Package: gedcomS7 (via r-universe)

May 29, 2026

Title Create Family Tree GEDCOM Files with S7 Objects

Version 0.4.0

Description Import, create, edit and export family tree GEDCOM files. Genealogical Data Communication (GEDCOM) is a flat text file format which contains many nested data structures representing genealogical records. This package automates away the complexity in constructing these files, representing them as S7 objects.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE, roclets = c(``namespace", ``rd", ``roxytest::testthat_roclet"))

RoxygenNote 7.3.3

Depends R (>= 4.1)

Imports S7, tools

Suggests testthat (>= 3.0.0), roxytest, withr, uuid, rmarkdown

Config/testthat/edition 3

URL https://jl5000.github.io/gedcomS7/, https://github.com/jl5000/gedcomS7

Repository https://jl5000.r-universe.dev

Date/Publication 2026-02-28 00:34:58 UTC

RemoteUrl https://github.com/jl5000/gedcomS7

RemoteRef HEAD

RemoteSha 9a251dd27fe0b3ef0926e494c1a4e6fe87591188

Contents

add_children 3
add_parents 4
add_siblings 5

add_spouse . . . 6
Address . . . 6
Association . . . 7
ChangeDate . . . 8
CreationDate . . . 9
date_exact_current . . . 9
DateApprox . . . 10
DateCalendar . . . 10
DateExact . . . 11
DatePeriod . . . 11
DateRange . . . 12
DateSorting . . . 12
DateValue . . . 13
df_fam . . . 13
df_fam_facts . . . 14
df_indi . . . 14
df_indi_facts . . . 15
df_media . . . 15
df_note . . . 16
df_repo . . . 16
df_sour . . . 17
df_subm . . . 17
fact_rules_df . . . 18
FactsRecorded . . . 18
FamilyAttribute . . . 19
FamilyEvent . . . 21
FamilyLinkChild . . . 24
FamilyLinkSpouse . . . 25
FamilyRecord . . . 25
GedcomHeader . . . 27
GedcomS7 . . . 28
GedcomSource . . . 29
get_ancestors . . . 30
get_descendants . . . 31
get_fam_as_child . . . 32
get_fam_as_spouse . . . 33
get_fam_children . . . 33
get_fam_partners . . . 34
get_indi_children . . . 34
get_indi_cousins . . . 35
get_indi_fathers . . . 35
get_indi_mothers . . . 36
get_indi_parents . . . 36
get_indi_partners . . . 37
get_indi_siblings . . . 37
get_records_by_pattern . . . 38
get_supporting_recs . . . 38
get_unused_recs . . . 39

IndividualAttribute 40
IndividualEvent 42
IndividualRecord 44
MediaFile 46
MediaLink 47
MediaRecord 48
new_gedcom 49
NonEvent 50
Note 50
NoteRecord 51
Ordinance 53
parse_gedcom_age 54
parse_gedcom_date 54
PersonalName 55
PersonalNamePieces 56
PersonalNameTran 56
Place 57
pull_record 58
push_record 59
read_gedcom 59
RepositoryCitation 60
RepositoryRecord 60
rm_living 62
rm_records 63
SourceCallNumber 63
SourceCitation 64
SourceRecord 65
SpouseSealing 67
SubmitterRecord 68
Time 70
TranslationText 71
val_record_types 71
write_gedcom 72

Index **74**

`add_children` *Create children records for a family*

Description

Create children records for a family

Usage

```
add_children(x, xref, sexes, chil_names = NULL, pedigrees = "BIRTH")
```

Arguments

x	A gedcom object.
xref	The xref of a Family record.
sexes	A character string giving the sexes (from val_sexes()) of each child. For example, "FFM" to add two daughters and one son.
chil_names	A character vector of children's names. If provided, it must be the same length as the number of sexes. If you don't want to provide a name for a child, set the name to "". Surnames must be enclosed in forward slashes. If all names you supply do not contain forward slashes then surnames will be taken from the father (or mother).
pedigrees	A character vector of pedigrees from val_pedigree_types(). This must be a vector of length one (which will be recycled) or the same size as the number of children. A value of NULL means no pedigrees will be defined. If you don't want to provide a pedigree for a child, set the pedigree to "". Defaults to "BIRTH".

Value

A gedcom object with additional child records.

---

add_parents *Add parent records for an individual*

Description

This function adds records for an individual's parents.

Usage

```
add_parents(x, xref, inc_sex = TRUE, fath_name = NULL, moth_name = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
inc_sex	Whether to populate the sex of the parents. This will ensure that there is one male and one female parent. Otherwise the sex will be assigned as "U" (undetermined).
fath_name, moth_name	Optional names to give to the parents. Surnames must be enclosed in forward slashes.

Details

This function may also create a Family record and will not modify existing parents.

Value

A gedcom object with additional parent records.

`add_siblings` *Create sibling records for an individual*

Description

Create sibling records for an individual

Usage

```
add_siblings(x, xref, sexes, sib_names = NULL, pedigrees = "BIRTH")
```

Arguments

`x`	A gedcom object.
`xref`	The xref of an Individual record.
`sexes`	A character string giving the sexes (from `val_sexes()`) of each sibling. For example, "FFM" to add two sisters and one brother.
`sib_names`	A character vector of sibling's names. If provided, it must be the same length as the number of sexes. If you don't want to provide a name for a sibling, set the name to "". Surnames must be enclosed in forward slashes. If all names you supply do not contain forward slashes then surnames will be taken from the father (or mother).
`pedigrees`	A character vector of pedigrees from `val_pedigree_types()`. This must be a vector of length one (which will be recycled) or the same size as the number of siblings. A value of NULL means no pedigrees will be defined. If you don't want to provide a pedigree for a sibling, set the pedigree to "". Defaults to "BIRTH".

Details

This function may also create a Family record and will not modify existing siblings.

Value

A gedcom object with additional sibling records.

add_spouse — *Add a spouse record for an individual*

Description

This creates a record for a spouse and potentially their Family record.

Usage

```
add_spouse(x, xref, sex = "U", spou_name = NULL, fam_xref = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
sex	The sex of the spouse.
spou_name	Optional name to give to the spouse. Surnames must be enclosed in forward slashes.
fam_xref	The cross-reference identifier of the Family record if it already exists. If this is not provided, a new Family record will be created.

Value

A gedcom object with additional spouse and Family records.

Address — *Create an address object*

Description

Create an address object

Usage

```
Address(
  full = character(0),
  city = character(0),
  state = character(0),
  postal_code = character(0),
  country = character(0),
  adr1 = character(0),
  adr2 = character(0),
  adr3 = character(0)
)
```

Arguments

full	A full address as it would appear on a mailing label, with lines separated by a new line character. For example: "The White House\n1600 Pennsylvania Avenue N.W.\nWashington\nDC 20500\nUnited States of America"
city	The city component of the address.
state	The state component of the address.
postal_code	The postal code component of the address.
country	The country component of the address.
adr1	Deprecated.
adr2	Deprecated.
adr3	Deprecated.

Value

An S7 object representing a GEDCOM ADDRESS_STRUCTURE.

Association *Create an association object*

Description

Create an association object

Usage

```
Association(
  indi_xref = "@VOID@",
  indi_phrase = character(0),
  relation_is = character(0),
  relation_phrase = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list()
)
```

Arguments

indi_xref	The cross-reference identifier of an individual record. If the individual does not have a record, then this can be left blank and a void xref will be used. However, you should define an @indi_phrase.
indi_phrase	Textual information that cannot be expressed in the @indi_xref.
relation_is	The nature of the association. This must be a value from `val_roles()`. If a value of "OTHER" is used, a @relation_phrase must be given.

`relation_phrase`
: Textual information that cannot be expressed in the relation.

`note_xrefs`
: A character vector of relevant note record cross-reference identifiers.

`notes`
: Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

`citations`
: Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

Value

An S7 object representing a GEDCOM ASSOCIATION_STRUCTURE.

`ChangeDate` *Create a change date object*

Description

Create a change date object

Usage

```
ChangeDate(
  date_exact = <object>,
  time = character(0),
  note_xrefs = character(0),
  notes = list()
)
```

Arguments

`date_exact`
: An exact date given either as a formatted GEDCOM string, or a `DateExact()` object. If not given, it will default to today's date.

`time`
: The time given either as a formatted GEDCOM string, or a `Time()` object.

`note_xrefs`
: A character vector of relevant note record cross-reference identifiers.

`notes`
: Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

Value

An S7 object representing a GEDCOM CHANGE_DATE.

CreationDate — *Create a creation date object*

Description

Create a creation date object

Usage

```
CreationDate(date_exact = <object>, time = character(0))
```

Arguments

`date_exact`	An exact date given either as a formatted GEDCOM string, or a `DateExact()` object. If not given, it will default to today's date.
`time`	The time given either as a formatted GEDCOM string, or a `Time()` object.

