Microsoft® Class Server

Learning Resource Development Guide

Version 4.0

Documentation

Contents

1 Introduction

Learning Resources are educational content that teachers can assign with Microsoft Class Server. This guide covers the Learning Resource formats (LRM, IMS, and IMS+) supported by Class Server.

This document is intended for use by the following Class Server audiences:

- Learning Resource publishers who want to publish to the LRM, IMS, or IMS+ format.
- Software developers who want to create utilities for reading or writing Learning Resources.

This document assumes that the reader is already familiar with using Class Server, XML, DHTML, and JScript.

1.1 Finding Further Information

December 13, 2006 – This and later versions of the Learning Resource Development Guide are provided as a Class Server file format reference for the SharePoint Learning Kit community. Some links and references in this document may be out of date.

The Class Server software development kit (SDK) can be found on the Web at http://ClassServer.msn.com/SDK/Install/.

The *Class Server Integration Guide* and *CSData Overview* are part of the SDK.

Information on installing and configuring Class Server can be found in the *Class Server Administrator's Guide* on the Class Server CD or on the Web at http://classserver.msn.com/Support/Doc

Information about becoming a Class Server partner can be found on the Web at http://www.microsoft.com/education/partner/?ID=ClassServerPartners.

The Class Server Online Discussion Forum can be found on the Web at http://groups.msn.com/ClassServerUsers/_homepage.msnw?pgmarket=en-us.

For more information on publishing Learning Resources for Class Server, send email to: cspubs@microsoft.com.

2 Format Overview

Class Server supports three formats: the industry standard Learning Resource format (IMS), Class Server's Learning Resource format (LRM), and the enhanced IMS+ format. IMS+ combines the advanced functionality of LRM with the open interoperability of IMS. Microsoft recommends authoring Learning Resources in the IMS+ format. This document focuses on the LRM format and the mappings used to create IMS+ from the LRM format.

2.1 IMS Format

Support for the IMS format was introduced in Class Server 3.0. IMS Learning Resources generally consists of a descriptive XML metadata file and a collection of Web pages. IMS files are packaged using the standard ZIP format and the IMS manifest conforms to the *IMS Content Packaging version 1.1.2* and *Metadata Standard version 1.2.1* specifications. For information on IMS, see http://www.imsglobal.org.

The Class Server implementation of IMS also includes support for the Sharable Content Object Reference Model (SCORM) version 1.2, an API defined by Advanced Distributed Learning at http://www.adlnet.org.

2.2 LRM Format

The LRM format is the native format for Class Server and offers features not available in the IMS format. Like IMS, the LRM format generally consists of a descriptive XML metadata file and a collection of Web pages. However, LRM is a richer format that enables features not available in IMS, such as assessment markup, curriculum standards, and publishing features.

LRM Learning Resources are stored in a multi-part MIME message with an .lrm file extension and are generally opened and edited using Class Server.

2.3 IMS+ Format

The IMS+ format is an application profile of IMS that offers the advantages IMS and LRM formats. The IMS+ format supports all the advanced features of LRM in Class Server while also providing compatibility with other IMS compatible Learning Management Systems.

Learning Resources exported from Class Server – Teacher are in IMS+ format by default. Learning Resources may come into Class Server in native LRM or IMS format, be edited to add certain LRM capabilities, and then be exported in the IMS+ format.

For a scaled publishing solution, you may use the mappings described in this document as a model of how Class Server represents its data in the imsmanifest of the IMS format. Some publisher-specific LRM capabilities are not available in Class Server editing, and require using the format details provided in this document.

2.4 Features of LRM and IMS+

LRM and IMS+ offer unique features in Class Server that go beyond those offered by IMS alone, such as assessment markup, curriculum standards, and publishing features. These features are described below.

2.4.1 Assessment Markup

The LRM and IMS+ formats support assessment markup. Assessment markup is specialized HTML syntax that is interpreted by Class Server and expressed as Web content with specialized behaviors. Assessment markup enables automated online assignments with questions. Learning Resource authors can specify questions and grading information, students can enter answers in a Web browser, and teachers can review and grade student answers with rubric and autograding assistance.

Assessment markup types include single and multiple answers, multiple choice, matching questions, essays, and file attachments. Grading for simple answer types can be automatic (autograding), and for other answer types may be assisted with grading rubrics.

The chart below highlights the utility of assessment markup in the lifecycle of an assignment:

Assignment Step	IMS Learning Resources	LRM and IMS+ Learning Resources
Completion by Student	Students may view the assignment in a Web browser, but they must submit their answers separately.	Students may enter answers into an assignment while viewing it in a Web browser. Their answers are posted and stored in Class Server using assessment markup.
Grading by Teacher	Teachers manually grade the papers, and then enter scores and comments manually into Class Server or a grade book. Teachers then return the papers to the students.	Teachers may use the Learning Resource and have autograding and rubric grading capabilities. Teacher may add inline comments for students, if they wish.
Review by Student	Students review their results on paper.	Students review teacher comments and autograding results online.
Reporting by Administrator	Data cannot be retrieved without a separate teacher submission process.	Grades and curriculum standards are stored in Class Server for back up and reporting purposes.

Assessment markup can be easily edited in Class Server's Learning Resource Editor if it is organized in specially formatted question tables. Question tables are used to add or delete a question, change a question's point total, or alter the answer key or scoring rubric.

Assessment markup may also be hidden from view for the student, the teacher, or both. This is useful when reporting results from complex content, such as content created with Macromedia Flash.

2.4.2 Curriculum Standards

Class Server allows teachers and publishers to align LRM and IMS+ Learning Resources with curriculum standards and then assess their students against those standards. Curriculum standards in LRM and IMS+ are copied in from a comprehensive curriculum standards XML file specified by a school district. For more information on the format for curriculum standards XML, see the *Class Server Integration Guide* in this SDK.

Once curriculum standards are aligned to Learning Resources, progress reporting is enabled. For example, it may be determined which standards are not being met by current curriculum, and where students are falling behind. It is also possible to administer standardized tests from a computer lab and receive reports against school district standards. Information on aligning Learning Resources with curriculum standards is detailed later in this document.

2.4.3 Publishing Learning Resources

Learning Resources authored in LRM and IMS+ support advanced authoring and distribution capabilities. LRM and IMS+ features of interest to Learning Resource publishers are as follows:

- **Licensing:** Learning Resource licensing allows users to preview Learning Resources, but prevents use without a valid school license. For security reasons, licensing is generally used with the LRM format, but not the IMS+ format, so that licensing cannot be bypassed through another learning management system. Learning Resource licensing is detailed later in this document.

- **Remote content:** Remote content is a feature that allows publishers to host Learning Resources directly from a Web site while enabling Class Server integration. For example, remote content can be assigned in Class Server, and then student results can be reported back to Class Server. However, the Learning Resource is being viewed and graded from the publisher's server. Please contact cspubs@microsoft.com for more information.
- **Distribution:** Learning Resource distribution can occur through CD-ROM or Web site. Class Server provides assistance and methodologies for getting Learning Resources from your system to a school curriculum library. Please contact cspubs@microsoft.com for more information.

Class Server also has partners that can help with the entire process of working with the LRM and IMS+ formats. Please contact cspubs@microsoft.com for more information.

3 LRM Structure and Format

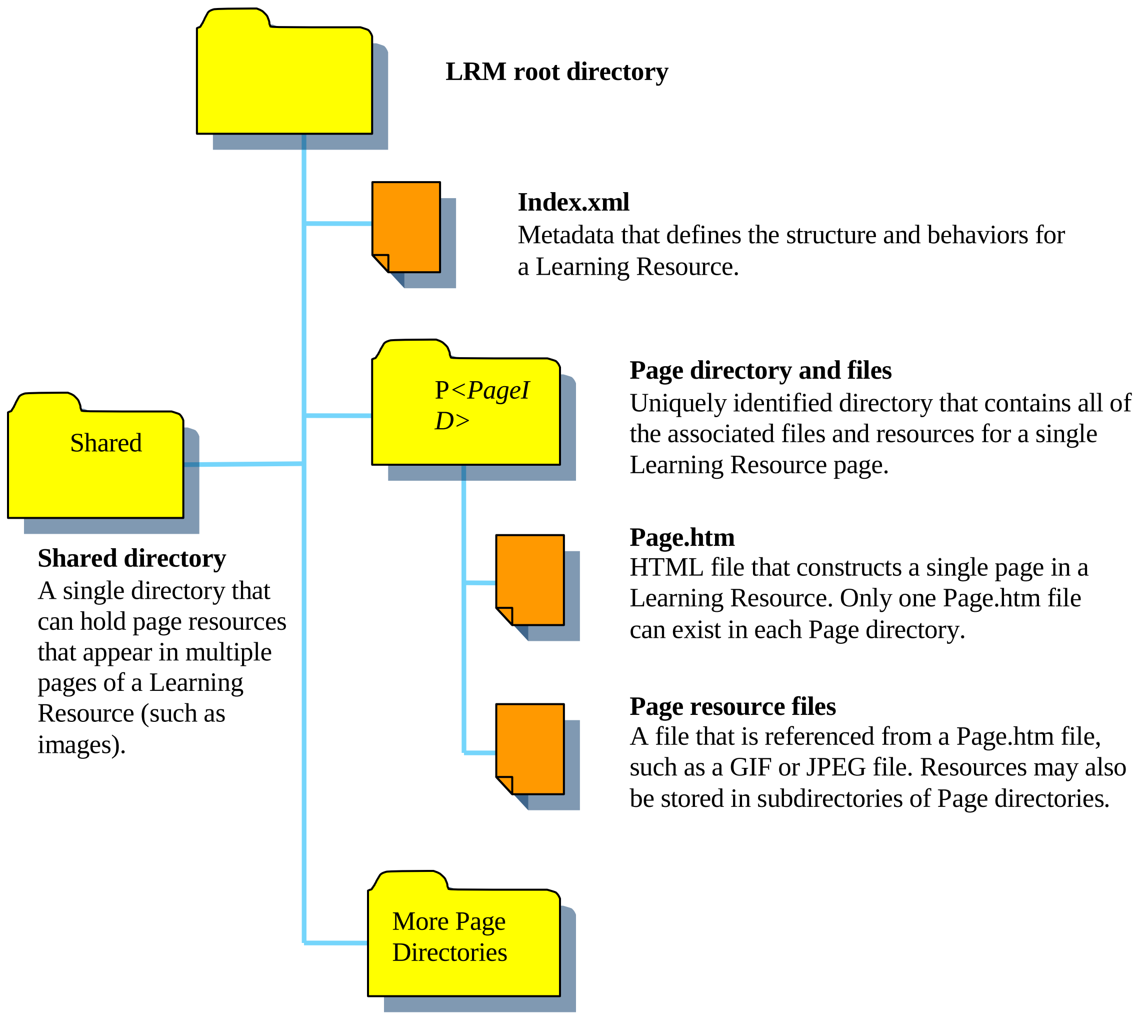
This section describes the LRM format, including the directory structure, the Index.xml manifest file, pages with assessment markup, and licensing. You want to be familiar with the LRM format if you are creating Learning Resources in .lrm or .ims file formats.

The IMS+ format supports all the advanced features of LRM in Class Server. The IMS+ format is essentially the LRM format plus the IMS format in a common folder. You can determine the IMS+ format from the LRM format by looking at notes through this section, and by exporting Learning Resources from Class Server. Content created in Class Server is exported in IMS+ format by default.

3.1 LRM Directory Structure

The LRM format supports browser-compatible content (Learning Resources) running from any location. For example, you can stream large media files or administer assessment content from a Web site. However, the default directory structure for LRM is required if you want teachers to be able to edit a Learning Resource.

The default LRM directory structure contains all files needed by the LRM, except streaming media or linked Web pages. The graphic below shows the default LRM structure.


LRM root directory
Index.xml
Metadata that defines the structure and behaviors for a Learning Resource.
P<*PageID*>
Page directory and files
Uniquely identified directory that contains all of the associated files and resources for a single Learning Resource page.
Shared
Shared directory
A single directory that can hold page resources that appear in multiple pages of a Learning Resource (such as images).
Page.htm
HTML file that constructs a single page in a Learning Resource. Only one Page.htm file can exist in each Page directory.
Page resource files
A file that is referenced from a Page.htm file, such as a GIF or JPEG file. Resources may also be stored in subdirectories of Page directories.
More Page Directories

All files in the LRM directory should be less than 2 MB in total size, to prevent warnings and performance issues in Class Server. Streaming large media files from a Web site is one way to bypass this limit. In addition, the relative paths to the files in the LRM directory must be less than or equal to 100 characters.

