Oracle® User Productivity Kit
Content Deployment

E65158-01
September 2015

ORACLE®

Oracle User Productivity Kit Content Deployment

E65158-01

September 2015

Contents

1 Deploy the Player

When you publish your content to a Player package in the Developer, you have several options for deployment. You can:

- place the Player package on a Web server and provide your users with a link to launch it
- place the Player package on a CD or other removable media so that users can play it offline
- launch the Player through a link in an application, called In-Application Support
- deploy the Player through a Learning Management system (LMS)
- deploy the content through Knowledge Center
- create links to specific topics

In all these cases, users play content through a browser interface. No software needs to be installed on the client workstation to play content, unless there is sound. If the content has sound, the client workstation must have Flash 9.0 or higher installed to play it. Note, however, that a web server hosting content with sound may not serve Abode Flash audio until the MIME types FLV and SWF are set (for example, application/x-shockwave-flash).

Setting Defaults for the Player
You set the default preferences for the Player during publishing. You can also choose to prevent users from changing the preferences. If you want to change the default preferences, republish the content with the new preference settings and copy it to the Web server. See the *Publishing Content* section in the Content Development manual for more information about publishing Player content and setting preferences.

1.1 Deploy Player Content on a Web Server

Place the Player package on a web server to make it available to your users. You do this by copying the contents of the PlayerPackage folder in the published content. You can then create a link to the index.html or default.htm page from an existing web site, or make the index.html or default.htm page itself available to users.

You can also deploy and manage Player content on a certified web server using the Content Manager.

Note: If you embed the Player in a browser frame in a portal, you should create a link to the tocx.html file to launch the Player Table of Contents.

1.1.1 Access Deployed Content using Internet Explorer

If you launch the Player from the Internet using Internet Explorer, you could have issues with pop-up blockers, full screen play, and the Do It! mode window not staying on top.

Note: Do It! mode does not stay on top in Internet Explorer 11 or later versions.

Pop-up Blockers
Pop-up blockers prevent the main Table of Contents window from launching. When attempting to launch, an information box appears below the Address Bar, stating a pop-up has been blocked. In addition, an Information Bar dialog box may appear.

Clicking in the Information Bar enables you to temporarily Allow Pop-ups, Always Allow Pop-ups from This Site, or change Settings. The possible settings changes are to turn off Pop-up Blocker, enable/disable Show Information Bar for Pop-ups, or More Settings. Selecting More Settings will open the Pop-up Blocker Settings dialog box. It is recommended that you not make changes using these options. You should instead put the web site with the content into a Trusted zone, as described below.

Note: If you select Always Allow Pop-ups from This Site, the site is automatically added to the Allowed sites in the Pop-up Blocker Settings dialog box. You can access this dialog box from Tools menus in Internet Explorer.

Do It! Mode and Full Screen Topic Play

To allow See It! and Try It! modes to launch full-screen and for Do It! mode to stay on top, the Player must be accessed from a zone in which the Allow Script-initiated Windows Without Size or Position Constraints setting is enabled. In a default configuration of Internet Explorer, it is recommended you add the web site where the content is located to either the:

- Local intranet zone where the Allow Script-initiated Windows Without Size or Position Constraints setting is enabled by default

- Trusted sites zone and enable the Allow Script-initiated Windows Without Size or Position Constraints setting

If you do not add the web site to either of these zones, See It! and Try It! modes appear in browser windows, and scroll bars appear as necessary to allow the user to view the content. In addition, Do It! mode appears as a browser window and will not stay on top.

🖱 To add the web site to the Local intranet zone:

1. Open Internet Explorer.

2. On the **Tools** menu, choose **Internet Options**.

3. Click the **Security** tab.

4. Click the **Local Intranet** icon.

5. Click **Sites**.

6. Click **Advanced**.

7. In the **Add this Web site to the zone** field, enter the URL for the Player.

8. Deselect the **Require server verification (https:) for all sites in this zone** option, if necessary.

9. Click **Add**.

10. Click **Close**.

11. Click **OK** to close the Local Intranet dialog box.

12. Click **OK** to close the Internet Options dialog box.

🖱 To add the web site as a Trusted site and enable the Allow script-initiated windows without size or position constraints setting:

1. Open Internet Explorer.

2. On the **Tools** menu, choose **Internet Options**.

3. Click the **Security** tab.

4. Click the **Trusted sites** icon.

5. Click **Sites**.

6. In the **Add this Web site to the zone** field, enter the URL for the Player.

7. Deselect the **Require server verification (https:) for all sites in this zone** option, if necessary.

8. Click **Add**.

9. Click **Close**.

10. Click **Custom Level**.

11. Enable the **Allow script-initiated windows without size or position constraints** setting.

12. Click **OK** to close the Security Settings - Trusted Sites Zone dialog box.

13. Click **OK** to close the Internet Options dialog box.

The Player interface should now launch correctly. In addition, the play modes will not have title bars, status bars, or a window border, and the Do It! mode will launch.

1.1.2 Access Deployed Content using Firefox

When the user launches a topic in the Player in Do It! mode using Firefox, the Do It! mode window will not stay on top as they interact with the target application window. To allow the Do It! window to stay on top, the user must have the Stay-on-Top plug-in installed on their client workstation. The plug-in maintains a list of trusted sites which are allowed to have stay-on-top windows.

The plug-in can be installed in one of three ways:

- Deployed by the system administrator via the SmartHelp installer

  The administrator pushes out the SmartHelp.exe to each user's machine. See *Use the SmartHelp Installer* in the Install SmartHelp for Browser-based Applications in the Implement In-Application Support chapter of the In-Application Support manual for instructions on how to create a silent install script.

- Downloaded and installed on the user's machine

  The administrator provides the user with the location of the SmartHelp.exe file and instructions on how to run the executable to install the SmartHelp button in Firefox, including the URL to the content. See *Use the SmartHelp Installer* in the Install SmartHelp for Browser-based Applications in the Implement In-Application Support chapter of the In-Application Support manual for specific instructions.

