

Package: ALSdownloaderR (via r-universe)

October 2, 2026

Title Discover, Inspect and Download Airborne Laser Scanning Data

Version 0.2.0

Description Discover, visualize and download airborne laser scanning data from an international catalogue through R functions and a 'shiny' application. Spatial searches identify point-cloud tiles intersecting an area of interest and retain acquisition metadata, source citations and licence links. Supports three-dimensional visualization, visual comparison, download planning reports and transfers of original files. Integrates supported provider services and configured spatial indexes, including approved community contributions. Distinguishes integrated access from external portals; availability depends on provider coverage.

License GPL-3

Contact Cesar Ivan Alvites Diaz <calvites1990@gmail.com>

URL <https://cesarito2021.r-universe.dev/ALSdownloaderR>,
https://github.com/Cesarito2021/als_downloader

BugReports https://github.com/Cesarito2021/als_downloader/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports xml2, sf, s2, httr, curl, jsonlite, digest, parallelly,
future, future.apply, shiny (>= 1.8.0), leaflet,
leaflet.extras, DT, callr, zip, stats, tools, utils

Suggests lidR, testthat (>= 3.0.0), knitr, rmarkdown, tinytex,
pkgload, openssl

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Roxygen list(markdown = TRUE)

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

Repository <https://cesarito2021.r-universe.dev>

Date/Publication 2026-10-02 13:46:32 UTC

RemoteUrl https://github.com/Cesarito2021/als_downloader

RemoteRef main

RemoteSha 05ce4f6d20d272f38400c7a135016de557d0f528

Contents

als_app	3
als_report	4
aoi_area	5
configure_reviewer_access	6
country_links	7
create_als_report	7
download_als_data	8
download_tiles	9
download_worker_policy	10
extract_als_dates	11
find_als_data	12
find_tiles	13
get_als_file_sizes	14
import_zenodo_inbox	15
inspect_zenodo	16
launch_app	16
prepare_zenodo_submission	18
preview_points	19
preview_remote_tile	19
provider_catalog	21
read_aoi	21
read_forest_preview	22
read_preview	23
read_tile_index	24
review_zenodo_submission	25
submit_zenodo	26
summarize_als_data	27
summarize_als_download	27
update_country_link	28
zenodo_submissions	29

Index	30
--------------	-----------

als_app	<i>Create the Shiny application object</i>
---------	--

Description

Create the Shiny application object

Usage

```
als_app(  
  mode = "local",  
  tile_index_dir = NULL,  
  provider_limit = 2L,  
  submission_dir = NULL,  
  reviewer = NULL  
)
```

Arguments

mode	"local" for configurable transfers or "hosted" for a serial download service. Set by the administrator, not by browser input.
tile_index_dir	Optional administrator-supplied OpenTopography or contributed index directory. Not exposed as a server path control in hosted mode.
provider_limit	Transfer concurrency ceiling after checking provider terms. Defaults to two; hosted mode always uses one.
submission_dir	Optional private directory for Zenodo proposals and approved indexes. No directory is created until a proposal is submitted.
reviewer	Optional trusted maintainer name; enables private review UI only in local mode with email-link authentication configured by configure_reviewer_access() . Never expose the reviewer app publicly.

Value

A shiny.appobj suitable for [shiny::runApp\(\)](#) or a deployment entry point.

Examples

```
if (interactive()) shiny::runApp(als_app())
```

als_report	<i>Generate an HTML or PDF summary of a search/download selection</i>
------------	---

Description

Generate an HTML or PDF summary of a search/download selection

Usage

```
als_report(
  tiles,
  output_dir,
  format = c("html", "pdf"),
  aoi_area_km2 = NA_real_,
  aoi = NULL,
  figures = character(),
  details = FALSE,
  map_image = character(),
  map_credits = "",
  figure_captions = NULL
)
```

Arguments

tiles	An sf or data frame as returned by find_tiles() or download_tiles() . Only used to summarise the selection; no network request is made and no source file is read.
output_dir	Directory to write the report into. Created if it does not already exist.
format	"html" (default; self-contained, requires Pandoc) or "pdf" (requires Pandoc and a working tinytex installation; falls back to "html" with a warning otherwise).
aoi_area_km2	Optional AOI size in square kilometres, shown for context only.
aoi	Optional sf polygon (the AOI) as passed to find_tiles() . When supplied alongside tiles with geometry, the report includes a simple map figure of the tile footprints and the AOI outline. Geometry only – not a basemap image, and nothing is fetched to draw it.
figures	Optional paths to up to six exported PNG figures, included unchanged with their embedded legends and credits. Maximum 10 MiB each.
details	Include a technical appendix with transfer-time scenarios and a sample file table. Defaults to FALSE for a concise visual report.
map_image	Optional single PNG path, used instead of the geometry-only map. The app captures a centred basemap with AOI and selected tiles.
map_credits	Basemap attribution accompanying map_image.
figure_captions	Optional short captions, one per attached figure.

