Package ‘abbyyR’

June 25, 2019

Title Access to Abbyy Optical Character Recognition (OCR) API
Version 0.5.5
Maintainer Gaurav Sood <gsood07@gmail.com>

Description Get text from images of text using Abbyy Cloud Optical Character Recognition (OCR) API. Easily OCR images, barcodes, forms, documents with machine readable zones, e.g. passports. Get the results in a variety of formats including plain text and XML. To learn more about the Abbyy OCR API, see <http://ocrsdk.com/>.

URL http://github.com/soodoku/abbyyR

BugReports http://github.com/soodoku/abbyyR/issues

Depends R (>= 3.2.0)
License MIT + file LICENSE

LazyData true
VignetteBuilder knitr
Imports httr, XML, curl, readr, plyr, progress
Suggests testthat, rmarkdown, knitr (>= 1.11), lintr

RoxygenNote 6.1.1

NeedsCompilation no

Author Gaurav Sood [aut, cre]

Repository CRAN

Date/Publication 2019-06-25 04:30:04 UTC

R topics documented:

  abbyyR-package ......................................................... 2
  abbyy_check ........................................................... 3
  abbyy_GET ............................................................ 3
  abbyy_POST ........................................................... 4
  deleteTask ............................................................. 4
  getAppInfo ............................................................ 5
Description

OCR images, barcodes, forms, documents with machine readable zones, e.g. passports, right from R. Get the results in any one of a wide variety of formats, from text to XML. The package provides access to Abbyy Cloud OCR. For more information, see http://ocrsdk.com/. Details about results of calls to the API can be found at http://ocrsdk.com/documentation/specifications/status-codes/.

To learn how to use abbyyR, see this vignette: https://CRAN.R-project.org/package=abbyyR/vignettes/overview.html. Or, see how to scrape text from a folder of static Wisconsin Ads storyboards: https://CRAN.R-project.org/package=abbyyR/vignettes/wiscads.html.

You need to get credentials (application ID and password) to use this application. If you haven’t already, get these at http://ocrsdk.com/. And set these using setapp

Author(s)

Gaurav Sood
abbyy_check  Request Response Verification

**Description**

Request Response Verification

**Usage**

abbyy_check(req)

**Arguments**

- req  request

**Value**

in case of failure, a message

abbyy_GET  Base POST AND GET functions. Not exported.

**Description**

GET

**Usage**

abbyy_GET(path, query, ...)

**Arguments**

- path  path to specific API request URL
- query  query list
- ...  Additional arguments passed to GET.

**Value**

list
deleteTask

**Description**

POST

**Usage**

abbyy_POST(path, query, body = "", ...)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>path to specific API request URL</td>
</tr>
<tr>
<td>query</td>
<td>query list</td>
</tr>
<tr>
<td>body</td>
<td>passing image through body</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments passed to POST.</td>
</tr>
</tbody>
</table>

**Value**

list

**deleteTask**

*Delete Task*

**Description**

Deletes task and associated data. While Abbyy says, "If you try to delete the task that has already been deleted, the successful response is returned," it doesn’t appear to hold. Hence, the function now defaults to checking the status of the task via getTaskStatus, and deletes only if it hasn’t been deleted.

The function by default prints the status of the task you are trying to delete. It will show up as 'deleted' if successful.

**Usage**

deleteTask(taskId = NULL, ...)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taskId</td>
<td>Required; ID of the task; String</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments passed to abbyy_GET.</td>
</tr>
</tbody>
</table>
getAppInfo

Value

Data frame with all the details of the task you are trying to delete: id (task id), registrationTime, statusChangeTime, status (Submitted, Queued, InProgress, Completed, ProcessingFailed, Deleted, NotEnoughCredits), filesCount (No. of files), credits, resultUrl (URL for the processed file if applicable)

References

http://ocrsdk.com/documentation/apireference/deleteTask/

Examples

```r
## Not run:
deleteTask(taskId = task_id)
## End(Not run)
```

getAppInfo Get Application Info

Description

Get Information about the Application, such as number of pages (given the money), when the application credits expire etc. The function prints the details, and returns a data.frame with the details.

Usage

getAppInfo(...)

Arguments

... Additional arguments passed to abbyy_GET.