Value

An S7 object representing a GEDCOM CREATION_DATE.

date_exact_current — *Create a GEDCOM Exact Date object for today*

Description

Create a GEDCOM Exact Date object for today

Usage

```
date_exact_current()
```

Value

An S7 object representing a GEDCOM Exact Date for today.

DateApprox — *Create a GEDCOM Approximate Date object*

Description

Create a GEDCOM Approximate Date object

Usage

```
DateApprox(date_cal = character(0), about = TRUE, calc = FALSE, est = FALSE)
```

Arguments

date_cal	A Calendar date given either as a formatted GEDCOM string, or a DateCalendar object.
about	Whether the date is near to the date given.
calc	Whether the date is calculated from other values.
est	Whether the date is near to the date given, and is calculated from other values.

Value

An S7 object representing a GEDCOM Approximate Date.

DateCalendar — *Create a GEDCOM Calendar Date object*

Description

Create a GEDCOM Calendar Date object

Usage

```
DateCalendar(
  year = integer(0),
  month = integer(0),
  day = integer(0),
  julian = FALSE
)
```

Arguments

year	The year given as an integer (not 0). Negative years are interpreted as Before the Common Era (BCE). If BCE, only year should be provided.
month	The month of the year given as an integer between 1 and 12.
day	The day of the month given as an integer between 1 and 31.
julian	Whether the date is given in the Julian calendar. If not, it is assumed to be in the Gregorian calendar.

Value

An S7 object representing a GEDCOM Calendar Date.

DateExact — *Create a GEDCOM Exact Date object*

Description

Create a GEDCOM Exact Date object

Usage

```
DateExact(year = integer(0), month = integer(0), day = integer(0))
```

Arguments

`year`	The year given as an integer (greater than 0).
`month`	The month of the year given as an integer between 1 and 12.
`day`	The day of the month given as an integer between 1 and 31.

Value

An S7 object representing a GEDCOM Exact Date.

DatePeriod — *Create a GEDCOM Date Period object*

Description

Create a GEDCOM Date Period object

Usage

```
DatePeriod(start_date = character(0), end_date = character(0))
```

Arguments

`start_date`	The start of the period/range given either as a formatted GEDCOM string, or a `DateCalendar()` object.
`end_date`	The end of the period/range given either as a formatted GEDCOM string, or a `DateCalendar()` object.

Value

An S7 object representing a GEDCOM Date Period.

DateRange — *Create a GEDCOM Date Range object*

Description

Create a GEDCOM Date Range object

Usage

```
DateRange(start_date = character(0), end_date = character(0))
```

Arguments

start_date	The start of the period/range given either as a formatted GEDCOM string, or a DateCalendar() object.
end_date	The end of the period/range given either as a formatted GEDCOM string, or a DateCalendar() object.

Value

An S7 object representing a GEDCOM Date Range.

DateSorting — *Create a GEDCOM Sorting Date object*

Description

Create a GEDCOM Sorting Date object

Usage

```
DateSorting(
  date = character(0),
  date_phrase = character(0),
  time = character(0)
)
```

Arguments

date	The date given either as a formatted GEDCOM string, or a DateCalendar object.
date_phrase	Textual information that cannot be expressed in the date.
time	The time given either as a formatted GEDCOM string, or a Time() object.

Value

An S7 object representing a GEDCOM Sorting Date.

DateValue — *Create a GEDCOM Date Value object*

Description

Create a GEDCOM Date Value object

Usage

```
DateValue(date = character(0), date_phrase = character(0), time = character(0))
```

Arguments

`date`	The date given either as a formatted GEDCOM string, or a `DateValue()` object.
`date_phrase`	Textual information that cannot be expressed in the date.
`time`	The time given either as a formatted GEDCOM string, or a `Time()` object.

Value

An S7 object representing a GEDCOM Date Value.

df_fam — *Summarise Family records in a dataframe*

Description

Summarise Family records in a dataframe

Usage

```
df_fam(x, xrefs = NULL)
```

Arguments

`x`	A gedcom object.
`xrefs`	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Family record on each row.

df_fam_facts — *Summarise a family's attributes/events in a dataframe*

Description

Summarise a family's attributes/events in a dataframe

Usage

```
df_fam_facts(x, xref)
```

Arguments

x	A gedcom object.
xref	The cross-reference identifier of a family record.

Value

A dataframe summarising an attribute/event on each row.

df_indi — *Summarise Individual records in a dataframe*

Description

Summarise Individual records in a dataframe

Usage

```
df_indi(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising an Individual record on each row.

df_indi_facts — *Summarise an individual's attributes/events in a dataframe*

Description

Summarise an individual's attributes/events in a dataframe

Usage

```
df_indi_facts(x, xref)
```

Arguments

x	A gedcom object.
xref	The cross-reference identifier of an individual record.

Value

A dataframe summarising an attribute/event on each row.

df_media — *Summarise Multimedia records in a dataframe*

Description

Summarise Multimedia records in a dataframe

Usage

```
df_media(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Multimedia record on each row.

df_note — *Summarise Note records in a dataframe*

Description

Summarise Note records in a dataframe

Usage

```
df_note(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Note record on each row.

df_repo — *Summarise Repository records in a dataframe*

Description

Summarise Repository records in a dataframe

Usage

```
df_repo(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Repository record on each row.

df_sour — *Summarise Source records in a dataframe*

Description

Summarise Source records in a dataframe

Usage

```
df_sour(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Source record on each row.

df_subm — *Summarise Submitter records in a dataframe*

Description

Summarise Submitter records in a dataframe

Usage

```
df_subm(x, xrefs = NULL)
```

Arguments

x	A gedcom object.
xrefs	A vector of xrefs to summarise. If this is left NULL, all relevant xrefs will be used.

Value

A dataframe summarising a Submitter record on each row.

fact_rules_df *Property requirements for fact objects.*

Description

Property requirements for fact objects.

Usage

```
fact_rules_df(type = NULL)
```

Arguments

type	A character string to filter the table.

Value

A dataframe detailing the property requirements for each type of fact.

FactsRecorded *Create an object recording facts covered in a source record*

Description

Create an object recording facts covered in a source record

Usage

```
FactsRecorded(
  fact_types = character(0),
  date_period = character(0),
  date_phrase = character(0),
  territory = NULL
)
```

Arguments

fact_types	A character string indicating the types of events that were recorded in a particular source. Each event type is separated by a comma and space. For example, a parish register of births, deaths, and marriages would be BIRT, DEAT, MARR.
date_period	A date period given either as a formatted GEDCOM string, or a `DatePeriod()` object.
date_phrase	Textual information that cannot be expressed in the date.
territory	The territory associated with the events covered. This can either be a `Place()` object or a character string (a comma-separated string of region names, ordered from smallest to largest).

Value

An S7 object representing a GEDCOM SOUR.EVEN structure.

`FamilyAttribute` *Create a family attribute object*

Description

Create a family attribute object

Usage

```
FamilyAttribute(
  fact_type = character(0),
  fact_val = character(0),
  fact_desc = character(0),
  date = character(0),
  place = NULL,
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  agency = character(0),
  relig_affil = character(0),
  cause = character(0),
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  date_sort = character(0),
  associations = list(),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  unique_ids = character(0),
  husb_age = character(0),
  husb_age_phrase = character(0),
  wife_age = character(0),
  wife_age_phrase = character(0)
)
```

Arguments

`fact_type` A code indicating the type of fact. This must be taken from one of `val_individual_event_types()`, `val_individual_attribute_types()`, `val_family_event_types()`, or `val_family_attribute_typ` A generic event ("EVEN") or attribute ("FACT") can also be defined for more

bespoke facts. See `fact_rules_df()` for the set of rules surrounding the codes allowed.

`fact_val` A value associated with the fact. For example for "NCHI" this would be the number of children. See `fact_rules_df()` for the set of rules surrounding the need for values and the values allowed.

`fact_desc` A further classification of the fact. This is required for generic events or attributes. See `fact_rules_df()` for the set of rules surrounding the need for this.

`date` The date given either as a formatted GEDCOM string, or a `DateValue()` object.

`place` The associated place. This can either be a `Place()` object or a character string (a comma-separated string of region names, ordered from smallest to largest).

`address` The address given either as a `Address()` object or as a character string. This would be as written on a mailing label with new lines separated by \n.

`phone_numbers` A character vector of phone numbers.

`emails` A character vector of email addresses.

`faxes` A character vector of fax numbers.

`web_pages` A character vector of web page URLs.

`agency` The organization, institution, corporation, person, or other entity that has responsibility for the associated fact. Examples are an employer of a person of an associated occupation, or an educational establishment that has awarded a scholastic award.

`relig_affil` A religious denomination associated with the fact.

`cause` Used in special cases to record the reasons which precipitated the fact (e.g. cause of death).

`confidential` A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.

`locked` A logical value indicating whether the associated record/fact should be treated as read-only.

`private` A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.

`date_sort` The date given either as a formatted GEDCOM string, or a `DateSorting()` object.

`associations` Associated individuals. This can either be a `Association()` object or a list of them.

`note_xrefs` A character vector of relevant note record cross-reference identifiers.

`notes` Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

`citations` Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

`media_links` Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.

`unique_ids` A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.

`husb_age` A character string that indicates the age in years, months, weeks and/or days that the husband was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.

`husb_age_phrase` Free text information that cannot be expressed in the husband's age.

`wife_age` A character string that indicates the age in years, months, weeks and/or days that the wife was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.

`wife_age_phrase` Free text information that cannot be expressed in the wife's age.

Value

An S7 object representing a GEDCOM FAMILY_ATTRIBUTE_STRUCTURE.

`FamilyEvent` *Create a family event object*

Description

Create a family event object

Usage