- Run the SmartHelpSetup.html from the user's machine

  The administrator must set the appropriate MIME types on the web server, provide the user with the location of the SmartHelpSetup.html file to install the SmartHelp button in Firefox, provide the user with information on post installation tasks such as adding the button to the tool bar via Customize option (if necessary) and add the location of the deployed content to the add-on exception list. See *Install SmartHelp for Mozilla Firefox* in the Install SmartHelp for Browser-based Applications in the Implement In-Application Support chapter of the In-Application Support manual for specific instructions.

After the plug-in is installed and the user launches a topic in Do It! mode for the first time, the Stay-on-Top dialog opens and prompts the user with these options:

- **Always**: The site is added to the list of trusted sites and Do It! mode will stay on top. The user is not prompted when Do It! mode is launched in subsequent sessions.

- **Just this time**: The site stays on top for this session. The next time Do It! mode launches, the user is prompted to allow it.

- **Never**: The site is added to the list of denied sites and does not stay on top.

The user's choice affects how the web site is treated on future launches. Each time the Player is launched, the URL is checked against a list of trusted sites for the plug-in. The list of trusted sites is maintained on the Stay-On-Top Preferences dialog box. The user can access this dialog box from the Preferences button on the Stay-On-Top dialog box or from the Stay-On-Top Preferences item on the Firefox Tools menu.

1.2 Play Content from a Hard Drive or CD

When launching the Player Package directly from removable media or a local hard drive, you should use play.exe instead of launching index.html or default.htm. Using play.exe avoids several operating system limitations associated with HTML playback from the file system, such as no sound playback support and Internet Explorer security warning messages in the Player window.

1.2.1 Use Play.exe

When you launch the Player using play.exe, the Play dialog box opens to allow you to select a browser installed on your workstation. This dialog box opens each time you launch play.exe unless you mark the Do not ask me again checkbox.

Note: If you opted to not have the Play dialog box display, the last browser selected is used when you launch play.exe. To reinstate the Play dialog upon play.exe launch, use the Reset Dialogs button under the General category (under Developer) in the Options dialog box to reset the option. If you do not have access to the Developer, you can run play.exe with the /reset option.

Play.exe can be used with these command line parameters:

/browser=[firefox,safari,ie,system]
This parameter sets the browser to use when you launch play.exe. If you select the system option, the system default browser is used. If the system default is not a supported browser, Internet Explorer is used.

/reset
This parameter resets the browser to the system default browser. Also, resets the Do not ask again box on the Play dialog box to unmarked.

/log
This parameter creates a log file in the directory where play.exe is launched.

/i
This parameter forces the Play dialog box to display. This is used to bring back the Play dialog box after the Do not ask me again checkbox has been marked.

/?, /h, /help
These parameters display a list of all the command line options.

URL
This parameter launches the topic from the provided URL. If no URL is provided, play.exe launches index.html from the same directory in which play.exe is located.

To run play.exe from a command prompt:

1. Open a command prompt or the Run dialog box.

2. Change the directory or enter the full path to play.exe.

3. Type **play.exe** and append the desired parameters.

4. Press ENTER to launch the Player.

1.3 In-Application Support

The Player deployment format that is published from the Developer is not only useful in various training and educational initiatives; it is also designed to provide in-application support for end users.

There are specific add-ons and instructions for certain major enterprise applications to accommodate context-sensitive in-application support; these applications are referred to as target applications throughout this document. Content for target applications can generally be deployed using one or both of the following methods, depending on the target application:

- Help Integration: This method allows users to invoke context-sensitive in-application support content using the existing target application Help menu or Help link.

- SmartHelp: This is a toolbar button and a sidebar for browsers that end users may use to access context-sensitive in-application support content. SmartHelp also allows for full text search of content directly from the browser sidebar.

The mechanism for context-sensitivity used for target applications is known as ExactMatch. This technique allows the Player to display a list of content that is related to the end user's activity in the application. The relevancy of the content list is determined based on content items that contain exactly the same context information as recorded during content creation in the Developer.

The mechanism for context-sensitivity for browser-based non-target applications is known as SmartMatch. This technique uses statistical analysis to rank the most probable content items and generates a list of relevant content. This allows you to use the SmartHelp toolbar button to deploy context-sensitive in-application support for any browser-based application.

Note: There are implications on applicable results in the Player when the content author changes the Applicable outline display option from the default. See Applicable outline display in the Player Defaults topic of the Set Developer Defaults section of the Get Started using the Developer chapter in the Content Development guide.

See the *In-Application Support* guide for specific instructions about help integration and SmartHelp content deployment.

1.3.1 Plan for Deploying In-Application Support

Once you decide to deploy content for in-application support, some of the fundamental questions that need to be addressed are:

- What deployment mechanism is most appropriate for your application, Help Integration with a target application or SmartHelp-based deployment?

- Who needs to be involved, what needs to be done, and when does it have to happen?

Understanding the answers to these questions will give you the information you need to implement help integration and context-sensitive help successfully.

These are the tasks which need to be completed to successfully deploy in-application support.

Content Author

Create content in the Developer: While context for many applications is automatically captured, it is recommended that you validate that context is being captured. You can check this by opening the topic in the Topic Editor and then opening the Context ID editor. If the Type field displays SmartMatch or ExactMatch, context was captured. If it displays NoMatch, no context was captured. In addition, while you are recording a topic, there is an icon in the Recorder which shows context recognition status.

Create an IAS Configuration document in the Developer: Consult with the application and content web server administrators to find out the necessary URLs.

Give the published content to the web server administrator.

Web Server Administrator

Deploy the published content to a web server.

Application Administrator

Perform the configuration steps detailed in this guide for each application.

