Package 'rqti'

May 14, 2026

Title Create Tests According to QTI 2.1 Standard

Version 1.2.1

Description Create tests and tasks compliant with the Question & Test Interoperability (QTI) information model version 2.1. Input sources are Rmd/md description files or S4-class objects. Output formats include standalone zip or xml files. Supports the generation of basic task types (single and multiple choice, order, pair association, matching tables, filling gaps and essay) and provides a comprehensive set of attributes for customizing tests.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Imports htmltools, xml2, yaml, rmarkdown, servr, rstudioapi, stringr, methods, magrittr, httr2, curl, knitr, getPass, keyring, zip, textutils, lifecycle, exams, base64enc

Suggests covr, dplyr, testthat (>= 3.0.0), XML, readr, RCurl, chromote, callr, pkgload, httpuv

Config/testthat/edition 3

Config/testthat/parallel false

URL https://github.com/shevandrin/rqti, https://shevandrin.github.io/rqti/

BugReports https://github.com/shevandrin/rqti/issues

Collate 'rqti.R' 'QtiMetadata.R' 'ModalFeedback.R' 'AssessmentItem.R' 'AssessmentSection.R' 'AssessmentTest.R' 'AssessmentTestOpal.R' 'AssessmentTestOpenOlat.R' 'Choice.R' 'CorrectFeedback.R' 'MatchTable.R' 'DirectedPair.R' 'Entry.R' 'Essay.R' 'Gap.R' 'InlineChoice.R' 'LMS.R' 'MultipleChoice.R' 'MultipleChoiceTable.R' 'NumericGap.R' 'OneInColTable.R' 'OneInRowTable.R' 'Opal.R' 'Ordering.R' 'SingleChoice.R' 'TextGap.R' 'TextGapOpal.R' 'WrongFeedback.R' 'character.R' 'exams_integration.R' 'extract_results.R' 'helpers.R' 'knit_functions.R' 'object_builder.R' 'qti_task.R' 'qti_test.R' 'response_processing.R' 'rqti-package.R' 'rqti_project.R' 'section_builder.R' 'utils-pipe.R' 'verify_qti.R' 'zzz.R'

Depends R (>= 2.10)

NeedsCompilation no

Author Andrey Shevandrin [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0003-0807-2546>),
Petr Bondarenko [ctb] (ORCID: <https://orcid.org/0000-0002-5154-9664>),
Ivonne Ojeda [ctb],
Johannes Titz [aut, cph] (ORCID:
<https://orcid.org/0000-0002-1102-5719>),
Brian Mottershead [cph] (Author of QTIJS library),
Stiftung für Innovation in der Hochschullehre [fnd]

Maintainer Andrey Shevandrin <shevandrin@gmail.com>

Repository CRAN

Date/Publication 2026-05-14 17:40:03 UTC

Contents

`AssessmentItem-class` *Class AssessmentItem*

Description

Abstract class `AssessmentItem` is responsible for creating a root element 'assessmentItem' in XML task description according to QTI 2.1. This class is not meant to be instantiated directly; instead, it serves as a base for derived classes.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`assessmentSection` *Create an object AssessmentSection*

Description

Create an AssessmentSection `rqti`-object as part of a test content

Usage

```
assessmentSection(
  assessment_item,
  identifier = generate_id(type = "section"),
  title = identifier,
  selection = NA_integer_,
  time_limit = NA_integer_,
  visible = TRUE,
  shuffle = FALSE,
  max_attempts = NA_integer_,
  allow_comment = TRUE
)
```

Arguments

`assessment_item`	A list containing AssessmentSection and/or Assessment item objects, such as SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, and DirectedPair.
`identifier`	A character value indicating the identifier of the test file. By default, it is generated as 'id_section_dddd', where dddd represents random digits.
`title`	A character value, optional, representing the file title. By default, it takes the value of the `identifier`.
`selection`	An integer value, optional, defining how many children of the section are delivered in the test. Default is `NA_integer_`, meaning "no selection".
`time_limit`	An integer value, optional, controlling the amount of time in munutes a candidate is allowed for this part of the test.
`visible`	A boolean value, optional, indicating whether the title of this section is shown in the hierarchy of the test structure. Default is `TRUE`.
`shuffle`	A boolean value, optional, responsible for randomizing the order in which the assessment items and subsections are initially presented to the candidate. Default is `FALSE`.
`max_attempts`	An integer value, optional, enabling the maximum number of attempts allowed for a candidate to pass this section.
`allow_comment`	A boolean value, optional, enabling candidates to leave comments on each question of the section. Default is `TRUE`.

Value

An object of class AssessmentSection.

See Also

section(), test(), test4opal()

Examples

```
sc <- singleChoice(prompt = "Question", choices = c("A", "B", "C"))
es <- essay(prompt = "Question")
# Since ready-made S4 "AssessmentItem" objects are taken, in this example a
#permanent section consisting of two tasks is created.
s <- assessmentSection(list(sc, es), title = "Section with nonrandomized tasks")
```

AssessmentSection-class

Class "AssessmentSection"

Description

Class AssessmentSection is responsible for forming a section in the test XML specification according to QTI 2.1.

Slots

identifier A character representing the unique identifier of the assessment section. By default, it is generated as 'id_section_dddd', where dddd represents random digits.

title A character representing the title of the section in the test. By default, it takes the value of the identifier.

time_limit A numeric value, optional, controlling the amount of time *in munutes* a candidate is allowed for this part of the test.

visible A boolean value, optional. If TRUE, it shows this section in the hierarchy of the test structure. Default is TRUE.

assessment_item A list containing AssessmentSection and/or Assessment item objects, such as SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, and DirectedPair.

shuffle A boolean value, optional, responsible for randomizing the order in which the assessment items and subsections are initially presented to the candidate. Default is FALSE.

selection A numeric value, optional, defining how many children of the section are delivered in the test.

max_attempts A numeric value, optional, enabling the maximum number of attempts a candidate is allowed to pass in this section.

allow_comment A boolean value, optional, enabling to allow the candidate to leave comments in each question of the section. Defautl is TRUE.

See Also

section(), test(), test4opal(), AssessmentTest, AssessmentTestOpal.

Examples

```
sc1 <- new("SingleChoice", prompt = "Example task 1", title = "SC1",
            identifier = "q1", choices = c("a", "b", "c"))
sc2 <- new("SingleChoice", prompt = "Example task 2", title = "SC2",
            identifier = "q2", choices = c("A", "B", "C"))
sc3 <- new("SingleChoice", prompt = "Example task 3", title = "SC3",
            identifier = "q3", choices = c("aa", "bb", "cc"))
exam_section <- new("AssessmentSection",
                    identifier = "sec_id",
                    title = "Section",
                    time_limit = 20,
                    visible = FALSE,
                    assessment_item = list(sc1, sc2, sc3),
                    shuffle = FALSE,
                    selection = 1,
                    max_attempts = 1,
                    allow_comment = FALSE)
```

---

assessmentTest *Create an object AssessmentTest*

Description

Create an AssessmentTest rqti-object.

Usage

```
assessmentTest(
  section,
  identifier = generate_id(type = "test"),
  title = identifier,
  time_limit = NULL,
  max_attempts = 1L,
  fallback_titles = "generic",
  academic_grading = NULL,
  grade_label = c(en = "Grade", de = "Note"),
  table_label = c(en = "Grade", de = "Note"),
  navigation_mode = "nonlinear",
  submission_mode = "individual",
  allow_comment = TRUE,
  rebuild_variables = TRUE,
  metadata = qtiMetadata(),
  stylesheet_path = NULL,
  points = NA_real_
)
```

Arguments

`section`	A list containing AssessmentSection objects.
`identifier`	A character value indicating the identifier of the test file. By default, it is generated as 'id_test_dddd', where dddd represents random digits.
`title`	A character value, optional, representing the file title. By default, it takes the value of the identifier.
`time_limit`	An integer value, optional, controlling the time given to a candidate for the test in minutes. Default is `NULL`.
`max_attempts`	An integer value, optional, indicating the maximum number of attempts allowed for the candidate. Default is 1.
`fallback_titles`	A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".
`academic_grading`	A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test. Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme. The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.
`grade_label`	A character value, optional; a short message that shows with a grade in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
`table_label`	A character value, optional; a concise message to display as the column title of the grading table in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
`navigation_mode`	A character value, optional, determining the general paths that the candidate may have during the exam. Two mode options are possible: - 'linear': Candidate is not allowed to return to previous questions. - 'nonlinear': Candidate is free to navigate; used by default.
`submission_mode`	A character value, optional, determining when the candidate's responses are submitted for response processing. One of two mode options is possible: - 'individual': Submit candidates' responses on an item-by-item basis; used by default. - 'simultaneous': Candidates' responses are submitted all together by the end of the test.
`allow_comment`	A boolean, optional, enabling the candidate to leave comments in each question. Default is `TRUE`.

`rebuild_variables`
: A boolean, optional, enabling the recalculation of variables and reshuffling the order of choices for each item-attempt. Default is `TRUE`.

`metadata`
: An object of class QtiMetadata that holds metadata information about the test. By default it creates QtiMetadata object. See `qtiMetadata()`.

`stylesheet_path`
: A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When `academic_grading` is set, the default stylesheet `styles/rqti.css` is included automatically; a user-defined stylesheet is added in addition and may override default styles.

`points`
: Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

Value

An AssessmentTest object.

See Also

`test()`, `test4opal()`, `section()`, AssessmentTest, AssessmentSection

Examples

```
sc <- sc <- singleChoice(prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
s <- section(c(sc, es), title = "Section with nonrandomized tasks")
t <- assessmentTest(list(s), title = "Example of the Exam")
```

`AssessmentTest-class` *Class "AssessmentTest"*

Description

Class `AssessmentTest` is responsible for creating XML exam files according to the QTI 2.1 standard.

Details

Test consists of one or more sections. Each section can have one or more questions/tasks and/or one or more sub sections.

Slots

`identifier` A character representing the unique identifier of the assessment test. By default, it is generated as 'id_test_dddd', where dddd represents random digits.

`title` A character representing the title of the test. By default, it takes the value of the identifier.

`points` Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

`test_part_identifier` A character representing the identifier of the test part.

`navigation_mode` A character value, optional, determining the general paths that the candidate may have during the exam. Possible values:

- "linear" - candidate is not allowed to return to the previous questions.
- "nonlinear" - candidate is free to navigate. This is used by default.

`submission_mode` A character value, optional, determining when the candidate's responses are submitted for response processing. Possible values:

- "individual" - submit candidates' responses on an item-by-item basis. This is used by default.
- "simultaneous" - candidates' responses are submitted all together by the end of the test.

`section` A list containing one or more AssessmentSection objects.

`time_limit` A numeric value, optional, controlling the amount of time in minutes which a candidate is allowed for this part of the test.

`max_attempts` A numeric value, optional, enabling the maximum number of attempts that a candidate is allowed to pass.

`allow_comment` A boolean value, optional, enabling to allow candidates to leave comments in each question.

`rebuild_variables` A boolean value, optional, enabling to recalculate variables and reshuffle the order of choices for each item-attempt.

`fallback_titles` A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".

`academic_grading` A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test.

Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme.

The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.

`grade_label` A character value, optional, representing a short message to display with a grade in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`table_label` A character value, optional, representing a concise message to display as the column title of the grading table in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`stylesheet_path` A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When `academic_grading` is set, the default stylesheet `styles/rqti.css` is included automatically; a user-defined stylesheet is added in addition and may override default styles.

`metadata` An object of class QtiMetadata that holds metadata information about the test.

See Also

AssessmentSection, AssessmentTestOpal, `test()`, `test4opal()`, `section()`.

Examples

```
# This example creates test 'exam' with one section 'exam_section' which
# consists of two questions/tasks: essay and single choice types
task1 <- new("Essay", prompt = "Test task", title = "Essay",
             identifier = "q1")
task2 <- new("SingleChoice", prompt = "Test task", title = "SingleChoice",
             choices = c("A", "B", "C"), identifier = "q2")
exam_section <- new("AssessmentSection", identifier = "sec_id",
                    title = "section", assessment_item = list(task1, task2))
exam <- new("AssessmentTest",
            identifier = "id_test_1234",
            title = "Example of Exam",
            navigation_mode = "linear",
            submission_mode = "individual",
            section = list(exam_section),
            time_limit = 90,
            max_attempts = 1,
            grade_label = "Preliminary grade")
```

---

assessmentTestOpal *Create an object AssessmentTestOpal*

Description

Create an AssessmentTestOpal `rqti`-object.

Usage

```
assessmentTestOpal(
  section,
  identifier = generate_id(type = "test"),
  title = identifier,
  time_limit = NULL,
  max_attempts = 1L,
  fallback_titles = "generic",
  academic_grading = NULL,
  grade_label = c(en = "Grade", de = "Note"),
```

```
  table_label = c(en = "Grade", de = "Note"),
  navigation_mode = "nonlinear",
  submission_mode = "individual",
  allow_comment = TRUE,
  rebuild_variables = TRUE,
  show_test_time = TRUE,
  calculator = NA_character_,
  mark_items = TRUE,
  keep_responses = FALSE,
  metadata = qtiMetadata(),
  points = NA_real_
)
```

Arguments

`section`	A list containing AssessmentSection objects.
`identifier`	A character value indicating the identifier of the test file. By default, it is generated as 'id_test_dddd', where dddd represents random digits.
`title`	A character value, optional, representing the file title. By default, it takes the value of the identifier.
`time_limit`	An integer value, optional, controlling the time given to a candidate for the test in minutes. Default is `NULL`.
`max_attempts`	An integer value, optional, indicating the maximum number of attempts allowed for the candidate. Default is 1.
`fallback_titles`	A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".
`academic_grading`	A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test. Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme. The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.
`grade_label`	A character value, optional; a short message that shows with a grade in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
`table_label`	A character value, optional; a concise message to display as the column title of the grading table in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.

navigation_mode
: A character value, optional, determining the general paths that the candidate may have during the exam. Two mode options are possible: - 'linear': Candidate is not allowed to return to previous questions. - 'nonlinear': Candidate is free to navigate; used by default.

submission_mode
: A character value, optional, determining when the candidate's responses are submitted for response processing. One of two mode options is possible: - 'individual': Submit candidates' responses on an item-by-item basis; used by default. - 'simultaneous': Candidates' responses are submitted all together by the end of the test.

allow_comment
: A boolean, optional, enabling the candidate to leave comments in each question. Default is TRUE.

rebuild_variables
: A boolean, optional, enabling the recalculation of variables and reshuffling the order of choices for each item-attempt. Default is TRUE.

show_test_time
: A boolean, optional, determining whether to show candidate elapsed processing time without a time limit. Default is TRUE.

calculator
: A character value, optional, determining whether to show a calculator to the candidate. Possible values: - "simple" - "scientific".

mark_items
: A boolean, optional, determining whether to allow candidate marking of questions. Default is TRUE.

keep_responses
: A boolean, optional, determining whether to save the candidate's answers from the previous attempt. Default is FALSE.

metadata
: An object of class QtiMetadata that holds metadata information about the test. By default it creates QtiMetadata object. See qtiMetadata().

points
: Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

Value

An AssessmentTestOpal object.

See Also

test(), test4opal(), section(), assessmentTest(), AssessmentTest, AssessmentSection

Examples

```
sc <- sc <- singleChoice(prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
s <- section(c(sc, es), title = "Section with nonrandomized tasks")
t <- assessmentTest(list(s), title = "Example of the Exam")
```

`AssessmentTestOpal-class` *Class "AssessmentTestOpal"*

Description

Class `AssessmentTestOpal` is responsible for creating XML exam files according to the QTI 2.1 standard for LMS Opal.

Details

Test consists of one or more sections. Each section can have one or more questions/tasks and/or one or more sub sections.

Slots

`identifier` A character representing the unique identifier of the assessment test. By default, it is generated as 'id_test_dddd', where dddd represents random digits.

`title` A character representing the title of the test. By default, it takes the value of the identifier.

`points` Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

`test_part_identifier` A character representing the identifier of the test part.

`navigation_mode` A character value, optional, determining the general paths that the candidate may have during the exam. Possible values:

- "linear" - candidate is not allowed to return to the previous questions.
- "nonlinear" - candidate is free to navigate. This is used by default.

`submission_mode` A character value, optional, determining when the candidate's responses are submitted for response processing. Possible values:

- "individual" - submit candidates' responses on an item-by-item basis. This is used by default.
- "simultaneous" - candidates' responses are submitted all together by the end of the test.

`section` A list containing one or more AssessmentSection objects.

`time_limit` A numeric value, optional, controlling the amount of time in minutes which a candidate is allowed for this part of the test.

`max_attempts` A numeric value, optional, enabling the maximum number of attempts that a candidate is allowed to pass.

`allow_comment` A boolean value, optional, enabling to allow candidates to leave comments in each question.

`rebuild_variables` A boolean value, optional, enabling to recalculate variables and reshuffle the order of choices for each item-attempt.

