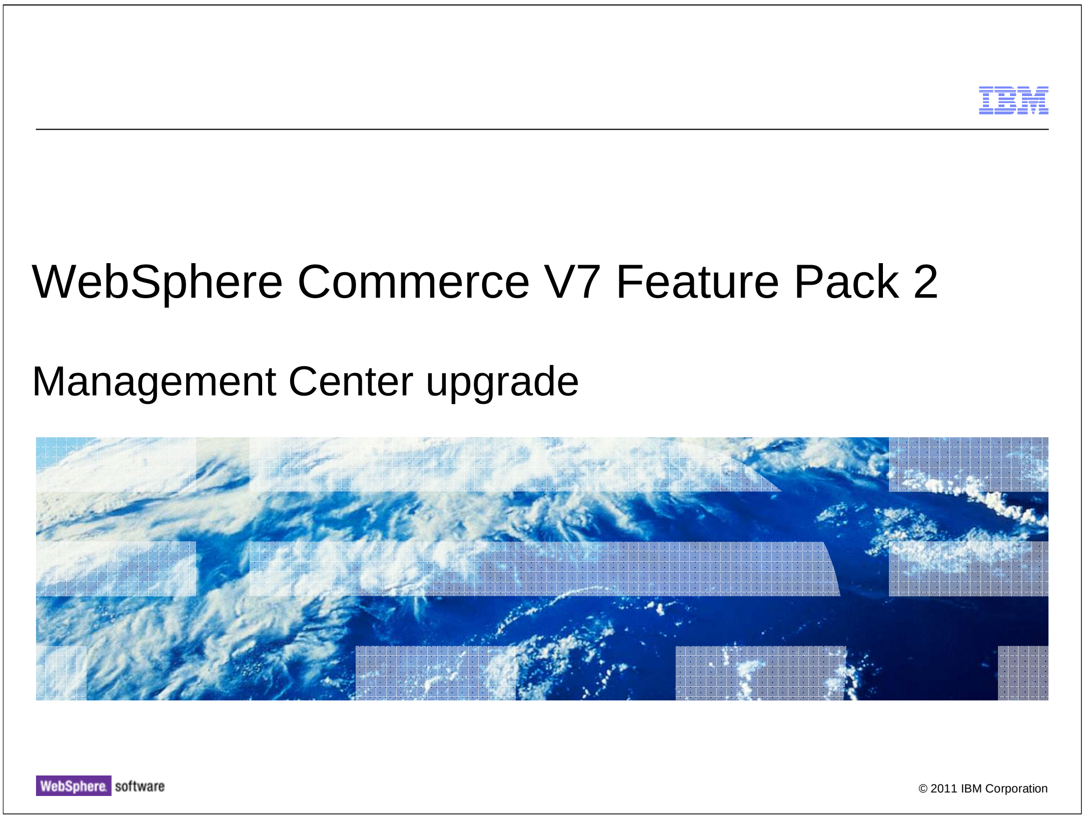


WebSphere Commerce V7 Feature Pack 2

Management Center upgrade

This presentation provides an overview of the changes to the Management Center framework in feature pack 2. You should be familiar with the existing Mangement Center framework before viewing this presentation.

