# Package: WorldFlora (via r-universe)

January 9, 2025

**Type** Package

**Title** Standardize Plant Names According to World Flora Online Taxonomic Backbone

**Version** 1.14-5

**Date** 2024-9-9

**Author** Roeland Kindt [cre, aut] (<https://orcid.org/0000-0002-7672-0712>)

**Maintainer** Roeland Kindt <RoelandCEKindt@gmail.com>

**Description** World Flora Online is an online flora of all known plants, available from <https://www.worldfloraonline.org/>. Methods are provided of matching a list of plant names (scientific names, taxonomic names, botanical names) against a static copy of the World Flora Online Taxonomic Backbone data that can be downloaded from the World Flora Online website. The World Flora Online Taxonomic Backbone is an updated version of The Plant List (<http://www.theplantlist.org/>), a working list of plant names that has become static since 2013.

**License** GPL-3

**Depends** R (>= 3.5.0)

**Suggests** data.table, utils, stringr, dplyr, fuzzyjoin, stringdist

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2024-09-10 09:50:15 UTC

## Contents

---

`new.backbone` *Develop a User-created Taxonomic Backbone data set*

---

## Description

Instead of using the taxonomic backbone data set from World Flora Online, it is possible to use matching functions of WorldFlora with alternative taxonomic backbone. The function creates new variables that correspond to key variables in the World Flora Online backbone so that matching functions WFO.match and WFO.one can be applied.

## Usage

```
new.backbone(x,
    taxonID = "taxonID", scientificName = "scientificName",
    scientificNameAuthorship = "scientificNameAuthorship",
    acceptedNameUsageID = NULL, taxonomicStatus = NULL
)
```

## Arguments

| | |
|---|---|
| `x` | data.frame with the variables. |
| `taxonID` | name of the variable with the identification |
| `scientificName` | name of the variable with the full taxon name |
| `scientificNameAuthorship` | name of the variable with the naming authors |
| `acceptedNameUsageID` | ID of the record with the current (accepted) name. Should respond to an ID in the 'taxonID' column. In case the taxonomic name is current, then this field should be left blank. This field is used by function WFO.match to find the accepted name of a species. |
| `taxonomicStatus` | Variable that indicates whether the record is for a current name or a synonym. This variable is used by function WFO.one to discriminate situations where best matches include matches with current names and synonyms. |

## Details

This function allows a user to create a new taxonomic backbone data set that is understood by WFO.match and WFO.one.

Alternative examples with the Mammal Diversity Database (https://www.mammaldiversity.org/) and the World Checklist of Vascular Plants (https://powo.science.kew.org/about-wcvp) are provided in the Kindt 2021a,b RPubs.

## Value

The function returns a data.table that can be understood by WFO.match and WFO.one for standardizing taxonomic names.

## Author(s)

Roeland Kindt (World Agroforestry)

## References

Kindt, R. 2021a. Standardizing mammal species names with the Mammal Species Database via exact and fuzzy matching functions from the WorldFlora package. https://rpubs.com/Roeland-KINDT

Kindt, R. 2021b. Standardizing GlobalTreeSearch tree species names with World Flora Online and the World Checklist of Vascular Plants https://rpubs.com/Roeland-KINDT

## See Also

WFO.match, WFO.one

## Examples

```
## Not run:

# load the World Flora Online taxonomic backbone
WFO.remember()

# get a list of Sapotaceae species
Sapotaceae <- WFO.data[WFO.data$family == "Sapotaceae",]
Sapotaceae <- Sapotaceae[Sapotaceae$taxonRank == "SPECIES", ]
Sapotaceae <- Sapotaceae[Sapotaceae$taxonomicStatus == "Accepted", ]
Sapotaceae <- Sapotaceae[, c("scientificName", "scientificNameAuthorship")]
Sapotaceae <- data.frame(ID = c(1:nrow(Sapotaceae)), Sapotaceae)
names(Sapotaceae)[2:3] <- c("species.name", "author")
head(Sapotaceae)

# create a new backbone from the GlobalTreeSearch database,
# after copying locally from https://tools.bgci.org/global_tree_search.php
GTS.dir <- "E://Roeland//R///World Flora Online//2021"
GTS <- read.csv(paste0(GTS.dir, "//global_tree_search.csv"))
GTS <- GTS[, 1:2]
GTS <- data.frame(GTS.ID = paste0("GTS-", c(1:nrow(GTS))), GTS)
nrow(GTS)

# create the new backbone
GTS.data <- new.backbone(GTS,
                         taxonID="GTS.ID",
                         scientificName="TaxonName",
                         scientificNameAuthorship="Author")
head(GTS.data)

# Check and standardize Sapotaceae
```

```
Sapotaceae.match <- WFO.one(WFO.match(Sapotaceae,
                                      WFO.data = GTS.data,
                                      spec.name = "species.name",
                                      Authorship = "author"))

nrow(Sapotaceae.match[Sapotaceae.match$Fuzzy == FALSE, ] )
nrow(Sapotaceae.match[Sapotaceae.match$Fuzzy == TRUE &
        Sapotaceae.match$Fuzzy.dist < 4, ] )
Sapotaceae.match[Sapotaceae.match$Fuzzy == TRUE &
        Sapotaceae.match$Fuzzy.dist < 4,
        c("ID", "species.name", "Fuzzy.dist", "scientificName")]


## End(Not run)
```

