Adobe® InDesign® CS6 IDML Read Me

This document contains information about IDML resources available for Adobe InDesign CS6 including:

- A summary of IDML documentation.
- Information on the IDML schema.
- A description of the IDMLTools Java library. This library contains classes, sample code, and tools for working with IDML.

For late-breaking IDML news, see the latest version of this file on the InDesign devnet page, http://www.adobe.com/devnet/indesign.

IDML Documentation

IDML documentation is available in the InDesign CS65 products SDK. It also can be downloaded from http://www.adobe.com/devnet/indesign. Click the **Documentation** tab and scroll down to the **IDML** section.

The following IDML documentation is available:

- *Adobe InDesign IDML File Format Specification* — A formal specification of the IDML file format.
- *Adobe InDesign Markup Language (IDML) Cookbook* — A less formal, "how to" guide for working with IDML.

IDML Schema

Adobe InDesign Markup Language (IDML) Cookbook covers how to produce a RELAX NG schema and describes why this is specific to your plug-in configuration. For your convenience, a version of the schema, built from the standard set of Adobe plug-ins, is available with the IDML documentation at http://www.adobe.com/devnet/indesign. It also is available in the InDesign CS6 Products SDK at `SDK/docs/references/idml-schema.zip`.

The archive expands to the following directories:

- `idml-schema/package` — Contains the schema for IDML files.
- `idml-schema/single` — Contains the schema for snippets (IDMS), InCopy stories (ICML), and InCopy assignment files (ICMA).

IDMLTools

IDMLTools contains Java and XSLT-based tools, APIs, and samples for working with IDML. This support is implemented in several Java classes. The package can be used from the command line or within Java programs.

Windows and Mac OS shell scripts are included with each sample, to make it easy to run these classes from the command line.

Requirements

Windows

➤ Java JSE version 5 (JDK 1.5) or later.

➤ Ant — Available for download at http://ant.apache.org/.

Mac OS (Java and Ant are preinstalled on Mac OS X 10.5 and later.)

Set-up

Follow these steps:

1. Create an environment variable called `IDMLTOOLS_HOME`. This is used to set the `classpath` in `bat` and `sh` scripts. It should contain an absolute path to your `idmltools` directory.

   NOTE: Do not terminate the `IDMLTOOLS_HOME` string with a directory separator (`/` or `\`). The current release does not handle that in its formation of classpaths.

2. Add *`sdk`*`/devtools/sdktools/idmltools/bin` to your system path. This allows you to use the validate and package scripts from any directory.

On Windows, environment variables are set in the Systems Properties panel. You can get to your System Properties via the Control Panel or by right-clicking **My Computer** and selecting **Properties**. In System Properties, select the **Advanced** tab and click **Environment Variables**. Create an `IDMLTOOLS_HOME` variable containing the absolute path to the `IDMLTools` directory. Add the absolute path to `idmltools/bin` to the user `PATH` variable. (If this variable does not exist, create it.)

On Mac OS, there are several ways to add environment variables. If you are using Bash (the default terminal shell), consider adding them to your shell start-up file. Look for a file called `~/.bashrc`. If you do not have such a file, create one. Then define a variable as follows:

```
export IDMLTOOLS_HOME="/sdk/devtools/sdktools/idmltools"
export PATH="$PATH:$IDMLTOOLS_HOME/bin"
```

Change the contents to match the absolute path to your `idmltools` folder. You need to start a new shell before these changes take effect.

Source Code

All IDMLTools source code is included in the following location.

```
IDMLTOOLS/src
```

The code is divided across two packages.

➤ `com.adobe.idml` contains reusable classes.

➤ `com.adobe.idml.samples` contains code that is specific to a sample.

All classes are built into one JAR file:

```
IDMLTOOLS/jars/idmltools.jar
```

For Javadoc information on the package, uncompress *`IDMLTOOLS`*`/docs.zip`.

To rebuild the package, run `ant.bat` (Windows) or `ant` (Mac OS) in the *`IDMLTOOLS`* directory.

To create an Eclipse project from this `build.xml` file, run Eclipse and choose File > New > Project. Choose **Java Project from Existing Ant Buildfile**. Check the box for **Link to the buildfile in the file system**. This creates a new Eclipse project that works with the existing Ant build file.

Tools

IDMLTools includes two useful programs or tools. To use these tools, follow the set-up instructions above.

➤ The `package` tool allows you to compress, decompress, and inspect an IDML package. For information on exercising this class from the command line, run:

Windows:	`package.bat -h`
Mac OS:	`package.sh -h`

➤ The `validate` tool allows you to validate IDML, IDMS, ICML, and ICMA files. For information on validating, run:

Windows:	`validate.bat -h`
Mac OS:	`validate.sh -h`

Samples

IDMLTools contains the following Java and XSLT-based samples, demonstrating working with IDML. In the following table, the entries in the **samplename** column are used in place of the *`samplename`* variable in the instructions for running the samples (see "Running the Samples" on page 4").

Sample	samplename
Add Catalog Pages	`addcatalogpages`
Conditional Text	`conditionaltext`
Copy Styles	`copystyles`
ICML Builder	`icmlbuilder`
Import XML Template	`importxmltemplate`
Notes	`notes`
Page Builder	`pagebuilder`
Replace Images	`replaceimages`
Replace Story	`replacestory`

NOTE: These samples use Saxon and XSLT 1.0.

