   Note: For more information, see the documentation for the Eclipse IDE software.

4 Install the LDD SDK plug-in. For more information, see "Installing the LDD SDK plug-in" on page 16.

   The LDD SDK menu is available after installing the Solution Builder Toolkit.

5 Enable the Lexmark Solution Development perspective. For more information, see "Enabling the Lexmark Solution Development perspective" on page 17.

   Note: A perspective provides predefined layouts and editor sets in the Eclipse IDE software.

6 Install an Eclipse software plug-in for editing JavaScript files, such as JSEclipse.

   To install JSEclipse, type **http://hyperion.mw.lexmark.com/lddsdk/jseclipse/** in the Available Software dialog box.

   Notes:

   - The automatic completion of LDD script elements is available only on JSEclipse. If JavaScript plug-in is not installed, then associate the JavaScript files manually by using the Eclipse software text editor.
   - The JSEclipse plug-in requires a workspace folder name with no spaces.

7 Enable the automatic completion of libraries for JSEclipse. Do the following:

   a In the Eclipse software, click **LDD SDK** > **Update JSEclipse Autocomplete**.

      Note: If JSEclipse is not installed, then the menu item is unavailable.

   b Restart the Eclipse software.

   Note: When updating plug-ins, make sure that the Lexmark Solution Development perspective is enabled. For more information, see "Enabling the Lexmark Solution Development perspective" on page 17.

Installing the LDD SDK plug-in

Offline installation

1 On your computer, extract the LDD SDK plug-in ZIP file.

2 From the Eclipse IDE software, click **Help** > **Install New Software** > **Add** > **Local**.

3 Browse to the folder where you extracted the plug-in, and then click **OK**.

4 Add the local site.

5 Select **SBT**, and then click **Next**.

6 Check the list of items to be installed, and then click **Next**.

7 Accept the license agreement, and then click **Finish**.

8 If necessary, when you receive a security warning indicating that the software contains unsigned content, click **OK**.

9 Restart the Eclipse IDE software.

Online installation

1 From the Eclipse IDE software, click **Help** > **Install New Software** > **Add**.

2 In the Location field, type `http://hyperion.mw.lexmark.com/lddsdk/current/`, and then click **OK**.

3 If necessary, enter your user name and password, and then click **OK**.

4 Select **SBT**, and then click **Next**.

5 Check the list of items to be installed, and then click **Next**.

6 Accept the license agreement, and then click **Finish**.

7 If necessary, when you receive a security warning indicating that the software contains unsigned content, click **OK**.

8 Restart the Eclipse IDE software.

The LDD SDK menu is available after installing the Solution Builder Toolkit.

Notes:

- To update the plug-in automatically, from the Preferences window, configure the Automatic Updates settings.
- When updating plug-ins, make sure that the Lexmark Solution Development perspective is enabled.

Enabling the Lexmark Solution Development perspective

When developing solutions for LDD 5.x, use the Lexmark Solution Development perspective.

Note: A perspective provides predefined layouts and editor sets in the Eclipse IDE software.

1 From the Solution Builder Toolkit, click **Window** > **Open Perspective** > **Other**.

2 Select **Lexmark Solution Development**, and then click **OK**.

Note: When using the Lexmark Solution Development perspective, the LDD Build Log appears in the workbench.

Manually associating JavaScript files with the Eclipse software

If a plug-in for editing JavaScript files (such as JSEclipse) is not installed, then associate JavaScript files manually with the Eclipse software text editor.

1 In the Eclipse software, click **Window** > **Preferences**.

2 In the Preferences dialog, select **General** > **Editors** > **File Associations**.

3 Beside the File types list, click **Add**.

4 Type ***.js**, and then click **OK**.

5 Make sure that the new entry for *.js is selected, and then click **Add** beside the Associated editors list.

6 Make sure that **Internal Editors** is selected, select **Text Editor** from the list, and then click **OK**.

7 In the Preferences dialog, click **OK**.

Accessing the *Lexmark Document Distributor Script Reference Guide*

1 In the Eclipse software SDK, click **LDD SDK** > **Reference Material**.

2 On the page that appears, click **LDD 4.6, 4.7, 4.8, 4.9 Script Reference** or **LDD 5.0 Script Reference** for working with LDD 5.0 scripts. The script reference guides for previous versions may also be available.

3 Select topics from the menu at the left side of the page.

Developing workflow solutions

Best practices for solution design

Follow these guidelines when developing the workflow and user interface of a solution:

- Make it clear to the user when document input is expected. Depending on the nature of the solution, a prompt asking the user to place a certain document on the scanner may be beneficial. Alternatively, if the purpose of the solution makes the expectation of document input obvious, a prompt may be unnecessary.
- When possible, present all prompts before scanning the document. If a user does not know some of the information requested through prompts, then cancel the job before scanning since it takes the most time. On e‑Task 2 or later printers, you may choose to present prompts or confirmation after scanning and partially processing the document when appropriate. On e‑Task printers, however, no prompts can be used after the scan task.
- Handle user-initiated cancels at all stages.
- When possible, allow the user to batch process multiple documents. To avoid forcing the user to wait between scan tasks, scan and store all documents before processing and routing.
- To avoid unexpected errors due to incorrect or missing input, validate documents to scan the correct documents and the correct number of documents or pages.
- When possible, complete all prompting and scanning before beginning processing and routing. Again, on e‑Task 2 or later printers, you may choose to present prompts or confirmation after scanning and partially processing the document when appropriate. Users do not expect a long wait between prompts, so minimize such intermediate processing.
- After processing and routing, provide confirmation of success or failure to the user. On e‑Task devices, confirmation must be printed, and no further action may be taken in case of failure. On e‑Task 2 or later devices, you can use more prompts to deliver confirmation, and allow a user option to take corrective action in case of failure.
- Report the percentage of completion to the system when the task is completed, or as accurately as possible when the task fails.
- Handle all errors and recover whenever possible. On e‑Task 2 or later devices, allow a user option to take corrective action in case of errors. For example, the user may be prompted to rescan a document when a bar code could not be read because it was scanned upside down. On e‑Task devices, errors that occur during or after a scan can be reported only on the confirmation page.
- Log all errors regardless of recovery ability, and report them to the user whenever possible. Log non-fatal errors at the "ERROR" level and fatal errors at the "FATAL" level.
- Log potential problems that are not errors at the "WARN" level.
- To save space in the log and maintain better system performance, log messages that only an administrator can see during normal operation at the "INFO" level. Remember that an administrator can see the overall completion status of a job without a log message using the Jobs task in Lexmark Management Console (LMC).
- To determine the location of a bug, log messages at the "DEBUG" level. Messages at the "DEBUG" level are not recorded unless specifically enabled by an administrator, so these messages do not affect space and performance during normal operation.

Creating a workflow solution project

Each LDD solution is maintained as a project within the Eclipse software.

1 In the Eclipse software, click **File** > **New**, and then click the solution setting of your LDD version.

Note: If the Lexmark Solution Development perspective is not open, then click **File** > **New** > **Project**. From the Lexmark folder, select the solution setting of your LDD version, and then click **Next**.

2 Type a container name for the project.

The container name is the name of the workflow solution project in the Eclipse IDE software.

3 Configure the optional properties. Do one or more of the following:

- To use a solution name different from the container name, clear **Use Project Name**, and then type a solution name.
   The solution name is the name of the solution that appears in LMC when the solution is installed.
- To start a workflow solution project at a version other than 1.0, type a value in the Initial Version field.
- To use a package name different from the default that is based on the solution name, clear **Use Default**, and then type a package name.
   The package name is the name of the solution file built from the workflow solution project.

4 Select the version of the LDD system where you want to install the solution, and then click **Next**.

5 From the Available Templates list, select the appropriate template for the solution you plan to develop, and then click **Next**.

6 Type a description of the solution and the name of the author.

7 Click **Finish**.

Understanding the contents of a workflow solution project

The following folder structure details the contents of a workflow solution project. Some folders may not appear in all solutions.

ProjectFolder—This is the Container folder that contains all contents of the project. The Container folder is named when creating the project.

docs—This contains files for use during development. The contained files are not included with the compiled solution.

platform**—This contains files and components accessed by the solution and shared with other solutions. All contents of this folder are copied to the application folder (\Lexmark\Solutions\apps\wf-ldss\) when the solution is installed on a system. This folder is accessible by scripts through the **taskInfo.platformDir property.

Note: If files are included in this folder, then deployment of the solution may require a restart of the LDD system. Files that do not need to be shared with other solutions should be placed in the src \solution\WEB-INF\ and src\solution\WEB-INF\lib\ folders instead.

src—This contains the source files of the project.

conf—This contains the configuration files for the project.

devicePolicy.ldd—This is the configuration file that manages the profiles and associated policy settings associated with a solution.

forms.ldd—This is the configuration file that manages the formsets included with or linked to the solution.

reports.ldd—This is the configuration file that manages the custom reports included with the solution.

solutionSettings.ldd—This is the configuration file that manages the global and local solution settings available for a solution. Settings created here are accessible by scripts and can be changed using Lexmark Management Console.

statusPage.properties—This is the configuration file that manages the look and feel of the custom status page that may be used instead of Lexmark Management Console for non-technical users.

esf—This contains any .fls files for eSF applications used with a hybrid solution.

esf-app-source-code—This contains the source files from any eSF applications used with a hybrid solution.

forms**—This contains any formsets included with the solution. (Use the **forms.ldd configuration file to add forms.)

images—This contains images associated with the solution, such as home screen images added using the device policy editor. Files in this folder are copied to the images\ folder of the solution on the LDD system when the solution is installed.

reports**—This contains compiled Jasper custom report files associated with the solution. (Use the **reports.ldd configuration file to add reports.)

scripts—This contains the JavaScript scripts included in the solution. This folder may be empty for solutions that deploy only an eSF application or LDD component. Each main script located here should be assigned to a profile in devicePolicy.ldd.

scheduled—This contains scripts that can be scheduled using Lexmark Management Console after a solution is deployed.

solution—This contains files and components accessed only by the current solution. This folder is accessible by scripts through the `taskInfo.solutionDir` property.

WEB-INF—This contains property files for custom components specific to a solution that should not be shared with other solutions.

lib—This contains custom components specific to a solution that should not be shared with other solutions.

target—This contains the compiled solution after a project build, in addition to associated class files and XML descriptor files. This folder appears only in a new project after the first build.

Note: This folder may be deleted to ensure a clean build.

install—This contains the compiled solution, with the extension .solution. The file contained here can be distributed independently as a complete solution.

solution—This contains copies of all files contained in the solution file from the time of the build.

Note: File and folder names within a workflow solution project cannot contain double-byte characters. However, files themselves may contain double-byte characters.

Managing solution projects

The following operations are performed the same way for LDD solution projects and for any project in the Eclipse software:

- Importing existing projects
- Importing archive files
- Creating scripts
- Building projects

For more information, see the Eclipse software documentation.

Configuring a workflow solution project

1 In the Package Explorer window, right-click the solution folder, and then click **Properties**.

2 Select the properties setting of your LDD version.

3 Configure the settings.

4 Click **OK**.

Note: Changing the LDD server target does not change the contents of scripts. Convert scripts between different versions manually.

Referencing a workflow solution project

1 Make sure the project to be referenced has been built. If Build Automatically is not enabled on the Project menu, then do the following:

  a Select the workflow solution project in Package Explorer.

  b Click **Project** > **Build Project**.

  The target\install folder should contain a solution file after a successful build.

2 Determine whether to include or reference the solution project:

- An included solution does not update with changes made to the original component project. Do the following to include a solution:

  a Make sure the parent solution project appears in Package Explorer.

  b Click and drag the solution file from the target\install folder of the solution project to be included to the root folder of the parent solution project.

- A referenced solution updates if the original solution project is changed and rebuilt. Do the following to reference a solution:

  a Right-click the parent solution project folder in the Package Explorer pane, and then click **Properties**.

  b Select **Project References** from the list.

  c Select the check box beside the solution project you want to reference in the workflow solution project.

    Note: A project must appear in Package Explorer to appear in the Project References list.

  d Click **OK**.

Evaluating code quality with JSLint

To check for problems in your script code during builds, you can enable the included version of JSLint. JSLint evaluates code for overall quality, including syntax, structure, and style conventions, which may help avoid runtime errors by finding issues that cause compiler errors.

For more information on the specific checks carried out by JSLint, see the JSLint instructions at **www.jslint.com**.

Note: Some checks specific to the use of JavaScript with HTML are omitted from LDD. A different comment, described in the following, is used to identify functions defined later in or outside of the script and LDD objects as global variables.

When JSLint is enabled, code is parsed during project builds. Any warnings generated are shown in the Problems view.

Note: If the Problems view does not appear in the Eclipse software, then see the documentation for the Eclipse software.

To enable JSLint during the build, do the following:

1 In the Eclipse software, click **Window** > **Preferences**.

2 In the category list, expand **Lexmark**, and then select **JSLint** > **Enable JSLint**.

3 If necessary, configure the checks for JSLint to carry out, and then click **OK**.

Note: Project builds for large solutions may be noticeably slower when JSLint is enabled. You may want to enable JSLint only while debugging.

Avoiding warnings for functions and LDD objects

To identify LDD objects and indicate functions defined later in or outside of the script to JSLint, use the comment **/**jsLint *variableName* */**.

To identify the LDD top-level objects, include this comment in the script. For example,

```
/**jsLint caller,confirm,confirmTarget,context,credentials,printerIP,taskInfo */
```

Understanding types of scripts

Using profile scripts

Most workflow solution projects include at least one profile script, which, after being assigned to a profile using the Device Policy Editor, can be executed from a printer or software client. A profile script may have any file name other than the names reserved for auto-configure scripts, and each profile script file should contain a `main()` function.

All LDD objects are available for use in profile scripts, and profile scripts can access library scripts.

The script file TestMFP.js in the TestMFP example template is an example of a profile script.

Using library scripts

Library scripts, which can be accessed only by other scripts, can be used to contain common functionality, such as functions for database access, logging, and progress monitoring. Each function in a library script should use objects and methods appropriate only for scripts that will call it. For example, a library script called by a scheduled script should not use prompt objects or any scan methods.

The script file library.js in the TestSNS example template is an example of a library script.

Adding a library script to a solution

A library script does not require a specific file name. To begin adding a library script, add a blank script in the scripts\ folder.

