

Package: arcgeocoder (via r-universe)

October 27, 2026

Title Address and Coordinate Search with the 'ArcGIS' REST API

Version 0.4.1

Description Provides a lightweight interface to the 'ArcGIS' REST API for converting addresses and structured address components into geographic coordinates, finding places by category and converting coordinates into addresses. It uses the ArcGIS service documented at <https://developers.arcgis.com/rest/geocode/api-reference/overview-world-geocoding-service.htm>. No API key is required.

License MIT + file LICENSE

URL <https://dieghernan.github.io/arcgeocoder/>,
<https://github.com/dieghernan/arcgeocoder>

BugReports <https://github.com/dieghernan/arcgeocoder/issues>

Depends R (>= 4.1.0)

Imports dplyr (>= 1.0.0), jsonlite (>= 1.7.0), utils

Suggests curl, ggplot2, knitr, quarto, sf, testthat (>= 3.2.0),
tibble, withr (>= 3.0.0)

VignetteBuilder quarto

Config/Needs/website dieghernan/gitdevr, leaflet, terra, tidyterra,
maptiles, styler, mapSpain, remotes, reactable, crosstalk,
sessioninfo, tidygeocoder, nominatimlite, pak

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.1.1

Config/testthat/edition 3

Config/testthat/parallel false

Encoding UTF-8

LazyData true

X-schema.org-applicationCategory cartography

X-schema.org-keywords r, arcgis, address, coordinates, location-search, rstats, r-package, api-wrapper, api-rest, arcgis-api, cran-r, gis, cran, spatial-search, geocoder, geocoding, reverse-geocoder, reverse-geocoding

Repository <https://dieghernan.r-universe.dev>

Date/Publication 2026-10-27 13:02:12 UTC

RemoteUrl <https://github.com/dieghernan/arcgeocoder>

RemoteRef HEAD

RemoteSha 6a45d993ebebe75947e370e71c265a9363fb9c33

Contents

arc_categories	2
arc_geo	4
arc_geo_categories	6
arc_geo_multi	9
arc_reverse_geo	12
arc_spatial_references	15
Index	17

arc_categories	<i>Place categories supported by the ArcGIS REST API</i>
----------------	--

Description

A dataset of categories that can be used to filter results from [arc_geo\(\)](#), [arc_geo_multi\(\)](#) and [arc_geo_categories\(\)](#).

Format

A [tibble](#) with 383 rows and three variables:

- level_1** Top-level category.
- level_2** Second-level category.
- level_3** Child-level category.

Details

See [ArcGIS REST API category filtering](#) for details and examples.

The ArcGIS geocoding service supports searches for many types of addresses and places around the world, so applications do not need to anticipate the types of places that users may search for. However, ambiguous searches can match many different places and produce unexpected results. For example, a search for a city may match a street name, or an airport code may match a country abbreviation.

In these cases, the `category` argument can filter out unwanted results. It limits the types of places that the service searches for, which can eliminate false-positive matches and speed up the search.

The dataset lists categories at three hierarchy levels (`level_1`, `level_2` and `level_3`). If a `level_1` category is requested (for example, `POI`), its child categories may also be included in the results.

Note

Data extracted on **15 January 2026**.

Source

[ArcGIS REST API category filtering](#).

See Also

[arc_geo\(\)](#), [arc_geo_multi\(\)](#) and [arc_geo_categories\(\)](#).

Examples

```
# Get all possible values.
data("arc_categories")
arc_categories

# Use categories.

sea_1 <- arc_geo("sea",
  custom_query = list(outFields = c("LongLabel", "Type")),
  limit = 2
)

dplyr::glimpse(sea_1)

# Categories can disambiguate the result.

sea_2 <- arc_geo("sea",
  custom_query = list(outFields = c("LongLabel", "Type")),
  limit = 2, category = "Food"
)

dplyr::glimpse(sea_2)

# Use multiple categories.
sea_3 <- arc_geo("sea",
  custom_query = list(outFields = c("LongLabel", "Type")),
  sourcecountry = "UK", limit = 5,
  category = c("Amusement Park", "Aquarium")
)

dplyr::glimpse(sea_3)
```

arc_geo

*Geocode addresses with the ArcGIS REST API***Description**

Converts single-line addresses into geographic coordinates and returns one or more matches for each query.

