

Package: learnitdown (via r-universe)

August 31, 2024

Type Package

Version 1.6.0

Title R Markdown, Bookdown and Learnr Additions for Learning Material

Description Extension to R Markdown, Bookdown and Learnr for building better learning and e-learning material: H5P integration, course-contextual divs, differed loading of Shiny and learnr applications, and much more ...

Maintainer Philippe Grosjean <phgrosjean@sciviews.org>

Depends R (>= 3.0.0)

Imports getPass, glue, gradethis, httpuv, httr, jsonlite, keyring, later, learnr, magick, mongolite, PKI, remotes, rstudioapi, shiny, shinylogs, shinytoastr, stats, utils, webshot

Suggests covr, knitr, rmarkdown, spelling, testthat (>= 3.0.0)

Remotes rstudio/gradethis

License MIT + file LICENSE

URL <https://github.com/learnitr/learnitdown>,
<https://learnitr.github.io/learnitdown/>

BugReports <https://github.com/learnitr/learnitdown/issues>

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

VignetteBuilder knitr

Encoding UTF-8

Language en-US

ByteCompile yes

Config/testthat/edition 3

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

RemoteUrl <https://github.com/learnitr/learnitdown>

RemoteRef HEAD

RemoteSha 7bc6e85db095867e5696157b72765ce65ec948c8

Contents

assignment	2
checker_ack_learnr	6
config	7
diagnose_login	10
h5p	10
launch_shiny	12
learnitdownLearnrSetup	14
learnitdownShiny	15
learnitdown_init	16
learnr	18
obfuscate	19
read_shinylogs	20
record_learnr	20
record_shiny	21
run	22
send_mail_learnr	24
show_ex_toc	25
trackEvents	25
webshot_shiny	28

Index	30
--------------	-----------

assignment	<i>Insert a GitHub (Classroom) assignment or challenge in the document</i>
------------	--

Description

Insert a block of class assign, assign2, challenge or challenge2 with data related to a GitHub (Classroom) assignment.

Usage

```
assignment(
  name,
  url,
  alturl = url,
  course.ids = NULL,
  course.urls = NULL,
  course.starts = NULL,
  course.ends = NULL,
  part = NULL,
  course.names = c(course1 = "Data Science"),
  toc = "",
  clone = TRUE,
  level = 3,
  n = 1,
```

```
    type = if (n == 1) "ind. github" else "group github",
    acad_year = "",
    term = "",
    texts = assignment_en(),
    assign.img = "images/list-assign.png",
    assign.link = "github_assignment",
    block = "assign",
    template = "assignment_en.html",
    baseurl = "/"
)

assignment2(
  name,
  url,
  alturl = url,
  course.ids = NULL,
  course.urls = NULL,
  course.starts = NULL,
  course.ends = NULL,
  part = NULL,
  course.names = c(course1 = "Data Science"),
  toc = "",
  clone = TRUE,
  level = 3,
  n = 1,
  type = if (n == 1) "ind. github" else "group github",
  acad_year = "",
  term = "",
  texts = assignment2_en(),
  assign.img = "images/list-assign2.png",
  assign.link = "github_assignment",
  block = "assign2",
  template = "assignment_en.html",
  baseurl = "/"
)

challenge(
  name,
  url,
  alturl = url,
  course.ids = NULL,
  course.urls = NULL,
  course.starts = NULL,
  course.ends = NULL,
  part = NULL,
  course.names = c(course1 = "Data Science"),
  toc = "",
  clone = TRUE,
```

```

    level = 3,
    n = 1,
    type = if (n == 1) "ind. challenge" else "group challenge",
    acad_year = "",
    term = "",
    texts = challenge_en(),
    assign.img = "images/list-challenge.png",
    assign.link = "github_challenge",
    block = "challenge",
    template = "assignment_en.html",
    baseurl = "/"
)

challenge2(
  name,
  url,
  alturl = url,
  course.ids = NULL,
  course.urls = NULL,
  course.starts = NULL,
  course.ends = NULL,
  part = NULL,
  course.names = c(course1 = "Data Science"),
  toc = "",
  clone = TRUE,
  level = 3,
  n = 1,
  type = if (n == 1) "ind. challenge" else "group challenge",
  acad_year = "",
  term = "",
  texts = challenge2_en(),
  assign.img = "images/list-challenge2.png",
  assign.link = "github_challenge",
  block = "challenge2",
  template = "assignment_en.html",
  baseurl = "/"
)

assignment_en(title, part.name, alt, sub, course, toc.def)
assignment_fr(title, part.name, alt, sub, course, toc.def)
assignment2_en(title, part.name, alt, sub, course, toc.def)
assignment2_fr(title, part.name, alt, sub, course, toc.def)
challenge_en(title, part.name, alt, sub, course, toc.def)

```

```
challenge_fr(title, part.name, alt, sub, course, toc.def)
```

```
challenge2_en(title, part.name, alt, sub, course, toc.def)
```

```
challenge2_fr(title, part.name, alt, sub, course, toc.def)
```