To define the script as a library script, a specific structure is used:
- Begin the library section of the script with an empty **library()** function.
- Use the format **library.prototype.<functionName> = function(<arguments>)** for function headers.
- Create a new **library** object. This **must** be done as the last line of the script.

The following example shows the basic structure of a library script:

```
function library()
{
}
…
library.prototype.myFunction = function(myArgument1,myArgument2)
{
   …
}
…
new library();
```

Accessing the functions of a library script

To access the functions of a library script from another script, create an object reference to the library script using the **context.callTask()** method. The functions in the library script are then available as members of the new object.

The first argument in the **callTask()** method is always **taskInfo.solutionName**. The second argument is the script file name with no extension. For example, if the previous library script is named "mylibrary.js," the following line creates an object reference to that library:

```
var myLib = context.callTask(taskInfo.solutionName, "mylibrary");
```

The included functions can then be accessed as follows:

```
myLib.myFunction(arg1,arg2)
```

Using auto-configure scripts

If you have detailed knowledge of the LDD system where a solution is deployed, then you can write an auto-configure script that can:
- Create a device group.
- Add printers to a device group.
- Deploy the solution to a device group.

- Configure home screens.
- Discover devices.
- Perform a policy update.
- Set the server online.

Several events can trigger auto-configure scripts. The following script names determine the events that trigger the scripts:

- **configureNewSolution.js**—The script is triggered when a solution is installed on the LDD system.
- **configureNewSolutionServer.js**—The script is triggered when a solution is added to each LDD server in the LDD system.
- **configureUpgradeSolution.js**—The script is triggered after an existing solution is upgraded in the LDD system.
- **configureUpgradeSolutionServer.js**—The script is triggered after a solution is upgraded on each server in the LDD system.
- **configureRemoveSolution.js**—The script is triggered when a solution is removed from the LDD system.
- **configureRemoveSolutionServer.js**—The script is triggered when a solution is removed from each server in the LDD system.
- **configureSolutionRestart.js**—The script is triggered when a solution is installed or upgraded, or a server is restarted, in which case it is run for each solution.

To add an auto-configure script to a workflow solution, create a script using the selected name in the scripts\ folder of the workflow solution project.

Note: If a solution performs several actions using an auto-configure script, then increase the timeout setting for exporting a solution for deployment. You can configure this setting in the Lexmark preferences.

Using auto-configure methods

All methods used for auto-configure are members of the **caller** object, which is detailed in the *Lexmark Document Distributor Script Reference*. These methods can also be used in a scheduled script in which the **caller** object is assigned as follows:

```
var caller = context.getObject("autoconfigUtilities")
```

For an example of an auto-configure script that uses several of the available methods, see the configureNewSolution.js file in the TestMFP template in the Eclipse software. Also, the simplelog.js file in the scheduled\ folder of the TestMFP template contains the preceding **var** statement and uses auto-configure methods.

Configuring the home screen with auto-configure

The method **caller.setWelcomeScreen(groupName, solutionName, fileName)** configures the home screen for the specified solution in the specified group. The specified XML file, located in the "src\solution\welcomescreen\" folder, provides a description of the layout of the home screen.

The description of a home screen for an individual device class is specified using the element **<welcomescreen model="*class*">**, where *class* is one of the following:

- **etask**—For e-Task MFPs
- **sfp_etask**—For e-Task SFPs

- **etask2**—For e-Task 2 MFPs
- **etask2+**—For e-Task 2+ MFPs
- **sfp_etask2**—For e-Task 2 SFPs
- **etask3**—For e-Task 3 MFPs
- **sfp_etask3**—For SFP e-Task 3 SFPs
- **etask4**—For e-Task 4 MFPs
- **etask4(4.3)**—For e-Task 4 MFPs with a 4.3-inch screen
- **sfp_etask4(7)**—For e-Task 4 SFPs with a 7-inch screen
- **sfp_etask4**—For e-Task 4 SFPs with a 7-inch screen
- **etask5**—For e-Task 5 MFPs
- **sfp_etask5**—For e-Task 5 SFPs
- **x642**—For X642 printers
- **T656**—For T656 printers (SFP with a touch screen)

The layout is specified as an integer in the contents of the **<buttonLayout>** element. The settings apply as shown in the following table.

Buttons for the home screen are contained within a **<buttons>** element. Each button is defined using the **<button>** element, which contains the following child elements to define the properties of a button:

- **<action>**—Specifies the action associated with the button. Values available for each device class are shown in the following table. To leave a space on the home screen, do not include this element. If **Placeholder** is specified for the action, then do not specify any other properties for the button.
- **<displayText>**—The custom display text for the button.
- **<displayIcon>**—The custom icon for the button.
- **<profileName>**—The profile to assign to the button when **<action>** contains **Single Profile**, **Copy + Profile**, **Fax + Profile**, or **Email + Profile**.
- **<shortcut>**—The shortcut number assigned to the button when **<action>** contains **Shortcut**.

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask`	**1–9** The value specifies the number of buttons.	**Standard printer functions** • `Copy` • `Fax` • `Email` • `Forms` • `Printer Panel` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile`
`sfp_etask`	**1–*n*** The value specifies the number of buttons and must be the same as the number of `<button>` elements defined.	**LDD profile** • `Single Profile`
`etask2`	**1–9** The value specifies the number of buttons on the first page of the home screen. Three subsequent 9-button pages are included in the layout, providing 27 more buttons.	**Standard printer functions** • `Copy` • `Fax` • `Email` • `Ftp` • `Held Jobs` • `Search Held Jobs` • `Held Faxes` • `Lock Device` **LDD profile** • `Single Profile` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `Placeholder`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask2+`	**1−9** The value specifies the number of buttons on the first page of the home screen. Three subsequent 9-button pages are included in the layout, providing 27 more buttons.	**Standard printer functions** • `Change Language` • `Copy` • `Copy Shortcuts` • `Fax` • `Fax Shortcuts` • `Email` • `Email Shortcuts` • `Ftp` • `Ftp Shortcuts` • `Held Jobs` • `Search Held Jobs` • `Held Faxes` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` • `Placeholder` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile` • `Ftp + Profile`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`sfp_etask2+`	**1−9** The value specifies the number of buttons on the first page of the home screen. Three subsequent 9-button pages are included in the layout, providing 27 more buttons.	**Standard printer functions** • `Change Language` • `Search Held Jobs` • `Held Jobs` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` • `Placeholder`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask3`	**1−10** The value specifies the number of buttons on the first page of the home screen. Three subsequent 10-button pages are included in the layout, providing 30 more buttons.	**Standard printer functions** • `Change Language` • `Copy` • `Copy Shortcuts` • `Fax` • `Fax Shortcuts` • `Email` • `Email Shortcuts` • `Ftp` • `Ftp Shortcuts` • `Held Jobs` • `Search Held Jobs` • `Held Faxes` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile` • `Ftp + Profile`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`sfp_etask3`	**1−4** The value specifies the number of buttons on the first page of the home screen. Three subsequent 4-button pages are included in the layout, providing 12 more buttons.	**Standard printer functions** • `Change Language` • `Search Held Jobs` • `Held Jobs` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask4`	**1−10** The value specifies the number of buttons on the first page of the home screen. Three subsequent 10-button pages are included in the layout, providing 30 more buttons.	**Standard printer functions** • `Change Language` • `Copy` • `Copy Shortcuts` • `Fax` • `Fax Shortcuts` • `Email` • `Email Shortcuts` • `Ftp` • `Ftp Shortcuts` • `Search Held Jobs` • `Held Jobs` • `Release Held Faxes` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile` • `Ftp + Profile`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask4(4.3)`	**1−4** The value specifies the number of buttons on the first page of the home screen. Three subsequent 4-button pages are included in the layout, providing 12 more buttons.	**Standard printer functions** • `Change Language` • `Copy` • `Copy Shortcuts` • `Fax` • `Fax Shortcuts` • `Email` • `Email Shortcuts` • `Ftp` • `Ftp Shortcuts` • `Search Held Jobs` • `Held Jobs` • `Release Held Faxes` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile` • `Ftp + Profile`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`sfp_etask4`	**1–4** The value specifies the number of buttons on the first page of the home screen. Three subsequent 4-button pages are included in the layout, providing 12 more buttons.	**Standard printer functions** • `Change Language` • `Search Held Jobs` • `Held Jobs` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Placeholder for an eSF application installed with the solution** • `App Reservation`
`sfp_etask4(7)`	**1–10** The value specifies the number of buttons on the first page of the home screen. Three subsequent 10-button pages are included in the layout, providing 30 more buttons.	**Standard printer functions** • `Change Language` • `Search Held Jobs` • `Held Jobs` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Placeholder for an eSF application installed with the solution** • `App Reservation`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`etask5`	**1−15** The value specifies the number of buttons on the first page of the home screen. Four subsequent 15-button pages are included in the layout, providing 60 more buttons.	**Standard printer functions** • `Copy` • `Email` • `Fax` • `Held Faxes` • `Release Held Faxes` • `Status/Supplies` • `Job Queue` • `Change Language` • `Settings` • `Shortcuts` • `USB Drive` • `Held Jobs` • `Bookmarks` • `FTP` • `Scan Profiles` • `Lock Device` **LDD profile** • `App Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` • `Placeholder` **Standard functions overridden by profiles** • `Copy + Profile` • `Fax + Profile` • `Email + Profile` • `Ftp + Profile`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`sfp_etask5`	**1–8** The value specifies the number of buttons on the first page of the home screen. Eight subsequent 8-button pages are included in the layout, providing 64 more buttons.	**Standard printer functions** • `Status/Supplies` • `Job Queue` • `Change Language` • `Settings` • `USB Drive` • `Held Jobs` • `Bookmarks` • `Lock Device` **LDD profile** • `App Profiles` **Placeholder for an eSF application installed with the solution** • `App Reservation` • `Placeholder`
`x642`	**1–4** If the value is **1–3**, then it specifies the number of buttons. If the value is **4**, then it specifies a four-page layout of 4 buttons each, providing 16 total buttons.	**Standard printer functions** • `Copy` • `Fax` • `Email` • `Ftp` • `Held Jobs` **LDD profile** • `Single Profile` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `Placeholder`

Device class	Valid contents of the `<buttonLayout>` element	Valid contents of the `<action>` element
`t656`	**1–9** The value specifies the number of buttons on the first page of the home screen. Three subsequent 9-button pages are included in the layout, providing 27 more buttons.	**Standard printer functions** • `Change Language` • `Held Jobs` • `Search Held Jobs` • `Usb Drive` • `Bookmarks` • `Jobs by user` • `Lock Device` **LDD profile** • `Single Profile` **List of profiles** • `Profiles` **Shortcut** • `Shortcut` **Placeholder for an eSF application installed with the solution** • `App Reservation` • `Placeholder`

Determine the placement of each button by the order in which the buttons are defined.

- **MFPs and SFPs with touch screens**—The first button specified is the button nearest to the upper-left corner of the home screen in the specified layout. The order of the buttons progresses from left to right, and then from top to bottom by rows. If subsequent pages are present, then each page begins after all buttons have been specified on the previous page. To leave a blank space in the specified layout, define the button using the `<button>` element. Do not include the `<action>` element or any other properties of the button.

- **SFPs without touch screens**—The buttons appear as menu items in the order in which they are defined.

The following sample XML file specifies a home screen for e-Task 2 devices:

```
<?xml version="1.0" encoding="UTF-8"?>
<welcomescreen model="etask2">
    <buttonLayout>5</buttonLayout>
    <buttons>
        <!-- Buttons in page 1 -->
        <!-- A layout of 5 buttons is used, but only positions
             2 and 3 on the first page contain buttons -->
        <button/>
        <button>
            <action>Copy</action>
        </button>
        <button/>
        <button>
            <action>Fax</action>
            <displayText>My Fax</displayText>
        </button>
        <button/>

        <!-- Buttons in page 2 (page 2 starts after specifying all
             5 buttons on the first page) -->
        <!-- Place a profile button in position 5, using empty
             buttons for the first 4 -->
        <button/>
```

```
        <button/>
        <button/>
        <button/>
        <button>
            <action>Single Profile</action>
            <profileName>TestMFP</profileName>
            <displayText>Test MFP</displayText>
            <displayIcon>images\\testmfp_up.gif</displayIcon>
        </button>
    </buttons>
</welcomescreen>
```

An example file for the home screen on each device class can be found in the "\src\solution\welcomescreen" folder of the TestMFP example template.

Using forms merge scripts

You can use forms merge scripts to perform advanced management of forms merge operations or modify forms merge data and output documents. Using a forms merge script, you can:

- Send merged documents to multiple destinations, including e-mail messages, printers, network shares, FTP locations, or destinations provided by custom components included with the solution.
- Retrieve and modify input data.
- Create and insert new data pages.
- Retrieve the number and description of the port used to submit the job.
- Retrieve the name of the formset associated with the job.
- Retrieve global solution settings.
- Start another merge within the current merge.

The following scripting elements are unique to forms merge scripts:

- **mergeContext top-level object**—This provides fields and methods for forms merge jobs.
- **DataPage object**—This is a page of input data.
- **FormSetManageClass object**—This is used to manage the formset, allowing you to add, delete, and update formsets through the methods specified in this class.
- **MergeClass object**—This is used to merge data with a formset.

The script file TestMergeScript.js in the TestMergeScript example template is an example of a forms merge script.

Scripting for different stages of a forms merge

Forms merge scripts use callbacks to define functions that are activated at various stages of a forms merge job. The **main()** function in the script contains any actions executed when the job is first submitted, as well as assignments of callback functions. You only need to define and assign callback functions for the stages where scripted actions are necessary.

