|  |
| --- |
|  |

|  |
| --- |
| **Tracking External Drive** |
| Achieved through JAVA programming language |
|  |
|  |
|  |
| **Ayush Garg** |
| **Akshay Bindal** |
|  |

ACKNOWLEDGEMENT

I take the opportunity to express my deep gratitude and sincere thanks to my computer teacher Mr. for her stimulating and thoughtful guidance. I am also thankful to my laboratory assistant Mr. who helped me a lot while completing the project. It is thoroughly their suggestions and advice that helped me in pursuing my work and in accomplishment of this project.

INDEX

|  |  |  |
| --- | --- | --- |
| S No. | TOPIC |  |
| 1. | Acknowledgement |  |
| 2. | 0 Level DFD |  |
| 3. | Menu diagram |  |
| 4. | About the project |  |
| 5. | Coding |  |
| 6. | Output screens |  |
| 7. | Bibliography |  |

TO NAVIGATE IN PROJECT

1. The application is to be run in command prompt (in windows).
2. It will wait for the user to insert or eject an external disk.
3. If the user inserts an external disk a new window will appear with following menu:
4. Display Total Space
5. Display Free Space
6. Display Space occupied
7. Display contents of the drive
8. Display only folders and files
9. Return to program
10. Exit

Press 1 to see the total capacity of the disk in GB.

Press 2 to see the free space in the disk in GB.

Press 3 to see the total space occupied by the disk in GB.

Press 4 to see all the files and folders in the drive.

Press 5 to see files and folder content of the drive.

Press 6 to return to command prompt window.

Press 7 to exit the program.

1. If the user removes the disk, then a message is shown that the disk has been removed.
2. Multiple drives can be run simultaneously.

DATA FLOW DIAGRAMS AND MENU DIAGRAMS

0 LEVEL DFD

INSERTING EXTERNAL DRIVE MEMORY SPECIFICATIONS

CANDIDATE

REMOVING EXTERNAL DRIVE CONTENTS OF DRIVE

POLLING BOTH

MENU DIAGRAM

ALL FILES AND FOLDERS

THIS DISPLAYS ALL THE FILES AND FOLDERS PRESENT IN THE DRIVE

MENU

1 TOTAL SPACE

2 FREE SPACE

3. SPACE OCCUPIED

4. ALL THE FILES AND FOLDERS

5. ONLY THE VISIBLE CONTENTS OF DRIVE

6 . RETURN BACK

7. EXIT

TOTAL SPACE

THIS DISPLAYS THE TOTAL CAPACITY OF EXTERNAL DRIVE

VISIBLE CONTENTS

THIS DISPLAYS THE VISIBLE FILES AND FOLDERSOF THE DRIVE

FREE SPACE

THIS DISPLAYS THE FREE SPACE IN EXTERNAL DRIVE

RETURN BACK

THIS RETURNS THE USER BACK TO THE PROGRAM

SPACE OCCUPIED

THIS DISPLAYS THE SPACE OCCUPIED BY EXTERNAL DRIVE

EXIT

THIS IS TO EXIT FROM THE PROGRAM

ABOUT THE PROJECT

This project deals with the possible operations that can be done on any external drive that id plugged in the system. The major operations followed by this project are as follows:-

* Detecting if any external drive is inserted or removed
* Displaying various memory details of the external drive
* Displaying the contents of the external disk.

CODING OF THE PROJECT

import java.util.Scanner;

import java.io.\*;

import javax.swing.\*;

class q5

{

static //To print all the drives present on the system only once

{

long j=0;

File [] q=File.listRoots();

System.out.println("\n................................");

System.out.println("\n\tPRESENT DRIVES\n");

System.out.println("................................\n");

for(File r : q)

{

System.out.println(r);

j=r.getTotalSpace()/(1024\*1024\*1024);

if(j==0)

{

System.out.println("Total Space "+(float)r.getTotalSpace()/(1024\*1024)+"MB");

System.out.println("Free Space "+(float)r.getFreeSpace()/(1024\*1024)+"MB");

