

Package ‘psgc’

July 25, 2026

Type Package

Title Philippine Standard Geographic Code

Version 0.1.2

Description Provides access to the Philippine Standard Geographic Code (PSGC), an official classification system for geographic areas in the Philippines published by the Philippine Statistics Authority (PSA). Includes area names, geographic levels (Region, Province, City, Municipality, Sub-Municipality, and Barangay), and census population figures across multiple PSA publication releases. Offers utilities to look up individual codes, filter by geographic level, track code changes across releases via a built-in crosswalk, and retrieve population data in long or wide format.

Maintainer Bhas Abdulsamad <aeabdulsamad@gmail.com>

License MIT + file LICENSE

Encoding UTF-8

Imports cli

Suggests jsonlite, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Depends R (>= 3.5)

RoxygenNote 7.3.3

URL <https://yng-me.github.io/psgc/>

NeedsCompilation no

Author Bhas Abdulsamad [aut, cre, cph] (ORCID:
<https://orcid.org/0009-0002-5891-8124>)

Repository CRAN

Date/Publication 2026-07-25 06:40:02 UTC

Contents

get_population	2
get_psgc	3
get_psgc_wide	4
latest_release	5
list_releases	5
map_psgc	6
psgc_info	6
Index	8

get_population	<i>Get population data for a specific release</i>
----------------	---

Description

Returns the census population figures bundled with a PSGC release.

Usage

```
get_population(  
  release = latest_release(),  
  details = FALSE,  
  geographic_level = NULL,  
  wide = FALSE  
)
```

Arguments

release	A release name from [list_releases()]. Defaults to [latest_release()].
details	Logical. If 'TRUE', adds 'area_name' and 'geographic_level' columns from the PSGC release data. Defaults to 'FALSE'.
geographic_level	A character vector of geographic levels to filter by. Accepts the same canonical codes and aliases as [get_psgc()] (e.g. "Reg", "Region", "Prov", "city_mun", "Barangay", etc.). 'NULL' (default) returns all levels. Implies 'details = TRUE' internally to resolve the filter; the column is only included in the result when 'details = TRUE' is also requested.
wide	Logical. If 'TRUE', pivots census years to columns named 'population_<year>' (e.g. 'population_2015', 'population_2020', 'population_2024'), yielding one row per PSGC code. Defaults to 'FALSE'.

Value

A data frame. In long format (default): columns 'psgc_code', 'population', 'year', plus optionally 'area_name' and 'geographic_level'. In wide format: columns 'psgc_code', 'population_<year>' per census year, plus optionally 'area_name' and 'geographic_level'.

Examples

```
head(get_population())
get_population("Q1_2023")
get_population(details = TRUE)
get_population(geographic_level = "Reg")
get_population(geographic_level = "Region", wide = TRUE)
get_population(geographic_level = "Reg", wide = TRUE, details = TRUE)
```

get_psgc

Get PSGC data for a specific release

Description

Get PSGC data for a specific release

Usage

```
get_psgc(
  release = latest_release(),
  geographic_level = NULL,
  include_population_data = FALSE
)
```

Arguments

release	A release name from [list_releases()]. Defaults to [latest_release()].
geographic_level	A character vector of geographic levels to filter by. Accepts canonical codes ("Reg", "Prov", "City", "Mun", "SubMun", "Bgy") as well as common aliases such as "Region", "Province", "Municipality", "Barangay", "Sub-Municipality", etc. Use "city_mun" (or aliases like "City-Municipality") to include both cities and municipalities. 'NULL' (default) returns all levels.
include_population_data	Logical. If 'TRUE', census population figures are joined onto the result, adding 'population' (integer) and 'year' columns. Each geographic unit produces one row per available census year. Defaults to 'FALSE'.

Value

A data frame of PSGC entries for the given release, optionally filtered to the requested geographic level(s) and/or enriched with population data.

