

Package: c403 (via r-universe)

September 30, 2026

Version 0.9-7

Date 2026-07-20

Title Exam Tools for Department of Statistics (c403), Uni Innsbruck

Description Support tools for managing lectures and exams at Uni Innsbruck, specifically for automatic generation and evaluation of mathematics and statistics exams.

Depends R (>= 4.1.0), exams (>= 2.3-1)

Suggests base64enc, openxlsx, psychotools, tth, xlsx, XML, knitr, rmarkdown, tinytex, tidy, dplyr, randomNames, digest

Imports graphics, grDevices, stats, tools, utils, xml2

SystemRequirements pandoc (>= 2.0)

License GPL-2 | GPL-3

VignetteBuilder knitr

URL <https://codeberg.org/zeileis/c403/>

BugReports <https://codeberg.org/zeileis/c403/issues>

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs cmake make libuv1-dev libxml2-dev

Repository <https://zeileis.r-universe.dev>

Date/Publication 2026-07-20 10:30:32 UTC

RemoteUrl <https://codeberg.org/zeileis/c403>

RemoteRef HEAD

RemoteSha a4f9a45f6c0e1bb82be9826893369b8744b01512

Contents

exams2nops	2
exams2olat	5
exams2openolat	6
nops_eval	8
nops_feedback	10
nops_itemresp	11
olat_course_results	12
olat_eval	13
olat_eval_adjust_lang	14
olat_eval_export	15
olat_eval_guess_lang	16
olat_eval_lang	16
olat_exercise	17
olat_extract_get_test_responses_and_text	18
olat_extract_get_user_answers_and_scores	19
olat_extract_html_results	19
olat_feedback	23
olat_results_pseudonymize_data	24
olat_test_results	24
olat_test_results_get_attempts	30
olat_test_results_get_userdata	30
olat_test_results_guess_language	31
olat_test_results_of_attempt	32
olat_test_results_read_manifest	33
read_olat_results	33
read_vis	34
uibkmark	35
vis_groups	36
Index	37

exams2nops

Deprecated: Generation of NOPS Exams (Uni Innsbruck)

Description

Unexported legacy interface to [exams2nops](#) with different default values as used at Uni Innsbruck. Instead it is recommended to use [exams2nops](#) directly.

Usage

```
exams2nops(
  file,
  n = 1L,
  dir = NULL,
  name = NULL,
```

```

    language = "de",
    title = "Klausur",
    course = "",
    institution = "Universit\\\\"at Innsbruck",
    logo = "uibk-logo-bw.png",
    date = Sys.Date(),
    replacement = TRUE,
    intro = NULL,
    blank = NULL,
    duplex = TRUE,
    pages = NULL,
    usepackage = NULL,
    encoding = "",
    startid = 1L,
    points = NULL,
    showpoints = FALSE,
    reglength = 8L,
    ...
)

```

Arguments

file	character. A specification of a (list of) exercise files.
n	integer. The number of copies to be compiled from file.
dir	character. The default is either display on the screen or the current working directory.
name	character. A name prefix for resulting exercises and RDS file.
language	character. Path to a DCF file with a language specification. Currently, "en" and "de" are shipped with the exams package.
title	character. Title of the exam, e.g., "Statistische Datenanalyse".
course	character. Course number, e.g., "403001".
institution	character. Name of the institution at which the exam is conducted.
logo	character. Path to a logo image. If the logo is not found, it is simply omitted.
date	character or "Date" object specifying the date of the exam.
replacement	logical. Should a replacement exam sheet be included?
intro	character with LaTeX code for introduction text at the beginning of the exam.
blank	integer. Number of blank pages to be added at the end. (Default is chosen to be half of the number of exercises.)
duplex	logical. Should blank pages be added after the title page (for duplex printing)?
pages	character. Path(s) to additional PDF pages to be included at the end of the exam.
usepackage	character. Names of additional LaTeX packages to be included.
encoding	character, passed to Sweave .
startid	integer. Starting ID for the exam numbers (defaults to 1).

points	integer. How many points should be assigned to each exercise? Note that this argument overrules any exercise points that are provided within the expoints tags of the exercise files (if any). The vector of points supplied should either have length 1 or the number of exercises in the exam.
showpoints	logical. Should the PDF show the number of points associated with each exercise (if specified in the Rnw/Rmd exercise or in points)?
reglength	integer. Number of digits in the registration ID. The default is 8 and it can be increased up to 10.
...	arguments passed on to exams2pdf .

Details

exams2nops is a convenience interface for [exams2nops](#) with somewhat different defaults: German titles/descriptions, Uni Innsbruck logo, replacement sheets enabled, duplex printing enabled.

Value

A list of exams as generated by [xexams](#) is returned invisibly.

See Also

nops_eval nops_register

Examples

```
## load package and enforce par(ask = FALSE)
library("exams")
options(device.ask.default = FALSE)

## define an exams (= list of exercises)
myexam <- list(
  "boxplots",
  c("tstat", "ttest", "confint"),
  c("regression", "anova"),
  c("scatterplot", "boxhist"),
  "relfreq"
)

if(interactive()) {
  ## compile a single random exam (displayed on screen)
  exams2nops(c("tstat2", "anova", "boxplots"))
}
```

Description

Old legacy interface for producing QTI 1.2 (rather than QTI 2.1) exams for OpenOlat at Uni Innsbruck. By now superseded by [exams2openolat](#).

Usage

```
exams2olat(
  file,
  n = 1L,
  dir = ".",
  name = "olattest",
  maxattempts = 1L,
  cutvalue = 1000,
  solutionswitch = FALSE,
  stitle = "Aufgabe",
  ititle = "Frage",
  adescription = "Bitte bearbeiten Sie folgende Aufgaben.",
  sdescription = "Bitte beantworten Sie folgende Frage.",
  eval = list(partial = FALSE, negative = FALSE),
  ...
)
```

Arguments

<code>file</code>	list, of md/Rmd files to be used
<code>n</code>	integer, number of randomized tests to be created (default 1L)
<code>dir</code>	character, where to store the resulting file(s) (default .)
<code>name</code>	character name of the test/quiz
<code>maxattempts</code>	integer, the maximum attempts for one question (must be smaller than 100000L).
<code>cutvalue</code>	numeric, the cutvalue at which the exam is passed
<code>solutionswitch</code>	logical Should the question/item solutionswitch be enabled? In OLAT this means that the correct solution is shown after an incorrect solution was entered by an examinee (i.e., this is typically only useful if <code>maxattempts</code> = 1).
<code>stitle</code>	character A title that should be used for the sections. May be a vector of length 1 to use the same title for each section, or a vector containing different section titles.
<code>ititle</code>	character A title that should be used for the assessment items. May be a vector of length 1 to use the same title for each item, or a vector containing different item titles. Note that the maximum of different item titles is the number of sections/questions that are used for the exam.

