JENKINS

Contents

[DIFFERENT WAYS TO INSTALL JENKINS 1](#_Toc73119513)

[UNINSTALLING JENKINS 3](#_Toc73119514)

[Jenkins Questions 5](#_Toc73119515)

[**Pipeline syntax** 6](#_Toc73119516)

[**Jenkins Pipeline Concepts** 7](#_Toc73119517)

[Jenkins Backup 8](#_Toc73119518)

[Groovy Language 9](#_Toc73119519)

[First groovy script 10](#_Toc73119520)

[Working with string, number, Boolean values 10](#_Toc73119521)

[Control Structures (if-else, for loop) 10](#_Toc73119522)

[Sub Routines (function) 11](#_Toc73119523)

[Classes and Objects 11](#_Toc73119524)

# DIFFERENT WAYS TO INSTALL JENKINS

**Jenkins Documentation:**

<https://www.jenkins.io/doc/book/using/>

**Jenkins Pipeline Documentation:**

<https://www.jenkins.io/doc/book/pipeline/>

https://github.com/g0t4/course-jenkins-getting-started

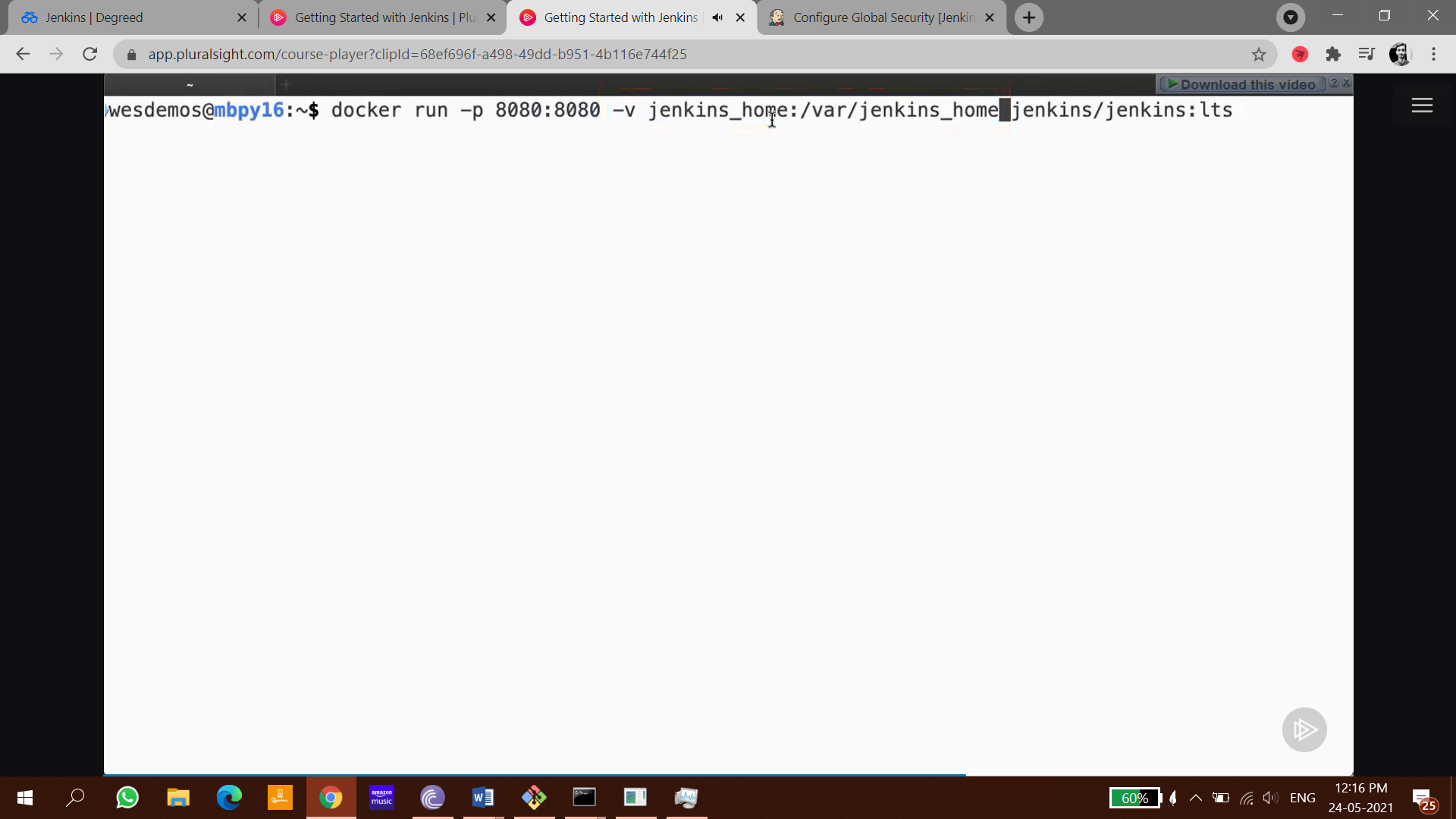
**Check Java Version:**

Javac -version

Java is pre-requisite for jenkins

**You can also use jenkins via docker.**

You have to pull: docker pull jenkins



**Install jenkins via docker-compose:**

|  |
| --- |
| version: "3.8" |
|  |  |
|  | services: |
|  |  |
|  | jenkins: |
|  | image: jenkins/jenkins:2.255 |
|  | ports: |
|  | - "127.0.0.1:9080:8080" |
|  | volumes: |
|  | - ./jenkins\_home\_on\_host:/var/jenkins\_home |
|  | # using a bind mount to the host `./jenkins\_home` means I can easily peruse the jenkins\_home files without needing to "get into the container" |
|  | restart: unless-stopped |
|  | # a named volume is fine too - jenkins\_home:/var/jenkins\_home |
|  |  |
|  |  |
|  |  |
|  | # I like docker-compose because I can easily spin up multiple apps like a test email server alongside jenkins! for testing email notifications from jenkins (later) |
|  | mails: |
|  | image: mailhog/mailhog |
|  | restart: unless-stopped |
|  | ports: |
|  | - "127.0.0.1:8025:8025" # mailhog's web app (think test instance of gmail) |
|  | # "127.0.0.1:1025:1025" # jenkins will use `mail:1025` to send emails, only map to host if you want to send files |
|  |  |
|  |  |
|  | ## NOTES: |
|  | # Host port bindings are constrained to listening on 127.0.0.1 |
|  | # - avoids external access to services |
|  | # - To open external access: |
|  | # - Remove IP address constraint to open the flood gates |
|  | # - Bind to another IP |
|  | # - https://docs.docker.com/config/containers/container-networking/#published-ports |

Blue Ocean is a GUI to create a pipeline

In pipeline we define sequence of steps

We can say Job is a part of pipeline

**Execute jenkins via war file**=> Java –jar jenkins.war

Open Console/Command line --> Go to your Jenkins installation directory. Execute the following commands respectively:

**to stop:**  
jenkins.exe stop

**to start:**  
jenkins.exe start

**to restart:**jenkins.exe restart

# UNINSTALLING JENKINS

If you want a clean reinstall, you will need to delete some more, in addition to uninstalling and deleting the Jenkins home folder. Otherwise, they will remain there forever and will show up after reinstalling. To completely remove everything for Jenkins from Windows (I am giving this information from Windows 10, 64 bit):

1. Open Windows **Control Panel** and click Programs and Features
2. Right-click **Jenkins** x.xxx.x (where x.xxx.x, is the Jenkins version), and select Uninstall.
3. Follow the directions.
4. When done, delete the following folders: (**note:** DO NOT remove the jenkins.jenkins or jenkinswar folders if you want to keep the projects and plugins.)

