Package 'planisphere'

July 7, 2026

Title Map Projections

Version 0.1.0

Description Applies cartographic projections to spatial data frames containing geographic coordinates. Projection methods are based on the 'D3.js' ecosystem <doi:10.1109/TVCG.2011.185> and use spherical geometry rather than ellipsoidal geodesic models.

License GPL (>= 3)

URL https://riatelab.github.io/planisphere/

BugReports https://github.com/riatelab/planisphere/issues/

Encoding UTF-8

Depends R (>= 4.1.0)

Imports utils, sf, V8, geojsonsf, graphics

Suggests knitr, r2d3

Language en-US

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Nicolas Lambert [aut, cre] (ORCID: <https://orcid.org/0000-0003-4976-6560>)

Maintainer Nicolas Lambert <nicolas.lambert@cnrs.fr>

Repository CRAN

Date/Publication 2026-07-07 09:30:02 UTC

Contents

`display` *Display a projected planisphere*

Description

Render a complete planisphere from a projected object created with `project()`. The display includes the projected sphere outline, graticule, and geographic features.

Usage

```
display(x, title = NULL, extent = NULL)
```

Arguments

`x`	A list returned by `project()`, containing three `sf` objects: • `basemap`: projected geographic features • `sphere`: outline of the projected globe • `graticule`: projected latitude/longitude grid
`title`	Character string. Optional title to add to the plot. Default is NULL.
`extent`	object. A spatial extent used to clip or define the mapping area (‘sf‘, ‘sfc‘, or ‘bbox‘).

Details

The visualization follows the D3.js spherical projection model used in the package. The rendering order is: 1. Sphere background 2. Graticule 3. Projected landmass (basemap) 4. Sphere outline

All geometries are expected to be already projected in spherical coordinates. This function is intended for fast visualization and exploration of D3-based projections.

Value

a base R plot.

Examples

```
library(sf)
world <- st_read(
  system.file("gpkg/land.gpkg", package = "planisphere"),
  quiet = TRUE
)

result <- planisphere::project(
  x = world,
  proj = "Eckert1",
  additional_layers = TRUE
)
planisphere::display(result)
```

`gallery` *Display a gallery of map projections*

Description

Creates a gallery of projected world maps using the projections available in **planisphere**. By default, a random sample of projections is selected from the projection registry and displayed in a multi-panel layout.

Usage

```
gallery(
  projections = NULL,
  sample = 12,
  ncol = 4,
  title = TRUE,
  verbose = TRUE
)
```

Arguments

`projections`	Character vector of projection names. If ‘NULL‘, projections are selected from ‘registry()‘.
`sample`	Integer. Number of projections to display when ‘projections = NULL‘. Ignored if ‘projections‘ is provided. Set to ‘NULL‘ to display all available projections.
`ncol`	Integer. Number of columns in the gallery layout.
`title`	Logical. Should projection names be displayed as titles?
`verbose`	Logical. Display progress messages while rendering projections.

Details

The function uses the bundled world land dataset and applies each selected projection through [project()]. Maps are displayed using [display()] in a base R graphics layout.

Value

Draws a gallery of projected maps in the current graphics device and returns ‘NULL‘ invisibly.

Examples

```
# Display a random sample of 12 projections
gallery(verbose = FALSE)

# Display selected projections
gallery(
  projections = c(
    "Mercator",
```

```
    "Robinson",
    "Orthographic",
    "EqualEarth"
  ),
  ncol = 2,
  verbose = FALSE
)
```

`new_v8_context` *New V8 context*

Description

Create a new V8 JavaScript context preloaded with the D3.js geospatial stack.

Usage

```
new_v8_context(
 libs = c("https://cdn.jsdelivr.net/npm/d3@7", "https://cdn.jsdelivr.net/npm/d3-geo@3",
    "https://cdn.jsdelivr.net/npm/d3-geo-polygon@2",
    "https://cdn.jsdelivr.net/npm/d3-geo-projection@4"),
  verbose = TRUE
)
```

Arguments

`libs`	Character vector of JavaScript library URLs to load. By default, libraries are loaded from the jsDelivr CDN. Users may replace these with local files, pinned versions, or alternative CDNs.
`verbose`	Logical. If TRUE, prints loading status for each library.

Details

This function initializes a fresh V8 JavaScript runtime and automatically loads the latest versions of the required D3.js libraries from the jsDelivr CDN.

The following JavaScript libraries are loaded by default:

- d3 (core library): https://cdn.jsdelivr.net/npm/d3 - d3-geo (spherical geometry and geographic projections): https://cdn.jsdelivr.net/npm/d3-geo - d3-geo-projection (extended cartographic projections): https://cdn.jsdelivr.net/npm/d3-geo-projection - d3-geo-polygon (spherical polygon clipping): https://cdn.jsdelivr.net/npm/d3-geo-polygon

These libraries are fetched dynamically at runtime from the jsDelivr CDN, ensuring that the most recent compatible versions are used unless explicitly overridden via the ‘libs‘ argument.