This function uses the SingleLine approach detailed in the [ArcGIS REST API documentation](#). For structured queries that use specific address components, use [arc_geo_multi\(\)](#).

Usage

```
arc_geo(
  address,
  lat = "lat",
  long = "lon",
  limit = 1,
  full_results = FALSE,
  return_addresses = TRUE,
  verbose = FALSE,
  progressbar = TRUE,
  outsr = NULL,
  langcode = NULL,
  sourcecountry = NULL,
  category = NULL,
  custom_query = list()
)
```

Arguments

address	Single-line address text (for example, "1600 Pennsylvania Ave NW, Washington") or a vector of addresses (for example, c("Madrid", "Barcelona")).
lat	Name of the latitude or y-coordinate column in the output. The default is "lat".
long	Name of the longitude or x-coordinate column in the output. The default is "lon".
limit	Maximum number of results to return per input address. Each query has a hard API limit of 50 results.
full_results	A logical value indicating whether to return all available API fields via outFields = "*". The default is FALSE.
return_addresses	A logical value indicating whether to include the input query in the output.
verbose	A logical value indicating whether to display API request details.
progressbar	A logical value indicating whether to display a progress bar for multiple queries.
outsr	Spatial reference of the output coordinates. The default is NULL, which uses the service default. See Details and arc_spatial_references .

langcode	Language of the returned addresses.
sourcecountry	Country filter using ISO codes (for example, "USA"). Multiple values can be supplied as a comma-separated string.
category	Place or address type used to filter results. Multiple values are accepted (for example, c("Cinema", "Museum")). See arc_categories .
custom_query	A named list with additional API parameters.

Details

See the [ArcGIS REST API documentation](#) for more information and valid values.

Value

A [tibble](#) with one or more matches for each query. For descriptions of the available fields, see [ArcGIS REST API service output](#).

outsr

The spatial reference can be specified as a well-known ID (WKID). If not specified, the spatial reference of the output locations is the same as that of the service (WGS 84, that is, WKID 4326).

See [arc_spatial_references](#) for values and examples.

References

[ArcGIS REST API findAddressCandidates](#).

See Also

Geocoding and reverse geocoding functions: [arc_geo_categories\(\)](#), [arc_geo_multi\(\)](#), [arc_reverse_geo\(\)](#)

Examples

```
arc_geo("Madrid, Spain")

library(dplyr)

# Several addresses with additional output fields.
with_params <- arc_geo(c("Madrid", "Barcelona"),
  custom_query = list(outFields = c("LongLabel", "CntryName"))
)

with_params |>
  select(lat, lon, CntryName, LongLabel)

# Restrict the search to the USA.
with_params_usa <- arc_geo(c("Madrid", "Barcelona"),
  sourcecountry = "USA",
  custom_query = list(outFields = c("LongLabel", "CntryName"))
)
```

```
with_params_usa |>
  select(lat, lon, CntryName, LongLabel)
```

arc_geo_categories	<i>Geocode places by category in an area</i>
--------------------	--

Description

Finds places that match one or more categories near a location or within a bounding box.

See [arc_categories](#) for a detailed explanation and available values.

To obtain results, provide either a pair of coordinates, x and y, as a search origin or a bounding box in bbox to define the search extent.

You can combine both approaches by providing x, y and bbox. See **Examples**.

