Package 'yummlyr'

November 24, 2015

Type Package

Title R Bindings for Yummly API

Version 0.1.1

Author Roman Tsegelskyi

Maintainer Roman Tsegelskyi <roman.tsegelskyi@gmail.com>

Description Yummly.com is one of the world's largest and most powerful recipe search sites and this package aims to provide R bindings for publicly available Yummly.com Recipe API (https://developer.yummly.com/).

URL https://github.com/RomanTsegelskyi/yummlyr

BugReports https://github.com/RomanTsegelskyi/yummlyr/issues

License GPL (>= 2)

LazyData TRUE

Depends R (>= 2.10)

Suggests testthat, knitr

Imports httr, jsonlite

VignetteBuilder knitr

RoxygenNote 5.0.0

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-24 07:46:55

R topics documented:

`add_argument` *Add argument to a query*

Description

Add argument to a query

Usage

```
add_argument(argument_values, argument_name, type, query, check = TRUE)
```

Arguments

`argument_values`	value of the argument
`argument_name`	name of the argument
`type`	argument type from metadata
`query`	existing query to append to
`check`	if to check again metadata values

`check_arguments` *Check ingredients*

Description

Check ingredients list against predifined ingredients by Yummly

Usage

```
check_arguments(arguments, type)
```

Arguments

`arguments`	arguments from metadata to check to check
`type`	type of arguments from metadata

Note

Predifined list is downloaded from Metadata Dictionaries

get_metadata *Get metadata*

Description

Return information about metadata

Usage

```
get_metadata(type)
```

Arguments

type	metadata type

get_recipe *Get recipe from Yummly.com*

Description

This call is equivalent in functionality to a Yummly recipe page.

Usage

```
get_recipe(recipe_id, app_id = auth_cache$APP_ID,
  app_key = auth_cache$APP_KEY)
```

Arguments

recipe_id	recipe ID
app_id	application ID
app_key	application KEY

Note

This function resembles viewing a recipe on Yummly.com

References

- Yummly Developer Guide https://developer.yummly.com/documentation

Examples

```
## Not run:
# to request the response for French Onion Soup by Ree Drummond The Pioneer Woman
# with id French-Onion-Soup-The-Pioneer-Woman-Cooks-_-Ree-Drummond-41364
get_recipe("French-Onion-Soup-The-Pioneer-Woman-Cooks-_-Ree-Drummond-41364")

## End(Not run)
```

`parse_jsonp` *Parse JSONP returned by Yummly for metadata*

Description

This function parses JSONP that yummly uses as a response. It is based on assumption that list of elements is returned.

Usage

```
parse_jsonp(jsonp)
```

Arguments

`jsonp`	jsonp string

`perform_query` *Process query*

Description

Query Yummly API and check return codes

Usage

```
perform_query(query)
```

Arguments

`query`	string query to execute

prepare_array_parameter *Prepare search parameter*

Description

Prepare search parameter from direction

Usage

```
prepare_array_parameter(param, name)
```

Arguments

param	vector parameter to use
name	name for parameter to use

save_yummly_credentials *Save API credentials for later use*

Description

This functions caches the credentials to avoid need for entering it when calling other functions

Usage

```
save_yummly_credentials(app_id, app_key)
```

Arguments

app_id	application ID
app_key	application key

Examples

```
# since not checking is preformed not to waste API calls
# it falls on the user to save correct information
save_yummly_credentials("APP_ID", "APP_KEY")
```

search_recipes *Search recipes on Yummly.com*

Description

Query Yummly.com API to search for recipes with certain parameter. All parameters are optional and can be used in any combination. The criteria you pass via the various parameters are combined with the AND operator (set conjunction). In other words, every recipe has to match the search phrase and satisfy the ingredient, cuisine, course, holiday, time, nutrition, and taste restrictions as described below. If you specify a multi-word phrase to the q parameter, every word has to match something in each matching recipe:

Usage

```
search_recipes(search_words, require_pictures, allowed_ingredient,
  excluded_ingredient, allowed_diet, allowed_allergy, allowed_cuisine,
  excluded_cuisine, allowed_course, excluded_course, allowed_holiday,
  excluded_holiday, max_total_time, max_results, start, nutrition, flavor,
  facet_field, app_id = auth_cache$APP_ID, app_key = auth_cache$APP_KEY)
```

