Package ‘infochimps’

March 22, 2011

Type Package

Title An R wrapper for the infochimps.com API services

Version 0.3

Date 2011-03-22

Author Drew Conway

Maintainer Drew Conway `<drew.conway@nyu.edu>`

Depends RCurl, RJSONIO

Description This package provides functions to access all of the APIs
currently available infochimps.com. For more information see http://api.infochimps.com/.

License BSD

LazyLoad yes

R topics documented:

infochimps-package *An R wrapper for the infochimps.com API services*

Description

This package provides functions to access all of the APIs currently available infochimps.com.

Details

Package:	infochimps
Type:	Package
Version:	0.3
Date:	2011-22-2011
License:	BSD
LazyLoad:	yes

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/.

Examples

```
library(infochimps)

infochimps("your.api.key")
drew<-influence("drewconway")
```

census *Gather U.S. Census data for a given IP address*

Description

A function to return combined census data for a given IP address using the inforchimps.com APIs

Usage

```
census(ip.address)
```

Arguments

`ip.address`	Properly formatted IP address as character string

Value

list : see reference for listing of all data returned (extensive)

Author(s)

Drew Conway, <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/web/an/ip_census/combined

Examples

```
infochimps("your.api.key")
nyu<-census("128.122.79.165")
```

`conversations` *Create data frame of recent conversations between two Twitter users*

Description

A function to return the interactions between two Twitter users with infochimps.com API

Usage

```
conversations(screen.name.a, screen.name.b, user.id.a = NA, user.id.b = NA)
```

Arguments

`screen.name.a`	The name of a Twitter user
`screen.name.b`	The name of a Twitter user
`user.id.a`	a Twitter user ID
`user.id.b`	a Twitter user ID

Value

Data frame with the following columns:

`user.id.a`	First Twitter user (numeric)
`user.id.b`	Second Twitter user (numeric)
`conversation.id`	Internal Twitter ID for tweet (numeric)
`conversation.type`	Factor describing conversation type (factor). See ref.
`reply.to.id`	If RE type, internal Twitter ID for reply-to tweet (numeric)

If user.name not found, or no data, return NA

Author(s)

Drew Conway, <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/conversation

Examples

```
infochimps("my.api.key")
jd.tweets<-conversations("drewconway","CMastication")
head(jd.tweets)
```

`demographics` *Gather demographic data for a given IP address from the U.S. Census*

Description

A function to return infochimps.com census data for a given IP address from the Digital Elements IP data and U.S. censu data with infochimps.com APIs.

Usage

```
demographics(ip.address)
```

Arguments

`ip.address`	Properly formatted IP address as character string

Value

list : see reference for listing of all data returned (extensive)

Author(s)

Drew Conway, <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/web/an/de/demographics

Examples

```
infochimps("your.api.key")
nyu<-demographics("128.122.79.165")
```

domain *Return domain information for a given domian*

Description

A function to return Digitial Elements IP domain data from the infochimps.com API

Usage

```
domain(ip.address)
```

Arguments

`ip.address`	Properly formatted IP address as character string

Value

A list containing the following elements:

`domain`	Domain name (character)
`company`	Registered company name (character)
`isp`	Internet service provider (character)
`proxy_type`	Proxy type (character)
`naics_code`	NAICS Code (numeric)

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/web/an/de/domain

Examples

```
infochimps("your.api.key")
nyu<-domain("128.122.79.165")
```

`geo.names` *Autocomplete any place name in the world.*

Description

Place names request. A prefix to autocomplete. One or both of country and state can be supplied to constrain the results.

Usage

```
geo.names(prefix, country="", state="")
```

Arguments

`prefix`	A simple prefix, eg. "port"
`country`	An (optional) string ISO-3166 2-letter country code, 2 characters
`state`	Administrative subdivision one code (eg. TX or AK when country is US)

Value

Returns a character vector of all completions given search criteria.

