

Package: datasuite.ui (via r-universe)

October 11, 2026

Title Shiny Interface, Chart Options and Report Builder for DataSuite Apps

Version 0.5.2

Description The shared interface of DataSuite's Shiny apps: chart options that restyle any 'ggplot2' chart (texts, fonts, colours, axes, legend, show or hide an element), a Word-like report builder writing Word, PowerPoint and PDF files, and the Shiny and React components, page frame and navigation the apps are built from. It holds nothing specific to one app's data: an app gives it its pages, its charts and its data through small interfaces.

License AGPL (>= 3)

URL <https://github.com/damurka/datasuite.ui>

BugReports <https://github.com/damurka/datasuite.ui/issues>

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1)

Imports cli (>= 3.6.2), flextable (>= 0.10.0), fontawesome (>= 0.5.3), ggplot2 (>= 3.5.1), graphics, grDevices, grid, htmltools (>= 0.5.9), jquerylib (>= 0.1.4), jsonlite (>= 2.0.0), openxlsx (>= 4.2.8.1), purrr (>= 1.2.2), quire (>= 0.2.17), rlang (>= 1.1.0), scales (>= 1.3.0), shiny (>= 1.14.0), shiny.i18n (>= 0.3.0), shiny.react (>= 0.4.0), stats, stringr (>= 1.6.0), tools, utils

Suggests dplyr (>= 1.2.1), countrycode (>= 1.8.0), magick (>= 2.9.1), officer (>= 0.7.5), pdftools (>= 3.9.0), png (>= 0.1-9), ragg (>= 1.5.2), rstudioapi (>= 0.19.0), rsvg (>= 2.7.0), svglite (>= 2.2.2), systemfonts (>= 1.3.2), testthat (>= 3.0.0), V8 (>= 8.2.0), xml2 (>= 1.6.0), zip (>= 3.0.0)

Config/datasuite/onDemand chromote, pdftools

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev zlib1g-dev

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

Date/Publication 2026-10-11 22:17:16 UTC

RemoteUrl <https://github.com/damurka/datasuite.ui>

RemoteRef main

RemoteSha d6a9546608811bc2c946909ac0af7ce709c436ee

Contents

datasuite.ui-package	3
ai_action	5
ai_register_component	6
ai_reply_when	7
apply_chart_options	7
as_report_context	8
cd_chart_options	8
cd_chart_type	14
cd_page_body	15
cd_start_pages	16
chart_facet_info	16
chart_option_fields	17
ds_host_request	18
export_deck	19
export_report	20
merge_chart_options	21
render_report_block	21
report_apply_transforms	22
report_asset_data_url	22
report_block_size	23
report_blocks	23
report_chart_fields	25
report_context	25
report_converter	26
report_deck_converter	26
report_deck_layouts	27
report_default_cover	27
report_default_design	28
report_field_catalog	28
report_fields	28
report_final_pages	29
report_flag_file	30
report_fonts	30
report_in	31
report_kind	31

report_list	32
report_merge	33
report_page	33
report_plot_spec	34
report_project_blocks	34
report_read	35
report_register	36
report_resolve_block	37
report_slide_size	37
report_store_asset	38
report_theme_from_file	38
report_themes	39
report_translate	40
report_tx	40
report_update_blocks	41
report_validate_project	42
report_validate_spec	43
resolve_chart_options	43
save_report_chart	44
with_report_chart_options	45
Index	46

datasuite.ui-package *datasuite.ui: Shiny Interface, Chart Options and Report Builder for DataSuite Apps*

Description

The shared interface of DataSuite's Shiny apps: chart options that restyle any 'ggplot2' chart (texts, fonts, colours, axes, legend, show or hide an element), a Word-like report builder writing Word, PowerPoint and PDF files, and the Shiny and React components, page frame and navigation the apps are built from. It holds nothing specific to one app's data: an app gives it its pages, its charts and its data through small interfaces.

Details

An app built on datasuite.ui gives it its pages, its data and its report content through small interfaces; the package itself knows nothing about any app's data and never calls an app's own package. The Countdown to 2030 apps are the working example: cd2030.core adds the Countdown pages and report content on top of it, and each app package (cd2030.rmncah, cd2030.vaxx, cd2030.pooled) has a run_app() built from both.

The app frame and components

`app_frame()` builds a whole app: the page, the sidebar and header, and the server. An app gives it a page registry (`cd_page_def()`, `cd_use_pages()`), a nav tree (`cd_nav_section()`, `cd_nav_item()`) and a data_server returning its dataset. A page is a module whose UI is `cd_page_ui()`, made of cards (`cd_card()`, `cd_chart_card()`), charts with their download and Customize tools (`cd_plot_ui()`, `cd_plot_server()`), inputs, dialogs and feedback. Every kit function is listed in `?"interface-kit"`; the package README (the dataset contract, a minimal app) and docs/COMPONENTS.md in the repository describe each one.

Chart options

`cd_chart_options()` describes how a ggplot2 chart should look (texts, fonts, colours, axes, legend, grid, facets, marks, and showing or hiding each element) and `apply_chart_options()` applies it to a finished chart. `merge_chart_options()`, `resolve_chart_options()`, `chart_option_fields()` and `cd_chart_type()` support them.

Reports

A report is blocks on pages with a theme and a cover, edited in the browser (the Reports page, `reports_ui()` / `reports_server()`) and exported with `export_report()` (Word, PDF) or `export_deck()` (PowerPoint, PDF). An app registers what its reports offer once with `report_register()` (kinds of chart and table, standard reports, themes, fields) and gives the engine one dataset's data as a `report_context()`, or through an `as_report_context()` method for its own dataset class. Themes: `report_themes()`, `report_theme_from_file()`; fields: `report_field_catalog()`.

Translations

Text is a translation key. `cd_translations()` merges the kit's keys, the files packages registered with `cd_register_translations()` and the app's own file for shiny.i18n::init_i18n(); `cd_use_i18n()` makes the translator the one the components use.

Author(s)

Maintainer: David Kariuki <hello@damurka.com> ([ORCID](#))

Other contributors:

- African Population and Health Research Center (APHRC) [copyright holder, funder]

See Also

Useful links:

- <https://github.com/damurka/datasuite.ui>
- Report bugs at <https://github.com/damurka/datasuite.ui/issues>

ai_action

*An action the AI can ask an app to do***Description**

Apps built with `app_frame()` answer requests from DataSuite's chat (see docs/AI-BRIDGE.md). The frame has the built-in actions (the page, its charts and tables, navigation); an app adds its own with `ai_action()`, passed in `app_frame(ai_actions = )`.

Usage

```
ai_action(
  name,
  fn,
  kind = c("read", "view", "add", "replace", "change"),
  description = "",
  args = list(),
  classify = NULL,
  summary = NULL
)
```

Arguments

name	The action's name, e.g. "setFilters".
fn	function(args, session): args is the named list the AI sent. Returns the result (anything JSON can hold: lists, vectors, data frames), or <code>ai_reply_when()</code> to answer once the browser has caught up. To refuse, stop with a message for the user.
kind	How much the action changes, which decides whether DataSuite asks the user first: • "read": looks, changes nothing the user sees. • "view": changes only what the app shows (a page, a tab, the filters). Never asked about. • "add": adds something new the user can remove (a saved report, a chart, a new file). Not asked about. • "replace": overwrites or deletes something saved. Asked about (setting <code>datasuite.shinyApps.aiAppControl</code>). • "change": the old name for a change of any size; treated as "replace". With the setting "off", DataSuite runs none but "read" actions.
description	One sentence telling the AI what the action does.
args	A named list describing each argument, name = "what it is".
classify	Optional function(args, session) returning the kind of one call when it depends on its arguments (e.g. "add" for a new report, "replace" for one that exists). By default, kind.

summary Optional function(args, session) returning one sentence saying what this call would do, for the user to confirm (e.g. Save the report "X" in data.rds). By default DataSuite shows the action and its arguments.