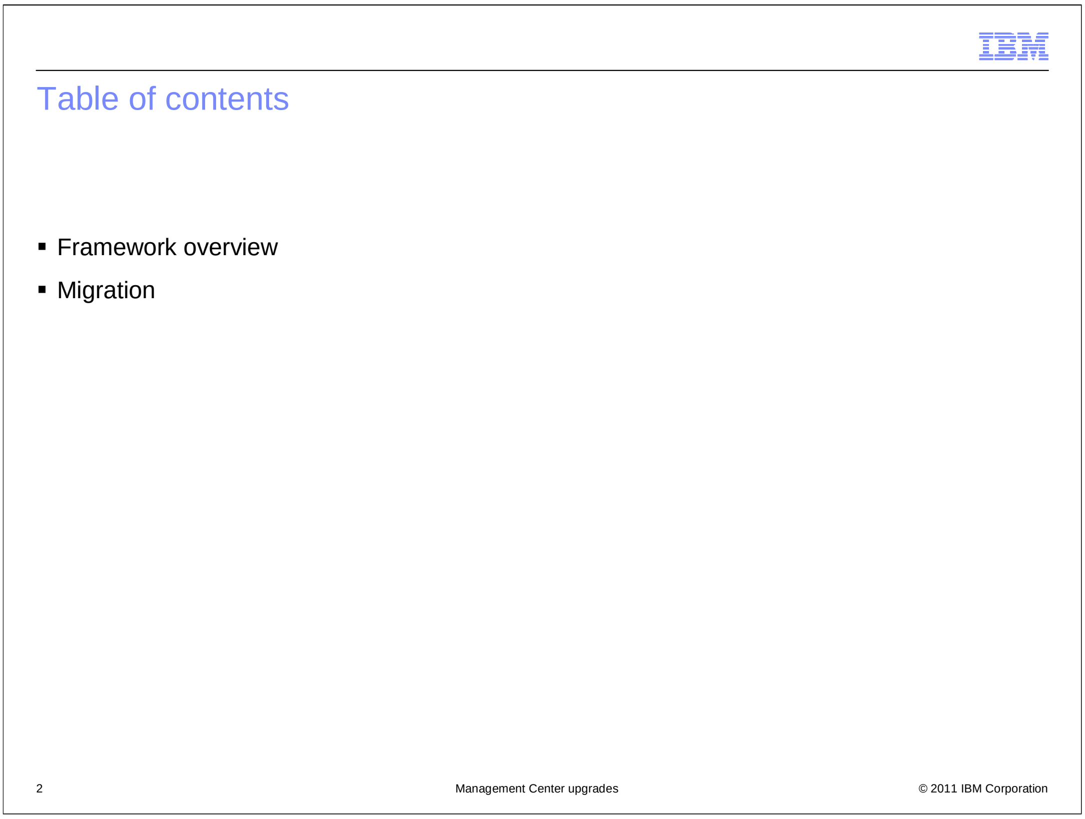


Table of contents

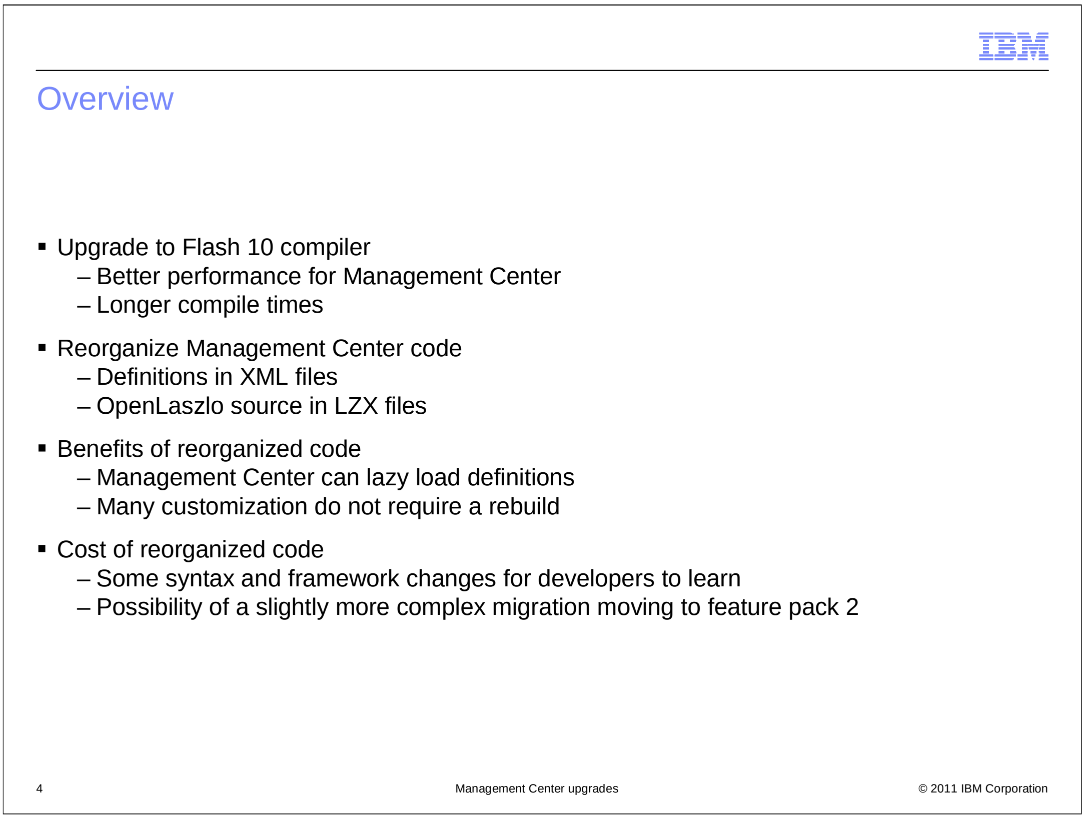
- Framework overview
- Migration

The first half of this presentation highlights the changes to the Management Center framework. The second half of the presentation focuses on the process for migrating previously customized files to the feature pack 2 level of Management Center.

Section

Framework overview

This section highlights the changes to the Management Center framework.



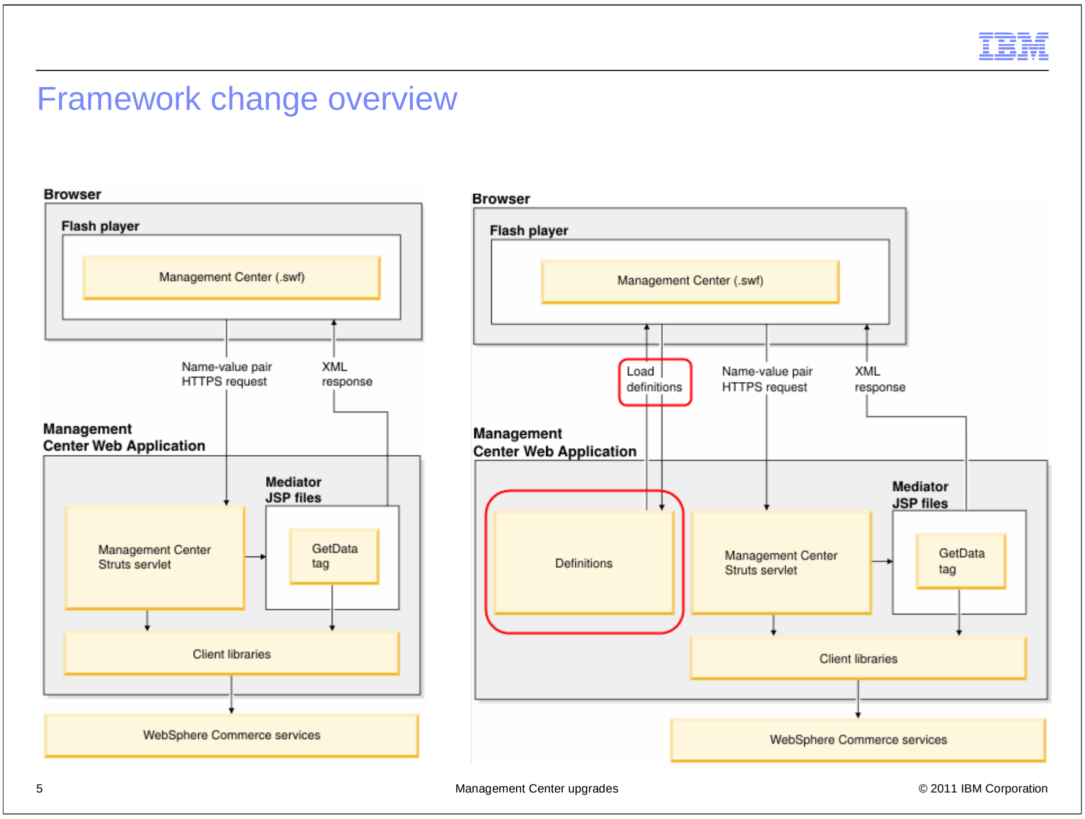
Overview

- Upgrade to Flash 10 compiler
  - Better performance for Management Center
  - Longer compile times
- Reorganize Management Center code
  - Definitions in XML files
  - OpenLaszlo source in LZX files
- Benefits of reorganized code
  - Management Center can lazy load definitions
  - Many customization do not require a rebuild
- Cost of reorganized code
  - Some syntax and framework changes for developers to learn
  - Possibility of a slightly more complex migration moving to feature pack 2

In feature pack 2, WebSphere Commerce Developer has been updated to use the Flash 10 compiler for compiling OpenLaszlo files. Upgrading to the Flash 10 compiler provides better performance of the compiled Management Center application. The tradeoff is the compiler takes longer to run. To reduce both the compilation time and the frequency that the application needs to be recompiled, the Management Center code has been restructured. The definition of most Management Center components has moved into XML definition files. All remaining code is in OpenLaszlo source files.

The benefit of this reorganization is the Management Center application can load object definitions as they are required, improving the initial start time. Also, adding or customizing object definitions can be done without the need to recompile the OpenLaszlo source files.

The cost of this change for adopters of feature pack 2 is the time required for tool developers to learn the new syntax and framework changes. If you have existing customized code, the migration process to feature pack 2 is slightly more complex. This presentation will describe how the migration tool has been updated to reduce the complexity as much as possible.



Framework change overview

Browser

Flash player

Management Center (.swf)

Name-value pair HTTPS request

XML response

Management Center Web Application

Mediator JSP files

Management Center Struts servlet

GetData tag

Client libraries

WebSphere Commerce services

Browser

Flash player

Management Center (.swf)

