

Package ‘CoreAPI’

October 4, 2016

Type Package

Title Core API fucntions

Version 0.1.0

Author Craig Parman

Maintainer Craig Parman <cparman@coreinformatics.com>

Description The package provides funtions for intacting with the Core API.

License GPL-2 | GPL-3

LazyData TRUE

Imports httr,jsonlite

Suggests testthat, knitr, rmarkdown

RoxygenNote 5.0.1

VignetteBuilder knitr

RemoteType github

RemoteHost https://api.github.com

RemoteRepo CoreAPI

RemoteUsername ceparman

RemoteRef master

RemoteSha 5c3f146223788d0f9388c2427fd43ee334116c6e

GithubRepo CoreAPI

GithubUsername ceparman

GithubRef master

GithubSHA1 5c3f146223788d0f9388c2427fd43ee334116c6e

R topics documented:

apiCall	2
attachFile	3
authBasic	4
buildUrl	5

coreAPI	6
createEntity	7
createExperimentSample	8
createSampleLot	9
experimentPublish	10
experimentUnPublish	11
getCellContents	12
getContainerLineage	13
getEntityByBarcode	14
getEntityById	15
getExperimentSamples	16
logOut	17
transferCellContents	18
updateCellContents	19
updateEntity	20
updateExperimentContainers	21
updateExperimentSampleData	22
Index	24

apiCall	<i>apiCall - Base call to Core REST API.</i>
---------	----------------------------------------------

Description

apiCall base call to Core REST API.

Usage

apiCall(coreApi, body, encode, special = NULL, useVerbose = FALSE)

Arguments

- | | |
|------------|---------------------------------------------------------------------------|
| coreApi | coreApi object with valid jsessionid |
| body | body for request |
| encode | encoding to use for request option are "multipart", "form", "json", "raw" |
| special | - passed to buildUrl for special sdk endpoints |
| useVerbose | Use verbose communication for debugging |

Details

apiCall base call to Core REST API.

Value

RETURN return the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
response <-CoreAPI::apiCall(login$coreApi,body,"json",,special=NULL,useVerbose=FALSE)
logOut(login$coreApi )

## End(Not run)
```

attachFile

attachFile - attaches a file to an entity or attribute.

Description

attachFile Attaches a file to entity identified by barcode or one of its attributes.

Usage

```
attachFile(coreApi, barcode, filename, filepath, targetAttributeName = "",
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
barcode	User provided barcode as a character string
filename	name to use for the attached file
filepath	path to the file to attach
targetAttributeName	- if included the name of the attribute to attach the file to. Must be in all caps.
useVerbose	Use verbose communication for debugging

Details

attachFile attaches a file to an entity or attribute.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
newitem<-CoreAPI::attachFile(response$coreApi,barcode,filename,
                             filepath,targetAttributeName="",useVerbose=FALSE)
logOut(login$coreApi )

## End(Not run)
```

authBasic	<i>authBasic - Authenticates against the LIMS using basic authentication.</i>
-----------	-------------------------------------------------------------------------------

Description

authBasic Logs in and returns a fully populated coreApi object in \$coreAPI.

Usage

```
authBasic(coreApi, useVerbose = FALSE)
```

Arguments

coreApi	object of class coreApi that contains user, password, baseURL and account. account is required if user has access to multiple tenants.
useVerbose	- Use verbose settings for HTTP commands

Details

authBasic authenticates to Core API

Value

returns a list with coreApi which returns the passed coreApi object with jsessionid, awsseb and employeeid populated, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
response<- CoreAPI::authBasic(api)
logOut(response$coreApi,useVerbose=TRUE )

## End(Not run)
```

buildUrl	<i>buildUrl - build URL for call to Core REST API.</i>
----------	--------------------------------------------------------

Description

buildUrl build URL for call to Core REST API.

Usage

```
buildUrl(coreApi, special = NULL, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
useVerbose	Use verbose communication for debugging

Details

apiCall base call to Core REST API.

Value

RETURN Core REST URL

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
response <-CoreAPI::apiCall(login$coreApi,body,"json",useVerbose=FALSE)
logOut(login$coreApi )

## End(Not run)
```

coreAPI	<i>coreAPI - Creates a object of class coreAPI that contains user and connection information.</i>
---------	---------------------------------------------------------------------------------------------------

Description

coreAPI - Creates a object of class coreAPI that contains user and connection information.

Usage

coreAPI(CoreAccountInfo)

Arguments

CoreAccountInfo
file with account information in json format.

