# Package 'charlatan'

May 8, 2026

**Type** Package

**Title** Make Fake Data

**Description** Make fake data that looks realistic, 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** 0.6.2

**License** MIT + file LICENSE

**URL** https://docs.ropensci.org/charlatan/, https://github.com/ropensci/charlatan

**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, spelling

**RoxygenNote** 7.3.3

**Config/testthat/edition** 3

**Depends** R (>= 2.10)

**Collate** 'address-provider-en_GB.R' 'address-provider-en_NZ.R' 'address-provider-en_US.R' 'address-provider-nl_NL.R' 'datetime-provider.R' 'address-provider.R' 'available_locales.R' 'available_providers.R' 'base-provider.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-en_US.R'
'internet-provider-fa_IR.R' 'internet-provider-fr_FR.R'
'internet-provider-hr_HR.R' 'internet-provider.R'
'isbn-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' 'languages.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.R'
'phonenumber-provider-en.R' 'phonenumber-provider-es.R'
'phonenumber-provider-fr.R' 'phonenumber-provider-nl.R'
'phonenumber-provider-rest.R' 'phonenumber-provider.R'
'sequence-provider.R' 'sequences.R' 'ssn-provider-all.R'
'ssn-provider.R' 'ssn.R' 'taxonomy-provider-all.R'
'taxonomy-provider.R' 'taxonomy.R' 'useragent-provider-all.R'
'useragent-provider.R' 'zzz.R'

**NeedsCompilation** no

**Author** Roel M. Hogervorst [cre, aut] (ORCID:

<https://orcid.org/0000-0001-7509-0328>),
Scott Chamberlain [aut] (ORCID:
<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] (ROR: <https://ror.org/019jywm96>)

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

**Repository** CRAN

**Date/Publication** 2026-01-14 06:10:43 UTC

# Contents

---

`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)

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

Scott Chamberlain

Kyle Voytovich

Martin Pedersen

## See Also

Useful links:

- https://docs.ropensci.org/charlatan/
- https://github.com/ropensci/charlatan
- Report bugs at https://github.com/ropensci/charlatan/issues

## 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_en_US$new()
x$color_name()
x$hex_color()
```

---

`AddressProvider` *AddressProvider*

---

## Description

Object to create addresses for a locale. Makes use of PersonProvider for creating street names.

## Details

When there is no PersonProvider for this locale, we default back to en_US.

## Value

Returns an AddressProvider object.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `AddressProvider`

## Methods

### Public methods:

- AddressProvider$new()
- AddressProvider$address()
- AddressProvider$city()
- AddressProvider$street_name()
- AddressProvider$street_address()
- AddressProvider$postcode()
- AddressProvider$init_person_provider()
- AddressProvider$clone()

**Method** `new()`**:** Create a new `AddressProvider` object

*Usage:*

```
AddressProvider$new()
```

*Returns:* A new `AddressProvider` object

**Method** `address()`**:** Create an address, a combination of street, postal code and city.

*Usage:*

```
AddressProvider$address()
```

**Method** `city()`**:** Create a city

*Usage:*

```
AddressProvider$city()
```

**Method** `street_name()`**:** Create a street name.

*Usage:*

```
AddressProvider$street_name()
```

**Method** `street_address()`**:** Create a street address, a combination of streetname and house indicator.

*Usage:*

```
AddressProvider$street_address()
```

**Method** `postcode()`**:** Create a postal code

*Usage:*

```
AddressProvider$postcode()
```

**Method** `init_person_provider()`**:** initialize the person provider (for use in addresses based on names)

*Usage:*

```
AddressProvider$init_person_provider(locale)
```

*Arguments:*

`locale` locale

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Note

You cannot instantiate the Parent providers. You must use one of the localized one.

## See Also

Other ParentProviders: `ColorProvider`, `CompanyProvider`, `InternetProvider`, `JobProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

---

`AddressProvider_en_GB` *AddressProvider for English, Great Britain*

---

## Description

Object to create addresses for a locale. Makes use of PersonProvider for creating street names.

## Details

When there is no PersonProvider for this locale, we default back to en_US.

## Value

Returns an AddressProvider object.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::AddressProvider` -> `AddressProvider_en_GB`

## Methods

### Public methods:

- `AddressProvider_en_GB$address()`
- `AddressProvider_en_GB$city()`
- `AddressProvider_en_GB$street_name()`
- `AddressProvider_en_GB$street_address()`
- `AddressProvider_en_GB$postcode()`
- `AddressProvider_en_GB$building_number()`
- `AddressProvider_en_GB$clone()`

**Method** `address()`: Create an address, a combination of street, postal code and city.

*Usage:*

```
AddressProvider_en_GB$address()
```

**Method** `city()`: Create a city

*Usage:*

```
AddressProvider_en_GB$city()
```

**Method** `street_name()`: Create a street name.

*Usage:*

```
AddressProvider_en_GB$street_name()
```

**Method** `street_address()`: Create a street address, a combination of streetname and house indicator.

*Usage:*

```
AddressProvider_en_GB$street_address()
```

**Method** `postcode()`: Create a postal code

*Usage:*

```
AddressProvider_en_GB$postcode()
```

**Method** `building_number()`: Create a building number

*Usage:*

```
AddressProvider_en_GB$building_number()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_NZ`, `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es_MX`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other GB: `PersonProvider_en_GB`, `PhoneNumberProvider_en_GB`

## Examples

```
(z <- AddressProvider_en_GB$new())
z$locale
z$postcode()
z$street_name()
z$address()
z$city()
```

AddressProvider_en_NZ *AddressProvider for New-Zealand*

## Description

Object to create addresses for a locale. Makes use of PersonProvider for creating street names.

## Details

When there is no PersonProvider for this locale, we default back to en_US.

## Value

Returns an AddressProvider object.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::AddressProvider` -> `AddressProvider_en_NZ`

## Methods

### Public methods:

- `AddressProvider_en_NZ$address()`
- `AddressProvider_en_NZ$city()`
- `AddressProvider_en_NZ$street_name()`
- `AddressProvider_en_NZ$street_address()`
- `AddressProvider_en_NZ$postcode()`
- `AddressProvider_en_NZ$clone()`

**Method** `address()`**:** Create an address, a combination of street, postal code and city.

*Usage:*

```
AddressProvider_en_NZ$address()
```

**Method** `city()`**:** Create a city

*Usage:*

```
AddressProvider_en_NZ$city()
```

**Method** `street_name()`**:** Create a street name

*Usage:*

```
AddressProvider_en_NZ$street_name()
```

**Method** `street_address()`**:** Create a street address , a combination of streetname and house indicator.

*Usage:*

```
AddressProvider_en_NZ$street_address()
```

**Method** `postcode()`**:** Create a postal code

*Usage:*

```
AddressProvider_en_NZ$postcode()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es_MX`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other NZ: `PersonProvider_en_NZ`, `PhoneNumberProvider_en_NZ`

## Examples

```
(z <- AddressProvider_en_NZ$new())
z$postcode()
z$street_name()
z$address()
z$city()
```

---

# `AddressProvider_en_US` *AddressProvider for United States of America*

---

## Description

Object to create addresses for a locale. Makes use of PersonProvider for creating street names.

## Details

When there is no PersonProvider for this locale, we default back to en_US.

## Value

Returns an AddressProvider object.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::AddressProvider -> AddressProvider_en_US

## Methods

### Public methods:

- AddressProvider_en_US$address()
- AddressProvider_en_US$civ_address()
- AddressProvider_en_US$mil_address()
- AddressProvider_en_US$city()
- AddressProvider_en_US$street_name()
- AddressProvider_en_US$street_address()
- AddressProvider_en_US$postcode()
- AddressProvider_en_US$building_number()
- AddressProvider_en_US$state()
- AddressProvider_en_US$clone()

**Method** `address()`**:** address

*Usage:*

```
AddressProvider_en_US$address()
```

**Method** `civ_address()`**:** civilian address, the type of address you would expect. Not to be confused with Military address which is also available for this locale.

*Usage:*

```
AddressProvider_en_US$civ_address()
```

**Method** `mil_address()`**:** Military address

*Usage:*

```
AddressProvider_en_US$mil_address()
```

**Method** `city()`**:** city

*Usage:*

```
AddressProvider_en_US$city()
```

**Method** `street_name()`**:** street name

*Usage:*

```
AddressProvider_en_US$street_name()
```

**Method** `street_address()`**:** street address

*Usage:*

```
AddressProvider_en_US$street_address()
```

**Method** `postcode()`**:** postal code

*Usage:*

```
AddressProvider_en_US$postcode()
```

**Method** `building_number()`**:** building number

*Usage:*

```
AddressProvider_en_US$building_number()
```

**Method** `state()`**:** state

*Usage:*

```
AddressProvider_en_US$state()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_NZ`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es_MX`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other US: `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_US`, `PhoneNumberProvider_en_US`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`

## Examples

```
(z <- AddressProvider_en_US$new())
z$locale
z$postcode()
z$street_name()
z$address()
z$city()
z$mil_address()
z$civ_address()
```

AddressProvider_nl_NL *AddressProvider for The Netherlands*

## Description

Object to create addresses for a locale. Makes use of PersonProvider for creating street names.

## Details

When there is no PersonProvider for this locale, we default back to en_US.

## Value

Returns an AddressProvider object.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::AddressProvider` -> `AddressProvider_nl_NL`

## Methods

### Public methods:

- `AddressProvider_nl_NL$address()`
- `AddressProvider_nl_NL$city()`
- `AddressProvider_nl_NL$street_name()`
- `AddressProvider_nl_NL$street_address()`
- `AddressProvider_nl_NL$postcode()`
- `AddressProvider_nl_NL$building_number()`
- `AddressProvider_nl_NL$province()`
- `AddressProvider_nl_NL$clone()`

**Method** `address()`**:** Create an address, a combination of street, postal code and city. The three components street, postal code and city are generated independently, so they are not related.

*Usage:*

```
AddressProvider_nl_NL$address()
```

**Method** `city()`**:** Create a city

*Usage:*

```
AddressProvider_nl_NL$city()
```

**Method** `street_name()`**:** Create a street name

*Usage:*

```
AddressProvider_nl_NL$street_name()
```

**Method** `street_address()`: Create a street address, a combination of streetname and house indicator.

*Usage:*

```
AddressProvider_nl_NL$street_address()
```

**Method** `postcode()`: Create a postal code, does not exclude impossible postcodes in The Netherlands (leading zero for examples) but looks good enough for most purposes.

*Usage:*

```
AddressProvider_nl_NL$postcode()
```

**Method** `building_number()`: building number.

*Usage:*

```
AddressProvider_nl_NL$building_number()
```

**Method** `province()`: Create a province.

*Usage:*

```
AddressProvider_nl_NL$province()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: `ElementProvider_nl_NL`, `JobProvider_nl_NL`, `PersonProvider_nl_NL`, `PhoneNumberProvider_nl_BE`, `PhoneNumberProvider_nl_NL`, `SSNProvider_nl_NL`, `dutch-language`

Other NL: `ElementProvider_nl_NL`, `JobProvider_nl_NL`, `PersonProvider_nl_NL`, `PhoneNumberProvider_nl_NL`, `SSNProvider_nl_NL`

## Examples

```
(z <- AddressProvider_nl_NL$new())
z$locale
z$postcode()
z$street_name()
z$address()
z$city()
z$province()
```

## arabic-language *Arabic Language*

### Description

Providers with the Arabic locale (ar).

### See Also

Other ar: LoremProvider_ar_AA

Other languages: bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## available_locales_df *Available locales*

### Description

A data.frame of locales available in **charlatan**. All localized providers are linked in the documentation.

### 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()

## available_providers *Available Providers*

### Description

A vector of locales available in **charlatan**

### Format

A vector with 22 items:

**provider** The base provider

## BareProvider *A NonLocalized Provider that contains all the selection and creation elements, but not the locales. That way we can still inherit an do useful stuff for providers that have no locale.*

### Description

A NonLocalized Provider that contains all the selection and creation elements, but not the locales. That way we can still inherit an do useful stuff for providers that have no locale.

A NonLocalized Provider that contains all the selection and creation elements, but not the locales. That way we can still inherit an do useful stuff for providers that have no locale.

### Active bindings

`provider` Display the provider name.

### Methods

#### Public methods:

- `BareProvider$random_element()`
- `BareProvider$random_element_prob()`
- `BareProvider$random_int()`
- `BareProvider$random_digit()`
- `BareProvider$random_digit_not_zero()`
- `BareProvider$random_digit_or_empty()`
- `BareProvider$random_digit_not_zero_or_empty()`
- `BareProvider$random_letter()`
- `BareProvider$numerify()`
- `BareProvider$lexify()`
- `BareProvider$bothify()`

- BareProvider$randomize_nb_elements()
- BareProvider$print()
- BareProvider$clone()

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

*Usage:*

```
BareProvider$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:*

```
BareProvider$random_element_prob(x)
```

*Arguments:*

`x` vector or list

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

*Usage:*

```
BareProvider$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:*

```
BareProvider$random_digit()
```

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

*Usage:*

```
BareProvider$random_digit_not_zero()
```

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

*Usage:*

```
BareProvider$random_digit_or_empty()
```

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

*Usage:*

```
BareProvider$random_digit_not_zero_or_empty()
```

**Method** `random_letter()`: random letter

*Usage:*

```
BareProvider$random_letter()
```

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

*Usage:*

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

*Arguments:*

`text` (character) a string

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

*Usage:*

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

*Arguments:*

`text` (character) a string

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

*Usage:*

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

*Arguments:*

`text` (character) a string

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

*Usage:*

```
BareProvider$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** `print()`**:** Print method for provider

*Usage:*

```
BareProvider$print(...)
```

*Arguments:*

`...` ignored by this method

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

---

`BaseProvider` *BaseProvider*

---

## Description

BaseProvider

BaseProvider

## Super class

`charlatan::BareProvider` -> `BaseProvider`

## Active bindings

`locale` (character) locale of this Provider.