Value

A data.frame with the following fields: Name of the Application (name), No. of pages remaining (pages), No. of fields remaining (fields), when the application credits expire (expires) and type of application (type).

References

http://ocrsdk.com/documentation/apireference/getApplicationInfo/

http://ocrsdk.com/schema/appInfo-1.0.xsd
getResults

**Description**

Get data from all the processed images.

**Usage**

getResults(output = "", save_to_file = TRUE)

**Arguments**

- **output**: Optional; folder to which you want to save the data from the processed images; Default is same folder as the script
- **save_to_file**: Required; Default is TRUE; If true, it saves to file. Otherwise returns a data frame with results + other attributes from Abbyy

**Details**

The function calls listFinishedTasks, goes through the finishedTasks data frame and downloads all the files in resultsUrl Results can be stored in memory or written to the hard disk. By default, the function writes to the disk. If the user wants the results to be written to disk, a data frame with paths to local file paths is returned. If the user wants to store the results in memory, data frame with a column carrying the results is returned.

**Value**

data frame returned by listFinishedTasks plus either a column that contains paths to local files (when writing to disk), or actual results returned.

**References**

http://ocrsdk.com/documentation/apireference/getTaskStatus/

**Examples**

```r
## Not run:
getAppInfo()

## End(Not run)
```

```r
getResults(save_to_file = FALSE)

## End(Not run)
```
getTaskStatus

Description
This function gets task status for a particular task ID. The function prints the status of the task by default. The function returns a data.frame with all the task details: id (task id), registrationTime, statusChangeTime, status (Submitted, Queued, InProgress, Completed, ProcessingFailed, Deleted, NotEnoughCredits), filesCount (No. of files), credits, resultUrl (URL for the processed file if applicable).

Usage
getTaskStatus(taskId = NULL, ...)

Arguments
taskId Required, Id of the task
... Additional arguments passed to abbyy_GET.

Value
A data.frame with all the available details about the task

References
http://ocrsdk.com/documentation/apireference/getTaskStatus/

Examples
## Not run:
getTaskStatus(taskId="task_id")

## End(Not run)

listFinishedTasks

Description
List all the finished tasks in the application. The tasks are ordered by the time of the end of processing. No more than 100 tasks can be returned at one method call. The function prints number of finished tasks by default.
Usage

listFinishedTasks(...)  

Arguments  

... Additional arguments passed to abbyy.GET.

Value

A data.frame with the following columns: id (task id), registrationTime, statusChangeTime, status (Completed), filesCount (No. of files), credits, resultUrl (URL for the processed file)

References

http://ocrsdk.com/documentation/apireference/listFinishedTasks/

Examples

```r
## Not run:  
listFinishedTasks()

## End(Not run)
```

---

listTasks  

List Tasks

Description

List all the tasks in the application. You can specify a date range and whether or not you want to include deleted tasks. The function prints total number of tasks and no. of finished tasks by default.

Usage

listTasks(fromDate=NULL, toDate=NULL, excludeDeleted = FALSE, ...)

Arguments

fromDate  

Optional; format: yyyy-mm-ddTh:mm:ssZ

toDate  

Optional; format: yyyy-mm-ddTh:mm:ssZ

excludeDeleted  

Optional; Boolean, Default=FALSE

... Additional arguments passed to abbyy.GET.

Value

A data.frame with the following columns: id (task id), registrationTime, statusChangeTime, status (Submitted, Queued, InProgress, Completed, ProcessingFailed, Deleted, NotEnoughCredits), filesCount (No. of files), credits, resultUrl (URL for the processed file). If no tasks are finished, the last column (resultUrl) isn’t returned.
## References

http://ocrsdk.com/documentation/apireference/listTasks/

## Examples

```r
## Not run:
listTasks()
listTasks(fromDate = "2015-11-10T00:00:00Z", toDate = "2016-11-10T00:00:00Z")
listTasks(fromDate = "2015-11-10T00:00:00Z")

## End(Not run)
```

### ocrFile

#### OCR File

**Description**

Want to quick OCR a local file and get the results? Use this function.

