Package: identifix (via r-universe)

November 28, 2026

Type Package

Title Match Sample Names that are Written Differently

Version 0.1.0

Description Matches two sets of sample names (or other identifiers) that refer to the same objects but are written differently, e.g. with other whitespace, case, separators, accents, leading zeros, date formats and languages, word order, typos or abbreviations. Matching runs in steps from strict to lenient that can be switched on and off. Pairs are only accepted if they are unambiguous within both complete sets, so that names which cannot be matched safely are reported instead of being paired wrongly.

License AGPL-3

URL https://github.com/MatthiasPucher/identifix

BugReports https://github.com/MatthiasPucher/identifix/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports stats, utils

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

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Matthias Pucher [aut, cre] (ORCID: <https://orcid.org/0000-0002-9093-2199>)

Maintainer Matthias Pucher <matthias.pucher@ecosight.at>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-28 14:30:02 UTC

RemoteUrl https://github.com/cran/identifix

RemoteRef HEAD

RemoteSha 265551ed56dff7a6efe2f3f52de31d0cc55ff1f1

Contents

idfix_extradat 2
idfix_known 3
idfix_match 4
idfix_multi 5
idfix_rename 7
idfix_report 8
idfix_review 9
idfix_steps 10

Index **12**

`idfix_extradat` *Extract dates from names*

Description

Finds dates written in many formats and languages inside names, for example ‘2026-09-01‘, ‘20260901‘, ‘01.09.2026‘, ‘1 Sept 2026‘, ‘2026September01‘, ‘2026Setember01‘ (one typo in a month name), ‘September 1st, 2026‘, ‘13 novembre 2026‘ or Chinese/Japanese dates written with the characters for year, month and day. Dates without day (‘Sept_2026‘) and without year (‘03.04.‘, ‘3. April‘) are recognised as well.

Usage

```
idfix_extradat(x, date_order = c("auto", "dmy", "mdy"))
```

Arguments

`x`	Character vector of names.
`date_order`	How to read ambiguous numeric dates: ‘"auto"‘ (default), ‘"dmy"‘ (day first) or ‘"mdy"‘ (month first).

Details

Numeric dates such as ‘03/04/2026‘ can mean 3 April or 4 March. With ‘date_order = "auto"‘ the order is taken from unambiguous dates in ‘x‘ that use the same separator (e.g. ‘22/04/2026‘ shows that ‘/‘ is day-first). Without such evidence, dot-separated dates are read day-first (common convention); all others stay ambiguous and both readings are reported.

Value

A data frame with one row per element of ‘x‘:

name the input.

date ISO representation of the first date found: ‘"2026-09-01"‘, ‘"2026-09"‘ (no day) or ‘"–09-01"‘ (no year); ‘NA‘ if none.

alternative second reading of an ambiguous date, otherwise ‘NA‘.

n_dates number of dates found.

Examples

```
idfix_extradat(c("Exp_2026-09-01", "2026Setember01_B", "01.09.2026",
                 "13 novembre 2026", "Trial_Sept_2026", "S_04/03/2026"))
idfix_extradat("S_04/03/2026", date_order = "mdy")
```

`idfix_known` *Decisions about pairs, for reuse*

Description

Returns all decisions about pairs as a table that can be saved (e.g. with [utils::write.csv()]) and passed to [idfix_match()] as ‘known‘ next time. Confirmed pairs are then matched first (step ‘"known"‘), rejected pairs are never matched or suggested again. This is useful when the same names come up repeatedly, e.g. in a monitoring programme: only new names have to be checked.

Usage

```
idfix_known(x)
```

Arguments

`x`	Result of [idfix_match()].

Details

The table contains * the decisions made with [idfix_review()], * corrections by hand (‘m[3] <- 7L‘ confirms the new pair and rejects the pair found before; ‘m[3] <- NA‘ rejects the pair found before), also when they are made after a review: the latest change counts, * the ‘known‘ table given to [idfix_match()], so that decisions accumulate.

A pair is identified by the two names, regardless of which set they come from. If there are several decisions for the same pair, the latest counts.

Value

A data frame with columns ‘n1‘, ‘n2‘, ‘decision‘ (‘"confirmed"‘ or ‘"rejected"‘) and ‘date‘ (‘"YYYY-MM-DD"‘), one row per pair.

Examples

```
m <- idfix_match(c("Std_10ng", "Probe_A1", "Site 7"),
                 c("Standard_10ng", "probe_a1", "Site-7b"))
m
# the user rejects the first pair and assigns the third by hand
m[1] <- NA
m[3] <- 3L
known <- idfix_known(m)
known
```

```
# next time, these decisions are used
idfix_report(idfix_match(c("Site 7", "Std_10ng"), c("Site-7b", "Standard_10ng"),
                         known = known))
```

`idfix_match` *Match two sets of sample names*

Description

Finds for every name in ‘n1‘ the corresponding name in ‘n2‘, even if the names are written differently (whitespace, case, separators, accents, leading zeros, date formats and languages, word order, typos, abbreviations). The names can be in any order.