Usage

```
arc_geo_categories(
  category,
  x = NULL,
  y = NULL,
  bbox = NULL,
  name = NULL,
  lat = "lat",
  long = "lon",
  limit = 1,
  full_results = FALSE,
  verbose = FALSE,
  custom_query = list(),
  ...
)
```

Arguments

category	A place or address type used to filter results. Multiple values can be supplied as a vector (for example, c("Cinema", "Museum")), which performs one call for each value. See Details .
x	A numeric vector of longitude values in the range $[-180, 180]$.
y	A numeric vector of latitude values in the range $[-90, 90]$.
bbox	A numeric vector specifying a bounding box used to limit the search. It must contain longitude (x) and latitude (y) values in the form c(xmin, ymin, xmax, ymax). See Details .
name	An optional string containing the name or address to match.

lat	Name of the latitude or y-coordinate column in the output. The default is "lat".
long	Name of the longitude or x-coordinate column in the output. The default is "lon".
limit	Maximum number of results per query. The ArcGIS REST API limits a single request to 50 results.
full_results	A logical value indicating whether to return all available API fields via outFields = "*". The default is FALSE.
verbose	A logical value indicating whether to display API request details.
custom_query	A named list with additional API parameters.
...	Arguments passed on to arc_geo
sourcecountry	Country filter using ISO codes (for example, "USA"). Multiple values can be supplied as a comma-separated string.
outsr	Spatial reference of the output coordinates. The default is NULL, which uses the service default. See Details and arc_spatial_references .
langcode	Language of the returned addresses.

Details

Bounding boxes can be located using online tools, such as [Bounding Box Tool](#).

For a full list of valid categories, see [arc_categories](#). This function is vectorized over category, which means it performs one independent call to [arc_geo\(\)](#) for each category value.

`arc_geo_categories()` also understands a single string of categories separated by commas ("Cinema,Museum"), which is treated internally as `c("Cinema", "Museum")`.

Value

A [tibble](#) with one or more matches for each query. For descriptions of the available fields, see [ArcGIS REST API service output](#).

outsr

The spatial reference can be specified as a well-known ID (WKID). If not specified, the spatial reference of the output locations is the same as that of the service (WGS 84, that is, WKID 4326).

See [arc_spatial_references](#) for values and examples.

See Also

- [arc_categories](#) lists supported values.
- [ArcGIS REST API category filtering](#) provides API details.

Geocoding and reverse geocoding functions: [arc_geo\(\)](#), [arc_geo_multi\(\)](#), [arc_reverse_geo\(\)](#)

Examples

```
# Full workflow: gas stations near Carabanchel, Madrid.

# Geocode Carabanchel.
carab <- arc_geo("Carabanchel, Madrid, Spain")

# Extract the CRS.
carab_crs <- unique(carab$latestWkid)

library(ggplot2)

base_map <- ggplot(carab) +
  geom_point(aes(lon, lat), size = 5, color = "red") +
  geom_rect(aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax),
    fill = NA,
    color = "blue"
  ) +
  coord_sf(crs = carab_crs)

# Example 1: Search near Carabanchel (not restricted).
ex1 <- arc_geo_categories("Gas Station",
  # Use Carabanchel as the search origin.
  x = carab$lon, y = carab$lat,
  limit = 50, full_results = TRUE
)

# Reduce labels to the most common ones.
library(dplyr)

labs <- ex1 |>
  count(ShortLabel) |>
  slice_max(n = 7, order_by = n) |>
  pull(ShortLabel)

base_map +
  geom_point(data = ex1, aes(lon, lat, color = ShortLabel)) +
  scale_color_discrete(breaks = labs) +
  labs(
    title = "Example 1",
    subtitle = "Search near (points may be far away)"
  )

# Example 2: Include part of the name for different results.
ex2 <- arc_geo_categories("Gas Station",
  # Match this name.
  name = "Repsol",
  # Use Carabanchel as the search origin.
  x = carab$lon, y = carab$lat,
  limit = 50, full_results = TRUE
)
```

```
base_map +
  geom_point(data = ex2, aes(lon, lat, color = ShortLabel)) +
  labs(
    title = "Example 2",
    subtitle = "Search near with name"
  )

# Example 3: Search within a bounding box.
ex3 <- arc_geo_categories("Gas Station",
  name = "Repsol",
  bbox = c(carab$xmin, carab$ymin, carab$xmax, carab$ymax),
  limit = 50, full_results = TRUE
)

base_map +
  geom_point(data = ex3, aes(lon, lat, color = ShortLabel)) +
  labs(
    title = "Example 3",
    subtitle = "Search near with name and bounding box"
  )
```

arc_geo_multi*Geocode addresses with a structured ArcGIS REST API query*

Description

Converts structured address components into geographic coordinates and returns one or more matches for each query.