Assign callback functions to the following fields of the **mergeContext** top-level object at the end of the **main()** function (or, when using exception handling, at the end of the **try** block within the function):

mergeContext field to which the callback function is assigned	Stage of forms merge at which the assigned function is called	Parameters of assigned callback function
mergeContext.renderDataPage	After each page of input data is read	**page**—DataPage object containing the data from the page
mergeContext.endDoc	After input from a single form is completed and the resulting PDF is rendered	**fileName**—String containing the path and file name of the output PDF
mergeContext.endDataSet	After input from a single form is completed and the resulting dataset is generated **Note:** A dataset is generated only if this callback function is defined.	**fileName**—String containing the path and file name of the dataset, which can be sent to a printer with a forms card
mergeContext.endData	After all input data is read	None
mergeContext.endJob	After all PDFs included in the job have been rendered	**files**—Array of strings containing the file names of all output PDFs

A forms merge script with all callback functions defined is structured as follows:

```
function main()
{
    try
    {
        //Actions to take when job is first submitted
        …
        //Assignment of callback functions
        mergeContext.renderDataPage = myRenderDataPageFunction;
        mergeContext.endDoc = myEndDocFunction;
        mergeContext.endDataSet = myEndDataSetFunction;
        mergeContext.endData = myEndDataFunction;
        mergeContext.endJob = myEndJobFunction;
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}

function myRenderDataPageFunction(page)
{
    try
    {
        //Actions to take after each page of data
        //is read from the input data
        …
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}

function myEndDocFunction(fileName)
{
    try
    {
        //Actions to take after input from a single form is completed
        //and the resulting PDF is rendered
        …
```

```
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}

function myEndDataSetFunction(fileName)
{
    try
    {
        //Actions to take after input from a single form is completed
        //and the resulting dataset is generated
        …
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}

function myEndDataFunction()
{
    try
    {
        //Actions to take after all input data is read
        …
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}

function myEndJobFunction(files)
{
    try
    {
        //Actions to take after all PDFs included
        //in the job have been rendered
        …
    }
    catch(e)
    {
        //Logging and other exception handling
        …
        throw e;
    }
}
```

Using the PrintClass service with forms merge scripts

As in all scripts, you can use the **PrintClass** service to send a document to a printer.

Two fields are particularly important when using **PrintClass** in a forms merge script:

- **nativeSpooling**—Normally, the destination printer is identified by its IP address, but input data for a forms merge may contain the name of the output queue. Set this field to **true**, to identify the printer by queue name instead of by IP address.

- **optionMode**—This field should always be set to **PrintClass.OPTION_MODE_IGNORE** to make sure print settings from the form are used instead of settings that may be defined using **PrintClass**.

Printing forms on printers with forms cards

If you have printers with forms cards, you can choose to print forms using a dataset, an XML list of key-value pairs that represent input data. Most of the information about the final output form is stored on the forms card, and only the dataset is sent to the printer. Because less data is transferred over the network, use of datasets may provide better performance with limited network bandwidth.

To use a dataset, define **mergeContext.endDataSet** in **main()**, and then use the function assigned to the field to send the dataset to the printer.

Using scheduled scripts

Scheduled scripts can be created to assist with system maintenance. Since scheduled scripts are launched by the LDD system itself rather than a printer or software client, they cannot use objects that require user intervention, such as prompt objects and scan methods.

You can use auto-configure methods for making changes to the LDD system in scheduled scripts if the **caller** object is assigned as follows:

```
var caller = context.getObject("autoconfigUtilities")
```

You can also access an Additional Options field that an administrator can populate with free-form text when scheduling the task in LMC. You should provide the LDD administrator with documentation on any values that the script expects in the field. Scripts access the value supplied by the administrator using the field **taskInfo.additionalOptions**.

To create a scheduled script, create a new script in the src\scripts\scheduled\ folder.

For more information about adding scheduled scripts in LMC, see the *Lexmark Document Distributor Administrator's Guide*.

Examples of scheduled scripts can be found in the TestMFP and TestSNS example templates.

Sample script for an export event

```
Public Function fnJSAPIProfileCall(ByVal profileName As String,
pWorkdoc As SCBCdrPROJLib.SCBCdrWorkdoc)
   Dim SOAPResponse, XML

   SOAPResponse = fnJSAPIStart(profileName, pWorkdoc)

   ' Parse START response
   Set XML = CreateObject("Microsoft.XMLDOM")
   XML.setProperty "SelectionLanguage", "XPath"
   XML.async = False
   XML.loadXML(SOAPResponse)

   Dim names, values
   Set names = XML.selectNodes("//name")
   Set values = XML.selectNodes("//value")

   Dim key, sessionID, scanUrl, tomcatIP, webdavUser, webdavPasswd
   Dim idx
   For idx=0 To names.length-1
      key = names(idx).Text

      If key = "sessionID" Then
         sessionID = values(idx).Text
      ElseIf key = "tomcatIP" Then
         tomcatIP = values(idx).Text
      ElseIf key = "scanUrl" Then
         scanUrl = values(idx).Text
```

```
        ElseIf key = "webdavUser" Then
            webdavUser = values(idx).Text
        ElseIf key = "webdavPasswd" Then
            webdavPasswd = values(idx).Text
        End If
    Next

    ' Optionally put the original tiff to the correct webdav folder
    ' location for processing by the script being run.
    Dim doWebDavPut As Boolean
    doWebDavPut = True ' hardcoded True for this example
    If doWebDavPut Then
        Dim http, fNameWithoutExt, fName, fileObject, fileBytes
        ' Original tiff file
        fNameWithoutExt = Left(pWorkdoc.Filename, CStr(Len(pWorkdoc.Filename) - 3))
        fName = fNameWithoutExt & "tif"

        ' Load tif bytes
        Set fileObject = CreateObject("ADODB.Stream")
        fileObject.type = 1
        fileObject.Open
        fileObject.LoadFromFile fName

        fileBytes = fileObject.Read

        ' webdav put bytes
        Set http = CreateObject("MSXML2.XMLHTTP.3.0")
        http.Open "PUT", scanUrl & "/scan.0.file.0.tif", False, webdavUser, webdavPasswd
        http.setRequestHeader "translate", "f" ' Set this to prevent stream problems
        http.Send fileBytes ' Send the stream across
    End If

    fnJSAPIMetadata(sessionID, scanUrl, tomcatIP)
End Function
```

Understanding the objects used in solution development

Most tasks in scripted solutions for the Lexmark Document Distributor are completed by creating and manipulating objects. The Solution Builder Toolkit includes three main types of objects specific to development of solutions for Lexmark Document Distributor: top-level objects, prompts, and services.

Top-level objects

Top-level objects provide global references to the entity that launched the script, including references to the printer, confirmation page, prompts and services, and task information.

For example, the **taskInfo** object is used to access and set task information, such as the IP of the printer that originated the task or the status displayed with the job in LMC, and the **caller** object is used to execute a prompt on the printer.

```
function main()
{
  …
  // Use taskInfo to get the address of the originating printer
  var printerIP = taskInfo.printer;

  // Use taskInfo to set the task status in LMC
  taskInfo.status = "Running";
  …
  // Use the caller object to execute the prompt
  caller.ask(mystringprompt);
  …
}
```

Prompts

Prompts are used primarily during the document capture stage to receive inputs, including both documents and answers to prompts, from the caller (the printer or software client).
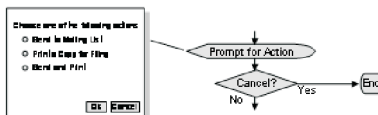
The following example uses a string prompt to illustrate the basic structure necessary to create, manipulate, and execute most prompts, although some prompts use a different structure. For more information and examples for specific prompts, see the *Lexmark Document Distributor Script Reference Guide*. Note that a separate function is used for the prompt object to enhance readability and reusability of code.

```
function main()
{
  …
  var value = promptForString()
  …
}
function promptForString()
{
  // Create instance of the StringPrompt object
  var myprompt = new StringPrompt;

  // Set text to display when the prompt is executed
  myprompt.text = "Enter a string";

  // Set the default value of the prompt
  myprompt.value = "sample response";

  // Execute the prompt
  caller.ask(myprompt);

  // Return the value of the prompt
  return myprompt.value;
}
```

Notes:

- No prompts should be used when developing a script to be run on a schedule.
- Some prompts are not supported for single-function printers. For more information, see "Supported printers" on page 10.

Services

Services are used after document capture for processing and routing documents, as well as providing confirmation of the job to the user. Services may also be used during document capture for processing answers to prompts, such as checking passwords. However, you should avoid excessive processing between successive prompts to prevent unexpected delays for the user.
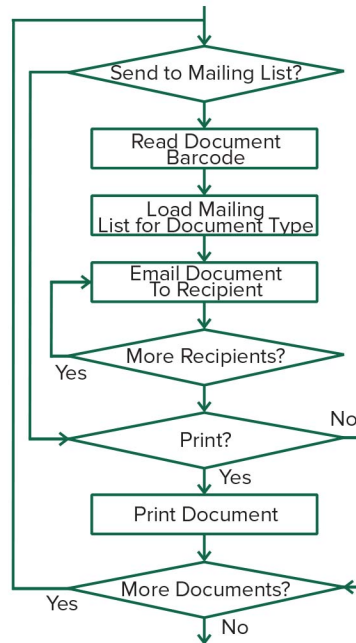
Use the following structure to create, manipulate, and execute a service:

```
// Create an instance of the service object "ServiceClass"
var myService = new ServiceClass();

// Set properties of the service object
myService.field1 = value1;
myService.field2 = value2;

// Execute the service
myService.execute();
```

Note: Services that require a user to scan a document should not be used when developing a script to be run on a schedule.

Configuring solutions

Managing solution settings

Solution settings can be accessed from scripts and are available to the LDD administrator through LMC during and after solution deployment. A solution setting is accessed from a script by name, just like a constant or variable.

Solution settings may be global or local. The following table lists the differences between global and local solution settings:

Setting type	Scope	Location in LMC	Time of first configuration	Common uses
Global	All device groups that use the solution	Configuration task on the Solutions tab	The solution is uploaded to LDD.	• Global logging on or off • Mail server address • Database server address
Local	A single device group that uses the solution	Solutions task on the Device Groups tab; dialog during deployment to a device group	The solution is added to a device group.	• Shared folder for a specific group. For example, one shared folder for a group of printers in the Accounting Department, and one for a group of printers in the Audit Department. • Scan settings. • E-mail address of the manager associated with each group.

Solution settings are generally configured throughout the solution development process as the need arises for each setting.

To create solution settings, double-click **solutionSettings.ldd** in the src\conf\ folder of the Package Explorer list. The Solution Settings Editor appears in the Editor pane. Separate lists of local and global solution settings are shown.

To edit an existing setting, select an item in the Local Solution Settings list or Global Solution Settings list.

To add a setting, click **New** beside the associated list.

To delete the current selection, click **Remove** beside the associated list.

To change the position of the current selection in LMC, click **Up** or **Down** beside the associated list.

Modify the properties of the selected or new setting using the fields on the right side of the window. While editing properties, click **Clear** to revert to the last saved properties for the setting, or click **Save** to save the new properties.

Property	Description
Key	The name used to access the property from a solution script.
Name	The name used for the property in LMC.
Read-Only (local solution settings only)	Determines whether the property is read-only. A read-only property cannot be modified using a script or in LMC.
Type	The type assignment for the property. See the following table of property types for a description of each type.

Type	Shown in LMC	Properties
Check	Check box	Values: • true • false

Type	Shown in LMC	Properties
Content Type	List of predefined content type settings	• Text • Photo • Mixed
Text	Value field	Any character **Note:** The Value field can be left blank.
Number	Value field	Any character **Note:** The Value field should not be left blank, even if the minimum value is set to zero.
Password	Value field	Any character
Darkness	List of predefined darkness values	Values: • 0 • 1 • 2 • 3 • 4 • 5 • 6 • 7 • 8
Depth	List of predefined color depth settings	Values: • Black and White • Grayscale • Color
Duplex	List of predefined duplex settings	Values: • Simplex • Long Edge Bound • Short Edge Bound
File Type	List of predefined file type settings	Values: • TIFF • JPEG • PDF • PS

Type	Shown in LMC	Properties
Media Size	List of predefined media-size settings	Values: • Letter • Legal • A4 • B5 • Executive • A5 • Business Card • Photo (3 x 5) • Photo (4 x 6) • Mixed
Resolution	List of predefined resolution settings	Values: • 150 • 200 • 300 • 400 • 600
Display Type	List of predefined printer standard prompt settings	Values: • Generic • Native

Managing device policies

The device policy developed within the solution determines the profiles that are made available to a printer or software client when the solution is deployed. Each profile is associated with a script within the solution. The policy also manages the default profile buttons, shortcut number, access control number, scan settings, and eSF applications associated with each profile. The device policy is generally configured near the end of the solution development process, since existing scripts are assigned to profiles.

To manage the device policy associated with the solution, double-click **devicePolicy.ldd** in the *src\conf* folder of the project in the Package Explorer pane. The Device Policy Editor appears in the Editor pane.

To add a profile, click **New**, type the key for the new profile, and then click **Create**.

To edit a profile, select it in the Device Profiles list.

To delete a profile, select it in the Device Profiles list, and then click **Remove**.

A device policy contains a separate sub-policy for each device class and for software clients. Select a device class from the Device Group list to access the sub-policy for each, or select **Software** to access the sub-policy for software clients.

Notes:

- Each sub-policy includes unique settings for Name, Task, No Scan, Profile buttons, Shortcut Number, Access Control Number, and Settings. Values are shown for only the currently selected Device Group.
- Sub-policies for single-function printers are created using their respective device classes. Select **No Scan** to enable deploying a profile to single-function printers.

- Scan Settings, Profile buttons, Shortcut Number, and Access Control Number do not apply to software clients. These settings are not available when Software is selected.

To reset all settings for a sub-policy, click **Reset** beside the Device Group list. While editing properties, click **Clear** to revert to the last saved properties for the profile, or **Save** to save the new properties.

Note: Changes in a sub-policy must be saved before selecting a different Device Group. If the sub-policy has been changed and you select a different Device Group, then you are asked whether to save changes to the current sub-policy.

