Design In-Memory File System

Try to solve the Design In-Memory File System problem.

Statement

Design an in-memory file system. The skeleton for the class `FileSystem` is provided to you. Simulate the following functions:

- `ls`: You're given a string representing a path. If it's a file path, return a list that only contains the file's name. If it's a directory path, return the list of file and directory names in this directory. Your function should return the output (file and directory names together) in lexicographical order.
- `mkdir`: If you have a directory path that does not exist, make a new directory according to the given path. The function should create all the middle directories in the path if they don't exist. This function's return type is `void`.
- `addContentToFile`: You're given a file path and file content in string format. If the file doesn't exist, create that file containing the given content. If the file already exists, append the given content to the original content. This function's return type is `void`.
- `readContentFromFile`: Given a file path, return its content in string format.

Examples

Operation	Output	Explanation
`FileSystem fs`	NULL	The constructor call returns nothing.
`fs.ls("/")`	[]	The directory `/` contains nothing, so we return an empty list.
`fs.mkdir("/dir1/dir2/dir3")`	NULL	Create a directory `dir1` in `/`. Then, create a directory `dir2` in the directory `dir1`. Lastly, create a directory `dir3` in the directory `dir2`.
`fs.addContentToFile("/dir1/dir2/dir3/file1", "File")`	NULL	Create a file `file1` with the contents `File` in the directory `/dir1/dir2/dir3`.
`fs.ls("/")`	["dir1"]	Only directory `dir1` exists in the `/` directory.
`fs.readContentFromFile("/dir1/dir2/dir3/file1")`	"File"	Output the file content.

Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

Design In-Memory File System

1

What is the output for the last two lines of code?

```
fileSystem.mkdir("/dir1/dir2")
fileSystem.addContentToFile("/dir1/dir2/file1", "File")
fs.ls("/")
fileSystem.readContentFromFile("/dir1/dir2/file1")
```

A) “File”
[“dir1”]

B) [“dir1”]
“File”

C) [“dir1”]

D) “File”

Submit Answer

‹ Question 1 of 2 ›
0 attempted

Reset Quiz ↻

Try it yourself

Implement your solution in the following coding playground:

Java

usercode > FileSystem.java

```java
import java.util.*;
public class FileSystem {
    public FileSystem() {
        // Write your code here
    }
    public List <String> ls(String path) {
        // Write your code here
        List < String > files = new ArrayList < > ();
        return files;
    }

    public void mkdir(String path) {
        // Write your code here
    }

    public void addContentToFile(String filePath, String content) {
```

```
    public String readContentFromFile(String filePath) {
        // Write your code here
        return "";
    }
}
```

Powered by AI ▷ Submit

Test Cases Results

Case 1 Case 2 Case 3

Input #1

["FileSystem","ls","mkdir","addContentToFile","ls","readContentFromFile"]

Input #2

[[],["/"],["/a/b/c"],["/a/b/c/d","hello"],["/"],["/a/b/c/d"]]

Design In-Memory File System

Need a Hint?

← Back

Same Tree

Next →

Design File System

☑ Mark as Completed