Examples

```
head(get_psgc())
get_psgc("Q1_2023")
get_psgc(geographic_level = "Reg")
get_psgc(geographic_level = "Region")
get_psgc(geographic_level = "city_mun")
get_psgc(geographic_level = c("Prov", "City"))
get_psgc(geographic_level = "Reg", include_population_data = TRUE)
```

get_psgc_wide

Get PSGC data in wide (denormalised) format

Description

Returns a data frame with one row per barangay and all four geographic levels—region, province, city/municipality, and barangay—spread into separate columns. Highly Urbanised Cities (HUCs) and Independent Component Cities (ICCs) are included as pseudo-province entries so their barangays have a ‘province’ value. Barangays in areas with no province layer in the PSGC (e.g. Pateros in NCR, City of Isabela, Special Geographic Areas) will have ‘NA’ for ‘province’ and ‘province_code’.

Usage

```
get_psgc_wide(release = latest_release())
```

Arguments

release A release name from [list_releases()]. Defaults to [latest_release()].

Value

A data frame with one row per barangay and the following columns, in order:

‘**area_code**’ 10-digit PSGC code of the barangay.

‘**region_code**’ 10-digit PSGC code of the region.

‘**province_code**’ 10-digit PSGC code of the province (or of the HUC/ICC acting as province).

‘**city_mun_code**’ 10-digit PSGC code of the city, municipality, or sub-municipality. ‘NA’ for barangays that sit directly under an HUC with no intervening city/municipality layer.

‘**region**’ Region name.

‘**province**’ Province name (or HUC/ICC name for province-free cities). ‘NA’ for areas with no province layer (e.g. Pateros in NCR, City of Isabela, Special Geographic Areas).

‘**city_mun**’ City / municipality / sub-municipality name. ‘NA’ for HUC barangays.

‘**barangay**’ Barangay name.

‘**urban_rural**’ Urban/rural classification of the barangay.

‘**island_region**’ Island group of the barangay.

Examples

```
head(get_psgc_wide())  
head(get_psgc_wide("Q1_2023"))
```

latest_release	<i>The most recent bundled PSGC release</i>
----------------	---

Description

The most recent bundled PSGC release

Usage

```
latest_release()
```

Value

A single character string naming the latest available release.

Examples

```
latest_release()
```

list_releases	<i>List available PSGC releases</i>
---------------	-------------------------------------

Description

List available PSGC releases

Usage

```
list_releases()
```

Value

A character vector of release names in chronological order.

Examples

```
list_releases()
```

map_psgc	<i>Map PSGC codes to a target release</i>
----------	---

Description

Map PSGC codes to a target release

Usage

```
map_psgc(code, from = "auto", to = latest_release(), changes_only = FALSE)
```

Arguments

code	A character vector of 10-digit PSGC codes.
from	Release the codes come from, or "auto" (default) to detect automatically using the earliest release that contains each code.
to	Target release name. Defaults to [latest_release()].
changes_only	Logical. If 'TRUE', only rows where the code actually changed (i.e. 'mapping_type' is not "direct") are returned. Codes that remained unchanged across all hops are dropped. Defaults to 'FALSE'.

Value

A data frame with columns 'old_code', 'new_code' ('NA' for abolished codes), 'mapping_type' ("direct", "renumbered", "split", "merged", or "abolished"), 'from_release', and 'to_release'. Split codes produce multiple rows.

Examples

```
map_psgc("0100000000")
map_psgc(c("0100000000", "0102800000"), to = "Q4_2023")
map_psgc(get_psgc(geographic_level = "Bgy")$psgc_code,
  from = "Q1_2023", to = "Q4_2023", changes_only = TRUE)
```

psgc_info	<i>Get metadata for one or more PSGC codes</i>
-----------	--

Description

Get metadata for one or more PSGC codes

Usage

```
psgc_info(code, release = latest_release())
```

Arguments

<code>code</code>	A character vector of 10-digit PSGC codes.
<code>release</code>	A release name from <code>[list_releases()]</code> . Defaults to <code>[latest_release()]</code> .

Value

A data frame with one row per code containing metadata columns ('area_name', 'geographic_level', 'correspondence_code', etc.) plus a 'release' column indicating which release was used.

Examples

```
psgc_info("0100000000")
psgc_info(c("0100000000", "0102800000"))
psgc_info("0100000000", release = "Q1_2023")
```

Index

`get_population`, [2](#)

`get_psgc`, [3](#)

`get_psgc_wide`, [4](#)

`latest_release`, [5](#)

`list_releases`, [5](#)

`map_psgc`, [6](#)

`psgc_info`, [6](#)