---

`vascular.families` *Orders and Higher Level Classifications of Vascular Plants*

---

## Description

This data set lists orders for families of vascular plants (angiosperms, gymnosperms and pteridophytes). For angiosperms, information from orders and higher levels of classification correspond to the fourth update of the Angiosperm Phylogeny Group (APG IV, doi:10.1111/boj.12385). Higher levels of classification correspond to names of nodes of the consensus tree (Figure 1 in doi:10.1111/boj.12385). Orders for gymnosperms and pteridophytes were obtained from the website of Missouri Botanical Garden.

## Usage

```
data(vascular.families)
```

## Format

A data frame with 476 observations on the following 10 variables.

`Group` Group.

`Family.ID` Unique ID for each family. For angiosperms, these correspond to APG IV.

`Family` Name of the plant family.

`Family.taxonID` taxonID retrieved from World Flora Online.

`Order` Name of the plant order.

`Order.taxonID` taxonID retrieved from World Flora Online.

`Node.1` Name of the node in the consensus tree.

`Node.2` Name of the node in the consensus tree, with Node.2 nested within Node.1.

`Node.3` Name of the node in the consensus tree, with Node.3 nested within Node.2.

`Node.4` Name of the node in the consensus tree, with Node.4 nested within Node.3.

## References

The Angiosperm Phylogeny Group, M. W. Chase, M. J. M. Christenhusz, M. F. Fay, J. W. Byng, W. S. Judd, D. E. Soltis, D. J. Mabberley, A. N. Sennikov, P. S. Soltis, P. F. Stevens, An update of the Angiosperm Phylogeny Group classification for the orders and families of flowering plants: APG IV, Botanical Journal of the Linnean Society 181: 1-20. doi:10.1111/boj.12385

## Examples

```
data(vascular.families)
```

---

# WFO.acceptable.match *Check for fuzzy matches that can be acceptable based on gender notations*

---

## Description

The function checks whether submitted and match names only differ by ending by -um, -us or -a. An extra check is done to accept differences that result from having 'ii' instead of 'i' in the submitted and matched name. An optional check ignores differences in vowels.

## Usage

```
WFO.acceptable.match(x, spec.name="spec.name",
  no.vowels=FALSE)
```

## Arguments

| | |
|---|---|
| x | Output for WFO.match, WFO.match.fuzzyjoin or WFO.match.one. |
| spec.name | Name of taxon submitted for matching. |
| no.vowels | Accept results if only vowels differ between submitted and matched name. |

## Details

The function was initially developed to check for changes in gender notations.

In new versions, also the following differences in species names are judged to be acceptable:

- hybrid and non-hybrid names (eg, Sorbus avonensis - Sorbus xavonensis)

- i vs. j (eg, Syzygium naiadum - Syzygium najadum)

- tt vs. t (eg, Ficus scott-elliottii - Ficus scott-elliotii)

- ll vs. l (eg, Garcinia moseleyana - Garcinia moselleyana)

- rr vs. r (eg, Hymenodictyon perrieri - Hymenodictyon perieri)

- mm vs. m (eg, Monteverdia schummaniana - Monteverdia schumanniana)

- nn vs. n (eg, Pyrus tamamschiannae - Pyrus tamamschianae)

- ff vs. f (eg, Dendropanax langsdorfii - Dendropanax langsdorffii)

- hh vs. h (eg, Gmelina leichardtii - Gmelina leichhardtii)

- dd vs. d (eg, Miconia buddlejoides - Miconia budlejoides)

- is vs. e (eg, Decarydendron ranomafanensis - Decarydendron ranomafanense)

- dt vs. d (eg, Stadtmannia acuminata - Stadmania acuminata)

## Value

The function returns a logical vector that indicates whether names could be acceptable.

## Author(s)

Roeland Kindt (World Agroforestry)

## Examples

```
## Not run:

data(WFO.example)

spec.test <- data.frame(spec.name=c("Faidherbia albida", "Acacia albida",
  "Faidherbia albidum", "Faidherbia albidus",
  "Faidherbia albiida",
  "Prunus africanus", "Prunos africanea",
  "Prunus afrocaneus", "Prunus afrocaneos"))

match1 <- WFO.match.fuzzyjoin(spec.data=spec.test, WFO.data=WFO.example,
  fuzzydist.max = 6)
match1[, c("spec.name", "scientificName")]

# check for gender differences (and ii - i)
WFO.acceptable.match(match1)

# ignore differences in vowels
WFO.acceptable.match(match1, no.vowels=TRUE)

accepted.cases <- WFO.acceptable.match(match1, no.vowels=TRUE)
match1.accepted <- match1[accepted.cases == TRUE, ]
match1.notaccepted <- match1[accepted.cases == FALSE, ]

## End(Not run)
```

---

`WFO.example` *World Flora Online (WFO) taxonomic backbone example data set*

---

## Description

This data set is a subset of the World Flora Online taxonomic backbone that allows running the first set of examples. In practical applications, users should first download a static copy of the Taxonomic Backbone data from https://www.worldfloraonline.org or https://zenodo.org/doi/10.5281/zenodo.7460141 (_DwC_backbone_R.zip).

