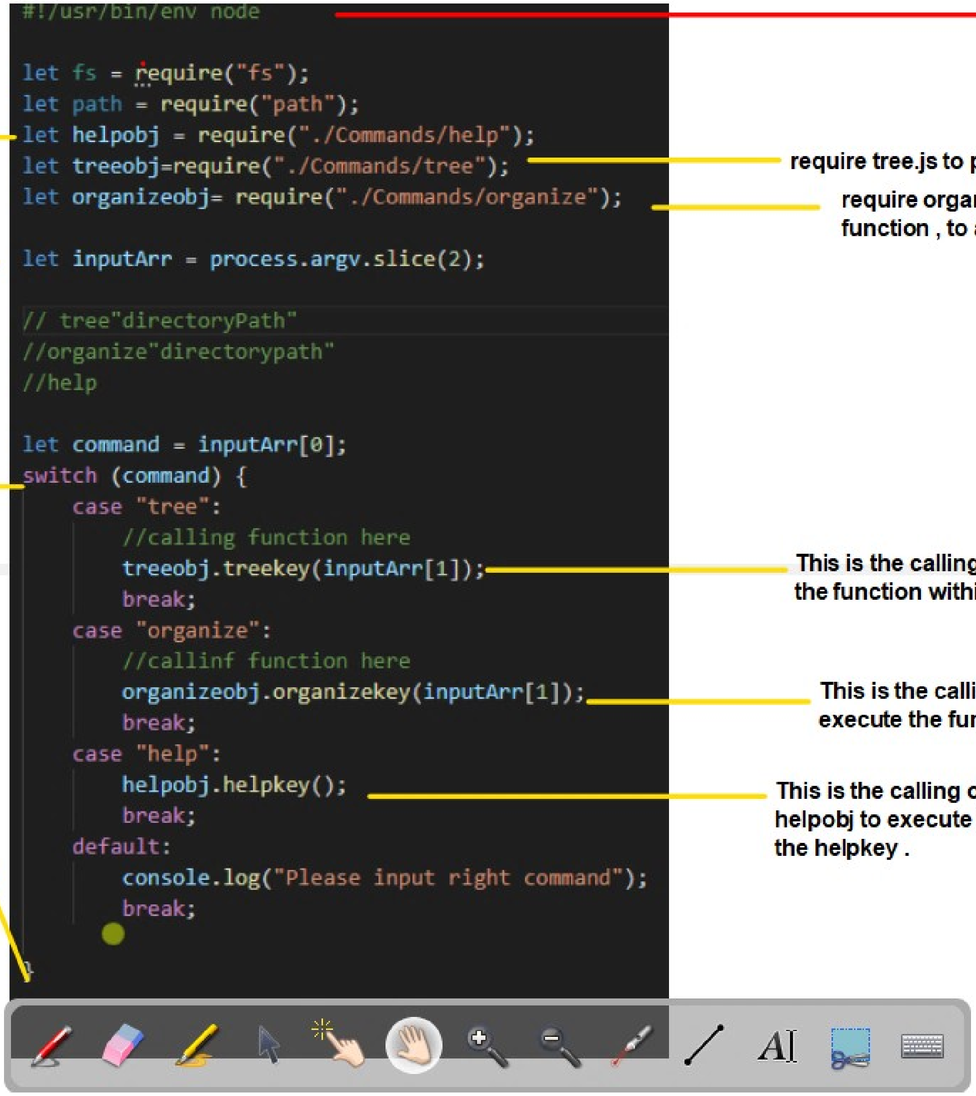


```
#!/usr/bin/env node

let fs = require("fs");
let path = require("path");
let helpobj = require("./Commands/help");
let treeobj=require("./Commands/tree");
let organizeobj= require("./Commands/organize");

let inputArr = process.argv.slice(2);

// tree"directoryPath"
//organize"directorypath"
//help

let command = inputArr[0];
switch (command) {
    case "tree":
        //calling function here
        treeobj.treekey(inputArr[1]);
        break;
    case "organize":
        //callinf function here
        organizeobj.organizekey(inputArr[1]);
        break;
    case "help":
        helpobj.helpkey();
        break;
    default:
        console.log("Please input right command");
        break;
}
```

shabang Syntax for nodejs

require help.js to provide the function available for help

require tree.js to provide tree function

require organize.js to provide organize function , to arrange files and folder .

input array using .slice()method.