C:\Program Files\Jenkins or C:\Program Files (x86)\Jenkins

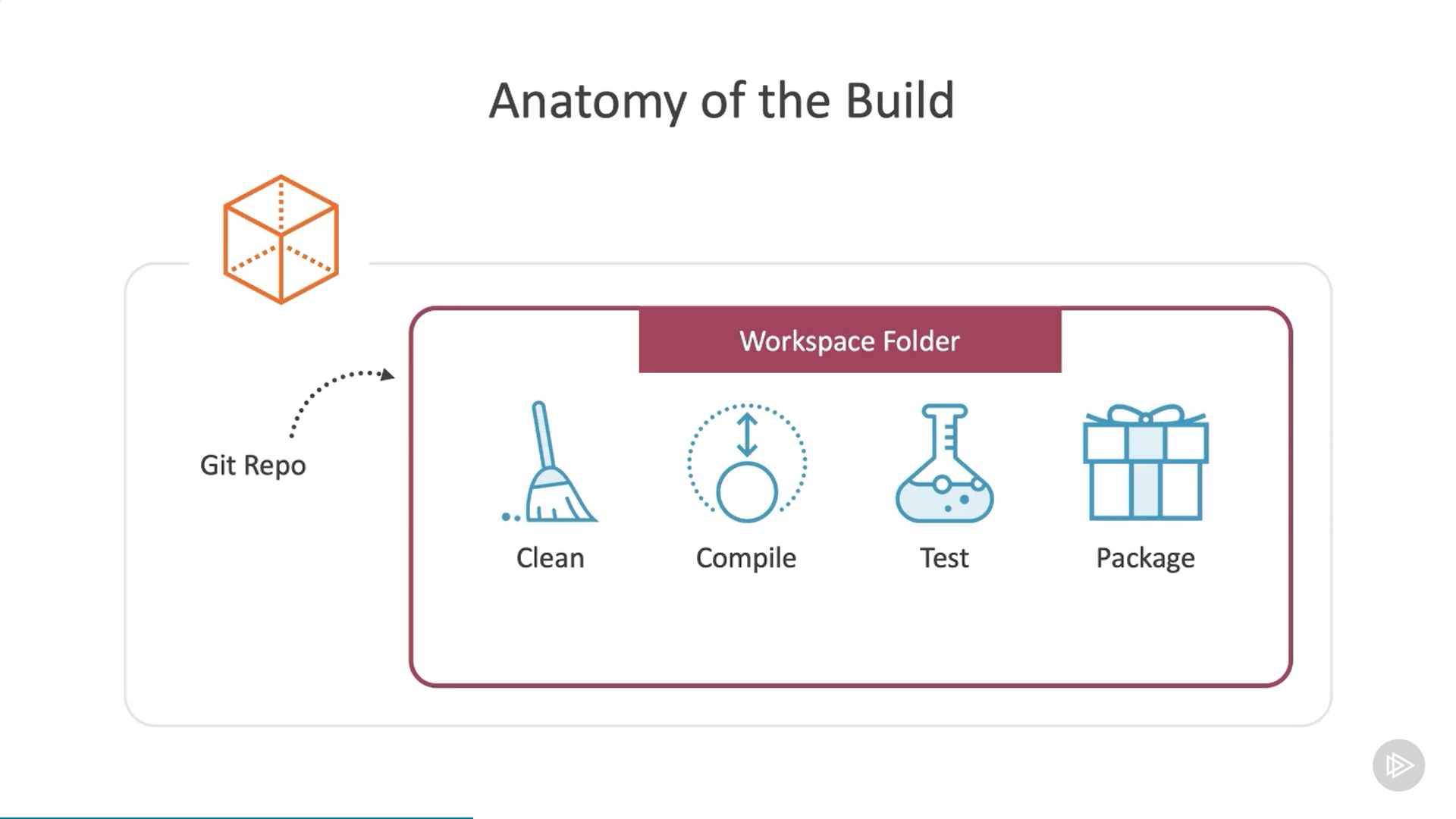
C:\Windows\System32\config\systemprofile\AppData\Local\Jenkins.jenkins

C:\Windows\System32\config\systemprofile\AppData\Local\Jenkinswar

1. Restart the computer.
2. **Clone the Git Repository**

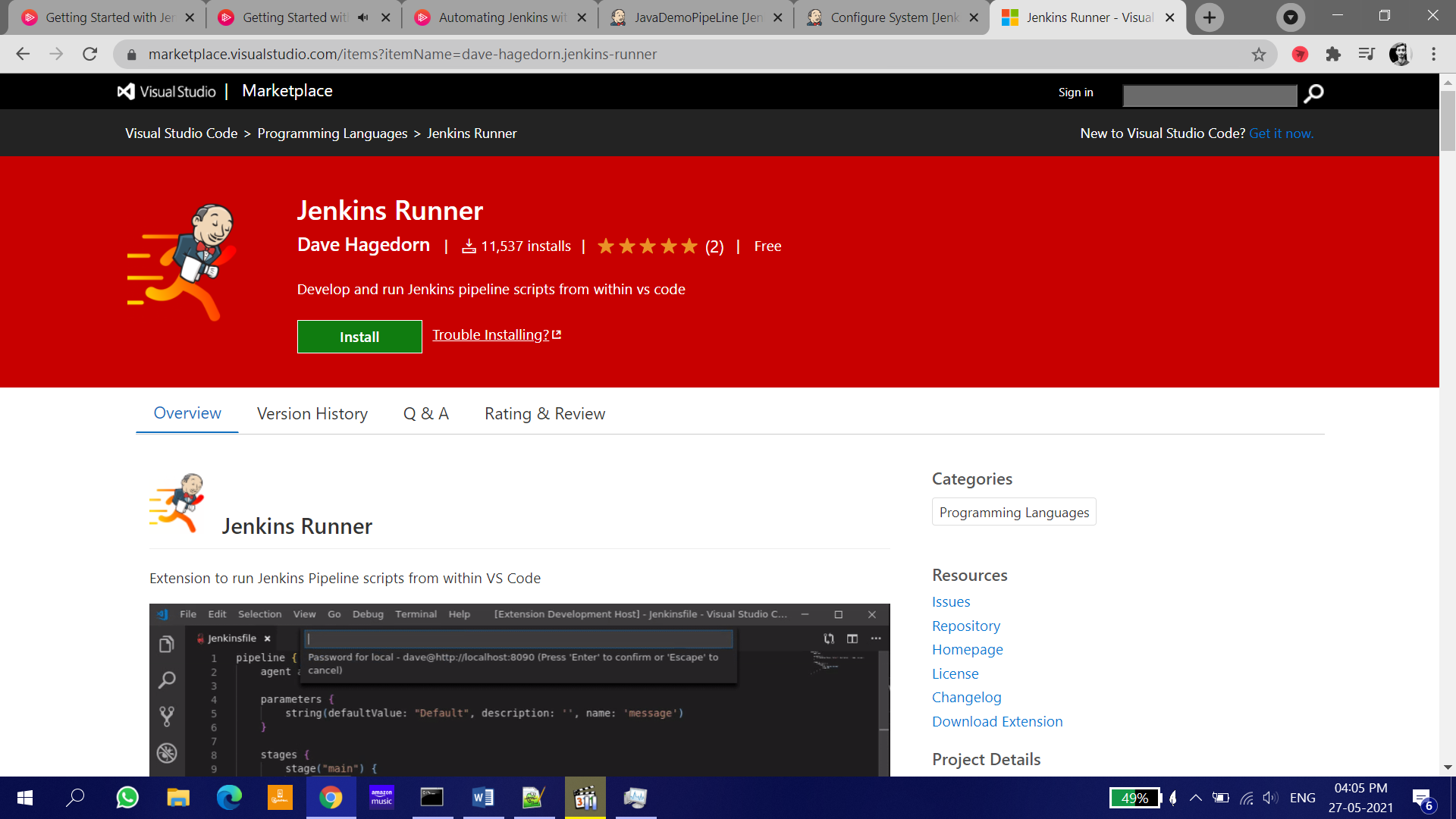
Git clone [git-repo-link.git]