Value

An ai_action object.

Examples

```
ai_action("highlight", function(args, session) list(done = TRUE), kind = "view",
  description = "Highlight a region on the map", args = list(region = "a region name"))
```

ai_register_component *Make an output reachable by the AI*

Description

Registers a chart or table so DataSuite's chat can list it, read the data it shows and screenshot it (see docs/AI-BRIDGE.md). `cd_plot_server()` does this for every chart; call it for other outputs, e.g. a table, from their module server.

Usage

```
ai_register_component(
  session,
  output_id,
  type = c("chart", "table"),
  data,
  about = NULL,
  id = sub("-$", "", session$ns(""))
)
```

Arguments

session	The module's session.
output_id	The output's full id (session\$ns("table")), so the page can find it.
type	"chart" or "table".
data	A function (or reactive) returning the data frame the output shows. It is only called when the AI asks for the data, never to fill in the state.
about	What the output is, for the app's AI: NULL, a list, or a function (or reactive) returning one – e.g. list(kind = "coverage", options = list(indicator = "anc4")). Passed to DataSuite as it is.
id	The component's id; by default the module path (session\$ns("")) without its trailing -).

Value

NULL, invisibly.

ai_reply_when

Answer an AI request once the browser has caught up

Description

A change the app asks the browser to make (open a page, set a filter) only shows in input after a round trip. An [ai_action\(\)](#) returns `ai_reply_when()` to answer when `ready()` is TRUE, or after timeout seconds, with `result()`.

Usage

```
ai_reply_when(ready, result, timeout = 3)
```

Arguments

<code>ready</code>	A function returning TRUE once the change has arrived (it may read inputs).
<code>result</code>	A function returning the result to send.
<code>timeout</code>	Seconds to wait at most.

Value

An object [ai_action\(\)](#) functions return.

apply_chart_options

Apply chart options to a ggplot

Description

Changes only what options sets (see [cd_chart_options\(\)](#) for the list). The plot passed in is not modified. A plot that is not a ggplot (base graphics) is returned unchanged.

Usage

```
apply_chart_options(p, options = NULL)
```

Arguments

<code>p</code>	A ggplot.
<code>options</code>	A <code>cd_chart_options</code> object (or a list of its fields, or NULL).

Value

The ggplot.

as_report_context	<i>Make a report context</i>
-------------------	------------------------------

Description

What every report function calls on its first argument. A `report_context` is returned as it is and `NULL` gives an empty one; an app adds a method for its own dataset class.

Usage

```
as_report_context(x, ...)
```

Arguments

<code>x</code>	A report context, <code>NULL</code> , or an object an app has a method for.
<code>...</code>	Passed to methods.

Value

A `report_context`.

cd_chart_options	<i>Chart options</i>
------------------	----------------------

Description

Describes what a user can change about a chart: its texts, fonts, sizes, colours, angles, legend, axes, grid, backgrounds and how its lines, points and labels are drawn. Every plot method in the package accepts one through its `options` argument (and any of these fields as a named argument in `...`), and an app can keep them per chart, so the choices survive reloads.

Usage

```
cd_chart_options(
  ...,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  tag = NULL,
  x_title = NULL,
  y_title = NULL,
  legend_title = NULL,
  title_wrap = NULL,
  font_family = NULL,
  text_scale = NULL,
```

```
line_height = NULL,  
title_size = NULL,  
subtitle_size = NULL,  
caption_size = NULL,  
axis_title_size = NULL,  
axis_text_size = NULL,  
x_title_size = NULL,  
y_title_size = NULL,  
x_text_size = NULL,  
y_text_size = NULL,  
legend_title_size = NULL,  
legend_text_size = NULL,  
strip_text_size = NULL,  
title_face = NULL,  
subtitle_face = NULL,  
caption_face = NULL,  
axis_title_face = NULL,  
axis_text_face = NULL,  
legend_title_face = NULL,  
legend_text_face = NULL,  
strip_text_face = NULL,  
text_color = NULL,  
title_color = NULL,  
subtitle_color = NULL,  
caption_color = NULL,  
axis_title_color = NULL,  
axis_text_color = NULL,  
legend_title_color = NULL,  
legend_text_color = NULL,  
strip_text_color = NULL,  
title_align = NULL,  
subtitle_align = NULL,  
caption_align = NULL,  
title_position = NULL,  
caption_position = NULL,  
x_text_angle = NULL,  
y_text_angle = NULL,  
plot_margin = NULL,  
legend_position = NULL,  
legend_direction = NULL,  
legend_justification = NULL,  
legend_ncol = NULL,  
legend_nrow = NULL,  
legend_reverse = NULL,  
legend_key_size = NULL,  
legend_background = NULL,  
legend_labels = NULL,  
legend_label_wrap = NULL,
```

```
category_labels = NULL,  
category_label_wrap = NULL,  
x_limits = NULL,  
y_limits = NULL,  
x_labels = NULL,  
y_labels = NULL,  
axis_line = NULL,  
axis_line_color = NULL,  
axis_ticks = NULL,  
flip = NULL,  
grid = NULL,  
grid_minor = NULL,  
grid_color = NULL,  
grid_linewidth = NULL,  
grid_linetype = NULL,  
panel_border = NULL,  
panel_border_color = NULL,  
panel_color = NULL,  
background_color = NULL,  
strip_background = NULL,  
legend_border = NULL,  
legend_border_color = NULL,  
strip_border = NULL,  
strip_border_color = NULL,  
theme_preset = NULL,  
colors = NULL,  
line_scale = NULL,  
point_scale = NULL,  
alpha = NULL,  
bar_width = NULL,  
label_size = NULL,  
label_angle = NULL,  
label_color = NULL,  
facet_ncol = NULL,  
facet_nrow = NULL,  
facet_scales = NULL,  
strip_position = NULL,  
show_title = NULL,  
show_subtitle = NULL,  
show_caption = NULL,  
show_x_title = NULL,  
show_y_title = NULL,  
show_x_text = NULL,  
show_y_text = NULL,  
show_legend = NULL,  
show_legend_title = NULL,  
show_labels = NULL,  
show_strips = NULL
```

)

Arguments

... colours, the other spelling of colors. Nothing else is accepted.

title, subtitle, caption, tag
Chart texts.

x_title, y_title
Axis titles.

legend_title Legend title.

title_wrap Wrap the title, subtitle and caption after this many characters.

font_family A font family available on the machine drawing the chart (e.g. "Arial", "serif").

text_scale A multiplier for every text size in the chart (1.25 = 25 percent larger). Applied before the explicit sizes below.

line_height Line spacing of text (a multiple of the font size, default about 0.9).

