# S3 CMD Backup

Alexis Valencia-Suarez

Table of Contents

S3 CMD Backup 1

Description 1

Requirements 1

Usage Instructions 1

Opening Instructions 1

Implementation 2

Flow 2

Functions 2

## Description

Python script that can backup a given directory and push it to AWS S3.

### Requirements

* Download & Install AWS CLI
* Configure AWS CLI with your AWS Credentials
* Make sure you have python 3 install
* Make sure you have boto3 module (pip install boto3)
* Download or Clone S3CMDBackup (<https://github.com/alexisv6/S3CMDBackup.git>)

### Usage Instructions

* Find the S3CMDBackup.py in the command line
  + S3CMDBackup/ S3CMDBackup.py
* Invoke the script; Ex:
  + **python S3CMDBackup.py C:\Users\av\Desktop\ bucket1**
* On Mac I had to call:
  + **python3 S3CMDBackup.py /Users/av/Desktop bucket1**
* For help use:
  + **python S3CMDBackup.py -h**
  + **python S3CMDBackup.py --help**

### Opening Instructions

* With Visual Studio you can open and run the project by selecting the **S3CMDBackup.sln** file inside of the S3CMDBackup folder
* Make sure to have a python environment setup in Visual Studio
* To view code open the file located at S3CMDBackup/S3CMDBackup.py

## Implementation

### Flow

#### Validate user-entered arguments and save them to variables

#### Print Introduction with entered arguments shown on screen

#### Make sure user entered valid directory

#### Check for existence of bucket in S3

If bucket doesn’t exist then prompt user whether they want to create a new bucket. Create the new bucket with new name provided by user

#### Begin Traversing Directory

Derive the keynames for each file found in dir

Check that keyname to see if already contained in s3

If already contained will only reupload if size is different (updated)

If not will upload and create with the key generated

#### Print files and status as it progresses

### Functions

#### prompt\_create\_bucket()

* Asks the user if they would like to create a bucket. Verifies that response is appropriate and then returns whether the user wants to create a bucket.

#### print\_separator()

* Prints a separator to the console screen for visual aid

#### Print\_intro(path, bucket)

* Prints the intro to the script and presents the user with the inputs they provided when it was started. Displays path and bucket.

#### Create\_bucket(s3, bucketname)

* Will take bucketname and try to create a new bucket on the s3 object.
* If there is an error, such as the bucketname already existing, the user will be prompted for another name
* Will return when new bucket has been created

#### Does\_file\_exist(s3, bucketname, keyname)

* Checks the s3 instance passed in to see if the keyname passed in is present within the bucketname passed in.
* Will be used to save on uploads by not re-uploading files

#### Get\_base\_foldername(path)

* Returns the folder name of the folder that is being backed up. This foldername is extracted from the path that is passed in.

#### Get\_s3\_keyname(path, backupDir, filename )

* Prints out the backup directory chosen
* Transforms the directory to make folder the base within S3. This is going to be used to as the keyname for the new file.

#### Is\_size\_equal(s3, bucketname, keyname, path)

* Checks the s3 instance passed in and compares the size of the file associated to the keyname in the buckename passed in to the file on the local disk.
* Returns if the files are of equal size or not
* Will be used to not re\_upload files that have not changed since last backup
* If size is different it will be uploaded to update s3

#### is\_dir(path)

* Checks a path to make sure it’s a directory then returns true if yes or false if not a dir

#### Get\_valid\_dir()

* Used to get a valid directory path from the user
* Insures we backup something that exists