The following table shows the properties available for each device profile:

Property	Description
Key	This property is the name used to access the profile from a script. The key applies to the policy as a whole and does not vary for sub-policies.
Device Group	This property allows the solution developer to select which device subtypes are in the profile. The device group can be any of the following: • **e-Task 2**—All devices with eSF framework support • **e-Task**—Devices without eSF support • **Software**—LDD printer ports, the Microsoft Windows application software Select'N'Send, hybrid eSF applications, and other applications that submit jobs to an LDD System using the LDD Job Submission Web Service
Name	This property is the name used for the profile in Lexmark Management Console. **Note:** For a profile used with a Lexmark Document Server Port, the name must be 14 or fewer characters.
Task	This property is the script associated with the policy.
No Scan (e-Task and e-Task 2)	This property indicates whether a scan task is part of the solution. A profile can be deployed to a single-function printer only when No Scan is selected.
No Prompt	This property is a control added to note indicating that the profile does not contain prompting. It helps improve the performance of the LDD printer ports for print servers that do not support prompting. This setting is also active for device groups which are set to Software. Select this setting for any existing Software solutions, such as print release, to improve the LDD printer port performance for print servers.
Setting (e-Task and e-Task 2)	This property is the setting type to add to or edit in the list of settings. See the following table of setting types for a description of each type and available values.
Mapped (e-Task and e-Task 2)	This property determines whether the setting is mapped to a solution setting defined in solutionSettings.ldd.
Value (e-Task and e-Task 2)	This property is the value of the setting to add to or edit in the list of settings. See the following table of setting types for the available values for each setting type.
Overrideable (e-Task and e-Task 2)	This property determines whether the user at the printer can override the setting when the solution is run.
Shortcut Number	This property is the shortcut number by which a user can access the profile from the printer.
Access Control Number	This property is the number used to assign access controls specifically to the profile, using one of the Solution 1–10 settings in the printer Embedded Web Server.

The following table shows the setting types available for device profiles:

Type	Available values
Content Type	• Text • Photo • Mixed
Darkness	• 0 • 1 • 2 • 3 • 4 • 5 • 6 • 7 • 8
Depth	• Black and White • Grayscale • Color
Duplex	• Simplex • Long Edge Bound
File Type	• TIFF • JPEG • PDF • PS
Media Size	• Letter • Legal • A4 • B5 • Executive • A5 • Business Card • Photo (3 x 5) • Photo (4 x 6) • Mixed
Orientation	• Portrait • Landscape
Resolution	• 150 • 200 • 300 • 400 • 600

Profile buttons

You can package profile buttons to appear on the home screen as part of the device policy for e-Task and e-Task 2 devices. When e-Task or e-Task 2 is selected in the Device Group list, the Profile Button(s) section appears. Additionally, when e-Task 2 is selected, a selection appears for e-Task 2 or e-Task 2+ profile buttons, representing older and newer e-Task 2 printers, respectively. You can determine which setting to use by matching the preview of the default image with the look of the existing buttons on the printer display.

You can load separate images for Up Image and Down Image. The Up Image shows when the button is in default state, and the Down Image shows while the user is pressing the button. If no Down Image is specified, then the Up Image is used for both conditions.

Note: File names of images used for the home screen cannot contain double-byte characters.

Images selected are resized to the following dimensions for each device class. For best results, resize or crop source images to the correct size before use.

- **e-Task 5**—140 x 140 pixels
- **e-Task 3** and **e-Task 4**—172 x 254 pixels
- **e-Task 2 and e-Task 2+, SFP e-Task 2+**—120 x 75 pixels
- **e-Task**—120 x 80 pixels
- **X642**—120 x 55 pixels

eSF applications

To associate an eSF application with the e-Task 2 sub-policy for a hybrid solution, click **Import** beside the eSF Applications list, and then locate an eSF application. Select the newly imported application in the list, and then make sure that **e-Task2** is selected in the Device Group list. To remove a previously imported eSF application, select it from the list, and then click **Remove**.

Note: File names of eSF applications used in a solution cannot contain double-byte characters.

To manage the settings associated with the eSF application and allow configuration in LMC, import and configure the descriptor file for the eSF application.

Notes:

- LDD cannot accept colons in XML tag names, which appear in some descriptor files for eSF applications. Uploading a solution with a descriptor file containing colons in tag names disables the Solutions tab and the eSF Configuration task for device groups. Restoring access requires intervention from Lexmark Professional Services. Before importing the descriptor file, open it in a text editor and remove any colons from tag names. For example, if the tag **<esf:display pattern="instptrn">** appears in the file, then change it to **<esfdisplay pattern="instptrn">**. Make sure to change matching closing tags as well.
- LDD can manage eSF application settings for e-Task 2+ printers only. Configure eSF application settings for e-Task 2 and X642 printers using the Embedded Web Server on each printer after the application is deployed.

1 Select the eSF application from the eSF Application list.

2 Click **Descriptor**.

3 Locate the XML descriptor file for the eSF application. The file name should be the same as the file name of the eSF application, with an ".xml" extension and "_desc" appended before the extension.

4 Click **Open**.

5 Select the eSF application from the eSF Application list.

6 Click the **eSF Descriptor Editor** tab at the bottom of the Device Policy Editor.

7 Modify the settings if necessary, and then click **Save**.

Managing formsets

You can use forms merge in a solution by including formsets created with Lexmark Forms Composer. Formsets included in a workflow solution project are uploaded to the LDD system when the workflow solution is installed.

Formsets are added as needed throughout the solution development process. To manage formsets, double-click **forms.ldd** in the src\conf\ folder of the Package Explorer list. The Forms List Editor appears in the Editor pane. The list of included formsets is shown.

To add a formset that will be installed with a workflow solution:

1 In the Forms List Editor, click **Import**.

2 Browse to a formset (.sdd or .fdd) file.

3 Click **Open**.

When uploaded, formsets may be shared among workflow solutions, and the administrator selects whether to remove associated formsets when removing solutions. To create a link with a formset included in another workflow solution:

1 In the Forms List Editor, click **Link**.

2 In the Form Name field, type the name of the formset.

3 Click **Create**.

To remove a formset from the project, select it from the Forms List, and then click **Remove**.

Managing custom reports

LDD uses JasperReports for custom reports, which can be included with a workflow solution during development. You can create custom reports using the open source application iReport 3.5.2. For more information about overall report design, see the documentation for iReport, and for specific information about developing custom reports for LDD, see the *Lexmark Document Distributor Administrator's Guide*.

Custom reports are added as needed throughout the solution development process. To manage custom reports, double-click **reports.ldd** in the src\conf\ folder of the Package Explorer list. The Reports List Editor appears in the Editor pane. The list of included reports is shown.

To add a report that will be installed with a workflow solution:

1 In the Reports List Editor, click **Import**.

2 Browse to a compiled JasperReports (.jasper) file.

3 Click **Open**.

To edit the settings of a report after it is imported:

1 Select the report file from the Reports list.

2 Type a descriptive report name.

3 Select an output format.

Note: Some reports are designed for a particular output format. For more information about a custom report, contact the designer of your custom report.

4 Select **Database** from the Datasource drop-down menu.

5 Select **Monitor** from the Source Choice drop-down menu.

6 If you are using auto-configure to schedule the report, then select the period for the report to cover.

7 Add a custom parameter for either a query parameter defined in the custom report file:

a In the Key field, type the name of the parameter defined within the report.

b Select the parameter type to identify the type of data contained in the parameter.

c Type a default value for the parameter.

Notes:

- To add multiple custom parameters, save the report settings, and then enter the information for the next custom parameter.
- To remove a custom parameter, select it from the list, and then click **Remove**.

8 Click **Save**.

Using the included document processing components

Bar code read support

The LDD SDK includes two bar code read packages: a premium Windows native package, and a lower-cost Java package. The following tables compare the details of each package:

	Premium package	Value package
LDD script objects	`BarcodeReadClass` `BarcodeConstant`	`BarcodeReadLiteClass` `BarcodeReadLiteConstant`
License fees	Required for each LDD server installation	Included in system cost
Code	Windows native (.NET or COM)	Java
Performance	Faster	Slower
Solution template project	TestBarcodeRead	TestBCRead
Ideal scan resolution	300 dpi	300 dpi
Expected quality	Excellent, Good, Fair, or Poor	Interval between scanned pixel rows, 1–100
Reading from a specified zone	Yes	No
Automatic detection of one-dimensional codes	Yes	No
Checksum validation	Yes	Yes
Fixed length	Yes	No

	Premium package	Value package
Minimum length	Yes	No
Page number	Yes	No
Coordinates	Yes	Yes
Multiple codes with directional sorting	Yes	Yes

One-dimensional symbology support

	Premium package	Value package
Codabar	Yes	Yes
Code 11	No	Yes
Code 25	Yes	No
Code 32	Yes	Yes
Code 39	Yes	Yes
Code 39 mod 43	Yes	No
Code 93	Yes	Yes
Code 128	Yes	Yes
EAN 8	Yes	Yes
EAN 8 + 2	Yes	Yes
EAN 8 + 5	Yes	Yes
EAN 13	Yes	Yes
EAN 13 + 2	Yes	Yes
EAN 13 + 5	Yes	Yes
GS1-128 (formerly UCC/EAN-128)	Yes **Note:** The GS-128 symbology is specified for the **type** field of **BarcodeReadClass** using the constant **BARCODE_UCC128**	No
GS1 DataBar (formerly RSS)	No	Yes **Note:** The GS1 DataBar symbology is specified for the **type** field of **BarcodeReadLiteClass** using the constant **BC_CODERSS14** or **BC_CODERSSLIMITED**.
ITF-14	Yes	No
Patch Code	Yes	Yes
PLANET	Yes	Yes
POSTNET	Yes	Yes
QR Code	No	Yes

	Premium package	Value package
RM4SCC	No	Yes
Telepen	No	Yes
UPC-A	Yes	Yes
UPC-E	Yes	Yes
USPS Intelligent Mail	Yes	No

Two-dimensional symbology support

	Premium package	Value package
Datamatrix	Yes	Yes
PDF417	Yes	Yes

Bar code write support

The LDD SDK includes two bar code write packages: a premium Windows native package, and an open source, Java-based Barbecue. The following tables compare the details of each package:

Features and usage

	Premium	Barbecue
LDD script objects	**BCWriteClass** **BCWriteClass.options** **BarcodeConstant**	**ImagingClass.options.overlay.barcode** **BarcodeConstant**
License fees	Required for each LDD server installation	Open source; no fees
Code	Windows native (.NET or COM)	Java
Custom minimum width	No	Yes
Human-readable text position	Above, below, or embedded	Below only
Human-readable text justification	Controllable	Fixed

	Premium	Barbecue
Codabar	Yes	Yes
Code 25	Yes	Yes
Code 25 Interleaved	Yes	Yes
Code 39	Yes	Yes
Code 93	Yes	No
Code 128	Yes	Yes
Code 128A	Yes	Yes

	Premium	Barbecue
Code 128B	Yes	Yes
Code 128C	Yes	Yes
EAN 8	Yes	No
EAN 8 + 2	Yes	No
EAN 8 + 5	Yes	No
EAN 13	Yes	Yes
EAN 13 + 2	Yes	No
EAN 13 + 5 (Bookland)	Yes	Yes
GS1-128 (formerly UCC/EAN-128) **Note:** The GS-128 symbology is specified for the `type` field using the constant `BARCODE_UCC128`	Yes	Yes
GS1-DataMatrix	Yes	Yes
ITF-14	Yes	No
POSTNET	Yes	Yes
QR Code	Yes	No
UPC-A	Yes	Yes
UPC-E	Yes	No
USPS Intelligent Mail	Yes	No

	Premium	Barbecue
PDF417	Yes	No

OCR support

LDD optionally includes an optical character recognition (OCR) module to retrieve text from scanned documents, enabling them for searching, editing, and indexing. Entire documents, specific pages of documents, or specific areas of pages can be stored as text documents.

The following table lists the features of the OCR module:

Feature	Details
Supported languages	- Arabic (LDD 5.0 or later) - Croatian (LDD 5.0 or later) - Cyrillic Serbian (LDD 5.0 or later) - Czech (LDD 4.8.5 or later) - Danish - Dutch - English - Finnish - French - German - Greek (LDD 5.0 or later) - Hebrew (LDD 5.0 or later) - Hungarian (LDD 5.0 or later) - Italian - Japanese (LDD 4.9 or later) - Korean (LDD 4.9 or later) - Latin Serbian (LDD 5.0 or later) - Norwegian - Polish - Portuguese - Romanian (LDD 5.0 or later) - Russian (LDD 5.0 or later) - Simplified Chinese (LDD 4.9 or later) - Slovakian (LDD 5.0 or later) - Slovenian (LDD 5.0 or later) - Spanish - Swedish - Traditional Chinese (LDD 4.9 or later) - Turkish (LDD 5.0 or later)
Dictionaries	Each language has one associated dictionary. The search order of language dictionaries can be configured in a script.
Supported text types	- Common typographic (serif, sans-serif, italic, monospace) - Typewriter-printed - Dot-matrix-printed - ZIP-code-style numerals - OCR-A - OCR-B - MICR (E-13B and CMC-7) - Gothic
Supported input text size	10 points–220 points

Feature	Details
Default output fonts	The following selections are made for default output fonts based on the input font: • **Serif fonts**—Times New Roman • **Sans-serif fonts**—Arial • **Monospaced fonts**—Courier New Output fonts can be changed to any supported TrueType fonts within a script, after the OCR operation, and before the text is exported to a document.
Zoning	• **Automatic**—The entire page is scanned and analyzed for blocks of text. • **Manual**—The script defines regions on a page for OCR scanning. This method is faster, since it does not require analyzing the entire page.
Available output formats	• HTML • Searchable PDF – PDF version 1.7 (LDD 4.9 or later) – PDF versions 1.3 to 1.6 (LDD 4.8.1 or later) – PDF/A (LDD 4.8.1 or later) – PDF/A-1a (LDD 4.8.1 or later) – PDF/A-1b (LDD 4.9 or later) – PDF/A-2a (LDD 4.9 or later) – PDF/A-2u (LDD 4.9 or later) – PDF/A-3a (LDD 4.9 or later) – PDF/A-3u (LDD 4.9 or later) – MRC compression (LDD 4.8.5 or later) • Plain text (TXT) • Rich Text Format (RTF) • Microsoft Office formats – docx (LDD 4.9 or later) – xls (LDD 4.9 or later) – xlsx (LDD 4.9 or later) – pptx (LDD 4.9 or later)
Document color	Supported for input: • Bitonal • Grayscale • Color Supported for output: • Bitonal • Color
Resolution	Input is accepted from documents scanned at 150–600 dpi. For best resolution, scan documents at 300 dpi.

