

Package ‘notebookutils’

January 26, 2026

Title Dummy R APIs Used in 'Azure Synapse Analytics' for Local Developments

Version 1.6.2

Description

This is a pure dummy interfaces package which mirrors 'MsSparkUtils' APIs <<https://learn.microsoft.com/en-us/azure/synapse-analytics/spark/microsoft-spark-utilities?pivots=programming-language-r>> of 'Azure Synapse Analytics' <<https://learn.microsoft.com/en-us/azure/synapse-analytics/>> for R users, customer of Azure Synapse can download this package from CRAN for local development.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

Maintainer runtimeexp <runtimeexpdg@microsoft.com>

Author runtimeexp [aut, cre],
Microsoft [cph]

NeedsCompilation no

Repository CRAN

Date/Publication 2026-01-26 08:40:02 UTC

Contents

display	4
display.config	5
display.configCustomOptions	6
displayHTML	6
mssparkutils.credentials.getConnectionStringOrCreds	7
mssparkutils.credentials.getFullConnectionString	7
mssparkutils.credentials.getPropertiesAll	8
mssparkutils.credentials.getSecret	8
mssparkutils.credentials.getSecretWithLS	9

mssparkutils.credentials.getToken	10
mssparkutils.credentials.help	10
mssparkutils.credentials.isValidToken	11
mssparkutils.credentials.putSecret	11
mssparkutils.credentials.putSecretWithLS	12
mssparkutils.env.getClusterId	13
mssparkutils.env.getJobId	13
mssparkutils.env.getPoolName	14
mssparkutils.env.getUserId	14
mssparkutils.env.getUserName	15
mssparkutils.env.getWorkspaceName	15
mssparkutils.env.help	16
mssparkutils.fs.append	16
mssparkutils.fs.cp	17
mssparkutils.fs.exists	17
mssparkutils.fs.fastcp	18
mssparkutils.fs.getMountPath	19
mssparkutils.fs.head	19
mssparkutils.fs.help	20
mssparkutils.fs.ls	21
mssparkutils.fs.mkdirs	21
mssparkutils.fs.mount	22
mssparkutils.fs.mounts	22
mssparkutils.fs.mountToDriverNode	23
mssparkutils.fs.mv	23
mssparkutils.fs.put	24
mssparkutils.fs.refreshMounts	25
mssparkutils.fs.rm	25
mssparkutils.fs.unmount	26
mssparkutils.fs.unmountFromDriverNode	26
mssparkutils.help	27
mssparkutils.lakehouse.create	27
mssparkutils.lakehouse.delete	28
mssparkutils.lakehouse.get	28
mssparkutils.lakehouse.help	29
mssparkutils.lakehouse.list	29
mssparkutils.lakehouse.update	30
mssparkutils.nbResPath	30
mssparkutils.notebook.exit	31
mssparkutils.notebook.help	31
mssparkutils.notebook.run	32
mssparkutils.notebook.runMultiple	33
mssparkutils.notebook.updateNBSEndpoint	33
mssparkutils.notebook.validateDAG	34
mssparkutils.runtime.context	34
mssparkutils.runtime.setHcReplId	35
mssparkutils.session.stop	35
notebookutils.conf.get	36

notebookutils.conf.set	36
notebookutils.connections.getCredential	37
notebookutils.connections.getHelpString	37
notebookutils.credentials.getConnectionStringOrCreds	38
notebookutils.credentials.getFullConnectionString	38
notebookutils.credentials.getPropertiesAll	39
notebookutils.credentials.getSecret	39
notebookutils.credentials.getSecretWithLS	40
notebookutils.credentials.getToken	41
notebookutils.credentials.help	41
notebookutils.credentials.isValidToken	42
notebookutils.credentials.putSecret	42
notebookutils.credentials.putSecretWithLS	43
notebookutils.env.getClusterId	44
notebookutils.env.getJobId	44
notebookutils.env.getPoolName	45
notebookutils.env.getUserId	45
notebookutils.env.getUserName	46
notebookutils.env.getWorkspaceName	46
notebookutils.env.help	47
notebookutils.fabricClient.delete	47
notebookutils.fabricClient.get	48
notebookutils.fabricClient.help	48
notebookutils.fabricClient.listCapacities	49
notebookutils.fabricClient.patch	49
notebookutils.fabricClient.post	50
notebookutils.fabricClient.put	50
notebookutils.fs.append	51
notebookutils.fs.cp	51
notebookutils.fs.exists	52
notebookutils.fs.fastcp	53
notebookutils.fs.getMountPath	53
notebookutils.fs.head	54
notebookutils.fs.help	55
notebookutils.fs.ls	55
notebookutils.fs.mkdirs	56
notebookutils.fs.mount	57
notebookutils.fs.mounts	57
notebookutils.fs.mountToDriverNode	58
notebookutils.fs.mv	58
notebookutils.fs.put	59
notebookutils.fs.refreshMounts	60
notebookutils.fs.rm	60
notebookutils.fs.unmount	61
notebookutils.fs.unmountFromDriverNode	61
notebookutils.help	62
notebookutils.initialize	62
notebookutils.lakehouse.create	63

notebookutils.lakehouse.delete	63
notebookutils.lakehouse.get	64
notebookutils.lakehouse.getDefinition	64
notebookutils.lakehouse.getWithProperties	65
notebookutils.lakehouse.help	65
notebookutils.lakehouse.list	66
notebookutils.lakehouse.listTables	66
notebookutils.lakehouse.loadTable	67
notebookutils.lakehouse.update	67
notebookutils.lakehouse.updateDefinition	68
notebookutils.nbResPath	68
notebookutils.notebook.create	69
notebookutils.notebook.delete	69
notebookutils.notebook.exit	70
notebookutils.notebook.get	70
notebookutils.notebook.help	71
notebookutils.notebook.list	71
notebookutils.notebook.run	72
notebookutils.notebook.update	73
notebookutils.notebook.updateDefinition	73
notebookutils.notebook.updateNBSEndpoint	74
notebookutils.notebook.validateDAG	74
notebookutils.runtime.context	75
notebookutils.runtime.help	76
notebookutils.runtime.setHcReplId	76
notebookutils.session.stop	77
notebookutils.udf.getFunctions	77
notebookutils.udf.getHelpString	78
notebookutils.udf.help	78
notebookutils.udf.run	79
notebookutils.variableLibrary.get	79
notebookutils.variableLibrary.getHelpString	80
notebookutils.variableLibrary.getLibrary	80

Index 81

display	<i>Set the dataframe info which needs to be visualized.</i>
---------	---

Description

Set the dataframe info which needs to be visualized.

Usage

```
display(dataFrame, isSummary = FALSE)
```

Arguments

dataFrame	the dataframe that needs to be visualized.
isSummary	whether show summary info of the dataframe.

Value

No return value, show the first part of passed dataframe.

Examples

```
data <- list(56,78,90,45,67)
df <- data.frame(t(sapply(data,c)))
display(df)
display(df, TRUE)
```

display.config	<i>Set the chart config metadata for current dataframe (set by display) which needs to be visualized.</i>
----------------	---

Description

Set the chart config metadata for current dataframe (set by display) which needs to be visualized.

Usage

```
display.config(
  commId,
  lastCommId = NULL,
  binsNumber = 10,
  category = "table",
  keys = NULL,
  values = NULL,
  series = NULL,
  aggregation = NULL,
  column = NULL
)
```

Arguments

commId	the id used to identify whether the API call from synapse notebook js client.
lastCommId	same with id parameter, but the previous value.
binsNumber	bins number for rendering histogram, default is 10.
category	the chart category as bar, line, default is table.
keys	the column names which useds to render x-axis.
values	the column names which used to render y-axis.

series	the column which used to render the chart series
aggregation	the aggregation operation type: sum, avg, min, max, count.
column	will deperated: the column name used to calculate the statistic info, as the column type, unique values, missing values, etc.

`display.configCustomOptions`

Set the visualizer custom options related to the current dataframe, it will be consumed when executing display.

Description

Set the visualizer custom options related to the current dataframe, it will be consumed when executing display.

Usage

```
display.configCustomOptions(options)
```

Arguments

options	custom options serialized from JSON format
---------	--

displayHTML	<i>Construct an specific html fragment to synapse notebook front-end for rendering based on user-input html content.</i>
-------------	--

Description

Construct an specific html fragment to synapse notebook front-end for rendering based on user-input html content.

Usage

```
displayHTML(content)
```

Arguments

content	html content which user want to render
---------	--

Value

No return value, print the content to mimic the render behavior when used in azure synapse runtime.

Examples

```
displayHTML('<b>Hello world!</b>')
```

```
mssparkutils.credentials.getConnectionStringOrCreds
```

Take linked service name as input and return connection string or credentials depending on the configuration of the linked service.

Description

Take linked service name as input and return connection string or credentials depending on the configuration of the linked service.

