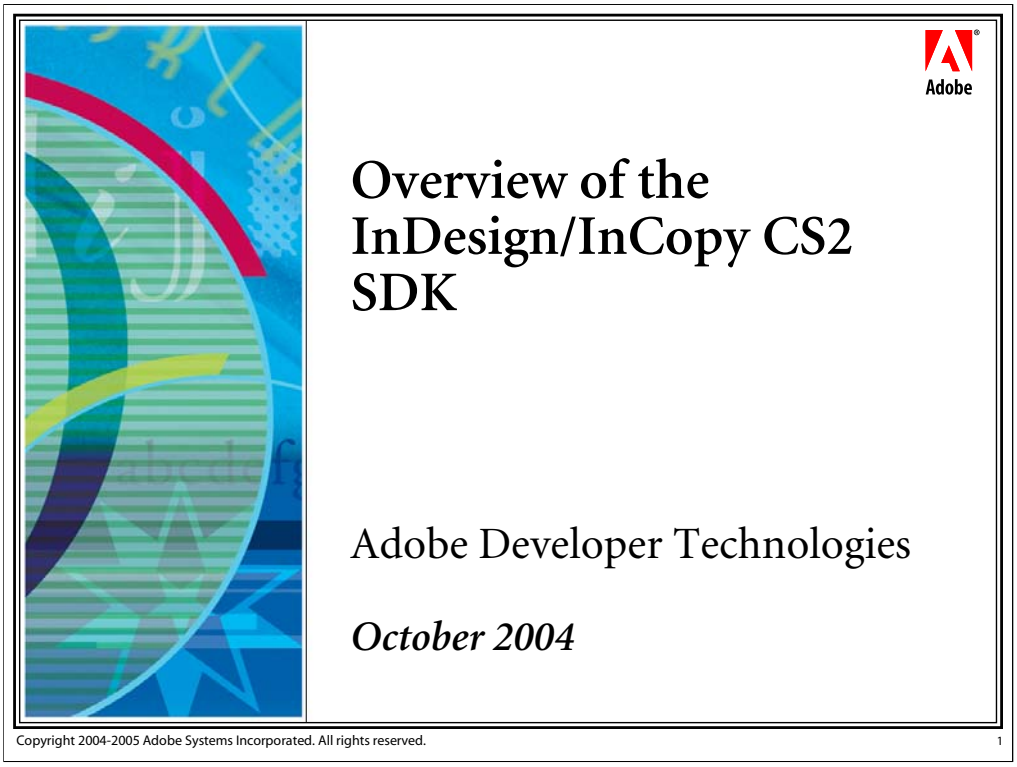
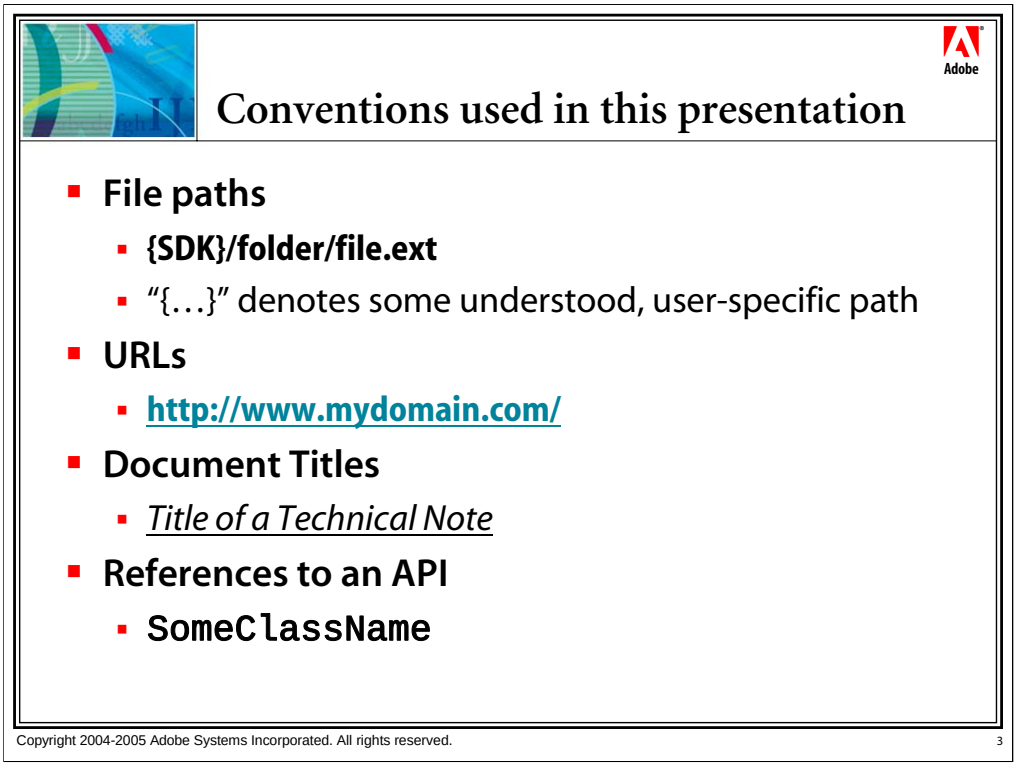


Goals of the training sessions

- To give an overview of the InDesign CS2/InCopy CS2 SDK.
- To outline some of the new API areas in this version (CS2).
- To briefly discuss porting a plug-in from CS to CS2.



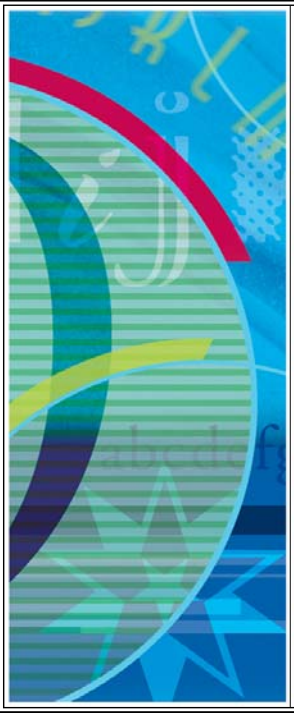
conventions used in the slides.

•File paths are shown in a bold condensed font. {…} denotes some understood path or URL that is user-specific, and is meant to be replaced with the path that applies to your system or context.

•URLs will be in blue.