adescription	character Description (of length 1) for the overall assessment (i.e., exam).
sdescription	character Vector of descriptions for each section.
eval	named list, specifies the settings for the evaluation policy, see function exams_eval
...	forwarded to exams2qti12

Details

exams2olat is the old convenience interface to produce QTI 1.2 tests/exams for OpenOlat. It has been superseded by [exams2openolat](#) which offers more options and flexibility.

exams2openolat	<i>Deprecated: Generation of Exams for OpenOlat (Uni Innsbruck)</i>
----------------	---

Description

Unexported legacy interface to [exams2openolat](#) with slightly different default values as used at the Department of Statistics, Uni Innsbruck. Instead it is recommended to use [exams2openolat](#) directly.

Usage

```
exams2openolat(
  file,
  n = 1L,
  dir = ".",
  name = "olatatest",
  maxattempts = 1,
  cutvalue = 1000,
  solutionswitch = FALSE,
  qti = "2.1",
  stitle = "Aufgabe",
  ititle = "Frage",
  adescription = "",
  sdescription = "",
  eval = list(partial = FALSE, negative = FALSE),
  template = "qti21",
  ...
)
```

Arguments

file	character. A specification of a (list of) exercise files.
n	integer. The number of copies to be compiled from file.
dir	character. The default is either display on the screen or the current working directory.

name	character. A name prefix for resulting ZIP and RDS file.
maxattempts	integer. The maximum attempts for one question, may also be set to Inf.
cutvalue	numeric. The cutvalue at which the exam is passed.
solutionswitch	logical. Should the question/item solutionswitch be enabled?
qti	character indicating whether QTI "1.2" or "2.1" (default) should be generated.
stitle, ititle, adescription, sdescription	character. Descriptions for various titles/descriptions. Defaults to generic German titles.
eval	named list. Specifies the settings for the evaluation policy, see function exams_eval . The default is not to partial credits for multiple-choice exercises.
template	character. The IMS QTI 2.1 template that should be used. In addition to the default template this can be set to "qti21test" which uses almost the same template but with a PASS variable. Thus, no cutvalue is needed and the online tests just generate a SCORE.
...	arguments passed on to exams2openolat .

Details

`exams2openolat` is a convenience interface for [exams2openolat](#) with somewhat different defaults: German titles/descriptions, partial credits disabled, solution switch turned off, large cut value (so that the test cannot be passed), and with fancy quotes turned off in verbatim R output. Finally, an RDS file is stored as a by-product containing the [xexams](#) list. This enables extracting and displaying specific exercises from an online test in R.

Value

A list of exams as generated by [xexams](#) is returned invisibly.

See Also

`olat_eval` `olat_exercise`

Examples

```
## load package and enforce par(ask = FALSE)
library("exams")
options(device.ask.default = FALSE)

## define an exams (= list of exercises)
myexam <- list(
  "boxplots",
  c("tstat", "ttest", "confint"),
  c("regression", "anova"),
  c("scatterplot", "boxhist"),
  "relfreq"
)

## output directory
```

```

mydir <- tempdir()

## generate .zip with German OpenOlat exam in temporary directory
## using a few customization options
exams2openolat(myexam, n = 3, dir = mydir, maxattempts = 2)
dir(mydir)

```

nops_eval

Deprecated: Evaluate NOPS Exams (Uni Innsbruck)

Description

Unexported legacy interface to evaluate NOPS exams produced with [exams2nops](#), and scanned by [nops_scan](#). Instead it is recommended to use [nops_scan](#) directly.

Usage

```

nops_eval(
  register = Sys.glob("*.csv"),
  solutions = Sys.glob("*.rds"),
  scans = Sys.glob("nops_scan_*.zip"),
  points = NULL,
  eval = exams_eval(partial = FALSE, negative = 0.25),
  mark = c(0.5, 0.6, 0.75, 0.85),
  dir = ".",
  results = "nops_eval",
  html = NULL,
  col = hcl(c(0, 0, 60, 120), c(70, 0, 70, 70), 90),
  language = "de",
  module = NULL,
  interactive = TRUE,
  string_scans = Sys.glob("nops_string_scan_*.zip"),
  string_points = seq(0, 1, 0.25)
)

nops_eval_write_uibk(
  results = "nops_eval.csv",
  file = "exam_eval",
  dir = ".",
  language = "en",
  ...
)

nops_register(
  file = Sys.glob("*.xls*"),
  startid = 1L,

```

```

    tab = !identical(startid, FALSE),
    pdf = !identical(startid, FALSE),
    split = NULL,
    info = NULL,
    verbose = TRUE,
    ...
)

```

Arguments

register	character. File name of a CSV file (semicolon-separated) of the registered students, e.g., as produced by <code>nops_register</code> based on the VIS registration lists. Must contain columns "Matrikelnr", "Name", "Account" (and "LV", in case module marks should be computed). The file name should not contain spaces, umlaut or other special characters (something like "GP-2014-02.csv" is recommended).
solutions	character. File name of the RDS exercise file produced by exams2nops .
scans	character. File name of the ZIP file with scanning results (containing Daten.txt and PNG files) as produced by nops_scan (or the FSS).
points	numeric. Vector of points per exercise. By default read from solutions.
eval	list specification of evaluation policy as computed by exams_eval .
mark	logical or numeric. If mark = FALSE, no marks are computed. Otherwise mark needs to be a numeric vector with four threshold values to compute marks from 5 to 1. The thresholds can either be relative (all lower than 1) or absolute. In case results exactly matching a threshold, the better mark is used.
dir	character. File path to the output directory (the default being the current working directory).
results	character. Prefix for output files.
html	character. File name for individual HTML files, by default the same as register with suffix .html.
col	character. Hex color codes used for exercises with negative, neutral, positive, full solution.
language	character. Path to a DCF file with a language specification. Currently, "en" and "de" are shipped with the package.
module	logical or numeric. Should module marks (in addition to the exam marks) be computed? If this is numeric, this can be a vector of two ECTS weights for the written exam and seminar, respectively (by default equal weights of 0.5 and 0.5 are used). If module is not FALSE, register needs to contain a column "LV" with the seminar marks.
interactive	logical. Should possible errors in the Daten.txt file be corrected interactively? Requires the png package for full interactivity.
string_scans	character. Optional file name of the ZIP file with scanning results of string exercise sheets (if any) containing Daten2.txt and PNG files as produced by nops_scan .
string_points	numeric. Vector of length 5 with points assigned to string results.

