

# Package: charlatan (via r-universe)

October 18, 2024

**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** 0.6.1

**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

**Roxygen** list(markdown = TRUE)

**VignetteBuilder** knitr

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

**Suggests** testthat, knitr, rmarkdown, ipaddress, stringi, spelling

**RoxygenNote** 7.3.2

**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'  
 'phonenummer-provider-en.R' 'phonenummer-provider-es.R'  
 'phonenummer-provider-fr.R' 'phonenummer-provider-nl.R'  
 'phonenummer-provider-rest.R' 'phonenummer-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'

**Repository** <https://ropensci.r-universe.dev>

**RemoteUrl** <https://github.com/ropensci/charlatan>

**RemoteRef** master

**RemoteSha** 15737c759846effc9790818d5a5ae440a8a34b96

## Contents

charlatan-package . . . . .	7
AddressProvider . . . . .	8
AddressProvider_en_GB . . . . .	10
AddressProvider_en_NZ . . . . .	12
AddressProvider_en_US . . . . .	13
AddressProvider_nl_NL . . . . .	16
arabic-language . . . . .	18
available_locales_df . . . . .	18
available_providers . . . . .	19
BareProvider . . . . .	19
BaseProvider . . . . .	22
bosnian-language . . . . .	23
bulgarian-language . . . . .	24
charlatan_locales . . . . .	24
charlatan_settings . . . . .	25
chinese-language . . . . .	25
ch_color . . . . .	26
ch_company . . . . .	27
ch_credit . . . . .	28
ch_currency . . . . .	29
ch_doi . . . . .	29
ch_generate . . . . .	30
ch_gene_sequence . . . . .	31
ch_job . . . . .	31
ch_missing . . . . .	32
ch_name . . . . .	33
ch_phone_number . . . . .	33
ch_ssn . . . . .	34
ColorProvider . . . . .	35
ColorProvider_en_US . . . . .	36
ColorProvider_uk_UA . . . . .	37
CompanyProvider . . . . .	38
CompanyProvider_bg_BG . . . . .	40
CompanyProvider_cs_CZ . . . . .	41
CompanyProvider_de_DE . . . . .	42
CompanyProvider_en_US . . . . .	43
CompanyProvider_es_MX . . . . .	44
CompanyProvider_fa_IR . . . . .	45
CompanyProvider_fr_FR . . . . .	46
CompanyProvider_hr_HR . . . . .	47
CompanyProvider_it_IT . . . . .	48

coordinates . . . . .	49
croatian-language . . . . .	49
czech-language . . . . .	50
danish-language . . . . .	50
DateTimeProvider . . . . .	51
date_time . . . . .	53
dutch-language . . . . .	54
ElementProvider_en_US . . . . .	54
ElementProvider_nl_NL . . . . .	55
elements . . . . .	56
english-language . . . . .	57
farsi-language . . . . .	57
FileProvider_en_US . . . . .	58
finnish-language . . . . .	59
fraudster . . . . .	59
french-language . . . . .	60
german-language . . . . .	61
greek-language . . . . .	61
hebrew-language . . . . .	62
hindi-language . . . . .	62
hungarian-language . . . . .	63
indonesian-language . . . . .	63
InternetProvider . . . . .	64
InternetProvider_bg_BG . . . . .	68
InternetProvider_cs_CZ . . . . .	69
InternetProvider_de_DE . . . . .	70
InternetProvider_en_AU . . . . .	71
InternetProvider_en_NZ . . . . .	72
InternetProvider_en_US . . . . .	73
InternetProvider_fa_IR . . . . .	74
InternetProvider_fr_FR . . . . .	75
InternetProvider_hr_HR . . . . .	76
ISBNProvider . . . . .	77
italian-language . . . . .	78
japanese-language . . . . .	78
JobProvider . . . . .	79
JobProvider_da_DK . . . . .	80
JobProvider_en_US . . . . .	81
JobProvider_fa_IR . . . . .	82
JobProvider_fi_FI . . . . .	83
JobProvider_fr_CH . . . . .	84
JobProvider_fr_FR . . . . .	85
JobProvider_hr_HR . . . . .	86
JobProvider_nl_NL . . . . .	87
JobProvider_pl_PL . . . . .	88
JobProvider_ru_RU . . . . .	89
JobProvider_uk_UA . . . . .	90
JobProvider_zh_TW . . . . .	91

korean-language . . . . .	92
latin-language . . . . .	92
latvian-language . . . . .	93
lithuanian-language . . . . .	93
LoremProvider . . . . .	94
LoremProvider_ar_AA . . . . .	96
LoremProvider_el_GR . . . . .	97
LoremProvider_en_US . . . . .	98
LoremProvider_he_IL . . . . .	99
LoremProvider_ja_JP . . . . .	100
LoremProvider_la . . . . .	101
LoremProvider_ru_RU . . . . .	102
LoremProvider_zh_CN . . . . .	103
LoremProvider_zh_TW . . . . .	104
norwegian-language . . . . .	105
numerics . . . . .	106
PersonProvider . . . . .	107
PersonProvider_bg_BG . . . . .	110
PersonProvider_cs_CZ . . . . .	111
PersonProvider_da_DK . . . . .	112
PersonProvider_de_AT . . . . .	113
PersonProvider_de_DE . . . . .	114
PersonProvider_en_GB . . . . .	115
PersonProvider_en_NZ . . . . .	116
PersonProvider_en_US . . . . .	117
PersonProvider_es_ES . . . . .	119
PersonProvider_es_MX . . . . .	120
PersonProvider_fa_IR . . . . .	121
PersonProvider_fi_FI . . . . .	122
PersonProvider_fr_CH . . . . .	123
PersonProvider_fr_FR . . . . .	124
PersonProvider_hr_HR . . . . .	125
PersonProvider_it_IT . . . . .	126
PersonProvider_ja_JP . . . . .	127
PersonProvider_ko_KR . . . . .	129
PersonProvider_lt_LT . . . . .	130
PersonProvider_lv_LV . . . . .	131
PersonProvider_ne_NP . . . . .	132
PersonProvider_nl_NL . . . . .	133
PersonProvider_no_NO . . . . .	134
PersonProvider_pl_PL . . . . .	135
PhoneNumberProvider . . . . .	136
PhoneNumberProvider_bg_BG . . . . .	137
PhoneNumberProvider_bs_BA . . . . .	138
PhoneNumberProvider_cs_CZ . . . . .	139
PhoneNumberProvider_da_DK . . . . .	140
PhoneNumberProvider_de_DE . . . . .	141
PhoneNumberProvider_dk_DK . . . . .	142