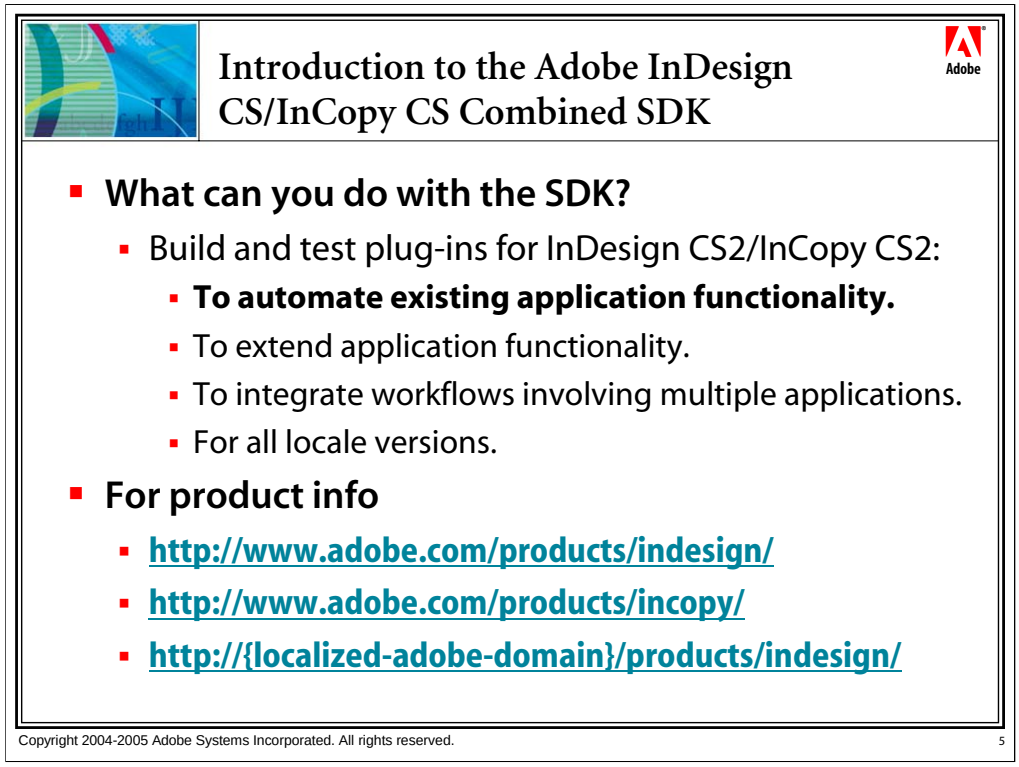
•Document titles will be underlined and italicized.

•References to an API will be in a monospace font.



The InDesign CS2/InCopy CS2 Combined Software Development Kit

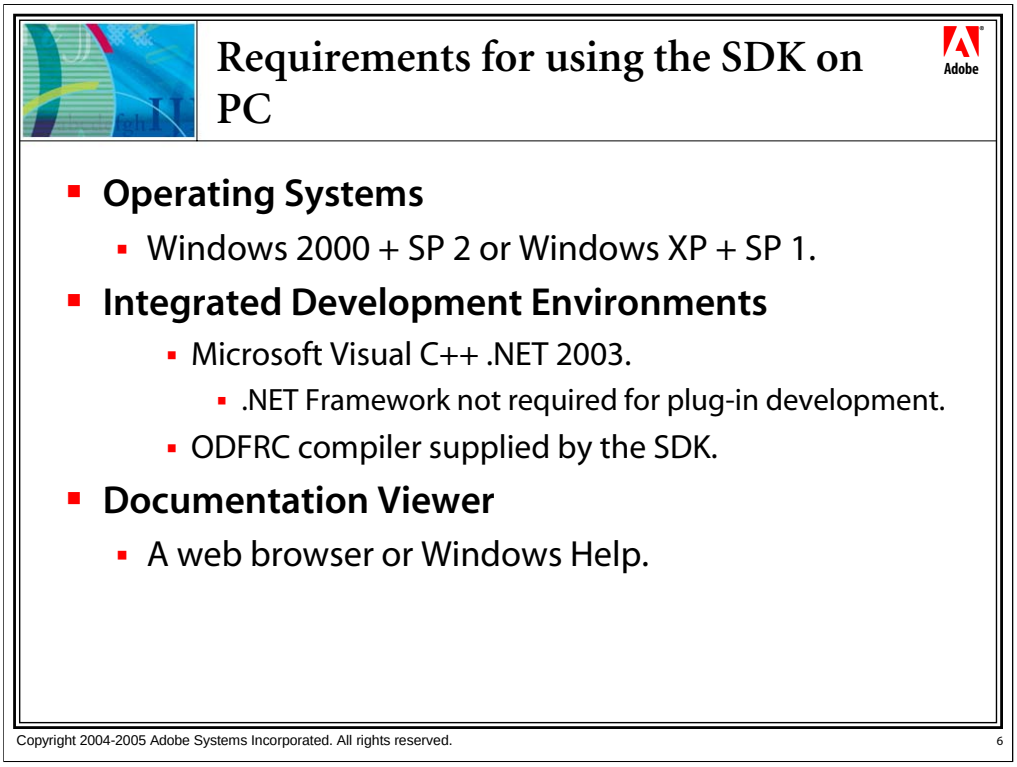
4



Note that you can use the same SDK to develop plug-ins for all locales. Adobe directly supports InDesign/InCopy CS2 in 13 locales. Middle Eastern and Central European locale versions are also available.

For more product info, please refer to the URLs shown on the slide.

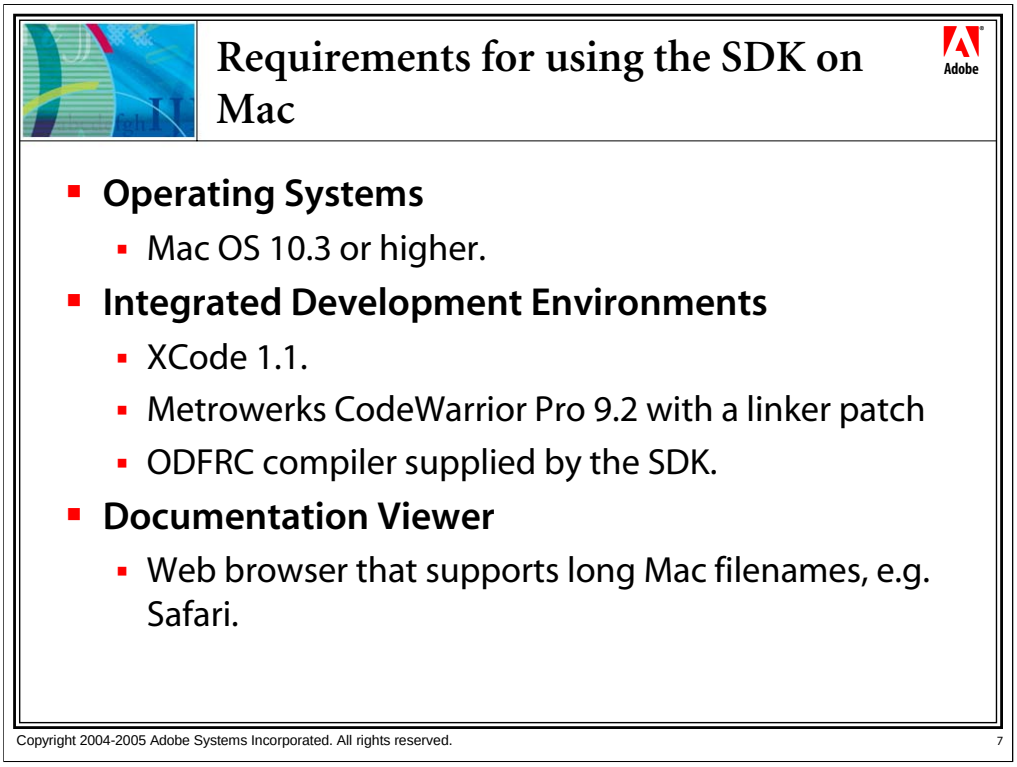
For localized product info, in most cases, you can use the same URL sub-path after the domain. For instance, if you want to look for the Japanese version of InDesign, goto http://www.adobe.co.jp/**products/indesign**/, and for the Italian version, goto http://www.adobe.it/**products/indesign**/. Note the "/**products/indesign**/" part of the URL is common.



