Package 'charlatan'

September 13, 2023

Type Package

Title Make Fake Data

Description Make fake data, supporting addresses, person names, dates, times, colors, coordinates, currencies, digital object identifiers ('DOIs'), jobs, phone numbers, 'DNA' sequences, doubles and integers from distributions and within a range.

Version 0.5.1

License MIT + file LICENSE

URL https://docs.ropensci.org/charlatan/ (website)
https://github.com/ropensci/charlatan (devel)

BugReports https://github.com/ropensci/charlatan/issues

LazyData true

Encoding UTF-8

Language en-US

VignetteBuilder knitr

Imports R6 (>= 2.2.0), tibble (>= 1.2), whisker

Suggests testthat, knitr, rmarkdown, ipaddress, stringi

RoxygenNote 7.2.2

Collate 'address-provider-en_GB.R' 'address-provider-en_NZ.R' 'address-provider-en_US.R' 'address-provider-nl_NL.R' 'base-provider.R' 'datetime-provider.R' 'address-provider.R' 'available_locales.R' 'charlatan-package.R' 'charlatan_settings.R' 'color-provider-en_US.R' 'color-provider-uk_UA.R' 'color-provider.R' 'color.R' 'company-provider-bg_BG.R' 'company-provider-cs_CZ.R' 'company-provider-de_DE.R' 'company-provider-en_US.R' 'company-provider-es_MX.R' 'company-provider-fa_IR.R' 'company-provider-fr_FR.R' 'company-provider-hr_HR.R' 'company-provider-it_IT.R' 'company-provider.R' 'company.R' 'coordinate-provider.R' 'coordinates.R' 'credit_card-provider.R' 'credit_card.R' 'currency-provider.R'

'currency.R' 'date_time.R' 'doi-provider.R' 'doi.R'
'element-provider.R' 'elements.R' 'file-provider.R'
'fraudster.R' 'generate.R' 'globals.R'
'internet-provider-bg_BG.R' 'internet-provider-cs_CZ.R'
'internet-provider-de_DE.R' 'internet-provider-en_AU.R'
'internet-provider-en_NZ.R' 'internet-provider-fa_IR.R'
'internet-provider-fr_FR.R' 'internet-provider-hr_HR.R'
'internet-provider.R' 'job.R' 'jobs-provider-da_DK.R'
'jobs-provider-en_US.R' 'jobs-provider-fa_IR.R'
'jobs-provider-fi_FI.R' 'jobs-provider-fr_CH.R'
'jobs-provider-fr_FR.R' 'jobs-provider-hr_HR.R'
'jobs-provider-nl_NL.R' 'jobs-provider-pl_PL.R'
'jobs-provider-ru_RU.R' 'jobs-provider-uk_UA.R'
'jobs-provider-zh_TW.R' 'jobs-provider.R'
'lorem-provider-ar_AA.R' 'lorem-provider-el_GR.R'
'lorem-provider-en_US.R' 'lorem-provider-he_IL.R'
'lorem-provider-ja_JP.R' 'lorem-provider-la.R'
'lorem-provider-ru_RU.R' 'lorem-provider-zh_CN.R'
'lorem-provider-zh_TW.R' 'lorem-provider.R' 'misc-provider.R'
'missing-data-provider.R' 'missing.R' 'name.R'
'numerics-provider.R' 'numerics.R' 'onload.R'
'person-provider-bg_BG.R' 'person-provider-cs_CZ.R'
'person-provider-da_DK.R' 'person-provider-de_AT.R'
'person-provider-de_DE.R' 'person-provider-en_GB.R'
'person-provider-en_NZ.R' 'person-provider-en_US.R'
'person-provider-es_ES.R' 'person-provider-es_MX.R'
'person-provider-fa_IR.R' 'person-provider-fi_FI.R'
'person-provider-fr_CH.R' 'person-provider-fr_FR.R'
'person-provider-hr_HR.R' 'person-provider-it_IT.R'
'person-provider-ja_JP.R' 'person-provider-ko_KR.R'
'person-provider-lt_LT.R' 'person-provider-lv_LV.R'
'person-provider-ne_NP.R' 'person-provider-nl_NL.R'
'person-provider-no_NO.R' 'person-provider-pl_PL.R'
'person-provider.R' 'phone-number-provider-all.R'
'phone_number.R' 'phonenumbers-provider.R'
'sequence-provider.R' 'sequences.R' 'ssn-provider.R' 'ssn.R'
'taxonomy-provider.R' 'taxonomy.R' 'useragent-provider.R'
'zzz.R'