LRM directories are shared between computers using the .lrm, .ims, or .zip formats. These formats bundle or package all the files in the LRM directory into a single file (with a .lrm, .ims, or .zip file extension). For information on these formats, see *Learning Resource Bundling Formats* later in this document.

3.1.1 LRM Root Directory

Each LRM has a root directory. This directory contains the *Index.xml* manifest file and the subdirectories described below. The *Index.xml* file contains metadata for the Learning Resource, and it describes the structure of the Learning Resource (for example, a list of pages).

The LRM root directory may also contain an *IMSManifest.xml* file in the case of IMS+ Learning Resources. IMS+ content may also contain other directories and files not described below.

3.1.2 Page Directories

In the root directory of an LRM there is a directory for each page of the Learning Resource. Each directory contains the files for one page of the Learning Resource, and the directory is named P<*PageID*>, where <*PageID*> is the PageID attribute corresponding to the page. The page directory contains a Page.htm file and any supporting files (e.g., images, style sheets, etc) required to render the page. Page.htm files contain the necessary information to create a specific page of a Learning Resource. In addition to standard HTML markup, Page.htm files contain assessment markup, which is question-and-answer-key components that enable Class Server to store and grade student work automatically.

Files used by Page.htm (such as .css, .js, and other media files) are also included in the page directory. The directory may also contain subdirectories with files used by Page.htm. It is important that Page.htm use only relative paths to refer to files or pages in the LRM directory.

The order of pages is determined by the order of <PAGE> nodes in the Index.xml file, not by the page ID numbers (for example, *P1012* may uniquely identify the first page in an LRM, and *P1003* may identify the second page). When the Class Server Learning Resource Editor is used to add or remove pages in an LRM, page ID's are not changed. This ensures that links between pages will continue to work when pages are sorted in the editor.

3.1.3 Shared Directory

The Shared directory holds files, such as images, that are shared across pages. Shared files can save storage space if the files are used across multiple Learning Resource pages. Shared files are not removed when pages are deleted from an LRM using the Learning Resource Editor.

The Shared directory also contains files common to the LRM, including the teacher notes file, the themes file, and the LRM icon. Some of these files are described in Index.xml.

3.2 Index.xml Structure and IMS Mapping

Index.xml contains metadata describing an LRM. IMSManifest.xml contains IMS metadata. Most of the contents of Index.xml map to similar data in IMSManifest.xml. Class Server performs this mapping automatically when importing and exporting Learning Resources. If both an Index.xml and IMSManifest.xml file are included in an IMS+ format Learning Resource that is imported into Class Server, only the Index.xml file is read.

The following sections list Index.xml nodes, a description of their attributes, and how their values map to values in IMSManifest.xml when Class Server exports Learning Resources in IMS+ format. For all attributes listed in this document, the attribute value "" should be interpreted as if the attribute were absent. A list of data types and a sample Index.xml file follow the listing of nodes and attributes.

Note: The list of IMS metadata described below is not complete and is only provided to aid understanding of how Class Server maps IMS metadata with Index.xml. Other IMS metadata can be used, but will be ignored by Class

Server. For complete information on IMS metadata, see the *IMS Content Packaging and Metadata Standard* on http://www.imsglobal.org.

All nodes and attributes that are not explicitly marked as optional are required.

3.2.1 LearningResource (Root Node)

Attribute	Description	IMSManifest.xml Written by Class Server
Version required float attribute	Software that reads LRMs should expect Version to be "1.0", "2.0", or "3.0". If "1.0", then the LRM can be loaded in all versions of Class Server. If "2.0", then the LRM can be loaded into Class Server 2.0 or later. If "3.0", then the LRM can be loaded into Class Server 3.0 or later. Currently, the only legal values are "1.0", "2.0", and "3.0". Use "3.0" if the Learning Resource includes Curriculum Standards alignment. Use "2.0" if the Learning Resource takes advantage of licensing features. Use "1.0" if the Learning Resource does not use licensing or standards alignment features.	None
xmlns required namespace attribute	"urn:schemas-microsoft-com:learning-resource"	None

3.2.2 PackageDescription/General (Node of LearningResource)

Attribute	Description	IMSManifest.xml Written by Class Server

Attribute	Description	IMSManifest.xml Written by Class Server
Keywords optional string node	Words or phrases that describe the contents of the Learning Resource to distinguish it from other Learning Resources. Adjacent keywords or phrases are separated by commas (for example, "Canada, United States, North America" contains three keywords: "Canada," "United States," and "North America"). Length: <= 255 characters	One or more instances: **general.keyword** (langstring†)
Language optional string node	The country code for the language of the LRM (for example, "en" for English). This is the Win32 LCTYPE LOCALE_SABBREVLANGNAME that the Windows operating system uses. This element is required. Length: <= 5 characters	**general.language**
Title required node	The title of the LRM. Note: Do not include line breaks in this element. Length: <= 255 characters	**general.title** (langstring†)

† langstring, datetime and vcard datatypes are described in the IMS specifications (http://www.imsglobal.org)

3.2.3 PackageDescription/Language (Node of LearningResource)

The country code for the language of the LRM (for example, "en" for English). This is the Win32 LCTYPE LOCALE_SABBREVLANGNAME that the Windows operating system uses. This element is required.

Length: <= 5 characters

IMSManifest.xml written by Class Server: **general.language**

3.2.4 LearningResourceData (Node of General)

Attribute	Description	IMSManifest.xml Written by Class Server

AgeMax optional int attribute	The maximum age level for which the content is appropriate. The age levels are in the Mapping Age to Year/Grade table later in this document.	**educational.typicalagerange**
AgeMin optional int attribute	The minimum age level for which the Learning Resource is appropriate. The age levels are in the Mapping Age to Year/Grade table later in this document.	**educational.typicalagerange**
Assessment optional string node	Name of the standard assessment with which the Learning Resource is aligned. For example, “Virginia Standards of Learning Assessments: Computer Technology, Grade 5”. Length: <= 255 characters	None
Attachment optional string attribute	The path to a file containing notes or instructions for teacher use pertaining to the Learning Resource. The file must be in the Learning Resource’s Shared folder, and should contain the path relative to that folder. For example, “SubfolderOfShared/TeacherNotes.doc”. Length: <= 100 characters	A **manifest.resources** element with identifier=”CSAttachment” and href=<path to attachment>
Author required string attribute	The author’s name (for example, Ben Smith). Length: <= 255 characters	An element in **lifecycle.contribute** (vcard†) where .role=”Author”
AuthorEmail optional string attribute	The e-mail address of the author (for example, someone@microsoft.com). Length: <= 255 characters	An element in **lifecycle.contribute** (vcard†) where .role=”Author”
BloomTaxonomy optional enumset attribute	Bloom's Taxonomy attributes of the LRM. This attribute may contain zero or more of the following: "Knowledge|Comprehension|Application|Analysis|Synthesis|Evaluation" Length: <= 255 characters	None
BrowseOnly optional bool attribute of PublisherURL node	If true (1), data is not posted to the PublisherURL link. See the PublisherURL node description for more information.	None
ChapterName optional string attribute	Name of the chapter of the book or other curriculum series member to which the Learning Resource belongs. Length: <= 255 characters	None

Copyright optional string attribute	Copyright information for the Learning Resource (for example, "Copyright (c) Name of copyright holder"). The "(c)" notation and the copyright dates, if any, should be included in the string. Length: <= 255 characters	Using **rights.copyrightandotherrestrictions,** set: **value.source** = "LOMv1.0" **.value.langstring** = "yes" and the copyright string written to **rights.description** (langstring†)
CurriculumSequence optional float attribute	The sequence number (for example, 4.2) of the Learning Resource in the curriculum series. This is used primarily as a sort key (for example, to order the display of Learning Resources in a list).	None
CurriculumSeries optional string attribute	Name of the curriculum or textbook series to which the Learning Resource belongs. Length: <= 255 characters	None
DateCreated required date attribute	The date and time the Learning Resource was created. See *Data Types* for information about the format of this value.	**metametadata.contribute** record with **.role.source** = "LOMv1.0" **.role.value** = "Creator" **.date.datetime** = *DateCreated*, ISO 8601 formatted
DateModified required date attribute	The date and time the Learning Resource was last modified. See *Data Types* for information about the format of this value.	A new **lifecycle.contribute** element with **.role.source** = "LOMv1.0" **.role.value** = "Unknown" **.centity.vcard** = vcard for *LastModifiedBy* **.date.datetime** = *DateModified*, ISO 8601 formatted
GardnerMultipleIntelligences optional enumset attribute	Gardner's Multiple Intelligences attributes of the Learning Resource. This attribute may contain zero or more of the following: "Verbal-Linguistic|Logical-Mathematical|Visual-Spatial|Body-Kinesthetic|Musical-Rhythmic|Interpersonal|Intrapersonal|Naturalist" Length: <= 255 characters	None
Icon optional string attribute	The name of an image file that can be used to represent the Learning Resource (for example, Thumbnail.gif). The associated file should be in the Learning Resource's Shared directory. The icon image size is 86 x 59 pixels. Templates display the icon in the Learning Resource Editor. Note: Class Server does not currently use this attribute. Length: <= 255 characters	A **resource** in **manifest.resources** with **.identifer** = "CSIcon" **href** = path to the icon file
Instructions optional string node	Instructions for students assigned the Learning Resource. Length: <= 4096 characters	**educational.description** (langstring†)

IsLocked optional bool attribute	If true (1), a user will not be able to open the Learning Resource in the Learning Resource Editor. This should be set to true if there are multiple object tags on any page in the Learning Resource. See the Embedded Objects information later in this document.	None
IsPublishable optional bool attribute	If true (1), Class Server can enable a feature that helps users freely share this Learning Resource over the Internet. If false (0, or not present), Class Server will not attempt to enable the Internet sharing feature. Content may still be shared among users of the same server and may be shared through other means such as ftp. This attribute blocks Internet sharing only from schools. The Learning Resource can still be explicitly shared by a publisher to Class Server Internet Find, but cannot be shared out of a school.	None
IsTemplate optional bool attribute	If true (1), this Learning Resource can be used as a template to create Learning Resources in Class Server. Learning Resource templates are displayed when opening the Learning Resource Editor to create a Learning Resource.	None
KeyStage optional int attribute	Key stage in the United Kingdom curriculum (a value from 1 to 5). It can be useful in searching for Learning Resources.	Find or create a **classification** element where • **classification.purpose.value** = “Educational Level”, *and* • **classification.taxonpath.source.langstring** = “NC Metadata Standard: Keystage” The keystage is then written to the **classification.taxonpath.taxon.taxon.id** from this classification.
LastModifiedBy optional string attribute	The name of the person who last modified the Learning Resource. Length: <= 255 characters	Create a new **lifecycle.contribute** element with **.role.source** = “LOMv1.0” **.role.value** = “Unknown” **.centity.vcard** = vcard for *LastModifiedBy* **.date.datetime** = *DateModified*, ISO 8601 formatted
PointsPossible optional float attribute	The maximum number of points that a student can receive for completing the assignment using this Learning Resource. PointsPossible must be between 0 and 10,000.	None

Publisher optional string attribute	The Learning Resource publisher's name. Length: <= 255 characters	Find or create a **lifecycle.contribute** element where **.role** = "Publisher". Modify the **lifecycle.contriubte.centity.vcard** element to contain this information.
PublisherURL optional string node	The publisher's URL for obtaining additional information about the Learning Resource or about acquiring licensing. In Class Server – Teacher, when the URL is clicked and the **BrowseOnly** attribute on the node is false, values are posted for school name (**SchoolName**), School GUID (**SchoolID**), and Creation ID (**CreationID**). This data can then be used by the **PublisherURL** site to build a license file for the Learning Resource or to direct the teacher to related Learning Resources. The publisher's URL must begin with http:// or https://. Length: <= 2048 characters	Find or create a **lifecycle.contribute** element where **.role** = "Publisher". Modify the **lifecycle.contriubte.centity.vcard** element to contain this information.
RegionID optional int attribute	The ID of the region for which this Learning Resource was intended. Valid RegionIDs are in the Regions table later in this document.	None
RelatedStandard optional string attribute	Free form text description of the related curriculum benchmark objective, or standard with which this Learning Resource is aligned. Length: <= 255 characters.	None
Subject required string attribute	A subject from the <*SubjectTopicList*>_<*RegionID*>.xml file (for example, GEM_1.xml) located in the Class Server - Teacher installation directory. <*RegionID*> is the RegionID for the Learning Resource. Length: <= 255 characters	Modify or create a **classification** element with **.purpose** = "Discipline" **.taxonpath.source** = *SubjectTopicList_RegionID* **.taxonpath.taxon.entry** = *Subject* **.taxonpath.taxon.taxon.entry** = *Topic*
SubjectTopicList required string attribute	This attribute should be set to "GEM". See the **Subject** and **Topic** attributes for more information.	None
SuitableFor optional enumset attribute	Instructional setting for which the Learning Resource may be suitable. This attribute may contain zero or more of the following: "Individual|Small group|Class".	None