## Usage

```
data(WFO.example)
```

## Source

World Flora Online. An Online Flora of All Known Plants. https://www.worldfloraonline.org

## Examples

```
data(WFO.example)
```

---

`WFO.match` *Standardize plant names according to World Flora Online taxonomic backbone*

---

## Description

This package checks a list of taxa (typically species) against the World Flora Online (WFO) taxonomic backbone. The user needs to first download a static copy of the Taxonomic Backbone data from https://www.worldfloraonline.org or https://zenodo.org/doi/10.5281/zenodo.7460141 (_DwC_backbone_R.zip).

## Usage

```
WFO.match(spec.data = NULL, WFO.file = NULL, WFO.data = NULL,
    no.dates = TRUE,
    spec.name = "spec.name", Genus = "Genus", Species = "Species",
  Infraspecific.rank = "Infraspecific.rank", Infraspecific = "Infraspecific",
    Authorship = "Authorship", First.dist = FALSE,
    acceptedNameUsageID.match = TRUE,
    Fuzzy = 0.1, Fuzzy.force = FALSE, Fuzzy.max = 250, Fuzzy.min = TRUE,
    Fuzzy.shortest = FALSE, Fuzzy.within = FALSE,
    Fuzzy.two = TRUE, Fuzzy.one = TRUE,
    squish = TRUE,
  spec.name.tolower = FALSE, spec.name.nonumber = TRUE, spec.name.nobrackets = TRUE,
    exclude.infraspecific = FALSE,
    infraspecific.excluded = c("cultivar.", "f.", "sect.", "subf.", "subg.",
        "subsp.", "subvar.", "var", "var.", "[infraspec.]", "fo.", "forma",
        "nothosubsp.", "nothovar.", "sect."),
    spec.name.sub = TRUE,
```

```
  sub.pattern=c(" sp[.] A", " sp[.] B", " sp[.] C", " sp[.]", " spp[.]", " pl[.]",
     " indet[.]", " ind[.]", " gen[.]", " g[.]", " fam[.]", " nov[.]", " prox[.]",
         " cf[.]", " aff[.]", " s[.]s[.]", " s[.]l[.]",
         " p[.]p[.]", " p[.] p[.]", "[?]", " inc[.]", " stet[.]", "Ca[.]",
      "nom[.] cons[.]", "nom[.] dub[.]", " nom[.] err[.]", " nom[.] illeg[.]",
      " nom[.] inval[.]", " nom[.] nov[.]", " nom[.] nud[.]", " nom[.] obl[.]",
     " nom[.] prot[.]", " nom[.] rej[.]", " nom[.] supp[.]", " sensu auct[.]"),
     verbose = TRUE, counter = 1000)

WFO.url(WFO.result = NULL, browse = FALSE, browse.rows = c(1:1), ...)

WFO.one(WFO.result = NULL, priority = "Accepted",
  spec.name = NULL, Auth.dist = NULL, Old.author.dist=NULL, First.dist = NULL,
     verbose = TRUE, counter = 1000)

WFO.browse(taxon, WFO.file = NULL, WFO.data = NULL,
    accepted.only = FALSE, acceptedNameUsageID.match = TRUE, ...)

WFO.synonyms(taxon, WFO.file = NULL, WFO.data = NULL, ...)

WFO.family(taxon, WFO.file = NULL, WFO.data = NULL, ...)
```

## Arguments

| | |
|---|---|
| spec.data | A data.frame containing variables with species names. In case that a character vector is provided, then this vector will be converted to a data.frame |
| WFO.file | File name of the static copy of the Taxonomic Backbone. If not NULL, then data will be reloaded from this file. |
| WFO.data | Data set with the static copy of the Taxonomic Backbone. Ignored if WFO.file is not NULL. |
| no.dates | Speeding up the loading of the WFO.data by not loading fields of 'created' and 'modified'. |
| spec.name | Name of the column with taxonomic names. In case that a spec.name is provided, then separate genus and species names will be ignored. For function WFO.one, giving the name for this columns results in copying a submitted but unmatched plant name into the scientificName of the results. |
| Genus | Name of the column with the genus names. |
| Species | Name of the column with the species names. |
| Infraspecific.rank | Name of the column with the infraspecific rank (such as "subsp.", "var." or "cultivar."). |
| Infraspecific | Name of the column with the infraspecific names. |
| Authorship | Name of the column with the naming authorities. |
| First.dist | If TRUE, then calculate the fuzzy distance between the first words of the submitted and matched names (these are typically the genus names) . |

`acceptedNameUsageID.match`
: If `TRUE`, obtain the accepted name and others details from the earlier acceptedNameUsageID.

`Fuzzy`
: If larger than 0, then attempt fuzzy matching in case an identical taxonomic name is not found in the World Flora Online. This argument will be used as argument `max.distance` in the internally called `agrep`. Note that fuzzy matching is only possible for the `spec.name`.

