Package 'rcanvec'

October 14, 2022

Type Package

Title Access and Plot CanVec and CanVec+ Data for Rapid Basemap Creation in Canada

Version 0.2.1

Date 2016-09-21

Author Dewey Dunnington <dewey@fishandwhistle.net>

Maintainer Dewey Dunnington <dewey@fishandwhistle.net>

Description Provides an interface to the National Topographic System (NTS), which is the way in which a number of freely available Canadian datasets are organized. CanVec and CanVec+ datasets, which include all data used to create Canadian topographic maps, are two such datasets that are useful in creating vector-based maps for locations across Canada. This packages searches CanVec data by location, plots it using pretty defaults, and exports it to human-readable shapefiles for use in another GIS.

License GPL-2

Depends R (>= 2.10), sp

Imports rgdal

URL https://github.com/paleolimbot/rcanvec

BugReports https://github.com/paleolimbot/rcanvec/issues

LazyData true

Suggests prettymapr, testthat

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2016-09-22 23:01:11

R topics documented:

`rcanvec-package` *Find, load, and plot Canadian vector basemap data*

Description

Provides an interface to the National Topographic System (NTS), which is the way in which several a number of freely available Canadian datasets are available. CanVec and CanVec+ datasets, which include all data used to create Canadian topographic maps, are two such items that are useful in creating vector-based maps for locations across Canada.

Details

This package provides access to the CanVec/CanVec+ datasets via the canvec.download function. CanVec data is organized by National Topographic System number (e.g. 021H01), so the nts function is provided to look up NTS number by location (e.g. lat/lon), parsed text (e.g. "21h1"), or bounding box. Searching a bounding box using `prettymapr::searchbbox` based on human-readable location may be helpful.(e.g. `prettymapr::searchbbox("wolfville ns")`). Using the canvec.qplot function, this data can be plotted using default options for standard layers. Combining the `searchbbox()` and canvec.qplot functions, it is possible to make a vector-based map of any location in Canada with one incredible super simple line of code (data is downloaded automatically): `canvec.qplot(bbox=searchbbox("wolfville ns"))`. For prettier maps, the canvec.export function exports shapefiles in a human-readable format (e.g. building_021H01.shp). If more refined

plotting or manipulation is desired, further functions are available to manually load or obtain filenames of cached files. Files are downloaded to the working directory by default, although this location can be defined using the `cachedir` argument available to many functions.

Author(s)

Dewey Dunnington <dewey@fishandwhistle.net>

References

CanVec+ Product Specifications, National Topographic System (NTS) Documentation

Examples

```
library(prettymapr)

#nts() function generates nts references based on lat, lon, or
#bounding coordinates
nts('21h')
nts('21h1')
nts('21h1', '21a16', '21A15')
nts(lat=45.2, lon=-64.32)
nts(lat=c(45.2, 46.2), lon=c(-64.32, -64.81))
nts(bbox=makebbox(45.125, -64.25, 44.875, -64.75))

#variant ntsstring() converts nts to string formats or takes the same
#arguments as nts() and returns a string vector instead.
ntsstring(c("021", "H", "01"))
ntsstring(bbox=makebbox(45.125, -64.25, 44.875, -64.75))

#bbox functions from {prettymapr} make it easy to manipulate bounding boxes
wolfville <- searchbbox("wolfville ns", source="google")
wolfvillezoomedout <- zoombbox(wolfville, 0.5)

#easy plotting with canvec.qplot()
canvec.qplot(bbox=searchbbox("wolfville ns", source="google"))

#download canvec or canvec+ data. 250k references use canvec+ (large amounts of data)
#and 50k references use canvec data (older but distributed in smaller chunks).
canvec.download(nts('21h1'))

#load data
buildings <- canvec.load(nts("21h1"), "building")
lakes <- canvec.load(nts("21h1"), "waterbody")
rivers <- canvec.load(nts('21h1'), "river")
roads <- canvec.load(nts('21h1'), "road")
contours <- canvec.load(nts('21h1'), "contour")

#plot data
sp::plot(lakes, col="lightblue", border="lightblue")
sp::plot(rivers, add=TRUE, col="lightblue")
sp::plot(buildings, add=TRUE, pch=".")
```

```
#zoomed in
sp::plot(lakes, col="lightblue", border="lightblue",
     xlim=c(-64.4,-64.35), ylim=c(45.05,45.1))
sp::plot(contours, add=TRUE, col="brown", lwd=0.2)
sp::plot(rivers, add=TRUE, col="lightblue")
sp::plot(buildings, add=TRUE, pch=".")
sp::plot(roads, add=TRUE, lwd=0.5)


#equivalent syntax in canvec.qplot()
canvec.qplot(nts("21h1"), layers=c("waterbody", "contour", "river", "road"))
canvec.qplot(bbox=makebbox(45.1, -64.35, 45.05, -64.4),
            layers=c("waterbody", "contour", "river", "building", "road"))

#method returns plot data argument so data does not need to be loaded each time.
#this will not work when changing nts sheets.
plotdata <- canvec.qplot(nts("21h1"), layers=c("waterbody", "contour", "river"))
plotdata <- canvec.qplot(bbox=makebbox(45.1, -64.35, 45.05, -64.4),
                         layers=c("waterbody", "contour", "river"),
                         data=plotdata)

#easy exporting with human readable names
canvec.export(nts("21h01"), "~/canvecdata", layers=c("road", "river"))
```