title_size, subtitle_size, caption_size, axis_title_size,
axis_text_size, legend_title_size, legend_text_size, strip_text_size
Text sizes in points. axis_* sets both axes; use the x_/y_ ones below to set one.

x_title_size, y_title_size, x_text_size, y_text_size
Size of one axis' title or tick labels.

title_face, subtitle_face, caption_face, axis_title_face,
axis_text_face, legend_title_face, legend_text_face, strip_text_face
One of "plain", "bold", "italic", "bold.italic".

text_color Colour of all text; the *_color fields below override it for one kind of text.

title_color, subtitle_color, caption_color, axis_title_color,
axis_text_color, legend_title_color, legend_text_color,
strip_text_color
Colour of one kind of text.

title_align, subtitle_align, caption_align
One of "left", "center", "right".

title_position, caption_position
"panel" aligns the text with the plotting area, "plot" with the whole chart.

x_text_angle, y_text_angle
Angle of the tick labels in degrees (e.g. 45, 90); the alignment is adjusted so the labels stay attached to the axis.

plot_margin Space around the chart in points: one number, or four (top, right, bottom, left).

legend_position
One of "right", "bottom", "top", "left", "none".

legend_direction
"horizontal" or "vertical".

legend_justification
Where the legend sits along its side: "center", "top", "bottom", "left", "right".

legend_ncol, legend_nrow
Number of columns or rows the legend entries are laid out in.

legend_reverse	TRUE to reverse the order of the legend entries.
legend_key_size	Size of the legend keys in millimetres.
legend_background	Fill colour of the legend box.
legend_labels	A named character vector (or list): names are the legend entries the plot draws (or the text currently shown for them), values the text to show instead.
legend_label_wrap	Wrap the legend entries after this many characters.
category_labels	The same as legend_labels for the categories on a discrete axis.
category_label_wrap	Wrap the category labels after this many characters.
x_limits, y_limits	Two numbers (NA leaves an end open): the range to show. Zooms; data is not dropped.
x_labels, y_labels	How numbers on a continuous axis are written: "number", "comma", "percent" (0.5 becomes 50%), "percent_points" (50 becomes 50%), "scientific", "compact" (1.2K, 3M).
axis_line	TRUE/FALSE to draw or hide the axis lines.
axis_line_color	Colour of the axis lines.
axis_ticks	TRUE/FALSE to draw or hide the tick marks.
flip	TRUE to swap the axes, FALSE to keep the chart as drawn, NULL for automatic.
grid	Which grid lines to show: "both", "horizontal", "vertical" or "none".
grid_minor	TRUE/FALSE to show or hide the minor grid lines.
grid_color, grid_linewidth, grid_linetype	Style of the grid lines (linetype is one of "solid", "dashed", "dotted", "dotdash", "longdash", "twodash").
panel_border	TRUE/FALSE to draw or hide a border around the plotting area.
panel_border_color	Colour of that border.
panel_color	Fill colour of the plotting area.
background_color	Fill colour of the whole chart.
strip_background	Fill colour of facet strips.
legend_border, strip_border	TRUE/FALSE to draw or hide a border around the legend or around each facet strip (panel heading).
legend_border_color, strip_border_color	Colour of that border.

theme_preset	Start from a different look before the other options are applied: "grey", "minimal", "classic", "bw", "light", "linedraw", "dark", "void". It replaces the plot's own theme.
colors	A named character vector of colours for a colour or fill legend (names are entries as in legend_labels); entries not named keep their colour.
line_scale, point_scale	Multiply the width of lines and the size of points (1.5 = 50 percent bigger).
alpha	Transparency of every mark, from 0 (invisible) to 1.
bar_width	Width of the bars (and boxes) as a share of the space each has, above 0 and up to 1 (0.9 is ggplot2's default).
label_size	Size in points of text drawn on the chart (data labels).
label_angle, label_color	Angle and colour of that text.
facet_ncol, facet_nrow	For a chart split into panels with facet_wrap(): the number of columns or rows of panels (a whole number from 1 to 20). Setting only one lets ggplot2 work out the other. Charts split with facet_grid() keep their layout (one row or column per value).
facet_scales	Whether panels share their axes: "fixed" (all the same), "free" (each its own), "free_x" or "free_y" (only that axis varies). Ignored on maps and other charts whose coordinates cannot vary by panel.
strip_position	Where the panel labels (strips) go: "top", "bottom", "left" or "right". With facet_grid(), "top"/"bottom" place the column labels and "left"/"right" the row labels. Use chart_facet_info() to see whether a chart has panels at all.
show_title, show_subtitle, show_caption, show_x_title, show_y_title, show_x_text, show_y_text, show_legend, show_legend_title, show_labels, show_strips	Show or hide one element of the chart: the title, subtitle, caption, an axis title, an axis' tick labels (show_x_text, show_y_text), the whole legend, the legend's title, the data labels drawn on the chart (geom_text(), geom_label() and their ggrepel versions) or the panel names of a chart split into panels (show_strips). FALSE hides it and leaves no space for it; TRUE shows it again where the chart itself hid it (a legend the chart put at "none" goes to the right, a text the chart's theme blanked is drawn), which only works when the chart has that text at all; NULL keeps what the chart draws. An empty text (title = "", x_title = "", legend_title = "...") hides that element the same way.

Details

A field left NULL means "keep what the plot draws by default"; only what is set is changed. Text fields accept "", which hides that text and the space it took (as the show_* fields do). Colours are anything R understands ("red", "#1b7837", "transparent"). Sizes are in points unless said otherwise. x_* and y_* always mean the data's x and y aesthetics, so they follow the data when the chart is flipped.

Value

An object of class `cd_chart_options` (a list of the fields that are set).

See Also

`apply_chart_options()`, `merge_chart_options()`

Examples

```
cd_chart_options(title = "Coverage", legend_position = "bottom", text_scale = 1.2)
cd_chart_options(x_title = "Year", legend_labels = c(dhis2 = "Routine data"))
cd_chart_options(x_text_angle = 45, grid = "horizontal", grid_color = "grey90",
                 y_labels = "percent_points")
```

`cd_chart_type`

The type of graph a plot is

Description

A key made from the geoms a ggplot is built from (lines and points, bars, tiles...), used for a chart that has no id of its own.

Usage

```
cd_chart_type(p)
```

Arguments

`p` A ggplot.

Value

A string such as `"geom:Errorbar+Line+Point"`, or `NULL` when `p` is not a ggplot with layers.

cd_page_body	<i>A whole page: filter bar, page header, denominator row, content</i>
--------------	--

Description

The page assembled from its parts. Modules do not call this directly; they call cd_page_ui(), which fills in the title, section and subtitle from the page registry.

Usage

```
cd_page_body(  
    dashboardId,  
    dashboardTitle,  
    i18n,  
    ...,  
    filters = NULL,  
    include_report = FALSE,  
    include_notes = FALSE,  
    include_help = TRUE,  
    eyebrow = NULL,  
    subtitle = NULL,  
    include_denominator = FALSE  
)
```

Arguments

- dashboardId Unique module ID
- dashboardTitle Localized dashboard title
- i18n Translator object
- ... Page content, passed to cd_page_content()
- filters Optional cd_filter_bar(...) with the page's filter inputs
- include_report, include_notes, include_help
 Show the header's report, notes and help buttons.
- eyebrow The small label above the title (a translation key).
- subtitle The line under the title (a translation key).
- include_denominator
 Show the row the app's page header server fills (Countdown: the denominators).

cd_start_pages	<i>Start the servers of pages not opened yet</i>
----------------	--

Description

An app starts a page's server the first time the page is opened (`cd_pages_server()`). What needs a page that may not have been opened (the AI reading a chart on another page) starts it with this. Nothing happens for a page already started, or in an app whose pages are all started with the session.