Arguments

<code>name</code>	The name of the assignment or the challenge (usually the same as the name as the GitHub Classroom assignment).
<code>url</code>	The URL of the assignment or challenge (could be a named list for different courses).
<code>alturl</code>	An alternate URL to propose to external users not registered in a course. If not provided, it is the same as <code>url</code> .
<code>course.ids</code>	Named vector with the Classroom identifiers for the assignments for each course and also possibly, for each group.
<code>course.urls</code>	Named vector with the Classroom URLs for each course, and also possibly for groups. Names are the course identifiers in Moodle, or the groups defined for the users. If NULL, no course-specific sections are added.
<code>course.starts</code>	Named vector (same logic as for <code>course.urls</code>) with starting dates (characters like "YYYY-mm-dd HH:MM:SS"). A corresponding entry in <code>course.urls</code> is required, or the dates will be ignored.
<code>course.ends</code>	Named vector (same logic as for <code>course.urls</code>) with ending dates (characters like "YYYY-mm-dd HH:MM:SS"). A corresponding entry in <code>course.urls</code> is required, or the dates will be ignored.
<code>part</code>	If the assignment presents several parts in the README.md file (because the same assignment is used at different places), indicate the part here. Otherwise, leave the default value NULL if there are no parts.
<code>course.names</code>	A named character vector with the name of the course. Names used must be the same as for <code>course.urls</code> , but there may be more here, and not necessarily in the same order.
<code>toc</code>	The exercise table of content (ex-toc) label. If <code>toc = ""</code> (default), a generic label is calculated using <code>toc.def</code> from <code>texts</code> . To not display the exercise in the table of content, use <code>toc = NULL</code> .
<code>clone</code>	Should the exercise be listed for cloning the repositories (TRUE by default)? If TRUE, an entry is added in the list of assignments whose repositories should be cloned.
<code>level</code>	The difficulty level (1 = easiest, 2 = more difficult, ...)
<code>n</code>	The number of students per project (by default, <code>n = 1</code> for individual assignments and <code>2</code> for group assignments).
<code>type</code>	The type of exercise. By default, it is "ind. github" or "ind. challenge" if <code>n = 1</code> , or "group github"/"group challenge" otherwise.
<code>acad_year</code>	The academic year (e.g., 2021-2022).
<code>term</code>	The term (e.g., Q1, Q2, Q3).

texts	Various sentences used to construct the assignment bloc. You can make a call to <code>assignment_en()</code> or <code>assignment_fr()</code> as a basis and modify only the sentences you want.
assign.img	The relative path to the image to use before the label in the ex-toc.
assign.link	The link to the assignment help page (when the user clicks on the image in the ex-toc).
block	The class of the div, or the LaTeX environment to use for the assignment block.
template	The template file to use for the URL redirection page (currently only <code>assignment_en.html</code> or <code>assignment_fr.html</code>).
baseurl	The base URL for the web site.
title	The title of the block.
part.name	The word to use for "part".
alt	The text to display for alternate access to the repository (for non-registered users). Use a string following the <code>glue()</code> syntax to replace variables.
sub	The text that appears at the bottom of the assignment block.
course	The text for items corresponding to courses.
toc.def	The default ex-toc label (using <code>glue()</code> syntax).

Details

If the URL contains several entries, names are used to create show/hide divs according to the `icourse` user information.

Value

Markdown code that generates the GitHub Classroom assignment block. It is most conveniently used inside an R chunk in your R Markdown document. If you do not want to break your code chunks inside RStudio, you may use something like `if (exists("assignment")) assignment(...)`.

checker_ack_learnr *A default checker that just acknowledges submission*

Description

Check code submitted during an exercise. This version just acknowledges reception of the submission. This function is used internally by the tutorials and is not intended for the end-user.

Usage

```
checker_ack_learnr(  
  label,  
  user_code,  
  solution_code,  
  check_code,  
  envir_result,  
  evaluate_result,  
  ...  
)
```

Arguments

label	The label for the learnr exercise.
user_code	The code submitted by the user.
solution_code	The code provided by the "-solution" chunk.
check_code	The code provided by the "-check" chunk.
envir_result	The environment after the execution of the chunk.
evaluate_result	Result from evaluation of the code.
...	Additional parameters (currently not used).

Details

This is a simple checker function for learnitdown learnr applications that just indicates to the user that its answer is taken into account.

Value

A list with components message, correct and location.

See Also

[record_learnr\(\)](#)

config

Configure the R environment for the course (including database information) and provide (or cache) user information in ciphered form.

Description

Call these functions every time you need to get environment variables set, like the URL, user and password of the MongoDB database used by the course.

Usage

```
config(  
    url,  
    password,  
    cache = file.path(tempdir(), ".learnitdown_config"),  
    debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0  
)  
  
sign_in(  
    data,  
    password,  
    cipher = "aes-256-cbc",  
    iv = NULL,  
    cache = file.path(tempdir(), ".learnitdown_user"),  
    debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0  
)  
  
sign_out(  
    title = "Signing out",  
    message = "Do you really want to sign out with learnitdown?",  
    cache = file.path(tempdir(), ".learnitdown_user"),  
    debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0  
)  
  
encrypt(  
    object,  
    password,  
    cipher = "aes-256-cbc",  
    iv = NULL,  
    serialize = TRUE,  
    base64 = FALSE,  
    url.encode = FALSE  
)  
  
decrypt(  
    object,  
    password,  
    cipher = "aes-256-cbc",  
    iv = NULL,  
    unserialize = TRUE,  
    base64 = FALSE,  
    url.decode = FALSE  
)  
  
lock(  
    object,  
    password,  
    key = "",
```

```

    message = "Password for learnitdown:",
    reset = FALSE,
    ref1 = NULL,
    ref2 = NULL
)

unlock(
  object,
  password,
  key = "",
  message = "Password for learnitdown:",
  reset = FALSE,
  ref1 = NULL,
  ref2 = NULL
)