```
FamilyEvent(
  fact_type = character(0),
  fact_val = character(0),
  fact_desc = character(0),
  date = character(0),
  place = NULL,
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  agency = character(0),
  relig_affil = character(0),
  cause = character(0),
```

```
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  date_sort = character(0),
  associations = list(),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  unique_ids = character(0),
  husb_age = character(0),
  husb_age_phrase = character(0),
  wife_age = character(0),
  wife_age_phrase = character(0)
)
```

Arguments

fact_type	A code indicating the type of fact. This must be taken from one of val_individual_event_types(), val_individual_attribute_types(), val_family_event_types(), or val_family_attribute_typ A generic event ("EVEN") or attribute ("FACT") can also be defined for more bespoke facts. See fact_rules_df() for the set of rules surrounding the codes allowed.
fact_val	A value associated with the fact. For example for "NCHI" this would be the number of children. See fact_rules_df() for the set of rules surrounding the need for values and the values allowed.
fact_desc	A further classification of the fact. This is required for generic events or attributes. See fact_rules_df() for the set of rules surrounding the need for this.
date	The date given either as a formatted GEDCOM string, or a DateValue() object.
place	The associated place. This can either be a Place() object or a character string (a comma-separated string of region names, ordered from smallest to largest).
address	The address given either as a Address() object or as a character string. This would be as written on a mailing label with new lines separated by \n.
phone_numbers	A character vector of phone numbers.
emails	A character vector of email addresses.
faxes	A character vector of fax numbers.
web_pages	A character vector of web page URLs.
agency	The organization, institution, corporation, person, or other entity that has responsibility for the associated fact. Examples are an employer of a person of an associated occupation, or an educational establishment that has awarded a scholastic award.
relig_affil	A religious denomination associated with the fact.
cause	Used in special cases to record the reasons which precipitated the fact (e.g. cause of death).

`confidential` A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.

`locked` A logical value indicating whether the associated record/fact should be treated as read-only.

`private` A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.

`date_sort` The date given either as a formatted GEDCOM string, or a `DateSorting()` object.

`associations` Associated individuals. This can either be a `Association()` object or a list of them.

`note_xrefs` A character vector of relevant note record cross-reference identifiers.

`notes` Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

`citations` Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

`media_links` Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.

`unique_ids` A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.

`husb_age` A character string that indicates the age in years, months, weeks and/or days that the husband was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.

`husb_age_phrase` Free text information that cannot be expressed in the husband's age.

`wife_age` A character string that indicates the age in years, months, weeks and/or days that the wife was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.

`wife_age_phrase` Free text information that cannot be expressed in the wife's age.

Value

An S7 object representing a GEDCOM FAMILY_EVENT_STRUCTURE.

FamilyLinkChild *Create a family link (as child) object*

Description

Create a family link (as child) object

Usage

```
FamilyLinkChild(
  fam_xref = character(0),
  note_xrefs = character(0),
  notes = list(),
  pedigree = character(0),
  pedigree_phrase = character(0),
  confidence = character(0),
  confidence_phrase = character(0)
)
```

Arguments

`fam_xref`	The cross-reference identifier of a family record.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`pedigree`	An optional value from `val_pedigree_types()` indicating the nature of the link.
`pedigree_phrase`	An optional free-text phrase describing the nature of the link. This is required if `@pedigree` is "OTHER".
`confidence`	An optional value from `val_confidence_types()` indicating the confidence of the link.
`confidence_phrase`	An optional free-text phrase expanding on the confidence of the link.

Value

An S7 object representing a GEDCOM family link as a child.

FamilyLinkSpouse *Create a family link (as spouse) object*

Description

Create a family link (as spouse) object

Usage

```
FamilyLinkSpouse(
  fam_xref = character(0),
  note_xrefs = character(0),
  notes = list()
)
```

Arguments

fam_xref	The cross-reference identifier of a family record.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

Value

An S7 object representing a GEDCOM family link as a spouse.

FamilyRecord *Create a family record object*

Description

Create a family record object

Usage

```
FamilyRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
```

```
  citations = list(),
  media_links = list(),
  created = NULL,
  updated = NULL,
  facts = list(),
  non_events = list(),
  husb_xref = character(0),
  wife_xref = character(0),
  chil_xrefs = character(0),
  associations = list(),
  subm_xrefs = character(0),
  spouse_sealings = list()
)
```

Arguments

`XREF`	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
`confidential`	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
`locked`	A logical value indicating whether the associated record/fact should be treated as read-only.
`private`	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
`user_ids`	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
`unique_ids`	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
`ext_ids`	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`citations`	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.
`media_links`	Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.
`created`	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
`updated`	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
`facts`	Events and/or attributes for this family. A `FamilyEvent()`/`FamilyAttribute()` object, or a list of them.

`non_events` Events that this family did not experience. A `NonEvent()` object, or a list of them.

`husb_xref`, `wife_xref`, `chil_xrefs`
The cross-reference identifier(s) of member's individual records. If the individual does not have a record, then the value "@VOID@" can be used. However, you will need to describe the individual by using a named vector (a description can be used in either case), e.g. c("Joe Bloggs" = "@VOID@") or c("Joe Bloggs" = "@I1@").

`associations` Associated individuals. This can either be a `Association()` object or a list of them.

`subm_xrefs` A character vector of relevant submitter record cross-reference identifiers.

`spouse_sealings`
A `SpouseSealing()` object or a list of them detailing the sealing of a husband and wife in a temple ceremony of The Church of Jesus Christ of Latter-day Saints.

Value

An S7 object representing a GEDCOM FAMILY_RECORD.

`GedcomHeader` *Create a GEDCOM header object*

Description

Create a GEDCOM header object

Usage

```
GedcomHeader(
  gedcom_version = character(0),
  ext_tags = character(0),
  source = NULL,
  destination = character(0),
  creation_date = <object>,
  creation_time = character(0),
  subm_xref = character(0),
  gedcom_copyright = character(0),
  default_language = character(0),
  default_place_form = character(0),
  notes = list(),
  note_xrefs = character(0)
)
```

Arguments

- `gedcom_version` The version number of the official specification that this GEDCOM conforms to. This must include the major and minor version (for example, "7.0"); it may include the patch as well (for example, "7.0.1"), but doing so is not required.
- `ext_tags` Not supported.
- `source` A `GedcomSource()` object describing the software that has generated the GEDCOM.‘
- `destination` An identifier for the system expected to receive this GEDCOM.
- `creation_date` The creation date of the file given either as a formatted GEDCOM string, or a `DateExact()` object.
- `creation_time` The creation time of the file given either as a formatted GEDCOM string, or a `Time()` object.
- `subm_xref` The cross-reference identifier of the primary submitter.
- `gedcom_copyright` A copyright statement, as appropriate for the copyright laws applicable to this data.
- `default_language` The default language for the entire GEDCOM object.
- `default_place_form` The default form for place names in the GEDCOM object. A comma-separated string of jurisdictional titles. For example "City, County, State, Country".
- `notes` Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
- `note_xrefs` A character vector of relevant note record cross-reference identifiers.

Value

An S7 object representing a GEDCOM header.

GedcomS7 *Create a GEDCOM object*

Description

You shouldn't need to use this directly to create new GEDCOM objects. Instead, use `new_gedcom()` which populates relevant defaults.

Usage