file	character. Name of the VIS file with the registration list.
...	further arguments passed to read_vis (e.g., subset).
startid	integer or logical, default 1L. If FALSE no random seats are generated.
tab	logical. Should a tab-separated file with the seat information be generated for OpenOlat? Defaults to TRUE if random seats are generated.
pdf	logical. Should PDF files with participant lists be generated for printing? Defaults to TRUE if random seats are generated.
split	integer. Number of participant lists ordered by seat.
info	character. Vector of length 4 with information about the exam: (1) type of exam (GP, LVP, VO, ...), (2) title of exam, (3) date/time (YYYY-MM-DD HH:MM), (4) location/room. By default the information is inferred from the VIS file.
verbose	logical. Should information about the registrations be printed to the screen?

Details

`nops_eval` is a companion function for [exams2nops](#) and [nops_scan](#). It calls [nops_eval](#) from the **exams** package which evaluates the scanned exams by computing the sums of the points achieved and (if desired) mapping them to marks (and to module marks). Furthermore HTML reports for each individual student are generated for upload into OpenOlat. In addition to this function from the **exams** package, the function adds the marks in the Uni Innsbruck-specific format in an Excel spreadsheet.

`nops_register` is another companion function for preprocessing the registration lists that are provided by VIS. The function assigns random seats for every student and saves the result in both CSV and XLSX format as well as a tab-separated text file with the seat numbers for import into OLAT. The underlying workhorse function is [read_vis](#).

Value

A data frame with the detailed exam results is returned invisibly. It is also written to a CSV file in the current directory, along with a ZIP file containing the HTML reports (for upload into OLAT), and an XLSX file (for importing the marks into VIS).

See Also

[exams2nops](#), [nops_scan](#), [nops_eval](#), [read_vis](#)

nops_feedback

Generate nops test feedback files (answer/solution)

Description

Similar to what [olat_eval](#) does but in a more detailed way. Creates html files for each participant with feedback about his/her test results (including questions and solutions).

Usage

```
nops_feedback(res, xexam, name = "nops_feedback")
```

Arguments

res	data.frame, content of the csv file created by nops_eval .
xexam	list as returned from reading the rds file
name	character, name of the test, will be used to name the zip archive file and the html files

Value

Returns the name of the zip file created.

Author(s)

Reto Stauffer

nops_itemresp	<i>Extract Item-Response Data from NOPS Exams</i>
---------------	---

Description

Process data from NOPS evaluation results (via [nops_eval](#)) for subsequent IRT (item response theory) modeling.

Usage

```
nops_itemresp(
  eval = "nops_eval.csv",
  exam = Sys.glob("*.rds"),
  psychotools = NULL,
  labels = NULL,
  ...
)
```

Arguments

eval	character. File name of CSV output from nops_eval .
exam	character. File name of RDS output from exams2nops.
psychotools	logical. Should itemresp from psychotools be used as the class for item response columns? By default, this is used if the psychotools package is available. If FALSE, matrices with dummy codings are used.
labels	function for extracting exercise labels from each \$metainfo. By default the original file name \$metainfo\$file is used.
...	additional arguments (such as colClasses) passed to read.csv2(eval, dec = ".", ...).

Details

`nops_itemresp` returns a data frame with several item response outcomes for each student: `solved` indicates whether or not an exercise was fully solved, `partial` whether or not it was at least partially solved. `points` gives the points achieved for each exercise. The corresponding `nsolved`, `npartial`, and `npoints` are the sums of these for each student. Moreover, `solved2`, `partial2`, and `points2` distinguish not only the exercises within the exam but also the actual source template within each exercise.

Value

A `data.frame`.

See Also

`nops_eval`

<code>olat_course_results</code>	<i>Extracting Olat Course Results</i>
----------------------------------	---------------------------------------

Description

Loading and preparing 'course results' (user details and score) from the `course_results.xlsx` files shipped with the ZIP file extracted via OLAT. For a single test element, or a (partial) archive of test elements extracted. In contrast to `olat_test_results()` this function solely relies on the XLSX file (final points).

Usage

```
olat_course_results(zipfile, verbose = FALSE)
```

Arguments

<code>zipfile</code>	character vector with one or multiple ZIP file names (archives exported via Open Olat). Can be a named vector for a 'wide' format (see details).
<code>verbose</code>	logical, if TRUE some information is written to stdout.

Details

A data frame with the course results is returned, including username, name, institution, as well as the scores of the different elements and where they come from.

If `zipfile` is unnamed, the variable `element` describes where the scores (`score`) comes from. If `zipfile` is a named character vector, a wide format is returned where the scores are stored according to the name of the character vector `zipfile`.

Author(s)

Reto

Examples

```
## Not run:
## Pseudonymization
library("dotenv")
load_dot_env(".env_demo")

ptzip <- c("PT01" = "praesenztest_01.zip",
          "PT02" = "praesenztest_02.zip",
          "PT03" = "praesenztest_03.zip",
          "PT04" = "praesenztest_04.zip",
          "PT05" = "praesenztest_05.zip",
          "PT06" = "praesenztest_06.zip")
otzip <- c("onlinetests_1-11.zip", "onlinetests_12-14_temporary.zip")

## Extracting data
p1 <- olat_course_results(ptzip)
p2 <- olat_course_results(unname(ptzip))
ot <- olat_course_results(otzip)

## Head
head(p1, n = 2L)
head(p2, n = 2L)
head(ot, n = 2L)

## End(Not run)
```

olat_eval

Evaluate OLAT Exams

Description

Evaluate OLAT exams produced with [exams2olat](#), and carried out and exported in (Open)OLAT.

Usage

```
olat_eval(file, plot = TRUE, export = FALSE)
```

Arguments

file	character. Base file name of RDS and XLS file with exam generated by exams2olat and exported OLAT results, respectively.
plot	logical. Should barplots with the aggregated results be displayed on the screen?
export	logical. Should detailed questions along with individual results be exported to HTML files in a ZIP archive for convenient import into OLAT?

Details

olat_eval is a companion function for [exams2olat](#). It evaluates the exams carried out in OLAT for further processing outside of OLAT (in CSV format) and optionally exports detailed individual HTML reports (in a ZIP archive) for reimport into OLAT.

Value

A data.frame with the detailed exam results is returned invisibly. It is also written to a CSV file in the current directory, along with a ZIP file containing the HTML reports (for upload into OLAT).