Usage

```
mssparkutils.credentials.getConnectionStringOrCreds(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic credentials returned by azure synapse runtime for linkedService.

Examples

```
mssparkutils.credentials.getConnectionStringOrCreds('AzureDataLakeStorage1')
```

```
mssparkutils.credentials.getFullConnectionString
```

Take linked service name as input and return full connection string with credentials.

Description

Take linked service name as input and return full connection string with credentials.

Usage

```
mssparkutils.credentials.getFullConnectionString(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic connection string returned by azure synapse runtime for linkedService.

Examples

```
mssparkutils.credentials.getConnectionStringOrCreds('AzureDataLakeStorage1')
```

```
mssparkutils.credentials.getPropertiesAll
```

Return all the properties of a given linked service in string format.

Description

Return all the properties of a given linked service in string format.

Usage

```
mssparkutils.credentials.getPropertiesAll(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic properties string returned by azure synapse runtime for linkedService.

Examples

```
mssparkutils.credentials.getPropertiesAll('AzureDataLakeStorage1')
```

```
mssparkutils.credentials.getSecret
```

Return AKV secret.

Description

Return AKV secret.

Usage

```
mssparkutils.credentials.getSecret(akvName, secret, linkedService = NULL)
```

Arguments

akvName Azure Key Vault name.

secret name of the secret being fetched.

linkedService linkedService name of the AKV linked service.

Value

A empty string used to mimic secret returned by azure synapse runtime for given akvName and secret.

Examples

```
mssparkutils.credentials.getSecret('akvName', 'secretName')  
mssparkutils.credentials.getSecret('akvName', 'secretName', 'AzureDataLakeStorage1')
```

```
mssparkutils.credentials.getSecretWithLS
```

Return AKV secret using linkedService.

Description

Return AKV secret using linkedService.

Usage

```
mssparkutils.credentials.getSecretWithLS(linkedService, secret)
```

Arguments

linkedService	linkedService name of the AKV linked service.
secret	name of the secret being fetched.

Value

A empty string used to mimic secret returned by azure synapse runtime for given linkedService and secret.

Examples

```
mssparkutils.credentials.getSecretWithLS('AzureDataLakeStorage1', 'secretName')
```

```
mssparkutils.credentials.getToken
```

Get AAD token for a resource.

Description

Get AAD token for a resource.

Usage

```
mssparkutils.credentials.getToken(audience, name = "")
```

Arguments

audience	token audience.
name	token audience.

Value

A empty string used to mimic token returned by azure synapse runtime for accessing resource audience.

Examples

```
mssparkutils.credentials.getToken('synapse')  
mssparkutils.credentials.getToken('storage')  
mssparkutils.credentials.getToken('storage', 'storage')
```

```
mssparkutils.credentials.help
```

Get help message.

Description

Get help message.

Usage

```
mssparkutils.credentials.help()
```

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils credentials module when used in azure synapse runtime.

Examples

```
mssparkutils.credentials.help()
```

```
mssparkutils.credentials.isValidToken
```

Returns true if the input token is valid (i.e, hasn't expired).

Description

Returns true if the input token is valid (i.e, hasn't expired).

Usage

```
mssparkutils.credentials.isValidToken(token)
```

Arguments

token	token to validate.
-------	--------------------

Value

FALSE to mimic the result if token is invalid.

Examples

```
mssparkutils.credentials.isValidToken('dummyToken')
```

```
mssparkutils.credentials.putSecret
```

Put AKV secret using with or without linkedService.

Description

Put AKV secret using with or without linkedService.

Usage

```
mssparkutils.credentials.putSecret(  
    akvName,  
    secretName,  
    secretValue,  
    linkedService = NULL  
)
```

Arguments

akvName	Azure Key Vault name.
secretName	name of the secret being written.
secretValue	value of the secret being written.
linkedService	name of the AKV linked service.

Value

The secretValue been written.

Examples

```
mssparkutils.credentials.putSecret('akvName', 'secretName', 'secretValue')
mssparkutils.credentials.putSecret('akvName', 'secretName', 'secretValue', 'AzureDataLakeStorage1')
```

```
mssparkutils.credentials.putSecretWithLS
```

Put AKV secret using linkedService.

Description

Put AKV secret using linkedService.

Usage

```
mssparkutils.credentials.putSecretWithLS(
  linkedService,
  secretName,
  secretValue
)
```

Arguments

linkedService	name of AKV linked service.
secretName	name of the secret being written.
secretValue	value of the secret being written.

Value

The secretValue been written.

Examples

```
mssparkutils.credentials.putSecretWithLS('AzureDataLakeStorage1', 'secretName', 'secretValue')
```

```
mssparkutils.env.getClusterId
```

Get cluster id.

Description

Get cluster id.

Usage

```
mssparkutils.env.getClusterId()
```

Value

A empty string used to mimic cluster id of azure synapse runtime.

Examples

```
mssparkutils.env.getClusterId()
```

```
mssparkutils.env.getJobId
```

Get job Id.

Description

Get job Id.

Usage

```
mssparkutils.env.getJobId()
```

Value

A empty string used to mimic the id of spark job been submitted to azure synapse runtime.

Examples

```
mssparkutils.env.getJobId()
```

```
mssparkutils.env.getPoolName
```

Get pool name.

Description

Get pool name.

Usage

```
mssparkutils.env.getPoolName()
```

Value

A empty string used to mimic the name of user's azure synapse spark pool.

Examples

```
mssparkutils.env.getPoolName()
```

```
mssparkutils.env.getUserId
```

Get user Id.

Description

Get user Id.

Usage

```
mssparkutils.env.getUserId()
```

Value

A empty string used to mimic the id of user.

Examples

```
mssparkutils.env.getUserId()
```

```
mssparkutils.env.getUserName
```

Get user name.

Description

Get user name.

Usage

```
mssparkutils.env.getUserName()
```

Value

A empty string used to mimic the name of user.

Examples

```
mssparkutils.env.getUserName()
```

```
mssparkutils.env.getWorkspaceName
```

Get workspace name.

Description

Get workspace name.

Usage

```
mssparkutils.env.getWorkspaceName()
```

Value

A empty string used to mimic the id of the user's azure synapse workspace.

Examples

```
mssparkutils.env.getWorkspaceName()
```

```
mssparkutils.env.help
```

Get help message.

Description

Get help message.

Usage

```
mssparkutils.env.help()
```

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils env module when used in azure synapse runtime.

Examples

```
mssparkutils.env.help()
```

```
mssparkutils.fs.append
```

Append the given String to a file, encoded in UTF-8.

Description

Append the given String to a file, encoded in UTF-8.

Usage

```
mssparkutils.fs.append(file, content, createFileIfNotExists = FALSE)
```

Arguments

file	FileSystem URI
content	Content needs to be append to file, encoded in System default charset.
createFileIfNotExists	If set to true, will firstly try to create file if not exists.

Value

FALSE to mimic the result if file content append fail.

Examples

```
mssparkutils.fs.append("/tmp/my-file", "Hello world!")  
mssparkutils.fs.append("/tmp/my-file", "Hello world!", TRUE)
```

mssparkutils.fs.cp	<i>Copies a file or directory, possibly across FileSystems.</i>
--------------------	---

Description

Copies a file or directory, possibly across FileSystems.

Usage

```
mssparkutils.fs.cp(from, to, recurse = FALSE)
```

Arguments

from	FileSystem URI of the source file or directory
to	FileSystem URI of the destination file or directory
recurse	if TRUE, all files and directories will be recursively copied

Value

FALSE to mimic the result if file or directory from fail to copy to to.

Examples

```
mssparkutils.fs.cp("/tmp/my-folder/a", "adls://xxx/tmp/b")  
mssparkutils.fs.cp("/tmp/my-folder/a", "adls://xxx/tmp/b", TRUE)
```

mssparkutils.fs.exists	<i>Check if a file or directory exists.</i>
------------------------	---

Description

Check if a file or directory exists.

Usage

```
mssparkutils.fs.exists(file)
```

Arguments

file	FileSystem URI
------	----------------

Value

TRUE if the file or directory exists

Examples

```
## Not run:
mssparkutils.fs.exists("/tmp/my-file")

## End(Not run)
```

mssparkutils.fs.fastcp

Copies a file or directory via azcopy, possibly across FileSystems.

Description

Copies a file or directory via azcopy, possibly across FileSystems.

Usage

```
mssparkutils.fs.fastcp(from, to, recurse = TRUE, extraConfigs = NULL)
```

Arguments

from	FileSystem URI of the source file or directory
to	FileSystem URI of the destination file or directory
recurse	if TRUE, all files and directories will be recursively copied
extraConfigs	extra configs for azcopy, includes flags, timeout, aadToken, sourceLinkedService, destinationLinkedService

Value

TRUE if all files were successfully copied

Examples

```
## Not run:
mssparkutils.fs.fastcp("file:/tmp/my-folder/a", "adls://xxx/tmp/b")

## End(Not run)
```

`mssparkutils.fs.getMountPath`*Gets the local path of the mount point.*

Description

Gets the local path of the mount point.

Usage