Receive the content URL from the web server administrator.

Perform help integration steps for URL configuration, if applicable.

Test User

Verify the published content appears from the application using the Help Menu Integration option or the SmartHelp button, as deployed.

If you are deploying context-sensitive help, the test user should coordinate with the content developer to note which parts of the application were used to create content and then verify that the content launches with an appropriate list of relevant topics.

1.4 Deploy the Player through an LMS

The Player package includes the files necessary for use in any AICC or SCORM compliant Learning Management System (LMS).

Note: AICC does not recognize double-byte character sets, such as Japanese or Chinese. If your content uses double-byte characters in outline document titles, the published Player package only supports a SCORM compliant Learning Management System.

There are several published learning specifications to which learning management systems can comply. Published content is compliant with the following specifications:

AICC AGR-010: Web based CMI Guidelines (Version 3.5)

- CBT Course

- Individual CBT AU

SCORM Specifications: Version 1.2

- SCORM Version 1.2 Run-Time Environment Conformant (SCO-RTE1+Mandatory+Optional)

- SCORM Version 1.2 Content Packaging XML Conformant (ADLCP-PIF1)

SCORM Specifications: Version 1.3 (SCORM 2004 4th Edition)

- CP SCORM 2004 4th Ed. Compliant (CP CAM 1.1 & CP RTE 1.1)

- SCO SCORM 2004 4th Ed. Compliant (SCO RTE 1.1)

For more information regarding AICC please visit *www.aicc.org*, and for more information regarding ADL and SCORM please visit *www.adlnet.org*.

Content consists of modules and sections with concepts, inline questions, assessments, and topics which have been recorded. When content is published, each of these documents is represented in the course structure files as an Assignable Unit (AU) for AICC or Shareable Content Object (SCO) for SCORM. In addition, all modules and sections, including those with no concepts, are represented in course structure files as a block (AICC) or item (SCORM).

1.4.1 AICC Compliance

The Player package adheres to the following compliance criteria as defined in the AICC specifications for AGR-010 (Web-based CMI):

- A set of AICC specified course structure files are provided that can be imported into an LMS.

- Content must be launched by an AICC-compliant LMS.

- Content communicates with the LMS with the HTTP based AICC/CMI protocol (HACP).

1.4.1.1 Content Structure and Description Files

The Developer generates the following files to conform with AICC Level I (descriptions for course content and course structure) requirements. These files are located at the root of the PlayerPackage folder in a published Player package.

The Developer will create the following AICC files:

Course Description File (.CRS): **output.crs**

Assignable Unit File (.AU): **output.au**

Descriptor File (.DES): **output.des**

Course Structure File (.CST): **output.cst**

Of these, the output.au file deserves special attention. This file contains URLs for each of the assignable units to be imported into the LMS. The AICC specification requires that these URLs be absolute URLs referencing the location where the content will reside on the content server. If at publishing time you know where on your content server you will be placing the published content, you can specify a Prefix URL in the Publishing Wizard and the output.au file will be generated accordingly. If not, you will need to manually edit the output.au file to reflect the location on the content server before you can import the content into your LMS.

1.4.1.2 AICC API Function Calls

During play, the Player package makes the following AICC calls:

- GetParam
  Called at AU startup to read all variables from the LMS.

- PutParam
  Called when the user navigates away from the AU, prior to terminating the session, to write back all variables written to the LMS.

- ExitAU
  Called after PutParam when the user navigates away from the AU. This closes the LMS session.

Note: Unlike when communicating with a SCORM LMS, when communicating with an AICC LMS the Player package only communicates when the AU starts up and when it shuts down. There is no immediate communication of completion status changes.

1.4.1.3 AICC Data Element Usage

The following AICC data elements are used (HACP data element names are in brackets):

- Core.Student ID [Student_ID] - Read

- Core.Student Name [Student_Name] - Read

- Core.Lesson Location [Lesson_Location] - Read/Write

  For an LMS Group, this element holds the item number in the outline of the last item the user visited. For more information about LMS Groups, see the LMS Groups section in Use Content through an LMS.

- Core.Lesson Status [Lesson_Status] - Read/Write

  This element holds the completion status of the item. For scored items (assessments, inline questions, and topics with Know It? mode available), this is either "passed" or "failed" and Core.Score holds the score of the item. For unscored items (module/section concepts, topics without Know It? mode available, and LMS Groups), this is either "completed" or "incomplete". If the item has never been viewed, the LMS may return either "not attempted" or "unknown", both of which the Player handles as "not attempted".

- Core.Exit [{appended to Lesson_Status}] - Write

  This element sends the exit state of the item to the LMS. If the user leaves the item prior to completing it, ", suspend" is appended to the value of Lesson_Status and the information stored in [Core_Lesson] is used when the user returns to the item.

- Core.Score [Score] - Write

  This element holds the raw score (0-100) for scored items (assessments, inline questions, and topics without Know It? mode available). For unscored items (module/section concepts, topics without Know It? mode available, and LMS Groups), no score information is sent to the LMS.

- Core.Session Time [Time] - Write

- Suspend Data [[Core_Lesson]] - Read/Write

  Holds detailed completion state information for topics, assessments, and LMS Groups.

  For topics, this stores the state of each of the play modes of the topic, along with a score and pass/fail indicator if Know It? mode was completed. Example (showing See It!, Try It!, and Know It? have been completed, with a 100% correct execution of Know It? mode; Do It! and Print It! are incomplete):
  S=C,T=C,K=C,D=I,P=I,PASS=1,SCORE=100

  For assessments, this stores a checksum of the questions in the assessment along with an indication of the states of each of the questions in the assessment. For LMS Groups, this stores a checksum of the outline along with a run length encoded indication of the state of each of the items in the outline.

