Package 'widr'

March 16, 2026

Title Interface to the World Inequality Database (WID)

Version 0.1.0

Description
Interface to the World Inequality Database (WID) API <https://wid.world>. Downloads distributional national accounts data with filters for country, year, percentile, age group, and population type. Includes code validation and reference tables. Independent implementation unaffiliated with the World Inequality Lab (WIL) or the Paris School of Economics.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports base64enc, jsonlite, digest, ggplot2, httr2, scales, tools, utils

Suggests testthat (>= 3.0.0), covr, knitr, rmarkdown, withr

VignetteBuilder knitr

URL https://github.com/cherylisabella/widr

BugReports https://github.com/cherylisabella/widr/issues

LazyData true

NeedsCompilation no

Author Cheryl Isabella Lim [aut, cre]

Maintainer Cheryl Isabella Lim <cheryl.academic@gmail.com>

Repository CRAN

Date/Publication 2026-03-16 16:40:02 UTC

Contents

`widr-package` *widr: Interface to the World Inequality Database (WID)*

Description

Interface to the World Inequality Database (WID) API `https://wid.world`. Downloads distributional national accounts data with filters for country, year, percentile, age group, and population type. Includes code validation and reference tables. Independent implementation unaffiliated with the World Inequality Lab (WIL) or the Paris School of Economics.

Author(s)

Maintainer: Cheryl Isabella Lim <cheryl.academic@gmail.com>

See Also

Useful links:

- https://github.com/cherylisabella/widr
- Report bugs at https://github.com/cherylisabella/widr/issues

---

`download_wid` *Download data from WID.world*

Description

Returns a `wid_df`, a classed `data.frame` that works natively with dplyr, ggplot2, and all base-R operations.

Usage

```
download_wid(
  indicators = "all",
  areas = "all",
  years = "all",
  perc = "all",
  ages = "992",
  pop = "j",
  metadata = FALSE,
  include_extrapolations = TRUE,
  verbose = FALSE,
  cache = TRUE
)

download(
  indicators = "all",
  areas = "all",
  years = "all",
  perc = "all",
  ages = "992",
  pop = "j",
  metadata = FALSE,
  include_extrapolations = TRUE,
  verbose = FALSE,
  cache = TRUE
)
```

Arguments

`indicators`	WID variable codes or `"all"`.
`areas`	ISO-2 area codes or `"all"`.

years	Integer vector or "all".
perc	Percentile codes or "all".
ages	Age codes or "all". Default "992".
pop	Population unit codes or "all". Default "j".
metadata	Logical. If TRUE, fetches metadata as a "wid_meta" attribute on the returned object. Default FALSE.
include_extrapolations	Logical. Default TRUE.
verbose	Logical. Default FALSE.
cache	Logical. Default TRUE.

Value

A wid_df (a classed data.frame) with columns country, variable, percentile, year, value, age, pop. If metadata = TRUE, a "wid_meta" attribute is attached.

Examples

```
if (nzchar(Sys.getenv("WID_API_KEY"))) {
  d <- download_wid(
    indicators = "sptinc992j",
    areas      = "US",
    perc       = "p99p100",
    years      = 2010:2020
  )
}
```

---

print.wid_df *Print a wid_df object*

Description

This method prints a summary of a wid_df object, including the number of rows, countries, and variables.

Usage

```
## S3 method for class 'wid_df'
print(x, ...)
```

Arguments

x	A wid_df object.
...	Additional arguments passed to print().

Value

Invisibly returns the input `x`.

Examples

```
d <- data.frame(country = "US", variable = "sptinc992j",
                percentile = "p99p100", year = "2020",
                value = 0.19, age = "992", pop = "j",
                stringsAsFactors = FALSE)
class(d) <- c("wid_df", "data.frame")
print(d)
```

`print.wid_query` *Print a wid_query object*

Description

Displays the contents of a `wid_query` object.

Usage

```
## S3 method for class 'wid_query'
print(x, ...)
```

Arguments

`x`	A `wid_query` object.
`...`	Additional arguments (currently ignored).

Value

Invisibly returns the input `x`.

Examples

```
q <- wid_query(indicators = "sptinc992j", areas = "US")
print(q)
```

wid_ages *Age group codes (\S2.1.3)*

Description

Zero-padded three-digit strings.