Feature	Details
Image-processing capabilities	The following image-processing tasks are available: • Despeckle • Deskew • Rotation (90-degree increments) • Identification of text without output of text

Accuracy factors

The accuracy of the OCR module depends on the following original document factors:

- **Media type**—More-reflective specialty media may result in poor-quality scanned images.
- **Document condition**—Aged, wrinkled, or creased documents may result in poor-quality scanned images.
- **Document image quality**—Low resolution or color depth of the scanned image or poor print quality in the scanned document may result in poor-quality scanned images.
- **Document alignment**—Skewed or slightly rotated documents, such as pages fed through a misaligned ADF may cause difficulty in reading the scanned image.

Understanding the included solution database

LDD uses Firebird® as the default database. When using Firebird, you can create and modify tables without creating a custom database. Data in the included database is automatically migrated to a new version when the Firebird database used by LDD is upgraded.

The included database is SOLUTIONINFO.fdb, located in the **\firebird\data** folder where the LDD database is installed.

The following are the credentials for the database:

- **User name**—`framework`
- **Password**—`rivet`

Understanding the Job Submission Web Service

LDD 5.x provides an interface called the Job Submission API (JSAPI) that allows web service–capable client software to interact with an LDD system. It supports both Representational State Transfer (REST) and SOAP web services. Client applications (such as eSF applications, the Microsoft Windows application software Select'N'Send, Lexmark Document Server Ports, and other applications) can use these services to run an LDD script or profile. Using JSAPI, the client software can also pass extra metadata to the running script or profile for additional processing or data capture.

Through the web service, clients can do the following:

- Query the system for the profiles available to a client system based on the IP address of the client.
- Submit documents to the LDD system.
- Execute a profile to process documents.
- Query the system for task progress and status.

The following requirements must be met for the web service to be available to a client:

- A software client license is installed for the client.
- The IP address of the client has been added to a software client group.
- A solution with the client software script or profile is installed and added to a software client group.

  Note: Multiple software client groups and client software solutions, scripts, or profiles can be installed on an LDD system.

- The LDD system is online.

Using REST JSAPI

REST JSAPI is available only in LDD 4.6.4.1 or later.

Notes:

- REST JSAPI supports both HTTP and HTTPS.
- Strings passed to and from the API must be UTF-8 encoded.
- Dates returned from the API are in POSIX time format.

Use the following HTTP methods when sending requests to the server:

- POST—Create a resource.
- GET—Retrieve a resource.
- PUT—Edit a resource.
- DELETE—Remove a resource.

The base URI for RESTful JSAPI is **http://loadbalancer:9780/lmc/rws/jsapi**, where **loadbalancer** is the name or IP address of the load balancer.

Submitting jobs

Sample request

```
URI: /jsapi/v1/jobs
Method: POST
Version: v1
Content-type: multipart/form-data
HTTP Form Parameters: (*Required parameter)
```

The following table lists the HTTP form parameters used when submitting a job:

Parameter key or object	Value type	Description
profileName*	String	The name of the LDD profile to execute that is in an LDD SDK solution installed on an LDD system.
		For example, use **Test SNS** with the TestSNS example solution.
authId*	String	The name of the user who submitted the job.
clientIPAddr*	String	The IPv4 address of the client submitting the job.
		Note: IPv6 addresses are not supported.
* Required parameter		

Parameter key or object	Value type	Description
clientHostName*	String	The host name of the client submitting the job. A fully qualified domain name (FQDN) is preferred. **Note:** If the host name cannot be determined, then use the IP address.
clientMACAddr*	String	The network card MAC address of the client submitting the job. For example, **12:BA:4F:67:91:B4**.
submitTime*	Long	The time on the client when job submission started. **Note:** Use POSIX time format. For example, **1340125377 GMT**.
appId*	String	The application ID where the job is coming from. Use one of the following IDs: • **PNS**—Use for jobs sent from LDD printer ports. • **SNS**—Use for jobs sent from the Microsoft Windows application software Select'N'Send for LDD. • **eSF**—Use for jobs sent from eSF applications. • **LSP**—Use for jobs sent from Lexmark Solutions Platform (LSP). • **mobile**—Use for jobs sent from mobile applications. **Note:** The TestSNS example solution only looks for PNS and SNS IDs.
resolver	String	Use either of the following: • **software**—Use for clients that are associated with a software client group and client software script or profile. • **hosted**—Use if the client is not associated with LDD software client group settings. **Note:** If no value is added, then the parameter defaults to **software**.
language	String	The language that is used by the client. Use the format **ISO 639-1_ISO 3166-1**. For example, **en_US**. The first two letters are important, and the next two letters are used only to distinguish between Simplified (zh_CN) and Traditional (zh_TW) Chinese. **Notes:** • This language ID format is used by Lexmark printers. • For more language strings, navigate to the LDD SDK script reference, and then click **taskInfo** > **language information (taskInfo.language)**. You can also open a web browser and search for "ISO 639-1" and "ISO 3166-1." • If no value is added, then the parameter defaults to **en_US**.
xyz	String, integer, and so on	Other metadata (key and value pairs) that is needed by the specific profile launched by the job submitted.
file object	File object	File binary, file name, file extension, and so on. The key name must be **prompt.[x].file.[y]**, where **x** is the scan prompt number and **y** is the number of the file returned by the prompt. For example, **prompt.0.file.0**. **x** and **y** are both zero-based.
* Required parameter		

201 Created indicates a successful response, and the HTTP header value indicates the new resource location.

Sample response

```
201 Created
Location: http://xxx.xxx.xxx.xxx:9780/jsapi/v1/jobs/123456
Content-Type: application/json
[json-data]
```

Sample success JSON response

```
{
"jobId":"8087","tomcatIP":"http://10.195.x.y:9788","noprompt":"true","prompt_timeout":"30"
}
```

If the submission fails, then one of the following errors is returned:

- **400 - client did not send required data -** *list of required parameters that are missing*
- **403 - client software not licensed**
- **403 - client address is not in the list of allowed devices**
- **403 - client user is not authorized**
- **404 - profile name does not exist**

Sample error JSON response for a 403 error

```
{
"reason": "client software is not licensed"
}
```

Querying a job status

Sample request

```
URI: /jsapi/v1/jobs/{jobID}
Method: GET
Version: v1
Parameters: (*Required parameter)
```

{jobID} is the ID returned from the submit job request.

The following table lists the keys returned when querying a job status:

Key	Value type	Description
jobID	String	The ID of the job you are querying.
state	String	The status of the job you are querying. The status can be one of the following: • **Running**—The profile initiated by the job is still running. • **Completed**—The profile initiated by the job is completed. • **Error**—The profile initiated by the job is stopped because of an error.
percentComplete	Integer	The percentage of how much the profile initiated by the job is completed. **Note:** If the **percentComplete** value is not **100** or the **state** value is not **error**, then do not update **percentComplete** in the profile.

Key	Value type	Description
solutionName	String	The name of the solution on the LDD system that contains the profile.
taskName	String	The name of the task or script running on the LDD system for the profile.
submitTime	Long	The time on the client when job submission started. The returned time is in POSIX time format. For example, **1340125377 GMT**.
startTime	Long	The time on the LDD system when the job started. The returned time is in POSIX time format. For example, **1340125377 GMT**. **Note:** If the job is not started, then this key is not added in the returned JSON data.
endTime	Long	The time on the LDD system when the job ended. The returned time is in POSIX time format. For example, **1340125377 GMT**. **Note:** If the job is not started, then this key is not added in the returned JSON data.
authID	String	The name of the user who submitted the job.
clientIPAddr	String	The IPv4 address of the client submitting the job. **Note:** IPv6 addresses are not supported.
clientHostName	String	The host name of the client submitting the job. **Note:** If the host name cannot be determined, then the IP address appears.
clientMACAddr	String	The network card MAC address of the client submitting the job.
appID	String	The application ID where the job is coming from. The ID can be one of the following: • **PNS**—For jobs sent from LDD printer ports. • **SNS**—For jobs sent from the Microsoft Windows application software Select'N'Send for LDD. • **eSF**—For jobs sent from eSF applications. • **LSP**—For jobs sent from Lexmark Solutions Platform (LSP). • **mobile**—For jobs sent from mobile applications. **Note:** The TestSNS example solution only looks for PNS and SNS IDs.
language	String	The language used by the client.
srvIPAddr	String	The IP address of the LDD server processing this job.
srvHostName	String	The host name of the LDD server processing the job. **Note:** If the host name cannot be determined, then the IP address appears.
srvThreadName	String	The name of the thread that is processing the job. For example, **workflow-29**.

Sample success JSON response

```
200 OK
Location: http://xxx.xxx.xxx.xxx:9780/jsapi/v1/jobs/123456
Content-Type: application/json
Content-Length: xxxx
{
"jobId": "123456",
"state": "running",
"percentComplete": 40,
"solutionName": "TestSNS",
"taskName": "testsns",
```

```
"submitTime": 1340125377,
"startTime": 1340125400,
"authId": "smith",
"clientIPAddr": "10.190.64.123",
"clientHostName": "my-pc.area.company.com",
"clientMACAddr": "12:BA:4F:67:91:B4",
"appId": "mobile",
"language": "en_US",
"srvIPAddr": "10.190.64.75"
"srvHostName": "lddsrv1.area.company.com"
"srvThreadName": "workflow-29"
}
```

Using SOAP JSAPI

Note: SOAP JSAPI is not supported in LDD 5.4 or later.

SOAP JSAPI is available in LDD 4.3.x.x or later. We recommend using RESTful JSAPI instead of SOAP JSAPI to work with your client applications. For more information, contact your Lexmark representative.

The following table lists the methods used with the SOAP Web Service:

Method	Target	Description	Parameters	Return values
start	Load balancer	Initiates a job submission session.	• **resolver**—The resolver used to determine the device group, solution, and task settings. The default resolver uses clientIP and profileName only. • **protocol**—The protocol used to extract prompt answers from the metadata. The default is **webdav**. • **authId**—The client user name. • **clientIP**—The client IP address. • **deviceGroup**— The device group associated with the job. • **profileName**— The profile used for the job.	• **sessionID**— The session ID, which must be used in later calls. • **tomcatIP**—The address of the LDD server, which must be the target for later calls. • **scanUrl**—The WebDAV folder where documents are submitted.
post	WebDAV	Submits files in response to **ScanPrompt**.	See the WebDAV documentation.	

Method	Target	Description	Parameters	Return values
metadata	LDD server	Provides answers for other prompts.	• **continue**—Indicates that more metadata follows. • **scan.*x*.file.*y***—The path of a scanned file in the WebDAV repository. *x* indicates the sequence number of a scan prompt, and *y* indicates the sequence number of a file from that scan prompt. *x* and *y* are both zero-based.	Reserved.
nextPrompt (optional)	LDD server	Requests for the next prompt.	**sessionID**—The session ID returned from the initial **start** call.	**return nextPrompt**—The parameters of returned changes based on the prompt type. **finish**—Shows the finish button. **back**—Lets users go to the previous prompt. **type**—Prompt type.
queryTaskStatus (optional)	LDD server	Calls this method with the **sessionID** to determine if the job is successful.	**sessionID**—The session ID returned from the initial **start** call.	**percentComplete**—The percentage of the job completion that is updated as the script runs. **id**—The session ID from the start of the job. **status**—Shows the status of the job. **startTime**—The time when the job started. **endTime**—The time when the job ended.

Understanding dynamic prompting support

Many solutions involve submitting jobs that require information such as account numbers, passwords, or processing options. When a profile is selected, solutions may either prompt you for information or start a job automatically.

The Job Submission Web Service provides prompting capability to Lexmark Document Server Printer Port clients. The following prompts are supported:

- **MessagePrompt**
- **BooleanPrompt**
- **IntegerPrompt**
- **ListPrompt**
- **ArrayPrompt**
- **PasswordPrompt**
- **ScanPrompt**
- **MessagePrompt**
- **EndPrompt**

The timeout period for the prompt wizard is 30 seconds.

Prompting depends on whether there are more required information or documents to process. If there are more, then select whether to continue or not. On the last prompt, a summary dialog box of the answers appears.

Notes:

- Canceling the job does not undo the processing of the job.
- LDD 4.6.3.x printer ports on a clustered print server are not supported.

Using Kerberos authentication

If a user logs in at a printer using Kerberos, then the LDD system uses the credentials to manipulate network files or interact with ECM systems.

Notes:

- Enable secure communication between printers and servers in the LMC for any device group that uses a solution with Kerberos authentication.
- Make sure that the date and time on the printer, LDD server, and KDC server are synchronized.

The following objects use Kerberos authentication:

- FileClass
- LdapClass
- SharePointClass
- TextFileClass

If necessary, Kerberos tickets are accessed, and no additional scripting is necessary.

Note: Credentials set for an object in a solution script override Kerberos credentials.

You can set a KDC address and realm within a script using the **credentials** top-level object:

```
credentials.add("kdc", "myhost.mycompany.com");
credentials.add("realm", "myrealm.mycompany.com");
```

Developing workflow solutions for earlier versions of LDD

To develop workflow solutions for earlier versions of LDD 5.x, use the current version of the LDD SDK.

Note: Automatic complete is installed only on the latest version of LDD. For more information, see the *Lexmark Document Distributor Script Reference*.

Testing and deploying workflow solutions

Testing with the MFP emulator and performance simulator

The MFP emulator and performance simulator are used to test LDD solutions without a physical printer. The emulator interactively emulates an e-Task 2 printer control panel. The performance simulator lets you set parameters and quickly run multiple tests simulating either an e-Task or an e-Task 2 printer. The interactive emulator is useful for preliminary debugging, and the performance simulator is useful for comprehensive functional and stress testing.

Note: In Windows 7 or later, make sure to run the Eclipse software and LDD SDK as an administrator, for the simulator and emulator to function properly.

Accessing the MFP emulator or performance simulator

- To access the interactive MFP emulator, in the Eclipse software, click **LDD SDK** > **MFP** > **Emulator (eTask2)**.
- To access the performance simulator, in the Eclipse software, click **LDD SDK** > **MFP** > **Performance Simulator**.

If you are running the MFP emulator or performance simulator for the first time, then the installation window appears. To install the application, do the following:

1 If necessary, change the folder where you want to install the application.

   Note: The installation path cannot contain double-byte characters.

2 If necessary, select **Launch Emulator** to launch the emulator or performance simulator after installation.

3 Click **Finish**.

If you selected **Launch Emulator** or if the MFP emulator is already installed, then the interactive MFP Emulator or MFP Simulator configuration window appears.