**Usage**

```r
```

**Arguments**

- `file_path` path to file containing OCR’d text; required
- `output_dir` path to output directory. file name will be same as input file name (except for the extension)
- `exportFormat` optional, default: txt; options: txt, txtUnstructured, rtf, docx, xlsx, pptx, pdfSearchable, pdfTextAndImages, pdfa, xml, xmlForCorrectedImage, alto
- `save_to_file` Required, Boolean, Default is TRUE, but if not, returns result to memory

**Value**

path to output file

**Examples**

```r
## Not run:
ocrFile(file_path = "path_to_ocr_file", output_dir = "path_to_output_dir")

## End(Not run)
```
processBarcodeField  

Process Bar Code Field

Description

Process the bar code field in an image.

Usage

processBarcodeField(file_path = "", barcodeType = "autodetect",  
region = "-1,-1,-1,-1", containsBinaryData = "false",  
pdfPassword = "", description = "", ...)

Arguments

file_path  
path of the document

barcodeType  
optional, default: "autodetect"

region  
coordinates of region from top left, 4 values: top left bottom right; optional;  
default: "-1,-1,-1,-1" (entire image)

containsBinaryData  
optional, default: "false"

dfPassword  
optional, default: ""

description  
optional, default: ""

...  
Additional arguments passed to abbyy_POST.

Value

Data frame with details of the task associated with the submitted Image

References

http://ocrsdk.com/documentation/apireference/processBarcodeField/

Examples

## Not run:
processBarcodeField(file_path = "file_path")

## End(Not run)
**processBusinessCard**  

**Process Business Card**

**Description**

Processes a Business Card

**Usage**

```
processBusinessCard(file_path = "", language = "English",
imageSource = "auto", correctOrientation = "true",
correctSkew = "true", exportFormat = "vCard", description = "",
pdfPassword = "", ...)
```

**Arguments**

- **file_path**: required, path of the document, default: ""
- **language**: optional, default: English
- **imageSource**: optional, default: auto
- **correctOrientation**: optional, default: true
- **correctSkew**: optional, default: true
- **exportFormat**: optional, default: "vCard"
- **description**: optional, default: ""
- **pdfPassword**: optional, default: NULL
- ... Additional arguments passed to `abbyy_POST`.

**Value**

Data frame with details of the task associated with the submitted Business Card

**References**


**Examples**

```r
## Not run:
processBusinessCard(file_path="file_path", language="English")

## End(Not run)
```
### processCheckmarkField

#### Description

Processes Checkmark Field

#### Usage

```r
processCheckmarkField(file_path = "", checkmarkType = "empty",
region = "-1,-1,-1,-1", correctionAllowed = "false",
pdfPassword = "", description = "", ...)
```

#### Arguments

- **file_path**: required, path of the document, default: ""
- **checkmarkType**: optional, default: "empty"
- **region**: coordinates of region from top left, 4 values: top left bottom right; optional; default: ":-1,-1,-1,-1" (entire image)
- **correctionAllowed**: optional, default: "false"
- **pdfPassword**: optional, default: ""
- **description**: optional, default: ""
- ... Additional arguments passed to `abbyy_POST`.

#### Value

Data frame with details of the task associated with the submitted Image

#### References

[http://ocrsdk.com/documentation/apireference/processCheckmarkField/](http://ocrsdk.com/documentation/apireference/processCheckmarkField/)

For supported image types, see [http://ocrsdk.com/documentation/specifications/image-formats/](http://ocrsdk.com/documentation/specifications/image-formats/)

#### Examples

```r
## Not run:
processCheckmarkField(file_path = "file_path")

## End(Not run)
```
processDocument

**Description**

This function processes several images for the same task and results in a multi-page document. For instance, upload pages of the book individually via submitImage to the same task. And then process it via ProcessDocument to get a multi-page pdf.