`fallback_titles` A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".

`academic_grading` A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test.

Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme.

The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.

`grade_label` A character value, optional, representing a short message to display with a grade in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`table_label` A character value, optional, representing a concise message to display as the column title of the grading table in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`stylesheet_path` A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When `academic_grading` is set, the default stylesheet `styles/rqti.css` is included automatically; a user-defined stylesheet is added in addition and may override default styles.

`metadata` An object of class QtiMetadata that holds metadata information about the test.

`show_test_time` A boolean value, optional, determining whether to show the candidate elapsed processing time without time limit. Default is `FALSE`.

`calculator` A character value, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific".

`mark_items` A boolean value, optional, determining whether to allow candidate marking of questions. Default is `TRUE`.

`keep_responses` A boolean value, optional, determining whether to save candidate's answers from the previous attempt. Default is `FALSE`.

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

See Also

AssessmentSection, AssessmentTest, `test()`, `test4opal()`, `section()`.

Examples

```
# This example creates test 'exam' with one section 'exam_section' which
# consists of two questions/tasks: essay and single choice types
task1 <- new("Essay", prompt = "Test task", title = "Essay",
             identifier = "q1")
task2 <- new("SingleChoice", prompt = "Test task", title = "SingleChoice",
             choices = c("A", "B", "C"), identifier = "q2")
exam_section <- new("AssessmentSection", identifier = "sec_id",
```

```
                    title = "section", assessment_item = list(task1, task2))
exam <- new("AssessmentTestOpal",
            identifier = "id_test_1234",
            title = "Example of Exam",
            navigation_mode = "linear",
            submission_mode = "individual",
            section = list(exam_section),
            time_limit = 90,
            max_attempts = 1,
            grade_label = "Preliminary grade",
            show_test_time = TRUE,
            calculator = "scientific-calculator",
            mark_items = TRUE,
            files = "text_book.pdf")
```

`assessmentTestOpenOlat`

Create an object AssessmentTestOpenOlat

Description

Create an AssessmentTestOpenOlat `rqti`-object.

Usage

```
assessmentTestOpenOlat(
  section,
  identifier = generate_id(type = "test"),
  title = identifier,
  time_limit = NULL,
  max_attempts = 1L,
  fallback_titles = "generic",
  academic_grading = NULL,
  grade_label = c(en = "Grade", de = "Note"),
  table_label = c(en = "Grade", de = "Note"),
  navigation_mode = "nonlinear",
  submission_mode = "individual",
  allow_comment = TRUE,
  rebuild_variables = TRUE,
  metadata = qtiMetadata(),
  points = NA_real_,
  cancel = FALSE,
  suspend = FALSE,
  scoreprogress = FALSE,
  questionprogress = FALSE,
  maxscoreitem = TRUE,
  menu = TRUE,
  titles = TRUE,
```

```
  notes = FALSE,
  hidelms = TRUE,
  hidefeedbacks = FALSE,
  blockaftersuccess = FALSE,
  attempts = 1L,
  anonym = FALSE,
  manualcorrect = FALSE
)
```

Arguments

section	A list containing AssessmentSection objects.
identifier	A character value indicating the identifier of the test file. By default, it is generated as 'id_test_dddd', where dddd represents random digits.
title	A character value, optional, representing the file title. By default, it takes the value of the identifier.
time_limit	An integer value, optional, controlling the time given to a candidate for the test in minutes. Default is 90 minutes.
max_attempts	An integer value, optional, indicating the maximum number of attempts allowed for the candidate. Default is 1.
fallback_titles	A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".
academic_grading	A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test. Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme. The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.
grade_label	A character value, optional; a short message that shows with a grade in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
table_label	A character value, optional; a concise message to display as the column title of the grading table in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
navigation_mode	A character value, optional, determining the general paths that the candidate may have during the exam. Two mode options are possible: - 'linear': Candidate

is not allowed to return to previous questions. - 'nonlinear': Candidate is free to navigate; used by default.

`submission_mode`
: A character value, optional, determining when the candidate's responses are submitted for response processing. One of two mode options is possible: - 'individual': Submit candidates' responses on an item-by-item basis; used by default. - 'simultaneous': Candidates' responses are submitted all together by the end of the test.

`allow_comment`
: A boolean, optional, enabling the candidate to leave comments in each question. Default is `TRUE`.

`rebuild_variables`
: A boolean, optional, enabling the recalculation of variables and reshuffling the order of choices for each item-attempt. Default is `TRUE`.

`metadata`
: An object of class QtiMetadata that holds metadata information about the test. By default it creates QtiMetadata object. See `qtiMetadata()`.

`points`
: Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

`cancel`
: A logical value, optional, indicating whether participants are allowed to cancel an exam after starting it. Default is `FALSE`.

`suspend`
: A logical value, optional, indicating whether participants are allowed to suspend an exam after starting it and continue later. Default is `FALSE`.

`scoreprogress`
: A logical value, optional, indicating whether the progress of the score achieved so far should be displayed during the exam. Default is `FALSE`.

`questionprogress`
: A logical value, optional, indicating whether the number of solved questions should be displayed during the exam. Default is `FALSE`.

`maxscoreitem`
: A logical value, optional, indicating whether the maximum score of an item should be displayed. Default is `TRUE`.

`menu`
: A logical value, optional, indicating whether the menu should be displayed during the exam. Default is `TRUE`.

`titles`
: A logical value, optional, indicating whether question titles should be displayed during the exam. Default is `TRUE`.

`notes`
: A logical value, optional, indicating whether participants can take notes in OpenOlat during the exam. Default is `FALSE`.

`hidelms`
: A logical value, optional, indicating whether access to the OpenOlat learning management system should be hidden during the exam. Default is `TRUE`.

`hidefeedbacks`
: A logical value, optional, indicating whether feedback should be hidden. Default is `FALSE`.

`blockaftersuccess`
: A logical value, optional, indicating whether the exam should be blocked after successful completion. Default is `FALSE`.

`attempts`
: An integer value, optional, indicating how many attempts are allowed for the exam as a whole. Default is `1`.

`anonym` A logical value, optional, indicating whether anonymous users are allowed to take the exam. Default is `FALSE`.

`manualcorrect` A logical value, optional, indicating whether points and pass/fail status should be evaluated manually. Default is `FALSE`.

Value

An AssessmentTestOpenOlat object.

See Also

`test()`, `section()`, `assessmentTest()`, AssessmentTest, AssessmentSection

Examples

```
sc <- sc <- singleChoice(prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
s <- section(c(sc, es), title = "Section with nonrandomized tasks")
t <- assessmentTestOpenOlat(list(s), title = "Example of the Exam")
```

AssessmentTestOpenOlat-class

Class "AssessmentTestOpenOlat"

Description

Class `AssessmentTestOpenOlat` is responsible for creating XML exam files according to the QTI 2.1 standard for LMS OpenOlat.

Details

Test consists of one or more sections. Each section can have one or more questions/tasks and/or one or more sub sections.

Slots

`identifier` A character representing the unique identifier of the assessment test. By default, it is generated as 'id_test_dddd', where dddd represents random digits.

`title` A character representing the title of the test. By default, it takes the value of the identifier.

`points` Do not use directly; the maximum number of points for the exam/test. It is calculated automatically as a sum of points of included tasks.

`test_part_identifier` A character representing the identifier of the test part.

`navigation_mode` A character value, optional, determining the general paths that the candidate may have during the exam. Possible values:

- "linear" - candidate is not allowed to return to the previous questions.

- "nonlinear" - candidate is free to navigate. This is used by default.

`submission_mode` A character value, optional, determining when the candidate's responses are submitted for response processing. Possible values:

- "individual" - submit candidates' responses on an item-by-item basis. This is used by default.
- "simultaneous" - candidates' responses are submitted all together by the end of the test.

`section` A list containing one or more AssessmentSection objects.

`time_limit` A numeric value, optional, controlling the amount of time in minutes which a candidate is allowed for this part of the test.

`max_attempts` A numeric value, optional, enabling the maximum number of attempts that a candidate is allowed to pass.

`allow_comment` A boolean value, optional, enabling to allow candidates to leave comments in each question.

`rebuild_variables` A boolean value, optional, enabling to recalculate variables and reshuffle the order of choices for each item-attempt.

`fallback_titles` A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".

`academic_grading` A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test.

Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme.

The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.

`grade_label` A character value, optional, representing a short message to display with a grade in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`table_label` A character value, optional, representing a concise message to display as the column title of the grading table in the final feedback. For multilingual usage, it hat to be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system. Default is c(en="Grade", de="Note").

`stylesheet_path` A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When `academic_grading` is set, the default stylesheet `styles/rqti.css` is included automatically; a user-defined stylesheet is added in addition and may override default styles.

`metadata` An object of class QtiMetadata that holds metadata information about the test.

`cancel` A logical value, optional, indicating whether participants are allowed to cancel an exam after starting it. Default is `FALSE`.

`suspend` A logical value, optional, indicating whether participants are allowed to suspend an exam after starting it and continue later. Default is `FALSE`.

`scoreprogress` A logical value, optional, indicating whether the progress of the score achieved so far should be displayed during the exam. Default is `FALSE`.

`questionprogress` A logical value, optional, indicating whether the number of solved questions should be displayed during the exam. Default is `FALSE`.

`maxscoreitem` A logical value, optional, indicating whether the maximum score of an item should be displayed. Default is `TRUE`.

`menu` A logical value, optional, indicating whether the menu should be displayed during the exam. Default is `TRUE`.

`titles` A logical value, optional, indicating whether question titles should be displayed during the exam. Default is `TRUE`.

`notes` A logical value, optional, indicating whether participants can take notes in OpenOlat during the exam. Default is `FALSE`.

`hidelms` A logical value, optional, indicating whether access to the OpenOlat learning management system should be hidden during the exam. Default is `TRUE`.

`hidefeedbacks` A logical value, optional, indicating whether feedback should be hidden. Default is `FALSE`.

`blockaftersuccess` A logical value, optional, indicating whether the exam should be blocked after successful completion. Default is `FALSE`.

`attempts` An integer value, optional, indicating how many attempts are allowed for the exam as a whole. Default is `1`.

`anonym` A logical value, optional, indicating whether anonymous users are allowed to take the exam. Default is `FALSE`.

`manualcorrect` A logical value, optional, indicating whether points and pass/fail status should be evaluated manually. Default is `FALSE`.

See Also

AssessmentSection, AssessmentTest, `section()`.

Examples

```
# This example creates test 'exam' with one section 'exam_section' which
# consists of two questions/tasks: essay and single choice types
task1 <- new("Essay", prompt = "Test task", title = "Essay",
             identifier = "q1")
task2 <- new("SingleChoice", prompt = "Test task", title = "SingleChoice",
             choices = c("A", "B", "C"), identifier = "q2")
exam_section <- new("AssessmentSection", identifier = "sec_id",
                    title = "section", assessment_item = list(task1, task2))
exam <- new("AssessmentTestOpenOlat",
            identifier = "id_test_1234",
            title = "Example of Exam",
            navigation_mode = "linear",
            submission_mode = "individual",
            section = list(exam_section),
            time_limit = 90,
            max_attempts = 1,
            grade_label = "Preliminary grade")
```

authLMS *Authenticate with LMS*

Description

A generic function to handle authentication with a Learning Management System (LMS).

The method to handle authentication with LMS.

Usage

```
authLMS(object, ...)

## S4 method for signature 'LMS'
authLMS(object, ...)
```

Arguments

object	an instance of the S4 object Opal, LMS
...	Additional arguments to be passed to the method, if applicable.

buildAssessmentSection *Build tags for AssessmentSection in assessmentTest*

Description

Generic function for tags that contains assessementSection in assessnetTest

Usage

```
buildAssessmentSection(object, folder = NULL, verify = FALSE)

## S4 method for signature 'AssessmentItem'
buildAssessmentSection(object, folder)

## S4 method for signature 'AssessmentSection'
buildAssessmentSection(object, folder = NULL, verify = FALSE)

## S4 method for signature 'character'
buildAssessmentSection(object, folder = NULL, verify = FALSE)
```

Arguments

object	an instance of the S4 object (AssessmentSection and all types of AssessmentItem)
folder	string; a folder to store xml file
verify	boolean, optional; check validity of xml file, default FALSE

Choice-class *Class "Choice"*

Description

Abstract class Choice is not meant to be instantiated directly; instead, it serves as a base for derived classes SingleChoice and MultipleChoice.

Slots

choices A character vector defining a set of answer options in the question.

choice_identifiers A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.

shuffle A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.

orientation A character, determining whether to place answers in vertical or horizontal mode. Possible values:

- "vertical" - Default.
- "horizontal"

---

correctFeedback *Create object CorrectFeedback*

Description

Create object CorrectFeedback

Usage

```
correctFeedback(content = list(), title = character(0), show = TRUE)
```

Arguments

content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
title	A character value, optional, representing the title of the feedback window.
show	A boolean value, optional, determining whether to show (TRUE) or hide (FALSE) the feedback. Default is TRUE.

Value

An object of class CorrectFeedback

Examples

```
cfb <- correctFeedback(content = "Some comments", title = "Feedback")
```

CorrectFeedback-class *Class "CorrectFeedback"*

Description

Class CorrectFeedback is responsible for delivering feedback messages to the candidate in case of a correct answer on the entire exercise.

Slots

`outcome_identifier` A character representing the unique identifier of the outcome declaration variable that relates to feedback. Default is "FEEDBACKMODAL".

`show` A boolean value, optional, determining whether to show (TRUE) or hide (FALSE) the modal feedback. Default is TRUE.

`title` A character value, optional, representing the title of the modal feedback window.

`content` A list of character content to form the text of the modal feedback, which can include HTML tags.

`identifier` A character value representing the identifier of the modal feedback item. Default is "correct". cfb <- new("CorrectFeedback", title = "Right answer", content = list("Some demonstration"))

---

createAssessmentTest *Create an element assessmentTest of a qti-xml document for test*

Description

Generic function for creating assessmentTest element for XML document of specification the test following the QTI schema v2.1

Usage

```
createAssessmentTest(object, folder, verify = FALSE)

## S4 method for signature 'AssessmentTest'
createAssessmentTest(object, folder, verify = FALSE)

## S4 method for signature 'AssessmentTestOpal'
createAssessmentTest(object, folder, verify = FALSE)
```

Arguments

object	an instance of the S4 object AssessmentTest or AssessmentTestOpal
folder	string, optional; a folder to store xml file; working directory by default
verify	boolean, optional; to check validity of xml file, default FALSE

---

createConfigurationFile

Create a configuration file for QTI test

Description

Generic function for creating additional configuration file for the archive with test

Usage

```
createConfigurationFile(object, output)

## S4 method for signature 'AssessmentTest'
createConfigurationFile(object, output)

## S4 method for signature 'AssessmentTestOpenOlat'
createConfigurationFile(object, output)
```

Arguments

object	an instance of the S4 object AssessmentTest
output	string, a folder to store an xml configuration files

---

createItemBody

Create an element itemBody of a qti-xml document

Description

Generic function for creating itemBody element for XML document of specification the question following the QTI schema v2.1

Usage

```
createItemBody(object)

## S4 method for signature 'DirectedPair'
createItemBody(object)

## S4 method for signature 'Entry'
createItemBody(object)

## S4 method for signature 'Essay'
createItemBody(object)

## S4 method for signature 'MultipleChoice'
createItemBody(object)

## S4 method for signature 'MultipleChoiceTable'
createItemBody(object)

## S4 method for signature 'OneInColTable'
createItemBody(object)

## S4 method for signature 'OneInRowTable'
createItemBody(object)

## S4 method for signature 'Ordering'
createItemBody(object)

## S4 method for signature 'SingleChoice'
createItemBody(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair)

---

createMetadata *Create an element of metadata*

Description

Create an element of metadata

Usage

```
createMetadata(object)

## S4 method for signature 'QtiContributor'
```

```
createMetadata(object)

## S4 method for signature 'AssessmentItem'
createMetadata(object)

## S4 method for signature 'AssessmentTest'
createMetadata(object)
```

Arguments

object	an instance of the S4 object (QtiContributor, QtiMetadata

---

createOutcomeDeclaration

Create an element outcomeDeclaration of a qti-xml document

Description

Generic function for creating outcomeDeclaration element for XML document of specification the question following the QTI schema v2.1

Usage

```
createOutcomeDeclaration(object)

## S4 method for signature 'AssessmentItem'
createOutcomeDeclaration(object)

## S4 method for signature 'AssessmentTest'
createOutcomeDeclaration(object)

## S4 method for signature 'Entry'
createOutcomeDeclaration(object)

## S4 method for signature 'Gap'
createOutcomeDeclaration(object)

## S4 method for signature 'SingleChoice'
createOutcomeDeclaration(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

createQtiTask-methods *Create XML or zip file for question specification*

Description

Create XML or zip file for question specification

Usage

```
createQtiTask(object, dir = ".", verification = FALSE, zip = FALSE)

## S4 method for signature 'AssessmentItem'
createQtiTask(object, dir = ".", verification = FALSE, zip = FALSE)

## S4 method for signature 'character'
createQtiTask(object, dir = getwd(), verification = FALSE, zip = FALSE)
```

Arguments

object	An instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair).
dir	A character value, optional; a folder to store xml file; working directory is used by default.
verification	A boolean value, optional; to check validity of xml file. Default is FALSE.
zip	A boolean value, optional; the TRUE value allows to create a zip archive with the manifest and task files inside. Default is FALSE.