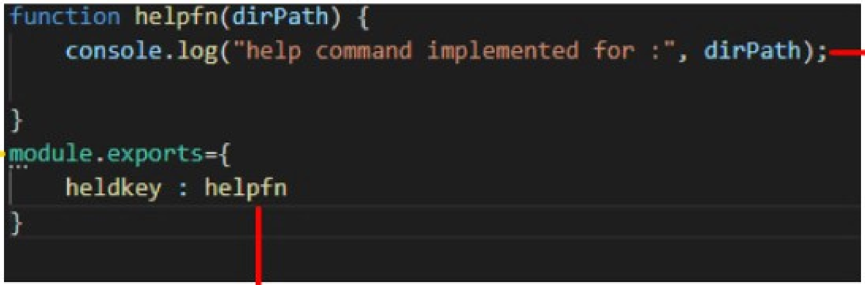
Switch cases for different cases.

This is the calling of treekey from treeobj to execute the function within the treekey .

This is the calling of organizekey from oragnizeobj to execute the function within the oragnizekey .

This is the calling of helpkey from helpobj to execute the function within the helpkey .

Help function



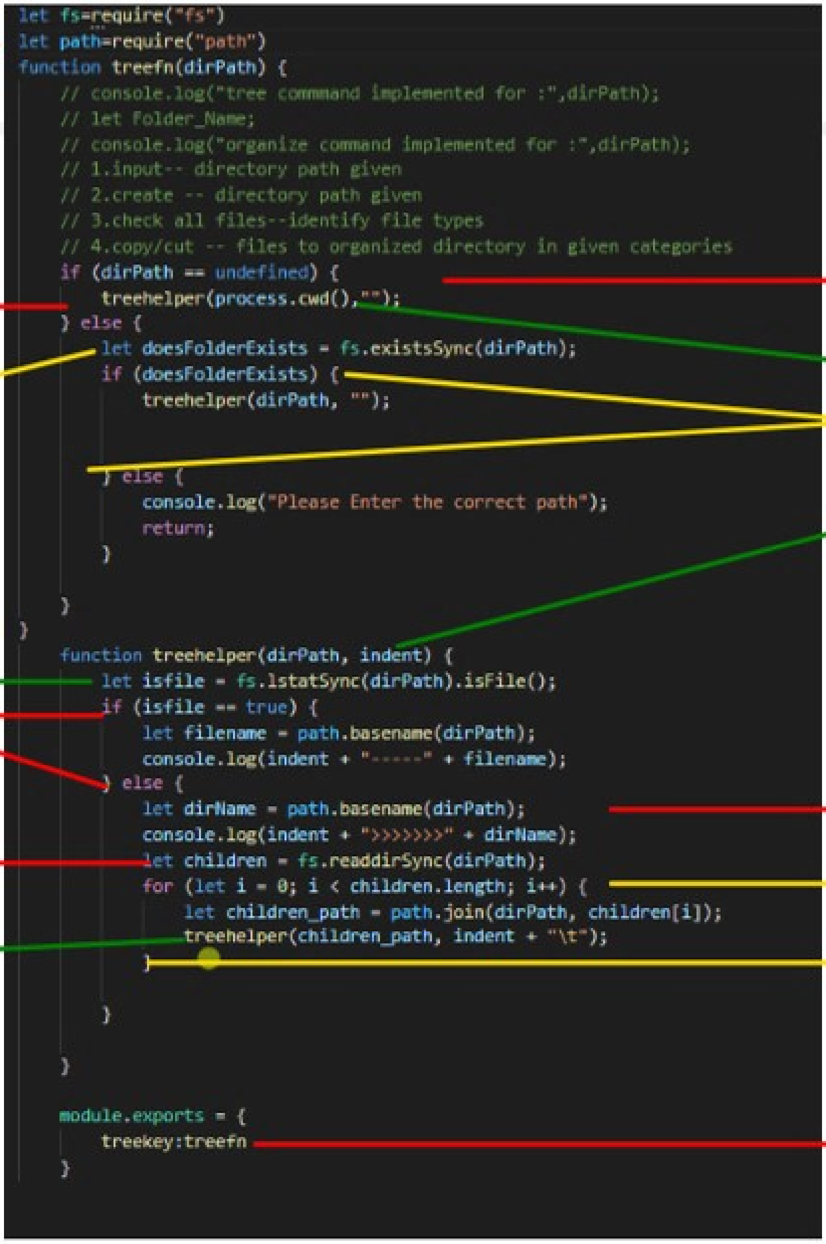
```
function helpfn(dirPath) {
    console.log("help command implemented for :", dirPath);

}
module.exports={
    heldkey : helpfn
}
```