The operating systems requirements for developing with the SDK are identical to the products themselves.

The Integrated Development Environments (IDEs) required for development are shown. In general, we recommend that you use these IDEs.

•For Windows, we require that you use the mentioned IDE. In addition, the Java 2 Runtime and Visual Basic 6 Runtime engines are required for some tools.



For more information, please read the porting guide.

How to obtain the SDK

- The pre-release SDK is only available to partners who have been invited into the developer pre-release program.
- SDK can be downloaded from: https://us.prerelease.adobe.com
- If you have any questions or issues, contact:
  - Mark Niemann-Ross (mnr@adobe.com)
- The User Forum is the place to discuss the pre-release SDK issues.

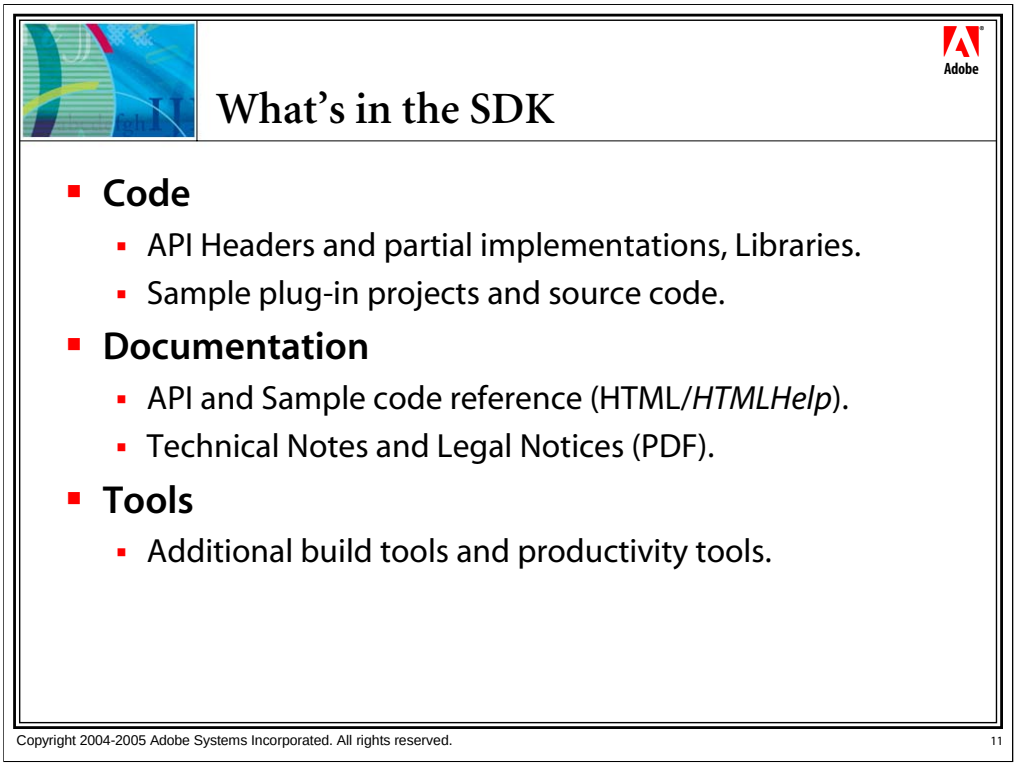
Installing the SDK

- Run the installer from the download
  - Mac: **Install Adobe InDesign CSx SDK**
  - Win: **Install Adobe InDesign CS SDK.exe**

9

What's in the SDK

- **CS2 SDK folder structure is almost identical as the CS SDK except:**
  - The Macintosh SDK frameworks/libraries are moved back in a packagefolder.
    - in CS: libraries were in **{SDK}/build/mac/libd|libr.**
    - In CS2: libraries and frameworks are in **{SDK}/build/mac/debug/packagefolder.**
- **Identical footprint between Mac and Win except a couple of platform specific tools.**



The SDK contains three major components.

•Code: comes in two forms:

- •API headers and partial class implementations you compile with, as well as libraries you link with
- •Sample plug-in projects and source code you can use to learn about the API

•Documentation: also comes in two types:

- •API and sample code reference in HTML and HTMLHelp format (for Windows only). The location of these files and instructions for decompressing the HTML tar-ball {SDK}/docs/references/sdkdocs.tar.gz is shown in the {SDK}/Readme.txt and ReadMeJ.txt files. There is also a text file that describes how to turn the HTML-based documentation into an Apple Help bundle for use on the Macintosh.
- •Technical notes and legal notices in PDF. We assume you have at least the free Adobe Reader.

•Tools for helping you develop with the SDK:

- •IDE plug-ins such as ODFRC and build tools.
- •Productivity tools such as GrayGoose and DollyXs.

What's in the SDK: Development Productivity Tools

- **DollyXs**
  - Revamped cross-platform project generator.
- **Graygoose**
  - Mac project converter that convert a CS (CW 8.3 based, PEF format) project mcp to a CW 9.2 project that will build plug-in as package bundle in Mach-O format.
- **DebugWindow (Mac only)**
  - For viewing **TRACE** outputs with Debug build.

What's in the SDK: Documentation

- **Reference Documentation: {SDK}/docs/references**
  - About API classes.
  - About "Boss classes".
  - About SDK Samples.
  - Overview (**{SDK}/docs/references/index.html**).
    - Explode **{SDK}/docs/references/sdkdocs.tar.gz** first.
- **Technical Notes: {SDK}/docs/guides**
  - More in-depth coverage about specific topics.
  - See **{SDK}/Readme.txt** for an overview.

13

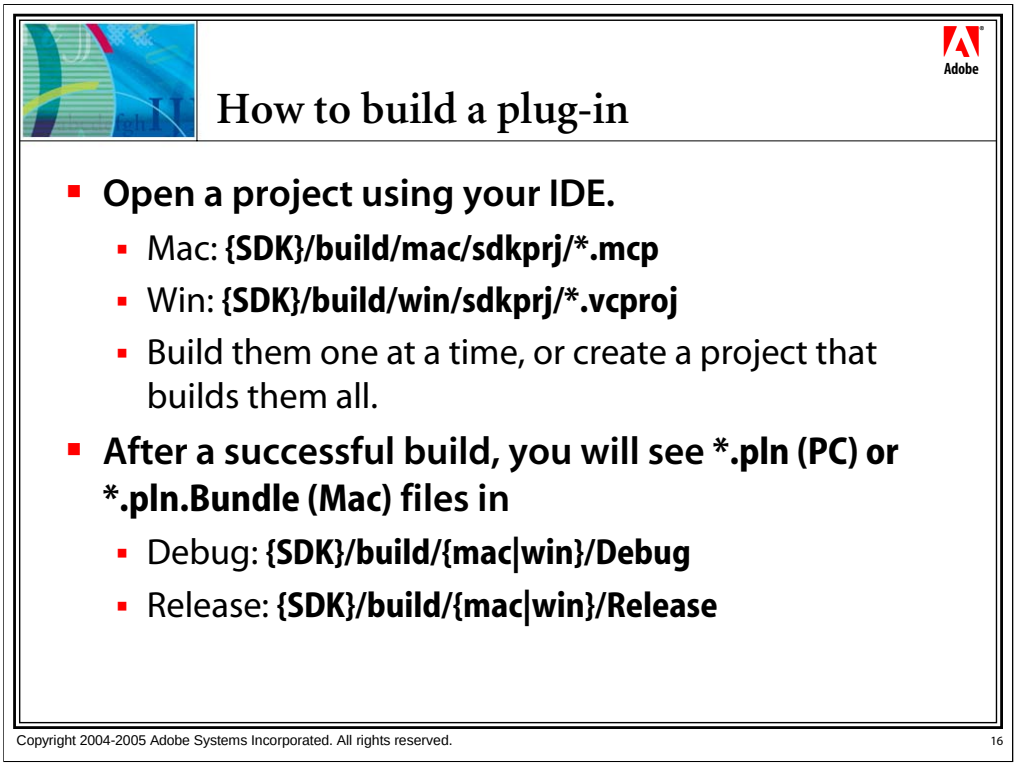
How do we get started?

