

Name: Krishna Mitra

Roll No. 60

Batch T13

## Experiment No-5

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to Test and deploy an application over the tomcat server**

### Programming in Jenkins:

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.” In simple way, Continuous integration (CI) is the practice of frequently building and testing each change done to your code automatically.

Jenkins is a self-contained, open-source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

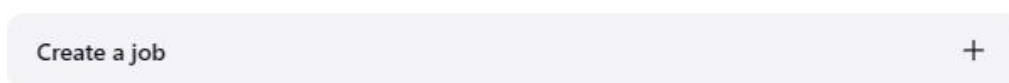
Our first job will execute the shell commands. The freestyle project provides enough options and features to build the complex jobs that you will need in your projects.

### Example 1


#### Example 1.1: Deploying a freestyle app in Jenkins

Creating a job:

#### **Start building your software project**



Naming the job and setting it as freestyle:



Dashboard24\_34\_31\_45\_40\_42\_37\_41

Status

</> Changes

Workspace

Build Now

Configure

Delete Project

Rename

24\_34\_31\_45\_40\_42\_37\_41

Add description

Permalinks

- Last build (#6), 31 min ago
- Last stable build (#6), 31 min ago
- Last successful build (#6), 31 min ago
- Last completed build (#6), 31 min ago

Buils

\*\*\*

Q Filter/

Today

#6 10:41 AM

#5 10:37 AM

#4 10:37 AM

#3 10:37 AM

#2 10:37 AM

#1 10:37 AM

Selecting build type as “Execute shell”:

## Build Steps

Add build step ^

Filter

Execute Windows batch command

Execute shell

Invoke Ant

Invoke Gradle script

Invoke top-level Maven targets

Run with timeout

Set build status to "pending" on GitHub commit

Entering a simple command for the shell execution:

Dashboard24\_34\_31\_45\_40\_42\_37\_41Configuration

Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Execute Windows batch command

Command

See the list of available environment variables

javac C:\Users\richminds\Desktop\sepm\24.java  
java C:\Users\richminds\Desktop\sepm\24.java

Advanced

Add build step

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Add post-build action

Apply

Applying and saving the project configuration:

A blue 'Save' button and a light blue 'Apply' button are shown. Below them is a green rectangular button with a white checkmark and the text 'Saved'.

Building the project:

A light blue button with a play icon and the text 'Build Now'.

Console output (after building):

Dashboard24\_34\_31\_45\_40\_42\_37\_41#1Console Output

Status

Changes

Console Output

Edit Build Information

Delete build '#1'

Timings

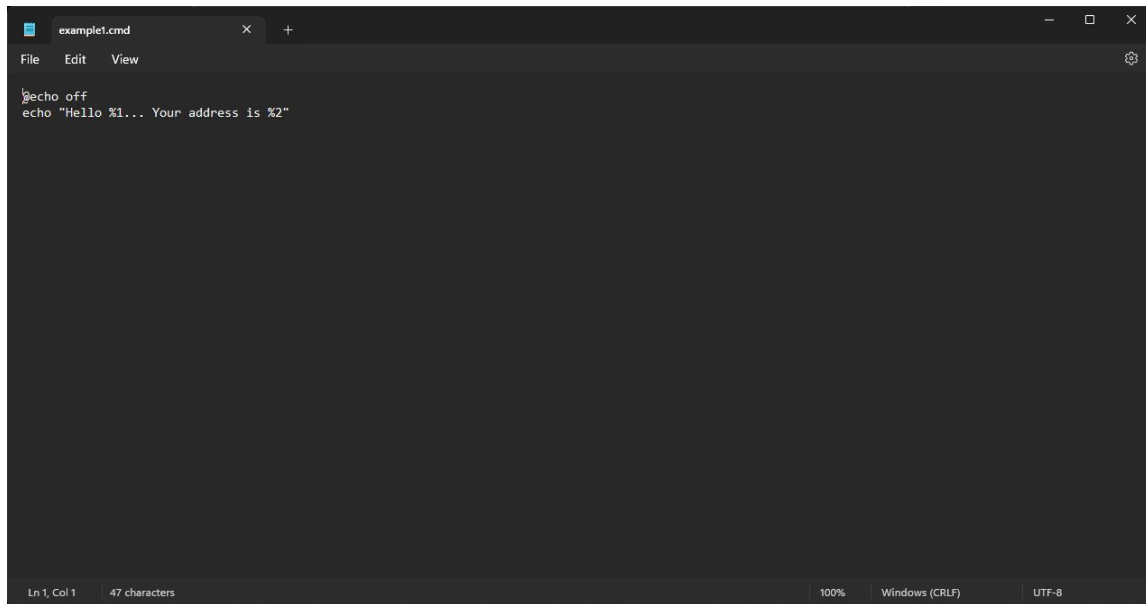
Next Build

Console Output

DownloadCopyView as plain text

Started by user Aditya Dikonda  
Running as SYSTEM  
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\24\_34\_31\_45\_40\_42\_37\_41  
[24\_34\_31\_45\_40\_42\_37\_41] \$ cmd /c call C:\WINDOWS\TEMP\jenkins3499849351956503998.bat  
  
