Package: idem (via r-universe)

June 3, 2026

Title XLSForm Comparison and Validation

Version 2026.6.3

Description Reads and compares XLSForm survey files, validating that a target form is consistent with a development form.

License MIT + file LICENSE

URL https://impact-initiatives.github.io/idem/

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports cli, lifecycle, purrr, readxl, rlang (>= 1.2.0), stats, stringr, tibble, tidyr

Suggests dplyr, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

LazyData true

Config/Needs/website rmarkdown

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev

Repository https://impact-initiatives.r-universe.dev

Date/Publication 2026-06-03 08:50:53 UTC

RemoteUrl https://github.com/impact-initiatives/idem

RemoteRef HEAD

RemoteSha fba1f8f96bef950e57b01320196312afb85376a2

Contents

`idem_passing_lists` *Default list names skipped by* `validate_choices()`

Description

A character vector of XLSForm list names whose choice options are expected to differ between forms (admin boundaries, cluster IDs, enumerator IDs). Used as the default for the `passing_lists` argument of `validate_choices()` and `validate_xlsform()`.

Usage

```
idem_passing_lists
```

Examples

```
idem_passing_lists
```

`msna_template_required`
MSNA template XLSForm (required questions)

Description

An `xlsform` object containing the required questions from the Multi-Sector Needs Assessment (MSNA) template form. This dataset serves as the reference (development) form against which collected XLSForms can be validated with `validate_xlsform()`.

Usage

```
msna_template_required
```

Format

An `xlsform` object — a named list of two tibbles with class `c("xlsform", "list")`:

`survey` — 313 rows × 17 columns: # nolint: line_length_linter.

type XLSForm question type (e.g. `"select_one"`, `"integer"`).

name Variable name.

`label::english (en)` Question label in English.

`label::french (fr)` Question label in French.

`hint::english (en)` Enumerator hint in English.

`hint::french (fr)` Enumerator hint in French.

calculation XLSForm calculation expression.

required Whether the question is required (`TRUE/FALSE/NA`).

relevant XLSForm relevance expression.

constraint XLSForm constraint expression.

default Default value.

repeat_count Repeat count expression for repeat groups.

`constraint_message::english (en)` Constraint violation message in English.

`constraint_message::french (fr)` Constraint violation message in French.

appearance XLSForm appearance attribute.

choice_filter Choice filter expression.

parameters Additional XLSForm parameters.

`choices` — 549 rows × 8 columns: # nolint: line_length_linter.

list_name Choice list identifier referenced in `survey$type`.

name Choice option value.

`label::english (en)` Choice label in English.

`label::french (fr)` Choice label in French.

parent_country Country-level cascade filter value.

parent_admin1 Admin1-level cascade filter value.

parent_admin2 Admin2-level cascade filter value.

parent_admin3 Admin3-level cascade filter value.

Versioning

The dataset carries a `version` attribute recording the package version under which it was generated. Inspect it with:

```
attr(msna_template_required, "version")
```

The dataset is updated in lockstep with package releases, so the version attribute ties each snapshot of the reference form to a specific release.

Source

Derived from the MSNA template XLSForm bundled in `inst/extdata/form.xlsx`. Regenerate with `data-raw/msna_template_required.R`.

See Also

`read_xlsform()`, `validate_xlsform()`

Examples

```
msna_template_required

xlsform_questions(msna_template_required)

attr(msna_template_required, "version")
```

---

read_xlsform *Read an XLSForm file*

Description

Reads an XLSForm `.xlsx` file from disk and returns an `xlsform` object — a named list of tibbles (one per sheet) with the source file path stored as an attribute. This is the standard entry point for working with XLSForms in idem.

Usage

```
read_xlsform(
  path,
  required_sheets = c("survey", "choices"),
  optional_sheets = character()
)
```

Arguments

`path`	Path to the `.xlsx` file.
`required_sheets`	Character vector of sheet names that must be present in the workbook. Defaults to `c("survey", "choices")`. An absent required sheet is an error.
`optional_sheets`	Character vector of sheet names to read if present. Defaults to `character()`. An absent optional sheet produces a warning and is silently excluded from the returned object.