```

Arguments

url	The URL of the encrypted file that contains the configuration information.
password	The password to crypt, decrypt, lock or unlock the data.
cache	The path to the file to use to store a cached version of these data. Access to the database will be checked, and if it fails, the configuration data are refreshed from the URL.
debug	Do we issue debugging messages? By default, it is set according to the LEARNITDOWN_DEBUG environment variable (yes, if this variable is not \emptyset).
data	The fingerprint data in clear or ciphered form (in this case, the string must start with "fingerprint=").
cipher	The cryptography algorithm to use.
iv	The initialization vector for the cipher.
title	The title of the dialog box prompting to sign out.
message	The message of the dialog box prompting to sign out, or asking for a password.
object	An object to be encrypted, decrypted, locked or unlocked.
serialize	Do we serialize object before ciphering it (TRUE by default)?
base64	Do we encode/decode base64 the object (FALSE by default)?
url.encode	Do we encode for URL (query) use (FALSE by default) ?
unserialize	Do we unserialize the resulting object after deciphering it (TRUE by default)?
url.decode	Do we decode URL before deciphering it (FALSE by default)?
key	The key that stores the password. It is advised to use something like <code>course_year</code> , so that you can manage different passwords for the same course given at different academic years. Using a key, the course password must be entered only once. If not provided, the password is not stored. Note also that the name of an environment variable could be used too for the key. This is convenient on a server like RStudio Connect, for instance.
reset	Should we reset the password (FALSE by default)?
ref1	Code to check the validity of the password
ref2	Second code to check password validity.

Value

Invisibly returns TRUE if success, or FALSE otherwise for `config()`. The encrypted/decrypted object for `encrypt()` and `decrypt()`, or the locked/unlocked object for `lock()` and `unlock()`. The user information for `sign_in()`.

<code>diagnose_login</code>	<i>Insert code in an R Markdown document to diagnose personal data information</i>
-----------------------------	--

Description

This function is used, inside an R Markdown document, to dynamically elaborate a short report on the client-side (by using JavaScript code) that displays the login status of the user (login, contextual data, ...).

Usage

```
diagnose_login(block = "info", lang = "en")
```

Arguments

<code>block</code>	Should the diagnostic be included in a block construct (not if NULL)?
<code>lang</code>	The language to use for diagnostic output (only 'en' or 'fr' currently)

Details

Wherever you want to display this diagnostic report in your bookdown page, add on its own line the following instruction: ``r learnitdown::diagnose_login()`` between backquotes (you can also change `lang`, if you want).

Value

Nothing, the function is used for its side-effect of adding diagnostic code in the document

<code>h5p</code>	<i>Insert H5P content in the document</i>
------------------	---

Description

Insert H5P content in the document

Usage

```

h5p(
  id,
  baseurl,
  idurl = "wp-admin/admin-ajax.php?action=h5p_embed&id=",
  width = 780,
  height = 500,
  toc = "",
  toc.def = "H5P exercise {id}",
  h5p.img = "images/list-h5p.png",
  h5p.link = "h5p"
)

```

Arguments

<code>id</code>	The ID of the H5P content in your Wordpress.
<code>baseurl</code>	The first part of the URL for your domain, usually something like <code>https://my.site.com</code> without the trailing <code>/</code> .
<code>idurl</code>	The URL to the H5P content, usually something like <code>""</code> for exercises from <code>h5p.org</code> or <code>h5p.com</code> , or <code>"wp-admin/admin-ajax.php?action=h5p_embed&id="</code> for Wordpress (by default).
<code>width</code>	The width of the iframe where the H5P content is displayed.
<code>height</code>	The height of the iframe.
<code>toc</code>	Entry to use in the exercises table of content (NULL if no entry, <code>""</code> for a default entry based on <code>toc.def</code> =).
<code>toc.def</code>	Text for a default toc entry using <code>glue()</code> syntax for replacement, e.g., <code>{var}</code> .
<code>h5p.img</code>	The image to display in front of the toc entry
<code>h5p.link</code>	The link when the image is clicked (sends to an help page about learnr tutorials).

Details

This function is designed to work inside a Wordpress site where the H5P plugin has been installed. You should also serve your bookdown pages as a subdirectory inside of the same Wordpress site to allow free communication between the parent (the bookdown page) and the child document in the iframe (the H5P content).

Value

HTML code that generates the iframe. It is most conveniently used inside and R inline expression in your R Markdown document on its own line with one blank line above and below it.

 launch_shiny

Launch Shiny application by clicking on its screenshot.

Description

Shiny applications can be embedded in certain R Markdown documents. However, the application is automatically loaded at the same time as the main page, and that may not be the desired behavior. With `launch_shiny()`, you display just a screenshot of the Shiny application in the page. The user has to click on it to actually launch the application (that replaces the screenshot on the page).

