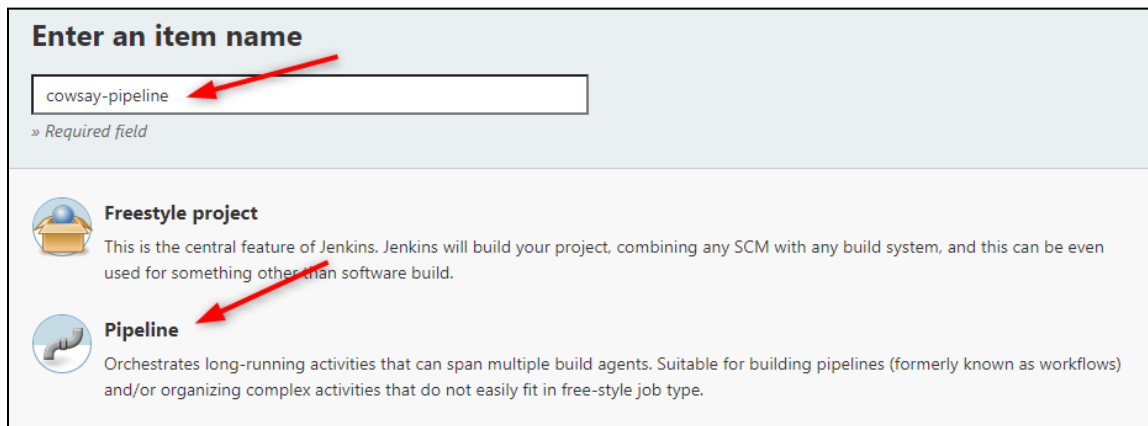


## Pipeline-driven Project

**Objective:** In this lab you will create a **pipeline** project, with the same functionality of the freestyle project you created earlier.

### Project configuration

1. Create a new **"Pipeline"** item, named **"cowsay-pipeline"**



**Enter an item name**

cowsay-pipeline *» Required field*

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**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.

2. Enter a description of your liking.
3. Scroll to the bottom. Under 'Pipeline', under 'Definition', choose "Pipeline script". This is usually the default setting.
4. In the script field, enter the following code:

## Pipeline

### Definition

Pipeline script

Script ?

```
1 pipeline {  
2   agent any  
3  
4   stages {  
5     stage('Hello World') {  
6       steps {  
7         sh '/usr/games/cowsay "Hello World"'  
8       }  
9     }  
10  }  
11 }  
12
```

### 5. Save and Build Now

### 6. Your Build screen should look like this:

## Stage View

Average stage times:  
(Average full run time: ~5s)

#1

Apr 04  
15:39

No  
Changes

Hello World

588ms

588ms

## Permalinks

7. Your console output should look like this:

## Console Output

```
Started by user XXXXXXXXXX
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /root/.jenkins/workspace/cowsay-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello World)
[Pipeline] sh
+ /usr/games/cowsay Hello World

  _____
< Hello World >
  -----
      \   ^__^
       \  (oo)\_______
          (__)\       )\/\
              ||----w |
              ||     ||

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

8. You should have a cow

## Add Parameterization

9. Get back into the configuration for this project
10. After the 'agent any' line, but before the 'stages' line, add this block:

```
parameters {  
    string(name: 'cow_pipeline_param', defaultValue: '$JOB_NAME', description:  
    'What the pipeline cow says')  
}
```

```
1 pipeline {  
2     agent any  
3     parameters {  
4         string(name: 'cow_pipeline_param', defaultValue: '$JOB_NAME', description: 'What the pipeline cow says')  
5     }  
6     stages {  
7         stage('Hello World') {
```

11. Update the '**Hello World**' stage to use the new parameter. You are looking to print out "Hello cowsay-pipeline".

So change the command in this line:

```
sh 'cowsay "Hello World" '
```

to...

What do you think it should look like? Look at the parameter string you added and try to create the correct cowsay line. Refer to your "cow-world" freestyle project to see how to write the linux line to have cowsay use a parameter, and combine that with the new parameter you are using in this lab.

**NOTE: If you are unable to work this step out on your own, the solution is listed at the bottom of this lab doc**

Save, build and watch the magic

**NOTE: Due to a marvelous quirk in Jenkins Pipeline jobs, you may have to run this build twice to get it to use the parameter for the first time**



## Console Output

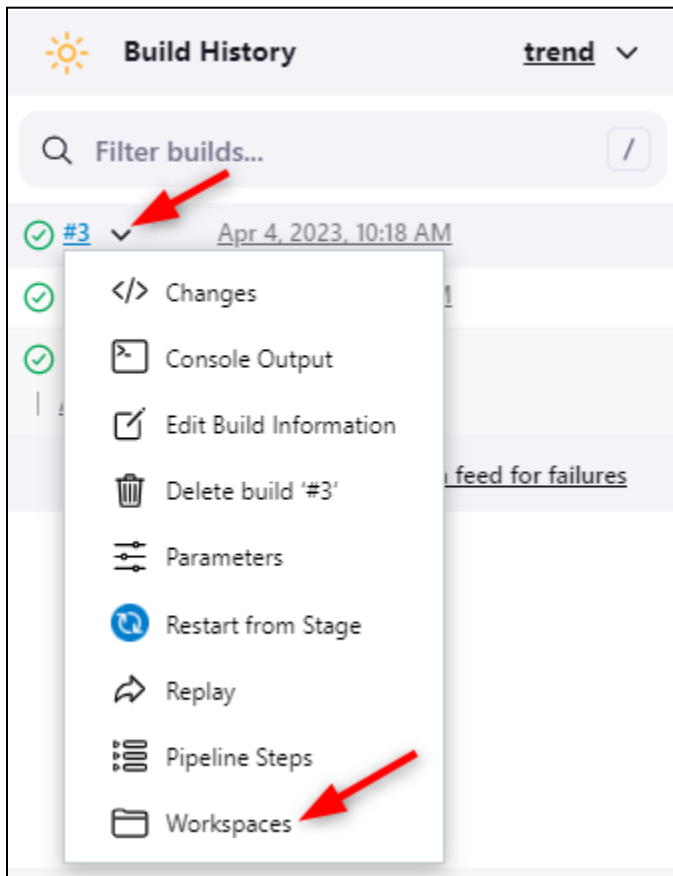
```
Started by user XXXXXXXXXX
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /home/ubuntu/.jenkins/workspace/cowsay-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello World)
[Pipeline] sh
+ /usr/games/cowsay Hello cowsay-pipeline

-----
< Hello cowsay-pipeline >
-----
      \  ^__^
       \ (oo)\_______
          (__)\       )\/\
             ||----w |
             ||     ||

[Pipeline] }
[Pipeline] // stage
```

### Write to a file in the workspace

12. Go back to the configuration of your project.
13. Change the linux command in the 'Hello World' stage so it writes to a file, called cowput.txt. If you are not able to do this, look at the solution at the bottom of this lab doc.
14. Run your build, find your file and view it. You'll need to look in the **Build History dropdown** for this build number to find the workspace.



15. Remember to use your browser back button to return to Jenkins after viewing the `cowput.txt` file. If you close this tab, you will close the Jenkins interface

Notify your instructor that you are done with the lab

END OF LAB

**If you were unable to make the code for those last two steps work, the solution is below:**

...

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

#### Script

```
1 pipeline {
2   agent any
3   parameters {
4     string(name: 'cow_pipeline_param', defaultValue: '$JOB_NAME', description: 'What the pipeline cow says')
5   }
6   stages {
7     stage('Hello World') {
8       steps {
9         sh '/usr/games/cowsay "Hello $cow_pipeline_param" | tee cowput.txt'
10      }
11    }
12  }
13 }
14
```