`canvec.cachedir` *Get Cache Directory*

Description

Get the default cache directory, which is the folder rcanvec.cache in the current working directory. Modify this behaviour by passing a `cachedir` argument to `canvec.download()`, `canvec.load()`, or `canvec.qplot()`.

Usage

```
canvec.cachedir()
```

Value

A character string of the cache directory path

canvec.cleanup *Remove CanVec Data Files*

Description

Deletes files downloaded by `canvec.download()`. Use `all=TRUE` to remove the cache directory entirely.

Usage

```
canvec.cleanup(ntsid = NULL, cachedir = NULL, all = FALSE,
  keeparchives = FALSE, keepfolders = FALSE)
```

Arguments

`ntsid`	One or more NTS References as generated by `nts()`
`cachedir`	The same `cachedir` that was passed to `canvec.download()`
`all`	Use `all=TRUE` to recursively delete the cache directory.
`keeparchives`	Pass `TRUE` to keep .zip files downloaded by `canvec.download()`
`keepfolders`	Pass `TRUE` to keep folders extracted by `canvec.download()`

Examples

```
canvec.download(nts('21h1'))
canvec.cleanup(nts('21h1'))
#or
canvec.cleanup(all=TRUE)
```

canvec.defaultoptions *Get Default Options For Plotting Layers*

Description

Get Default Options For Plotting Layers

Usage

```
canvec.defaultoptions(layerid)
```

Arguments

`layerid`	The layer id as defined in `canvec_layers$id`

Value

a `list` object that can be passed to `canvec.plot()`

`canvec.download` *Download and Extract CanVec or CanVec+ Data*

Description

Downloads CanVec or CanVec+ data (as applicable) to `cachedir` and extracts the archive.

Usage

```
canvec.download(..., forcedownload = FALSE, forceextract = FALSE,
  extract = TRUE, cachedir = NULL)
```

Arguments

`...`	A list of NTS References as generated by `nts()`
`forcedownload`	A boolean describing if the file should be re-downloaded, even if already present.
`forceextract`	Force the exctraction of the archive even if the folder is already present.
`extract`	Pass `extract=FALSE` to download the archive without extracting.
`cachedir`	Pass a specific cache directory in which to download and extract the file. Default value is that returned by `canvec.cachedir()`

Examples

```
canvec.download(nts('21h1'))
```

`canvec.export` *Export CanVec Data*

Description

Export `layers` for one or more NTS reference(s) `ntsid` to path `tofolder`, automatically renaming layers based on their layerid. Pass `crs` to re-project data, or pass `driver` to convert file format.