Topic required string attribute	A topic from the <*SubjectTopicList*>_<*RegionID*>.xml file belonging to the defined **Subject** attribute. Length: <= 255 characters	Modify or create a **classification** element with **.purpose** = "Discipline" **.taxonpath.source** = *SubjectTopicList_RegionID* **.taxonpath.taxon.entry** = *Subject* **.taxonpath.taxon.taxon.entry** = *Topic*
Type required string attribute	Type of Learning Resource (for example, worksheet, quiz, or assessment). Length: <= 255 characters	None
UnitName required string attribute	Name of the curriculum unit to which the Learning Resource belongs. Length: <= 255 characters	None

† langstring, datetime and vcard datatypes are described in the IMS specifications (http://www.imsglobal.org)

3.2.4.1 Regions

Regions are geographical areas that share a common language (such as English) but have slight variations in educational terminology. For instance, in the United States, a score provided to a student for an assignment is a *grade*, whereas in the UK, it is a *mark*. The following table lists the regions and their corresponding RegionIDs used within the English version of Class Server 4.0. The Class Server version column indicates which version of Class Server the region was first available. Other language releases and future versions of Class Server may contain other valid RegionIDs.

Region	RegionID	Class Server version
Australia	4	1.0 and later
Canada	2	1.0 and later
England	3	1.0 and later
Ireland	10	4.0 and later
New Zealand	9	4.0 and later
Northern Ireland	6	3.0 and later
Scotland	8	3.0 and later
Singapore	5	3.0 and later
United States	1	1.0 and later
Wales	7	3.0 and later

3.2.4.2 Mapping Age to Grade or Year

AgeMin and AgeMax attributes in Index.xml are mapped to regionalized grade or year levels when displayed in Class Server - Teacher. The region is configured on the school server. The following table shows some of the regions for the English product and their mapped grade or year level. This table does not display all region information. Other languages and future versions of Class Server may contain additional regions mapped to their associated grade or year level.

Age	4	5	6	7	8	9	10	11	12	13	14	15	16	17
U.S./Canada: Grade	Pre-K	K	1	2	3	4	5	6	7	8	9	10	11	12
UK (England, Scotland, Wales, Northern Ireland): Year	R	1	2	3	4	5	6	7	8	9	10	11	12	13
Australia: Year	Pre-K	K	1	2	3	4	5	6	7	8	9	10	11	12
Ireland: Year	ES	Jl	Sl	P1	P2	P3	P4	P5	P6	PP1	PP2	PP3	PP4	PP5
New Zealand: Year	Pre	1	2	3	4	5	6	7	8	9	10	11	12	13
Singapore: Year	K1	K2	P1	P2	P3	P4	P5	P6	Sec. 1	Sec. 2	Sec. 3	Sec. 4	Sec. 5	Sec. 6

3.2.5 ID (Optional Node of LearningResourceData)

The ID node is used for tracking and comparing Learning Resources in Class Server.

Attribute	Description	IMSManifest.xml Written by Class Server
BranchID conditionally required guid attribute	A GUID assigned to the Learning Resource when a user who is not the creator of the Learning Resource modifies it. If a Learning Resource has never been branched then the **BranchID** should be the same as the **CreationID**. The **BranchID** is used to compare Learning Resources when they are opened in Class Server. This attribute is required if the element is present.	**general.identifier**
CreationID conditionally required guid attribute	A GUID assigned to the Learning Resource when it is created. This GUID does not change during the lifetime of a Learning Resource and can be used to uniquely identify the Learning Resource, even after it has been edited. This attribute is required if the element is present.	**metadata.identifier**

3.2.6 Format (Optional Node of LearningResourceData)

The Format node determines where and when the Learning Resource was modified. Publishers do not need to define this node.

Attribute	Description	IMSManifest.xml Written by Class Server
DateModifiedVersion conditionally required date attribute	Date/time that the Learning Resource was last modified. From Index.xml format version 2.0 and later, this attribute is identical to the **DateModified** attribute listed earlier in this document. However, both attributes must be present for correct operation. This attribute is required if the element is present.	None
VersionModifiedLast conditionally required float attribute	Version number of the Index.xml format for which it was written last, currently 3.0. When the Index.xml format changes, the version number is incremented. Then, depending on the utility last used to modify Index.xml, write the appropriate version number of the **VersionModifiedLast** attribute. The current Index.xml format version is 3.0. Note that other utilities may modify Index.xml. This attribute is required if the element is present.	None
VersionModifiedMax conditionally required float attribute	Maximum version number of the Index.xml format in which it was ever written. This attribute is required if the element is present.	None

3.2.7 Licenses/License (Optional Node of LearningResourceData)

For more information on licensing, see the *Licensing* section of this document.

The optional Licenses node (within LearningResourceData) contains zero or more License nodes.

If the Learning Resource contains a Licenses node, the LearningResource\Version attribute cannot be “1.0”.

Attribute	Description	IMSManifest.xml Written by Class Server
ID conditionally required guid attribute	A GUID that is used as a key to lock the content until a license file with a corresponding GUID activates the content. If no License nodes are specified, then the content is unlicensed and can freely be used in the classroom. As a matter of convenience we recommend that you use the **CreationID** GUID listed earlier for the first License ID. Licensing is not enforced in Class Server 1.0, but the **Version** attribute of the root **LearningResource** node can be incremented to prevent the use of Learning Resources in Class Server 1.0. This attribute is required if the element is provided. For more information about licensing, contact cspubs@microsoft.com.	None

3.2.8 Standards/Standard (Optional Node of LearningResourceData)

Curriculum standards are commonly presented as high-level, general concepts, followed by specific details. In Class Server, the specific details are referred to as benchmarks and skills. A benchmark or skill is a specific level of a standard that can be used to evaluate a student’s work. For example, *Add and subtract whole numbers* is a skill, *Computation* is its parent benchmark, and *Mathematics* is its parent standard.

The Class Server curriculum standards format is documented in this SDK in the *Class Server Integration Guide*. The Standard node that is described below is an abbreviation of a skill or benchmark in the curriculum standards format. A Standard node may be used in a school with a different curriculum standards document, but some capabilities are lost (for example, viewing the full description).

The Standards node is optional. Any number of Standard nodes may exist within the Standards node, though for performance and usability reasons, we recommend that there be no more than 100 Standard nodes in a Learning Resource.

If the Learning Resource contains a Standards node, the LearningResource\Version attribute must be "3.0".

Attribute	Description	IMSManifest.xml Written by Class Server
GUID conditionally required guid attribute	The GUID assigned to a particular Benchmark or Skill in the source curriculum standards document. A GUID may not be repeated in a Learning Resource. This attribute is required if the element is provided.	None
ID optional string attribute	A short identifier, for example, "17.4(a)". Length: <= 20 characters	None
Title optional string attribute	A short title, for example, "Add and subtract whole numbers." Length: <= 80 characters	None
Desc optional string attribute	A description, for example, "Student can perform addition and subtraction of positive integers." Length: <= 200 characters	None
PBID optional string attribute	The **ID** of the parent benchmark if the Standard node is a skill or an empty string if the Standard node is a benchmark. Length: <= 20 characters	None
PBTitle optional string attribute	The **Title** of the parent benchmark if the Standard node is a skill or empty string if the Standard node is a benchmark. Length: <= 80 characters	None
PSID optional string attribute	The **ID** of the parent standard. Length: <= 20 characters	None
PSTitle optional string attribute	The **Title** of the parent standard. Length: <= 80 characters	None

3.2.9 RemoteContent (Optional Node of LearningResourceData)

The RemoteContent node is used in place of the Organization node of the Learning Resource. If you want to implement a remote content Web site, please contact cspubs@microsoft.com.

Attribute	Description	IMSManifest.xml Written by Class Server

Attribute	Description	IMSManifest.xml Written by Class Server
href conditionally required string attribute	The URL to which RedirectRemoteContent.aspx will redirect, along with any publisher-specific parameters. This attribute is required if the element is present.	None

3.2.10 Technical (Optional Node of PackageDescription)

Attribute	Description	IMSManifest.xml Written by Class Server
Duration optional int node	The typical time it takes a student to complete a Learning Resource, measured in seconds (for example, "3600" refers to one hour and "86400" refers to one day).	**educational.typicallearningtime** (datetime†)

3.2.11 Classification (Optional Node of PackageDescription)

Attribute	Description	IMSManifest.xml Written by Class Server
EducationalObjective optional string node	Statement describing the learning objectives, aims, curriculum standards, or attainment targets with which this Learning Resource is aligned. Length: <= 255 characters.	The **classification.description** element of the first **classification** element whose **classification.purpose** = "Educational Objective".

3.2.12 Organization/TableOfContents/Page (Required Node of LearningResource)

There is one Page node per LRM page. The order of Page nodes is the order of pages in the LRM.

Attribute	Description	IMSManifest.xml Written by Class Server
PageID required int attribute	An integer (unique in this LRM) that identifies the page. The directory P<*PageID*> holds the contents of the page. 1 <= Value < 1999999999.	**organizations.item.identifierref** **resources.resource.identifier**
Title required string attribute	The title of the page. Length: <= 200 characters.	**organizations.item.title**
href optional string attribute	The URL or relative path of an IMS page to be used instead of P<*PageID*>/Page.htm. If this attribute is specified, then Page.htm files and assessment markup are not used. The **href** attribute is used by Class Server when importing IMS Learning Resources.	**resources.resource.href** **resources.resource.file.href**
ImsId optional string attribute	The resource identifier of the page in the IMS manifest. Class Server uses this value to coordinate data updates in Index.xml with IMSManifest.xml. Length: <= 255 characters.	None

† langstring, datetime and vcard datatypes are described in the IMS specifications (http://www.imsglobal.org)

3.2.13 Data Types

The Index.xml schema includes the following XML data types:

Data Type	Description
bool attribute	Represented as "0" (false) or "1" (true). If the attribute is not present, it is assumed to be false.
date attribute or node value	Represented by OLE DATE values. OLE DATE values are the number of fractional days since midnight, December 30, 1899, GMT (for example, Monday, July 17, 2000 12:48 PM, Pacific Daylight Time is represented by the number 36724.82566).
enum string attribute	Contains one of a fixed number of strings. These strings are treated as constant values and they are always represented in English in the XML file, even if they are translated in a user interface.
enumset attribute	The same as an **enum** attribute, except that the attribute value can contain zero or more strings from a fixed set. If multiple strings are present, they are separated with a "|" character (for example, "123|XYZ").
float attribute or node value	Contains a double-precision floating-point value represented as a string in decimal form using the "C" locale (for example, "3.14", not "3,14", even in Europe). It does not use thousands separators (for example, "1234.56" is valid, but "1,234.56" is not).
guid attribute or node value	Contains a guid of the form: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx. For instance, CCA7A565-3EE7-45AD-A075-2F05B25DEFD2
int attribute or node value	Contains a signed 32-bit integer (between -2147483648 and 2147483647, inclusive) represented as a string in decimal form without thousands separators (for example, "12345" is valid, but "12,345" is not).