Load definitions

Name-value pair HTTPS request

XML response

Management Center Web Application

Mediator JSP files

Definitions

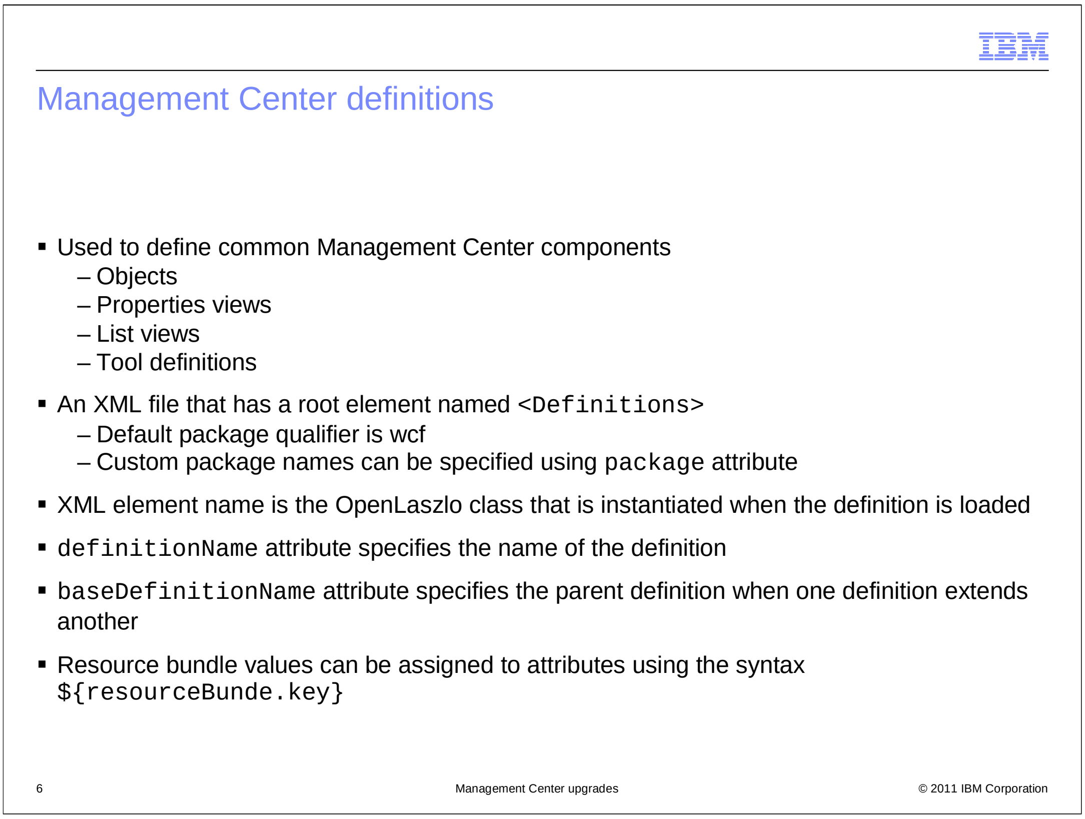
Management Center Struts servlet

GetData tag

Client libraries

WebSphere Commerce services

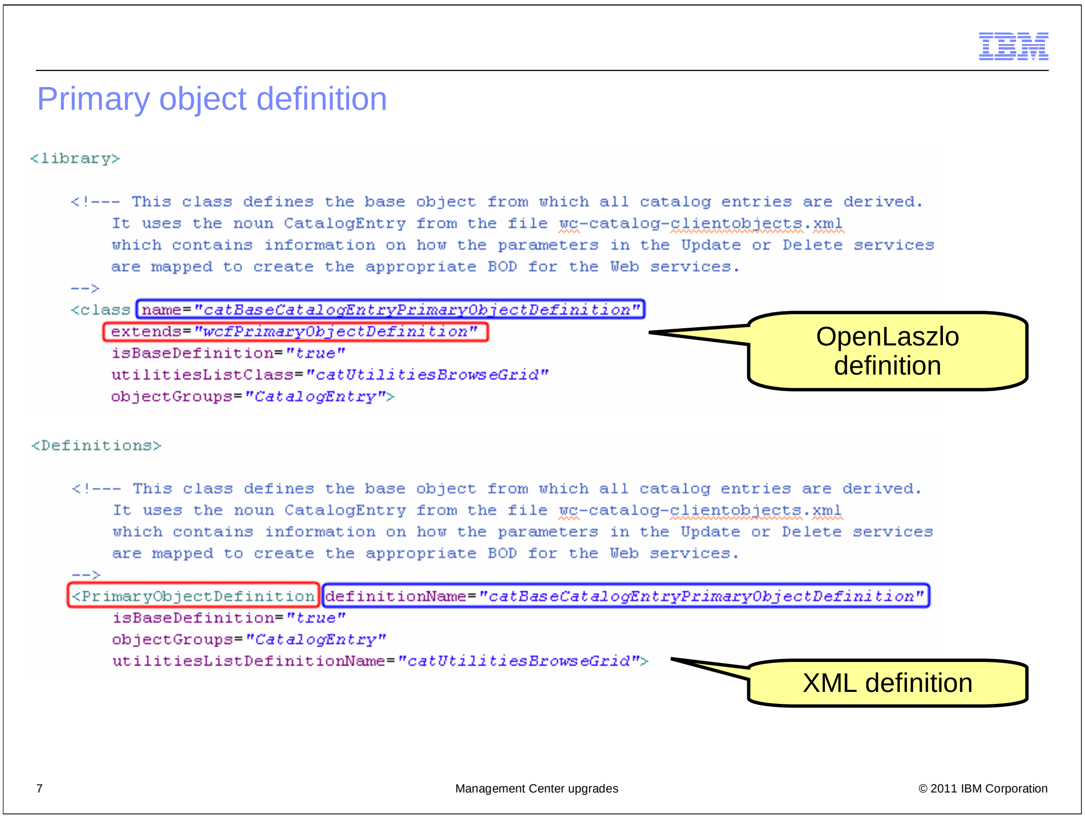
The diagrams on this slide compare the Management Center framework before feature pack 2 to the framework in feature pack 2. The changes are circled in red in the diagram on the right. The Management Center Flash application loads configuration information (called definitions) from the Management Center web application. The definitions are used by the Management Center framework to display all of the tools, business objects, properties views, and list views in Management Center.



Management Center definitions

- Used to define common Management Center components
  - Objects
  - Properties views
  - List views
  - Tool definitions
- An XML file that has a root element named `<Definitions>`
  - Default package qualifier is wcf
  - Custom package names can be specified using `package` attribute
- XML element name is the OpenLaszlo class that is instantiated when the definition is loaded
- `definitionName` attribute specifies the name of the definition
- `baseDefinitionName` attribute specifies the parent definition when one definition extends another
- Resource bundle values can be assigned to attributes using the syntax `${resourceBunde.key}`

All types of Management Center components can be described using definitions. This includes parent and child objects, property and list views and tool definitions. A root element named <Definition> replaces the <Library> root element used in OpenLaszlo files. All definitions shipped with feature pack 2 have a default package qualifier of wcf. You can specify your own package name for custom files by including the package attribute. The name of each tag in a definition file is the OpenLaszlo class that is instantiated when the definition is loaded. Some examples are shown on the next slide. The name of the object created is specified in the definitionName attribute. A definition can extend another definition by using the baseDefinitionName attribute to specify the parent definition. Resource bundle values can be assigned to attributes within a definition using the syntax shown on the slide.



Primary object definition

```
<library>

    <!--- This class defines the base object from which all catalog entries are derived.
        It uses the noun CatalogEntry from the file wc-catalog-clientobjects.xml
        which contains information on how the parameters in the Update or Delete services
        are mapped to create the appropriate BOD for the Web services.
    -->
    <class name="catBaseCatalogEntryPrimaryObjectDefinition"
        extends="wcfPrimaryObjectDefinition"
        isBaseDefinition="true"
        utilitiesListClass="catUtilitiesBrowseGrid"
        objectGroups="CatalogEntry">
```