**Usage**

```r
processDocument(taskId = NULL, language = "English", 
profile = c("documentConversion", "documentArchiving", 
"textExtraction", "fieldLevelRecognition", "barcodeRecognition"),
textType = c("normal", "typewriter", "matrix", "index", "ocrA", "ocrB", 
"e13b", "cmc7", "gothic"), imageSource = c("auto", "photo", "scanner"),
correctOrientation = c("true", "false"), correctSkew = c("true", 
"false"), readBarcodes = c("false", "true"), exportFormat = c("txt", 
"txtUnstructured", "rtf", "docx", "xlsx", "pptx", "pdfSearchable", 
"pdfTextAndImages", "pdfa", "xml", "xmlForCorrectedImage", "alto"),
description = NULL, pdfPassword = NULL, ...)
```

**Arguments**

- `taskId` Only tasks with Submitted, Completed or NotEnoughCredits status can be processed using this function.
- `language` String. Optional; default: English
- `profile` String. Optional; default: documentConversion Options: documentConversion, documentArchiving, textExtraction, fieldLevelRecognition, barcodeRecognition
- `textType` String. Optional; default: normal Options: normal, typewriter, matrix, index, ocrA, ocrB, e13b,cmc7, gothic
- `imageSource` String. Optional; default: auto Options: auto, photo, scanner
- `correctOrientation` String. Optional; default: true. Options: true or false
- `correctSkew` String. Optional; default: true. Options: true or false
- `readBarcodes` Optional; Options: true or false
- `exportFormat` optional, default: txt Options: txt, txtUnstructured, rtf, docx, xlsx, pptx, pdfSearchable, pdfTextAndImages, pdfa, xml, xmlForCorrectedImage, alto
- `description` Optional; default: ""
- `pdfPassword` Optional; default: NULL

... Additional arguments passed to abbyy_GET.

**Value**

data.frame with details of the task associated with the submitted Document
**processFields**

**References**

http://ocrsdk.com/documentation/apireference/processDocument/

**Examples**

```r
## Not run:
processDocument(taskId = "task_id")
## End(Not run)
```

---

**processFields**

**Process Fields**

**Description**

This function gets Information about a particular application

**Usage**

```r
processFields(file_path = "", taskId = NULL, description = "", ...)
```

**Arguments**

- **file_path**: path of the document
- **taskId**: Only tasks with Submitted, Completed or NotEnoughCredits status can be processed using this function.
- **description**: optional, default: ""
- ... Additional arguments passed to `abbyy_POST`.

**Value**

`data.frame` with details of the task associated with the submitted Image

**References**

http://ocrsdk.com/documentation/apireference/processFields/

**Examples**

```r
## Not run:
processFields(file_path = "file_path", taskId = "task_id", description = "")
## End(Not run)
```
processImage  

**Process Image**

**Description**

This function processes an image

**Usage**

```r
processImage(file_path = "", language = "English",
profile = c("documentConversion", "documentArchiving",
"textExtraction", "barcodeRecognition"), textType = c("normal",
"typewriter", "matrix", "index", "ocrA", "ocrB", "e13b", "cmc7",
"gothic"), imageSource = c("auto", "photo", "scanner"),
correctOrientation = c("true", "false"), correctSkew = c("true",
"false"), readBarcodes = c("false", "true"), exportFormat = c("txt",
"txtUnstructured", "rtf", "docx", "xlsx", "pptx", "pdfSearchable",
"pdfTextAndImages", "pdfa", "xml", "xmlForCorrectedImage", "alto"),
description = "", pdfPassword = "", ...)```

**Arguments**

- `file_path`: path to the document
- `language`: optional, default: English
- `textType`: String. Optional; default: `normal` Options: `normal`, `typewriter`, `matrix`, `index`, `ocrA`, `ocrB`, `e13b`, `cmc7`, `gothic`
- `imageSource`: String. Optional; default: `auto` Options: `auto`, `photo`, `scanner`
- `correctOrientation`: String. Optional; default: `true`. Options: `true` or `false`
- `correctSkew`: String. Optional; default: `true`. Options: `true` or `false`
- `readBarcodes`: Optional; Options: `true` or `false`
- `exportFormat`: optional, default: `txt` options: `txt`, `txtUnstructured`, `rtf`, `docx`, `xlsx`, `pptx`, `pdfSearchable`, `pdfTextAndImages`, `pdfa`, `xml`, `xmlForCorrectedImage`, `alto`
- `description`: optional, default: ""
- `pdfPassword`: optional, default: NULL
- ...

Additional arguments passed to `abbyy_POST`.

**Value**

A data.frame with details of the task associated with the submitted Image

**References**

processMRZ

**Description**

Extract data from Machine Readable Zone in an Image

**Usage**

