DBUtils

Name: Aathirainathan P

Date: 04-12-2024

1.dbutils.help()

```
dbutils.help()
```

This module provides various utilities for users to interact with the rest of Databricks.

credentials: DatabricksCredentialUtils -> Utilities for interacting with credentials within notebooks
data: DataUtils -> Utilities for understanding and interacting with datasets (EXPERIMENTAL)
fs: DbfsUtils -> Manipulates the Databricks filesystem (DBFS) from the console
jobs: JobsUtils -> Utilities for leveraging jobs features
library: LibraryUtils -> Utilities for session isolated libraries
meta: MetaUtils -> Methods to hook into the compiler (EXPERIMENTAL)
notebook: NotebookUtils -> Utilities for the control flow of a notebook (EXPERIMENTAL)
preview: Preview -> Utilities under preview category
secrets: SecretUtils -> Provides utilities for leveraging secrets within notebooks
widgets: WidgetsUtils -> Methods to create and get bound value of input widgets inside notebooks

2.dbutils.fs.help()

▶ ⌄ ✓ Just now (<1s)   2   Python 🗑 ✧ ⌑ ⋮

```
dbutils.fs.help()
```

dbutils.fs provides utilities for working with FileSystems. Most methods in this package can take either a DBFS path (e.g., "/foo" or "dbfs:/foo"), or another FileSystem URI. For more info about a method, use **dbutils.fs.help("methodName")**. In notebooks, you can also use the %fs shorthand to access DBFS. The %fs shorthand maps straightforwardly onto dbutils calls. For example, "%fs head --maxBytes=10000 /file/path" translates into "dbutils.fs.head("/file/path", maxBytes = 10000)".

fsutils

cp(from: String, to: String, recurse: boolean = false): boolean -> Copies a file or directory, possibly across FileSystems
head(file: String, maxBytes: int = 65536): String -> Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8
ls(dir: String): Seq -> Lists the contents of a directory
mkdirs(dir: String): boolean -> Creates the given directory if it does not exist, also creating any necessary parent directories
mv(from: String, to: String, recurse: boolean = false): boolean -> Moves a file or directory, possibly across FileSystems
put(file: String, contents: String, overwrite: boolean = false): boolean -> Writes the given String out to a file, encoded in UTF-8
rm(dir: String, recurse: boolean = false): boolean -> Removes a file or directory

mount

mount(source: String, mountPoint: String, encryptionType: String = "", owner: String = null, extraConfigs: Map = Map.empty[String, String]): boolean -> Mounts the given source directory into DBFS at the given mount point
mounts: Seq -> Displays information about what is mounted within DBFS
refreshMounts: boolean -> Forces all machines in this cluster to refresh their mount cache, ensuring they receive the most recent information
unmount(mountPoint: String): boolean -> Deletes a DBFS mount point
updateMount(source: String, mountPoint: String, encryptionType: String = "", owner: String = null, extraConfigs: Map = Map.empty[String, String]): boolean -> Similar to mount(), but updates an existing mount point (if present) instead of creating a new one

3.dbutils.fs.help('cp')

▶ ✓ Just now (<1s)   3

```
dbutils.fs.help('cp')
```

```
/**
 * Copies a file or directory, possibly across FileSystems..
 *
 * Example: cp("/mnt/my-folder/a", "s3n://bucket/b")
 *
 * @param from FileSystem URI of the source file or directory
 * @param to FileSystem URI of the destination file or directory
 * @param recurse if true, all files and directories will be recursively copied
 * @return true if all files were successfully copied
 */
cp(from: java.lang.String, to: java.lang.String, recurse: boolean = false): boolean
```