See Also

[exams2olat](#)

olat_eval_adjust_lang *Adjust language of variables/columns*

Description

Modifies the names of the variables in the data.frame as read from the xlsx file (OpenOLAT). Converts the variable names to English such that we do no longer have to care about language in all other methods/functions.

Usage

```
olat_eval_adjust_lang(x)
```

Arguments

x data.frame read from the xlsx file

Details

Input 'x' is the data.frame as read from the xlsx file which contains user meta information and the detailed information about the individual questions of the test. Problem: depending on the user language settings of OLAT the names and order of the columns differs. This function takes input 'x' and manipulates the variable or column names in a way that the rest of the code ([olat_eval](#)) is not language dependent anymore.

- tries to guess the language by calling [olat_eval_guess_lang](#)
- loads the search-replace-data.frame (internally)
- search and replace variable names
- return input object x with new variable names

Note: Even English to English will rename some of the variables.

The function uses the data set olat_eval_lang which is shipped with the package (see data("olat_eval_lang")).

Value

Returns the same `data.frame` (same dimension and data) with adjusted names.

Author(s)

Reto

olat_eval_export	<i>Olat eval export</i>
------------------	-------------------------

Description

Takes the results from `read_olat_results` and the information from the `rds` file with the individual questions/answers to generate a zip archive file with individual test results (html file). This zip file can be used to upload to OLAT.

Usage

```
olat_eval_export(
  results,
  xexam,
  file = "olat_eval.zip",
  html = "Testergebnisse.html",
  col = hcl(c(0, 0, 60, 120), c(70, 0, 70, 70), 90)
)
```

Arguments

<code>results</code>	<code>data.frame</code> , results from <code>read_olat_results</code>
<code>xexam</code>	list the object loaded from the <code>rds</code> file which contains the individual questions/answers. The length of the list corresponds to the number of randomized tests, each list element contains <code>N</code> elements (<code>N</code> = number of questions) with all the information required to generate the output.
<code>file</code>	character, name of the zip file, the final archive file where to store the exported html files
<code>html</code>	character, name of the output files (html files)
<code>col</code>	character vector of length 4L with hex colors, default is <code>hcl(c(0, 0, 60, 120), c(70, 0, 70, 70), 90)</code>

Value

Return of the `zip()` call.

olat_eval_guess_lang	<i>Guess language of the OLAT results file</i>
----------------------	--

Description

Given a set of character vectors this function tries to guess the language of the imported file. This allows to evaluate exams from OpenOLAT in different languages. Used by [olat_eval](#) to rename the columns to make the evaluation independent from the language used when exporting the results via OLAT.

Usage

```
olat_eval_guess_lang(x)
```

Arguments

x	character vector with column names
---	------------------------------------

Value

Returns the language name ("en", "de") if the function is able to guess the language. Else the script will stop.

Author(s)

Reto

olat_eval_lang	<i>Olat evaluation language data set</i>
----------------	--

Description

Depending on the user profile language settings OLAT exports test results (xlsx file) in different languages. To make things easier in the **c403** package the variable names are modified using this language search-and-replace data set.

Usage

```
olat_eval_lang
```

Format

A data frame with a language flag (given your export was in German, only rows with lang == "de" will be considered). The two columns search and replace are used for `gsub()` (regular expression string replacement).

Author(s)

Reto Stauffer

olat_exercise

*Extract (and Display) Exercises from OpenOlat Exams***Description**

Extract (and display) selected exercises from OpenOlat exams produced with [exams2openolat](#) in order to see both question and solution.

Usage

```
olat_exercise(x, ..., fixed = TRUE, show = TRUE, mathjax = TRUE)
```

Arguments

x	character or list. Either an OpenOlat exam list as produced by exams2openolat or a character with the file path to an .rds file containing such an exam.
...	character. Either a single numeric index of the exam to be selected. Or, alternatively, patterns to be searched for in the question text of the exams in x.
fixed	logical. Should the search pattern(s) be matched as is?
show	logical. Should the exercise(s) found be shown in the browser?
mathjax	logical. Should the JavaScript from http://www.MathJax.org/ be included for rendering mathematical formulas?

Details

olat_exercise is a companion function for [exams2openolat](#). As OpenOlat has no option during an exam to look at the precise question of a particular student – and more importantly the corresponding solution – one strategy is to search for particular words, numbers, or other strings in the database of all questions from an OpenOlat exam.

olat_exercise goes through all questions in the exam and selects those question(s) that match(es) the given search patterns. By default the question(s)/solution(s) are displayed in the browser and returned invisibly.

Value

A list containing either a single exercise or a list of such exercises (in case the search patterns do not yield a unique question).

See Also

[exams2openolat](#)

`olat_extract_get_test_responses_and_text`*Extracting Exercise Text, Choice Options, and user Responses*

Description

Given single or multiple choice questions are present, the details are required for mapping the candidate answer as well as later on calculate the 'order' (i.e., binary '00101' solution/answer strings).

Usage

```
olat_extract_get_test_responses_and_text(  
  d,  
  identifier,  
  verbose = FALSE,  
  envir = NULL  
)
```

Arguments

<code>d</code>	string, name of the directory where the Open Olat ZIP file was extracted. Contains the XML files required.
<code>identifier</code>	string, identifier of the question.
<code>verbose</code>	logical, if TRUE some information is written to stdout.
<code>envir</code>	either NULL or an R environment. Used when <code>cache_xml = TRUE</code> when calling <code>olat_test_results()</code> .

Details

Given the `identifier` this function is looking for a file called `"*/test*/<identifier>.xml"` containing the potential is returned. Attached to each choice is the text of the exercise, in case of single-choice, multiple-choice, or cloze questions the same text is duplicated in multiple rows. These duplicates have to be carried along and are later removed when calculating the summaries.

This has to be done for each test question. If one has a small number of randomizations and/or a large number of participants, multiple users may have gotten the same question. In this case the same XML file would need to be parsed multiple times. To avoid this, we store the extracted choice items in a custom environment if `envir` is set and re-use it if the same id is seen a second time.

Value

Either NULL if no choice questions are present, or a `olat_test_responses` data frame containing question/answer identifiers, binary flag whether or not the answer is a correct answer, as well as its original content (i.e., the one shown to the users).

olat_extract_get_user_answers_and_scores

Extracting User Answers and Scores

Description

Each time a user takes an attempt on a test, a XML file is placed in `.../userdata/<username>/<attempt_X>` folder which contains the candidate (users) answer as well as the score given by Open Olat. For single choice and multiple choice questions, the candidates answer is a choice identifier, translated to a readable value via the choices object.