```r
processMRZ(file_path = "", ...)  
```

**Arguments**

- `file_path` : path to the document
- `...` : Additional arguments passed to `abbyy_POST`.

**Value**

Data frame with details of the task associated with the submitted MRZ document

**References**

http://ocrsdk.com/documentation/apireference/processMRZ/

**Examples**

```r
## Not run:
processMRZ(file_path = "file_path")
```

## End(Not run)
processPhotoId

Description
Get data from a Photo ID. The function is under testing and may not work fully.

Usage
processPhotoId(file_path = "", idType = "auto", imageSource = "auto", correctOrientation = "true", correctSkew = "true", description = "", pdfPassword = "", ...)

Arguments
- file_path: path to file; required
- idType: optional; default = "auto"
- imageSource: optional; default = "auto"
- correctOrientation: String. Optional; default: true. Options: true or false
- correctSkew: String. Optional; default: true. Options: true or false
- description: optional; default = ""
- pdfPassword: optional; default = ""
- ... Additional arguments passed to abbyy_POST.

Value
Data frame with details of the task associated with the submitted Photo ID image

References
http://ocrsdk.com/documentation/apireference/processPhotoId/

Examples
```r
## Not run:
processPhotoId(file_path = "file_path", idType = "auto", imageSource = "auto")

## End(Not run)
```
processRemoteImage  

**Description**

This function gets information about a particular application.

**Usage**

```r
processRemoteImage(img_url = NULL, language = "English", 
                   profile = c("documentConversion", "documentArchiving", 
                                "textExtraction", "fieldLevelRecognition", "barcodeRecognition"),
                   textType = c("normal", "typewriter", "matrix", "index", "ocrA", "ocrB", 
                                "e13b", "cmc7", "gothic"),
                   imageSource = c("auto", "photo", "scanner"),
                   correctOrientation = c("true", "false"),
                   correctSkew = c("true", "false"),
                   readBarcodes = c("false", "true"),
                   exportFormat = c("txt", "txtUnstructured", "rtf", "docx", "xlsx", "pptx", "pdfSearchable", 
                                     "pdfTextAndImages", "pdfa", "xml", "xmlForCorrectedImage", "alto"),
                   description = NULL, pdfPassword = NULL, ...)
```

**Arguments**

- `img_url`: Required; url to remote image.
- `language`: Optional; default: English.
- `correctOrientation`: String. Optional; default: true. Options: true or false.
- `correctSkew`: String. Optional; default: true. Options: true or false.
- `readBarcodes`: Optional; default:
- `exportFormat`: Optional; default: txt. Must be one of the following: txt, rtf, docx, xlsx, pptx, pdfSearchable, pdfTextAndImages, pdfa, xml, alto.
- `description`: Optional; default: ""
- `pdfPassword`: Optional; default: NULL.
- `...`: Additional arguments passed to abbyy_GET.

**Value**

Data frame with details of the task associated with the submitted Remote Image.

**References**

processTextField 

**Examples**

```r
## Not run:
processRemoteImage(img_url = "img_url")
## End(Not run)
```

**processTextField**  
**Process Text Field**

**Description**

This function gets information about a particular application.

**Usage**