```
mssparkutils.fs.getMountPath(mountPoint, scope = "")
```

Arguments

mountPoint	The directory that was previously mounted.
scope	Mount point level, job or workspace, default is job.

Value

Empty string to mimic the local mounted path related to mountPoint.

Examples

```
mssparkutils.fs.getMountPath("/mnt")  
mssparkutils.fs.getMountPath("/mnt", " job")
```

`mssparkutils.fs.head` *Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8.*

Description

Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8.

Usage

```
mssparkutils.fs.head(file, maxBytes = 65535)
```

Arguments

file	FileSystem URI
maxBytes	Maximum number of bytes to read

Value

Empty string to mimic the returned content of file.

Examples

```
mssparkutils.fs.head("/tmp/my-folder/my-file")  
mssparkutils.fs.head("/tmp/my-folder/my-file", 1000)
```

mssparkutils.fs.help *mssparkutils.fs provides utilities for working with various FileSystems.*

Description

Below is overview about the available methods:

Usage

```
mssparkutils.fs.help(methodName = "")
```

Arguments

methodName method name to get more information.

Details

mssparkutils.fs.cp: Copies a file or directory, possibly across FileSystems mssparkutils.fs.mv: Moves a file or directory, possibly across FileSystems mssparkutils.fs.ls: Array -> Lists the contents of a directory mssparkutils.fs.mkdirs: Creates the given directory if it does not exist, also creating any necessary parent directories mssparkutils.fs.put: Writes the given String out to a file, encoded in UTF-8 mssparkutils.fs.head: Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8 mssparkutils.fs.append: Append the content to a file mssparkutils.fs.rm: Removes a file or directory

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils fs module when used in azure synapse runtime.

Examples

```
mssparkutils.fs.help()  
mssparkutils.fs.help("ls")
```

mssparkutils.fs.ls	<i>Lists the contents of a directory.</i>
--------------------	---

Description

Lists the contents of a directory.

Usage

```
mssparkutils.fs.ls(dir)
```

Arguments

dir	FileSystem URI
-----	----------------

Value

Empty list to mimic the file list under dir.

Examples

```
mssparkutils.fs.ls("/tmp/my-folder/")
```

mssparkutils.fs.mkdirs

*Creates the given directory if it does not exist, also creating any necessary parent * directories.*

Description

Creates the given directory if it does not exist, also creating any necessary parent * directories.

Usage

```
mssparkutils.fs.mkdirs(dir)
```

Arguments

dir	FileSystem URI
-----	----------------

Value

FALSE to mimic the result if dir creation fail.

Examples

```
mssparkutils.fs.mkdirs("/tmp/a/b/c")
```

`mssparkutils.fs.mount` *Attach remote storage (Blob, Gen2, Azure File Share) to all working nodes (driver node and worker nodes)*

Description

Attach remote storage (Blob, Gen2, Azure File Share) to all working nodes (driver node and worker nodes)

Usage

```
mssparkutils.fs.mount(source, mountPoint, extraConfigs = NULL)
```

Arguments

<code>source</code>	FileSystem URI that contains the source data.
<code>mountPoint</code>	The directory of remote source to mount the source.
<code>extraConfigs</code>	Extra configurations.

Value

FALSE to mimic the result if mountPoint creation fail.

Examples

```
mssparkutils.fs.mount("abfss://xxx.dfs.core.windows.net", "/mnt")
```

`mssparkutils.fs.mounts`

Show information about what is mounted. Any credentials used to mount the mount points listed will not be displayed.

Description

Show information about what is mounted. Any credentials used to mount the mount points listed will not be displayed.

Usage

```
mssparkutils.fs.mounts(extraConfigs = NULL)
```

Arguments

<code>extraConfigs</code>	Extra configurations.
---------------------------	-----------------------

Value

The list of MountPointInfo.

```
mssparkutils.fs.mountToDriverNode
```

Attach remote storage (Blob, Gen2, Azure File Share) to driver node

Description

Attach remote storage (Blob, Gen2, Azure File Share) to driver node

Usage

```
mssparkutils.fs.mountToDriverNode(source, mountPoint, extraConfigs = NULL)
```

Arguments

source	FileSystem URI that contains the source data.
mountPoint	The directory of remote source to mount the source.
extraConfigs	Extra configurations.

Value

TRUE if the path was successfully mounted.

```
mssparkutils.fs.mv
```

Moves a file or directory, possibly across FileSystems. For intra-FileSystem, it is implemented by hadoop fs rename operation. For inter-FileSystem, This is implemented as a copy followed by delete.

Description

Moves a file or directory, possibly across FileSystems. For intra-FileSystem, it is implemented by hadoop fs rename operation. For inter-FileSystem, This is implemented as a copy followed by delete.

Usage

```
mssparkutils.fs.mv(from, to, createPath = FALSE, overwrite = FALSE)
```

Arguments

from	FileSystem URI of the source file or directory.
to	FileSystem URI of the destination file or directory.
createPath	if TRUE, will firstly create the parent dir if not exists before move op.
overwrite	if TRUE, will overwrite the destination folder if exists.

Value

FALSE to mimic the result of mv operation fail.

Examples

```
mssparkutils.fs.mv("/tmp/my-folder/", "adls:/xxx/tmp/b")
```

mssparkutils.fs.put	<i>Writes the given String out to a file, encoded in UTF-8.</i>
---------------------	---

Description

Writes the given String out to a file, encoded in UTF-8.

Usage

```
mssparkutils.fs.put(file, content, overwrite = FALSE)
```

Arguments

file	FileSystem URI.
content	Content of file to write, encoded in System default charset.
overwrite	If set to TRUE, the file will be overwritten if it existed already. Note that if overwrite is TRUE and the the write fails, the original file. may still be deleted.

Value

FALSE to mimic the result of file put operation fail.

Examples

```
mssparkutils.fs.put("/tmp/my-file", "Hello world!", TRUE)
```

```
mssparkutils.fs.refreshMounts
```

Refresh workspace level mount points.

Description

Refresh workspace level mount points.

Usage

```
mssparkutils.fs.refreshMounts()
```

Value

FALSE to mimic the refreshMounts fail to refresh mount info.

Examples

```
mssparkutils.fs.refreshMounts()
```

```
mssparkutils.fs.rm
```

Removes a file or directory.

Description

Removes a file or directory.

Usage

```
mssparkutils.fs.rm(dir, recurse = FALSE)
```

Arguments

dir	FileSystem URI for a single file or a directory.
recurse	if TRUE, all files and directories will be recursively deleted.

Value

FALSE to mimic the result of dir deletion fail.

Examples

```
mssparkutils.fs.rm("/tmp/my-folder/", TRUE)
```

```
mssparkutils.fs.unmount
```

Removes a mount point.

Description

Removes a mount point.

Usage

```
mssparkutils.fs.unmount(mountPoint)
```

Arguments

mountPoint The directory that was previously mounted.

Value

FALSE to mimic the result of unmount mountPoint fail.

Examples

```
mssparkutils.fs.unmount("/mnt")
```

```
mssparkutils.fs.unmountFromDriverNode
```

Removes a mount point from driver node.

Description

Removes a mount point from driver node.

Usage

```
mssparkutils.fs.unmountFromDriverNode(mountPoint)
```

Arguments

mountPoint The directory that was previously mounted.

Value

TRUE if the mount point was successfully unmounted.

mssparkutils.help	<i>Get help message for this module.</i>
-------------------	--

Description

Get help message for this module.

Usage

```
mssparkutils.help(methodName = "")
```

Arguments

methodName	method name to get more information.
------------	--------------------------------------

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils module when used in azure synapse runtime.

Examples

```
mssparkutils.help()
```

mssparkutils.lakehouse.create	<i>Create a lakehouse</i>
-------------------------------	---------------------------

Description

Create a lakehouse

Usage

```
mssparkutils.lakehouse.create(  
  name,  
  description = "",  
  definition = "",  
  workspaceId = ""  
)
```

Arguments

name	Name of the lakehouse
description	Description of the lakehouse
definition	Definition of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

```
mssparkutils.lakehouse.delete
```

Delete a lakehouse

Description

Delete a lakehouse

Usage

```
mssparkutils.lakehouse.delete(name, workspaceId = "")
```

Arguments

name	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

```
mssparkutils.lakehouse.get
```

Get a lakehouse

Description

Get a lakehouse

Usage

```
mssparkutils.lakehouse.get(name = "", workspaceId = "")
```

Arguments

name	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

`mssparkutils.lakehouse.help`*The lakehouse module.*

Description

`mssparkutils.lakehouse.create(name: String, description: String, workspaceId: String): Lakehouse`
-> Create a lakehouse `mssparkutils.lakehouse.get(name: String, workspaceId: String): Lakehouse`
-> Get a lakehouse `mssparkutils.lakehouse.delete(name: String, workspaceId: String): void` ->
Delete a lakehouse `mssparkutils.lakehouse.update(name: String, newName: String, description:
String, workspaceId: String): Lakehouse` -> Update a lakehouse

Usage

```
mssparkutils.lakehouse.help(methodName = "")
```

Arguments

<code>methodName</code>	method name to get more information
-------------------------	-------------------------------------

`mssparkutils.lakehouse.list`*List all lakehouses*

Description

List all lakehouses

Usage

```
mssparkutils.lakehouse.list(workspaceId = "", maxResults = 1000L)
```

Arguments

<code>workspaceId</code>	Workspace id of the lakehouse, default to current workspace
<code>maxResults</code>	Maximum number of lakehouses to return, default to 1000

Value

A list of lakehouse objects

```
mssparkutils.lakehouse.update
```

Update a lakehouse

Description

Update a lakehouse

Usage

```
mssparkutils.lakehouse.update(  
  name,  
  newName,  
  description = "",  
  workspaceId = ""  
)
```

Arguments

name	Name of the lakehouse
newName	New name of the lakehouse
description	Description of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

```
mssparkutils.nbResPath
```

Gets the local path of the notebook resource.

Description

Gets the local path of the notebook resource.

Usage

```
mssparkutils.nbResPath()
```

Value

The local path of the notebook resource.