Value

A path to xml or zip file.

Examples

```
essay <- new("Essay", prompt = "Test task", title = "Essay")
## Not run:
# creates folder with XML (side effect)
createQtiTask(essay, "result")
# creates folder with zip (side effect)
createQtiTask(essay, "result", zip = TRUE)

## End(Not run)
```

createQtiTest-methods *Create zip-archive of the qti test specification*

Description

Create zip-archive of the qti test specification

Usage

```
createQtiTest(object, dir = NULL, verification = FALSE, zip_only =
  FALSE)

## S4 method for signature 'AssessmentItem'
createQtiTest(object, dir = ".", verification = FALSE, zip_only = FALSE)

## S4 method for signature 'AssessmentTest'
createQtiTest(object, dir = getwd(), verification = FALSE, zip_only = FALSE)

## S4 method for signature 'character'
createQtiTest(object, dir = getwd())
```

Arguments

object	An instance of the AssessmentTest, AssessmentTestOpal or AssessmentItem S4 object.
dir	A character value, optional; a folder to store xml file; working directory is used by default.
verification	A boolean value, optional; to check validity of xml files. Default is FALSE.
zip_only	A boolean value, optional; returns only zip file in case of TRUE or zip, xml and downloads files in case of FALSE value. Default is FALSE.

Value

A path to zip and xml files.

Examples

```
essay <- new("Essay", prompt = "Test task", title = "Essay",
             identifier = "q1")
sc <- new("SingleChoice", prompt = "Test task", title = "SingleChoice",
          choices = c("A", "B", "C"), identifier = "q2")
exam_section <- new("AssessmentSection", identifier = "sec_id",
                    title = "section", assessment_item = list(essay, sc))
exam <- new("AssessmentTestOpal", identifier = "id_test",
           title = "some title", section = list(exam_section))
## Not run:
# creates folder with zip (side effect)
```

```
createQtiTest(exam, "exam_folder", "TRUE")

## End(Not run)
```

---

createResponseDeclaration *Create an element responseDeclaration of a qti-xml document*

Description

Generic function for creating responseDeclaration element for XML document of specification the question following the QTI schema v2.1

Usage

```
createResponseDeclaration(object)

## S4 method for signature 'AssessmentItem'
createResponseDeclaration(object)

## S4 method for signature 'MatchTable'
createResponseDeclaration(object)

## S4 method for signature 'Entry'
createResponseDeclaration(object)

## S4 method for signature 'Essay'
createResponseDeclaration(object)

## S4 method for signature 'InlineChoice'
createResponseDeclaration(object)

## S4 method for signature 'MultipleChoice'
createResponseDeclaration(object)

## S4 method for signature 'MultipleChoiceTable'
createResponseDeclaration(object)

## S4 method for signature 'NumericGap'
createResponseDeclaration(object)

## S4 method for signature 'Ordering'
createResponseDeclaration(object)

## S4 method for signature 'SingleChoice'
createResponseDeclaration(object)
```

```
## S4 method for signature 'TextGap'
createResponseDeclaration(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

`createResponseProcessing` *Create an element responseProcessing of a qti-xml document*

Description

Generic function for creating responseProcessing element for XML document of specification the question following the QTI schema v2.1

Usage

```
createResponseProcessing(object)

## S4 method for signature 'AssessmentItem'
createResponseProcessing(object)

## S4 method for signature 'Entry'
createResponseProcessing(object)

## S4 method for signature 'Essay'
createResponseProcessing(object)

## S4 method for signature 'Gap'
createResponseProcessing(object)

## S4 method for signature 'NumericGap'
createResponseProcessing(object)

## S4 method for signature 'Ordering'
createResponseProcessing(object)

## S4 method for signature 'SingleChoice'
createResponseProcessing(object)

## S4 method for signature 'TextGapOpal'
createResponseProcessing(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

---

createText *Compose a set of html elements to display question in qti-xml document*

Description

Generic function for creating a set of html elements to display question for XML document of specification the question following the QTI schema v2.1

Usage

```
createText(object)

## S4 method for signature 'Gap'
createText(object)

## S4 method for signature 'InlineChoice'
createText(object)

## S4 method for signature 'character'
createText(object)
```

Arguments

object	an instance of the S4 object (Gap, InlineChoice, character)

---

createZip *Create an Zip archive of QTI test*

Description

Generic function for creating zip archive with set of XML documents of specification the test following the QTI schema v2.1

Usage

```
createZip(object, input, output, file_name, zip_only)

## S4 method for signature 'AssessmentTest'
createZip(object, input, output, file_name, zip_only)

## S4 method for signature 'AssessmentTestOpal'
createZip(object, input, output, file_name, zip_only)
```

Arguments

object	an instance of the S4 object AssessmentTest or AssessmentTestOpal
input	string, optional; a source folder with xml files
output	string, optional; a folder to store zip and xml files; working directory by default
file_name	string, optional; file name of zip archive
zip_only	boolean, optional; returns only zip file in case of TRUE or zip, xml and downloads files in case of FALSE value

---

create_assessment_item *Compose a root element AssessmentItem of xml task*

Description

create_assessment_item() creates html structure with AssessmentItem root element (shiny.tag) for xml qti task description according QTI 2.1

Usage

```
create_assessment_item(object)
```

Arguments

object	an instance of the S4 object

Value

A list() with a shiny.tag class

---

create_qti_task *Create XML file for question specification*

Description

Create XML file for question specification

Usage

```
create_qti_task(object, dir = NULL, verification = FALSE)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair).
dir	string, optional; a folder to store xml file; working directory by default
verification	boolean, optional; to check validity of xml file, default FALSE

Value

xml document.

`create_qti_test` *Create XML file for exam test specification*

Description

Create XML file for exam test specification

Usage

```
create_qti_test(object, path = ".", verification = FALSE, zip_only
  = FALSE)
```

Arguments

`object`	an instance of the AssessmentTest S4 object
`path`	string, optional; a path to folder to store zip file with possible file name; working directory by default
`verification`	boolean, optional; to check validity of xml file, default `FALSE`
`zip_only`	boolean, optional; returns only zip file in case of TRUE or zip, xml and downloads files in case of FALSE value

Value

xml document.

`create_question_object`
Create rqti S4 AssessmentItem Object from Rmd

Description

Generates an rqti S4 AssessmentItem object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair) from an Rmd file.

Usage

```
create_question_object(file)
```

Arguments

`file`	A string representing the path to an Rmd file.

Value

One of the rqti S4 AssessmentItem objects: SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, or DirectedPair.

Examples

```
create_question_object("file.Rmd")
```

`directedPair` *Create object DirectedPair*

Description

Create object DirectedPair

Usage

```
directedPair(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  rows,
  rows_identifiers,
  cols,
  cols_identifiers,
  answers_identifiers,
  answers_scores = NA_real_,
  shuffle = TRUE,
  shuffle_rows = TRUE,
  shuffle_cols = TRUE,
  feedback = list(),
  orientation = "vertical",
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

`identifier`	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
`title`	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
`content`	A character string or a list of character strings to form the text of the question, which may include HTML tags.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. If not provided, the default is calculated as 0.5 points per pair.

`rows` A character vector specifying answer options as the first elements in couples.

`rows_identifiers` A character vector, optional, specifies identifiers of the first elements in couples.

`cols` A character vector specifying answer options as the second elements in couples.

`cols_identifiers` A character vector, optional, specifies identifiers of the second elements in couples.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only for the first elements of the answer tuples. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only for the second elements of the answer tuples. Default is `TRUE`.

`feedback` A list containing feedback message-object ModalFeedback for candidates.

`orientation` A character, optional, determining whether to place answers in vertical or horizontal mode. Possible values:

- "vertical" - Default.
- "horizontal".

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific".

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class DirectedPair

Examples

```
dp_min <- directedPair(content = "<p>\"Directed pairs\" task</p>",
                       rows = c("alfa", "beta", "gamma"),
                       rows_identifiers = c("a", "b", "g"),
                       cols = c("A", "B", "G;"),
                       cols_identifiers = c("as", "bs", "gs"),
                       answers_identifiers = c("a as", "b bs", 'g gs'))

dp <- directedPair(identifier = "id_task_1234",
                   title = "Directed Pair Task",
                   content = "<p>\"Directed pairs\" task</p>",
                   prompt = "Plain text, can be used instead of the content",
                   rows = c("alfa", "beta", "gamma"),
                   rows_identifiers = c("a", "b", "g"),
                   cols = c("A", "B", "G"),
                   cols_identifiers = c("as", "bs", "gs"),
                   answers_identifiers = c("a as", "b bs", "g gs"),
                   answers_scores = c(1, 0.5, 0.1),
                   shuffle_rows = FALSE,
                   shuffle_cols = TRUE,
                   orientation = "horizontal")
```

DirectedPair-class *Class "DirectedPair"*

Description

Class `DirectedPair` is responsible for creating assessment tasks according to the QTI 2.1 standard, where a candidate has to make binary associations between answer options.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.

- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`rows` A character vector specifying answer options as row names in the table or the first elements in couples in DirectedPair.

`rows_identifiers` A character vector, optional, specifying identifiers for answer options defined in rows of the table or identifiers of the first elements in couples in DirectedPair.

`cols` A character vector specifying answer options as column headers in the table or the second elements in couples in DirectedPair.

`cols_identifiers` A character vector, optional, specifying identifiers for answer options defined in columns of the table or identifiers of the second elements in couples in DirectedPair.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only in rows. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only in columns. Default is `TRUE`.

`orientation` A character, optional, determining whether to place answers in vertical or horizontal mode. Possible values:

- "vertical" - Default.
- "horizontal"

Examples

```
dp <- new("DirectedPair",
          identifier = "id_task_1234",
          title = "Directed pair",
```

```
content = list("<p>\"Directed pairs\" task</p>"),
points = 5,
rows = c("row1", "row2", "row3"),
rows_identifiers = c("a", "b", "c"),
cols = c("alfa", "beta", "gamma"),
cols_identifiers = c("k", "l", "m"),
answers_identifiers = c("a k", "b l", 'c m'),
shuffle = TRUE,
orientation = "vertical")
```

---

dropdown *Create YAML string for InlineChoice object (dropdown list)*

Description

Create YAML string for InlineChoice object (dropdown list)

Usage

```
dropdown(
  choices,
  solution_index = 1,
  points = 1,
  shuffle = TRUE,
  response_identifier = NULL
)
```

Arguments

choices	A numeric or character vector; contains values of possible answers. If you use a named vector, the names will be used as identifiers.
solution_index	An integer value, optional; the number of right answer in the choices vector. Default is 1.
points	A numeric value, optional; the number of points for this gap. Default is 1.
shuffle	A boolean, optional; is responsible to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
response_identifier	A character string, optional; an identifier for the answer.

Value

A character string mapped as yaml.

See Also

gap_text(), gap_numeric(), mdlist()

Examples

```
dropdown(c("Option A", "Option B"), response_identifier = "task_dd_list")
```

`entry` *Create object Entry*

Description

Create object Entry

Usage

```
entry(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  feedback = list(),
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

`identifier`	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
`title`	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
`content`	A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as `TextGap`, `TextGapOpal`, `NumericGap`, or `InlineChoice`.
`prompt`	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
`points`	A numeric value, it is calculated as the sum of the gap points by default.
`feedback`	A list containing feedback message-object ModalFeedback for candidates.
`calculator`	A character, optional, determining whether to show a calculator to the candidate. Possible values: • "simple" • "scientific".
`files`	A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class Entry

See Also

[textGap()][numericGap()][textGapOpal()]

Examples

```
gap_min <- entry(content = list("Question and Test Interoperability",
                                textGap("QTI")))

gap <- entry(identifier = "id_task_1234",
           title = "Essay Task",
                   content = list("Question and Test Interoperability:",
                                  textGap("QTI")),
                   prompt = "Plain text, can be used instead of content",
                   points = 2,
                   feedback = list(new("ModalFeedback",
                                   content = list("Model answer"))),
                   calculator = "scientific-calculator",
                   files = "text_book.pdf")
```

Entry-class *Class "Entry"*

Description

Class Entry is responsible for creating assessment tasks according to the QTI 2.1 standard. These tasks include one or more instances of text input fields (with numeric or text answers) or dropdown lists.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.

- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

See Also

NumericGap, TextGap, TextGapOpal, InlineChoice

Examples

```
entry_gaps <- new("Entry", content = list("<p>In mathematics, the common
logarithm is the logarithm with base", new("NumericGap",
                                           response_identifier = "numeric_1",
                                           solution = 10,
                                           placeholder = "it is a number"),
". It is also known as the decimal", new("TextGap",
                                         response_identifier = "text_1",
                                         solution = "logarithm",
                                         placeholder = "it is a text"),
 ".</p>"),
                  title = "entry with number and text in answers",
                  identifier = "entry_example")
entry_dropdown <- new("Entry", content = list("<p>In mathematics, the common
logarithm is the logarithm with base", new("InlineChoice",
                                           response_identifier = "numeric_1",
                                           choices = c("10", "7", "11")),
". It is also known as the decimal", new("InlineChoice",
                                         response_identifier = "text_1",
                                         choices = c("logarithm", "limit")),
 ".</p>"),
                  title = "entry with dropdown lists for answers",
                  identifier = "entry_example")
```

essay *Create object Essay*

Description

Create object Essay

Usage

```
essay(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  feedback = list(),
  expected_length = length_expected(feedback),
  expected_lines = lines_expected(feedback),
  words_max = max_words(feedback),
  words_min = NA_integer_,
  data_allow_paste = TRUE,
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

identifier	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
title	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
prompt	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
points	A numeric value, optional, representing the number of points for the entire task. Default is 1.
feedback	A list containing feedback message-object ModalFeedback for candidates.
expected_length	A numeric, optional. Responsible for setting the size of the text input field in the content delivery engine. By default it will be calculated according to model answer in the slot content of ModalFeedback.
expected_lines	A numeric, optional. Responsible for setting the number of rows of the text input field in the content delivery engine. By default it will be calculated according to model answer in the slot content of ModalFeedback.

`words_max` A numeric, optional. Responsible for setting the maximum number of words that a candidate can write in the text input field. By default it will be calculated according to model answer in the slot `content` of `ModalFeedback`.

`words_min` A numeric, optional. Responsible for setting the minimum number of words that a candidate should write in the text input field.

`data_allow_paste` A boolean, optional. Determines whether it is possible for a candidate to copy text into the text input field. Default is TRUE. Only works on OPAL and OpenOlat.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific".

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class Essay

Examples

```
es_min <- essay(content = list("<h2>Open question</h2>", "Write your answer here"))

es <- essay(identifier = "id_task_1234",
            title = "Essay Task",
            content = "<h2>Open question</h2> Write your answer here",
            prompt = "Plain text, can be used instead of content",
            points = 2,
            expected_length = 100,
            expected_lines = 5,
            words_max = 100,
            words_min = 1,
            data_allow_paste = TRUE,
            feedback = list(new("ModalFeedback",
                            content = list("Model answer"))),
            calculator = "scientific-calculator",
            files = "text_book.pdf")
```

Essay-class *Class "Essay"*

Description

Class `Essay` is responsible for creating essay type of assessment task according to QTI 2.1.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`expected_length` A numeric, optional. Responsible for setting the size of the text input field in the content delivery engine.

`expected_lines` A numeric, optional. Responsible for setting the number of rows of the text input field in the content delivery engine.

`words_max` A numeric, optional. Responsible for setting the maximum number of words that a candidate can write in the text input field.

`words_min` A numeric, optional. Responsible for setting the minimum number of words that a candidate should write in the text input field.

`data_allow_paste` A logical, optional. Determines whether it is possible for a candidate to copy text into the text input field. Default is `TRUE`. Only works on OPAL and OpenOlat.

Note

If 'ModalFeedback' is given, default values for slots related to the text input field are calculated automatically.

Examples

```
es <- new("Essay",
          identifier = "id_task_1234",
          title = "Essay Task",
          content = list("<p>Develop some idea and write it down in
                                  the text field</p>"),
          prompt = "Write your answer in text field",
          points = 1,
          feedback = list(),
          calculator = "scientific-calculator",
          files = "text_book.pdf",
          expected_length = 100,
          expected_lines = 5,
          words_max = 200,
          words_min = 10,
          data_allow_paste = FALSE)
```

`exams_task` *Convert an exams Rmd task to a QTI 2.1 XML file*

Description

Helper function for using tasks written in the `exams` Rmd format inside `rqti` sections. The function converts the given Rmd file with `exams::exams2qti21()` and returns the path to the generated XML file.