- **The best place to start is the *Porting Guide***
  - **{SDK}/docs/guides/portingguide.pdf**
  - How to set up your IDEs.
  - On the Mac: How to create a plug-in project from scratch, or how to convert InDesign/InCopy CS plug-in projects for use with InDesign CS/InCopy CS2.
  - On the PC: The CS guideline is still true for this version.

Preparing the IDEs

- **Win: Visual C++ .NET 2003**
  - See *Porting Guide*:
    - "How do I set up Visual C++ .NET 2003 to access tools required to build SDK plug-ins?"
- **Mac: CodeWarrior**
  - See *Porting Guide*:
    - "How do I set up Metrowerks CodeWarrior 9.2 to compile plug-ins?"
    - "How do I set up CodeWarrior to access tools required to build SDK plug-ins?"

15



Once you get the IDE setup properly (as per previous slide), building a sample plug-in is as easy as loading a project in your IDE and selecting the Build command!

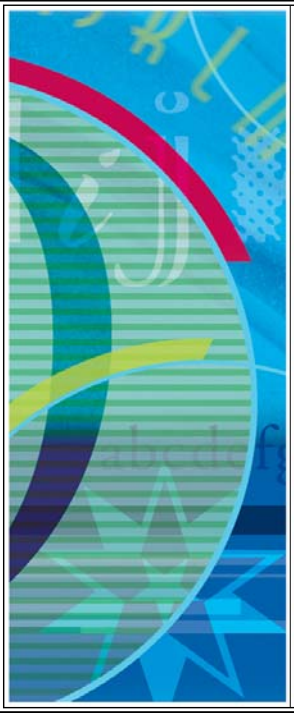
•VC++ .NET 2003:

- •Select the Active Solution Configuration ("Build >> Configuration Manager" menu, then select the "Active Solution Configuration" pulldown) as either Debug or Release
- •Select the "Build >> Build {ProjectName}" menu to build.

•CodeWarrior

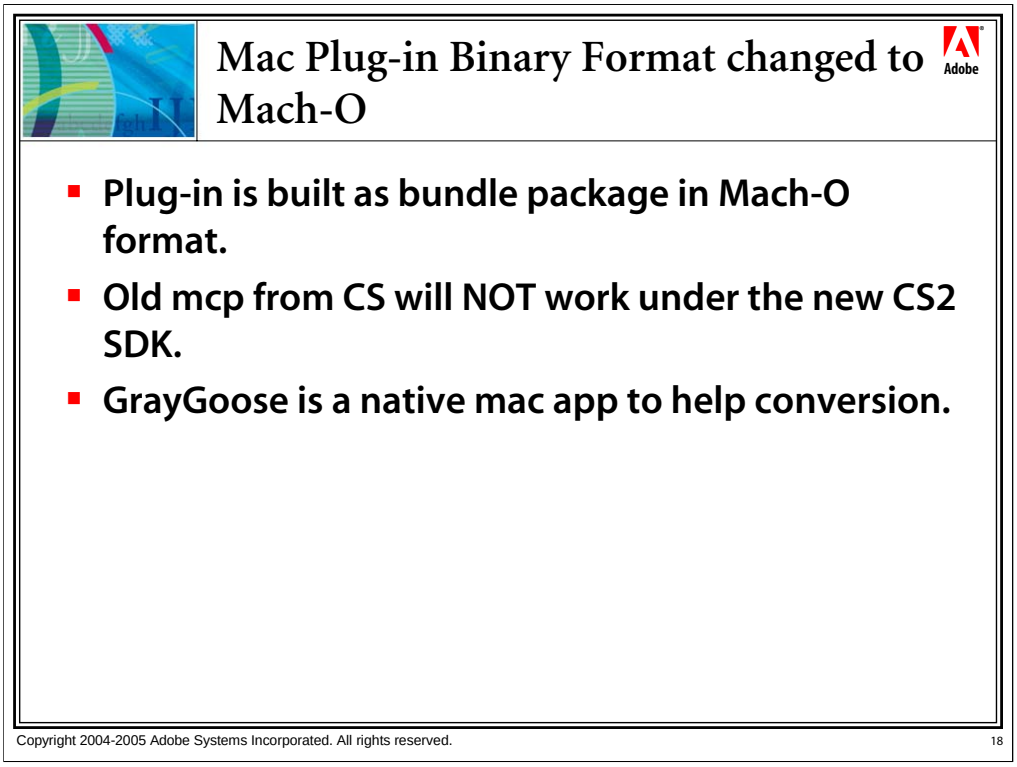
- •Select the build target in the Project window
- •Click on the Make button in the Project window, or press Command-M

After the project completes building, you will see a plug-in file (and possibly debug symbols as well), in the folders mentioned on the slide.



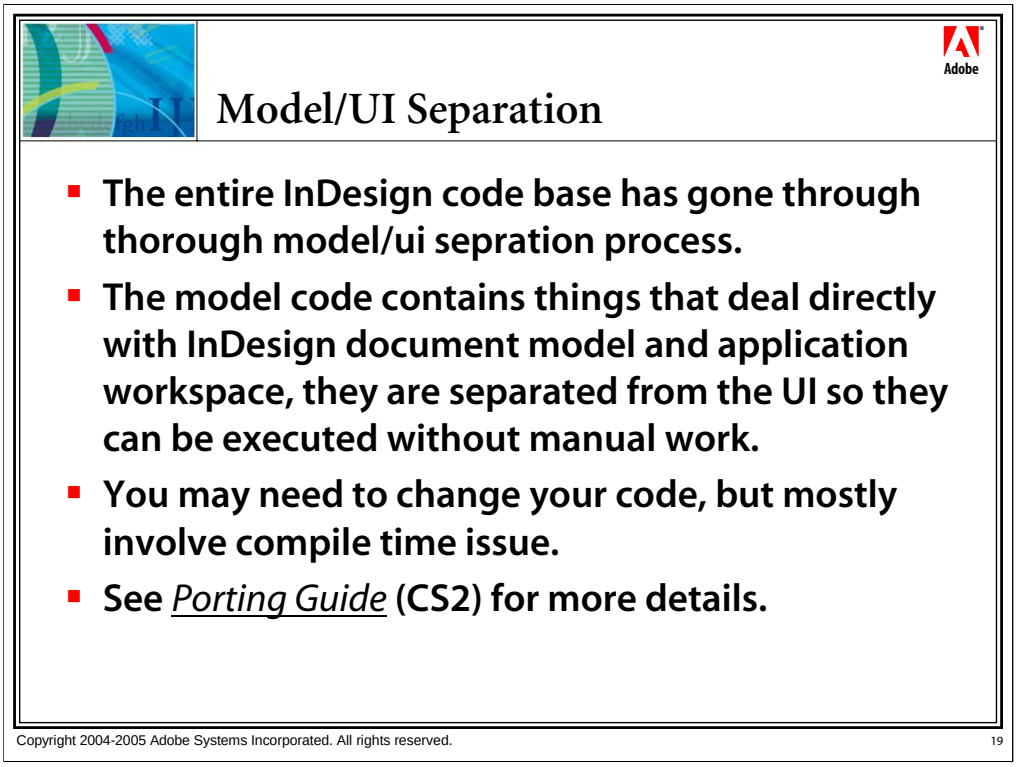
What's New in InDesign CS2/InCopy CS2

17



Note that in CS, plug-in was built as shared library in PEF format.