```
mssparkutils.notebook.exit
```

This method lets you exit a notebook with a value.

Description

This method lets you exit a notebook with a value.

Usage

```
mssparkutils.notebook.exit(value)
```

Arguments

value the value to return when exiting.

Value

No return value, mimic behavior to set the notebook run exit value using value.

Examples

```
mssparkutils.notebook.exit('exitVal')
```

```
mssparkutils.notebook.help
```

The notebook module.

Description

The notebook module.

Usage

```
mssparkutils.notebook.help(methodName = "")
```

Arguments

methodName method name to get more information.

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils notebook module when used in azure synapse runtime.

Examples

```
mssparkutils.notebook.help()  
mssparkutils.notebook.help("run")
```

```
mssparkutils.notebook.run
```

Runs a notebook and returns its exit value. The notebook will run in the current livy session context by default.

Description

Runs a notebook and returns its exit value. The notebook will run in the current livy session context by default.

Usage

```
mssparkutils.notebook.run(  
  path,  
  timeoutSeconds = 90,  
  arguments = NULL,  
  workspace = ""  
)
```

Arguments

path	absolute path to the notebook, e.g. /path/to/notebook.
timeoutSeconds	timeout in seconds for the called notebook.
arguments	string map of arguments to pass to the notebook.
workspace	Workspace of the notebook.

Value

Empty string to mimic the exitVal set by mssparkutils.notebook.exit.

Examples

```
mssparkutils.notebook.run('NB1')  
mssparkutils.notebook.run('NB1', 200)  
mssparkutils.notebook.run('NB1', 200, list("input"=30))
```

`mssparkutils.notebook.runMultiple`

Runs multiple notebooks concurrently with support for dependency relationships. Details can be found in `mssparkutils.notebook.help("runMultiple")`.

Description

Runs multiple notebooks concurrently with support for dependency relationships. Details can be found in `mssparkutils.notebook.help("runMultiple")`.

Usage

```
mssparkutils.notebook.runMultiple(pathsOrPipeline)
```

Arguments

`pathsOrPipeline`

A list of notebook names or a complex data structure (JSON string) that meets the requirements of the `com.microsoft.spark.notebook.msutils.impl.MsNotebookPipeline` scala class.

Value

a list of exit values and exceptions for each notebook

`mssparkutils.notebook.updateNBSEndpoint`

provide a way to make people can update the endpoint

Description

provide a way to make people can update the endpoint

Usage

```
mssparkutils.notebook.updateNBSEndpoint(endpoint)
```

Arguments

`endpoint` the new point

```
mssparkutils.notebook.validateDAG
```

Check if the DAG is correctly defined, if the syntax is correct, and if notebooks are found in the workspace without actual execution. Details can be found in `mssparkutils.notebook.help("validateDAG")`.

Description

Check if the DAG is correctly defined, if the syntax is correct, and if notebooks are found in the workspace without actual execution. Details can be found in `mssparkutils.notebook.help("validateDAG")`.

Usage

```
mssparkutils.notebook.validateDAG(dag)
```

Arguments

<code>dag</code>	A list of notebook names or a complex data structure (JSON string) that meets the requirements of the <code>com.microsoft.spark.notebook.msutils.impl.MsNotebookPipeline</code> scala class.
------------------	--

Value

check result

```
mssparkutils.runtime.context
```

Get runtime properties

Description

Get runtime properties

Usage

```
mssparkutils.runtime.context()
```

Value

A dummy env object to mimic the result of runtime context method when used in azure synapse runtime.

Examples

```
mssparkutils.runtime.context()
```

```
mssparkutils.runtime.setHcReplId
```

Set runtime high concurrency mode repl id

Description

Set runtime high concurrency mode repl id

Usage

```
mssparkutils.runtime.setHcReplId(replId)
```

Arguments

replId	High concurrency mode repl id
--------	-------------------------------

```
mssparkutils.session.stop
```

Stop an interactive session

Description

Stop an interactive session

Usage

```
mssparkutils.session.stop(detach = TRUE)
```

Arguments

detach	If detach is True, stop session from standard session, or detach current notebook from high concurrency session; if detach is False, stop session in any session. Default is TRUE.
--------	--

`notebookutils.conf.get`*Get a config value*

Description

Dummy in-memory implementation for local development.

Usage

```
notebookutils.conf.get(key, default = "")
```

Arguments

key	Config key
default	Default value when key is missing

`notebookutils.conf.set`*Set a config key/value*

Description

Dummy in-memory implementation for local development.

Usage

```
notebookutils.conf.set(key, value)
```

Arguments

key	Config key
value	Config value

```
notebookutils.connections.getCredential
```

Get credential for a connection

Description

Dummy implementation for local development.

Usage

```
notebookutils.connections.getCredential(connectionId, artifactId = "")
```

Arguments

connectionId	Connection id
artifactId	Artifact id

Value

A dummy credential object (list)

```
notebookutils.connections.getHelpString
```

Get help string for a method.

Description

Get help string for a method.

Usage

```
notebookutils.connections.getHelpString(funcName = "", namespace = "")
```

Arguments

funcName	function name
namespace	namespace

```
notebookutils.credentials.getConnectionStringOrCreds
```

Take linked service name as input and return connection string or credentials depending on the configuration of the linked service.

Description

Take linked service name as input and return connection string or credentials depending on the configuration of the linked service.

Usage

```
notebookutils.credentials.getConnectionStringOrCreds(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic credentials returned by azure synapse runtime for linkedService.

Examples

```
notebookutils.credentials.getConnectionStringOrCreds('AzureDataLakeStorage1')
```

```
notebookutils.credentials.getFullConnectionString
```

Take linked service name as input and return full connection string with credentials.

Description

Take linked service name as input and return full connection string with credentials.

Usage

```
notebookutils.credentials.getFullConnectionString(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic connection string returned by azure synapse runtime for linkedService.

Examples

```
notebookutils.credentials.getConnectionStringOrCreds('AzureDataLakeStorage1')
```

```
notebookutils.credentials.getPropertiesAll
```

Return all the properties of a given linked service in string format.

Description

Return all the properties of a given linked service in string format.

Usage

```
notebookutils.credentials.getPropertiesAll(linkedService)
```

Arguments

linkedService Linked service name.

Value

A empty string used to mimic properties string returned by azure synapse runtime for linkedService.

Examples

```
notebookutils.credentials.getPropertiesAll('AzureDataLakeStorage1')
```

```
notebookutils.credentials.getSecret
```

Return AKV secret.

Description

Return AKV secret.

Usage

```
notebookutils.credentials.getSecret(akvName, secret, linkedService = NULL)
```

Arguments

akvName Azure Key Vault name.

secret name of the secret being fetched.

linkedService linkedService name of the AKV linked service.

Value

A empty string used to mimic secret returned by azure synapse runtime for given akvName and secret.

Examples

```
notebookutils.credentials.getSecret('akvName', 'secretName')
notebookutils.credentials.getSecret('akvName', 'secretName', 'AzureDataLakeStorage1')
```

```
notebookutils.credentials.getSecretWithLS
```

Return AKV secret using linkedService.

Description

Return AKV secret using linkedService.

Usage

```
notebookutils.credentials.getSecretWithLS(linkedService, secret)
```

Arguments

linkedService	linkedService name of the AKV linked service.
secret	name of the secret being fetched.

Value

A empty string used to mimic secret returned by azure synapse runtime for given linkedService and secret.

Examples

```
notebookutils.credentials.getSecretWithLS('AzureDataLakeStorage1', 'secretName')
```

```
notebookutils.credentials.getToken
```

Get AAD token for a resource.

Description

Get AAD token for a resource.

Usage

```
notebookutils.credentials.getToken(audience, name = "")
```

Arguments

audience	token audience.
name	token audience.

Value

A empty string used to mimic token returned by azure synapse runtime for accessing resource audience.

Examples