## *1.4.1.4 Configure Content for a non-LMS Server using AICC*

The Player package uses a Java applet to handle communication with the LMS. This applet is signed, as required by recent versions of Java, but is set to still run within the sandbox security restrictions. If you install a Player package on a web server other than the LMS server, the applet will not be able to communicate with the LMS server due to cross domain restrictions enforced by the browser. When you launch the content, you receive an error (java.security.AccessControlException) and the attempt is not tracked by the LMS.

In previous versions of UPK, this could be handled by moving the Java applet from the content server to the LMS server. This is no longer necessary and the cross domain issue is now handled by deployment of a crossdomain.xml file to the document root of the LMS server. A suitable crossdomain.xml file is written to the aicc directory of published courses (the same directory in which the applet's .jar file is located).

Although we do not believe that moving the Java applet to the LMS server will normally be necessary, it is possible that some server configurations might still require this. The following section describes how to do this.

Moving the Java Applet to the LMS Server

Note: It is unlikely that you will need to do this and should only do so if your particular server configuration makes this necessary. Note that only one copy of the Java applet is needed on the LMS server for multiple LMS Packages on the content server.

To deploy the Java applet to the LMS server:

1. Locate the file gethttppostdata.jar in the …\aicc folder where you published the content.

2. Copy this file to a location on the LMS web server that can be referenced through a URL. Note the full URL to the directory containing the file. For example:

    http://lmsserver/somedirectory

3. Open the config.js file in the root directory of the location where you published the content using any text editor such as Notepad.

4. Locate the following line in the file:

    AiccAppletPath : "",

5. Change the AiccAppletPath value to the URL that was noted in step #2. For example:

    AiccAppletPath : "http://lmsserver/somedirectory",

6. Save the file.

Note: If you will normally be deploying your content this way, you may want to edit the Player publishing style to avoid having to perform the steps above each time you publish content. In the Player style, you edit the config.xml file, locate the <item name="AiccAppletPath" line, and add the URL before the close of the item tag such as:

'http://127.0.0.1:81';">http://lmsserver/somedirectory</item>

For more information about customizing Player styles, see *Customize the Player* in the Publish Content section in the Content Development manual.

1.4.1.5 Configure Content for SSL (HTTPS) Protocol

The AICC specification is unclear on what protocol should be used as the default when no protocol is specified in the AICC_URL parameter on the launch URL. By default, the Player uses the HTTP protocol if no protocol is specified. You can change this to use HTTPS instead if you need to. Note that this is ignored if the LMS passes the protocol in the AICC_URL parameter.

To set the default protocol to HTTPS:

1. Open the config.js file in the root directory of the location where you published the content using any text editor such as Notepad.

2. Locate the following line in the file:

    AiccUrlProtocol : "http://",

3. Change the AiccUrlProtocol value to https:

    AiccUrlProtocol : "https://",

4. Save the file.

Note: As with the AiccAppletPath, you may want to edit config.xml in the Player publishing style if you always use HTTPS as the default protocol for your published content.

1.4.2 SCORM Compliance

The Developer generates Learning Content that is SCORM 1.2 and SCORM 2004 4th Edition compliant.

Content may be imported into, and launched from, either a SCORM 1.2 or SCORM 2004 (1.3) compliant LMS.

1.4.2.1 Content Structure and Metadata Files for SCORM 1.2 and SCORM 2004

The Developer generates the following files in accordance with SCORM 1.2 Conformance and SCORM 2004 4th Edition Compliance.

Manifest files:

*   imsmanifest.xml
    The manifest file in either SCORM 1.2 or SCORM 2004 format, depending on the SCORM version setting in the Publishing Wizard.

*   imsmanifest12.xml
    A SCORM 1.2 conformant manifest file if the SCORM version is set to 1.3 (2004) in the Publishing Wizard.

*   imsmanifest13.xml
    A SCORM 2004 4th Edition compliant manifest file if the SCORM version is set to 1.2 in the Publishing Wizard

These documents are located at the root of the PlayerPackage folder in a published Player package.

1.4.2.2 SCORM API Function Calls

During play, the Player package makes the following SCORM API calls (SCORM 1.2 API names in brackets):

*   Initialize [LMSInitialize]
    Called when the SCO is first launched by the LMS.

- Terminate [LMSFinish]
  Called when the user navigates away from the SCO.

- GetValue [LMSGetValue]
  Called at SCO startup to read all variables from the LMS.

- SetValue [LMSSetValue]
  Called when the user navigates away from the SCO, prior to terminating the session, to write back all variables to the LMS. In addition, the SCO's completion status and score (if any) are written to the LMS any time the completion status of the SCO changes. This allows the LMS to immediately change the SCO's indicator in its user interface, if the LMS supports this.

- Commit [LMSCommit]
  Explicitly called whenever the SCO's completion status is reported back to the LMS. When the user navigates away from the SCO, an implicit commit takes place when the session is terminated.

- GetLastError [LMSGetLastError]
  Called whenever an LMS function returns an error.

- GetErrorString [LMSGetErrorString]
  Called whenever an LMS function returns an error. This error is then reported through the Player interface.

1.4.2.3 SCORM Data Element Usage

The following SCORM data elements are used by the Player package (SCORM 1.2 data element names are in brackets):

- cmi.learner_id [cmi.core.student_id] - Read

- cmi.learner_name [cmi.core.student_name] - Read

- cmi.session_time [cmi.core.session_time] - Write

- cmi.completion_status, cmi.success_status [cmi.core.lesson_status] - Read/Write

  This element holds the completion status of the item. For scored items (assessments, inline questions, and topics with Know It? mode available), this will be either "passed" or "failed" and cmi.score.raw [cmi.core.score.raw] will hold the item's score. For unscored items (module/section concepts, topics without Know It? mode available, and LMS Groups), this will be either "completed" or "incomplete". If the item has never been viewed, the LMS may return either "not attempted" or "unknown", both of which the Player handles as "not attempted".