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/global-place-names-autocomplete-geonames

Examples

```
infochimps("your.api.key")
ny.cities <- geo.names("city", country="US", state="NY")
```

`influence` *Find the level of influence for a given Twitter user*

Description

A function to return infochimps.com influence scores for a Twitter user

Usage

```
influence(screen.name, user.id = NA)
```

Arguments

`screen.name`	The name of a Twitter user
`user.id`	a Twitter user ID

Value

list : see reference for listing of all data returned (extensive)

Author(s)

Drew Conway, <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/influence

Examples

```
infochimps("your.api.key")
drew<-influence("drewconway")
```

`infochimps` *Create an infochimps.com API session.*

Description

List object to hold a user's API key, as well as all API URLS. Needed in all functions

Usage

```
infochimps(api.key)
```

Arguments

`api.key`	A valid infochimps.com API key

Value

`api.key`	A valid infochimps.com API key

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

To get an API key from infochimps.com see, http://api.infochimps.com/about/features-and-pricing

Examples

```
infochimps("your.api.key")

## The function is currently defined as
function(api.key) {
    if(is.character(api.key)){
        .InfochimpsEnv$data$api.key<-api.key
    }
    else{
        warning("API key must be a string")
    }
}
```

`ip.geo` *IP address geo-location*

Description

A function to return Digitial Elements IP Intelligence geo-loaction data from the infochimps.com API

Usage

```
ip.geo(ip.address)
```

Arguments

`ip.address`	Properly formatted IP address as character string

Value

list : see reference for listing of all data returned (extensive)

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/web/an/de/geo

Examples

```
infochimps("your.api.key")
nyu<-ip.geo("128.122.79.165")
```

strong.links *Find all of the Strong Links of a given Twitter user*

Description

A function to return infochimps.com Strong Links data

Usage

```
strong.links(screen.name, user.id = NA)
```

Arguments

screen.name	The name of a Twitter user
user.id	a Twitter user ID

Value

A data frame with the following columns:

user.id	Twitter user ID (numeric)
strong.link	Twitter user ID with Strong Link (numeric)
link.weight	Strength of Strong Link (numeric)

If user.name not found, return NA

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/strong_links

Examples

```
infochimps("your.api.key")
drew.links<-strong.links("drewconway")
head(drew.links)
```

trstrank — *Get the trstrank score for a given Twitter user*

Description

A function to return infochimps.com trstrank score for a Twitter user

Usage

```
trstrank(screen.name, user.id = NA)
```

Arguments

`screen.name`	The name of a Twitter user
`user.id`	a Twitter user ID

Value

A list with the following elements:

`user_id`	A Twitter user ID (numeric)
`screen_name`	Screen name of a Twitter user (character)
`trstrank`	trstrank score (numeric)
`tq`	trstrank quotient (numeric)

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/trstrank

Examples

```
infochimps("your.api.key")
trstrank("drewconway")
```

`tw.Autocomplete` *Matches partial Twitter user names for completion.*

Description

Uses the same type of autocomplete that the Twitter iPhone app uses. Given a twitter screen name prefix returns up to 100 possible screen name completions sorted by trstrank.

Usage

```
tw.Autocomplete(prefix)
```

Arguments

`prefix`	A three-character or more prefix of a Twitter screen name

Value

Returns a character vector of all completions given search criteria.

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/twitter-screen-name-autocomplete

Examples

```
infochimps("your.api.key")
tw.Autocomplete("drew")
```

`tw.PeopleSearch` *Full text search, on any or all fields, of twitter user profiles.*

Description

A function to return infochimps.com Twitter people search. Full text search, on any or all fields, of twitter user profiles.

Usage

```
tw.PeopleSearch(search.string, from=0, to.df=TRUE)
```

Arguments

`search.string`
The term to search for. You may use "field:term" to search for that term in only a given field (or even "field.subfield:term" if there are nested fields

`from` Integer offset to start results at. A multiple of 100.

`to.df` Option to return search query as a data frame. Default is TRUE, and if FALSE returned as list keyed by data values.