```
GedcomS7(
  header = GedcomHeader(),
  records = GedcomRecords(),
  update_change_dates = FALSE,
  add_creation_dates = FALSE
)
```

Arguments

`header` An S7 object whose properties contain information about the GEDCOM object as a whole.

`records` An S7 object whose properties contain information about all records. Do not edit properties in capitals directly. See Details for more information.

`update_change_dates` Whether to automatically update change dates when updating records. This happens when the record is pushed to the gedcom object.

`add_creation_dates` Whether to automatically add creation dates when creating records. This happens when the record is pushed to the gedcom object.

Details

All information about records is contained in the `@records` property.

The `@prefixes` property is a named vector containing any alphanumeric string (up to 6 characters long) which will precede the number given to identify new records, of which there are currently 7 types:

Individual (INDI) Family (FAM) Source (SOUR) Repository (REPO) Multimedia (OBJE) Note (SNOTE) Submitter (SUBM)

This vector must be of a particular length with specific names. For example: c(SUBM = "U", INDI = "I", FAM = "F", SOUR = "S", REPO = "R", OBJE = "M", SNOTE = "N").

The order that these records appear in the vector will also dictate the order in which records will appear in the exported file.

Value

An S7 object representing a GEDCOM file.

`GedcomSource` *Create a GEDCOM source object*

Description

Create a GEDCOM source object

Usage

```
GedcomSource(
  product_id = character(0),
  product_name = character(0),
  product_version = character(0),
  business_name = character(0),
  business_address = NULL,
  phone_numbers = character(0),
```

```
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  data_name = character(0),
  data_pubdate = character(0),
  data_pubtime = character(0),
  data_copyright = character(0)
)
```

Arguments

- `product_id` An identifier for the product producing this dataset.
- `product_name` The name of the product producing this dataset.
- `product_version` The version of the product producing this dataset.
- `business_name` The name of the business, corporation, or person that produced or commissioned the product.
- `business_address` The address of the business, corporation, or person that produced or commissioned the product. The address is given either as a `Address()` object or as a character string. This would be as written on a mailing label with new lines separated by \n.
- `phone_numbers` A character vector of phone numbers.
- `emails` A character vector of email addresses.
- `faxes` A character vector of fax numbers.
- `web_pages` A character vector of web page URLs.
- `data_name` Deprecated.
- `data_pubdate` Deprecated.
- `data_pubtime` Deprecated.
- `data_copyright` Deprecated.

Value

An S7 object representing a GEDCOM HEAD.SOUR.

`get_ancestors` *Identify all ancestors for an individual*

Description

This function identifies records in an entire branch of the family tree above a certain individual.

Usage

```
get_ancestors(
  x,
  xref,
  inc_indi = FALSE,
  inc_sibs = FALSE,
  inc_fam = FALSE,
  inc_supp = FALSE,
  pedigrees = NULL
)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
inc_indi	Whether to also include the individual themselves.
inc_sibs	Whether to also include all siblings of ancestors (siblings of this individual will only be included if the individual is included).
inc_fam	Whether to also include all Family records where this individual is a child (and all ancestors' Family records).
inc_supp	Whether to also include all supporting records (Note, Source, Repository, Multimedia).
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

---

get_descendants *Identify all descendants for an individual*

Description

This function identifies records in an entire branch of the family tree below a certain individual.

Usage

```
get_descendants(
  x,
  xref,
  inc_indi = FALSE,
  inc_part = FALSE,
  inc_fam = FALSE,
```

```
  inc_supp = FALSE,
  pedigrees = NULL
)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
inc_indi	Whether to also include the individual themselves.
inc_part	Whether to also include all partners of this individual (and their descendants and descendants' partners).
inc_fam	Whether to also include all Family records where this individual is a partner (and all descendants' Family records).
inc_supp	Whether to also include all supporting records (Note, Source, Repository, Multimedia).
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

---

get_fam_as_child *Identify all families for an individual where they are a child*

Description

Identify all families for an individual where they are a child

Usage

```
get_fam_as_child(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_fam_as_spouse *Identify all families for an individual where they are a partner*

Description

Identify all families for an individual where they are a partner

Usage

```
get_fam_as_spouse(x, xref)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.

Value

A character vector of xrefs.

get_fam_children *Identify all children in a family*

Description

Identify all children in a family

Usage

```
get_fam_children(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of a Family record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_fam_partners *Identify all partners in a family*

Description

Identify all partners in a family

Usage

```
get_fam_partners(x, xref)
```

Arguments

x	A gedcom object.
xref	The xref of a Family record.

Value

A character vector of xrefs.

get_indi_children *Identify all children for an individual*

Description

Identify all children for an individual

Usage

```
get_indi_children(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_indi_cousins *Identify all cousins for an individual*

Description

Identify all cousins for an individual

Usage

```
get_indi_cousins(x, xref, degree = 1, inc_half = FALSE)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
degree	Whether to return first cousins (degree = 1), second cousins (degree = 2), etc.
inc_half	Whether to include half cousins.

Value

A character vector of xrefs.

get_indi_fathers *Identify all fathers for an individual*

Description

Identify all fathers for an individual

Usage

```
get_indi_fathers(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_indi_mothers *Identify all mothers for an individual*

Description

Identify all mothers for an individual

Usage

```
get_indi_mothers(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_indi_parents *Identify all parents for an individual*

Description

Identify all parents for an individual

Usage

```
get_indi_parents(x, xref, pedigrees = NULL)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).

Value

A character vector of xrefs.

get_indi_partners *Identify all partners for an individual*

Description

Identify all partners for an individual

Usage

```
get_indi_partners(x, xref)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.

Value

A character vector of xrefs.

get_indi_siblings *Identify all siblings for an individual*

Description

Identify all siblings for an individual

Usage

```
get_indi_siblings(x, xref, pedigrees = NULL, inc_half = FALSE)
```

Arguments

x	A gedcom object.
xref	The xref of an Individual record.
pedigrees	A character vector of allowed family-child linkages. By default, NULL means all pedigrees (e.g. inc. ADOPTED). If it includes "BIRTH" then this will also pick up non-existent values (since BIRTH is assumed to be the default).
inc_half	Whether to include siblings that only share one parent.

Value

A character vector of xrefs.

get_records_by_pattern

Identify all records that contain a pattern

Description

Identify all records that contain a pattern

Usage

```
get_records_by_pattern(x, pattern, return_context = FALSE)
```

Arguments

x	A gedcom object.
pattern	A regular expression. Case is ignored.
return_context	Whether to return a named list of matching lines in the GEDCOM (TRUE) or just the xrefs (FALSE, default).

Value

Either a vector of matching xrefs, or a named list of matching GEDCOM lines.

get_supporting_recs

Identify all supporting records for a set of records

Description

This function gets all supporting records (and onwards dependencies) for a set of records. Supporting records are note, multimedia, source, submitter and repository records, i.e. those providing supporting evidence and comments.

Usage

```
get_supporting_recs(
  x,
  xrefs,
  inc_note = TRUE,
  inc_media = TRUE,
  inc_sour = TRUE,
  inc_repo = TRUE,
  inc_subm = TRUE
)
```

Arguments

`x`	A gedcom object.
`xrefs`	The xrefs of records to get supporting records for.
`inc_note`	Whether to include Note records.
`inc_media`	Whether to include Multimedia records.
`inc_sour`	Whether to include Source records.
`inc_repo`	Whether to include Repository records.
`inc_subm`	Whether to include Submitter records.

Value

A character vector of xrefs.

`get_unused_recs` *Identify unreferenced records*

Description

This function identifies records that are not referenced in any other records.

Usage

```
get_unused_recs(x)
```

Arguments

`x`	A gedcom object.

Details

You would expect every record to be referenced by another in some way. For example, Individual records should reference Family records (and vice-versa), Repository records should be referenced by Source records, and Source records should be cited by other records.

Value

A character vector of xrefs.

IndividualAttribute *Create an individual attribute object*

Description

Create an individual attribute object

Usage

```
IndividualAttribute(
  fact_type = character(0),
  fact_val = character(0),
  fact_desc = character(0),
  date = character(0),
  place = NULL,
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  agency = character(0),
  relig_affil = character(0),
  cause = character(0),
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  date_sort = character(0),
  associations = list(),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  unique_ids = character(0),
  age = character(0),
  age_phrase = character(0)
)
```

Arguments

`fact_type`	A code indicating the type of fact. This must be taken from one of `val_individual_event_types()`, `val_individual_attribute_types()`, `val_family_event_types()`, or `val_family_attribute_typ` A generic event ("EVEN") or attribute ("FACT") can also be defined for more bespoke facts. See `fact_rules_df()` for the set of rules surrounding the codes allowed.
`fact_val`	A value associated with the fact. For example for "NCHI" this would be the number of children. See `fact_rules_df()` for the set of rules surrounding the need for values and the values allowed.

fact_desc	A further classification of the fact. This is required for generic events or attributes. See fact_rules_df() for the set of rules surrounding the need for this.
date	The date given either as a formatted GEDCOM string, or a DateValue() object.
place	The associated place. This can either be a Place() object or a character string (a comma-separated string of region names, ordered from smallest to largest).
address	The address given either as a Address() object or as a character string. This would be as written on a mailing label with new lines separated by \n.
phone_numbers	A character vector of phone numbers.
emails	A character vector of email addresses.
faxes	A character vector of fax numbers.
web_pages	A character vector of web page URLs.
agency	The organization, institution, corporation, person, or other entity that has responsibility for the associated fact. Examples are an employer of a person of an associated occupation, or an educational establishment that has awarded a scholastic award.
relig_affil	A religious denomination associated with the fact.
cause	Used in special cases to record the reasons which precipitated the fact (e.g. cause of death).
confidential	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
locked	A logical value indicating whether the associated record/fact should be treated as read-only.
private	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
date_sort	The date given either as a formatted GEDCOM string, or a DateSorting() object.
associations	Associated individuals. This can either be a Association() object or a list of them.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a Note() object, a list of them, or a character vector of notes.
citations	Associated sources. This can either be a SourceCitation() object, a list of them, or a character vector of XREFs of source records.
media_links	Associated multimedia. This can either be a MediaLink() object, a list of them, or a character vector of XREFs of multimedia records.
unique_ids	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with uuid::UUIDgenerate().
age	A character string that indicates the age in years, months, weeks and/or days that the individual was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.
age_phrase	Free text information that cannot be expressed in the individual's age.