Usage

```
launch_shiny(
  url,
  app = sub("\\?.+$$", "", basename(url)),
  imgdir = "images/shinyapps",
  img = paste0(imgdir, "/", app, ".png"),
  createimg = TRUE,
  width = 790,
  height = 500,
  fun = NULL,
  alt1 = "*Click to start the Shiny application.*",
  alt2 = paste0("*Click to start",
    "or [run `run.cmd`]({run.url}{run.arg}){{target=\"_blank\"}}.*"),
  toc = "",
  toc.def = "Shiny application {app}",
  run.url = "start_rstudio.html?runrcode=",
  run.cmd = glue("{fun}({app})"),
  run.arg = URLEncode(run.cmd, reserved = TRUE),
  app.img = "images/list-app.png",
  app.link = "shiny_app",
  ...
)
```

Arguments

<code>url</code>	The URL of the Shiny application. If both <code>app =</code> and <code>baseurl =</code> are provided, you don't need to specify it.
<code>app</code>	Name of the shiny application (cannot be duplicated on a page).
<code>imgdir</code>	The directory without trailing "/" where images relative to Shiny applications are stored. By default, it is relative to current directory, in <code>images/shinyapps</code> subdirectories.
<code>img</code>	The relative or absolute path to the image with a screenshot of the Shiny application, as produced by <code>webshot_shiny()</code> .
<code>createimg</code>	If the app image (<code>img</code>) is not found, and there is no default image in <code>imgdir =</code> , do we put it there (yes be default)?

width	The width of the image and iframe for the app.
height	The height of image and iframe.
fun	The function to run as alternative to start the Shiny application locally. It is better to fully specify it (<code>package::function</code>), and it should take one argument which is the application name in <code>app =</code> .
alt1	Alternate text to display at the bottom of the screenshot when nothing is provided for <code>fun =</code> . If NULL, nothing is displayed below the screenshot.
alt2	Alternate text to display at the bottom of the screenshot in case <code>fun =</code> is provided.
toc	Entry to use in the exercises table of content (NULL if no entry, "" for a default entry based on <code>toc.def =</code>).
toc.def	Text for a default toc entry using <code>glue()</code> syntax for replacement, e.g., <code>{app}</code> .
run.url	The URL to use to start the Shiny application in RStudio server in the SciViews Box. It should generally end with <code>?runrcode=</code> , and the R code to execute will be appended to it from <code>run.arg =</code> .
run.cmd	The command to use to launch the Shiny application in RStudio.
run.arg	The URL encoded version of <code>run.cmd =</code> .
app.img	The image to display in front of the toc entry
app.link	The link when the image is clicked (sends to an help page about Shiny applications).
...	Not used here, but it allows to add more arguments used by the screenshot addin, like <code>delay =</code> , <code>offsetx =</code> or <code>offsety =</code> , see webshot_shiny() .

Value

The HTML content that creates the image and the iframe. The function must be called from within an R inline expression or from an R chunk with `results='asis'` in an HTML-rendered version of the R Markdown document to get the correct result.

See Also

[webshot_shiny\(\)](#)

Examples

```
# TODO...
```

 learnitdownLearnrSetup

Set up a learnitdown Learnr application

Description

This function eases the configuration of the learnr document to get user and database info, record events, use grade this and parameterize learnr.

Usage

```
learnitdownLearnrSetup(
  config,
  sign_in,
  time.limit = 60,
  cap = "R Code",
  echo = FALSE,
  comment = NA,
  use.gradethis = TRUE,
  event.recorder = learnitdown::record_learnr,
  debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0
)
```

```
learnitdownLearnrBanner(
  title,
  text,
  image,
  align = "left",
  msg.nologin = "Anonymous user, no record!",
  msg.login = "Recording activated for ",
  msg.error = "Error recording activity! "
)
```

```
learnitdownLearnrServer(input, output, session)
```

Arguments

<code>config</code>	The <code>config()</code> command to use to get database info.
<code>sign_in</code>	The <code>sign_in()</code> command to use to get user info.
<code>time.limit</code>	The maximum time allowed to evaluate R code.
<code>cap</code>	The caption for R code widgets.
<code>echo</code>	Do we echo commands in R chunks?
<code>comment</code>	The prefix added before each line of R chunk output.
<code>use.gradethis</code>	Do we use gradethis?

<code>event.recorder</code>	The function to use as event recorder. you should probably not change the default value here.
<code>debug</code>	Do we issue additions debugging informations?
<code>title</code>	The Title for the banner.
<code>text</code>	Text to print beneath the title.
<code>image</code>	URL to an image to display in the banner.
<code>align</code>	How is the image aligned: "left" (default), "right", "middle", "top" or "bottom".
<code>msg.nologin</code>	The message to display if no user is logged in.
<code>msg.login</code>	The message to display if a user is logged in (will be followed by the login).
<code>msg.error</code>	The message when an error during recording of activity in the database occurs.
<code>input</code>	The Shiny input.
<code>output</code>	The Shiny output.
<code>session</code>	The Shiny session.

Value

Nothing. The function is used to setup the learnr environment.

learnitdownShiny

Create and manage learnitdown Shiny applications

Description

A learnitdown Shiny application is an application whose events (start, stop, inputs, outputs, errors, result, quit) are recorded. It also provides a 'Submit Answer' and a 'Quit' buttons that manage to check the answer provided by the user and to close the application cleanly.