For a single-line address, use [arc_geo\(\)](#).

Usage

```
arc_geo_multi(
  address = NULL,
  address2 = NULL,
  address3 = NULL,
  neighborhood = NULL,
  city = NULL,
  subregion = NULL,
  region = NULL,
  postal = NULL,
  postalex = NULL,
  countrycode = NULL,
  lat = "lat",
  long = "lon",
  limit = 1,
```

```

    full_results = FALSE,
    return_addresses = TRUE,
    verbose = FALSE,
    progressbar = TRUE,
    outsr = NULL,
    langcode = NULL,
    category = NULL,
    custom_query = list()
)

```

Arguments

address, address2, address3, neighborhood, city, subregion	Address components. See Details .
region, postal, postalext, countrycode	Additional address components. See Details .
lat	Name of the latitude or y-coordinate column in the output. The default is "lat".
long	Name of the longitude or x-coordinate column in the output. The default is "lon".
limit	Maximum number of results to return per input address. Each query has a hard API limit of 50 results.
full_results	A logical value indicating whether to return all available API fields via outFields = "*". The default is FALSE.
return_addresses	A logical value indicating whether to include the input query in the output.
verbose	A logical value indicating whether to display API request details.
progressbar	A logical value indicating whether to display a progress bar for multiple queries.
outsr	Spatial reference of the output coordinates. The default is NULL, which uses the service default. See Details and arc_spatial_references .
langcode	Language of the returned addresses.
category	Place or address type used to filter results. Multiple values are accepted (for example, c("Cinema", "Museum")). See arc_categories .
custom_query	A named list with additional API parameters.

Details

See the [ArcGIS REST API documentation](#) for more information and valid values.

Value

A [tibble](#) with one or more matches for each query. For descriptions of the available fields, see [ArcGIS REST API service output](#).

The output also includes the input arguments as columns prefixed with q_ to help track the results.

Address components

This function performs structured queries using separate address components. At least one component must not be NA or NULL.

Each argument can be a vector to geocode multiple addresses. Vectors supplied to different arguments must have the same length.

The following list provides a brief description of each argument:

- **address:** A string representing the first line of a street address. It usually contains the street name and house number, but can also contain a building or place name.
- **address2:** A string that represents the second line of a street address. It can include a street name and house number, building name, place name or subunit.
- **address3:** A string that represents the third line of a street address. It can include a street name and house number, building name, place name or subunit.
- **neighborhood:** The smallest administrative division associated with an address, typically a neighborhood or a section of a larger populated place.
- **city:** The next largest administrative division associated with an address, typically a city or municipality.
- **subregion:** The next largest administrative division associated with an address. Depending on the country, a subregion can represent a county, state or province.
- **region:** The largest administrative division associated with an address, typically a state or province.
- **postal:** The standard postal code for an address, typically a three- to six-character alphanumeric code.
- **postalext:** A postal code extension, such as the United States Postal Service ZIP+4 code.
- **countrycode:** A value representing the country. Providing this value can increase search speed. Acceptable values include the full country name in English or the official language of the country, the two-character country code or the three-character country code.

outsr

The spatial reference can be specified as a well-known ID (WKID). If not specified, the spatial reference of the output locations is the same as that of the service (WGS 84, that is, WKID 4326).

See [arc_spatial_references](#) for values and examples.

