Package 'amapGeocode'

January 20, 2026

Type Package

Title An Interface to the 'AutoNavi Maps' API Geocoding Services

Version 1.0.0

Description Getting and parsing data of location geocode/reverse-geocode and administrative regions from 'AutoNavi Maps'<https://lbs.amap.com/api/webservice/summary> API.

License MIT + file LICENSE

Depends R (>= 4.1.0)

Imports httr2, jsonlite, rlang, digest, xml2, tibble, dplyr, utils

Suggests knitr, rmarkdown, testthat, httptest2, vcr, spelling, covr, shiny, bslib, DT, readr, tools

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

Language en-US

URL https://womeimingzi11.github.io/amapGeocode/, https://github.com/womeimingzi11/amapGeocode

BugReports https://github.com/womeimingzi11/amapGeocode/issues

NeedsCompilation no

Author Han Chen [aut, cre], Wanyanhan Jiang [ctb]

Maintainer Han Chen <chenhan28@gmail.com>

Repository CRAN

Date/Publication 2026-01-20 13:10:02 UTC

Contents

`amap_config` *Configure Amap settings*

Description

Configure Amap settings

Usage

```
amap_config(
  signature = NULL,
  secret = NULL,
  key = NULL,
  enabled = TRUE,
  max_active = NULL,
  throttle = NULL
)
```

Arguments

`signature`	Optional. Signature configuration. Use 'FALSE' to disable, a single string secret, or a list.
`secret`	Optional. Secret key used for request signing.
`key`	Optional. Optional API key override when signing is enabled.
`enabled`	Optional. Logical flag to enable or disable signing.
`max_active`	Optional. Maximum number of active concurrent HTTP requests when bulk operations are executed with 'httr2::req_perform_parallel()'. Defaults to 3.

throttle — Optional. Throttling configuration for outgoing HTTP requests. Use 'FALSE' to disable throttling, 'TRUE' to enable with defaults, or a list with any of the following fields: 'enabled' (logical), 'rate' (numeric), 'capacity' (numeric), 'fill_time_s' (numeric), and 'realm' (character).

Defaults are safe for AutoNavi's QPS limits: 'max_active = 3' and 'throttle = list(rate = 3, fill_time_s = 1)'.

---

amap_gui *Launch the amapGeocode Graphical Interface*

Description

Launches a Shiny application that provides a graphical user interface for accessing the functionality of 'amapGeocode'. The app supports:

- Geocoding (Address to Coordinates) - Single and Batch (CSV)
- Reverse Geocoding (Coordinates to Location) - Single and Batch (CSV)
- Coordinate Conversion - Single and Batch
- Configuration of API Key and Rate Limits

Usage

```
amap_gui()
```

Details

The application requires the following suggested packages: 'shiny', 'bslib', 'DT', and 'readr'. If they are not installed, the function will prompt the user to install them.

Value

No return value, called for side effects (launching the application).

Examples

```
## Not run:
if (interactive()) {
  amap_gui()
}

## End(Not run)
```

amap_sign *Generate Amap signature*

Description

Generate Amap signature

Usage

```
amap_sign(params, secret, path)
```

Arguments

params	Required. Named list of request parameters to sign.
secret	Required. Secret key used for request signing.
path	Required. Request path portion of the API URL.

convertCoord *Convert coordinates to the AutoNavi system*

Description

Convert coordinates to the AutoNavi system

Usage

```
convertCoord(
  locations,
  key = NULL,
  coordsys = NULL,
  sig = NULL,
  output = "tibble",
  keep_bad_request = TRUE,
  ...
)
```

Arguments

locations	Required. Coordinate string(s) to convert. Accepts a character vector.
key	Optional. AutoNavi API key. You can also set this globally via ‘options(amap_key = "your-key")‘.
coordsys	Optional. Source coordinate system (‘gps‘, ‘mapbar‘, ‘baidu‘, ‘autonavi‘).
sig	Optional. Manual digital signature. Most workflows can enable automatic signing via [with_amap_signature()] or [amap_config()].

`output` Optional. Output data structure. Supported values are ‘"tibble"‘ (default), ‘"JSON"‘, and ‘"XML"‘.

`keep_bad_request` Optional. When ‘TRUE‘ (default) API errors are converted into placeholder rows so that batched workflows continue. When ‘FALSE‘ errors are raised as ‘amap_api_error‘ conditions.

`...` Optional. Included for forward compatibility only.