## Methods

### Public methods:

- `BaseProvider$check_locale()`
- `BaseProvider$allowed_locales()`
- `BaseProvider$new()`
- `BaseProvider$print()`
- `BaseProvider$clone()`

**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** `allowed_locales()`**:** fetch the allowed locales for this provider

*Usage:*

```
BaseProvider$allowed_locales()
```

**Method** `new()`**:** Create a new Provider object

*Usage:*

```
BaseProvider$new()
```

*Returns:* A new object

**Method** `print()`**:** Print method for provider

*Usage:*

```
BaseProvider$print(...)
```

*Arguments:*

`...` ignored by this method

**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.

## Note

You cannot instantiate the Parent providers. You must use one of the localized one.

## Examples

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

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

---

`bosnian-language` *Bosnian Language*

---

## Description

Providers with the Bosnian locale (bs).

## See Also

Other bs: `PhoneNumberProvider_bs_BA`

Other languages: `arabic-language`, `bulgarian-language`, `chinese-language`, `croatian-language`, `czech-language`, `danish-language`, `dutch-language`, `english-language`, `farsi-language`, `finnish-language`, `french-language`, `german-language`, `greek-language`, `hebrew-language`, `hindi-language`, `hungarian-language`, `indonesian-language`, `italian-language`, `japanese-language`, `korean-language`, `latin-language`, `latvian-language`, `lithuanian-language`, `norwegian-language`, `polish-language`, `portuguese-language`, `russian-language`, `spanish-language`, `swedish-language`, `thai-language`, `turkish-language`, `ukrainian-language`

## bulgarian-language *Bulgarian Language*

### Description

Providers with the Bulgarian locale (bg).

### See Also

Other bg: CompanyProvider_bg_BG, InternetProvider_bg_BG, InternetProvider_en_NZ, PersonProvider_bg_BG, PhoneNumberProvider_bg_BG

Other languages: arabic-language, bosnian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## 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_en_US$new()
x$messy
charlatan_settings(messy = TRUE)
x <- PersonProvider_en_US$new()
x$messy
```

## chinese-language *Chinese Language*

### Description

Providers with the Chinese locale (zh).

## See Also

Other zh: JobProvider_zh_TW, LoremProvider_zh_CN, LoremProvider_zh_TW, PhoneNumberProvider_zh_TW

Other languages: arabic-language, bosnian-language, bulgarian-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

# 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 `colors()$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)
# or even ch_color_name(500)

ch_safe_color_name()
ch_safe_color_name(10)

ch_hex_color()
ch_hex_color(10)
# or even 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 `company()$allowed_locales()` for locales supported. |

## See Also

CompanyProvider

## Examples

```
ch_company()
ch_company(10)
# or even 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)
# or even ch_credit_card_number(n = 500)

ch_credit_card_security_code()
ch_credit_card_security_code(n = 10)
# or even 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)
# or even 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)
# or even 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_en_US

### Examples

```
ch_name()
ch_name(10)
# or even 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)
# or even 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)
```

ColorProvider *ColorProvider*

## Description

methods for colors create color names, hex values, rgb values or css values.

## Value

A ColorProvider object that can generate colors.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> ColorProvider

## Active bindings

all_colors (character) xxx

safe_colors (character) xxx

## Methods

### Public methods:

- ColorProvider$color_name()
- ColorProvider$hex_from_name()
- ColorProvider$safe_color_name()
- ColorProvider$hex_color()
- ColorProvider$safe_hex_color()
- ColorProvider$rgb_color()
- ColorProvider$rgb_css_color()
- ColorProvider$clone()

**Method** color_name(): color name

*Usage:*

```
ColorProvider$color_name()
```

**Method** hex_from_name(): get color by name

*Usage:*

```
ColorProvider$hex_from_name(name)
```

*Arguments:*

name color name

*Returns:* hex value

**Method** safe_color_name(): safe color name

*Usage:*

```
ColorProvider$safe_color_name()
```

**Method** `hex_color()`**:** hex color

*Usage:*

```
ColorProvider$hex_color()
```

**Method** `safe_hex_color()`**:** safe hex color

*Usage:*

```
ColorProvider$safe_hex_color()
```

**Method** `rgb_color()`**:** RGB color

*Usage:*

```
ColorProvider$rgb_color()
```

**Method** `rgb_css_color()`**:** RGB CSS color

*Usage:*

```
ColorProvider$rgb_css_color()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: `AddressProvider`, `CompanyProvider`, `InternetProvider`, `JobProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

---

`ColorProvider_en_US` *ColorProvider*

---

## Description

methods for colors create color names, hex values, rgb values or css values.

## Value

A ColorProvider object that can generate colors.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::ColorProvider` -> `ColorProvider_en_US`

## Methods

### Public methods:

- `ColorProvider_en_US$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_NZ`, `AddressProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es_MX`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other US: `AddressProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_US`, `PhoneNumberProvider_en_US`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`

## Examples

```
x <- ColorProvider_en_US$new()
x$locale
x$color_name()
x$safe_color_name()
x$hex_color()
x$safe_hex_color()
x$rgb_color()
x$rgb_css_color()
```

---

# ColorProvider_uk_UA *ColorProvider Ukrainian (Ukraine)*

---

## Description

methods for colors create color names, hex values, rgb values or css values.

## Value

A ColorProvider object that can generate colors.

### Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::ColorProvider -> ColorProvider_uk_UA

### Methods

#### Public methods:

- ColorProvider_uk_UA$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

### See Also

Other uk: JobProvider_uk_UA, PhoneNumberProvider_uk_UA, ukrainian-language

Other UA: JobProvider_uk_UA, PhoneNumberProvider_uk_UA

### Examples

```
x <- ColorProvider_uk_UA$new()
x$locale
x$color_name()
x$safe_color_name()
x$hex_color()
x$safe_hex_color()
x$rgb_color()
x$rgb_css_color()
```

---

## CompanyProvider *CompanyProvider*

---

### Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

### Value

A CompanyProvider object that can create companies.

### Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> CompanyProvider

## Methods

### Public methods:

- CompanyProvider$new()
- CompanyProvider$company()
- CompanyProvider$catch_phrase()
- CompanyProvider$bs()
- CompanyProvider$clone()

**Method** `new()`**:** Create a new `CompanyProvider` object

*Usage:*

```
CompanyProvider$new()
```

*Returns:* A new `CompanyProvider` object

**Method** `company()`**:** a company name

*Usage:*

```
CompanyProvider$company()
```

**Method** `catch_phrase()`**:** a catch phrase

*Usage:*

```
CompanyProvider$catch_phrase()
```

**Method** `bs()`**:** BS words

*Usage:*

```
CompanyProvider$bs()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `InternetProvider`, `JobProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

`CompanyProvider_bg_BG` *CompanyProvider for Bulgarian (Bulgaria)*

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_bg_BG`

## Methods

### Public methods:

- `CompanyProvider_bg_BG$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other bg: `InternetProvider_bg_BG`, `InternetProvider_en_NZ`, `PersonProvider_bg_BG`, `PhoneNumberProvider_bg_BG`
`bulgarian-language`

Other BG: `InternetProvider_bg_BG`, `InternetProvider_en_NZ`, `PersonProvider_bg_BG`, `PhoneNumberProvider_bg_B`

## Examples

```
x <- CompanyProvider_bg_BG$new()
x$locale
x$company()
```

`CompanyProvider_cs_CZ` *CompanyProvider for Czech*

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_cs_CZ`

## Methods

### Public methods:

- `CompanyProvider_cs_CZ$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other cs: `InternetProvider_cs_CZ`, `PersonProvider_cs_CZ`, `PhoneNumberProvider_cs_CZ`, `czech-language`

Other CZ: `InternetProvider_cs_CZ`, `PersonProvider_cs_CZ`, `PhoneNumberProvider_cs_CZ`

## Examples

```
x <- CompanyProvider_cs_CZ$new()
x$locale
x$company()
```

CompanyProvider_de_DE *CompanyProvider for German (Germany)*

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::CompanyProvider -> CompanyProvider_de_DE

## Methods

### Public methods:

- CompanyProvider_de_DE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other de: InternetProvider_de_DE, PersonProvider_de_AT, PersonProvider_de_DE, PhoneNumberProvider_de_DE, german-language

Other DE: InternetProvider_de_DE, PersonProvider_de_DE, PhoneNumberProvider_de_DE

## Examples

```
x <- CompanyProvider_de_DE$new()
x$locale
x$company()
```

CompanyProvider_en_US *CompanyProvider for English (United States)*

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::CompanyProvider -> CompanyProvider_en_US

## Methods

### Public methods:

- CompanyProvider_en_US$company()
- CompanyProvider_en_US$clone()

**Method** `company()`**:** a company name

*Usage:*

```
CompanyProvider_en_US$company()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
x <- CompanyProvider_en_US$new()
x$locale
x$company()
x$catch_phrase()
x$bs()
```

---

`CompanyProvider_es_MX` *CompanyProvider Spanish (Mexico)*

---

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_es_MX`

## Methods

### Public methods:

- `CompanyProvider_es_MX$company()`
- `CompanyProvider_es_MX$clone()`

**Method** `company()`**:** a company name

*Usage:*

```
CompanyProvider_es_MX$company()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other es: `PersonProvider_es_ES`, `PersonProvider_es_MX`, `PhoneNumberProvider_es_ES`, `PhoneNumberProvider_es_P`
`spanish-language`

Other MX: `PersonProvider_es_MX`, `PhoneNumberProvider_es_MX`

## Examples

```
x <- CompanyProvider_es_MX$new()
x$locale
x$company()
x$catch_phrase()
x$bs()
```

---

`CompanyProvider_fa_IR` *CompanyProvider Persian (Iran)*

---

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_fa_IR`

## Methods

### Public methods:

- `CompanyProvider_fa_IR$company()`
- `CompanyProvider_fa_IR$clone()`

**Method** `company()`**:** a company name

*Usage:*

```
CompanyProvider_fa_IR$company()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fa: `InternetProvider_fa_IR`, `JobProvider_fa_IR`, `PersonProvider_fa_IR`, `PhoneNumberProvider_fa_IR`, `farsi-language`

Other IR: `InternetProvider_fa_IR`, `JobProvider_fa_IR`, `PersonProvider_fa_IR`, `PhoneNumberProvider_fa_IR`

## Examples

```
x <- CompanyProvider_fa_IR$new()
x$locale
x$company()
x$catch_phrase()
x$bs()
```

---

`CompanyProvider_fr_FR` *CompanyProvider for France (French)*

---

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_fr_FR`

## Methods

### Public methods:

- `CompanyProvider_fr_FR$catch_phrase()`
- `CompanyProvider_fr_FR$clone()`

**Method** `catch_phrase()`**:** generate a catch phrase for a company.

*Usage:*

```
CompanyProvider_fr_FR$catch_phrase()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other FR: `InternetProvider_fr_FR`, `JobProvider_fr_FR`, `PersonProvider_fr_FR`, `PhoneNumberProvider_fr_FR`

Other fr: `InternetProvider_fr_FR`, `JobProvider_fr_CH`, `JobProvider_fr_FR`, `PersonProvider_fr_CH`, `PersonProvider_fr_FR`, `PhoneNumberProvider_fr_CH`, `PhoneNumberProvider_fr_FR`, `french-language`

## Examples

```
x <- CompanyProvider_fr_FR$new()
x$locale
x$company()
x$catch_phrase()
```

---

`CompanyProvider_hr_HR` *CompanyProvider Croatian (Croatia)*

---

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::CompanyProvider` -> `CompanyProvider_hr_HR`

## Methods

### Public methods:

- `CompanyProvider_hr_HR$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other hr: `InternetProvider_hr_HR`, `JobProvider_hr_HR`, `PersonProvider_hr_HR`, `PhoneNumberProvider_hr_HR`, `croatian-language`

Other HR: `InternetProvider_hr_HR`, `JobProvider_hr_HR`, `PersonProvider_hr_HR`, `PhoneNumberProvider_hr_HR`

## Examples

```
x <- CompanyProvider_hr_HR$new()
x$locale
x$company()
x$catch_phrase()
x$bs()
```

CompanyProvider_it_IT *CompanyProvider Italian (Italy)*

## Description

company name/etc. methods

Note that you cannot instantiate this class, you can only use the localized versions such as CompanyProvider_en_US.

## Value

A CompanyProvider object that can create companies.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::CompanyProvider -> CompanyProvider_it_IT

## Methods

### Public methods:

- CompanyProvider_it_IT$company()
- CompanyProvider_it_IT$clone()

**Method** `company()`**:** a company name

*Usage:*

```
CompanyProvider_it_IT$company()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other it: PersonProvider_it_IT, PhoneNumberProvider_it_IT, italian-language