Details

Content is limited to data already carried by tiles: filename, dataset, provider, provider-reported acquisition dates, known size and the recorded citation/licence per tile. Transfer-time scenarios use known bytes and illustrative aggregate speeds, not CPU core counts. Unknown file sizes are explicitly excluded. Nothing is fetched. The report is written only to output_dir and is never transmitted anywhere.

Value

Invisibly, the path to the rendered report file.

Examples

```
# Render a local planning report from synthetic metadata; no downloads.
# Rendering can take more than 5 seconds and requires Pandoc.

if (requireNamespace("rmarkdown", quietly = TRUE) && rmarkdown::pandoc_available()) {
  index <- system.file("extdata", "example-index", "demo.tiles.geojson",
    package = "ALSdownloaderR")
  tiles <- read_tile_index(index)
  output_dir <- tempfile("als-report-")
  report <- als_report(tiles, output_dir, format = "html")
  basename(report)
  file.exists(report)
  unlink(output_dir, recursive = TRUE)
}
```

aoi_area	<i>Calculate AOI size</i>
----------	---------------------------

Description

Calculate AOI size

Usage

```
aoi_area(aoi)
```

Arguments

aoi Polygon input accepted by `read_aoi()`.

Value

Geodesic union area in square kilometres, without double-counting overlapping polygons.

Examples

```
ring <- matrix(c(0,0, 1,0, 1,1, 0,1, 0,0), ncol = 2, byrow = TRUE)
aoi_area(sf::st_sfc(sf::st_polygon(list(ring))), crs = 4326))
```

```
configure_reviewer_access
```

Configure email-link access to local proposal review

Description

Configure email-link access to local proposal review

Usage

```
configure_reviewer_access(path, emails)
```

Arguments

path	Private allowlist-file path outside public assets and Git.
emails	Administrator mailboxes allowed to receive review invitations.

Details

Access requires a single-use private review link delivered to an allowed mailbox by the administrator-configured mail service. Preview mail cannot authenticate. Reconfiguration revokes sessions and outstanding invitations. Restrict the file's parent directory to the host owner (use Windows ACLs where applicable). The reviewer remains localhost-only; remote HTTPS hosting is separate.

Value

Invisibly, the allowlist-file path. No password is requested or stored. Set option `ALSdownloaderR.reviewer_credentials` or environment variable `ALS_REVIEWER_CREDENTIALS` to this path at launch.

Examples

```
# Create a temporary allowlist only; this does not send invitations or email.
if (requireNamespace("openssl", quietly = TRUE)) {
  path <- tempfile(fileext = ".rds")
  configure_reviewer_access(path, "reviewer@example.org")
  file.exists(path)
  unlink(path)
}
```

country_links	<i>List persistent country source links</i>
---------------	---

Description

List persistent country source links

Usage

```
country_links(path = country_links_path(), history = FALSE)
```

Arguments

path	Registry file. Defaults to the package's user configuration directory; override with option <code>ALSdownloadR.country_links</code> for a shared deployment.
history	Return previous revisions instead of current links.

Details

Reading does not create a file or contact a website. Country names follow the bundled world map. These links do not imply national ALS coverage or enable an in-app download adapter. Restart the app after updating links.

Value

A data frame of country, country_code, source_id, name, url, updated_on and notes. The initial records are the packaged external portals.

Examples

```
country_links()
```

create_als_report	<i>Create an ALS download planning report</i>
-------------------	---

Description

Create an ALS download planning report

Usage

```
create_als_report(...)
```

Arguments

...	Arguments passed to <code>als_report()</code> .
-----	---

Value

The generated report path, invisibly.

Examples

```
# Render a local planning report from synthetic metadata; no downloads.
# Rendering can take more than 5 seconds and requires Pandoc.

if (requireNamespace("rmarkdown", quietly = TRUE) && rmarkdown::pandoc_available()) {
  index <- system.file("extdata", "example-index", "demo.tiles.geojson",
    package = "ALSdownloaderR")
  tiles <- read_tile_index(index)
  output_dir <- tempfile("als-report-")
  report <- create_als_report(tiles, output_dir, format = "html")
  basename(report)
  file.exists(report)
  unlink(output_dir, recursive = TRUE)
}
```

download_als_data	<i>Download selected airborne laser scanning files</i>
-------------------	--

Description

Download selected airborne laser scanning files

Usage

```
download_als_data(...)
```

Arguments

... Arguments passed to download_tiles().

Value

A transfer status table, as returned by download_tiles().