Value

When ‘output = "tibble"‘, a ‘tibble‘ with columns ‘lng‘ and ‘lat‘ is returned. The table preserves the input order and gains a ‘rate_limit‘ attribute containing any rate limit headers returned by the API. When ‘output‘ is ‘"JSON"‘ or ‘"XML"‘, the parsed body is returned without further processing.

See Also

[extractConvertCoord()], [with_amap_signature()], [amap_config()]

Examples

```
## Not run:
convertCoord("116.481499,39.990475", coordsys = "gps")

## End(Not run)
```

`extractAdmin` *Extract subordinate administrative regions from a district response*

Description

Extract subordinate administrative regions from a district response

Usage

```
extractAdmin(res, include_polyline = FALSE)
```

Arguments

`res` Required. Response object returned by [getAdmin()] with ‘output = "JSON"‘ or by the AutoNavi district API.

`include_polyline` Logical indicating whether to include the polyline column (requires ‘extensions = "all"‘). Defaults to ‘FALSE‘.

Value

A ‘tibble‘ describing each administrative region present in the response. The table includes parent metadata (‘parent_name‘, ‘parent_adcode‘, ‘parent_level‘), centre coordinates (‘lng‘, ‘lat‘), and a ‘depth‘ column describing the nesting level (0 for the matched region, 1+ for subregions). When no results are present a single placeholder row filled with ‘NA‘ values is returned.

See Also

[getAdmin()]

Examples

```
## Not run:
raw <- getAdmin("Sichuan Province", output = "JSON")
extractAdmin(raw)

## End(Not run)
```

`extractConvertCoord` *Extract converted coordinates from a conversion response*

Description

Extract converted coordinates from a conversion response

Usage

```
extractConvertCoord(res)
```

Arguments

`res`	Required. Response object returned by [convertCoord()] with ‘output = "JSON"‘ or by the AutoNavi coordinate conversion API.

Value

A ‘tibble‘ with columns ‘lng‘ and ‘lat‘. When no data is present a single placeholder row filled with ‘NA‘ values is returned.

See Also

[convertCoord()]

Examples

```
## Not run:
raw <- convertCoord("116.481499,39.990475", coordsys = "gps", output = "JSON")
extractConvertCoord(raw)

## End(Not run)
```

`extractCoord` *Extract coordinate from a geocoding response*

Description

Extract coordinate from a geocoding response

Usage

```
extractCoord(res)
```

Arguments

`res`	Required. Response object returned by [getCoord()] with ‘output = "JSON"‘ or by the AutoNavi geocoding API.

Value

A ‘tibble‘ with one row per geocode candidate. The table contains the original columns provided by the API alongside a ‘match_rank‘ column that indicates the ordering reported by AutoNavi. When the response does not contain any matches a single placeholder row filled with ‘NA‘ values is returned.

See Also

[getCoord()]

Examples

```
## Not run:
raw <- getCoord("IFS Chengdu", output = "JSON")
extractCoord(raw)

## End(Not run)
```

`extractLocation` *Extract location from coordinate request*

Description

Extract location from coordinate request

Usage

```
extractLocation(res, details = NULL)
```

Arguments

`res`	Required. Response object returned by [getLocation()] with ‘output = "JSON"‘ or by the AutoNavi reverse-geocoding API.
`details`	Optional. Character vector describing which extended detail payloads to parse into list-columns. Valid values are ‘"pois"‘, ‘"roads"‘, ‘"roadinters"‘, and ‘"aois"‘. Use ‘"all"‘ to include every detail payload.

Value

A ‘tibble‘ describing the parsed reverse-geocode results. Each row corresponds to an element in the API response. When no data is present a single placeholder row filled with ‘NA‘ values is returned.

See Also

[getLocation()]

Examples

```
## Not run:
raw <- getLocation(104.043284, 30.666864, output = "JSON")
extractLocation(raw, details = c("pois", "roads"))

## End(Not run)
```

`getAdmin` *Get subordinate administrative regions from keywords*

Description

Get subordinate administrative regions from keywords

Usage

```
getAdmin(
  keywords,
  key = NULL,
  subdistrict = NULL,
  page = NULL,
  offset = NULL,
  extensions = NULL,
  filter = NULL,
  callback = NULL,
  output = "tibble",
  keep_bad_request = TRUE,
  include_polyline = FALSE,
  ...
)
```