Details

Creates a object of class coreAPI that contains user name, password base url, and account if needed. It has slots for account, jsessionID, AWSELB, and base URL. Requires a json file that contains the user pwd and account.

```
#'Creates a object of class coreAPI that contains account information
coreAPI("path to json")
```

The json must include the fields shown below. The account value may be set to "" if the user only has access to one tenant.

Example json object.

```
[{      "user": "xxxxxxxxx",      "pwd": "xxxxxxxxx",      "coreUrl": "xxxxx
```

Value

Object of class coreAPI

Examples

```
## Not run:
api<-coreApi("/home")

## End(Not run)
```

createEntity	<i>createEntity - Create a new instance of a entity.</i>
--------------	----------------------------------------------------------

Description

createEntity Creates a new entity instance. Required inputs are url, jsessionId and entityType.

Usage

```
createEntity(coreApi, entityType, attributeValues = NULL, locationId = NULL,
  projectIds = NULL, barcode = NULL, associations = NULL,
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get as character string
attributeValues	atributes as a list of key-vlaues pairs
locationId	location ID for inital location as character string
projectIds	project comma seperated list of project IDs as character string
barcode	User provided barcode as a character string
associations	association as a list of dataframes (see details)
useVerbose	Use verbose communitcaion for debuggins

Details

createEntity Creates a new entity instance.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
item<-createEntityt(login$coreApi,"Entity_Type")
logOut(login$coreApi )

## End(Not run)
```

createExperimentSample

createExperimentSample- Creates an experiment sample.

Description

createExperimentSample Creates a new sample lot using the parent sample barcode

Usage

```
createExperimentSample(coreApi, entityType, sampleLotBarcode, experimentBarcode,
    attributeValues = NULL, locationId = NULL, projectIds = NULL,
    barcode = NULL, associations = NULL, concentration = NULL,
    concentrationUnit = NULL, timeMin = NULL, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get as character string
sampleLotBarcode	parent sample barcode
experimentBarcode	experiment barcode
attributeValues	atributes as a list of key-vlaues pairs
locationId	location ID for inital location as character string
projectIds	project comma seperated list of project IDs as character string
barcode	User provided barcode as a character string
associations	association as a list of dataframes (see details)
concentration	sample lot concentration
concentrationUnit	concentration unit
timeMin	min time value
useVerbose	Use verbose communitcaion for debuggins

Details

createExperimentSample Creates an experiment sample.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
item<-createExperimentSample(login$coreApi,
logOut(login$coreApi )

## End(Not run)
```

createSampleLot	<i>createSampleLot - Creates a lot of a sample.</i>
-----------------	-----------------------------------------------------

Description

createSampleLot Creates a new sample lot using the parent sample barcode

Usage

```
createSampleLot(coreApi, entityType, sampleBarcode, attributeValues = NULL,
  locationId = NULL, projectIds = NULL, barcode = NULL,
  associations = NULL, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get as character string
sampleBarcode	parent sample barcode
attributeValues	atributes as a list of key-vlaues pairs
locationId	location ID for initial location as character string
projectIds	project comma seperated list of project IDs as character string
barcode	User provided barcode as a character string
associations	association as a list of dataframes (see details)
useVerbose	Use verbose communitcaion for debuggins

Details

createSampleLot Creates a lot of a sample.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
item<-createSampleLot(login$coreApi,"Sample_Lot_Name","sample_barcode"
logOut(login$coreApi )

## End(Not run)
```

experimentPublish	experimentPublish <i>publishes an experiment.</i>
-------------------	---------------------------------------------------

Description

experimentPublish - publishes an experiment.

Usage

```
experimentPublish(coreApi, experimentType, experimentBarcode,
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
experimentType	experiment entity type
experimentBarcode	barcode of the experiment
useVerbose	Use verbose communication for debugging

Value

RETURN returns a list \$entity contains updated experiment information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
update<- CoreAPI::experimentPublish(login$coreApi,experimentType, exptbarcode,useVerbose = TRUE)
logOut(login$coreApi )

## End(Not run)
```

experimentUnPublish	experimentUnPublish <i>unpublishes an experiment.</i>
---------------------	-------------------------------------------------------

Description

experimentUnPublish - publishes an experiment.