4.dbutils.fs.help('ls')

```
dbutils.fs.help('ls')
```

```
/**
 * Lists the contents of a directory.
 *
 * Example: display(ls("/mnt/my-folder/"))
 *
 * The FileInfo object that is returned has the following helper methods:
 * val files = ls("/mnt/my-folder/")
 * files.map(_.name) // [myFile, myDir/]
 * files.map(_.length) // [1286, 0]
 * files.map(_.path) // [/mnt/my-folder/myFile, /mnt/my-folder/myDir/]
 * files.map(_.isDir) // [false, true]
 * files.map(_.isFile) // [true, false]
 * files.map(_.modificationTime) // [1615937446000, 1615937224000]
 *
 * @param dir FileSystem URI
 * @return Ordered sequence of FileInfos containing the name, size and modification time
 * of each file.
 */
ls(dir: java.lang.String): scala.collection.Seq
```

5.dbutils.fs.ls

```
dbutils.fs.ls
```

```
<bound method prettify_exception_message.<locals>.f_with_exception_handling of Package
'dbutils.fs'. For more information, type 'display(dbutils.fs)' in a cell.>
```

6. ls (Listing all the datasets):

```
%fs
ls /databricks-datasets
```

	path	name	
1	dbfs:/databricks-datasets/COVID/	COVID/	
2	dbfs:/databricks-datasets/README.md	README.md	
3	dbfs:/databricks-datasets/Rdatasets/	Rdatasets/	
4	dbfs:/databricks-datasets/SPARK_README.md	SPARK_README.md	
5	dbfs:/databricks-datasets/adult/	adult/	
6	dbfs:/databricks-datasets/airlines/	airlines/	
7	dbfs:/databricks-datasets/amazon/	amazon/	
8	dbfs:/databricks-datasets/asa/	asa/	
9	dbfs:/databricks-datasets/atlas_higgs/	atlas_higgs/	
10	dbfs:/databricks-datasets/bikeSharing/	bikeSharing/	
11	dbfs:/databricks-datasets/cctvVideos/	cctvVideos/	
12	dbfs:/databricks-datasets/credit-card-fraud/	credit-card-fraud/	
13	dbfs:/databricks-datasets/cs100/	cs100/	
14	dbfs:/databricks-datasets/cs110x/	cs110x/	

55 rows | 1.81 seconds runtime   Refreshed now

7. dbutils.fs.ls("Listing a particular dataset")

```python
dbutils.fs.ls('dbfs:/databricks-datasets/COVID/')
```

```
[FileInfo(path='dbfs:/databricks-datasets/COVID/.DS_Store', name='.DS_Store', size=61
48, modificationTime=1615901201000),
 FileInfo(path='dbfs:/databricks-datasets/COVID/CORD-19/', name='CORD-19/', size=0, m
odificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/CSSEGISandData/', name='CSSEGISandDat
a/', size=0, modificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/ESRI_hospital_beds/', name='ESRI_hosp
ital_beds/', size=0, modificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/IHME/', name='IHME/', size=0, modific
ationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/USAFacts/', name='USAFacts/', size=0,
modificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/coronavirusdataset/', name='coronavir
usdataset/', size=0, modificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/covid-19-data/', name='covid-19-dat
a/', size=0, modificationTime=1733287302249),
 FileInfo(path='dbfs:/databricks-datasets/COVID/download_daily_covid-19_datasets.sh',
name='download_daily_covid-19_datasets.sh', size=1083, modificationTime=160929693200
0),
 FileInfo(path='dbfs:/databricks-datasets/COVID/estimated_patient_impact_and_hospital
 capacity by state/', name='estimated patient impact and hospital capacity by stat
```

