# Package: vantorr (via r-universe)

November 13, 2026

**Title** Access 'Maxar' and 'Vantor' Satellite Imagery APIs

**Version** 0.3.0

**Description** Provides a client for the 'Maxar' and 'Vantor' 'REST' and 'OGC' APIs (<https://developers.maxar.com/docs/>), enabling programmatic access to high-resolution satellite imagery (such as 'WorldView', 'Vivid' base maps, and analysis-ready data). Includes functions for 'OAuth2' and API key authentication, discovery and 'STAC' catalog search, 'OGC' web services ('WMS', 'WMTS', 'WFS'), survey fieldwork resource generation (printable field atlases, index grids, 'Survey Solutions' and 'ODK' assignment tables, offline 'GeoPackage' and 'Esri' tile packages), interactive 'leaflet' and 'shiny' application modules, and spatial data retrieval integrated with 'sf', 'terra', and 'stars'.

**License** MIT + file LICENSE

**URL** https://github.com/michael-cw/vantorr, https://michael-cw.github.io/vantorr/, https://developers.maxar.com/docs/

**BugReports** https://github.com/michael-cw/vantorr/issues

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.3.2

**Depends** R (>= 4.1.0)

**Imports** cli, httr2, sf

**Suggests** bslib, knitr, leaflet, markdown, rmarkdown, shiny, stars, terra, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Michael Wild [aut, cre]

**Maintainer** Michael Wild <michael.wild@me.com>

**Repository** https://cran.r-universe.dev

**Date/Publication** 2026-11-13 17:56:35 UTC

**RemoteUrl** https://github.com/cran/vantorr

**RemoteRef** HEAD

**RemoteSha** 1df08fb7007b1ac604132c11c7582a6d1bee3d42

# Contents

add_vantor_basemap . . . 3
add_vantor_imagery . . . 4
add_vantor_seamlines . . . 6
convert_bbox_to_3857 . . . 8
create_maxar_request . . . 8
get_maxar_token . . . 9
get_maxar_wms_basemap . . . 10
get_maxar_wms_tiles . . . 12
refresh_maxar_token . . . 13
run_vantor_demo_app . . . 14
vantor_as_bbox . . . 15
vantor_auth . . . 16
vantor_auth_clear . . . 17
vantor_auth_server . . . 18
vantor_auth_ui . . . 19
vantor_explorer_server . . . 19
vantor_explorer_ui . . . 20
vantor_export_capi . . . 21
vantor_field_atlas . . . 23
vantor_get_seamlines . . . 25
vantor_get_usage . . . 26
vantor_has_auth . . . 27
vantor_offline_package . . . 28
vantor_plus_code . . . 30
vantor_survey_grid . . . 31
vantor_vintage_report . . . 32
vantor_wmts_url . . . 33
vantor_write_tpk . . . 34

**Index** **37**

add_vantor_basemap *Add a Vantor Vivid Mosaic Basemap to a Leaflet Map*

## Description

Adds a high-resolution Vantor / Maxar Vivid Mosaic basemap (`VIVID_STANDARD_30` or `VIVID_ADVANCED_15`) to a `leaflet` or `leafletProxy` map widget using either pre-cached OGC WMTS tiles (recommended for fast interactive panning) or dynamic OGC WMS rendering.

## Usage

```
add_vantor_basemap(
  map,
  product = "VIVID_STANDARD_30",
  service = c("WMTS", "WMS"),
  api_key = NULL,
  group = "Vantor Vivid",
  layerId = NULL,
  show_seamlines = FALSE,
  cql_filter = NULL,
  shoreline_masking = FALSE,
  max_zoom = 21L,
  attribution = "Imagery &copy; Vantor / Maxar Technologies",
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)

addVantorBasemap(
  map,
  product = "VIVID_STANDARD_30",
  service = c("WMTS", "WMS"),
  api_key = NULL,
  group = "Vantor Vivid",
  layerId = NULL,
  show_seamlines = FALSE,
  cql_filter = NULL,
  shoreline_masking = FALSE,
  max_zoom = 21L,
  attribution = "Imagery &copy; Vantor / Maxar Technologies",
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `map` | A `leaflet` or `leaflet_proxy` map widget object. |

| | |
|---|---|
| product | Character string. Vivid product name. Default is "VIVID_STANDARD_30". Options include "VIVID_STANDARD_30" (30 cm) and "VIVID_ADVANCED_15" (15 cm). |
| service | Character string. OGC service protocol: "WMTS" (default, fastest in browser) or "WMS". |
| api_key | Character string. Optional Vantor/Maxar API key. If NULL, resolved from session authentication (vantor_auth()) or environment variables (VANTOR_API_KEY / MAXAR_API_KEY). |
| group | Character string. Name of the leaflet layer group. Default is "Vantor Vivid". |
| layerId | Character string. Optional leaflet layer ID. |
| show_seamlines | Logical. If TRUE, also overlays the corresponding cutline/seamline layer via add_vantor_seamlines(). Default is FALSE. |
| cql_filter | Character string. Optional custom CQL filter. |
| shoreline_masking | Logical. Apply shoreline masking (used when service = "WMS"). Default is FALSE. |
| max_zoom | Integer. Maximum zoom level for the tile layer. Default is 21L. |
| attribution | Character string. Map attribution text. |
| options | A list of tile options passed to leaflet::tileOptions() or leaflet::WMSTileOptions(). |
| verbose | Logical. Print status messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

The modified leaflet or leaflet_proxy map object.

## Examples

```
if (requireNamespace("leaflet", quietly = TRUE)) {
  m <- leaflet::leaflet() |>
    add_vantor_basemap(
      product = "VIVID_STANDARD_30",
      api_key = "example_api_key",
      verbose = FALSE
    )
}
```

---

# add_vantor_imagery *Add Vantor Archive Streaming Imagery to a Leaflet Map*

---

## Description

Streams individual archive satellite imagery strips or filtered collections from the Vantor / Maxar Streaming Imagery OGC WMS endpoint (https://api.maxar.com/streaming/v1/ogc/wms) onto a leaflet or leafletProxy map.

## Usage

```
add_vantor_imagery(
  map,
  feature_id = NULL,
  cql_filter = NULL,
  api_key = NULL,
  group = "Vantor Archive Imagery",
  layerId = NULL,
  format = "image/vnd.jpeg-png",
  transparent = TRUE,
  max_zoom = 21L,
  attribution = "Imagery &copy; Vantor / Maxar Technologies",
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)

addVantorImagery(
  map,
  feature_id = NULL,
  cql_filter = NULL,
  api_key = NULL,
  group = "Vantor Archive Imagery",
  layerId = NULL,
  format = "image/vnd.jpeg-png",
  transparent = TRUE,
  max_zoom = 21L,
  attribution = "Imagery &copy; Vantor / Maxar Technologies",
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| map | A leaflet or leaflet_proxy map widget object. |
| feature_id | Character string. Optional catalog / feature ID of a specific satellite strip (e.g., "10300100EBEDF600"). Used to construct cql_filter = "featureId='<feature_id>'" if cql_filter is NULL. |
| cql_filter | Character string. Optional custom CQL filter, such as "(acquisitionDate>='2023-01-01')AND(cloud |
| api_key | Character string. Optional Vantor/Maxar API key. |
| group | Character string. Leaflet layer group name. Default is "Vantor Archive Imagery". |
| layerId | Character string. Optional leaflet layer ID. |
| format | Character string. WMS tile image format. Default is "image/vnd.jpeg-png". |
| transparent | Logical. Request transparent background for no-data pixels. Default is TRUE. |
| max_zoom | Integer. Maximum zoom level. Default is 21L. |
| attribution | Character string. Map attribution text. |

| | |
|---|---|
| options | Optional `leaflet::WMSTileOptions()` list. |
| verbose | Logical. Print status messages. Default is `getOption("vantorr.verbose", TRUE)`. |

## Value

The modified `leaflet` or `leaflet_proxy` map object.