Arguments

keywords	Required. Search keywords. Accepts a character vector; each element is queried in turn.
key	Optional. AutoNavi API key. You can also set this globally via ‘options(amap_key = "your-key")‘.
subdistrict	Optional. Subordinate administrative depth (0-3). Defaults to the API’s behaviour.
page	Optional. Page number when multiple pages are available.
offset	Optional. Maximum records per page (maximum 20).
extensions	Optional. Either ‘"base"‘ or ‘"all"‘. Required for polyline data.
filter	Optional. Filter by designated administrative divisions (adcode).
callback	Optional. JSONP callback. When supplied, the raw response string is returned.
output	Optional. Output data structure. Supported values are ‘"tibble"‘ (default), ‘"JSON"‘, and ‘"XML"‘.
keep_bad_request	Optional. When ‘TRUE‘ (default) API errors are converted into placeholder rows so that batched workflows continue. When ‘FALSE‘ errors are raised as ‘amap_api_error‘ conditions.
include_polyline	Optional. When ‘TRUE‘, and when the request is made with ‘extensions = "all"‘, polyline strings are included in the parsed output.
...	Optional. Included for forward compatibility only.

Value

When ‘output = "tibble"‘, a ‘tibble‘ containing administrative region details is returned. The table preserves the input order and includes parent metadata (‘parent_name‘, ‘parent_adcode‘, ‘parent_level‘) and a ‘depth‘ column describing the nesting level. A ‘rate_limit‘ attribute is attached when rate limit headers are present. When ‘output‘ is ‘"JSON"‘ or ‘"XML"‘, the parsed body is returned without further processing.

See Also

[extractAdmin()], [with_amap_signature()], [amap_config()]

Examples

```
## Not run:
getAdmin("Sichuan Province", subdistrict = 1)

# Include polylines (requires extensions = "all")
getAdmin("Sichuan Province",
  subdistrict = 1,
  extensions = "all", include_polyline = TRUE
)

## End(Not run)
```

---

getCoord *Get coordinate from location*

Description

Get coordinate from location

Usage

```
getCoord(
  address,
  key = NULL,
  city = NULL,
  sig = NULL,
  output = "tibble",
  callback = NULL,
  keep_bad_request = TRUE,
  mode = c("best", "all"),
  batch = FALSE,
  ...
)
```

Arguments

address	Required. Structured address information. The value can be a character vector; each element will be queried in turn.
key	Optional. AutoNavi API key. You can also set this globally via ‘options(amap_key = "your-key")‘.
city	Optional. City hint that narrows down the search scope. When ‘batch = TRUE‘, only a single city value is supported.

`sig` Optional. Digital signature supplied manually. Most users should instead enable automatic signing via [with_amap_signature()] or [amap_config()].

`output` Optional. Output data structure. Supported values are ‘"tibble"‘ (default), ‘"JSON"‘, and ‘"XML"‘.

`callback` Optional. JSONP callback. When supplied the raw body is returned as a character vector.

`keep_bad_request` Optional. When ‘TRUE‘ (default) API errors are converted into placeholder rows so that vectorised or batched workflows continue. When ‘FALSE‘ errors are raised as ‘amap_api_error‘ conditions.

`mode` Optional. Controls how geocode candidates are returned. Use ‘"best"‘ (default) to keep the highest ranked candidate for each query or ‘"all"‘ to return all matches alongside ranking metadata.

`batch` Optional. When ‘TRUE‘, requests are chunked into groups of ten addresses using the API’s batch mode.

Bulk requests are executed with ‘httr2::req_perform_parallel()‘ (curl multi; no additional R sessions) and are protected by throttling configured via [amap_config()].

`...` Optional. Included for forward compatibility only.

Value

When ‘output = "tibble"‘, a ‘tibble‘ containing geocode results is returned. The table preserves the input order and gains a ‘rate_limit‘ attribute containing any rate limit headers returned by the API. When ‘mode = "all"‘, additional metadata columns (‘query‘, ‘query_index‘, and ‘match_rank‘) are included. When ‘output‘ is ‘"JSON"‘ or ‘"XML"‘, the parsed body is returned without further processing.