Examples

```
# Requires an interactive session, Internet and your own polygon file.
# Provider availability and original tile sizes vary; allow more than 5 seconds.

if (interactive()) {
  aoi <- read_aoi(file.choose())
  tiles <- find_tiles(aoi, provider = "usgs3dep")
  if (nrow(tiles) > 0L) {
    output_dir <- tempfile("als-download-")
```

```

    result <- download_als_data(tiles[1, ], output_dir = output_dir, workers = 1)
    result[, c("status", "path")]
    # The downloaded file and its manifest remain in output_dir for inspection.
  }
}

```

download_tiles

*Download selected tiles with resumable transfer records***Description**

Download selected tiles with resumable transfer records

Usage

```

download_tiles(
  tiles,
  output_dir,
  workers = 1L,
  mode = c("local", "hosted"),
  provider_limit = 2L,
  retries = 2L,
  timeout = 600,
  progress_dir = NULL
)

```

Arguments

tiles	An sf or data frame returned by find_tiles() .
output_dir	Explicit writable destination directory. A directory is created only when this function is called.
workers	Requested transfer workers. Defaults to one for scripted use.
mode	"local" or "hosted". Hosted forces serial transfers.
provider_limit	Provider concurrency ceiling; defaults to two until an operator has verified the provider's applicable terms.
retries	Number of retries for transient failures. Authentication and rate-limit responses are not retried automatically.
timeout	Per-transfer timeout in seconds.
progress_dir	Optional directory for per-tile progress records.

Details

Uses `future.apply::future_lapply()` and restores the caller's future plan. Resume requires both a completed sidecar record and a matching file checksum. HTTP status, reported length and LAS signature are checked; complete point-stream decompression is not verified. Existing untracked destination files are never overwritten. Credentials are not written to the exported manifest. Large datasets should be downloaded locally.

Value

A data frame with one row per asset, transfer status, canonical source URL, byte count, checksum and local path. Failed tiles remain in the result.

Examples

```
# Requires an interactive session, Internet and your own polygon file.
# Provider availability and original tile sizes vary; allow more than 5 seconds.

if (interactive()) {
  aoi <- read_aoi(file.choose())
  tiles <- find_tiles(aoi, provider = "usgs3dep")
  if (nrow(tiles) > 0L) {
    output_dir <- tempfile("als-download-")
    result <- download_tiles(tiles[1, ], output_dir = output_dir, workers = 1)
    result[, c("status", "path")]
    # The downloaded file and its manifest remain in output_dir for inspection.
  }
}
```

download_worker_policy

Plan download concurrency

Description

Plan download concurrency

Usage

```
download_worker_policy(
  mode,
  available_cores = NA_integer_,
  requested = NULL,
  provider_limit = Inf,
  jobs = Inf
)
```

Arguments

mode	Deployment mode: "hosted" or "local".
available_cores	Available logical CPU cores. Missing values use one.
requested	User-selected worker count, or NULL for the recommendation.
provider_limit	Maximum simultaneous transfers allowed by the provider.
jobs	Number of pending tile transfers.

Details

Hosted mode uses one worker. Local mode leaves four available cores unused when possible and initially recommends at most ten workers. During automated package checks callers must explicitly request no more than two.

Value

A list containing recommended, maximum, requested and effective counts.

Examples

```
download_worker_policy("local", available_cores = 8)
download_worker_policy("hosted", available_cores = 16, requested = 4)
```

extract_als_dates	<i>Extract acquisition years and retain their evidence</i>
-------------------	--

Description

Extract acquisition years and retain their evidence

Usage

```
extract_als_dates(tiles, lookup = TRUE, lookup_budget = 60)
```

Arguments

tiles	An ALS asset table.
lookup	Whether to consult linked provider metadata.
lookup_budget	Maximum elapsed seconds before starting more metadata lookups.

Value

The table with acquisition_year, source, precision, scope and status. Missing values remain NA. Filename years are labelled unverified references.

Examples

```
# Synthetic catalogue values, read locally without contacting their URLs.
index <- system.file("extdata", "example-index", "demo.tiles.geojson",
                     package = "ALSdownloader")
tiles <- read_tile_index(index)
dated <- extract_als_dates(tiles, lookup = FALSE)
dated[, c("acquisition_year", "date_source", "date_status")]
```

find_als_data	<i>Find airborne laser scanning data</i>
---------------	--

Description

Find airborne laser scanning data

Usage

```
find_als_data(  
  aoi,  
  provider = c("usgs3dep", "planetary", "opentopography", "contributed", "ahn6",  
    "swisstopo", "ignfr", "canelevation"),  
  start = NULL,  
  end = NULL,  
  tile_index_dir = NULL,  
  max_items = 10000L  
)
```

Arguments

aoi	Polygon input accepted by <code>read_aoi()</code> .
provider	Either "usgs3dep" (USGS The National Map) or "planetary" (optional COPC mirror) or "opentopography" (verified hosted TileIndex catalogue), or "contributed" (maintainer-approved local *.tiles.geojson indexes), "ahn6" (native AHN6 index), "swisstopo" (swissSURFACE3D STAC), "ignfr" (IGN LiDAR HD, indexed through a public STAC catalogue maintained by UMR TETIS / INRAE; not IGN's own WFS), or "canelevation" (Canada; official NRCan spatial tile service, or a configured local .gpkg/.shp index).
start, end	Optional inclusive acquisition dates in YYYY-MM-DD format. Unknown acquisition dates remain in the results.
tile_index_dir	Directory of OpenTopography *_TileIndex.zip files, approved contributed *.tiles.geojson files, or CanElevation .gpkg/ .shp indexes. Required for contributed sources; optional for OpenTopography and Canada. Indexes require an embedded CRS and a url field pointing to the original file.
max_items	Maximum number of tiles to return. An incomplete result raises an error instead of silently reporting partial coverage.