Value

An S7 object representing a GEDCOM INDIVIDUAL_ATTRIBUTE_STRUCTURE.

IndividualEvent *Create an individual event object*

Description

Create an individual event object

Usage

```
IndividualEvent(
  fact_type = character(0),
  fact_val = character(0),
  fact_desc = character(0),
  date = character(0),
  place = NULL,
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  agency = character(0),
  relig_affil = character(0),
  cause = character(0),
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  date_sort = character(0),
  associations = list(),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  unique_ids = character(0),
  age = character(0),
  age_phrase = character(0),
  fam_xref = character(0),
  adop_parent = character(0),
  adop_parent_phrase = character(0)
)
```

Arguments

`fact_type` A code indicating the type of fact. This must be taken from one of `val_individual_event_types()`, `val_individual_attribute_types()`, `val_family_event_types()`, or `val_family_attribute_typ

A generic event ("EVEN") or attribute ("FACT") can also be defined for more bespoke facts. See `fact_rules_df()` for the set of rules surrounding the codes allowed.

`fact_val` A value associated with the fact. For example for "NCHI" this would be the number of children. See `fact_rules_df()` for the set of rules surrounding the need for values and the values allowed.

`fact_desc` A further classification of the fact. This is required for generic events or attributes. See `fact_rules_df()` for the set of rules surrounding the need for this.

`date` The date given either as a formatted GEDCOM string, or a `DateValue()` object.

`place` The associated place. This can either be a `Place()` object or a character string (a comma-separated string of region names, ordered from smallest to largest).

`address` The address given either as a `Address()` object or as a character string. This would be as written on a mailing label with new lines separated by \n.

`phone_numbers` A character vector of phone numbers.

`emails` A character vector of email addresses.

`faxes` A character vector of fax numbers.

`web_pages` A character vector of web page URLs.

`agency` The organization, institution, corporation, person, or other entity that has responsibility for the associated fact. Examples are an employer of a person of an associated occupation, or an educational establishment that has awarded a scholastic award.

`relig_affil` A religious denomination associated with the fact.

`cause` Used in special cases to record the reasons which precipitated the fact (e.g. cause of death).

`confidential` A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.

`locked` A logical value indicating whether the associated record/fact should be treated as read-only.

`private` A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.

`date_sort` The date given either as a formatted GEDCOM string, or a `DateSorting()` object.

`associations` Associated individuals. This can either be a `Association()` object or a list of them.

`note_xrefs` A character vector of relevant note record cross-reference identifiers.

`notes` Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

`citations` Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

`media_links` Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.

`unique_ids` A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.

`age` A character string that indicates the age in years, months, weeks and/or days that the individual was at the time of the fact. Any combination of these is permitted. It is recommended that this be an age from a cited source document. Any labels must come after their corresponding number, for example; "4y 8m 1w 3d". Age bounds can also be included, for example; "< 40y". If the age doesn't fit this format then describe the age in the corresponding phrase parameter.

`age_phrase` Free text information that cannot be expressed in the individual's age.

`fam_xref` The cross-reference identifier of a family record.

`adop_parent` The adopting parent. Must be a value from `val_adoptive_parents()`. For adoption events only.

`adop_parent_phrase` A free text description of the adoptive parent (if the value itself is insufficient).

Value

An S7 object representing a GEDCOM INDIVIDUAL_EVENT_STRUCTURE.

`IndividualRecord` *Create an individual record object*

Description

Create an individual record object

Usage

```
IndividualRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  created = NULL,
  updated = NULL,
  pers_names = list(),
  sex = "U",
  facts = list(),
  non_events = list(),
```

```
  ordinances = list(),
  fam_links_chil = list(),
  fam_links_spou = list(),
  subm_xrefs = character(0),
  associations = list(),
  alia_xrefs = character(0),
  anci_xrefs = character(0),
  desi_xrefs = character(0)
)
```

Arguments

XREF	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
confidential	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
locked	A logical value indicating whether the associated record/fact should be treated as read-only.
private	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
user_ids	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
unique_ids	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
ext_ids	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
citations	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.
media_links	Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.
created	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
updated	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
pers_names	A `PersonalName()` object or a list of them, giving the names of this individual. A simple character vector of names can be provided instead, but this is not recommended.
sex	The sex of the individual. Either "M" (male), "F" (female), "X" (other), or "U" (undetermined, the default).
facts	Events and/or attributes for this individual. An `IndividualEvent()`/`IndividualAttribute()` object, or a list of them.

non_events	Events that this individual did not experience. A NonEvent() object, or a list of them.
ordinances	An Ordinance() object, or a list of them.
fam_links_chil	A FamilyLinkChild() object or a list of them, giving the Family records that this individual is a member of as a child. A character vector of Family record xrefs can also be provided. This will be automatically updated if the individual's membership in a Family record changes.
fam_links_spou	A FamilyLinkSpouse() object or a list of them, giving the Family records that this individual is a member of as a spouse. A character vector of Family record xrefs can also be provided. This will be automatically updated if the individual's membership in a Family record changes.
subm_xrefs	A character vector of relevant submitter record cross-reference identifiers.
associations	Associated individuals. This can either be a Association() object or a list of them.
alia_xrefs	A named character vector of relevant individual record cross-reference identifiers whose records also represent this individual. The vector names may provide a description of these records.
anci_xrefs	A character vector of relevant submitter record cross-reference identifiers who are interested in the ancestors of this individual.
desi_xrefs	A character vector of relevant submitter record cross-reference identifiers who are interested in the descendants of this individual.

Value

An S7 object representing a GEDCOM INDIVIDUAL_RECORD.

MediaFile *Create a media file object*

Description

Create a media file object

Usage

```
MediaFile(
  location = character(0),
  title = character(0),
  media_type = character(0),
  medium = character(0),
  medium_phrase = character(0),
  media_alt = character(0)
)
```

Arguments

`location`	An absolute or relative URL to the file.
`title`	The title of the multimedia record.
`media_type`	The media type as defined in RFC 2045.
`medium`	A value from `val_medium_types()`. If "OTHER" is selected then a `@medium_phrase` must be given. This should describe the original medium from which it was derived. So if it is a digital image scanned from a physical photograph, it should be "PHOTO" instead of "ELECTRONIC".
`medium_phrase`	A free text description of the medium. This is mandatory if `@medium` is "OTHER".
`media_alt`	A named vector of the media in alternative media forms, c(form = location)

Value

An S7 object representing a GEDCOM multimedia file substructure.

`MediaLink` *Create a multimedia link object*

Description

Create a multimedia link object

Usage

```
MediaLink(
  media_xref = "@VOID@",
  title = character(0),
  top = integer(0),
  left = integer(0),
  height = integer(0),
  width = integer(0)
)
```

Arguments

`media_xref`	The cross-reference identifier of a multimedia record.
`title`	The title of the multimedia record.
`top`	The number of pixels to omit from the top side of the image.
`left`	The number of pixels to omit from the left side of the image.
`height`	The height in pixels of the cropped region.
`width`	The width in pixels of the cropped region.

Details

The properties @left and @top indicate the top left corner of the region to display. The properties @width and @height indicate the dimensions of the region to display.

If the multimedia record contains multiple files, then the crop parameters only applies to the first file.

Value

An S7 object representing a GEDCOM MULTIMEDIA_LINK.

MediaRecord *Create a multimedia record object*

Description

Create a multimedia record object

Usage

```
MediaRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  created = NULL,
  updated = NULL,
  files = list()
)
```

Arguments

XREF	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
confidential	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
locked	A logical value indicating whether the associated record/fact should be treated as read-only.
private	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.