Usage

```
idfix_match(
  n1,
  n2,
 steps = c("exact", "whitespace", "case", "punctuation", "zeros", "dates", "reorder",
    "typo", "abbreviation", "date_context"),
  date_order = c("auto", "dmy", "mdy"),
  known = NULL
)
```

Arguments

`n1, n2`	Character vectors (or anything coercible, e.g. factors) with names. ‘NA‘ and empty strings are never matched. Duplicated names are an error: they cannot be matched safely. If duplicates mean the same sample (e.g. data in long format), match ‘unique(x)‘; [idfix_rename()] then renames all occurrences.
`steps`	Character vector of steps to use; see [idfix_steps()] for all steps and their order. Steps always run in that order.
`date_order`	How to read ambiguous numeric dates like ‘03/04/2026‘: ‘"auto"‘ (default) uses unambiguous dates with the same separator in the same set; without such evidence, dotted dates are read day-first and other dates stay ambiguous: they then only match if exactly one reading fits a name in the other set. ‘"dmy"‘ or ‘"mdy"‘ force an order.
`known`	Optional table of earlier decisions as returned by [idfix_known()] (columns ‘n1‘, ‘n2‘, ‘decision‘). Confirmed pairs are matched first (step ‘"known"‘), rejected pairs are never matched or suggested. Pairs are found by exact names in either orientation.

Details

The matching runs in steps (see [idfix_steps()]). Each step only looks at names that are still unmatched; matched pairs are removed before the next step. Steps go from strict to lenient, so a pair is always found by the strictest comparison possible.

Wrong pairs are avoided rather than gaps. In every step (except ‘"exact"‘), a pair is only accepted if both names have exactly one candidate in the *complete* other set and no look-alike in their own set. For example, if ‘n1‘ contains both ‘"Sample_A"‘ and ‘"sample_a"‘, case is obviously meaningful and neither name is matched by ignoring case. Numbers and dates must always agree, and typos at the start or end of a word (‘sampleA‘ / ‘sampleB‘) are never corrected. Names that cannot be matched safely stay ‘NA‘ and are listed by [idfix_report()] with a suggestion.

Normalisations accumulate: disabling ‘"case"‘, ‘"punctuation"‘ (also accents), ‘"zeros"‘ or ‘"dates"‘ also switches them off in all later steps.

Value

An integer vector of the same length as ‘n1‘ with the position of the matching name in ‘n2‘ (‘NA‘ if none), so that ‘n2[result]‘ is aligned with ‘n1‘. It has class ‘idfix_match‘ and keeps the details needed by [idfix_report()]. Use ‘as.vector()‘ to get a plain integer vector. Matches can be corrected by assignment, e.g. ‘m[3] <- 7L‘.

Examples

```
n1 <- c("Sample_001", "Exp_2026-09-01", "DNA_Sample_002", "Muster A",
        "sampleA", "Only_in_n1")
n2 <- c("2026Setember01_Exp", "muster_a", "sample_1", "DNA_Smaple_002",
        "sampleB")
m <- idfix_match(n1, n2)
m
n2[m]
idfix_report(m)
```

`idfix_multi` *Match more than two sets of sample names*

Description

Matches several sets of names and returns a table with one row per sample and one column per set. The first set (or the one given by ‘reference‘) is the starting point; every further set is matched with [idfix_match()] against the rows found so far. Names that match no row get a new row, so later sets can still be matched to samples that are missing in the reference ("growing reference"). Each row is represented by its name in the reference, or, if missing there, by the name of the set that added the row.

Usage

```
idfix_multi(
  ...,
  reference = 1,
  value = c("name", "index"),
 steps = c("exact", "whitespace", "case", "punctuation", "zeros", "dates", "reorder",
    "typo", "abbreviation", "date_context"),
  date_order = c("auto", "dmy", "mdy"),
  known = NULL
)
```

Arguments

...	Two or more vectors of names, preferably named (‘idfix_multi(lab = x, field = y)‘); alternatively one list of such vectors. Unnamed sets are called ‘n1‘, ‘n2‘, ... Duplicated names within a set are an error.
reference	Number or name of the set to start with.
value	‘"name"‘ (default) to fill the table with the names, ‘"index"‘ to fill it with the positions within each set.
steps	Character vector of steps to use; see [idfix_steps()] for all steps and their order. Steps always run in that order.
date_order	How to read ambiguous numeric dates like ‘03/04/2026‘: ‘"auto"‘ (default) uses unambiguous dates with the same separator in the same set; without such evidence, dotted dates are read day-first and other dates stay ambiguous: they then only match if exactly one reading fits a name in the other set. ‘"dmy"‘ or ‘"mdy"‘ force an order.
known	Optional table of earlier decisions ([idfix_known()]); it is used for every single matching.