Usage

```
learnitdownShiny(title, windowTitle = title)
```

```
learnitdownShinyVersion(version)
```

```
submitAnswerButton(
  inputId = "learnitdown_submit_",
  label = "Submit",
  class = "btn-primary",
  ...
)
```

```
quitButton(
  inputId = "learnitdown_quit_",
  label = "Quit",
  class = "btn-secondary",
```

```

    ...
  )

  submitQuitButtons()

```

Arguments

<code>title</code>	The title of the Shiny application
<code>windowTitle</code>	The title of the window that holds the Shiny application, by default, it is the same as <code>title</code> . If <code>title = NULL</code> , no title is added.
<code>version</code>	The version number (in a string character format) of the Shiny application to use in the events logger.
<code>inputId</code>	The identifier of the button ("learnitdown_submit_" or "learnitdown_quit_").
<code>label</code>	The button text ("Submit" or "Quit").
<code>class</code>	The bootstrap class of the button.
<code>...</code>	Further arguments passed to <code>shiny::actionButton()</code> .

Value

The code to be inserted at the beginning of the Shiny application UI for `learnitdownShiny()` or where the buttons should be located for the other function.

See Also

[trackEvents\(\)](#), [record_shiny\(\)](#)

Examples

```
learnitdownShiny("My title")
```

learnitdown_init	<i>Initialize learnitdown features in an R Markdown document</i>
------------------	--

Description

This function must be called in a script run by `before_chapter_script` entry in `_bookdown.yml` to create required `style.css` and `header.html` files.

Usage

```

learnitdown_init(
  shiny = TRUE,
  h5p = TRUE,
  use.query = FALSE,
  iframe.links = TRUE,
  details.css = TRUE,

```

```

baseurl = "https://example.org",
institutions = c("institution1", "institution2"),
courses = c("course1", "course2", "course3"),
style = "style.css",
style0 = "style0.css",
header = "header.html",
header0 = "header0.html",
hide.code.msg = "See code"
)

```

Arguments

shiny	Do we use Shiny applications and do we want to pass parameters and or launch the application on a click?
h5p	Do we use H5P served from a Wordpress site in the same domain as our R Markdown document? The H5P integration plugin, and the H5PxAPIkatchu Wordpress plugins must be installed in order to serve H5P apps and to record the H5P events through the xAPI interface.
use.query	Do we collect user/course/institution data through the URL query string (the part after the question mark in the URL).
iframe.links	If our document is displayed in an iframe, external link should better target their parent window. With this option, external links with no defined target are automatically retargeted when the page loads.
details.css	Do we want to enhance the <details> section with a summary surrounded by a light gray box in order to better evidence it.
baseurl	The URL where the site is server from (for H5P integration), it is also the base URL for the associated Wordpress server with H5P plugin. Provide it without the trailing /!
institutions	The list of possible institutions that have specific sections in this document.
courses	The list of courses with specific sections in this document.
style	The path to the 'style.css' file.
style0	The path to a file with additional content to add to the 'style.css' file.
header	The path to the 'header.html' file.
header0	The path to a file with additional content to add to 'header.html'.
hide.code.msg	The message to display for hidden code.

Value

A list with `css` and `html` components with the content that was added to respective files is returned invisibly for debugging purposes (the function is mainly used for its side effect of creating `style.css` and `header.html` files for the bookdown format).

Examples

```
# This is better placed in a setup R chunk or an R inline expression on its
# own line. To see the code injected, use `cat()` at the R prompt:
odir <- setwd(tempdir())
dir.create("temp")
setwd("temp")
# Create fake style0.css and header0.html files to see what happens
cat("\n/* Content from style0.css */\n", file = "style0.css")
cat("\n<!-- Content from header0.html -->\n", file = "header0.html")
# Create style.css and header.html files
(learnitdown_init())
cat(readLines('style.css'), sep = "\n")
cat(readLines('header.html'), sep = "\n")
setwd("../")
unlink("temp")
setwd(odir)
rm(odir)
```

learnr

Insert a block for a learnr tutorial

Description

Insert Markdown text to link to a learnr tutorial.

Usage

```
learnr(
  id,
  title = NULL,
  package,
  toc = "",
  text = "Now let's make the exercises in the following tutorial:",
  toc.def = "Tutorial {id}",
  rstudio.url = "start_rstudio.html",
  tuto.img = "images/list-tuto.png",
  tuto.link = "tutorial"
)
```

Arguments

<code>id</code>	Identifier of the learnr tutorial (as <code>tutorial:id</code> in the YAML section).
<code>title</code>	The title of the tutorial (<code>title</code> in the YAML section). If <code>NULL</code> , the <code>id</code> is used instead.
<code>package</code>	Package where the learnr tutorial is defined.

toc	Entry to use in the exercises table of content (NULL if no entry, "" for a default entry based on toc.def =).
text	The text to display in the learnr block.
toc.def	Text for a default toc entry using <code>glue()</code> syntax for replacement, e.g., {id}.
rstudio.url	The URL to open a page in RStudio server in the SciViews box.
tuto.img	The image to display in front of the toc entry
tuto.link	The link when the image is clicked (sends to an help page about learnr tutorials).

Value

The Markdown chunk to insert a learnr tutorial block in the document.

obfuscate	<i>Obfuscate answers in learnr documents</i>
-----------	--

Description

Obfuscate answers in learnr documents

Usage

```
obfuscate(object, password)
```

```
._(object)
```

```
obfuscate_logical(x, y, r = FALSE)
```

```
ans(correct, text, message = NULL)
```

Arguments

object	An R object to obfuscate, or Obfuscation string to decrypt.
password	The password to use (otherwise, the password defined by the application is used)
x	A logical value or an integer.
y	Obfuscation parameter.
r	Reset obfuscation (not intended for end-user)?
correct	Is the answer is correct or not (either a logical value, or an integer that obfuscates the results).
text	Text of the option.
message	Message displayed if the item is selected

Value

The encrypted or decrypted object.