`Fuzzy.force`
: If `TRUE`, always use the fuzzy matching algorithm, even when the `spec.name` was matched exactly.

`Fuzzy.max`
: Maximum number of fuzzy matches.

`Fuzzy.min`
: If `TRUE`, limit the matching of names to those with the smallest Levenshtein distance, calculated via adist.

`Fuzzy.shortest`
: If `TRUE`, limit the matching of names to those with the most similar length of characters (this feature is expected to eliminate matches at infraspecific levels, see examples).

`Fuzzy.within`
: If `TRUE`, limit the matching of names to those that contain exactly the submitted plant name (this feature is expected to be useful when submitting plant names that only contain a subset of the first characters of the species name, in order to check for best matches manually afterwards).

`Fuzzy.two`
: If `TRUE`, in case that there were no fuzzy matches, limit the terms to be matched to the first two (these are expected to be genus and species names).

`Fuzzy.one`
: If `TRUE`, in case that there were no fuzzy matches, limit the terms to be matched to the first one (expected to be the genus name).

`squish`
: If `TRUE`, remove repeated whitespace and white space from the start and end of the submitted full name via str_squish.

`spec.name.tolower`
: If `TRUE`, then convert all characters of the `spec.name` to lower case via tolower.

`spec.name.nonumber`
: If `TRUE`, then submitted `spec.name` that contain numbers will be interpreted as genera, only matching the first word.

`spec.name.nobrackets`
: If `TRUE`, then submitted `spec.name` then sections of the submitted name after '(' will be removed. Note that this will also remove sections after ')', such as authorities for plant names that are in a separate column of WFO.

`exclude.infraspecific`
: If `TRUE`, then exclude records that contain the infraspecific levels defined by `infraspecific.excluded`.

`infraspecific.excluded`
: Infraspecific levels (available from column 'verbatimTaxonRank') excluded in the results. Note that levels are excluded both in direct matches and matches with the accepted name.

`spec.name.sub`
: If `TRUE`, then delete sections of the `spec.name` that match the `sub.pattern`.

`sub.pattern`
: Sections of the `spec.name` to be deleted

`verbose`
: Give details on the fuzzy matching process.

| | |
|---|---|
| counter | Progress on the matching process is reported by multiples of this counter. |
| WFO.result | Result obtained via WFO.match. |
| browse | If TRUE, then browse urls specified by browse.rows. |
| browse.rows | Indices of row with the urls to be browsed. |
| priority | Method of selecting the 1-to-1 matches. Option Accepted first limits candidates to accepted names, with a possible second step of eliminating accepted names that are synonyms. Option Synonym first limits candidates to those that are not synonyms, with a possible second step of eliminating names that are not accepted. When the number of matches is larger than one after these steps, a third algorithm picks the candidate with the smallest taxonID. |
| Auth.dist | In case that the name of the variable with the Levenshtein distance between the authorship names is provided, then the algorithm first prioritizes records with the best match between the submitted and matched author names. |
| Old.author.dist | In case that the name of the variable with the Levenshtein distance between the authorship names for the synonym matches is provided, then the algorithm first prioritizes records with the best match between the submitted and matched author names. |
| taxon | Character string with the name of the taxon for which information will be given (for families, different genera; for genera, different specieds; for species, infraspecific levels). |
| accepted.only | If TRUE, then only provide taxa with accepted names. |
| ... | Other arguments for browseURL (WFO.url) or WFO.match (WFO.browse). |

## Details

The principal function (WFO.match) matches plant names. Columns retrieved from the World Flora Online are added to the provided input data.frame. In case that there are multiple matches, then rows from the input data.frame are repeated.

Column 'Unique' shows whether there was a unique match (or not match) in the WFO.

Column 'Matched' shows whether there was a match in the WFO.

Column 'Fuzzy' shows whether matching was done by the fuzzy method.

Column 'Fuzzy.dist' gives the Levenshtein distance calculated between submitted and matched plant names adist.

Column 'Auth.dist' gives the Levenshtein distance calculated between submitted and matched authorship names, if the former were provided adist.

Column 'Subseq' gives different numbers for different matches for the same plant name.

Column 'Hybrid' shows whether there was a hybrid character in the scientificName.

Column 'New.accepted' shows whether the species details correspond to the current accepted name.

Column 'Old.status' gives the taxonomic status of the first match with the non-blank acceptedNameUsageID.

Column 'Old.ID' gives the ID of the first match with the non-blank acceptedNameUsageID.

Column 'Old.name' gives the name of the first match with the non-blank acceptedNameUsageID.

The function was inspired on the Taxonstand package that matches plant names against The Plant List. Note that The Plant List has been static since 2013, but was used as the starting point for the Taxonomic Backbone of the World Flora Online.

Function WFO.one finds one unique matching name for each submitted name. Via priority = "Accepted", it first limits candidates to accepted names, with a possible second step of eliminating accepted names that are synonyms. Via priority = "Synonym", it first limits candidates to those that are not synonyms, with a possible second step of eliminating names that are not accepted. When the number of matches is larger than one after these steps, a third algorithm picks the candidate with the smallest taxonID. When a spec.name is given to WFO.one, the original submitted name is inserted for the scientificName.