Usage

```
experimentUnPublish(coreApi, experimentType, experimentBarcode,
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
experimentType	experiment entity type
experimentBarcode	barcode of the experiment
useVerbose	Use verbose communication for debugging

Value

RETURN returns a list \$entity contains updated experiment information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
update<- CoreAPI::experimentUnPublish(login$coreApi,experimentType, exptbarcode,useVerbose = TRUE)
logOut(login$coreApi )

## End(Not run)
```

getCellContents	<i>getCellContents - Get information about container cell contents.</i>
-----------------	-------------------------------------------------------------------------

Description

getCellContents - Get information about container cell contents.

Usage

```
getCellContents(coreApi, containerBarcode, containerCellNum,  
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
containerBarcode	container barcode
containerCellNum	container cel number as a string
useVerbose	Use verbose communication for debugging

Details

getCellContents puts a cell lot in a container cell

Value

RETURN returns a list \$entity contains cell information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:  
api<-CoreAPI("PATH TO JSON FILE")  
login<- CoreAPI::authBasic(api)  
cell<-getCellContents(login$coreApi,"VIA9","1")  
logOut(login$coreApi )  
  
## End(Not run)
```

getContainerLineage	<i>getContainerLineage - Get lineage for a container by barcode</i>
---------------------	---------------------------------------------------------------------

Description

getContainerLineage Get the parent and child containers for a container referenced by barcode..

Usage

```
getContainerLineage(coreApi, barcode, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
barcode	barcode of container to get lineage for
useVerbose	TRUE or FALSE to indicate if verbose options should be used in http POST

Details

getContainerLineage get an entity from the LIMS by barcode and

Value

returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
lineage<-getContainerLineage(login$coreApi,"barcode")
parents <- ineage$entity$parents
logOut(login$coreApi )

## End(Not run)
```

getEntityByBarcode	<i>getEntityByBarcode - Get an entity from the LIMS by barcode.</i>
--------------------	---------------------------------------------------------------------

Description

getEntityByBarcode Get an entity from the LIMS by barcode and entityType.

Usage

```
getEntityByBarcode(coreApi, entityType, barcode, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get
barcode	barcode of entity to get
useVerbose	TRUE or FALSE to indicate if verbose options should be used in http POST

Details

getEntityByBarcode get an entity from the LIMS by barcode and

Value

returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
item<-getEntityByBarcode(login$coreApi,"entityType","barcode")
logOut(login$coreApi)

## End(Not run)
```

getEntityById	<i>getEntityById - Get an entity from the LIMS by entity ID.</i>
---------------	------------------------------------------------------------------

Description

getEntityById Get an entity from the LIMS by ID and entityType.

Usage

```
getEntityById(coreApi, entityType, entityId, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get
entityId	ID of entity to get
useVerbose	TRUE or FALSE to indicate if verbose options should be used in http POST

Details

getEntityById get an entity from the LIMS by ID.

Value

returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
item<-getEntityById(login$coreApi,"entityType","entityId")
logOut(login$coreApi)

## End(Not run)
```

getExperimentSamples *getExperimentSamples - Gets experiment and experiment samples from experiment identified by barcode.*

Description

getExperimentSamples Gets experiment and experiment samples from experiment identified by barcode.

Usage

```
getExperimentSamples(coreApi, entityType, barcode, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	experiment entity type to get
barcode	barcode of entity to get
useVerbose	TRUE or FALSE to indicate if verbose options should be used in http POST

Details

getExperimentSamples Gets experiment and experiment samples from experiment identified by barcode.

Value

returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
experiment<- getExperimentSamples(login$coreApi,"entityType","barcode")
logOut(login$coreApi)

## End(Not run)
```

logOut	<i>logOut -Log user out of the LIMS.</i>
--------	------------------------------------------

Description

logOut logs out of the current jsession.

Usage

```
logOut(coreApi, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object returned during log in
useVerbose	use verbose option for debuggin in http POST

Details

logOut logs user out of the LIMS using Core API

Value

returns list with \$success = TRUE when sucessful, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
response<- CoreAPI::authBasic(api)
logOut(response$coreApi,useVerbose=TRUE )

## End(Not run)
```

transferCellContents *transferCellContents - Transfers contents from one cell to another.*

Description

transferCellContents - Get information about container cell contents.

Usage

```
transferCellContents(coreApi, sourceCellID, destCellID, amount, concentration,
  amountUnit, concentrationUnit, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
sourceCellID	source cell ID
destCellID	destination cell ID
amount	amount to transfer
concentration	concentration for destination cell
amountUnit	valid units for amount
concentrationUnit	valid units for concentration
useVerbose	use verbose messaging for debugging

Details

transferCellContents puts a cell lot in a container cell

Value

RETURN returns a list \$entity contains cell information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
newcell<-transferCellContents(login$coreApi,"98811","93221","1",".1","mg/ML","uL")
logOut(login$coreApi )

## End(Not run)
```

updateCellContents	<i>updateCellContents - Puts a cell lot in a container cell.</i>
--------------------	------------------------------------------------------------------

Description

updateCellContents - Puts a cell lot in a container cell.