Details

By default the `survey` and `choices` sheets are required. Pass additional sheet names (e.g. `"external_choices"`) via `required_sheets`, or request sheets that may not be present (e.g. `"settings"`) via `optional_sheets`.

Value

An `xlsform` object: a named list of tibbles, one per sheet successfully read, with a `path` attribute holding the source file path and class `c("xlsform", "list")`.

See Also

`xlsform()` to construct an `xlsform` object from in-memory data frames.

Examples

```
path <- system.file("extdata/form.xlsx", package = "idem")

# Read the default sheets (survey + choices)
form <- read_xlsform(path)
form

# Inspect the survey sheet directly
form$survey

# Opportunistically read the settings sheet (no error if absent)
read_xlsform(path, optional_sheets = "settings")
```

`validate_choices` *Validate choice options between two XLSForms*

Description

For every list name that exists in *both* `target` and `dev`'s choices sheets, checks that each choice option name present in `target` also exists in `dev`. Returns a tibble row for each option found in `target` that is absent from `dev` for the same list.

Usage

```
validate_choices(target, dev, passing_lists = idem_passing_lists)
```

Arguments

`target`	An `xlsform` object representing the authoritative reference form.
`dev`	An `xlsform` object representing the form being validated.
`passing_lists`	A character vector of list names to skip entirely. Defaults to idem_passing_lists. Pass `character(0)` to disable all bypasses.

Details

Scope of this check:
This check only compares lists that are defined in *both* forms. Lists that appear in `target` but are entirely absent from `dev` are not reported here — use `validate_list_names()` to catch those gaps first.
A typical validation workflow runs `validate_list_names()` before `validate_choices()`, or simply calls `validate_xlsform()` which runs both.

Value

A tibble with columns `check`, `severity`, `name`, `list_name`, and `detail`. Has zero rows when all choice options in `target` are present in `dev` for every shared list.

See Also

`validate_xlsform()` to run all checks together; `validate_list_names()` for checking that lists themselves exist in `dev`; `xlsform_choices()` to extract choice options from a form.

Examples

```
target <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# No issues: all choice options in target also exist in dev
validate_choices(target, target)

# Issues found: drop one option from a non-passing list
non_passing_row <- which(
  !is.na(target$choices$list_name) &
    !target$choices$list_name %in% idem_passing_lists
)[1]
dev_trimmed <- xlsform(
  survey  = target$survey,
  choices = target$choices[-non_passing_row, ]
)
validate_choices(target, dev_trimmed)

# Extend the default passing_lists with a project-specific list
validate_choices(
  target, target,
  passing_lists = c(idem_passing_lists, "l_my_project_list")
)
```

---

validate_list_names *Validate defined list names between two XLSForms*

Description

Checks that every list name *defined* in `target`'s choices sheet also exists as a defined list in `dev`'s choices sheet. Returns a tibble row for each list name present in `target`'s choices but absent from `dev`'s choices.

Usage

```
validate_list_names(target, dev)
```

Arguments

target	An xlsform object representing the authoritative reference form.
dev	An xlsform object representing the form being validated.

Details

Relationship to other checks:

This check is a prerequisite for validate_choices(): because validate_choices() only compares options for lists that exist in *both* forms' choices sheets, any list that target defines but dev omits would be silently skipped. validate_list_names() catches those gaps explicitly.

To verify that the same lists are also actively *used* in both forms' survey questions (not just defined in choices), see validate_survey_list_names().

Value

A tibble with columns check, severity, name, list_name, and detail. Has zero rows when all list names defined in target's choices are also defined in dev's choices.

See Also

validate_xlsform() to run all checks together; validate_survey_list_names() for the complementary survey-side check; xlsform_defined_list_names() to extract defined list names from a form.

Examples

```
target <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# No issues: all lists defined in target's choices are also defined in dev
validate_list_names(target, target)

# Issues found: dev has no choice lists at all, but target defines some
dev_empty_choices <- xlsform(
  survey  = target$survey,
  choices = data.frame(list_name = character(), name = character())
)
validate_list_names(target, dev_empty_choices)
```

validate_question_names *Validate question names between two XLSForms*

Description

Checks that every question name present in `target`'s survey sheet also exists in `dev`'s survey sheet. Returns a tibble row for each question name found in `target` but absent from `dev`.