Usage

```
cd_start_pages(session = getDefaultReactiveDomain(), which = NULL)
```

Arguments

session	The Shiny session.
which	The ids of the pages to start; every page when NULL.

Value

NULL, invisibly.

chart_facet_info	<i>Whether a chart is split into panels, and how</i>
------------------	--

Description

Tells whether a ggplot is split into panels (`facet_wrap()` or `facet_grid()`) and how they are laid out now, so an editor can offer the panel options of `cd_chart_options()` (`facet_ncol`, `facet_nrow`, `facet_scales`, `strip_position`) only for charts that have panels.

Usage

```
chart_facet_info(plot)
```

Arguments

plot	A ggplot (anything else gives NULL).
------	--------------------------------------

Value

NULL when the chart has no panels. Otherwise a list with

- type: "wrap" or "grid";
- ncol, nrow: the number of columns and rows of panels as drawn (integer; NA when the plot cannot be built and the facet does not say);
- scales: "fixed", "free", "free_x" or "free_y";
- strip_position: "top", "bottom", "left" or "right" (for facet_grid(), where the column labels are, or the row labels when it only has rows);
- panels: the number of panels (integer, NA when the plot cannot be built).

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(wt, mpg)) + geom_point() + facet_wrap(~cyl, ncol = 2)
chart_facet_info(p)
chart_facet_info(ggplot(mtcars, aes(wt, mpg)) + geom_point())
```

chart_option_fields	<i>The names of the chart options</i>
---------------------	---------------------------------------

Description

Every field `cd_chart_options()` accepts, for code that keeps only the chart options out of a list of settings.

Usage

```
chart_option_fields()
```

Value

A character vector.

ds_host_request	<i>Ask DataSuite for something</i>
-----------------	------------------------------------

Description

What an app asks of DataSuite, the application running it: open the chat, print a page, install packages. Inside a Jovian R kernel whose host answers it (DataSuite starts the app with `CDSUITE_HOST_UI=1`), the request goes on Jovian's host channel as the method `"datasuite.<action>"` (`hera::host_notify()`, or `hera::host_ask()` to wait for the answer while the app keeps running); otherwise it is one line on the R session's output, `DATASUITE_HOST_REQUEST <json>`, with the action in the JSON object beside `args`, and an answer is read from a done file named in the request. The requests, their arguments and answers are described in the package's docs/`HOST-REQUESTS.md`.

Usage

```
ds_host_request(action, args = list(), wait = FALSE, timeout = 120)
```

Arguments

<code>action</code>	The request: <code>"openChat"</code> , <code>"print"</code> , <code>"installPackages"</code> .
<code>args</code>	Its arguments: a named list (a JSON object; give an array as a list, so that one element stays an array).
<code>wait</code>	TRUE to wait for DataSuite's answer.
<code>timeout</code>	For <code>wait = TRUE</code> on the line protocol: how long to wait for the answer, in seconds.

Details

It asks whenever it is called: a caller checks first that the app runs inside DataSuite (`CDSUITE_SHINY_ID` is set, or `CDSUITE_PRINT` for printing).

Value

With `wait = FALSE`, TRUE invisibly once the request is sent. With `wait = TRUE`, DataSuite's answer (a list, as in the done file: ok, and error when it failed), or NULL when it gave none.

Examples

```
## Not run:
ds_host_request("openChat", list(query = "Explain this chart: "))

## End(Not run)
```

export_deck

Export a slide deck

Description

Draws every chart and table of a deck from the loaded data and writes it as a PowerPoint file, or as a PDF made from that file (by PowerPoint or LibreOffice, see [report_deck_converter\(\)](#)), with the report builder's own writers (Quire's, through [quire::quire_export\(\)](#)). Every item is placed at its box: text as text boxes in the design's fonts and colours, charts and pictures as pictures, tables as PowerPoint tables. The slides' speaker notes are kept.

Usage

```
export_deck(
  context,
  project,
  file,
  format = c("pptx", "pdf"),
  i18n = NULL,
  progress = NULL,
  converter = c("auto", "powerpoint", "libreoffice")
)
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
project	The deck (see the top of R/report-deck.R): kind = "deck", name, design, region, slides.
file	The file to write.
format	"pptx" or "pdf".
i18n	An object with a t(key) method, or NULL.
progress	A function called with a number between 0 and 1 as the export advances, or NULL.
converter	What makes the PDF: "auto" (PowerPoint, else LibreOffice), "powerpoint" or "libreoffice".

Value

file, invisibly, with attribute converter (what made the PDF; NULL for a PowerPoint file).

export_report

Export a report built in the report builder

Description

Draws every chart and table from the loaded data and writes the report as a Word document or a PDF, with the report builder's own writers (Quire's, through `quire:quire_export()`). The PDF is made from the Word file (see `report_converter()`), so the two match page for page. A slide deck (`kind = "deck"`) is written by `export_deck()`, as a PowerPoint file or a PDF.

Usage

```
export_report(
  context,
  project,
  file,
  format = c("docx", "pdf", "pptx"),
  i18n = NULL,
  subtitle = NULL,
  progress = NULL,
  converter = c("auto", "word", "libreoffice", "browser", "powerpoint")
)
```

Arguments

context	A report context (<code>report_context()</code>), or what <code>as_report_context()</code> turns into one.
project	The report: <code>list(name, design, cover, blocks)</code> (such as the app's standard reports, see <code>report_register()</code>), or a slide deck.
file	The file to write.
format	"docx" or "pdf" for a document; "pptx" or "pdf" for a slide deck (by default "docx" for a document, "pptx" for a deck).
i18n	An object with a <code>t(key)</code> method, or NULL.
subtitle	Not used any more: the cover page has its own subtitle. Kept so older calls work.
progress	A function called with a number between 0 and 1 as the export advances, or NULL.
converter	"auto" (Word, else LibreOffice, else a browser), or one of "word", "libreoffice", "browser"; for a deck "auto", "powerpoint" or "libreoffice" ("word" is taken as PowerPoint).

Value

file, invisibly, with attribute `converter`: what made the PDF ("word", "libreoffice", "browser", "powerpoint") or, for Word, what finished the file ("word", or NULL).

merge_chart_options	<i>Combine chart options</i>
---------------------	------------------------------

Description

Later arguments win, field by field; legend_labels, category_labels and colors combine name by name. Use it to layer a dataset's defaults, a chart's saved options and what the caller passes now.

Usage

```
merge_chart_options(...)
```

Arguments

... cd_chart_options objects, lists of fields, or NULLs.

Value

A cd_chart_options object.