3.2.14 Sample Index.xml

The following is a simple example of an Index.xml file that will be part of a Learning Resource that contains two pages:

```xml
<?xml version="1.0"?>
<LearningResource xmlns="urn:schemas-microsoft-com:learning-resource" Version="3.0" >
    <PackageDescription>
        <General>
            <Language>en</Language>
            <Title>Differentiation Practice #3</Title>
            <Description>This is a simple differentiation worksheet.</Description>
            <LearningResourceData DateCreated="36724.82566" DateModified="36725.83528"
              Author="Ben Smith" Publisher="Lucerne Publishing." AgeMin="15" AgeMax="17"
              SubjectTopicList="GEM" Subject="Mathematics" Topic="Calculus" Type="Worksheet”
                Unit=”Differentiation”>
                <Instructions>
                  The student needs to understand basic differentiation before
                  completing this assignment.
                </Instructions>
                <PublisherURL BrowseOnly=”1”>http://lucernepublishing.com/LR.asp</PublisherURL>
                <ID CreationID="53B48F16-22B7-443a-9410-0C78D175CBE2"
                  BranchID="53B48F16-22B7-443a-9410-0C78D175CBE2" />
                <Format VersionModifiedLast="3" VersionModifiedMax="3"
                  DateModifiedVersion="36725.83528" />
                <Licenses>
                    <License ID="53B48F16-22B7-443a-9410-0C78D175CBE2" />
                </Licenses>
                <Standards>
                    <Standard GUID="2B8F0390-0C3F-48BC-862E-A02A13FE20E3" ID=""
                      Title="Earth/Space Science, Hydrosphere/atmosphere."
                      Desc="Relate global atmospheric movement and the formation of ocean
                      currents to weather and climate."
                      PBID="1.3" PBTitle="The student understands and uses scientific
                      concepts and principles." PSID="" PSTitle="Science (5-7)" />
                    <Standard GUID="85528576-5044-4358-B8E3-CFF912EC6D16" ID=""
                      Title="Earth/Space Science, Interactions in the solar system and beyond."
                      Desc="Describe how the regular and predictable motions of most objects
                      in the solar system account for such phenomena as the day, year,
                      phases of the moon, eclipses, seasons, and ocean tides."
                      PBID="1.3" PBTitle="The student understands and uses scientific
                      concepts and principles." PSID="" PSTitle="Science (5-7)" />
                </Standards>
            </LearningResourceData>
        </General>
        <Technical>
            <Duration>3600</Duration>
        </Technical>
    </PackageDescription>
    <Organization>
            <TableOfContents>
                <Page PageID="1000" Title="Simple Differentiation"/>
                <Page PageID="1001" Title="Advanced Differentiation"/>
            </TableOfContents>
        </Organization>
</LearningResource>
```

3.3 IMS Additional Requirements

Learning Resources in IMS format which are imported in to Class Server must comply with the *IMS Content Packaging and Metadata Standard* as defined by the IMS organization, see http://www.imsglobal.org.

In addition to the minimum requirements outlined in the IMS standard, the following metadata is also required for Learning Resources imported into Class Server.

```
<metadata>
         <imsmd:lom>
         </imsmd:lom>
 </metadata>
 <organizations default="Testing">
     <organization identifier="Testing" structure="hierarchical">
         <item identifier="1001" identifierref="folder1">
             <title>Page 1</title>
         </item>
     </organization>
 </organizations>
 <resources>
     <resource identifier="folder1" type="webcontent" href="index.html">
         <file href="index.html" />
     </resource>
 </resources>
```

Specifically, Class Server requires a <metadata> node and must also have either a <lom> or <record> subelement of the <metadata> node. At least one <resource> node must also exist. This is in addition to having the manifest and organization node that IMS requires.

3.4 Page.htm and Assessment Markup

Most Learning Resource pages are authored as a P<*PageID*>/Page.htm file (unless specified otherwise in Index.xml). Page.htm files must be used to enable the assessment markup and editing features of the LRM format. Assessment markup provides online assessment delivery, autograding, and rubric grading features. This section describes rules and structures for creating and editing Page.htm files.

3.4.1 HTML Format Requirements

A Page.htm file is a UTF-8 encoded text file containing Hypertext Markup Language (HTML), which can be interpreted by a Web browser. However, there are a few features and restrictions that make Page.htm files unique. The following requirements apply when you are editing the HTML source code for Page.htm files.

- Page.htm must be encoded using the UTF-8 character set (defined in RFC 2279).
- The HEAD element is required, and should be placed between the <HTML></HTML> tags.
- The META element (specifying UTF-8 encoding) is required, and should be placed between <HEAD></HEAD> tags.
- All HTML content must be placed in the HEAD or BODY sections of the file.
- If the STYLE element exists, it must be placed in the HEAD section of the file. Inline styles can appear in the BODY section of the file.
- HTML element styles must not be overridden.
- FORM elements are prohibited.
- FRAMESET elements are prohibited.
- Links that have targets in the same page must explicitly specify target=”_self”. If a target is not specified, a link target appears in a new window with target=”_blank”.

- To navigate between pages of a Learning Resource without saving question answers, you may use links of the form:
  <A href="../P1001/Page.htm">...</A>.
- Special syntax and rules for constructing questions and hidden fields, as defined in subsequent sections of this document, must be obeyed.
- SCRIPT elements, and the scripts they contain, should be in the HEAD and BODY sections of the file.

3.4.1.1 Manipulating Page.htm with Scripts

Test content that contains scripts to ensure that it works properly in all Class Server views on all the browsers appropriate for the content. Script errors usually occur when a student performs an unexpected action, such as clicking **Pause** while playing an audio file, or dragging an unsupported file type into a list.

When authoring script in Page.htm, obey the following rules:

- The script must be located in the HEAD or BODY sections of the Page.htm file.

All script in the Page.htm file must be in the HEAD or BODY sections of the file, where they will be merged with Class Server scripts. Class Server parses the Page.htm file, extracts the HEAD and BODY sections, and then merges the sections into the final page depending on the view, such as Preview, Assigned, Grading, or Posted. Scripts placed outside the HEAD or BODY tags will not execute.

- Script should execute only in Class Server student views.

Write scripts that will execute in the Class Server student views only. Scripts that execute in teacher views may cause conflict with Class Server – Teacher. To determine if you are in one of the student views when editing an LRM using a script, check the value of the ECS_ViewType global variable.

A simple example is shown below.

```
function foo(){

        if (ECS_ViewType == 2) {

                document.write(" Good luck with the assignment. ");

        }

        /* Do nothing in other views */

}
```

See the next section of this document for the definition of the view types.

Note: Any attributes specified for HTML elements in Page.htm (such as the BODY element) will be merged with attributes that Class Server adds in different views. This means that any attributes, such as event handlers, defined in those tags will continue to exist when Class Server processes the page.

- Script should not assume that page elements are visible

In some views (for example, print views), Class Server will set the style of all elements to hidden. Assuming the elements are visible, for instance by attempting to set the focus to an element in the page, can cause the page to render incorrectly.

3.4.2 Learning Resources and Class Server Views

One of the advantages of LRM and IMS+ content is that the assessment markup will render differently depending on how the content is to be used. For example, a text entry field will appear as a standard HTML input box for the student completing a quiz, but after it is graded it will appear as written text with no editing capabilities. The content developer does not need a lot of details of how this works; it is part of the Learning Resource content management.

Class Server has six different views of a document, and each corresponds to different phases in the lifecycle of an assignment: Authoring, Preview, Assigned, Grading, Graded, Print and Web Grading. Print view is also used when saving a learning resource in IMS+ format.

3.4.2.1 Determining the Current View Type Through Script

Though a lot of this functionality comes with the LRM format, a developer building custom functionality may need to know which view is currently being rendered. Each page of the learning resource will have a variable named ECS_ViewType that indicates the view in which that page has been rendered:

Class Server view	ECS_ViewType value
Authoring	0
Preview	1
Assigned	2
Grading	3
Graded	4
Print	5
WebGrading	6

For an example using ECS_ViewType, see the *Sample LRM*, described later in this document.

3.4.3 Assessment Markup Format

Assessment markup is special HTML syntax that specifies metadata about autograded questions. This format is specially designed to enable flexible editing and online submission capabilities. Assessment markup is similar to XML data islands, but the source file can be directly and intuitively edited.

The *QuestionSample.lrm* in the SDK contains examples of various question types, and includes comments in the code to assist understanding of the assessment markup syntax.

3.4.3.1 Assessment Markup Types and Views

Assessment markup is the parts of a *question* or *item* in an LRM. For example, each option button for a multiple-choice question is assessment markup, or text form elements for multi-part questions. Assessment markup is grouped into questions by ID (for example, question 3A).

The following is the current list of assessment markup types and views.

Assessment Markup Type	Purpose of Assessment Markup	Authoring View*	Assigned View**	Grading View	Graded View	Print View
TYPE=1 **Text**	Single-line fill-in-the-blank question or sub-item. May be graded automatically or manually.	Image of a single-line text input form element.	Single-line text input form element.	HTML editing control containing text that a student entered. If an answer key was specified, the assessment markup contains autograding annotations indicating whether the answer is correct or not.	Text that a student entered, plus any annotations from autograding or manual grading.	Image of a single-line text input form element.
TYPE=2 **Text Area**	Multi-line fill-in-the- blank question or sub-item. Graded manually, autograding is not available for this type.	Image of a multi-line text input form element.	Multi-line text input form element (text area).	HTML editing control containing text that a student entered. If an answer key was specified, the assessment markup contains autograding annotations that say whether the answer is correct or not.	Text that a student entered, plus any annotations from autograding or manual grading.	Image of a multi-line text input form element.
TYPE=3 **Radio**	Sub-item within a single select multiple-choice question. Typically graded automatically.	Image of an option button form element.	Option button form element.	Image of an option button form element indicating whether the answer is right or wrong.	Image of an option button form element indicating whether the answer is right or wrong.	Image of an option button form element.
TYPE=4 **Check Box** **Note**: This type of assessment markup is not supported in the Question Wizard in the Learning Resource Editor.	Sub-item within a multi-select multiple-choice question. Typically graded automatically.	Image of a check box form element.	Check box form element.	Image of a check box form element indicating whether the answer is right or wrong.	Image of a check box form element, indicating whether answer is right or wrong.	Image of a check box form element.

Assessment Markup Type	Purpose of Assessment Markup	Authoring View*	Assigned View**	Grading View	Graded View	Print View
TYPE=5 **Item Score**	Item score (points) for a specific question.	A box that displays the maximum score for the item or question.	Text indicating the maximum score for the item or question.	Text input form element where the teacher can enter or override the item score, plus text indicating the maximum score for the item.	Text displaying the points awarded and maximum points for the item.	A box that displays the maximum score for the item or question.
TYPE=7 **Select**	Matching item question or sub-item.	Image of a drop-down list box form element.	Drop-down list box form element. List box items are always sorted alphabetically.	Text containing the option the student selected. If an answer key was specified, the assessment markup contains autograding annotations that say whether the answer is correct or not.	Text containing the option the student selected.	Text containing the option the correct answer.
TYPE=8 **Rubric**	Grading information, plus an assigned number of points. Multiple rubrics can apply to a single question.	Image of a check box, plus a score in a box.	The image of a gray check box and text indicating the score that would be added to the item if the teacher selects the check box.	Check box and text indicating the score that is added to the item if the teacher selects the check box.	The image of a gray check box, checked if teacher had selected it during grading, cleared if not. Also displays text indicating the score that was associated with that check box.	Image of a check box, plus a score in a box.
TYPE=9 **File** (attachment)	Question for which the student attaches a file, such as a Word document.	Image of a file upload form element.	File upload form element and a hyperlink to the previously uploaded file, if any.	Hyperlink that enables the teacher to view the attached file.	Hyperlink to the file as annotated by the teacher.	Text stating "Upload a file."

* The Print view is used when printing the learning resource or its answer key.

** In this table, information on Assigned view also applies to Preview view, Print view, and save to IMS+ format.

3.4.3.2 Assessment Markup Syntax

In a Page.htm file, an assessment markup is represented as an <IMG> element that has a *src* attribute of the following form:

```
protocol://server/mslamrk,arguments
```

The *mslamrk* in the src attribute identifies the <img>element as an assessment markup. Any other attributes of the <IMG> element should be ignored, and the various views will replace the entire <img>element with whatever that assessment markup represents in that specific view.

A more specific URL syntax is required for authoring utilities to work:

```
http://localhost:3535/mslamrk,arguments
```

The arguments string is a comma-separated set of arguments. The following example includes the <img> element syntax:

```
<img src="http://localhost:3535/mslamrk,type=2,id=8A,rows=4,cols=35">
```

In this example, the output of the assessment markup when the student completes the assignment is a 4-row 35-column <TEXTAREA> (type=2) identified as belonging to question (assessment item) 8A. "The value *localhost:3535* is the address of the authoring utility component that generates assessment markup images in Authoring view. This address is not used in any other view.