Value

An sf asset table including acquisition evidence and file sizes when retrievable. The download summary is available with `summarize_als_download()`.

Examples

```
# Read a small synthetic catalogue supplied with the package; no Internet.
index_dir <- system.file("extdata", "example-index", package = "ALSdownloader")
index <- read_tile_index(file.path(index_dir, "demo.tiles.geojson"))
aoi <- read_aoi(sf::st_geometry(index))
# Search the local demonstration index and inspect the matching record.
tiles <- find_als_data(aoi, provider = "contributed", tile_index_dir = index_dir)
tiles[, c("tile_id", "acquired_start", "acquired_end", "size_bytes")]
# The example.org URLs are placeholders: do not download these demo records.
```

find_tiles

Find point-cloud tiles intersecting an AOI

Description

Find point-cloud tiles intersecting an AOI

Usage

```
find_tiles(
  aoi,
  provider = c("usgs3dep", "planetary", "opentopography", "contributed", "ahn6",
    "swisstopo", "ignfr", "canelevation"),
  start = NULL,
  end = NULL,
  tile_index_dir = NULL,
  max_items = 10000L
)
```

Arguments

aoi	Polygon input accepted by <code>read_aoi()</code> .
provider	Either "usgs3dep" (USGS The National Map) or "planetary" (optional COPC mirror) or "opentopography" (verified hosted TileIndex catalogue), or "contributed" (maintainer-approved local *.tiles.geojson indexes), "ahn6" (native AHN6 index), "swisstopo" (swissSURFACE3D STAC), "ignfr" (IGN LiDAR HD, indexed through a public STAC catalogue maintained by UMR TETIS / INRAE; not IGN's own WFS), or "canelevation" (Canada; official NRCAN spatial tile service, or a configured local .gpkg/.shp index).
start, end	Optional inclusive acquisition dates in YYYY-MM-DD format. Unknown acquisition dates remain in the results.
tile_index_dir	Directory of OpenTopography *_TileIndex.zip files, approved contributed *.tiles.geojson files, or CanElevation .gpkg/.shp indexes. Required for contributed sources; optional for OpenTopography and Canada. Indexes require an embedded CRS and a url field pointing to the original file.
max_items	Maximum number of tiles to return. An incomplete result raises an error instead of silently reporting partial coverage.

Details

Network access occurs only when this function is explicitly called for a remote source. OpenTopography searches the packaged, audited airborne-LiDAR registry by default and fetches matching provider indexes with SHA-256 verification. An explicit local index directory overrides that route in the R API. Public tile downloads do not require an API key. Asset licenses vary per dataset.

Value

An sf table of assets with stable identifiers, canonical URLs, acquisition dates, citation information and tile geometry in EPSG:4326. `campaign_id` contains the reported 3DEP project or IGN mission identifier. `campaign_group` and `campaign_basis` also describe fallback groups from recognized project directories, delivery blocks or datasets. Fallback groups do not establish distinct flight dates.

Examples

```
# Read a small synthetic catalogue supplied with the package; no Internet.
index_dir <- system.file("extdata", "example-index", package = "ALSdownloaderR")
index <- read_tile_index(file.path(index_dir, "demo.tiles.geojson"))
aoi <- read_aoi(sf::st_geometry(index))
# Search the local demonstration index and inspect the matching record.
tiles <- find_tiles(aoi, provider = "contributed", tile_index_dir = index_dir)
tiles[, c("tile_id", "acquired_start", "acquired_end", "size_bytes")]
# The example.org URLs are placeholders: do not download these demo records.
```

<code>get_als_file_sizes</code>	<i>Retrieve sizes of ALS files without downloading point clouds</i>
---------------------------------	---

Description

Retrieve sizes of ALS files without downloading point clouds

Usage

```
get_als_file_sizes(tiles, timeout = 15, budget = 60, connections = 4L)
```

Arguments

<code>tiles</code>	An ALS asset table.
<code>timeout</code>	Maximum seconds per HTTP request.
<code>budget</code>	Maximum seconds for each batch of requests.
<code>connections</code>	Maximum simultaneous connections (1 to 8).

Value

The asset table with `size_bytes` and size provenance/status. NA sizes are not zero. Requests use HEAD and, when necessary, a bounded byte range.