Examples

```
merge_chart_options(cd_chart_options(font_family = "serif", title = "A"),
                    cd_chart_options(title = "B"))
```

render_report_block	<i>Draw one chart or table of a report</i>
---------------------	--

Description

Draw one chart or table of a report

Usage

```
render_report_block(context, block, i18n = NULL, design = NULL)
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
block	A chart or table block (see the top of this file for its fields). A region of " <code>@report</code> " is the report's region: the caller puts it in first (report_resolve_block()).
i18n	An object with a <code>t(key)</code> method for translated labels, or NULL for English.
design	The report's design: its palette colours the chart's series (see report_themes()). NULL keeps the chart's own colours.

Value

list(type = "plot", value = <ggplot>), list(type = "table", value = <flextable>), or list(type = "error", message = <text>) when it cannot be drawn with this dataset.

report_apply_transforms

Apply a custom chart's transforms to its data

Description

Apply a custom chart's transforms to its data

Usage

```
report_apply_transforms(data, transform = list())
```

Arguments

data	A data frame.
transform	The spec's transform steps (see report_validate_spec()).

Value

The data frame, transformed. Stops with a message naming a missing column.

Examples

```
d <- data.frame(region = c("A", "B", "A"), year = c(2022, 2022, 2023), anc4 = c(50, 60, 55))
report_apply_transforms(d, list(list(filter = list(year = 2022)),
                                list(select = c("region", "anc4"))))
```

report_asset_data_url *A report picture as a data URL*

Description

The picture a block refers to as "asset:<id>" (see `CacheConnection$set_report_asset()`), for showing in a browser.

Usage

```
report_asset_data_url(context, id)
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
id	The picture's id (with or without "asset:").

Value

A data: URL, or NULL when the dataset has no such picture.

report_block_size	<i>The size a chart or image block is shown at, in inches</i>
-------------------	---

Description

Worked out from the page (see [report_page\(\)](#)): a full-width block spans the text, a half-width one half of it. Shared by the builder's previews and by the exported files, so the page on screen and the file agree. A block on a slide carries its box, box = c(width, height) in inches (see [report_project_blocks\(\)](#)): it is drawn at that size.

Usage

```
report_block_size(block, design = NULL)
```

Arguments

block	A chart or image block.
design	The report's design (NULL: the default page).

Value

c(width, height) in inches.

report_blocks	<i>The blocks of a report</i>
---------------	-------------------------------

Description

What a standard report is written with. Texts may be plain strings or [report_tx\(\)](#).

Usage

```

report_heading(text, level = 1)

report_paragraph(text)

report_note(text)

report_page_break()

report_chart(
  kind,
  indicator = NULL,
  admin_level = "national",
  size = "full",
  variant = NULL,
  type = "chart",
  region = NULL,
  year = NULL,
  title = NULL
)

report_table(kind, ...)

report_questions(
  ...,
  title = report_tx("Interpretations", "Interprétations", "Interpretações")
)

```

Arguments

text	The text: plain for a heading, formatted (a small subset of HTML) for a paragraph or note.
level	A heading's level (1 or 2).
kind, indicator, admin_level, size, variant, region, year, title	A chart's or table's kind (one of the registered kinds, see report_register()) and what to draw; size is "full", "half" or "third".
type	"chart" or "table".
...	For <code>report_table()</code> , passed to <code>report_chart()</code> ; for <code>report_questions()</code> , the questions (strings or report_tx()), one per line.

Value

A block (a list).

report_chart_fields	<i>Fill the fields that describe the chart below a text</i>
---------------------	---

Description

{chart_indicator} and {chart_year} in a text, heading or caption are the indicator (its name, in the report's language) and the year of the first chart or table after it in the report, so a title above a chart follows the chart when its indicator or year is changed. {chart_indicators} names the indicators of all the charts and tables after it up to the next heading ("ANC 4 Visits and Measles 1"), for a heading above charts side by side. A text with no chart after it keeps the field as typed.

Usage

```
report_chart_fields(blocks, context = NULL, i18n = NULL)
```

Arguments

blocks	The report's blocks, in order.
context	A report context (report_context()), or what as_report_context() turns into one.
i18n	The report's translations (NULL: English).

Value

The blocks, their texts filled.

report_context	<i>A report context: one dataset's side of the report builder</i>
----------------	---

Description

Each argument is a function; those left out do nothing (no pictures kept, no saved chart options, no fields).

Usage

```
report_context(
  draw = NULL,
  asset_get = NULL,
  asset_set = NULL,
  chart_options = NULL,
  fields = NULL,
  years = NULL,
  regions = NULL,
  flag = NULL
)
```

Arguments

draw	function(block, i18n) returning a ggplot or a flextable for a chart or table block (an error when it cannot be drawn).
asset_get	function(id) returning a kept picture or template, list(type, data = <raw>), or NULL.
asset_set	function(id, value) keeping one.
chart_options	function() returning the chart options saved with the data: a named list, "default" for every chart and one per chart id or type of graph (cd_chart_type()).
fields	function(project, lang) returning a named list of field values (e.g. country).
years	function() returning the years the data covers.
regions	function() returning the regions a report can be about.
flag	function() returning the path of a flag picture for the cover, or NULL.

Value

An object of class report_context.

report_converter	<i>The program that turns the Word file into a PDF</i>
------------------	--

Description

Microsoft Word (Windows) or LibreOffice. With one of them the PDF has exactly the Word file's pages. (Quire's: [quire::quire_converter\(\)](#).)

Usage

```
report_converter()
```

Value

"word", "libreoffice", or NULL when neither is installed.

report_deck_converter	<i>The program that turns a slide deck into a PDF</i>
-----------------------	---

Description

Microsoft PowerPoint (Windows) or LibreOffice. (Quire's: [quire::quire_converter\(\)](#).)

Usage

```
report_deck_converter()
```

Value

"powerpoint", "libreoffice", or NULL when neither is installed.

report_deck_layouts	<i>The layouts of a slide</i>
---------------------	-------------------------------

Description

Where a new slide's placeholders go, as PowerPoint places them: each layout lists its boxes as fractions (0 to 1) of the slide's width and height, so they fit a 16:9 or a 4:3 slide.

Usage

```
report_deck_layouts(lang = NULL)
```

Arguments

lang	NULL for the names in the three languages (list(en, fr, pt)), or "en", "fr", "pt" for one.
------	--

Value

A list of layouts, each list(id, name, items); each item is list(role, type, x, y, w, h) with role "title", "subtitle", "heading", "body" or "picture" (a picture placeholder: nothing is written for it until a picture is put in it), and type "text" (a text box) or "content" (a text, chart, table or picture).

report_default_cover	<i>The default cover page</i>
----------------------	-------------------------------

Description

The default cover page

Usage

```
report_default_cover()
```

Value

A cover (see the top of R/report-theme.R for its fields).

`report_default_design` *The default report design*

Description

The default report design

Usage

```
report_default_design()
```

Value

A design (see the top of `R/report-theme.R` for its fields).

`report_field_catalog` *The fields a report's text can contain*

Description

The report's own (title, date, editors, region, years, the chart below), then those the app registered ([report_register\(\)](#), fields).

Usage

```
report_field_catalog()
```

Value

A list of `list(key, group, label)`; key is the word written in braces.