The *type*= argument specifies the type of the assessment markup.

Assessment markup argument values, for example, the string after the equal sign (=) in *akey=Canada*, can contain arbitrary characters (except ASCII 0 to 31, inclusive), but they need to be encoded. For example, *akey=Blue%20Sky* or *akey=Blue+Sky* means that the argument value for akey= is "Blue Sky." The rules of character encoding are as follows:

The following characters need to be encoded as "%XX", where XX is two hexadecimal digits (for example, "%3D" for "="): " # % & + , \ ~

The space character may be encoded as "+" or "%20", or left as is.

Characters above ASCII 127 should be UTF-8 encoded. The Learning Resource Editor encodes these characters in the following way, but it is not recommended for other utilities. Creating learning resources that rely on this behavior may cause problems in software that reads it.

- Characters greater than ASCII 127 and less than or equal to ASCII 255 are encoded using the same "%XX" format described above.
- Characters above ASCII 255 are encoded in the form ~XXXX, where XXXX is four hexadecimal digits (for example, "~898B" for the Kanji character representing "mi").
- Leading and trailing white spaces are removed.
- Multiple white spaces are compressed into one space.

3.4.3.3 Assessment Markup Arguments

Each assessment markup in a Page.htm file has a set of arguments. The following table lists assessment markup types and their required and optional arguments.

Assessment Markup Type	Required Arguments	Optional Arguments
Type=1 (Text)	type, id, cols	akey, uak
Type=2 (Text Area)	type, id, cols, rows	None
Type=3 (Radio)	type, id, maxpts	None
Type=4 (Check Box)	type, id, maxpts	None
Type=5 (Item Score)	type, id, maxpts	csid
Type=7 (Select)	type, id, text	None
Type=8 (Rubric)	type, id, maxpts	None
Type=9 (File)	type, id, cols	None

The following table describes the assessment markup arguments:

Argument	Example	Description
akey=	akey=Canada	Answer key. For **Text** assessment markup, akey= (if provided) specifies an answer key that indicates the correct answer for automatic grading. Semicolons are used to separate multiple correct answers (for example, "akey=color;colour"). Autograding matching is case-insensitive. If one student types "Color" in the previous example, and another student types "colour," both answers will be graded as correct. During grading, if akey= is not provided, the teacher must grade the question manually, either by entering a grade in the space provided by the **Item Score** assessment markup, or by selecting the appropriate check boxes provided by **Rubric** assessment markup.
cols=	cols=40	Specifies the width in characters of the associated form element. Depending on the element, this value is used as the value for the **cols** or **size** attribute of the HTML element.
csid=	csid=C12345;C234567	This feature is deprecated and should not be used when developing new Learning Resources. Specifies CSID values. For an **Item Score** assessment markup, csid= specifies the list of curriculum standard IDs associated with the question that is associated with that **Item Score** assessment markup. Each CSID value must start with an uppercase C followed by 5 to 8 digits. Multiple CSID values must be separated by semicolons with no white space between them.
id=	id=8A	Required item. This argument specifies the question (assessment item) number. All sub-items within a given question must have the same id= *value*. The id= *value* can not be longer than 31 characters. The characters are limited to digits (0-9), letters (a-z and A-Z), periods (.), hyphens (-), colons (:), and underscores (_).
maxpts=	maxpts=2.5	Maximum points. A floating-point number (positive, negative, or zero) that specifies the maximum number of points for an item. maxpts= is used in **Item Score** assessment markup to indicate the score for a given question. It is also used in **Radio**, **Check Box**, and **Rubric** assessment markup to specify a number of points (positive or negative) for that particular sub-item. The points must be within the range: -1000 <= maxpts <= 1000.
rows=	rows=5	Specifies the height (in lines) of the associated form element.

text=	text=A. Sacramento	For a **Select** assessment markup, text= specifies the text of one of the lines in the drop-down list box and the correct answer for this sub-item. For example, if a question has three **Select** sub-items, with "text=Beta" on the first, "text=Gamma" on the second, and "text=Alpha" on the third, then in Assigned view (that is, while the student is working on the learning resource) each of the three sub-items will appear as a drop-down list box with these four lines: "" (blank), "Alpha", "Beta", and "Gamma" (sorted alphabetically, with "" selected by default).
type=	type=2	Required item. Specifies the assessment markup type.
uak=	uak=1	Unordered answer key. Either "0" or "1". This argument is used by **Text** assessment markup to indicate that the student answers may appear in any order (among all sub-items in a question). For example, if the question is "name the three primary colors," and three **Text** sub-items are provided, one sub-item can have "akey=red", the second can have "akey=green", and the third can have "akey=blue", and all can have "uak=1", which means the student will get full credit for entering "red", "green", and "blue" in the three text boxes in any order. Note: If uak is provided, then akey must also be provided.

3.4.4 LREQ2 Question Format

LREQ2 questions allow assessment markup for an entire question to be edited at once. Specifically, they are questions implemented as HTML tables with the LREQ2 attribute defined. Class Server versions 3.0 and later provide the Question Wizard, accessible through the Learning Resource Editor, to create and edit LREQ2 questions.

An LREQ2 question table consists of the following rows of text and assessment markup:

- A question row
- Zero or more answer rows
- An optional rubric row (for certain essay and file attachment questions)

Depending on the question type, each answer section may have multiple possible answers. Generally, all answer rows are constructed in the same format. CSS styles applied to the table cells are used to format the components of the question.

LREQ2 questions must follow these guidelines:

- Table cells may not contain nested tables, except for a rubric table. The rubric table is contained in the rubric cell.
- All assessment markup in an LREQ2 table must have the same ID, and the ID must match the question number in the question row.

3.4.4.1 LREQ2 Table Definition

The definition of an LREQ2 table is:

<TABLE class=QTable width=95% LREQ2="***qtype***">

qtype is an identifier representing the question type, as defined in the following table.

Question Type	qtype

Question Type	qtype
Essay	1
Short Answer	2
Multiple Choice	3
True or False	4
Matching	5
Fill-In-The-Blank	6
Single/Multiple Answer Blanks	7
Attachment Request	8

3.4.4.2 Question Row

Each question has a question row containing the question number, teacher instructions, and the ITEMSCORE assessment element. The syntax for a question row is:

<TR style="vertical-align:***align***">

<TD style="vertical-align:***align***" class=QNum>***qnumber***.</TD>

<TD style="vertical-align:***align***" class=QText width="100%" colspan=***span***>***qtext***</TD>

<TD style="vertical-align:***align***" class=QPts style="text-align: right">

<img src=http://localhost:3535/mslamrk,type=5,id=***qnumber***,maxpts=***maxpts***>

</TD>

</TR>

Valid values for given parameters may differ depending on the question type.

Parameter	Value
align	Depends on the question type: For ***qtype*** of 6 (Fill-in-the-Blank), ***align***=*baseline*. For all others, ***align***=*top*.
qnumber	The value is the question number, for example "1". Note: ***qnumber*** is used in several places, and must be the same for all instances in a given question.
span	Depends on the question type: For ***qtype*** of 1,2,6, or 8, ***span***=*1*. For ***qtype*** of 3,4, or 5, ***span***=*3*. For ***qtype*** of 7, ***span***=*2*.

Parameter	Value
qtext	Depends on the question type: For ***qtype*** of 6 (Fill-in-the-Blank), ***qtext*** is created by concatenating information from each answer blank in the UI, as shown in the following example: `<TD class=AText width="100%">`***atext***$_0$ <img src="http://localhost:3535/mslamrk,type=1,id=***qnumber***,cols=15, akey=***akey***$_1$,uak=***ordered***" align=absMiddle> <span>***atext***$_1$</span> <img src="http://localhost:3535/mslamrk,type=1,id=***qnumber***,cols=15, akey=***akey***$_2$,uak=***ordered***" align=absMiddle> <span>***atext***$_2$</span> …. <img src="http://localhost:3535/mslamrk,type=1,id=***qnumber***,cols=15, akey=***akey***$_j$,uak=***ordered***" align=absMiddle> <span>***atext***$_j$</span> `</TD>` ***atext*** - Textual context for the blanks. ***akey*** - Answer key for the appropriate blank, valid values separated by semicolons. ***ordered*** - Whether blanks are part of an ordered or unordered list, 0 for ordered, 1 for unordered. If the blanks are ordered, the answers must be provided in the order specified in order to be marked correct. For all other question types, ***qtext*** is the question text as entered by the teacher. It should not include HTML.
maxpts	The maximum number of points possible for this question. For autograded questions, this value is computed from the points for each answer in the question depending on question type: For ***qtype*** of 1, 2, or 8, ***maxpts*** is the maximum score a student can achieve with the rubric, if present, or the maximum score assigned to the question. For ***qtype*** of 3 or 4, ***maxpts*** is the highest value assigned to an individual answer in the question. For ***qtype*** of 5, 6, or 7, ***maxpts*** is the number of answer blanks or items to match in the question. In all cases, the value must be within the range: -1000 <= maxpts <= 1000.

Note: For Fill-in-the-Blanks (***qtype*** = 6), the teacher instruction is the actual question, so the class for the cell is AText, not QText.

This is a sample multiple choice question row for the first question in a learning resource, with a point value of 2.:

```
<TR style="vertical-align:top">
        <TD style="vertical-align:top" class=QNum>1.</TD>
        <TD style="vertical-align:top" class=QText width="100%" colspan=3>What is the state capital of
        Texas?</TD>
        <TD style="vertical-align:top" class=QPts style="text-align: right">
        <img src=http://localhost:3535/mslamrk,type=5,id=1,maxpts=2>
        </TD>
</TR>
```

See *QuestionSample.lrm* in the SDK for examples of other question types.

3.4.4.3 Answer Row

With the exception of the Blank question and the Fill-in-the-Blanks question, each question has one or more answer rows; the syntax depends on the question type.

qtype	Answer Row

1 – Essay
 	
2 – Short Answer
	These questions have only one answer row:
	`<TR style="vertical-align: top">`
	&nbsp;&nbsp;`<TD>&nbsp;</TD>`
	  `<TD class=AText width="100%">`
	    <img src="http://localhost:3535/mslamrk,type=2,id=***qnumber***,
	      cols=***cols***,rows=***rows***">
	  `</TD>`
	&nbsp;&nbsp;`<TD>&nbsp;</TD>`
	`</TR>`
	
	cols and **rows** are the size of the text edit control.
	We recommend the following values:
	Cols:
	Small = 40
	Medium = 60
	Large = 80
	
	Rows:
	Small = 6
	Medium = 15
	Large = 25
	We recommend using the Small values for Short Answer questions.

qtype	Answer Row
3 – Multiple Choice 4 – True or False	These questions have one answer row per possible choice: <TR style="vertical-align: top"> <TD>&nbsp;</TD> <TD> <img src="http://localhost:3535/mslamrk,type=3,id=***qnumber***, maxpts=***points***$_i$" align=absMiddle> </TD> <TD class=ANum>***anumber***.</TD> <TD class=AText width="100%">***atext***</TD> <TD>&nbsp;</TD> </TR> ***points -*** The number of points this choice is worth. If left blank, ***points***$_i$ = 0. Values may be negative. ***anumber*** – For English content, this is a letter from **a** to **z**, depending on the position of the choice with respect to other choices in the question. The answer rows are displayed in the order they exist in the Learning Resource, therefore in English content we recommend the first choice be **a**, the second **b**, etc. Content in other languages may use characters from their alphabet. ***atext*** - The text for the choice.
5 – Matching	This question has one answer row per item to match: <TR style="vertical-align: top"> <TD>&nbsp;</TD> <TD class=ANum>***anumber***$_i$</TD> <TD class=AText width="30%">***atext***$_i$</TD> <TD class=AText width="70%"> <img src="http://localhost:3535/mslamrk,type=7,id=***qnumber***,text=***answer***$_i$"> </TD> <TD>&nbsp;</TD> </TR> ***anumber*** - A letter from **a.** to **z.**, depending on the position of the choice with respect to other choices in the question. The first choice is **a.**, the second **b.**, etc. ***atext*** - The text for the item to match. ***answer*** - The correct matching answer for this item.

qtype	Answer Row
6 – Fill-in-the-Blank	This type of question has no answer row. The answers are provided as part of the question row.