## Examples

```
if (requireNamespace("leaflet", quietly = TRUE)) {
  m <- leaflet::leaflet() |>
    add_vantor_imagery(
      feature_id = "10300100EBEDF600",
      api_key = "example_api_key",
      verbose = FALSE
    )
}
```

---

# add_vantor_seamlines *Add Vantor Basemap Seamlines to a Leaflet Map*

---

## Description

Overlays Vantor / Maxar Vivid Mosaic seamlines onto a `leaflet` or `leafletProxy` map. Supports either fast OGC WMS raster cutline rendering (`mode = "wms"`, default) or interactive OGC WFS vector polygons (`mode = "wfs"`) with clickable popups displaying acquisition date (`acq_time`), resolution (`gsd`), sensor (`sensor`), and catalog ID (`catid`).

## Usage

```
add_vantor_seamlines(
  map,
  product = "VIVID_STANDARD_30",
  mode = c("wms", "wfs"),
  bbox = NULL,
  api_key = NULL,
  group = "Vantor Seamlines",
  layerId = NULL,
  cql_filter = NULL,
  color = "#00E5FF",
  weight = 2,
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)

addVantorSeamlines(
```

```
  map,
  product = "VIVID_STANDARD_30",
  mode = c("wms", "wfs"),
  bbox = NULL,
  api_key = NULL,
  group = "Vantor Seamlines",
  layerId = NULL,
  cql_filter = NULL,
  color = "#00E5FF",
  weight = 2,
  options = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| map | A leaflet or leaflet_proxy map widget object. |
| product | Character string. Vivid product name. Default is "VIVID_STANDARD_30". |
| mode | Character string. Either "wms" (default, transparent raster overlay across all zoom levels) or "wfs" (fetches vector polygons via vantor_get_seamlines() for bbox with interactive metadata popups). |
| bbox | Optional bounding box or spatial object required when mode = "wfs". |
| api_key | Character string. Optional Vantor/Maxar API key. |
| group | Character string. Leaflet layer group name. Default is "Vantor Seamlines". |
| layerId | Character string. Optional leaflet layer ID. |
| cql_filter | Character string. Optional custom CQL filter. |
| color | Character string. Stroke color for vector polygons when mode = "wfs". Default is "#00E5FF". |
| weight | Numeric. Stroke weight when mode = "wfs". Default is 2. |
| options | Optional leaflet::WMSTileOptions() list when mode = "wms". |
| verbose | Logical. Print status messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

The modified leaflet or leaflet_proxy map object.

## Examples

```
if (requireNamespace("leaflet", quietly = TRUE)) {
  m <- leaflet::leaflet() |>
    add_vantor_seamlines(
      product = "VIVID_STANDARD_30",
      api_key = "example_api_key",
      verbose = FALSE
    )
}
```

## convert_bbox_to_3857 *Convert Lat/Lon Bounding Box to Web Mercator*

### Description

Helper function to convert a bounding box or spatial object from EPSG:4326 (lat/lon) to EPSG:3857 (Web Mercator).

### Usage

```
convert_bbox_to_3857(bbox_4326)
```

### Arguments

| | |
|---|---|
| bbox_4326 | Numeric vector of length 4 (c(lon_min, lat_min, lon_max, lat_max)) or any spatial object supported by vantor_as_bbox(). |

### Value

Numeric vector of length 4 with bounding box coordinates in EPSG:3857: c(x_min, y_min, x_max, y_max).

### Examples

```
# Denver area in lat/lon
bbox_latlon <- c(-105.01, 39.72, -104.96, 39.75)
bbox_mercator <- convert_bbox_to_3857(bbox_latlon)
```

## create_maxar_request *Create Maxar / Vantor API Request with Authentication*

### Description

Helper function to create an httr2 request with Vantor/Maxar bearer token or API key authentication and built-in retry logic for transient HTTP errors.

### Usage

```
create_maxar_request(url, token = NULL, api_key = NULL, verbose = FALSE)
```

### Arguments

| | |
|---|---|
| url | Character string. The API endpoint URL. |
| token | Character string. Optional bearer token (access_token from get_maxar_token()). |
| api_key | Character string. Optional Maxar/Vantor API key. |
| verbose | Logical. If TRUE, prints confirmation of the configured request. Defaults to FALSE. |

## Value

An `httr2_request` object with authentication headers and retry policies set.

## Examples

```
req <- create_maxar_request(
  "https://api.maxar.com/discovery/v1",
  api_key = "example_key"
)
```

---

`get_maxar_token` *Get Maxar / Vantor API Bearer Token*

---

## Description

Retrieves a bearer token from the Maxar/Vantor Authentication API using the OAuth2 Password Flow and caches it in the current session for automatic refresh.

## Usage

```
get_maxar_token(
  username = Sys.getenv("VANTOR_USERNAME", unset = Sys.getenv("MAXAR_USERNAME", unset =
    "")),
  password = Sys.getenv("VANTOR_PASSWORD", unset = Sys.getenv("MAXAR_PASSWORD", unset =
    "")),
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `username` | Character string. The email address associated with your Maxar/Vantor account. Defaults to `Sys.getenv("VANTOR_USERNAME", Sys.getenv("MAXAR_USERNAME"))`. |
| `password` | Character string. The password for your Maxar/Vantor account. Defaults to `Sys.getenv("VANTOR_PASSWORD", Sys.getenv("MAXAR_PASSWORD"))`. |
| `verbose` | Logical. If `TRUE`, prints token status details. Defaults to `getOption("vantorr.verbose", TRUE)`. |

## Value

A list containing the token response (`access_token`, `expires_in`, `refresh_token`, `token_type`), or `NULL` (with an alert) if the API cannot be reached.

## Examples

```
if (vantor_has_auth() && nzchar(Sys.getenv("VANTOR_USERNAME"))) {
  token_response <- get_maxar_token(verbose = FALSE)
}
```

---

`get_maxar_wms_basemap` *Get Maxar / Vantor WMS Basemap as GeoTIFF or Spatial Raster Object*

---

## Description

Retrieves a Maxar/Vantor Vivid Mosaic basemap (30 cm or 15 cm) via OGC WMS for a specified area of interest. Supports spatial inputs from `sf`, `terra`, and `stars`, as well as numeric bounding boxes, and can return a file path, `terra::SpatRaster`, or `stars` object.

## Usage

```
get_maxar_wms_basemap(
  bbox,
  token = NULL,
  output_file = tempfile(pattern = "vantor_basemap_", fileext = ".tif"),
  width = NULL,
  height = NULL,
  resolution = 0.3,
  product = "VIVID_STANDARD_30",
  crs = c("EPSG:3857", "EPSG:4326"),
  cql_filter = NULL,
  shoreline_masking = FALSE,
  api_key = NULL,
  return_type = c("file", "SpatRaster", "stars", "stars_proxy"),
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `bbox` | A spatial object (`sf`, `sfc`, `bbox`, `SpatRaster`, `SpatVector`, `SpatExtent`, `stars`, `stars_proxy`) or a numeric vector of length 4: `c(xmin, ymin, xmax, ymax)`. Automatically transformed to the target `crs`. |
| `token` | Character string. Optional OAuth2 bearer token from `get_maxar_token()`. If `NULL`, resolved via `api_key`, session cache (`vantor_auth()`), or environment variables. |
| `output_file` | Character string. Path where the GeoTIFF file will be saved. Defaults to `tempfile(pattern = "vantor_basemap_", fileext = ".tif")` in compliance with CRAN policies. |

| | |
|---|---|
| width | Integer. Width of the output image in pixels. If NULL, calculated from resolution. |
| height | Integer. Height of the output image in pixels. If NULL, calculated from resolution. |
| resolution | Numeric. Target ground resolution in meters. Default is 0.3 (30 cm). Used to calculate width and height if not explicitly provided. |
| product | Character string. Vivid product name. Default is "VIVID_STANDARD_30". Options include "VIVID_STANDARD_30" and "VIVID_ADVANCED_15". |
| crs | Character string. Output CRS requested from WMS ("EPSG:3857" or "EPSG:4326"). Default is "EPSG:3857". |
| cql_filter | Character string. Optional custom CQL filter. If NULL, constructed automatically from product and shoreline_masking. |
| shoreline_masking | Logical. Apply shoreline masking. Default is FALSE. |
| api_key | Character string. Optional Maxar/Vantor API key (maxar-api-key). |
| return_type | Character string. Return format: "file" (default, returns the file path invisibly), "SpatRaster" (returns a terra::SpatRaster), "stars" (returns an in-memory stars object), or "stars_proxy". |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

Depending on return_type, the path to the saved GeoTIFF file (invisibly when return_type = "file"), a SpatRaster, or a stars object. Returns NULL invisibly (with an informative message) if the API request fails.

## Examples