Usage

```
olat_extract_get_user_answers_and_scores(
  node,
  choices,
  tz = "",
  verbose = FALSE
)
```

Arguments

node	an xml_nodeset, specifically the <code><itemResults></code> node.
choices	object of class <code>olat_test_responses</code> as returned by olat_extract_get_test_responses_and_text()
tz	time zone used for date/time conversions. Defaults to <code>""</code> (uses local/system time zone).
verbose	logical, if TRUE some information is written to stdout.

Value

TODO(R)

Author(s)

Reto

olat_extract_html_results

Extracting OpenOLAT Results from HTML Results Files

Description

To get detailed information from the individual questions of a test, this function processes the HTML results files (test summaries) created by OpenOLAT, and (if required) combines this the data with the information provided by [exams2openolat](#) to create the link to the question source files (i.e., R/exams Rmd/Rnw questions). Is able to process results exported via Open Olat (zipfile) in both German or English for now.

Usage

```
olat_extract_html_results(rds, zipfile = NULL, verbose = TRUE)
```

Arguments

rds	NULL or character. If character, name of the RDS file procuded by exams2openolat when the test was generated used to extract original exam question file name and 'exname'. If NULL only the information from the zipfile is used.
zipfile	character, name/path to the ZIP file (exported via OpenOlat) containing the HTML results files (amongst other files needed).
verbose	logical, if TRUE some information is written to stdout.

Details

A brief summary of the individual steps:

1. Reading and flattening the rds file (source file name, and question name (exname)).
2. Unzips the zipfile, tries to guess the language of the content in the ZIP file, and checks if the content looks as expected.
3. Searching for the manifest (imsmanifest.xml) containing the OpenOLAT IDs; combine with the information from step (1) to create the link between source files and OpenOLAT IDs.
4. Identifying all HTML results files (HTML test summary) containing the required detailed information.
5. Parsing all HTML files, extracting user information and details about each individual question (OpenOLAT ID, Score, user Answer, correct Solution). This is combined with the information from step (2).

After that a few small modifications are made on the resulting data.frame before it is returned.

Value

Returns a (tibble) data frame of dimension $N \times K$ where N corresponds to the total number of questions. If 100 participants have taken a test with 20 questions each, this would result in $N = 2000$. The columns contain:

- Username: Participants user name (c-Kennung).
- Institution: Institution identifier (Matrikelnummer)
- Name: Full name of the participant.
- ID: OpenOLAT question identifier.
- Email: Participant email as shown in the HTML results file.
- Status: Status provided by OpenOLAT.
- Score: The participant's score for this question ("My Score" in the HTML file).
- Answer: The participant's answer, can be NA if empty.
- Solution: The correct solution.
- RawText: Raw question text, unformatted, can be used to search for keywords in the text.

- Section/Item: Section and item ID extracted from the exams olat identifier.

If an additional RDS file (argument `rds`) is provided, the following columns are added:

- file: Source file name (modified), path/name of the original R/exams file (Rmd/Rnw).
- exname: Name of the question (the exname meta-information from the R/exams question).

Besides Score (numeric), Section/Item (int) everything is of class character.

Deprecation Warning

Note that this function should no longer be used and will be removed in the future. Use `olat_test_results()` or `olat_course_results()` instead.

Disclaimer and missing features

TODO: This is a first implementation of this feature and there is a series of todo's and missing features. Incomplete list of known missing features:

- Depending on the (user specific) OpenOLAT language setting, the exported ZIP file comes in different languages. We 'auto guess' the language based on the content of the ZIP archive, tested for German and English.
- Originally implemented for string, num, and schoice questions. Not sure if it works for mchoice, definitively no support for cloze questions at the moment.
- Handling of multiple attempts (evaluate first, second, last attempt?).
- Support for multiple tests (if the "Results" folder contains multiple "test*" directories).
- Variations of questions (limited support; import works but post-processing might get difficult/impossible).

Author(s)

Reto

Examples

```
## Not run:

# 'Minimal' of the analysis performed in December 2024 (for which this
# function was originally written); few ggplot2 plots plus code to prepare
# the data matrix for estimating a Rasch model. Can't be run by end-users as
# we can't provide the data needed to run the example, this serves as a
# template for future developments/ideas.
library("c403")

rds      <- "GP06Dezember2024.rds"
zipfile  <- "results_GP06Dezember2024.zip"
results <- olat_extract_html_results(rds, zipfile)

# -----
# Some plotting
```

```

# -----

library("ggplot2")
library("patchwork")

# Some demo plots; assumes that the students could only
# gain 0 points (incorrect) or 2 points (correct)
results$label <- with(results, paste(file, exname, sep = "\n"))
results$scoreLabel <- factor(results$Score > 0,
                             levels = c(FALSE, TRUE),
                             labels = c("Score == 0 (incorrect)",
                                          "Score > 0 (counted as correct)"))

g1 <- ggplot(results) + geom_bar(aes(y = label, fill = Status), stat = "count") +
  labs(title = "Stacked barplot of answered/unanswered questions") +
  labs(subtitle = paste("Number of participants: ", length(unique(results$User)))) +
  theme_minimal()
g2 <- ggplot(results) + geom_bar(aes(y = label, fill = scoreLabel), stat = "count") +
  labs(title = "Stacked barplot of answered/unanswered questions") +
  labs(subtitle = paste("Number of participants: ", length(unique(results$User)))) +
  theme_minimal()

plot(g1 + g2) # Patchwork plot

# Histogram of Score distribution; here assuming the threshold was 22 points
library("dplyr")
results %>% group_by(Name) %>% summarize(Score = sum(Score)) %>%
  ggplot() + geom_histogram(aes(x = Score), binwidth = 1) +
  geom_vline(xintercept = 22, col = "tomato", lwd = 2) +
  labs(title = "Histogram of total Score") +
  labs(subtitle = paste("Number of participants: ", length(unique(results$User)))) +
  theme_minimal()

# -----
# Estimating Rasch model
# -----

library("tidyr")
library("psychotools")

# Prepare result to binary matrix for Rasch model
tmp <- transform(subset(results, select = c(Name, file, Score)), Score = as.integer(Score > 0)) |>
  pivot_wider(names_from = file, values_from = Score) |>
  as.data.frame()
tmp <- as.matrix(tmp[, !grepl("^Name$", names(tmp))])

# Estimate Rasch model
mr <- raschmodel(tmp)

# Default plots
hold <- par(no.readonly = TRUE)
par(mar = c(20, 12, 2.5, 1))
plot(mr, type = "profile")

```

```
par(hold)
plot(mr, type = "piplot")

## End(Not run)
```

olat_feedback	<i>Generate OLAT test feedback files (answer/solution)</i>
---------------	--

Description

Similar to what [olat_eval](#) does but in a more detailed way. Creates html files for each participant with feedback about his/her test results (including questions and solutions). For OLAT tests use [olat_feedback](#), for nops tests (written tests) use [nops_feedback](#).