Usage

```
updateCellContents(coreApi, containerType, containerBarcode, containerCellNum,  
    sampleLotBarcode, amount, amountUnit, concentration, concentrationUnit,  
    useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
containerType	container entity type
containerBarcode	container barcode
containerCellNum	container cel number
sampleLotBarcode	barcode of lot to add to cell
amount	amount to add (numeric)
amountUnit	units
concentration	(numeric)
concentrationUnit	concentration units

Details

updateCellContents puts a cell lot in a container cell

Value

RETURN returns a list \$entity contains updated container information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
cell<-
logOut(login$coreApi )

## End(Not run)
```

updateEntity	<i>updateEntity - Create a new instance of a entity.</i>
--------------	----------------------------------------------------------

Description

updateEntity Creates a new entity instance.

Usage

```
updateEntity(coreApi, entityType, barcode, attributeValues = NULL,
  locationId = NULL, projectIds = NULL, associations = NULL,
  useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get as character string
barcode	User provided barcode as a character string
attributeValues	attributes as a list of key-values pairs
locationId	location ID for initial location as character string
projectIds	project comma separated list of project IDs as character string
associations	association as a list of dataframes (see details)
useVerbose	Use verbose communication for debugging

Details

updateEntity Creates a new entity instance.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
newitem<-CoreAPI::updateEntity(login$coreApi,entityType,barcode)
logOut(login$coreApi ) response<- CoreAPI::authBasic(coreApi)

## End(Not run)
```

updateExperimentContainers

updateExperimentContainers - Adds a container to an experiment.

Description

updateExperimentContainers - Adds a container to an experiment.

Usage

```
updateExperimentContainers(coreApi, containerBarcode, experimentType,
  experimentBarcode, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
containerBarcode	barcode of container that has a cell lot in it
experimentType	experiment entity type
experimentBarcode	barcode of the experiment
useVerbose	Use verbose communication for debugging

Details

updateExperimentContainers puts a cell lot in a container cell

Value

RETURN returns a list \$entity contains updated experiment information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:
api<-CoreAPI("PATH TO JSON FILE")
login<- CoreAPI::authBasic(api)
update<- CoreAPI::updateExperimentContainers(login$coreApi,containerBarcode, experimentType, newExptbarcode,use
logOut(login$coreApi )

## End(Not run)
```

updateExperimentSampleData

updateExperimentSampleData - Upate experiment sample data.

Description

updateExperimentSampleData Update experiment sample data.

Usage

```
updateExperimentSampleData(coreApi, entityType, experimentSamplebarcode,
  assayAttributeValues, useVerbose = FALSE)
```

Arguments

coreApi	coreApi object with valid jsessionid
entityType	entity type to get as character string
experimentSamplebarcode	User provided barcode as a character string
assayAttributeValues	assay attributes as a list of key-values pairs
useVerbose	Use verbose communication for debugging

Details

updateExperimentSampleData Upate experiment sample data.

Value

RETURN returns a list \$entity contains entity information, \$response contains the entire http response

Author(s)

Craig Parman

Examples

```
## Not run:  
api<-CoreAPI("PATH TO JSON FILE")  
login<- CoreAPI::authBasic(api)  
newdata<-CoreAPI::updateExperimentSampleData(login$coreApi,entityType,experimentSampleBarcode,assayAttributeVal  
logOut(login$coreApi ) response<- CoreAPI::authBasic(coreApi)  
  
## End(Not run)
```

Index

apiCall, [2](#)
attachFile, [3](#)
authBasic, [4](#)

buildUrl, [5](#)

coreAPI, [6](#)
createEntity, [7](#)
createExperimentSample, [8](#)
createSampleLot, [9](#)

experimentPublish, [10](#)
experimentUnPublish, [11](#)

getCellContents, [12](#)
getContainerLineage, [13](#)
getEntityByBarcode, [14](#)
getEntityById, [15](#)
getExperimentSamples, [16](#)

logOut, [17](#)

transferCellContents, [18](#)

updateCellContents, [19](#)
updateEntity, [20](#)
updateExperimentContainers, [21](#)
updateExperimentSampleData, [22](#)