When the user specifies the column with the Auth.dist, documenting the Levenshtein distance between the submitted and matched authorities, then WFO.one first prioritizes records with best match between Authorities.

Function WFO.browse lists all the genera for a family, all species for a genus or all infraspecific levels for a species.

Function WFO.synonyms gives all records with the acceptedNameUsageID equal to the matched accepted species shown in the first row.

Function WFO.family provides information on the order of vascular plants, based on information available from vascular.families. Based on an internal list of bryophyte families, when the submitted plant name is a bryophyte, the function returns 'bryophyte' instead.

## Value

The main function returns a data.set with the matched species details from the WFO.

## Author(s)

Roeland Kindt (World Agroforestry, CIFOR-ICRAF)

## References

World Flora Online. An Online Flora of All Known Plants. https://www.worldfloraonline.org

Sigovini M, Keppel E, Tagliapietra. 2016. Open Nomenclature in the biodiversity era. Methods in Ecology and Evolution 7: 1217-1225.

Kindt, R. 2020. WorldFlora: An R package for exact and fuzzy matching of plant names against the World Flora Online taxonomic backbone data. Applications in Plant Sciences 8(9): e11388

## See Also

WFO.match.fuzzyjoin

## Examples

```
data(WFO.example)

spec.test <- data.frame(spec.name=c("Faidherbia albida", "Acacia albida",
    "Omalanthus populneus", "Pygeum afric"))
```

```
WFO.match(spec.data=spec.test, WFO.data=WFO.example, counter=1, verbose=TRUE)

# Also calculate the Levenshtein distance for the genus
WFO.match(spec.data=spec.test, WFO.data=WFO.example, First.dist=TRUE,
    counter=1, verbose=TRUE)

# Show all the fuzzy matches, which included those at infraspecifc level
e1 <- WFO.match(spec.data=spec.test, WFO.data=WFO.example, counter=1,
    Fuzzy.min=FALSE, Fuzzy.shortest=FALSE, verbose=TRUE)
e1

# Use function WFO.one for a 1-to-1 match between submitted and matched names
WFO.one(e1)

# Hybrid species
WFO.match("Arabis divaricarpa", WFO.data=WFO.example)
WFO.match("Arabis x divaricarpa", WFO.data=WFO.example)

# Convert capitals to lower case
WFO.match("FAIDHERBIA ALBIDA", WFO.data=WFO.example, spec.name.tolower=TRUE)

# Remove sections of plant names that are equal to ' sp.' or ' indet. '
WFO.match("Prunus sp.", WFO.data=WFO.example, spec.name.sub=TRUE)

# Get urls, but do not open any
e2 <- WFO.match(spec.data=spec.test, WFO.data=WFO.example, counter=1, verbose=TRUE)
WFO.url(e2, browse=FALSE, browse.rows=c(1:nrow(e2)))

# Include input species names where no matches were found
# This happens when the name with original species names is provided to WFO.one
x1 <- WFO.match("World agroforestry", WFO.data=WFO.example)
WFO.one(x1, spec.name="spec.name")

## Not run:

# Cross-check with Taxonstand results
library(Taxonstand)
data(bryophytes)

# Give the file with the static copy of the Taxonomic Backbone data ('classification.txt')
# that was downloaded from \url{https://www.worldfloraonline.org/downloadData}.
# Possibly first use unzip(file.choose()) for the downloaded WFO_Backbone.zip
WFO.file.RK <- file.choose()

# check species name
w1 <- WFO.match(bryophytes[1:20, ], WFO.file=WFO.file.RK, spec.name="Full.name", counter=1)
w1


# check species name from list of names
w1 <- WFO.match(bryophytes$Full.name[1:20], WFO.file=WFO.file.RK, counter=1)

# re-check species names obtained via Taxonstand
```

```
# note that Taxonstand did not match some infraspecific names ('Higher.level')
r1 <- Taxonstand::TPL(bryophytes$Full.name[1:20], corr = TRUE)
w2 <- WFO.match(r1, WFO.file=WFO.file.RK, Genus="New.Genus", Species="New.Species",
       Infraspecific.rank="New.Infraspecific.rank", Infraspecific="New.Infraspecific", counter=1)
w2

# only check genus and species
# specify different names for infraspecific columns as default to Taxonstand
w3 <- WFO.match(r1, WFO.file=WFO.file.RK, Genus="New.Genus", Species="New.Species",
          Infraspecific.rank="none", Infraspecific="none", counter=1)

# note that the method above also retrieved infraspecific levels
# to only retrieve at the species level, match infraspecific levels with an empty column
r1$empty <- rep("", nrow(r1))
w4 <- WFO.match(r1, WFO.file=WFO.file.RK, Genus="New.Genus", Species="New.Species",
          Infraspecific.rank="empty", Infraspecific="empty", counter=1)

# as an alternative to the method above, exclude all documented infraspecific levels
# from the results
w5 <- WFO.match(r1, WFO.file=WFO.file.RK, Genus="New.Genus", Species="New.Species",
          exclude.infraspecific=TRUE, counter=1)

# save results to file
# utils::write.table(w4, quote=F, sep="\t", row.names=F, append=FALSE)

# limit the fuzzy matches to those that contain a shortened version of a species name
w6 <- WFO.match("Acacia caes", WFO.file=WFO.file.RK, Fuzzy=0.01, Fuzzy.within=TRUE, verbose=TRUE)

# show all the matches for a genus
spec.test1 <- data.frame(Genus=c("Casimiroa"))
w8 <- WFO.match(spec.test1, WFO.file=WFO.file.RK, exclude.infraspecific=TRUE, verbose=TRUE)

# show all listings at a next hierarchical level
WFO.data1 <- data.table::fread(WFO.file.RK, encoding="UTF-8")

WFO.browse("Pinaceae", WFO.data=WFO.data1)
WFO.browse("Pinaceae", WFO.data=WFO.data1, accepted.only=T)

WFO.browse("Tsuga", WFO.data=WFO.data1)
WFO.browse("Tsuga", WFO.data=WFO.data1, accepted.only=T)

WFO.browse("Olea europaea", WFO.data=WFO.data1)
WFO.browse("Olea europaea", WFO.data=WFO.data1, accepted.only=T)

# browsing only works at family, genus and species levels
# for orders, however, information is given from vascular.families
WFO.browse("Polypodiales", WFO.data=WFO.data1)

# submitting no name results in a list of all families
WFO.browse(, WFO.data=WFO.data1)

# give synonyms
WFO.synonyms("Olea europaea", WFO.data=WFO.data1)
```

```
# give order and other higher levels from family
WFO.family("Olea europaea", WFO.data=WFO.data1)

## End(Not run)
```