Usage

```
olat_feedback(res, xexam, name = "olat_feedback")

olat_feedback_render_one(res, xexam, i, htmlfile = "Result.html", show = FALSE)
```

Arguments

res	data.frame, result from olat_eval
xexam	list as returned from reading the rds file
name	character, name of the test, will be used to name the zip archive file and the html files
i	integer, row index (which row in 'x' to render)
htmlfile	character, name of the output file
show	logical, if set to TRUE the html file will be opened in a browser

Value

Returns the name of the zip file created.

Author(s)

Reto Stauffer

olat_results_pseudonymize_data
Pseudonymizing User Data

Description

When a dotenv file was loaded previous to calling the `olat_test_results()` or `olat_course_results()` function, all user related details will be pseudonymized. This includes the institution, username, and name of the participant.

Usage

```
olat_results_pseudonymize_data(x)
```

Arguments

x a data frame.

Details

Expects a series of environment variables to be present as loaded from the dotenv file. If pseudonymization is requested, this function looks for variables in the data frame x to be pseudonymized and replaces them.

Value

If no pseudonymization takes place, the return is an unmodified copy of x. Else columns with user specific information will be modified and a modified version of x is returned with re-ordered rows (observations).

Author(s)

Reto

olat_test_results *Extracting Detailed Open Olat Test Results*

Description

To get detailed information from the individual questions of Open Olat test elements, this function processes the content of a ZIP file downloaded via Open Olat (via export all results or the archiving function) and, if requested, combines this data with the information with the details in the RDS file provided by `exams2openolat()`. Should work for both, results exported in German or English.

Usage

```

olat_test_results(
  zipfile,
  rds = NULL,
  attempt = c("last", "best", "first", "all"),
  tz = "",
  verbose = FALSE,
  cache_xml = FALSE,
  cores = NULL,
  ...
)

## S3 method for class 'olat_test_details'
format(x, ...)

## S3 method for class 'olat_test_result'
summary(
  object,
  level = c("user", "task", "item"),
  type = c("wide", "long"),
  ...
)

## S3 method for class 'olat_test_results'
summary(
  object,
  level = c("user", "task", "item"),
  type = c("wide", "long"),
  ...
)

```

Arguments

zipfile	character, name/path to the ZIP file (exported via OpenOlat). See Section 'ZIP File Content' for more details.
rds	NULL or character vector. If character, name(s) of the RDS file(s) procuded by exams2openolat() when the test was generated, used to extract original exam question file name and 'exname' to enrich the return. If the ZIP file (see Section 'ZIP File Content') contains a single element, rds must be of length one. Else rds must be a named character vector where the names match directory names the individual tests in the ZIP file.
attempt	character, defaults to "last". Can be changed to "first" (results of first attempt), "best" (best attempt), or "all" (return results of all attempts).
tz	time zone used for date/time conversions. Defaults to "" (uses local/system time zone).
verbose	logical, if TRUE some information is written to stdout.

cache_xml	logical, defaults to FALSE. If set TRUE a custom environment is used to 'cache' the content of the test-related XML (see olat_extract_get_test_responses_and_text()) and is re-used if the same randomization is used multiple times. Can speed up things if the number of randomizations is lower than the number of total attempts.
cores	number of cores to be used. Either NULL or a positive integer. If NULL, all detected cores but two will be used.
...	currently unused.
x	object of class <code>olat_test_details</code>
object	an object of class <code>olat_test_results</code> .
level	aggregation level to be used, defaults to "user".
type	character, type of summary to be returned, defaults to "wide".

Details

This function replaces the old function [olat_extract_html_results\(\)](#) which was designed with a similar purpose but limited features and was using the HTML files rather than the XML files.

A brief summary of the individual steps:

Value

Returns an object of class `c("olat_test_result", "data.frame")` if the ZIP file contains results of one single Open Olat element, or an object of class `olat_test_results` (plural; named list) containing a series of `c("olat_test_result", "data.frame")` objects if the ZIP contains results from multiple elements. Each `olat_test_result` element can contain variable amount of variables including user and task details.

- `xmlfile`: The main XML file the test results are based on
- `username`: Open Olat account name or user name
- `institution`: Institution identifier (Matrikelnummer)
- `name`: Full name of the participant
- `attempt`: Integer, attempt number (see argument `attempt`)
- `total_score`: Overall test score
- `max_score`: Maximum test score possible
- `task_k`: One or multiple columns named `task_1`, `task_2`, ..., `task_K`. These are objects of class `olat_test_details` containing data frames with all the test details of a task (see Section 'Olat Test Details').

If the result is a list, the names correspond to the name of the Open Olat element, or to be more precise, the name of the folder in the ZIP file containing the elements results.

ZIP File Content

Open Olat allows to extract detailed test results in two different places. For a single Open Olat Element (i.e., a single test) this can be done via the `assessment > element > export results`. In this case the ZIP file will contain test results of the selected element in the root node of the ZIP file. If so, this function returns a single `olat_test_result` object.

Alternatively one can export test results of one or multiple Open Olat elements (i.e., multiple tests) via the archiving function. In this case the ZIP file exported contains a series of folders, one folder per Open Olat element, each then containing the corresponding test details extracted by this function. In this case, this function will return an object of class `olat_test_results` (named list) containing a series of `olat_test_result` objects.