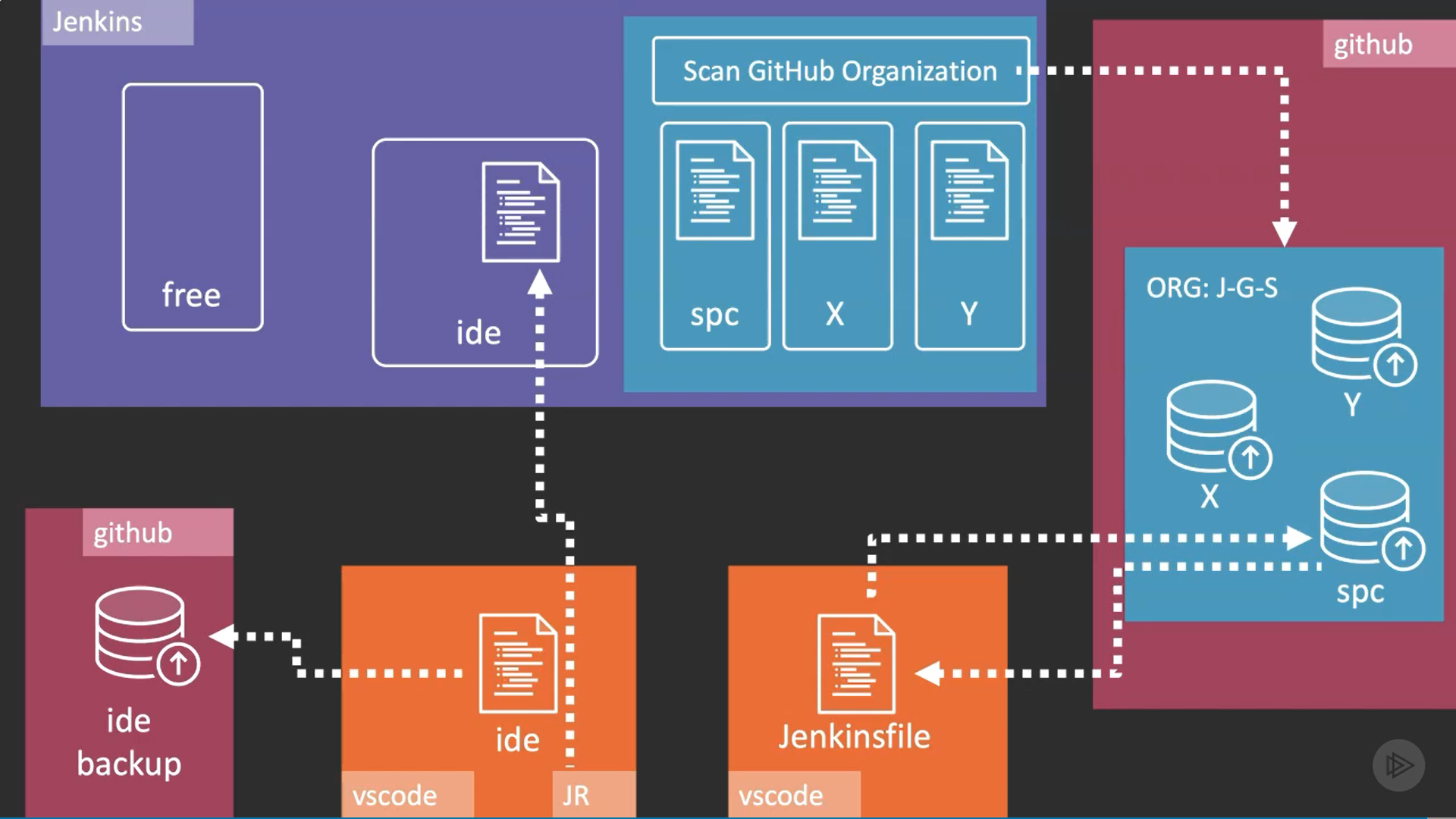
If you want to install maven dependencies=> ./mvnw [install/clean/package]