  In SCORM 2004, the completion status is stored separately from the success (passed/failed) status. When a scored item is completed, cmi.completion_status will be set to "completed" and cmi.success_status will be set to either "passed" or "failed"

- cmi.suspend_data [cmi.suspend_data] - Read/Write

  This element holds the detailed completion state information for topics, assessments, and LMS Groups.

  For topics, this stores the state of each of the play modes of the topic, along with a score and pass/fail indicator if Know It? mode was completed. Example (showing See It!, Try It!, and Know It? have been completed, with a 100% correct execution of Know It? mode; Do It! and Print It! are incomplete):
  S=C,T=C,K=C,D=I,P=I,PASS=1,SCORE=100

For assessments, this stores a checksum of the questions in the assessment along with an indication of the states of each of the questions in the assessment. For LMS Groups, this stores a checksum of the outline along with a run length encoded indication of the state of each of the items in the outline.

- cmi.location [cmi.lesson_location] - Read/Write

  For an LMS Group (see LMS Groups), this element holds the item number in the outline of the last item the user visited.

- cmi.exit [cmi.core.exit] - Write

  This element sends the exit state of the item to the LMS. If the user has completed the item, exit state is set to "normal". If the user leaves the item prior to completing it, exit state is set to "suspend" and the information stored in cmi.suspend_data is used when the user returns to the item.

- cmi.score.raw [cmi.core.score.raw] - Write

  This element holds the raw score (0-100) for scored items (assessments, inline questions, and topics with Know It? mode available). For unscored items (module/section concepts, topics without Know It? mode available, and LMS Groups), no score information is sent to the LMS.

- cmi.score.scaled - Write

  This element holds the score for the item, scaled to the range 0.0-1.0. SCORM 2004 only.

- cmi.score.min [cmi.core.score.min] - Write

  This element is the item's minimum possible score. Always 0.

- cmi.score.max [cmi.core.score.max] - Write

  This element is the item's maximum possible score. Always 100.

1.4.2.4 Switch SCORM Versions in Published Packages

The published content contains the necessary files to support both SCORM versions. SCORM, however, requires that the manifest be named imsmanifest.xml and that it reside at the root of the Player package. This file will contain the manifest for the version you selected during publishing. The manifest for the other version will be named either imsmanifest12.xml or imsmanifest13.xml, depending on your selections during publishing.

For example, when you publish for SCORM 1.2 conformance, the published package includes an imsmanifest.xml file for version 1.2 and an imsmanifest13.xml file for version 1.3. Alternatively, when you publish for SCORM 1.3 compliance, the published package includes an imsmanifest.xml file for version 1.3 and an imsmanifest12.xml file for version 1.2. Content published for one SCORM version can easily be used for the other by renaming the manifest files for the version.

To switch SCORM versions in published packages:

1. Rename **imsmanifest.xml** to either **imsmanifest12.xml** or **imsmanifest13.xml** depending on what the package is currently defaulting to. If you published for version 1.2, rename the imsmanifest.xml file to imsmanifest12.xml. If you published for version 1.3, rename the imsmanifest.xml file to imsmanifest13.xml.

2. Rename either **imsmanifest12.xml** to **imsmanifest.xml** to use SCORM version 1.2 or **imsmanifest13.xml** to **imsmanifest.xml** to use SCORM version 1.3.

1.4.3 Use Content through an LMS

When you launch an item from an LMS, how it progresses to completion depends on the type of content being launched.

1.4.3.1 Sections/Concepts

When you develop your content in the Developer, you organize the content into a hierarchical outline of modules and sections. Along with containing content items such as topics and assessments, the sections themselves may contain content in the form of a section concept. Both SCORM and AICC, however, require that each item in the tree be either a content item (SCO or AU) or have children, but not both. When publishing for LMS deployment, section concepts are published as the first child of the section. In other words, a section with a concept containing two topics as children appears in the LMS as having three children; that is, the concept followed by the two topics. The section concept is marked complete as soon as it is selected from the LMS.

1.4.3.2 Topics

When a topic is launched from the LMS, the concept and available play modes are shown. The topic is considered started and the completion status is set to "incomplete". If the user launches and completes the required mode for the topic, the completion status is set to "completed" for required modes other than Know It?. If the required mode is Know It?, then the completion status is set to either "passed" or "failed" depending on the score received. If the required mode is completed, the topic is marked complete whether the user has played the topic in any other mode or not.

If the user navigates away from the topic before completing it, the completion status remains "incomplete" and the exit state is set to "suspend". The user can later come back to the topic, complete the required mode, and complete the topic.

Required Mode

When you publish content to the Player, you can specify which topic play modes are available to the user in an LMS. The options you choose affects which play mode(s) are optional and which mode is required for completion. If the required play mode is complete, the topic is marked complete whether the user has played the topic in any other mode or not.

Know It? mode is the highest priority mode, followed by Try It!, See It!, Print It!, and Do It!. The highest priority mode available is the mode required for completion.

Note: A topic can be published with no play modes available. In this case, the topic is completed when its concept is viewed.

1.4.3.3 Assessments

When a user selects an assessment in the LMS, a launch page appears. Depending on the previous completion status of the assessment, the launch page allows the user to start the assessment, to continue an incomplete assessment, or to retake a previously completed assessment. If a user continues an assessment that was left incomplete, only the remaining questions are presented. Questions that were previously answered maintain their previous answer values. The assessment itself does not begin until the user clicks the Start (Continue/Retake) button on the launch page. Therefore, until the assessment is launched, no change in completion status takes place. If the user navigates away from the assessment without launching it, no attempt at the assessment has been made.

Once the assessment is launched, its completion status is set to "incomplete". If the user navigates away from the assessment before completing it, the assessment remains incomplete and the exit state is set to "suspend". The user can later come back to the assessment, answer the remaining questions, and complete the assessment.