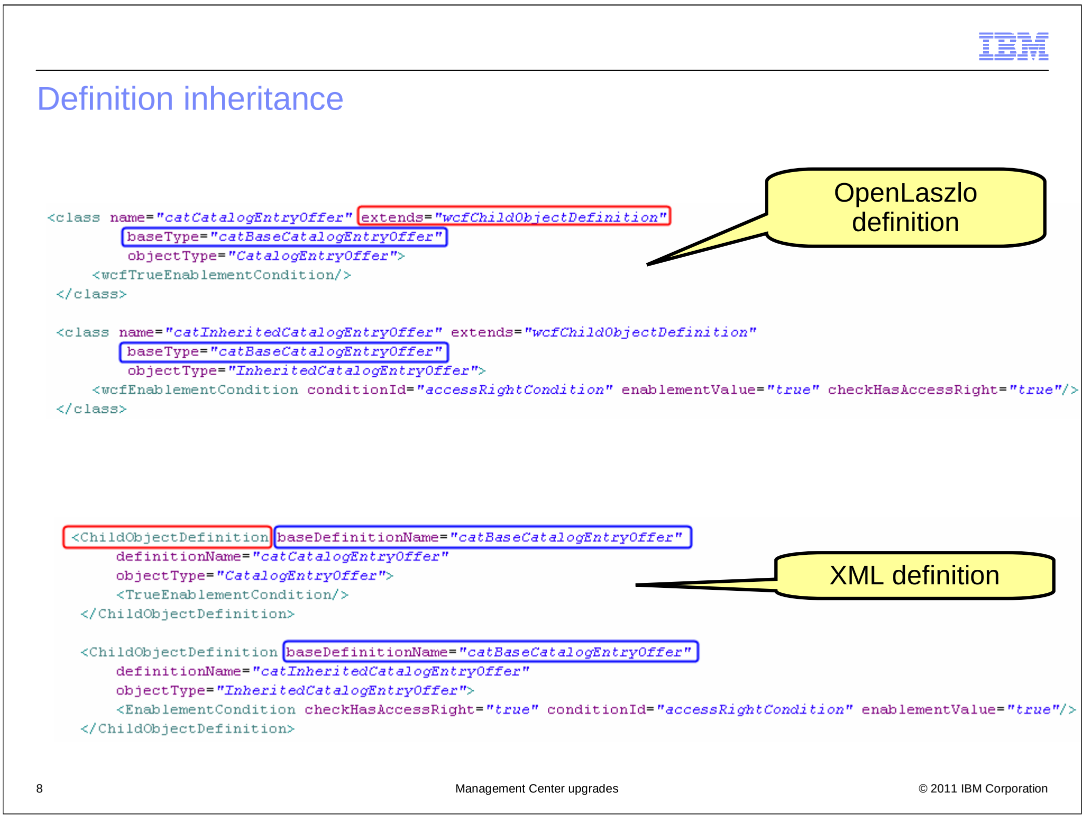
OpenLaszlo definition

```
<Definitions>

    <!--- This class defines the base object from which all catalog entries are derived.
        It uses the noun CatalogEntry from the file wc-catalog-clientobjects.xml
        which contains information on how the parameters in the Update or Delete services
        are mapped to create the appropriate BOD for the Web services.
    -->
    <PrimaryObjectDefinition definitionName="catBaseCatalogEntryPrimaryObjectDefinition"
        isBaseDefinition="true"
        objectGroups="CatalogEntry"
        utilitiesListDefinitionName="catUtilitiesBrowseGrid">
```

XML definition

This slide compares the syntax for a primary object definition. On the top half of the slide is the syntax for defining the object in OpenLaszlo. On the bottom half of the slide is the syntax for defining the object in a definition file. The area circled in red defines the type of OpenLaszlo object that is created. The area circled in blue defines the name of the object.



Definition inheritance

OpenLaszlo definition

```
<class name="catCatalogEntryOffer" extends="wcfChildObjectDefinition"
        baseType="catBaseCatalogEntryOffer"
        objectType="CatalogEntryOffer">
    <wcfTrueEnablementCondition/>
</class>

<class name="catInheritedCatalogEntryOffer" extends="wcfChildObjectDefinition"
        baseType="catBaseCatalogEntryOffer"
        objectType="InheritedCatalogEntryOffer">
    <wcfEnablementCondition conditionId="accessRightCondition" enablementValue="true" checkHasAccessRight="true"/>
</class>
```

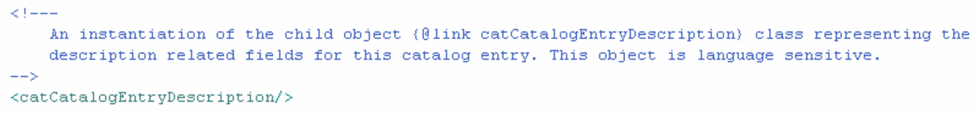
XML definition

```
<ChildObjectDefinition baseDefinitionName="catBaseCatalogEntryOffer"
     definitionName="catCatalogEntryOffer"
     objectType="CatalogEntryOffer">
     <TrueEnablementCondition/>
  </ChildObjectDefinition>

  <ChildObjectDefinition baseDefinitionName="catBaseCatalogEntryOffer"
     definitionName="catInheritedCatalogEntryOffer"
     objectType="InheritedCatalogEntryOffer">
     <EnablementCondition checkHasAccessRight="true" conditionId="accessRightCondition" enablementValue="true"/>
  </ChildObjectDefinition>
```

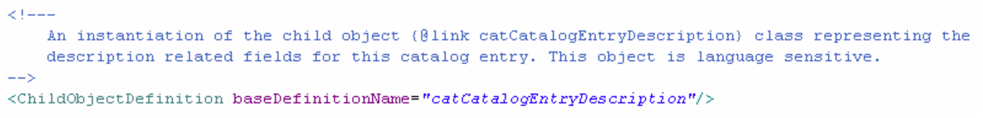
This slide shows a similar comparison for objects that extend other objects. The object type is still circled in red. This time the parent object name is circled in blue.

Instantiating objects



```
<!---
    An instantiation of the child object {@link catCatalogEntryDescription} class representing the
    description related fields for this catalog entry. This object is language sensitive.
-->
<catCatalogEntryDescription/>
```

OpenLaszlo file

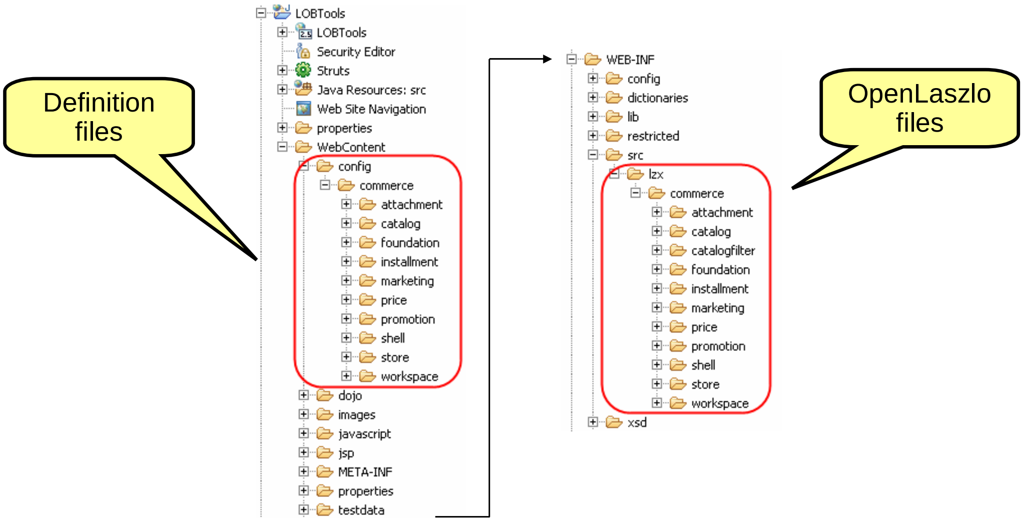


```
<!---
    An instantiation of the child object {@link catCatalogEntryDescription} class representing the
    description related fields for this catalog entry. This object is language sensitive.
-->
<ChildObjectDefinition baseDefinitionName="catCatalogEntryDescription"/>
```

Definition file

This example compares how objects are instantiated. The definition file makes use of the baseDefinitionName attribute to specify the definition to be instantiated.