See Also

[extractCoord()], [with_amap_signature()], [amap_config()]

Examples

```
## Not run:
# Basic lookup (best match only)
getCoord("IFS Chengdu")

# Retrieve all candidates for a single query
getCoord("LOS ANGELES", mode = "all")

# Batch ten addresses at a time
getCoord(rep("Chengdu IFS", 12), batch = TRUE)

# Temporarily enable automatic request signing
with_amap_signature("your-secret", getCoord("IFS Chengdu"))

## End(Not run)
```

getLocation *Get location from coordinate*

Description

Get location from coordinate

Usage

```
getLocation(
  lng,
  lat,
  key = NULL,
  poitype = NULL,
  radius = NULL,
  extensions = NULL,
  roadlevel = NULL,
  sig = NULL,
  output = "tibble",
  callback = NULL,
  homeorcorp = 0,
  keep_bad_request = TRUE,
  batch = FALSE,
  details = NULL,
  ...
)
```

Arguments

lng	Required. Longitude in decimal degrees. Can be a numeric vector.
lat	Required. Latitude in decimal degrees. Must be the same length as ‘lng‘.
key	Optional. AutoNavi API key. You can also set this globally via ‘options(amap_key = "your-key")‘.
poitype	Optional. Return nearby POI types. Only meaningful when ‘extensions = "all"‘.
radius	Optional. Search radius in metres (0-3000).
extensions	Optional. Either ‘"base"‘ (default) or ‘"all"‘ to request extended detail payloads.
roadlevel	Optional. Road level filter. Only applies when ‘extensions = "all"‘.
sig	Optional. Manual digital signature. Most workflows can enable automatic signing via [with_amap_signature()] or [amap_config()].
output	Optional. Output format. Supported values are ‘"tibble"‘ (default), ‘"JSON"‘, and ‘"XML"‘.
callback	Optional. JSONP callback. When supplied the raw response string is returned.
homeorcorp	Optional. Optimise POI ordering: ‘0‘ (default) for none, ‘1‘ for home-centric, ‘2‘ for corporate-centric ordering.

`keep_bad_request`
: Optional. When ‘TRUE‘ (default) API errors are converted into placeholder rows so that batched workflows continue. When ‘FALSE‘ errors are raised as ‘amap_api_error‘ conditions.

`batch`
: Optional. When ‘TRUE‘, requests are chunked into groups of ten coordinates using the API’s batch mode.

  Bulk requests are executed with ‘httr2::req_perform_parallel()‘ (curl multi; no additional R sessions) and are protected by throttling configured via [amap_config()].

`details`
: Optional. Character vector describing which extended list-columns to include in the parsed output. Supported values are ‘"pois"‘, ‘"roads"‘, ‘"roadinters"‘, and ‘"aois"‘. Use ‘"all"‘ to include every detail payload. Defaults to ‘NULL‘, which omits nested payloads.

`...`
: Optional. Included for forward compatibility only.

Value

When ‘output = "tibble"‘, a ‘tibble‘ with one row per coordinate is returned. The table preserves the input order and gains a ‘rate_limit‘ attribute containing any rate limit headers returned by the API. When ‘details‘ are requested, corresponding list-columns (‘pois‘, ‘roads‘, ‘roadinters‘, ‘aois‘) contain nested ‘tibble‘ objects. When ‘output‘ is ‘"JSON"‘ or ‘"XML"‘, the parsed body is returned without further processing.

See Also

[extractLocation()], [with_amap_signature()], [amap_config()]

Examples

```
## Not run:
getLocation(104.043284, 30.666864)

# Request extended POI details
getLocation(104.043284, 30.666864,
  extensions = "all", details = "pois"
)

# Batch reverse-geocode ten points at a time
lngs <- rep(104.043284, 12)
lats <- rep(30.666864, 12)
getLocation(lngs, lats, batch = TRUE)

## End(Not run)
```

`num_coord_to_str_loc` *Take longitude and latitude from location string out.*

Description

Take longitude and latitude from location string out.

Usage

```
num_coord_to_str_loc(lng, lat)
```

Arguments

`lng`	Required. Longitude in decimal
`lat`	Required. Latitude in decimal

Value

Comma binded coordinate string

`str_loc_to_num_coord` *Take longitude and latitude from location string out.*

Description

Take longitude and latitude from location string out.

Usage

```
str_loc_to_num_coord(str_location)
```

Arguments

`str_location`	Required. Location string from response

Value

vector contains Longitude and Latitude in numeric

`with_amap_signature` *Execute code with temporary signature settings*

Description

Execute code with temporary signature settings

Usage

```
with_amap_signature(secret, expr, key = NULL, enabled = TRUE)
```

Arguments

`secret`	Required. Secret key used for request signing.
`expr`	Required. Expression to evaluate with signing enabled.
`key`	Optional. Optional API key override when signing is enabled.
`enabled`	Optional. Logical flag to enable or disable signing.

Index