JAVA.LANG.RUNTIME.EXEC METHOD

Description

The **java.lang.Runtime.exec***Stringcommand, String[]envp, Filedir* method Executes the specified string command in a separate process with the specified environment and working directory. This is a convenience method. An invocation of the form exec*command, envp, dir* behaves in exactly the same way as the invocation exec*cmdarray, envp, dir*, where cmdarray is an array of all the tokens in command.

More precisely, the command string is broken into tokens using a StringTokenizer created by the call new StringTokenizer*command* with no further modification of the character categories. The tokens produced by the tokenizer are then placed in the new string array cmdarray, in the same order.

Declaration

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

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

Parameters

- **command** -- a specified system command.
- **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, or one of the elements of envp is null
- **IllegalArgumentException** -- If command is empty

Example

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 {

   // print a message
   System.out.println("Executing notepad.exe...");
```

```
   // create a file with the working directory we wish
   File dir = new File("C:/");

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

   // print another message
   System.out.println("Notepad 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...
Notepad should now open.
```