Other IT: PersonProvider_it_IT, PhoneNumberProvider_it_IT

## Examples

```
x <- CompanyProvider_it_IT$new()
x$locale
x$company()
x$catch_phrase()
x$bs()
```

## 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))
```

## croatian-language *Croatian Language*

### Description

Providers with the Croatian locale (hr).

## See Also

Other hr: CompanyProvider_hr_HR, InternetProvider_hr_HR, JobProvider_hr_HR, PersonProvider_hr_HR, PhoneNumberProvider_hr_HR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

# czech-language *Czech Language*

---

## Description

Providers with the Czech locale (cs).

## See Also

Other cs: CompanyProvider_cs_CZ, InternetProvider_cs_CZ, PersonProvider_cs_CZ, PhoneNumberProvider_cs_CZ

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

# danish-language *Danish Language*

---

## Description

Providers with the Danish locale (da).

## See Also

Other da: JobProvider_da_DK, PersonProvider_da_DK, PhoneNumberProvider_da_DK

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

DateTimeProvider *DateTimeProvider*

## Description

date and time methods

## Super class

charlatan::BareProvider -> DateTimeProvider

## Public fields

centuries (character) centuries in roman numerals

countries (list) countries list

## Methods

### Public methods:

- DateTimeProvider$unix_time()
- DateTimeProvider$date()
- DateTimeProvider$date_time()
- DateTimeProvider$date_time_fromtimestamp()
- DateTimeProvider$iso8601()
- DateTimeProvider$year()
- DateTimeProvider$century()
- DateTimeProvider$timezone()
- DateTimeProvider$date_time_between()
- DateTimeProvider$clone()

**Method** unix_time(): Get a timestamp between January 1, 1970 and now, unless passed explicit start_date or end_date values

*Usage:*

```
DateTimeProvider$unix_time(start_date = NULL, end_date = "now")
```

*Arguments:*

start_date start date, a valid date format

end_date start date, a valid date format, default: "now"

**Method** date(): Generate a date between January 1, 1970 and now, with given pattern

*Usage:*

```
DateTimeProvider$date(pattern = "%Y-%m-%d")
```

*Arguments:*

pattern date pattern, default: %Y-%m-%d

**Method** `date_time()`: Generate a date time between January 1, 1970 and now

*Usage:*

```
DateTimeProvider$date_time(tzinfo = NULL)
```

*Arguments:*

`tzinfo` timezone, see timezone

**Method** `date_time_fromtimestamp()`: Generate a iso8601 format date

*Usage:*

```
DateTimeProvider$date_time_fromtimestamp(timestamp, tzinfo = NULL)
```

*Arguments:*

`timestamp` a timestamp

`tzinfo` timezone, see timezone

**Method** `iso8601()`: Generate a iso8601 format date

*Usage:*

```
DateTimeProvider$iso8601(date, tzinfo = NULL)
```

*Arguments:*

`date` a date, in a valid date format

`tzinfo` timezone, see timezone

**Method** `year()`: generate a year

*Usage:*

```
DateTimeProvider$year()
```

**Method** `century()`: generate a century

*Usage:*

```
DateTimeProvider$century()
```

**Method** `timezone()`: generate a timezone

*Usage:*

```
DateTimeProvider$timezone()
```

**Method** `date_time_between()`: Generate a date time based on a random date between two given dates

*Usage:*

```
DateTimeProvider$date_time_between(start_date, end_date = "now", tzinfo = NULL)
```

*Arguments:*

`start_date` start date, a valid date format

`end_date` start date, a valid date format, default: "now"

`tzinfo` timezone, see timezone

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## References

https://en.wikipedia.org/wiki/Unix_time

https://en.wikipedia.org/wiki/Unix_time

## Examples

```
z <- DateTimeProvider$new()
z$countries
z$centuries
z$century()
z$timezone()
z$unix_time()
z$date("%Y-%M-%d")
z$date_time()
z$year()
z$iso8601("1932-02-12 05:32:12")
# z$iso8601("January 4, 1981")

# date time between a range of dates
(start_date <- Sys.time() - 604800L)
z$date_time_between(start_date = start_date)
# in the year 1900
z$date_time_between("1900-01-01 00:00:00 PST", "1900-12-31 00:00:00 PST")
z$date_time_between("1900-01-01", "1900-12-31")
```

---

# 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)
```

---

`dutch-language` *Dutch Language*

---

## Description

Providers with the Dutch locale (nl).

## See Also

Other nl: `AddressProvider_nl_NL`, `ElementProvider_nl_NL`, `JobProvider_nl_NL`, `PersonProvider_nl_NL`, `PhoneNumberProvider_nl_BE`, `PhoneNumberProvider_nl_NL`, `SSNProvider_nl_NL`

Other languages: `arabic-language`, `bosnian-language`, `bulgarian-language`, `chinese-language`, `croatian-language`, `czech-language`, `danish-language`, `english-language`, `farsi-language`, `finnish-language`, `french-language`, `german-language`, `greek-language`, `hebrew-language`, `hindi-language`, `hungarian-language`, `indonesian-language`, `italian-language`, `japanese-language`, `korean-language`, `latin-language`, `latvian-language`, `lithuanian-language`, `norwegian-language`, `polish-language`, `portuguese-language`, `russian-language`, `spanish-language`, `swedish-language`, `thai-language`, `turkish-language`, `ukrainian-language`

---

`ElementProvider_en_US` *ElementProvider for USA*

---

## Description

chemical elements methods

## Details

Data from Wikipedia at https://en.wikipedia.org/wiki/Chemical_element

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::ElementProvider` -> `ElementProvider_en_US`

## Methods

### Public methods:

- ElementProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
z <- ElementProvider_en_US$new()
z$symbol()
z$element()
```

---

# ElementProvider_nl_NL *ElementProvider for the Netherlands*

---

## Description

chemical elements methods

## Details

Data from Wikipedia at https://nl.wikipedia.org/wiki/Lijst_van_chemische_elementen

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::ElementProvider -> ElementProvider_nl_NL

## Methods

### Public methods:

- ElementProvider_nl_NL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_BE, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL, dutch-language

Other NL: AddressProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL

## Examples

```
z <- ElementProvider_nl_NL$new()
z$symbol()
z$element()
```

---

`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)
```

---

`english-language` *English Language*

---

## Description

Providers with the English locale (en).

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_NZ`, `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es` `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`

Other languages: `arabic-language`, `bosnian-language`, `bulgarian-language`, `chinese-language`, `croatian-language`, `czech-language`, `danish-language`, `dutch-language`, `farsi-language`, `finnish-language`, `french-language`, `german-language`, `greek-language`, `hebrew-language`, `hindi-language`, `hungarian-language`, `indonesian-language`, `italian-language`, `japanese-language`, `korean-language`, `latin-language`, `latvian-language`, `lithuanian-language`, `norwegian-language`, `polish-language`, `portuguese-language`, `russian-language`, `spanish-language`, `swedish-language`, `thai-language`, `turkish-language`, `ukrainian-language`

---

`farsi-language` *Farsi Language*

---

## Description

Providers with the Farsi locale (fa).

## See Also

Other fa: CompanyProvider_fa_IR, InternetProvider_fa_IR, JobProvider_fa_IR, PersonProvider_fa_IR, PhoneNumberProvider_fa_IR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

# FileProvider_en_US *File Provider for United States English*

---

## Description

Creates files and file types.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::FileProvider -> FileProvider_en_US

## Methods

### Public methods:

- FileProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
(x <- FileProvider_en_US$new())
x$locale
x$mime_type()
x$file_extension()
x$file_name()
x$file_path()
x$file_path(depth = 2)
x$file_path(depth = 3)
x$file_path(depth = 6)
```

---

# finnish-language *Finnish Language*

---

## Description

Providers with the Finnish locale (fl).

## See Also

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

# 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()
```

---

# french-language *French Language*

---

## Description

Providers with the French locale (fr).

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language,

korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

`german-language` *German Language*

---

## Description

Providers with the German locale (de).

## See Also

Other de: CompanyProvider_de_DE, InternetProvider_de_DE, PersonProvider_de_AT, PersonProvider_de_DE, PhoneNumberProvider_de_DE

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

`greek-language` *Greek Language*

---

## Description

Providers with the Greek locale (el).

## See Also

Other el: LoremProvider_el_GR, PhoneNumberProvider_el_GR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

hebrew-language *Hebrew Language*

---

## Description

Providers with the Hebrew locale (he).

## See Also

Other he: LoremProvider_he_IL, PhoneNumberProvider_he_IL

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

hindi-language *Hindi Language*

---

## Description

Providers with the Hindi locale (hi).

## See Also

Other hi: PhoneNumberProvider_hi_IN

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## hungarian-language *Hungarian Language*

### Description

Providers with the Hungarian locale (hu).

### See Also

Other hu: PhoneNumberProvider_hu_HU

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## indonesian-language *Indonesian Language*

### Description

Providers with the Indonesian locale (id).

### See Also

Other id: PhoneNumberProvider_id_ID

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

`InternetProvider` *InternetProvider*

## Description

internet methods, e.g., email addresses, domain names

Note that if a locale you set doesn't have a locale specific set of data for PersonProvider or CompanyProvider we fall back to `en_US` Also note that you

## Value

A InternetProvider object with specific functions for internet.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `InternetProvider`

## Methods

### Public methods:

- `InternetProvider$new()`
- `InternetProvider$to_ascii()`
- `InternetProvider$email()`
- `InternetProvider$safe_email()`
- `InternetProvider$free_email()`
- `InternetProvider$company_email()`
- `InternetProvider$ascii_email()`
- `InternetProvider$ascii_safe_email()`
- `InternetProvider$ascii_free_email()`
- `InternetProvider$ascii_company_email()`
- `InternetProvider$user_name()`
- `InternetProvider$tld()`
- `InternetProvider$free_email_domain()`
- `InternetProvider$url()`
- `InternetProvider$domain_name()`
- `InternetProvider$domain_word()`
- `InternetProvider$ipv4()`
- `InternetProvider$ipv6()`
- `InternetProvider$mac_address()`
- `InternetProvider$uri_page()`
- `InternetProvider$uri_path()`
- `InternetProvider$uri_extension()`
- `InternetProvider$uri()`

- InternetProvider$slug()
- InternetProvider$image_url()
- InternetProvider$clone()

**Method** `new()`**:** Create a new Provider object

*Usage:*

```
InternetProvider$new()
```

*Returns:* A new object

**Method** `to_ascii()`**:** convert to ascii

*Usage:*

```
InternetProvider$to_ascii(x)
```

*Arguments:*

`x` the stringn to convert to ascii

**Method** `email()`**:** get an email address

*Usage:*

```
InternetProvider$email(domain = NULL)
```

*Arguments:*

`domain` (character) a domain name, if not given, a random name is chosen

**Method** `safe_email()`**:** get a safe email address

*Usage:*

```
InternetProvider$safe_email()
```

**Method** `free_email()`**:** a free email address

*Usage:*

```
InternetProvider$free_email()
```

**Method** `company_email()`**:** company email address

*Usage:*

```
InternetProvider$company_email()
```

**Method** `ascii_email()`**:** ascii email address

*Usage:*

```
InternetProvider$ascii_email()
```

**Method** `ascii_safe_email()`**:** safe ascii email address

*Usage:*

```
InternetProvider$ascii_safe_email()
```

**Method** `ascii_free_email()`**:** an ascii free email address

*Usage:*

```
InternetProvider$ascii_free_email()
```

**Method** `ascii_company_email()`: ascii company email address

*Usage:*

```
InternetProvider$ascii_company_email()
```

**Method** `user_name()`: a user name

*Usage:*

```
InternetProvider$user_name()
```

**Method** `tld()`: a tld

*Usage:*

```
InternetProvider$tld()
```

**Method** `free_email_domain()`: free email domain name

*Usage:*

```
InternetProvider$free_email_domain()
```

**Method** `url()`: a url

*Usage:*

```
InternetProvider$url(schemes = NULL)
```

*Arguments:*

`schemes` (character vector) a url scheme, defaults are http and https

**Method** `domain_name()`: Produce an Internet domain name with the specified number of sub-domain levels

*Usage:*

```
InternetProvider$domain_name(levels = 1)
```

*Arguments:*

`levels` (integer) how many levels, must be >1

**Method** `domain_word()`: a domain word

*Usage:*

```
InternetProvider$domain_word()
```

**Method** `ipv4()`: an ipv4 address or network

*Usage:*

```
InternetProvider$ipv4(network = FALSE)
```

*Arguments:*

`network` (logical) produce a network

**Method** `ipv6()`: an ipv6 address or network

*Usage:*

```
InternetProvider$ipv6(network = FALSE)
```

*Arguments:*

`network` (logical) produce a network

**Method** `mac_address()`: a mac address

*Usage:*

```
InternetProvider$mac_address()
```

**Method** `uri_page()`: a uri page

*Usage:*

```
InternetProvider$uri_page()
```

**Method** `uri_path()`: a uri path

*Usage:*

```
InternetProvider$uri_path(deep = NULL)
```

*Arguments:*

`deep` how deep to go, an integer, if not given an integer between 1 and 4 (inclusive) is chosen

**Method** `uri_extension()`: a uri extension

*Usage:*

```
InternetProvider$uri_extension()
```

**Method** `uri()`: a uri

*Usage:*

```
InternetProvider$uri()
```

**Method** `slug()`: a slug

*Usage:*

```
InternetProvider$slug(value = NULL)
```

*Arguments:*

`value` (character) a string, if given, returns itself, if not, uses LoremProvider to get a random string. default: `NULL`