PhoneNumberProvider_el_GR . . . . .	143
PhoneNumberProvider_en_AU . . . . .	144
PhoneNumberProvider_en_CA . . . . .	145
PhoneNumberProvider_en_GB . . . . .	146
PhoneNumberProvider_en_NZ . . . . .	147
PhoneNumberProvider_en_US . . . . .	148
PhoneNumberProvider_es_ES . . . . .	149
PhoneNumberProvider_es_MX . . . . .	150
PhoneNumberProvider_es_PE . . . . .	151
PhoneNumberProvider_fa_IR . . . . .	152
PhoneNumberProvider_fi_FI . . . . .	153
PhoneNumberProvider_fr_CH . . . . .	154
PhoneNumberProvider_fr_FR . . . . .	155
PhoneNumberProvider_he_IL . . . . .	156
PhoneNumberProvider_hi_IN . . . . .	157
PhoneNumberProvider_hr_HR . . . . .	158
PhoneNumberProvider_hu_HU . . . . .	159
PhoneNumberProvider_id_ID . . . . .	160
PhoneNumberProvider_it_IT . . . . .	161
PhoneNumberProvider_ja_JP . . . . .	162
PhoneNumberProvider_ko_KR . . . . .	163
PhoneNumberProvider_lt_LT . . . . .	164
PhoneNumberProvider_lv_LV . . . . .	165
PhoneNumberProvider_ne_NP . . . . .	166
PhoneNumberProvider_nl_BE . . . . .	167
PhoneNumberProvider_nl_NL . . . . .	168
PhoneNumberProvider_nn_NO . . . . .	169
PhoneNumberProvider_no_NO . . . . .	170
PhoneNumberProvider_pl_PL . . . . .	171
PhoneNumberProvider_pt_BR . . . . .	172
PhoneNumberProvider_pt_PT . . . . .	173
PhoneNumberProvider_ru_RU . . . . .	174
PhoneNumberProvider_sk_SK . . . . .	175
PhoneNumberProvider_sv_SE . . . . .	175
PhoneNumberProvider_th_TH . . . . .	176
PhoneNumberProvider_tr_TR . . . . .	177
PhoneNumberProvider_uk_UA . . . . .	178
PhoneNumberProvider_zh_TW . . . . .	179
polish-language . . . . .	180
portuguese-language . . . . .	180
russian-language . . . . .	181
SequenceProvider . . . . .	181
spanish-language . . . . .	182
SSNProvider . . . . .	182
SSNProvider_en_US . . . . .	183
SSNProvider_nl_NL . . . . .	184
subclass . . . . .	185
swedish-language . . . . .	186

taxonomy . . . . .	186
TaxonomyProvider . . . . .	187
TaxonomyProvider_en_US . . . . .	189
thai-language . . . . .	190
turkish-language . . . . .	190
ukrainian-language . . . . .	191
UserAgentProvider . . . . .	191
UserAgentProvider_en_US . . . . .	193

<b>Index</b>	<b>195</b>
--------------	------------

---

charlatan-package	<i>charlatan</i>
-------------------	------------------

---

## 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	<i>Available Providers</i>
---------------------	----------------------------

---

**Description**

A vector of locales available in **charlatan**

**Format**

A vector with 22 items:

**provider** The base provider

---

BareProvider	<i>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.</i>
--------------	--

---

**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$numerify("#%asf221?")  
x$lexify("#%asf221?")  
x$bothify("#%asf221?")
```

---

bosnian-language	<i>Bosnian Language</i>
------------------	-------------------------

---

### 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	<i>Bulgarian Language</i>
--------------------	---------------------------

---

**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	<i>Available locales</i>
-------------------	--------------------------

---

**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 through `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 <code>company()\$allowed_locales()</code> 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	<i>Create fake credit card data</i>
-----------	-------------------------------------

---

## 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	<i>Create fake currencies</i>
-------------	-------------------------------

---

**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	<i>Create fake DOIs (Digital Object Identifiers)</i>
--------	--

---

**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	<i>Generate a fake dataset</i>
-------------	--------------------------------

---

### Description

Generate a fake dataset

### Usage

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

### Arguments

...	columns to include. must be in the allowed set. See <b>Allowed column names</b> 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	<i>Create fake gene sequences</i>
------------------	-----------------------------------

---

**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	<i>Create fake jobs</i>
--------	-------------------------

---

**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 <code>JobProvider\$new()\$allowed_locales()</code> 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**

<code>x</code>	Input vector, can be any class - only 1 vector
<code>n</code>	(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	<i>Create fake person names</i>
---------	---------------------------------

---

**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 <code>PersonProvider\$new()\$allowed_locales()</code> 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	<i>Create fake phone numbers</i>
-----------------	----------------------------------

---

**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	<i>ColorProvider</i>
---------------	----------------------

---

**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	<i>Create fake coordinates</i>
-------------	--------------------------------

---

**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	<i>Croatian Language</i>
-------------------	--------------------------

---

**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	<i>Czech Language</i>
----------------	-----------------------

---

**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	<i>Danish Language</i>
-----------------	------------------------

---

**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	<i>DateTimeProvider</i>
------------------	-------------------------

---

**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)

[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	<i>Dutch Language</i>
----------------	-----------------------

---

**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	<i>ElementProvider for USA</i>
-----------------------	--------------------------------

---

**Description**

chemical elements methods

**Details**

Data from Wikipedia at [https://en.wikipedia.org/wiki/Chemical\\_element](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](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	<i>English Language</i>
------------------	-------------------------

---

**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\\_en\\_US](#), [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	<i>Farsi Language</i>
----------------	-----------------------

---

**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	<i>Finnish Language</i>
------------------	-------------------------

---

**Description**

Providers with the Finnish locale (fi).

**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	<i>Fraudster - catch all client to make all types of fake data</i>
-----------	--

---

**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	<i>InternetProvider</i>
------------------	-------------------------

---

### 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://placeholder.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\\_en\\_US](#), [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	<i>ISBNProvider</i>
--------------	---------------------

---

## 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](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	<i>Job provider for United States</i>
-------------------	---------------------------------------

---

**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	<i>Job provider for Russia</i>
-------------------	--------------------------------

---

### 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	<i>Job provider for Ukraine</i>
-------------------	---------------------------------

---

### 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	<i>Korean Language</i>
-----------------	------------------------

---

**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	<i>Latin Language</i>
----------------	-----------------------

---

**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	<i>LoremProvider</i>
---------------	----------------------

---

## 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\\_e1\\_GR\\$clone\(\)](#)

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

*Usage:*

```
LoremProvider_e1_GR$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other e1: [PhoneNumberProvider\\_e1\\_GR](#), [greek-language](#)