8. dbutils.fs.ls

```python
dbutils.fs.ls("databricks-datasets")
```

```
 FileInfo(path='dbfs:/databricks-datasets/iot-stream/', name='iot-stream/', size=0, m
odificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/learning-spark/', name='learning-spark/', s
ize=0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/learning-spark-v2/', name='learning-spark-v
2/', size=0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/lending-club-loan-stats/', name='lending-cl
ub-loan-stats/', size=0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/med-images/', name='med-images/', size=0, m
odificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/media/', name='media/', size=0, modificatio
nTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/mnist-digits/', name='mnist-digits/', size=
0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/news20.binary/', name='news20.binary/', siz
e=0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/nyctaxi/', name='nyctaxi/', size=0, modific
ationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/nyctaxi-with-zipcodes/', name='nyctaxi-with
-zipcodes/', size=0, modificationTime=1733287382389),
 FileInfo(path='dbfs:/databricks-datasets/online_retail/', name='online_retail/', siz
```

9.Listing all paths

```python
for foldername in dbutils.fs.ls("databricks-datasets"):
    print(foldername.path)
```

```
dbfs:/databricks-datasets/COVID/
dbfs:/databricks-datasets/README.md
dbfs:/databricks-datasets/Rdatasets/
dbfs:/databricks-datasets/SPARK_README.md
dbfs:/databricks-datasets/adult/
dbfs:/databricks-datasets/airlines/
dbfs:/databricks-datasets/amazon/
dbfs:/databricks-datasets/asa/
dbfs:/databricks-datasets/atlas_higgs/
dbfs:/databricks-datasets/bikeSharing/
dbfs:/databricks-datasets/cctvVideos/
dbfs:/databricks-datasets/credit-card-fraud/
dbfs:/databricks-datasets/cs100/
dbfs:/databricks-datasets/cs110x/
dbfs:/databricks-datasets/cs190/
dbfs:/databricks-datasets/data.gov/
dbfs:/databricks-datasets/definitive-guide/
dbfs:/databricks-datasets/delta-sharing/
dbfs:/databricks-datasets/flights/
dbfs:/databricks-datasets/flower_photos/
dbfs:/databricks-datasets/flowers/
```
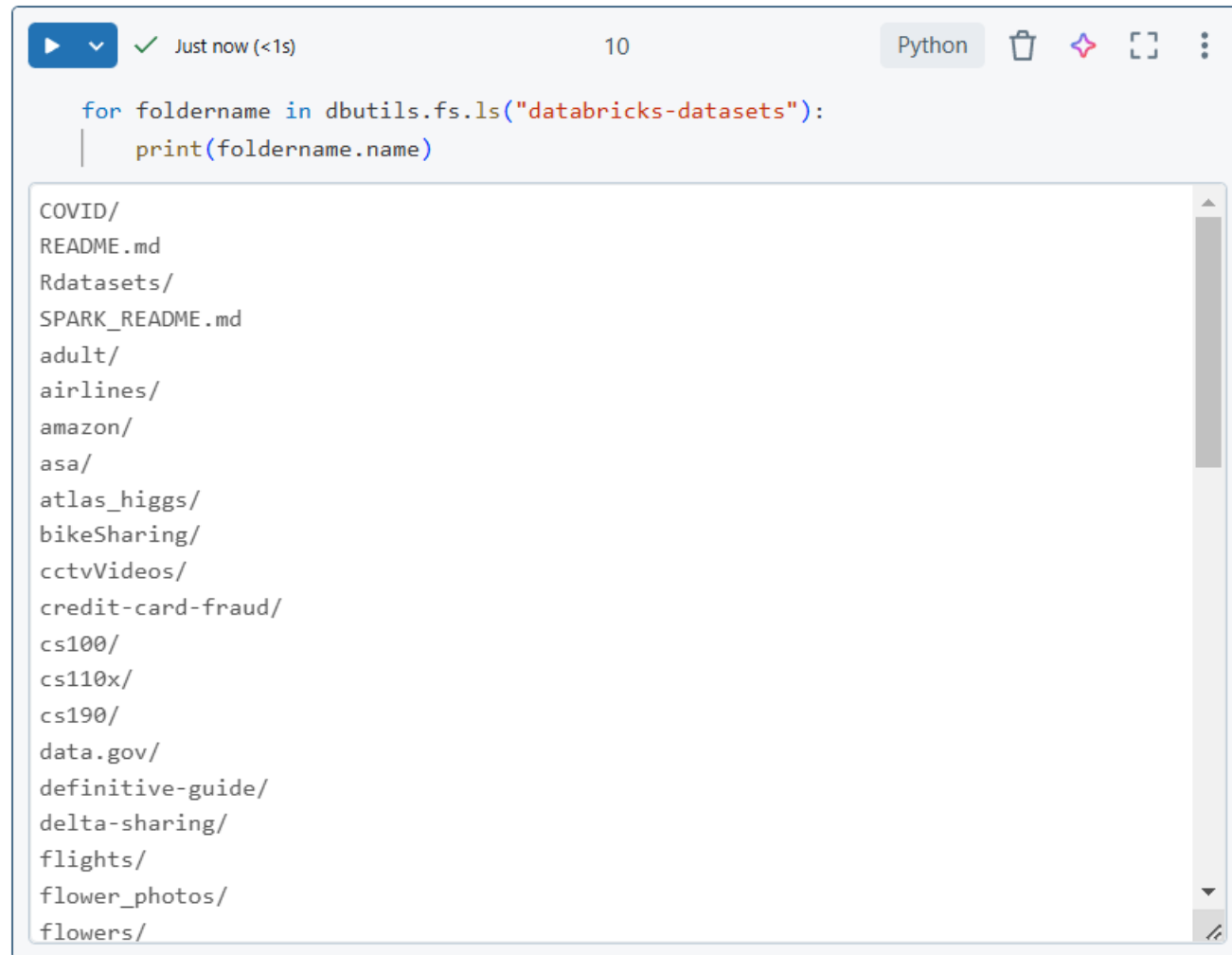
10.Listing all names

```python
for foldername in dbutils.fs.ls("databricks-datasets"):
    print(foldername.name)
```

```
COVID/
README.md
Rdatasets/
SPARK_README.md
adult/
airlines/
amazon/
asa/
atlas_higgs/
bikeSharing/
cctvVideos/
credit-card-fraud/
cs100/
cs110x/
cs190/
data.gov/
definitive-guide/
delta-sharing/
flights/
flower_photos/
flowers/
```