Usage

```
validate_question_names(target, dev)
```

Arguments

`target`	An `xlsform` object representing the authoritative reference form.
`dev`	An `xlsform` object representing the form being validated.

Details

This check catches situations where the authoritative `target` form contains questions that the work-in-progress `dev` form has not yet included — for example, a localised adaptation that dropped required questions, or a form version that has fallen behind the central reference.

Value

A tibble with columns `check`, `severity`, `name`, `list_name`, and `detail`. Has zero rows when all question names in `target` are present in `dev`.

See Also

`validate_xlsform()` to run all checks together; `xlsform_questions()` to extract question names from a form.

Examples

```
target <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# No issues: every question in target also exists in dev
validate_question_names(target, target)

# Issues found: target has a question that dev is missing
extra_row <- target$survey[1L, ]
extra_row$name <- "required_question"
target_extra <- xlsform(
  survey  = rbind(target$survey, extra_row),
  choices = target$choices
)
validate_question_names(target_extra, target)
```

validate_survey_list_names *Validate survey-referenced list names between two XLSForms*

Description

Checks that every list name *referenced* in `target`'s survey questions is also referenced in `dev`'s survey questions. Returns a tibble row for each list name actively used by `target`'s survey that is absent from `dev`'s survey.

Usage

```
validate_survey_list_names(target, dev)
```

Arguments

`target`	An `xlsform` object representing the authoritative reference form.
`dev`	An `xlsform` object representing the form being validated.

Details

How it differs from `validate_list_names()`:

`validate_list_names()` compares the lists *defined* in each form's choices sheet. `validate_survey_list_names()` compares the lists actively *used* by survey questions — the second token in `type` values like `select_one list_a`.

The two checks are complementary. A list can be defined in choices but never used in the survey (orphaned list), or — after a question type change from `select_one list_a` to `text` — it may still be defined in choices while no longer referenced in any survey question. This check surfaces the latter case.

Value

A tibble with columns `check`, `severity`, `name`, `list_name`, and `detail`. Has zero rows when all list names referenced by `target`'s survey are also referenced by `dev`'s survey.

See Also

`validate_xlsform()` to run all checks together; `validate_list_names()` for the complementary choices-side check; `xlsform_referenced_list_names()` to extract referenced list names from a form.

Examples

```
target <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# No issues: all lists target's survey uses are also used in dev's survey
validate_survey_list_names(target, target)
```

```
# Issues found: dev's survey has all select questions replaced by text,
# so none of target's referenced lists appear in dev's survey
dev_no_selects <- xlsform(
  survey = data.frame(
    type = rep("text", nrow(target$survey)),
    name = target$survey$name
  ),
  choices = target$choices
)
validate_survey_list_names(target, dev_no_selects)
```

`validate_xlsform` *Validate an XLSForm against a reference form*

Description

Runs one or more validation checks comparing a `dev` (work-in-progress) XLSForm against a `target` (authoritative reference) XLSForm. The default direction checks that everything present in `target` also exists in `dev` — i.e., `target` is a valid subset of `dev`.

Usage

```
validate_xlsform(
  target,
  dev,
  checks = c("question_names", "list_names", "survey_list_names", "choices"),
  passing_lists = idem_passing_lists
)
```

Arguments

`target`	An `xlsform` object representing the authoritative reference form.
`dev`	An `xlsform` object representing the form being validated.
`checks`	A character vector of check names to run. Defaults to all four checks: `c("question_names", "list_names", "survey_list_names", "choices")`.
`passing_lists`	Passed to `validate_choices()`. A character vector of list names whose choice options are not compared. Defaults to idem_passing_lists.

Details

This is the main entry point for form validation. It delegates to the individual `validate_*()` functions and combines their results into a single tibble.