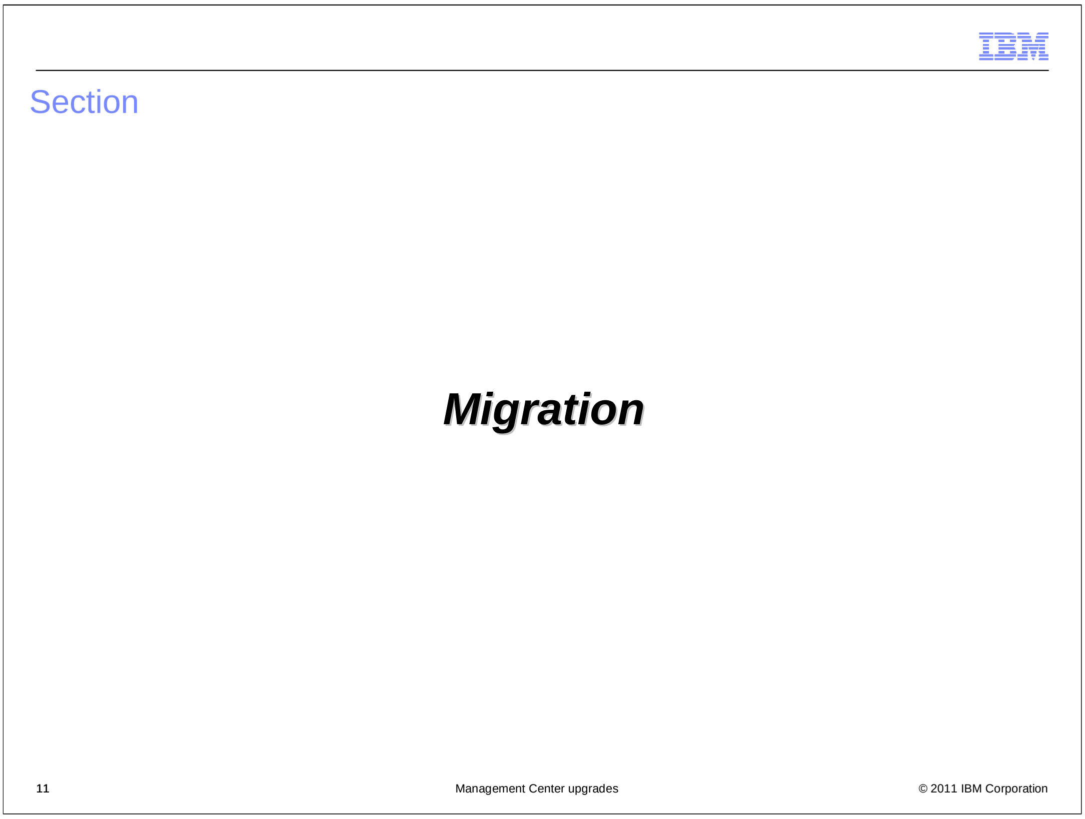
New project structure



Definition files

OpenLaszlo files

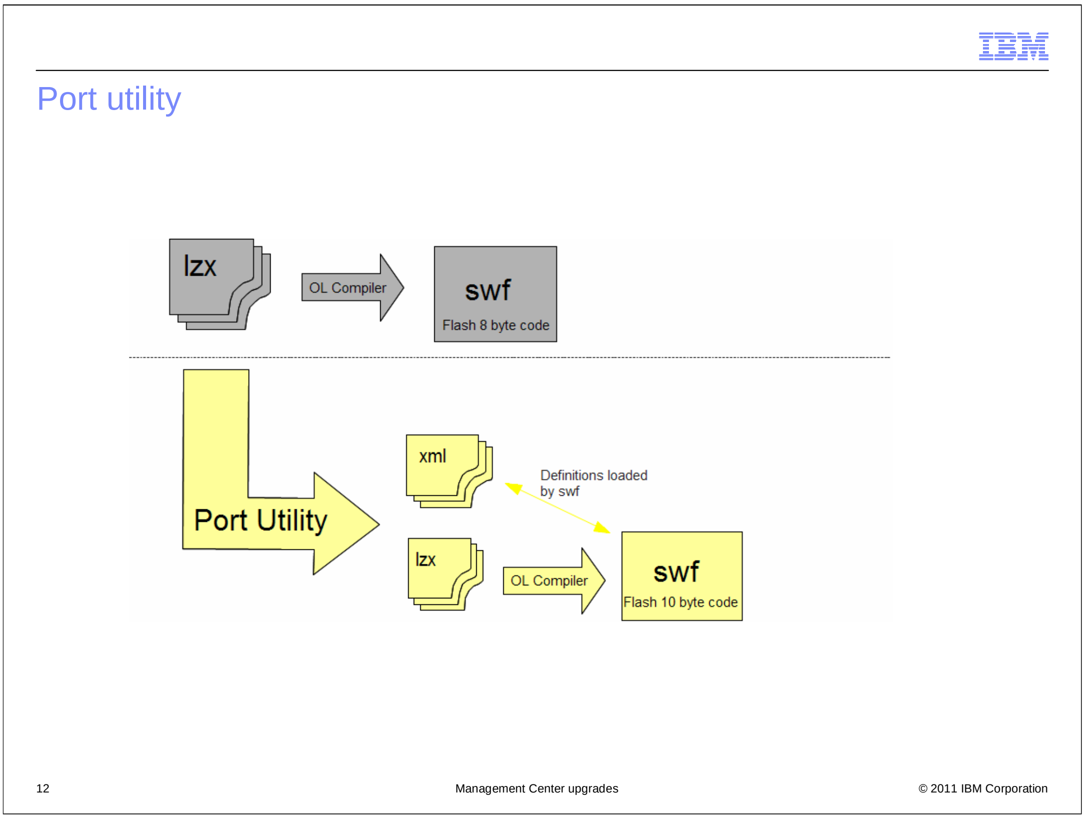
The Management Center definition files have moved to a new location within the LOBTools project. All definitions are in subdirectories under WebContent/config. The location of the OpenLaszlo files has not changed.