```
if (vantor_has_auth()) {
  # Define area of interest using an sf bounding box (WGS84)
  aoi <- sf::st_bbox(
    c(xmin = -105.005, ymin = 39.735, xmax = -105.000, ymax = 39.740),
    crs = sf::st_crs(4326)
  )

  # Retrieve basemap to a temporary GeoTIFF file
  tif_file <- get_maxar_wms_basemap(
    bbox = aoi,
    resolution = 0.5,
    verbose = FALSE
  )
}
```

`get_maxar_wms_tiles` *Get Multiple WMS Basemap Tiles for Large Areas*

## Description

Splits a large area of interest into a regular grid of tiles, downloads each tile via `get_maxar_wms_basemap()`, and optionally merges them into a single GeoTIFF, `terra::SpatRaster`, or `stars` object.

## Usage

```
get_maxar_wms_tiles(
  bbox,
  token = NULL,
  api_key = NULL,
  tile_size = 2500,
  output_dir = tempfile(pattern = "vantor_tiles_"),
  merge_tiles = TRUE,
  final_output = tempfile(pattern = "vantor_merged_", fileext = ".tif"),
  return_type = c("file", "SpatRaster", "stars", "stars_proxy"),
  progress_callback = NULL,
  verbose = getOption("vantorr.verbose", TRUE),
  ...
)
```

## Arguments

| | |
|---|---|
| `bbox` | A spatial object (`sf`, `sfc`, `bbox`, `SpatRaster`, `SpatVector`, `SpatExtent`, `stars`, `stars_proxy`) or a numeric vector of length 4. |
| `token` | Character string. Optional OAuth2 bearer token. |
| `api_key` | Character string. Optional Maxar/Vantor API key. |
| `tile_size` | Numeric. Size of each tile side in meters. Default is `2500` (2.5 km). |
| `output_dir` | Character string. Directory for individual tile files. Defaults to a new directory inside `tempdir()` in compliance with CRAN policy. |
| `merge_tiles` | Logical. Merge downloaded tiles into a single mosaic. Default is `TRUE` (requires the `terra` package). |
| `final_output` | Character string. File path for the merged GeoTIFF. Defaults to `tempfile(pattern = "vantor_merged_", fileext = ".tif")`. |
| `return_type` | Character string. One of `"file"`, `"SpatRaster"`, `"stars"`, or `"stars_proxy"`. |
| `progress_callback` | Optional function taking `(i, n_tiles)` called after each tile download (useful for integrating with `shiny::incProgress()`). |
| `verbose` | Logical. Print progress messages. Default is `getOption("vantorr.verbose", TRUE)`. |
| `...` | Additional arguments passed to `get_maxar_wms_basemap()`. |

## Value

If `merge_tiles = TRUE`, returns the merged output formatted according to `return_type`. If `merge_tiles = FALSE`, returns a character vector of downloaded tile file paths.

## Examples

```
if (vantor_has_auth()) {
  large_aoi <- sf::st_bbox(
    c(xmin = -105.02, ymin = 39.72, xmax = -104.98, ymax = 39.75),
    crs = sf::st_crs(4326)
  )
  merged_path <- get_maxar_wms_tiles(
    bbox = large_aoi,
    tile_size = 2000,
    resolution = 1.0,
    verbose = FALSE
  )
}
```

---

# refresh_maxar_token *Refresh Maxar / Vantor API Bearer Token*

---

## Description

Refreshes a bearer token using a refresh token, avoiding the need to resend username and password, and updates the session token cache.

## Usage

```
refresh_maxar_token(
  refresh_token = .vantorr_env$refresh_token,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `refresh_token` | Character string. The refresh token from a previous token response. Defaults to the cached session refresh token if available. |
| `verbose` | Logical. If `TRUE`, prints token refresh details. Defaults to `getOption("vantorr.verbose", TRUE)`. |

## Value

A list containing the new token response, or `NULL` if refresh fails.

## Examples

```
if (nzchar(Sys.getenv("VANTOR_USERNAME")) && nzchar(Sys.getenv("VANTOR_PASSWORD"))) {
  tok <- get_maxar_token(verbose = FALSE)
  if (!is.null(tok$refresh_token)) {
    new_token <- refresh_maxar_token(tok$refresh_token, verbose = FALSE)
  }
}
```

---

`run_vantor_demo_app` *Launch Interactive Vantor Explorer Demo Shiny App*

---

## Description

Launches a standalone Shiny application combining `vantor_auth_ui()` and `vantor_explorer_ui()` for interactively browsing Vantor Vivid basemaps, inspecting seamline acquisition dates, monitoring account usage, and exporting GeoTIFFs.

## Usage

```
run_vantor_demo_app(
  initial_bbox = c(xmin = -105.01, ymin = 39.73, xmax = -104.98, ymax = 39.75),
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `initial_bbox` | Numeric vector of length 4 (`c(xmin, ymin, xmax, ymax)` in `EPSG:4326`) or any spatial object supported by `vantor_as_bbox()`. Defaults to downtown Denver, CO. |
| `verbose` | Logical. Print status messages. Default is `getOption("vantorr.verbose", TRUE)`. |

## Value

A `shiny.appobj` object.

## Examples

```
if (interactive()) {
  run_vantor_demo_app()
}
```

vantor_as_bbox *Coerce Spatial Objects to a Bounding Box in a Target CRS*

## Description

S3 generic function that extracts a bounding box from spatial objects across major R geography packages (sf, terra, stars) or numeric vectors and transforms it to a target Coordinate Reference System (CRS).

## Usage

```
vantor_as_bbox(x, target_crs = 3857, source_crs = NULL, ...)
```

## Arguments

| | |
|---|---|
| x | A spatial object (sf, sfc, bbox, SpatRaster, SpatVector, SpatExtent, stars, stars_proxy) or a numeric vector of length 4: c(xmin, ymin, xmax, ymax). |
| target_crs | Integer or character string. Target EPSG code or CRS definition. Default is 3857 (Web Mercator). |
| source_crs | Integer or character string. Source CRS used only when x is a plain numeric vector without a "crs" attribute. If NULL, heuristically detects EPSG:4326 if coordinates fall within valid longitude/latitude bounds ([-180, 180] and [-90, 90]), or EPSG:3857 otherwise. |
| ... | Additional arguments passed to methods. |

## Value

An sf bbox object (c(xmin, ymin, xmax, ymax)) in target_crs.

## Examples

```
# From a lat/lon numeric vector (automatically detected as EPSG:4326)
vantor_as_bbox(c(-105.01, 39.72, -104.96, 39.75), target_crs = 3857)