dirPath = it is directory path variable.

module.exports ,it is a used to export the this module to main program file ,where we want to call this function,

intializing a helpkey to invoke the help function when called throughkey.

Tree Function



```
let fs=require("fs")
let path=require("path")
function treefn(dirPath) {
    // console.log("tree command implemented for :",dirPath);
    // let Folder_Name;
    // console.log("organize command implemented for :",dirPath);
    // 1.input-- directory path given
    // 2.create -- directory path given
    // 3.check all files--identify file types
    // 4.copy/cut -- files to organized directory in given categories
    if (dirPath == undefined) {
        treehelper(process.cwd(),"");
    } else {
        let doesFolderExists = fs.existsSync(dirPath);
        if (doesFolderExists) {
            treehelper(dirPath, "");


        } else {
            console.log("Please Enter the correct path");
            return;
        }

    }
}
    function treehelper(dirPath, indent) {
        let isfile = fs.lstatSync(dirPath).isFile();
        if (isfile == true) {
            let filename = path.basename(dirPath);
            console.log(indent + "-----" + filename);
        } else {
            let dirName = path.basename(dirPath);
            console.log(indent + ">>>>>>>" + dirName);
            let children = fs.readdirSync(dirPath);
            for (let i = 0; i < children.length; i++) {
                let children_path = path.join(dirPath, children[i]);
                treehelper(children_path, indent + "\t");
            }

        }

    }

    module.exports = {
        treekey:treefn
    }
```

require path: it is an inbuilt module ,used for path manipulation of given file or folder

undefined: it is shown , when we do not give filepath or give incorrect filepath.

calling the function in current working directory.

this is given for identation/space.

Checking if the given directory path exist or not.

if directory path exists , call helper()function.

Checking if at the given dirPath is a file and if it is file ,give the name of file at the base of path.
or at the last of path.

If there is folder at the base of the path , give the folder name.