In the alpha SDK, plug-in was output as framework bundle in Mach-O format, if you've already started your development since alpha SDK, you should be aware that we switched to bundle package now in beta.



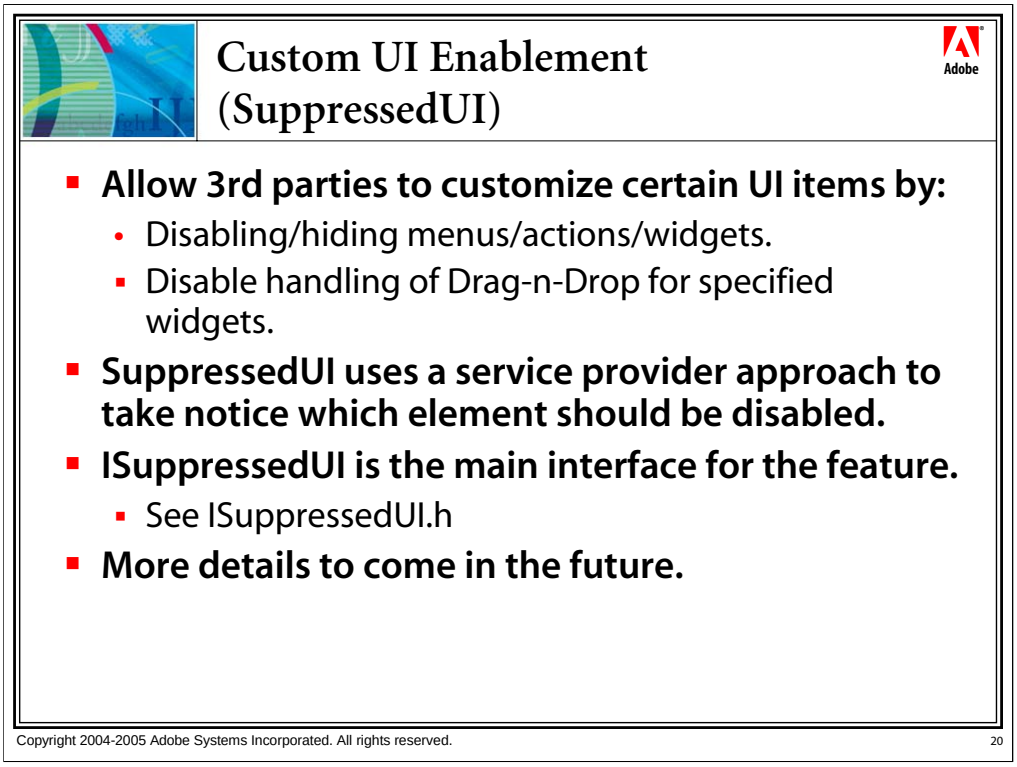
Here are a few guidelines regarding model/ui separation :

1. Interfaces with model and UI methods were refactored to separate interfaces. An example is ILayoutUtils, which had both model and UI methods in InDesign CS, but now it is split into 2 interfaces, ILayoutUtils and ILayoutUIUtils.

2. As a result of some plug-ins being split between model and UI, their associated ID headers are separated according to the nature of the IDs too. For example, the old LayoutID.h now is split into LayoutID.h and LayoutUIID.h. Some of the UI related boss, interface, and implementation IDs, such as kLayoutWindowBoss, are moved to the LayoutUIID.h.

3. Global function QueryPreferences has new signatures.

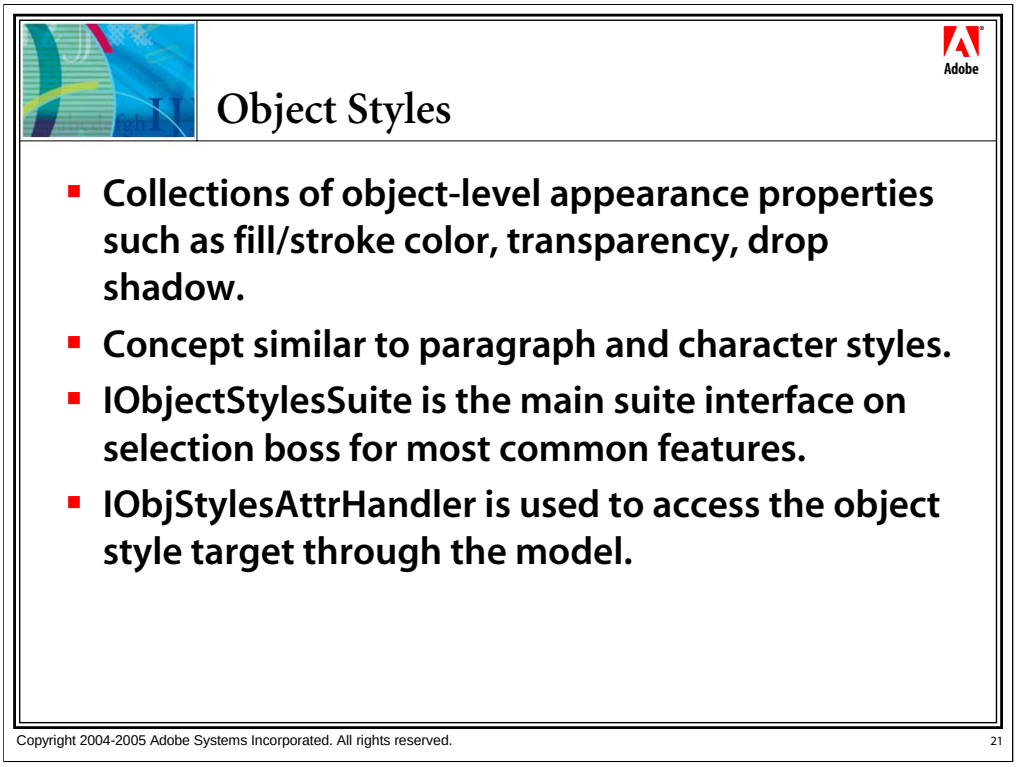
4. If you are planning on developing plug-ins for InDesign Server CS, you may refer to a technote titled "InDesign Server Plug-in Techniques" for more details on how to separate your plug-ins that mix model and UI components into separate plug-ins.



There are 2 provided ways for suppressing UI elements.

- The first way is to provide an implementation of the ISuppressedUI interface in a service provider boss. There can be several service provider bosses of this type that have ISuppressedUI implementations, each having the chance to suppress every widget, menu, etc. If any of the suppressedUI services respond that they would like to suppress a given UI element, then that UI element will be suppressed.