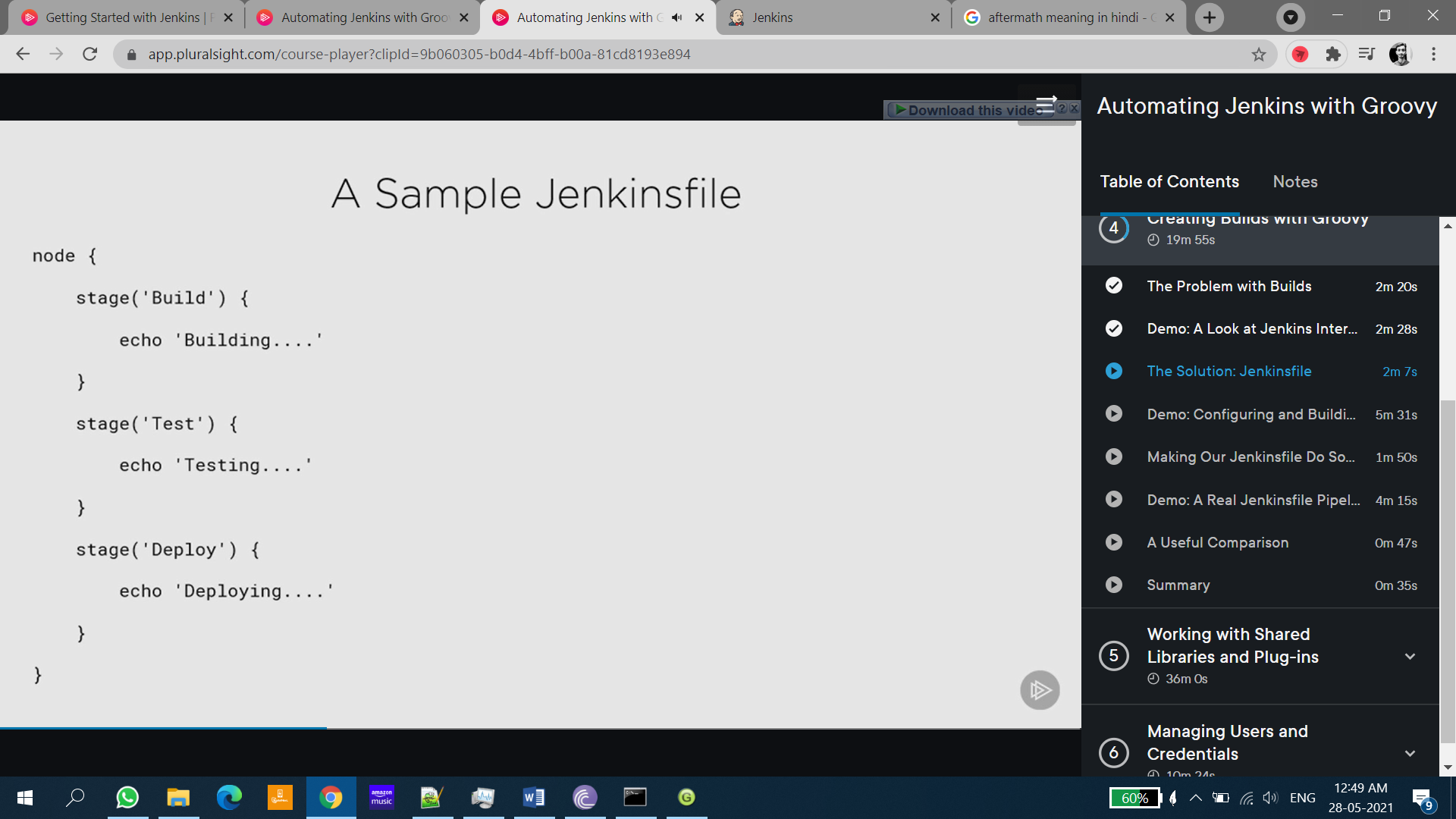




If you want to execute jenkins script on vs code







# Jenkins Questions

#### **How do you install Jenkins?**

**Answer:** To install Jenkins, make sure the following are installed –

* Java (version 8)
* Apache Tomcat (version 9)

Download the Jenkins war file and deploy it using Tomcat. You can choose to install the plugins suggested by Jenkins during the installation itself. Once the installation is done, you will be able to see the Jenkins dashboard.

#### **Question: What is Groovy?**

**Answer:** [Groovy](https://hackr.io/tutorials/learn-apache-groovy?ref=blog-post) from Apache is a language for Java platform. It is the native scripting language for Jenkins. Groovy-based plugins enhance Jenkins with great interfaces and build reports that are dynamic and consistent.

#### **Question: What is a job in Jenkins?**

**Answer:** A job or build job is a task or step in the entire build process. It could be compiling the source code, running unit tests, deploying the application to the web server and so on.

#### **Question: What is meant by Jenkins pipeline?**

**Answer:** A pipeline is a group of interlinked jobs done one after the other in a sequence. To integrate and implement continuous delivery pipelines, Jenkins pipelines provides a combination of plugins. The instructions to be performed are given through code. A continuous delivery pipeline can be represented as –

#### **Question: What are the types of pipelines in Jenkins?**

**Answer:** There are 3 types –

1. CI CD pipeline (Continuous Integration Continuous Delivery)
2. Scripted pipeline
3. Declarative pipeline

#### **Question: Describe the process to create a backup and copy files in Jenkins?**

**Answer:**To create a backup, you should periodically back up your JENKINS\_HOME directory.

This directory contains all the build jobs configurations, slave node configurations, and build history.

To create a backup, copy this directory, or you can also copy a job directory to replicate any job or rename the directory.

#### **Question: Explain the process of creating the Multibranch Pipeline in Jenkins?**

**Answer:**The process is as follows:

* Open the Jenkins dashboard and create a new item by clicking on ‘new item’ from the top left menu.
* Enter your project name and from the options shown, select ‘Multibranch pipeline’ and click on OK.
* Then you should select the repository location, branch source (GitHub/Bitbucket), and add the credentials of the branch source.
* Save the project.
* Jenkins automatically creates new Multibranch Pipelines for repositories that have branches and pulls requests containing Jenkins files.
* To connect to the GitHub repo, we need the HookURL. Get this URL from the repository settings.
* Add this HookURL to the Webhooks section.
* Once the jobs are created, build is automatically triggered by Jenkins.

#### **Question: What is Agent Directive in Jenkins?**

**Answer:**The agent directive directs Jenkins on how and where to execute the Pipeline or its subsets. All the pipelines require agents. The agent causes a workspace to be allocated that contains checkout files from source control and other additional working files required for the Pipeline. It also causes the steps required for execution to be executed by Jenkins when an executor is available.

#### **Question: What is a backup plugin? Why is it used?**

**Answer:** This is a useful plugin that backs up all the critical settings and configurations to be used in the future. This is especially useful when there is a failure so that we don’t lose the settings.

## **Pipeline syntax**

Two types of syntax are used for defining your JenkinsFile.

* Declarative
* Scripted

**Declarative:**

Declarative pipeline syntax offers a simple way to create pipelines. It consists of a predefined hierarchy to create Jenkins pipelines. It provides you the ability to control all aspects of a pipeline execution in a simple, straightforward manner.

**Scripted:**

Scripted Jenkins pipeline syntax runs on the Jenkins master with the help of a lightweight executor. It uses very few resources to convert the pipeline into atomic commands.

## **Jenkins Pipeline Concepts**

**Pipeline:** This is the user-defined block, which contains all the processes such as build, test, deploy, etc. it is a group of all the stages in a JenkinsFile. All the stages and steps are defined in this block. It is used in declarative pipeline syntax.

1. pipeline{
2. }

**Node:** The node is a machine on which Jenkins runs is called a node. A node block is used in scripted pipeline syntax.