Section

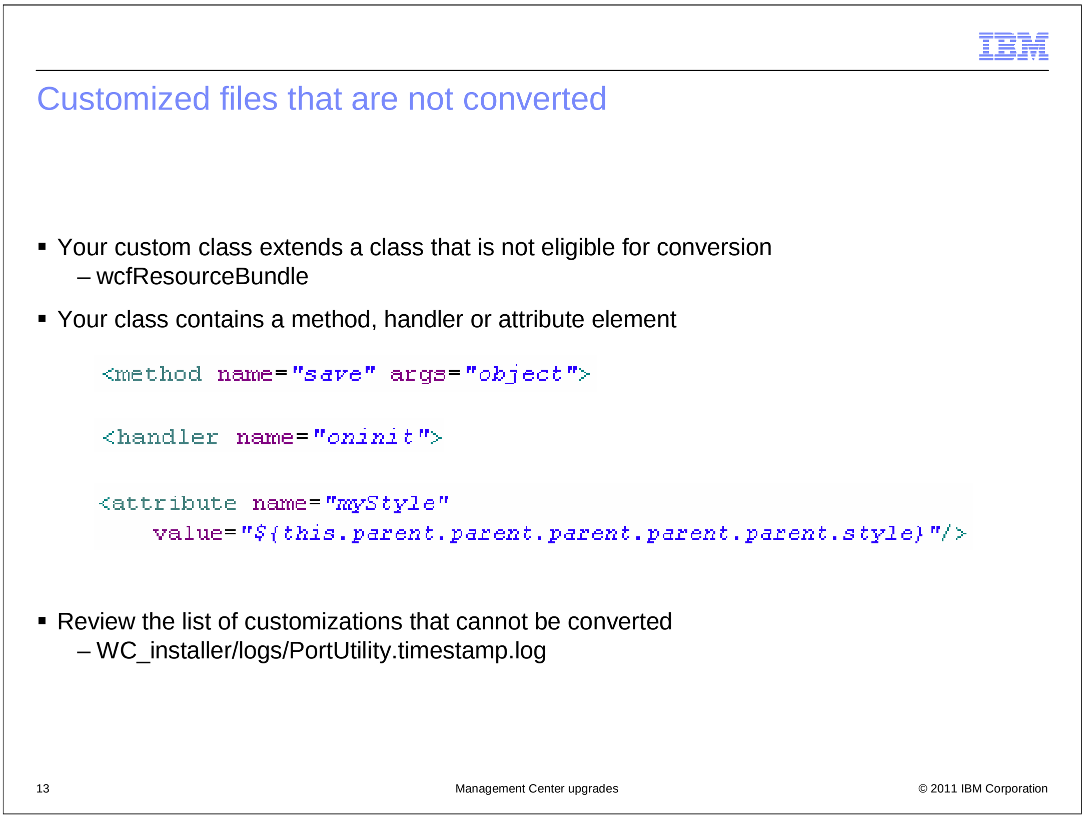
Migration

This section covers the migration process for customized Management Center files.



Port utility

When you enable the management-center feature in a new release, some automated migration steps are performed on your existing LOBTools project. They include backing up your existing project so you can compare your customized files with the new versions of those files and copy customized code into the new files. With the change in the structure of Management Center files in feature pack 2, a direct comparison between your current customized files and the new feature pack 2 files is difficult. To help overcome this difficulty, a port utility is included as part of the automated backup of your existing files. The port utility splits your current version of Management Center into definition files and OpenLaszlo source files. This provides a more direct comparison of existing versus new files and simplifies the manual part of the migration process.



Customized files that are not converted

- Your custom class extends a class that is not eligible for conversion
  - wcfResourceBundle
- Your class contains a method, handler or attribute element

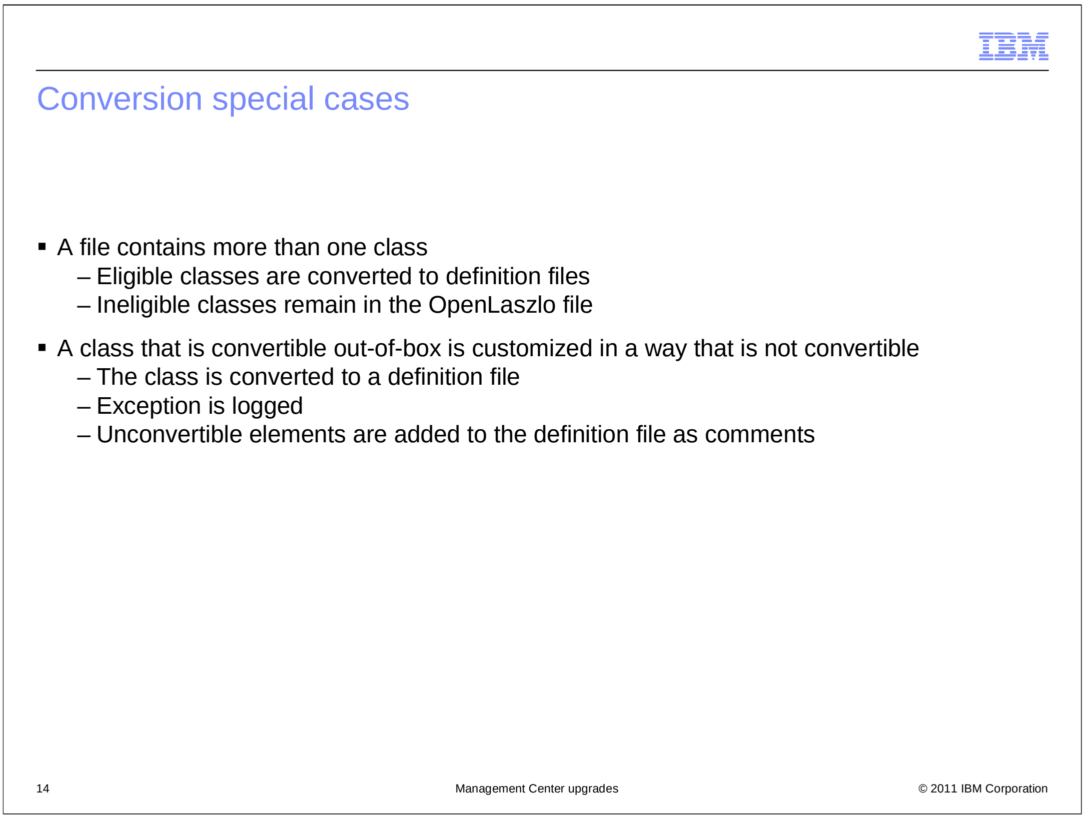
```
<method name="save" args="object">

<handler name="oninit">

<attribute name="myStyle"
    value="${this.parent.parent.parent.parent.parent.style}"/>
```

- Review the list of customizations that cannot be converted
  - WC_installer/logs/PortUtility.timestamp.log

Some of your customized code might not be eligible for conversion to definition files. If you extend a class that is not convertible, such as wcfResourceBundle, your custom class is not convertible. Another common reason your custom code is not converted into a definition file is it contains logic in the form of methods, handlers or attribute assignments that need to be evaluated. These are code assets and need to remain in OpenLaszlo source files. The port utility creates a log file of all customizations that cannot be converted.