Usage

```
wid_ages
```

```
wid_ages
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

wid_cache *Interact with the WID response cache*

Description

Low-level access to get, set, list, or clear cached API responses.

Usage

```
wid_cache(action, key = NULL, value = NULL)
```

Arguments

`action`	One of `"get"`, `"set"`, `"list"`, `"clear"`.
`key`	Cache key string (required for `"get"` and `"set"`).
`value`	Value to store (`"set"` only).

Value

For `"get"`: the cached object or `NULL`. For `"list"`: character vector of cache keys. For `"set"` and `"clear"`: invisibly, the stored value or file count removed.

Examples

```
wid_cache("list")
```

---

wid_cache_clear *Clear all cached WID API responses*

Description

Clear all cached WID API responses

Usage

```
wid_cache_clear()
```

Value

Invisibly, the number of files removed.

Examples

```
wid_cache_clear()
```

---

wid_cache_list *List cached WID API responses*

Description

List cached WID API responses

Usage

```
wid_cache_list()
```

Value

Character vector of cache file names.

Examples

```
wid_cache_list()
```

wid_concepts *Concept codes (\S3-9)*

Description

Letters 2-6 of a WID variable code. Scraped from \S3-6, 8-9 tables; \S7 hardcoded from prose.

Usage

```
wid_concepts
```

```
wid_concepts
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

wid_convert *Convert a wid_df to another currency*

Description

Convert a wid_df to another currency

Usage

```
wid_convert(data, target = "usd", base_year = NULL)
```

Arguments

`data`	A `wid_df`.
`target`	One of `"lcu"`, `"usd"`, `"eur"`, `"gbp"`, `"ppp"`, `"yppp"`.
`base_year`	Pin to a specific year's exchange rate; `NULL` uses contemporaneous rates.

Value

The same `wid_df` with values converted. Dimensionless series (shares, ratios) are returned unchanged with a message.

Examples

```
if (nzchar(Sys.getenv("WID_API_KEY"))) {
  d <- download_wid("aptinc992j", areas = "US", perc = "p99p100",
                    years = 2010:2020)
  wid_convert(d, target = "ppp")
}
```

`wid_countries` *Country and region codes (\S2.2)*

Description

Country and region codes (\S2.2)

WID country and region codes

Usage

```
wid_countries
```

```
wid_countries
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

`wid_decode` *Parse a WID variable code into its components*

Description

Parse a WID variable code into its components

Usage

```
wid_decode(x, strict = TRUE)
```

Arguments

x	Character string, e.g. "sptinc992j".
strict	TRUE (default) stops on error; FALSE warns and returns NULL.

Value

Named list with elements series_type (character), concept (character), age (character or NULL), and pop (character or NULL).

Examples

```
wid_decode("sptinc992j")
wid_decode("mnninc")        # no age or pop
```

---

wid_encode *Build a WID variable code from its components*

Description

Build a WID variable code from its components

Usage

```
wid_encode(series_type, concept = NULL, age = NULL, pop = NULL)
```

Arguments

series_type	1-letter series type, or a list from wid_decode.
concept	5-6 letter concept code.
age	3-digit age code or NULL.
pop	1-letter population code or NULL.

Value

A character string of the form <type><concept>[age][pop].

Examples

```
wid_encode("s", "ptinc", "992", "j")   # "sptinc992j"
wid_encode("m", "nninc")               # "mnninc"
wid_encode(wid_decode("sptinc992j"))   # round-trip
```

wid_fetch *Execute a wid_query*

Description

Fetch data from WID.world using a `wid_query` object.

Usage

```
wid_fetch(query, ...)
```

Arguments

`query`	A `wid_query` object.
`...`	Parameter overrides.

Value

A `wid_df` (a classed `data.frame`); see `download_wid`.

Examples

```
if (nzchar(Sys.getenv("WID_API_KEY"))) {
  q <- wid_query(indicators = "sptinc992j", areas = "US", years = 2010:2020)
  wid_fetch(q)
}
```

wid_filter *Update a wid_query object*

Description

Modify fields of a `wid_query` object, returning an updated query.

Usage

```
wid_filter(query, ...)
```

Arguments

`query`	A `wid_query` object.
`...`	Fields to update.

Value

The updated `wid_query` object.

Examples

```
q <- wid_query(indicators = "sptinc992j", areas = "US")
q <- wid_filter(q, years = 2010:2020)
```

`wid_gini` *Compute Gini coefficient from a share series*

Description

Requires an `"s"` (share) series with `pXpY` percentile codes covering the full distribution.

Usage

```
wid_gini(data, variable = NULL, country = NULL)
```

Arguments

`data`	A `wid_df` with share series.
`variable`	Variable code filter.
`country`	Country code filter.

Value

A `data.frame` with columns `country`, `year`, and `gini` (numeric, 0-1).

Examples