7 – Single or Multiple Answer Blanks	This question has one row per blank to fill in:
	<TR style="vertical-align: top">
	    <TD>&nbsp;</TD>
	    <TD class=ANum>***anumber***$_i$</TD>
	    <TD class=AText width="100%"><br> <img src="http://localhost:3535/mslamrk,type=1,id=***qnumber***,cols=15,
	            akey=***akey***$_i$, uak=***ordered***">
	    </TD>
	    <TD>&nbsp;</TD>
	</TR>
	
	anumber - A letter from **a.** to **z.**, depending on the position of the choice with respect to other choices in the question. The first choice is **a.**, the second **b.**, etc.
	akey – Answer key for the appropriate blank, valid values separated by semicolons.
	ordered - Whether blanks are part of an ordered or unordered list; 0 for ordered, 1 for unordered. If the blanks are ordered, the answers must be provided in the order specified for them to be marked correct.

8 – Attachment	This question has only one answer row:
	<TR style="vertical-align: top">
	    <TD>&nbsp;</TD>
	    <TD class=AText width="100%"><br> <br>**studentinstructions**<p>
	        <img src="http://localhost:3535/mslamrk,type=9,id=***qnumber***,cols=30">
	    </TD>
	    <TD>&nbsp;</TD>
	</TR>
	
	Studentinstructions – Instruction text for students. This text is optional.

See *QuestionSample.lrm* in the SDK for examples of other examples of answers.

3.4.4.4 Rubric Row

Questions that can have rubrics may have a rubric row as well. Question types 1, 2, and 8 allow rubrics. If a question has a rubric attached, it will have an additional row after the answer row(s):

<TR style="vertical-align: top">

<TD colspan=3>

rubric

</TD>

</TR>

Rubric is the rubric table for the rubric. Class Server 3.0 and later supports three types of rubrics, which are identified in a similar way as questions:

<TABLE class=**RTable** lrer2=***rtype***>

Rtype is an identifier representing the rubric type:

Rubric Type	rtype
Single-element	1
Multi-element	2
6+1 TRAITS™	3

In addition to its type, a rubric can be visible to the student or hidden from the student. Hidden tables have a **BGColor** attribute set to a specific value of **#FF99CB**. In order to produce the desired effect, they also have to have a **style="BACKGROUND-COLOR: #FF99CB"**. If authors want to color their rubric tables differently, they should add a **background-color** property to the other rubric styles.

This section of the spec assumes rubrics are visible to the student. For an example of a hidden rubric, see Short Answer with Multi-Element Hidden Rubric, below.

3.4.4.4.1 Single-Element Rubric

A single-element rubric is composed of a title row, followed by one or more grading rows, as follows:

Row	HTML
Title Row	<TR style="vertical-align: top"> <TD class=RTitle colSpan=2>***title***</TD> </TR> ***Title*** is the rubric title as entered by the teacher.

Row	HTML

Grading Row(s)	<TR style="vertical-align: top">
	    <TD class=RPts width="1%" nowrap>
	        <img src="http://localhost:3535/mslamrk,type=8,id=***qnumber***,
	          maxpts=***points***">
	    </TD>
	    <TD class=RText>***text***</TD>
	</TR>
	
	points - The number of points this grading row is worth.
	text – The text description of the rubric.

3.4.4.4.2 Multi-Element Rubric

A multi-element rubric is composed of a title row, a heading row, and one or more element rows, as follows:

Row	HTML
Title Row	<TR style="vertical-align: top"> <TD class=RTitle>&nbsp;</TD> <TD class=RTitle colspan=2>***bad***</TD> <TD class=RTitle colspan=2>***better***</TD> <TD class=RTitle colspan=2>***best***</TD> </TR> There must be one cell in this row for each level of achievement. You must have exactly 3 levels of achievement. ***bad*** - Text that labels an insufficient level of achievement, such as, "Needs Improvement." ***better*** - Text that labels an acceptable level of achievement, such as, "Satisfactory." ***best*** - Text that labels an excellent level of achievement, such as, "Exemplary."

Row	HTML

Element Row(s)	`<TR style="vertical-align: top">`
	    `<TD class=RHeader>`***element***`</TD>`
	
	    `<TD class=RPts width="1%">`
	        `<img src="http://localhost:3535/mslamrk,type=8,id=`***qnumber***`,maxpts=`***points*** $_1$`">`
	    `</TD>`
	    `<TD class=RText>`***criterion*** $_1$`</TD>`
	
	    `<TD class=RPts width="1%">`
	        `<img src="http://localhost:3535/mslamrk,type=8,id=`***qnumber***`,maxpts=`***points*** $_2$`">`
	    `</TD>`
	    `<TD class=RText>`***criterion*** $_2$`</TD>`
	
	    `<TD class=RPts width="1%">`
	        `<img src="http://localhost:3535/mslamrk,type=8,id=`***qnumber***`,maxpts=`***points*** $_3$`">`
	    `</TD>`
	    `<TD class=RText>`***criterion*** $_3$`</TD>`
	`</TR>`
	
	For each level of achievement defined in the Title Row, there must be a points and criterion cell in the Element Row.
	
	element – The title of this element, describing the overall objective being measured according to the various criteria.
	
	points - The number of points for performance at the level represented by ***criterion***.
	
	criterion - A description of the performance required to achieve the level described in the header row over the column containing this criterion.

3.4.4.4.3 6+1 TRAITS™ Rubric

The 6+1 TRAITS™ rubric is a special rubric that is not modifiable. Copyright: http://www.nwrel.org/ .

3.4.5 Embedded Objects

Media files can create a dynamic, interactive learning experience, but be aware of the following distribution, security, and formatting issues. Also, make sure to use the appropriate utilities for authoring the media content. For example, to view Macromedia Flash or Shockwave files using Netscape Navigator, use the Netscape plug-in for Shockwave, instead of the Internet Explorer ActiveX control.

Note: The Class Server Learning Resource Editor is not compatible with pages containing more than one object tag. If you are creating content with multiple objects, either distribute the objects across multiple LRM pages, or set the IsLocked attribute of the LearningResource\General node in Index.XML to true.

Media files have many configuration dependencies and should be thoroughly tested in all configurations.

3.4.5.1 Macromedia

Integrating Macromedia Flash and Shockwave content into Learning Resources allows publishers great latitude in the sort of learning content they can offer to teachers and students. With a little extra code, Flash content can interact with the LRM assessment markup and grading functionality of Class Server.

For details, see the *Sample LRM*, described below.

3.4.5.2 ActiveX Controls

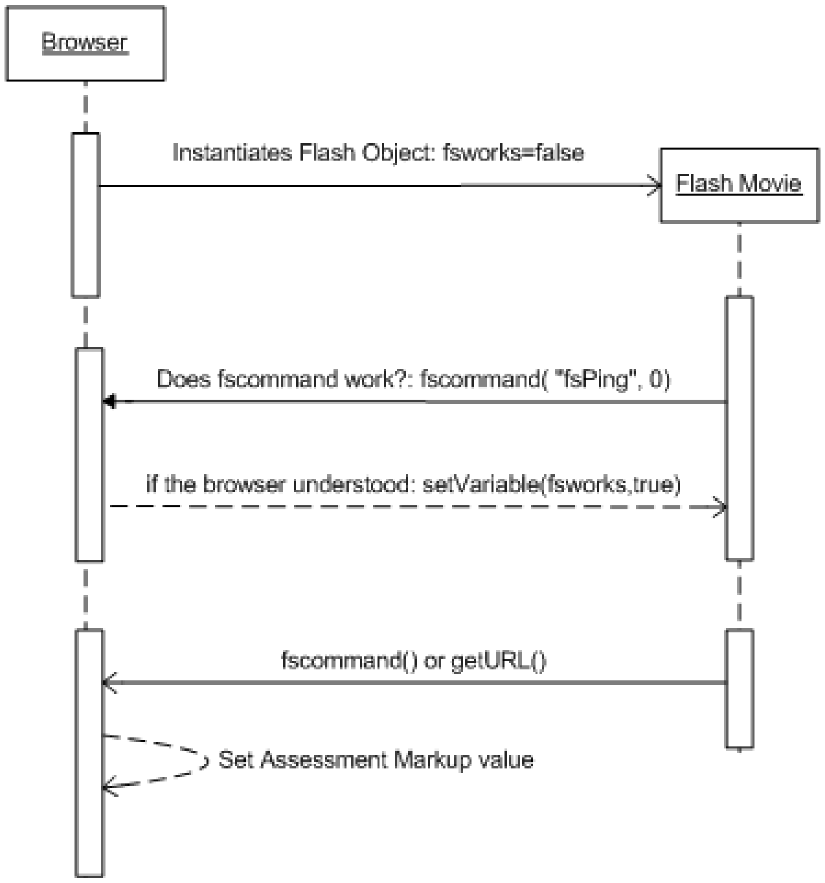
As with Flash or Shockwave objects, content containing plug-ins or ActiveX controls requires users to install the appropriate software. Plug-ins and ActiveX controls may not work as expected on older browsers, or on older Windows or Macintosh operating systems.

To maximize security, and to protect student and teacher computers from viruses, make sure that media controls are downloaded directly from manufacturer Web sites and that they are digitally signed. A control that is not digitally signed displays a warning message prior to installation stating that the content is not signed. The message prompts to accept or cancel the installation.

3.4.5.3 Sample LRM Using Flash Content

The *Sample LRM* included in this SDK demonstrates methods for using Flash content with assessment markup. The sample could be generalized to work with other forms of ActiveX content. The sample demonstrates several capabilities:

- **Communicating with script from Flash while inside frames and across browsers.** The problem of communicating with script from Flash is not unique to Class Server but is a difficult problem to overcome. The sample determines if the browser being used supports the Flash fscommand() function, and provides a smooth workaround via the getURL() function if the browser does not. The sample provides a standardized way to send commands between Flash and the browser that will work in both cases. This technique ensures that Flash-enabled LRMs work with as many of the browsers supported by Class Server as possible, as well as work in framed views, such as the Class Server – Teacher preview window.


Browser
Instantiates Flash Object: fsworks=false
Flash Movie
Does fscommand work?: fscommand("fsPing", 0)
if the browser understood: setVariable(fsworks,true)
fscommand() or getURL()
Set Assessment Markup value

- **Setting a score from Flash content**. ActiveX or Flash content is often designed to do its own grading and report back a final score and summary to the teacher. Assessment markup can be used to relay this data from Flash in the student view to the teacher in grading view. The sample code shows how to set a score by using a question with several check boxes, each worth a different value (1, 2, 4, 8, etc). When the user presses the "Save Student Score" button in the Flash content, Flash sends a command to the browser page telling it to turn on the check boxes necessary to total the desired score. When the LRM is graded, this value will automatically appear as the score for a question.

  Note that a message, "Warning: If there is more than one correct answer, make sure the sum of the correct answer points equals the value of the total question points." or similar may appear in the Preview view of the Sample.lrm. This warning message may be safely ignored in this case. The reason the message appears is that the sum of the checkboxes hidden inside the assessment markup is greater than the max score of the question. In this example, for simplicity and convenience, the checkboxes are power-of-two values: 1, 2, 4, 8, 16, 32, 64, 128, 245, and 512. This allows the Flash object to report a score up to the maximum score allowed by Class Server, which is 1000 points per question. However, for this particular sample question the maximum score is set to 100. The sum of the checkboxes is 1023. Hence, the warning message.

- **Setting and using text data.** Assessment markup includes essays. Essays can be used to save and load the state of a Flash object, report student performance to a teacher (such as time taken), or simply to relay student work back to the teacher. In a simple demonstration of this, the sample responds to keystrokes in a Flash InputText field and changes the value of the associated LREQ2 question to match. When the Learning Resource is graded, the text entered by the user will appear and can be graded by using a rubric.

To use this sample in a Class Server installation without Windows SharePoint Services, open it in Class Server – Teacher and review it in all Class Server views. The key source code for this sample is contained in the LRM, in Page.htm and FlashSample.fla. All of the LRM files can be modified and reused to contain your own content.

Another way to view this sample is to sign-in as a teacher to a school web site that is configured to use SharePoint with the Class Server web parts. Upload the sample into a document library. Choose "View Resource" to view a preview of the sample. Choose "Assign" to assign the sample to students. Sign-in as a student and submit the assignment. As a teacher, grade the assignment and view the graded assignment.