Since assessments are graded, a completed assessment is never actually be set to "completed", but rather to "passed" or "failed". Note that SCORM 2004 separates completion status from success status, and so under SCORM 2004 the completion status for the assessment is set to "completed", with its success status set to "passed" or "failed".

If a user retakes a previously completed assessment, the completion status and score of the new attempt is reported back to the LMS. If the user previously passed the assessment but failed it on retaking it, a failed status and score is sent to the LMS. It is up to the LMS implementation whether it shows the last attempt (failed) or the best attempt (passed) in the user interface.

Note: The SCORM 1.2 specification does not allow "not attempted" to be sent *to* the LMS, only for it to be received *from* the LMS. It also specifies that if no completion status is sent to the LMS, then "not attempted" is automatically changed to "incomplete". Therefore, if the user navigates away from an assessment prior to launching it, on a SCORM 1.2 LMS the completion status is set to "incomplete".

The order of question presentation, answer presentation, and required score for passing are all controlled by properties set in the Developer. These are not controlled by settings in the LMS itself.

1.4.3.4 Questions

When a question is selected in the LMS, the question is directly presented to the user. However, the completion status of the question remains "not attempted" at this point. If the user navigates away from the question before answering it, no attempt at the question has been made.

Note: The SCORM 1.2 specification does not allow "not attempted" to be sent to the LMS, only for it to be received from the LMS. It also specifies that if no completion status is sent to the LMS, then "not attempted" is automatically changed to "incomplete". Therefore, if the user navigates away from a question prior to answering it, on a SCORM 1.2 LMS the completion status is set to "incomplete".

When the user answers the question and clicks OK (or Do Not Know), the answer is judged (Do Not Know is automatically wrong). Since questions are graded, a completed question is never actually set to "completed", but rather to "passed" or "failed". Note that SCORM 2004 separates completion status from success status, and so under SCORM 2004 the question's completion status is set to "completed", with its success status set to "passed" or "failed".

1.4.3.5 LMS Groups

When you develop content in the Developer, each item in the outline is imported as a separate item in the LMS (section concepts are inserted as new first children within their section). You may, however, decide that you want the LMS reporting on a less granular level, on each section for example, rather than on each item in the section. The Player allows you to configure how much material gets grouped together into a single tracked LMS item.

In the Developer, you can set the Group for LMS property of a section (or module) to indicate that that section and all of its children should be grouped together into a single LMS item. You can make your entire course a single item, make each first level section separate items, or whatever arrangement you need. When you import the content into the LMS, the items marked as LMS Groups are imported as single items and the LMS has no specific knowledge of the children that make up that item.

When a user selects one of these items in the LMS, a launch page appears. When the item is launched, the Player opens in a separate window and allows the user to navigate within the children of the LMS item.

When you mark a section of your outline as an LMS Group, two sets of completion statuses must be kept: the status of the LMS Group itself, reported back to the LMS, and the status of each of the children in the outline, maintained by the Player. The completion status of the LMS Group is maintained very much like that of an assessment: if it is selected from the LMS but not launched, its completion status does not change; if it is launched, its completion status changes to "incomplete" and remains "incomplete" if the user navigates away without completing the LMS Group; if the user completes the LMS Group, its completion status changes to "completed".

Within the LMS Group, completion status for each of the children is maintained by the Player. To complete the LMS Group, the user must complete each of items in the group in the Player. The Player displays completion indicators for each of the items. If the user navigates away from the LMS Group before completing it, the completion status of children in the outline is saved as suspend data in the LMS and is restored when the user later returns to the LMS Group to continue.

Although scored items within the LMS Group (assessments, inline questions, and topics with Know It? mode available) display a pass or fail indicator in the outline when completed, the LMS Group as a whole is an unscored item and its completion status is either "completed" or "incomplete".

1.4.4 Using Debug Mode to Debug LMS Issues

The Player includes a debug mode to display the communications between the LMS and content and assist in debugging LMS deployment issues. If debug mode is enabled, and you launch an item from the LMS, a separate debug window also opens. Note that you may need to clear your browser cache. The debug window logs several events during the lifetime of the object launched.

Note: In some browser versions, the debug window may be empty when initially opened. If this is the case, simply refresh the window and the initial events are displayed. The window now automatically updates when additional events are logged.

To enable debug mode:

1. Open the config.js file in the root directory of the location where you published the content using any text editor such as Notepad.

2. Locate the following line in the file:

   DebugMode : false

3. Change the DebugMode value to true:

   DebugMode : true

4. Save the file.

1.5 Player Content on iPad

You can play content on iPad. Since iPad uses the iOS operating system, some adaptations were made to the Player.

- You cannot record content for iOS applications so all content will be of Windows-based applications. Recorded topics could have such steps as double-clicks, right mouse clicks, and keyboard events that have no equivalent on iOS such as "Press ENTER". Users should be aware of this.

- To help solve the problem mentioned in the previous bullet, special templates have been created for playback on iOS. These templates tell the user to "Swipe left to continue", rather than "Press ENTER to continue". In addition, for all unsupported actions such as mouse clicks and keyboard events, the user can advance using swipe left. Of course, the user can always tap in the action area for a step. The first time a user launches a topic in each session, a message displays instructing users to use swipe left to advance. This message only appears once.

- If an Adobe Acrobat (.pdf) file is attached to a concept in the Player outline, the file does not scroll on iPad. To enable scrolling, an Open in New Window link appears in the Player window. This opens the file in a new window, in which scrolling is enabled.

- Content can be installed in Knowledge Center and launched through the Player link. Usage is tracked. Also, if the content is installed in Knowledge Center and the user is directly launching the content from the content root, then usage is tracked.

- If the font for string inputs, bubble text, or web pages is MS Sans Serif or Tahoma, a font substitution is made automatically, as these fonts do not exist in an iPad environment.