**Method** `image_url()`: Returns URL to placeholder image - Example: http://placehold.it/640x480

*Usage:*

```
InternetProvider$image_url(width = NULL, height = NULL)
```

*Arguments:*

`width` image width, in pixels

`height` image height, in pixels

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `CompanyProvider`, `JobProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

InternetProvider_bg_BG

*Internet provider for Bulgarian (Bulgaria)*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_bg_BG

## Methods

### Public methods:

- InternetProvider_bg_BG$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other bg: CompanyProvider_bg_BG, InternetProvider_en_NZ, PersonProvider_bg_BG, PhoneNumberProvider_bg_BG, bulgarian-language

Other BG: CompanyProvider_bg_BG, InternetProvider_en_NZ, PersonProvider_bg_BG, PhoneNumberProvider_bg_BG

## Examples

```
x <- InternetProvider_bg_BG$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

`InternetProvider_cs_CZ` *Internet provider Czech*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::InternetProvider` -> `InternetProvider_cs_CZ`

## Methods

### Public methods:

- `InternetProvider_cs_CZ$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other cs: `CompanyProvider_cs_CZ`, `PersonProvider_cs_CZ`, `PhoneNumberProvider_cs_CZ`, `czech-language`

Other CZ: `CompanyProvider_cs_CZ`, `PersonProvider_cs_CZ`, `PhoneNumberProvider_cs_CZ`

## Examples

```
x <- InternetProvider_cs_CZ$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

InternetProvider_de_DE

*Internet provider German (Germany)*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_de_DE

## Methods

### Public methods:

- InternetProvider_de_DE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other de: CompanyProvider_de_DE, PersonProvider_de_AT, PersonProvider_de_DE, PhoneNumberProvider_de_DE, german-language

Other DE: CompanyProvider_de_DE, PersonProvider_de_DE, PhoneNumberProvider_de_DE

## Examples

```
x <- InternetProvider_de_DE$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

InternetProvider_en_AU

*Internet provider English (Australia)*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_en_AU

## Methods

### Public methods:

- InternetProvider_en_AU$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other AU: PhoneNumberProvider_en_AU

## Examples

```
x <- InternetProvider_en_AU$new()
x$mac_address()
```

`InternetProvider_en_NZ`

*Internet provider for New-Zealand*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::InternetProvider` -> `InternetProvider_en_NZ`

## Methods

### Public methods:

- `InternetProvider_en_NZ$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other bg: `CompanyProvider_bg_BG`, `InternetProvider_bg_BG`, `PersonProvider_bg_BG`, `PhoneNumberProvider_bg_BG`, `bulgarian-language`

Other BG: `CompanyProvider_bg_BG`, `InternetProvider_bg_BG`, `PersonProvider_bg_BG`, `PhoneNumberProvider_bg_BG`

## Examples

```
x <- InternetProvider_en_NZ$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

InternetProvider_en_US

*Internet provider for United States*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_en_US

## Methods

### Public methods:

- InternetProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_er SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
x <- InternetProvider_en_US$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

InternetProvider_fa_IR *Internet provider for Iran*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_fa_IR

## Methods

### Public methods:

- InternetProvider_fa_IR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fa: CompanyProvider_fa_IR, JobProvider_fa_IR, PersonProvider_fa_IR, PhoneNumberProvider_fa_IR, farsi-language

Other IR: CompanyProvider_fa_IR, JobProvider_fa_IR, PersonProvider_fa_IR, PhoneNumberProvider_fa_IR

## Examples

```
x <- InternetProvider_fa_IR$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

InternetProvider_fr_FR *Internet provider for France*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::InternetProvider -> InternetProvider_fr_FR

## Methods

### Public methods:

- InternetProvider_fr_FR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR, french-language

Other FR: CompanyProvider_fr_FR, JobProvider_fr_FR, PersonProvider_fr_FR, PhoneNumberProvider_fr_FR

## Examples

```
x <- InternetProvider_fr_FR$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

`InternetProvider_hr_HR`

*Internet provider for Croatian (Croatia)*

## Description

methods for internet related data, like email addresses, usernames, and websites.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::InternetProvider` -> `InternetProvider_hr_HR`

## Methods

### Public methods:

- `InternetProvider_hr_HR$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other hr: `CompanyProvider_hr_HR`, `JobProvider_hr_HR`, `PersonProvider_hr_HR`, `PhoneNumberProvider_hr_HR`, `croatian-language`

Other HR: `CompanyProvider_hr_HR`, `JobProvider_hr_HR`, `PersonProvider_hr_HR`, `PhoneNumberProvider_hr_HR`

## Examples

```
x <- InternetProvider_hr_HR$new()
x$email()
x$free_email()
x$mac_address()
x$company_email()
```

ISBNProvider *ISBNProvider*

## Description

International Standard Book Number - Provider. ISBN starts with group code, all English language ISBN-10 codes start with a 0 or 1, and all German language books start with a 3. see https://en.wikipedia.org/wiki/List_of_ISBN_registration_groups.

Charlatan does not provide further helpers for you, but you can supply the prefix yourself, if for instance you want to create Mexican ISBNs you can by supplying the ISBN10 prefix 970, or for Andorra supply the ISBN 13 prefix 97899920 (that is 978 for ISBN13, and 99920 for Andorra).

## Super class

charlatan::BareProvider -> ISBNProvider

## Methods

### Public methods:

- ISBNProvider$isbn10()
- ISBNProvider$isbn13()
- ISBNProvider$clone()

**Method** `isbn10()`**:** Make a ISBN10 This is a completely random (apart from the prefix), but valid ISBN10 number.

*Usage:*

```
ISBNProvider$isbn10(n = 1, prefix = NULL)
```

*Arguments:*

`n` (integer) number of ISBN10s to make, default=1

`prefix` (integer/character) prefix for ISBN

**Method** `isbn13()`**:** Make a ISBN13. This is a completely random (apart from the prefix), but valid ISBN13 number.

*Usage:*

```
ISBNProvider$isbn13(n = 1, prefix = NULL)
```

*Arguments:*

`n` (integer) number of ISBN10s to make, default=1

`prefix` (integer/character) prefix for ISBN

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Examples

```
z <- ISBNProvider$new()
z$isbn10()
z$isbn13()
z$isbn10(10)
z$isbn13(100)
# or even z$isbn10(500)
```

---

`italian-language` *Italian Language*

---

## Description

Providers with the Italian locale (it).

## See Also

Other it: `CompanyProvider_it_IT`, `PersonProvider_it_IT`, `PhoneNumberProvider_it_IT`

Other languages: `arabic-language`, `bosnian-language`, `bulgarian-language`, `chinese-language`, `croatian-language`, `czech-language`, `danish-language`, `dutch-language`, `english-language`, `farsi-language`, `finnish-language`, `french-language`, `german-language`, `greek-language`, `hebrew-language`, `hindi-language`, `hungarian-language`, `indonesian-language`, `japanese-language`, `korean-language`, `latin-language`, `latvian-language`, `lithuanian-language`, `norwegian-language`, `polish-language`, `portuguese-language`, `russian-language`, `spanish-language`, `swedish-language`, `thai-language`, `turkish-language`, `ukrainian-language`

---

`japanese-language` *Japanese Language*

---

## Description

Providers with the Japanese locale (ja).

## See Also

Other ja: `LoremProvider_ja_JP`, `PersonProvider_ja_JP`, `PhoneNumberProvider_ja_JP`

Other languages: `arabic-language`, `bosnian-language`, `bulgarian-language`, `chinese-language`, `croatian-language`, `czech-language`, `danish-language`, `dutch-language`, `english-language`, `farsi-language`, `finnish-language`, `french-language`, `german-language`, `greek-language`, `hebrew-language`, `hindi-language`, `hungarian-language`, `indonesian-language`, `italian-language`, `korean-language`, `latin-language`, `latvian-language`, `lithuanian-language`, `norwegian-language`, `polish-language`, `portuguese-language`, `russian-language`, `spanish-language`, `swedish-language`, `thai-language`, `turkish-language`, `ukrainian-language`

JobProvider *JobProvider*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `JobProvider`

## Methods

### Public methods:

- `JobProvider$render()`
- `JobProvider$clone()`

**Method** `render()`**:** Make a job

*Usage:*

```
JobProvider$render()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `CompanyProvider`, `InternetProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

JobProvider_da_DK *Job provider for Danish*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_da_DK

## Methods

### Public methods:

- JobProvider_da_DK$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other da: PersonProvider_da_DK, PhoneNumberProvider_da_DK, danish-language

Other DK: PersonProvider_da_DK, PhoneNumberProvider_da_DK, PhoneNumberProvider_dk_DK

## Examples

```
x <- JobProvider_da_DK$new()
x$render()
```

JobProvider_en_US *Job provider for United States*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_en_US

## Methods

### Public methods:

- JobProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
x <- JobProvider_en_US$new()
x$render()
```

JobProvider_fa_IR *Job provider for Iran (Persian)*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_fa_IR

## Methods

### Public methods:

- JobProvider_fa_IR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fa: CompanyProvider_fa_IR, InternetProvider_fa_IR, PersonProvider_fa_IR, PhoneNumberProvider_fa_IR, farsi-language

Other IR: CompanyProvider_fa_IR, InternetProvider_fa_IR, PersonProvider_fa_IR, PhoneNumberProvider_fa_IR

## Examples

```
x <- JobProvider_fa_IR$new()
x$render()
```

JobProvider_fi_FI *Job provider for Finnish*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_fi_FI

## Methods

### Public methods:

- JobProvider_fi_FI$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fi: PersonProvider_fi_FI, PhoneNumberProvider_fi_FI

Other FI: PersonProvider_fi_FI, PhoneNumberProvider_fi_FI

## Examples

```
x <- JobProvider_fi_FI$new()
x$render()
```

JobProvider_fr_CH *Job provider for Zwitserland*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_fr_CH

## Methods

### Public methods:

- JobProvider_fr_CH$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_FR, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR, french-language

Other CH: PersonProvider_fr_CH, PhoneNumberProvider_fr_CH

## Examples

```
x <- JobProvider_fr_CH$new()
x$render()
```

JobProvider_fr_FR *Job provider for France*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_fr_FR

## Methods

### Public methods:

- JobProvider_fr_FR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR, french-language

Other FR: CompanyProvider_fr_FR, InternetProvider_fr_FR, PersonProvider_fr_FR, PhoneNumberProvider_fr_FR

## Examples

```
x <- JobProvider_fr_FR$new()
x$render()
```

JobProvider_hr_HR *Job provider for Croatia*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_hr_HR

## Methods

### Public methods:

- JobProvider_hr_HR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other hr: CompanyProvider_hr_HR, InternetProvider_hr_HR, PersonProvider_hr_HR, PhoneNumberProvider_hr_HR, croatian-language

Other HR: CompanyProvider_hr_HR, InternetProvider_hr_HR, PersonProvider_hr_HR, PhoneNumberProvider_hr_HR

## Examples

```
x <- JobProvider_hr_HR$new()
x$render()
```

JobProvider_nl_NL *Job provider for Netherlands*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_nl_NL

## Methods

### Public methods:

- JobProvider_nl_NL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, ElementProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_BE, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL, dutch-language

Other NL: AddressProvider_nl_NL, ElementProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL

## Examples

```
x <- JobProvider_nl_NL$new()
x$render()
```

JobProvider_pl_PL *Job provider for Poland*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_pl_PL

## Methods

### Public methods:

- JobProvider_pl_PL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other pl: PersonProvider_pl_PL, PhoneNumberProvider_pl_PL, polish-language

Other PL: PersonProvider_pl_PL, PhoneNumberProvider_pl_PL

## Examples

```
x <- JobProvider_pl_PL$new()
x$render()
```

JobProvider_ru_RU *Job provider for Russia*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_ru_RU

## Methods

### Public methods:

- JobProvider_ru_RU$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ru: LoremProvider_ru_RU, PhoneNumberProvider_ru_RU, russian-language

Other RU: LoremProvider_ru_RU, PhoneNumberProvider_ru_RU

## Examples

```
x <- JobProvider_ru_RU$new()
x$render()
```

JobProvider_uk_UA *Job provider for Ukraine*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_uk_UA

## Methods

### Public methods:

- JobProvider_uk_UA$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other uk: ColorProvider_uk_UA, PhoneNumberProvider_uk_UA, ukrainian-language

Other UA: ColorProvider_uk_UA, PhoneNumberProvider_uk_UA

## Examples

```
x <- JobProvider_uk_UA$new()
x$render()
```

JobProvider_zh_TW *Job provider for Taiwan*

## Description

generate jobs

## Value

A JobProvider object with methods for jobs

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::JobProvider -> JobProvider_zh_TW

## Methods

### Public methods:

- JobProvider_zh_TW$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other zh: LoremProvider_zh_CN, LoremProvider_zh_TW, PhoneNumberProvider_zh_TW, chinese-language

Other TW: LoremProvider_zh_TW, PhoneNumberProvider_zh_TW

## Examples

```
x <- JobProvider_zh_TW$new()
x$render()
```

## korean-language *Korean Language*

### Description

Providers with the Korean locale (ko).

### See Also

Other ko: PersonProvider_ko_KR, PhoneNumberProvider_ko_KR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## latin-language *Latin Language*

### Description

Providers with the Latin locale (la).

### See Also

Other la: LoremProvider_la

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## latvian-language *Latvian Language*

### Description

Providers with the Latvian locale (lv).