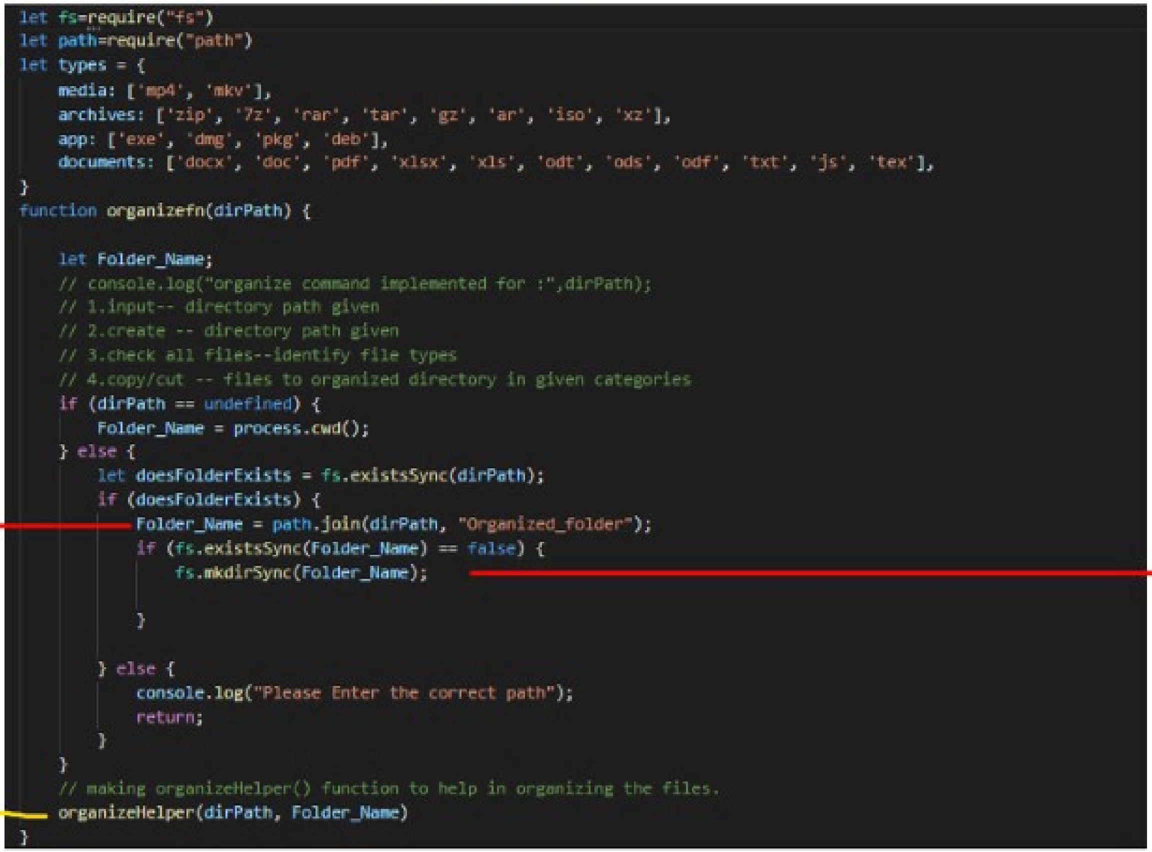
Reading inside the folder to check for further child files or folder.

Array to check within the folder for its child files and folder and getting there individual path.

calling of treehelper() function within the array to again check for files and folder and do the same thing again.
That is, if it is file ,give the filename and if, it is folder ,give its name and check within the folder for its child files and folders

key and value pair within the module .
key is called from the main program and it invokes the function(value) within it.