```
d <- data.frame(
  country = rep("US", 3L), year = rep("2020", 3L),
  variable = rep("sptinc992j", 3L),
  percentile = c("p0p50", "p50p90", "p90p100"),
  value = c(0.20, 0.40, 0.40), age = "992", pop = "j",
  stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_gini(d)
```

wid_is_valid *Test WID code components without throwing*

Description

Test WID code components without throwing

Usage

```
wid_is_valid(...)
```

Arguments

`...`	Arguments passed to `wid_validate`.

Value

Scalar logical: `TRUE` if all supplied components are valid, `FALSE` if any would cause an error. Warning-level issues return `TRUE`.

Examples

```
wid_is_valid(series_type = "s", concept = "ptinc")  # TRUE
wid_is_valid(series_type = "Z")                      # FALSE
```

wid_metadata *Fetch metadata for variables in a wid_df*

Description

Fetch metadata for variables in a wid_df

Usage

```
wid_metadata(data)
```

Arguments

`data`	A `wid_df`.

Value

A `data.frame` with columns `variable`, `country`, `source`, `method`, `quality`, and `imputation`.

Examples

```
if (nzchar(Sys.getenv("WID_API_KEY"))) {
  d <- download_wid("sptinc992j", areas = "US", metadata = TRUE)
  wid_metadata(d)
}
```

`wid_percentiles` *Percentile group codes (\S2.3)*

Description

Enumerated from the pXpY grammar: key groups, deciles, centiles, and top-1\

Usage

```
wid_percentiles

wid_percentiles
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

`wid_percentile_ratio` *Percentile threshold ratio (e.g. P90/P10)*

Description

Percentile threshold ratio (e.g. P90/P10)

Usage

```
wid_percentile_ratio(
  data,
  numerator = "p90",
  denominator = "p10",
  variable = NULL,
  country = NULL
)
```

Arguments

data	A wid_df with threshold ("t") series.
numerator	Upper percentile code, e.g. "p90".
denominator	Lower percentile code, e.g. "p10".
variable	Variable code filter.
country	Country code filter.

Value

A data.frame with columns country, year, and ratio (numeric: numerator threshold divided by denominator threshold).

Examples

```
d <- data.frame(
  country = rep("US", 2L), year = rep("2020", 2L),
  variable = rep("tptinc992j", 2L),
  percentile = c("p90", "p10"), value = c(200000, 10000),
  age = "992", pop = "j", stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_percentile_ratio(d)
```

---

wid_plot_compare *Cross-country comparison bar/point chart*

Description

Cross-country comparison bar/point chart

Usage

```
wid_plot_compare(
  data,
  year = NULL,
  variable = NULL,
  country = NULL,
  country_labels = NULL,
  type = "bar"
)
```

Arguments

data	A wid_df.
year	Integer year (defaults to most recent common year).
variable	Variable code to plot.
country	Country codes to include.
country_labels	Named character vector for renaming countries.
type	"bar" (default) or "point".

Value

A ggplot object.

Examples

```
d <- data.frame(
  country = rep(c("US", "FR"), each = 3L),
  year    = rep(c("2018", "2019", "2020"), 2L),
  variable = "sptinc992j", percentile = "p99p100",
  value = c(0.20, 0.19, 0.19, 0.11, 0.11, 0.12),
  age = "992", pop = "j", stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_plot_compare(d, year = 2020)
```

---

wid_plot_lorenz *Lorenz curve*

Description

Lorenz curve

Usage

```
wid_plot_lorenz(data, variable = NULL, country = NULL, country_labels = NULL)
```

Arguments

data	A wid_df with share ("s") series.
variable	Variable code to plot.
country	Country codes to include.
country_labels	Named character vector for renaming countries.

Value

A ggplot object.

Examples

```
d <- data.frame(
  country = rep("US", 3L), year = rep("2020", 3L),
  variable = "sptinc992j",
  percentile = c("p0p50", "p50p90", "p90p100"),
  value = c(0.20, 0.40, 0.40), age = "992", pop = "j",
  stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_plot_lorenz(d)
```

`wid_plot_timeseries` *Time-series line chart for WID data*

Description

Time-series line chart for WID data

Usage

```
wid_plot_timeseries(
  data,
  variable = NULL,
  country = NULL,
  country_labels = NULL,
  facet = FALSE
)
```

Arguments

`data`	A `wid_df`.
`variable`	Variable code to plot.
`country`	Country codes to include.
`country_labels`	Named character vector for renaming countries.
`facet`	Logical. Facet by country. Default `FALSE`.

