JAVA.LANG.RUNTIME.EXEC METHOD

Description

The **java.lang.Runtime.exec*String[]cmdarray, String[]envp, Filedir*** method executes the specified command and arguments in a separate process with the specified environment and working directory. Given an array of strings cmdarray, representing the tokens of a command line, and an array of strings envp, representing "environment" variable settings, this method creates a new process in which to execute the specified command.

Starting an operating system process is highly system-dependent. Among the many things that can go wrong are:

- The operating system program file was not found.
- Access to the program file was denied.
- The working directory does not exist.

In such cases an exception will be thrown. The exact nature of the exception is system-dependent, but it will always be a subclass of IOException.

Declaration

Following is the declaration for **java.lang.Runtime.exec** method

```
public Process exec(String[] cmdarray, String[] envp, File dir)
```

Parameters

- **cmdarray** -- array containing the command to call and its arguments.
- **envp** -- array of strings, each element of which has environment variable settings in the format name=value, or null if the subprocess should inherit the environment of the current process.
- **dir** -- the working directory of the subprocess, or null if the subprocess should inherit the working directory of the current process.

Return Value

This method returns a new Process object for managing the subprocess

Exception

- **SecurityException** -- If a security manager exists and its checkExec method doesn't allow creation of the subprocess
- **IOException** -- If an I/O error occurs
- **NullPointerException** -- If command is null
- **IndexOutOfBoundsException** -- If cmdarray is an empty array *haslength0*

Example

This example requires a file named **test.txt** in our C:/ folder with the following contents:

```
Hello
```

The following example shows the usage of lang.Runtime.exec method.

```
package com.tutorialspoint;

import java.io.File;

public class RuntimeDemo {

   public static void main(String[] args) {
   try {

   // create a new array of 2 strings
   String[] cmdArray = new String[2];

   // first argument is the program we want to open
   cmdArray[0] = "notepad.exe";

   // second argument is a txt file we want to open with notepad
   cmdArray[1] = "test.txt";

   // print a message
   System.out.println("Executing notepad.exe and opening test.txt");

   // create a file which contains the directory of the file needed
   File dir = new File("c:/");

   // create a process and execute cmdArray and currect environment
   Process process = Runtime.getRuntime().exec(cmdArray, null, dir);

   // print another message
   System.out.println("test.txt should now open.");

   } catch (Exception ex) {
   ex.printStackTrace();
   }

   }
}
```

Let us compile and run the above program, this will produce the following result:

```
Executing notepad.exe and opening test.txt
test.txt should now open.
```

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js