Usage

```
canvec.export(ntsid, tofolder, layers = NULL, crs = NULL, cachedir = NULL,
  driver = NULL, combine = TRUE, overwrite = TRUE, ...)
```

Arguments

ntsid	One or more NTS References as generated by nts()
tofolder	A directory to which files should be copied.
layers	One or more layer ids as listed in canvec_layers$id. Defaults to all layers.
crs	A CRS (as generated by sp::CRS()) in which to project the data.
cachedir	Pass a specific cache directory in which files have been extracted. Default value is that returned by canvec.cachedir()
driver	A rgdal driver with which to save data. ESRI Shapefile, KML, CSV, and GML have been tested; others returned by rgdal::ogrDrivers() may also work.
combine	TRUE if output should be one file per layer, FALSE otherwise
overwrite	TRUE if files should overwrite files already in output directory.
...	Arguments passed on to sp::writeOGR()

Examples

```
canvec.download(nts("21h01"))
canvec.export(nts("21h01"), "exporteddata", layers=c("road", "river"))
canvec.export(nts("21h01"), "exporteddataUTM", layers=c("road", "river"),
                crs=sp::CRS("+init=epsg:26920"))
canvec.export(nts("21h01"), "exporteddata", layers=c("road", "river"),
               driver="KML")
canvec.export(nts("21h01"), "exporteddataALL")
```

---

canvec.findlayer *Get File Prefix of a CanVec Layer*

Description

Find directory and file prefix for a layer id (as listed in canvec_layers$id) in the directory specified. If the layer is not available, a warning will be issued.

Usage

```
canvec.findlayer(directory, layerid)
```

Arguments

directory	A directory where CanVec shapefiles are located.
layerid	A single layer id as listed in canvec_layers$id

Value

The file prefix of the layer, or NA if the layer does not exist

canvec.load *Load CanVec Data*

Description

Load layerid for NTS reference(s) that were previously downloaded to cachedir.

Usage

```
canvec.load(ntsid, layerid, cachedir = NULL)
```

Arguments

ntsid	One or more NTS References as generated by nts()
layerid	A single layer id as listed in canvec_layers$id
cachedir	Pass a specific cache directory in which files have been extracted. Default value is that returned by canvec.cachedir()

Value

A sp::Spatial* object loaded from the given shapefile or a list of Spatial* objects if more than one directory is specified.

Examples

```
buildings <- canvec.load(nts("21h1"), "building")
```

canvec.loadfromdir *Load CanVec Data From Directory*

Description

Load layerid from a directory or directories that contain(s) CanVec data.

Usage

```
canvec.loadfromdir(directory, layerid)
```

Arguments

directory	A directory or directories that contain(s) CanVec or CanVec+ data.
layerid	A single layer id as listed in canvec_layers$id

Value

A sp::Spatial* object loaded from the given shapefile or a `list` of Spatial* objects if more than one directory is specified.

See Also

canvec.load

---

canvec.plot *Plot CanVec Spatial Data*

Description

Plot CanVec Spatial Data

Usage

```
canvec.plot(loaded, options = NULL, crs = NULL, add = NULL)
```

Arguments

`loaded`	A Spatial* object or list of Spatial* objects such as those generated by `canvec.load()`
`options`	A `list` object with the graphical options to be applied to the layers specified
`crs`	A CRS (as generated by `sp::CRS()`) in which to project the data.
`add`	TRUE if layer or layers should be added to the current plot, FALSE if all layers should be plotted on a fresh plot (not reccomended) or NULL for default behaviour, which will create a new plot for the first layer and add each subsequent layer

---

canvec.qplot *Quickly Plot Canvec Data*

Description

Quickly plot CanVec data with options to change the plotting style of each. If data does not exist in the cache it will be downloaded. Simplest usage uses `searchbbox()` to find an appropriate bounding box (e.g. `canvec.qplot(bbox=searchbbox("Wolfville NS"))`). Be careful plotting feature-intensive layers (e.g. "road", "building") over large areas (e.g. `searchbbox("toronto, on")`). This will happily run but plotting the map may take up to 20 minutes!