`report_fields` *The values of a report's fields*

Description

The values of a report's fields

Usage

```
report_fields(context, project, date = Sys.Date(), lang = "en")
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
project	The report (name, cover).
date	The date the report is dated.
lang	The report's language ("en", "fr", "pt"): the date is written in it.

Value

A named list of strings, one per field in [report_field_catalog\(\)](#) that has a value.

report_final_pages	<i>The pages of a report as they will be printed</i>
--------------------	--

Description

Writes the Word file, makes the PDF from it and renders each page as a picture: what the builder's "Final pages" view shows. Needs Microsoft Word or LibreOffice ([report_converter\(\)](#)). For a slide deck, the PowerPoint file and its PDF (PowerPoint or LibreOffice, [report_deck_converter\(\)](#)): one picture per slide.

Usage

```
report_final_pages(context, project, i18n = NULL, dpi = 60, progress = NULL)
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
project	The report: <code>list(name, design, cover, blocks)</code> (such as the app's standard reports, see report_register()), or a slide deck.
i18n	An object with a <code>t(key)</code> method, or <code>NULL</code> .
dpi	Resolution of the page pictures.
progress	A function called with a number between 0 and 1 as the export advances, or <code>NULL</code> .

Value

`list(pages = <character vector of PNG files>, converter, pdf = <path>)`. The files are in a temporary folder the caller removes.

report_flag_file	<i>A country's flag</i>
------------------	-------------------------

Description

Downloaded once from flagcdn.com and kept in the package's context folder. NULL when the country is not known or the flag cannot be downloaded (no internet): the cover is then drawn without it.

Usage

```
report_flag_file(iso3)
```

Arguments

iso3	The country's ISO3 code.
------	--------------------------

Value

The path of a PNG file, or NULL.

report_fonts	<i>The fonts a report can use</i>
--------------	-----------------------------------

Description

Typefaces made for print that come with Microsoft Office, so the page in the editor, the charts, the Word file and the PDF all use the same letters, on any computer that opens the file: serif faces for reading text, then sans serif faces for charts, tables and headings. Only those installed where the charts are drawn are given (all of them when that cannot be told).

Usage

```
report_fonts(group = c("all", "serif", "sans"), installed = TRUE)
```

Arguments

group	"all", "serif" or "sans".
installed	FALSE for every font a report may name, installed here or not.

Value

A character vector.

report_in	<i>A text, block or report in one language</i>
-----------	--

Description

Every `report_tx()` in `x` (a block, a list of blocks, a cover...) replaced by its text in `lang`.

Usage

```
report_in(x, lang = "en")
```

Arguments

<code>x</code>	Anything holding texts made with <code>report_tx()</code> .
<code>lang</code>	The language.

Value

`x` in that language.

report_kind	<i>One kind of chart or table</i>
-------------	-----------------------------------

Description

One kind of chart or table

Usage

```
report_kind(  
  type,  
  group,  
  label,  
  indicators = NULL,  
  levels = NULL,  
  variants = NULL,  
  year = FALSE,  
  regional = FALSE,  
  groups = NULL,  
  tall = FALSE  
)
```

Arguments

type	"chart" or "table".
group	The group it is shown under in the builder.
label	Its name.
indicators	The indicators it can be drawn for (a character vector, an app-defined keyword, or NULL).
levels, variants	The levels (e.g. national, regional) and variants it offers.
year	Whether a year is chosen.
regional	Whether it is drawn for one region.
groups	The app's groups of indicators it is for.
tall	Whether it is drawn taller than wide (maps, charts with a panel per district), see report_block_size() .

Value

A list.

report_list	<i>The saved reports, as the AI lists them</i>
-------------	--

Description

The saved reports, as the AI lists them

Usage

```
report_list(projects)
```

Arguments

projects	The saved reports, a named list by id (a dataset's report_projects).
----------	--

Value

A list, the last edited first, of list(id, name, kind, lang, blocks, updated); kind is "report" (a document) or "deck" (a slide deck), blocks the number of blocks (a deck's: of its slides' items).

Examples

```
report_list(list(r1 = list(name = "Coverage", blocks = list(report_paragraph("Hello")))))
```

report_merge	<i>Two designs (or covers) merged</i>
--------------	---------------------------------------

Description

Field by field, b over a: lists such as a cover's editors or logos are replaced whole, not merged item by item.

Usage

```
report_merge(a, b)
```

Arguments

a, b Two designs or covers.

Value

The merged list.

report_page	<i>The page of a report, in inches</i>
-------------	--

Description

Page size, margins and the width and height left for content. Shared by the builder and the exported files.

Usage

```
report_page(design = report_default_design())
```

Arguments

design A design.

Value

A list: width, height, top, bottom, left, right, text_width, text_height.

report_plot_spec	<i>Draw a custom chart's plot description</i>
------------------	---

Description

Generic: a table in, a ggplot out. Chart options (the customize panel) apply to the result like to any other chart.

Usage

```
report_plot_spec(data, plot, title = NULL)
```

Arguments

data	The (transformed) data frame.
plot	The plot description (see report_validate_spec()): geom (line, col, point, area, tile), x, y, and optionally colour, fill, facet, position (dodge, stack, fill), labels (list(x, y, title, colour, fill)), percent (the y values are percentages, 0 to 100) and flip (bars go across).
title	A title, used when plot\$labels\$title isn't given.

Value

A ggplot.

Examples

```
d <- data.frame(region = c("A", "B"), value = c(55, 61))
report_plot_spec(d, list(geom = "col", x = "region", y = "value", percent = TRUE))
```

report_project_blocks	<i>Every block of a report</i>
-----------------------	--------------------------------

Description

A document's blocks, or every item's block of a slide deck, in order, so a caller can count, check or draw them the same way for both. A slide item's block gets the item's id and its box (box = c(width, height), inches), which [report_block_size\(\)](#) draws it at. A document's free-layout page (a block of type "canvas", whose items are placed like a slide's) is given as its items' blocks, in the same way; the canvas block itself is not in the list.

Usage

```
report_project_blocks(project)
```

Arguments

project A report (kind = "deck" for a slide deck; anything else is a document).

Value

A list of blocks.

report_read	<i>A saved report, as the AI reads it</i>
-------------	---

Description

Every block in order, compactly: its id and type; a heading's, paragraph's or note's text (plain, the formatting left out); a chart's or table's kind, settings, chart options and layout, and – with data = TRUE – a short table of what it shows (drawn as the report draws it: the chart's title, subtitle and caption, and the values it plots, or the table's rows), so the AI can write about it from its numbers.

Usage

```
report_read(  
  context,  
  project,  
  id = project$id,  
  i18n = NULL,  
  lang = "en",  
  data = TRUE,  
  max_rows = 25,  
  max_cols = 8,  
  max_chars = 2000  
)
```

Arguments

context A report context ([report_context\(\)](#)), or what [as_report_context\(\)](#) turns into one.

project The report.

id The report's id (by default its own).

i18n A translator (t(key)) the charts are drawn with. By default the app's, in the report's language.

lang The language a report without one of its own is written in (it follows the app's).

data Whether to draw the charts and tables for their data.

max_rows, max_cols The most rows and columns of each chart's or table's data.

max_chars The most characters of each text.

Value

list(id, name, kind, lang, region, blocks). A slide deck's blocks are its slides' items (each with its slide); a document's free page (canvas) lists its items' blocks in items.

report_register	<i>Register what an app's reports offer</i>
-----------------	---

Description

Called once by an app package (usually in its .onLoad()); later calls replace only what they give.