- The second way is using a default XML based ISuppressedUI implementation that will read in an XML file you provide, this XML will provide information on what UI elements to disable. A default XML based implementation is provided, but you still have to create the service provider boss. In this boss, you will also aggregate an ISysFileData interface that contains the path to the XML file.



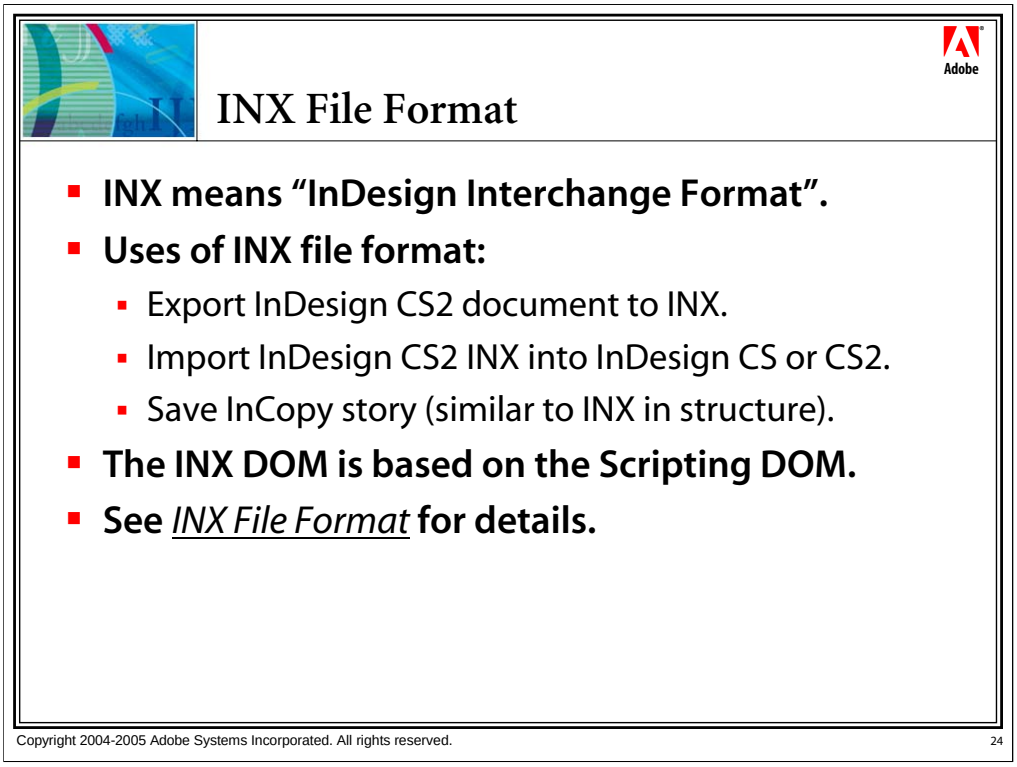
Editing a object style updates the objects with that style applied. Style management tasks such as redefining, synchronizing, loading, deleting are supported.

XML Re-architecture

- The XML import architecture has been made more extensible.
- "XML Snippets" allows logical subsections of a document to be exported to XML and imported back in.
- Tagging of tables, table cells, and content within table cells.
- XML linking to keep track of imported XML files.
- Many more features, see more details in *XML Fundamentals*.

Shared Resources

- Save/Load application preferences to an external file.
- ISnippetExport::ExportAppPrefs will export application preferences to a stream.
- ISnippetImport::ImportFromStream will import the preferences.
- You can specify which elements you want exported.
- IAppPrefsImportDelegate, and IAppPrefsExportDelegate provide customization for the import/export.

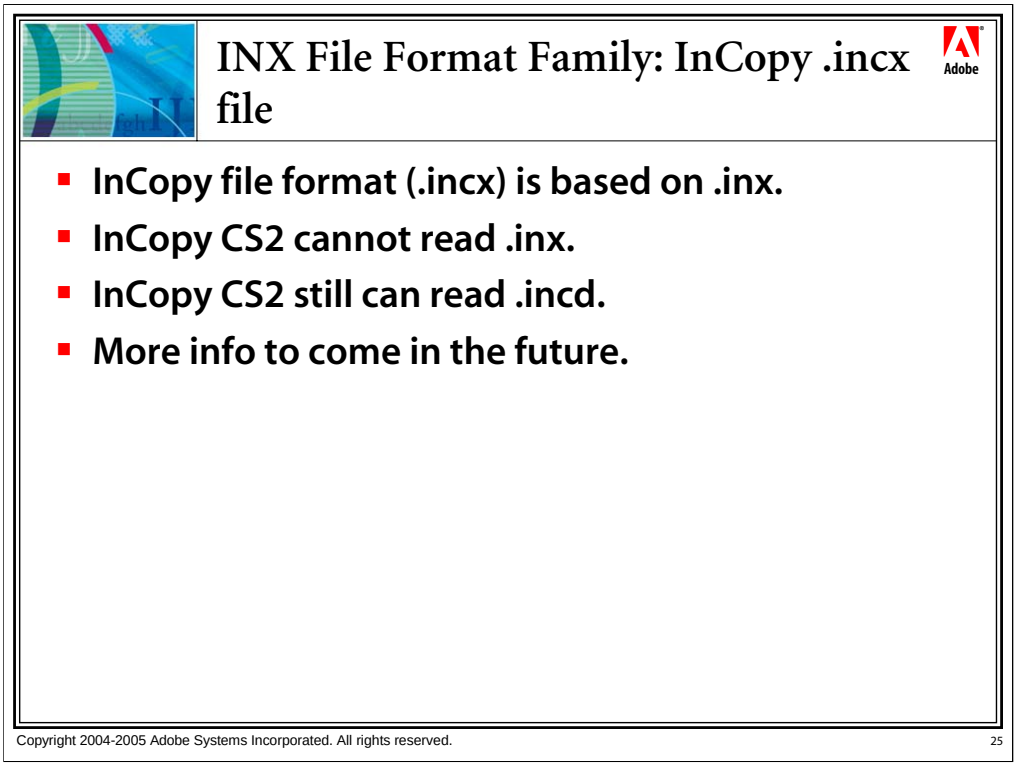


Every InDesign document is described by a tree structure of Scripting objects, rooted at a document object. For example, the document will have several children that are color objects, spreads, and so on. Besides having children, each object can have a set of properties. A color will have properties that describe its name, color model, and color values (among other things). When we export an InDesign document to an INX file, we translate the value of every object and property into XML. When we import an INX file, we reverse the process.