Other GR: [PhoneNumberProvider\\_e1\\_GR](#)

## Examples

```
x <- LoremProvider_e1_GR$new()
x$word()
x$words(3)
x$words(6)
x$sentence()
x$paragraph()
x$paragraphs(3)
cat(x$paragraphs(6), sep = "\n")
x$text(19)
x <- LoremProvider_e1_GR$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	<i>PersonProvider</i>
----------------	-----------------------

---

## 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\\_BG](#), [bulgarian-language](#)

Other BG: [CompanyProvider\\_bg\\_BG](#), [InternetProvider\\_bg\\_BG](#), [InternetProvider\\_en\\_NZ](#), [PhoneNumberProvider\\_bg\\_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 Britain)*

---

**Description**

Person Provider for English (Great Britain)

Person Provider for English (Great Britain)

**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 phonenumber.

## 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\_e1\_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\\_e1\\_GR](#)

## Methods

### Public methods:

- [PhoneNumberProvider\\_e1\\_GR\\$clone\(\)](#)

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

*Usage:*

```
PhoneNumberProvider_e1_GR$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

## See Also

Other el: [LoremProvider\\_e1\\_GR](#), [greek-language](#)

Other GR: [LoremProvider\\_e1\\_GR](#)

## Examples

```
z <- PhoneNumberProvider_e1_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 phonenumber.

## 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 phonenumber.

### 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 phonenumber.

## 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 phonenumber.

## 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 phonenumber.

## 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 phonenumber.

## 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 Switzerland*

---

## 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 phonenumber.

## 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 phonenumber.

## 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 phonenumber.

## 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 phonenumber.

## 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*

---

## 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*

---

## 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\\_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 phonenumber.

## 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 phonenumber.

## 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 phonenumber.

**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	<i>Russian Language</i>
------------------	-------------------------

---

### 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	<i>SequenceProvider</i>
------------------	-------------------------

---

### 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	<i>Spanish Language</i>
------------------	-------------------------

---

**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	<i>SSNProvider</i>
-------------	--------------------

---

**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\\_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](#), [PhoneNumberProvider\\_en\\_US](#), [TaxonomyProvider\\_en\\_US](#), [UserAgentProvider\\_en\\_US](#)

**Examples**

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

---

SSNProvider_nl_NL	<i>SSNProvider the Netherlands</i>
-------------------	------------------------------------

---

**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-proof).

*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	<i>Swedish Language</i>
------------------	-------------------------

---

**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	<i>Create fake taxonomic names</i>
----------	------------------------------------

---

**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\\_en\\_US](#), [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$species(authority = TRUE)
## FIXME - datetimetypeprovider slow - may be related to unix time problem
# z$species(authority = TRUE, date = TRUE)
```

---

thai-language	<i>Thai Language</i>
---------------	----------------------

---

**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	<i>Turkish Language</i>
------------------	-------------------------

---

**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\\_en\\_US](#), [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

- \* **AA**
  - LoremProvider\_ar\_AA, [96](#)
- \* **AT**
  - PersonProvider\_de\_AT, [113](#)
- \* **AU**
  - InternetProvider\_en\_AU, [71](#)
  - PhoneNumberProvider\_en\_AU, [144](#)
- \* **BA**
  - PhoneNumberProvider\_bs\_BA, [138](#)
- \* **BE**
  - PhoneNumberProvider\_nl\_BE, [167](#)
- \* **BG**
  - CompanyProvider\_bg\_BG, [40](#)
  - InternetProvider\_bg\_BG, [68](#)
  - InternetProvider\_en\_NZ, [72](#)
  - PersonProvider\_bg\_BG, [110](#)
  - PhoneNumberProvider\_bg\_BG, [137](#)
- \* **BR**
  - PhoneNumberProvider\_pt\_BR, [172](#)
- \* **CA**
  - PhoneNumberProvider\_en\_CA, [145](#)
- \* **CH**
  - JobProvider\_fr\_CH, [84](#)
  - PersonProvider\_fr\_CH, [123](#)
  - PhoneNumberProvider\_fr\_CH, [154](#)
- \* **CN**
  - LoremProvider\_zh\_CN, [103](#)
- \* **CZ**
  - CompanyProvider\_cs\_CZ, [41](#)
  - InternetProvider\_cs\_CZ, [69](#)
  - PersonProvider\_cs\_CZ, [111](#)
  - PhoneNumberProvider\_cs\_CZ, [139](#)
- \* **DE**
  - CompanyProvider\_de\_DE, [42](#)
  - InternetProvider\_de\_DE, [70](#)
  - PersonProvider\_de\_DE, [114](#)
  - PhoneNumberProvider\_de\_DE, [141](#)
- \* **DK**
  - JobProvider\_da\_DK, [80](#)
  - PersonProvider\_da\_DK, [112](#)
  - PhoneNumberProvider\_da\_DK, [140](#)
  - PhoneNumberProvider\_dk\_DK, [142](#)
- \* **ES**
  - PersonProvider\_es\_ES, [119](#)
  - PhoneNumberProvider\_es\_ES, [149](#)
- \* **FI**
  - JobProvider\_fi\_FI, [83](#)
  - PersonProvider\_fi\_FI, [122](#)
  - PhoneNumberProvider\_fi\_FI, [153](#)
- \* **FR**
  - CompanyProvider\_fr\_FR, [46](#)
  - InternetProvider\_fr\_FR, [75](#)
  - JobProvider\_fr\_FR, [85](#)
  - PersonProvider\_fr\_FR, [124](#)
  - PhoneNumberProvider\_fr\_FR, [155](#)
- \* **GB**
  - AddressProvider\_en\_GB, [10](#)
  - PersonProvider\_en\_GB, [115](#)
  - PhoneNumberProvider\_en\_GB, [146](#)
- \* **GR**
  - LoremProvider\_el\_GR, [97](#)
  - PhoneNumberProvider\_el\_GR, [143](#)
- \* **HR**
  - CompanyProvider\_hr\_HR, [47](#)
  - InternetProvider\_hr\_HR, [76](#)
  - JobProvider\_hr\_HR, [86](#)
  - PersonProvider\_hr\_HR, [125](#)
  - PhoneNumberProvider\_hr\_HR, [158](#)
- \* **HU**
  - PhoneNumberProvider\_hu\_HU, [159](#)
- \* **ID**
  - PhoneNumberProvider\_id\_ID, [160](#)
- \* **IL**
  - LoremProvider\_he\_IL, [99](#)
  - PhoneNumberProvider\_he\_IL, [156](#)
- \* **IN**
  - PhoneNumberProvider\_hi\_IN, [157](#)
- \* **IR**