Examples

```
obfuscate("test", "password")
```

read_shinylogs	<i>Read shinylogs log data and format them in a data.frame</i>
----------------	--

Description

Learntidown Shiny applications are a special kind of Shiny applications that logs events and check results. It uses the shinylogs package to log Shiny events and `read_shinylogs()` reads such logs and convert their data into a format that is suitable to include, say in a MongoDB database.

Usage

```
read_shinylogs(file, version = "0", log.errors = TRUE, log.outputs = FALSE)
```

Arguments

file	The path to the RDS file that contains the shinylogs log data.
version	The version of the Shiny application. Version is not recorded by shinylogs, so, we must provide it here.
log.errors	Do we record errors too?
log.outputs	Do we record outputs too (note that results are recorded with inputs)?

Value

A data frame with the different events logged (one per line).

See Also

[record_shiny\(\)](#), [trackEvents\(\)](#)

record_learnr	<i>Record results of learnr exercises in a MongoDB database</i>
---------------	---

Description

Record tutorial submissions in a MongoDB database. The function is used by learntidown learnr tutorials and is not for end-users.

Usage

```
record_learnr(tutorial_id, tutorial_version, user_id, event, data)
```

```
user_name(value)
```

```
user_email(value)
```

Arguments

tutorial_id	The identifier of the tutorial.
tutorial_version	The version of the tutorial.
user_id	The user identifier for this learnr process.
event	The event that triggers the record, like <code>exercise_submission</code> or <code>question_submission</code>
data	A JSON field with event-dependent data content. If NULL, only a test to see if the database is responding is performed.
value	The new value for user name or email (if not provided, the current value is returned).

Value

Nothing. The function is used for its side-effects.

See Also

[send_mail_learnr\(\)](#)

record_shiny

Record Shiny events in a MongoDB database

Description

Given a path that contains shinylogs events in .rds format, read these events and transfer them into a MongoDB database.

Usage

```
record_shiny(  
  path,  
  url,  
  db,  
  collection = "shiny",  
  version = "0",  
  log.errors = TRUE,  
  log.outputs = FALSE,  
  drop.dir = FALSE,  
  debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0  
)
```

Arguments

path	The directory that contains shinylogs .rds files
url	The mongodb url.
db	The database name.
collection	The name of the collection where to insert the documents.
version	The version of the running Shiny application.
log.errors	Do we record errors too?
log.outputs	Do we record outputs too (note that results are recorded with inputs)?
drop.dir	If TRUE and path is empty at the end of the process, drop the logs directory.
debug	Do we debug the events recording by issuing extra messages? By Default the value in the environment variable LEARNITDOWN_DEBUG is used and debugging will be done if this value is different to 0.

Value

TRUE if there where log files to export, FALSE otherwise.

See Also

[read_shinylogs\(\)](#), [trackEvents\(\)](#)

run	<i>Run learnitdown learnr tutorials or Shiny apps from a package after update</i>
-----	---

Description

These functions are convenient in the framework of a course whose learnrs and Shiny applications may be updated during the course. The [update\(\)](#) function checks if an update is available (respecting the version of R), and the [run\(\)](#) (for learnrs), or [runApp\(\)](#) (for Shiny applications) manage to run the item in a friendly way.

Usage

```
run(
  tutorial,
  package,
  github_repos = NULL,
  ...,
  update = ask,
  ask = interactive(),
  upgrade = "never"
)
```

```

run_app(
  app,
  package,
  github_repos = NULL,
  ...,
  update = ask,
  ask = interactive(),
  upgrade = "never",
  in.job = TRUE,
  max.wait = 60
)

update_pkg(package, github_repos, upgrade = "never")

```

Arguments

tutorial	The name of the tutorial to use. If not provided, a list of available tutorials is displayed.
package	The package from where to run the tutorial.
github_repos	The GitHub repository where the package is developed (for updates), use NULL to prevent any updates.
...	Further arguments passed to <code>run_tutorial()</code> (for learnrs), or to <code>runApp()</code> (for Shiny applications).
update	Do we check for an updated version first, and if it is found, update the package automatically?
ask	In case <code>tutorial</code> or <code>app</code> is not provided, do we ask to select in a list?
upgrade	When a new version of the main package is found, do we also upgrade dependencies? By default, never, but use "ask" to ask user.
app	The name of the Shiny application to run. If not provided, a list of available apps is displayed.
in.job	Should the application be run in a Job in RStudio (TRUE by default)?
max.wait	How many seconds do we wait for the Shiny app to start (60 sec by default)?

Value

The result returned by `run_tutorial()` for `run()`, or by `runApp()` for `run_app()`. The `update()` function return TRUE or FALSE, depending if the package is updated or not.

See Also

[run_tutorial\(\)](#), [runApp\(\)](#)

Examples

```

## Not run:
#' # To start from a list of available tutorials:
run(package = "my_package")

```

```
run("my_tutorial", package = "my_package")
run_app(package = "mypackage")
run_app("my_shiny_app", package = "mypackage")

## End(Not run)
```

send_mail_learnr *Send your learnr submissions by email*

Description

Your submissions are sent to a central database. However, in case that database is not accessible, the data is stored locally. This function uses your plain email to send your records. Note that, once the email is created, the local version of your records is reset. So, if you finally decide to NOT send the email, these records are lost (in this case, call your teachers to recover them, if you have to.)

