



IMS Question and Test Interoperability Addendum

Version 2.1 Public Draft Specification

Copyright © 2008 IMS Global Learning Consortium, Inc. All Rights Reserved.

The IMS Logo is a registered trademark of IMS GLC

Document Name: IMS Question and Test Interoperability Addendum

Date Issued: 28 March 2008

Caution: this specification is incomplete in its current state. The IMS QTI project group is in the process of evolving this specification based on input from market participants. Suppliers of products and services are encouraged to participate by contacting Mark McKell at mmckell@imglobal.org. This specification will be superseded by an updated release based on the input of the project group participants.

Please note that supplier's claims as to implementation of QTI v2.1 and conformance to it HAVE NOT BEEN VALIDATED by IMS GLC. While such suppliers are likely well-intentioned, IMS GLC member organizations have not yet put in place the testing process to validate these claims. IMS GLC currently grants a conformance mark to the Common Cartridge profile of QTI v1.2.1. The authoritative source of products and services that meet this conformance is contained in the IMS online product directory

<http://www.imglobal.org/ProductDirectory/directory.cfm>

Thank you for your interest in and support of IMS QTI.

IPR and Distribution Notices

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imglobal.org/ipr/imsipr_policyFinal.pdf.

Copyright © 2008 IMS Global Learning Consortium. All Rights Reserved.

If you wish to copy or distribute this document, you must complete a valid Registered User license registration with IMS and receive an email from IMS granting the license to distribute the specification. To register, follow the instructions on the IMS website: <http://www.imglobal.org/specificationdownload.cfm>.

This document may be copied and furnished to others by Registered Users who have registered on the IMS website provided that the above copyright notice and this paragraph are included on all such copies. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to IMS, except as

needed for the purpose of developing IMS specifications, under the auspices of a chartered IMS project group.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: <http://www.imsglobal.org/license.html>.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

Overview

Based on feedback from implementers and reviewers of the specification, the QTI project group has made bug fixes and updates to some of the examples, the specification documents, and the XML schema. The changes are summarized in this Addendum document and the updated examples and schema are also made available on the IMS website:

<http://www.imsglobal.org/question/index.html>

Table of Contents

[Overview](#)

[1. Implementation Guide](#)

[1. §3.2 Simple Items](#)

[1. §3.6 Adaptive Items](#)

[1. §3.8.6 Number Formatting](#)

[1. §4 Tests \(Assessments\)](#)

[2. Assessment Test, Section and Item Information Model](#)

[2. §5 Item Variables](#)

[2. §5.1 Response Variables](#)

[2. §7.1 Simple Interactions](#)

[2. §7.2 Text-based Interactions](#)

[2. §8.2 Generalized Response Processing](#)

[2. §10.3 Template Processing](#)

[2. §11.2 Test Structure](#)

[2. §12 Outcome Processing](#)

[2. §15.2 Expressions Used only in Outcomes Processing](#)

[2. §15.3 Operators](#)

[2. §16 Item and Test Fragments](#)

[2. §17 Basic Data Types](#)

[3. Meta-data and Usage Data](#)

[3. §3 New meta-data elements in IMS QTI v2.0](#)

[4. Results Reporting](#)

[4. §3 Assessment Result](#)

[5. Integration Guide](#)

[5. §5 Simple Sequencing](#)

[6. XML Binding](#)

[7. Example Files](#)

[8. Appendix](#)

[About this document](#)

1. Implementation Guide

§3.2 Simple Items

A new example has been added:

Writing a Postcard with Rubric Information

http://www.imsglobal.org/question/ktiv2p1pd2/examples/items/extended_text_rubric.xml

A `rubricBlock` can be used to add instructions about the way the item should be scored by a human scorer. The view attribute is used to indicate that the information should only be made visible to users in certain roles.

§3.6 Adaptive Items

*The following example now has an updated description: **Mexican President with adaptive feedback***

http://www.imsglobal.org/question/ktiv2p1pd2/examples/items/feedback_adaptive.xml

In this adaptive example, the candidate receives different feedback for each attempt. The item allows for four incorrect attempts before the correct answer is provided. This example takes advantage of the built-in response variable `numAttempts` and uses the persistence of outcome values in adaptive items to keep track of the responses given so far (using the variable `PREVIOUSRESPONSES`) adjusting the inline feedback accordingly.

§3.8.6 Number Formatting

A typographical error in row 6 of the table has been corrected. The example format specification demonstrating the hyphen flag has been corrected from `%-8f` to `%-8i`

§4 Tests (Assessments)

In the explanation of Golden (required) Items and Sections the first paragraph ("This example shows...") was copied in error and has been deleted.