# From an sf bbox object
bb <- sf::st_bbox(c(xmin = -77.04, ymin = 38.88, xmax = -77.03, ymax = 38.89),
                  crs = sf::st_crs(4326))
vantor_as_bbox(bb, target_crs = 3857)
```

vantor_auth *Authenticate with the Vantor / Maxar API*

## Description

Sets or retrieves authentication credentials for the current R session. Supports both long-lived API keys (`maxar-api-key`) and OAuth2 Bearer tokens (`Authorization: Bearer <token>`) with automatic refresh when expired.

## Usage

```
vantor_auth(
  api_key = NULL,
  username = NULL,
  password = NULL,
  token = NULL,
  refresh_token = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `api_key` | Character string. A Vantor/Maxar API key. |
| `username` | Character string. Email address for OAuth2 authentication. |
| `password` | Character string. Password for OAuth2 authentication. |
| `token` | Character string. An existing OAuth2 access token. |
| `refresh_token` | Character string. An existing OAuth2 refresh token. |
| `verbose` | Logical. Print status messages. Defaults to `getOption("vantorr.verbose", TRUE)`. |

## Details

By default, `vantor_auth()` checks the following environment variables if arguments are not explicitly supplied:

- `VANTOR_API_KEY` or `MAXAR_API_KEY`
- `VANTOR_USERNAME` or `MAXAR_USERNAME`
- `VANTOR_PASSWORD` or `MAXAR_PASSWORD`

## Value

Invisibly returns a list representing the active authentication state, or `NULL` if no credentials could be resolved.

### Note

**Authentication Scope Across Vantor Endpoints:**

- **OGC & Non-Administrative APIs** (`get_maxar_wms_basemap()`, `get_maxar_wms_tiles()`, `vantor_get_seamlines()`): Accept **either** an API key (`maxar-api-key`) or an OAuth2 Bearer token.
- **Browser-Based** `leaflet` **Tile Layers** (`add_vantor_basemap()`, `add_vantor_seamlines()`, `add_vantor_imagery()`, `vantor_wmts_url()`): Require an **API key** (`maxar_api_key` URL query parameter).
- **Account Services / Admin APIs** (`vantor_get_usage()`): Require an **OAuth2 Bearer token** (`username` and `password`, or `token`). Per the Vantor API specification, Admin APIs do not accept API keys.

### Examples

```
# Check if credentials are configured in the environment
vantor_has_auth(verbose = FALSE)

# Set an API key for the current session
vantor_auth(api_key = "your_api_key_here", verbose = FALSE)

# Clear session credentials
vantor_auth_clear(verbose = FALSE)
```

---

## vantor_auth_clear *Clear Stored Vantor Credentials*

---

### Description

Removes any cached API key or OAuth2 tokens from the internal session environment.

### Usage

```
vantor_auth_clear(verbose = getOption("vantorr.verbose", TRUE))
```

### Arguments

| | |
|---|---|
| `verbose` | Logical. Print confirmation message. Defaults to `getOption("vantorr.verbose", TRUE)`. |

### Value

Invisibly returns `TRUE`.

### Examples

```
vantor_auth_clear(verbose = FALSE)
```

vantor_auth_server *Vantor Authentication Shiny Module Server*

## Description

Server logic for the vantor_auth_ui() Shiny module. Manages session credentials, provides reactive authentication state for downstream modules (such as vantor_explorer_server()), and automatically refreshes OAuth2 bearer tokens in the background so long-running Shiny sessions do not expire.

## Usage

```
vantor_auth_server(
  id,
  auto_refresh_ms = 3e+05,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| id | Character string. The Shiny module namespace ID. |
| auto_refresh_ms | Numeric. Interval in milliseconds to check and automatically refresh expiring OAuth2 tokens via shiny::invalidateLater(). Default is 300000 (5 minutes). |
| verbose | Logical. Print status messages to the R console. Default is getOption("vantorr.verbose", TRUE). |

## Value

A shiny::reactive expression returning a list with elements: authenticated (logical), type ("api_key", "bearer", or "none"), api_key (character or NULL), and updated_at (POSIXct).

## Examples

```
if (requireNamespace("shiny", quietly = TRUE)) {
  server <- function(input, output, session) {
    auth_state <- vantor_auth_server("auth", verbose = FALSE)
  }
}
```

vantor_auth_ui *Vantor Authentication Shiny Module UI*

## Description

Creates a compact, namespace-safe Shiny UI panel for configuring Vantor / Maxar API credentials (API Key or OAuth2 Username/Password), displaying connection status, and optionally checking account credit usage.

## Usage

```
vantor_auth_ui(id, show_usage = TRUE)
```

## Arguments

| | |
|---|---|
| id | Character string. The Shiny module namespace ID. |
| show_usage | Logical. Include a button and output area to query account credit usage via vantor_get_usage(). Default is TRUE. |

## Value

A shiny.tag.list UI definition.

## Examples

```
if (requireNamespace("shiny", quietly = TRUE)) {
  ui <- shiny::fluidPage(
    vantor_auth_ui("auth")
  )
}
```

vantor_explorer_server *Vantor Interactive Imagery Explorer Shiny Module Server*

## Description

Server logic for the vantor_explorer_ui() Shiny module. Renders an interactive leaflet map with Vantor Vivid basemaps, updates layers via leaflet::leafletProxy(), queries WFS seamlines for the current viewport, and provides a shiny::downloadHandler() with progress bar integration for exporting GeoTIFFs via get_maxar_wms_basemap() or get_maxar_wms_tiles().

## Usage

```
vantor_explorer_server(
  id,
  initial_bbox = c(xmin = -105.01, ymin = 39.73, xmax = -104.98, ymax = 39.75),
  auth_reactive = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| id | Character string. The Shiny module namespace ID. |
| initial_bbox | Numeric vector of length 4 (c(xmin, ymin, xmax, ymax) in EPSG:4326) or any spatial object supported by vantor_as_bbox() defining the initial map view. Defaults to downtown Denver, CO. |
| auth_reactive | Optional shiny::reactive returned by vantor_auth_server() to automatically refresh map layers when credentials change. |
| verbose | Logical. Print status messages to the R console. Default is getOption("vantorr.verbose", TRUE). |

## Value

A list of shiny::reactive expressions:

- bbox: Reactive sf::bbox (EPSG:4326) of the current map viewport
- seamlines: Reactive sf object of WFS seamlines when inspected
- product: Reactive selected Vivid product name
- resolution: Reactive selected ground resolution in meters

## Examples

```
if (requireNamespace("shiny", quietly = TRUE) &&
    requireNamespace("leaflet", quietly = TRUE)) {
  server <- function(input, output, session) {
    explorer <- vantor_explorer_server("explorer", verbose = FALSE)
  }
}
```

---

# vantor_explorer_ui *Vantor Interactive Imagery Explorer Shiny Module UI*

---

## Description

Creates a drop-in, namespace-safe Shiny UI component featuring an interactive leaflet map with Vantor Vivid Mosaic basemap switching (VIVID_STANDARD_30 and VIVID_ADVANCED_15), seamline cutline overlay, WFS seamline vintage inspection, live AOI bounding box metrics, and one-click GeoTIFF export.

### Usage

```
vantor_explorer_ui(id, height = "550px")
```

### Arguments

| | |
|---|---|
| id | Character string. The Shiny module namespace ID. |
| height | Character string. CSS height of the leaflet map widget. Default is "550px". |

### Value

A shiny.tag.list UI definition.