Organize Function



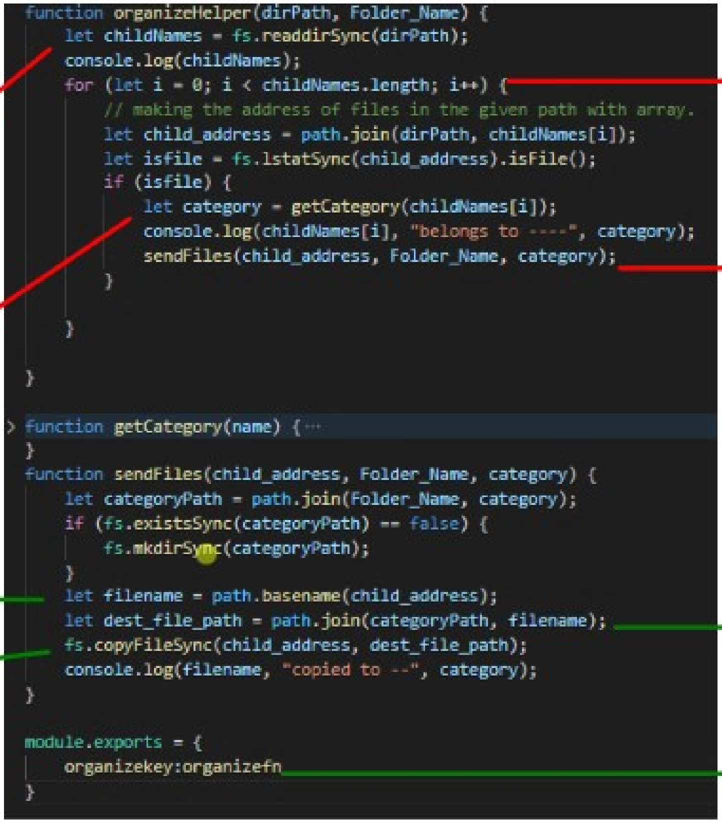
```
let fs=require("fs")
let path=require("path")
let types = {
    media: ['mp4', 'mkv'],
    archives: ['zip', '7z', 'rar', 'tar', 'gz', 'ar', 'iso', 'xz'],
    app: ['exe', 'dmg', 'pkg', 'deb'],
    documents: ['docx', 'doc', 'pdf', 'xlsx', 'xls', 'odt', 'ods', 'odf', 'txt', 'js', 'tex'],
}
function organizefn(dirPath) {

    let Folder_Name;
    // console.log("organize command implemented for :",dirPath);
    // 1.input-- directory path given
    // 2.create -- directory path given
    // 3.check all files--identify file types
    // 4.copy/cut -- files to organized directory in given categories
    if (dirPath == undefined) {
        Folder_Name = process.cwd();
    } else {
        let doesFolderExists = fs.existsSync(dirPath);
        if (doesFolderExists) {
            Folder_Name = path.join(dirPath, "Organized_folder");
            if (fs.existsSync(Folder_Name) == false) {
                fs.mkdirSync(Folder_Name);

            }

        } else {
            console.log("Please Enter the correct path");
            return;
        }
    }
    // making organizeHelper() function to help in organizing the files.
    organizeHelper(dirPath, Folder_Name)
}
```

it is a types object ,with different attributes.
It will be used in the program to categorize the types of files present according to attributes given in the types object.

folder_name variable with the path and name of destination folder, known as Organized_folder.

making the folder/ directory with this method.

calling of organizeHelper()function, with arguments passes are source address and destionation folder.



```
function organizeHelper(dirPath, Folder_Name) {
    let childNames = fs.readdirSync(dirPath);
    console.log(childNames);
    for (let i = 0; i < childNames.length; i++) {
        // making the address of files in the given path with array.
        let child_address = path.join(dirPath, childNames[i]);
        let isfile = fs.lstatSync(child_address).isFile();
        if (isfile) {
            let category = getCategory(childNames[i]);
            console.log(childNames[i], "belongs to ----", category);
            sendFiles(child_address, Folder_Name, category);
        }

    }

}

> function getCategory(name) {···
}
function sendFiles(child_address, Folder_Name, category) {
    let categoryPath = path.join(Folder_Name, category);
    if (fs.existsSync(categoryPath) == false) {
        fs.mkdirSync(categoryPath);
    }
    let filename = path.basename(child_address);
    let dest_file_path = path.join(categoryPath, filename);
    fs.copyFileSync(child_address, dest_file_path);
    console.log(filename, "copied to --", category);
}

module.exports = {
    organizekey:organizefn
}
```

loop to read the content within the childNames , with content of the directory at the base of the dirPath.

reading the content of the given directory at the base of the given path.

calling sendFiles()function ,with arguments as the address of childs of the source directory ,destination folder and category .