Using the interactive MFP emulator

Using the interactive MFP emulator with an LDD system

1 From the MFP Emulator configuration window, click **Interactive Mode** > **MFP** > **Emulator (eTask2)**.

   The home screen window appears, and the emulator is accessible as a printer from LMC.

2 Make sure that the LDD system is online.

3 Upload the solution to be tested in LMC. For more information, see the *Lexmark Document Distributor Administrator's Guide*.

4 Add the MFP emulator to a device group in LMC, and then discover it.

   Note: The IP address of the emulator is the IP address of the computer where it is running. This address appears beside Client IP in the MFP emulator configuration window, and on the upper-left corner of the home screen.

5 Deploy the solution, modify the home screen for the device group as necessary, and then perform a policy update.

6 If you include the profile on the home screen, then click the icon for the profile. If the profile does not appear on the home screen, then do the following:

    a From the printer home screen, do either of the following:

- For e-Task 5 printers, click **App Profiles**.
- For e-Task 2+, e-Task 3, or e-Task 4 printers, click **Held Jobs** > **Profiles**.

    b Click the icon for the profile.

The profile launches the associated script on an LDD server.

Note: Some prompts are not supported in interactive mode. When a script includes an unsupported prompt, a message appears on the emulated printer control panel. Click **Next** to continue the script after the prompt.

Simulating a scan task

1 Type or browse to the path of a TIFF, JPEG, PDF, or PostScript file to simulate the document to be scanned.

2 Click **OK**.

To assign a default image file to appear in each Scan File dialog, click **File** > **Properties**.

Saving print jobs

Print jobs initiated by a profile are discarded by default. To enable saving print jobs:

1 Make sure you have run the interactive emulator at least once. The configuration that contains the settings for the interactive emulator is created automatically the first time it is run.

2 Locate the \Profiles\interactive\conf\ folder where the MFP emulator is installed. The default is C:\Program Files\mfpsimulator\.

3 Open **sim-interactive.properties** in a text editor.

4 If necessary, modify the location where print jobs are saved beside `interactive.advanced.pj.output=`.

5 Change the value beside `interactive.advanced.pj.save=` to `true`.

6 If the interactive emulator is open, then close, and then restart it.

Using the performance simulator

Preparing and running tests in the performance simulator

1 From the MFP Simulator configuration window, click **File** > **New Configuration**.

2 Type the IP address or host name of the LDD system in the Load Balancer IP field.

3 Type the profile name to execute.

Note: The profile name refers to a device profile created using the Device Policy Editor in the solution. A list of profiles can also be found in the Profiles task after the solution has been deployed to the relevant device group in LMC.

4 Select whether to simulate an e-Task 2 or e-Task device.

5 Discover the performance simulator and deploy the appropriate solution in LMC:

Note: This step is only necessary if the interactive MFP emulator or performance simulator has not been discovered, or if the necessary solution has not been deployed.

a To set the simulator to discovery mode, type **0** for both "Number of MFPs" and Repetitions fields.

b Click to save the test profile, and then click to start the simulator in discovery mode.

c Make sure the LDD system is online.

d Upload the solution to be tested in LMC.

e Add the performance simulator to a device group in LMC, and then discover it.

Note: The IP address of the simulator is the IP address of the computer where it is running. This address appears in the Client IP field in the MFP Simulator configuration window.

f Deploy the solution and modify the home screen for the device group as necessary, and then perform a policy update.

g Click to stop the simulator.

6 Type the number of MFPs to simulate.

7 Type the number of repetitions for the test.

8 In the Scan File field, browse to a TIFF, JPEG, PDF or PostScript file to use in simulating a scan task.

9 If necessary, configure advanced settings, including any answers to prompts that should be different than the default. For more information, see "Configuring advanced properties" on page 68.

10 To create a log during the test in Log4J, select **Enable UI Logging**.

11 Click to save the test profile, and then click to start testing.

You can stop a test that is still in progress by clicking .

Configuring advanced properties

From the MFP Simulator, in the Advanced and Configuration tabs, you can configure advanced properties for the performance simulator, including custom prompt answers.

To save the test configuration after configuring advanced properties, click .

Simulator properties

Property	Description	Default
`serverAddress`	The IP address or host name of the LDD system.	None
`profileName`	The profile to run on the LDD system. **Note:** The profile name is a Device Profile that is created using the Device Policy Editor in the solution. A list of profiles are also in the **Profiles** task for the relevant device group in LMC.	None

Property	Description	Default
cancelProbability	The probability that the profile must be canceled, expressed as a decimal between **0** and **1**. In repetitive testing and testing on multiple printers, the profile is canceled for the specified percentage of the tests, with canceled tests that are randomly selected. When the application determines that a particular test must be canceled, a random prompt is selected within that test for the point of cancellation. The setting **0** specifies that no tests are canceled, and the setting **1** indicates that every test is canceled at a random prompt.	**0.0**
ui.logging	The test is logged in Log4J.	**false**

Basic prompting properties

Property	Description	Default
basic.addressRange	The IP addresses used for emulated e-Task printers. One IP address can be used to emulate multiple e-Task printers at one IP address because the emulator can use multiple HTTP connections for e-Task printers. Multiple IP addresses are specified as a range, in the ***xxx.xxx.xxx.xxx-xxx.xxx.xxx.xxx*** format, and each included IP address must be bound to a network adapter in the local computer. If the **basic.multiIP** property is set to **false**, and multiple IP addresses are specified, then only the first IP address is used.	Local IP address (set automatically when the simulator is started)
basic.scanfiles	The TIFF, JPEG, PDF, or PostScript files to use in simulating the scan task. This property can contain a comma-delimited list of multiple files that are submitted together for the scan task. **Note:** To change the delimiter used in the list, use the **scanFileDelimiter** property.	None
basic.promptAnswers	A comma-delimited list of answers for prompts that the profile asks for. If this property is empty, then the default response is used for all prompts. If you need to provide an answer for any prompt, then provide all answers for the applicable logic path. For more information, see "Supplying answers to prompts" on page 77. **Note:** To change the delimiter used in the list, use the **delimiter** property.	None
basic.repetitions	The number of test repetitions on each emulated printer. To enter discovery mode, use **0** in the following properties: • **basic.repetitions** • **basic.nummfps**	0

Property	Description	Default
`basic.nummfps`	The number of printers to emulate. To enter discovery mode, use **0** in the following properties: • `basic.repetitions` • `basic.nummfps` **Note:** We do not recommend emulating more than 250 printers.	**0**
`basic.bind`	The local IP address is bound to the print listener when receiving print jobs.	**true**
`basic.prompting`	This property indicates whether the profile contains prompts.	**true**
`basic.scanning`	This property indicates whether the profile contains a scan task.	**true**
`basic.profilerunTimeThreshold`	The time in milliseconds allowed for a profile to run before a warning message is logged.	**30000**
`basic.timeToFirstPromptThreshold`	The time in milliseconds allowed for the first prompt in the profile to show before a warning message is logged.	**2000**
`basic.multiIP`	Multiple IP addresses are used for emulating multiple e-Task printers. If set to **false**, then only the first IP address specified in the `basic.addressRange` property is used. If set to **true**, then the number of IP addresses specified in the following properties must match: • `basic.addressRange` • `basic.nummfps`	**true** **Note:** If the test is run without specifying multiple IP addresses, then this value is automatically changed to **false**.
`basic.readTimeout`	The timeout in milliseconds for reading data from the LDD system.	**360000**
`basic.connectionTimeout`	The timeout in milliseconds for connection requests to the LDD system.	**180000**
`native`	For LDD 5.x, set this property to **false**.	**false**
`basic.useRandomFileFromDirectory`	A random file is used from the directory specified in the `basic.randomFileDirectoryPath` property for simulating the scan task. If set to **true**, then the following properties override the files specified in the `basic.scanfiles` • `basic.useRandomFileFromDirectory` • `basic.randomFileDirectoryPath`	**false**

Property	Description	Default
`basic.randomFileDirectoryPath`	The directory where image files for scans are located if the **`basic.useRandomFileFromDirectory`** property is set to **true**.	None

Advanced prompting properties

Property	Description	Default
`advanced.promptAnswers`	A comma-delimited list of answers for prompts that the profile asks for. If empty, then the default response is used for all prompts. If you need to provide an answer for any prompt, then provide all answers for the applicable logic path. For more information, see "Supplying answers to prompts" on page 77. **Note:** To change the delimiter used in the list, use the **`delimiter`** property.	None
`advanced.addressRange`	The IP addresses used for emulated e-Task 2 printers. One IP address can be used to emulate multiple e-Task 2 printers at one IP address because the emulator can use multiple HTTP connections for e-Task 2 printers. Multiple IP addresses are specified as a range, in the ***xxx.xxx.xxx.xxx-xxx.xxx.xxx.xxx*** format, and each included IP address must be bound to a network adapter in the local computer. If the **`advanced.multiIP`** property is set to **true**, and multiple IP addresses are specified, then only the first IP address is used.	Local IP address (set automatically when the simulator is started)
`advanced.repetitions`	The number of test repetitions on each emulated printer. To enter discovery mode, use **0** in the following properties: • **`advanced.repetitions`** • **`advanced.nummfps`**	**0**
`advanced.nummfps`	The number of printers to emulate. To enter discovery mode, use **0** in the following properties: • **`advanced.repetitions`** • **`advanced.nummfps`** **Note:** We do not recommend emulating more than 250 printers.	**0**

Property	Description	Default
advanced.multiIP	Multiple IP addresses are used for emulating multiple e-Task 2 printers. If set to **false**, then only the first IP address specified in the **advanced.addressRange** property is used.	**true** **Note:** If the test is run without specifying multiple IP addresses, then this value is automatically changed to **false**.
advanced.profilerunTimeThreshold	The time in milliseconds allowed for a profile to run before a warning message is logged.	**30000**
advanced.timeToFirstPromptThreshold	The time in milliseconds allowed for the first prompt in the profile to show before a warning message is logged.	**2000**
advanced.useSecureWebdav	This property determines whether WebDAV communication with the LDD system must be secure. If the LDD system is v4.4.0.2 or later, then set this property to **true**. Otherwise, set it to **false**.	**true**
advanced.webdavUsername	The user name used for secure WebDAV communication with the LDD system.	**1dd**
advanced.webdavPassword	The password used for secure WebDAV communication with the LDD system.	**1dd**
advanced.useRandomFileFromDirectory	A random file is used from the directory specified in the **advanced.randomFileDirectoryPath** property for simulating scan tasks. If set to **true**, then the following properties override the files specified in the **advanced.scanfiles*N*** property: • **advanced.useRandomFileFromDirectory** • **advanced.randomFileDirectoryPath**	**false**
advanced.randomFileDirectoryPath	The directory where image files for scans are located if the **advanced.useRandomFileFromDirectory** property is set to **true**.	None

Property	Description	Default
advanced.scanfiles1– advanced.scanfiles*N*	The TIFF, JPEG, PDF, or PostScript files to use in simulating scan tasks. Each property represents a single scan task, so properties after the **advanced.scanfiles1** property are only used for profiles that contain multiple scan tasks. Each property can contain a comma-delimited list of multiple files that are submitted together for the scan task. **Note:** To change the delimiter used in the list, use the **scanFileDelimiter** property.	None

Basic prompting scan properties

Property	Description	Default
basic.1info.name	The profile name that is associated with the scan task.	**${***CurrentTestProfile***.profileName}** **Note:** This value refers to the **profileName** property.
basic.1info.resolution	Resolution for the scan task.	**300**
basic.1info.format	Image format for the scan task.	**TIFF**
basic.1info.depth	Bit depth for the scan task.	**8**
basic.1info.orientation	Orientation for the scan task.	**PORTRAIT**
basic.1info.papersize	Paper size for the scan task.	**LETTER**
basic.1info.numpages	Number of pages for the scan task.	**1**

Property	Description	Default
advanced.1info.name	The profile name that is associated with the scan task.	**${***CurrentTestProfile***.profileName}** **Note:** This value refers to the **profileName** property.
advanced.1info.resolution	Resolution for the scan task.	**300**
advanced.1info.format	Image format for the scan task.	**TIFF**
advanced.1info.depth	Bit depth for the scan task.	**8**
advanced.1info.orientation	Orientation for the scan task.	**PORTRAIT**
advanced.1info.papersize	Paper size for the scan task.	**LETTER**

Property	Description	Default
`advanced.linfo.numpages`	Number of pages for the scan task.	`1`

Property	Description	Default
`basic.pj.addressRange`	The IP address range that receives print jobs from the LDD system.	`${CurrentTestProfile.basic.addressRange}` **Note:** This value refers to the `basic.addressRange` property.
`basic.pj.output`	The folder where simulated print jobs must be saved.	The **/Profiles/x/conf/data/PrintJobs** folder where the MFP emulator and performance simulator are installed, where **x** is the name of the current test profile.
`basic.pj.ext`	The file extension of saved print jobs. Use one of the following: • `pdf` • `ps` • `tif` **Note:** Do not include a period (`.`) before the extension. The extension implies the file type used.	`ps`
`basic.pj.save`	Print jobs from the LDD system are saved.	`false`
`basic.pj.poolSize`	The size of the thread pool to use for print jobs.	`${CurrentTestProfile.serverPoolSize}`

Advanced prompting print job properties

Property	Description	Default
`advanced.pj.addressRange`	The IP address range that receives print jobs from the LDD system.	`${CurrentTestProfile.advanced.addressRange}` **Note:** This value refers to the `advanced.addressRange` property.
`advanced.pj.output`	The folder where simulated print jobs are saved.	The **/Profiles/x/conf/data/PrintJobs** folder where the MFP emulator and performance simulator are installed, where **x** is the name of the current test profile.