Value

A `ggplot` object.

Examples

```
d <- data.frame(
  country = rep(c("US", "FR"), each = 3L),
  year    = rep(c("2018", "2019", "2020"), 2L),
  variable = "sptinc992j", percentile = "p99p100",
  value = c(0.20, 0.19, 0.19, 0.11, 0.11, 0.12),
  age = "992", pop = "j", stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_plot_timeseries(d)
```

`wid_pop_types` *Population unit codes (\S2.1.4)*

Description

Population unit codes (\S2.1.4)

WID population unit codes

Usage

```
wid_pop_types
```

```
wid_pop_types
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

wid_query *Build a reusable WID query object*

Description

Creates a `wid_query` object that stores query parameters for WID data fetching.

Usage

```
wid_query(...)
```

Arguments

`...`	Named arguments matching `download_wid()` parameters.

Value

A `wid_query` object (a named list with class `"wid_query"`).

Examples

```
q <- wid_query(indicators = "sptinc992j", areas = "US", years = 2010:2020)
```

wid_search *Search WID lookup tables by regex*

Description

Search WID lookup tables by regex

Usage

```
wid_search(query, tables = "concepts", type = NULL)
```

Arguments

`query`	Regular expression.
`tables`	Table names to search. Default `"concepts"`.
`type`	Optional series-type letter to filter results.

Value

A `data.frame` with columns `table`, `variable`, and `description` for each matching row, or an invisible empty `data.frame` if no matches are found.

Examples

```
wid_search("national income")
wid_search("wealth", type = "s")
wid_search("^US", tables = "countries")
```

`wid_series_types` *Series type codes (\S2.1.1)*

Description

Series type codes (\S2.1.1)

WID series type codes

Usage

```
wid_series_types

wid_series_types
```

Format

data.frame(code, description)

A data frame with columns `code` and `description`.

Source

https://wid.world/codes-dictionary/

https://wid.world/codes-dictionary/

`wid_set_key` *Set WID API key*

Description

Set WID API key

Usage

```
wid_set_key(key)
```

Arguments

`key`	character API key

Value

Invisibly, the key string.

wid_tidy *Tidy a wid_df: decode variable codes and coerce column types*

Description

Tidy a wid_df: decode variable codes and coerce column types

Usage

```
wid_tidy(data, decode = TRUE, country_names = TRUE)
```

Arguments

data	A wid_df.
decode	Logical. Add decoded columns. Default TRUE.
country_names	Logical. Join country display names. Default TRUE.

Value

A plain data.frame with year as integer, value as double, and optionally indicator, series_type, type_label, and country_name columns appended.

Examples

```
d <- data.frame(
  country = "US", variable = "sptinc992j", percentile = "p99p100",
  year = "2020", value = "0.19", age = "992", pop = "j",
  stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_tidy(d, country_names = FALSE)
```

wid_top_share *Top income or wealth share*

Description

Top income or wealth share

Usage

```
wid_top_share(data, top = 0.1, variable = NULL, country = NULL)
```

Arguments

data	A wid_df with share series.
top	Top fraction, e.g. 0.1 = top 10\%.
variable	Variable code filter.
country	Country code filter.

Value

A data.frame with columns country, year, top (the requested fractile), and share (numeric).

Examples

```
d <- data.frame(
  country = "US", year = "2020", variable = "sptinc992j",
  percentile = "p99p100", value = 0.19, age = "992", pop = "j",
  stringsAsFactors = FALSE
)
class(d) <- c("wid_df", "data.frame")
wid_top_share(d, top = 0.01)
```

---

wid_validate *Validate WID code components*

Description

Validate WID code components

Usage

```
wid_validate(
  series_type = NULL,
  concept = NULL,
  age = NULL,
  pop = NULL,
  years = NULL,
  areas = NULL,
  perc = NULL
)
```

Arguments

series_type	One-letter series type code.
concept	5-6 letter concept code.
age	Three-digit age code.
pop	One-letter population unit code.
years	Integer vector of years.
areas	ISO-2 area codes.
perc	Percentile codes of the form pXpY.

Value

Invisibly, a named list of normalised inputs with the same names as the parameters, with `age` zero-padded and `years` coerced to integer.

Examples

```
wid_validate(series_type = "s", concept = "ptinc", age = 992, pop = "j")
wid_validate(areas = c("US", "FR", "US-CA"))
```

Index