Several of the examples used the placeholder name Steve's Test. These have been updated as follows:

Controlling the duration of an item attempt

<http://www.imsglobal.org/question/ktiv2p1pd2/examples/tests/rtest09.xml>

Early termination of test based on accumulated item outcomes

<http://www.imsglobal.org/question/ktiv2p1pd2/examples/tests/rtest10.xml>

Branching based on the response to an assessmentItem

<http://www.imsglobal.org/question/ktiv2p1pd2/examples/tests/rtest13.xml>

Mapping item outcomes prior to aggregation

<http://www.imsglobal.org/question/ktiv2p1pd2/examples/tests/rtest26.xml>

The following example also has an updated description:

Branching based on the response to an assessmentItem

<http://www.imsglobal.org/question/ktiv2p1pd2/examples/tests/rtest13.xml>

This example shows the support for branching based on the response to an assessmentItem. The example uses the preCondition element and the branchRule element.

The preCondition element sets the conditions that need to be met for an assessmentItem or assessmentSection to be displayed. In nonlinear mode, pre-conditions are ignored.

The branchRule element contains a rule, evaluated during the test, for setting an alternative target as the next item or section. As with preconditions, branch rules are ignored in nonlinear mode. The second branchRule element contains a special targetItem EXIT_SECTION which means exit this section of the test.

2. Assessment Test, Section and Item Information Model

The use of the {ordered} constraint in the definitions was generally inconsistent and has caused some confusion. The {ordered} constraint has now been added to all definitions where the order of multiple attributes or child objects is significant. Although this information is not represented in the resulting XML schema, it may be used by implementers when determining the appropriate data structures to use internally in their applications and should also be used when comparing instances of the declared classes. For example, in the definition of assessmentItem at the beginning of §4 the order of the optional stylesheet and modalFeedback children is significant whereas the order of responseDeclarations (for example) is not. For completeness, the affected definitions are listed in the Appendix.

§5 Item Variables

The ordering constraint for defaultValue.value is now clarified as follows:

Contains : value [1..*] {ordered}

The order of the values is significant only if the variable being set has ordered cardinality.

§5.1 Response Variables

The first paragraph of the definition of responseDeclaration is clarified as follows:

Response variables are declared by response declarations and bound to interactions in the itemBody. Each response variable declared may be bound to one and only one interaction.

The ordering constraint for correctResponse.value is now clarified as follows:

Contains : value [1..*] {ordered}

The order of the values is significant only when the response is of ordered cardinality.

§7.1 Simple Interactions

The first paragraph of the definition of choiceInteraction has had the sentence "There is no corresponding minimum number of choices." removed. This text is inconsistent with the new minChoices attribute that was introduced in the second public draft.

§7.2 Text-based Interactions

The definition of inlineChoice given was missing the declaration of its children. This affected the resulting XML schema documents causing validation failures when this element was used. This has been corrected as follows:

Class : inlineChoice (choice)

Associated classes:

inlineChoiceInteraction

A simple run of text to be displayed to the user, may be subject to variable value substitution with printedVariable.

Contains : textOrVariable [*] {ordered}

§8.2 Generalized Response Processing

As well as adding the {ordered} constraint to the definition of responseProcessing.responseRule, the following clarifying statement has been added to emphasize the change:

Result rules are followed in the order given. Variables updated by a rule take their new value when evaluated as part of any following rules.

§10.3 Template Processing

As well as adding the {ordered} constraint to the definition of templateProcessing.templateRule, the following clarifying statement has been added to emphasize the change:

Template rules are followed in the order given. Variables updated by a rule take their new value when evaluated as part of any following rules.

§11.2 Test Structure

The definition of the weight class was inconsistent with the definition of the corresponding element in the XML schema and the examples were mutually inconsistent. This was due to errors in the schema and the examples which have now been corrected. The published definition is unchanged.

§12 Outcome Processing

As well as adding the {ordered} constraint to the definition of outcomeProcessing.outcomeRule, the following clarifying statement has been added to emphasize the change:

Outcome rules are followed in the order given. Variables updated by a rule take their new value when evaluated as part of any following rules.