### Examples

```
if (requireNamespace("shiny", quietly = TRUE) &&
    requireNamespace("leaflet", quietly = TRUE)) {
  ui <- shiny::fluidPage(
    vantor_explorer_ui("explorer")
  )
}
```

---

## vantor_export_capi *Prepare and Export Survey Sample Points for CAPI Fieldwork Apps*

---

### Description

Enriches a set of survey sample locations (sf points or polygons, such as household structures, enumeration clusters, or ground-truth plots) with navigation and preload fields ready for **ODK Collect**, **KoboToolbox**, **ArcGIS Survey123**, **World Bank Survey Solutions**, and **QField**:

- sample_id: Unique formatted sample identifier
- longitude, latitude: WGS84 decimal degree coordinates
- odk_geopoint: Preload string ("<lat> <lon> 0 0") used by ODK / Kobo geopoint questions
- plus_code: Offline-capable 10-character Google Plus Code via vantor_plus_code()
- google_maps_url: One-click navigation link for mobile enumerators
- Optional grid cell assignment (cell_id) if a grid from vantor_survey_grid() is supplied
- Optional imagery vintage metadata (acq_date, sensor, gsd_m) if seamlines are supplied

## Usage

```
vantor_export_capi(
  samples,
  id_col = NULL,
  grid = NULL,
  seamlines = NULL,
  output_csv = NULL,
  output_geojson = NULL,
  output_suso_tab = NULL,
  suso_gps_var = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| samples | An sf object (POINT or POLYGON; polygons are converted to representative points via sf::st_point_on_surface()). |
| id_col | Character string. Optional existing column name in samples to use as sample_id. If NULL, generates sequential IDs ("S001", "S002", ...). |
| grid | Optional sf grid from vantor_survey_grid() used to assign each sample to its Enumeration Area / grid cell_id. |
| seamlines | Optional sf seamline polygons from vantor_get_seamlines() used to attach imagery vintage (imagery_acq_date, imagery_sensor) to each sample point. |
| output_csv | Optional file path to write a CSV preload table (for ODK Collect, KoboToolbox, and ArcGIS Survey123). Default is NULL. |
| output_geojson | Optional file path to write a GeoJSON file for mobile GIS map overlays (QField, Mergin Maps). Default is NULL. |
| output_suso_tab | Optional file path to write a tab-delimited (.tab) assignment preload file formatted for **World Bank Survey Solutions** Headquarters (Upload Assignments). Includes composite GPS columns (<suso_gps_var>__Latitude and <suso_gps_var>__Longitude) so sampled structures appear as interactive pins on the Interviewer tablet Map Dashboard over the .tpk basemap, plus _target_area (from cell_id), plus_code, and imagery_acq_date. Default is NULL. |
| suso_gps_var | Optional character string. Variable name of the identifying GPS question in Survey Solutions Designer (e.g., "hh_gps"). When provided (or when output_suso_tab is set), the coordinate columns on both the returned sf object and the exported .tab file are named <suso_gps_var>__Latitude and <suso_gps_var>__Longitude instead of latitude and longitude. Default is NULL (uses "hh_gps" automatically when output_suso_tab is supplied). |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

## Details

Optionally writes the enriched dataset to a CSV preload table (output_csv) and/or a GeoJSON file (output_geojson).

## Value

An enriched `sf` point object in `EPSG:4326`.

## Examples

```
pts <- sf::st_as_sf(
  data.frame(
    hh_id = c("HH_01", "HH_02"),
    lon = c(-105.005, -105.002),
    lat = c(39.735, 39.738)
  ),
  coords = c("lon", "lat"),
  crs = 4326
)

capi_sf <- vantor_export_capi(pts, id_col = "hh_id", suso_gps_var = "hh_gps", verbose = FALSE)
head(capi_sf)
```

---

# vantor_field_atlas *Generate a Printable Multi-Page Field Atlas (PDF Map Book)*

---

## Description

Creates a multi-page printable PDF Field Atlas (Map Book) for field supervisors and survey enumerators:

- **Page 1 (Overview Index Map)**: Displays the full Area of Interest, all Enumeration Area / survey grid boundaries labeled by `id_col` (e.g., `"A1"`, `"A2"`), sample points, scale bar, and north arrow.
- **Subsequent Pages (Detail Enumeration Sheets)**: One page per grid cell or Enumeration Area polygon, zoomed in with a metric margin (`buffer_m`), optional high-resolution Vantor RGB satellite imagery background, labeled sample points, scale bar, north arrow, and a metadata header/footer with coordinates and imagery vintage.

## Usage

```
vantor_field_atlas(
  grid,
  points = NULL,
  raster = NULL,
  output_file = tempfile(pattern = "vantor_field_atlas_", fileext = ".pdf"),
  id_col = "cell_id",
  point_label_col = "sample_id",
  title = "Vantor Field Enumeration Atlas",
  buffer_m = 50,
  fetch_basemap = FALSE,
  resolution = 0.5,
```

```
  product = "VIVID_STANDARD_30",
  paper_width = 11,
  paper_height = 8.5,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| grid | An sf polygon object representing survey grid cells (e.g. from vantor_survey_grid()) or Enumeration Area boundaries. |
| points | Optional sf point object of sample locations (e.g. from vantor_export_capi()) to plot and label on each map sheet. |
| raster | Optional terra::SpatRaster or GeoTIFF file path to use as the RGB satellite imagery background. |
| output_file | Character string. Output PDF file path. Defaults to tempfile(pattern = "vantor_field_atlas_", fileext = ".pdf") in compliance with CRAN policy. |
| id_col | Character string. Column name in grid used to label each enumeration sheet. Default is "cell_id". |
| point_label_col | Character string. Optional column name in points used to label sample markers on detail sheets. Default is "sample_id". |
| title | Character string. Title printed on the overview cover map. Default is "Vantor Field Enumeration Atlas". |
| buffer_m | Numeric. Buffer distance in meters around each detail cell. Default is 50. |
| fetch_basemap | Logical. If TRUE and raster is NULL, downloads Vantor WMS satellite imagery for the atlas extent via get_maxar_wms_basemap(). Default is FALSE. |
| resolution | Numeric. Ground resolution in meters when fetch_basemap = TRUE. Default is 0.5. |
| product | Character string. Vivid product name when fetch_basemap = TRUE. Default is "VIVID_STANDARD_30". |
| paper_width | Numeric. PDF page width in inches. Default is 11 (landscape). |
| paper_height | Numeric. PDF page height in inches. Default is 8.5. |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

## Details

Works both **offline** (using a pre-downloaded raster SpatRaster or vector-only mode when fetch_basemap = FALSE) and **online** (fetching Vantor WMS basemap imagery automatically when fetch_basemap = TRUE).

## Value

Invisibly returns the file path to the generated PDF Field Atlas.

## Examples

```
aoi <- sf::st_bbox(
  c(xmin = -105.01, ymin = 39.73, xmax = -105.00, ymax = 39.74),
  crs = sf::st_crs(4326)
)
grid_sf <- vantor_survey_grid(aoi, cell_size = 600, verbose = FALSE)
pdf_path <- vantor_field_atlas(grid_sf, verbose = FALSE)
unlink(pdf_path)
```

---

# vantor_get_seamlines *Get Vantor Basemap Seamlines as an* sf *Object (OGC WFS)*

---

## Description

Queries the Vantor / Maxar Basemap Seamlines OGC Web Feature Service (WFS) to retrieve vector polygons representing the individual satellite imagery strips that make up a Vivid Mosaic basemap within an area of interest.