Olat Test Details

Each task column in the object returned contains an object of class `olat_test_details` (a named list with data frames) containing all the extracted information. Each row of these data frames corresponds to a question, thus cloze questions have multiple rows. Each data frame contains the following variables:

- `question_number`: Question number, order of how they were shown in the task (integer)
- `datetime`: Date and time when the participants solved the task (POSIXct)
- `duration`: Duration in minutes the participants worked on the task (numeric)
- `max_score`: Maximum score possible (numeric)
- `type`: Type of question (num, schoice, ...; character)
- `choice_choices`: NA if the question was no single choice or multiple choice question, else a character with all the options shown in the order as displayed in the task (character). The individual options/choices are separated by line breaks (`\\n`; character)
- `choice_solution`: NA if no single choice or multiple choice question, else a character of the form "01001" where "1" indicate correct choices (character)
- `solution`: The correct solution (character, for all question types)
- `choice_answer`: NA if no single choice or multiple choice question, else a character of the form "01001" where "1" indicates the answer given by the candidate (character)
- `answer`: Answer given by the candidate (character, for all question types)
- `score`: Score given to the participant (numeric)

If an additional RDS file (argument `rds`) is provided, the following columns are added:

- `exfile`: Source file name (modified), path/name of the original R/exams file (Rmd/Rnw; character)
- `exname`: Name of the question (the exname meta-information from the R/exams question; character)

Adding functionality

TODO: The plan is to add additional functionality to produce output similar to what has been implemented already for the "nops workflow", to allow performing the same checks/analysis no matter where the results come from. See:

- <https://git.uibk.ac.at/econstat/exams/demo/-/tree/main/gp-2024-02>

Author(s)

Reto

Examples

```
## Not run:

## Extracting test details from ZIP
x1 <- olat_test_results("olat_results.zip")

## Providing additional rds file to enrich the return
## (adds name of the R/exams file and the exname).
x2 <- olat_test_results("olat_results.zip", "olat_test.rds",
                        cache = TRUE, verbose = TRUE)

## Archive ZIP (see Section 'ZIP File Content'): When not using the
## additional rds files everything works the same, except the
## return is a named list.
## When providing rds files, they must match the element folder names
## within the ZIP. Imagine the ZIP file is an archive of four tests
## from four groups, where the same test was used twice each, and
## the tests are named "Test_A", "Test_B", "Test_C", and "Test_D",
## the call will look something as follows:
rdsfiles <- c("Test_A" = "onlinetest_version1.rds",
              "Test_B" = "onlinetest_version2.rds",
              "Test_C" = "onlinetest_version3.rds",
              "Test_D" = "onlinetest_version4.rds")
archive_x <- olat_test_results("olat_archive.zip", rds = rdsfiles)

## 'archive_x': List with four elements named after names(rdsfiles),
## each containing an object of class olat_test_result, the same
## object returned when the ZIP file contains results of one single
## element (see above; x1, x2).

# -----
# Pseudonymization: `olat_test_results()` allows to pseudonymize
# user information by providing a dotenv file.
# -----

## Write demo env file (.env_demo)
write("
# Integer, if 0 no pseudonymization will take place (when the dotenv is loaded),
# else user credentials will be pseudonymized. If this dotenv is not loaded at
# all, no alteration will take place anyways.
C403_PSEUDONYMIZE=1

# Used to set a seed
C403_SEED=1

# String to create the pseudonymized hashes for the users
C403_SECRET=demo
", ".env_demo")
```

```

## Loading the dotenv package. Automatically loads a .env file
## if it exists when attached or manually load a specific file.
library("dotenv")
load_dot_env(".env_demo")

## Same as `x2` but pseudonymized
x3 <- olat_test_results("full_OT11.zip", "onlinetest11.rds")

# -----
# Summary/reshaping the data
# -----

## Calculate basic summary (identical to level = "user", type = "long")
head(s <- summary(x3))

## Summary on task level, wide format
head(swide <- summary(x3, level = "task", type = "wide"))

## Summary on item level, long format
head(slong <- summary(x3, level = "item", type = "long"))

library("tinyplot")
tinyplot(duration ~ 1 | as.factor(task), facet = "by", data = slong, theme = "clean2")
tinyplot(score ~ duration | factor(question_number), type = "jitter",
         facet = ~task, data = slong,
         facet.args = list(nrow = 1), xlab = "Duration [min]", ylab = "Score")

# -----
# Estimating Rasch model
# -----

library("tidyr")
library("psychotools")

## Reshaping the data
tmp <- pivot_wider(subset(slong, select = c("username", "task", "score")),
                  values_fn = function(x) as.integer(sum(x) > 0),
                  names_from = "task", values_from = "score") |> as.data.frame()

tmp <- as.matrix(tmp[, !grepl("^username", names(tmp))])

## Estimate Rasch model
mr <- raschmodel(tmp)

## Default plots
plot(mr, type = "profile")
plot(mr, type = "piplot")

## End(Not run)

```

olat_test_results_get_attempts
Identify Attempts Taken

Description

Olat allows to have multiple attempts per user on one test. For each attempt an XML file containing the users answers and scores achieved contained in the Open Olat ZIP file. This function searches for all matching XML files and returns the attempt details.

Usage

```
olat_test_results_get_attempts(d, lang, verbose = FALSE)
```

Arguments

d	string, name of the directory where the Open Olat ZIP file was extracted. Contains the XML files required.
lang	an object of class "olat_extract_lang" as returned by olat_test_results_guess_language() .
verbose	logical, if TRUE some information is written to stdout.

Value

A data frame containing the attempt (integer), the username (c-Kennung), the name of the participant, as well as the XML file containing the results.

Author(s)

Reto

See Also

[olat_test_results_guess_language\(\)](#)

olat_test_results_get_userdata
Extracting User Details

Description

Loading the user details from the 'user details file', an XLSX file included in the Open Olat results ZIP file.

Usage

```
olat_test_results_get_userdata(lang)
```

Arguments

lang object of class olat_extract_lang as returned by `olat_test_results_guess_language()`.
Used to be able to process results in different languages.

Value

A data frame with institution identifier (Matrikelnummer), username (Open Olat account or user name), and name (full name of the participants).

Author(s)

Reto

olat_test_results_guess_language
Guessing ZIP Content Language

Description

When exporting the results ZIP file via Open Olat, the user language setting is used. This function guesses the language based on the content (file names) of the ZIP file and returns an object used by `olat_test_results()` and its child functions.

Usage

```
olat_test_results_guess_language(d, verbose = FALSE)
```

Arguments

d character, name of the directory where the ZIP content was extracted.
verbose logical, defaults to FALSE.

Details

What differs? Depending on the language the main folder with the results we are interested in is either called "Results" or "Resultate". Similarly, the user attempt folders are called "Attempt_1" or "Versuch_1".

In addition, the user details file (XLSX) containing user name, name etc. comes with specific columns, e.g., "Username" versus "Anmeldename".

Value

An object of class olat_extract_lang, a named list with the required language-dependent specifications.

Author(s)

Reto

`olat_test_results_of_attempt`*Extracting Test Details of Participants Attempt*

Description

This can be seen as the core of `olat_test_results()`, extracting all details of an attempt.