Usage

```
report_register(  
  themes = NULL,  
  default_theme = NULL,  
  cover = NULL,  
  kinds = NULL,  
  presets = NULL,  
  indicator_name = NULL,  
  fields = NULL,  
  chart_id = NULL  
)
```

Arguments

themes	A named list of designs (see report_default_design()), each with name (a string, or a list of en, fr, pt): offered beside the built-in themes, first.
default_theme	The id of the theme new reports start from.
cover	A list of cover fields that new reports start from (see report_default_cover()), e.g. a kicker.
kinds	A function returning the kinds of chart and table, a named list of report_kind() .
presets	A function of lang returning the standard reports (see the app's documentation of its own).
indicator_name	A function of indicator and i18n returning an indicator's name in the report's language, for {chart_indicator}.
fields	A list of extra fields for report_field_catalog() , each list(key, group, label); their values come from the report context's fields().
chart_id	A function of a chart's data and its ggplot returning the chart's id: the key its options are kept under, for the screen and for reports (the same chart gets the same id in both). By default, cd_chart_type() .

Value

Invisibly, NULL.

report_resolve_block	<i>A block as it is drawn in a report</i>
----------------------	---

Description

Puts the report's region in a block that asks for it (region = "@report"). A report without a region is a national report: such a block is then drawn for the whole country (or asks for a region when it needs one).

Usage

```
report_resolve_block(block, project, regions = NULL)
```

Arguments

block	A block.
project	The report (its region; NULL or "": national).
regions	Not used; kept for older callers.

Value

The block.

report_slide_size	<i>The size of a slide deck's slides, in inches</i>
-------------------	---

Description

The size of a slide deck's slides, in inches

Usage

```
report_slide_size(design = NULL)
```

Arguments

design	A design; its slide_size is "16:9" (13.333 x 7.5 in, the default) or "4:3" (10 x 7.5 in).
--------	---

Value

c(width, height) in inches.

report_store_asset	<i>Store a picture for the reports</i>
--------------------	--

Description

Keeps a picture once in the dataset, from a data: URL (as the report builder uploads it) or from a web address (downloaded now, so the report does not need the internet later). A picture larger than 4000 pixels is made smaller; formats Word cannot show (WebP, for example) are converted to PNG.

Usage

```
report_store_asset(context, id, src)
```

Arguments

context	A report context (report_context()), or what as_report_context() turns into one.
id	The picture's id.
src	A data: URL, or an http(s):// address.

Value

A list: id, ratio (height / width) and url (a data: URL of the picture as stored), invisibly.

report_theme_from_file	<i>A report theme from a PowerPoint or Word file</i>
------------------------	--

Description

Reads the colours and fonts of an Office theme or template (PowerPoint .pptx / .potx, Word .docx / .dotx) into a report theme, as [report_themes\(\)](#) gives them: the accent is the theme's first accent colour, the text colour its first dark colour, the chart palette its six accent colours, the heading and body fonts its heading and body fonts. A Word file's styles give the heading colour (Heading 1's), the fonts and the text sizes when they set them, and its page size, orientation and margins. A PowerPoint file's master gives the title colour and fonts, the slide size and the background colour. The reading is Quire's ([quire::quire_theme_from_file\(\)](#)).

Usage

```
report_theme_from_file(path, name = NULL)
```

Arguments

path	The file.
name	The theme's name (by default the file's name without its extension).

Details

The same file can be given to the export as the design's template (its path): see [export_deck\(\)](#) and [export_report\(\)](#).

Value

A design (see [report_default_design\(\)](#)) with theme ("custom_<hash of the file>"), name (list(en, fr, pt)), and template_kind ("pptx" or "docx"); for a PowerPoint file also slide_size ("16:9" or "4:3", the nearer), background (the master's background colour, or NULL) and slide_designs: list(title, content), the look of the file's title slide and of its other slides (either NULL when not found). A slide design is list(background, decor, title, subtitle, body): background a colour or NULL; decor what is drawn behind a slide's content, back first, each list(type = "image", file, x, y, w, h) (a logo: file is the picture, copied out of the file into a temporary folder) or list(type = "rect", fill, fill_opacity, outline, x, y, w, h) (a band or bar), each also with from ("master", "layout" or "slide": where it was found); title, subtitle, body the style of that text, list(x, y, w, h, fill, fill_opacity, color, font_size, bold, align, font), or NULL. Boxes are in inches from the slide's top-left corner; colours are "#rrggbb". [export_deck\(\)](#) draws a design's background and decor on its slides.

report_themes	<i>The report themes</i>
---------------	--------------------------

Description

The ready-made looks a report can start from. Each is a complete design (see [report_default_design\(\)](#)).

Usage

```
report_themes()
```

Value

A named list of designs; each also has name.

report_translate	<i>A translated string</i>
------------------	----------------------------

Description

A translated string

Usage

```
report_translate(i18n, key, fallback = key)
```

Arguments

- i18n An object with a `t(key)` method (an app's translator), or NULL.
- key The key.
- fallback What to give when there is no translation (or no `i18n`).

Value

A string.

report_tx	<i>Text in several languages</i>
-----------	----------------------------------

Description

A standard report is written once, in the app's languages; `report_in()` gives it in one.

Usage

```
report_tx(en, fr = en, pt = en)
```

Arguments

- en, fr, pt The text in English, French and Portuguese (French and Portuguese default to the English).

Value

An object of class `rb_tx`.

report_update_blocks *Change blocks of a saved report*

Description

Applies targeted changes, in order, to a report's blocks, and checks the result as a whole ([report_validate_project\(\)](#)). Only the blocks named change; the design, the cover and every other block stay as they are. Each change is one of:

Usage

```
report_update_blocks(
    project,
    changes,
    kinds = .ds_report_kinds(),
    members = NULL
)
```

Arguments

project	The report.
changes	A list of changes.
kinds	The kinds of chart and table (a named list of report_kind()); by default the app's.
members	Passed to report_validate_project() for custom charts.

Details

- `list(blockId, <fields>)`: changes that block.
 - A heading, paragraph or note: text (plain text; a line break starts a new line, **bold** is bold), type ("heading", "paragraph" or "note"), level (a heading's, 1 to 6), align, space_before, space_after (pt).
 - A chart or table: kind (another kind; the settings the new kind does not take are left out, those it needs get its defaults), its settings indicator, admin_level, region, year, variant (as the kind allows; NULL removes one), options (chart options, see [cd_chart_options\(\)](#); merged into the chart's own, NULL removes one), size ("full", "half", "third"), title (NULL: the chart's own), caption (FALSE hides the chart's caption).
 - A picture: caption, alt, size, width (percent of its column), align.
 - Any block of a document: pageBreakBefore (TRUE starts a new page before it, FALSE removes that).
- `list(afterBlockId, insert = <block>)`: a new heading, paragraph, note, page break, chart or table after that block ("`@start`": at the start). Several inserted after the same block keep their order.
- `list(blockId, delete = TRUE)`: removes the block.
- `list(blockId, moveAfter = <id>)`: moves the block after another ("`@start`": to the start).

A slide deck's items can be changed, but not added, removed or moved.

Value

`list(project, changes)`: the changed report (without its sigs on changed charts: the builder signs them again) and one sentence per change, saying what it did.