---

`WFO.match.fuzzyjoin` *Standardize plant names according to World Flora Online taxonomic backbone*

---

## Description

An alternative and typically faster method of matching records than WFO.match that allows for different methods of calculating the fuzzy distance via stringdist.

## Usage

```
WFO.match.fuzzyjoin(spec.data = NULL, WFO.file = NULL, WFO.data = NULL,
    no.dates = TRUE,
    spec.name = "spec.name",
    Authorship = "Authorship",
    stringdist.method = "lv", fuzzydist.max = 4,
    Fuzzy.min = TRUE,
    acceptedNameUsageID.match = TRUE,
    squish = TRUE,
 spec.name.tolower = FALSE, spec.name.nonumber = TRUE, spec.name.nobrackets = TRUE,
    spec.name.sub = TRUE,
 sub.pattern=c(" sp[.] A", " sp[.] B", " sp[.] C", " sp[.]", " spp[.]", " pl[.]",
    " indet[.]", " ind[.]", " gen[.]", " g[.]", " fam[.]", " nov[.]", " prox[.]",
        " cf[.]", " aff[.]", " s[.]s[.]", " s[.]l[.]",
        " p[.]p[.]", " p[.] p[.]", "[?]", " inc[.]", " stet[.]", "Ca[.]",
     "nom[.] cons[.]", "nom[.] dub[.]", " nom[.] err[.]", " nom[.] illeg[.]",
     " nom[.] inval[.]", " nom[.] nov[.]", " nom[.] nud[.]", " nom[.] obl[.]",
    " nom[.] prot[.]", " nom[.] rej[.]", " nom[.] supp[.]", " sensu auct[.]"))
```

## Arguments

| | |
|---|---|
| `spec.data` | A data.frame containing variables with species names. In case that a character vector is provided, then this vector will be converted to a data.frame |
| `WFO.file` | File name of the static copy of the Taxonomic Backbone. If not `NULL`, then data will be reloaded from this file. |
| `WFO.data` | Data set with the static copy of the Taxonomic Backbone. Ignored if `WFO.file` is not `NULL`. |
| `no.dates` | Speeding up the loading of the WFO.data by not loading fields of 'created' and 'modified'. |
| `spec.name` | Name of the column with taxonomic names. |

`Authorship` Name of the column with the naming authorities.

`stringdist.method`
Method used to calculate the fuzzy distance as used by in the internally called `stringdist`.

`fuzzydist.max` Maximum distance used for joining as in `stringdist_join`.

`Fuzzy.min` Limit the results of fuzzy matching to those with the smallest distance.

`acceptedNameUsageID.match`
If `TRUE`, obtain the accepted name and others details from the earlier acceptedNameUsageID.

`squish` If `TRUE`, remove repeated whitespace and white space from the start and end of the submitted full name via str_squish.

`spec.name.tolower`
If `TRUE`, then convert all characters of the `spec.name` to lower case via tolower.

`spec.name.nonumber`
If `TRUE`, then submitted `spec.name` that contain numbers will be interpreted as genera, only matching the first word.

`spec.name.nobrackets`
If `TRUE`, then submitted `spec.name` then sections of the submitted name after '(' will be removed. Note that this will also remove sections after ')', such as authorities for plant names that are in a separate column of WFO.

`spec.name.sub` If `TRUE`, then delete sections of the `spec.name` that match the `sub.pattern`.