## Usage

```
vantor_get_seamlines(
  bbox,
  product = "VIVID_STANDARD_30",
  cql_filter = NULL,
  token = NULL,
  api_key = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| bbox | A spatial object (sf, sfc, bbox, SpatRaster, SpatVector, SpatExtent, stars, stars_proxy) or a numeric vector of length 4: c(xmin, ymin, xmax, ymax). |
| product | Character string. Vivid mosaic product name. Default is "VIVID_STANDARD_30". Options include "VIVID_STANDARD_30" and "VIVID_ADVANCED_15". |
| cql_filter | Character string. Optional custom CQL filter. If NULL, constructed automatically from product and bbox. |
| token | Character string. Optional OAuth2 bearer token. |
| api_key | Character string. Optional Maxar/Vantor API key. |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

### Details

Each returned polygon feature typically includes metadata attributes such as:

- `acq_time`, `acq_time_earliest`, `acq_time_latest`: Imagery acquisition timestamp(s)
- `gsd`: Ground sample distance (resolution)
- `sensor` / `vehicle_name`: Satellite sensor identifier (e.g., `"WV03"`, `"WV02"`)
- `ona`: Off-nadir angle
- `catid`: Archive catalog ID of the source strip

### Value

An `sf` object (in `EPSG:4326`) containing the seamline polygons and their acquisition metadata, or `NULL` invisibly if the API request fails.

### Examples

```
if (vantor_has_auth()) {
  aoi <- sf::st_bbox(
    c(xmin = -105.01, ymin = 39.73, xmax = -104.99, ymax = 39.75),
    crs = sf::st_crs(4326)
  )
  seamlines <- vantor_get_seamlines(aoi, verbose = FALSE)
}
```

---

## vantor_get_usage *Get Vantor Account Usage and Credit Overview*

---

### Description

Queries the Vantor / Maxar Usage API (`v2/activation/overview`) to retrieve activation credit limits, total usage, and categorized consumption metrics. Useful for monitoring account quota in interactive R sessions or Shiny dashboards before launching large multi-tile downloads.

### Usage

```
vantor_get_usage(
  category = c("PRODUCT", "DAY", "WEEK", "MONTH", "QUARTER"),
 unit = c("CREDIT", "FEATURE_COUNT", "REQUEST_COUNT", "MAP_VIEW", "GIGABYTE", "SQ_KM"),
  token = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

### Arguments

| | |
|---|---|
| category | Character string. Grouping category for usage breakdown. One of "PRODUCT" (default), "DAY", "WEEK", "MONTH", or "QUARTER". |
| unit | Character string. Measurement unit for usage metrics. One of "CREDIT" (default), "FEATURE_COUNT", "REQUEST_COUNT", "MAP_VIEW", "GIGABYTE", or "SQ_KM". |
| token | Character string. Optional OAuth2 bearer token (access_token from get_maxar_token()). If NULL, resolved from the active OAuth2 session cache or VANTOR_USERNAME / VANTOR_PASSWORD environment variables. |
| verbose | Logical. Print summary messages. Default is getOption("vantorr.verbose", TRUE). |

### Value

A list containing the parsed usage overview response (including activations, usageCategories, and a tidy summary_df data frame when available), or NULL invisibly if the API request fails.

### Note

**Authentication Requirement (OAuth2 Bearer Token Only):** The Usage API (https://api.maxar.com/usage/v2/activ is part of the Vantor **Account Services (Admin API)** (https://developers.maxar.com/docs/admin/guides/usage). Per the Vantor API specification, **API keys** (maxar-api-key) **are not accepted by Admin APIs**. You must authenticate using an OAuth2 Bearer Token via vantor_auth(username = "...", password = "...") (or VANTOR_USERNAME and VANTOR_PASSWORD in .Renviron), and your account must have Admin permissions to view activation usage.

### Examples

```
if (nzchar(Sys.getenv("VANTOR_USERNAME")) && nzchar(Sys.getenv("VANTOR_PASSWORD"))) {
  usage <- vantor_get_usage(category = "PRODUCT", unit = "CREDIT", verbose = FALSE)
}
```

---

## vantor_has_auth *Check if Vantor Authentication is Available*

---

### Description

Checks whether an active session token, API key, or environment credentials are available. Useful for guarding examples and automated tests.

### Usage

```
vantor_has_auth(verbose = FALSE)
```

### Arguments

| | |
|---|---|
| verbose | Logical. If TRUE, prints the detected authentication source. Defaults to FALSE. |

### Value

Logical TRUE if authentication credentials are available, FALSE otherwise.

### Examples

```
vantor_has_auth()
```

---

## vantor_offline_package

*Package Offline Fieldwork Vector Layers and Satellite Basemaps*

---

### Description

Bundles survey grid cells, sample points, seamline vintage polygons, and high-resolution Vantor satellite imagery into a self-contained directory ready to copy onto mobile tablets for offline use in **QField**, **Mergin Maps**, **ArcGIS Field Maps**, **World Bank Survey Solutions**, or **ODK Collect**:

- Writes all vector layers (`grid`, `samples`, `seamlines`) into a single multi-layer OGC **GeoPackage** (`.gpkg`).
- Writes a CSV preload table (`<package_name>_samples.csv`) and a **World Bank Survey Solutions** tab-delimited assignment preload file (`<package_name>_suso_preload.tab`) when `samples` is provided.
- Saves or downloads the satellite basemap as a Web Mercator (`EPSG:3857`) georeferenced **GeoTIFF** (`.tif`).
- Optionally builds a multi-zoom Esri **Tile Package** (`.tpk` **or** `.tpkx`) via `vantor_write_tpk()` for zoomable offline basemaps in **Survey Solutions** (ArcGIS Maps SDK for .NET v200.x) and **ArcGIS Field Maps**.
- Optionally exports zipped **Esri Shapefiles** (`_shapes.zip`) for uploading vector Enumeration Area boundaries to Survey Solutions Headquarters.

### Usage

```
vantor_offline_package(
  output_dir = tempfile(pattern = "vantor_offline_pkg_"),
  grid = NULL,
  samples = NULL,
  seamlines = NULL,
  raster = NULL,
  fetch_basemap = FALSE,
  create_tpk = FALSE,
  tpk_zoom = 15:16,
  create_shapefiles = FALSE,
```

```
  suso_gps_var = "hh_gps",
  resolution = 0.5,
  product = "VIVID_STANDARD_30",
  package_name = "vantor_fieldwork",
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| output_dir | Character string. Target directory for the offline bundle. Defaults to tempfile(pattern = "vantor_offline_pkg_") in compliance with CRAN policy. |
| grid | Optional sf polygon layer (e.g., from vantor_survey_grid()). |
| samples | Optional sf point layer (e.g., from vantor_export_capi()). |
| seamlines | Optional sf seamline vintage layer (e.g., from vantor_get_seamlines()). |
| raster | Optional terra::SpatRaster or GeoTIFF file path to include in the offline bundle. Automatically reprojected to Web Mercator (EPSG:3857) for Survey Solutions / Esri compatibility. |
| fetch_basemap | Logical. If TRUE and raster is NULL, downloads the Vantor WMS basemap for the extent of grid or samples. Default is FALSE. |
| create_tpk | Logical or character ("tpk" or "tpkx"). If TRUE or "tpk", generates an Esri .tpk (Compact Cache V1) tile package; if "tpkx", generates an open-specification .tpkx (Compact Cache V2) tile package supported by Survey Solutions v24.09+ (ArcGIS Maps SDK v200.x). Default is FALSE. |
| tpk_zoom | Integer vector. Web Mercator zoom levels (0 to 22) to include when create_tpk is enabled. Default is 15:16. |
| create_shapefiles | Logical. If TRUE, also writes grid and samples as EPSG:4326 Esri Shapefiles bundled into <package_name>_shapes.zip for Survey Solutions Headquarters upload. Default is FALSE. |
| suso_gps_var | Character string. Variable name of the identifying GPS question in your Survey Solutions Designer questionnaire used to format <suso_gps_var>__Latitude and <suso_gps_var>__Longitude columns in <package_name>_suso_preload.tab. Default is "hh_gps". |
| resolution | Numeric. Ground resolution in meters when fetch_basemap = TRUE. Default is 0.5. |
| product | Character string. Vivid product name when fetch_basemap = TRUE. Default is "VIVID_STANDARD_30". |
| package_name | Character string. Base filename prefix for bundled files. Default is "vantor_fieldwork". |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

A list containing the paths to the generated offline bundle files: output_dir, gpkg_path, layers, csv_path, suso_tab_path, raster_path, tpk_path, tpk_zip_path, and shapefile_zip_path.

## Examples