Usage

```
exams_task(path, title = NULL)
```

Arguments

`path`	Character string. Path to an Rmd file written in the `exams` format.
`title`	Character string or `NULL`. Optional title passed to `exams::exams2qti21()` via the `ititle` argument.

Details

This function is mainly useful when `exams` tasks need to be combined with native `rqti` tasks in one section, for example: `section(exams_task(path_to_file, title = "Exams task"))`.

Value

Character string. Path to the QTI 2.1 XML file generated by `exams::exams2qti21()` in a temporary directory.

Examples

```
path <- system.file("exams", "example.Rmd", package = "rqti")
section(exams_task(path, title = "Exams task"))
```

`extract_results` *Create data frame with test results*

Description

The function `extract_results()` takes Opal zip archive "Export results" or xml file and creates two kinds of data frames (according to parameter 'level'), see the 'Details' section.

Usage

```
extract_results(file, level = "task", hide_filename = TRUE)
```

Arguments

`file`	A string with a path of the xml test result file.
`level`	A string with two possible values: task and item.
`hide_filename`	A boolean value, TRUE to hide original file names by default.

Value

A dataframe with attribues of the candidates outcomes and result variables.

Note

1.With option level = "task" data frame consists of columns:

- 'file' - name of the xml file with test results (to identify candidate)
- 'date' - date and time of test
- 'id_question' - question item identifier
- 'duration' - time in sec. what candidate spent on this item
- 'score_candidate' - points that were given to candidate after evaluation
- 'score_max' - max possible score for this question
- 'is_answer_given' - TRUE if candidate gave the answer on question, otherwise FALSE
- 'title' - the values of attribute 'title' of assessment items
- 'scorer_comment' - scorer's comment for manually scored items (if available).

2.With option level = "item" data frame consists of columns:

- 'file' - name of the xml file with test results (to identify candidate)
- 'date' - date and time of test

- 'id_question' - question item identifier
- 'base_type' - type of answer (identifier, string or float)
- 'cardinalities' - defines whether this question is single, multiple or ordered -value
- 'qti_type' - specifies the type of the task
- 'id_answer' - identifier of each response variable
- 'expected_response' - values that considered as right responses for question
- 'candidate_response' - values that were given by candidate
- 'score_candidate' - - points that were given to candidate after evaluation
- 'score_max' - max possible score for this question item
- 'is_response_correct' - TRUE if candidate gave the right response, otherwise FALSE
- 'title' - the values of attribute 'title' of assessment items

Examples

```
file <- system.file("test_results.zip", package = "rqti")
df <- extract_results(file, level = "item")
```

`Gap-class` *Class "Gap"*

Description

Abstract class `Gap` is not meant to be instantiated directly; instead, it serves as a base for derived classes such as NumericGap, TextGap, TextGapOpal and InlineChoice.

Slots

`response_identifier` A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.

`points` A numeric value, optional, representing the number of points for this gap. Default is `1`.

`placeholder` A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine.

`expected_length` A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine.

See Also

NumericGap, TextGap, TextGapOpal and InlineChoice.

gap_numeric *Create YAML string for NumericGap object*

Description

Create YAML string for NumericGap object

Usage

```
gap_numeric(
  solution,
  tolerance = 0,
  tolerance_type = "absolute",
  points = 1,
  response_identifier = NULL,
  include_lower_bound = TRUE,
  include_upper_bound = TRUE,
  expected_length = size_gap(solution),
  placeholder = NULL
)
```

Arguments

solution	A numeric value; contains right answer for this numeric entry.
tolerance	A numeric value, optional; specifies the value for up and low boundaries of tolerance rate for candidate answer. Default is 0.
tolerance_type	A character string, optional; specifies tolerance mode; possible values:"exact", "absolute" (by default), "relative".
points	A numeric value, optional; the number of points for this gap. Default is 1.
response_identifier	A character string, optional; an identifier for the answer.
include_lower_bound	A boolean, optional; specifies whether or not the lower bound is included in tolerance rate.
include_upper_bound	A boolean, optional; specifies whether or not the upper bound is included in tolerance rate.
expected_length	An integer value, optional; is responsible to set a size of text input field in content delivery engine.
placeholder	A character string, optional; is responsible to place some helpful text in text input field in content delivery engine.

Value

A character string mapped as yaml.

See Also

gap_text(), dropdown(), mdlist()

Examples

```
gap_numeric(5.0, tolerance = 10, tolerance_type = "relative")
```

---

gap_text *Create YAML string for TextGap object*

Description

Create YAML string for TextGap object

Usage

```
gap_text(
  solution,
  tolerance = NULL,
  case_sensitive = FALSE,
  points = 1,
  response_identifier = NULL,
  expected_length = size_gap(solution),
  placeholder = NULL
)
```

Arguments

solution	A character vector containing values considered as correct answers.
tolerance	An integer value, optional; defines the number of characters to tolerate spelling mistakes in evaluating candidate answers.
case_sensitive	A boolean, optional; determines whether the evaluation of the correct answer is case sensitive. Default is FALSE.
points	A numeric value, optional; the number of points for this gap. Default is 1.
response_identifier	A character string (optional) representing an identifier for the answer.
expected_length	An integer value, optional; sets the size of the text input field in the content delivery engine.
placeholder	A character string, optional; places helpful text in the text input field in the content delivery engine.

Value

A character string mapped as yaml.

See Also

gap_numeric(), dropdown(), mdlist()

Examples

```
gap_text(c("Solution", "Solutions"), tolerance = 2)
```

---

german_grading *German grading scale*

Description

A helper function that returns a named numeric vector representing a common German grading scheme. The names correspond to grades and the values define the minimum proportion of points required to achieve the respective grade.

Usage

```
german_grading()
```

Value

A named numeric vector with grades as names and minimum score proportions as values.

Examples

```
german_grading()
```

---

getAssessmentItems *Get list of AssessmentItems for AssessmentSection*

Description

Generic function for

Usage

```
getAssessmentItems(object)

## S4 method for signature 'AssessmentItem'
getAssessmentItems(object)

## S4 method for signature 'AssessmentSection'
getAssessmentItems(object)

## S4 method for signature 'character'
getAssessmentItems(object)
```

Arguments

object	an instance of the S4 object (AssessmentSection, AssessmentItem)

---

getCalculator-methods *Get value of the slot 'calculator'*

Description

Get value of the slot 'calculator'

Usage

```
getCalculator(object)

## S4 method for signature 'AssessmentItem'
getCalculator(object)

## S4 method for signature 'AssessmentSection'
getCalculator(object)

## S4 method for signature 'character'
getCalculator(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

`getContributors-methods` *Get list of contributors values*

Description

Get list of contributors values

Usage

```
getContributors(object)

## S4 method for signature 'AssessmentItem'
getContributors(object)

## S4 method for signature 'AssessmentSection'
getContributors(object)

## S4 method for signature 'character'
getContributors(object)
```

Arguments

`object`	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

`getCourseElements` *Get elements of the course by courseId from LMS*

Description

This method gets elements of the user's course by its courseId on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getCourseElements(object, course_id)

## S4 method for signature 'missing'
getCourseElements(object, course_id)

## S4 method for signature 'Opal'
getCourseElements(object, course_id)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
course_id	A length one character vector with course id.

Value

A dataframe with the elements of the course.

A dataframe with the data of the elements of the course (fields: nodeId, shortTitle, shortName, longTitle) on LMS Opal.

---

getCourseGroups *Get groups from a course*

Description

This method retrieves groups from a course on the Learning Management System (LMS) by its course id. It returns a data frame with all accessible attributes of the groups. If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

This method retrieves groups from a course on LMS Opal by its course id. The groups are returned as a data frame with all accessible attributes, such as group id, name, description, participant limits, and group settings. If no Opal connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getCourseGroups(object, course_id)

## S4 method for signature 'missing'
getCourseGroups(object, course_id)

## S4 method for signature 'Opal'
getCourseGroups(object, course_id)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
course_id	A character vector of length one specifying the course resource ID. Note that this is not the course ID shown in the URL, but a longer identifier available via the "Show more information" option within the course.

Value

A data frame with all accessible attributes of the course groups.

A data frame with course groups and their attributes.

Examples

```
groups <- getCourseGroups("89068111333293")


groups <- getCourseGroups("89068111333293")
```

`getCourseResult` *Get zip with course results by resource id and node id*

Description

This method retrieves zip with course results by its resource id and node id on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getCourseResult(object, resource_id, node_id, path_outcome = ".", ...)

## S4 method for signature 'missing'
getCourseResult(object, resource_id, node_id, path_outcome = ".", ...)

## S4 method for signature 'Opal'
getCourseResult(
  object,
  resource_id,
  node_id,
  path_outcome = ".",
  rename = TRUE
)
```

Arguments

`object`	An S4 object of class Opal that represents a connection to the LMS.
`resource_id`	A length one character vector with resource id.
`node_id`	A length one character vector with node id (test).
`path_outcome`	A length one character vector with path, where the zip should be stored. Default is working directory.
`...`	Additional arguments to be passed to the method, if applicable.
`rename`	A boolean value; optional; Set `TRUE` value to take the short name of the course element for naming zip (results_shortName.zip). `FALSE` combines in zip name course id and node id. Default is `TRUE`.

Value

It downloads a zip and return a character string with path.

Examples

```
zip_file <- getCourseResult("89068111333293", "1617337826161777006")
```

`getFiles-methods` *Get file paths for attachment of test*

Description

Get file paths for attachment of test

Usage

```
getFiles(object)

## S4 method for signature 'AssessmentItem'
getFiles(object)

## S4 method for signature 'AssessmentSection'
getFiles(object)

## S4 method for signature 'character'
getFiles(object)
```

Arguments

`object`	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

`getGroupUsers` *Get users from a group*

Description

This method retrieves users from a group on the Learning Management System (LMS) by its group id. It returns a data frame with all accessible attributes of the group members. If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

This method retrieves users from a group on LMS Opal by its group id. It returns a data frame with user attributes obtained from the LMS, including user id, login, first name, last name, email, and additional properties (e.g., status). If no Opal connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getGroupUsers(object, group_id)

## S4 method for signature 'missing'
getGroupUsers(object, group_id)

## S4 method for signature 'Opal'
getGroupUsers(object, group_id)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
group_id	A character vector specifying the group id.

Value

A data frame with all accessible attributes of the group members.

A data frame with group users and their attributes. Each row represents a user and may include the following columns:

key Unique user id.

login User login name.

firstName User first name.

lastName User last name.

email User email address.

status User status (e.g., STATUS_ACTIVE).

Examples

```
groups <- getGroupUsers("89068111333293")


users <- getGroupUsers("89068111333293")
```

---

getIdentifier-methods *Get identifier*

Description

Get identifier

Usage

```
getIdentifier(object)

## S4 method for signature 'AssessmentItem'
getIdentifier(object)

## S4 method for signature 'AssessmentSection'
getIdentifier(object)

## S4 method for signature 'Gap'
getIdentifier(object)

## S4 method for signature 'character'
getIdentifier(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

---

getLMSResources *Get records of all current user's resources on LMS*

Description

This method retrieves data about all resources associated with the current user on the Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getLMSResources(object, ...)

## S4 method for signature 'missing'
getLMSResources(object)

## S4 method for signature 'Opal'
getLMSResources(object)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
...	Additional arguments to be passed to the method, if applicable.

Value

A dataframe with attributes of user's resources.

`getLMSResourcesByName` *Get select records about user resources by name.*

Description

This method retrieves data about a user's resource by its name on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getLMSResourcesByName(object, ...)

## S4 method for signature 'missing'
getLMSResourcesByName(object, display_name, rtype = NULL)

## S4 method for signature 'Opal'
getLMSResourcesByName(object, display_name, rtype = NULL)
```

Arguments

`object`	An S4 object of class Opal that represents a connection to the LMS.
`...`	Additional arguments to be passed to the method, if applicable.
`display_name`	A string value withe the name of resource.
`rtype`	A string value with the type of resource. Possible values: "FileResource.TEST", "FileResource.QUESTION", or "FileResource.SURVEY".

Value

A dataframe with attributes of user's resources.

Examples

```
df <- getLMSResourcesByName("task_name")
```

getLMSResourceURL *Create an URL using the resource's display name on LMS*

Description

This method creates an URL for user's resource by its name on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
getLMSResourceURL(object, display_name)

## S4 method for signature 'missing'
getLMSResourceURL(object, display_name)

## S4 method for signature 'Opal'
getLMSResourceURL(object, display_name)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
display_name	A length one character vector to entitle file in OPAL; it takes file name without extension by default; optional.

Value

A string value of URL.

getObject-methods *Get object*

Description

Get object

Usage

```
getObject(object)

## S4 method for signature 'AssessmentItem'
getObject(object)

## S4 method for signature 'AssessmentSection'
getObject(object)
```

```
## S4 method for signature 'character'
getObject(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

---

getPoints-methods *Get points from AssessmentItem object*

Description

Get points from AssessmentItem object

Usage

```
getPoints(object)

## S4 method for signature 'AssessmentItem'
getPoints(object)

## S4 method for signature 'AssessmentSection'
getPoints(object)

## S4 method for signature 'MultipleChoice'
getPoints(object)

## S4 method for signature 'character'
getPoints(object)
```

Arguments

object	an instance of the S4 object (SingleChoice, MultipleChoice, Essay, Entry, Ordering, OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair, TextGap, NumericGap, InlineChoice)

getResponse *Get and process a piece of question content*

Description

Generic function to get and process a different types of question content (text with instances of gaps or dropdown lists) for XML document of specification the question following the QTI schema v2.1

Usage

```
getResponse(object)

## S4 method for signature 'InlineChoice'
getResponse(object)

## S4 method for signature 'NumericGap'
getResponse(object)

## S4 method for signature 'TextGap'
getResponse(object)

## S4 method for signature 'character'
getResponse(object)
```

Arguments

object	an instance of the S4 object (NumericGap, TextGap, InlineChoice, character)

inlineChoice *Create object InlineChoice*

Description

Create object InlineChoice

Usage

```
inlineChoice(
  choices,
  solution_index = 1,
  response_identifier = generate_id(type = "gap"),
  choices_identifiers = paste0("Choice", LETTERS[seq(choices)]),
  points = 1,
  shuffle = TRUE,
  placeholder = "",
  expected_length = size_gap(choices)
)
```

Arguments

choices	A character vector containing the answers shown in the dropdown list.
solution_index	A numeric value, optional, representing the index of the correct answer in the options vector. Default is 1.
response_identifier	A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.
choices_identifiers	A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "OptionD", where D is a letter representing the alphabetical order of the answer in the list.
points	A numeric value, optional, representing the number of points for this gap. Default is 1
shuffle	A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
placeholder	A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine. Default is "".
expected_length	A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine. Default value is adjusted by the first choice size.

Value

An object of class InlineChoice

See Also

[entry()][numericGap()][textGap()][textGapOpal()]

Examples

```
dd_min <- inlineChoice(c("answer1", "answer2", "answer3"))

dd <- inlineChoice(choices = c("answer1", "answer2", "answer3"),
                   solution_index = 2,
                   response_identifier  = "id_gap_1234",
                   choices_identifiers = c("a", "b", "c"),
                   points = 2,
                   shuffle = FALSE,
                   placeholder = "answers",
                   expected_length = 10)
```

`InlineChoice-class` *Class "InlineChoice"*

Description

Class `InlineChoice` is responsible for creating instances of dropdown lists as answer options in Entry type assessment tasks according to the QTI 2.1 standard.

Slots

`response_identifier` A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.

`points` A numeric value, optional, representing the number of points for this gap. Default is 1.

`placeholder` A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine.

`expected_length` A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine.

`choices` A character vector containing the answers shown in the dropdown list.

`solution_index` A numeric value, optional, representing the index of the correct answer in the options vector. Default is 1.

`choices_identifiers` A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "OptionD", where D is a letter representing the alphabetical order of the answer in the list.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

See Also

Entry, NumericGap, TextGap, TextGapOpal

Examples

```
dd <- new("InlineChoice",
          response_identifier = "id_gap_1234",
          points = 1,
          choices =  c("answer1", "answer2", "answer3"),
          solution_index = 1,
          choices_identifiers = c("OptionA", "OptionB", "OptionC"),
          shuffle = TRUE)
```

isUserLoggedIn *Check if User is Logged in LMS*

Description

This method checks whether a user is logged into an LMS (Learning Management System) by sending a request to the LMS server and evaluating the response.

This method checks whether a user is logged into an LMS Opal by sending a request to the LMS server and evaluating the response.

Usage

```
isUserLoggedIn(object)

## S4 method for signature 'Opal'
isUserLoggedIn(object)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.

Value

A logical value (TRUE if the user is logged in, FALSE otherwise).

A logical value (TRUE if the user is logged in, FALSE otherwise).