Usage

```
canvec.qplot(ntsid = NULL, bbox = NULL, layers = c("waterbody", "forest",
  "contour", "river", "road"), options = NULL, data = NULL,
  cachedir = NULL, plotdata = TRUE, atscale = nts.SCALE50K,
  stoponlargerequest = TRUE, epsg = NULL, ...)
```

Arguments

ntsid	One or more NTS References as generated by nts()
bbox	A bounding box describing the desired extent. If no ntsid is provided, nts(bbox=bbox) will be invoked to find the appropriate NTS Reference(s)
layers	A list of layers as defined in canvec_layers$id in the order in which they should be plotted
options	A list object containing the options for each layer in the form options$layerid <- list(col="lightblue")
data	A list object that contains the loaded Spatial* data to be plotted. This should always be an object that was returned by canvec.qplot()
cachedir	Pass a specific cache directory in which files have been extracted. Default value is that returned by canvec.cachedir()
plotdata	TRUE if data should be plotted, FALSE if data should just be loaded.
atscale	One of nts.SCALE50K (CanVec data) or nts.SCALE250K (CanVec+ data)
stoponlargerequest	Stop if a large (greater than 4 tiles) area is requested. Defaults to TRUE.
epsg	The epsg code in which to plot the data, or NULL for automatic. Use epsg=3857 to layer on Open Street Map tiles, or epsg=269XX (where XX is the UTM Zone) for a UTM projection. Defaults to no projection, although sp::plot adjusts the aspect such that the default does not appear distorted.
...	A list of graphical parameters passed to the inital call to plot(). Use add=TRUE to layer on an existing plot.

Value

A list object that contains the Spatial* data that was plotted

Examples

```
#simplest use using searchbbox() from {prettymapr}
library(prettymapr)
wolfville <- searchbbox("Wolfville NS", source="google")
canvec.qplot(bbox=wolfville)
canvec.qplot(bbox=wolfville, layers=c("waterbody", "forest"))

#can also plot by NTS sheet and use bbox= or xlim, ylim to zoom.
canvec.qplot(nts("21h1"), layers=c("waterbody", "forest", "contour", "river", "road"))
canvec.qplot(bbox=makebbox(45.1, -64.35, 45.05, -64.4),
       layers=c("waterbody", "contour", "river", "building", "building_poly", "road"))

#method returns plot data argument so data does not need to be loaded each time.
#this will not work when changing nts sheets.
plotdata <- canvec.qplot(nts("21h1"), layers=c("waterbody", "forest", "contour", "river"))
plotdata <- canvec.qplot(bbox=makebbox(45.1, -64.35, 45.05, -64.4),
                      layers=c("waterbody", "contour", "river"),
                      data=plotdata)
```

```
#use with prettymapr::addscalebar() and prettymapr::addnortharrow()
library(prettymapr)
wolfville <- searchbbox("Wolfville NS", source="google")
canvec.qplot(bbox=wolfville)
addscalebar()
addnortharrow()

#or use with prettymapr::prettymap() to set margins and add north arrow/
#scalebar
prettymap(canvec.qplot(bbox=wolfville))
```

`canvec.url` *Get CanVec or CanVec+ data URL*

Description

Get CanVec or CanVec+ data URL based on the NTS Reference (as generated by `nts()`) provided. The URL generated may or may not exist depending whether or not the sheet or area is available. CanVec data is available by mapsheet (e.g. 021H01; 1:50k), CanVec+ data is available by map area (e.g. 021H; 1:250k). If a the `ntsid` provided is a 1:50k reference, a CanVec URL will be generated. Otherwise, a CanVec+ url is generated.

Usage

```
canvec.url(ntsid,
  server = "http://ftp.geogratis.gc.ca/pub/nrcan_rncan/vector/")
```

Arguments

`ntsid`	A single NTS Reference as generated by nts().
`server`	The server to download from (default: `http://ftp.geogratis.gc.ca/pub/nrcan_rncan/vector/`)

Value

A URL where the given data can be found.

canvec_layers *Canvec Layers*

Description

As documented by the Canvec and canvec+ documentation found at http://ftp2.cits.rncan.gc.ca/pub/canvec+/doc/CanVec+_di

Usage

```
canvec_layers
```

Format

A data frame with six variables: `name`, `id`, `theme`, `filename`, `geometry`, and (`geometry_ext`. These are used to generate pretty filenames and provide human-readable access to canvec and canvec+ data.

nts *Generate NTS References*

Description

Generate one or more NTS references based on arguments provided.