§15.2 Expressions Used only in Outcomes Processing

The definition of the `weightIdentifier` attribute of the class `testVariables` has been clarified as follows:

Attribute : `weightIdentifier` [0..1]: identifier

If specified, the defined weight is applied to each variable as described in the definition of weightIdentifier for a single variable. The behavior of this attribute is only defined if the `baseType` attribute is float or omitted. When a weighting is specified the result of the expression always has base-type float. Note that this option is incompatible with `baseType` integer. This restriction ensures that the value of the `baseType` attribute remains consistent with the resulting container type.

§15.3 Operators

The handling of `NULL` values by the index operator has been clarified in the definition of the index class as follows:

The index operator takes a sub-expression with an ordered container value and any base-type. The result is the *n*th value of the container. The result has the same base-type as the sub-expression but single cardinality. The first value of a container has index 1, the second 2 and so on. *n* must be a positive integer. If *n* exceeds the number of values in the container (or the sub-expression is `NULL`) then the result of the index operator is `NULL`.

§16 Item and Test Fragments

The definition of the `include` class was confusing and its representation was incorrect in the schema. The introductory paragraph has been reworded as follows:

An Item Fragment is part of an item that is managed independently of the items that depend on it. Similarly, a Test Fragment is part of a test that is managed independently of the tests that depend on it. Fragments are packaged as separate resources and can be transported independently. A fragment may appear anywhere in the model where one of the following abstract classes may appear: flowStatic, inlineStatic, blockStatic, responseRule, sectionPart or outcomeRule. For example, an item fragment may be a division of the itemBody represented by an instance of the div class or a single interaction.

The class definition has been updated to show the correct list of classes from which it inherits:

Class : include (blockStatic, flowStatic, inlineStatic, outcomeRule, responseRule, sectionPart)

§17 Basic Data Types

The definition of the identifier type has been updated to reflect the new use of the period character as a prefix delimiter. This change replaces the reference to "future ues" and re-enforces the fact that the period must not be used except when used as a prefix delimiter:

Datatype: identifier

An identifier is simply a logical reference to another object in the item, such as an itemVariable or choice. An identifier is a string of characters that must start with a Letter or an underscore ('_') and contain only Letters, underscores, hyphens ('-'), period ('.', a.k.a. full-stop), Digits, CombiningChars and Extenders. Identifiers containing the period character are reserved for use in prefixing, as described in the definition of variable. The character classes Letter, Digit, CombiningChar and Extender are defined in the Extensible Markup Language (XML) 1.0 (Second Edition) [XML]. Note particularly that identifiers may not contain the colon (':') character. Identifiers should have no more than 32 characters for compatibility with version 1. They are always compared case-sensitively.

3. Meta-data and Usage Data

§3 New meta-data elements in IMS QTI v2.0

The definition of the enumeration `interactionType` has been updated to include the `mediaInteraction` which was omitted in error. This change also affects the XML schema.

4. Results Reporting

The {ordering} constraint has been applied where necessary (see above for more details).

§3 Assessment Result

The usage of correctResponse in responseVariable has been clarified as follows:

Contains : correctResponse [0..1]

The correct response may be output as part of the report if desired. Systems are not limited to reporting correct responses declared in responseDeclarations. For example, a correct response may be set by a templateRule or may simply have been suppressed from the declaration passed to the delivery engine (e.g., for security).

As well as adding the {ordered} constraint the definition of candidateResponse.value has been updated as follows:

Contains : value [*] {ordered}

The value(s) of the response variable. A NULL value, resulting from no response, is indicated by the absence of any value. The order of the values is significant only if the response was declared with ordered cardinality.

As well as adding the {ordered} constraint the definition of outcomeVariable.value has been updated as follows:

Contains : value [*] {ordered}

The value(s) of the outcome variable. The order of the values is significant only if the outcome was declared with ordered cardinality.

As well as adding the {ordered} constraint the definition of templateVariable.value has been updated as follows:

Contains : value [*] {ordered}

The value(s) of the template variable. The order of the values is significant only if the template variable was declared with ordered cardinality.

5. Integration Guide

§5 Simple Sequencing

The first sentence of the last paragraph has been updated to read:

Note that assessment items have resource type `imsqti_item_xmlv2p1` (or `imsqti_item_xmlv2p0`) and not `webcontent`.

6. XML Binding

The XML schema has been updated to include fixes (as described above) for the correct usage of `xi:include` and for the definitions of the `inlineChoice` and `weight` elements.

7. Example Files

The example files have been corrected to validate against the updated schema (where necessary) and to reflect current best practice in the use of the specification. These changes have affected the entire example set.