LMS-class *LMS Class*

Description

The LMS class is an abstract representation of a Learning Management System (LMS). It provides a foundation for defining LMS-specific implementations.

Slots

name A character string representing the name or identifier of the LMS.

api_user A character string containing the username for authentication.

endpoint A character string specifying the LMS API endpoint. By default, this value is retrieved from the environment variable RQTI_API_ENDPOINT. To set this variable globally, use: Sys.setenv(RQTI_API_ENDPOINT = 'your_endpoint'), or add it to your .Renviron file for persistence across sessions.

`MatchTaable-classs` *Class "MatchTable"*

Description

Abstract class `MatchTable` is not meant to be instantiated directly; instead, it serves as a base for derived classes such as OneInRowTable, OneInColTable, MultipleChoiceTable, and DirectedPair.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`rows` A character vector specifying answer options as row names in the table or the first elements in couples in DirectedPair.

`rows_identifiers` A character vector, optional, specifying identifiers for answer options defined in rows of the table or identifiers of the first elements in couples in DirectedPair.

`cols` A character vector specifying answer options as column headers in the table or the second elements in couples in DirectedPair.

`cols_identifiers` A character vector, optional, specifying identifiers for answer options defined in columns of the table or identifiers of the second elements in couples in DirectedPair.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only in rows. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only in columns. Default is `TRUE`.

See Also

OneInRowTable, OneInColTable, MultipleChoiceTable, DirectedPair

---

`mdlist` *Create a markdown list for answer options*

Description

Create a markdown list for answer options

Usage

```
mdlist(vect, solutions = NULL, gaps = NULL)
```

Arguments

`vect`	A string or numeric vector of answer options for single/multiple choice task.
`solutions`	An integer value, optional; indexes of right answer options in `vect`.
`gaps`	numeric or string vector, optional; provides primitive gap description if there is a need to build a list of gaps.

Value

A markdown list.

See Also

`gap_text()`, `gap_numeric()`, `dropdown()`

Examples

```
#list for multiple choice task
mdlist(c("A", "B", "C"), c(2, 3))
# it returns:
#- A
#- *B*
#- *C*
#list of gaps
mdlist(c("A", "B", "C"), c(2, 3), c(1, 2, 3))
# it returns:
#- A <gap>1</gap>
#- *B* <gap>2</gap>
#- *C* <gap>3</gap>
```

`modalFeedback` *Create object ModalFeedback*

Description

Create object ModalFeedback

Usage

```
modalFeedback(content = list(), title = character(0), show = TRUE)
```

Arguments

`content`	A character string or a list of character strings to form the text of the question, which may include HTML tags.
`title`	A character value, optional, representing the title of the modal feedback window.
`show`	A boolean value, optional, determining whether to show (`TRUE`) or hide (`FALSE`) the modal feedback. Default is `TRUE`.

Value

An object of class ModalFeedback

Examples

```
fb <- modalFeedback(content = "Model answer", title = "Feedback")
```

ModalFeedback-class *Class "ModalFeedback"*

Description

Class ModalFeedback is responsible for delivering feedback messages to the candidate, regardless of whether the answer was correct or incorrect.

Slots

outcome_identifier A character representing the unique identifier of the outcome declaration variable that relates to feedback. Default is "FEEDBACKMODAL".

show A boolean value, optional, determining whether to show (TRUE) or hide (FALSE) the modal feedback. Default is TRUE.

title A character value, optional, representing the title of the modal feedback window.

content A list of character content to form the text of the modal feedback, which can include HTML tags.

identifier A character value representing the identifier of the modal feedback item. Default is "modal_feedback".

Examples

```
fb <- new("ModalFeedback",
          title = "Possible solution",
          content = list("<b>Some explanation</b>"))
```

multipleChoice *Create object MultipleChoice*

Description

Create object MultipleChoice

Usage

```
multipleChoice(
  identifier = generate_id(),
  title = identifier,
  choices,
  choice_identifiers = paste0("Choice", LETTERS[seq(choices)]),
  content = list(),
  prompt = "",
  points = 1,
  feedback = list(),
```

```
  orientation = "vertical",
  shuffle = TRUE,
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

identifier	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
title	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
choices	A character vector defining a set of answer options in the question.
choice_identifiers	A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.
content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
prompt	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
points	A numeric vector, required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the choices. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.
feedback	A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.
orientation	A character, determining whether to place answers in vertical or horizontal mode. Possible values: • "vertical" - Default, • "horizontal".
shuffle	A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
calculator	A character, optional, determining whether to show a calculator to the candidate. Possible values: • "simple" • "scientific".
files	A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class MultipleChoice

Examples

```
mc_min <- multipleChoice(choices = c("option1", "option2", "option3"),
                         points = c(0, 0.5, 0.5))

mc <- multipleChoice(identifier = "id_task_1234",
                     title = "Multiple Choice Task",
                     content = "<p>Pick up the right options</p>",
                     prompt = "Plain text, can be used instead of content",
                     points = c(0, 0.5, 0.5),
                     feedback = list(new("WrongFeedback",
                                         content = list("Wrong answer"))),
                     calculator = "scientific-calculator",
                     files = "text_book.pdf",
                     choices = c("option 1", "option 2", "option 3"),
                     choice_identifiers = c("ChoiceA", "ChoiceB", "ChoiceC"),
                     shuffle = TRUE,
                     orientation = "vertical")
```

MultipleChoice-class *Class "MultipleChoice"*

Description

Class MultipleChoice is responsible for creating multiple choice assessment task according to QTI 2.1.

Slots

identifier A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

title A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

content A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

prompt An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

points A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.

- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`choices` A character vector defining a set of answer options in the question.

`choice_identifiers` A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.

`shuffle` A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`orientation` A character, determining whether to place answers in vertical or horizontal mode. Possible values:

- "vertical" - Default.
- "horizontal"

Examples

```
mc <- new("MultipleChoice",
          identifier = "id_task_1234",
          title = "Multiple Choice Task",
          content = list("<p>Pick up the right options</p>"),
          prompt = "Plain text, can be used instead of content",
          points = c(1, -1, 1, -1),
          feedback = list(new("WrongFeedback", content = list("Wrong answer"))),
          calculator = "scientific-calculator",
          files = "text_book.pdf",
          choices = c("option 1", "option 2", "option 3", "option 4"),
          choice_identifiers = c("ChoiceA", "ChoiceB", "ChoiceC", "ChoiceD"),
          shuffle = TRUE,
          orientation = "vertical")
```

`multipleChoiceTable` *Create object MultipleChoiceTable*

Description

Create object MultipleChoiceTable

Usage

```
multipleChoiceTable(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  rows,
  rows_identifiers,
  cols,
  cols_identifiers,
  answers_identifiers,
  answers_scores = NA_real_,
  shuffle = TRUE,
  shuffle_rows = TRUE,
  shuffle_cols = TRUE,
  feedback = list(),
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

`identifier`	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
`title`	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
`content`	A character string or a list of character strings to form the text of the question, which may include HTML tags.
`prompt`	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
`points`	A numeric value, optional, representing the number of points for the entire task. It can also be calculated as the sum of points for individual answers, when provided. Default is 1.
`rows`	A character vector specifying answer options defined in rows of the table.
`rows_identifiers`	A character vector, optional, specifies identifiers of the rows of the table

`cols` A character vector specifying answer options defined in columns of the table.

`cols_identifiers`
A character vector, optional, specifies identifiers of the columns of the table.

`answers_identifiers`
A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only for the first elements of the answer tuples. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only for the second elements of the answer tuples. Default is `TRUE`.

`feedback` A list containing feedback message-object ModalFeedback for candidates.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific".

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class MultipleChoiceTable

Examples

```
mt_min <- multipleChoiceTable(content = "<p>\"Multiple choice table\" task</p>",
                      rows = c("alfa", "beta", "gamma", "alpha"),
                      rows_identifiers = c("a", "b", "g", "aa"),
                      cols = c("A", "B", "G", "a"),
                      cols_identifiers = c("as", "bs", "gs", "aas"),
                 answers_identifiers = c("a as", "b bs", "g gs", "aa as", "a aas", "aa aas"))

mt <- multipleChoiceTable(identifier = "id_task_1234",
                    title = "Table with many possible answers in rows and cols",
                    content = "<p>\"Multiple choice table\" task</p>",
                    prompt = "Plain text, can be used instead of the content",
                    rows = c("alfa", "beta", "gamma", "alpha"),
                    rows_identifiers = c("a", "b", "g", "aa"),
                    cols = c("A", "B", "G", "a"),
                    cols_identifiers = c("as", "bs", "gs", "aas"),
                answers_identifiers = c("a as", "b bs", "g gs", "aa as", "a aas", "aa aas"),
                    answers_scores = c(1, 0.5, 0.1, 1, 0.5, 1),
                    shuffle_rows = FALSE,
                    shuffle_cols = TRUE)
```

MultipleChoiceTable-class
Class "MultipleChoiceTable"

Description

Class `MultipleChoiceTable` is responsible for creating assessment tasks according to the QTI 2.1 standard with a table of answer options, where many correct answers in each row and column are possible.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`rows` A character vector specifying answer options as row names in the table or the first elements in couples in DirectedPair.

`rows_identifiers` A character vector, optional, specifying identifiers for answer options defined in rows of the table or identifiers of the first elements in couples in DirectedPair.

`cols` A character vector specifying answer options as column headers in the table or the second elements in couples in DirectedPair.

`cols_identifiers` A character vector, optional, specifying identifiers for answer options defined in columns of the table or identifiers of the second elements in couples in DirectedPair.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only in rows. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only in columns. Default is `TRUE`.

`mapping` Do not use directly; values are initialized automatically. This slot contains a named numeric vector of points, where names correspond to all possible combinations of row and column identifiers.

Examples

```
mt <- new("MultipleChoiceTable",
          identifier = "id_task_1234",
          title = "Multiple choice table",
          content = list("<p>Match table task</p>",
                         "<i>table description</i>"),
          points = 5,
          rows = c("row1", "row2", "row3"),
          rows_identifiers = c("a", "b", "c"),
          cols = c("alfa", "beta", "gamma"),
          cols_identifiers = c("a", "b", "c"),
          answers_identifiers = c("a a", "b b", "b c"),
          shuffle = TRUE)
```

`numericGap` *Create object NumericGap*

Description

Create object NumericGap

Usage

```
numericGap(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  tolerance = 0,
  tolerance_type = "absolute",
  include_lower_bound = TRUE,
  include_upper_bound = TRUE
)

gapNumeric(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  tolerance = 0,
  tolerance_type = "absolute",
  include_lower_bound = TRUE,
  include_upper_bound = TRUE
)
```

Arguments

`solution`	A numeric value containing the correct answer for this numeric entry.
`response_identifier`	A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.
`points`	A numeric value, optional, representing the number of points for this gap. Default is 1
`placeholder`	A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine. Default is "".
`expected_length`	A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine. Default value is adjusted by solution size.
`tolerance`	A numeric value, optional, specifying the value for the upper and lower boundaries of the tolerance rate for candidate answers. Default is 0.
`tolerance_type`	A character value, optional, specifying the tolerance mode. Possible values: • "exact" • "absolute" - Default. • "relative"
`include_lower_bound`	A boolean value, optional, specifying whether the lower bound is included in the tolerance rate. Default is `TRUE`.

include_upper_bound
: A boolean value, optional, specifying whether the upper bound is included in the tolerance rate. Default is TRUE.

Value

An object of class NumericGap

See Also

[entry()][textGap()][textGapOpal()]

Examples

```
ng_min <- numericGap(5.1)

ng <- numericGap(solution = 5.1,
                 response_identifier  = "id_gap_1234",
                 points = 2,
                 placeholder = "put your answer here",
                 expected_length = 4,
                 tolerance = 5,
                 tolerance_type = "relative")
```

NumericGap-class *Class "NumericGap"*

Description

Class NumericGap is responsible for creating instances of input fields with numeric type of answers in question Entry type assessment tasks according to the QTI 2.1 standard.

Slots

response_identifier A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.

points A numeric value, optional, representing the number of points for this gap. Default is 1.

placeholder A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine.

expected_length A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine.

solution A numeric value containing the correct answer for this numeric entry.

tolerance A numeric value, optional, specifying the value for the upper and lower boundaries of the tolerance rate for candidate answers. Default is 0.

tolerance_type A character value, optional, specifying the tolerance mode. Possible values:

- "exact"

- "absolute" - Default.
- "relative"

include_lower_bound A boolean value, optional, specifying whether the lower bound is included in the tolerance rate. Default is TRUE.

include_upper_bound A boolean value, optional, specifying whether the upper bound is included in the tolerance rate. Default is TRUE.

See Also

Entry, TextGap, TextGapOpal and InlineChoice.

Examples

```
ng <- new("NumericGap",
          response_identifier = "id_gap_1234",
          points = 1,
          placeholder = "use this format xx.xxx",
          solution = 5,
          tolerance = 1,
          tolerance_type = "relative",
          include_lower_bound = TRUE,
          include_upper_bound = TRUE)
```

---

oneInColTable *Create object OneInColTable*

Description

Create object OneInColTable

Usage

```
oneInColTable(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  rows,
  rows_identifiers,
  cols,
  cols_identifiers,
  answers_identifiers,
  answers_scores = NA_real_,
  shuffle = TRUE,
  shuffle_rows = TRUE,
  shuffle_cols = TRUE,
```

```
  feedback = list(),
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

identifier	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
title	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
prompt	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
points	A numeric value, optional, representing the number of points for the entire task. It can also be calculated as the sum of points for individual answers, when provided. Default is 1.
rows	A character vector specifying answer options defined in rows of the table.
rows_identifiers	A character vector, optional, specifies identifiers of the rows of the table
cols	A character vector specifying answer options defined in columns of the table.
cols_identifiers	A character vector, optional, specifies identifiers of the columns of the table.
answers_identifiers	A character vector specifying couples of identifiers that combine the correct answers.
answers_scores	A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.
shuffle	A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
shuffle_rows	A boolean value, optional, determining whether to randomize the order of the choices only for the first elements of the answer tuples. Default is TRUE.
shuffle_cols	A boolean value, optional, determining whether to randomize the order of the choices only for the second elements of the answer tuples. Default is TRUE.
feedback	A list containing feedback message-object ModalFeedback for candidates.
calculator	A character, optional, determining whether to show a calculator to the candidate. Possible values: • "simple" • "scientific".
files	A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class OneInColTable

Examples

```
ct_min <- oneInColTable(content = "<p>\"One in column table\" task</p>",
                        rows = c("alfa", "beta", "gamma"),
                        rows_identifiers = c("a", "b", "g"),
                        cols = c("A", "B", "G", "a"),
                        cols_identifiers = c("as", "bs", "gs", "aas"),
                        answers_identifiers = c("a as", "b bs", "g gs", "a aas"))

ct <- oneInColTable(identifier = "id_task_1234",
                    title = "Table with one answer per column",
                    content = "<p>\"One in column table\" task</p>",
                    prompt = "Plain text, can be used instead of the content",
                    rows = c("alfa", "beta", "gamma"),
                    rows_identifiers = c("a", "b", "g"),
                    cols = c("A", "B", "G", "a"),
                    cols_identifiers = c("as", "bs", "gs", "aas"),
                    answers_identifiers = c("a as", "b bs", "g gs", "a aas"),
                    answers_scores = c(1, 0.5, 0.1, 1),
                    shuffle_rows = FALSE,
                    shuffle_cols = TRUE)
```

OneInColTable-class *Class "OneInColTable"*

Description

Class OneInColTable is responsible for creating assessment tasks according to the QTI 2.1 standard with a table of answer options, where only one correct answer in each column is possible.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.

- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`rows` A character vector specifying answer options as row names in the table or the first elements in couples in DirectedPair.

`rows_identifiers` A character vector, optional, specifying identifiers for answer options defined in rows of the table or identifiers of the first elements in couples in DirectedPair.

`cols` A character vector specifying answer options as column headers in the table or the second elements in couples in DirectedPair.

`cols_identifiers` A character vector, optional, specifying identifiers for answer options defined in columns of the table or identifiers of the second elements in couples in DirectedPair.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only in rows. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only in columns. Default is `TRUE`.

Examples

```
mt <- new("OneInColTable",
          identifier = "id_task_1234",
          title = "One in Col choice table",
          content = list("<p>\"One in col\" table task</p>",
```

```
                "<i>table description</i>"),
points = 5,
rows = c("row1", "row2", "row3", "row4"),
rows_identifiers = c("a", "b", "c", "d"),
cols = c("alfa", "beta", "gamma"),
cols_identifiers = c("k", "l", "m"),
answers_identifiers = c("a k", "d l", 'd m'),
shuffle = TRUE)
```

---

oneInRowTable *Create object OneInRowTable*

Description

Create object OneInRowTable

Usage