Examples

```
# Synthetic catalogue values, read locally without contacting their URLs.
index <- system.file("extdata", "example-index", "demo.tiles.geojson",
                     package = "ALSdownloaderR")
tiles <- read_tile_index(index)
# Sizes are already recorded, so no HTTP lookup is needed.
sized <- get_als_file_sizes(tiles)
sized[, c("size_bytes", "size_source", "size_status")]
```

import_zenodo_inbox	<i>Rebuild an inbox contribution in the private review queue</i>
---------------------	--

Description

Rebuild an inbox contribution in the private review queue

Usage

```
import_zenodo_inbox(fields, queue)
```

Arguments

fields	Named list of fields received through the configured submission inbox. Untrusted metadata are re-fetched from Zenodo; links are not followed unless they exactly match a file in that record.
queue	Administrator-controlled persistent private queue directory.

Details

This maintainer operation supports the link-only Formspree fields generated by ALSdownloaderR. It retrieves Zenodo metadata and, for polygon coverage, one bounded boundary file. It never downloads point clouds. Access to the inbox is configured separately; no inbox API key is bundled.

Value

Pending proposal ID. Repeated imports are deduplicated. No approval or publication is performed.

Examples

```
# Maintainer-only: requires an actual inbox export, Internet and a private queue.
# The JSON file must contain the named fields received by the submission inbox.
## Not run:
fields <- jsonlite::read_json(file.choose(), simplifyVector = TRUE)
queue <- readline("Private review queue directory: ")
id <- import_zenodo_inbox(fields, queue)
zenodo_submissions(queue)

## End(Not run)
```

inspect_zenodo	<i>Inspect a public Zenodo record without downloading point clouds</i>
----------------	--

Description

Inspect a public Zenodo record without downloading point clouds

Usage

```
inspect_zenodo(link)
```

Arguments

link	Published Zenodo record URL, DOI or numeric record ID.
------	--

Details

Only retrieves a bounded JSON metadata response from zenodo.org. Publication dates are never treated as acquisition dates. No catalogue changes.

Value

Metadata including authors, licence, description, file sizes and links.

Examples

```
# Requires Internet and a published, openly accessible Zenodo record.

if (interactive()) {
  link <- readline("Published Zenodo record URL or DOI: ")
  metadata <- inspect_zenodo(link)
  metadata[c("title", "doi", "license_url")]
}
```

launch_app	<i>Launch the ALSdownloaderR Shiny application</i>
------------	--

Description

Launch the ALSdownloaderR Shiny application

Usage

```
launch_app(
  mode = c("local", "hosted"),
  tile_index_dir = NULL,
  provider_limit = 2L,
  host = "127.0.0.1",
  port = NULL,
  launch.browser = interactive(),
  submission_dir = NULL,
  reviewer = NULL
)
```

Arguments

mode	"local" for configurable transfers or "hosted" for a serial download service. Set by the administrator, not by browser input.
tile_index_dir	Optional administrator-supplied OpenTopography or contributed index directory. Not exposed as a server path control in hosted mode.
provider_limit	Transfer concurrency ceiling after checking provider terms. Defaults to two; hosted mode always uses one.
host, port	Bind address and optional port passed to shiny::runApp() .
launch.browser	Whether to open a browser. Defaults to interactive use.
submission_dir	Optional private directory for Zenodo proposals and approved indexes. No directory is created until a proposal is submitted.
reviewer	Optional trusted maintainer name; enables private review UI only in local mode with email-link authentication configured by configure_reviewer_access() . Never expose the reviewer app publicly.

Details

Launch is explicit: loading the package does not start a browser, contact providers, change the working directory or install packages. Background transfer processes are terminated when the owning session ends. Optional browser submissions use option `ALSdownloader.formsfree`, a list with endpoint (a public <https://formsfree.io/f/> form URL) and attachments (FALSE by default; TRUE requires a file-upload plan). Alternatively set environment variable `ALS_FORMSPREE_ENDPOINT` for links-only submissions. The community form is the default when no local submission directory is set. Set option `ALSdownloader.formsfree = FALSE` to disable remote submission. An explicitly configured transport takes precedence over the local queue button. It requires explicit submission and Internet access, but no contributor account. Service quotas apply; receiving a proposal never approves its publication.

Value

Invisibly, the return value of [shiny::runApp\(\)](#). Runs until stopped.

Examples

```
if (interactive()) launch_app()
```

```
prepare_zenodo_submission
```

Prepare a Zenodo contribution for maintainer review

Description

Prepare a Zenodo contribution for maintainer review

Usage

```
prepare_zenodo_submission(
  link,
  boundary,
  acquired = "",
  platform = "ALS",
  email = ""
)
```

Arguments

link	Public Zenodo DOI or record link.
boundary	GeoJSON, GeoPackage, zipped Shapefile or sf polygons. For multiple assets include file_key with exact Zenodo filenames.
acquired	Collection year, year range or YYYY-MM-DD / YYYY-MM-DD; blank means unknown and never substitutes the publication date.
platform	ALS or UAV-LiDAR, confirmed by the contributor.
email	Optional private contact email, excluded from approved indexes.