```
notebookutils.credentials.getToken('synapse')
notebookutils.credentials.getToken('storage')
notebookutils.credentials.getToken('storage', 'storage')
```

```
notebookutils.credentials.help
```

Get help message.

Description

Get help message.

Usage

```
notebookutils.credentials.help()
```

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils credentials module when used in azure synapse runtime.

Examples

```
notebookutils.credentials.help()
```

```
notebookutils.credentials.isValidToken
```

Returns true if the input token is valid (i.e, hasn't expired).

Description

Returns true if the input token is valid (i.e, hasn't expired).

Usage

```
notebookutils.credentials.isValidToken(token)
```

Arguments

token	token to validate.
-------	--------------------

Value

FALSE to mimic the result if token is invalid.

Examples

```
notebookutils.credentials.isValidToken('dummyToken')
```

```
notebookutils.credentials.putSecret
```

Put AKV secret using with or without linkedService.

Description

Put AKV secret using with or without linkedService.

Usage

```
notebookutils.credentials.putSecret(  
    akvName,  
    secretName,  
    secretValue,  
    linkedService = NULL  
)
```

Arguments

akvName	Azure Key Vault name.
secretName	name of the secret being written.
secretValue	value of the secret being written.
linkedService	name of the AKV linked service.

Value

The secretValue been written.

Examples

```
notebookutils.credentials.putSecret('akvName', 'secretName', 'secretValue')
notebookutils.credentials.putSecret('akvName', 'secretName', 'secretValue', 'AzureDataLakeStorage1')
```

```
notebookutils.credentials.putSecretWithLS
```

Put AKV secret using linkedService.

Description

Put AKV secret using linkedService.

Usage

```
notebookutils.credentials.putSecretWithLS(
    linkedService,
    secretName,
    secretValue
)
```

Arguments

linkedService	name of AKV linked service.
secretName	name of the secret being written.
secretValue	value of the secret being written.

Value

The secretValue been written.

Examples

```
notebookutils.credentials.putSecretWithLS('AzureDataLakeStorage1', 'secretName', 'secretValue')
```

```
notebookutils.env.getClusterId
```

Get cluster id.

Description

Get cluster id.

Usage

```
notebookutils.env.getClusterId()
```

Value

A empty string used to mimic cluster id of azure synapse runtime.

Examples

```
notebookutils.env.getClusterId()
```

```
notebookutils.env.getJobId
```

Get job Id.

Description

Get job Id.

Usage

```
notebookutils.env.getJobId()
```

Value

A empty string used to mimic the id of spark job been submitted to azure synapse runtime.

Examples

```
notebookutils.env.getJobId()
```

```
notebookutils.env.getPoolName
```

Get pool name.

Description

Get pool name.

Usage

```
notebookutils.env.getPoolName()
```

Value

A empty string used to mimic the name of user's azure synapse spark pool.

Examples

```
notebookutils.env.getPoolName()
```

```
notebookutils.env.getUserId
```

Get user Id.

Description

Get user Id.

Usage

```
notebookutils.env.getUserId()
```

Value

A empty string used to mimic the id of user.

Examples

```
notebookutils.env.getUserId()
```

```
notebookutils.env.getUserName
```

Get user name.

Description

Get user name.

Usage

```
notebookutils.env.getUserName()
```

Value

A empty string used to mimic the name of user.

Examples

```
notebookutils.env.getUserName()
```

```
notebookutils.env.getWorkspaceName
```

Get workspace name.

Description

Get workspace name.

Usage

```
notebookutils.env.getWorkspaceName()
```

Value

A empty string used to mimic the id of the user's azure synapse workspace.

Examples

```
notebookutils.env.getWorkspaceName()
```

```
notebookutils.env.help
```

Get help message.

Description

Get help message.

Usage

```
notebookutils.env.help()
```

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils env module when used in azure synapse runtime.

Examples

```
notebookutils.env.help()
```

```
notebookutils.fabricClient.delete
```

Send a DELETE request to Fabric.

Description

Send a DELETE request to Fabric.

Usage

```
notebookutils.fabricClient.delete(path, headers = list())
```

Arguments

path	Path of the request
headers	Headers of the request

```
notebookutils.fabricClient.get
```

Send a GET request to Fabric.

Description

Send a GET request to Fabric.

Usage

```
notebookutils.fabricClient.get(path, headers = list())
```

Arguments

path	Path of the request
headers	Headers of the request

Value

RestResponse Response of the request

```
notebookutils.fabricClient.help
```

Get help string for a method.

Description

Get help string for a method.

Usage

```
notebookutils.fabricClient.help(methodName = "")
```

Arguments

methodName	Name of the method
------------	--------------------

`notebookutils.fabricClient.listCapacities`*List all capacities in the workspace.*

Description

List all capacities in the workspace.

Usage

```
notebookutils.fabricClient.listCapacities(maxResults = 1000L)
```

Arguments

<code>maxResults</code>	Maximum number of capacities to return, default is 1000
-------------------------	---

Value

Array of Capacity objects

`notebookutils.fabricClient.patch`*Send a PATCH request to Fabric.*

Description

Send a PATCH request to Fabric.

Usage

```
notebookutils.fabricClient.patch(path, content, headers = list())
```

Arguments

<code>path</code>	Path of the request
<code>content</code>	Content of the request
<code>headers</code>	Headers of the request

Value

RestResponse Response of the request

notebookutils.fabricClient.post

Send a POST request to Fabric.

Description

Send a POST request to Fabric.

Usage

notebookutils.fabricClient.post(path, content, headers = list())

Arguments

path	Path of the request
content	Content of the request
headers	Headers of the request

Value

RestResponse Response of the request

notebookutils.fabricClient.put

Send a PUT request to Fabric.

Description

Send a PUT request to Fabric.

Usage

notebookutils.fabricClient.put(path, content, headers = list())

Arguments

path	Path of the request
content	Content of the request
headers	Headers of the request

Value

RestResponse Response of the request

`notebookutils.fs.append`*Append the given String to a file, encoded in UTF-8.*

Description

Append the given String to a file, encoded in UTF-8.

Usage

```
notebookutils.fs.append(file, content, createFileIfNotExists = FALSE)
```

Arguments

<code>file</code>	FileSystem URI
<code>content</code>	Content needs to be append to file, encoded in System default charset.
<code>createFileIfNotExists</code>	If set to true, will firstly try to create file if not exists.

Value

FALSE to mimic the result if file content append fail.

Examples

```
notebookutils.fs.append("/tmp/my-file", "Hello world!")  
notebookutils.fs.append("/tmp/my-file", "Hello world!", TRUE)
```

`notebookutils.fs.cp` *Copies a file or directory, possibly across FileSystems.*

Description

Copies a file or directory, possibly across FileSystems.

Usage

```
notebookutils.fs.cp(from, to, recurse = FALSE)
```

Arguments

<code>from</code>	FileSystem URI of the source file or directory
<code>to</code>	FileSystem URI of the destination file or directory
<code>recurse</code>	if TRUE, all files and directories will be recursively copied

Value

FALSE to mimic the result if file or directory from fail to copy to to.

Examples

```
notebookutils.fs.cp("/tmp/my-folder/a", "adls://xxx/tmp/b")
notebookutils.fs.cp("/tmp/my-folder/a", "adls://xxx/tmp/b", TRUE)
```

```
notebookutils.fs.exists
```

Check if a file or directory exists.

Description

Check if a file or directory exists.

Usage

```
notebookutils.fs.exists(file)
```

Arguments

file	FileSystem URI
------	----------------

Value

TRUE if the file or directory exists

Examples

```
## Not run:
notebookutils.fs.exists("/tmp/my-file")

## End(Not run)
```

`notebookutils.fs.fastcp`*Copies a file or directory via azcopy, possibly across FileSystems.*

Description

Copies a file or directory via azcopy, possibly across FileSystems.

Usage

```
notebookutils.fs.fastcp(from, to, recurse = TRUE, extraConfigs = NULL)
```

Arguments

<code>from</code>	FileSystem URI of the source file or directory
<code>to</code>	FileSystem URI of the destination file or directory
<code>recurse</code>	if TRUE, all files and directories will be recursively copied
<code>extraConfigs</code>	extra configs for azcopy, includes flags, timeout, aadToken, sourceLinkedService, destinationLinkedService

Value

TRUE if all files were successfully copied

Examples

```
## Not run:  
notebookutils.fs.fastcp("file:/tmp/my-folder/a", "adls://xxx/tmp/b")  
  
## End(Not run)
```

`notebookutils.fs.getMountPath`*Gets the local path of the mount point.*

Description

Gets the local path of the mount point.

Usage

```
notebookutils.fs.getMountPath(mountPoint, scope = "")
```

Arguments

mountPoint	The directory that was previously mounted.
scope	Mount point level, job or workspace, default is job.

Value

Empty string to mimic the local mounted path related to mountPoint.

Examples