### See Also

Other lv: PersonProvider_lv_LV, PhoneNumberProvider_lv_LV

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## lithuanian-language *Lithuanian Language*

### Description

Providers with the Lithuanian locale (lv).

### See Also

Other lt: PersonProvider_lt_LT, PhoneNumberProvider_lt_LT

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

LoremProvider *LoremProvider*

## Description

lorem ipsum methods for generating random words in a language. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> LoremProvider

## Methods

### Public methods:

- LoremProvider$new()
- LoremProvider$word()
- LoremProvider$words()
- LoremProvider$sentence()
- LoremProvider$sentences()
- LoremProvider$paragraph()
- LoremProvider$paragraphs()
- LoremProvider$text()
- LoremProvider$clone()

**Method** `new()`**:** Create a new `LoremProvider` object

*Usage:*

```
LoremProvider$new(sentence_punctuation = ".", word_connector = " ")
```

*Arguments:*

`sentence_punctuation` (character) End of sentence punctuation

`word_connector` (character) Default connector between words

*Returns:* A new `LoremProvider` object

**Method** `word()`**:** Generate a random word

*Usage:*

```
LoremProvider$word(ext_words = NULL)
```

*Arguments:*

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* a single word

**Method** `words()`**:** Generate a character vector of random words

*Usage:*

```
LoremProvider$words(nb = 3, ext_words = NULL)
```

*Arguments:*

`nb` (integer) how many words to return

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* many words

**Method** `sentence()`**:** Generate a random sentence

*Usage:*

```
LoremProvider$sentence(
  nb_words = 6,
  variable_nb_words = TRUE,
  ext_words = NULL
)
```

*Arguments:*

`nb_words` (integer) around how many words the sentence should contain

`variable_nb_words` set to `FALSE` if you want exactly `nb` words returned, otherwise the result may include a number of words of `nb` +/-40% (with a minimum of 1)

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* a single sentence

**Method** `sentences()`**:** Generate a character vector of random sentences

*Usage:*

```
LoremProvider$sentences(nb = 3, ext_words = NULL)
```

*Arguments:*

`nb` (integer) how many sentences to return

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* many sentences

**Method** `paragraph()`**:** Generate a single paragraph

*Usage:*

```
LoremProvider$paragraph(
  nb_sentences = 3,
  variable_nb_sentences = TRUE,
  ext_words = NULL
)
```

*Arguments:*

`nb_sentences` (integer) around how many sentences the paragraph should contain

`variable_nb_sentences` set to `FALSE` if you want exactly `nb` sentences returned, otherwise the result may include a number of sentences of `nb` +/-40% (with a minimum of 1)

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* a single paragraph

**Method** `paragraphs()`**:** Generate many paragraphs

*Usage:*

```
LoremProvider$paragraphs(nb = 3, ext_words = NULL)
```

*Arguments:*

`nb` (integer) how many paragraphs to return

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* many paragraphs

**Method** `text()`**:** Generate a random text string. Depending on the `max_nb_chars`, returns a string made of words, sentences, or paragraphs.

*Usage:*

```
LoremProvider$text(max_nb_chars = 200, ext_words = NULL)
```

*Arguments:*

`max_nb_chars` Maximum number of characters the text should contain (minimum 5)

`ext_words` a character vector of words you would like to have instead of "Lorem ipsum"

*Returns:* character string of words

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Note

You cannot instantiate the Parent providers. You must use one of the localized one.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `CompanyProvider`, `InternetProvider`, `JobProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

---

`LoremProvider_ar_AA` *Lorem provider Arabic*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::LoremProvider` -> `LoremProvider_ar_AA`

## Methods

### Public methods:

- LoremProvider_ar_AA$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other ar: arabic-language

## Examples

```
x <- LoremProvider_ar_AA$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_ar_AA$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_el_GR *Lorem provider Greek (Greece)*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_el_GR

## Methods

### Public methods:

- LoremProvider_el_GR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other el: PhoneNumberProvider_el_GR, greek-language

Other GR: PhoneNumberProvider_el_GR

## Examples

```
x <- LoremProvider_el_GR$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_el_GR$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_en_US *Lorem provider English (USA)*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_en_US

## Methods

### Public methods:

- LoremProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
x <- LoremProvider_en_US$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_en_US$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_he_IL *Lorem provider Hebrew*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

### Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_he_IL

### Methods

#### Public methods:

- LoremProvider_he_IL$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

### See Also

Other he: PhoneNumberProvider_he_IL, hebrew-language

Other IL: PhoneNumberProvider_he_IL

### Examples

```
x <- LoremProvider_he_IL$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_he_IL$new(word_connector = " --- ")
x$paragraph(4)
```

---

## LoremProvider_ja_JP *Lorem provider Japanese*

---

### Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

### Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_ja_JP

## Methods

### Public methods:

- LoremProvider_ja_JP$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ja: PersonProvider_ja_JP, PhoneNumberProvider_ja_JP, japanese-language

Other JP: PersonProvider_ja_JP, PhoneNumberProvider_ja_JP

## Examples

```
x <- LoremProvider_ja_JP$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_ja_JP$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_la *Lorem provider Latin*

---

## Description

lorem ipsum methods for generating random words in a language. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_la

## Methods

### Public methods:

- `LoremProvider_la$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other la: `latin-language`

## Examples

```
x <- LoremProvider_la$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_la$new(word_connector = " --- ")
x$paragraph(4)
```

---

`LoremProvider_ru_RU` *Lorem provider Russian (Russia)*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::LoremProvider` -> `LoremProvider_ru_RU`

## Methods

### Public methods:

- LoremProvider_ru_RU$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other ru: JobProvider_ru_RU, PhoneNumberProvider_ru_RU, russian-language

Other RU: JobProvider_ru_RU, PhoneNumberProvider_ru_RU

## Examples

```
x <- LoremProvider_ru_RU$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_ru_RU$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_zh_CN *Lorem provider Chinese (China)*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_zh_CN

## Methods

### Public methods:

- LoremProvider_zh_CN$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other zh: JobProvider_zh_TW, LoremProvider_zh_TW, PhoneNumberProvider_zh_TW, chinese-language

## Examples

```
x <- LoremProvider_zh_CN$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_zh_CN$new(word_connector = " --- ")
x$paragraph(4)
```

---

# LoremProvider_zh_TW *Lorem provider Chinese (Taiwan)*

---

## Description

Methods for Lorem Ipsum generation. Lorem Ipsum is a placeholder text commonly used to demonstrate the visual form of a document or a typeface without relying on meaningful content.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::LoremProvider -> LoremProvider_zh_TW

## Methods

### Public methods:

- LoremProvider_zh_TW$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other zh: JobProvider_zh_TW, LoremProvider_zh_CN, PhoneNumberProvider_zh_TW, chinese-language

Other TW: JobProvider_zh_TW, PhoneNumberProvider_zh_TW

## Examples

```
x <- LoremProvider_zh_TW$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_zh_TW$new(word_connector = " --- ")
x$paragraph(4)
```

---

# norwegian-language *Norwegian Language*

---

## Description

Providers with the Norwegian locale (no).

## See Also

Other no: PersonProvider_no_NO, PhoneNumberProvider_no_NO

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

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)
```

PersonProvider *PersonProvider*

## Description

PersonProvider

PersonProvider

## Details

Methods for Persons, methods for generating names.

## Value

A PersonProvider object that can create names.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `PersonProvider`

## Active bindings

`messy` show current setting for messy. Either TRUE or FALSE depending on configuration and if this is even possible for the locale.

## Methods

### Public methods:

- `PersonProvider$new()`
- `PersonProvider$messy_is_possible()`
- `PersonProvider$change_messy()`
- `PersonProvider$render()`
- `PersonProvider$messy_switch()`
- `PersonProvider$first_name()`
- `PersonProvider$first_name_female()`
- `PersonProvider$first_name_male()`
- `PersonProvider$last_name()`
- `PersonProvider$last_name_female()`
- `PersonProvider$last_name_male()`
- `PersonProvider$prefix()`
- `PersonProvider$prefix_female()`
- `PersonProvider$prefix_male()`
- `PersonProvider$suffix()`
- `PersonProvider$suffix_female()`
- `PersonProvider$suffix_male()`

- PersonProvider$clone()

**Method** `new()`: Create a new `PersonProvider` object

*Usage:*

```
PersonProvider$new(messy = FALSE)
```

*Arguments:*

`messy` make it messy

**Method** `messy_is_possible()`: internal function to figure out if messy is a valid option for this locale.

*Usage:*

```
PersonProvider$messy_is_possible()
```

**Method** `change_messy()`: Change messy (if possible)

*Usage:*

```
PersonProvider$change_messy(messy)
```

*Arguments:*

`messy` TRUE or FALSE

**Method** `render()`: Make a person's name

*Usage:*

```
PersonProvider$render(fmt = NULL)
```

*Arguments:*

`fmt` (character) a name format, default: `NULL`

**Method** `messy_switch()`: messy switch (internal). Always return a text, when messy is allowed return a messy version, but otherwise return a clean version.

*Usage:*

```
PersonProvider$messy_switch(clean_choice, messy_choice)
```

*Arguments:*

`clean_choice` the clean version

`messy_choice` the messy version

**Method** `first_name()`: make a first name

*Usage:*

```
PersonProvider$first_name()
```

**Method** `first_name_female()`: make a female first name

*Usage:*

```
PersonProvider$first_name_female()
```

**Method** `first_name_male()`: make a male first name

*Usage:*

```
PersonProvider$first_name_male()
```

**Method** `last_name()`: make a last name

*Usage:*

```
PersonProvider$last_name()
```

**Method** `last_name_female()`: make a female last name

*Usage:*

```
PersonProvider$last_name_female()
```

**Method** `last_name_male()`: make a male last name

*Usage:*

```
PersonProvider$last_name_male()
```

**Method** `prefix()`: make a name prefix

*Usage:*

```
PersonProvider$prefix()
```

**Method** `prefix_female()`: make a female name prefix

*Usage:*

```
PersonProvider$prefix_female()
```

**Method** `prefix_male()`: make a male name prefix

*Usage:*

```
PersonProvider$prefix_male()
```

**Method** `suffix()`: make a name suffix

*Usage:*

```
PersonProvider$suffix()
```

**Method** `suffix_female()`: make a female name suffix

*Usage:*

```
PersonProvider$suffix_female()
```

**Method** `suffix_male()`: make a male name suffix

*Usage:*

```
PersonProvider$suffix_male()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Note

You cannot instantiate the Parent providers. You must use one of the localized one.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `CompanyProvider`, `InternetProvider`, `JobProvider`, `LoremProvider`, `PhoneNumberProvider`, `TaxonomyProvider`, `UserAgentProvider`

PersonProvider_bg_BG *Person Provider for Bulgarian (Bulgaria)*

## Description

Person Provider for Bulgarian (Bulgaria)

Person Provider for Bulgarian (Bulgaria)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_bg_BG

## Methods

### Public methods:

- PersonProvider_bg_BG$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other bg: CompanyProvider_bg_BG, InternetProvider_bg_BG, InternetProvider_en_NZ, PhoneNumberProvider_bg_B
bulgarian-language

Other BG: CompanyProvider_bg_BG, InternetProvider_bg_BG, InternetProvider_en_NZ, PhoneNumberProvider_bg_

## Examples

```
x <- PersonProvider_bg_BG$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

PersonProvider_cs_CZ *Person Provider for Czech (Czech Republic)*

---

## Description

Person Provider for Czech (Czech Republic)

Person Provider for Czech (Czech Republic)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_cs_CZ

## Methods

### Public methods:

- PersonProvider_cs_CZ$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other cs: CompanyProvider_cs_CZ, InternetProvider_cs_CZ, PhoneNumberProvider_cs_CZ, czech-language

Other CZ: CompanyProvider_cs_CZ, InternetProvider_cs_CZ, PhoneNumberProvider_cs_CZ

## Examples

```
x <- PersonProvider_cs_CZ$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_da_DK *Person Provider for Danish (Denmark)*

---

## Description

Person Provider for Danish (Denmark)

Person Provider for Danish (Denmark)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_da_DK

## Methods

### Public methods:

- PersonProvider_da_DK$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other da: JobProvider_da_DK, PhoneNumberProvider_da_DK, danish-language

Other DK: JobProvider_da_DK, PhoneNumberProvider_da_DK, PhoneNumberProvider_dk_DK

## Examples

```
x <- PersonProvider_da_DK$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_de_AT *Person Provider for Austrian German*

---

## Description

Person Provider for Austrian German

Person Provider for Austrian German

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_de_AT

## Methods

### Public methods:

- PersonProvider_de_AT$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other de: CompanyProvider_de_DE, InternetProvider_de_DE, PersonProvider_de_DE, PhoneNumberProvider_de_DE, german-language

## Examples

```
x <- PersonProvider_de_AT$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_de_DE *Person Provider for German (Germany)*

---

## Description

Person Provider for German (Germany)

Person Provider for German (Germany)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_de_DE

## Methods

### Public methods:

- PersonProvider_de_DE$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other de: CompanyProvider_de_DE, InternetProvider_de_DE, PersonProvider_de_AT, PhoneNumberProvider_de_DE, german-language

Other DE: CompanyProvider_de_DE, InternetProvider_de_DE, PhoneNumberProvider_de_DE

## Examples

```
x <- PersonProvider_de_DE$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_en_GB *Person Provider for English (Great Brittain)*

---

## Description

Person Provider for English (Great Brittain)

Person Provider for English (Great Brittain)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_en_GB

## Methods

### Public methods:

- PersonProvider_en_GB$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other GB: AddressProvider_en_GB, PhoneNumberProvider_en_GB

## Examples