Value

A pending proposal; no files are published and no point clouds read.

Examples

```
# Requires Internet and coverage polygons matching the files in your record.
# This prepares a proposal but does not submit it or approve its publication.

if (interactive()) {
  link <- readline("Published Zenodo record URL or DOI: ")
  boundary <- read_aoi(file.choose())
  proposal <- prepare_zenodo_submission(link, boundary, acquired = "", platform = "ALS")
  proposal$id
}
```

preview_points	<i>Make a bounded point-cloud preview</i>
----------------	---

Description

Make a bounded point-cloud preview

Usage

```
preview_points(points, max_points = 100000L)
```

Arguments

points	A data frame or matrix with numeric X, Y, Z columns.
max_points	Maximum displayed points, between one and one million.

Details

Sampling is deterministic and affects only the returned preview. The source cloud is not modified. Elevation is not normalized canopy height.

Value

A data frame of sampled coordinates with an origin attribute. Coordinates are translated to a local origin for display precision. An optional numeric Classification column is retained as an aligned classification attribute; missing or invalid codes become NA. Optional numeric Intensity is retained as an aligned intensity attribute. No classification or intensity is inferred.

Examples

```
preview_points(data.frame(X = 1:10, Y = 1:10, Z = 11:20), 5)
```

preview_remote_tile	<i>Download one remote tile temporarily and read a bounded preview</i>
---------------------	--

Description

Download one remote tile temporarily and read a bounded preview

Usage

```
preview_remote_tile(
  tile,
  max_bytes = 1024 * 1024^2,
  path = tempfile(fileext = ".laz"),
  reader = function(path) read_preview(path, 100000L),
  progress = NULL,
  zip_member = NULL,
  max_uncompressed_bytes = 2 * 1024^3
)
```

Arguments

tile	A single-row tile data frame, as returned by find_tiles() .
max_bytes	Maximum known HTTP file size accepted for the temporary download, in bytes.
path	Temporary local path for the downloaded file. Removed after reader runs or on error.
reader	Function called with path once the download completes; defaults to read_preview() . Used by read_forest_preview() for the app's forest close-up preview.
progress	Optional function receiving short preview-stage messages.
zip_member	Optional exact LAS/LAZ member name when a ZIP contains multiple point clouds. A single point cloud is selected automatically.
max_uncompressed_bytes	Maximum total uncompressed ZIP size, in bytes.

Details

A temporary, single-file preview; never persists a cloud in the source catalog. Exported (rather than internal-only) so a background [callr::r_bg\(\)](#) worker can call it by namespace-qualified name – a plain closure passed as a callr argument can lose access to sibling package-internal helpers it calls, such as [require_data_terms\(\)](#).

Value

Whatever reader returns.

Examples

```
# Requires an interactive session, Internet and your own polygon file.
# Provider availability and original tile sizes vary; allow more than 5 seconds.

if (interactive() && requireNamespace("lidR", quietly = TRUE)) {
  aoi <- read_aoi(file.choose())
  tiles <- find_tiles(aoi, provider = "usgs3dep")
  if (nrow(tiles) > 0L) {
    # Downloads one original tile temporarily (up to 10 MiB here).
    points <- preview_remote_tile(tiles[1, ], max_bytes = 10 * 1024^2)
    head(points)
  }
}
```

```
}
}
```

provider_catalog	<i>Inspect provider access and implementation status</i>
------------------	--

Description

Inspect provider access and implementation status

Usage

```
provider_catalog()
```

Value

A data frame with provider identifiers, access notes, source links, and whether a tile-search adapter is implemented. Implemented does not mean that every dataset or endpoint is currently available.

Examples

```
provider_catalog()
```

read_aoi	<i>Read and validate a polygon AOI</i>
----------	--

Description

Read and validate a polygon AOI

Usage

```
read_aoi(x, layer = NULL)
```

Arguments

x	An sf or sfc polygon, a path to a vector file, or a Shiny upload object containing name and datapath.
layer	Layer name for a multi-layer GeoPackage. Required when there is more than one layer.

Details

Zippped Shapefiles must contain one Shapefile with a declared CRS. Archives with unsafe paths or more than 200 MB uncompressed are rejected. No coordinate reference system is guessed.

Value

An sf polygon object in EPSG:4326. Empty geometries are removed.

