|  |
| --- |
| #!/bin/bash |
|  |  |
|  | # Test for non-existence of bin and create if it doesn't not exist. |
|  | createBin() { |
|  | if [ ! -d $recyclepath ] ; then |
|  | mkdir $recyclepath |
|  | touch $HOME/.restore.info |
|  | fi |
|  |  |
|  | # If directory exist and .restore.info does not not exist, create .restore.info |
|  | if [ -d $recyclepath ] && [ ! -e $HOME/.restore.info ] ; then |
|  | touch $HOME/.restore.info |
|  | fi |
|  | } |
|  |  |
|  | # If $RMCFG exist, then use the environment variable. |
|  | # If $RMCFG does not exist, $HOME/.rm.cfg will be used. |
|  | # If neither exist, the recycle bin is in the default location: $HOME/deleted. |
|  | if [ $RMCFG ] ; then |
|  | recyclepath=$RMCFG |
|  | createBin |
|  | elif [ -e $HOME/.rm.cfg ] ; then |
|  | eval recyclepath=$(head -1 $HOME/.rm.cfg) |
|  | createBin |
|  | else |
|  | recyclepath=$HOME/deleted |
|  | createBin |
|  | fi |
|  |  |
|  | # If you use "safe\_rm" without a name of a file, rm produces an error message. |
|  | if [ $# -eq 0 ] ; then |
|  | echo "Error: Please provide the name of the file! ex: safe\_rm filename" |
|  | exit |
|  | fi |
|  |  |
|  | removeDir() { |
|  | recursivepath=$(echo $dirList | cut -d " " -f1) |
|  | if [ $noOption = true ] ; then |
|  | rm -r $recursivepath |
|  | elif [ $interactive = true ] && [ $verbose = true ] ; then |
|  | rm -ivr $recursivepath |
|  | elif [ $interactive = true ] ; then |
|  | rm -ir $recursivepath |
|  | elif [ $verbose = true ] ; then |
|  | rm -vr $recursivepath |
|  | fi |
|  | } |
|  |  |
|  | # This function filters out folder(s) and call recursiveFile to process file(s) |
|  | # After that, it will remove all the directories by using rm. |
|  | recursiveFolder() { |
|  | for i in $\* |
|  | do |
|  | folder=$i |
|  | if [ -d $folder ] ; then |
|  | continue |
|  | else |
|  | recursiveFile $i |
|  | fi |
|  | done |
|  |  |
|  | dirList=$(find $ff -type d 2>/dev/null) |
|  |  |
|  | for i in $dirList |
|  | do |
|  | folder=$i |
|  | if [ -d $folder ] ; then |
|  | removeDir |
|  | else |
|  | continue |
|  | fi |
|  | done |
|  | } |
|  |  |
|  | # If it is a file, it will call boolFunction |
|  | # else, it will call recursiveFolder. |
|  | recursiveFile() { |
|  | for i in $\* |
|  | do |
|  | file=$i |
|  | if [ -d $file ] ; then |
|  | recursiveFolder $i |
|  | else |
|  | boolFunction |
|  | fi |
|  | done |
|  | } |
|  |  |
|  | # Call writeFile to write the file |
|  | boolFunction() { |
|  | if [ $noOption = true ] ; then |
|  | writeFile |
|  | elif [ $interactive = true ] && [ $verbose = true ] ; then |
|  | if [ -s $file ] ; then |
|  | read -p "safe\_rm: remove regular empty file '$file' ? [y/n] " response |
|  | else |
|  | read -p "safe\_rm: remove regular file '$file' ? [y/n] " response |
|  | fi |
|  | selectCase |
|  | echo "removed '$file' to the Recycle Bin" |
|  | elif [ $interactive = true ] ; then |
|  | if [ -s $file ] ; then |
|  | read -p "safe\_rm: remove regular empty file '$file' ? [y/n] " response |
|  | else |
|  | read -p "safe\_rm: remove regular file '$file' ? [y/n] " response |
|  | fi |
|  | selectCase |
|  | elif [ $verbose = true ] ; then |
|  | writeFile |
|  | echo "removed '$file' to the Recycle Bin" |
|  | fi |
|  | } |
|  |  |
|  | mainProcess() { |
|  | for i in $\* |
|  | do |
|  | file=$i |
|  | # If you use "rm file1" and file1 does not exist, rm produces an error message. |
|  | if [ ! -e $file ] ; then |
|  | echo "Error: The file $file does not exist!" |
|  | exit |
|  | fi |
|  |  |
|  | # If you use "rm dir1" and dir1 is a directory, rm produces an error message. |
|  | if [ -d $file ] ; then |
|  | echo "safe\_rm: cannot remove '$file' : Is a directory" |
|  | file="!Error" |
|  | fi |
|  |  |
|  | boolFunction |
|  | done |
|  | } |
|  |  |
|  | writeFile() { |
|  | # If no errors mv the file to the relevant folder and add to restore info. |
|  | if [ $file != "!Error" ] ; then |
|  | inode=$(ls -i $file | cut -d " " -f1) |
|  | filename=$(basename $file) |
|  | newfilename=$filename\\_$inode |
|  | fixedPath=$(readlink -fn $file) |
|  | echo $newfilename:$fixedPath >> $HOME/.restore.info |
|  | mv $file $recyclepath/$newfilename |
|  | fi |
|  | } |
|  |  |
|  | selectCase() { |
|  | convertToLower |
|  | case $response in |
|  | y) writeFile ;; |
|  | yes) writeFile ;; |
|  | n) continue ;; |
|  | no) continue ;; |
|  | \*) read -p "Error: Invalid response! [y/n] " response |
|  | selectCase ;; |
|  | esac |
|  | } |
|  | convertToLower() { |
|  | response=$(echo $response | tr [:upper:] [:lower:]) |
|  | } |
|  |  |
|  | # Main |
|  | noOption=true |
|  | interactive=false |
|  | verbose=false |
|  | isFolder=false |
|  |  |
|  | while getopts ivrR opt |
|  | do |
|  | case $opt in |
|  | i) interactive=true |
|  | noOption=false |
|  | ;; |
|  | v) verbose=true |
|  | noOption=false |
|  | ;; |
|  | r) isFolder=true |
|  | ;; |
|  | R) isFolder=true |
|  | ;; |
|  | esac |
|  | done |
|  | shift $(($OPTIND - 1)) |
|  |  |
|  | if [ $isFolder = true ] ; then |
|  | for i in $\* |
|  | do |
|  | ff=$i |
|  | if [ -d $ff ] || [ -e $ff ] ; then |
|  | dirList=$(find $ff) |
|  | recursiveFolder $dirList |
|  | else |
|  | echo "Error: The folder/file $ff does not exist!" |
|  | fi |
|  | done |
|  | else |
|  | # Call mainProcess to do the work, if those are just files: |
|  | mainProcess $\* |
|  | fi |
|  | # END OF MAIN |