`sub.pattern` Sections of the `spec.name` to be deleted

## Details

This function matches plant names by using the `stringdist_left_join` function internally. The results are provided in a similar formatto those from `WFO.match`; therefore the `WFO.one` function can be used in a next step of the analysis.

For large data sets the function may fail due to memory limits. A solution is to analyse different subsets of large data, as for example shown by Kindt (2023).

Column 'Unique' shows whether there was a unique match (or not match) in the WFO.

Column 'Matched' shows whether there was a match in the WFO.

Column 'Fuzzy' shows whether matching was done by the fuzzy method.

Column 'Fuzzy.dist' gives the fuzzy distance calculated between submitted and matched plant names, calculated internally with stringdist_left_join.

Column 'Auth.dist' gives the Levenshtein distance calculated between submitted and matched authorship names, if the former were provided. This distance is calculated in the same way as for the WFO.match function via adist.

Column 'Subseq' gives different numbers for different matches for the same plant name.

Column 'Hybrid' shows whether there was a hybrid character in the scientificName.

Column 'New.accepted' shows whether the species details correspond to the current accepted name.

Column 'Old.status' gives the taxonomic status of the first match with the non-blank acceptedNameUsageID.

Column 'Old.ID' gives the ID of the first match with the non-blank acceptedNameUsageID.

Column 'Old.name' gives the name of the first match with the non-blank acceptedNameUsageID.

## Value

The main function returns a data.set with the matched species details from the WFO.

## Author(s)

Roeland Kindt (World Agroforestry, CIFOR-ICRAF)

## References

World Flora Online. An Online Flora of All Known Plants. https://www.worldfloraonline.org

Sigovini M, Keppel E, Tagliapietra. 2016. Open Nomenclature in the biodiversity era. Methods in Ecology and Evolution 7: 1217-1225.

Kindt, R. 2020. WorldFlora: An R package for exact and fuzzy matching of plant names against the World Flora Online taxonomic backbone data. Applications in Plant Sciences 8(9): e11388

Kindt, R. 2023. Standardizing tree species names of GlobalTreeSearch with WorldFlora while testing the faster matching function of WFO.match.fuzzyjoin. https://rpubs.com/Roeland-KINDT/996500

## See Also

WFO.match

## Examples

```
## Not run:
data(WFO.example)

library(fuzzyjoin)

spec.test <- data.frame(spec.name=c("Faidherbia albida", "Acacia albida",
    "Faidherbia albiad",
    "Omalanthus populneus", "Pygeum afric"))

WFO.match.fuzzyjoin(spec.data=spec.test, WFO.data=WFO.example)

# Using the Damerau-Levenshtein distance
WFO.match.fuzzyjoin(spec.data=spec.test, WFO.data=WFO.example,
    stringdist.method="dl")

## End(Not run)
```

---

# WFO.prepare *Prepare a data set for analysis with WFO.match*

---

## Description

The main function of WFO.prepare attempts to split a list of species names with naming authorities in different fields of botanical names and authorities.

## Usage

```
WFO.prepare(spec.data = NULL, spec.full="spec.full",
    squish = TRUE, spec.name.nonumber = TRUE,
        spec.name.sub = TRUE,
     sub.pattern = c(" sp[.] A", " sp[.] B", " sp[.] C", " sp[.]", " spp[.]", " pl[.]",
        " indet[.]", " ind[.]", " gen[.]", " g[.]", " fam[.]", " nov[.]", " prox[.]",
            " cf[.]", " aff[.]", " s[.]s[.]", " s[.]l[.]",
            " p[.]p[.]", " p[.] p[.]", "[?]", " inc[.]", " stet[.]", "Ca[.]",
         "nom[.] cons[.]", "nom[.] dub[.]", " nom[.] err[.]", " nom[.] illeg[.]",
         " nom[.] inval[.]", " nom[.] nov[.]", " nom[.] nud[.]", " nom[.] obl[.]",
        " nom[.] prot[.]", " nom[.] rej[.]", " nom[.] supp[.]", " sensu auct[.]"),
    genus.2.flag = TRUE, species.2.flag = TRUE,
    punctuation.flag = TRUE, pointless.flag = TRUE,
    trinomial = c("cultivar.", "f.", "sect.", "subf.", "subg.",
        "subsp.", "subvar.", "var.",
        "CULTIVAR.",  "SECT.", "SUBF.", "SUBG.", "SUBSP.", "SUBVAR.", "VAR."),
    authors.ending.f=c("Aiton f.", "Baker f.", "Bak. f.", "Burm. f.",
                       "Cheng f.", "Chrtek f.",
                       "De Marco f.", "Fang f.", "Ferry f.", "Forsyth f.",
                       "Forster f.", "Fraser f.", "G.Don f.", "Haller f.",
                       "Hallier f.", "Hook. f.", "Hooker f.", "Hsueh f.",
                       "J.Kickx f.", "J. Kickx f.", "Keng f.",
                       "Kickx f.", "Klokov f.", "Koster f.",
                       "Liou f.", "L. f.", "Ma f.", "Mikan f.",
                       "Occhioni f.", "Rchb. f.",
                       "Schultes f.", "Schult. f.", "Stapf f."),
    verbose = TRUE, counter = 1000)

WFO.preprepare(spec.data=NULL, spec.full="spec.full",
    trinomial.first="subsp.", trinomial.second="var.")
```