}

else

{

System.out.println("Total Space:"+(float)r.getTotalSpace()/(1024\*1024\*1024)+"GB");

System.out.println("Free Space:"+(float)r.getFreeSpace()/(1024\*1024\*1024)+"GB");

}

System.out.println("............................");

}

}

public static void main(String ... as) //To check if the drive has been Removed or Inserted

{

Scanner sc=new Scanner(System.in);

File [] q=File.listRoots();

int j=q.length;

while(true)

{

File [] root=File.listRoots();

int i=root.length;

if( i>j )

{

File w=findDrive(q);

Menu m=new Menu(w);

m.start(); //Starts the thread of Menu

main(); //To start the drive checking again

}

else if(i<j)

{

System.out.println("\n............................");

System.out.println("EXTERNAL DRIVE REMOVED!!!!!!!!\n");

System.out.println("............................\n");

main();

}

}

}

static File findDrive(File [] driv) //To find the Drive Inserted or Removed

{

int y;

int flag=1;

int pos=0;

File [] qw=File.listRoots();

for(y=0;y<driv.length;y++)

{

if(qw[y].equals(driv[y]))

{

flag=1;

}

else

{

flag=0;

pos=y;

break;

}

}

if(flag==1)

pos=driv.length;

return qw[pos];

}

}

class Menu extends Thread //Thread running prallel to main

{

File drive;

Menu(File rt)

{

drive=rt;

}

public void run()

{

int choice;

long q;

do

{

Scanner sc=new Scanner(System.in);

String ch=JOptionPane.showInputDialog(drive+" DRIVE INSERTED\n"+"MENU\n1.Display total space\n2.Display free space\n3.Space Occupied\n4.Display contents of Drive\n5.Display only Folders and Files\n6.Return to program\n7.Exit","Your choice");

char [] c=ch.toCharArray();

for(int j=0;j<c.length;j++)

{

if((c[j]<48)||(c[j]>55))

{

System.out.println("Wrong choice entered!!!!!");

break;

}

}

choice=(int)c[0]-48;

switch(choice)

{

case 1: q=drive.getTotalSpace()/(1024\*1024\*1024);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(drive);

if(q==0)

System.out.println("Total Space "+(float)drive.getTotalSpace()/(1024\*1024)+"MB");

else

System.out.println("Total Space:"+(float)drive.getTotalSpace()/(1024\*1024\*1024)+" GB");

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

case 2: q=drive.getFreeSpace()/(1024\*1024\*1024);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(drive);

if(q==0)

System.out.println("Free Space "+(float)drive.getFreeSpace()/(1024\*1024)+"MB");

else

System.out.println("\nFree Space:"+(float)drive.getFreeSpace()/(1024\*1024\*1024)+" GB");

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

case 3: q=drive.getFreeSpace()/(1024\*1024\*1024);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(drive);

if(q==0)

System.out.println("\nSpace Occupied:"+(float)(drive.getTotalSpace()-drive.getFreeSpace())/(1024\*1024)+" MB");

else

System.out.println("\nSpace Occupied:"+(float)(drive.getTotalSpace()-drive.getFreeSpace())/(1024\*1024\*1024)+" GB");

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

case 4: System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(drive);

System.out.println("\nFiles Present:\n");

if(drive.getTotalSpace()==drive.getFreeSpace())

{

System.out.println("Drive is Empty");

break;

}

showFiles(drive);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

case 5: System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(drive);

String [] p=drive.list();

System.out.println("\nFiles Present:\n");

for(String u : p)

System.out.println(" -> "+u);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

case 6:q5.main();

case 7:System.exit(0);

}

}while(choice!=6);

}

static void showFiles(File x) //Passing the Address only

{

File [] name; //Storing Address in a File Array

name=x.listFiles(); //listing files in name array

for(int z=0;z<name.length;z++)

{

if(name[z].isDirectory())

{

File a;

a=name[z].getAbsoluteFile();

showFiles(a);

}

else

System.out.println(name[z]);

}

}

}