Note that as stated above, a warning message appears in preview and print views: "Warning: If there is more than one correct answer, make sure the sum of the correct answer points equals the value of the total question points." In editing, grading, and graded views the warning message does not appear, because the assessment markup is shown. In student view, neither the warning message nor the assessment markup is shown.

The *Sample LRM* is not designed to be used for production code, although it may be freely reused. There are additional issues with Flash-enabled LRMs, which are not covered in this example:

- **Managing re-entrance on a Learning Resource page.** When the student wants to submit his or her responses to the LRM, they must navigate to another page. If they choose to return to the flash-enabled page, the answers they entered (or the state they left the Flash content in) should match the state at the time they left the page. In addition, when an assignment is returned to a student, data the student entered should be displayed. This could be done by serializing the data from the Flash movie into a hidden field on the Learning Resource page, which could then be de-serialized to restore the values in the Flash movie. If this is not practical, a simpler alternative is to alert the student upon reloading a page that their data has been saved and will be sent when the Learning Resource is submitted, and that any changes they make will overwrite the saved information.

- **Controlling where and when Flash Questions and Answers are shown.** By using the ECS_ViewType variable, publishers can create their content where the Flash and grading portions are displayed only in certain views. For instance, the sample hides grading and LREQ2 portions in the student and preview views, but the publisher may wish to hide the flash content during grading or printing. Optionally, a publisher might want 2 pieces of Flash content, one that is displayed to the teacher explaining how the content will be used, and another for the student to complete.
- **Obfuscating scores set from Flash.** The Sample LRM sets the final score using a set of check boxes that can be deciphered by the student. By attaching a script debugger to the browser running the Flash content, a student could set their own score manually. This can be made more difficult by splitting the point values up across several questions, changing the order of the point values, or using JavaScript to transform the value when the contents are saved.

Note: For more information about browser compatibility, distributing Flash Player or Shockwave Player, or hints on how to embed Flash and Shockwave files that are compatible with a wide range of browsers, visit the Macromedia Web site at http://www.macromedia.com.

3.5 Licensing

One of the features of the LRM format is the ability to include licensing. LRM license files exist in two formats:

- Uncompressed XML format
- Compressed .lrm format (required for use with Class Server)

Only license files in .lrm format can be processed by Class Server – Teacher. When a user double-clicks a license file, or downloads a license file from a server, Class Server – Teacher will upload the license file to Class Server, which merges the license into the school's database of licenses (after verifying that the license is for that school). After the license file is uploaded to Class Server, teachers can assign the licensed content for the period of time specified by the license.

Note: Class Server will not allow import of license files that are 1 MB or larger.

3.5.1 Creating License Files

To create a license file for Class Server, first create the file in uncompressed XML format, and then compress the file into .lrm format.

To create the uncompressed XML format license file, you can do the following:

- Create the file manually. You can then verify the syntax of the license file using *Validate.exe* (see below).
- Create the file programmatically. For example, you can use a Web server to generate license files for customers. It is recommended that you use Validate.exe to test the output from the software.

  If you use an ASP page to generate the license, consider using an ISAPI filter, such as LRLicenseFlt.dll, so that a Web browser will correctly process the downloaded license file and cause Class Server - Teacher to be invoked. See the LRLicenseFlt directory in the SDK for more information.

To convert uncompressed XML format license file to .lrm format, you can do the following:

- Use *MakeLicense.js* to manually convert an XML-format license file to .lrm format, for example:

  ```
  CScript MakeLicense.js input-license.xml output-license.lrm
  ```

- Use the TeacherClientHelp ActiveX object to programmatically convert an XML format license file to .lrm format. See the License.asp file for an example.

Note: Both MakeLicense.js and the TeacherClientHelp ActiveX object require that Class Server be installed on the computer.

3.5.2 License File XML Format

This section describes the XML format of the Class Server license file. See *SampleLicense.xml* for an example of a license file in XML format. Note that a license file must be bundled using .lrm format before it can be opened in Class Server.

The root XML node of a XML-form license file is the <LicenseFile> node, which specifies the namespace:

```
xmlns="http://schemas.microsoft.com/classServer/2002/02/license"
```

In the <LicenseFile> node are three child nodes:

- A *<Version>* node contains the version number, which must be 2.0.
- A *<LicensedSchools>* node contains a sequence of <LicensedSchool> nodes. Each *<LicensedSchool>* node specifies the GUID (globally unique identifier) of a school to which the license applies. A single license file can be used for multiple schools. For example, a school district may license content, with one license, for all the schools in that district.
- A *<Licenses>* node contains a sequence of <License> nodes. Each *<License>* node specifies the licensing rights (for each school listed in the <LicensedSchools> node) for a particular LRM or collection of LRMs.

3.5.2.1 <LicensedSchool> Node Format

A <LicensedSchool> node contains one attribute, which is ID. The ID contains the GUID of a school to which the license applies. The GUID of a school can be determined in several ways, such as the following:

- Run Class Server Configuration, which is installed on the Class Server computer. The School GUID appears on the first tab.
- Go to the following URL in a browser:

```
http://server-name/Schoolname/post/hello
```

For example: http://myserver/JeffersonHighSchool/post/hello

This will display an XML string containing the school GUID, as well as other attributes. The school GUID is the value of the SchGUID attribute).

For example, see the following LicensedSchool node:

```
<LicensedSchool ID="1058e324-6d5a-4733-8f50-8300f7726c91" />
```

3.5.2.2 <License> Node Format

A <License> node specifies the licensing rights for a school listed in the <LicensedSchools> node. These are the rights for a particular LRM or collection of LRMs.

The ID attribute of the <License> node identifies one or more LRMs. Each of these LRMs must have the same GUID specified in the license information section of its Index.xml file.

A license ID can identify an unbounded set of content. For example, an ID may refer to a collection of 6th-grade American History LRMs. In the future, if more LRMs are added to the collection, or existing LRMs in the collection are updated, the school's right to the collection remains intact.

Note: A license applies to an entire school.

In addition to the ID attribute, a license file also has the following attributes:

- The *Rights* attribute that specifies what licensing rights the school has. This attribute is a sequence of parenthesized substrings. Currently, the only valid value of Rights is *(A).*
- The *SDt* attribute (optional) that specifies the start date of the license. This attribute is an integer in the OLE DATE format (for example: 37200 is November 5, 2001). The school may begin using the licensed content at the beginning of the specified day. If the SDt attribute is not specified, the school may begin using the content immediately.

- The *EDt* attribute (optional) that specifies the end date of the license. This attribute is an integer in the OLE DATE format. The school may use the licensed content until the end of the specified day. Note: Class Server may internally add between zero and one day (a grace period) to EDt to account for time zone variations. If the EDt attribute is not specified, the school may use the content indefinitely.

For example, see the following License node, which provides a school a specified amount of time:

```
<License ID="98ccf86f-e18b-4fbc-9eb6-beb0e4930d8b" Rights="(A)" SDt="30003"
EDt="30103" />
```

3.5.2.3 Merging License Files

Once a license has been issued to a school, there is no mechanism for revoking the license. You may also not restrict licensing by uploading a more limited permission set for the same content.

When multiple license files are uploaded to a school, the broader permissions set is applied to licensed content. For example, a license file X is uploaded to a school using Class Server. If a license file Y is uploaded to the same school, with the same LRMs, but license file Y is more restrictive (has a later SDt date, for example), then license file Y is ignored. The upload process to Class Server will not reduce a school's LRM usage rights.

Another example is that a license file Y was granted to the school first, and later license file X is granted (to extend the license end date, for example), Class Server will ensure that the school's right to use the content won't be reduced if the license file Y is accidentally uploaded over license file X (from restoring a back up of Class Server, for example).

3.5.2.4 Validating License XML

License.xsd contains the XML schema for license files in XML Schema (XSD) format.

You can run Validate.exe from a command-line prompt to validate the syntax of a license file. The following command line validates the syntax of the SampleLicense.xml file:

```
        Validate.exe License.xsd SampleLicense.xml
```

Validate.exe only validates that an XML format license file has the correct syntax. It will not verify that the file can be used by Class Server with no errors. Always test a license file with Class Server to verify that it functions correctly.

Note: Validate.exe requires that the .NET Framework be installed.

3.5.3 License File .lrm Format

A license file bundled in the .lrm format is similar to a Learning Resource (LRM) bundled into .lrm format. A licensed LRM has the following additions to the file:

- "X-LRM-License-Version: 2.0" appears in the file header instead of "X-LRM-Version: 1.0".
- The LRM license file contains a single file section, with the Content-Location equal to License.xml. This section contains the license file in XML format, compressed using the normal LRM compression method.

3.5.4 Sample: SampleLicense.xml

The *SampleLicense.xml* sample is an example of a license file in XML format. Note that a license file must be converted to LRM format before it can be used by Class Server. Also note that you must replace the IDs and dates in the file to match your school and your LRMs before using it.

4 Learning Resource Bundling Formats

4.1 IMS (ZIP)

The IMS file format bundles one or more Learning Resources into a single file with an *.ims* or *.zip* extension. IMS files can be stored on content servers and downloaded as files through HTTP.

IMS Packages that contain multiple Learning Resources contain only a single manifest file (IMSManifest.xml) that contains all of the information from the individual manifests; they are rebuilt when the manifest is unbundled.

The bundling format for IMS content is defined by the IMS Global Consortium (http://www.imsglobal.org) as being zip based.

You can use the *MakeIMS.js* utility to create an LRM file from either an LRM directory or a directory containing a set of LRM subdirectories. The directory can then be unbundled using *UnIMS.js*. The *MakeIMS.js* and *UnIMS.js* utilities are provided as part of Microsoft Class Server – Teacher. Alternatively, an application can zip and unzip an IMS+ directory using available utilities.

4.1.1.1 MakeIMS.js

The MakeIMS utility will convert a directory that contains an LRM to an IMS+ file with the .ims extension.

MakeIMS can be started by double-clicking the MakeIMS.js file, which is in the web\client directory under the directory where Class Server - Teacher is installed. It will then prompt you for the directory to bundle and then a file to store it in.

MakeIMS can also be run from the command line with one of the following forms:

```
makeims <inputdirectory> <outputfile>

makeims <inputdirectory>

makeims
```

The user will be prompted for any information that was not included on the command line.

4.1.1.2 UnIMS.js

The UnIMS utility will take a file in IMS+ format (with the IMS or ZIP extension) and open it into a directory on the user’s drive.

UnIMS can be started by double-clicking the UnIMS.js file, which is in the web\client directory under the directory where Class Server - Teacher is installed. It will then prompt you for the file to unbundled and the directory to store it in.

UnIMS can also be run from the command line with one of the following forms:

```
unims <inputfile> <outputdirectory>

unims <inputfile>

unims
```

The user will be prompted for any information that was not included on the command line.

If there are multiple Learning Resources stored in the IMS package then a directory for each will be created under the directory chosen by the user.

4.2 Bundled LRMs

One or more LRM or IMS+ directories can be bundled into a single file with an .lrm extension. The .lrm files can be stored on content servers and downloaded as files through HTTP.

As with .ims packaged content, you can combine multiple LRM or IMS+ directories into a single .lrm. The limit on the size of a .lrm created is directly related to the amount of RAM in the computer creating the .lrm. The computer opening a .lrm does not have this requirement (for example, a teacher's computer with 64 MB RAM can open a 100 MB .lrm). A smaller .lrm is easier for teachers to handle (copy, import, and so on). Therefore, we recommend that .lrm files be less than 100 MB in size.

4.2.1 Methods For Creating LRMs

You can use the *MakeLRM.js* and *UnLRM.js* utilities to work with .lrm files interactively or as part of a build process. Alternatively, an application can link into LRMDLL to create a .lrm file programmatically. LRMDLL can be statically linked and redistributed with other applications.

4.2.1.1 MakeLRM.js

The MakeLRM utility will convert a directory that contains an LRM to a file with the .lrm extension. MakeLRM will also convert a directory that contains subdirectories of LRM or IMS+ content into a single .lrm file.

MakeLRM can be started by double-clicking the MakeLRM.js file, which is in the web/client directory under the directory where Class Server - Teacher is installed. It will then prompt you for the directory to bundle and then a file to store it in.

MakeLRM can also be run from the command line with one of the following forms:

```
makelrm <inputdirectory> <outputfile>

makelrm <inputdirectory>

makelrm
```

The user will be prompted for any information that was not included on the command line.