```
oneInRowTable(
  identifier = generate_id(),
  title = identifier,
  content = list(),
  prompt = "",
  points = 1,
  rows,
  rows_identifiers,
  cols,
  cols_identifiers,
  answers_identifiers,
  answers_scores = NA_real_,
  shuffle = TRUE,
  shuffle_rows = TRUE,
  shuffle_cols = TRUE,
  feedback = list(),
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

identifier	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
title	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
content	A character string or a list of character strings to form the text of the question, which may include HTML tags.

prompt — An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

points — A numeric value, optional, representing the number of points for the entire task. It can also be calculated as the sum of points for individual answers, when provided. Default is 1.

rows — A character vector specifying answer options defined in rows of the table.

rows_identifiers — A character vector, optional, specifies identifiers of the rows of the table

cols — A character vector specifying answer options defined in columns of the table.

cols_identifiers — A character vector, optional, specifies identifiers of the columns of the table.

answers_identifiers — A character vector specifying couples of identifiers that combine the correct answers.

answers_scores — A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

shuffle — A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.

shuffle_rows — A boolean value, optional, determining whether to randomize the order of the choices only for the first elements of the answer tuples. Default is TRUE.

shuffle_cols — A boolean value, optional, determining whether to randomize the order of the choices only for the second elements of the answer tuples. Default is TRUE.

feedback — A list containing feedback message-object ModalFeedback for candidates.

calculator — A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific".

files — A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class OneInRowTable

Examples

```
rt_min <- oneInRowTable(content = "<p>\"One in row table\" task</p>",
                        rows = c("alfa", "beta", "gamma", "alpha"),
                        rows_identifiers = c("a", "b", "g", "aa"),
                        cols = c("A", "B", "G"),
                        cols_identifiers = c("as", "bs", "gs"),
                        answers_identifiers = c("a as", "b bs", "g gs", "aa as"))
```

```
rt <- oneInRowTable(identifier = "id_task_1234",
                    title = "Table with one answer per row",
                    content = "<p>\"One in row table\" task</p>",
                    prompt = "Plain text, can be used instead of the content",
                    rows = c("alfa", "beta", "gamma", "alpha"),
                    rows_identifiers = c("a", "b", "g", "aa"),
                    cols = c("A", "B", "G"),
                    cols_identifiers = c("as", "bs", "gs"),
                    answers_identifiers = c("a as", "b bs", "g gs", "aa as"),
                    answers_scores = c(1, 0.5, 0.1, 1),
                    shuffle_rows = FALSE,
                    shuffle_cols = TRUE)
```

`OneInRowTable-class` *Class "OneInRowTable"*

Description

Class `OneInRowTable` is responsible for creating assessment tasks according to the QTI 2.1 standard with a table of answer options, where only one correct answer in each row is possible.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`rows` A character vector specifying answer options as row names in the table or the first elements in couples in DirectedPair.

`rows_identifiers` A character vector, optional, specifying identifiers for answer options defined in rows of the table or identifiers of the first elements in couples in DirectedPair.

`cols` A character vector specifying answer options as column headers in the table or the second elements in couples in DirectedPair.

`cols_identifiers` A character vector, optional, specifying identifiers for answer options defined in columns of the table or identifiers of the second elements in couples in DirectedPair.

`answers_identifiers` A character vector specifying couples of identifiers that combine the correct answers.

`answers_scores` A numeric vector, optional, where each number determines the number of points awarded to a candidate if they select the corresponding answer. If not assigned, the individual values for correct answers are calculated from the task points and the number of correct options.

`shuffle` A boolean value, optional, determining whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`shuffle_rows` A boolean value, optional, determining whether to randomize the order of the choices only in rows. Default is `TRUE`.

`shuffle_cols` A boolean value, optional, determining whether to randomize the order of the choices only in columns. Default is `TRUE`.

Examples

```
mt <- new("OneInRowTable",
          identifier = "id_task_1234",
          title = "One in Row choice table",
          content = list("<p>\"One in row\" table task</p>",
                         "<i>table description</i>"),
          points = 5,
          rows = c("row1", "row2", "row3", "row4"),
          rows_identifiers = c("a", "b", "c", "d"),
          cols = c("alfa", "beta", "gamma"),
          cols_identifiers = c("k", "l", "m"),
          answers_identifiers = c("a k", "b l", "c l", "d m"),
          shuffle = TRUE)
```

opal *Create an Opal LMS Connection Object*

Description

This helper function initializes an `Opal` object, a subclass of `LMS`, representing a connection to the Opal Learning Management System (LMS).

Usage

```
opal(api_user = NA_character_, endpoint = NA_character_)
```

Arguments

`api_user`	A character string specifying the API username.
`endpoint`	A character string specifying the API endpoint for the LMS.

Value

An object of class `Opal`, inheriting from `LMS`, which can be used to interact with the Opal LMS API.

Opal-class *Class Opal*

Description

The `Opal` class represents a specific implementation of a Learning Management System (LMS) that extends the abstract `LMS` class. This class is designed to facilitate interactions with the Opal LMS API.

Slots

`name` A character string representing the name/identifier of the LMS. Defaults to `"Opal"`.

`api_user` A character string specifying the API username for authentication.

`endpoint` A character string containing the API endpoint of the Opal LMS. This can be set using the environment variable `RQTI_API_ENDPOINT` with `Sys.setenv(RQTI_API_ENDPOINT='xxxxxxxxxxxxxx')` or placed in the `.Renviron` file.

See Also

LMS-class for the parent class.

`ordering` *Create object Ordering*

Description

Create object Ordering

Usage

```
ordering(
  identifier = generate_id(),
  title = identifier,
  choices,
  choices_identifiers = paste0("Choice", LETTERS[seq(choices)]),
  content = list(),
  prompt = "",
  points = 1,
  points_per_answer = TRUE,
  shuffle = TRUE,
  feedback = list(),
  calculator = NA_character_,
  files = NA_character_
)
```

Arguments

`identifier`	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
`title`	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
`choices`	A character vector containing the answers. The order of answers in the vector represents the correct response for the task.
`choices_identifiers`	A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.
`content`	A character string or a list of character strings to form the text of the question, which may include HTML tags.
`prompt`	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
`points`	A numeric value, optional, representing the number of points for the entire task. Default is 1.
`points_per_answer`	A boolean value indicating the scoring method. If `TRUE`, each selected answer will be scored individually. If `FALSE`, only fully correct answers will be scored with the maximum score. Default is `TRUE`.

shuffle	A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
feedback	A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.
calculator	A character, optional, determining whether to show a calculator to the candidate. Possible values: • "simple" • "scientific".
files	A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

Value

An object of class Ordering

Examples

```
ord_min <- ordering(prompt = "Set the right order:",
                        choices = c("Step1", "Step2", "Step3"))

ord <- ordering(identifier = "id_task_1234",
             title = "Order Task",
             choices = c("Step1", "Step2", "Step3"),
             choices_identifiers = c("a", "b", "c"),
             content = "<p>Set the right order</p>",
             prompt = "Plain text, can be used instead of content",
             points = 2,
             points_per_answer = FALSE,
             shuffle = FALSE,
             feedback = list(new("WrongFeedback",
                                 content = list("Wrong answer"))),
             calculator = "scientific-calculator",
             files = "text_book.pdf")
```

Ordering-class *Class "Ordering"*

Description

Class Ordering is responsible for creating assessment task according to QTI 2.1., where candidate has to place answers in a specific order

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`choices` A character vector containing the answers. The order of answers in the vector represents the correct response for the task.

`choices_identifiers` A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically. By default, identifiers are generated automatically according to the template "ChoiceL", where L is a letter representing the alphabetical order of the answer in the list.

`shuffle` A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.

`points_per_answer` A boolean value indicating the scoring method. If `TRUE`, each selected answer will be scored individually. If `FALSE`, only fully correct answers will be scored with the maximum score. Default is `TRUE`.

Examples

```
ord <- new("Ordering",
           identifier = "id_task_1234",
           title = "order",
           content = list("<p>Put these items in a right order</p>"),
           prompt = "",
           points = 2,
           feedback = list(),
           choices = c("first", "second", "third"),
           choices_identifiers = c("ChoiceA", "ChoiceB", "ChoiceC"),
           shuffle = TRUE,
           points_per_answer = TRUE)
```

`prepareQTIJSFiles-methods`

Prepare files to render them with QTIJS

Description

Prepare files to render them with QTIJS

Usage

```
prepareQTIJSFiles(object, dir = NULL)

## S4 method for signature 'AssessmentItem'
prepareQTIJSFiles(object, dir = "")

## S4 method for signature 'AssessmentSection'
prepareQTIJSFiles(object, dir = NULL)

## S4 method for signature 'AssessmentTest'
prepareQTIJSFiles(object, dir = NULL)

## S4 method for signature 'character'
prepareQTIJSFiles(object, dir = NULL)
```

Arguments

`object`	an instance of AssessmentItem, AssessmentTest, AssessmentTestOpal, AssessmentSection, or string path to xml, rmd or md files
`dir`	QTIJS path

prepare_renderer *Prepare qtijs renderer.*

Description

Starts server for qtijs, returns path of qtijs and the url of the server.

Usage

```
prepare_renderer(qtijs_path = qtijs_pkg_path())
```

Arguments

qtijs_path	The path to the qtijs renderer (qti.js), which will be started with servr::httw and to which xml files will be copied. Default is the QTIJS folder in the R package rqti local installation via the helper qtijs_pkg_path().

print.qti_validation_result *Print a QTI validation result*

Description

Print a QTI validation result

Usage

```
## S3 method for class 'qti_validation_result'
print(x, ...)
```

Arguments

x	A qti_validation_result object.
...	Unused.

Value

The input object, invisibly.

provide_file *Embed a local file as a downloadable hyperlink in R Markdown*

Description

Designed for inline use in R Markdown, e.g. `` `r provide_file("attachment.pdf")` ``.

Usage

```
provide_file(
  path,
  label = basename(path),
  mime = NULL,
  download = TRUE,
  warn_size_mb = 5
)
```

Arguments

path	Path to the local file.
label	Text shown to the user. Defaults to the file name.
mime	MIME type. If NULL, it is guessed from the extension.
download	Logical. If TRUE, adds the HTML download attribute.
warn_size_mb	Warn if file is larger than this many MB.

Details

The function reads a local file, encodes it as Base64, and returns an HTML anchor tag with a data: URI. This allows the file to be embedded directly into the rendered output instead of being referenced externally.

Value

knitr_asis object with an HTML hyperlink.

publishCourse *Publish a course on LMS*

Description

Publish a course on LMS

Usage

```
publishCourse(object, course_id)
```

Arguments

object	An S4 object of class LMS that represents a connection to the LMS.
course_id	A character string with course id in the LMS.

Value

Status code of the HTTP request.

---

publishCourse,missing-method

Publish a course on LMS

Description

This method publishes the course by its course id on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
## S4 method for signature 'missing'
publishCourse(object, course_id)
```

Arguments

object	An S4 object of class LMS that represents a connection to the LMS.
course_id	A character string with course id in the LMS.

Value

Status code of the HTTP request.

---

publishCourse,Opal-method

Publish a course on LMS Opal

Description

Publish a course on LMS Opal

Usage

```
## S4 method for signature 'Opal'
publishCourse(object, course_id)
```

Arguments

object	An S4 object of class Opal that represents a connection to the LMS.
course_id	A character string with course id in the LMS.

Value

Status code of the HTTP request.

---

qtiContributor *Constructor function for class QtiContributor*

Description

Creates object of QtiContributor-class

Usage

```
qtiContributor(
  name = Sys.getenv("RQTI_AUTHOR"),
  role = "author",
  contribution_date = ifelse(name != "", Sys.Date(), as.Date(NA))
)
```

Arguments

name	A character string representing the name of the author.
role	A character string kind of contribution. Possible values: author, publisher, unknown, initiator, terminator, validator, editor, graphical designer, technical implementer, content provider, technical validator, educational validator, script writer, instructional designer, subject matter expert. Default is "author".
contribution_date	A character string representing date of the contribution. Default is the current system date, when contributor is assigned.

Examples

```
creator= qtiContributor("Max Mustermann", "technical validator")
```

QtiContributor-class *Class QtiContributor*

Description

This class stores metadata information about contributors.

Slots

`name` A character string representing the name of the author. By default it takes value from environment variable 'RQTI_AUTHOR'.

`role` A character string kind of contribution. Possible values: author, publisher, unknown, initiator, terminator, validator, editor, graphical designer, technical implementer, content provider, technical validator, educational validator, script writer, instructional designer, subject matter expert. Default is "author".

`contribution_date` A character string representing date of the contribution. Default is the current system date.

qtijs_pkg_path *Shortcut for the qtijs path of the rqti package local installation.*

Description

Shortcut for the qtijs path of the rqti package local installation.

Usage

```
qtijs_pkg_path()
```

Examples

```
qtijs_pkg_path()
```

qtiMetadata *Constructor function for class QtiMetadata*

Description

Creates object of QtiMetadata-class

Usage

```
qtiMetadata(
  contributor = list(),
  description = "",
  rights = Sys.getenv("RQTI_RIGHTS"),
  version = NA_character_
)
```

Arguments

contributor	A list of objects QtiContributor-type that holds metadata information about the authors.
description	A character string providing a textual description of the content of this learning object.
rights	A character string describing the intellectual property rights and conditions of use for this learning object. By default it takes value from environment variable 'RQTI_RIGHTS'.
version	A character string representing the edition/version of this learning object.

Examples

```
creator= qtiMetadata(qtiContributor("Max Mustermann"),
                     description = "Task description",
                     rights = "This file is Copyright (C) 2024 Max
                     Mustermann, all rights reserved.",
                     version = "1.0")
```

QtiMetadata-class *Class QtiMetadata*

Description

This class stores metadata information such as contributors, description, rights and version for QTI-compliant tasks or tests.

Slots

`contributor` A list of objects QtiContributor-type that holds metadata information about the authors.

`description` A character string providing a textual description of the content of this learning object.

`rights` A character string describing the intellectual property rights and conditions of use for this learning object. By default it takes value from environment variable 'RQTI_RIGHTS'.

`version` A character string representing the edition/version of this learning object.

`qti_contributor` *Constructor function for class QtiContributor*

Description

Creates object of QtiContributor-class

Usage

```
qti_contributor(
  name = Sys.getenv("RQTI_AUTHOR"),
  role = "author",
  contribution_date = ifelse(name != "", Sys.Date(), as.Date(NA))
)
```

Arguments

`name`	A character string representing the name of the author.
`role`	A character string kind of contribution. Possible values: author, publisher, unknown, initiator, terminator, validator, editor, graphical designer, technical implementer, content provider, technical validator, educational validator, script writer, instructional designer, subject matter expert. Default is "author".
`contribution_date`	A character string representing date of the contribution. Default is the current system date, when contributor is assigned.

Examples

```
creator= qti_contributor("Max Mustermann", "technical validator")
```

qti_metadata *Constructor function for class QtiMetadata*

Description

Creates object of QtiMetadata-class

Usage

```
qti_metadata(
  contributor = list(),
  description = "",
  rights = Sys.getenv("RQTI_RIGHTS"),
  version = NA_character_
)
```

Arguments

contributor	A list of objects QtiContributor-type that holds metadata information about the authors.
description	A character string providing a textual description of the content of this learning object.
rights	A character string describing the intellectual property rights and conditions of use for this learning object. By default it takes value from environment variable 'RQTI_RIGHTS'.
version	A character string representing the edition/version of this learning object.

Examples

```
creator= qti_metadata(qtiContributor("Max Mustermann"),
                      description = "Task description",
                      rights = "This file is Copyright (C) 2024 Max
                      Mustermann, all rights reserved.",
                      version = "1.0")
```

render_opal *Render Rmd directly in Opal via API*

Description

Render Rmd directly in Opal via API

Usage

```
render_opal(input, ...)
```

Arguments

input	(the path to the input Rmd document)
...	required for passing arguments when knitting

Details

Customize knit function in the Rmd file using the following YAML setting after the word knit `knit: rqti::render_opal`.

Value

A list with the key, display name, and URL of the resource in Opal.

Examples

```
file <- system.file("exercises/sc1.Rmd", package = 'rqti')
render_opal(file)
```

---

render_qtijs *Render an Rmd/md/xml file or rqti-object as qti xml with qtijs.*

Description

Generates the qti xml file via rmd2xml. The xml is copied into the qtijs folder which transforms the xml into HTML. Finally, the HTML is displayed and the user sees a preview of the exercise or test.

Usage

```
render_qtijs(
  input,
  preview_feedback = FALSE,
  qtijs_path = qtijs_pkg_path(),
  ...
)
```

Arguments

input	The path to the input Rmd/md/xml document or an AssessmentItem, AssessmentTest, AssessmentTestOpal, AssessmentSection object.
preview_feedback	A boolean value; optional. Set TRUE value to always display feedback (for example, as a modal answer). Default is FALSE.
qtijs_path	The path to the qtijs renderer (qti.js), which will be started with servr::httw and to which xml files will be copied. Default is the QTIJS folder in the R package rqti local installation via the helper qtijs_pkg_path().
...	required for passing arguments when knitting

Details

Requires a running qtijs server, which can be started with start_server().

The preview is automatically loaded into the RStudio viewer pane if run in RStudio. Alternatively you can just open the browser at the corresponding local server url which is displayed after rendering is finished. Since the function is supposed to be called via the Knit-Button in RStudio, it defaults to the RStudio viewer pane.

Customize knit function in the Rmd file using the following YAML setting after the word knit `knit: rqti::render_qtijs`.

Value

An URL of the corresponding local server to display the rendering result.

Examples