Property	Description	Default
advanced.pj.ext	The file extension of saved print jobs. Use one of the following: • **pdf** • **ps** • **tif** **Note:** Do not include a period (**.**) before the extension. The extension implies the file type used.	**ps**
advanced.pj.save	Print jobs from the LDD system are saved.	**false**
advanced.pj.poolSize	The size of the thread pool to use for print jobs.	**${*CurrentTestProfile*.serverPoolSize}**

Delay properties

Property	Description	Default
clientStartUpDelay	The delay in milliseconds between starting each group of emulated MFPs when testing with more than one MFP.	**500**
clientStartUpDelaySize	The number of MFPs to start after each interval that is specified in the **clientStartUpDelay** property.	**10**
startDelayMin[*]	The minimum delay in milliseconds between test repetitions.	**2000**
startDelayMax[*]	The maximum delay in milliseconds between test repetitions.	**2000**
promptDelayMin[*]	The minimum delay in milliseconds before answering prompts.	**2000**
promptDelayMax[*]	The maximum delay in milliseconds before answering prompts.	**2000**
initScanDelayMin[*]	The minimum delay in milliseconds before submitting a simulated scan.	**2000**
initScanDelayMax[*]	The maximum delay in milliseconds before submitting a simulated scan.	**2000**
betweenScanDelayMin[*]	The minimum delay in milliseconds between submitting multiple files in a simulated scan task.	**2000**
betweenScanDelayMax[*]	The maximum delay in milliseconds between submitting multiple files in a simulated scan task.	**2000**
[1] The delay properties are useful in simulating real-world delays where user input is expected when stress-testing a solution. For each delay specified, a random value is selected between the minimum and maximum values.		

Prompt answer formats and advanced emulator properties

Property	Description	Default
`delimiter`	The delimiter used between subsequent prompt answers in the following properties: • `basic.promptAnswers` • `advanced.promptAnswers` **Note:** A colon (`:`) cannot be used as the delimiter.	`,`
`scanFileDelimiter`	The delimiter used between multiple scan files in the following properties: • `basic.scanfiles` • `advanced.scanfilesN`	`,`
`defaultAnswer`	The string used in the following properties to indicate the default prompt answer: • `basic.promptAnswers` • `advanced.promptAnswers`	`[DEFAULT]`
`cancelAnswer`	The string used in the following properties to indicate that the prompt is cancelled: • `basic.promptAnswers` • `advanced.promptAnswers`	`[CANCEL]`
`randomAnswer`	The string used in the following properties to indicate that a random answer must be selected: • `basic.promptAnswers` • `advanced.promptAnswers`	`[RANDOM]`
`serverPoolSize`	The maximum number of threads in server thread pools.	`50`
`JMXHTTPAdaptorPort`	The HTTP port that an adaptor uses for JMX-based remote management.	`8090`

Basic scheduled task properties

Property	Description	Default
`basic.pauseCronExp`	A CRON expression that schedules a pause in the test.	None
basic.unpauseCronExp	A CRON expression that schedules for the test to resume after a pause has been scheduled.	None
`basic.addMfpCronExp`	A CRON expression that determines the interval before a new printer is added to the test.	None
`basic.removeMfpCronExp`	A CRON expression that determines the interval before a printer is removed from the test.	None
`basic.numScheduledMfpsToAdd`	The number of printers to add at the interval specified in the `basic.addMfpCronExp` property.	`1`
`basic.numScheduledMfpsToRemove`	The number of printers to remove at the interval specified in the `basic.removeMfpCronExp` property.	`1`

Advanced scheduled task properties

Property	Description	Default
`advanced.pauseCronExp`	A CRON expression that schedules a pause in the test.	None
`advanced.unpauseCronExp`	A CRON expression that schedules for the test to resume after a pause has been scheduled.	None
`advanced.addMfpCronExp`	A CRON expression that determines the interval before a new printer is added to the test.	None
`advanced.removeMfpCronExp`	A CRON expression that determines the interval before a printer is removed from the test.	None
`advanced.numScheduledMfpsToAdd`	The number of printers to add at the interval specified in the `advanced.addMfpCronExp` property.	1
`advanced.numScheduledMfpsToRemove`	The number of printers to remove at the interval specified in the `advanced.removeMfpCronExp` property.	1

Supplying answers to prompts

You can supply a comma-delimited list of answers to supply for prompts asked by the profile in the setting `basic.promptAnswers` (for e-Task printers) or `advanced.promptAnswers` (for e-Task 2 printers). If the setting is left blank, then the default response is used for all prompts. The delimiter for the list may be changed using the `delimiter` setting.

If you need to supply an answer for any prompt, then you must supply all answers for the applicable logic path. To specify the default answer for a prompt in the path, use `[DEFAULT]`, or the answer otherwise specified by the `defaultAnswer` setting. The logic path may be different depending on prompt answers, so you must plan answers for the specific path you want to take through the prompts. It may be helpful to test the path by first using the interactive MFP emulator.

The following table shows the valid answer values for each prompt type:

Prompt type	Valid answer values	Default value if no default is specified by the script
Array	Zero-based array index	**0**
Authentication (magnetic stripe card or RFID data)	A value, separated by a colon from a colon-delimited list of key=value pairs that simulate data from a magnetic stripe card or RFID device, using the following standard keys: **FormatCode** **Name** **Track1** **Track1Raw** **Track1Pan** **Track1AdditionalData** **Track2** **Track2Raw** **Track2Pan** **Track2AdditionalData** **Track3** **Track3Raw** **Track3Pan** **Track3AdditionalData** For example: `4444555566667777:Track2Raw=;4444555566667777=09051010000041600000?:Track1Raw=%B4444555566667777^USER/JOEQ^09051010000000000000000000000000416000000?:FormatCode=B:Track1AdditionalData=09051010000000000000000000000000416000000:Track1Pan=4444555566667777Track2AdditionalData=09051010000041600000:Name=USER/JOE Q` **Note:** Line breaks may be included in Authentication values. However, line breaks in the example are included only for clarity.	Empty string
Boolean	**true** **false**	**true**
Copy	**[DEFAULT]** only	N/A
CopyUI	**[DEFAULT]** only	N/A
Custom VLML	**[DEFAULT]** only	N/A
Email	**[DEFAULT]** only	N/A
EmailUI	**[DEFAULT]** only	N/A
Fax	**[DEFAULT]** only	N/A
FaxUI	**[DEFAULT]** only	N/A
Image boolean	**true** **false**	**true**

Prompt type	Valid answer values	Default value if no default is specified by the script
Image list	Zero-based array index	**0**
Image message	**[DEFAULT]** only	N/A
Integer	Numeric integer	**0**
List	Zero-based array index	**0**
Message	**[DEFAULT]** only	N/A
Numeric	Numeric integer	**0**
Password	Text string	Empty string
Scan	None; scan prompts are handled by the **basic.scanfiles** and **advanced.scanfilesN** settings. Answers for scan prompts should not be included in the sequence of prompt answers.	N/A
ScanUI	**[DEFAULT]** only	N/A
String	Text string	Empty string

Using JConsole to monitor and modify a running test

You can use the *Java Monitoring and Management Console* (JConsole) or another JMX-enabled application to access attributes and operations of the performance simulator process during a test. The following steps detail accessing these attributes and operations using JConsole specifically, but the listed attributes and operations are the same for other JMX-enabled applications.

1 Run JConsole.

Note: JConsole is installed with the Java SE Development Kit. The default location for JConsole is C:\Program Files\Java\jdk<*current version*>\bin\jconsole.exe.

2 While a performance simulator test is running, select **com.lexmark.workflow.simulator.App** from the list of processes on the Local tab.

Note: If a test is started after the "Connect to Agent" dialog is opened, then click the empty area within the list of processes to refresh it.

3 Click **Connect** > **MBeans** tab.

4 Expand the simulator folder, and then expand the folder for the currently running test profile.

5 For e-Task 2, expand the Advanced folder.

or

For e-Task, expand the Basic folder.

6 Select **Simulation**.

From the Attributes tab, you can see the following details of a running test:

- **ActiveMfps**—This is the number of MFPs currently being emulated for the test.
- **BetweenClientStartUpDelaySize**—This is the number of MFPs started after each interval specified by the **clientStartUpDelay** setting in the test profile.

- **Cancels**—This is the number of tests canceled, which is controlled by the **cancelProbability** setting in the test profile.
- **Failures**—This is the number of tests run that did not complete successfully.
- **Name**—This is the name of the test, which includes the test profile name; Advanced for e-Task 2 or Basic for e-Task.
- **PausedMfps**—This is the number of paused MFPs.
- **RunCount**—This is the total number of completed tests.
- **StartTime**—This is the date and time when the test was started.
- **Successes**—This is the number of successfully completed tests.

Only the BetweenClientStartUpDelaySize value can be changed from the Attributes tab.

From the Operations tab, you can perform any of the following actions by clicking the corresponding button:

- **getBetweenClientStartUpDelaySize**—Retrieve the current setting for the number of MFPs started after each interval specified by the **clientStartUpDelay** setting in the test profile.
- **setBetweenClientStartUpDelaySize**—Set the number of MFPs started after each interval specified by the **clientStartUpDelay** setting in the test profile.
- **getActiveMfps**—Retrieve the number of MFPs currently being emulated for the test.
- **getRunCount**—Retrieve the total number of completed tests.
- **getCancels**—Retrieve the number of tests canceled.
- **getSuccesses**—Retrieve the number of successfully completed tests.
- **getFailures**—Retrieve the number of tests run that did not complete successfully.
- **pause**—Pause the test.
- **getPausedMfps**—Retrieve the number of paused MFPs.
- **addMfps**—Add the number of MFPs specified for p1 to the test.
- **removeMfps**—Remove the number of MFPs specified for p1 from the test.
- **shutdown**—Stop the test.
- **getName**—Retrieve the name of the test.
- **resume**—Resume a paused test.
- **getStartTime**—Retrieve the date and time when the test was started.

Determining the single-server throughput for a solution

When building or upgrading an LDD system around a particular solution, determine the single-server throughput of the solution to help in sizing the system. Determine this value by testing the solution with the performance simulator.

For more information on system sizing using the value determined by this process, see the *Lexmark Document Distributor Administrator's Guide*.

Note: Perform this test either on a separate test LDD system, or during a time that your production system is not processing any other jobs. Make sure that the test LDD system has at least one server matching the performance of your production system.

1 On the LDD system where the solution is being tested, set all but one server offline.

   Note: If there are unequal servers in the system, then test the solution on each server.

2 In the Eclipse software, click **LDD SDK** > **MFP** > **Performance Simulator**.

   Notes:

   - In Windows 7 or later, make sure to run the Eclipse software and LDD SDK as an administrator, for the simulator and emulator to function properly.
   - If you have not installed the MFP emulator and performance simulator, then follow the instructions on the computer screen.

3 From the MFP Simulator configuration window, click **File** > **New Configuration**.

4 Configure the simulator and LDD system for the test. Make sure that you have discovered the simulator in the LDD system and have deployed the solution to the simulator.

   After the MFP simulator has been discovered, use the following settings in the MFP Simulator configuration window:

   - Configuration tab:
     – **Number of MFPs to simulate**—1
     – **Repetitions**—A number high enough to run at least an hour. The following typical throughputs on a server with the recommended hardware may help determine a rough estimate:

Solution processing load	Functions used	Average single-server throughput
Typical	• Some image-processing • Printing	6000–8000 jobs per hour
Heavy	• Extensive image-processing • Bar codes • External processes • Small to medium Document Producer (e-forms) jobs	2000–3000 jobs per hour
Very heavy (OCR)	• OCR • Large Document Producer (e-forms) jobs	100–200 jobs per hour

Note: Using less than the recommended RAM significantly reduces throughput. For example, a dual-processor server with only 2GB of RAM can process only 600–800 jobs per hour when using a "heavy" solution.

- **Scan File**—If a scan task is part of the profile, then a TIFF, JPEG, PDF, or PostScript file is required to simulate the task.

  Note: If different files must be used for separate scan tasks for an e-Task 2 solution, then use the advanced property `advanced.scanfilesN` instead.

- Advanced tab:

  - **basic.promptAnswers/advanced.promptAnswers**—A list of answers to prompts that results in the logic path with the heaviest possible processing load. You may omit this setting if the default answer to each prompt results in the same logic path.

Note: You can change any other properties that help simulate your system more accurately.

5 Click ▦ to save the test profile, and then click ▶ to start testing.

Let the test to run for an hour.

Deploying a solution directly to an LDD server system

Notes:

- Solutions can be deployed directly from the Eclipse software only when using LDD 4.3 or later.
- If necessary, update the proxy settings in the Eclipse software for communication with the LDD server system. If the LDD system is inside a proxy server in the Eclipse software, then add the load balancer host name to the "No Proxy for" list. You can update this list in the Network Connections preferences.
- If a solution performs several actions using an auto-configure script, then increase the timeout setting for exporting a solution for deployment. You can configure this setting in the Lexmark preferences.

1 In the Package Explorer pane, select the solution.

2 If Build Automatically is not enabled on the Project menu, then click **Project** > **Build Project**.

3 Click **File** > **Export**.

4 Expand the Lexmark folder, and then select **Deploy to LDD 4.3+ Server**.

5 Click **Next**.

6 If the server has not been accessed from the Eclipse software before, then do the following:

  a Select **Deploy to new LDD System**.

  b Type the host name or IP address of the load balancer.

  c Type the user name and password used to log in to LMC.

    Note: Do not type double-byte characters in the Password field. To use a password with double-byte characters, type it in a text editor, and then copy and paste it to the Password field.

  d Click **Add**.

7 Select **Deploy to existing LDD System**.

8 Select the server from the list.

9 Click **Finish**.

Restarting the Lexmark Solutions Application Server

If you install a workflow solution that includes a component, then you may need to restart the Lexmark Solutions Application Server for the solution to fully function.

Before restarting the Lexmark Solutions Application Server, set the server offline on the System tab in LMC. Make sure the server is offline before proceeding. Click **Refresh**, and then make sure **offline** is reported in the Status column for the server.

You can restart the Lexmark Solutions Application Server from the Windows Services control panel.

Note: When you restart the Lexmark Solutions Application Server service, any solution-related file in the \Lexmark\Solutions\apps\wf-ldss\ or \Lexmark\Solutions\apps\wf-ldss\solutions folders, such as a property file for a solution, is reverted to the version stored with the solution package as it was initially installed or upgraded, and any manual changes are lost.

Distributing a solution file and installing the solution using LMC

When the development system does not have access to the LDD server system, you can distribute the solution file and install the solution using LMC.

Note: The following procedure applies to LDD 4.3 or later. For information about manually installing a solution on an LDD 4.2 system, see the documentation for LDD 4.2.

1 If Build Automatically is not enabled on the Project menu, then select the solution in the Package Explorer pane, and then click **Project** > **Build Project**.