Available checks:

Check name	What it tests

"question_names"	Every question name in target must exist in dev.
"list_names"	Every list name *defined* in target's choices sheet must also be defined in dev's choices sheet.
"survey_list_names"	Every list name *referenced* in target's survey questions must also be referenced in dev's survey qu
"choices"	For every shared list, every choice option in target must exist in the same list in dev.

Return value structure:

Each row in the returned tibble represents one validation issue:

Column	Description
check	Which check produced this issue.
severity	Currently always "error".
name	The name of the offending question or choice option.
list_name	The choices list involved (NA for question-level checks).
detail	A human-readable description of the problem.

Value

A tibble with columns check, severity, name, list_name, and detail. Has zero rows when no issues are found.

See Also

validate_question_names(), validate_list_names(), validate_survey_list_names(), validate_choices() for the individual checks.

Examples

```
target <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# No issues: a form is always a valid subset of itself
validate_xlsform(target, target)

# Run only a subset of checks
validate_xlsform(target, target, checks = c("question_names", "choices"))

# Introduce issues: dev is missing a question and a choice option
non_passing_row <- which(
  !is.na(target$choices$list_name) &
    !target$choices$list_name %in% idem_passing_lists
)[1]
dev_trimmed <- xlsform(
  survey  = target$survey[-nrow(target$survey), ],
  choices = target$choices[-non_passing_row, ]
)
issues <- validate_xlsform(target, dev_trimmed)
issues

# Extend the default passing_lists with a project-specific list
validate_xlsform(
```

```
  target, target,
  passing_lists = c(idem_passing_lists, "l_my_project_list")
)
```

`xlsform` *Construct an xlsform object from data frames*

Description

Builds an `xlsform` object directly from in-memory data frames, without reading from a file. The resulting object is structurally identical to one produced by `read_xlsform()`, making it useful for testing, creating minimal reproducible examples, or programmatically assembling forms.

Usage

```
xlsform(..., path = NA_character_)
```

Arguments

`...`	Named data frames, one per sheet. Names become the sheet names (e.g. `survey =`, `choices =`). All arguments must be named and must be data frames.
`path`	A string recording the (notional) source path. Defaults to `NA_character_` for in-memory objects.

Details

Most idem functions expect at least a `survey` sheet and, for choice-related operations, a `choices` sheet.

Value

An `xlsform` object: a named list of data frames with class `c("xlsform", "list")` and a `path` attribute.

See Also

`read_xlsform()` to load an `xlsform` object from an `.xlsx` file.

Examples

```
# Minimal form with two select_one questions sharing a yes/no list
survey <- data.frame(
  type = c("select_one yn", "select_one yn", "text"),
  name = c("consent", "satisfied", "comments")
)
choices <- data.frame(
  list_name = c("yn", "yn"),
  name      = c("yes", "no"),
  label     = c("Yes", "No")
```

```
)
form <- xlsform(survey = survey, choices = choices)
form

# In-memory forms can be passed directly to validate_xlsform()
validate_xlsform(form, form)
```

`xlsform_choices` *Get choice options from an XLSForm*

Description

Returns a named list of character vectors, where each name is a list name and each element contains the choice option `name` values for that list. Both the `choices` sheet and, when present, the `external_choices` sheet are combined.

Usage

```
xlsform_choices(x, ...)

## Default S3 method:
xlsform_choices(x, ...)

## S3 method for class 'xlsform'
xlsform_choices(x, ...)
```

Arguments

`x`	An `xlsform` object.
`...`	Ignored; present for S3 method compatibility.

Details

This is useful for inspecting which options are available for a given `select_one` or `select_multiple` question, and is used internally by `validate_choices()` to compare option sets across two forms.

Value

A named list of character vectors. Each name is a list name; each element is the character vector of option `name` values for that list. Rows with `NA` in either `list_name` or `name` are silently dropped.

See Also

`xlsform_defined_list_names()` for just the list names; `validate_choices()` to compare choice options across two forms.