C:\ProgramData\Jenkins\jenkins\workspace\24\_34\_31\_45\_40\_42\_37\_41>javac C:\Users\richminds\Desktop\sepm\24.java  
  
C:\ProgramData\Jenkins\jenkins\workspace\24\_34\_31\_45\_40\_42\_37\_41>java C:\Users\richminds\Desktop\sepm\24.java  
This is T12  
  
C:\ProgramData\Jenkins\jenkins\workspace\24\_34\_31\_45\_40\_42\_37\_41>exit 0  
Finished: SUCCESS

REST APIJenkins 2.503

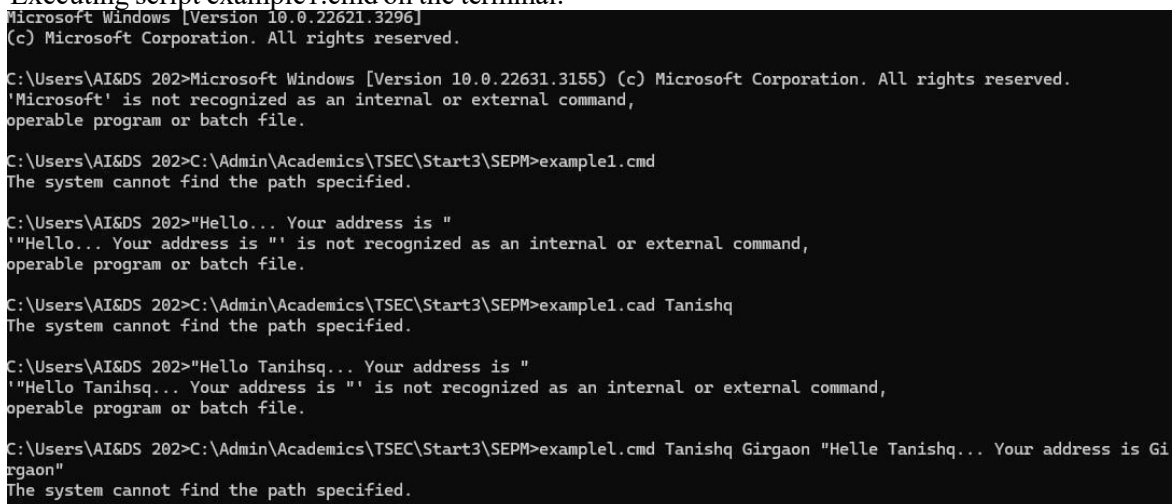


```
example1.cmd
File Edit View
echo off
echo "Hello %1... Your address is %2"
Ln 1, Col 1 47 characters 100% Windows (CRLF) UTF-8
```

### Example 1.2: Taking parameters through files

Contents of script example1.cmd:

Executing script example1.cmd on the terminal:



```
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AI&DS 202>Microsoft Windows [Version 10.0.22631.3155] (c) Microsoft Corporation. All rights reserved.
'Microsoft' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cmd
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello... Your address is "
'Hello... Your address is ' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cad Tanishq
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello Tanishq... Your address is "
'Hello Tanishq... Your address is ' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cmd Tanishq Girgaon "Helle Tanishq... Your address is Gi
rgaon"
The system cannot find the path specified.
```

Modifying the Jenkins project to execute the script while supplying required parameters:



Console output after building the modified project:



: Running

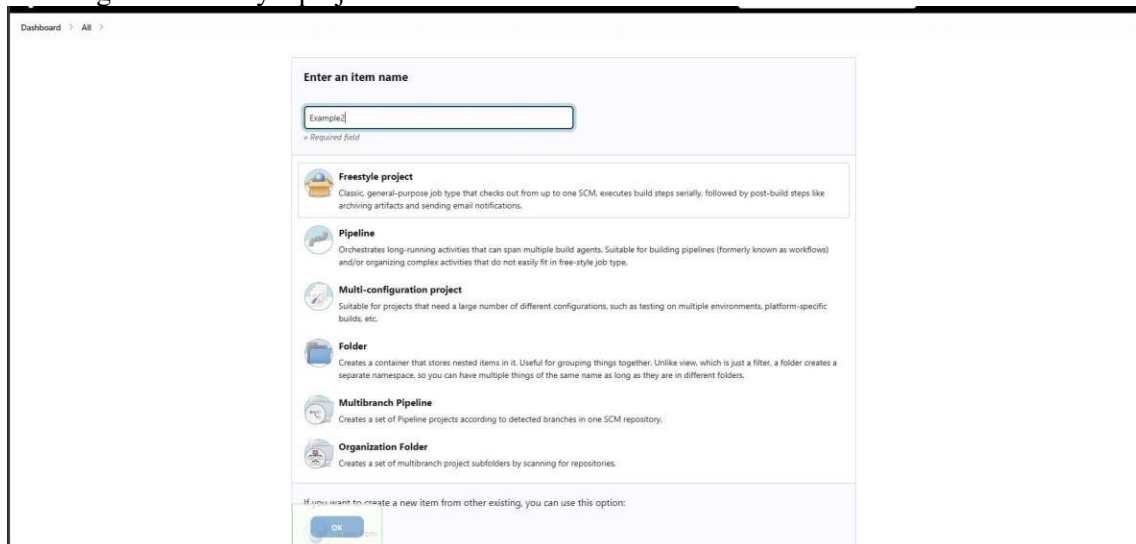
## a Java program under Jenkins

Creating a simple Java program:

Compiling and running the program on the terminal:

```
C:\Users\richminds\Desktop\sepm>javac 24.java
C:\Users\richminds\Desktop\sepm>java 24.java
This is T12
C:\Users\richminds\Desktop\sepm>
```

Creating a new freestyle project:



Configure new project:

Command

See the list of available environment variables

```
javac C:\Users\richminds\Desktop\sepm\24.java
java C:\Users\richminds\Desktop\sepm\24.java
```

Console output after building:

## Console Output

[Download](#)[Copy](#)[View as plain text](#)

```
Started by user Aditya Dikonda
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_41
[24_34_31_45_40_42_37_41] $ cmd /c call C:\WINDOWS\TEMP\jenkins3970528995341461278.bat

C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_41>javac C:\Users\richminds\Desktop\sepm\24.java

C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_41>java C:\Users\richminds\Desktop\sepm\24.java
This is T12

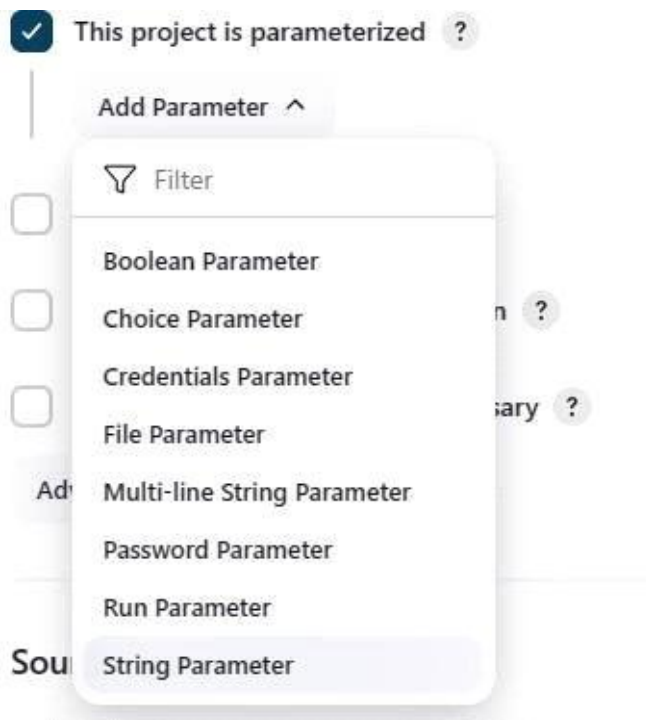
C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_41>exit 0
Finished: SUCCESS
```

## Example 3

### Example 3.1: Parameterise build

Creating a new freestyle project:

Enabling parameterisation and adding a String parameter:



Configuring the string parameter as Fname:

String Parameter ?

Name ?

Fname

Default Value ?

Description ?

Plain text [Preview](#)

☐ Trim the string ?

Adding a choice parameter and configuring it as City with the following choices:

Choice Parameter

Name

City

Choices

Ambernath  
Badlapur  
Kalyan  
Dombivli

! Requires Choices.

Description

Configuring build steps:

Build Steps

Execute Windows batch command ?