Usage

```
send_mail_learnr(
  address,
  subject = "Learnr activity",
  file = Sys.getenv("LOCAL_STORAGE")
)
```

Arguments

address	The mail address to send the data to.
subject	The title of the mail.
file	The file that contains your learnr activity information.

Details

In case the MongoDB cannot be reached, learnr events are stored in a local file. This function allows to submit its content through email as an alternate way to collect learnr activity.

Value

The data are returned invisibly.

See Also

[record_learnr\(\)](#)

Examples

```
## Not run:
send_mail_learnr("me@mymail.org")

## End(Not run)
```

show_ex_toc	<i>Insert a table of content for the exercises at the end of a bookdown chapter</i>
-------------	---

Description

For the various exercise types (h5p, shiny apps, learnr & GitHub assignments/challenges) we add toc entries with `h5p()`, `launch_shiny()`, `learnr()`, `assignment()`, and `challenge()` respectively. This function creates the table of content for all these exercises.

Usage

```
show_ex_toc(header = "", clear.it = TRUE)

clean_ex_toc()
```

Arguments

header	A Markdown text to place as header of the exercises toc.
clear.it	Do we clear the toc list (TRUE by default)?

Value

The Markdown chunk with the exercises toc.

trackEvents	<i>Track events, the submit or the quit buttons</i>
-------------	---

Description

These functions provide the required code to be inserted in the server part of a Shiny application to track events (start, stop, inputs, outputs, errors, result or quit), to check the answer when the user clicks on the submit button and to cleanly close the application when the user clicks on the quit button.

Usage

```
trackEvents(
  session,
  input,
  output,
  sign_in.fun = NULL,
  url = Sys.getenv("MONGO_URL"),
  url.server = Sys.getenv("MONGO_URL_SERVER"),
  db = Sys.getenv("MONGO_BASE"),
```

```

user = Sys.getenv("MONGO_USER"),
password = Sys.getenv("MONGO_PASSWORD"),
version = getOption("learnitdown.shiny.version"),
path = Sys.getenv("LEARNITDOWN_LOCAL_STORAGE", "shiny_logs"),
log.errors = TRUE,
log.outputs = FALSE,
drop.dir = TRUE,
config = NULL,
debug = Sys.getenv("LEARNITDOWN_DEBUG", 0) != 0
)

trackSubmit(
  session,
  input,
  output,
  solution = NULL,
  max_score = NULL,
  comment = "",
  message.success = "Correct",
  message.error = "Incorrect",
  score.txt = "Score",
  check.solution = check_shiny_solution
)

trackQuit(session, input, output, delay = 60)

check_shiny_solution(answer, solution)

```

Arguments

session	The current Shiny session.
input	The Shiny input object.
output	The Shiny output object.
sign_in.fun	The function that can get user info from the sign_in() , or NULL by default to disable using local user data.
url	The URL to reach the MongoDB database. By default, it is read from the MONGO_URL environment variable.
url.server	The URL to reach the MongoDB database. By default, it is read from the MONGO_URL_SERVER environment variable.
db	The database to populate in the MongoDB database. By default, it is read from the MONGO_BASE environment variable.
user	The user login to the MongoDB database. By default, it is read from the MONGO_USER environment variable.
password	The password to access the MongoDB database. By default, it is read from the MONGO_PASSWORD environment variable.
version	The version of the current Shiny application. By default, it is the <code>learnitdown.shiny.version</code> option, as set by learnitdownShinyVersion() .

path	The path where the temporary shinylogs log files are stored. By default, it is set to the LEARNITDOWN_LOCAL_STORAGE environment variable, or to shiny_logs subdirectory of the application if not defined. If that directory is not writable, a temporary directory is used instead.
log.errors	Do we log error events (yes by default)?
log.outputs	Do we log output events (no by default)?
drop.dir	Do we erase the directory indicated by path = if it is empty at the end of the process (yes by default).
config	The result of the call to <code>config()</code> , if done.
debug	Do we debug recording of events using extra messages? By default, it is the value of the environment variable LEARNITDOWN_DEBUG, and debugging is activated when that value is different to 0.
solution	The correct solution as a named list. Names are the application inputs to check and their values are the correct values. The current state of these inputs will be compared against the solution, and the <code>message.success</code> or <code>message.error</code> is displayed accordingly. Also a result event is triggered with the <code>correct</code> field set to TRUE or FALSE accordingly. If <code>solution = NULL</code> , then the answer is considered to be always correct (use this if you just want to indicate that the user's answer is recorded in the database). For numeric values, use <code>'c(min = x, max = y)</code> to check if values are within a range.
max_score	The highest score value that could be attributed if the exercise is correctly done. By default, it is NULL meaning that the higher score would be equal to the number of items to check in solution.
comment	A string with a comment to append to the value of the result event.
message.success	The message to display is the answer is correct ("Correct" by default).
message.error	The message to display if the answer is wrong ("Incorrect" by default).
score.txt	The word to use for "score" ("Score" by default).
check.solution	The function to use to check if solution provided by a Shiny application is correct or not. By default, it is <code>check_shiny_solution()</code> .
delay	The time to wait before we close the Shiny application in sec (60 sec by default). If <code>delay = -1</code> , the application is not closed, only the session is closed when the user clicks on the quit button.
answer	A named list of the answer data to check against solution. This object is provided by <code>trackSubmit()</code> .