```
file <- system.file("exercises/sc1.Rmd", package = 'rqti')
render_qtijs(file)
```

`render_xml` *Render a single xml file with qtijs.*

Description

Uses qtijs to render a single xml file in the RStudio viewer pane with a local server.

Usage

```
render_xml(input, qtijs_path = qtijs_pkg_path())
```

Arguments

`input`	The path to the input Rmd/md/xml document or an AssessmentItem, AssessmentTest, AssessmentTestOpal, AssessmentSection object.
`qtijs_path`	The path to the qtijs renderer (qti.js), which will be started with servr::httw and to which xml files will be copied. Default is the QTIJS folder in the R package rqti local installation via the helper qtijs_pkg_path().

Value

nothing, has side effects

Examples

```
file <- system.file("exercises/sc1d.xml", package = 'rqti')
render_qtijs(file)
```

render_zip *Render a zipped qti archive with qtijs.*

Description

Uses qtijs to render a zipped qti archive in the RStudio viewer pane with a local server.

Usage

```
render_zip(input, qtijs_path = qtijs_pkg_path())
```

Arguments

input	The path to the input Rmd/md/xml document or an AssessmentItem, AssessmentTest, AssessmentTestOpal, AssessmentSection object.
qtijs_path	The path to the qtijs renderer (qti.js), which will be started with servr::httw and to which xml files will be copied. Default is the QTIJS folder in the R package rqti local installation via the helper qtijs_pkg_path().

Value

nothing, has side effects

rmd2xml *Create qti-XML task file from Rmd (md) description*

Description

Create XML file for question specification from Rmd (md) description according to qti 2.1 information model

Usage

```
rmd2xml(file, path = getwd(), verification = FALSE)
```

Arguments

file	A string of path to file with markdown description of question.
path	A string, optional; a folder to store xml file. Default is working directory.
verification	A boolean value, optional; enable validation of the xml file. Default is FALSE.

Value

The path string to the xml file.

Examples

```
## Not run:
# creates folder with xml (side effect)
rmd2xml("task.Rmd", "target_folder", TRUE)

## End(Not run)
```

`rmd2zip` *Create test zip file with one task xml file from Rmd (md) description*

Description

Create zip file with test, that contains one xml question specification generated from Rmd (md) description according to qti 2.1 information model

Usage

```
rmd2zip(file, path = getwd(), verification = FALSE)
```

Arguments

`file`	A string of path to file with markdown description of question.
`path`	A string, optional; a folder to store xml file. Default is working directory.
`verification`	A boolean value, optional; enable validation of the xml file. Default is `FALSE`.

Value

The path string to the zip file.

Examples

```
## Not run:
# creates folder with zip (side effect)
rmd2zip("task.Rmd", "target_folder", TRUE)

## End(Not run)
```

`section` *Create a section as part of a test content*

Description

Create an AssessmentSection `rqti`-object as part of a test content. The function accepts `rqti` tasks and sections directly, as well as task definitions stored in native `rqti` Rmd/md files, `exams`-style Rmd files, or XML files.

Usage

```
section(
  content,
  n_variants = 1L,
  seed_number = NULL,
  id = NULL,
  by = "variants",
  selection = NULL,
  title = character(0),
  time_limits = NA_integer_,
  visible = TRUE,
  shuffle = FALSE,
  max_attempts = NA_integer_,
  allow_comment = TRUE
)
```

Arguments

`content`	A list of Rmd, md, or xml files, or task (AssessmentItem) or section (AssessmentSection) objects. Supported Rmd/md inputs include native `rqti` files and Rmd files written in the `exams` format.
`n_variants`	An integer value indicating the number of task variants to create from Rmd files. Default is `1`.
`seed_number`	An integer vector, optional, specifying seed numbers to reproduce the result of calculations.
`id`	A character value, optional, serving as the identifier of the assessment section.
`by`	A character with two possible values: "variants" or "files", indicating the type of the test structure. Default is "variants".
`selection`	An integer value, optional, defining how many children of the section are delivered in the test. Default is `NULL`, meaning "no selection".
`title`	A character value, optional, representing the title of the section. If not provided, it defaults to `identifier`.
`time_limits`	An integer value, optional, controlling the amount of time a candidate is allowed for this part of the test.

visible	A boolean value, optional, indicating whether the title of this section is shown in the hierarchy of the test structure. Default is TRUE.
shuffle	A boolean value, optional, responsible for randomizing the order in which the assessment items and subsections are initially presented to the candidate. Default is FALSE.
max_attempts	An integer value, optional, enabling the maximum number of attempts allowed for a candidate to pass this section.
allow_comment	A boolean value, optional, enabling candidates to leave comments on each question of the section. Default is TRUE.

Value

An object of class AssessmentSection.

See Also

test(), test4opal()

Examples

```
sc <- new("SingleChoice", prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
# Since ready-made S4 "AssessmentItem" objects are taken, in this example a
#permanent section consisting of two tasks is created.
s <- section(c(sc, es), title = "Section with nonrandomized tasks")

# Since Rmd files with randomization of internal variables are taken,
#in this example 2 variants are created with a different seed number for each.
path <- system.file("rmarkdown/templates/", package='rqti')
file1 <- file.path(path, "singlechoice-simple/skeleton/skeleton.Rmd")
file2 <- file.path(path, "singlechoice-complex/skeleton/skeleton.Rmd")
s <- section(c(file1, file2), n_variants = 2,
title = "Section with two variants of tasks")
```

---

singleChoice *Create object SingleChoice*

Description

Create object SingleChoice

Usage

```
singleChoice(
  identifier = generate_id(),
  title = identifier,
  choices,
  choice_identifiers = paste0("Choice", LETTERS[seq(choices)]),
```

```
  solution = 1,
  content = list(),
  prompt = "",
  points = 1,
  feedback = list(),
  orientation = "vertical",
  shuffle = TRUE,
  calculator = NA_character_,
  files = NA_character_,
  scoring_scheme = "standard"
)
```

Arguments

identifier	A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.
title	A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.
choices	A character vector defining a set of answer options in the question.
choice_identifiers	A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.
solution	A numeric value, optional. Represents the index of the correct answer in the choices slot. By default, the first item in the choices slot is considered the correct answer. Default is 1.
content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
prompt	An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".
points	A numeric value, optional, representing the number of points for the entire task. Default is 1.
feedback	A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.
orientation	A character, determining whether to place answers in vertical or horizontal mode. Possible values: • "vertical" - Default, • "horizontal".
shuffle	A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is TRUE.
calculator	A character, optional, determining whether to show a calculator to the candidate. Possible values: • "simple" • "scientific".

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`scoring_scheme` A character value, optional, defining how response options are scored. Possible values:

- "standard" - the correct answer receives full points and incorrect answers receive 0. This is used by default.
- "penalty" - the correct answer receives full points and incorrect answers receive a negative score of -points/(k-1), where k is the number of response options.

Value

An object of class SingleChoice

Examples

```
sc_min <- singleChoice(prompt = "Question?",
                       choices = c("Answer1", "Answer2", "Answer3"))

sc <- singleChoice(identifier = "id_task_1234",
                   title = "Single Choice Task",
                   content = "<p>Pick up the right option</p>",
                   prompt = "Plain text, can be used instead of content",
                   points = 2,
                   feedback = list(new("WrongFeedback",
                                   content = list("Wrong answer"))),
                   calculator = "scientific-calculator",
                   files = "text_book.pdf",
                   choices = c("option 1", "option 2", "option 3"),
                   choice_identifiers = c("ChoiceA", "ChoiceB", "ChoiceC"),
                   shuffle = TRUE,
                   orientation = "vertical",
                   solution = 2)
```

SingleChoice-class *Class "SingleChoice"*

Description

Class `SingleChoice` is responsible for creating single-choice assessment tasks according to the QTI 2.1 standard.

Slots

`identifier` A character representing the unique identifier of the assessment task. By default, it is generated as 'id_task_dddd', where dddd represents random digits.

`title` A character representing the title of the XML file associated with the task. By default, it takes the value of the identifier.

`content` A list of character content to form the text of the question, which can include HTML tags. For tasks of the Entry type, it must also contain at least one instance of Gap objects, such as TextGap, TextGapOpal, NumericGap, or InlineChoice.

`prompt` An optional character representing a simple question text, consisting of one paragraph. This can supplement or replace content in the task. Default is "".

`points` A numeric value, optional, representing the number of points for the entire task. Default is 1, but pay attention:

- For tasks of the Entry type, it is calculated as the sum of the gap points by default.
- For tasks of the DirectedPair, the default is calculated as 0.5 points per pair.
- For tasks of the MatchTable type, it can also be calculated as the sum of points for individual answers, when provided.
- For tasks of the MultipleChoice type, points is numeric vector and required. Each number in this vector determines the number of points that will be awarded to a candidate if they select the corresponding answer. The order of the scores must match the order of the `choices`. It is possible to assign negative values to incorrect answers. All answers with a positive score are considered correct.

`feedback` A list containing feedback messages for candidates. Each element of the list should be an instance of either ModalFeedback, CorrectFeedback, or WrongFeedback class.

`calculator` A character, optional, determining whether to show a calculator to the candidate. Possible values:

- "simple"
- "scientific"

`files` A character vector, optional, containing paths to files that will be accessible to the candidate during the test/exam.

`metadata` An object of class QtiMetadata that holds metadata information about the task.

`choices` A character vector defining a set of answer options in the question.

`choice_identifiers` A character vector, optional, containing a set of identifiers for answers. By default, identifiers are generated automatically according to the template "ChoiceD", where D is a letter representing the alphabetical order of the answer in the list.

`shuffle` A boolean value indicating whether to randomize the order in which the choices are initially presented to the candidate. Default is `TRUE`.

`orientation` A character, determining whether to place answers in vertical or horizontal mode. Possible values:

- "vertical" - Default.
- "horizontal"

`solution` A numeric value, optional. Represents the index of the correct answer in the `choices` slot. By default, the first item in the `choices` slot is considered the correct answer. Default is `1`.

`scoring_scheme` A character value, optional, defining how response options are scored. Possible values:

- "standard" - the correct answer receives full points and incorrect answers receive 0. This is used by default.
- "penalty" - the correct answer receives full points and incorrect answers receive a negative score of -1/(k-1), where k is the number of response options.

Examples

```
sc <- new("SingleChoice",
          identifier = "id_task_1234",
          title = "Single Choice Task",
          content = list("<p>Pick up the right option</p>"),
          prompt = "Plain text, can be used instead of content",
          points = 2,
          feedback = list(new("WrongFeedback", content = list("Wrong answer"))),
          calculator = "scientific-calculator",
          files = "text_book.pdf",
          choices = c("option 1", "option 2", "option 3", "option 4"),
          choice_identifiers = c("ChoiceA", "ChoiceB", "ChoiceC", "ChoiceD"),
          shuffle = TRUE,
          orientation = "vertical",
          solution = 2)
```

---

start_server *Start qtijs renderer as a local server.*

Description

This function starts an http server with the qtijs renderer. The renderer performs the conversion of qti.xml into HTML.

Usage

```
start_server(qtijs_path = qtijs_pkg_path(), daemon = T)
```

Arguments

qtijs_path	The path to the qtijs renderer (qti.js), which will be started with servr::httw and to which xml files will be copied. Default is the QTIJS folder in the R package rqti local installation via the helper qtijs_pkg_path().
daemon	This parameter is forwarded to `servr::httw` and should always be TRUE (the default). FALSE is only used for testing purposes when called via `callr::bg()`

Details

The server has to be started manually by the user, otherwise the Knit-Button will not work. The Knit-Button starts a new session and invoking a server there does not work. You can automatically start the server via an .RProfile file on start up.

Value

The URL string of the qtijs server.

Examples

```
## Not run:
start_server()

## End(Not run)
```

---

stop_server *Stop QTIJS local server*

Description

Stop QTIJS local server

Usage

```
stop_server()
```

Value

nothing, has side effects

---

test *Create a test*

Description

Create an AssessmentTest `rqti`-object.

Usage

```
test(
  content,
  identifier = "test_identifier",
  title = "Test Title",
  time_limit = NULL,
  max_attempts = 1L,
  fallback_titles = "generic",
  academic_grading = NULL,
  grade_label = c(en = "Grade", de = "Note"),
  table_label = c(en = "Grade", de = "Note"),
  navigation_mode = "nonlinear",
  submission_mode = "individual",
```

```
  allow_comment = TRUE,
  rebuild_variables = TRUE,
  stylesheet_path = NULL,
  contributor = list(),
  description = "",
  rights = Sys.getenv("RQTI_RIGHTS"),
  version = "0.0.9"
)
```

Arguments

content	A list containing AssessmentSection objects.
identifier	A character value indicating the identifier of the test file. Default is 'test_identifier'.
title	A character value, optional, representing the file title. Default is 'Test Title'.
time_limit	An integer value, optional, controlling the time given to a candidate for the test in minutes. Default is NULL.
max_attempts	An integer value, optional, indicating the maximum number of attempts allowed for the candidate. Default is 1.
fallback_titles	A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".
academic_grading	A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test. Each grade corresponds to the minimum percentage score required to achieve it. A helper function german_grading() is available to generate a common German grading scheme. The default is NULL, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use german_grading().
grade_label	A character value, optional; a short message that shows with a grade in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
table_label	A character value, optional; a concise message to display as the column title of the grading table in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.
navigation_mode	A character value, optional, determining the general paths that the candidate may have during the exam. Two mode options are possible: - 'linear': Candidate is not allowed to return to previous questions. - 'nonlinear': Candidate is free to navigate; used by default.

`submission_mode`
: A character value, optional, determining when the candidate's responses are submitted for response processing. One of two mode options is possible: - 'individual': Submit candidates' responses on an item-by-item basis; used by default. - 'simultaneous': Candidates' responses are submitted all together by the end of the test.

`allow_comment`
: A boolean, optional, enabling the candidate to leave comments in each question. Default is `TRUE`.

`rebuild_variables`
: A boolean, optional, enabling the recalculation of variables and reshuffling the order of choices for each item-attempt. Default is `TRUE`.

`stylesheet_path`
: A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When `academic_grading` is set, the default stylesheet `styles/rqti.css` is included automatically; a user-defined stylesheet is added in addition and may override default styles.

`contributor`
: A list of objects QtiContributor-type that holds metadata information about the authors.

`description`
: A character string providing a textual description of the content of this learning object.

`rights`
: A character string describing the intellectual property rights and conditions of use for this learning object. By default it takes value from environment variable 'RQTI_RIGHTS'.

`version`
: A character string representing the edition/version of this learning object.

Value

An AssessmentTest object.

See Also

`test4opal()`, `section()`, AssessmentTest, AssessmentSection

Examples

```
sc <- new("SingleChoice", prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
s <- section(c(sc, es), title = "Section with nonrandomized tasks")
t <- test(s, title = "Example of the Exam")
```

`test4opal` *Create a test for LMS OPAL*

Description

Create an AssessmentTestOpal `rqti`-object.

Usage