- The Share feature works differently on iPad. If you navigate to a topic and use the Share feature, you get a link that when opened shows the topic page with the play mode buttons. However, if you navigate to the concept for a topic and use the share feature, you get a link that when opened shows the concept page and does not show the play mode buttons.

- To play sound on an iPad, the following MIME type must be registered on the server where the content is deployed. If the content is deployed on a Knowledge Center server, the MIME type is already present.

  .m4a audio/mp4

2 Launch Content from Other Applications

With the Player, you not only have the option of launching it with the full outline, you also have the option of launching a specific document in the outline. The document could be a module, a section, or a topic. If it is a topic, you can launch it in its own "player", to give the user a choice of which play mode to view, or you can launch it directly into a play mode.

The launch point is a link that you can create and embed in a web page, attach to a button in an application, or place in any document.

Copy the contents of the PlayerPackage folder in the published output to a web server and then create a hyperlink to a module, section, or topic in the content. The content should be placed on a web server for the links to work properly. Some things to note:

- Depending on the type of web server you are using, case sensitivity may be necessary. In this case, be sure to create the hyperlink with the appropriate capital letters.

- If you are launching a topic from certain document types such as Microsoft Word or Microsoft PowerPoint, the question mark <?> in the example link shown in the next section should be changed to a pound sign <#>.

- URL addresses should not contain spaces. You should use %20 to indicate any spaces.

Warning! Note that if the output file structure of a published package changes in future versions of the Developer, the hyperlinks will be broken. You will need to update them manually.

2.1 Launch a Document

You can launch a module, section, question, or assessment directly through a link. You can also launch the concept of a topic, instead of launching the topic into a play mode.

Create a hyperlink consisting of:

<server name>
The name of the Web server.

<virtual directory name>
The name of the virtual directory where the content is located on the Web server.

index.html
A page that causes the topic to launch in a separate window in the browser. Without this page, the topic opens in the current browser window, and when the user finishes playing the topic, the browser window closes.

Guid=
The document ID of the item.

Example:
http://<server name>/<virtual directory name>/index.html?Guid=04f9d570-8966-4170-b38c-3d399e38a948

You can create the hyperlink in two ways. One way is simply by typing it; however, there is a "shortcut" that can save you some time. Within the folder where you published the content, there is a file called kp.html. When you open this file in your browser, it contains the outline structure and provides a list of all the outline items, with links next to them. You can use these links to create the hyperlink.

Note: If you use this method, a back parameter is automatically added to the hyperlink. This parameter causes the browser to navigate back one page in the browser history to the page before the gateway page. It is optional.

To create a link by copying:

1. In Windows Explorer, navigate to the location of the published content.

2. Open the folder containing the published content.

3. Open the **kp.html** file in your browser.

4. Right-click the **Concept** link next to the module, section, or topic, or the **Question** or **Assessment** link next to the question or assessment.

5. Select the **Copy Shortcut** command from the pop-up menu.

6. Navigate to the document or location where you want to create the link.

7. Paste the link into the file or location.

8. Edit the link as necessary to refer to the correct URL. Note that by default, the link uses a file protocol, so you should update it to support the virtual directory location.

2.2 Launch a Topic

You have a choice if you want to launch a topic from a link. You can launch a topic in its own Player with all play modes available, or you can also launch a topic into a specific play mode. See the previous section to launch a topic concept.

Create a hyperlink as shown in the previous section and append the following parameters to the link:

Mode=
Indicates the mode in which the topic will play.

T = Try It! Mode
S = See It! Mode
K = Know It? Mode
D = Do It! Mode
E = Test It! Mode
P = Print It! Mode

Frame=
Optional parameter for the ID of a frame if you want play to start with a specific frame instead of from the beginning.

An example hyperlink to launch a topic in Try It! mode:

http://<server name>/<virtual directory name>/index.html?Guid=04f9d570-8966-4170-b38c-3d399e38a948&Mode=T

You can create the hyperlink in two ways. One way is simply by typing it; however, there is a "shortcut" that can save you some time. Within the folder where you published the content, there is a file called kp.html. When you open this file in your browser, it contains the outline structure and provides a list of all the outline items, with links next to them. You can use these links to create the hyperlink.

Note: If you use this method, a back parameter is automatically added to the hyperlink. This parameter causes the browser to navigate back one page in the browser history to the page before the gateway page. It is optional.

You should not use the links in kp.html to play topics or specific playback modes, as some features may not work. All playback should be from a server.

🖰 To create a link by copying:

1. In Windows Explorer, navigate to the location of the published content.

2. Open the folder containing the published content.

3. Open the **kp.html** file in your browser.

4. Right-click the **Topic** link or the link for a specific play mode next to a topic.

5. Select the **Copy Shortcut** command from the pop-up menu.

6. Navigate to the document or location where you want to create the link.

7. Paste the link into the file or location.

8. Edit the link as necessary to refer to the correct URL.

3 Deploy an HTML Web Site

The HTML Web site is a very simple set of HTML pages with the steps to perform the transaction but without the high interactivity of the other deployment formats.

When you publish content an HTML Web site package, the result is an HTMLPages folder. This folder contains an index.html file which is the main page; it contains the outline you chose to publish with links to the topics. The keywords.html in this folder lists all keywords for the published content and contains links to the associated topics. Below this folder, there is a tpc subfolder which contains a subfolder for each topic and attachment in the content you chose to publish. There are .html files that display all the steps for a particular topic, as well as .html files for each frame of each topic and all attachments. There are also .png files for each topic frame bitmap and each attachment.

Note: The HTML Web Site can be included when you publish a Player package. If a user has a non-supported browser, the HTML Web Site launches instead.

To make published content available to your users via the Web, copy or move the entire file structure to your Web server. Make sure that the upper and lower case letters of all the file and folder names are preserved in the copy exactly as they were created by the publishing process. Then, create a link from your existing Web page to index.html or default.htm.