Arguments

search_words	search phrase, can be supplied in from of vector of words
require_pictures	set to TRUE if only to return recipes with photos
allowed_ingredient	ingredient that all search results must include
excluded_ingredient	ingredient that all search results should not contain
allowed_diet	search results will only include recipes whose ingredients are allowed for that diet
allowed_allergy	only include recipes whose ingredients are allowed for that allergy
allowed_cuisine	search results will only include recipes with that cuisine
excluded_cuisine	search results will only exclude recipes with that cuisine
allowed_course	search results will only include recipes with that cuisine
excluded_course	search results will only exclude recipes with that cuisine
allowed_holiday	search results will only include recipes with that holiday
excluded_holiday	search results will only exclude recipes with that holiday

max_total_time	search for recipes that do not exceed a specified max total cook + prep time in seconds
max_results	number of results to return
start	start with specific result in search
nutrition	set the range of allowed values for a given nutrition attribute (see below for the list of supported nutrition attributes) by setting a min and/or a max
flavor	set the ranges for taste attributes (this corresponds to the taste sliders on the Yummly.com search page). The values of min and max are between 0 and 1.
facet_field	facet counts for ingredient and diet. When this parameter is called, the response will include a facetCounts object that lists the matching diets or ingredients and how many results match each diet or ingredient.
app_id	application ID
app_key	application key

Note

This function resembles search query to Yummly API

References

- Yummly Developer Guide https://developer.yummly.com/documentation

Examples

```
## Not run:
# search for recipes with bacon
search_recipes("bacon")

# search for recipes with bacon that have pictures
search_recipes("bacon", require_pictures = TRUE)

# search for "Onion Soup" recipes which include garlic and cognac
search_recipes("Onion Soup", allowed_ingredient = c("garlic", "cognac"))

# search for "Onion Soup" recipes which do not include "onion soup mix"
search_recipes("Onion Soup", excluded_ingredient = c("onion soup mix"))

# search for "Onion Soup" recipes that are Dairy-Free and Gluten-Free
search_recipes("bacon", allowed_allergy =c("Dairy-Free", "Gluten-Free"))

# search for "Onion Soup" recipes that are Pescetarian and Lacto vegetarian
search_recipes("bacon", allowed_diet =c("Pescetarian", "Lacto vegetarian")

# search for "Onion Soup" recipes that match American Cuisine
search_recipes("bacon", allowed_cuisine =c("American")

# exclude American recipes from a search for "Onion Soup"
search_recipes("bacon", excluded_cuisine =c("American")
```

```
# search for "Onion Soup" recipes that are Appetizers
search_recipes("bacon", allowed_course =c("Appetizers")

# exclude Appetizer recipes from a search for "Onion Soup"
search_recipes("bacon", excluded_course =c("Appetizers")

# search for "Onion Soup" recipes for Thanksgiving
search_recipes("bacon", allowed_holiday =c("Thanksgiving")

# exclude Thanksgiving recipes from a search for "Onion Soup"
search_recipes("bacon", excluded_holiday =c("Thanksgiving")

# if you want 20 recipes per page and want to see the second page of results
search_recipes("bacon", max_results = 20)

# if you want to start with position 20
search_recipes("bacon", start = 20)

# looking for recipes with a lot of Potassium, try setting a min of 3000 mg
# and a max of the Daily Suggested Value of 3500 mg
search_recipes("bacon", nutrition = list(Calcium=list(min=3, max=3.5)))

#  search for recipes which are very sweet but are not very spicy,
search_recipes("bacon", flavor = list(sweet=list(min=0.1, max=1)))

## End(Not run)
```

`yummlyr_options` *Querying/setting yummlyr option*

Description

To list all `yummlyr` options, just run this function without any parameters provided. To query only one value, pass the first parameter. To set that, use the `value` parameter too.

Usage

```
yummlyr_options(o, value)
```

Arguments

`o`	option name (string). See below.
`value`	value to assign (optional)

Details

The following `yummlyr` options are available:

- `log`: `NULL` or an optionally passed *logger name* from **futile.logger** to record all info, trace, debug and error messages.

Index