`user_ids`	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
`unique_ids`	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
`ext_ids`	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`citations`	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.
`media_links`	Not used.
`created`	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
`updated`	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
`files`	A `MediaFile` object or a list of them. This refers to 1 or more external digital files. Grouped files should each pertain to the same context.

Value

An S7 object representing a GEDCOM MULTIMEDIA_RECORD.

`new_gedcom` *Create a new GEDCOM object*

Description

Create a new GEDCOM object

Usage

```
new_gedcom(my_language = "en")
```

Arguments

`my_language`	The primary language in which data will be stored. The language code should adhere to BCP 47.

Details

See documentation for `GedcomS7()` and `GedcomSource()` for object properties.

Value

A minimal gedcom S7 object.

NonEvent — *Create a non-event object*

Description

Create a non-event object

Usage

```
NonEvent(
  event_type = character(0),
  date_period = character(0),
  date_phrase = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list()
)
```

Arguments

event_type	A code indicating the type of event that didn't happen. This must be taken from `val_event_types()`.
date_period	A date period given either as a formatted GEDCOM string, or a `DatePeriod()` object.
date_phrase	Textual information that cannot be expressed in the date.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
citations	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

Value

An S7 object representing a GEDCOM NON_EVENT_STRUCTURE.

Note — *Create a note structure object*

Description

Create a note structure object

Usage

```
Note(
  text = character(0),
  language = character(0),
  media_type = character(0),
  translations = list(),
  citations = list()
)
```

Arguments

`text`	A character string. New lines are created with \n.
`language`	A character string of language tags as defined in BCP 47.
`media_type`	The media type as defined in RFC 2045.
`translations`	A `TranslationText()` object or a list of them. One for each alternate translation of the text.
`citations`	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

Details

The shared note (SNOTE) alternative of this structure is defined separately in relevant structures.

Value

An S7 object representing a GEDCOM NOTE_STRUCTURE.

`NoteRecord` *Create a shared note record object*

Description

Create a shared note record object

Usage

```
NoteRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
```

```
  citations = list(),
  media_links = list(),
  created = NULL,
  updated = NULL,
  text = character(0),
  media_type = character(0),
  language = character(0),
  translations = list()
)
```

Arguments

XREF	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
confidential	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
locked	A logical value indicating whether the associated record/fact should be treated as read-only.
private	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
user_ids	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
unique_ids	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
ext_ids	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
note_xrefs	Not used.
notes	Not used.
citations	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.
media_links	Not used.
created	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
updated	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
text	A character string. New lines are created with \n.
media_type	The media type as defined in RFC 2045.
language	A character string of language tags as defined in BCP 47.
translations	A `TranslationText()` object or a list of them. One for each alternate translation of the text.

Value

An S7 object representing a GEDCOM SHARED_NOTE_RECORD.

`Ordinance` *Create an individual ordinance object*

Description

Create an individual ordinance object

Usage

```
Ordinance(
  ord_type = character(0),
  date = character(0),
  temple_name = character(0),
  place = NULL,
  ord_state = character(0),
  state_date = character(0),
  state_time = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  fam_xref = character(0)
)
```

Arguments

`ord_type`	A value from `val_individual_ordinance_types()`.
`date`	The date given either as a formatted GEDCOM string, or a `DateValue()` object.
`temple_name`	The name of a temple of The Church of Jesus Christ of Latter-day Saints.
`place`	The associated place. This can either be a `Place()` object or a character string (a comma-separated string of region names, ordered from smallest to largest).
`ord_state`	An optional value from `val_ordinance_states(@ord_type)`.
`state_date`	The ordinance date given either as a formatted GEDCOM string, or a `DateExact` object.
`state_time`	The ordinance time given either as a formatted GEDCOM string, or a `Time` object.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`citations`	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.
`fam_xref`	The cross-reference identifier of a family record.

Value

An S7 object representing a GEDCOM LDS_INDIVIDUAL_ORDINANCE.

parse_gedcom_age — *Convert a GEDCOM age at event into decimalised years*

Description

Convert a GEDCOM age at event into decimalised years

Usage

```
parse_gedcom_age(age_string)
```

Arguments

age_string	A string describing an age at an event, e.g. "14y 3m 1w 2d".

Value

A numeric value giving the age in years.

parse_gedcom_date — *Convert a GEDCOM date into a date object*

Description

Convert a GEDCOM date into a date object

Usage

```
parse_gedcom_date(date_string, minimise = TRUE)
```

Arguments

date_string	A GEDCOM date value string.
minimise	Whether to fill in missing date pieces so that the date is minimised. For example, if no month is given, January is used. If minimise = FALSE, December will be used.

Value

A date.

PersonalName *Create a personal name object*

Description

Create a personal name object

Usage

```
PersonalName(
  pers_name = character(0),
  name_type = character(0),
  type_phrase = character(0),
  name_pieces = NULL,
  name_translations = list(),
  notes = list(),
  note_xrefs = character(0),
  citations = list()
)
```

Arguments

pers_name	The full name of the individual. Surnames should be enclosed in forward slashes.
name_type	An optional name type taken from val_name_types().
type_phrase	An optional free text description of the name type. This is required if the name type is "OTHER".
name_pieces	A PersonalNamePieces() object defining the pieces of the full name.
name_translations	A PersonalNameTran() object or a list of them, providing different translations of this personal name.
notes	Associated notes. This can either be a Note() object, a list of them, or a character vector of notes.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
citations	Associated sources. This can either be a SourceCitation() object, a list of them, or a character vector of XREFs of source records.

Value

An S7 object representing a GEDCOM PERSONAL_NAME_STRUCTURE.

PersonalNamePieces *Create a name pieces object*

Description

Create a name pieces object

Usage

```
PersonalNamePieces(
  prefix = character(0),
  given = character(0),
  nickname = character(0),
  surname_prefix = character(0),
  surname = character(0),
  suffix = character(0)
)
```

Arguments

prefix	The name prefix, e.g. Cmdr.
given	The given name or earned name.
nickname	A descriptive or familiar name that is used instead of, or in addition to, one's proper name.
surname_prefix	Surname prefix or article used in a family name. For example in the name "de la Cruz", this value would be "de la".
surname	Surname or family name.
suffix	Name piece that appears after the given name and surname parts, e.g. Jr.

Value

An S7 object representing a GEDCOM PERSONAL_NAME_PIECES.

PersonalNameTran *Create a name translation object*

Description

Create a name translation object

Usage

```
PersonalNameTran(
  pers_name = character(0),
  language = character(0),
  name_pieces = NULL
)
```

Arguments

`pers_name`	The full name of the individual. Surnames should be enclosed in forward slashes.
`language`	A character string of language tags as defined in BCP 47.
`name_pieces`	A `PersonalNamePieces()` object defining the pieces of the full name.

Value

An S7 object representing a GEDCOM personal name translation substructure.

`Place` *Create a place structure object*

Description

Create a place structure object

Usage

```
Place(
  place_name = character(0),
  place_form = character(0),
  language = character(0),
  place_translations = character(0),
  lat_long = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list()
)
```

Arguments

`place_name`	A comma-separated string of region names, ordered from smallest to largest. The specific meaning of each element is given by the @place_form, or in the `@default_place_form` of the gedcom object if there is no @place_form defined. Elements should be left blank if they are unknown, do not apply to the location, or are too specific for the region in question. For example "Baltimore, , Maryland, USA".

`place_form` A comma-separated string of jurisdictional titles, which has the same number of elements as @place_form. For example "City, County, State, Country".

`language` A character string of language tags as defined in BCP 47.

`place_translations` A named character vector of translations of the place name. The vector values must follow the same form as the @place_name and the vector names must be a language value as defined by @language.

`lat_long` The latitude and longitude of the place, separated by a space. The latitude coordinate is the direction North or South from the equator in degrees and fraction of degrees. The longitude coordinate is in degrees and fraction of degrees East or West of the zero or base meridian coordinate. For example: 18 degrees, 9 minutes, and 3.4 seconds North, 168 degrees, 9 minutes, and 3.4 seconds East would be formatted as "N18.150944 E168.150944".

`ext_ids` A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.

`note_xrefs` A character vector of relevant note record cross-reference identifiers.

`notes` Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.

Value

An S7 object representing a GEDCOM PLACE_STRUCTURE.

`pull_record` *Pull a record from a GEDCOM object for editing*

Description

Pull a record from a GEDCOM object for editing

Usage

```
pull_record(x, xref)
```

Arguments

`x` A gedcom object.

`xref` The xref of the record to pull.

Details

The record is not removed from the gedcom object, rather a copy is taken.

Value

An S7 object representing the record.

push_record — *Push an edited record back into a GEDCOM object*

Description

Push an edited record back into a GEDCOM object

Usage

```
push_record(x, record)
```

Arguments

x	A gedcom object.
record	An object representing the record to place back into the GEDCOM object.