```
test4opal(
  content,
  identifier = "test_identifier",
  title = "Test Title",
  time_limit = NULL,
  max_attempts = 1L,
  files = NULL,
  calculator = NULL,
  fallback_titles = "generic",
  academic_grading = NULL,
  grade_label = c(en = "Grade", de = "Note"),
  table_label = c(en = "Grade", de = "Note"),
  navigation_mode = "nonlinear",
  submission_mode = "individual",
  allow_comment = TRUE,
  rebuild_variables = TRUE,
  show_test_time = TRUE,
  mark_items = TRUE,
  keep_responses = FALSE,
  stylesheet_path = NULL,
  contributor = list(),
  description = "",
  rights = Sys.getenv("RQTI_RIGHTS"),
  version = "0.0.9"
)
```

Arguments

`content`	A list containing AssessmentSection objects.
`identifier`	A character value indicating the identifier of the test file. Default is 'test_identifier'.
`title`	A character value, optional, representing the file title. Default is 'Test Title'.
`time_limit`	An integer value, optional, controlling the time given to a candidate for the test in minutes. Default is `NULL`.
`max_attempts`	An integer value, optional, indicating the maximum number of attempts allowed for the candidate. Default is 1.
`files`	A character vector, optional; paths to files that will be accessible to the candidate during the test/exam.

`calculator` A character, optional; determines whether to show a calculator to the candidate. Possible values:

- 'simple'
- 'scientific'. Default is `NULL`.

`fallback_titles`

A character value, optional, controlling how titles are assigned when no explicit title is provided. Possible values are "filename" (use filenames as titles) and "generic" (use generic labels such as "Section 1", "Section 1.2", or "Task 1.2.1"). Default is "generic".

`academic_grading`

A named numeric vector that defines the grade table shown to the candidate as feedback at the end of the test.

Each grade corresponds to the minimum percentage score required to achieve it. A helper function `german_grading()` is available to generate a common German grading scheme.

The default is `NULL`, which means that no grading table is shown. To display a grading table, provide a named numeric vector or use `german_grading()`.

`grade_label` A character value, optional; a short message that shows with a grade in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.

`table_label` A character value, optional; a concise message to display as the column title of the grading table in the final feedback; for multilingual use, it can be a named vector with two-letter ISO language codes as names (e.g., c(en="Grade", de="Note")); during test creation, it takes the value for the language of the operating system; c(en="Grade", de="Note")is default.

`navigation_mode`

A character value, optional, determining the general paths that the candidate may have during the exam. Two mode options are possible: - 'linear': Candidate is not allowed to return to previous questions. - 'nonlinear': Candidate is free to navigate; used by default.

`submission_mode`

A character value, optional, determining when the candidate's responses are submitted for response processing. One of two mode options is possible: - 'individual': Submit candidates' responses on an item-by-item basis; used by default. - 'simultaneous': Candidates' responses are submitted all together by the end of the test.

`allow_comment` A boolean, optional, enabling the candidate to leave comments in each question. Default is `TRUE`.

`rebuild_variables`

A boolean, optional, enabling the recalculation of variables and reshuffling the order of choices for each item-attempt. Default is `TRUE`.

`show_test_time` A boolean, optional, determining whether to show candidate elapsed processing time without a time limit. Default is `TRUE`.

mark_items	A boolean, optional, determining whether to allow candidate marking of questions. Default is TRUE.
keep_responses	A boolean, optional, determining whether to save the candidate's answers from the previous attempt. Default is FALSE.
stylesheet_path	A character value, optional, specifying the path to a custom CSS stylesheet. If provided, the stylesheet is included at the assessment test level and applied during rendering. When academic_grading is set, the default stylesheet styles/rqti.css is included automatically; a user-defined stylesheet is added in addition and may override default styles.
contributor	A list of objects QtiContributor-type that holds metadata information about the authors.
description	A character string providing a textual description of the content of this learning object.
rights	A character string describing the intellectual property rights and conditions of use for this learning object. By default it takes value from environment variable 'RQTI_RIGHTS'.
version	A character string representing the edition/version of this learning object.

Value

An AssessmentTestOpal object

See Also

test(), section(), AssessmentTestOpal, AssessmentSection

Examples

```
sc <- new("SingleChoice", prompt = "Question", choices = c("A", "B", "C"))
es <- new("Essay", prompt = "Question")
s <- section(c(sc, es), title = "Section with nonrandomized tasks")
t <- test4opal(s, title = "Example of the Exam",
show_test_time = FALSE)
```

---

textGap *Create object TextGap*

Description

Create object TextGap

Usage

```
textGap(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  case_sensitive = FALSE
)

gapText(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  case_sensitive = FALSE
)
```

Arguments

solution	A character vector containing the values considered as correct answers.
response_identifier	A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.
points	A numeric value, optional, representing the number of points for this gap. Default is 1
placeholder	A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine. Default is "".
expected_length	A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine. Default value is adjusted by solution size.
case_sensitive	A boolean value, determining whether the evaluation of the correct answer is case sensitive. Default is FALSE.

Value

An object of class TextGap

See Also

[entry()][numericGap()][textGapOpal()]

Examples

```
tg_min <- textGap("answer")

tg <- textGap(solution = "answer",
```

```
response_identifier  = "id_gap_1234",
points = 2,
placeholder = "put your answer here",
expected_length = 20,
case_sensitive = TRUE)
```

TextGap-class *Class "TextGap"*

Description

Class TextGap is responsible for creating instances of input fields with text type of answers in question Entry type assessment tasks according to the QTI 2.1 standard.

Slots

response_identifier A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.

points A numeric value, optional, representing the number of points for this gap. Default is 1.

placeholder A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine.

expected_length A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine.

solution A character vector containing the values considered as correct answers.

case_sensitive A boolean value, determining whether the evaluation of the correct answer is case sensitive. Default is FALSE.

See Also

Entry, NumericGap, TextGapOpal and InlineChoice.

Examples

```
tg <- new("TextGap",
          response_identifier = "id_gap_1234",
          points = 2,
          placeholder = "do not put special characters",
          expected_length = 20,
          solution = c("answer", "answerr", "aanswer"),
          case_sensitive = FALSE)
```

textGapOpal *Create object TextGapOpal*

Description

Create object TextGapOpal

Usage

```
textGapOpal(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  case_sensitive = FALSE,
  tolerance = 0
)

gapTextOpal(
  solution,
  response_identifier = generate_id(type = "gap"),
  points = 1,
  placeholder = "",
  expected_length = size_gap(solution),
  case_sensitive = FALSE,
  tolerance = 0
)
```

Arguments

solution	A character vector containing the values considered as correct answers.
response_identifier	A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.
points	A numeric value, optional, representing the number of points for this gap. Default is 1
placeholder	A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine. Default is "".
expected_length	A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine. Default value is adjusted by solution size.
case_sensitive	A boolean value, determining whether the evaluation of the correct answer is case sensitive. Default is FALSE.
tolerance	A numeric value defining how many characters will be taken into account to tolerate spelling mistakes in the evaluation of candidate answers. Default is 0.

Value

An object of class TextGapOpal

See Also

[entry()][numericGap()][textGap()]

Examples

```
tgo_min <- textGapOpal("answer")

tgo <- textGapOpal(solution = "answer",
             response_identifier  = "id_gap_1234",
             points = 2,
             placeholder = "put your answer here",
             expected_length = 20,
             case_sensitive = TRUE,
             tolerance = 2)
```

TextGapOpal-class *Class "TextGapOpal"*

Description

Class `TextGapOpal` is responsible for creating instances of input fields with text type of answers in question Entry type assessment tasks according to the QTI 2.1 standard for LMS Opal.

Slots

`response_identifier` A character value representing an identifier for the answer. By default, it is generated as 'id_gap_dddd', where dddd represents random digits.

`points` A numeric value, optional, representing the number of points for this gap. Default is `1`.

`placeholder` A character value, optional, responsible for placing helpful text in the text input field in the content delivery engine.

`expected_length` A numeric value, optional, responsible for setting the size of the text input field in the content delivery engine.

`solution` A character vector containing the values considered as correct answers.

`case_sensitive` A boolean value, determining whether the evaluation of the correct answer is case sensitive. Default is `FALSE`.

`tolerance` A numeric value defining how many characters will be taken into account to tolerate spelling mistakes in the evaluation of candidate answers. Default is `0`.

See Also

Entry, NumericGap, TextGap and InlineChoice.

Examples

```
tgo <- new("TextGapOpal",
          response_identifier = "id_gap_1234",
          points = 2,
          placeholder = "do not put special characters",
          expected_length = 20,
          solution = "answer",
          case_sensitive = FALSE,
          tolerance = 1)
```

---

updateCourseElementResource

Update the referenced learning resource of a course element in the LMS

Description

Update the referenced learning resource of a course element in the LMS

Usage

```
updateCourseElementResource(object, course_id, ...)
```

Arguments

object	An S4 object of class LMS that represents a connection to the LMS.
course_id	A character string with course id in the LMS.
...	Additional arguments to be passed to the method, if applicable.

Value

Response of the HTTP request.

---

updateCourseElementResource,missing-method

Update the referenced learning resource of a course element in the LMS

Description

This method updates the learning resource by its course id on Learning Management System (LMS). If no LMS connection object is provided, it attempts to guess the connection using default settings (e.g., environment variables). If the connection cannot be established, an error is thrown.

Usage

```
## S4 method for signature 'missing'
updateCourseElementResource(object, course_id, ...)
```

Arguments

object	An S4 object of class LMS that represents a connection to the LMS.
course_id	A character string with course id in the LMS.
...	Additional arguments to be passed to the method, if applicable.

Value

Response of the HTTP request.

---

updateCourseElementResource,Opal-method

Update the referenced learning resource of a course element in the LMS Opal

---

Description

Update the referenced learning resource of a course element in the LMS Opal

Usage

```
## S4 method for signature 'Opal'
updateCourseElementResource(
  object,
  course_id,
  node_id,
  resource_id,
  publish = TRUE
)
```

Arguments

object	An S4 object of class LMS that represents a connection to the LMS.
course_id	A character string with the course ID. You can find this in the course's details (Resource ID) in the LMS.
node_id	A character string with the course element ID. This can be found, for example, in the course editor under the "Title and Description" tab of the respective course element in the LMS Opal.
resource_id	A character string with the ID of the resource. This can be found in the details view of the desired resource within the LMS.

publish A boolean value that determines whether the course should be published after the resource is updated. If TRUE (default), the course will be published immediately after the update. If FALSE, the course will not be published automatically, leaving it in an unpublished state until manual publication.

Value

The response of the HTTP request made to update the resource. If the course is published, an additional message about the publishing status is returned.

---

upload2LMS *Upload content to LMS*

Description

This is a generic function that handles the process of uploading content to a Learning Management System (LMS). The content can be in the form of an AssessmentTest, AssessmentTestOpal, AssessmentItem object, or a file in Rmd, Markdown, zip or XML format.

This is a method that handles the process of uploading content to a Learning Management System (LMS). The content can be in the form of an AssessmentTest, AssessmentTestOpal, AssessmentItem object, or a file in Rmd, Markdown, zip or XML format.

This is a generic function that handles the process of uploading content to LMS Opal. The content can be in the form of an AssessmentTest, AssessmentTestOpal, AssessmentItem object, or a file in Rmd, Markdown, zip or XML format.

Usage

```
upload2LMS(object, test, ...)

## S4 method for signature 'LMS'
upload2LMS(object, test, ...)

## S4 method for signature 'Opal'
upload2LMS(
  object,
  test,
  display_name = NULL,
  access = 4,
  overwrite = TRUE,
  open_in_browser = TRUE,
  as_survey = FALSE
)
```

Arguments

object	An S4 object representing the LMS, such as an instance of the Opal class.
test	An AssessmentTest, AssessmentTestOpal or AssessmentItem objects, or a character string with path to Rmd/md, zip or XML files.
...	Additional arguments to be passed to the method, if applicable.
display_name	A length one character vector to entitle resource in OPAL; file name without extension or identifier of the object by default; optional.
access	An integer value, optional; it is responsible for publication status, where 1 - only those responsible for this learning resource; 2 - responsible and other authors; 3 - all registered users; 4 - registered users and guests. Default is 4.
overwrite	A boolean value. If TRUE, and a file with the specified display name already exists, it will be overwritten. Default is TRUE.
open_in_browser	A boolean value; optional; it controls whether to open a URL in default browser. Default is TRUE.
as_survey	A boolean value, optional. If TRUE, the resource will be treated as a survey; if FALSE, as a test. Default is FALSE.

---

upload2opal *Upload a resource on OPAL*

Description

Function upload2opal() takes full prepared zip archive of QTI-test or QTI-task and uploads it to the OPAL.

Usage

```
upload2opal(
  test,
  display_name = NULL,
  access = 4,
  overwrite = TRUE,
  endpoint = NULL,
  open_in_browser = TRUE,
  as_survey = FALSE,
  api_user = NULL
)
```

Arguments

test	A length one character vector of AssessmentTest, AssessmentTestOpal or AssessmentItem objects, Rmd/md or XML files; required.
display_name	A length one character vector to entitle file in OPAL; file name without extension by default; optional.

access	An integer value, optional; it is responsible for publication status, where 1 - only those responsible for this learning resource; 2 - responsible and other authors; 3 - all registered users; 4 - registered users and guests. Default is 4.
overwrite	A boolean value; if the value is TRUE, if only one file with the specified display name is found, it will be overwritten. Default is TRUE.
endpoint	A string of endpoint of LMS Opal; by default it is got from environment variable RQTI_API_ENDPOINT. To set a global environment variable, you need to call Sys.setenv(RQTI_API_ENDPOINT='xxxxxxxxxxxxxx') or you can put these command into .Renviron.
open_in_browser	A boolean value; optional; it controls whether to open a URL in default browser. Default is TRUE.
as_survey	A boolean value; optional; it controls resource type (test r survey). Default is FALSE.
api_user	A character value of the username in the OPAL.

Value

A list with the key, display name, and URL of the resource in Opal.

Examples

```
file <- system.file("exercises/sc1.Rmd", package='rqti')
upload2opal(file, "task 1", open_in_browser = FALSE)
```

---

verify_qti *Validate QTI XML*

Description

S4 generic for validating QTI documents in various formats.

Usage

```
verify_qti(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)

## S4 method for signature 'character'
```

```
verify_qti(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)

## S4 method for signature 'xml_document'
verify_qti(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)

## S4 method for signature 'AssessmentItem'
verify_qti(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)

## S4 method for signature 'AssessmentTest'
verify_qti(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)
```

Arguments

doc	A QTI document (file path, character string, xml_document, or S4 object)
extended_schema	Logical. Use extended rqti schema?

ctx	Integer. Context characters for error snippets.
color	Logical. Use ANSI colors?
engine	Character. Validation backend ("auto", "xml2", "xmllint").
ignore_import	Logical. Ignore import warnings?
print	Logical. Print results?

Value

A qti_validation_result or qti_validation_results_list object.

Functions

- verify_qti(character): Validate character input (file path or XML string)
- verify_qti(xml_document): Validate xml_document objects
- verify_qti(AssessmentItem): Validate assessmentItem objects
- verify_qti(AssessmentTest): Validate assessmentTest objects

---

verify_qti_impl *Validate an XML document against the QTI schema*

Description

Validates an XML document against the QTI schema or the extended rqti QTI schema. Input can be either a file path, a character string containing XML, or an xml2::xml_document.

Usage

```
verify_qti_impl(
  doc,
  extended_schema = FALSE,
  ctx = 40,
  color = TRUE,
  engine = c("auto", "xml2", "xmllint"),
  ignore_import = TRUE,
  print = TRUE
)
```

Arguments

doc	A QTI XML document. This can be a file path, a character string containing XML, or an xml2::xml_document.
extended_schema	Logical. Should the extended rqti schema be used? Defaults to FALSE.
ctx	Integer. Number of characters of context shown before and after the offending XML element in printed snippets. Defaults to 40.

color	Logical. Should ANSI colors be used in printed output? Defaults to TRUE.
engine	Character string specifying the validation backend. One of "auto", "xml2", or "xmllint". Defaults to "auto".
ignore_import	Logical. If TRUE, warnings related to <import> statements in the schema are ignored. Default is TRUE because rqti uses a locally saved schema instead of downloading from the internet.
print	Logical. Should the validation result be printed before it is returned? Defaults to TRUE.

Details

By default, the function chooses a validation backend automatically. If xmllint is available, the input is a file path, and the platform is not Windows, the xmllint backend is used. Otherwise, validation falls back to xml2.

The function returns an object of class "qti_validation_result". By default, that object is also printed in a human-readable form.

Value

An object of class "qti_validation_result" with components:

valid Logical scalar.

errors A list of parsed validation errors.

engine The backend used for validation.

Examples

```
## Not run:
f <- system.file("exercises", "sc1d.xml", package = "rqti")

res <- verify_qti(f)
res$valid

x <- xml2::read_xml(f)
res2 <- verify_qti(x)

summary(res2)

## End(Not run)
```

wrongFeedback *Create object WrongFeedback*

Description

Create object WrongFeedback

Usage

```
wrongFeedback(content = list(), title = character(0), show = TRUE)
```

Arguments

content	A character string or a list of character strings to form the text of the question, which may include HTML tags.
title	A character value, optional, representing the title of the feedback window.
show	A boolean value, optional, determining whether to show (TRUE) or hide (FALSE) the feedback. Default is TRUE.

Value

An object of class WrongFeedback

Examples

```
wfb <- wrongFeedback(content = "Some comments", title = "Feedback")
```

WrongFeedback-class *Class "WrongFeedback"*

Description

Class WrongFeedback is responsible for delivering feedback messages to the candidate in case of an incorrect answer on the entire exercise.

Slots

outcome_identifier A character representing the unique identifier of the outcome declaration variable that relates to feedback. Default is "FEEDBACKMODAL".

show A boolean value, optional, determining whether to show (TRUE) or hide (FALSE) the modal feedback. Default is TRUE.

title A character value, optional, representing the title of the modal feedback window.

content A list of character content to form the text of the modal feedback, which can include HTML tags.

identifier A character value representing the identifier of the modal feedback item. Default is "incorrect".

Examples

```
wfb <- new("WrongFeedback",
           title = "Wrong answer",
           content = list("<b>Some demonstration</b>"))
```

Index