- CompanyProvider\_fa\_IR, [45](#)
- InternetProvider\_fa\_IR, [74](#)
- JobProvider\_fa\_IR, [82](#)
- PersonProvider\_fa\_IR, [121](#)
- PhoneNumberProvider\_fa\_IR, [152](#)
- \* **IT**
  - CompanyProvider\_it\_IT, [48](#)
  - PersonProvider\_it\_IT, [126](#)
  - PhoneNumberProvider\_it\_IT, [161](#)
- \* **JP**
  - LoremProvider\_ja\_JP, [100](#)
  - PersonProvider\_ja\_JP, [127](#)
  - PhoneNumberProvider\_ja\_JP, [162](#)
- \* **KR**
  - PersonProvider\_ko\_KR, [129](#)
  - PhoneNumberProvider\_ko\_KR, [163](#)
- \* **LT**
  - PersonProvider\_lt\_LT, [130](#)
  - PhoneNumberProvider\_lt\_LT, [164](#)
- \* **LV**
  - PersonProvider\_lv\_LV, [131](#)
  - PhoneNumberProvider\_lv\_LV, [165](#)
- \* **MX**
  - CompanyProvider\_es\_MX, [44](#)
  - PersonProvider\_es\_MX, [120](#)
  - PhoneNumberProvider\_es\_MX, [150](#)
- \* **NL**
  - AddressProvider\_nl\_NL, [16](#)
  - ElementProvider\_nl\_NL, [55](#)
  - JobProvider\_nl\_NL, [87](#)
  - PersonProvider\_nl\_NL, [133](#)
  - PhoneNumberProvider\_nl\_NL, [168](#)
  - SSNProvider\_nl\_NL, [184](#)
- \* **NO**
  - PersonProvider\_no\_NO, [134](#)
  - PhoneNumberProvider\_nn\_NO, [169](#)
  - PhoneNumberProvider\_no\_NO, [170](#)
- \* **NP**
  - PersonProvider\_ne\_NP, [132](#)
  - PhoneNumberProvider\_ne\_NP, [166](#)
- \* **NZ**
  - AddressProvider\_en\_NZ, [12](#)
  - PersonProvider\_en\_NZ, [116](#)
  - PhoneNumberProvider\_en\_NZ, [147](#)
- \* **PE**
  - PhoneNumberProvider\_es\_PE, [151](#)
- \* **PL**
  - JobProvider\_pl\_PL, [88](#)
  - PersonProvider\_pl\_PL, [135](#)
  - PhoneNumberProvider\_pl\_PL, [171](#)
- \* **PT**
  - PhoneNumberProvider\_pt\_PT, [173](#)
- \* **ParentProviders**
  - AddressProvider, [8](#)
  - ColorProvider, [35](#)
  - CompanyProvider, [38](#)
  - InternetProvider, [64](#)
  - JobProvider, [79](#)
  - LoremProvider, [94](#)
  - PersonProvider, [107](#)
  - PhoneNumberProvider, [136](#)
  - TaxonomyProvider, [187](#)
  - UserAgentProvider, [191](#)
- \* **RU**
  - JobProvider\_ru\_RU, [89](#)
  - LoremProvider\_ru\_RU, [102](#)
  - PhoneNumberProvider\_ru\_RU, [174](#)
- \* **SE**
  - PhoneNumberProvider\_sv\_SE, [175](#)
- \* **SK**
  - PhoneNumberProvider\_sk\_SK, [175](#)
- \* **TH**
  - PhoneNumberProvider\_th\_TH, [176](#)
- \* **TR**
  - PhoneNumberProvider\_tr\_TR, [177](#)
- \* **TW**
  - JobProvider\_zh\_TW, [91](#)
  - LoremProvider\_zh\_TW, [104](#)
  - PhoneNumberProvider\_zh\_TW, [179](#)
- \* **UA**
  - ColorProvider\_uk\_UA, [37](#)
  - JobProvider\_uk\_UA, [90](#)
  - PhoneNumberProvider\_uk\_UA, [178](#)
- \* **US**
  - AddressProvider\_en\_US, [13](#)
  - ColorProvider\_en\_US, [36](#)
  - CompanyProvider\_en\_US, [43](#)
  - ElementProvider\_en\_US, [54](#)
  - FileProvider\_en\_US, [58](#)
  - InternetProvider\_en\_US, [73](#)
  - JobProvider\_en\_US, [81](#)
  - LoremProvider\_en\_US, [98](#)
  - PersonProvider\_en\_US, [117](#)
  - PhoneNumberProvider\_en\_US, [148](#)
  - SSNProvider\_en\_US, [183](#)
  - TaxonomyProvider\_en\_US, [189](#)