Details

The function will automatically keep family links for individuals updated. It will also update the `@updated` property if `@update_change_dates` in the gedcom object is set to TRUE.

Value

An updated GEDCOM object.

read_gedcom — *Import a GEDCOM file*

Description

Imports a *.ged file and creates a gedcom object.

Usage

```
read_gedcom(filepath = NULL, lines = NULL)
```

Arguments

filepath	The full filepath of the GEDCOM file. If this and the `lines` parameter are both NULL then you will be prompted to select a file. You may also provide the path to a GEDzip file (*.zip), where there should be a single GEDCOM file located in the root directory of the archive.
lines	If the filepath is not provided then a character vector of GEDCOM lines can be provided.

Value

A gedcom S7 object.

RepositoryCitation *Create a repository citation object*

Description

Create a repository citation object

Usage

```
RepositoryCitation(
  repo_xref = "@VOID@",
  notes = list(),
  note_xrefs = character(0),
  call_numbers = list()
)
```

Arguments

repo_xref	The cross-reference identifier of a repository record. If the repository does not have a record, then this can be left blank and a void xref will be used. However, you should describe the repository in @notes.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
call_numbers	Call number(s) used to file and retrieve items from the repository. This can either be a `SourceCallNumber` object, a list of them, or a character vector of call numbers.

Value

An S7 object representing a GEDCOM SOURCE_REPOSITORY_CITATION.

RepositoryRecord *Create a repository record object*

Description

Create a repository record object

Usage

```
RepositoryRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
  media_links = list(),
  created = NULL,
  updated = NULL,
  repo_name = character(0),
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0)
)
```

Arguments

`XREF`	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
`confidential`	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
`locked`	A logical value indicating whether the associated record/fact should be treated as read-only.
`private`	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
`user_ids`	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
`unique_ids`	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
`ext_ids`	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`citations`	Not used.
`media_links`	Not used.

created	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
updated	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
repo_name	The name of the repository.
address	The address given either as a `Address()` object or as a character string. This would be as written on a mailing label with new lines separated by \n.
phone_numbers	A character vector of phone numbers.
emails	A character vector of email addresses.
faxes	A character vector of fax numbers.
web_pages	A character vector of web page URLs.

Value

An S7 object representing a GEDCOM REPOSITORY_RECORD.

`rm_living` *Remove living individuals in a GEDCOM object*

Description

Remove living individuals in a GEDCOM object

Usage

```
rm_living(x, max_age = 100)
```

Arguments

x	A gedcom object.
max_age	The maximum age to assume for a living person (if a date of birth is given).

Value

A gedcom object cleansed of information on living individuals.

rm_records — *Remove records from a GEDCOM object*

Description

Remove records from a GEDCOM object

Usage

```
rm_records(x, xrefs, void_refs = TRUE)
```

Arguments

x	A gedcom object.
xrefs	A character vector of xrefs to remove.
void_refs	Whether to replace references to this record with a @VOID@ reference. This indicates to people that there was a reference to a record here. Note that if this is set to FALSE, you risk losing supplementary information (e.g. pedigree data in family links).

Value

The gedcom object with the records removed.

SourceCallNumber — *Create a source call number object*

Description

Create a source call number object

Usage

```
SourceCallNumber(
  call_number = character(0),
  medium = character(0),
  medium_phrase = character(0)
)
```

Arguments

call_number	The call number.
medium	A value from `val_medium_types()`. If "OTHER" is selected then a `@medium_phrase` must be given.
medium_phrase	A free text description of the medium. This is mandatory if `@medium` is "OTHER".

Value

An S7 object representing the CALN substructure of a GEDCOM SOURCE_REPOSITORY_CITATION.

SourceCitation *Create a source citation object*

Description

Create a source citation object

Usage

```
SourceCitation(
  sour_xref = "@VOID@",
  where = character(0),
  date = character(0),
  source_text = list(),
  fact_type = character(0),
  fact_phrase = character(0),
  role = character(0),
  role_phrase = character(0),
  certainty = character(0),
  media_links = list(),
  note_xrefs = character(0),
  notes = list()
)
```

Arguments

`sour_xref`	The cross-reference identifier of a source record. If the source does not have a record, then this can be left blank and a void xref will be used. However, you should describe the source in @where.
`where`	A specific location within the information referenced. For a published work, this could include the volume of a multi-volume work and the page number or numbers. For a periodical, it could include volume, issue, and page numbers. For a newspaper, it could include a date, page number, and column number. For an unpublished source or micro- filmed works, this could be a film or sheet number, page number, or frame number. A census record might have an enumerating district, page number, line number, dwelling number, and family number. It is recommended that the data in this field be formatted comma-separated with label: value pairs
`date`	The date given either as a formatted GEDCOM string, or a `DateValue()` object.
`source_text`	A verbatim copy of any description contained within the source. This can either be a `TranslationText()` object, a list of them, or a character vector of text.

fact_type	A code indicating the type of fact. This must be taken from one of val_individual_event_types(), val_individual_attribute_types(), val_family_event_types(), or val_family_attribute_typ A generic event ("EVEN") or attribute ("FACT") can also be defined for more bespoke facts. See fact_rules_df() for the set of rules surrounding the codes allowed.
fact_phrase	Textual information that cannot be expressed in the fact type.
role	What role this person played in this fact.
role_phrase	Textual information that cannot be expressed in the role.
certainty	An enumerated value indicating the credibility of a piece of information, based on its supporting evidence. Some systems use this feature to rank multiple conflicting opinions for display of most likely information first. It is not intended to eliminate the receivers' need to evaluate the evidence for themselves. "0" = unreliable/estimated data "1" = Questionable reliability of evidence "2" = Secondary evidence, data officially recorded sometime after event "3" = Direct and primary evidence used, or by dominance of the evidence
media_links	Associated multimedia. This can either be a MediaLink() object, a list of them, or a character vector of XREFs of multimedia records.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a Note() object, a list of them, or a character vector of notes.

Value

An S7 object representing a GEDCOM SOURCE_CITATION.

SourceRecord *Create a source record object*

Description

Create a source record object

Usage

```
SourceRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
```

```
  media_links = list(),
  created = NULL,
  updated = NULL,
  facts_recorded = list(),
  agency = character(0),
  data_note_xrefs = character(0),
  data_notes = list(),
  originator = character(0),
  full_title = character(0),
  short_title = character(0),
  publication_facts = character(0),
  source_text = list(),
  repo_citations = list()
)
```

Arguments

`XREF`	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
`confidential`	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
`locked`	A logical value indicating whether the associated record/fact should be treated as read-only.
`private`	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
`user_ids`	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
`unique_ids`	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
`ext_ids`	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
`note_xrefs`	A character vector of relevant note record cross-reference identifiers.
`notes`	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
`citations`	Not used.
`media_links`	Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.
`created`	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
`updated`	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
`facts_recorded`	The facts recorded by the source. This can either be a `FactsRecorded` object, a list of them, or a character vector of comma-delimited fact types. For example, a parish register of births, deaths, and marriages would be "BIRT, DEAT, MARR". The `val_fact_types()` function gives a list of possible fact types.

agency
: The organization, institution, corporation, person, or other entity that has responsibility for the associated fact. Examples are an employer of a person of an associated occupation, or an educational establishment that has awarded a scholastic award.

data_note_xrefs
: A character vector of note record cross-reference identifiers relevant to the source data.

data_notes
: Associated notes about the source data. This can either be a `Note` object, a list of them, or a character vector of notes.

originator
: The person, agency, or entity who created the record. For a published work, this could be the author, compiler, transcriber, abstractor, or editor. For an unpublished source, this may be an individual, a government agency, church organization, or private organization.

full_title
: The full title of the source.

short_title
: A shortened name of the source used for sorting, filing, and retrieving records.

publication_facts
: When and where the record was created. For published works, this includes information such as the city of publication, name of the publisher, and year of publication.

source_text
: A verbatim copy of any description contained within the source. This can either be a `TranslationText()` object, a list of them, or a character vector of text.

repo_citations
: Associated repositories. This can either be a `RepositoryCitation` object, a list of them, or a character vector of XREFs of repository records.

Value

An S7 object representing a GEDCOM SOURCE_RECORD.