NeedsCompilation no

Author Roel M. Hogervorst [cre, aut] (<https://orcid.org/0000-0001-7509-0328>),
Scott Chamberlain [aut] (<https://orcid.org/0000-0003-1444-9135>),
Kyle Voytovich [aut],
Martin Pedersen [ctb],
Brooke Anderson [rev] (Brooke Anderson reviewed the package for rOpenSci, see https://github.com/ropensci/onboarding/issues/94),
Tristan Mahr [rev] (Tristan Mahr reviewed the package for rOpenSci, see https://github.com/ropensci/onboarding/issues/94),
rOpenSci [fnd] (https://ropensci.org)

Maintainer Roel M. Hogervorst <hogervorst.rm@gmail.com>

Repository CRAN

Date/Publication 2023-09-13 13:40:02 UTC

R topics documented:

`charlatan-package` *charlatan*

Description

Make fake data, supporting addresses, person names, dates, times, colors, coordinates, currencies, digital object identifiers (DOIs), jobs, phone numbers, DNA sequences, doubles and integers from distributions and within a range.

Package API

- ch_generate(): generate a data.frame with fake data
- fraudster(): single interface to all fake data methods
- High level interfaces: There are high level functions prefixed with ch_ that wrap low level interfaces, and are meant to be easier to use and provide easy way to make many instances of a thing.
- Low level interfaces: All of these are R6 objects that a user can initialize and then call methods on the them.

Author(s)

Scott Chamberlain <myrmecocystus+r@gmail.com>

Kyle Voytovich

Martin Pedersen

Examples

```
# generate individual types of data
ch_name()
ch_phone_number()
ch_job()

# generate a data.frame
ch_generate()

# one interface to all data types - generate the class first
#  reports the locale to be used, can change optionally
(x <- fraudster())
x$job()
x$name()
x$color_name()
x$hex_color()

# low level interfaces to "data providers"
# these are exported by hidden from package man page
# as most users will likely not interact with these
x <- ColorProvider$new()
x$color_name()
x$hex_color()
```

---

available_locales_df *Available locales*

Description

A data.frame of locales available in **charlatan**

Format

A data frame with 45 rows and 4 variables:

Language language two letter code

Country country two letter code

Variant a variant code, if applicable

Name official locale two letter code

See Also

data.frame used in `charlatan_locales()`

`BaseProvider` *BaseProvider*

Description

BaseProvider

BaseProvider

Methods

Public methods:

- `BaseProvider$random_element()`
- `BaseProvider$random_element_prob()`
- `BaseProvider$random_int()`
- `BaseProvider$random_digit()`
- `BaseProvider$random_digit_not_zero()`
- `BaseProvider$random_digit_or_empty()`
- `BaseProvider$random_digit_not_zero_or_empty()`
- `BaseProvider$random_letter()`
- `BaseProvider$numerify()`
- `BaseProvider$lexify()`
- `BaseProvider$bothify()`
- `BaseProvider$check_locale()`
- `BaseProvider$randomize_nb_elements()`
- `BaseProvider$clone()`

Method `random_element()`**:** pick a random element from vector/list

Usage:

```
BaseProvider$random_element(x)
```

Arguments:

`x` vector or list

Returns: a single element from x

Method `random_element_prob()`: pick a random element with probability from vector/list

Usage:

```
BaseProvider$random_element_prob(x)
```

Arguments:

`x` vector or list

Method `random_int()`: any number of random integers from a min, max

Usage:

```
BaseProvider$random_int(min = 0, max = 9999, size = 1)
```

Arguments:

`min` the minimum value. default: 0

`max` the maximum value. default: 9999

`size` number of values to return. default: 1

Returns: random integer

Method `random_digit()`: random integer between 0 and 9

Usage:

```
BaseProvider$random_digit()
```

Method `random_digit_not_zero()`: random integer between 1 and 9

Usage:

```
BaseProvider$random_digit_not_zero()
```

Method `random_digit_or_empty()`: random integer between 0 and 9 or empty character string

Usage:

```
BaseProvider$random_digit_or_empty()
```

Method `random_digit_not_zero_or_empty()`: random integer between 1 and 9 or empty character string

Usage:

```
BaseProvider$random_digit_not_zero_or_empty()
```

Method `random_letter()`: random letter

Usage:

```
BaseProvider$random_letter()
```

Method `numerify()`: replace a template with numbers

Usage:

```
BaseProvider$numerify(text = "###")
```

Arguments:

`text` (character) a string

Method `lexify()`: replace a template with letters

Usage:

```
BaseProvider$lexify(text = "????")
```

Arguments:

`text` (character) a string

Method `bothify()`: both numerify and lexify together

Usage:

```
BaseProvider$bothify(text = "## ??")
```

Arguments:

`text` (character) a string

Method `check_locale()`: check a locale to see if it exists, if not, stop with error message

Usage:

```
BaseProvider$check_locale(x)
```

Arguments:

`x` a locale name, e.g, 'bg_BG'

Returns: returns nothing if locale is supported; stops w/ message if not

Method `randomize_nb_elements()`: Returns a random value near number

Usage:

```
BaseProvider$randomize_nb_elements(
  number = 10,
  le = FALSE,
  ge = FALSE,
  min = NULL,
  max = NULL
)
```

Arguments:

`number` value to which the result must be near

`le` result must be lower or equal to number

`ge` result must be greater or equal to number

`min` the minimum value. default: `NULL`

`max` the maximum value. default: `NULL`

Returns: a random int near number

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
BaseProvider$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
(x <- BaseProvider$new())

x$numerify("#%%asdf221?")
x$lexify("#%%asdf221?")
x$bothify("#%%asdf221?")

z <- PhoneNumberProvider$new()
x$numerify(z$render())

x$random_element(letters)
x$random_int()
x$random_digit()
x$random_digit_not_zero()
x$random_digit_or_empty()
x$random_digit_not_zero_or_empty()
x$random_letter()
x$check_locale("es_ES")
## fails
# x$check_locale("es_EQ")

x$randomize_nb_elements()
```

---

charlatan_locales *Available locales*

Description

Available locales

Usage

```
charlatan_locales()
```

Value

a data.frame of the available locales in this package. See available_locales_df for structure.

Not all functions support all locales. Check the docs for each one to see what locales they support.

You can find out more about each locale by running your locale though `stringi::stri_locale_info()`

Examples

```
charlatan_locales()
```

charlatan_settings *charlatan settings*

Description

charlatan settings

Usage

```
charlatan_settings(messy = NULL)
```

Arguments

messy	(logical) make some messy data. Default: NULL

More deets

- messy - When FALSE, nothing is different from normal. When TRUE, we select incorrect/wrong values with probability X. Messy mode is only available for **en-US** for now, and only for some data types. The default setting is NULL, meaning it is ignored.

Examples

```
charlatan_settings()
charlatan_settings(messy = TRUE)
charlatan_settings(messy = FALSE)

# with PersonProvider - overrides local messy param in all cases
x <- PersonProvider$new()
x$messy
charlatan_settings(messy = TRUE)
x <- PersonProvider$new()
x$messy
```

ch_color *Create fake colors*

Description

Create fake colors

Usage

```
ch_color_name(n = 1, locale = NULL)

ch_safe_color_name(n = 1, locale = NULL)

ch_hex_color(n = 1)

ch_safe_hex_color(n = 1)

ch_rgb_color(n = 1)

ch_rgb_css_color(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. See `ColorProvider$new()$allowed_locales()` for locales supported. Affects the `ch_color_name` and `ch_safe_color_name` functions

See Also

ColorProvider

Examples

```
ch_color_name()
ch_color_name(10)
ch_color_name(500)

ch_safe_color_name()
ch_safe_color_name(10)

ch_hex_color()
ch_hex_color(10)
ch_hex_color(1000)

ch_safe_hex_color()
ch_safe_hex_color(10)

ch_rgb_color()
ch_rgb_color(10)

ch_rgb_css_color()
ch_rgb_css_color(10)

ch_color_name(locale = "uk_UA")
ch_color_name(n = 10, locale = "uk_UA")

ch_safe_color_name(locale = "uk_UA")
ch_safe_color_name(n = 10, locale = "uk_UA")
```

ch_company *Create fake company names and other company bits*

Description

Create fake company names and other company bits

Usage

```
ch_company(n = 1, locale = NULL)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. See `CompanyProvider$new()$allowed_locales()` for locales supported.

See Also

CompanyProvider

Examples

```
ch_company()
ch_company(10)
ch_company(500)

ch_company(locale = "fr_FR", n = 10)
ch_company(locale = "cs_CZ", n = 10)
ch_company(locale = "es_MX", n = 10)
ch_company(locale = "hr_HR", n = 10)
```

ch_credit *Create fake credit card data*

Description

Create fake credit card data

Usage

```
ch_credit_card_provider(n = 1)

ch_credit_card_number(n = 1)

ch_credit_card_security_code(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer

See Also

CreditCardProvider

Examples

```
ch_credit_card_provider()
ch_credit_card_provider(n = 4)

ch_credit_card_number()
ch_credit_card_number(n = 10)
ch_credit_card_number(n = 500)

ch_credit_card_security_code()
ch_credit_card_security_code(n = 10)
ch_credit_card_security_code(n = 500)
```

---

ch_currency *Create fake currencies*

Description

Create fake currencies

Usage

```
ch_currency(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer

See Also

CurrencyProvider

Examples

```
ch_currency()
ch_currency(10)
ch_currency(500)
```

ch_doi *Create fake DOIs (Digital Object Identifiers)*

Description

Create fake DOIs (Digital Object Identifiers)

Usage

```
ch_doi(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer

See Also

DOIProvider

Examples

```
ch_doi()
ch_doi(10)
ch_doi(100)
```

ch_generate *Generate a fake dataset*

Description

Generate a fake dataset

Usage

```
ch_generate(..., n = 10, locale = NULL)
```

Arguments

...	columns to include. must be in the allowed set. See **Allowed column names** below. Three default columns are included (name, job, phone_number) if nothing is specified - but are overridden by any input.
n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. options: only supported for data types that have locale support, See each data provider for details.

Allowed column names

- name (default included)
- job (default included)
- phone_number (default included)
- currency
- color_name
- rgb_color
- rgb_css_color

Examples

```
ch_generate()
ch_generate(n = 1)
ch_generate(n = 100)

ch_generate("job")
ch_generate("job", "name")
ch_generate("job", "color_name")

# locale
ch_generate(locale = "en_US")
ch_generate(locale = "fr_FR")
ch_generate(locale = "fr_CH")
```

---

ch_gene_sequence *Create fake gene sequences*

Description

Create fake gene sequences

Usage

```
ch_gene_sequence(n = 1, length = 30)
```

Arguments

n	(integer) number of things to get, any non-negative integer
length	(integer) length of sequence to create

See Also

SequenceProvider

Examples

```
ch_gene_sequence()
ch_gene_sequence(10)
ch_gene_sequence(100)

ch_gene_sequence(length = 500)
ch_gene_sequence(10, length = 500)
```

---

ch_job *Create fake jobs*

Description

Create fake jobs

Usage

```
ch_job(n = 1, locale = NULL)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. Run `JobProvider$new()$allowed_locales()` for locales supported (default: en_US)

See Also

JobProvider

Examples

```
ch_job()
ch_job(10)
ch_job(500)

ch_job(locale = "da_DK", n = 10)
ch_job(locale = "fi_FI", n = 10)
ch_job(locale = "fr_FR", n = 10)
ch_job(locale = "fr_CH", n = 10)
ch_job(locale = "hr_HR", n = 10)
ch_job(locale = "fa_IR", n = 10)
ch_job(locale = "pl_PL", n = 10)
ch_job(locale = "ru_RU", n = 10)
ch_job(locale = "uk_UA", n = 10)
ch_job(locale = "zh_TW", n = 10)
```

ch_missing *Create missing data*

Description

Create missing data

Usage

```
ch_missing(x, n = 1)
```

Arguments

x	Input vector, can be any class - only 1 vetor
n	(integer) number of things to get, any non-negative integer

See Also

MissingDataProvider

Examples

```
ch_missing(letters)
ch_missing(letters, 10)
ch_missing(letters, 20)
```

ch_name *Create fake person names*

Description

Create fake person names

Usage

```
ch_name(n = 1, locale = NULL, messy = FALSE)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. See `PersonProvider$new()$allowed_locales()` for locales supported (default: en_US)
messy	(logical) make some messy data. Default: `FALSE`

See Also

PersonProvider

Examples

```
ch_name()
ch_name(10)
ch_name(500)

ch_name(locale = "fr_FR", n = 10)
ch_name(locale = "fr_CH", n = 10)
ch_name(locale = "fa_IR", n = 10)
ch_name(locale = "fi_FI", n = 10)
```

---

ch_phone_number *Create fake phone numbers*

Description

Create fake phone numbers

Usage

```
ch_phone_number(n = 1, locale = NULL)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. See `PhoneNumberProvider$new()$allowed_locales()` for locales supported (default: en_US)

See Also

PhoneNumberProvider

Examples

```
ch_phone_number()
ch_phone_number(10)
ch_phone_number(500)

# locales
ch_phone_number(locale = "fr_FR")
ch_phone_number(locale = "uk_UA")
ch_phone_number(locale = "en_CA")
ch_phone_number(locale = "lv_LV")
```

ch_ssn *Create fake Social Security Numbers*

Description

Create fake Social Security Numbers

Usage