- UserAgentProvider\_en\_US, 193
- \* **ar**
  - arabic-language, 18
  - LoremProvider\_ar\_AA, 96
- \* **bg**
  - bulgarian-language, 24
  - CompanyProvider\_bg\_BG, 40
  - InternetProvider\_bg\_BG, 68
  - InternetProvider\_en\_NZ, 72
  - PersonProvider\_bg\_BG, 110
  - PhoneNumberProvider\_bg\_BG, 137
- \* **bs**
  - bosnian-language, 23
  - PhoneNumberProvider\_bs\_BA, 138
- \* **cs**
  - CompanyProvider\_cs\_CZ, 41
  - czech-language, 50
  - InternetProvider\_cs\_CZ, 69
  - PersonProvider\_cs\_CZ, 111
  - PhoneNumberProvider\_cs\_CZ, 139
- \* **data**
  - available\_locales\_df, 18
  - available\_providers, 19
- \* **da**
  - danish-language, 50
  - JobProvider\_da\_DK, 80
  - PersonProvider\_da\_DK, 112
  - PhoneNumberProvider\_da\_DK, 140
- \* **de**
  - CompanyProvider\_de\_DE, 42
  - german-language, 61
  - InternetProvider\_de\_DE, 70
  - PersonProvider\_de\_AT, 113
  - PersonProvider\_de\_DE, 114
  - PhoneNumberProvider\_de\_DE, 141
- \* **dk**
  - PhoneNumberProvider\_dk\_DK, 142
- \* **el**
  - greek-language, 61
  - LoremProvider\_el\_GR, 97
  - PhoneNumberProvider\_el\_GR, 143
- \* **en**
  - AddressProvider\_en\_GB, 10
  - AddressProvider\_en\_NZ, 12
  - AddressProvider\_en\_US, 13
  - ColorProvider\_en\_US, 36
  - CompanyProvider\_en\_US, 43
  - ElementProvider\_en\_US, 54
  - english-language, 57
  - FileProvider\_en\_US, 58
  - InternetProvider\_en\_AU, 71
  - InternetProvider\_en\_US, 73
  - JobProvider\_en\_US, 81
  - LoremProvider\_en\_US, 98
  - PersonProvider\_en\_GB, 115
  - PersonProvider\_en\_NZ, 116
  - PersonProvider\_en\_US, 117
  - PhoneNumberProvider\_en\_AU, 144
  - PhoneNumberProvider\_en\_CA, 145
  - PhoneNumberProvider\_en\_GB, 146
  - PhoneNumberProvider\_en\_NZ, 147
  - PhoneNumberProvider\_en\_US, 148
  - PhoneNumberProvider\_es\_MX, 150
  - SSNProvider\_en\_US, 183
  - TaxonomyProvider\_en\_US, 189
  - UserAgentProvider\_en\_US, 193
- \* **es**
  - CompanyProvider\_es\_MX, 44
  - PersonProvider\_es\_ES, 119
  - PersonProvider\_es\_MX, 120
  - PhoneNumberProvider\_es\_ES, 149
  - PhoneNumberProvider\_es\_PE, 151
  - spanish-language, 182
- \* **fa**
  - CompanyProvider\_fa\_IR, 45
  - farsi-language, 57
  - InternetProvider\_fa\_IR, 74
  - JobProvider\_fa\_IR, 82
  - PersonProvider\_fa\_IR, 121
  - PhoneNumberProvider\_fa\_IR, 152
- \* **fi**
  - JobProvider\_fi\_FI, 83
  - PersonProvider\_fi\_FI, 122
  - PhoneNumberProvider\_fi\_FI, 153
- \* **fl**
  - finnish-language, 59
- \* **fr**
  - CompanyProvider\_fr\_FR, 46
  - french-language, 60
  - InternetProvider\_fr\_FR, 75
  - JobProvider\_fr\_CH, 84
  - JobProvider\_fr\_FR, 85
  - PersonProvider\_fr\_CH, 123
  - PersonProvider\_fr\_FR, 124
  - PhoneNumberProvider\_fr\_CH, 154
  - PhoneNumberProvider\_fr\_FR, 155

- \* **he**
  - hebrew-language, [62](#)
  - LoremProvider\_he\_IL, [99](#)
  - PhoneNumberProvider\_he\_IL, [156](#)
- \* **hi**
  - hindi-language, [62](#)
  - PhoneNumberProvider\_hi\_IN, [157](#)
- \* **hr**
  - CompanyProvider\_hr\_HR, [47](#)
  - croatian-language, [49](#)
  - InternetProvider\_hr\_HR, [76](#)
  - JobProvider\_hr\_HR, [86](#)
  - PersonProvider\_hr\_HR, [125](#)
  - PhoneNumberProvider\_hr\_HR, [158](#)
- \* **hu**
  - hungarian-language, [63](#)
  - PhoneNumberProvider\_hu\_HU, [159](#)
- \* **id**
  - indonesian-language, [63](#)
  - PhoneNumberProvider\_id\_ID, [160](#)
- \* **it**
  - CompanyProvider\_it\_IT, [48](#)
  - italian-language, [78](#)
  - PersonProvider\_it\_IT, [126](#)
  - PhoneNumberProvider\_it\_IT, [161](#)
- \* **ja**
  - japanese-language, [78](#)
  - LoremProvider\_ja\_JP, [100](#)
  - PersonProvider\_ja\_JP, [127](#)
  - PhoneNumberProvider\_ja\_JP, [162](#)
- \* **ko**
  - korean-language, [92](#)
  - PersonProvider\_ko\_KR, [129](#)
  - PhoneNumberProvider\_ko\_KR, [163](#)
- \* **languages**
  - arabic-language, [18](#)
  - bosnian-language, [23](#)
  - bulgarian-language, [24](#)
  - chinese-language, [25](#)
  - croatian-language, [49](#)
  - czech-language, [50](#)
  - danish-language, [50](#)
  - dutch-language, [54](#)
  - english-language, [57](#)
  - farsi-language, [57](#)
  - finnish-language, [59](#)
  - french-language, [60](#)
  - german-language, [61](#)
  - greek-language, [61](#)
  - hebrew-language, [62](#)
  - hindi-language, [62](#)
  - hungarian-language, [63](#)
  - indonesian-language, [63](#)
  - italian-language, [78](#)
  - japanese-language, [78](#)
  - korean-language, [92](#)
  - latin-language, [92](#)
  - latvian-language, [93](#)
  - lithuanian-language, [93](#)
  - norwegian-language, [105](#)
  - polish-language, [180](#)
  - portuguese-language, [180](#)
  - russian-language, [181](#)
  - spanish-language, [182](#)
  - swedish-language, [186](#)
  - thai-language, [190](#)
  - turkish-language, [190](#)
  - ukrainian-language, [191](#)
- \* **la**
  - latin-language, [92](#)
  - LoremProvider\_la, [101](#)
- \* **lt**
  - lithuanian-language, [93](#)
  - PersonProvider\_lt\_LT, [130](#)
  - PhoneNumberProvider\_lt\_LT, [164](#)
- \* **lv**
  - latvian-language, [93](#)
  - PersonProvider\_lv\_LV, [131](#)
  - PhoneNumberProvider\_lv\_LV, [165](#)
- \* **ne**
  - PersonProvider\_ne\_NP, [132](#)
  - PhoneNumberProvider\_ne\_NP, [166](#)
- \* **nl**
  - AddressProvider\_nl\_NL, [16](#)
  - dutch-language, [54](#)
  - ElementProvider\_nl\_NL, [55](#)
  - JobProvider\_nl\_NL, [87](#)
  - PersonProvider\_nl\_NL, [133](#)
  - PhoneNumberProvider\_nl\_BE, [167](#)
  - PhoneNumberProvider\_nl\_NL, [168](#)
  - SSNProvider\_nl\_NL, [184](#)
- \* **nn**
  - PhoneNumberProvider\_nn\_NO, [169](#)
- \* **no**
  - norwegian-language, [105](#)
  - PersonProvider\_no\_NO, [134](#)

