**חלק :1**

1. איך יוצרים תיקייה בגאווה?

ב-Java, כדי ליצור תיקייה ב-Java תחילה אנו משתמשים ב-class file.(import java.io.File)

שנית, נקרא לתיקייה שברצוננו ליצור ונציין את הנתיב שבו נרצה ליצור את התיקייה עם שם המשתנה cm מסוג File.

שלישית, אנו משתמשים ב mkdir() כדי ליצור תיקייה חדשה ומשתמשים בתנאי if כדי לוודא שהתיקייה נוצרה.

............

<https://how.dev/answers/how-to-create-a-directory-using-java>

............

import java.io.File;

public class CMD\_ex {

public static void main(String[] args){

File cm = new File ("C:\\Users\\duaaa\\Downloads\\myNewFolder");

// cm.mkdir() العبارة التي تنشأ الملف

if (cm.mkdir() == true) {

System.out.println("Directory has been created successfully");

}

else {

System.out.println("Directory cannot be created");

}

}

}

1. איך יוצרים קובץ?

ב־Java, ניתן ליצור קובץ חדש באמצעות המחלקה File והשיטה createNewFile(). פונקציה זו יוצרת קובץ חדש וריק. הפונקציה מחזירה true אם נתיב הקובץ המופשט אינו קיים ונוצר קובץ חדש. היא מחזירה false אם שם הקובץ כבר קיים.

.........

<https://www.geeksforgeeks.org/file-createnewfile-method-in-java-with-examples/>

………..

import java.io.File;

import java.io.IOException;

public class CMD\_ex {

public static void main(String[] args){

try {

// Get the file

File f = new File("C:\\Users\\duaaa\\Downloads\\Text");

// Create new file

// if it does not exist

if (f.createNewFile())

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

else

System.out.println("File already exists");

}

catch (Exception e) {

System.err.println(e);

}

}

}

1. איך בודקים אם קובץ/תיקייה קיימים ?

ב־Java, ניתן לבדוק אם קובץ או תיקייה קיימים באמצעות המחלקה File והשיטה exists().פונקציה זו קובעת האם קובץ או ספרייה המסומנים בשם הקובץ המופשט קיימים או לא. הפונקציה מחזירה אמת אם נתיב הקובץ המופשט קיים, אחרת מחזירה שקר.

...........

<https://www.geeksforgeeks.org/file-exists-method-in-java-with-examples/>

.........

import java.io.File;

public class CMD\_ex {

public static void main(String[] args){

// Get the file

File f = new File("C:\\Users\\duaaa\\Downloads\\Text");

if (f.exists())

// Show if the file exists

System.out.println("Exists");

else

// Show if the file does not exists

System.out.println("Does not Exists");

}

}

.4 איך מציגים את תוכן תיקייה?

רשימת כל הקבצים תיקייה ב-Java. במחלקה java.io.File המספקת מתודה מובנית listFiles() שתחזיר מערך של קבצים ותיקיות של הקבצים תיקייה.

........

<https://www.geeksforgeeks.org/how-to-list-all-files-in-a-directory-in-java/>

.......

import java.nio.file.Path;

import java.nio.file.Paths;

import java.io.File;

public class CMD\_ex {

public static void main(String[] args) {

// Using Path

Path path = Paths.get("myNewFolder");

System.out.println("Path: " + path.toString());

// Using File

File file = new File("myNewFolder");

System.out.println("File: " + file.getAbsolutePath());

}

}

........

1. .5 )רשות ) מה ההבדל בין File ל Path ?

1. Path הוא חלק מחבילת Java NIO (New Input/Output) שהוצגה ב-Java 7, המספקת פונקציונליות משופרת לטיפול בקבצים.

2. File הוא המחלקה הישנה יותר מ-Input/Output API המקורי ב-Java.

3. Path מספק שיטות טובות יותר לטיפול בנתיבי קבצים, כגון שילוב שני נתיבים או אחזור חלקים מנתיב.

4. שיטות File אינן תומכות בקישורים סמליים כמו שיטות Path.

5. ניתן להשתמש ב-Path עם מחלקות NIO אחרות כגון Files ו-FileSystems כדי לבצע פעולות קבצים מתקדמות.

.........

<https://www.javaguides.net/2023/11/difference-between-path-and-file-in-java-nio.html>

.........

import java.nio.file.Path;

import java.nio.file.Paths;

import java.io.File;

public class CMD\_ex {

public static void main(String[] args) {

// Using Path

Path path = Paths.get("myNewFolder");

System.out.println("Path: " + path.toString());

// Using File

File file = new File("myNewFolder");

System.out.println("File: " + file.getAbsolutePath());

}

}

**חלק 3: תיעוד וקובץ**

**README**

1. General description of the project

In this exercise, we will build a simple command line program in Java that simulates a terminal. The user will be able to type commands such as pwd, mkdir, ls, and cd and receive output accordingly, while working with the computer's file system

Instructions on how to compile and run the code 2.

We run the code by clicking the run button in Java, and when the code is run, we choose the operation we want

Example of running the commands3.