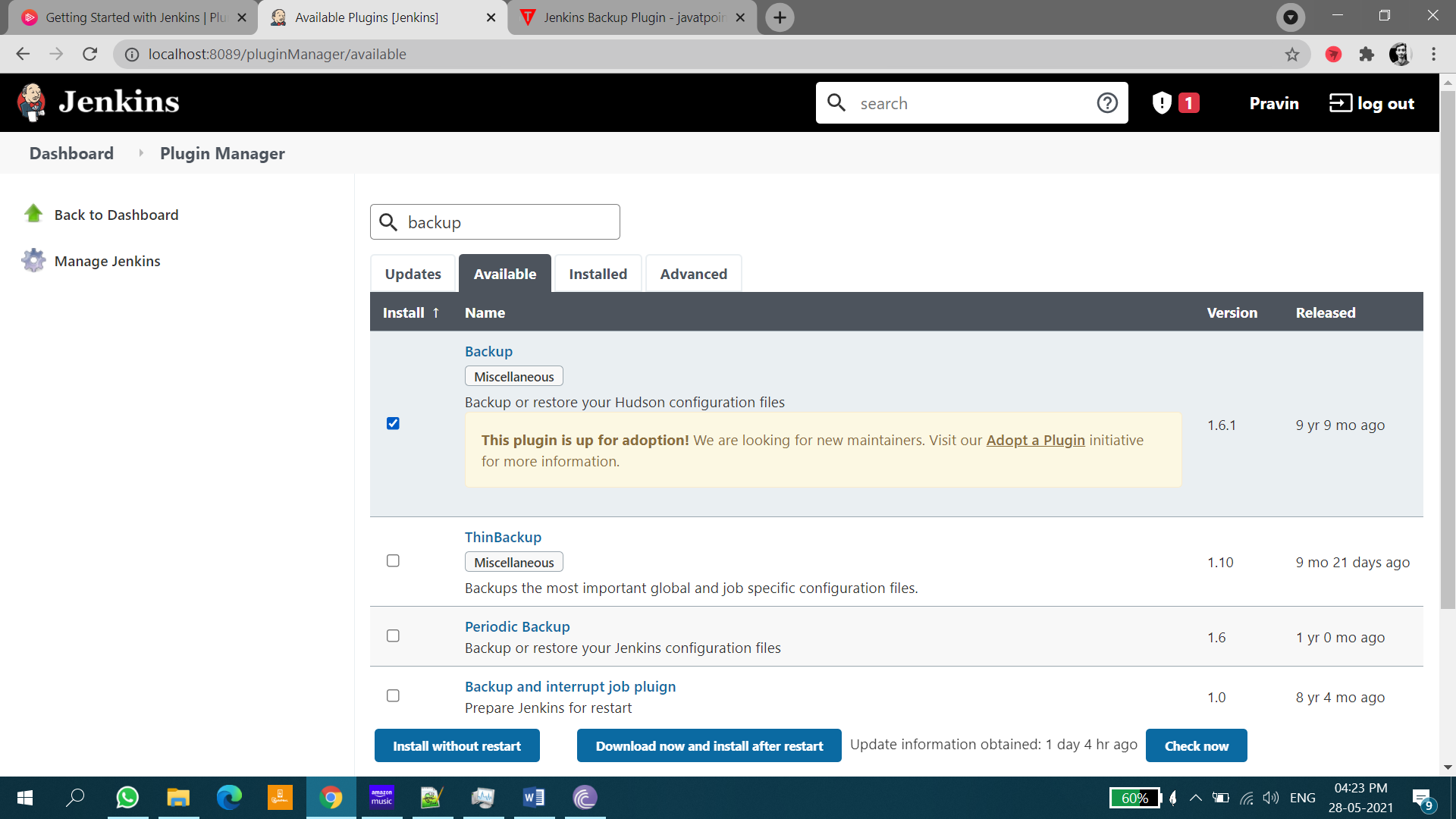
1. node{
2. }

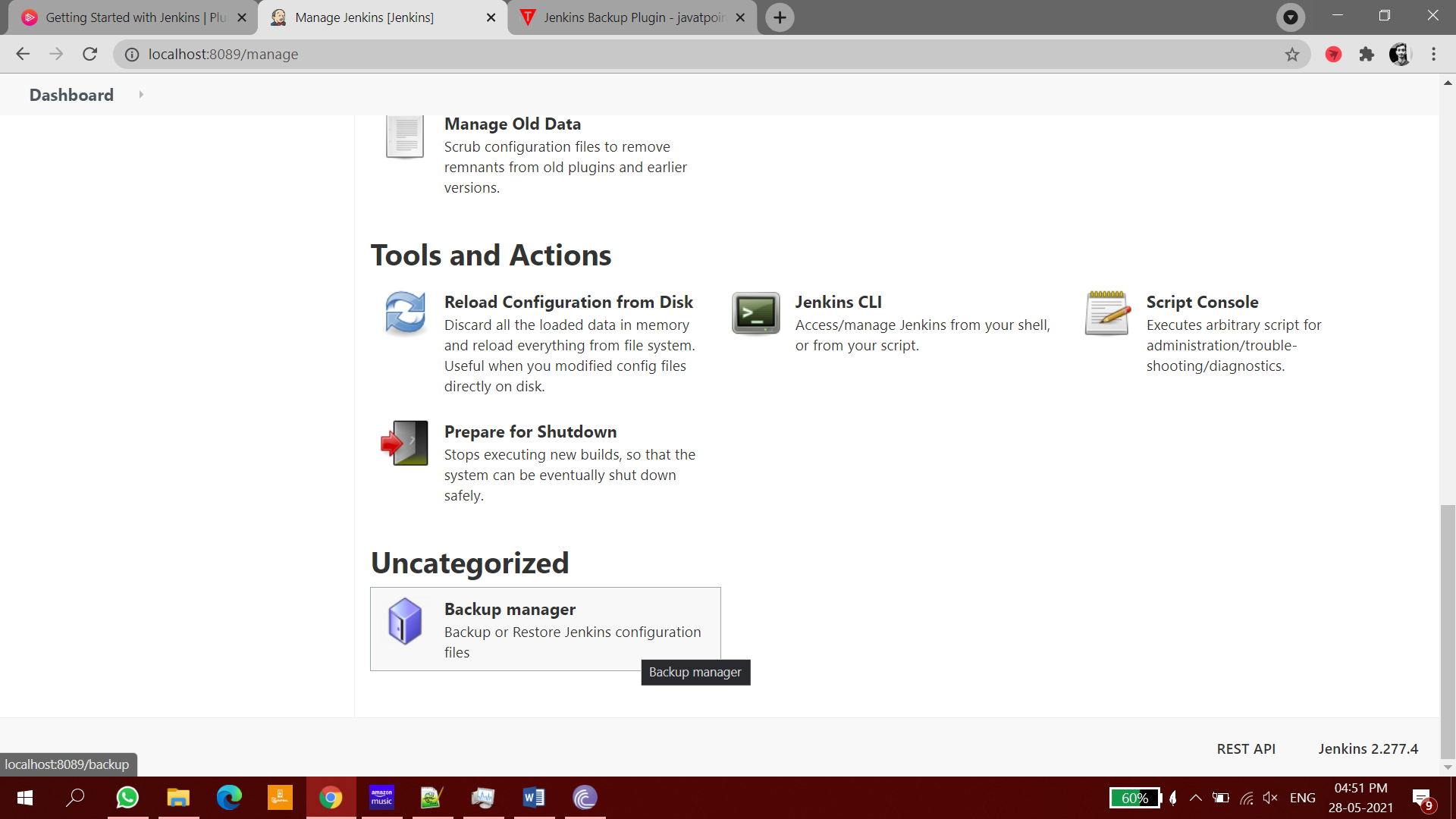
**Stage:** This block contains a series of steps in a pipeline. i.e., build, test, and deploy processes all come together in a stage. Generally, a stage block visualizes the Jenkins pipeline process.

Let's see an example for multiple stages, where each stage performs a specific task:

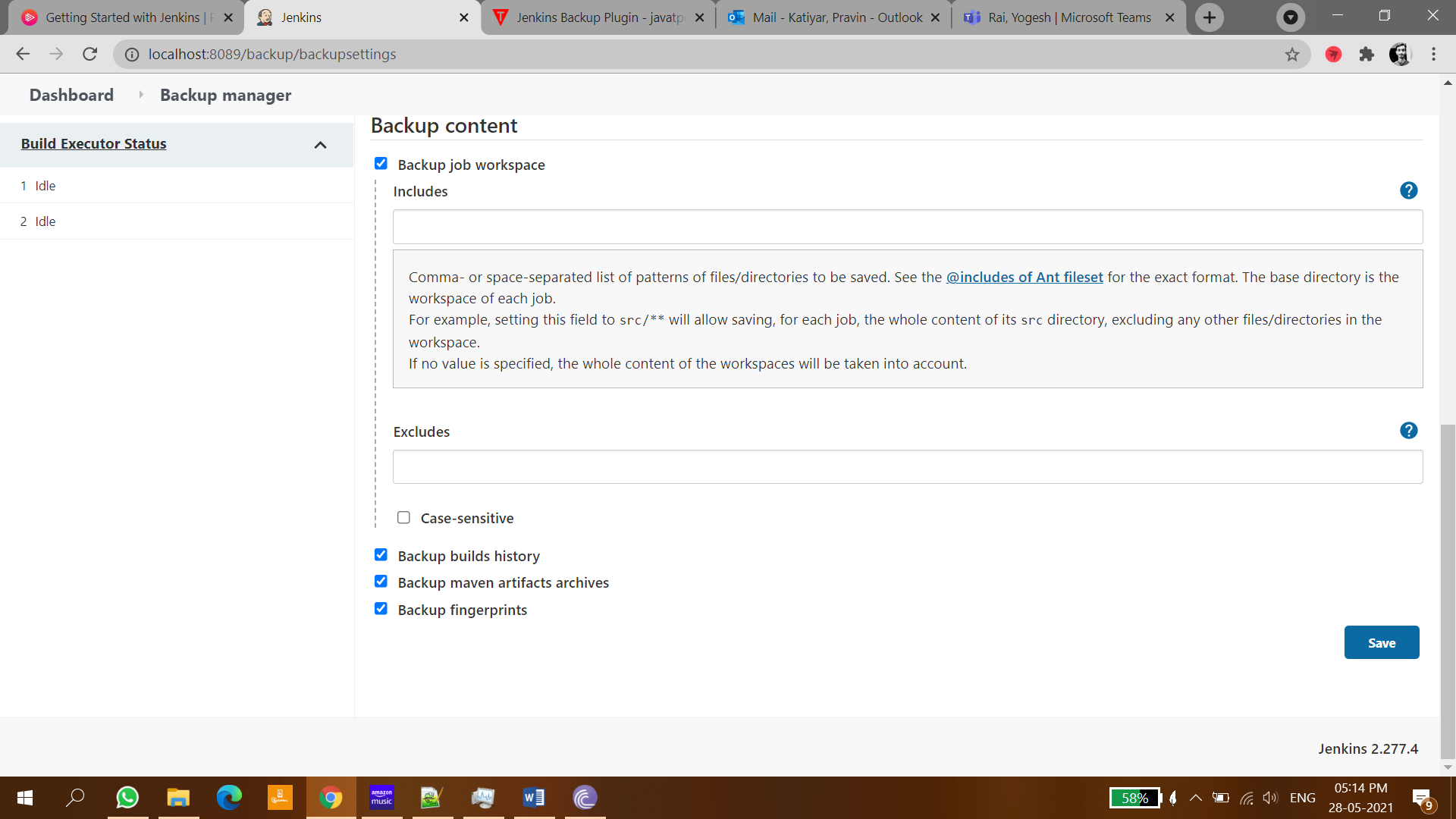
1. pipeline {
2. agent any
3. stages {
4. stage ('Build') {
5. ...
6. }
7. stage ('Test') {
8. ...
9. }
10. stage ('QA') {
11. ...
12. }
13. stage ('Deploy') {
14. ...
15. }
16. stage ('Monitor') {
17. ...
18. }
19. }
20. }