4.2.1.2 UnLRM.js

The UnLRM utility will take a file in .lrm format and open it into a directory on the user’s drive.

UnLRM can be started by double-clicking the UnLRM.js file, which in the web/client directory under the directory where Class Server - Teacher is installed. It will then prompt you for the file to unbundled and the directory to store it in.

UnLRM can also be run from the command line with one of the following forms:

```
unlrm <inputfile> <outputdirectory>

unlrm <inputfile>

unlrm
```

The user will be prompted for any information that was not included on the command line.

Each LRM stored in the .lrm file will be created under the directory chosen by the user, even if there is only one Learning Resource.

4.2.1.3 LRMDLL

LRMDLL can be linked into or run from the command line to save and open .lrm files. The functions performed are similar to the interactive MakeLRM.js and UnLRM.js utilities, but the code is suitable to a redistributable component. In particular, LRMDLL has no dependencies on Class Server components or particular system DLLs. For instance, you may add LRMDLL to an application so that the application can save files to the LRM format.

For more information, see readme.txt in the LRMDLL directory.

4.2.2 Mime Media Settings

For some Web browsers to support the Class Server one-click download feature properly, publishers hosting .lrm files on a Web server must make the Web server aware of the MIME Media-Type setting for .lrm and .ims files. The MIME type and subtype for server settings are as follows:

Application/vnd.ms-lrm

Application/vnd.ms-ims

Refer to your Web server documentation on how to configure a server to support MIME types and how to add the appropriate settings. Use different browsers such as Netscape Navigator and Internet Explorer to test how content is downloaded from your Web site. Make sure that the file opens directly when a user clicks it, and verify that it is imported into Class Server - Teacher automatically without the browser displaying a message asking if the file should be downloaded or saved.

4.2.3 Format

A .lrm file is a multi-part MIME message, containing headers, a blank line, and a body.

The headers (in standard MIME header format) include the following:

- Content-Type: multipart/related; boundary="boundary-string"
  where boundary-string is a string that is not contained anywhere else in the .lrm file, except in message part separators.
- MIME-Version: 1.0
  Standard MIME version number header.
- X-LRM-Version: 1.0
  LRM file format number. Encarta Class Server 1.0 and Microsoft Class Server will accept any .lrm file for which X-LRM-Version is greater than or equal to 1.0 and less than 3.0. (The first two characters must be "1.".)
- X-Multi-LR: LR-count
  This is an optional header indicating the number of Learning Resources in the package.
- If X-Multi-LR is absent, the .lrm file is in "single-LR format", containing a single LRM. In this case, each Content-Location header contains the name of a file in the Learning Resource, relative to the root of the Learning Resource directory (for example, "Index.xml" or "P1001\Something.gif"). See Message part headers later in this list.
- If X-Multi-LR is present, the .lrm file is capable of containing multiple LRMs in the single file. LR-count is the number of contained LRMs. In this case, each Content-Location header contains the name of a file in a specific LRM subdirectory, prefixed by an arbitrary subdirectory name. For example, if Content-Location for an image file is XYZ\P1001\Something.gif, the message part contains a file named P1001\Something.gif in an LRM labeled XYZ. In this example, the name XYZ is used purely in the .lrm file. When the .lrm file is imported into Class Server, the name XYZ is discarded because the name of the LRM is determined by the Title attribute within Index.xml.

Important: If the .lrm file is in "multi-LR format," the message parts corresponding to a particular LRM must all be contiguous.

The body contains a series of message parts, then the boundary string that is preceded by two hyphens (--) and followed by two hyphens and a CRLF (carriage return line feed). Each message part represents a single file in an LRM and consists of a message part header and a message part body.

Message part headers:

- Content-Location: *file-name* where *file-name* is the name of the file. Specifically:
- If the .lrm headers contain X-Multi-LR, then Content-Location, for example *XYZ\Index.xml* or *XYZ\P1000\Page.htm,* begins with an arbitrary label identifying the LRM.

- If the LRM headers do not contain X-Multi-LR, then Content-Location (for example, "Index.xml" or "P1000\Page.htm") is relative to the root of the LRM directory.
- X-Enc: encoding-format/unencoded-length where encoding-format specifies how the LRM file is encoded, and unencoded-length is the length of the unencoded version of the file. Currently, the only defined encoding format is "1". X-Enc is optional. If it is not present, the data is not encoded. Typically, HTML files (with .ht* extensions) and XML files (with .xml extensions) are encoded, and other files are not.

Message part body:

- The content is placed in the message part body, encoded if X-Enc is present in the headers.

4.2.3.1 Example of an LRM File In Single-LR Format

```
Content-Type: multipart/related; boundary="--------------------f9c0a7f4f5"
MIME-Version: 1.0
X-LRM-Version: 1.0

----------------------f9c0a7f4f5
Content-Location: Index.xml
X-Enc: 1/2817

<<encoded bits>>
----------------------f9c0a7f4f5
Content-Location: P1000\Page.htm
X-Enc: 1/49092

<<encoded bits>>
----------------------f9c0a7f4f5
Content-Location: Shared\Web.gif

<<unencoded GIF bits>>
----------------------f9c0a7f4f5--
```

Example of an LRM file in multi-LR format

```
Content-Type: multipart/related; boundary="--------------------f9c0a7f4f5"
MIME-Version: 1.0
X-LRM-Version: 1.0
X-Multi-LR: 2

----------------------f9c0a7f4f5
Content-Location: Act1\Index.xml
X-Enc: 1/2817

<<encoded bits>>
----------------------f9c0a7f4f5
Content-Location: Act1\P1000\Page.htm
X-Enc: 1/49092

----------------------f9c0a7f4f5
Content-Location: Act2\Index.xml
X-Enc: 1/2345

<<encoded bits>>
----------------------f9c0a7f4f5
Content-Location: Act2\P1000\Page.htm
X-Enc: 1/34567

<<encoded bits>>
----------------------f9c0a7f4f5--
```

4.3 Converting Between LRM and IMS Formats

Class Server – Teacher can export Learning Resources in either LRM or IMS+ formats. As a convenience to publishers and authors, the Class Server SDK includes a helper application called LearningResourceEditor.hta which allows you to use the editing and format conversion features of Class Server – Teacher without having to configure and connect to a Class Server server.

To use LearningResourceEditor.hta you must have already installed Class Server – Teacher. To use it for format conversions you simply open a file of one format (.lrm, .ims, .zip), and save it as a file with a different suffix.

LearningResourceEditor.hta can also be operated from the command line. Valid command line arguments can be found by invoking it with the /? argument.

LearningResourceEditor.hta can be found in the <Class Server SDK installation folder>\LearningResource\LearningResourceEditor folder in the Class Server SDK.

5 Testing Learning Resources in Class Server

It is recommended that you configure all browsers supported by Class Server to maximize browser compatibility. It is also important to test the content using all possible user roles and interaction. If you have not already done so, install Class Server – Teacher. For full testing you will need to create sample teacher, parent, student, and administrator accounts; and then complete an assignment life cycle.

For information on how to install Class Server – Teacher and create sample accounts, see the Class Server Administrator's Guide on the Class Server CD-ROM.

5.1 Quick Test: LearningResourceEditor.hta

The helper application LearningResourceEditor.hta described in section 4.3 can be used as an initial test of your Learning Resource. Open your Learning Resource with LearningResourceEditor.hta, and view all the pages. If you do not receive any error messages, and your Learning Resource looks correct, then it is likely that there are no serious issues with the format of your Learning Resource and you should move on to the assignment lifecycle test described in the next section.

5.2 Full Test: Assignment lifecycle

To fully test your content, you must review it as a teacher, parent, and student. Since Class Server – Teacher validates Learning Resources differently than the Class Server web parts, it's important to validate the assignment workflow using both features.

Follow this process to test your content:

1. Sign-in to Class Server - Teacher, import the content and then create, preview, and distribute an assignment.
2. Using a browser, sign-in as a teacher to a school web site that is configured to use SharePoint with the Class Server web parts. Upload the content into a document library and then view and assign the content. Note that the Class Server web parts do not operate on multi-LR Learning Resource packages.
3. Sign-in as a student, complete the assignments using all correct answers, and then submit them.
4. Sign-in as another student, complete the assignments using all incorrect answers, and then submit them.
5. Sign-in as a third student, complete the assignments using correct and incorrect answers, and then submit them.
6. Sign-in to Class Server – Teacher and then grade and return one assignment.
7. Using a browser, sign-in as a teacher to the school web site and grade and return the remaining assignment.

8. Sign-in as a parent and review the graded work.

For more detailed information on testing using an assignment lifecycle, see the *Class Server Administrator’s Guide*.

6 Glossary

The following acronyms and terms are used in this document:

Term	Description
answer key	The set of correct answers and point values for multiple-choice, fill-in-the-blank, and matching exercise questions.
assessment item (generally a question)	Content in Class Server that constructs an interrogative expression used to test knowledge. Usually this is expressed as a question with associated answer keys, and feedback similar to that of traditional testing methodologies. An assessment item might also be a rubric, which in Class Server is a subjective set of guidelines or criteria used for evaluating and scoring student performance and for providing feedback. Assessment items are built from an assemblage of Class Server Assessment Markup, generic text, and HTML, which are all organized in a specialized question or rubric table.
assessment item components	Web content associated with assessment types defined in Class Server to exhibit specific behaviors. There are eight collections of components for each assessment type that manifest themselves in the Class Server views, usually as a combination of text, images, and HTML form elements. The FORM element is defined in the HTML 3.2 specification and DOM Level 1. The yellow images representing form elements that appear inside the Learning Resource Editor when you insert a question with the Question Wizard are assessment components.
assessment item markup	Specialized markup syntax that is interpreted by Class Server and expressed as Web content with specialized behaviors. Normally created using the Learning Resource Editor, assessment item markup can also be manually authored in the HTML pages of a Learning Resource as individual IMG elements with specialized properties and attributes, one of which defines a Class Server assessment type for the element.
assessment item table	TABLE element containing special attributes that define it to Class Server as containing question or rubric assessment items. These specialized tables help Class Server track assessment items as compound entities in a Learning Resource. The TABLE element is defined in the HTML 3.2 specification and DOM Level 1.
authoring	In Class Server, the process of creating educational Web content in a specific format with corresponding XML metadata, and the process of testing newly formatted content to make sure it works correctly with Class Server software.
autograded question	Interrogative expression constructed of certain question component types that contain an answer key used to automatically assess student responses.
Class Server – Student	Web site that students and parents use to track, complete, and review assignments that teachers create using Class Server – Teacher.
Class Server – Teacher	Program used to create, assign, track, modify, and grade Web content for educational purposes. To create a Learning Resource in Class Server, you must create or download it using a computer configured with this program.
form element	The FORM element is defined in the HTML 3.2 specification and DOM Level 1. The yellow images representing form elements that appear inside the Learning Resource Editor when you insert a question with the Question Wizard are form elements.

Term	Description
GUID	A GUID is a Globally Unique Identifier, in the format xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx, where x is a hexadecimal digit.
Learning Resource	Information and questions structured as Web content in a defined format that is compatible with Class Server.
Learning Resource Editor	Specialized editor in Class Server – Teacher, My Learning Resources. You use this editor to create or modify Learning Resources.
Learning Resource Module	Single, MIME-encoded file containing all Web content for one or more Learning Resources. These files can be created using the Learning Resource Editor or the MakeLRM.js utility to enable one-click download of one or more Learning Resources at a time. MIME extensions are indexed at http://www.rfc-editor.org/.
MakeLRM.js	A bundling utility that combines all files and directories belonging to a Learning Resource into a single file called a Learning Resource Module, or LRM.
question-ID	Unique identifier for a question in a Learning Resource. A question-ID is generally composed of numbers, alphabetical characters, or both.
rubric	Subjective set of guidelines or criteria for evaluating and scoring student performance and for providing feedback. In Class Server, rubrics are formed from an assemblage of assessment item markup and behave in a similar fashion to questions.
working directory	Directory used by Class Server – Teacher to store Learning Resources and assignment information for offline use. This directory is synchronized with Class Server when a teacher is working online.
total question points	Value of the argument maxpts for the assessment type ITEMSCORE.
UTF-8	Encoding for Unicode. This encoding is used to create smaller files that primarily use the ASCII character set.
view	Web content that is exposed in one of six views to a Class Server client or connected device. This content is usually in the form of a Learning Resource.