References

[ArcGIS REST API findAddressCandidates.](#)

See Also

Geocoding and reverse geocoding functions: [arc_geo\(\)](#), [arc_geo_categories\(\)](#), [arc_reverse_geo\(\)](#)

Examples

```
simple <- arc_geo_multi(  
  address = "Plaza Mayor", limit = 10,  
  custom_query = list(outFields = c("LongLabel", "CntryName", "Region"))  
)  
  
library(dplyr)  
  
simple |>  
  select(lat, lon, CntryName, Region, LongLabel) |>  
  slice_head(n = 10)  
  
# Restrict search to Spain.  
simple2 <- arc_geo_multi(  
  address = "Plaza Mayor", countrycode = "ESP",  
  limit = 10,  
  custom_query = list(outFields = c("LongLabel", "CntryName", "Region"))  
)  
  
simple2 |>  
  select(lat, lon, CntryName, Region, LongLabel) |>  
  slice_head(n = 10)  
  
# Restrict to a region.  
simple3 <- arc_geo_multi(  
  address = "Plaza Mayor", region = "Segovia",  
  countrycode = "ESP",  
  limit = 10,  
  custom_query = list(outFields = c("LongLabel", "CntryName", "Region"))  
)  
  
simple3 |>  
  select(lat, lon, CntryName, Region, LongLabel) |>  
  slice_head(n = 10)
```

arc_reverse_geo

Reverse geocode coordinates with the ArcGIS REST API

Description

Converts longitude and latitude values into addresses. Latitudes must be in the range $[-90, 90]$ and longitudes in the range $[-180, 180]$. Returns one match for each coordinate pair.

Usage

```
arc_reverse_geo(  
  address = "Plaza Mayor", limit = 10,  
  custom_query = list(outFields = c("LongLabel", "CntryName", "Region"))  
)
```

```

x,
y,
address = "address",
full_results = FALSE,
return_coords = TRUE,
verbose = FALSE,
progressbar = TRUE,
outsr = NULL,
langcode = NULL,
featuretypes = NULL,
locationtype = NULL,
custom_query = list()
)

```

Arguments

x	A numeric vector of longitude values in the range $[-180, 180]$.
y	A numeric vector of latitude values in the range $[-90, 90]$.
address	Name of the address column in the output. The default is "address".
full_results	A logical value indicating whether to return all available API fields. The default, FALSE, returns only latitude, longitude and address.
return_coords	A logical value indicating whether to return the input coordinates with the results.
verbose	A logical value indicating whether to display API request details.
progressbar	A logical value indicating whether to display a progress bar for multiple queries.
outsr	Spatial reference of the output coordinates. The default is NULL, which uses the service default. See Details and arc_spatial_references .
langcode	Language of the returned addresses.
featuretypes	A character vector that limits the possible match types. The default is NULL, which does not filter by feature type. See Details .
locationtype	Location represented by the output geometry when featuretypes is "PointAddress" or "Subaddress". Valid values are "rooftop" and "street". The default is NULL.
custom_query	A named list with additional API parameters.

Details

See the [ArcGIS REST API documentation](#) for more information and valid values.

Value

A [tibble](#) with one match for each coordinate pair. The API output fields x and y are named lon and lat. These coordinates correspond to the matched feature and may differ from the input x and y values.

See [ArcGIS REST API output](#) for field details.

outsr

The spatial reference can be specified as a well-known ID (WKID). If not specified, the spatial reference of the output locations is the same as that of the service (WGS 84, that is, WKID 4326).

See [arc_spatial_references](#) for values and examples.

featuretypes

See `vignette("feature-types", package = "arcgeocoder")` for a detailed explanation of this argument.

This argument restricts the feature types returned by a reverse geocoding request. Possible values are "StreetInt", "DistanceMarker", "StreetAddress", "StreetName", "POI", "Subaddress", "PointAddress", "Postal" and "Locality".

Supply multiple values as a character vector, for example, `c("PointAddress", "StreetAddress")`.

References

[ArcGIS REST API reverseGeocode](#).