## Arguments

| | |
|---|---|
| spec.data | A data.frame containing variables with species names. In case that a character vector is provided, then this vector will be converted to a data.frame |
| spec.full | Name of the column with full taxonomic names. |
| squish | If TRUE, remove repeated whitespace and white space from the start and end of the submitted full name via str_squish. |
| spec.name.nonumber | If TRUE, then submitted spec.full that contain numbers will be interpreted as genera, only matching the first word. |
| spec.name.sub | If TRUE, then delete sections of the spec.full that match the sub.pattern. |
| sub.pattern | Sections of the spec.full to be deleted |
| genus.2.flag | Flag first part of the names with only 2 characters. |
| species.2.flag | Flag second part of the names with only 2 characters. |

`punctuation.flag`
: Flag if the retained plant name has punctuation characters.

`pointless.flag`
: Flag if the retained plant name has sub.pattern without the point.

`trinomial`
: Descriptors for trinomial names. In case a trinomial name is expected, the species name will be obtained from the first two words and the two words starting with the trinomial descriptor.

`authors.ending.f`
: Author names that end with ' f.', not confuse the function about trinomials with 'f.', indicating 'filius' ('son of').

`verbose`
: Give details on the process.

`counter`
: Progress on the process is reported by multiples of this counter.

`trinomial.first`
: Pattern to split species name in different columns.

`trinomial.second`
: Second pattern to split species name in different columns.

## Details

Function `WFO.prepare` splits submitted names into the botanical name ('spec.name') and the naming authority ('Authorship'). When the submitted name contains section between brackets that are not at the beginning of the naming authority, these sections will be removed. Function `WFO.preprepare` was designed to deal with situations where author names are given at species and infra-specific levels (see examples).

## Value

The function splits names in the botanical name and the naming authority.

## Author(s)

Roeland Kindt (World Agroforestry)

## Examples

```
## Not run:
WFO.prepare("Terminalia superba Engl. & Diels (**) (In review)")
WFO.prepare("Sorbus aucuparia subsp. praemorsa (Guss.) Nyman")
WFO.prepare("Ormosia aff. coarctata Jackson")
WFO.prepare("Ormosia aff coarctata Jackson")
WFO.prepare("Ormosia /coarctata Jackson")
WFO.prepare("Qualea TMG 148 Aubl.")
# Note that the sub.pattern is ' cf.'
WFO.prepare("cf Myrcia M1")
# Dealing with author names that end with ' f.' ('filius')
WFO.prepare("Malveopsis scabrosum Stapf f.")

# Using preprepare to deal with authorities at multiple levels
WFO.preprepare("Agave deserti Engelm. subsp. simplex Gentry")
WFO.preprepare("Zoysia matrella (L.) Merr. var. pacifica Goudsw.")
```

```
test.name <- paste0("Agastache pallidiflora (A. Heller) Rydb.",
  " subsp. neomexicana (Briq.) Lint & Epling",
  " var. havardii (A. Gray) R.W. Sanders")
WFO.preprepare(test.name)

## End(Not run)
```

---

`WFO.remember` *Remember the location of the Taxonomic Backbone data set*

---

## Description

The function remembers where the Taxonomic Backbone data was downloaded to. In case that no arguments are specified, then data.frame `WFO.data` will contain the previously specified Taxonomic Backbone data.

## Usage

```
WFO.download(WFO.url =
  paste0("https://files.worldfloraonline.org/files/WFO_Backbone/",
         "_WFOCompleteBackbone/WFO_Backbone.zip"),
save.dir = getwd(), WFO.remember = TRUE,
timeout = 500, ...)

WFO.remember(WFO.file = NULL, WFO.data = "WFO.data", WFO.pos = 1)
```

## Arguments

| | |
|---|---|
| `WFO.url` | Hyperlink to the download from the World Flora Online. |
| `save.dir` | Directory where the file will be downloaded and unzipped. |
| `WFO.remember` | Remember the location of the file for `WFO.remember`. |
| `timeout` | Timeout in seconds for some internet operations, to be modified among Options Settings. |
| `...` | Other arguments for download.file. |
| `WFO.file` | File path to the Taxonomic Backbone data ('classification.txt'). |
| `WFO.data` | Name of data set to be used by other WorldFlora functions. |
| `WFO.pos` | Argument pos as in assign. |

## Details

These functions avoid that a user needs to reload and re-specify the location of the Taxonomic Backbone data that was previously downloaded from the World Flora Online website. The location is saved in a text file in the 'etc' directory of the WorldFlora directory.

## Value

The function remembers the local location of the Taxonomic Backbone data.

## Author(s)

Roeland Kindt (World Agroforestry)

## Examples

```
## Not run:

# change the working directory
setwd(choose.dir())

# download the Taxonomic Backbone data
WFO.download()

# remember the previous download and avail the data as 'WFO.data'
WFO.remember()

# check
WFO.match("Faidherbia albida", WFO.data=WFO.data)

## End(Not run)
```

# Index