```
notebookutils.fs.getMountPath("/mnt")  
notebookutils.fs.getMountPath("/mnt", "job")
```

notebookutils.fs.head *Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8.*

Description

Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8.

Usage

```
notebookutils.fs.head(file, maxBytes = 65535)
```

Arguments

file	FileSystem URI
maxBytes	Maximum number of bytes to read

Value

Empty string to mimic the returned content of file.

Examples

```
notebookutils.fs.head("/tmp/my-folder/my-file")  
notebookutils.fs.head("/tmp/my-folder/my-file", 1000)
```

notebookutils.fs.help *notebookutils.fs provides utilities for working with various FileSystems.*

Description

Below is overview about the available methods:

Usage

```
notebookutils.fs.help(methodName = "")
```

Arguments

methodName method name to get more information.

Details

notebookutils.fs.cp: Copies a file or directory, possibly across FileSystems notebookutils.fs.mv: Moves a file or directory, possibly across FileSystems notebookutils.fs.ls: Array -> Lists the contents of a directory notebookutils.fs.mkdir: Creates the given directory if it does not exist, also creating any necessary parent directories notebookutils.fs.put: Writes the given String out to a file, encoded in UTF-8 notebookutils.fs.head: Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8 notebookutils.fs.append: Append the content to a file notebookutils.fs.rm: Removes a file or directory

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils fs module when used in azure synapse runtime.

Examples

```
notebookutils.fs.help()
notebookutils.fs.help("ls")
```

notebookutils.fs.ls *Lists the contents of a directory.*

Description

Lists the contents of a directory.

Usage

```
notebookutils.fs.ls(dir)
```

Arguments

dir FileSystem URI

Value

Empty list to mimic the file list under dir.

Examples

```
notebookutils.fs.ls("/tmp/my-folder/")
```

```
notebookutils.fs.mkdirs
```

*Creates the given directory if it does not exist, also creating any necessary parent * directories.*

Description

Creates the given directory if it does not exist, also creating any necessary parent * directories.

Usage

```
notebookutils.fs.mkdirs(dir)
```

Arguments

dir FileSystem URI

Value

FALSE to mimic the result if dir creation fail.

Examples

```
notebookutils.fs.mkdirs("/tmp/a/b/c")
```

`notebookutils.fs.mount`

Attach remote storage (Blob, Gen2, Azure File Share) to all working nodes (driver node and worker nodes)

Description

Attach remote storage (Blob, Gen2, Azure File Share) to all working nodes (driver node and worker nodes)

Usage

```
notebookutils.fs.mount(source, mountPoint, extraConfigs = NULL)
```

Arguments

<code>source</code>	FileSystem URI that contains the source data.
<code>mountPoint</code>	The directory of remote source to mount the source.
<code>extraConfigs</code>	Extra configurations.

Value

FALSE to mimic the result if mountPoint creation fail.

Examples

```
notebookutils.fs.mount("abfss://xxx.dfs.core.windows.net", "/mnt")
```

`notebookutils.fs.mounts`

Show information about what is mounted. Any credentials used to mount the mount points listed will not be displayed.

Description

Show information about what is mounted. Any credentials used to mount the mount points listed will not be displayed.

Usage

```
notebookutils.fs.mounts(extraConfigs = NULL)
```

Arguments

<code>extraConfigs</code>	Extra configurations.
---------------------------	-----------------------

Value

The list of MountPointInfo.

```
notebookutils.fs.mountToDriverNode
```

Attach remote storage (Blob, Gen2, Azure File Share) to driver node

Description

Attach remote storage (Blob, Gen2, Azure File Share) to driver node

Usage

```
notebookutils.fs.mountToDriverNode(source, mountPoint, extraConfigs = NULL)
```

Arguments

source	FileSystem URI that contains the source data.
mountPoint	The directory of remote source to mount the source.
extraConfigs	Extra configurations.

Value

TRUE if the path was successfully mounted.

```
notebookutils.fs.mv
```

Moves a file or directory, possibly across FileSystems. For intra-FileSystem, it is implemented by hadoop fs rename operation. For inter-FileSystem, This is implemented as a copy followed by delete.

Description

Moves a file or directory, possibly across FileSystems. For intra-FileSystem, it is implemented by hadoop fs rename operation. For inter-FileSystem, This is implemented as a copy followed by delete.

Usage

```
notebookutils.fs.mv(from, to, createPath = FALSE, overwrite = FALSE)
```

Arguments

from	FileSystem URI of the source file or directory.
to	FileSystem URI of the destination file or directory.
createPath	if TRUE, will firstly create the parent dir if not exists before move op.
overwrite	if TRUE, will overwrite the destination folder if exists.

Value

FALSE to mimic the result of mv operation fail.

Examples

```
notebookutils.fs.mv("/tmp/my-folder/", "adls:/xxx/tmp/b")
```

```
notebookutils.fs.put
```

Writes the given String out to a file, encoded in UTF-8.

Description

Writes the given String out to a file, encoded in UTF-8.

Usage

```
notebookutils.fs.put(file, content, overwrite = FALSE)
```

Arguments

file	FileSystem URI.
content	Content of file to write, encoded in System default charset.
overwrite	If set to TRUE, the file will be overwritten if it existed already. Note that if overwrite is TRUE and the the write fails, the original file. may still be deleted.

Value

FALSE to mimic the result of file put operation fail.

Examples

```
notebookutils.fs.put("/tmp/my-file", "Hello world!", TRUE)
```

`notebookutils.fs.refreshMounts`
Refresh workspace level mount points.

Description

Refresh workspace level mount points.

Usage

`notebookutils.fs.refreshMounts()`

Value

FALSE to mimic the refreshMounts fail to refresh mount info.

Examples

`notebookutils.fs.refreshMounts()`

`notebookutils.fs.rm` *Removes a file or directory.*

Description

Removes a file or directory.

Usage

`notebookutils.fs.rm(dir, recurse = FALSE)`

Arguments

<code>dir</code>	FileSystem URI for a single file or a directory.
<code>recurse</code>	if TRUE, all files and directories will be recursively deleted.

Value

FALSE to mimic the result of dir deletion fail.

Examples

`notebookutils.fs.rm("/tmp/my-folder/", TRUE)`

```
notebookutils.fs.unmount
```

Removes a mount point.

Description

Removes a mount point.

Usage

```
notebookutils.fs.unmount(mountPoint)
```

Arguments

mountPoint The directory that was previously mounted.

Value

FALSE to mimic the result of unmount mountPoint fail.

Examples

```
notebookutils.fs.unmount("/mnt")
```

```
notebookutils.fs.unmountFromDriverNode
```

Removes a mount point from driver node.

Description

Removes a mount point from driver node.

Usage

```
notebookutils.fs.unmountFromDriverNode(mountPoint)
```

Arguments

mountPoint The directory that was previously mounted.

Value

TRUE if the mount point was successfully unmounted.

<code>notebookutils.help</code>	<i>Get help message for this module.</i>
---------------------------------	--

Description

Get help message for this module.

Usage

```
notebookutils.help(methodName = "")
```

Arguments

<code>methodName</code>	method name to get more information.
-------------------------	--------------------------------------

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils module when used in azure synapse runtime.

Examples

```
notebookutils.help()
```

<code>notebookutils.initialize</code>	<i>Initialize notebookutils, using in pre-run code</i>
---------------------------------------	--

Description

Initialize notebookutils, using in pre-run code

Usage

```
notebookutils.initialize()
```

```
notebookutils.lakehouse.create
```

Create a lakehouse

Description

Create a lakehouse

Usage

```
notebookutils.lakehouse.create(  
    name,  
    description = "",  
    definition = "",  
    workspaceId = ""  
)
```

Arguments

name	Name of the lakehouse
description	Description of the lakehouse
definition	Definition of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

```
notebookutils.lakehouse.delete
```

Delete a lakehouse

Description

Delete a lakehouse

Usage

```
notebookutils.lakehouse.delete(name, workspaceId = "")
```

Arguments

name	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

```
notebookutils.lakehouse.get
```

Get a lakehouse

Description

Get a lakehouse

Usage

```
notebookutils.lakehouse.get(name = "", workspaceId = "")
```

Arguments

name	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

```
notebookutils.lakehouse.getDefinition
```

Get the definition of a lakehouse

Description

Get the definition of a lakehouse

Usage

```
notebookutils.lakehouse.getDefinition(name, workspaceId = "")
```

Arguments

name	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

The definition of the lakehouse

```
notebookutils.lakehouse.getWithProperties
```

Get the info of a Lakehouse with properties.

Description

Get the info of a Lakehouse with properties.

Usage

```
notebookutils.lakehouse.getWithProperties(name, workspaceId = "")
```

Arguments

name	Name of the Lakehouse.
workspaceId	Id of the workspace, default to current workspace.

Value

Artifact object. Please refer to: <https://learn.microsoft.com/en-us/rest/api/fabric/articles/item-management/properties/lakehouse-properties>