```
ch_ssn(n = 1, locale = NULL)
```

Arguments

n	(integer) number of things to get, any non-negative integer
locale	(character) the locale to use. See `SSNProvider$new()$allowed_locales()` for locales supported (default: en_US)

See Also

SSNProvider

Examples

```
ch_ssn()
ch_ssn(10)
```

coordinates *Create fake coordinates*

Description

Create fake coordinates

Usage

```
ch_lon(n = 1)

ch_lat(n = 1)

ch_position(n = 1, bbox = NULL)
```

Arguments

n	(integer) number of things to get, any non-negative integer
bbox	a bounding box of the form `[w,s,e,n]`

See Also

CoordinateProvider

Examples

```
ch_lon()
ch_lon(10)

ch_lat()
ch_lat(10)

ch_position()
ch_position(10)
ch_position(bbox = c(-120, 30, -110, 60))
```

`date_time` *Create dates and times*

Description

Create dates and times

Usage

```
ch_timezone(n = 1)

ch_unix_time(n = 1)

ch_date_time(n = 1)
```

Arguments

`n`	(integer) number of things to get, any non-negative integer

See Also

DateTimeProvider

Examples

```
ch_timezone()
ch_timezone(10)

ch_unix_time()
ch_unix_time(20)

ch_date_time()
ch_date_time(20)
```

elements *Get elements*

Description

Get elements

Usage

```
ch_element_symbol(n = 1)

ch_element_element(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer

See Also

ElementProvider

Examples

```
ch_element_symbol()
ch_element_symbol(10)
ch_element_symbol(50)

ch_element_element()
ch_element_element(10)
ch_element_element(50)
```

fraudster *Fraudster - catch all client to make all types of fake data*

Description

Fraudster - catch all client to make all types of fake data

Usage

```
fraudster(locale = NULL)
```

Arguments

locale	(character) the locale to use. options: en_US (default), fr_FR, fr_CH, hr_FR, fa_IR, pl_PL, ru_RU, uk_UA, zh_TW.

Examples

```
# English - the default locale
(x <- fraudster())
x$job()
x$name()
x$color_name()
x$safe_color_name()
x$hex_color()
x$safe_hex_color()
x$rgb_color()
x$rgb_css_color()

# different locales
## French
(y <- fraudster(locale = "fr_FR"))
y$job()

## Croatian
(z <- fraudster(locale = "hr_HR"))
z$job()

## Ukranian
(w <- fraudster(locale = "uk_UA"))
w$job()
w$color_name()

# geospatial
x$lat()
x$lon()
x$position()

# DOIs (Digital Object Identifier)
x$doi()
```

`numerics` *Create numbers*

Description

Create numbers

Usage

```
ch_double(n = 1, mean = 0, sd = 1)

ch_integer(n = 1, min = 1, max = 1000)

ch_unif(n = 1, min = 0, max = 9999)
```

```
ch_norm(n = 1, mean = 0, sd = 1)

ch_lnorm(n = 1, mean = 0, sd = 1)

ch_beta(n = 1, shape1, shape2, ncp = 0)
```

Arguments

n	(integer) number of things to get, any non-negative integer
mean	mean value
sd	standard deviation
min	minimum value
max	maximum value
shape1, shape2	non-negative parameters of the Beta distribution
ncp	non-centrality parameter

Examples

```
ch_double()
ch_double(10)
ch_double(100)

ch_integer()
ch_integer(10)
ch_integer(100)

ch_unif()
ch_norm()
ch_lnorm()
ch_beta(shape1 = 1, shape2 = 1)
```

---

taxonomy *Create fake taxonomic names*

Description

Create fake taxonomic names

Usage

```
ch_taxonomic_genus(n = 1)

ch_taxonomic_epithet(n = 1)

ch_taxonomic_species(n = 1)
```

Arguments

n	(integer) number of things to get, any non-negative integer

Names

Names were taken from Theplantlist. 500 genera names and 500 epithets were chosen at random from the set of 10,000 names in the dataset in the `taxize` package. Theplantlist is, as it says on the tin, composed of plant names - so these fake names are derived from plant names if that matters to you. These may generate names that match those of real taxa, but may not as well.

Taxonomic authority

Randomly, the taxonomic authority is in parentheses - which represents that the given authority was not the original authority.

See Also

TaxonomyProvider

Examples

```
ch_taxonomic_genus()
ch_taxonomic_genus(10)
ch_taxonomic_genus(500)

ch_taxonomic_epithet()
ch_taxonomic_epithet(10)
ch_taxonomic_epithet(500)

ch_taxonomic_species()
ch_taxonomic_species(10)
ch_taxonomic_species(500)
```

Index