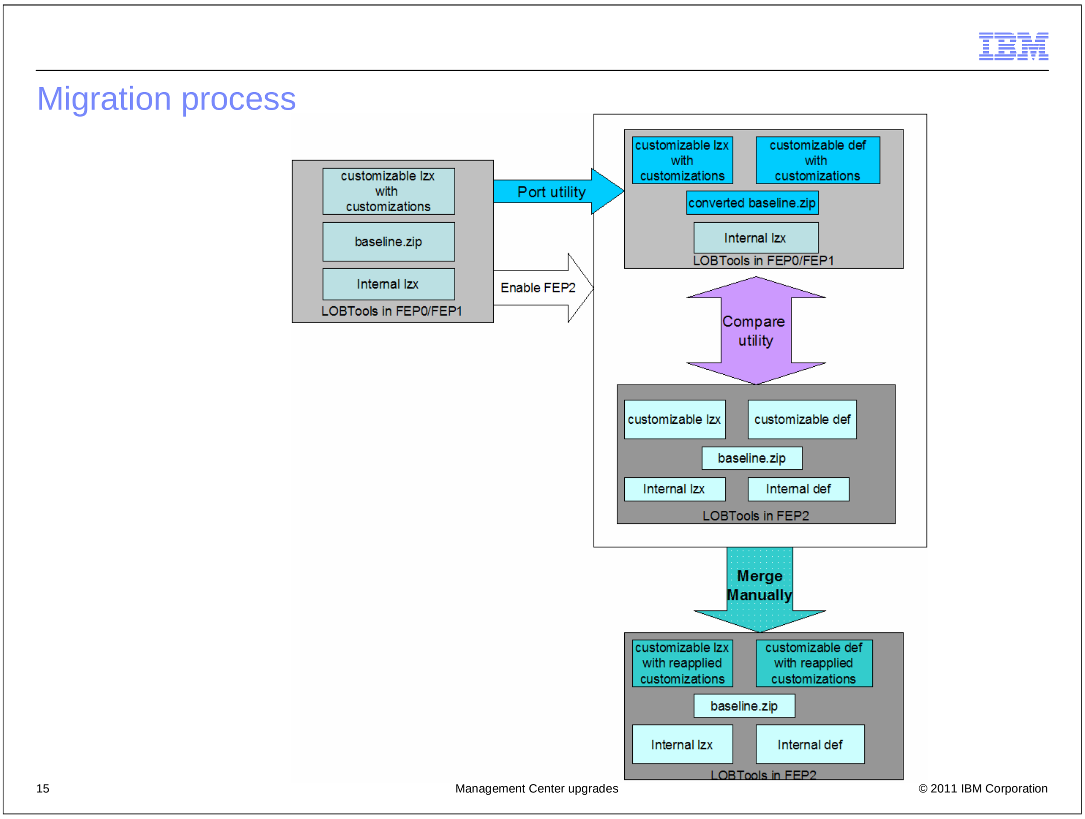


Conversion special cases

- A file contains more than one class
  - Eligible classes are converted to definition files
  - Ineligible classes remain in the OpenLaszlo file
- A class that is convertible out-of-box is customized in a way that is not convertible
  - The class is converted to a definition file
  - Exception is logged
  - Unconvertible elements are added to the definition file as comments

If your custom file contains multiple classes and only some are eligible for conversion, the file can be split into a definition file and an OpenLaszlo source file. The convertible classes are moved into the definition file.

For classes provided by IBM, if the class is convertible in it’s original state it is converted. If you have added custom logic into the file that is not convertible an exception is logged during the conversion process. Your custom code is included in the definition file as a comment.



Migration process

customizable lzx with customizations

baseline.zip

Internal lzx

LOBTools in FEP0/FEP1

Port utility

Enable FEP2

customizable lzx with customizations

customizable def with customizations

converted baseline.zip

Internal lzx

LOBTools in FEP0/FEP1

Compare utility

customizable lzx

customizable def

baseline.zip

Internal lzx

Internal def

LOBTools in FEP2

Merge Manually

customizable lzx with reapplied customizations

customizable def with reapplied customizations

baseline.zip

Internal lzx

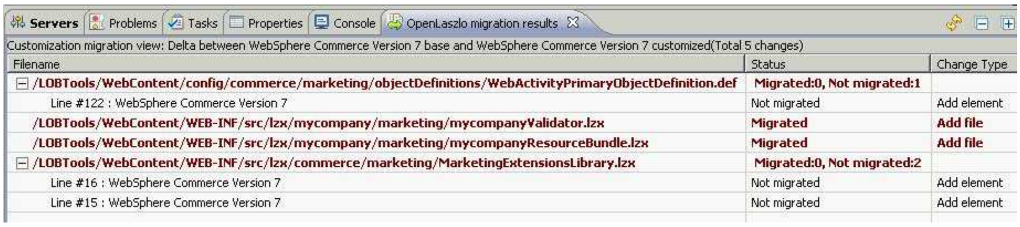
Internal def

LOBTools in FEP2

This flow chart shows the full migration process. When feature pack 2 is enabled, the customizable files are converted by the port utility into definition files and OpenLaszlo source files. Once the conversion is complete, you can use the compare utility to view the difference between the converted files and the new feature pack 2 files. With the assistance of the compare utility, you identify and manually merge the customized code that you want to keep. You should also review any exceptions logged by the port utility during the conversion process. Once the manual migration process is complete, your customized Management Center is ready to use in feature pack 2.

High level migration steps

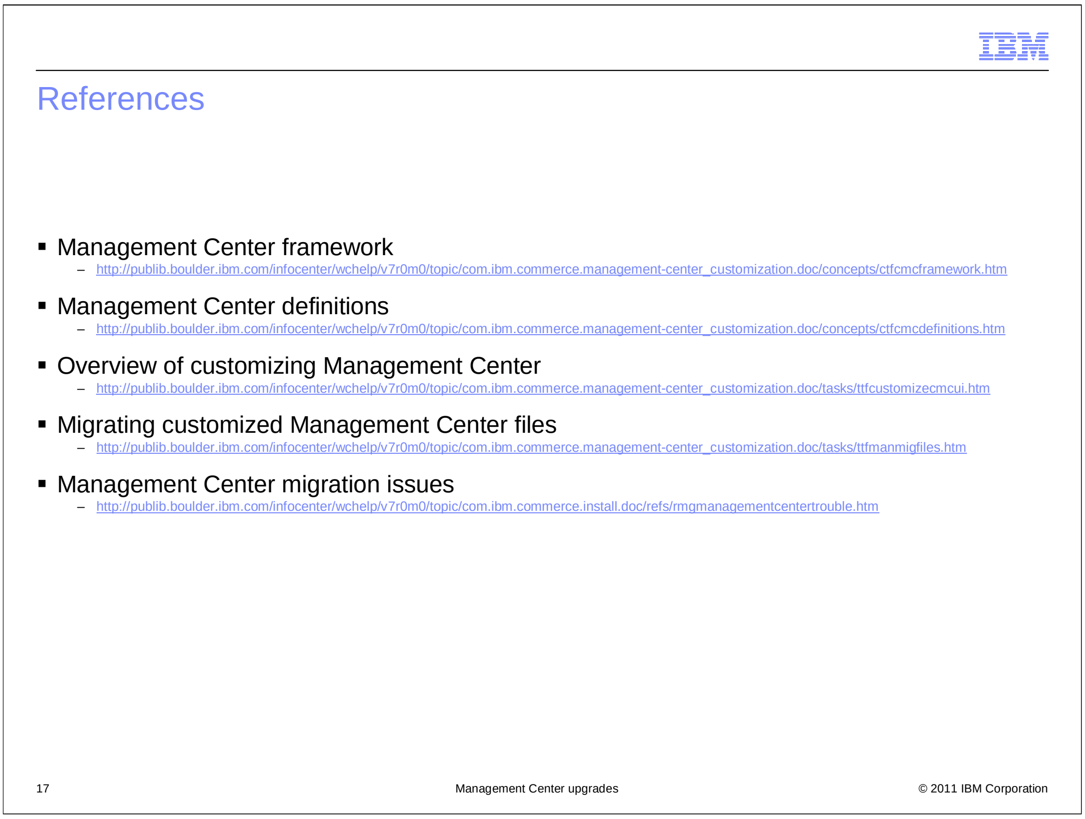
- Review your customizations before enabling feature pack 2
  - All customizations are in .lzx files
- Enable feature pack 2
- Use the comparison utility to view customization to be migrated
  - Customizations are in both .lzx and .def files
- Manually merge customizations



Customization migration view: Delta between WebSphere Commerce Version 7 base and WebSphere Commerce Version 7 customized(Total 5 changes)

Filename	Status	Change Type
⊟ /LOBTools/WebContent/config/commerce/marketing/objectDefinitions/WebActivityPrimaryObjectDefinition.def	Migrated:0, Not migrated:1	
Line #122 : WebSphere Commerce Version 7	Not migrated	Add element
/LOBTools/WebContent/WEB-INF/src/lzx/mycompany/marketing/mycompanyValidator.lzx	Migrated	Add file
/LOBTools/WebContent/WEB-INF/src/lzx/mycompany/marketing/mycompanyResourceBundle.lzx	Migrated	Add file
⊟ /LOBTools/WebContent/WEB-INF/src/lzx/commerce/marketing/MarketingExtensionsLibrary.lzx	Migrated:0, Not migrated:2	
Line #16 : WebSphere Commerce Version 7	Not migrated	Add element
Line #15 : WebSphere Commerce Version 7	Not migrated	Add element