- PhoneNumberProvider\_no\_NO, 170
- \* **package**
  - charlatan-package, 7
- \* **pl**
  - JobProvider\_pl\_PL, 88
  - PersonProvider\_pl\_PL, 135
  - PhoneNumberProvider\_pl\_PL, 171
  - polish-language, 180
- \* **pt**
  - PhoneNumberProvider\_pt\_BR, 172
  - PhoneNumberProvider\_pt\_PT, 173
  - portuguese-language, 180
- \* **ru**
  - JobProvider\_ru\_RU, 89
  - LoremProvider\_ru\_RU, 102
  - PhoneNumberProvider\_ru\_RU, 174
  - russian-language, 181
- \* **sk**
  - PhoneNumberProvider\_sk\_SK, 175
- \* **sv**
  - PhoneNumberProvider\_sv\_SE, 175
  - swedish-language, 186
- \* **th**
  - PhoneNumberProvider\_th\_TH, 176
  - thai-language, 190
- \* **tr**
  - PhoneNumberProvider\_tr\_TR, 177
  - turkish-language, 190
- \* **uk**
  - ColorProvider\_uk\_UA, 37
  - JobProvider\_uk\_UA, 90
  - PhoneNumberProvider\_uk\_UA, 178
  - ukrainian-language, 191
- \* **zh**
  - chinese-language, 25
  - JobProvider\_zh\_TW, 91
  - LoremProvider\_zh\_CN, 103
  - LoremProvider\_zh\_TW, 104
  - PhoneNumberProvider\_zh\_TW, 179
- AddressProvider, 8, 36, 39, 67, 79, 96, 109, 137, 188, 193
- AddressProvider\_en\_GB, 10, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- AddressProvider\_en\_NZ, 11, 12, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- AddressProvider\_en\_US, 11, 13, 13, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- AddressProvider\_nl\_NL, 16, 54, 56, 87, 134, 167, 168, 185
- arabic-language, 18
- available\_locales\_df, 18, 24
- available\_providers, 19
- BareProvider, 19
- BaseProvider, 22
- bosnian-language, 23
- bulgarian-language, 24
- ch\_beta (numerics), 106
- ch\_color, 26
- ch\_color\_name (ch\_color), 26
- ch\_company, 27
- ch\_credit, 28
- ch\_credit\_card\_number (ch\_credit), 28
- ch\_credit\_card\_provider (ch\_credit), 28
- ch\_credit\_card\_security\_code (ch\_credit), 28
- ch\_currency, 29
- ch\_date\_time (date\_time), 53
- ch\_doi, 29
- ch\_double (numerics), 106
- ch\_element\_element (elements), 56
- ch\_element\_symbol (elements), 56
- ch\_gene\_sequence, 31
- ch\_generate, 30
- ch\_generate(), 7
- ch\_hex\_color (ch\_color), 26
- ch\_integer (numerics), 106
- ch\_job, 31
- ch\_lat (coordinates), 49
- ch\_lnorm (numerics), 106
- ch\_lon (coordinates), 49
- ch\_missing, 32
- ch\_name, 33
- ch\_norm (numerics), 106
- ch\_phone\_number, 33
- ch\_position (coordinates), 49
- ch\_rgb\_color (ch\_color), 26
- ch\_rgb\_css\_color (ch\_color), 26
- ch\_safe\_color\_name (ch\_color), 26
- ch\_safe\_hex\_color (ch\_color), 26
- ch\_ssn, 34
- ch\_taxonomic\_epithet (taxonomy), 186

- ch\_taxonomic\_genus (taxonomy), 186
- ch\_taxonomic\_species (taxonomy), 186
- ch\_timezone (date\_time), 53
- ch\_unif (numerics), 106
- ch\_unix\_time (date\_time), 53
- charlatan (charlatan-package), 7
- charlatan-package, 7
- charlatan::AddressProvider, 10, 12, 14, 16
- charlatan::BareProvider, 8, 10, 12, 14, 16, 22, 35, 36, 38, 40–48, 51, 54, 55, 58, 64, 68–77, 79–91, 94, 96–98, 100–104, 107, 110–116, 118–127, 129–179, 181–184, 187, 189, 191, 193
- charlatan::BaseProvider, 8, 10, 12, 14, 16, 35, 36, 38, 40–48, 54, 55, 58, 64, 68–76, 79–91, 94, 96–98, 100–104, 107, 110–116, 118–127, 129–179, 182–184, 187, 189, 191, 193
- charlatan::ColorProvider, 36, 38
- charlatan::CompanyProvider, 40–48
- charlatan::ElementProvider, 54, 55
- charlatan::FileProvider, 58
- charlatan::InternetProvider, 68–76
- charlatan::JobProvider, 80–91
- charlatan::LoremProvider, 96–98, 100–104
- charlatan::PersonProvider, 110–116, 118–127, 129–135
- charlatan::PhoneNumberProvider, 137–179
- charlatan::SSNProvider, 183, 184
- charlatan::TaxonomyProvider, 189
- charlatan::UserAgentProvider, 193
- charlatan\_locales, 24
- charlatan\_locales(), 18
- charlatan\_settings, 25
- chinese-language, 25
- ColorProvider, 10, 26, 35, 39, 67, 79, 96, 109, 137, 188, 193
- ColorProvider\_en\_US, 11, 13, 15, 36, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- ColorProvider\_uk\_UA, 37, 90, 178, 191
- CompanyProvider, 10, 27, 36, 38, 64, 67, 79, 96, 109, 137, 188, 193
- CompanyProvider\_bg\_BG, 24, 40, 68, 72, 110, 138
- CompanyProvider\_cs\_CZ, 41, 50, 69, 112, 139
- CompanyProvider\_de\_DE, 42, 61, 70, 114, 115, 141
- CompanyProvider\_en\_US, 11, 13, 15, 37, 38, 40–43, 43, 44–48, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- CompanyProvider\_es\_MX, 44, 119, 121, 149–151, 182
- CompanyProvider\_fa\_IR, 45, 58, 74, 82, 122, 152
- CompanyProvider\_fr\_FR, 46, 60, 75, 84, 85, 124, 125, 154, 155
- CompanyProvider\_hr\_HR, 47, 50, 76, 86, 126, 158
- CompanyProvider\_it\_IT, 48, 78, 127, 161
- CoordinateProvider, 49
- coordinates, 49
- CreditCardProvider, 28
- croatian-language, 49
- CurrencyProvider, 29
- czech-language, 50
- danish-language, 50
- date\_time, 53
- DateTimeProvider, 51, 53
- DOIProvider, 29
- dutch-language, 54
- ElementProvider, 56
- ElementProvider\_en\_US, 11, 13, 15, 37, 43, 54, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- ElementProvider\_nl\_NL, 17, 54, 55, 87, 134, 167, 168, 185
- elements, 56
- english-language, 57
- farsi-language, 57
- FileProvider\_en\_US, 11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194
- finnish-language, 59
- fraudster, 59
- fraudster(), 7
- french-language, 60
- german-language, 61



- [greek-language](#), [61](#)
- [hebrew-language](#), [62](#)
- [hindi-language](#), [62](#)
- [hungarian-language](#), [63](#)
  