Value

A data frame of class ‘idfix_multi‘ with one column per set (in the order given) and one row per sample: first the names of the reference in their order, then names that are missing in the reference. Attribute ‘matches‘ holds the individual [idfix_match()] results (rows so far against each further set) for checking with [idfix_report()]. Use [idfix_rename()] to rename the names of one set into the spelling of another.

Examples

```
lab   <- c("Sample_001", "Exp_2026-09-01", "sampleA")
field <- c("sample_1", "2026Setember01_Exp", "sampleB")
gis   <- c("SAMPLE 1", "sampleB", "Site_9")
t <- idfix_multi(lab = lab, field = field, gis = gis)
t
idfix_multi(lab = lab, field = field, gis = gis, value = "index")
idfix_report(attr(t, "matches")$gis)
```

idfix_rename *Rename names into the spelling of the other set*

Description

Uses the result of [idfix_match()] or [idfix_multi()] to replace names by the matching names of another set. Without ‘data‘, the renamed names are returned as a character vector. With ‘data‘, the names are replaced directly in an object: the names of a vector or list, the values of a vector or data frame column, or the column or row names of a data frame or matrix.

Usage

```
idfix_rename(
  x,
  data = NULL,
  to = NULL,
  from = NULL,
  unmatched = c("keep", "na"),
  where = NULL
)
```

Arguments

x	Result of [idfix_match()] or [idfix_multi()].
data	Optional object that contains the names to rename (the names of the set given by ‘to‘’s counterpart, see ‘from‘). The values may repeat (e.g. a sample column in long format) and may be a subset of the set.
to	Spelling to use. For [idfix_match()]: ‘"n2"‘ (default, names of ‘n1‘ are replaced by the matching names of ‘n2‘) or ‘"n1"‘. For [idfix_multi()]: name of the target set (default: the reference).
from	Only for [idfix_multi()]: name of the set whose names are replaced. May be omitted if there are only two sets.
unmatched	What to do with names without a match: ‘"keep"‘ (default) keeps them, ‘"na"‘ replaces them by ‘NA‘.
where	Where the names are in ‘data‘: ‘"names"‘, ‘"values"‘, ‘"colnames"‘, ‘"rownames"‘ or a column name or number of a data frame. By default this is detected: the place whose entries all belong to the set being renamed. If several places qualify, ‘where‘ must be given.

Details

Check the matching first with [idfix_report()] and correct it if needed (e.g. ‘m[3] <- 7L‘); ‘idfix_rename()‘ uses the matching as it is. For [idfix_multi()] results, the table itself is used, so corrections in the table and removed rows are taken into account.

Value

Without ‘data‘, a character vector with the renamed names of the set given by ‘from‘ (for [idfix_match()]: ‘n1‘ if ‘to = "n2"‘, ‘n2‘ if ‘to = "n1"‘), in their original order. With ‘data‘, ‘data‘ with renamed names.

Examples

```
lab   <- c("Sample_001", "Exp_2026-09-01", "DNA_Sample_002", "sampleA")
field <- c("2026Setember01_Exp", "sample_1", "DNA_Smaple_002")
m <- idfix_match(lab, field)

idfix_rename(m)                   # lab names in field spelling
idfix_rename(m, to = "n1")        # field names in lab spelling
idfix_rename(m, unmatched = "na")

# rename the names of a named vector
conc <- c(Sample_001 = 1.2, `Exp_2026-09-01` = 3.4, DNA_Sample_002 = 0.7,
          sampleA = 2.2)
idfix_rename(m, conc)

# rename a column in long format
long <- data.frame(sample = rep(field, each = 2), rep = 1:2)
idfix_rename(m, long, to = "n1")

# more than two sets
gis <- c("SAMPLE 1", "DNA_Sample_2")
t <- idfix_multi(lab = lab, field = field, gis = gis)
idfix_rename(t, from = "gis")                # into lab spelling
idfix_rename(t, from = "gis", to = "field")
```

`idfix_report` *Report of a name matching*

Description

Shows for every name of ‘n1‘ the matched name of ‘n2‘, the step that found the match and, for unmatched names, why no match was accepted and the most similar unmatched name of the other set as suggestion. Unmatched names of ‘n2‘ are added at the end. Matches changed by hand (e.g. ‘m[3] <- 7L‘) are marked as ‘"manual"‘.

Usage

```
idfix_report(x, unmatched_only = FALSE)
```

Arguments

`x`	Result of [idfix_match()].
`unmatched_only`	If ‘TRUE‘, only rows without a match are returned.