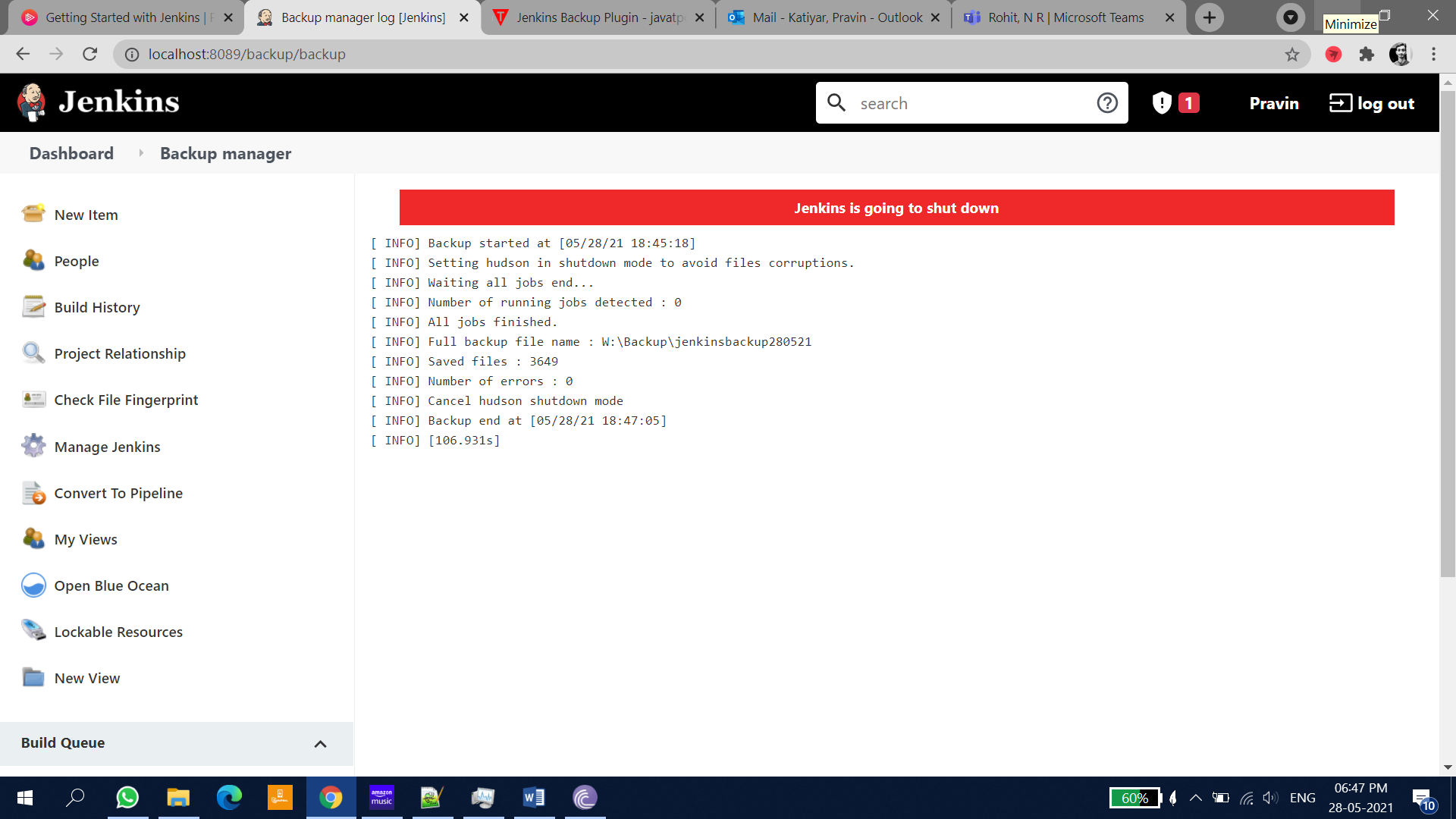
# Jenkins Backup







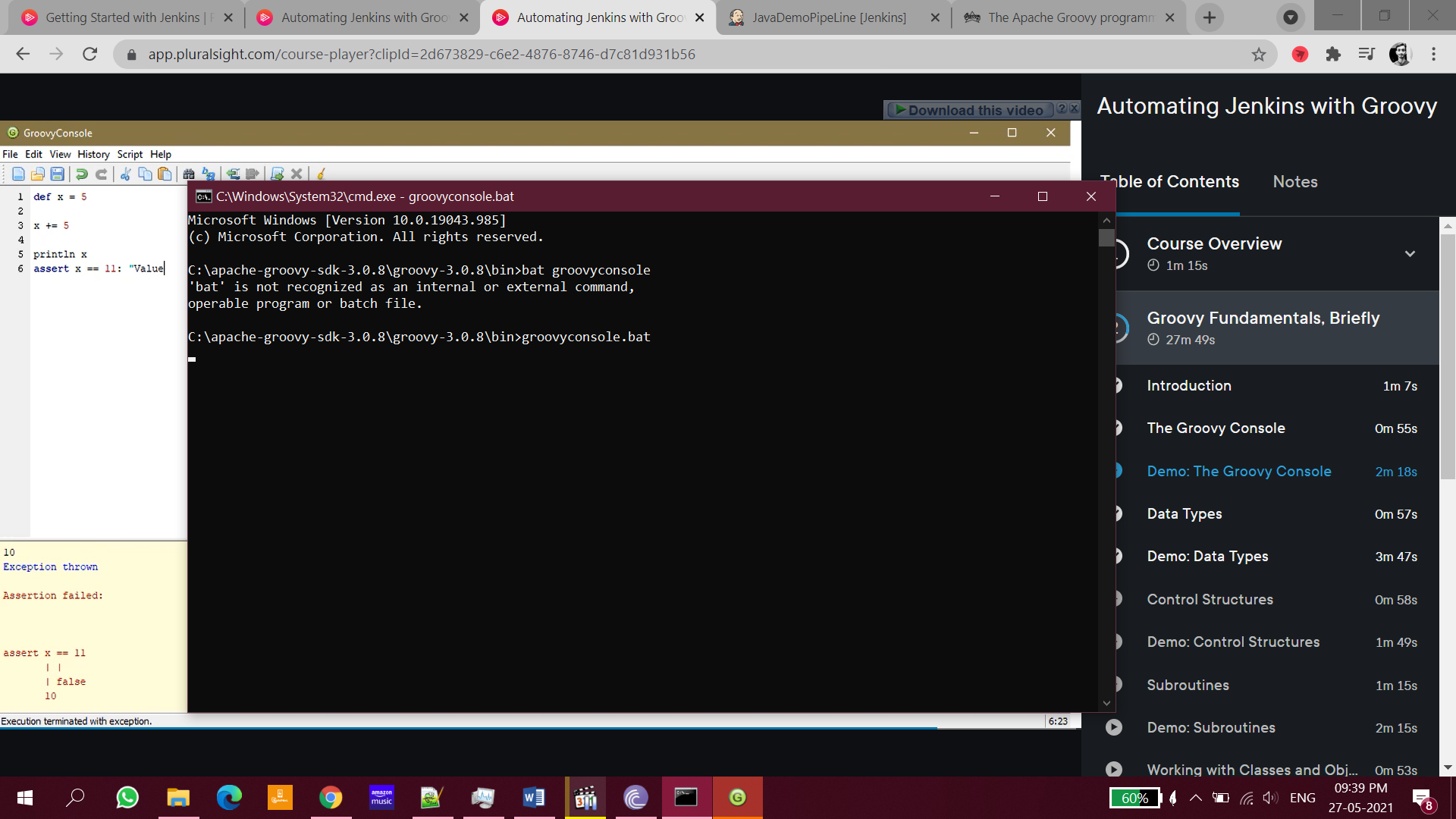




# Groovy Language

Download groovy Console

<https://groovy.apache.org/download.html>



## First groovy script

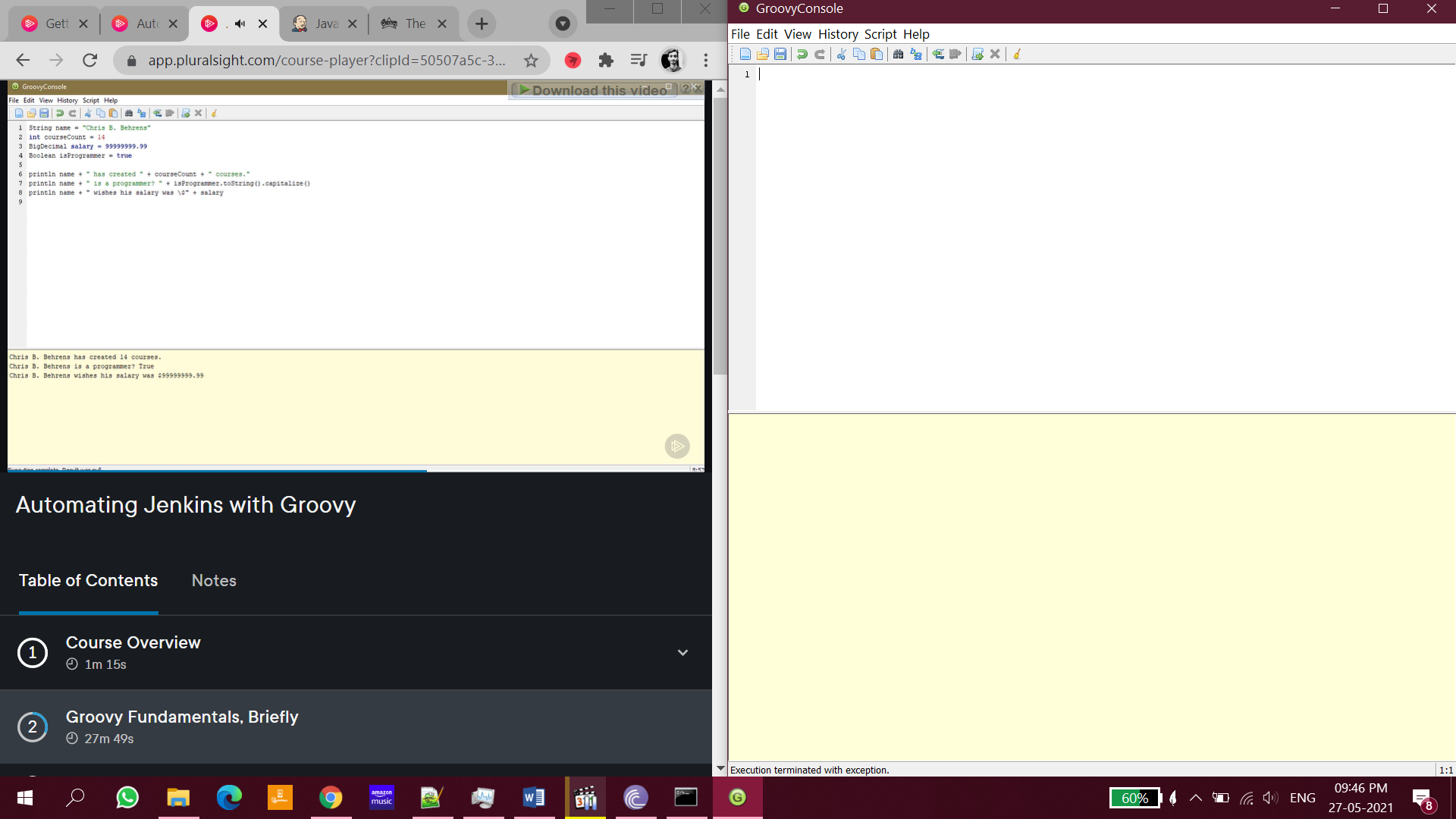
def x=5

x=x+5

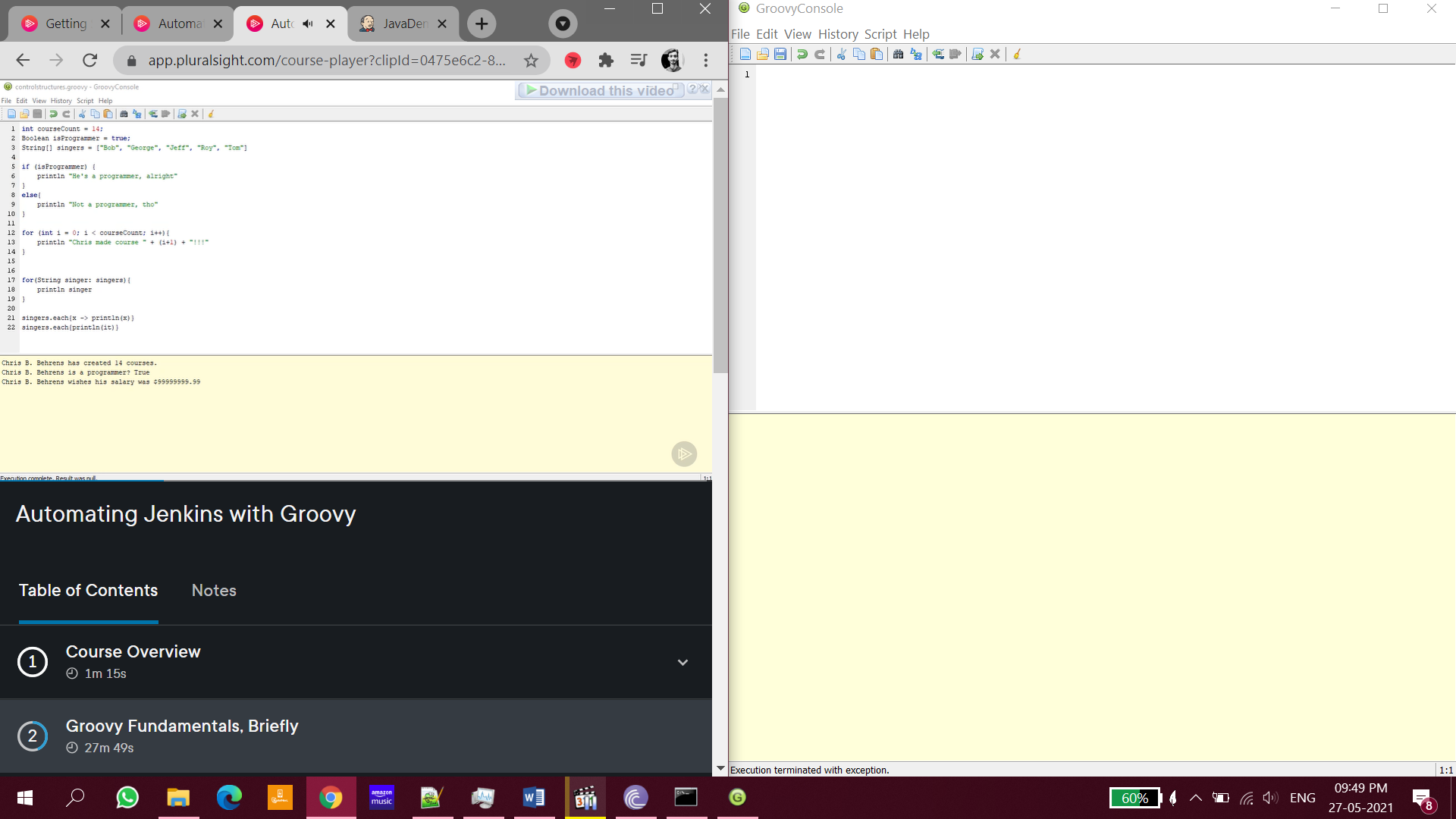
println x

assert x==12:"wrong answer"