- [indonesian-language](#), [63](#)
- [InternetProvider](#), [10](#), [36](#), [39](#), [64](#), [79](#), [96](#),  
[109](#), [137](#), [188](#), [193](#)
- [InternetProvider\\_bg\\_BG](#), [24](#), [40](#), [68](#), [72](#),  
[110](#), [138](#)
- [InternetProvider\\_cs\\_CZ](#), [41](#), [50](#), [69](#), [112](#),  
[139](#)
- [InternetProvider\\_de\\_DE](#), [42](#), [61](#), [70](#), [114](#),  
[115](#), [141](#)
- [InternetProvider\\_en\\_AU](#), [11](#), [13](#), [15](#), [37](#), [43](#),  
[55](#), [57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [116–118](#),  
[144–148](#), [150](#), [184](#), [189](#), [194](#)
- [InternetProvider\\_en\\_NZ](#), [24](#), [40](#), [68](#), [72](#),  
[110](#), [138](#)
- [InternetProvider\\_en\\_US](#), [11](#), [13](#), [15](#), [37](#), [43](#),  
[55](#), [57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [116–118](#),  
[144–148](#), [150](#), [184](#), [189](#), [194](#)
- [InternetProvider\\_fa\\_IR](#), [45](#), [58](#), [74](#), [82](#),  
[122](#), [152](#)
- [InternetProvider\\_fr\\_FR](#), [46](#), [60](#), [75](#), [84](#), [85](#),  
[124](#), [125](#), [154](#), [155](#)
- [InternetProvider\\_hr\\_HR](#), [47](#), [50](#), [76](#), [86](#),  
[126](#), [158](#)
- [ISBNProvider](#), [77](#)
- [italian-language](#), [78](#)
  
- [japanese-language](#), [78](#)
- [JobProvider](#), [10](#), [32](#), [36](#), [39](#), [67](#), [79](#), [96](#), [109](#),  
[137](#), [188](#), [193](#)
- [JobProvider\\_da\\_DK](#), [50](#), [80](#), [113](#), [140](#), [142](#)
- [JobProvider\\_en\\_US](#), [11](#), [13](#), [15](#), [37](#), [43](#), [55](#),  
[57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [116–118](#),  
[144–148](#), [150](#), [184](#), [189](#), [194](#)
- [JobProvider\\_fa\\_IR](#), [45](#), [58](#), [74](#), [82](#), [122](#), [152](#)
- [JobProvider\\_fi\\_FI](#), [83](#), [123](#), [153](#)
- [JobProvider\\_fr\\_CH](#), [46](#), [60](#), [75](#), [84](#), [85](#), [124](#),  
[125](#), [154](#), [155](#)
- [JobProvider\\_fr\\_FR](#), [46](#), [60](#), [75](#), [84](#), [85](#), [124](#),  
[125](#), [154](#), [155](#)
- [JobProvider\\_hr\\_HR](#), [47](#), [50](#), [76](#), [86](#), [126](#), [158](#)
- [JobProvider\\_nl\\_NL](#), [17](#), [54](#), [56](#), [87](#), [134](#), [167](#),  
[168](#), [185](#)
- [JobProvider\\_pl\\_PL](#), [88](#), [136](#), [171](#), [180](#)
  
- [JobProvider\\_ru\\_RU](#), [89](#), [103](#), [174](#), [181](#)
- [JobProvider\\_uk\\_UA](#), [38](#), [90](#), [178](#), [191](#)
- [JobProvider\\_zh\\_TW](#), [26](#), [91](#), [104](#), [105](#), [179](#)
  
- [korean-language](#), [92](#)
  
- [latin-language](#), [92](#)
- [latvian-language](#), [93](#)
- [lithuanian-language](#), [93](#)
- [LoremProvider](#), [10](#), [36](#), [39](#), [67](#), [79](#), [94](#), [109](#),  
[137](#), [188](#), [193](#)
- [LoremProvider\\_ar\\_AA](#), [18](#), [96](#)
- [LoremProvider\\_el\\_GR](#), [61](#), [97](#), [143](#)
- [LoremProvider\\_en\\_US](#), [11](#), [13](#), [15](#), [37](#), [43](#), [55](#),  
[57](#), [58](#), [71](#), [73](#), [81](#), [98](#), [116–118](#),  
[144–148](#), [150](#), [184](#), [189](#), [194](#)
- [LoremProvider\\_he\\_IL](#), [62](#), [99](#), [156](#)
- [LoremProvider\\_ja\\_JP](#), [78](#), [100](#), [128](#), [162](#)
- [LoremProvider\\_la](#), [92](#), [101](#)
- [LoremProvider\\_ru\\_RU](#), [89](#), [102](#), [174](#), [181](#)
- [LoremProvider\\_zh\\_CN](#), [26](#), [91](#), [103](#), [105](#), [179](#)
- [LoremProvider\\_zh\\_TW](#), [26](#), [91](#), [104](#), [104](#), [179](#)
  
- [MissingDataProvider](#), [32](#)
  
- [norwegian-language](#), [105](#)
- [numerics](#), [106](#)
  
- [PersonProvider](#), [8](#), [10](#), [12](#), [13](#), [16](#), [36](#), [39](#), [64](#),  
[67](#), [79](#), [96](#), [107](#), [137](#), [188](#), [193](#)
- [PersonProvider\\_bg\\_BG](#), [24](#), [40](#), [68](#), [72](#), [110](#),  
[138](#)
- [PersonProvider\\_cs\\_CZ](#), [41](#), [50](#), [69](#), [111](#), [139](#)
- [PersonProvider\\_da\\_DK](#), [50](#), [80](#), [112](#), [140](#), [142](#)
- [PersonProvider\\_de\\_AT](#), [42](#), [61](#), [70](#), [113](#), [115](#),  
[141](#)
- [PersonProvider\\_de\\_DE](#), [42](#), [61](#), [70](#), [114](#), [114](#),  
[141](#)
- [PersonProvider\\_en\\_GB](#), [11](#), [13](#), [15](#), [37](#), [43](#),  
[55](#), [57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [115](#), [117](#),  
[118](#), [144–148](#), [150](#), [184](#), [189](#), [194](#)
- [PersonProvider\\_en\\_NZ](#), [11](#), [13](#), [15](#), [37](#), [43](#),  
[55](#), [57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [116](#), [116](#),  
[118](#), [144–148](#), [150](#), [184](#), [189](#), [194](#)
- [PersonProvider\\_en\\_US](#), [11](#), [13](#), [15](#), [33](#), [37](#),  
[43](#), [55](#), [57](#), [58](#), [71](#), [73](#), [81](#), [99](#), [116](#),  
[117](#), [117](#), [144–148](#), [150](#), [184](#), [189](#),  
[194](#)
- [PersonProvider\\_es\\_ES](#), [44](#), [119](#), [121](#), [149](#),  
[151](#), [182](#)