The main issues addressed were:

- *Consistent use of the 2.1 schema in all examples including removal of local copies and associated references of the schema definition*
- *Updates to the manifest files in the packaged examples to use an updated and corrected schema for IMS Meta-data*
- *Corrections to examples using the weight element*
- *Corrections to some examples using time limits incorrectly*
- *Addition of descriptive titles to some test examples*

In addition to the above, a number of more minor typographical errors in the examples have been corrected.

8. Appendix

Elements that are newly labelled {ordered} in the specification:

assessmentItem.modalFeedback

assessmentItem.stylesheet

assessmentSection.rubricBlock

assessmentSection.sectionPart

assessmentTest.testFeedback

assessmentTest.testPart

associateInteraction.simpleAssociableChoice

atomicBlock.inline

candidateResponse.value

caption.inline

choiceInteraction.simpleChoice

colgroup.col

contains.expression

correctResponse.value

customOperator.expression

dd.flow

defaultValue.value

delete.expression

div.flow

divide.expression

dl.dlElement

dt.inline

durationGTE.expression

durationLT.expression

equal.expression

gapMatchInteraction.blockStatic

gapMatchInteraction.gapChoice

gapText.textOrVariable

graphicAssociateInteraction.associableHotspot

graphicGapMatchInteraction.associableHotspot

graphicGapMatchInteraction.gapImg

graphicOrderInteraction.hotspotChoice

gt.expression

gte.expression

hottext.inlineStatic

hottextInteraction.blockStatic

inlineChoice.textOrVariable

inlineChoiceInteraction.inlineChoice

integerDivide.expression

integerModulus.expression

interpolationTable.interpolationTableEntry

itemBody.block

li.flow

lt.expression

lte.expression

match.expression

matchTable.matchTableEntry

member.expression

modalFeedback.flowStatic

object.objectFlow

ol.li

orderInteractino.simpleChoice

ordered.expression

outcomeCondition.outcomeElseIf

outcomeElse.outcomeRule

outcomeElseIf.outcomeRule

outcomeIf.outcomeRule

outcomeProcessing.outcomeRule

outcomeProcessingFragment.outcomeRule

outcomeVariable.value

positionObjectStage.positionObjectInteraction

power.expression

prompt.inlineStatic

responseElse.responseRule

responseElseIf.responseRule

responseIf.responseRule

responseProcessing.responseRule

responseProcessingFragment.responseRule

responseVariable.choiceSequence

sectionPart.branchRule

sectionPart.preCondition

simpleAssociableChoice.flowStatic

simpleBlock.block

simpleChoice.flowStatic

simpleInline.inline

simpleMatchSet.simpleAssociableChoice

stringMatch.expression

substring.expression

subtract.expression

table.col

table.colgroup

table.tbody

tableCell.flow

tbody.tr

templateBlock.blockStatic

templateBlock.inlineStatic

templateCondition.templateElseIf

templateElse.templateRule

templateElseIf.templateRule

templateIf.templateRule

templateProcessing.templateRule

templateVariable.value

testFeedback.flowStatic

testPart.assessmentSection

testPart.branchRule

testPart.preCondition

testPart.testFeedback

tfoot.tr

thead.tr

tr.tableCell

ul.li

About this Document

Title	IMS Question and Test Interoperability Addendum
Editors	Steve Lay (University of Cambridge), Pierre Gorissen (SURF)
Co-chairs	Steve Lay (University of Cambridge), Pierre Gorissen (SURF)
Version	v2.1 (Public Draft v2.0 update)
Version Date	28 March 2008
Status	Public Draft
Summary	This Addendum provides an overview of the updates, bug fixes, corrections to the IMS Question and Test Interoperability v2.1 specification.
Revision Information	28 March 2008
Purpose	This document is circulated for public adoption. Organizations are encouraged to implement this version of the specification and to provide feedback to the project group.
Document Location	http://www.imsglobal.org/question/index.html

IMS Global Learning Consortium, Inc. ("IMS GLC") is publishing the information contained in this IMS Question and Test Interoperability Addendum ("Specification") for purposes of scientific, experimental, and scholarly collaboration only.

IMS GLC makes no warranty or representation regarding the accuracy or completeness of the Specification. This material is provided on an "As Is" and "As Available" basis.

The Specification is at all times subject to change and revision without notice.

It is your sole responsibility to evaluate the usefulness, accuracy, and completeness of the Specification as it relates to you.

IMS GLC would appreciate receiving your comments and suggestions.

Please contact IMS GLC through our website at <http://www.imsglobal.org>

Please refer to Document Name: IMS Question and Test Interoperability Addendum Date: 28 March 2008