Value

A data frame with the following data columns (all as characters):

`user_id` A Twitter user ID

`scraped_at` When the the infochimps flying monkeys scraped the user"s profile from the twitter api (YYYYMMddhhmmss)

`screen_name` Screen name of a Twitter user

`url` User"s web site

`location` User"s hand entered, raw text, location

`description` [unspecified]

`time_zone` Time zone as tzinfo compatible string

`utc_offset` [unspecified]

`lang` two letter language code, eg. "en"

`geo_enabled` The string "true" if the user has enabled geo location, "false" otherwise

`verified` Is this a twitter verified account? "true" or "false"

`contributors_enabled`
If the twitter account is a pro account allowing multiple users to control visit dev.twitter.com for details

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/twpeoplesearch

Examples

```
infochimps("your.api.key")
search.test <- tw.PeopleSearch("R hacks")
head(search.test)
```

ufo — *Search accounts of UFO sightings from the National UFO Reporting Center.*

Description

A function to return infochimps.com data on international UFO sightings.

Data is from a comprehensive listing of over 60,000 unidentified-flying-object (UFO) sightings in the United States (and some elsewhere). Fields of particular interest are the shape, which describes the shape of the UFO encountered as well as the description which is free-form descriptive text about the event.

Usage

```
ufo(search.string, from=0, to.df=TRUE)
```

Arguments

`search.string`	The term to search for. You may use 'field:term' to search for that term in only a given field (or even 'field.subfield:term' if there are nested fields
`from`	Integer offset to start results at. A multiple of 100.
`to.df`	Option to return search query as a data frame. Default is TRUE, and if FALSE returned as list keyed by data values.

Value

A data frame with the following data columns (all as characters):

`sighted_at`	Date ufo sighting occurred (YYYYMMdd)
`reported_at`	Date ufo sighting was reported (YYYYMMdd)
`location`	City and two letter state abbreviation sighting occurred in (City, State)
`shape`	Shape of the ufo. One of [changed, changing, chevron, cigar, circle, cone, crescent, cross, cylinder, delta, diamond, disk, dome, egg, fireball, flare, flash, formation, hexagon, light, other, pyramid, rectangle, round, sphere, teardrop, triangle, unknown]
`duration`	Human entered duration, eg. 2 min.
`description`	Full text description of the sighting.

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/60000-documented-ufo-sightings-with-text-descriptions-and-metada

Examples

```
infochimps("your.api.key")
cow.test <- ufo("cow field")
head(cow.test)
```

`wikipedia.abstratcs` *Search Wikipedia articles by title, by abstract content, or by both.*

Description

Provides access to infochimps.com search API of Wikipedia article abstracts.

Usage

```
wikipedia.abstratcs(search.string, from=0, to.df=TRUE)
```

Arguments

`search.string`	The term to search for. You may use "field:term" to search for that term in only a given field (or even "field.subfield:term" if there are nested fields
`from`	Integer offset to start results at. A multiple of 100.
`to.df`	Option to return search query as a data frame. Default is TRUE, and if FALSE returned as list keyed by data values.

Value

A data frame with the following data columns (all as characters):

`title`	Wikipedia article title
`url`	Web url of wikipedia article
`abstract`	Text content of wikipedia article abstract

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/wikipedia-articles-abstract-search

Examples

```
infochimps("your.api.key")
wiki.stats <- wikipedia.abstratcs("statistical computing")
head(wiki.stats)
```

word.bag *Find the words most associated with a given Twitter user*

Description

A function to return infochimps.com Word Bag for a Twitter user

Usage

```
word.bag(screen.name, user.id = NA)
```

Arguments

`screen.name`	The name of a Twitter user
`user.id`	a Twitter user ID

Value

A list with the following elements:

`user_id`	Twitter used ID (numeric)
`vocab`	Number of distinct tokens ever emitted (numeric)
`total.usages`	Total number of tokens emitted (numeric)
`tok.df`	Data frame with columns: user.id (numeric), rel.freq (numeric), tok user (character), freq.ppb (numeric)

If user.name not found, return NA

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/wordbag

Examples

```
infochimps("your.api.key")
hilary<-word.bag("hmason")
```

word.freq *Word Frequencies From the British National Corpus*

Description

The British National Corpus (BNC) is a 100-million-word text corpus of samples of written and spoken English from a wide range of sources. It was compiled as a general corpus (collection of texts) in the field of corpus linguistics. The corpus covers British English the late twentieth century from a wide variety of genres with the intention that it be a representative sample of spoken and written British English of that time.