- PersonProvider\_es\_MX, *44, 119, 120, 149–151, 182*  
 PersonProvider\_fa\_IR, *45, 58, 74, 82, 121, 152*  
 PersonProvider\_fi\_FI, *83, 122, 153*  
 PersonProvider\_fr\_CH, *46, 60, 75, 84, 85, 123, 125, 154, 155*  
 PersonProvider\_fr\_FR, *46, 60, 75, 84, 85, 124, 124, 154, 155*  
 PersonProvider\_hr\_HR, *47, 50, 76, 86, 125, 158*  
 PersonProvider\_it\_IT, *48, 78, 126, 161*  
 PersonProvider\_ja\_JP, *78, 101, 127, 162*  
 PersonProvider\_ko\_KR, *92, 129, 163*  
 PersonProvider\_lt\_LT, *93, 130, 164*  
 PersonProvider\_lv\_LV, *93, 131, 165*  
 PersonProvider\_ne\_NP, *132, 166*  
 PersonProvider\_nl\_NL, *17, 54, 56, 87, 133, 167, 168, 185*  
 PersonProvider\_no\_NO, *105, 134, 169, 170*  
 PersonProvider\_pl\_PL, *88, 135, 171, 180*  
 PhoneNumberProvider, *10, 34, 36, 39, 67, 79, 96, 109, 136, 188, 193*  
 PhoneNumberProvider\_bg\_BG, *24, 40, 68, 72, 110, 137*  
 PhoneNumberProvider\_bs\_BA, *23, 138*  
 PhoneNumberProvider\_cs\_CZ, *41, 50, 69, 112, 139*  
 PhoneNumberProvider\_da\_DK, *50, 80, 113, 140, 142*  
 PhoneNumberProvider\_de\_DE, *42, 61, 70, 114, 115, 141*  
 PhoneNumberProvider\_dk\_DK, *80, 113, 140, 142*  
 PhoneNumberProvider\_el\_GR, *61, 98, 143*  
 PhoneNumberProvider\_en\_AU, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144, 145–148, 150, 184, 189, 194*  
 PhoneNumberProvider\_en\_CA, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144, 145, 146–148, 150, 184, 189, 194*  
 PhoneNumberProvider\_en\_GB, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144, 145, 146, 147, 148, 150, 184, 189, 194*  
 PhoneNumberProvider\_en\_NZ, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–146, 147, 148, 150, 184, 189, 194*  
 PhoneNumberProvider\_en\_US, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–147, 148, 150, 184, 189, 194*  
 PhoneNumberProvider\_es\_ES, *44, 119, 121, 149, 151, 182*  
 PhoneNumberProvider\_es\_MX, *11, 13, 15, 37, 43, 44, 55, 57, 58, 71, 73, 81, 99, 116–118, 121, 144–148, 150, 184, 189, 194*  
 PhoneNumberProvider\_es\_PE, *44, 119, 121, 149, 151, 182*  
 PhoneNumberProvider\_fa\_IR, *45, 58, 74, 82, 122, 152*  
 PhoneNumberProvider\_fi\_FI, *83, 123, 153*  
 PhoneNumberProvider\_fr\_CH, *46, 60, 75, 84, 85, 124, 125, 154, 155*  
 PhoneNumberProvider\_fr\_FR, *46, 60, 75, 84, 85, 124, 125, 154, 155*  
 PhoneNumberProvider\_he\_IL, *62, 100, 156*  
 PhoneNumberProvider\_hi\_IN, *62, 157*  
 PhoneNumberProvider\_hr\_HR, *47, 50, 76, 86, 126, 158*  
 PhoneNumberProvider\_hu\_HU, *63, 159*  
 PhoneNumberProvider\_id\_ID, *63, 160*  
 PhoneNumberProvider\_it\_IT, *48, 78, 127, 161*  
 PhoneNumberProvider\_ja\_JP, *78, 101, 128, 162*  
 PhoneNumberProvider\_ko\_KR, *92, 129, 163*  
 PhoneNumberProvider\_lt\_LT, *93, 130, 164*  
 PhoneNumberProvider\_lv\_LV, *93, 132, 165*  
 PhoneNumberProvider\_ne\_NP, *133, 166*  
 PhoneNumberProvider\_nl\_BE, *17, 54, 56, 87, 134, 167, 168, 185*  
 PhoneNumberProvider\_nl\_NL, *17, 54, 56, 87, 134, 167, 168, 185*  
 PhoneNumberProvider\_nn\_NO, *135, 169, 170*  
 PhoneNumberProvider\_no\_NO, *105, 135, 169, 170*  
 PhoneNumberProvider\_pl\_PL, *88, 136, 171, 180*  
 PhoneNumberProvider\_pt\_BR, *172, 173, 180*  
 PhoneNumberProvider\_pt\_PT, *172, 173, 180*  
 PhoneNumberProvider\_ru\_RU, *89, 103, 174,*

*181*  
PhoneNumberProvider\_sk\_SK, *175*  
PhoneNumberProvider\_sv\_SE, *175, 186*  
PhoneNumberProvider\_th\_TH, *176, 190*  
PhoneNumberProvider\_tr\_TR, *177, 190*  
PhoneNumberProvider\_uk\_UA, *38, 90, 178, 191*  
PhoneNumberProvider\_zh\_TW, *26, 91, 104, 105, 179*  
polish-language, *180*  
portuguese-language, *180*  
  
russian-language, *181*  
  
SequenceProvider, *31, 181*  
spanish-language, *182*  
SSNProvider, *34, 182*  
SSNProvider\_en\_US, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 183, 189, 194*  
SSNProvider\_nl\_NL, *17, 54, 56, 87, 134, 167, 168, 184*  
subclass, *185*  
swedish-language, *186*  
  
taxonomy, *186*  
TaxonomyProvider, *10, 36, 39, 67, 79, 96, 109, 137, 187, 187, 193*  
TaxonomyProvider\_en\_US, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 194*  
thai-language, *190*  
timezone, *52*  
turkish-language, *190*  
  
ukrainian-language, *191*  
UserAgentProvider, *10, 36, 39, 67, 79, 96, 109, 137, 188, 191*  
UserAgentProvider\_en\_US, *11, 13, 15, 37, 43, 55, 57, 58, 71, 73, 81, 99, 116–118, 144–148, 150, 184, 189, 193*