All projections computed in this environment rely on D3.js spherical geometry, where the Earth is modeled as a sphere rather than an ellipsoid. This allows fast and consistent computation of map projections and geometric transformations.

Value

A V8 JavaScript context with the D3.js geospatial stack loaded and ready for cartographic computations.

Examples

```
ct <- planisphere::new_v8_context()
# Then:
library(sf)
world <- st_read(
  system.file("gpkg/land.gpkg", package = "planisphere"),
  quiet = TRUE
)
out <- planisphere::project(x = world, proj = "InterruptedBoggs", ct = ct)
```

---

planisphere *Package description*

Description

A wrapper around D3.js for cartographic projections to design and visualize world maps (planispheres). Projections are computed using D3.js spherical geometry rather than ellipsoidal geodesic models.

Author(s)

Maintainer: Nicolas Lambert <nicolas.lambert@cnrs.fr> (ORCID)

Authors:

- Nicolas Lambert <nicolas.lambert@cnrs.fr> (ORCID)

See Also

Useful links:

- https://riatelab.github.io/planisphere/
- Report bugs at https://github.com/riatelab/planisphere/issues

`project` *Project geographic data using a D3.js projection*

Description

Projects spatial data using D3.js geographic projections executed in a V8 JavaScript context. The projection is performed in spherical coordinates, following D3.js design principles, where the Earth is modeled as a sphere rather than an ellipsoid.

Usage

```
project(
  x,
  proj = "geoAzimuthalEqualArea",
  rotate = NULL,
  reflectX = NULL,
  reflectY = NULL,
  scale = 500,
  center = NULL,
  parallel = NULL,
  parallels = NULL,
  clipExtent = NULL,
  clip = TRUE,
  graticule = c(10, 10),
  additional_layers = FALSE,
  verbose = FALSE,
  ct = .planisphere$ct,
  ...
)
```

Arguments

`x`	An `sf` spatial dataframe to project.
`proj`	A D3 projection name or constructor (e.g. "geoInterruptedHomolosine").
`rotate`	Rotation parameters passed to the projection.
`reflectX`	Logical; whether to reflect the projection on the X axis.
`reflectY`	Logical; whether to reflect the projection on the Y axis.
`scale`	Projection scale factor (D3 spherical scale, not planar CRS units).
`center`	Optional projection center.
`parallel`	Optional standard parallel of the projection
`parallels`	Optional standard parallels of the projection
`clipExtent`	Projection's viewport clip extent to the specified bounds in pixels
`clip`	If TRUE, clips the projected geometries to the projected sphere.
`graticule`	Numeric vector of longitude/latitude step size for graticule generation.

additional_layers
: Logical. If TRUE, adds graticule and sphere layers. In this case, the function returns a list. If FALSE (default), it returns a spatial data frame.

verbose
: Logical. Display messages

ct
: A custom V8 JavaScript context if needed. See `new_v8_context()`.

...
: Additional parameters passed to the projection builder.

Details

D3 geographic projections are based on spherical geometry. All transformations assume a perfect sphere, which differs from traditional GIS pipelines that often rely on ellipsoidal datums (e.g., WGS84). As a result, this function is primarily intended for visualization and cartographic rendering rather than high-precision geodetic computation.

Value

If `additionalLayers = FALSE`, a single `sf` object corresponding to the projected basemap.

If `additionalLayers = TRUE`, a list of `sf` objects:

- `basemap`: projected input geometries
- `sphere`: outline of the projected globe
- `graticule`: projected graticule lines

Examples

```
library(sf)
world <- st_read(
  system.file("gpkg/land.gpkg", package = "planisphere"),
  quiet = TRUE
)

result <- planisphere::project(x = world, proj = "InterruptedBoggs")
```

---

registry *List available map projections*

Description

Returns the list of available map projections registered in **planisphere**. Projections are organized into thematic families to facilitate exploration and selection.

By default, all projections are returned as a single sorted character vector. Alternatively, a specific family of projections can be requested.

Usage

```
registry(type = NULL)
```

Arguments

type	Character. Optional projection family. Must be one of: "cylindrical", "conic", "azimuthal", "perspective", "pseudocylindrical", "interrupted", "polyhedral", "quincuncial", "regional", "pseudoazimuthal", "miscellaneous".

Value

A sorted unique character vector of projection names. If ‘type‘ is provided, only projections belonging to the selected family are returned.

Examples

```
# List all available projections

# List cylindrical projections
registry(type = "cylindrical")

# Explore regional projections
registry("regional")

# Visualize polyhedral projections
gallery(projections = registry("polyhedral"), verbose = FALSE)
```

Index