The important things to understand about INX are:

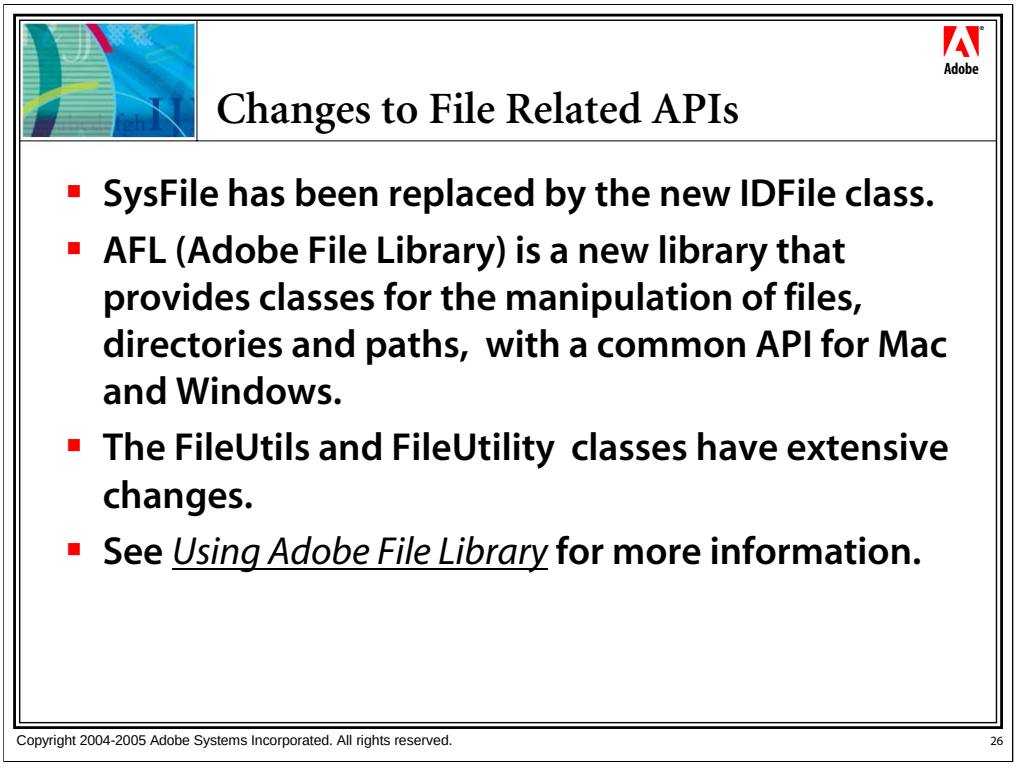
1. We create an entirely new InDesign document from a "foreign" file format (the INX). Because of differences between InDesign releases, the resulting document may not look exactly like the original. For example, if the composition algorithm changed, there could be slight differences in how text is laid out in a text frame.

2. The INX file might contain objects or object properties that did not exist in an earlier version of InDesign. These objects and properties will be ignored during import.



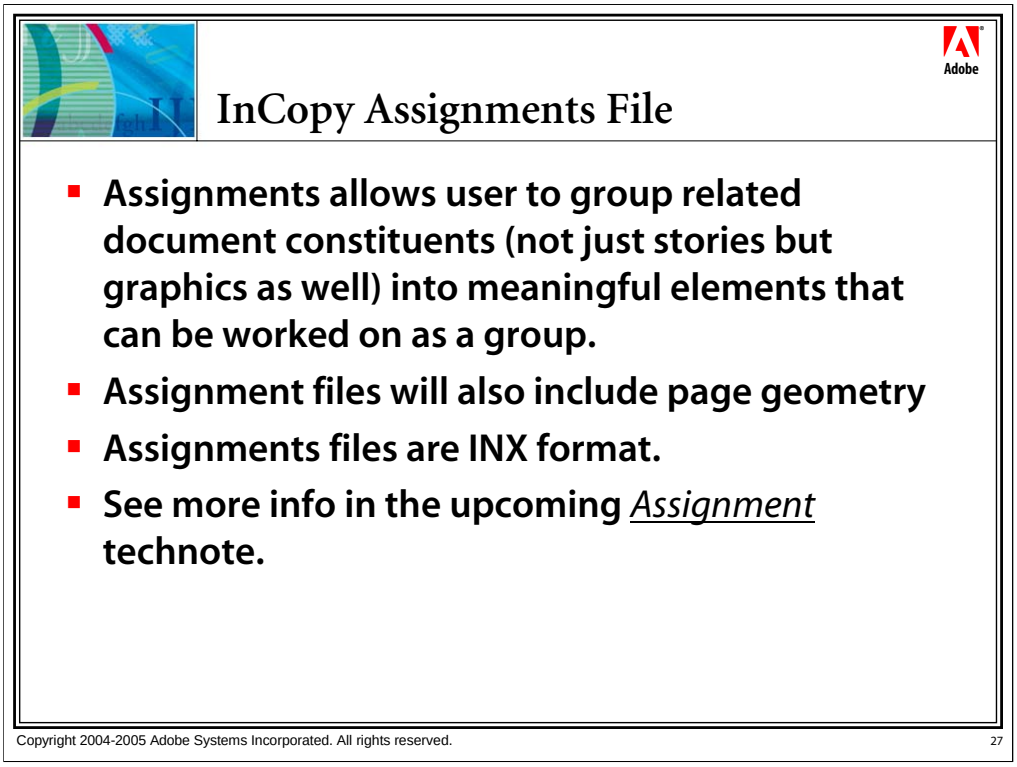
When you save a InCopy story in InCopy CS2, the save dialog defaults to a new file format called “InCopy Interchange”.

Note that the old InCopy CS story file format is still supported.



There are a few things to know about the new file library:

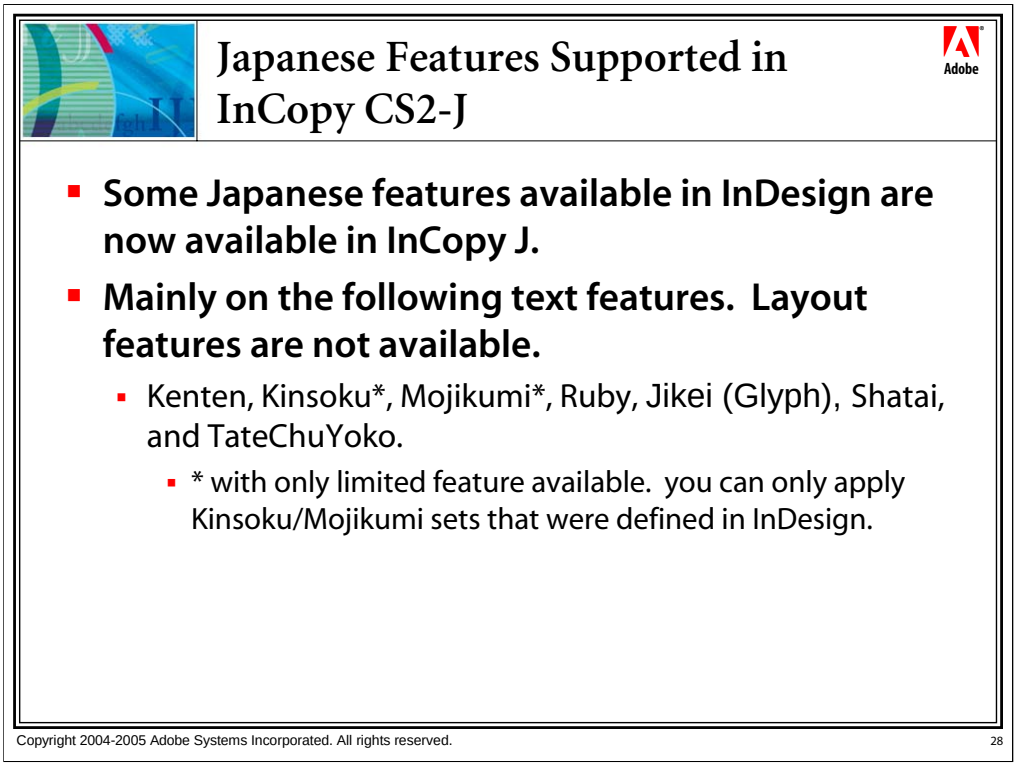
1. AFL is a new library that provides classes for the manipulation of files, directories and paths, with a common API for Mac and Windows. In addition, the library has utility classes that support file, path and OS specific operations.
2. SysFile is gone from the code base, replaced by the new IDFile class.
3. The IDFile class subclasses from the AFL's AFile class, and supports file and directory creation, deletion, and the retrieval and setting of system attributes.
4. The FileUtils and FileUtility classes have extensive changes. However, the behavior of the existing methods in both classes has not changed.
5. The rule is never to treat a SysFile or IDFile as a PMString. That assumption was blindly made in some Windows-specific code in the InDesign CS SDK, but has been corrected.