Examples

```
p <- list(name = "Notes", blocks = list(list(id = "b1", type = "paragraph", text = "")))
report_update_blocks(p, list(list(blockId = "b1", text = "Coverage rose."),
                             list(afterBlockId = "b1", insert = list(type = "heading", text = "Next steps"))))$changes
```

report_validate_project

Check a report before it is saved

Description

Checks the shape of a report project (see [export_report\(\)](#)) – one the AI built, say – before an app saves it: each block's type, and for charts and tables their kind. A `custom_chart` block may carry its description in `spec` or, as the AI bridge sends it, in `options`; either way it is checked with [report_validate_spec\(\)](#) and kept in `spec`.

Usage

```
report_validate_project(project, kinds = NULL, members = NULL)
```

Arguments

<code>project</code>	<code>list(name, design, cover, blocks)</code> .
<code>kinds</code>	The report kinds charts and tables may use (names), or <code>NULL</code> not to check them.
<code>members</code>	Passed to report_validate_spec() for custom charts.

Value

The project, tidied (each block has an id). Stops with a message saying what is wrong.

Examples

```
report_validate_project(list(name = "Notes", blocks = list(
  list(type = "heading", text = "Summary"),
  list(type = "paragraph", text = "Coverage rose.")
)))
```

report_validate_spec *Check a custom chart description*

Description

A custom chart (report kind "custom_chart") is described by data, never code: where its table comes from, a few fixed transforms and a plot description. report_validate_spec() checks one before it is saved, and returns it tidied (lists from JSON turned into vectors where a vector is meant).

Usage

```
report_validate_spec(spec, members = NULL)
```

Arguments

spec	The description: list(title, data = list(member, args), transform, plot). transform is a list of one-entry lists, each one of filter (list(column = values)), select (column names), rename (list(new = "old")), mutate_ratio (list(name, numerator, denominator, percent)), aggregate (list(by, fun, cols); fun one of sum, mean, median, min, max), pivot_longer (list(cols, names_to, values_to)), arrange (list(by, desc)), top_n (list(n, by, desc)). plot is list(geom, x, y, colour, fill, facet, position, labels, percent, flip) with geom one of line, col, point, area, tile.
members	The names data\$member may take, or NULL not to check it.

Value

The spec, tidied. Stops with a message saying what is wrong.

Examples

```
report_validate_spec(list(
  title = "Coverage by region",
  data = list(member = "calculate_coverage", args = list(admin_level = "adminlevel_1")),
  transform = list(list(filter = list(year = 2023))),
  plot = list(geom = "col", x = "adminlevel_1", y = "value")
))
```

resolve_chart_options *Resolve the chart options for a plot*

Description

What every plot method calls with its options argument and its ...: options (an object, a list or NULL) plus any chart-option field named in ... (unrelated arguments in ... are ignored). Named arguments in ... win.

Usage

```
resolve_chart_options(options = NULL, ...)
```

Arguments

options	A <code>cd_chart_options</code> object, a list of fields, or <code>NULL</code> .
...	Chart-option fields (or their older aliases).

Value

A `cd_chart_options` object.

save_report_chart	<i>Draw a chart block to a picture file</i>
-------------------	---

Description

Charts are written as SVG (vector: sharp at any size, on screen, printed or in the PDF Word makes) when the **svglite** package is installed and file ends in `.svg`; otherwise as PNG at dpi.

Usage

```
save_report_chart(rendered, block, file, dpi = 200, design = NULL)
```

Arguments

rendered	The result of <code>render_report_block()</code> with <code>type = "plot"</code> .
block	The block (for its size).
file	Path of the file to write: <code>.svg</code> or <code>.png</code> .
dpi	Resolution of a PNG.
design	The report's design (for the page size).

Value

file, invisibly.

`with_report_chart_options`*Run code with the report's saved chart options in effect*

Description

Run code with the report's saved chart options in effect

Usage

```
with_report_chart_options(context, code, design = NULL)
```

Arguments

context	A report context (<code>report_context()</code>), or what <code>as_report_context()</code> turns into one.
code	Code that draws charts.
design	The report's design, whose fonts the charts use (NULL: the charts keep their own).

Value

The value of code.

Index

ai_action, 5
ai_action(), 7
ai_register_component, 6
ai_reply_when, 7
ai_reply_when(), 5
app_frame(), 4, 5
apply_chart_options, 7
apply_chart_options(), 4, 14
as_report_context, 8
as_report_context(), 4, 19–21, 23, 25, 29, 35, 38, 45

cd_card(), 4
cd_chart_card(), 4
cd_chart_options, 8
cd_chart_options(), 4, 7, 16, 17, 41
cd_chart_type, 14
cd_chart_type(), 4, 26, 36
cd_nav_item(), 4
cd_nav_section(), 4
cd_page_body, 15
cd_page_def(), 4
cd_page_ui(), 4
cd_plot_server(), 4, 6
cd_plot_ui(), 4
cd_register_translations(), 4
cd_start_pages, 16
cd_translations(), 4
cd_use_i18n(), 4
cd_use_pages(), 4
chart_facet_info, 16
chart_facet_info(), 13
chart_option_fields, 17
chart_option_fields(), 4

datasuite.ui (datasuite.ui-package), 3
datasuite.ui-package, 3
ds_host_request, 18

export_deck, 19

export_deck(), 4, 20, 39
export_report, 20
export_report(), 4, 39, 42

merge_chart_options, 21
merge_chart_options(), 4, 14

quire::quire_converter(), 26
quire::quire_export(), 19, 20
quire::quire_theme_from_file(), 38

render_report_block, 21
render_report_block(), 44
report_apply_transforms, 22
report_asset_data_url, 22
report_block_size, 23
report_block_size(), 32, 34
report_blocks, 23
report_chart (report_blocks), 23
report_chart_fields, 25
report_context, 25
report_context(), 4, 19–21, 23, 25, 29, 35, 38, 45
report_converter, 26
report_converter(), 20, 29
report_deck_converter, 26
report_deck_converter(), 19, 29
report_deck_layouts, 27
report_default_cover, 27
report_default_cover(), 36
report_default_design, 28
report_default_design(), 36, 39
report_field_catalog, 28
report_field_catalog(), 4, 29, 36
report_fields, 28
report_final_pages, 29
report_flag_file, 30
report_fonts, 30
report_heading (report_blocks), 23
report_in, 31

report_in(), 40
report_kind, 31
report_kind(), 36, 41
report_list, 32
report_merge, 33
report_note (report_blocks), 23
report_page, 33
report_page(), 23
report_page_break (report_blocks), 23
report_paragraph (report_blocks), 23
report_plot_spec, 34
report_project_blocks, 34
report_project_blocks(), 23
report_questions (report_blocks), 23
report_read, 35
report_register, 36
report_register(), 4, 20, 24, 28, 29
report_resolve_block, 37
report_resolve_block(), 21
report_slide_size, 37
report_store_asset, 38
report_table (report_blocks), 23
report_theme_from_file, 38
report_theme_from_file(), 4
report_themes, 39
report_themes(), 4, 21, 38
report_translate, 40
report_tx, 40
report_tx(), 23, 24, 31
report_update_blocks, 41
report_validate_project, 42
report_validate_project(), 41
report_validate_spec, 43
report_validate_spec(), 22, 34, 42
reports_server(), 4
reports_ui(), 4
resolve_chart_options, 43
resolve_chart_options(), 4

save_report_chart, 44

with_report_chart_options, 45