```
notebookutils.lakehouse.help
```

The lakehouse module.

Description

```
notebookutils.lakehouse.create(name: String, description: String, workspaceId: String): Lakehouse  
-> Create a lakehouse notebookutils.lakehouse.get(name: String, workspaceId: String): Lakehouse  
-> Get a lakehouse notebookutils.lakehouse.delete(name: String, workspaceId: String): void ->  
Delete a lakehouse notebookutils.lakehouse.update(name: String, newName: String, description:  
String, workspaceId: String): Lakehouse -> Update a lakehouse
```

Usage

```
notebookutils.lakehouse.help(methodName = "")
```

Arguments

methodName	method name to get more information
------------	-------------------------------------

```
notebookutils.lakehouse.list
```

List all lakehouses

Description

List all lakehouses

Usage

```
notebookutils.lakehouse.list(workspaceId = "", maxResults = 1000L)
```

Arguments

workspaceId	Workspace id of the lakehouse, default to current workspace
maxResults	Maximum number of lakehouses to return, default to 1000

Value

A list of lakehouse objects

```
notebookutils.lakehouse.listTables
```

List all tables in a Lakehouse.

Description

List all tables in a Lakehouse.

Usage

```
notebookutils.lakehouse.listTables(  
  lakehouse = "",  
  workspaceId = "",  
  maxResults = 1000L  
)
```

Arguments

lakehouse	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace
maxResults	Maximum number of tables to return, default to 1000

Value

A list of table objects

```
notebookutils.lakehouse.loadTable
```

Starts a load table operation.

Description

Starts a load table operation.

Usage

```
notebookutils.lakehouse.loadTable(  
    loadOption,  
    table,  
    lakehouse = "",  
    workspaceId = ""  
)
```

Arguments

loadOption	string, loadOption Load options. Please refer to https://learn.microsoft.com/en-us/rest/api/fabric/lakehouse/tables/load-table
table	Name of the table
lakehouse	Name of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

boolean

```
notebookutils.lakehouse.update
```

Update a lakehouse

Description

Update a lakehouse

Usage

```
notebookutils.lakehouse.update(  
    name,  
    newName,  
    description = "",  
    workspaceId = ""  
)
```

Arguments

name	Name of the lakehouse
newName	New name of the lakehouse
description	Description of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

A lakehouse object

```
notebookutils.lakehouse.updateDefinition
```

Get the definition of a lakehouse

Description

Get the definition of a lakehouse

Usage

```
notebookutils.lakehouse.updateDefinition(name, definition, workspaceId = "")
```

Arguments

name	Name of the lakehouse
definition	Definition of the lakehouse
workspaceId	Workspace id of the lakehouse, default to current workspace

Value

The definition of the lakehouse

```
notebookutils.nbResPath
```

Gets the local path of the notebook resource.

Description

Gets the local path of the notebook resource.

Usage

```
notebookutils.nbResPath()
```

Value

The local path of the notebook resource.

```
notebookutils.notebook.create
```

Create a notebook

Description

Create a notebook

Usage

```
notebookutils.notebook.create(  
    name,  
    description = "",  
    content = "",  
    defaultLakehouse = "",  
    defaultLakehouseWorkspace = "",  
    workspaceId = ""  
)
```

Arguments

name	Name of the notebook
description	Description of the notebook
content	Definition of the notebook
defaultLakehouse	Default lakehouse of the notebook
defaultLakehouseWorkspace	Default lakehouse workspace of the notebook
workspaceId	Workspace id of the notebook, default to current workspace

Value

A notebook object

```
notebookutils.notebook.delete
```

Delete a notebook

Description

Delete a notebook

Usage

```
notebookutils.notebook.delete(name, workspaceId = "")
```

Arguments

name	Name of the notebook
workspaceId	Workspace id of the notebook, default to current workspace

```
notebookutils.notebook.exit
```

This method lets you exit a notebook with a value.

Description

This method lets you exit a notebook with a value.

Usage

```
notebookutils.notebook.exit(value)
```

Arguments

value	the value to return when exiting.
-------	-----------------------------------

Value

No return value, mimic behavior to set the notebook run exit value using value.

Examples

```
notebookutils.notebook.exit('exitVal')
```

```
notebookutils.notebook.get
```

Get a notebook

Description

Get a notebook

Usage

```
notebookutils.notebook.get(name, workspaceId = "")
```

Arguments

name	Name of the notebook
workspaceId	Workspace id of the notebook, default to current workspace

Value

A notebook object

```
notebookutils.notebook.help
```

The notebook module.

Description

The notebook module.

Usage

```
notebookutils.notebook.help(methodName = "")
```

Arguments

methodName method name to get more information.

Value

No return value, print empty string to mimic the behavior of help method of mssparkutils notebook module when used in azure synapse runtime.

Examples

```
notebookutils.notebook.help()  
notebookutils.notebook.help("run")
```

```
notebookutils.notebook.list
```

List all notebooks

Description

List all notebooks

Usage

```
notebookutils.notebook.list(workspaceId = "", maxResults = 1000L)
```

Arguments

workspaceId Workspace id of the notebook, default to current workspace
maxResults Maximum number of notebooks to return, default to 1000

Value

A list of notebook objects

```
notebookutils.notebook.run
```

Runs a notebook and returns its exit value. The notebook will run in the current livy session context by default.

Description

Runs a notebook and returns its exit value. The notebook will run in the current livy session context by default.

Usage

```
notebookutils.notebook.run(
  path,
  timeoutSeconds = 90,
  arguments = NULL,
  workspace = ""
)
```

Arguments

path	absolute path to the notebook, e.g. /path/to/notebook.
timeoutSeconds	timeout in seconds for the called notebook.
arguments	string map of arguments to pass to the notebook.
workspace	Workspace of the notebook.

Value

Empty string to mimic the exitVal set by mssparkutils.notebook.exit.

Examples

```
notebookutils.notebook.run('NB1')
notebookutils.notebook.run('NB1', 200)
notebookutils.notebook.run('NB1', 200, list("input"=30))
```

```
notebookutils.notebook.update
```

Update a notebook

Description

Update a notebook

Usage

```
notebookutils.notebook.update(  
    name,  
    newName,  
    description = "",  
    workspaceId = ""  
)
```

Arguments

name	Name of the notebook
newName	New name of the notebook
description	Description of the notebook
workspaceId	Workspace id of the notebook, default to current workspace

Value

A notebook object

```
notebookutils.notebook.updateDefinition
```

Get the definition of a notebook

Description

Get the definition of a notebook

Usage

```
notebookutils.notebook.updateDefinition(  
    name,  
    content,  
    defaultLakehouse = "",  
    defaultLakehouseWorkspace = "",  
    workspaceId = "",  
    environmentId = "",  
    environmentWorkspaceId = ""  
)
```

Arguments

name	Name of the notebook
content	Definition of the notebook
defaultLakehouse	Default lakehouse of the notebook
defaultLakehouseWorkspace	Default lakehouse workspace of the notebook
workspaceId	Workspace id of the notebook, default to current workspace
environmentId	Environment id of the notebook, default to current environment
environmentWorkspaceId	Environment workspace id of the notebook

Value

The definition of the notebook

notebookutils.notebook.updateNBSEndpoint
provide a way to make people can update the endpoint

Description

provide a way to make people can update the endpoint

Usage

notebookutils.notebook.updateNBSEndpoint(endpoint)

Arguments

endpoint	the new point
----------	---------------

notebookutils.notebook.validateDAG
Check if the DAG is correctly defined, if the syntax is correct, and if notebooks are found in the workspace without actual execution. Details can be found in mssparkutils.notebook.help("validateDAG").

Description

Check if the DAG is correctly defined, if the syntax is correct, and if notebooks are found in the workspace without actual execution. Details can be found in mssparkutils.notebook.help("validateDAG").

Usage

```
notebookutils.notebook.validateDAG(dag)
```

Arguments

dag	A list of notebook names or a complex data structure (JSON string) that meets the requirements of the <code>com.microsoft.spark.notebook.msutils.impl.MsNotebookPipeline</code> scala class.
-----	--

Value

check result

```
notebookutils.runtime.context
```

Get runtime properties

Description

Get runtime properties

Usage

```
notebookutils.runtime.context()
```

Value

A dummy env object to mimic the result of runtime context method when used in azure synapse runtime.

Examples

```
notebookutils.runtime.context()
```

```
notebookutils.runtime.help
```

*notebookutils.runtime is a utility to manage runtime context. context()
returns the runtime context as a list.*

Description

notebookutils.runtime is a utility to manage runtime context. context() returns the runtime context as a list.

Usage

```
notebookutils.runtime.help(methodName = "")
```

Arguments

methodName	method name to get more information.am
------------	--

```
notebookutils.runtime.setHcReplId
```

Set runtime high concurrency mode repl id

Description

Set runtime high concurrency mode repl id

Usage