Running the Samples

To run the samples:

Windows:	`cd SDK\devtools\sdktools\idmltools\samples\samplename`
	`run.bat`

Mac OS	`cd SDK/devtools/sdktools/idmltools/samples/samplename`
	`run.sh`

To get a usage message explaining the various options:

Windows:	`cd SDK\devtools\sdktools\idmltools\samples\samplename`
	`samplename.bat -h`

Mac OS:	`cd SDK/devtools/sdktools/idmltools/samples/samplename`
	`samplename.sh -h`

Add Catalog Pages

The add-catalog-pages sample demonstrates adding generated content to an existing IDML file. The content is generated from the data in an XML input file. The sample uses XSLT templates to create new story and spread files, and it uses Java APIs to alter the `designmap.xml` file. Before executing the sample, examine the `before.idml` file.

Running this sample produces a new IDML file, `after.idml`. Compare `before.idml` and `after.idml`. Notice the two additional pages in `after.idml`.

Conditional Text

The conditional-text sample demonstrates controlling conditional text in an IDML file. Before executing the sample, examine `ConditionalText.idml`. It contains text that uses several conditions.

Running the sample turns off the Print Only condition in `ConditionalText.idml` and writes the results to the `ConditionalText-PrintOnlyOff.idml` file. It then turns the condition back on in `ConditionalText-PrintOnlyOff.idml` and saves it to a file called `ConditionalText-PrintOnlyOn.idml`.

Copy Styles

The copy-styles sample demonstrates copying paragraph styles from one IDML document to another using Java and XSLT. Before executing the sample, examine the set of documents that start with `From`; these are the documents from which the styles will be copied. Next, examine the set of documents starting with `To`; these are the document to which the styles will be copied and written to a new file.

Running this sample produces several files that start with `Copy`. These are the destination files, with the styles copied from one document into another.

ICML Builder

The ICML-builder sample transforms a relatively simple XHTML (story.html) file into ICML.

The transformation supports headline, tables, emphasis, ordered lists, and unordered lists.

Running this sample produces the file `story.icml`. To see the file in use, open an InDesign document and place the file into an open spread.

It is possible to run this sample with other input files and style sheets. It is essentially a single file transformation. The one unique thing is that it passes a default table width to the style sheet. This is because InDesign table columns require a width. (Their width is not determined by content.)

To run this sample with other input, use the `icmlbuilder.bat` (Windows) or `icmlbuilder.sh` (Mac OS) scripts.

Import XML Template

The import-XML-template sample emulates the XML import functionality in InDesign. It demonstrates how XSLT can be used to transform tagged content. The IDML Package produced by this sample is identical to the file that would be produced if this file were imported into InDesign.

This sample supports multiple text and image changes in a one- or two-tier XML structure. The XML file that is imported was created by opening the structure view in InDesign and exporting the root to XML. The XML file is then modified outside of InDesign and reimported. When this file is reimported, the modifications made to the XML file are applied to the IDML file.

Running this sample produces the file `WomensShirts.idml`. This file is the result of importing the `Import.xml` file into the `MensShirts.idml` file.

To run this sample with other input, use the `importxmltemplate.bat` (Windows) or `importxmltemplate.sh` (Mac OS) scripts.

Notes

The notes sample demonstrates how notes can be extracted or removed from an IDML package. XPath expressions are used to extract notes, and XSLT is used to remove notes by transforming the XML in the package.

When notes are extracted from a package, they are copied from that package to a text file. When notes are removed from a package, a copy of that package is created with all notes removed. The original file is never modified.

The extra notes sample creates the `SampleNote_Notes.txt` file, which contains all notes found in the package.

The remove notes sample creates the `SampleNote_NotesRemoved.idml` file, which is a copy of the `SampleNote.idml` with the notes removed.

To run this sample with other input, use the `notes.bat` (Windows) or `notes.sh` (Mac OS) scripts.

Page Builder

The page-builder sample demonstrates transforming a simple XML input into an IDML package, using a fairly intelligent set of style sheets. These style sheets contain a combination of hard-coded IDML and XSLT constructs that react to the input.

The page-builder stylesheet and XML input can be used to build an IDML document containing a variable number of spreads. These spreads can contain a variable number of pages, with variable binding locations. Furthermore, any number of JPG images and text frames (single or linked) can be added to the pages of the document, using page-based coordinates.

Running this sample creates a file called `pagebuilder.idml` in the working directory.

To run this sample with other input, use the `pagebuilder.bat` (Windows) or `pagebuilder.sh` (Mac OS) scripts.

Replace Images

The replace-images sample shows how both embedded and linked images can be replaced with other linked images by changing the XML files in the IDML package. XSLT is used for the XML file transformations.

This sample shows two unique approaches to replacing images:

➤ Export all images and their links to an XML file which can be manually updated and reimported.

➤ Use the Java Swing user interface to select images for replacement.

To run this sample with other input, use the `replaceimages.bat` (Windows) or `replaceimages.sh` (Mac OS) scripts.

Replace Story

The replace-story sample demonstrates how to extract and replace stories in an IDML file. It assumes that any resources (styles, swatches, etc.) used in the story are available in the destination document.

Running this sample extracts all stories in `ReplaceStory2.idml` to a directory called `temp`. It then replaces a story in `ReplaceStory.idml` with one of the stories in `temp`.

You can use the sample to extract and replace stories in other files. To do so, use the `replacestory.bat` (Windows) or `replacestory.sh` (Mac OS) scripts.

The replace-story sample specifies stories by their Self attribute. To get the Self attributes of stories in a package, use the `package` tool and the `-i` (inspect) option:

Windows:	`package.bat -i ReplaceStory.idml`
Mac OS:	`package.sh -i ReplaceStory.idml`