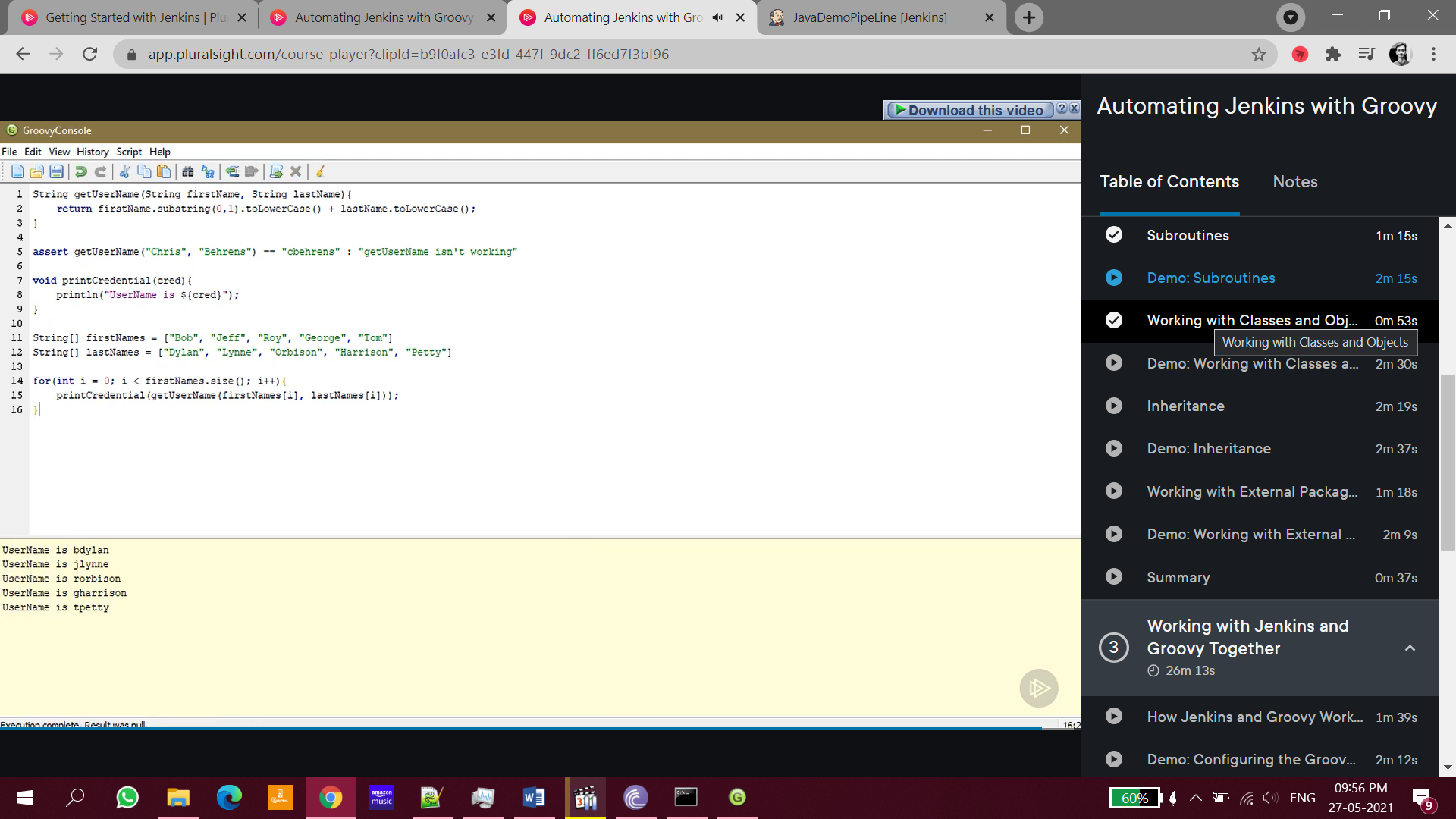
## Working with string, number, Boolean values



## Control Structures (if-else, for loop)



## Sub Routines (function)



Ex: *String getUserName(String fname,String lname){*

*return fname.substring(0,1).toUpperCase() +lname.toUpperCase();*

*}*

*println(getUserName("Rohit","Kumar"));*

## Classes and Objects