`SpouseSealing` *Create a spouse sealing object*

Description

Create a spouse sealing object

Usage

```
SpouseSealing(
  date = character(0),
  temple_name = character(0),
  place = NULL,
  ord_state = character(0),
  state_date = character(0),
  state_time = character(0),
  note_xrefs = character(0),
```

```
  notes = list(),
  citations = list()
)
```

Arguments

date	The date given either as a formatted GEDCOM string, or a `DateValue()` object.
temple_name	The name of a temple of The Church of Jesus Christ of Latter-day Saints.
place	The associated place. This can either be a `Place()` object or a character string (a comma-separated string of region names, ordered from smallest to largest).
ord_state	An optional value from `val_ordinance_states(@ord_type)`.
state_date	The ordinance date given either as a formatted GEDCOM string, or a `DateExact` object.
state_time	The ordinance time given either as a formatted GEDCOM string, or a `Time` object.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
citations	Associated sources. This can either be a `SourceCitation()` object, a list of them, or a character vector of XREFs of source records.

Value

An S7 object representing a GEDCOM LDS_SPOUSE_SEALING.

SubmitterRecord *Create a submitter record object*

Description

Create a submitter record object

Usage

```
SubmitterRecord(
  XREF = "@GEDCOMS7_ORPHAN@",
  confidential = FALSE,
  locked = FALSE,
  private = FALSE,
  user_ids = character(0),
  unique_ids = character(0),
  ext_ids = character(0),
  note_xrefs = character(0),
  notes = list(),
  citations = list(),
```

```
  media_links = list(),
  created = NULL,
  updated = NULL,
  subm_name = character(0),
  address = NULL,
  phone_numbers = character(0),
  emails = character(0),
  faxes = character(0),
  web_pages = character(0),
  languages = character(0)
)
```

Arguments

XREF	The cross-reference identifier for this record. You should not edit this at all as maintenance of these is done automatically.
confidential	A logical value indicating whether the associated record/fact should be treated as confidential. This allows them to be excluded on export.
locked	A logical value indicating whether the associated record/fact should be treated as read-only.
private	A logical value indicating whether the associated record/fact should be treated as private. This allows them to be excluded on export.
user_ids	A character vector of user-generated identifiers. The type of the identifiers can be given in the vector names, e.g. c("Driving license number" = "ABC123")
unique_ids	A character vector of enduring and globally-unique identifiers. These need to be formatted in line with RFC 9562 and can be generated with `uuid::UUIDgenerate()`.
ext_ids	A named character vector of identifiers maintained by an external authority. The names must be given as a URI. See the GEDCOM specification for more information.
note_xrefs	A character vector of relevant note record cross-reference identifiers.
notes	Associated notes. This can either be a `Note()` object, a list of them, or a character vector of notes.
citations	Not used.
media_links	Associated multimedia. This can either be a `MediaLink()` object, a list of them, or a character vector of XREFs of multimedia records.
created	A `CreationDate()` object containing the date the record was created. Creating an object with no parameters sets the date to today.
updated	A `ChangeDate()` object containing the date the record was updated. Creating an object with no parameters sets the date to today.
subm_name	The name of the submitter.
address	The address given either as a `Address()` object or as a character string. This would be as written on a mailing label with new lines separated by \n.
phone_numbers	A character vector of phone numbers.
emails	A character vector of email addresses.

faxes	A character vector of fax numbers.
web_pages	A character vector of web page URLs.
languages	A character vector of language tags as defined in BCP 47.

Value

An S7 object representing a GEDCOM SUBMITTER_RECORD.

Time — *Create a time object*

Description

Create a time object

Usage

```
Time(
  hour = integer(0),
  minute = integer(0),
  second = integer(0),
  fraction = integer(0),
  utc = TRUE
)
```

Arguments

hour	The hour of the day given as an integer between 0 and 23.
minute	The minute of the hour given as an integer between 0 and 59.
second	The second of the minute given as an integer between 0 and 59.
fraction	The fraction of the second given as an integer.
utc	Whether the time is in Coordinated Universal Time (UTC) (TRUE, the default) or is in local time (FALSE).

Value

An S7 object representing a GEDCOM time.

TranslationText *Create a text translation object*

Description

Create a text translation object

Usage

```
TranslationText(
  text = character(0),
  language = character(0),
  media_type = character(0)
)
```

Arguments

text	A character string. New lines are created with \n.
language	A character string of language tags as defined in BCP 47.
media_type	The media type as defined in RFC 2045.

Value

An S7 object representing a GEDCOM text translation substructure.

val_record_types *Lookup values*

Description

Lookup values

Usage

```
val_record_types()

val_adoptive_parents()

val_individual_attribute_types(inc_generic = FALSE)

val_individual_event_types(inc_generic = FALSE)

val_family_event_types(inc_generic = FALSE)

val_family_attribute_types(inc_generic = FALSE)
```

```
val_event_types(inc_generic = FALSE)

val_attribute_types(inc_generic = FALSE)

val_fact_types(inc_generic = FALSE)

val_individual_ordinance_types()

val_family_ordinance_types()

val_ordinance_states(ord_type)

val_medium_types()

val_pedigree_types()

val_certainty()

val_restriction()

val_roles()

val_sexes()

val_confidence_types()

val_name_types()
```

Arguments

`inc_generic`	Whether to include a generic facts.
`ord_type`	One of the values in `val_individual_ordinance_types()` or `val_family_ordinance_types()`.

Value

A vector of allowed values.

`write_gedcom` *Save a gedcom object to disk as a GEDCOM file*

Description

Save a gedcom object to disk as a GEDCOM file

Usage

```
write_gedcom(
  gedcom,
  filepath = file.choose(),
  inc_confid = TRUE,
  inc_private = TRUE,
  inc_living = TRUE
)
```

Arguments

gedcom	A gedcom object.
filepath	The full filepath to write to.
inc_confid	Whether to include records that are marked as confidential.
inc_private	Whether to include records that are marked as private.
inc_living	Whether to include individual records for suspected living people.

Details

This function prepares the gedcom object and then writes it to the filepath. Steps taken include filtering sensitive data, escaping "@" signs (with another "@") , and splitting long lines onto separate lines.

Writing to GEDzip files is not supported. If you want to do this then write to a GEDCOM file first and then move the file to the GEDzip archive.

Value

The filepath (invisibly).

Index

add_children, 3
add_parents, 4
add_siblings, 5
add_spouse, 6
Address, 6
Association, 7

ChangeDate, 8
CreationDate, 9

date_exact_current, 9
DateApprox, 10
DateCalendar, 10
DateExact, 11
DatePeriod, 11
DateRange, 12
DateSorting, 12
DateValue, 13
df_fam, 13
df_fam_facts, 14
df_indi, 14
df_indi_facts, 15
df_media, 15
df_note, 16
df_repo, 16
df_sour, 17
df_subm, 17

fact_rules_df, 18
FactsRecorded, 18
FamilyAttribute, 19
FamilyEvent, 21
FamilyLinkChild, 24
FamilyLinkSpouse, 25
FamilyRecord, 25

GedcomHeader, 27
GedcomS7, 28
GedcomSource, 29
get_ancestors, 30
get_descendants, 31
get_fam_as_child, 32
get_fam_as_spouse, 33
get_fam_children, 33
get_fam_partners, 34
get_indi_children, 34
get_indi_cousins, 35
get_indi_fathers, 35
get_indi_mothers, 36
get_indi_parents, 36
get_indi_partners, 37
get_indi_siblings, 37
get_records_by_pattern, 38
get_supporting_recs, 38
get_unused_recs, 39

IndividualAttribute, 40
IndividualEvent, 42
IndividualRecord, 44

MediaFile, 46
MediaLink, 47
MediaRecord, 48

new_gedcom, 49
NonEvent, 50
Note, 50
NoteRecord, 51

Ordinance, 53

parse_gedcom_age, 54
parse_gedcom_date, 54
PersonalName, 55
PersonalNamePieces, 56
PersonalNameTran, 56
Place, 57
pull_record, 58
push_record, 59

read_gedcom, 59

RepositoryCitation, 60
RepositoryRecord, 60
rm_living, 62
rm_records, 63

SourceCallNumber, 63
SourceCitation, 64
SourceRecord, 65
SpouseSealing, 67
SubmitterRecord, 68

Time, 70
TranslationText, 71

val_adoptive_parents
(val_record_types), 71
val_attribute_types (val_record_types),
71
val_certainty (val_record_types), 71
val_confidence_types
(val_record_types), 71
val_event_types (val_record_types), 71
val_fact_types (val_record_types), 71
val_family_attribute_types
(val_record_types), 71
val_family_event_types
(val_record_types), 71
val_family_ordinance_types
(val_record_types), 71
val_individual_attribute_types
(val_record_types), 71
val_individual_event_types
(val_record_types), 71
val_individual_ordinance_types
(val_record_types), 71
val_medium_types (val_record_types), 71
val_name_types (val_record_types), 71
val_ordinance_states
(val_record_types), 71
val_pedigree_types (val_record_types),
71
val_pedigree_types(), *4, 5*
val_record_types, 71
val_restriction (val_record_types), 71
val_roles (val_record_types), 71
val_sexes (val_record_types), 71
val_sexes(), *4, 5*

write_gedcom, 72