Examples

```
ring <- matrix(c(0,0, 1,0, 1,1, 0,1, 0,0), ncol = 2, byrow = TRUE)
aoi <- read_aoi(sf::st_sfc(sf::st_polygon(list(ring))), crs = 4326)
```

read_forest_preview	<i>Read a decimated, spatially windowed preview for the forest close-up viewer</i>
---------------------	--

Description

Read a decimated, spatially windowed preview for the forest close-up viewer

Usage

```
read_forest_preview(
  path,
  percent = 2,
  window = 100,
  center_x = 50,
  center_y = 50,
  voxel = 0,
  xy_units = "auto",
  z_units = "auto"
)
```

Arguments

path	Local LAS/LAZ file path.
percent	Approximate percentage of source points the reader keeps, between 0.1 and 100. The reader pool is also capped at 750,000 points.
window, center_x, center_y, voxel	Display windowing and optional voxel thinning; see <code>forest_display_sample()</code> .
xy_units, z_units	Source horizontal and elevation units: auto, m, ft (international foot) or us-ft (US survey foot). Explicit choices require confirmation from source documentation. Both must be known to scale display coordinates to metres; unknown units remain unchanged.

Details

Requires the optional package 'lidR'. Exported so a background `callr::r_bg()` worker can call it by namespace-qualified name; see `preview_remote_tile()` for why.

Value

The same translated coordinate table as `preview_points()`, with a `display_note` attribute describing the sampling actually used.

Examples

```
# Use a small LAS/LAZ example distributed with the optional lidR package.
if (requireNamespace("lidR", quietly = TRUE)) {
  path <- system.file("extdata", "MixedConifer.laz", package = "lidR")
  points <- read_forest_preview(path, percent = 1)
  head(points)
  attr(points, "origin")
  attr(points, "units")
}
```

read_preview	<i>Read a small preview from a local LAS or LAZ file</i>
--------------	--

Description

Read a small preview from a local LAS or LAZ file

Usage

```
read_preview(path, max_points = 100000L, xy_units = "auto", z_units = "auto")
```

Arguments

<code>path</code>	Local point-cloud filename.
<code>max_points</code>	Maximum preview size, between one and one million.
<code>xy_units, z_units</code>	Source horizontal and elevation units: <code>auto</code> , <code>m</code> , <code>ft</code> (international foot) or <code>us-ft</code> (US survey foot). Explicit choices require confirmation from source documentation. Both must be known to scale display coordinates to metres; unknown units remain unchanged.

Details

Requires the optional package 'lidR'. Reader-side decimation limits retained points before they enter R. The file may still need to be scanned in full; this function is not a remote COPC range reader.

Value

The same translated coordinate table as `preview_points()`.

review_zenodo_submission

Accept or reject a queued Zenodo contribution

Description

Accept or reject a queued Zenodo contribution

Usage

```
review_zenodo_submission(
  queue,
  id,
  decision = c("approve", "reject"),
  reviewer,
  confirmed = FALSE,
  reason = ""
)
```

Arguments

queue	Administrator-controlled private queue directory.
id	Proposal ID.
decision	approve or reject.
reviewer	Maintainer name for the audit record.
confirmed	Explicit confirmation of coverage/file mapping, dates, aerial LiDAR content, attribution and licence suitability. Required to approve.
reason	Optional private review note.

Details

Approval re-fetches Zenodo metadata and rejects changed records. Only approved indexes under queue/approved participate in catalogue search. Never downloads point clouds. Expose this operation only to trusted maintainers.

Value

Path to an approved index, or invisibly NULL on rejection.

Examples

```
# Maintainer-only example: records a real rejection in an existing private queue.
# Requires a pending proposal and an explicit review decision; not run automatically.
## Not run:
queue <- readline("Private review queue directory: ")
zenodo_submissions(queue)
```

```

id <- readline("ID of the proposal you have reviewed: ")
reviewer <- readline("Your maintainer name: ")
review_zenodo_submission(queue, id, decision = "reject", reviewer = reviewer,
                        reason = "Coverage or metadata needs correction.")

## End(Not run)

```

submit_zenodo

Submit a proposal to a private local review queue

Description

Submit a proposal to a private local review queue

Usage

```
submit_zenodo(proposal, queue)
```

Arguments

proposal	Result of prepare_zenodo_submission().
queue	Administrator-controlled private queue directory.

Details

Does not activate coverage or download data. Optional administrator configuration in option ALSdownloader.submission_m enables notifications; see the notification setup guide in the repository documentation.

Value

Proposal ID. Identical metadata/coverage proposals are deduplicated.

Examples

```

# Requires a real record, matching coverage and an administrator-owned queue.
# Run only when intending to submit; configured notifications may send email.
## Not run:
link <- readline("Published Zenodo record URL or DOI: ")
boundary <- read_aoi(file.choose())
proposal <- prepare_zenodo_submission(link, boundary, acquired = "", platform = "ALS")
queue <- readline("Private review queue directory: ")
id <- submit_zenodo(proposal, queue)
zenodo_submissions(queue)

## End(Not run)

```

summarize_als_data	<i>Summarize an ALS data selection</i>
--------------------	--

Description

Summarize an ALS data selection

Usage

```
summarize_als_data(tiles)
```

Arguments

tiles	An ALS asset table.
-------	---------------------

Value

Counts and total download size; incomplete totals remain NA.