Value

A data frame with columns ‘n1_index‘, ‘n1‘, ‘n2_index‘, ‘n2‘, ‘step‘, ‘confidence‘ (‘"known"‘, ‘"high"‘, ‘"medium"‘, ‘"low"‘ or ‘"unmatched"‘, see [idfix_steps()]), ‘review‘ (result of [idfix_review()], of corrections by hand or of earlier decisions passed as ‘known‘: ‘"confirmed"‘, ‘"rejected"‘, ‘"manual"‘, ‘"no_partner"‘ or ‘NA‘; confirmed pairs have the confidence ‘"known"‘ and the date of the confirmation in ‘note‘), ‘note‘, ‘suggestion‘ (index in the other set), ‘suggestion_name‘, ‘similarity‘ (0 to 1, plain edit distance) and ‘suggestion_check‘ (warns if numbers or dates of the suggestion differ).

Examples

```
m <- idfix_match(c("A_01", "B_02", "C_03"), c("b_2", "a_1", "X_09"))
idfix_report(m)

# correct by hand, then check again
m[3] <- 3L
idfix_report(m)
```

---

idfix_review *Check matched pairs and unmatched names*

Description

Shows the result of [idfix_match()] pair by pair so that you can confirm or reject each pair and choose a partner for names without a match. Differences between the two names are highlighted. The decisions are stored in the result: rejected pairs become ‘NA‘, chosen partners are entered, and [idfix_known()] returns all decisions for reuse in later matchings.

Usage

```
idfix_review(
  x,
  confidence = c("known", "high", "medium", "low", "unmatched"),
  mode = c("auto", "shiny", "console"),
  n_candidates = 3
)
```

Arguments

x	Result of [idfix_match()].
confidence	Which pairs to show, by confidence level (see [idfix_steps()]): any of ‘"known"‘ (confirmed earlier or set by hand), ‘"high"‘, ‘"medium"‘, ‘"low"‘ and ‘"unmatched"‘ (names without match). Default: all.
mode	‘"auto"‘ (default), ‘"shiny"‘ or ‘"console"‘.
n_candidates	Maximum number of candidates offered for each unmatched name (the most similar names of ‘n2‘ with a similarity of at least 0.3).

Details

With ‘mode = "shiny"‘ the review opens in the RStudio viewer or the browser (package **shiny** needed). Keys: arrow keys or ‘j‘/‘k‘ to move, ‘Enter‘ or ‘y‘ to confirm, ‘x‘ or ‘n‘ to reject, ‘1‘-‘9‘ to choose a candidate, ‘0‘ for "no partner", ‘Backspace‘ or ‘u‘ to undo. Finish with "Done". With ‘mode = "console"‘ the pairs are shown one after another in the R console. ‘mode = "auto"‘ uses shiny if it is installed and the console otherwise.

Pairs you do not decide on keep their match and are not recorded as decisions. Pairs confirmed earlier (in a previous review, by hand, or via ‘known‘ in [idfix_match()]) are shown as confirmed; reject them to revoke the confirmation. Names marked earlier as having no partner are shown that way, and rejected pairs are not offered again as candidates. Undo returns an item to its state at the start of the review.

Value

‘x‘ with the decisions applied (see [idfix_report()], columns ‘review‘ and ‘step‘). If the review is cancelled, ‘x‘ unchanged.

Examples

```
m <- idfix_match(c("DNA_Sample_001", "Std_10ng", "Exp_12/11/2026"),
                 c("DNA_Smaple_001", "Standard_10ng", "Exp_2026-11-12"))
if (interactive()) {
  m <- idfix_review(m)                          # everything
  m <- idfix_review(m, confidence = c("low", "unmatched"))
  idfix_report(m)
  idfix_known(m)
}
```

`idfix_steps` *Available matching steps*

Description

Lists the steps that [idfix_match()] can use, in the order in which they are run, whether they are used by default and what they do. Steps are applied one after another; names matched in one step are removed before the next step.

Usage

```
idfix_steps()
```

Details

Each step has a confidence level that [idfix_review()] uses to select the pairs to check: ‘"high"‘ (only whitespace, case, punctuation or leading zeros differ), ‘"medium"‘ (dates, word order, additional words next to a date) and ‘"low"‘ (typos, abbreviations, missing words). Pairs confirmed earlier (argument ‘known‘ of [idfix_match()]), confirmed in [idfix_review()] or set by hand have the level ‘"known"‘; pairs that rely on an ambiguous date are otherwise always ‘"low"‘.

Value

A data frame with columns ‘step‘, ‘default‘, ‘confidence‘ and ‘description‘.

Examples

```
idfix_steps()

# all default steps except typo correction
steps <- idfix_steps()
setdiff(steps$step[steps$default], "typo")
```

Index

idfix_extradat, 2
idfix_known, 3
idfix_match, 4
idfix_multi, 5
idfix_rename, 7
idfix_report, 8
idfix_review, 9
idfix_steps, 10