Usage

```
nts(..., lat = NULL, lon = NULL, bbox = NULL, atscale = nts.SCALE50K)
```

Arguments

`...`	An arbitrary number of strings in the form 21h1 to be parsed
`lat`	A vector of latitude values
`lon`	A vector of longitude values
`bbox`	A bounding box in the form returned by sp::bbox()
`atscale`	An integer value describing scale, either 1 (250k) or 2 (50k)

Value

one or more NTS references in the form c("021", "H", "01")

Examples

```
nts('21h')
nts('21h1')
nts('21h1', '21a16', '021A15')
nts(lat=45.2, lon=-64.32)
nts(lat=c(45.2, 46.2), lon=c(-64.32, -64.81))

library(prettymapr)
nts(bbox=makebbox(45.125, -64.25, 44.875, -64.75))
```

---

nts.bbox *Get Bounding Box of an NTS Reference*

Description

Calculates the bounding box in latitude/longitude described by a particular NTS Reference.

Usage

```
nts.bbox(ntsid)
```

Arguments

ntsid	One or more NTS References as generated by nts()

Value

A bbox like that returned by sp::bbox(), or a list of such objects.

Examples

```
nts.bbox(nts('21h'))
```

---

nts.bybbox *Get NTS References by Bounding Box*

Description

Retreive a list of NTS references at a given scale by bounding box. Bounding box is in the form returned by sp::bbox(). NTS References all have a valid series component (e.g. "021"), but map area (e.g. "H") and map sheet (e.g. "01") are not checked to make sure they exist.

Usage

```
nts.bybbox(bbox, atscale)
```

Arguments

bbox	A bounding box of lat/lon values in the form returned by sp::bbox().
atscale	One of nts.SCALESERIES, nts.SCALE250K, or nts.SCALE50K

Value

A list object containing zero or more NTS References.

See Also

nts

---

nts.idat *Get NTS Reference At A Location*

Description

Get NTS Reference(s) based on location at a given scale.

Usage

```
nts.idat(lat, lon, atscale)
```

Arguments

lat	A scalar or vector of latitude values
lon	A scalar or vector of longitude values of same length as lat
atscale	One of nts.SCALESERIES, nts.SCALE250K, or nts.SCALE50K

Value

A list of NTS References

See Also

nts

nts.SCALE250K *A constant denoting NTS Map Area (1:250k) scale (1) @export*

Description

A constant denoting NTS Map Area (1:250k) scale (1) @export

Usage

```
nts.SCALE250K
```

Format

An object of class `numeric` of length 1.

nts.SCALE50K *A constant denoting NTS Map Sheet (1:50k) scale (2) @export*

Description

A constant denoting NTS Map Sheet (1:50k) scale (2) @export

Usage

```
nts.SCALE50K
```

Format

An object of class `numeric` of length 1.

nts.SCALESERIES *A contstant denoting NTS Series scale (0)*

Description

A contstant denoting NTS Series scale (0)

Usage

```
nts.SCALESERIES
```

Format

An object of class `numeric` of length 1.

`ntsstring` *Generate NTS Reference Strings*

Description

Generate NTS strings from NTS references or other arguments to generate such a list from nts() based on arguments provided.

Usage

```
ntsstring(ntsid = NULL, lat = NULL, lon = NULL, bbox = NULL,
  atscale = nts.SCALE50K)
```

Arguments

`ntsid`	An arbitrary number of NTS references in the form c("021", "H", "01") or `list(c("021", "H", "01"), c("021", "A", "16"))` to be converted
`lat`	A vector of latitude values
`lon`	A vector of longitude values
`bbox`	A bounding box in the form returned by sp::bbox()
`atscale`	An integer value describing scale, either 1 (250k) or 2 (50k)

Value

a character vector of NTS reference strings in the form "021H01".

Examples

```
ntsstring(c("021", "H", "01"))

library(prettymapr)
ntsstring(bbox=makebbox(45.125, -64.25, 44.875, -64.75))
```

Index