FileSync.py

Folder Synchronization

José Luca Baptista Pereira

lucajk15@gmail.com

Introduction

In this simple python program, I implemented a folder synchronization system. It should maintain the contents of a replica folder equal to the source folder. It will compare the contents of the source folder with contents of the replica folder, and it will delete and copy all the files necessary to maintain the synchronization. This process will be done in a time interval, set by the user. You can find the source code of this small project at: https://github.com/LucaKnowsStuff/FileSync

FileSynv.py

The functions

calculate_files_md5 ()

<u>Arguments</u>:

chunk_size (size of the chunks in bits) (int),

file_path (Absolute path to the files we want to hash) (String)

<u>Returns</u>: Hash of the file (String)

Simple function that calculates the md5 hash of a file. The algorithm will take the files in chunks of size determined by the user, and it will include the name of the file in the hash to be able to differentiate files with different names but equal contents.

compare_dir ()

<u>Arguments</u>:

source (absolute path to the source folder) (String),

replica (absolute path to the replica folder) (String)

<u>Returns</u>:

files_to_copy (List of files that need to be copied from source to replica) (List),

files_to_delete (List of files that need to be deleted in the replica folder) (List),

dirs_to_copy (List of folders that need to be deleted the replica folder) (List),

protected_dirs (List of all the folders in the source folder) (List),

The most important and complex function. It iterates over the source folder and stores all the file hash and their paths in a dictionary. It will then go thought the replica folder and check what files are in the replica that aren't in the source and need to be deleted, and what files are in the source but aren't in the replica and need to be copied. It also checks if any folder in the replica needs to be deleted as well. It will return all these lists of files and a list of all the folders in the source folder that need to exist in the replica folder.

delete_files_and_dirs ()

<u>Arguments</u>:

files_to_delete (List of all the absolute paths of files that need to be deleted) (List),

dirs_to_delete (List of all the absolute paths of folders that need to be deleted)) (List)

This function will delete all the files and folders on both lists if they exist.

copy_files ()

Arguments:

files_to_copy (List of all the absolute paths of files that need to be copied from source to replica) (List),

protected_dirs (List of all the absolute paths of folders that need to be copied from source to replica)) (List)

source (absolute path to the source folder) (String),

replica (absolute path to the replica folder) (String)

This function will first create all the folders in the protected_dirs list that don't already exist in the replica folder. Then it will copy all the files from the copy list to the replica folder if they exist.

check_dir_exists ()

Arguments:

dir (Path to a file or folder) (String)

Very simple function that checks if a file or folder exist, if they don't it will throw send an error for the command line parser.

Returns:

dir (Path to a file or folder) (String)

def handle_arguments ()

This function will handle the command line arguments given by the user , and make sure they are valid.

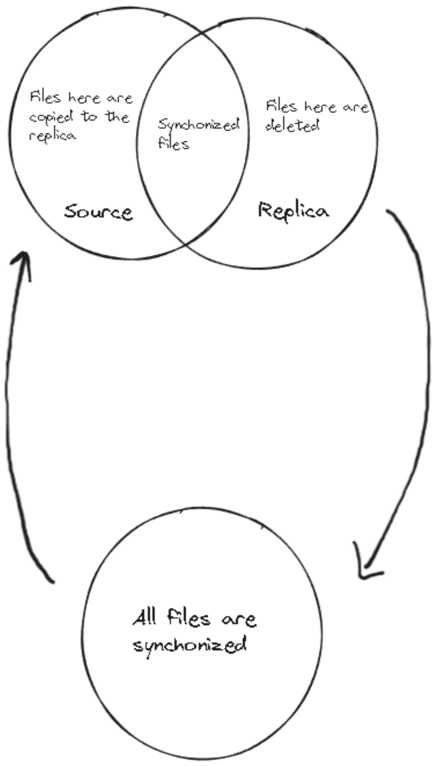
Returns:

args (Object containing all the arguments given by the user) (Obj)

How it works

After getting and verifying all the needed command line arguments the program will compare the files between the 2 folders. By default, we assume that all the files in the source folder to be copied to the replica. Then after the comparison all the files that are in the replica but not the source, or that have different contents are listed for deletion. If both the files are equal, they are removed from the copy list. We also keep a list of all the folders that exist in the replica but not in the source, they are listed for deletion as well.

After all the files and folders in the delete list are deleted and all the files are copied from the copy list both folders will be synchronized and have the same contents. This process is repeated in a time interval set by the user (default 15s)


Files here are copied to the replica
Synchonized files
Files here are deleted
Source
Replica
All files are synchonized

Run Instructions

The instructions to run the program are also listed in the README file.

You need Python3 installed to run this program.

Run **`python3 FileSync.py -h or python3 FileSync.py --help`** to receive more information in the command line.

Run **`python3 FileSync.py <Source Folder Absolute Path> <Replica Folder Absolute Path> <Log File Absolute Path> -t/--time <Time between cycles in seconds>`**

The time between cycles is an optional argument, if not stated it will between 15 seconds by default. All the files and folders passed in the command line need to exist.