```
x <- PersonProvider_en_GB$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_en_NZ *Person Provider for English (New Zealand)*

---

## Description

Person Provider for English (New Zealand)

Person Provider for English (New Zealand)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_en_NZ

## Methods

### Public methods:

- `PersonProvider_en_NZ$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_NZ`, `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es_MX`, `SSNProvider_en_US`, `TaxonomyProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other NZ: `AddressProvider_en_NZ`, `PhoneNumberProvider_en_NZ`

## Examples

```
x <- PersonProvider_en_NZ$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_en_US *Person Provider for English (United States)*

---

## Description

Person Provider for English (United States)

Person Provider for English (United States)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_en_US

## Methods

### Public methods:

- PersonProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
x <- PersonProvider_en_US$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

PersonProvider_es_ES *Person Provider for Spanish (Spain)*

## Description

Person Provider for Spanish (Spain)

Person Provider for Spanish (Spain)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_es_ES

## Methods

### Public methods:

- PersonProvider_es_ES$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other es: CompanyProvider_es_MX, PersonProvider_es_MX, PhoneNumberProvider_es_ES, PhoneNumberProvider_es_spanish-language

Other ES: PhoneNumberProvider_es_ES

## Examples

```
x <- PersonProvider_es_ES$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

`PersonProvider_es_MX` *Person Provider for Spanish (Mexico)*

---

## Description

Person Provider for Spanish (Mexico)

Person Provider for Spanish (Mexico)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::PersonProvider` -> `PersonProvider_es_MX`

## Methods

### Public methods:

- `PersonProvider_es_MX$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other es: CompanyProvider_es_MX, PersonProvider_es_ES, PhoneNumberProvider_es_ES, PhoneNumberProvider_es_spanish-language

Other MX: CompanyProvider_es_MX, PhoneNumberProvider_es_MX

## Examples

```
x <- PersonProvider_de_AT$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_fa_IR *Person Provider for Farsi (Iran)*

---

## Description

Person Provider for Farsi (Iran)

Person Provider for Farsi (Iran)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_fa_IR

## Methods

### Public methods:

- PersonProvider_fa_IR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fa: CompanyProvider_fa_IR, InternetProvider_fa_IR, JobProvider_fa_IR, PhoneNumberProvider_fa_IR, farsi-language

Other IR: CompanyProvider_fa_IR, InternetProvider_fa_IR, JobProvider_fa_IR, PhoneNumberProvider_fa_IR

## Examples

```
x <- PersonProvider_fa_IR$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_fi_FI *Person Provider for Finnish (Finland)*

---

## Description

Person Provider for Finnish (Finland)

Person Provider for Finnish (Finland)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_fi_FI

## Methods

### Public methods:

- PersonProvider_fi_FI$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fi: JobProvider_fi_FI, PhoneNumberProvider_fi_FI

Other FI: JobProvider_fi_FI, PhoneNumberProvider_fi_FI

## Examples

```
x <- PersonProvider_fi_FI$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_fr_CH *Person Provider for French (Switzerland)*

---

## Description

Person Provider for French (Switzerland)

Person Provider for French (Switzerland)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_fr_CH

## Methods

### Public methods:

- PersonProvider_fr_CH$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR, french-language

Other CH: JobProvider_fr_CH, PhoneNumberProvider_fr_CH

## Examples

```
x <- PersonProvider_fr_CH$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_fr_FR *Person Provider for French (France)*

---

## Description

Person Provider for French (France)

Person Provider for French (France)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_fr_FR

## Methods

### Public methods:

- PersonProvider_fr_FR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_CH, PhoneNumberProvider_fr_CH, PhoneNumberProvider_fr_FR, french-language

Other FR: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_FR, PhoneNumberProvider_fr_FR

## Examples

```
x <- PersonProvider_fr_FR$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_hr_HR *Person Provider for Croatian (Croatia)*

---

## Description

Person Provider for Croatian (Croatia)

Person Provider for Croatian (Croatia)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_hr_HR

## Methods

### Public methods:

- PersonProvider_hr_HR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other hr: CompanyProvider_hr_HR, InternetProvider_hr_HR, JobProvider_hr_HR, PhoneNumberProvider_hr_HR, croatian-language

Other HR: CompanyProvider_hr_HR, InternetProvider_hr_HR, JobProvider_hr_HR, PhoneNumberProvider_hr_HR

## Examples

```
x <- PersonProvider_hr_HR$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_it_IT *Person Provider for Italian (Italy)*

---

## Description

Person Provider for Italian (Italy)

Person Provider for Italian (Italy)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_it_IT

## Methods

### Public methods:

- PersonProvider_it_IT$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other it: CompanyProvider_it_IT, PhoneNumberProvider_it_IT, italian-language

Other IT: CompanyProvider_it_IT, PhoneNumberProvider_it_IT

## Examples

```
x <- PersonProvider_it_IT$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_ja_JP *Person Provider for Japanese (Japan)*

---

## Description

Person Provider for Japanese (Japan)

Person Provider for Japanese (Japan)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_ja_JP

## Methods

### Public methods:

- PersonProvider_ja_JP$kana_name()
- PersonProvider_ja_JP$first_kana_name()
- PersonProvider_ja_JP$last_kana_name()
- PersonProvider_ja_JP$first_kana_name_male()
- PersonProvider_ja_JP$first_kana_name_female()

- PersonProvider_ja_JP$clone()

**Method** `kana_name()`: create a kana name.

*Usage:*

```
PersonProvider_ja_JP$kana_name()
```

**Method** `first_kana_name()`: Generate first name (kana)

*Usage:*

```
PersonProvider_ja_JP$first_kana_name()
```

**Method** `last_kana_name()`: Generate last name (kana)

*Usage:*

```
PersonProvider_ja_JP$last_kana_name()
```

**Method** `first_kana_name_male()`: Generate first name male (kana)

*Usage:*

```
PersonProvider_ja_JP$first_kana_name_male()
```

**Method** `first_kana_name_female()`: Generate first name female (kana)

*Usage:*

```
PersonProvider_ja_JP$first_kana_name_female()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ja: LoremProvider_ja_JP, PhoneNumberProvider_ja_JP, japanese-language

Other JP: LoremProvider_ja_JP, PhoneNumberProvider_ja_JP

## Examples

```
x <- PersonProvider_ja_JP$new()
x$locale
x$render()
x$first_name()
x$first_kana_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

PersonProvider_ko_KR *Person Provider for Korean (Republic of Korea)*

## Description

Person Provider for Korean (Republic of Korea)

Person Provider for Korean (Republic of Korea)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_ko_KR

## Methods

### Public methods:

- PersonProvider_ko_KR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ko: PhoneNumberProvider_ko_KR, korean-language

Other KR: PhoneNumberProvider_ko_KR

## Examples

```
x <- PersonProvider_hr_HR$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
```

```
x$last_name_female()
x$last_name_male()
```

---

`PersonProvider_lt_LT` *Person Provider for Lithuanian (Lithuania)*

---

## Description

Person Provider for Lithuanian (Lithuania)

Person Provider for Lithuanian (Lithuania)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::PersonProvider` -> `PersonProvider_lt_LT`

## Methods

### Public methods:

- `PersonProvider_lt_LT$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other lt: `PhoneNumberProvider_lt_LT`, `lithuanian-language`

Other LT: `PhoneNumberProvider_lt_LT`

## Examples

```
x <- PersonProvider_lt_LT$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

`PersonProvider_lv_LV` *Person Provider for Latvian (Latvia)*

---

## Description

Person Provider for Latvian (Latvia)

Person Provider for Latvian (Latvia)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::PersonProvider` -> `PersonProvider_lv_LV`

## Methods

### Public methods:

- `PersonProvider_lv_LV$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other lv: PhoneNumberProvider_lv_LV, latvian-language

Other LV: PhoneNumberProvider_lv_LV

## Examples

```
x <- PersonProvider_lv_LV$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_ne_NP *Person Provider for Nepali (Nepal)*

---

## Description

Person Provider for Nepali (Nepal)

Person Provider for Nepali (Nepal)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_ne_NP

## Methods

### Public methods:

- PersonProvider_ne_NP$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ne: PhoneNumberProvider_ne_NP

Other NP: PhoneNumberProvider_ne_NP

## Examples

```
x <- PersonProvider_ne_NP$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_nl_NL *Person Provider for Dutch (Netherlands)*

---

## Description

Person Provider for Dutch (Netherlands)

Person Provider for Dutch (Netherlands)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_nl_NL

## Methods

### Public methods:

- PersonProvider_nl_NL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PhoneNumberProvider_nl_BE, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL, dutch-language

Other NL: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL

## Examples

```
x <- PersonProvider_nl_NL$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_no_NO *Person Provider for Norwegian (Norway)*

---

## Description

Person Provider for Norwegian (Norway)

Person Provider for Norwegian (Norway)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_no_NO

## Methods

### Public methods:

- PersonProvider_no_NO$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other no: PhoneNumberProvider_no_NO, norwegian-language

Other NO: PhoneNumberProvider_nn_NO, PhoneNumberProvider_no_NO

## Examples

```
x <- PersonProvider_no_NO$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PersonProvider_pl_PL *Person Provider for Polish (Poland)*

---

## Description

Person Provider for Polish (Poland)

Person Provider for Polish (Poland)

## Details

Note for female and male components that we fall back on generic versions if the locale doesn't provide a male/female version. e.g., if no female first name we use first name

## Value

A PersonProvider object that can create names.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PersonProvider -> PersonProvider_pl_PL

## Methods

### Public methods:

- PersonProvider_pl_PL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other pl: JobProvider_pl_PL, PhoneNumberProvider_pl_PL, polish-language

Other PL: JobProvider_pl_PL, PhoneNumberProvider_pl_PL

## Examples

```
x <- PersonProvider_pl_PL$new()
x$locale
x$render()
x$first_name()
x$first_name_female()
x$first_name_male()
x$last_name()
x$last_name_female()
x$last_name_male()
```

---

# PhoneNumberProvider *PhoneNumberProvider*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> PhoneNumberProvider

## Methods

### Public methods:

- PhoneNumberProvider$render()
- PhoneNumberProvider$clone()

**Method** `render()`**:** Make a phone number

*Usage:*

```
PhoneNumberProvider$render()
```

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other ParentProviders: AddressProvider, ColorProvider, CompanyProvider, InternetProvider, JobProvider, LoremProvider, PersonProvider, TaxonomyProvider, UserAgentProvider

---

# PhoneNumberProvider_bg_BG

*PhoneNumberProvider for Bulgaria*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_bg_BG

## Methods

### Public methods:

- PhoneNumberProvider_bg_BG$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other bg: CompanyProvider_bg_BG, InternetProvider_bg_BG, InternetProvider_en_NZ, PersonProvider_bg_BG, bulgarian-language

Other BG: CompanyProvider_bg_BG, InternetProvider_bg_BG, InternetProvider_en_NZ, PersonProvider_bg_BG

## Examples

```
z <- PhoneNumberProvider_bg_BG$new()
z$render()
```

---

# PhoneNumberProvider_bs_BA

*PhoneNumberProvider for Bosnia and Herzegovina*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_bs_BA

## Methods

### Public methods:

- PhoneNumberProvider_bs_BA$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other bs: bosnian-language

## Examples

```
z <- PhoneNumberProvider_bs_BA$new()
z$render()
```

PhoneNumberProvider_cs_CZ *PhoneNumberProvider for Chechia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_cs_CZ

## Methods

### Public methods:

- PhoneNumberProvider_cs_CZ$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other cs: CompanyProvider_cs_CZ, InternetProvider_cs_CZ, PersonProvider_cs_CZ, czech-language

Other CZ: CompanyProvider_cs_CZ, InternetProvider_cs_CZ, PersonProvider_cs_CZ

## Examples

```
z <- PhoneNumberProvider_cs_CZ$new()
z$render()
```

PhoneNumberProvider_da_DK *PhoneNumberProvider for Denmark*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_da_DK

## Methods

### Public methods:

- PhoneNumberProvider_da_DK$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other da: JobProvider_da_DK, PersonProvider_da_DK, danish-language

Other DK: JobProvider_da_DK, PersonProvider_da_DK, PhoneNumberProvider_dk_DK

## Examples

```
z <- PhoneNumberProvider_da_DK$new()
z$render()
```

PhoneNumberProvider_de_DE
*PhoneNumberProvider for Germany*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_de_DE

## Methods

### Public methods:

- PhoneNumberProvider_de_DE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other de: CompanyProvider_de_DE, InternetProvider_de_DE, PersonProvider_de_AT, PersonProvider_de_DE, german-language

Other DE: CompanyProvider_de_DE, InternetProvider_de_DE, PersonProvider_de_DE

## Examples

```
z <- PhoneNumberProvider_de_DE$new()
z$render()
```

PhoneNumberProvider_dk_DK
*PhoneNumberProvider for Denmark*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::PhoneNumberProvider` -> `PhoneNumberProvider_dk_DK`

## Methods

### Public methods:

- `PhoneNumberProvider_dk_DK$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other DK: `JobProvider_da_DK`, `PersonProvider_da_DK`, `PhoneNumberProvider_da_DK`

## Examples

```
z <- PhoneNumberProvider_en_GB$new()
z$render()
```

PhoneNumberProvider_el_GR *PhoneNumberProvider for Greece*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_el_GR