```
aoi <- sf::st_bbox(
  c(xmin = -105.01, ymin = 39.73, xmax = -105.00, ymax = 39.74),
  crs = sf::st_crs(4326)
)
grid_sf <- vantor_survey_grid(aoi, cell_size = 600, verbose = FALSE)
bundle <- vantor_offline_package(
  grid = grid_sf,
  create_shapefiles = TRUE,
  verbose = FALSE
)
unlink(bundle$output_dir, recursive = TRUE)
```

---

# vantor_plus_code *Encode Coordinates as Open Location Codes (Google Plus Codes)*

---

## Description

Converts WGS84 longitude and latitude coordinates into 10- or 11-character **Open Location Codes (Plus Codes)** (e.g., `"85FQPPV4+2V"`). Plus Codes are an open, offline-capable alphanumeric street address standard supported by Google Maps, ODK Collect, and field navigation apps without requiring an internet connection.

## Usage

```
vantor_plus_code(lon, lat, code_length = 10L)
```

## Arguments

| | |
|---|---|
| `lon` | Numeric vector of longitudes in decimal degrees (`-180` to `180`). |
| `lat` | Numeric vector of latitudes in decimal degrees (`-90` to `90`). |
| `code_length` | Integer. Number of significant characters in the Plus Code (`10` for ~14 x 14 m precision, `11` for ~3 x 3 m building-level precision). Default is `10L`. |

## Value

A character vector of formatted Plus Codes (with a `"+"` separator after the 8th character).

## Examples

```
# Encode coordinates in downtown Denver, CO
vantor_plus_code(lon = -104.995, lat = 39.740)
```

vantor_survey_grid *Generate an Alphanumeric Survey Index Grid over an Area of Interest*

## Description

Partitions an Area of Interest (AOI) into a regular grid of square or hexagonal enumeration cells with human-readable alphanumeric index labels (`"A1"`, `"A2"`, `"B1"`, etc.), cell centroids (`centroid_lon`, `centroid_lat`), and area in square meters (`area_m2`). Designed for planning field enumeration areas, printing map book index sheets with `vantor_field_atlas()`, and stratified spatial sampling.

## Usage

```
vantor_survey_grid(
  aoi,
  cell_size = 500,
  shape = c("square", "hexagonal"),
  clip_to_aoi = TRUE,
  crs = 4326,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `aoi` | A spatial object (`sf`, `sfc`, `bbox`, `SpatRaster`, `SpatVector`, `SpatExtent`, `stars`, `stars_proxy`) or a numeric bounding box vector of length 4 (`c(xmin, ymin, xmax, ymax)` in `EPSG:4326`). |
| `cell_size` | Numeric. Target width/diameter of each grid cell in meters (computed in Web Mercator `EPSG:3857`). Default is `500` (500 m). |
| `shape` | Character string. Grid cell geometry: `"square"` (default) or `"hexagonal"`. |
| `clip_to_aoi` | Logical. If `TRUE` (default) and `aoi` is an `sf`, `sfc`, or `SpatVector` polygon, retains only grid cells that spatially intersect the `aoi` geometry. |
| `crs` | Numeric or character. Output Coordinate Reference System for the returned `sf` grid. Default is `4326` (`EPSG:4326`). |
| `verbose` | Logical. Print status messages. Default is `getOption("vantorr.verbose", TRUE)`. |

## Value

An `sf` polygon object with columns:

- `cell_id`: Alphanumeric grid label (`"A1"`, `"A2"`, ..., `"B1"`, ...)
- `row_idx`: Row index (top-to-bottom, 1-based)
- `col_idx`: Column index (left-to-right, 1-based)
- `centroid_lon`, `centroid_lat`: Cell centroid coordinates in `EPSG:4326`
- `area_m2`: Approximate planar area in square meters
- `geometry`: Polygon geometry in `crs`

## Examples

```
aoi <- sf::st_bbox(
  c(xmin = -105.01, ymin = 39.73, xmax = -104.99, ymax = 39.75),
  crs = sf::st_crs(4326)
)
grid_sf <- vantor_survey_grid(aoi, cell_size = 500, verbose = FALSE)
head(grid_sf)
```

---

# vantor_vintage_report *Audit Basemap Imagery Vintage and Sensor Quality Across Survey Areas*

---

## Description

Evaluates the acquisition dates (`acq_time`), ground sample distance (`gsd`), off-nadir angle (`ona`), and contributing satellite sensors across an Area of Interest or a set of Enumeration Area / survey grid polygons. Can either query the Vantor OGC WFS Seamlines endpoint via `vantor_get_seamlines()` or use a pre-fetched `seamlines` `sf` object for offline workflows.

## Usage

```
vantor_vintage_report(
  aoi,
  seamlines = NULL,
  product = "VIVID_STANDARD_30",
  max_age_days = 730,
  reference_date = Sys.Date(),
  token = NULL,
  api_key = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| `aoi` | An `sf` object (such as Enumeration Areas or the output of `vantor_survey_grid()`), or any spatial bounding box supported by `vantor_as_bbox()`. |
| `seamlines` | Optional `sf` object of seamline polygons previously fetched via `vantor_get_seamlines()`. If `NULL`, fetched automatically from the Vantor WFS API. |
| `product` | Character string. Vivid product name when fetching seamlines. Default is `"VIVID_STANDARD_30"`. |
| `max_age_days` | Numeric. Threshold in days beyond which imagery is flagged with `vintage_warning = TRUE`. Default is `730` (2 years). |
| `reference_date` | `Date` or character string (`"YYYY-MM-DD"`) used to calculate `age_days`. Defaults to `Sys.Date()`. |
| `token` | Character string. Optional OAuth2 bearer token. |

| | |
|---|---|
| api_key | Character string. Optional Vantor/Maxar API key. |
| verbose | Logical. Print summary messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

An sf object representing aoi enriched with vintage audit columns:

- n_strips: Number of intersecting satellite mosaic strips
- earliest_acq_date, latest_acq_date: Earliest and latest strip acquisition dates (YYYY-MM-DD)
- max_age_days: Age in days of the oldest intersecting imagery strip relative to reference_date
- min_gsd_m, max_gsd_m: Minimum and maximum ground sample distance (meters)
- max_ona_deg: Maximum off-nadir angle (degrees)
- sensors: Comma-separated list of contributing satellite sensors
- vintage_warning: Logical TRUE if max_age_days > max_age_days threshold

## Examples

```
# Offline example using synthetic seamlines and a survey grid
aoi <- sf::st_bbox(
  c(xmin = -105.01, ymin = 39.73, xmax = -105.00, ymax = 39.74),
  crs = sf::st_crs(4326)
)
grid_sf <- vantor_survey_grid(aoi, cell_size = 600, verbose = FALSE)

mock_seamline <- sf::st_sf(
  acq_time = "2024-05-10T18:00:00Z",
  gsd = 0.30,
  ona = 11.5,
  sensor = "WV03",
  catid = "1040010012345600",
  geometry = sf::st_as_sfc(aoi)
)

audit_sf <- vantor_vintage_report(
  aoi = grid_sf,
  seamlines = mock_seamline,
  verbose = FALSE
)
```

---

# vantor_wmts_url *Construct a Vantor OGC WMTS Tile URL Template*

---

## Description

Generates an OGC Web Map Tile Service (WMTS) URL template ({z}/{x}/{y}) for streaming Vantor / Maxar Vivid Mosaic basemap tiles in EPSG:3857 (Web Mercator) within leaflet, mapgl, or tmap.

## Usage

```
vantor_wmts_url(
  product = "VIVID_STANDARD_30",
  api_key = NULL,
  format = "image/jpeg",
  cql_filter = NULL,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| product | Character string. Vivid product name. Default is "VIVID_STANDARD_30". Options include "VIVID_STANDARD_30" and "VIVID_ADVANCED_15". |
| api_key | Character string. Optional Vantor/Maxar API key. If NULL, resolved from vantor_auth() or environment variables (VANTOR_API_KEY / MAXAR_API_KEY). |
| format | Character string. Tile image format. Default is "image/jpeg". Options include "image/jpeg", "image/png", and "image/vnd.jpeg-png". |
| cql_filter | Character string. Optional custom CQL filter. If NULL, constructed automatically from product. |
| verbose | Logical. Print confirmation message. Default is FALSE. |

## Value

A character string containing the WMTS GetTile URL template with {z}, {x}, and {y} placeholders.