Examples

```
form <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# All choice options, organised by list name
xlsform_choices(form)

# Options for a specific list
xlsform_choices(form)[["yn"]]
```

`xlsform_defined_list_names` *Get list names defined in the choices sheets of an XLSForm*

Description

Extracts unique list names from the `list_name` column of all available in-workbook choices sheets. Two sheets are recognised:

Usage

```
xlsform_defined_list_names(x, ...)

## Default S3 method:
xlsform_defined_list_names(x, ...)

## S3 method for class 'xlsform'
xlsform_defined_list_names(x, ...)
```

Arguments

`x`	An `xlsform` object.
`...`	Ignored; present for S3 method compatibility.

Details

- `choices` — the standard sheet used by `select_one`, `select_multiple`, and `rank`.
- `external_choices` — the optional sheet used by `select_one_external` and `select_multiple_external`. Included automatically when present in the loaded form.

Note on file-based question types:
`select_one_from_file` and `select_multiple_from_file` reference external CSV/XML/GeoJSON files rather than any in-workbook sheet. This function cannot resolve those references and emits a warning for each such type it encounters.

Difference from `xlsform_referenced_list_names()`:
`xlsform_referenced_list_names()` returns lists *referenced* by survey questions; `xlsform_defined_list_names()` returns lists *defined* in the choices sheets. The two sets should match for a well-formed form, but can diverge when a question type is changed without updating the choices sheet (or vice versa).

Value

A character vector of unique list names drawn from all available in-workbook choices sheets.

See Also

`xlsform_referenced_list_names()` for list names referenced in the survey sheet; `xlsform_choices()` for the full choice options per list; `validate_list_names()` to compare defined lists across two forms.

Examples

```
form <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# All list names defined in the choices sheet
xlsform_defined_list_names(form)

# Cross-check: lists defined in choices vs. lists used in survey
# (both should be identical for a well-formed form)
all.equal(
  sort(xlsform_defined_list_names(form)),
  sort(xlsform_referenced_list_names(form))
)
```

---

xlsform_questions *Get question names from an XLSForm*

Description

Returns the values of the `name` column from the `survey` sheet, excluding any rows where `name` is `NA` (such as `begin_group` / `end_group` rows that carry no name).

Usage

```
xlsform_questions(x, ...)

## Default S3 method:
xlsform_questions(x, ...)

## S3 method for class 'xlsform'
xlsform_questions(x, ...)
```

Arguments

`x`	An `xlsform` object.
`...`	Ignored; present for S3 method compatibility.

Details

The returned vector is used internally by `validate_question_names()` to compare question inventories across two forms.

Value

A character vector of non-NA question names from the survey sheet.

See Also

`xlsform_referenced_list_names()` for list names referenced in the survey; `xlsform_defined_list_names()` for list names defined in the choices sheet.

Examples

```
form <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# All question names in the form
xlsform_questions(form)

# Count questions
length(xlsform_questions(form))
```

---

xlsform_referenced_list_names

Get list names referenced in an XLSForm's survey sheet

Description

Extracts the unique list names that are actively *referenced* in the `type` column of the `survey` sheet — that is, the second space-separated token for question types that link to a choices list:

Usage

```
xlsform_referenced_list_names(x, ...)

## Default S3 method:
xlsform_referenced_list_names(x, ...)

## S3 method for class 'xlsform'
xlsform_referenced_list_names(x, ...)
```

Arguments

`x`	An `xlsform` object.
`...`	Ignored; present for S3 method compatibility.

Details

Question type	Example type value	Extracted list name
`select_one`	`select_one yn`	`yn`
`select_multiple`	`select_multiple colors`	`colors`
`select_one_external`	`select_one_external regions`	`regions`
`select_multiple_external`	`select_multiple_external items`	`items`
`rank`	`rank priority`	`priority`

`select_one_from_file` and `select_multiple_from_file` are excluded because they reference external CSV/XML/GeoJSON files rather than any in-workbook choices sheet.

Value

A character vector of unique list names referenced in the survey.

See Also

`xlsform_defined_list_names()` for list names *defined* in the choices sheet; `validate_survey_list_names()` to compare referenced lists across two forms.

Examples

```
form <- read_xlsform(system.file("extdata/form.xlsx", package = "idem"))

# Lists actively used by survey questions
xlsform_referenced_list_names(form)

# Compare with lists defined in the choices sheet
xlsform_defined_list_names(form)
```

Index