Usage

```
olat_test_results_of_attempt(  
  d,  
  xmlfile,  
  tz,  
  olat_ids = NULL,  
  verbose = FALSE,  
  envir = NULL  
)
```

Arguments

<code>d</code>	path to the directory where the Open Olat ZIP file was extracted.
<code>xmlfile</code>	character, name of the user attempt XML file for which the results should be extracted.
<code>tz</code>	time zone used for date/time conversions. Defaults to "" (uses local/system time zone).
<code>olat_ids</code>	either NULL or a data frame with the information from the manifest XML enriched with the exam file name and title from the rds file created by exams2openolat (see <code>olat_test_results()</code>).
<code>verbose</code>	logical, if TRUE some information is written to stdout.
<code>envir</code>	either NULL or an R environment. Used when <code>cache_xml = TRUE</code> when calling <code>olat_test_results()</code> .

Author(s)

Reto

olat_test_results_read_manifest
Parsing Manifest File

Description

Parsing the "imsmanifest.xml" file.

Usage

```
olat_test_results_read_manifest(d, lang, verbose = FALSE)
```

Arguments

d	character, name of the directory where the ZIP content was extracted.
lang	object as returned by olat_test_results_guess_language() .
verbose	logical, defaults to FALSE.

Value

A data frame containing Olats internal ID as well as Section and Item identifier (integer).

Author(s)

Reto

read_olat_results *Read OLAT Results from XLSX file*

Description

Get OLAT test results (FIXME: eval support and cloze evaluation). TODO: Write proper documentation.

Usage

```
read_olat_results(file, xexam = NULL)
```

Arguments

file	name of the file ...
xexam	...

Value

...

read_vis

*Reading VIS Registrations***Description**

Read registration lists (for exams or courses) from the Excel export of VIS (which actually may or may not be XLS or HTML files).

Usage

```
read_vis(file, ...)
```

```
vis_register(file = Sys.glob("*.xls"), subset = TRUE)
```

Arguments

file	character with file name of an XLS file from VIS.
...	additional arguments passed to read.xlsx .
subset	logical or character. If logical: Should students without confirmed registration be omitted? If character: Select only the students with certain student IDs provided in subset.

Details

VIS offers Excel exports but in case of registration lists these are actually HTML files containing an HTML table. (Note that as of 2021 VIS offers an additional “real Excel” export.) HTMLtables are read using the XML package. However, some exports are also converted to actual Excel files which are read using the xlsx package. In either case some basic cleaning is done and additional meta-information is extracted.

The `vis_register` function loops over reading several VIS exports and then consolidates the resulting data frames.

Value

A data.frame with an additional attribute “info” providing details about the type of course (“LV”) or exam (“GP”).

uibkmark	<i>Auxiliary Formatting Functions</i>
----------	---------------------------------------

Description

Auxiliary functions for formatting elements of exams.

Usage

```
uibkmark(x, factor = TRUE)

mchoice2text(
  x,
  true = "\\textbf{Richtig}",
  false = "\\textbf{Falsch}"
)
```

Arguments

x	numeric (uibkmark) or logical (mchoice2text) vector.
factor	logical. Should the result be a factor or a character?
true	character. Text for true results.
false	character. Text for false results.

Details

The function uibkmark maps the numbers 1 to 5 to the mark labels SGT1, GUT2, etc. as used by UIBK.

The function mchoice2text masks the exams function of the same name in order to show German text.

Examples

```
uibkmark(1:5)
mchoice2text(c(TRUE, FALSE))
```

`vis_groups`*Randomly Assign VIS Participants into Groups (as HTML Table)*

Description

Randomly assign a vector of names (typically obtained from a VIS registration) into groups and display the result as an HTML table.

Usage

```
vis_groups(x, nrow = 5L, ncol = 2L, ...)
```

Arguments

<code>x</code>	character. Either a vector of names or a file name (csv or xls/xlsx from VIS) from which the Name column can be extracted.
<code>nrow, ncol</code>	numeric. Number of rows and columns into which the students should be assigned.
<code>...</code>	additional arguments passed to read_vis or read.csv2 .

Value

A character vector with the HTML code is returned invisibly.

Index

- * **OLAT**
 - olat_eval, 13
- * **datasets**
 - olat_eval_lang, 16
- * **utilities,**
 - olat_eval, 13
- * **utilities**
 - exams2nops, 2
 - exams2openolat, 6
 - nops_eval, 8
 - nops_itemresp, 11
 - olat_exercise, 17
 - read_vis, 34
 - uibkmark, 35
 - vis_groups, 36
- exams2nops, 2, 2, 4, 8–10
- exams2olat, 5, 13, 14
- exams2openolat, 5, 6, 6, 7, 17, 19, 20
- exams2openolat(), 24, 25
- exams2pdf, 4
- exams2qti12, 6
- exams_eval, 6, 7, 9
- format.olat_test_details
 - (olat_test_results), 24
- mchoice2text (uibkmark), 35
- nops_eval, 8, 10, 11
- nops_eval_write_uibk (nops_eval), 8
- nops_feedback, 10, 23
- nops_itemresp, 11
- nops_register (nops_eval), 8
- nops_scan, 8–10
- olat_course_results, 12
- olat_course_results(), 21, 24
- olat_eval, 10, 13, 14, 16, 23
- olat_eval_adjust_lang, 14
- olat_eval_export, 15
- olat_eval_guess_lang, 14, 16
- olat_eval_lang, 16
- olat_exercise, 17
- olat_extract_get_test_responses_and_text, 18
- olat_extract_get_test_responses_and_text(), 19, 26
- olat_extract_get_user_answers_and_scores, 19
- olat_extract_html_results, 19
- olat_extract_html_results(), 26
- olat_feedback, 23, 23
- olat_feedback_render_one
 - (olat_feedback), 23
- olat_results_pseudonymize_data, 24
- olat_test_results, 24
- olat_test_results(), 12, 18, 21, 24, 31, 32
- olat_test_results_get_attempts, 30
- olat_test_results_get_userdata, 30
- olat_test_results_guess_language, 31
- olat_test_results_guess_language(), 30, 31, 33
- olat_test_results_of_attempt, 32
- olat_test_results_read_manifest, 33
- read.csv2, 36
- read.xlsx, 34
- read_olat_results, 15, 33
- read_vis, 10, 34, 36
- summary.olat_test_result
 - (olat_test_results), 24
- summary.olat_test_results
 - (olat_test_results), 24
- Sweave, 3
- uibkmark, 35
- vis_groups, 36
- vis_register (read_vis), 34

xexams, [4](#), [7](#)