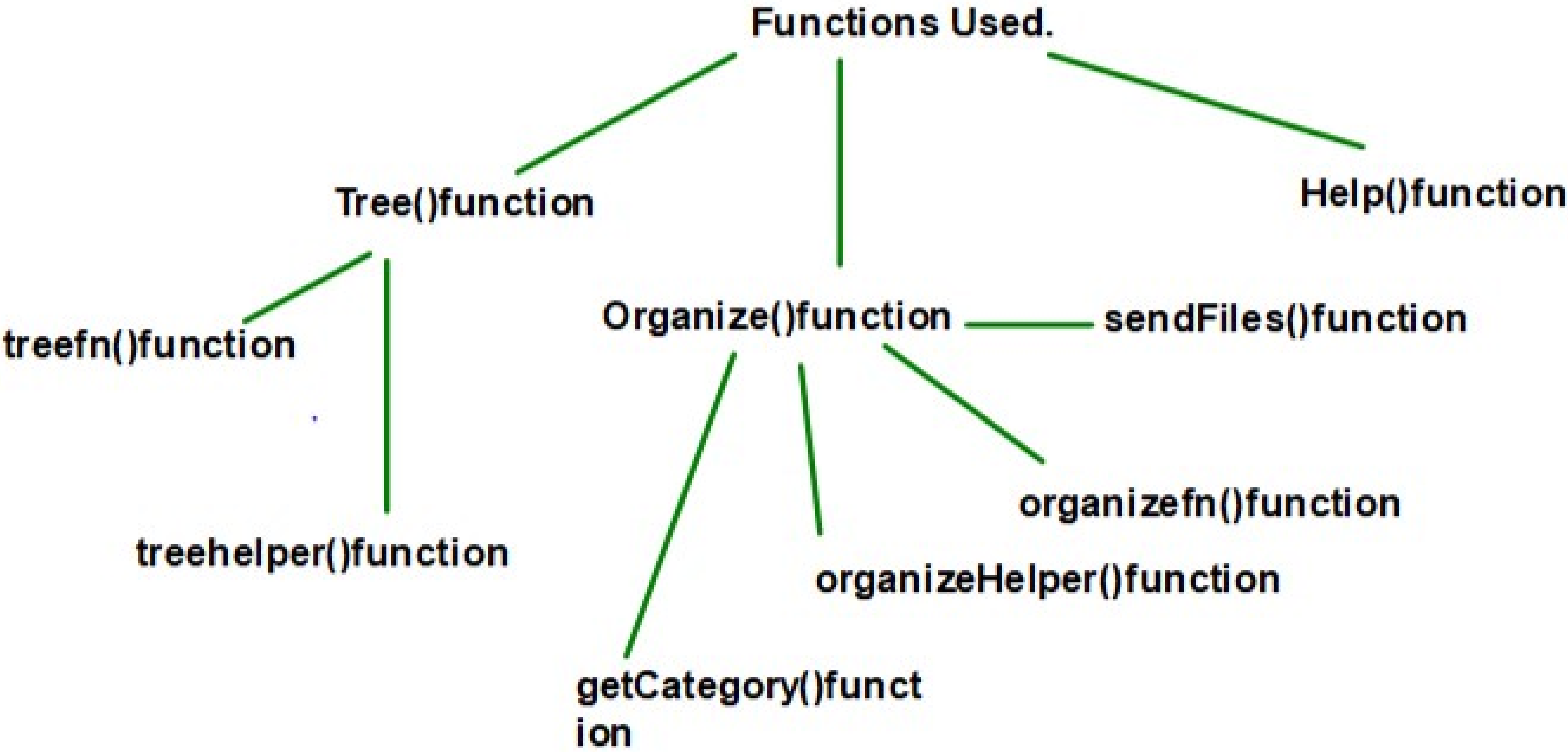
Getting the value of individual files or folder path with the array.
and the calling of getCategory function.

getting to know the filename by using the basename() method.

getting to know the destination organized folder address with the folder categories.

we are copying the content of the folder at the given child address and pasting it in the destination organized folder.

initializing the key and value pair for the given module export.
when we call the given organizeKey ,the function related with the key gets executed.


Functions Used.
Tree()function
Help()function
treefn()function
treehelper()function
Organize()function
sendFiles()function
organizefn()function
organizeHelper()function
getCategory()funct
ion