```r
processTextField(file_path = "", region = "-1,-1,-1,-1",
language = "English", letterSet = "", regExp = "",
textType = "normal", oneTextLine = "false",
oneWordPerTextLine = "false", markingType = "simpleText",
placeholdersCount = "1", writingStyle = "default",
description = "", pdfPassword = "", ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file_path</td>
<td>path of the document</td>
</tr>
<tr>
<td>region</td>
<td>coordinates of region from top left, 4 values: top left bottom right; optional; default: &quot;-1,-1,-1,-1&quot; (entire image)</td>
</tr>
<tr>
<td>language</td>
<td>optional; default: &quot;English&quot;</td>
</tr>
<tr>
<td>letterSet</td>
<td>letterset to be used for recognition, set by language but can be customized; optional; default: &quot;&quot;</td>
</tr>
<tr>
<td>regExp</td>
<td>which words are allowed in the field. see regular expression documentation; optional; default: &quot;&quot;</td>
</tr>
<tr>
<td>textType</td>
<td>type of the text in the field including typewriter, handprinted; optional; default: &quot;normal&quot;</td>
</tr>
<tr>
<td>oneTextLine</td>
<td>field contains only one text line or more; optional; default: &quot;false&quot;</td>
</tr>
<tr>
<td>oneWordPerTextLine</td>
<td>field contains one word per line or not; optional; default: &quot;false&quot;</td>
</tr>
<tr>
<td>markingType</td>
<td>only for handprint recognition, includes underlinedText etc.; optional; default: &quot;simpleText&quot;</td>
</tr>
<tr>
<td>placeholdersCount</td>
<td>No. of character cells for the field; optional; default: &quot;1&quot;</td>
</tr>
<tr>
<td>writingStyle</td>
<td>handprint writing style, see Abbyy FineReader documentation for values; optional; default: &quot;default&quot;</td>
</tr>
<tr>
<td>description</td>
<td>Description of processing task; optional; default: &quot;&quot;</td>
</tr>
<tr>
<td>pdfPassword</td>
<td>Password for pdf; optional; default: &quot;&quot;</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments passed to <code>abbyy_POST</code>.</td>
</tr>
</tbody>
</table>
setapp

**Value**
Data frame with details of the task associated with the submitted Image

**References**

**Examples**
```r
## Not run:
processTextField(file_path="file_path")
## End(Not run)
```

---

**setapp** *(Sets Application ID and Password)*

**Description**
Set Application ID and Password. Needed for interfacing with Abbyy FineReader Cloud OCR SDK. Run this before anything else.

**Usage**
```r
setapp(appdetails = NULLL force = FALSE)
```

**Arguments**
- `appdetails` A vector of app_id, app_password. Get these from [http://ocrsdk.com/](http://ocrsdk.com/). Set them before you use other functions.
- `force` Force change the app_id and app_password stored in the environment

**Details**
The function looks for AbbyyAppId and AbbyyAppPassword in the environment. If it doesn’t find them and if we don’t want to force change in them, it looks for arguments. And if no arguments are passed, it asks for user to input the values.

**References**

**Examples**
```r
## Not run:
setapp(c("app_id", "app_password"))
## End(Not run)
```
submitImage

Description

Add image to the existing task or creates a new task for the uploaded image. The new task isn’t processed till processDocument or processFields is called.

Usage

submitImage(file_path = ", taskId = ", pdfPassword = ", ...)

Arguments

file_path Required; Path to the document
taskId Optional; Assigns image to the task ID specified. If an empty string is passed, a new task is created.
pdfPassword Optional; If the pdf is password protected, put the password here.
... Additional arguments passed to abbyy_POST.

Value

Data frame with all the details of the submitted image: id (task id), registrationTime, statusChangeTime, status (Submitted, Queued, InProgress, Completed, ProcessingFailed, Deleted, NotEnoughCredits), filesCount (No. of files), credits

References

http://ocrsdk.com/documentation/apireference/submitImage/

Examples

## Not run:
submitImage(file_path="/images/image1.png", taskId="task_id", pdfPassword="pdf_password")

## End(Not run)
Index

*Topic Application
  setapp, 20
*Topic ID
  setapp, 20
*Topic Image
  processBusinessCard, 11
*Topic Password
  setapp, 20
*Topic Process
  processBusinessCard, 11
*Topic Remote
  processBusinessCard, 11
*Topic Sets
  setapp, 20
*Topic and
  setapp, 20

abbyy_check, 3
abbyy_GET, 3, 4, 5, 7, 8, 13, 18
abbyy_POST, 4, 10–12, 14–17, 19, 21
abbyyR (abbyyR-package), 2
abbyyR-package, 2

deleteTask, 4

GET, 3
getAppInfo, 5
getResult, 6
getTaskStatus, 4, 7

listFinishedTasks, 6, 7
listTasks, 8

ocrFile, 9

POST, 4
processBarcodeField, 10
processBusinessCard, 11
processCheckmarkField, 12
processDocument, 13
processFields, 14

processImage, 15
processMRZ, 16
processPhotoId, 17
processRemoteImage, 18
processTextField, 19

setapp, 2, 20
submitImage, 21