When users access the main page, they can click the links to display the concept for a module, section or topic, or they can click the topic name to display a list of all the steps for the selected topic. Users can click the Start Topic link at the bottom of the page to display the first step, or click the Step # link below the line of instruction text to move immediately to the indicated frame. On a frame page, users see the appropriate screenshot with the action area indicated by a thick red border. The frame's instructional text appears below the screenshot. To advance through the topic, users can click the action area or the Next Step link located below the frame's instructional text. Further, any Alternative Actions or Alternative Paths are described below the primary action. Users can also press ALT+Left Arrow or click the browser's Back button to return to the previous screen. There are also links at the bottom of the page to return to the Table of Contents or the Topic Overview page.

Note: The HTML Web site is designed to meet 508c compliant requirements for accessibility. The site meets the following requirements: it can be read by a screen reader such as JAWS; it is not color dependent; the font size can be changed by the user; it is fully functional without using a mouse; and does not depend on an audio track for functionality.

4 Deploy Presentation Content

The Presentation output allows the users to view and follow the steps of a transaction through a Microsoft PowerPoint slideshow. The user interacts with the slideshow to advance through the steps.

When you publish content for a Presentation, the result is a Presentation folder. The index.html file in this folder is the main page; when you launch it, it displays the outline you chose to publish, with links to the Microsoft PowerPoint presentations for each topic. A user can access the content by launching the index.html page or play.exe. This page displays a table of contents page showing the outline. If a concept exists for a module, section, or topic, a Concept link appears next to the name. The names of the topics are links which launch the Microsoft PowerPoint presentation for that topic in the browser window. Each frame is a separate slide in the presentation. The slide contains a graphic showing the screenshot, with the bubble text next to it. A box on the screenshot indicates the action area or areas for the frame. To navigate through the topic, users can either click in the action area or move to the next slide by clicking the window or pressing PAGE DOWN. If a frame has attachments, the attachment icons will appear above the bubble text. Users can click the icons to access the attachments, if the presentation is being viewed online. If the presentation is printed, the links are broken. Alternative Actions, Alternative Paths, and Decision frames are included as well, and links are provided to choose paths.

The Presentation folder in the published output also contains separate Microsoft PowerPoint files for each topic, which can be launched individually in Microsoft PowerPoint.

To make the content available to your users, copy or move the Presentation folder to a Web server or network location and provide a link to the index.html. You can also make slideshows from the Microsoft PowerPoint files and post them in a central location or distribute them through email.

5 Player Cookies

The following cookies are stored in your browser when viewing content through the Player.

Note: These cookies will expire after 1 year unless otherwise noted. If cookies are disabled in the browser, these preferences will not persist past the browser session. All cookie values are stored as plain text.

OnDemandToc
Stores the Player table of contents last window location.

- Toc Left: Location of the left side of the Player table of contents
- Toc Top: Location of the top of the Player table of contents
- Toc Width: Width of the Player table of contents
- Toc Height: Height of the Player table of contents.
- Separator: Location of the splitter between the Concept and Outline panes

Example: OnDemandToc = TocLeft:690&,
TocTop:229&,TocWidth:590&,TocHeight:722&,Separator:242;

OnDemandPlayer
Stores the user editable Player preferences.

- TimeStamp: Used internally to manage when to load values from the cookie
- Play Audio: Determines which type of audio plays
- ShowLeadIn: Show introduction text
- MarqueeColor: Determines the Marquee color
- DoIt_HotKey_Ctrl: Key combination for advancing in Do It! Mode
- DoIt_HotKey_Shift: Key combination for advancing in Do It! Mode
- DoIt_HotKey_Alt: Key combination for advancing in Do It! Mode
- DefaultPlayMode: Determines the default play mode

Example: OnDemandPlayer = TimeStamp:20091001539&, PlayAudio:all&, ShowLeadIn:all&,
MarqueeColor:blue&, DoIt_HotKey_Ctrl:L&, DoIt_HotKey_Shift:N&,
DoIt_HotKey_Alt:L&,DefaultPlayMode:T&;

GICookie
Stores the Do It! mode last window location.

- Toc Left: Location of the left side of the Do It! window
- Toc Top: Location of the top of the Do It! window
- Toc Width: Width of the Do It! window
- Toc Height: Height of the Do It! window
- Csss: Determines if the Do It! window shows the screenshot
- Cs2: Location of the Do It! window splitter

Example: GICookie = Cleft:1206&, Ctop:242&, Cwidth:340&, 493&, Csss:1&, Cs2:215

OnDemandPlayerRoles
Stores the Role-based filtering preferences.

- Filtering: Determines if Role-based filtering is on or off
- Roles: Lists the roles being filtered

Example: OnDemandPlayerRoles = Filtering:true&, Roles:abc+pdq+xxx;

Bookmark_(toc-hash)
Stores the last item in the Player outline the user was viewing. This is only set when not running under an LMS.

- Location: The ordinal number of the current document in the Player outline. The ordinal number of a document is its position in the outline, starting at 1, disregarding indentation levels.

Retention: 30 days
Example: Bookmark_v3yCrNVpMN5Ip7AmY+bCxVDyHOc=Location:5

PlayerStatusBarState
Determines if the LMS connection status bar should be shown.

Example: PlayerStatusBarState = PlayerStatusBarOpen:true

HemiCookie
Display debug information on the gateway page.

Example: HemiCookie=hemiparamtest:1;

Possible values: 0 or 1
0 – do not show debug information
1 – show debug information

UPKDpiDailyWarning
Persists whether to suppress the browser DPI warning message.

Retention: 1 day
Example: UPKDpiDailyWarning=DPIdailyWarning:1;

chkcookie
Cookie used to check if cookies are enabled in the browser.

Example: chkcookie=1288639798984