Value

The code to be inserted in the server part of the learntitdown Shiny application in order to properly identify the user and record the events.

See Also

[learnitdownShinyVersion\(\)](#), [sign_in\(\)](#)

Description

The `webshot_shiny()` function is designed to create the screenshot image of a Shiny application, with an icon suggesting to click on it for launching the application. The image created can then be used by `launch_shiny()` in a bookdown to differ the start of Shiny application, while displaying useful information to the user (how the Shiny application would look like if it was started).

Usage

```
webshot_shiny(
  url,
  app = basename(url),
  imgdir = "images/shinyapps",
  img = paste0(imgdir, "/", app, ".png"),
  width = 790,
  height = 500,
  offsetx = 30,
  offsety = 30,
  delay = 10,
  ...
)
```

Arguments

<code>url</code>	The URL to launch the Shiny app. If both <code>app =</code> and <code>baseurl =</code>
<code>app</code>	The name of the Shiny application. are provided, you don't need to specify it.
<code>imgdir</code>	The directory without trailing "/" where images relative to Shiny applications are stored. By default, it is relative to current directory, in <code>images/shinyapps</code> subdirectories.
<code>img</code>	The path to the image that is created. Not needed if <code>app =</code> and <code>imgdir =</code> are provided.
<code>width</code>	The requested weight of the screenshot (it may differ if the Shiny application defines other (limit) values).
<code>height</code>	The requested height of the screenshot (idem).
<code>offsetx</code>	The offset from left where to place the click icon in pixels.
<code>offsety</code>	The offset to bottom where to place the click icon in pixels.
<code>delay</code>	Time to wait (in sec) after the Shiny application has started and before the screenshot is taken. If the screenshot does not contain the complete application UI, try increase this value.
<code>...</code>	Further arguments passed to <code>launch_shiny()</code> but not used here.

Value

The path to the created image, invisibly.

See Also

[launch_shiny\(\)](#)

Examples

```
## Not run:  
# We wait 10 sec to make sure it is loaded when the screenshot is taken  
(webshot_shiny("https://phgrosjean.shinyapps.io/histogram/", delay = 10))  
# Now, look at this image. You can use it with launch_shiny()  
  
## End(Not run)
```

Index

- * **record events from the BioDataScience package**
 - checker_ack_learnr, 6
 - record_learnr, 20
- * **run interactive learnr documents from the BioDataScience package**
 - send_mail_learnr, 24
- * **run interactive learnr documents**
 - run, 22
- * **utilities**
 - checker_ack_learnr, 6
 - record_learnr, 20
 - run, 22
 - send_mail_learnr, 24
- ._ (obfuscate), 19
- ans (obfuscate), 19
- assignment, 2
- assignment(), 25
- assignment2 (assignment), 2
- assignment2_en (assignment), 2
- assignment2_fr (assignment), 2
- assignment_en (assignment), 2
- assignment_fr (assignment), 2
- challenge (assignment), 2
- challenge(), 25
- challenge2 (assignment), 2
- challenge2_en (assignment), 2
- challenge2_fr (assignment), 2
- challenge_en (assignment), 2
- challenge_fr (assignment), 2
- check_shiny_solution (trackEvents), 25
- checker_ack_learnr, 6
- clean_ex_toc (show_ex_toc), 25
- config, 7
- config(), 10, 27
- decrypt (config), 7
- decrypt(), 10
- diagnose_login, 10
- encrypt (config), 7
- encrypt(), 10
- glue(), 6, 11, 13, 19
- h5p, 10
- h5p(), 25
- launch_shiny, 12
- launch_shiny(), 12, 25, 28, 29
- learnitdown_init, 16
- learnitdownLearnrBanner
 - (learnitdownLearnrSetup), 14
- learnitdownLearnrServer
 - (learnitdownLearnrSetup), 14
- learnitdownLearnrSetup, 14
- learnitdownShiny, 15
- learnitdownShiny(), 16
- learnitdownShinyVersion
 - (learnitdownShiny), 15
- learnitdownShinyVersion(), 26, 27
- learnr, 18
- learnr(), 25
- lock (config), 7
- lock(), 10
- obfuscate, 19
- obfuscate_logical (obfuscate), 19
- quitButton (learnitdownShiny), 15
- read_shinylogs, 20
- read_shinylogs(), 20, 22
- record_learnr, 20
- record_learnr(), 7, 24
- record_shiny, 21
- record_shiny(), 16, 20
- run, 22
- run(), 22, 23

`run_app (run)`, 22
`run_app()`, 23
`run_tutorial()`, 23
`runApp()`, 22, 23

`send_mail_learnr`, 24
`send_mail_learnr()`, 21
`shiny::actionButton()`, 16
`show_ex_toc`, 25
`sign_in (config)`, 7
`sign_in()`, 10, 26, 27
`sign_out (config)`, 7
`submitAnswerButton (learnitdownShiny)`,
15
`submitQuitButtons (learnitdownShiny)`, 15

`trackEvents`, 25
`trackEvents()`, 16, 20, 22
`trackQuit (trackEvents)`, 25
`trackSubmit (trackEvents)`, 25

`unlock (config)`, 7
`unlock()`, 10
`update()`, 22, 23
`update_pkg (run)`, 22
`user_email (record_learnr)`, 20
`user_name (record_learnr)`, 20

`webshot_shiny`, 28
`webshot_shiny()`, 12, 13, 28