2 Distribute and install the solution file:

   a Open LMC.

   b Click the **Solutions** tab.

   c Select **All Solutions** from the Solutions list.

   d Click **Install/Upgrade**.

   e In the window that appears, type a path to a solution file, or click **Browse** to browse for the solution file.

      Note: For general distribution, the solution file is located in the \target\install folder within the main folder of the solution project. The solution file has an extension of .solution.

   f Click **Upload**.

Managing the LDD 4.x or later System List

To access the LDD 4.x or later System List window, click **LDD SDK**, and then click the System List option of LDD.

From the System List window, you can do one or more of the following:

- Add a server system.

   1 Click **New**.

   2 Type the host name or IP address of the load balancer.

   3 If necessary, change the port used to access LMC.

4 Type the user name and password used to log in to LMC.

Note: Do not type double-byte characters in the Password field. To use a password with double-byte characters, type it in a text editor, and then copy and paste it to the Password field.

5 If necessary, select the release (LDD version) of the targeted server system.

6 Click **Add**.

If authentication with the server system is successful, then the host name or IP address appears in the list.

- Edit settings for an existing system.

  1 Select a server system from the list.

  2 Edit the connection and authentication settings as necessary.

  3 Click **Update**.

- Remove an existing system.

  1 Select a server system from the list.

  2 Click **Remove**.

- Access LMC for an existing system.

  1 Select a server system from the list.

  2 Click **LMC**.

Note: If necessary, update the proxy settings in the Eclipse software for communication with an LDD server system. If the LDD system is inside a proxy server in the Eclipse software, then add the load balancer host name to the "No Proxy for" list. You can update this list in the Network Connections preferences.

Building a workflow solution from the command line

1 Create a working folder.

Example: `C:\solutionfolder\`

2 Copy the following files, located in the Eclipse \plugins folder, to the new folder:

- com.lexmark.lddsdk.core_1.0.0.*timestamp*.jar
- com.lexmark.lddsdk.sbtcore_1.0.0.*timestamp*.jar
- com.lexmark.lddsdk.sbt4*x*_1.0.0.*timestamp*.jar

timestamp represents the timestamp that appears on the latest version of each file, and *x* represents the target version of LDD.

3 Copy the Rhino JavaScript compiler, js-1.6R1.jar, to the new folder.

4 Add the working folder to the classpath.

Example: `java -classpath C:\solutionfolder\`

5 Remove the *timestamp* from the file names of com.lexmark.lddsdk.core_1.0.0.*timestamp*.jar and com.lexmark.lddsdk.sbtcore_1.0.0.*timestamp*.jar, so that the new file names are as follows:

- com.lexmark.lddsdk.core_1.0.0.jar
- com.lexmark.lddsdk.sbtcore_1.0.0.jar

6 From the working folder, execute this command, noting the replacements that follow:

```
java -jar com.lexmark.lddsdk.sbt4x_1.0.0.timestamp.jar project_container
working_folder_path/js-1.6R1.jar
```

Note: Multiple lines are shown for clarity. Type the entire command on a single line.

Make the following replacements in the command:

- **timestamp**—The timestamp contained in the file name

  Example: **com.lexmark.lddsdk.sbt44_1.0.0.v20081027‑1303.jar**

- **x**—The version number of the target version of LDD

  Example: **com.lexmark.lddsdk.sbt44_1.0.0.v20081027‑1303.jar**

- **project_container**—The full path to the container folder of the project in the workspace

  Example: **C:/workspace/TestMFP**

- **working_folder_path**—The full path of the working folder you created in <u>step 1</u>

Command example: **java -jar com.lexmark.lddsdk.sbt44_1.0.0.v20081027‑1303.jar
C:/workspace/TestMFP C:/solutionfolder/js-1.6R1.jar**

After a successful build, a new solution file can be found in the \target\install folder within the project container folder. The file extension of the solution file is .solution.

Developing components

Understanding component design

A typical component contains four classes:

- **Service interface**—This defines the contracts for the service implementation class and serves as a property type for the bean implementation class.
- **Service implementation**—This implements service interface class.
- **Bean interface**—This sets properties for the bean implementation class.
- **Bean implementation**—This implements the bean interface and calls the methods of the service implementation class.

Best practices for component design

- Implement a standard JavaScript component interface:
    - Implement the **new** operator to create instances of the component.
    - Use uppercase letters for constants.
    - Define all constants as children of the component class.
    - Implement the **setInput** method, and accept an array where appropriate.
    - Implement the **result** field, and return an array where appropriate.
- Each method is logged automatically to the confirmation page when called. Use **com.lexmark.workflow.framework.annotation.ConfirmBeanMethodLoggingOverride** to hide methods or arguments that should not appear in the confirmation page, such as passwords. To hide methods or arguments, do either of the following:
    - To prevent a method from appearing at all on the confirmation page, add **@ConfirmBeanMethodLoggingOverride(hide=true)** before the method declaration.
    - To prevent only the arguments of a method from appearing on the confirmation page, add **@ConfirmBeanMethodLoggingOverride(hideArguments=true)** before the method declaration. The arguments are replaced with the text **<hidden-args>** on the confirmation page.
- Log all activity using the Apache Software Foundation Apache log4j software framework, using the following levels:
    - **ERROR**—Fatal errors, such as system failures
    - **WARN**—Non-fatal errors, such as script exceptions
    - **INFO**—Diagnostic information, such as progress or status
    - **DEBUG**—Debugging information
- Handle script errors by throwing exceptions. Avoid catching exceptions within the component.
- Use **com.lexmark.workflow.framework.service.temporaryFile.TemporaryFileService** for output files and temporary files.
- Add a bean, **xxxVersionBean**, in applicationContext.xml to show the version number of the component in LMC.

Creating a component project

Like solutions, each LDD component is maintained as a project within the Eclipse software.

1 In the Eclipse software, click **File** > **New** > **Project**.

2 From the Lexmark folder, select the component version of your LDD solution, and then click **Next**.

3 Type a container name for the project.

The container name is the name of the component project in the Eclipse IDE software.

4 Configure the optional settings. Do one or more of the following:

- To use a component name different from the container, clear **Use Default**, and then type a component name.
- To use a base Java package name different from the default, clear **Use Default**, and then type a name for the base Java package.
- To start a component project at a version other than 1.0, type a value in the Initial Version field.
- To use a package name different from the default that is based on the solution name, clear **Use Default**, and then type a package name.

5 Select the version of the LDD system where you want to install the solution, and then click **Next**.

6 From the Available Templates list, select the appropriate template for the component you plan to develop.

7 Click **Next**.

8 Type a description for the solution and the name of the author.

9 Click **Finish**.

Understanding the contents of a component project

The following folder structure details the contents of a component project. Some folders may not appear in all components.

ProjectFolder—This is the container folder, named when creating the project, that contains all contents of the project.

docs—This contains documentation files for the component.

lib—This contains third-party .jar, .exe, and .dll files used by the component.

src—This contains the source files of the project.

conf—This contains the configuration files for the project

images—This contains images associated with the component.

java—This contains Java source files.

native—This contains any non-Java source files.

platform—This maps to the target\component\platform folder, which contains the .jar file built from the component project.

prop—This contains properties files.

test—This contains source files for unit and integration tests.

web\WEB-INF—This contains applicationContext*.xml class configuration files for Spring Framework.

> **Note:** To be recognized by Spring Framework, class configuration file names must begin with **applicationContext**. The default class configuration file is named **applicationContext-*ContainerName*.xml**.

target—This contains the compiled component after a project build, in addition to associated class files, XML descriptor files, and copies of script files from the time of the build. This folder only appears in a new project after the first build.

Note: This folder may be deleted to ensure a clean build.

component—This contains the XML descriptor of the component project.

platform—This contains the .jar file built from the component project.

install**—This contains the compiled component solution, with the extension **.solution.

Note: File and folder names within a component project cannot contain double-byte characters. However, files themselves may contain double-byte characters.

Including a component in a workflow solution project

1 Make sure the component project has been built. If Build Automatically is not enabled on the Project menu, then do the following:

a Select the component project in Package Explorer.

b Click **Project** > **Build Project**.

The src\platform folder should contain a .jar file after a successful build.

2 Determine whether to include or reference the component:

- An included component does not update with changes made to the original component project. Do the following to include a component:

  a Make sure the workflow solution project appears in Package Explorer.

  b Click and drag the .jar file from the src\platform folder of the component project to the root folder of the workflow solution project.

- A referenced component updates if the original component project is changed and rebuilt. Do the following to reference a component:

  a Right-click the workflow solution project folder in the Package Explorer pane, and then click **Properties**.

  b Select **Project References** from the list.

  c Select the check box beside the component project you want to reference in the workflow solution project.

  > **Note:** A project must appear in Package Explorer to appear in the Project References list.

  d Click **OK**.

Troubleshooting

Network connection problems occur when installing or updating plug-ins in the Eclipse software

If you connect to the Internet through a proxy server, then make sure that the proxy server connection is set up correctly in the Eclipse software. For more information, see "Installing the development environment" on page 15.

The Lexmark Solution Development perspective or other features are missing

Make sure that Java SE 6 is installed:

1 From the Windows Control Panel, double-click **Java**.

2 On the General tab, click **About**.

3 In the About Java window, make sure that the version number is at least 6.

4 If an older version is installed, then install the Java SE 6 Development Kit (JDK) and Java Runtime Environment (JRE), available from **java.sun.com**

Auto-complete does not work for LDD objects

Try one or more of the following:

- Make sure that JSEclipse is installed.
- JSEclipse requires a workspace folder name with no spaces. Make sure that your workspace folder name does not have spaces.

  1 In the Eclipse software, click **File** > **Switch Workspaces** > **Other**.

  2 In the Workspace field, type a path, or browse to a folder.

     Note: Make sure that the path has no spaces.

  3 To copy your current workbench layout and working sets, expand **Copy Settings**, and then select the options.

  4 Click **OK**.
     The Eclipse software restarts and uses the new workspace.

  5 In the Eclipse software, click **LDD SDK** > **LDD Update JSEclipse Autocomplete**.

  6 When prompted, click **Yes** to restart the Eclipse software.

- Make sure that the auto-complete libraries for JSEclipse have been updated for LDD.

  1 In the Eclipse software, click **LDD SDK** > **LDD Update JSEclipse Autocomplete**.

     Note: If JSEclipse is not installed, then the menu item is unavailable.

  2 When prompted, click **Yes** to restart the Eclipse software.

A properties file updated using revision control does not update

If a properties file is updated using revision control outside the Eclipse software, then changes may not appear in the associated editor in the Eclipse software.

To refresh the settings in a properties file in the Eclipse software, restart the software.

LDD system times out when exporting a solution

Try one or more of the following:

- If the LDD system is inside a proxy server in the Eclipse software, then add the load balancer host name to the "No Proxy for" list. You can update this list in the Network Connections preferences.
- If a solution performs several actions using an auto-configure script or the network is slow, then increase the timeout setting for exporting a solution for deployment.

    1 In the Eclipse software, click **Window** > **Preferences**.

    2 In the category list, select **Lexmark**.

    3 In the Solution Export Wait Time field, increase the value.

The MFP emulator does not run

Occasionally, a Java process associated with the MFP emulator may continue to run after the MFP Emulator window is closed, which prevents a new session from running correctly.

Restart the MFP emulator:

1 Make sure the MFP Emulator window is closed.

2 Right-click the taskbar, and then click **Task Manager**.

3 Click the **Processes** tab.

4 If java.exe appears in the list of processes, then do the following:

   a Select **java.exe**.

   b Click **End Process** > **Yes**.

5 If javaw.exe appears in the list of processes, then do the following:

   a Select **javaw.exe**.

   b Click **End Process** > **Yes**.

6 Reopen the MFP Emulator window.

The MFP emulator runs but cannot be discovered in LMC

The SNMP Windows service in Windows Server 2003 may cause a conflict with the MFP emulator. If the SNMP service is running, then the MFP emulator appears to run normally, but it cannot be discovered in LMC.

1 Close the MFP emulator.

2 Disable the SNMP service in the Services control panel.

3 Restart the MFP emulator, and then reattempt to discover the emulator in LMC.

The backspace and left arrow keys do not work when viewing LMC in the Eclipse software

When using LMC from the Web browser in the Eclipse software, the backspace and left arrow keys may stop working after the tab focus is changed.

To restore the function of the backspace and left arrow keys, do the following:

1 Click **Help** > **About Eclipse Platform** > **OK**.

2 Return to the LMC tab, and then continue editing.

Notices

Edition notice

December 2020

The following paragraph does not apply to any country where such provisions are inconsistent with local law: LEXMARK INTERNATIONAL, INC., PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in later editions. Improvements or changes in the products or the programs described may be made at any time.

References in this publication to products, programs, or services do not imply that the manufacturer intends to make these available in all countries in which it operates. Any reference to a product, program, or service is not intended to state or imply that only that product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any existing intellectual property right may be used instead. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by the manufacturer, are the user's responsibility.

For Lexmark technical support, visit **http://support.lexmark.com**.

For information on supplies and downloads, visit **www.lexmark.com**.

© **2012 Lexmark International, Inc.**

All rights reserved.

Trademarks

Lexmark and the Lexmark logo are trademarks or registered trademarks of Lexmark International, Inc. in the United States and/or other countries.

Firebird is a registered trademark of the Firebird Foundation.

Java is a registered trademark of Oracle and/or its affiliates.

All other trademarks are the property of their respective owners.

GOVERNMENT END USERS

The Software Program and any related documentation are "Commercial Items," as that term is defined in 48 C.F.R. 2.101, "Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 C.F.R. 12.212 or 48 C.F.R. 227.7202, as applicable. Consistent with 48 C.F.R. 12.212 or 48 C.F.R. 227.7202-1 through 227.7207-4, as applicable, the Commercial Computer Software and Commercial Software Documentation are licensed to the U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein.

The Apache Software License, Version 1.1

Copyright (c) 2000-2002 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3 The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

"This product includes software developed by the Apache Software Foundation (http://www.apache.org/)."

Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4 The names "Apache" and "Apache Software Foundation", "Jakarta-Oro" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5 Products derived from this software may not be called "Apache" or "Jakarta-Oro", nor may "Apache" or "Jakarta-Oro" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

==

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see **http://www.apache.org/**

Apache License Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1 Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or

otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2   Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3   Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4   Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

a   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

b   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

c (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

d (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5 Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6 Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7 Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8 Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9 Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Index