## Methods

### Public methods:

- PhoneNumberProvider_el_GR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other el: LoremProvider_el_GR, greek-language

Other GR: LoremProvider_el_GR

## Examples

```
z <- PhoneNumberProvider_el_GR$new()
z$render()
```

PhoneNumberProvider_en_AU *PhoneNumberProvider for Australia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_en_AU

## Methods

### Public methods:

- PhoneNumberProvider_en_AU$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other AU: InternetProvider_en_AU

## Examples

```
z <- PhoneNumberProvider_en_AU$new()
z$render()
```

PhoneNumberProvider_en_CA *PhoneNumberProvider for Canada*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_en_CA

## Methods

### Public methods:

- PhoneNumberProvider_en_CA$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

## Examples

```
z <- PhoneNumberProvider_en_CA$new()
z$render()
```

PhoneNumberProvider_en_GB

*PhoneNumberProvider for Great Britain*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_en_GB

## Methods

### Public methods:

- PhoneNumberProvider_en_GB$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other GB: AddressProvider_en_GB, PersonProvider_en_GB

## Examples

```
z <- PhoneNumberProvider_en_GB$new()
z$render()
```

PhoneNumberProvider_en_NZ

*PhoneNumberProvider for New Zealand*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_en_NZ

## Methods

### Public methods:

- PhoneNumberProvider_en_NZ$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_US, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other NZ: AddressProvider_en_NZ, PersonProvider_en_NZ

## Examples

```
z <- PhoneNumberProvider_en_NZ$new()
z$render()
```

PhoneNumberProvider_en_US

*PhoneNumberProvider for United States of America*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_en_US

## Methods

### Public methods:

- PhoneNumberProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_es_MX, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
z <- PhoneNumberProvider_en_US$new()
z$render()
```

PhoneNumberProvider_es_ES *PhoneNumberProvider for Spain*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_es_ES

## Methods

### Public methods:

- PhoneNumberProvider_es_ES$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other es: CompanyProvider_es_MX, PersonProvider_es_ES, PersonProvider_es_MX, PhoneNumberProvider_es_PE, spanish-language

Other ES: PersonProvider_es_ES

## Examples

```
z <- PhoneNumberProvider_es_ES$new()
z$render()
```

PhoneNumberProvider_es_MX *PhoneNumberProvider for Mexico*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_es_MX

## Methods

### Public methods:

- PhoneNumberProvider_es_MX$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other MX: CompanyProvider_es_MX, PersonProvider_es_MX

## Examples

```
z <- PhoneNumberProvider_es_MX$new()
z$render()
```

PhoneNumberProvider_es_PE *PhoneNumberProvider for Peru*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_es_PE

## Methods

### Public methods:

- PhoneNumberProvider_es_PE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other es: CompanyProvider_es_MX, PersonProvider_es_ES, PersonProvider_es_MX, PhoneNumberProvider_es_ES, spanish-language

## Examples

```
z <- PhoneNumberProvider_es_PE$new()
z$render()
```

PhoneNumberProvider_fa_IR *PhoneNumberProvider for Iran*

## Description

methods for generating phone numbers

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_fa_IR

## Methods

### Public methods:

- PhoneNumberProvider_fa_IR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fa: CompanyProvider_fa_IR, InternetProvider_fa_IR, JobProvider_fa_IR, PersonProvider_fa_IR, farsi-language

Other IR: CompanyProvider_fa_IR, InternetProvider_fa_IR, JobProvider_fa_IR, PersonProvider_fa_IR

## Examples

```
z <- PhoneNumberProvider_fa_IR$new()
z$render()
```

PhoneNumberProvider_fi_FI *PhoneNumberProvider for Finland*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_fi_FI

## Methods

### Public methods:

- PhoneNumberProvider_fi_FI$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other fi: JobProvider_fi_FI, PersonProvider_fi_FI

Other FI: JobProvider_fi_FI, PersonProvider_fi_FI

## Examples

```
z <- PhoneNumberProvider_fi_FI$new()
z$render()
```

PhoneNumberProvider_fr_CH *PhoneNumberProvider for Switserland*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_fr_CH

## Methods

### Public methods:

- PhoneNumberProvider_fr_CH$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_FR, french-language

Other CH: JobProvider_fr_CH, PersonProvider_fr_CH

## Examples

```
z <- PhoneNumberProvider_fr_CH$new()
z$render()
```

PhoneNumberProvider_fr_FR *PhoneNumberProvider for France*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_fr_FR

## Methods

### Public methods:

- PhoneNumberProvider_fr_FR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other fr: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_CH, JobProvider_fr_FR, PersonProvider_fr_CH, PersonProvider_fr_FR, PhoneNumberProvider_fr_CH, french-language

Other FR: CompanyProvider_fr_FR, InternetProvider_fr_FR, JobProvider_fr_FR, PersonProvider_fr_FR

## Examples

```
z <- PhoneNumberProvider_fr_FR$new()
z$render()
```

PhoneNumberProvider_he_IL *PhoneNumberProvider for Israel*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_he_IL

## Methods

### Public methods:

- PhoneNumberProvider_he_IL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other he: LoremProvider_he_IL, hebrew-language

Other IL: LoremProvider_he_IL

## Examples

```
z <- PhoneNumberProvider_he_IL$new()
z$render()
```

PhoneNumberProvider_hi_IN *PhoneNumberProvider for India*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_hi_IN

## Methods

### Public methods:

- PhoneNumberProvider_hi_IN$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other hi: hindi-language

## Examples

```
z <- PhoneNumberProvider_hi_IN$new()
z$render()
```

PhoneNumberProvider_hr_HR *PhoneNumberProvider for Croatia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_hr_HR

## Methods

### Public methods:

- PhoneNumberProvider_hr_HR$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other hr: CompanyProvider_hr_HR, InternetProvider_hr_HR, JobProvider_hr_HR, PersonProvider_hr_HR, croatian-language

Other HR: CompanyProvider_hr_HR, InternetProvider_hr_HR, JobProvider_hr_HR, PersonProvider_hr_HR

## Examples

```
z <- PhoneNumberProvider_en_GB$new()
z$render()
```

PhoneNumberProvider_hu_HU
*PhoneNumberProvider for Hungary*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_hu_HU

## Methods

### Public methods:

- PhoneNumberProvider_hu_HU$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other hu: hungarian-language

## Examples

```
z <- PhoneNumberProvider_hu_HU$new()
z$render()
```

PhoneNumberProvider_id_ID
*PhoneNumberProvider for Indonesia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_id_ID

## Methods

### Public methods:

- PhoneNumberProvider_id_ID$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other id: indonesian-language

## Examples

```
z <- PhoneNumberProvider_id_ID$new()
z$render()
```

PhoneNumberProvider_it_IT *PhoneNumberProvider for Italy*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_it_IT

## Methods

### Public methods:

- PhoneNumberProvider_it_IT$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other it: CompanyProvider_it_IT, PersonProvider_it_IT, italian-language

Other IT: CompanyProvider_it_IT, PersonProvider_it_IT

## Examples

```
z <- PhoneNumberProvider_it_IT$new()
z$render()
```

PhoneNumberProvider_ja_JP *PhoneNumberProvider for Japan*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_ja_JP

## Methods

### Public methods:

- PhoneNumberProvider_ja_JP$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ja: LoremProvider_ja_JP, PersonProvider_ja_JP, japanese-language

Other JP: LoremProvider_ja_JP, PersonProvider_ja_JP

## Examples

```
z <- PhoneNumberProvider_en_GB$new()
z$render()
```

PhoneNumberProvider_ko_KR *PhoneNumberProvider for Korean Republic*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_ko_KR

## Methods

### Public methods:

- PhoneNumberProvider_ko_KR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ko: PersonProvider_ko_KR, korean-language

Other KR: PersonProvider_ko_KR

## Examples

```
z <- PhoneNumberProvider_ko_KR$new()
z$render()
```

PhoneNumberProvider_lt_LT *PhoneNumberProvider for Lithuania*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_lt_LT

## Methods

### Public methods:

- PhoneNumberProvider_lt_LT$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other lt: PersonProvider_lt_LT, lithuanian-language

Other LT: PersonProvider_lt_LT

## Examples

```
z <- PhoneNumberProvider_lt_LT$new()
z$render()
```

PhoneNumberProvider_lv_LV *PhoneNumberProvider for Latvia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_lv_LV

## Methods

### Public methods:

- PhoneNumberProvider_lv_LV$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other lv: PersonProvider_lv_LV, latvian-language

Other LV: PersonProvider_lv_LV

## Examples

```
z <- PhoneNumberProvider_lv_LV$new()
z$render()
```

PhoneNumberProvider_ne_NP *PhoneNumberProvider for Nepal*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_ne_NP

## Methods

### Public methods:

- PhoneNumberProvider_ne_NP$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ne: PersonProvider_ne_NP

Other NP: PersonProvider_ne_NP

## Examples

```
z <- PhoneNumberProvider_ne_NP$new()
z$render()
```

PhoneNumberProvider_nl_BE
*PhoneNumberProvider for Belgium*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_nl_BE

## Methods

### Public methods:

- PhoneNumberProvider_nl_BE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_NL, SSNProvider_nl_NL, dutch-language

## Examples

```
z <- PhoneNumberProvider_nl_BE$new()
z$render()
```

PhoneNumberProvider_nl_NL *PhoneNumberProvider for the Netherlands*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_nl_NL

## Methods

### Public methods:

- PhoneNumberProvider_nl_NL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_BE, SSNProvider_nl_NL, dutch-language

Other NL: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, SSNProvider_nl_NL

## Examples

```
z <- PhoneNumberProvider_nl_NL$new()
z$render()
```

PhoneNumberProvider_nn_NO
*PhoneNumberProvider for Norway (nn_NO)*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_nn_NO

## Methods

### Public methods:

- PhoneNumberProvider_nn_NO$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other NO: PersonProvider_no_NO, PhoneNumberProvider_no_NO

## Examples

```
z <- PhoneNumberProvider_nn_NO$new()
z$render()
```

PhoneNumberProvider_no_NO

*PhoneNumberProvider for Norway (no_NO)*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_no_NO

## Methods

### Public methods:

- PhoneNumberProvider_no_NO$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other no: PersonProvider_no_NO, norwegian-language

Other NO: PersonProvider_no_NO, PhoneNumberProvider_nn_NO

## Examples

```
z <- PhoneNumberProvider_no_NO$new()
z$render()
```

PhoneNumberProvider_pl_PL *PhoneNumberProvider for Poland*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_pl_PL

## Methods

### Public methods:

- PhoneNumberProvider_pl_PL$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other pl: JobProvider_pl_PL, PersonProvider_pl_PL, polish-language

Other PL: JobProvider_pl_PL, PersonProvider_pl_PL

## Examples

```
z <- PhoneNumberProvider_pl_PL$new()
z$render()
```

PhoneNumberProvider_pt_BR
*PhoneNumberProvider for Brazil*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::PhoneNumberProvider` -> `PhoneNumberProvider_pt_BR`

## Methods

### Public methods:

- `PhoneNumberProvider_pt_BR$clone()`

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other pt: `PhoneNumberProvider_pt_PT`, `portuguese-language`

## Examples

```
z <- PhoneNumberProvider_pt_BR$new()
z$render()
```

PhoneNumberProvider_pt_PT
*PhoneNumberProvider for Portugal*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_pt_PT

## Methods

### Public methods:

- PhoneNumberProvider_pt_PT$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other pt: PhoneNumberProvider_pt_BR, portuguese-language

## Examples

```
z <- PhoneNumberProvider_pt_PT$new()
z$render()
```

PhoneNumberProvider_ru_RU *PhoneNumberProvider for Russia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_ru_RU

## Methods

### Public methods:

- PhoneNumberProvider_ru_RU$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ru: JobProvider_ru_RU, LoremProvider_ru_RU, russian-language

Other RU: JobProvider_ru_RU, LoremProvider_ru_RU

## Examples

```
z <- PhoneNumberProvider_ru_RU$new()
z$render()
```

PhoneNumberProvider_sk_SK *PhoneNumberProvider for Slovakia*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_sk_SK

## Methods

### Public methods:

- PhoneNumberProvider_sk_SK$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Examples

```
z <- PhoneNumberProvider_sk_SK$new()
z$render()
```

PhoneNumberProvider_sv_SE *PhoneNumberProvider for Sweden*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_sv_SE

## Methods

### Public methods:

- PhoneNumberProvider_sv_SE$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other sv: swedish-language

## Examples

```
z <- PhoneNumberProvider_sv_SE$new()
z$render()
```

---

# PhoneNumberProvider_th_TH

*PhoneNumberProvider for Thailand*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_th_TH

## Methods

### Public methods:

- PhoneNumberProvider_th_TH$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other th: thai-language

## Examples

```
z <- PhoneNumberProvider_th_TH$new()
z$render()
```

---

`PhoneNumberProvider_tr_TR`

*PhoneNumberProvider for Turkey*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_tr_TR

## Methods

### Public methods:

- PhoneNumberProvider_tr_TR$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other tr: turkish-language

## Examples

```
z <- PhoneNumberProvider_tr_TR$new()
z$render()
```

---

PhoneNumberProvider_uk_UA

*PhoneNumberProvider for Ukraine*

---

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_uk_UA

## Methods

### Public methods:

- PhoneNumberProvider_uk_UA$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other uk: ColorProvider_uk_UA, JobProvider_uk_UA, ukrainian-language

Other UA: ColorProvider_uk_UA, JobProvider_uk_UA

## Examples