Pwd

C:\Users\duaaa\Downloads > pwd

C:\Users\duaaa\Downloads

Mkdir

C:\Users\duaaa\Downloads > mkdir testFolder

Directory created: testFolder

Ls

C:\Users\duaaa\Downloads > ls

[FILE] (DUAA,HIBA)דוח ההצעה (1).docx

[FILE] (DUAA,HIBA)דוח ההצעה.docx

[FILE] 2a4f9994-27db-4352-ac6d-52fce1767bba.jpg

[FILE] 33caa297-56aa-4874-8c70-13d292877b68.jpg

[FILE] 5.pdf

[FILE] 60c68a6b-cc6d-41d7-a9a2-628718b7d5cc.jpg

[FILE] Azrieli Advanced 1 2025B Syllabus - Anne.docx

Cd

C:\Users\duaaa > cd Downloads

C:\Users\duaaa\Downloads

Touch

C:\Users\duaaa\Downloads\testFolder > touch myFile

File created: myFile

Help

> help

you need to handle the following commands:

pwd- Print working directory

ls- List files in current directory

cd - Change directory

mkdir - Create new directory

touch - Create a new file

help - Show help information exit- Exit the program

exit

C:\Users\duaaa\Downloads > exit

Goodbye!

4

import java.io.File;  
import java.io.IOException;  
  
public class ShellCommandHandler {  
 private File currentDirectory;  
  
//==================== Initialize currentDirectory to the user's current working directory  
 public ShellCommandHandler() {  
 currentDirectory = new File(System.*getProperty*("user.dir"));  
 }  
//====================Return the current directory  
 public File getCurrentDirectory() {  
 return currentDirectory;  
 }  
//====================print(Print the absolute path of the current directory)  
 public void printWorkingDirectory() {  
 System.*out*.println(currentDirectory.getAbsolutePath());  
 }  
//====================listDirectory(List all files and directories in the current directory)  
 public void listDirectory() {  
 File[] files = currentDirectory.listFiles();  
  
 if (files != null && files.length > 0) {  
 for (File file : files) {  
 if (file.isDirectory()) {  
 //For directories, prefix with "[DIR] "  
 System.*out*.println("[DIR] " + file.getName());  
 } else if (file.isFile()) {  
 // For files, prefix with "[FILE]  
 System.*out*.println("[FILE] " + file.getName());  
 }  
 }  
 } else {  
 System.*out*.println("The folder is empty.");  
 }  
 }  
//=====================changeDirectory(Implement the change directory command)  
public void changeDirectory(String argument) {  
 if (argument == null) {  
 System.*out*.println("Usage: cd [directory\_name]");  
 return;  
 }  
  
 if (argument.equals("..")) {  
 File parent = currentDirectory.getParentFile();  
 //If name is null, show usage message: "Usage: cd [directory\_name]"  
 if (parent != null && parent.exists()) {  
 currentDirectory = parent;  
 } else {  
 System.*out*.println("No parent directory");  
 }  
 //If name is "..", move to parent directory if possible  
 } else {  
 File newfile = new File(currentDirectory, argument);  
 if (newfile.exists() && newfile.isDirectory()) {  
 currentDirectory = newfile;  
  
 // move to the specified directory  
 } else {  
 System.*out*.println("Directory not found: " + argument);  
 }  
 }  
}  
//=======================makeDirectory(Implement the make directory command)  
public void makeDirectory(String argument) {  
// If name is null, show usage message: "Usage: mkdir [directory\_name]"  
 if (argument == null) {  
 System.*out*.println("Usage: mkdir [directory\_name]");  
 return;  
 }  
  
 File new\_one = new File(currentDirectory, argument);  
// If directory already exists, print: "Directory already exists."  
 if (new\_one.exists()) {  
 System.*out*.println("Directory already exists.");  
// If directory creation is successful, print: "Directory created: [name]"  
 } else {  
 if (new\_one.mkdir()) {  
 System.*out*.println("Directory created: " + argument);  
// If directory creation fails, print: "Failed to create directory."  
 } else {  
 System.*out*.println("Failed to create directory.");  
 }  
 }  
}  
//=========================createFile( Implement the create file command)  
public void createFile(String name) {  
// If name is null, show usage message: "Usage: touch [file\_name]"  
 if (name == null) {  
 System.*out*.println("Usage: touch [file\_name]");  
 return;  
 }  
  
 File newFile = new File(currentDirectory, name);  
  
 if (newFile.exists()) {  
 System.*out*.println("File already exists.");  
 } else {  
// If file creation is successful, print: "File created: [name]"  
 try {  
 if (newFile.createNewFile()) {  
 System.*out*.println("File created: " + name);  
 } else {  
 System.*out*.println("Failed to create file.");  
 }  
 } catch (IOException e) {  
// If file creation fails, print error message with exception details  
 System.*err*.println("Error creating file: " + e.getMessage());  
 }  
 }  
}  
//===========================printHelp(help command to print information about all available commands  
// List all commands with their descriptions)  
public void printHelp() {  
 System.*out*.println("you need to handle the following commands:");  
 System.*out*.println(" pwd- Print working directory");  
 System.*out*.println(" ls- List files in current directory");  
 System.*out*.println(" cd - Change directory");  
 System.*out*.println(" mkdir - Create new directory");  
 System.*out*.println(" touch - Create a new file");  
 System.*out*.println(" help - Show help information");  
 System.*out*.println(" exit- Exit the program");  
}  
//==========================