## Examples

```
url_template <- vantor_wmts_url(
  product = "VIVID_STANDARD_30",
  api_key = "example_api_key"
)
```

---

# vantor_write_tpk *Write an Esri Tile Package (.tpk or .tpkx) for Offline Field Basemaps*

---

## Description

Generates a multi-zoom Web Mercator (EPSG:3857) Esri Tile Package (.tpk Compact Cache V1 or .tpkx Compact Cache V2) in pure R without requiring proprietary ArcGIS software or ArcGIS Online credits. Designed for zoomable offline satellite basemaps in **World Bank Survey Solutions** (including v24.09+ built on the **ArcGIS Maps SDK for .NET v200.x**), **ArcGIS Field Maps**, and **ArcGIS Survey123**.

Tiles can either be sliced offline from a local terra::SpatRaster / GeoTIFF (raster), or fetched directly across zoom levels from the Vantor WMS/WMTS imagery service (raster = NULL).

## Usage

```
vantor_write_tpk(
  x = NULL,
  output_file = tempfile(fileext = ".tpk"),
  zoom = 15:17,
  format = c("auto", "tpk", "tpkx"),
  raster = NULL,
  product = "VIVID_STANDARD_30",
  max_tiles = 2500L,
  title = "Vantor Satellite Basemap",
  zip_for_suso = FALSE,
  aoi = NULL,
  verbose = getOption("vantorr.verbose", TRUE)
)
```

## Arguments

| | |
|---|---|
| x | Optional spatial object (sf, sfc, bbox, SpatRaster, SpatVector, SpatExtent, stars, or numeric c(xmin, ymin, xmax, ymax)) defining the Area of Interest. If NULL, the extent of raster is used. |
| output_file | Character string. Destination file path ending in .tpk or .tpkx. Defaults to tempfile(fileext = ".tpk") in compliance with CRAN policy. |
| zoom | Integer vector of Web Mercator (EPSG:3857) zoom levels (0 to 22) to render into the tile pyramid. Default is 15:17. |
| format | Character string: "auto" (inferred from output_file extension), "tpk" (Compact Cache V1 with .bundle + .bundlx and XML metadata), or "tpkx" (Apache-2.0 open specification Compact Cache V2 with .bundle and JSON metadata, supported by ArcGIS Maps SDK v200.x). |
| raster | Optional terra::SpatRaster or file path to a GeoTIFF raster. When supplied, tiles are resampled offline from raster. When NULL, tiles are downloaded from the Vantor WMS service using active credentials (vantor_auth()). |
| product | Character string. Vantor Vivid mosaic product identifier when fetching online tiles (raster = NULL). Default is "VIVID_STANDARD_30". |
| max_tiles | Integer. Safety cap on total tiles across all requested zoom levels to prevent accidental massive downloads. Default is 2500L. |
| title | Character string. Package title embedded in iteminfo. Default is "Vantor Satellite Basemap". |
| zip_for_suso | Logical. If TRUE, also creates a clean outer .zip archive (<stem>_tpk.zip or <stem>_tpkx.zip) using zip -q -X -j (stripping macOS __MACOSX/._* resource forks that Survey Solutions Headquarters rejects when users compress files via macOS Finder). Default is FALSE. |
| aoi | Optional alias for x (Area of Interest). |
| verbose | Logical. Print progress messages. Default is getOption("vantorr.verbose", TRUE). |

## Value

Invisibly returns the normalized `output_file` path, with attributes `n_tiles`, `zoom`, `format`, and `suso_zip_path` (when `zip_for_suso = TRUE`).

## Examples

```
if (requireNamespace("terra", quietly = TRUE)) {
  # Create a synthetic RGB SpatRaster in EPSG:3857
  r <- terra::rast(
    xmin = -11689500, xmax = -11688500,
    ymin = 4826500, ymax = 4827500,
    nrows = 128, ncols = 128, nlyrs = 3,
    crs = "EPSG:3857"
  )
  terra::values(r) <- sample(50:220, terra::ncell(r) * 3, replace = TRUE)

  # Export both .tpk (Compact Cache V1) and .tpkx (Compact Cache V2)
  tpk_file <- vantor_write_tpk(
    raster = r,
    output_file = tempfile(fileext = ".tpk"),
    zoom = 14:15,
    verbose = FALSE
  )
  tpkx_file <- vantor_write_tpk(
    raster = r,
    output_file = tempfile(fileext = ".tpkx"),
    zoom = 14:15,
    verbose = FALSE
  )
  unlink(c(tpk_file, tpkx_file))
}
```

# Index


add_vantor_basemap, 3
add_vantor_basemap(), *17*
add_vantor_imagery, 4
add_vantor_imagery(), *17*
add_vantor_seamlines, 6
add_vantor_seamlines(), *4*, *17*
addVantorBasemap *(*add_vantor_basemap*)*, 3
addVantorImagery *(*add_vantor_imagery*)*, 4
addVantorSeamlines
*(*add_vantor_seamlines*)*, 6

convert_bbox_to_3857, 8
create_maxar_request, 8

get_maxar_token, 9
get_maxar_token(), *27*
get_maxar_wms_basemap, 10
get_maxar_wms_basemap(), *12*, *17*, *19*, *24*
get_maxar_wms_tiles, 12
get_maxar_wms_tiles(), *17*, *19*

refresh_maxar_token, 13
run_vantor_demo_app, 14

tempdir(), *12*

vantor_as_bbox, 15
vantor_as_bbox(), *8*, *14*, *20*, *32*
vantor_auth, 16
vantor_auth(), *35*
vantor_auth_clear, 17
vantor_auth_server, 18
vantor_auth_server(), *20*
vantor_auth_ui, 19
vantor_auth_ui(), *14*, *18*
vantor_explorer_server, 19
vantor_explorer_server(), *18*
vantor_explorer_ui, 20
vantor_explorer_ui(), *14*, *19*
vantor_export_capi, 21
vantor_export_capi(), *24*, *29*
vantor_field_atlas, 23
vantor_field_atlas(), *31*
vantor_get_seamlines, 25
vantor_get_seamlines(), *7*, *17*, *22*, *29*, *32*
vantor_get_usage, 26
vantor_get_usage(), *17*, *19*
vantor_has_auth, 27
vantor_offline_package, 28
vantor_plus_code, 30
vantor_plus_code(), *21*
vantor_survey_grid, 31
vantor_survey_grid(), *21*, *22*, *24*, *29*, *32*
vantor_vintage_report, 32
vantor_wmts_url, 33
vantor_wmts_url(), *17*
vantor_write_tpk, 34
vantor_write_tpk(), *28*