```
z <- PhoneNumberProvider_uk_UA$new()
z$render()
```

PhoneNumberProvider_zh_TW *PhoneNumberProvider for Taiwan*

## Description

methods for generating phone numbers

## Value

A PhoneNumberProvider object that can create phonenumbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::PhoneNumberProvider -> PhoneNumberProvider_zh_TW

## Methods

### Public methods:

- PhoneNumberProvider_zh_TW$clone()

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other zh: JobProvider_zh_TW, LoremProvider_zh_CN, LoremProvider_zh_TW, chinese-language

Other TW: JobProvider_zh_TW, LoremProvider_zh_TW

## Examples

```
z <- PhoneNumberProvider_zh_TW$new()
z$render()
```

## polish-language *Polish Language*

### Description

Providers with the Polish locale (pl).

### See Also

Other pl: JobProvider_pl_PL, PersonProvider_pl_PL, PhoneNumberProvider_pl_PL

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## portuguese-language *Portuguese Language*

### Description

Providers with the Portuguese locale (pt).

### See Also

Other pt: PhoneNumberProvider_pt_BR, PhoneNumberProvider_pt_PT

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## russian-language *Russian Language*

### Description

Providers with the Russian locale (ru).

### See Also

Other ru: JobProvider_ru_RU, LoremProvider_ru_RU, PhoneNumberProvider_ru_RU

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, spanish-language, swedish-language, thai-language, turkish-language, ukrainian-language

## SequenceProvider *SequenceProvider*

### Description

genetic sequence generator

### Super class

charlatan::BareProvider -> SequenceProvider

### Methods

#### Public methods:

- SequenceProvider$render()
- SequenceProvider$clone()

**Method** `render()`**:** Make a sequence

*Usage:*

```
SequenceProvider$render(length = 30)
```

*Arguments:*

`length` (integer) length of sequence to create. default: 30

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## Examples

```
z <- SequenceProvider$new()
z$render()
z$render(10)
z$render(100)
# or even z$render(500)
```

---

spanish-language *Spanish Language*

---

## Description

Providers with the Spanish locale (es).

## See Also

Other es: CompanyProvider_es_MX, PersonProvider_es_ES, PersonProvider_es_MX, PhoneNumberProvider_es_ES, PhoneNumberProvider_es_PE

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, swedish-language, thai-language, turkish-language, ukrainian-language

---

SSNProvider *SSNProvider*

---

## Description

methods for generating social security numbers

## Value

a SSNProvider object for generating social security numbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> SSNProvider

## Methods

### Public methods:

- SSNProvider$render()
- SSNProvider$clone()

**Method** `render()`**:** Make a SSN

*Usage:*

```
SSNProvider$render()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

---

`SSNProvider_en_US` *SSNProvider United States*

---

## Description

methods for generating social security numbers

## Value

a SSNProvider object for generating social security numbers.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::SSNProvider` -> `SSNProvider_en_US`

## Methods

### Public methods:

- SSNProvider_en_US$render()
- SSNProvider_en_US$clone()

**Method** `render()`**:** Make a SSN

*Usage:*

```
SSNProvider_en_US$render()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es TaxonomyProvider_en_US, UserAgentProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, TaxonomyProvider_en_US, UserAgentProvider_en_US

## Examples

```
z <- SSNProvider_en_US$new()
z$render()
```

---

# SSNProvider_nl_NL *SSNProvider the Netherlands*

---

## Description

methods for generating social security numbers

## Value

a SSNProvider object for generating social security numbers.

## Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> charlatan::SSNProvider -> SSNProvider_nl_NL

## Methods

### Public methods:

- SSNProvider_nl_NL$render()
- SSNProvider_nl_NL$clone()

**Method** `render()`**:** Make a SSN Dutch SSN (BSN) is 9 digits that follow a certain proof (elf-proef).

*Usage:*

```
SSNProvider_nl_NL$render()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other nl: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_BE, PhoneNumberProvider_nl_NL, dutch-language

Other NL: AddressProvider_nl_NL, ElementProvider_nl_NL, JobProvider_nl_NL, PersonProvider_nl_NL, PhoneNumberProvider_nl_NL

## Examples

```
z <- SSNProvider_nl_NL$new()
z$render()
```

---

# subclass *Create Localized Provider*

---

## Description

Create Localized Provider

## Usage

```
subclass(provider, locale = NULL)
```

## Arguments

| | |
|---|---|
| provider | The name of the provider you want to create |
| locale | Locale to use |

## Value

Localized provider

## Examples

```
x <- subclass("AddressProvider")
```

## swedish-language *Swedish Language*

### Description

Providers with the Swedish locale (sv).

### See Also

Other sv: PhoneNumberProvider_sv_SE

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, thai-language, turkish-language, ukrainian-language

## taxonomy *Create fake taxonomic names*

### Description

Create fake taxonomic names

### Usage

```
ch_taxonomic_genus(n = 1, locale = "en_US")

ch_taxonomic_epithet(n = 1, locale = "en_US")

ch_taxonomic_species(n = 1, locale = "en_US")
```

### Arguments

| | |
|---|---|
| n | (integer) number of things to get, any non-negative integer |
| locale | Locale for provider |

### 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)
# or even ch_taxonomic_genus(500)

ch_taxonomic_epithet()
ch_taxonomic_epithet(10)
# or even ch_taxonomic_epithet(500)

ch_taxonomic_species()
ch_taxonomic_species(10)
# or even ch_taxonomic_species(500)
```

---

## TaxonomyProvider *TaxonomyProvider*

---

### Description

Taxonomy provider for Generating Taxonomic names.

### 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.

### Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `TaxonomyProvider`

## Methods

### Public methods:

- TaxonomyProvider$genus()
- TaxonomyProvider$epithet()
- TaxonomyProvider$species()
- TaxonomyProvider$new()
- TaxonomyProvider$clone()

**Method** `genus()`**:** Get a genus name

*Usage:*

```
TaxonomyProvider$genus()
```

**Method** `epithet()`**:** Get an epithet name

*Usage:*

```
TaxonomyProvider$epithet()
```

**Method** `species()`**:** Get a binomial name (genus + epithet)

*Usage:*

```
TaxonomyProvider$species(authority = FALSE, date = FALSE)
```

*Arguments:*

`authority` Include authority. default: `FALSE`

`date` Include authority date. If `authority = FALSE`, this is ignored. default: `FALSE`

**Method** `new()`**:** Initialize new Taxonomy Provider.

*Usage:*

```
TaxonomyProvider$new()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: AddressProvider, ColorProvider, CompanyProvider, InternetProvider, JobProvider, LoremProvider, PersonProvider, PhoneNumberProvider, UserAgentProvider

TaxonomyProvider_en_US *TaxonomyProvider en_US*

## Description

Taxonomy provider for Generating Taxonomic names.

## 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.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::TaxonomyProvider` -> `TaxonomyProvider_en_US`

## Methods

### Public methods:

- `TaxonomyProvider_en_US$clone()`

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: `AddressProvider_en_GB`, `AddressProvider_en_NZ`, `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_AU`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_GB`, `PersonProvider_en_NZ`, `PersonProvider_en_US`, `PhoneNumberProvider_en_AU`, `PhoneNumberProvider_en_CA`, `PhoneNumberProvider_en_GB`, `PhoneNumberProvider_en_NZ`, `PhoneNumberProvider_en_US`, `PhoneNumberProvider_es` `SSNProvider_en_US`, `UserAgentProvider_en_US`, `english-language`

Other US: `AddressProvider_en_US`, `ColorProvider_en_US`, `CompanyProvider_en_US`, `ElementProvider_en_US`, `FileProvider_en_US`, `InternetProvider_en_US`, `JobProvider_en_US`, `LoremProvider_en_US`, `PersonProvider_en_US`, `PhoneNumberProvider_en_US`, `SSNProvider_en_US`, `UserAgentProvider_en_US`

## Examples

```
(z <- TaxonomyProvider_en_US$new())
z$genus()
z$epithet()
z$species()
z$species(authority = TRUE)
## FIXME - datetimeprovider slow - may be related to unix time problem
# z$species(authority = TRUE, date = TRUE)
```

---

# thai-language *Thai Language*

---

## Description

Providers with the Thai locale (th).

## See Also

Other th: PhoneNumberProvider_th_TH

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, turkish-language, ukrainian-language

---

# turkish-language *Turkish Language*

---

## Description

Providers with the Turkish locale (tr).

## See Also

Other tr: PhoneNumberProvider_tr_TR

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, ukrainian-language

## ukrainian-language *Ukrainian Language*

### Description

Providers with the Ukrainian locale (uk).

### See Also

Other uk: ColorProvider_uk_UA, JobProvider_uk_UA, PhoneNumberProvider_uk_UA

Other languages: arabic-language, bosnian-language, bulgarian-language, chinese-language, croatian-language, czech-language, danish-language, dutch-language, english-language, farsi-language, finnish-language, french-language, german-language, greek-language, hebrew-language, hindi-language, hungarian-language, indonesian-language, italian-language, japanese-language, korean-language, latin-language, latvian-language, lithuanian-language, norwegian-language, polish-language, portuguese-language, russian-language, spanish-language, swedish-language, thai-language, turkish-language

## UserAgentProvider *UserAgentProvider*

### Description

user agent methods For instance mac_processor, user_agents, chrome and firefox user agents.

### Value

UserAgentProvider object.

### Super classes

charlatan::BareProvider -> charlatan::BaseProvider -> UserAgentProvider

### Methods

#### Public methods:

- UserAgentProvider$mac_processor()
- UserAgentProvider$linux_processor()
- UserAgentProvider$user_agent()
- UserAgentProvider$chrome()
- UserAgentProvider$firefox()
- UserAgentProvider$safari()
- UserAgentProvider$opera()
- UserAgentProvider$internet_explorer()

- UserAgentProvider$windows_platform_token()
- UserAgentProvider$linux_platform_token()
- UserAgentProvider$mac_platform_token()
- UserAgentProvider$clone()

**Method** `mac_processor()`: a mac processor

*Usage:*

```
UserAgentProvider$mac_processor()
```

**Method** `linux_processor()`: a linux processor

*Usage:*

```
UserAgentProvider$linux_processor()
```

**Method** `user_agent()`: a random user agent string

*Usage:*

```
UserAgentProvider$user_agent()
```

**Method** `chrome()`: a chrome user agent string

*Usage:*

```
UserAgentProvider$chrome(
  version_from = 13,
  version_to = 63,
  build_from = 800,
  build_to = 899
)
```

*Arguments:*

`version_from` (integer) minimum version

`version_to` (integer) maximum version

`build_from` (integer) minimum build

`build_to` (integer) maximum build

**Method** `firefox()`: a firefox user agent string

*Usage:*

```
UserAgentProvider$firefox()
```

**Method** `safari()`: a safari user agent string

*Usage:*

```
UserAgentProvider$safari()
```

**Method** `opera()`: an opera user agent string

*Usage:*

```
UserAgentProvider$opera()
```

**Method** `internet_explorer()`: an internet explorer user agent string

*Usage:*

```
UserAgentProvider$internet_explorer()
```

**Method** `windows_platform_token()`: a windows platform token

*Usage:*

```
UserAgentProvider$windows_platform_token()
```

**Method** `linux_platform_token()`: a linux platform token

*Usage:*

```
UserAgentProvider$linux_platform_token()
```

**Method** `mac_platform_token()`: a mac platform token

*Usage:*

```
UserAgentProvider$mac_platform_token()
```

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

*Usage:*

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

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other ParentProviders: `AddressProvider`, `ColorProvider`, `CompanyProvider`, `InternetProvider`, `JobProvider`, `LoremProvider`, `PersonProvider`, `PhoneNumberProvider`, `TaxonomyProvider`

---

`UserAgentProvider_en_US`

*UserAgentProvider for United States of America*

---

## Description

user agent methods For instance mac_processor, user_agents, chrome and firefox user agents.

## Value

UserAgentProvider object.

## Super classes

`charlatan::BareProvider` -> `charlatan::BaseProvider` -> `charlatan::UserAgentProvider` -> `UserAgentProvider_en_US`

## Methods

### Public methods:

- UserAgentProvider_en_US$clone()

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

*Usage:*

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

*Arguments:*

deep Whether to make a deep clone.

## See Also

Other en: AddressProvider_en_GB, AddressProvider_en_NZ, AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_AU, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_GB, PersonProvider_en_NZ, PersonProvider_en_US, PhoneNumberProvider_en_AU, PhoneNumberProvider_en_CA, PhoneNumberProvider_en_GB, PhoneNumberProvider_en_NZ, PhoneNumberProvider_en_US, PhoneNumberProvider_es SSNProvider_en_US, TaxonomyProvider_en_US, english-language

Other US: AddressProvider_en_US, ColorProvider_en_US, CompanyProvider_en_US, ElementProvider_en_US, FileProvider_en_US, InternetProvider_en_US, JobProvider_en_US, LoremProvider_en_US, PersonProvider_en_US, PhoneNumberProvider_en_US, SSNProvider_en_US, TaxonomyProvider_en_US

## Examples

```
(x <- UserAgentProvider_en_US$new())
x$locale
x$mac_processor()
x$linux_processor()
x$user_agent()
x$chrome()
x$firefox()
x$internet_explorer()
x$opera()
x$safari()
```

# Index