Of the two parts to the 10-million word spoken corpus, one is a demographic part, containing transcriptions of spontaneous natural conversations made by members of the public and the other a context-governed part, containing transcriptions of recordings made at specific types of meeting and event. All the original recordings transcribed for inclusion in the BNC have been deposited at the British Library Sound Archive.

In this data set, we provide plain text versions of the frequency lists contained in WFWSE. These are raw unedited frequency lists produced by our software and do not contain the many additional notes supplied in the book itself. The lists are tab delimited plain text so can be imported into your preferred spreadsheet format. For the main lists we provide a key to the columns. More details on the process undertaken in the preparation of the lists can be found in the introduction to the book. These lists show dispersion ranging between 0 and 1 rather than 0 and 100 as in the book. We multiplied the value by 100 and rounded to zero decimal places in the book for reasons of space. Log likelihood values are shown here to one decimal place rather than zero as in the book. Please note, all frequencies are per million words.

Usage

```
word.freq(word)
```

Arguments

word	The word being matched against the corpus

Value

Returns a list of characters with the following values:

head_word	Word type headword, see pp.4-5
head_word_freq_ppm	Rounded frequency per million word tokens (down to a minimum of 10 occurrences of a lemma per million), see pp. 5. Where BOTH head word and lemmas appear
head_word_range	Range, the number of sectors of the corpus (out of a maximum of 100) in which the word occurs. Where BOTH head word and lemmas appear
head_word_dispersion	Dispersion value (Juilland's D) from a minimum of 0.00 to a maximum of 1.00. Where BOTH head word and lemmas appear
variant_word	Variant form of headword

`variant_word_freq_ppm`
: Rounded frequency per million word tokens (down to a minimum of 10 occurrences of a lemma per million), see pp. 5. Where BOTH head word and lemmas appear

`variant_word_range`
: Range, the number of sectors of the corpus (out of a maximum of 100) in which the word occurs. Where BOTH head word and lemmas appear

`variant_word_dispersion`
: Dispersion value (Juilland's D) from a minimum of 0.00 to a maximum of 1.00. Where BOTH head word and lemmas appear

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/word-frequencies-from-the-british-national-corpus

Examples

```
infochimps("your.api.key")
word.freq("statistics")
```

`word.stats` *Get basic statistics associated with a given word on Twitter*

Description

A function to return infochimps.com Word Stats data

Usage

```
word.stats(tok)
```

Arguments

`tok`	The word you are searching (character)

Value

A list with the following elements:

`global_stdev_ppb`
: Standard deviation (numeric)

`range`
: Range (numeric)

`tok`
: The word (character)

`global_freq_ppb`
: Global frequency in parts-per-billion (numeric)

If tok not found, return NA

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://api.infochimps.com/describe/soc/net/tw/word_stats

Examples

```
infochimps("your.api.key")
word.stats("infochimps")
```

yahoo.stocks *Search historical stock market data.*

Description

This dataset consists of historical stock price data extracted from Yahoo Finance using the ichart (http://finance.yahoo.com/) service. The selection of stock symbols is gathered from nasdaq.com.

Usage

```
yahoo.stocks(symbol, begin.date, end.date, from=0, to.df=TRUE)
```

Arguments

`symbol`	The stock symbol
`from`	Integer offset to start results at. A multiple of 100.
`begin.date`	The first date in the time-series, formatted as YYYYMMdd
`end.date`	The last date in the time-series, formatted as YYYYMMdd
`to.df`	Option to return search query as a data frame. Default is TRUE, and if FALSE returned as list keyed by data values.

Value

A data frame with the following data columns (all as characters):

`exchange`	Exchange on which this stock is traded
`symbol`	Symbol for this stock
`date`	The date, formatted as YYYYMMdd
`open`	Opening price of this stock
`close`	Closing price of this stock
`adj_close`	Adjusted Closing Price
`low`	Lowest price of this stock sold on date
`high`	Highest price of this stock sold on date
`volume`	Number of shares sold on this day in units of 1000

Author(s)

Drew Conway <drew.conway@nyu.edu>

References

http://www.infochimps.com/datasets/yahoo-stock-search

Examples

```
infochimps("your.api.key")
appl.stock <- yahoo.stocks("aapl", begin.date="20010801", end.date="20010901")
head(appl.stock)
```

Index