See Also

Geocoding and reverse geocoding functions: [arc_geo\(\)](#), [arc_geo_categories\(\)](#), [arc_geo_multi\(\)](#)

Examples

```
arc_reverse_geo(x = -73.98586, y = 40.75728)

# Several coordinate pairs.
arc_reverse_geo(x = c(-73.98586, -3.188375), y = c(40.75728, 55.95335))

# Use additional API options.
sev <- arc_reverse_geo(
  x = c(-73.98586, -3.188375),
  y = c(40.75728, 55.95335),
  # Restrict results to specific feature types.
  featuretypes = "POI,StreetInt",
  # Return results in this WKID.
  outsr = 102100,
  verbose = TRUE, full_results = TRUE
)

dplyr::glimpse(sev)
```

arc_spatial_references*Spatial references supported by the ArcGIS REST API*

Description

A dataset of coordinate reference systems (CRSs) supported by the ArcGIS REST API.

Format

A [tibble](#) with 9,608 rows and eight variables:

projtype Projection type ("ProjectedCoordinateSystems", "GeographicCoordinateSystems" or "VerticalCoordinateSystems").

wkid Well-known ID (WKID).

latestWkid Current WKID if wkid is deprecated.

authority WKID authority (Esri or EPSG).

deprecated Whether wkid is deprecated.

description Human-readable description of the spatial reference.

areaname Area of use of the spatial reference.

wkt Well-known text (WKT) representation of the spatial reference. Useful when working with [sf](#) or [terra](#).

Details

This dataset is useful when using the `outsr` argument.

Some projection IDs have changed over time. For example, Web Mercator `wkid = 102100` is deprecated and its current equivalent is `wkid = 3857`. Both values work and return equivalent results.

Note

Data extracted on **15 January 2026**.

Source

[Esri Projection Engine factory](#).

See Also

- [sf::st_crs\(\)](#) inspects coordinate reference systems.
- [arc_geo\(\)](#), [arc_geo_multi\(\)](#), [arc_geo_categories\(\)](#) and [arc_reverse_geo\(\)](#) accept spatial references.

Examples

```
# Get all possible values.
data("arc_spatial_references")
arc_spatial_references

# Find the deprecated Web Mercator WKID.
library(dplyr)
wkid <- arc_spatial_references |>
  filter(latestWkid == 3857 & deprecated) |>
  slice(1)

glimpse(wkid)

add <- arc_geo("London, United Kingdom", outsr = wkid$wkid)

# Compare `lat`, `lon`, `wkid` and the current ID in `latestWkid`.
add |>
  select(lat, lon, wkid, latestWkid) |>
  glimpse()

# Look up the deprecated WKID.

try(sf::st_crs(wkid$wkid))

# Look up the current WKID.
try(sf::st_crs(wkid$latestWkid))

# Look up the WKT definition.
try(sf::st_crs(wkid$wkt))
```

Index

* datasets

- arc_categories, [2](#)
- arc_spatial_references, [15](#)

* geocoders

- arc_geo, [4](#)
- arc_geo_categories, [6](#)
- arc_geo_multi, [9](#)
- arc_reverse_geo, [12](#)

- arc_categories, [2](#), [5–7](#), [10](#)
- arc_geo, [4](#), [7](#)
- arc_geo(), [2](#), [3](#), [7](#), [9](#), [11](#), [14](#), [15](#)
- arc_geo_categories, [6](#)
- arc_geo_categories(), [2](#), [3](#), [5](#), [11](#), [14](#), [15](#)
- arc_geo_multi, [9](#)
- arc_geo_multi(), [2–5](#), [7](#), [14](#), [15](#)
- arc_reverse_geo, [12](#)
- arc_reverse_geo(), [5](#), [7](#), [11](#), [15](#)
- arc_spatial_references, [4](#), [5](#), [7](#), [10](#), [11](#), [13](#),
[14](#), [15](#)

- sf::st_crs(), [15](#)

- tibble, [2](#), [5](#), [7](#), [10](#), [13](#), [15](#)