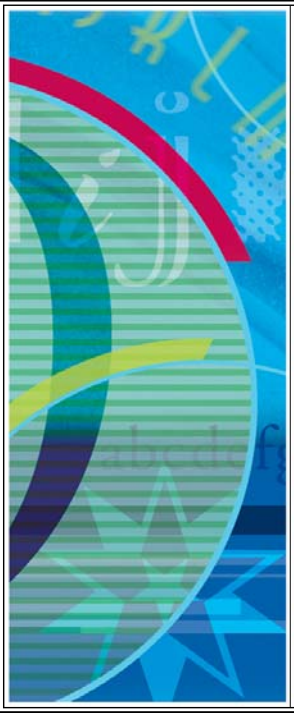
Assignment files will include page geometry, so InCopy users can see the content and layout of the frame they are editing, without impacting the time it takes to open an InDesign file.

When exporting a group of items from InDesign, an additional file—the assignment file—will be created that contains links or pointers to the grouped page elements and any transforms on those elements. This lets the user open a single file in InCopy and have editorial access to multiple stories.

Any stories in an Assignment are exported as InCopy files (INX format.) Geometry information and the relationship of the files are held in the Assignment File (also an INX format file.)



These features will only be displayed in the Layout view. Galley/Story views will be similar to the InDesign Story Editor.



Porting Overview

Compiler Settings

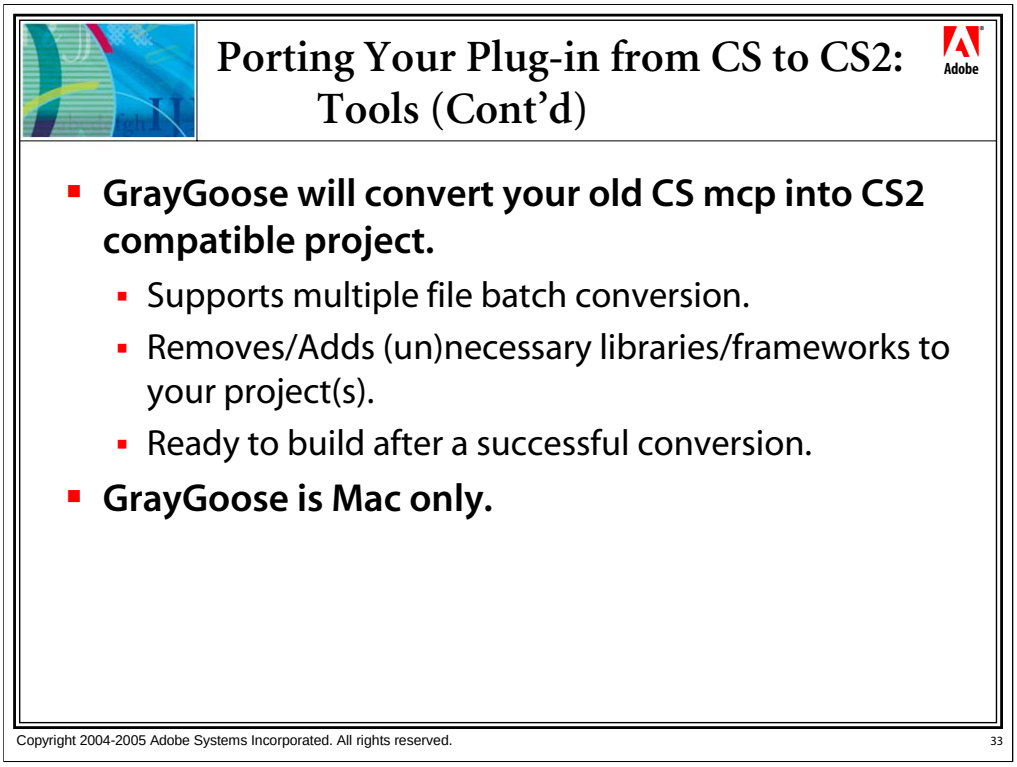
- **On PC, your CS vcproj file should work without modifications.**
- **On Mac, project settings need to be modified to accommodate the binary format and plug-in package format changes.**
  - Some old public libraries are gone, and new frameworks need to be added.
  - Should use Mach-O linker now.
  - Plug-in should be built as bundle package.
- **Porting Guide has detailed compiler settings information.**

Porting Your Plug-in from CS to CS2

1. Backup your old CS version project.
2. Duplicate your project (for Windows, you will use Graygoose to convert a project) and sources in the new SDK.
3. On the Mac, convert the mcp file into the new SDK sdkprj folder using tools such as DollyXs or GrayGoose. Make sure all the project settings are set according to the Porting Guide settings.
4. Build your project, fix any compiler errors.
5. Test your plug-ins in the debug build to see if there are any run time issues.

Porting Your Plug-in from CS to CS2: Tools

- **DollyXs is the project generation tool.**
  - It generates CS2 SDK compliant projects and associated template-based source files.
  - Revised in CS2 to include more features.
    - New UI.
    - It allows you to create project files and source where you specify, not necessarily in the SDK folder.



Since you do not need to convert your Visual Studio .vcproj files from CS to CS2, Snowgoose is not needed and Graygoose is for the Mac only.

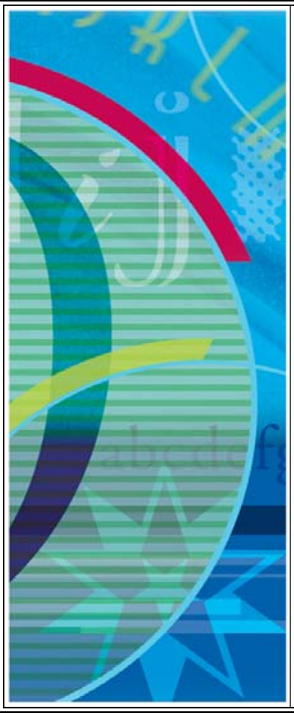
There are a few precautions to be aware of when using Graygoose, so make sure you read the help file before you process a project.

Porting Your Plug-in from CS to CS2: Porting Issues

- **The changes that took place when porting SDK sample code have been captured in the Porting Guide.**
- **When you see compiler errors:**
  - Check out the Troubleshooting section first.
  - Then check out the Porting Recipes.
- **We will take a look at some porting examples in another training session.**

Summary

- **In this session, we have learned:**
  - What's available in the InDesign CS2/InCopy CS2 Combined SDK.
  - Mac project file setting has changed quite a bit.
    - Use the tools provided by SDK to make sure you have correct settings.
  - Some new API areas that you should be aware of.
  - Basic porting strategy.



Adobe

Tools for the New Work™

36