```
notebookutils.runtime.setHcReplId(replId)
```

Arguments

replId	High concurrency mode repl id
--------	-------------------------------

notebookutils.session.stop
<i>Stop an interactive session</i>

Description

Stop an interactive session

Usage

notebookutils.session.stop(detach = TRUE)

Arguments

detach	If detach is True, stop session from standard session, or detach current notebook from high concurrency session; if detach is False, stop session in any session. Default is TRUE.
--------	--

notebookutils.udf.getFunctions
<i>Get UDF functions metadata.</i>

Description

Dummy implementation for local development.

Usage

notebookutils.udf.getFunctions(udf, workspaceId = "")

Arguments

udf	UDF artifact id or name
workspaceId	Workspace id

Value

A dummy UDF object (list)

notebookutils.udf.getHelpString
Get help string for a method.

Description

Get help string for a method.

Usage

```
notebookutils.udf.getHelpString(funcName = "", namespace = "")
```

Arguments

funcName	function name
namespace	namespace

notebookutils.udf.help
Get help string for a method.

Description

Get help string for a method.

Usage

```
notebookutils.udf.help(methodName = "")
```

Arguments

methodName	Name of the method
------------	--------------------

notebookutils.udf.run	<i>Run a User data functions (UDF).</i>
-----------------------	---

Description

Run a User data functions (UDF).

Usage

```
notebookutils.udf.run(  
  artifactId,  
  functionName,  
  parameters = list(),  
  workspaceId = "",  
  capacityId = ""  
)
```

Arguments

artifactId	The UDF artifact id.
functionName	The UDF function name.
parameters	The UDF parameters.
workspaceId	The UDF workspace id, if not provided, it will be retrieved from the current workspace.
capacityId	The UDF capacity id, if not provided, it will be retrieved from the current capacity.

Value

The UDF execution result.

notebookutils.variableLibrary.get	<i>Get a variable value</i>
-----------------------------------	-----------------------------

Description

Dummy implementation for local development.

Usage

```
notebookutils.variableLibrary.get(variableReference)
```

Arguments

variableReference
Variable reference

Value

A variable value (NULL)

notebookutils.variableLibrary.getHelpString
<i>Get help string for a method.</i>

Description

Get help string for a method.

Usage

notebookutils.variableLibrary.getHelpString(funcName = "", namespace = "")

Arguments

funcName function name
namespace namespace

notebookutils.variableLibrary.getLibrary
<i>Get a variable library</i>

Description

Dummy implementation for local development.

Usage

notebookutils.variableLibrary.getLibrary(variableLibraryName)

Arguments

variableLibraryName
Variable library name

Value

A dummy variable library object (list)

Index

display, [4](#)
display.config, [5](#)
display.configCustomOptions, [6](#)
displayHTML, [6](#)
mssparkutils.credentials.getConnectionStringOrCreds, [7](#)
mssparkutils.credentials.getFullConnectionString, [7](#)
mssparkutils.credentials.getPropertiesAll, [8](#)
mssparkutils.credentials.getSecret, [8](#)
mssparkutils.credentials.getSecretWithLS, [9](#)
mssparkutils.credentials.getToken, [10](#)
mssparkutils.credentials.help, [10](#)
mssparkutils.credentials.isValidToken, [11](#)
mssparkutils.credentials.putSecret, [11](#)
mssparkutils.credentials.putSecretWithLS, [12](#)
mssparkutils.env.getClusterId, [13](#)
mssparkutils.env.getJobId, [13](#)
mssparkutils.env.getPoolName, [14](#)
mssparkutils.env.getUserId, [14](#)
mssparkutils.env.getUserName, [15](#)
mssparkutils.env.getWorkspaceName, [15](#)
mssparkutils.env.help, [16](#)
mssparkutils.fs.append, [16](#)
mssparkutils.fs.cp, [17](#)
mssparkutils.fs.exists, [17](#)
mssparkutils.fs.fastcp, [18](#)
mssparkutils.fs.getMountPath, [19](#)
mssparkutils.fs.head, [19](#)
mssparkutils.fs.help, [20](#)
mssparkutils.fs.ls, [21](#)
mssparkutils.fs.mkdir, [21](#)
mssparkutils.fs.mount, [22](#)
mssparkutils.fs.mounts, [22](#)
mssparkutils.fs.mountToDriverNode, [23](#)
mssparkutils.fs.mv, [23](#)
mssparkutils.fs.put, [24](#)
mssparkutils.fs.refreshMounts, [25](#)
mssparkutils.fs.rm, [25](#)
mssparkutils.fs.unmount, [26](#)
mssparkutils.fs.unmountFromDriverNode, [26](#)
mssparkutils.help, [27](#)
mssparkutils.lakehouse.create, [27](#)
mssparkutils.lakehouse.delete, [28](#)
mssparkutils.lakehouse.get, [28](#)
mssparkutils.lakehouse.help, [29](#)
mssparkutils.lakehouse.list, [29](#)
mssparkutils.lakehouse.update, [30](#)
mssparkutils.nbResPath, [30](#)
mssparkutils.notebook.exit, [31](#)
mssparkutils.notebook.help, [31](#)
mssparkutils.notebook.run, [32](#)
mssparkutils.notebook.runMultiple, [33](#)
mssparkutils.notebook.updateNBSEndpoint, [33](#)
mssparkutils.notebook.validateDAG, [34](#)
mssparkutils.runtime.context, [34](#)
mssparkutils.runtime.setHcReplId, [35](#)
mssparkutils.session.stop, [35](#)
notebookutils.conf.get, [36](#)
notebookutils.conf.set, [36](#)
notebookutils.connections.getCredential, [37](#)
notebookutils.connections.getHelpString, [37](#)
notebookutils.credentials.getConnectionStringOrCreds, [38](#)
notebookutils.credentials.getFullConnectionString, [38](#)
notebookutils.credentials.getPropertiesAll, [39](#)
notebookutils.credentials.getSecret, [39](#)

notebookutils.credentials.getSecretWithLS, 40
notebookutils.credentials.getToken, 41
notebookutils.credentials.help, 41
notebookutils.credentials.isValidToken, 42
notebookutils.credentials.putSecret, 42
notebookutils.credentials.putSecretWithLS, 43
notebookutils.env.getClusterId, 44
notebookutils.env.getJobId, 44
notebookutils.env.getPoolName, 45
notebookutils.env.getUserId, 45
notebookutils.env.getUserName, 46
notebookutils.env.getWorkspaceName, 46
notebookutils.env.help, 47
notebookutils.fabricClient.delete, 47
notebookutils.fabricClient.get, 48
notebookutils.fabricClient.help, 48
notebookutils.fabricClient.listCapacities, 49
notebookutils.fabricClient.patch, 49
notebookutils.fabricClient.post, 50
notebookutils.fabricClient.put, 50
notebookutils.fs.append, 51
notebookutils.fs.cp, 51
notebookutils.fs.exists, 52
notebookutils.fs.fastcp, 53
notebookutils.fs.getMountPath, 53
notebookutils.fs.head, 54
notebookutils.fs.help, 55
notebookutils.fs.ls, 55
notebookutils.fs.mkdir, 56
notebookutils.fs.mount, 57
notebookutils.fs.mounts, 57
notebookutils.fs.mountToDriverNode, 58
notebookutils.fs.mv, 58
notebookutils.fs.put, 59
notebookutils.fs.refreshMounts, 60
notebookutils.fs.rm, 60
notebookutils.fs.unmount, 61
notebookutils.fs.unmountFromDriverNode, 61
notebookutils.help, 62
notebookutils.initialize, 62
notebookutils.lakehouse.create, 63
notebookutils.lakehouse.delete, 63
notebookutils.lakehouse.get, 64
notebookutils.lakehouse.getDefinition, 64
notebookutils.lakehouse.getWithProperties, 65
notebookutils.lakehouse.help, 65
notebookutils.lakehouse.list, 66
notebookutils.lakehouse.listTables, 66
notebookutils.lakehouse.loadTable, 67
notebookutils.lakehouse.update, 67
notebookutils.lakehouse.updateDefinition, 68
notebookutils.nbResPath, 68
notebookutils.notebook.create, 69
notebookutils.notebook.delete, 69
notebookutils.notebook.exit, 70
notebookutils.notebook.get, 70
notebookutils.notebook.help, 71
notebookutils.notebook.list, 71
notebookutils.notebook.run, 72
notebookutils.notebook.update, 73
notebookutils.notebook.updateDefinition, 73
notebookutils.notebook.updateNBSEndpoint, 74
notebookutils.notebook.validateDAG, 74
notebookutils.runtime.context, 75
notebookutils.runtime.help, 76
notebookutils.runtime.setHcReplId, 76
notebookutils.session.stop, 77
notebookutils.udf.getFunctions, 77
notebookutils.udf.getHelpString, 78
notebookutils.udf.help, 78
notebookutils.udf.run, 79
notebookutils.variableLibrary.get, 79
notebookutils.variableLibrary.getHelpString, 80
notebookutils.variableLibrary.getLibrary, 80