

# Package ‘oncrawlR’

November 18, 2019

**Type** Package

**Title** Machine Learning for S.E.O

**Version** 0.1.8

**Maintainer** Vincent Terrasi <vincent@oncrawl.com>

**Description** Measures different aspects of page content, structure and performance for SEO (Search Engine Optimization).

Aspects covered include HTML tags used in SEO, duplicate and near-duplicate content, structured data, on-

site linking structure and popularity transfer, and many other amazing things.

This package can be used to generate a real, full SEO audit report, which serves to detect errors or inefficiencies on a page that can be corrected in order to optimise its performance on search engines.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.4.0)

**Imports** RCurl (>= 1.8), jsonlite (>= 1.6), dplyr (>= 0.7), xgboost (>= 0.8), pROC (>= 1.1), ggplot2 (>= 3.1), caret (>= 6.0), DALEX (>= 0.3), rlist (>= 0.4.6), readr (>= 1.3), rlang (>= 0.3), formattable (>= 0.2), sparkline (>= 0.2), webshot (>= 0.5), e1071 (>= 1.7), pdp (>= 0.7), fs (>= 1.3), scales (>= 1.0.0), tidyr (>= 0.8.2), htmltools (>= 0.3.5)

**Suggests** devtools (>= 1.12.0), testthat

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Author** Vincent Terrasi [aut, cre],  
OnCrawl [cph, fnd]

**Repository** CRAN

**Date/Publication** 2019-11-18 16:40:02 UTC

**R topics documented:**

export_formattableWidget . . . . .	2
getCrawl . . . . .	3
getPageFields . . . . .	4
getProject . . . . .	5
initAPI . . . . .	6
listLinks . . . . .	6
listLogs . . . . .	7
listPages . . . . .	8
listProjects . . . . .	9
oncrawlCreateDashboard . . . . .	10
oncrawlCreateGraph . . . . .	11
oncrawlCreateSegmentation . . . . .	12
oncrawlExplainModel . . . . .	12
oncrawlSplitURL . . . . .	13
oncrawlTrainModel . . . . .	14
<b>Index</b>	<b>15</b>

---

export\_formattableWidget

*Transform HTML widget into picture*

---

**Description**

Transform HTML widget into picture

**Usage**

```
export_formattableWidget(w, file, width = 400, background = "white",
  delay = 0.2)
```

**Arguments**

w	HTML content to print
file	A vector of names of output files. Should end with .png, .pdf, or .jpeg. If several screenshots have to be taken and only one filename is provided, then the function appends the index number of the screenshot to the file name.
width	Viewport width. This is the width of the browser "window".
background	Background color for web page
delay	Time to wait before taking screenshot, in seconds. Sometimes a longer delay is needed for all assets to display properly.

---

`getCrawl`*Get a crawl*

---

**Description**

Get a crawl

**Usage**

```
getCrawl(crawlId)
```

**Arguments**

`crawlId`      Id of your crawl

**Details**

<<http://developer.oncrawl.com/#Get-a-crawl>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resources referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

The HTTP response is JSON object with a single `crawl` key containing the crawl's data

**Value**

Json

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
initAPI()  
project <- getCrawl("YOURCRAWLID")  
  
## End(Not run)
```

---

getPageFields	<i>List all available fields from a crawl</i>
---------------	---

---

**Description**

List all available fields from a crawl

**Usage**

```
getPageFields(crawlId)
```

**Arguments**

crawlId	ID of your crawl
---------	------------------

**Details**

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resource(s) referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

**Value**

Character Array

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
pages <- getFields(YOURCRAWLID)  
  
## End(Not run)
```

---

`getProject`*Get a project*

---

**Description**

Get a project

**Usage**

```
getProject(projectId)
```

**Arguments**

`projectId`      Id of your project

**Details**

<<http://developer.oncrawl.com/#Get-a-project>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resource(s) referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

The HTTP response is JSON object with three keys: - A project key with the project's data - A crawl\_configs key with a list of all the project's crawl configurations. - A crawls key with a list of all project's crawl.

**Value**

Json

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
initAPI()  
project <- getProject(YOURPROJECTID)  
  
## End(Not run)
```

---

initAPI	<i>Prepare Token for API calls</i>
---------	------------------------------------

---

**Description**

Prepare Token for API calls

**Usage**

```
initAPI(path)
```

**Arguments**

path	path of your conf file
------	------------------------

**Details**

Example file for oncrawl\_configuration.txt

```
key = 5516LP29W5Q9XXXXXXXXXXXXXXXXOEUGWHM9 debug = FALSE api = https://app.oncrawl.com/api/v2/
```

**Value**

ok if no error with API authentication

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
initAPI("oncrawl_configuration.txt")  
  
## End(Not run)
```

---

listLinks	<i>List all links from a crawl</i>
-----------	------------------------------------

---

**Description**

List all links from a crawl

**Usage**

```
listLinks(crawlId, originFilter = "", targetFilter = "")
```

**Arguments**

crawlId	ID of your crawl
originFilter	select a specific source
targetFilter	select a specific target

**Details**

<<http://developer.oncrawl.com/#Data-types>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resource(s) referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

**Value**

Json

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
links <- listLinks(YOURCRAWLID)  
  
## End(Not run)
```

---

listLogs

*List all pages from logs monitoring*

---

**Description**

List all pages from logs monitoring

**Usage**