Command

See [the list of available environment variables](#)

C:\Admin\Academics\TSEC\Start3\SEPM\example3.cmd %Fname% %City%

Advanced ▼

Add build step ▼

Entering parameters for build:

### Project Example3

This build requires parameters:

Fname

City

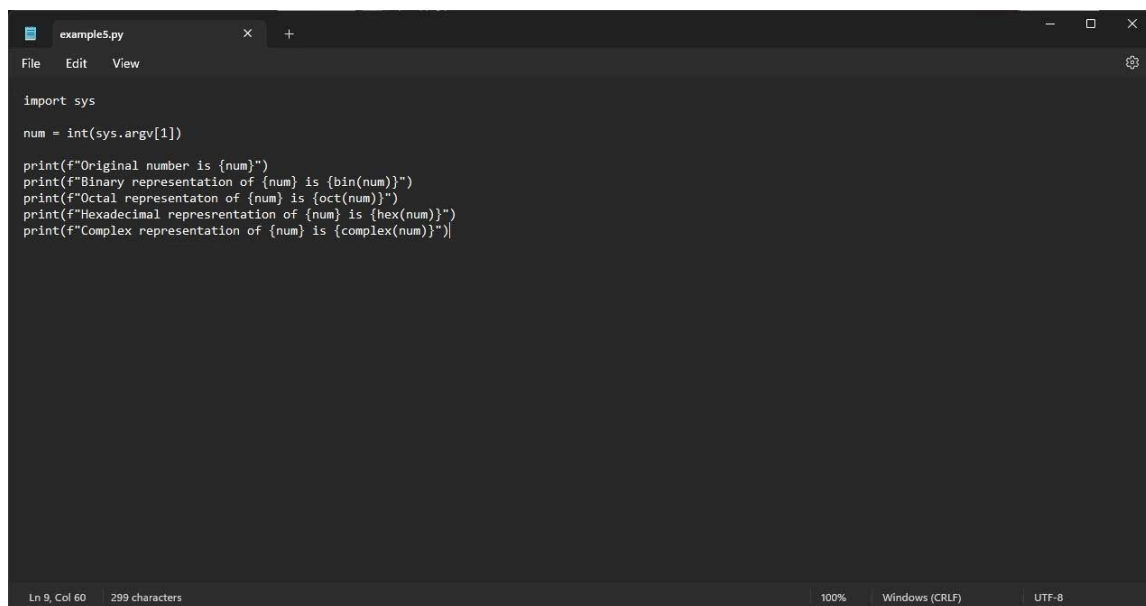
Console output after building:

#### ✓ Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example3
[Example3] $ cmd /c call C:\WINDOWS\TEMP\jenkins14094536165150986151.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example3>C:\Admin\Academics\TSEC\Start3\SEPH\example3.cmd Siddhant Bandra
Hello your name is Siddhant and your city is Bandra
Finished: SUCCESS
```

### Example 5



The screenshot shows a code editor window titled 'example5.py'. The code imports the 'sys' module and takes a command-line argument 'num'. It then prints out the original number and its binary, octal, hexadecimal, and complex representations. The status bar at the bottom indicates 'Ln 9, Col 60', '299 characters', '100%' zoom, 'Windows (CRLF)' line endings, and 'UTF-8' encoding.

```
import sys

num = int(sys.argv[1])

print(f"Original number is {num}")
print(f"Binary representation of {num} is {bin(num)}")
print(f"Octal representation of {num} is {oct(num)}")
print(f"Hexadecimal representation of {num} is {hex(num)}")
print(f"Complex representation of {num} is {complex(num)}")
```

#### Example 5.1: Running a Python program

Creating a simple Python script:

Running the Python script on the terminal:

```
C:\Users\richminds\Desktop\sepm>python 24.py 10
Number is 10

C:\Users\richminds\Desktop\sepm>
```


Creating a new freestyle project:





**Enter an item name**


Example5


» Required field


**Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

OK

om

Parameterising the project with a string parameter as follows:

☒ This project is parameterized ?

String Parameter ?

Name ?

num

Default Value ?

Description ?

Plain text [Preview](#)

☐ Trim the string ?

Add Parameter ▾

Configuring the build steps:

## Command

See the list of available environment variables

```
python C:\Users\richminds\Desktop\sepm\24.py
```

Setting the parameter for the build:

The screenshot shows the Jenkins web interface for a project named "Project 24\_34\_31\_45\_40\_42\_37\_41". The left sidebar contains a navigation menu with the following items: Status, Changes, Workspace, Build with Parameters, Configure, Delete Project, and Rename. The main content area displays the "Build with Parameters" configuration. It shows a parameter named "num" with a value of "10". Below the parameter configuration, there is a "Builds" section with a filter and a list of builds. The builds are listed as follows:

| Build | Time     | Status |
|-------|----------|--------|
| #6    | 10:41 AM | ✓      |
| #5    | 10:37 AM | ✓      |
| #4    | 10:37 AM | ✓      |
| #3    | 10:37 AM | ✓      |
| #2    | 10:37 AM | ✓      |
| #1    | 10:37 AM | ✓      |

**Conclusion:** Thus, we have successfully studied Continuous Integration and installed, configured, and understood programming with Jenkins.