Examples

```
# Synthetic catalogue values, read locally without contacting their URLs.  
index <- system.file("extdata", "example-index", "demo.tiles.geojson",  
                     package = "ALSdownloaderR")  
tiles <- read_tile_index(index)  
summarize_als_data(tiles)  
# Unknown sizes remain unknown rather than being counted as zero.  
tiles$size_bytes <- NA_real_  
summarize_als_data(tiles)
```

summarize_als_download	<i>Summarize the total download size of a selection</i>
------------------------	---

Description

Summarize the total download size of a selection

Usage

```
summarize_als_download(tiles)
```

Arguments

tiles	An ALS asset table, optionally enriched with <code>get_als_file_sizes()</code> .
-------	--

Value

A one-row data frame with file count, known bytes, missing sizes and complete total bytes (NA when any size is missing). GB uses 10^9 bytes.

Examples

```
# Synthetic catalogue values, read locally without contacting their URLs.
index <- system.file("extdata", "example-index", "demo.tiles.geojson",
                     package = "ALSdownloaderR")
tiles <- read_tile_index(index)
summarize_als_download(tiles)
# Unknown sizes remain unknown rather than being counted as zero.
tiles$size_bytes <- NA_real_
summarize_als_download(tiles)
```

update_country_link	<i>Add or update an external ALS source link for a country</i>
---------------------	--

Description

Add or update an external ALS source link for a country

Usage

```
update_country_link(
  country,
  url,
  updated_on,
  source_id = "country_portal",
  name = "ALS source portal",
  notes = "External access; consult source coverage and terms.",
  path = country_links_path()
)
```

Arguments

country	Country name (case-insensitive) or numeric country code from the bundled world map, for example "Italy" or 380.
url	An HTTPS source portal URL. It is recorded, not fetched or verified.
updated_on	Date supplied by the maintainer, in YYYY-MM-DD format. This is the link review date, not a LiDAR acquisition date.
source_id	Stable source identifier within a country. Use an existing identifier from country_links() to replace that source's URL.
name	Display name for the portal.
notes	Coverage and access notes. A regional portal is not national coverage.
path	Registry file. Defaults to the package's user configuration directory; override with option ALSdownloaderR.country_links for a shared deployment.

Details

Updates only the matching country/source pair; other entries remain. Prior revisions are retained in the registry and the previous file is saved with a .bak suffix. A lock rejects simultaneous writers. Registered links appear in the yellow External Access map layer and source catalogue after restarting the app, including countries with separate red survey footprints. No package installation files are modified. Share the registry path with the app process to use the same links in a hosted installation.

Value

The updated current country-link table, invisibly.

Examples

```
registry <- tempfile(fileext=".rds")
update_country_link("Italy", "https://example.org/lidar", "2026-09-22",
  name="Example portal (not a verified source)", path=registry)
country_links(path=registry)
unlink(c(registry, paste0(registry, ".bak")))
```

zenodo_submissions	List pending and reviewed Zenodo contributions
--------------------	--

Description

List pending and reviewed Zenodo contributions

Usage

```
zenodo_submissions(queue)
```

Arguments

queue Administrator-controlled private queue directory.

Value

A table of request IDs, DOI, title and status. Private contacts are omitted.

Examples

```
# A new private queue has no submissions; no network or files are created.
queue <- tempfile("als-review-")
zenodo_submissions(queue)
```

Index

als_app, [3](#)
als_report, [4](#)
aoi_area, [5](#)

callr::r_bg(), [20](#), [22](#)
configure_reviewer_access, [6](#)
configure_reviewer_access(), [3](#), [17](#)
country_links, [7](#)
create_als_report, [7](#)

download_als_data, [8](#)
download_tiles, [9](#)
download_tiles(), [4](#), [24](#)
download_worker_policy, [10](#)

extract_als_dates, [11](#)

find_als_data, [12](#)
find_tiles, [13](#)
find_tiles(), [4](#), [9](#), [20](#)

get_als_file_sizes, [14](#)

import_zenodo_inbox, [15](#)
inspect_zenodo, [16](#)

launch_app, [16](#)

prepare_zenodo_submission, [18](#)
preview_points, [19](#)
preview_points(), [23](#)
preview_remote_tile, [19](#)
preview_remote_tile(), [22](#)
provider_catalog, [21](#)

read_aoi, [21](#)
read_aoi(), [5](#), [12](#), [13](#)
read_forest_preview, [22](#)
read_forest_preview(), [20](#)
read_preview, [23](#)
read_preview(), [20](#)

read_tile_index, [24](#)
review_zenodo_submission, [25](#)

shiny::runApp(), [3](#), [17](#)
submit_zenodo, [26](#)
summarize_als_data, [27](#)
summarize_als_download, [27](#)

update_country_link, [28](#)

zenodo_submissions, [29](#)