```
listLogs(projectId)
```

**Arguments**

projectId	ID of your project
-----------	--------------------

**Details**

<<http://developer.oncrawl.com/#Data-types>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resources referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

**Value**

Json

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:
pages <- listLogs(YOURPROJECTID)

## End(Not run)
```

---

listPages

*List all pages from a crawl*

---

**Description**

List all pages from a crawl

**Usage**

```
listPages(crawlId)
```

**Arguments**

crawlId            ID of your crawl

**Details**

<<http://developer.oncrawl.com/#Data-types>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resource(s) referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error



**Value**

Json

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
pages <- listPages(YOURCRAWLID)  
  
## End(Not run)
```

---

<code>listProjects</code>	<i>List all projects</i>
---------------------------	--------------------------

---

**Description**

List all projects

**Usage**

```
listProjects(limit = 100)
```

**Arguments**

limit            number of projects

**Details**

<<http://developer.oncrawl.com/#List-projects>>

ResCode 400 : Returned when the request has incompatible values or does not match the API specification. 401 : Returned when the request is not authenticated. 403 : Returned the current quota does not allow the action to be performed. 404 : Returned when any of resources referred in the request is not found. 403 : Returned when the request is authenticated but the action is not allowed. 409 : Returned when the requested operation is not allowed for current state of the resource. 500 : Internal error

**Value**

Json

**Author(s)**

Vincent Terrasi

## Examples

```
initAPI()  
projects <- listProjects()
```

---

oncrawlCreateDashboard  
*Create a dashboard*

---

## Description

Create a dashboard

## Usage

```
oncrawlCreateDashboard(dataset, namefile, width, pathfile = tempdir())
```

## Arguments

dataset	Dataset with 3 columns : date, unique name of metric, value
namefile	Filename for the export
width	Width of your picture
pathfile	string. Optional. If not specified, the intermediate files are created under TEMPDIR, with the assumption that directory is granted for written permission.

## Value

a graph

## Author(s)

Vincent Terrasi

## Examples

```
## Not run:  
oncrawlCreateDashboard(res, "metric.png", 500, ".")  
  
## End(Not run)
```

---

oncrawlCreateGraph     *Create a graph*

---

**Description**

Create a graph

**Usage**

```
oncrawlCreateGraph(dataset, namefile, width, height,  
                  pathfile = tempdir())
```

**Arguments**

dataset	dataset generated by DALEX package
namefile	the filename for the export
width	width of your picture
height	height of your picture
pathfile	string. Optional. If not specified, the intermediate files are created under TMPDIR, with the assumption that directory is granted for written permission.

**Value**

file

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:  
oncrawlCreateGraph(res, "metric.png", width=5, height=4, ".")  
  
## End(Not run)
```

oncrawlCreateSegmentation

*Transform a character array of URLs into JSON file for OnCrawl platform*

---

### **Description**

Transform a character array of URLs into JSON file for OnCrawl platform

### **Usage**

```
oncrawlCreateSegmentation(list_urls, namefile, pathfile = tempdir())
```

### **Arguments**

list_urls	your urls
namefile	the filename for the JSON export
pathfile	string. Optional. If not specified, the intermediate files are created under TEMPDIR, with the assumption that directory is granted for written permission.

### **Value**

JSON file

### **Author(s)**

Vincent Terrasi

### **Examples**

```
mylist <- c("/cat/domain", "/cat/")  
oncrawlCreateSegmentation(mylist, "test.json")
```

---

oncrawlExplainModel *Explain XGBoost Model by displaying each importance variables*

---

### **Description**

Explain XGBoost Model by displaying each importance variables

### **Usage**

```
oncrawlExplainModel(model, x, y, max = 10, path = tempdir())
```

**Arguments**

model	your XgBoost model
x	your training data
y	your predicted data
max	the number of importance variable you want to explain
path	path of your conf file

**Value**

graphs

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:
list <- oncrawlTrainModel(dataset,200)
oncrawlExplainModel(list$model, list$x, list$y, 3)

## End(Not run)
```

---

oncrawlSplitURL	<i>Split URLs</i>
-----------------	-------------------

---

**Description**

Split URLs

**Usage**

```
oncrawlSplitURL(list_urls, limit = 15)
```

**Arguments**

list_urls	your urls
limit	the maximum of URLS you want

**Value**

data.frame

**Author(s)**

Vincent Terrasi

**Examples**

```
mylist <- c("/cat/domain/web/", "/cat/", "/cat/domain/")
oncrawlSplitURL(mylist, 2)
```

---

oncrawlTrainModel      *Train XGBoost Model*

---

**Description**

Train XGBoost Model

**Usage**

```
oncrawlTrainModel(dataset, nround = 300, verbose = 1)
```

**Arguments**

dataset	your data frame
nround	number of iterations
verbose	display errors ?

**Value**

a list with your ML model, your training data

**Author(s)**

Vincent Terrasi

**Examples**

```
## Not run:
list <- oncrawlTrainModel(dataset)
plot(list$roc)
print(list$matrix)

## End(Not run)
```

# Index

[export\\_formattableWidget, 2](#)

[getCrawl, 3](#)

[getPageFields, 4](#)

[getProject, 5](#)

[initAPI, 6](#)

[listLinks, 6](#)

[listLogs, 7](#)

[listPages, 8](#)

[listProjects, 9](#)

[oncrawlCreateDashboard, 10](#)

[oncrawlCreateGraph, 11](#)

[oncrawlCreateSegmentation, 12](#)

[oncrawlExplainModel, 12](#)

[oncrawlSplitURL, 13](#)

[oncrawlTrainModel, 14](#)