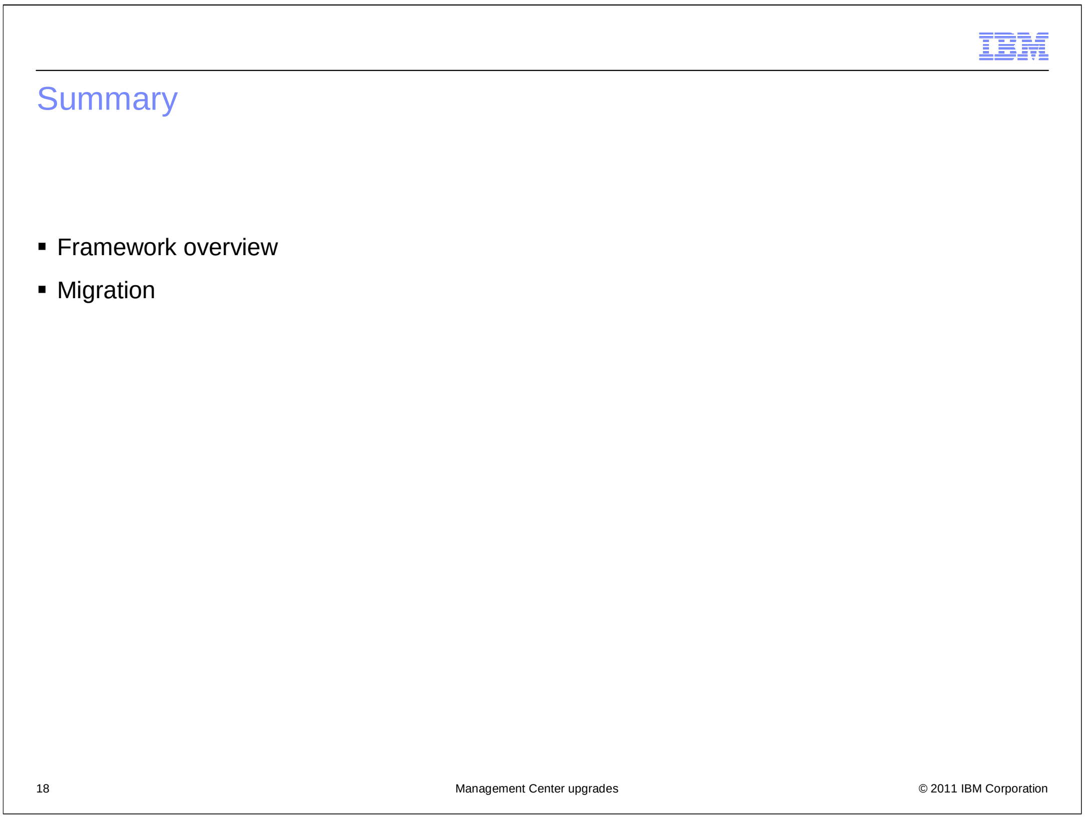
The slide summarizes the main steps in the migration process. You should start by reviewing all your customizations in their OpenLaszlo form before enabling feature pack 2. The comparison utility can help you with this review. Once you are familiar with the customized code and have identified any that is no longer needed, proceed with the feature pack 2 enablement. Use the comparison utility to view your customized code again. After enablement, the comparison utility shows your custom code in the new file structure. Using the migration tools to assist you, manually merge your customized code.



References

- Management Center framework
  - http://publib.boulder.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center_customization.doc/concepts/ctfcmcframework.htm
- Management Center definitions
  - http://publib.boulder.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center_customization.doc/concepts/ctfcmcdefinitions.htm
- Overview of customizing Management Center
  - http://publib.boulder.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center_customization.doc/tasks/ttfcustomizecmcui.htm
- Migrating customized Management Center files
  - http://publib.boulder.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center_customization.doc/tasks/ttfmanmigfiles.htm
- Management Center migration issues
  - http://publib.boulder.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.install.doc/refs/rmgmanagementcentertrouble.htm

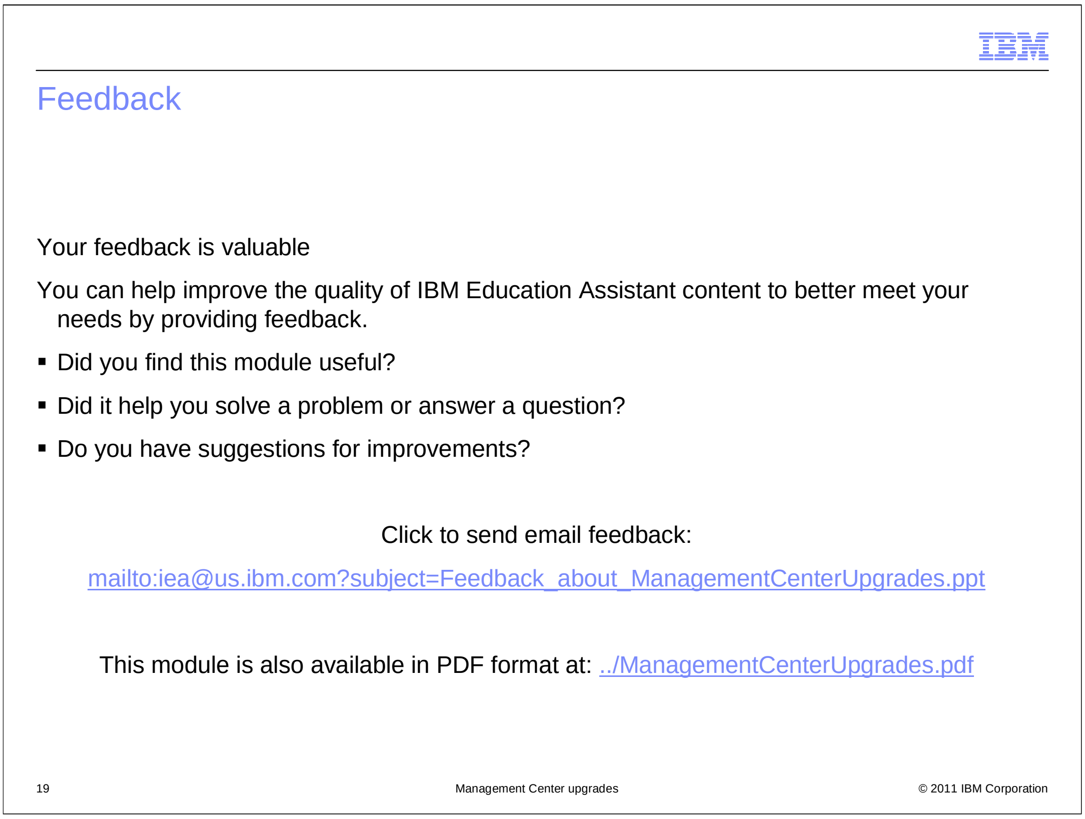
This slide contains some useful references.



Summary

- Framework overview
- Migration

The first half of this presentation introduced the changes to the Management Center framework. The second half of the presentation focused on the process for migrating previously customized files to the feature pack 2 level of Management Center.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_ManagementCenterUpgrades.ppt

This module is also available in PDF format at: ../ManagementCenterUpgrades.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.