10.Listing all the modification times:

```python
for foldername in dbutils.fs.ls("databricks-datasets"):
    print(foldername.modificationTime)
```

```
1733287622401
1532502320000
1733287622401
1516124912000
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
1733287622401
```

12. dbutils.notebook.help()

```python
dbutils.notebook.help()
```

The notebook module.

exit(value: String): void -> This method lets you exit a notebook with a value
run(path: String, timeoutSeconds: int, arguments: Map): String -> This method runs a notebook and
returns its exit value

13. dbutils.notebook.exit()

```
dbutils.notebook.exit("Existing")
```

```
Notebook exited: Existing
```

14. dbutils.notebook.run()

```
dbutils.notebook.run("./mynb",10)
```

Notebook Workflows

Start time	End time	Notebo...	Duration	Status	Error code	Run par...
Dec 04, 2024, 1(	Dec 04, 2024, 1(	...m/mynb	6s	⊘		

15. ls "dbfs:/"

```
%fs
ls "dbfs:/"
```

Table ∨ +				🔍 ▽ ◫

	ᴬᴮ_C path	ᴬᴮ_C name	1²₃ size	1²₃ modificationTime
1	dbfs:/Volume/	Volume/	0	0
2	dbfs:/Volumes/	Volumes/	0	0
3	dbfs:/databricks-dataset...	databricks-dataset...	0	0
4	dbfs:/databricks-results/	databricks-results/	0	0
5	dbfs:/volume/	volume/	0	0
6	dbfs:/volumes/	volumes/	0	0

⤓ 6 rows | 19.91 seconds runtime   Refreshed 21 minutes ago

Submitted by:
Aathirainathan P