✓ Maven → Java Developers, Testers

↳ Build Tool

JAVA Desktop app →

Build

Source Code → N classes

MobileApp → Android → App ↳ .apk

App iOS ↳ .ipa

app → .exe → Build

(.jar) → Build

MAVEN → Developers → Build Generation

↓

Build Tool → Testers → " , Dependencies Mgmt

Online/Global Repo [MAVEN Repository → https://mvnrepository.com/

→ Dependency / libs / jar

```
<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.6.1</version>
    <scope>test</scope>
</dependency>
```

Orgⁿ → HR policies → 10

→ Teams – Dev → } Documents/code

→ Testers

↓ 101

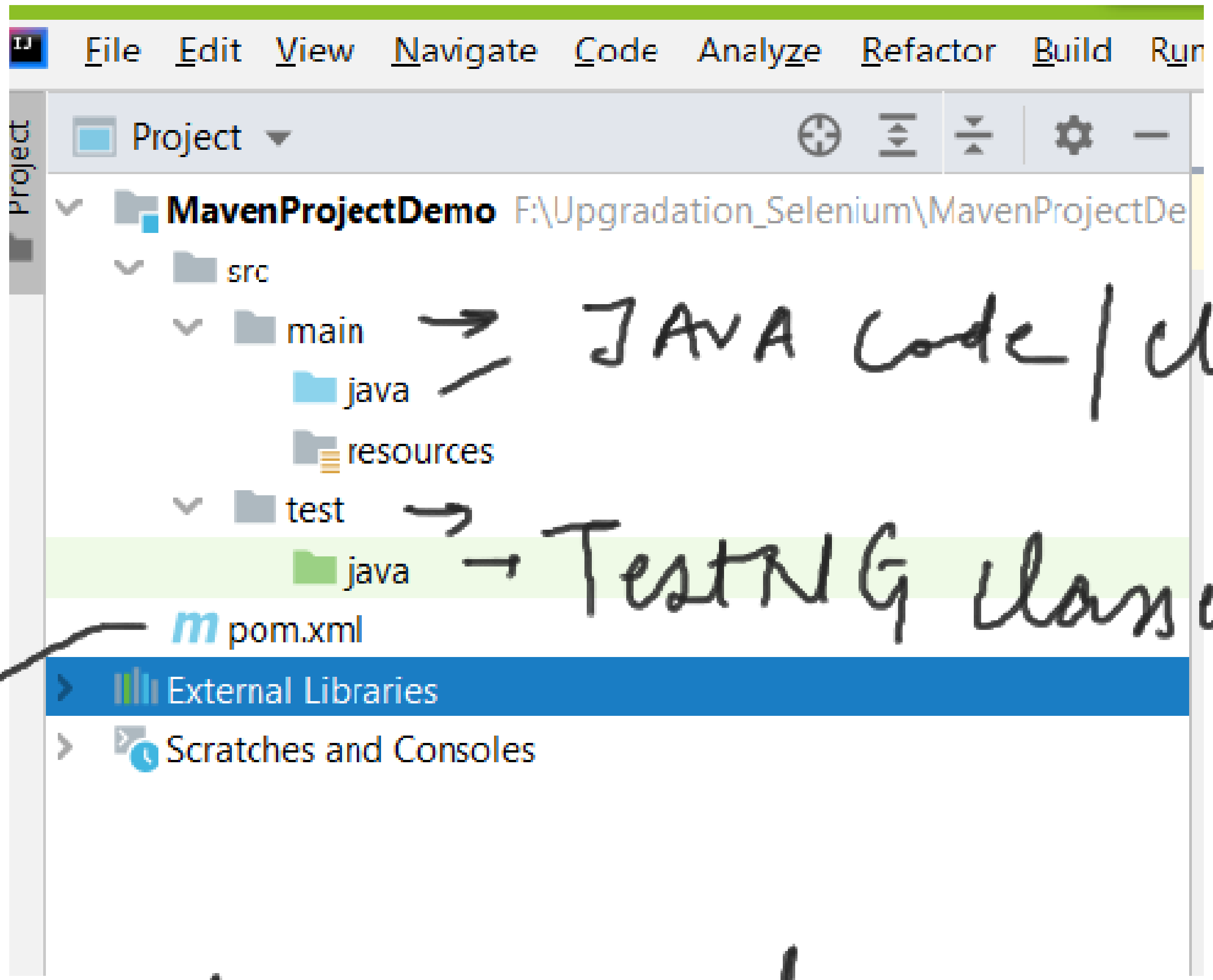
→ Lead → MAIL → 1000

↓

→ Common Server folder/location] Repository

MAVEN project →

folder Structure →



```
MavenProjectDemo  F:\Upgradation_Selenium\MavenProjectDe
  src
    main
      java
      resources
    test
      java
  pom.xml
External Libraries
Scratches and Consoles
```

main → JAVA code / classes

test → java → TestNG classes

project Object Model → pom.xml

→ project related info.

→ dependencies.

→ Build / plugins

```
<groupId>org.example</groupId>
   <artifactId>MavenProjectDemo</artifactId>
```

↓ Project Name

↓ packageName

pom.xml → project

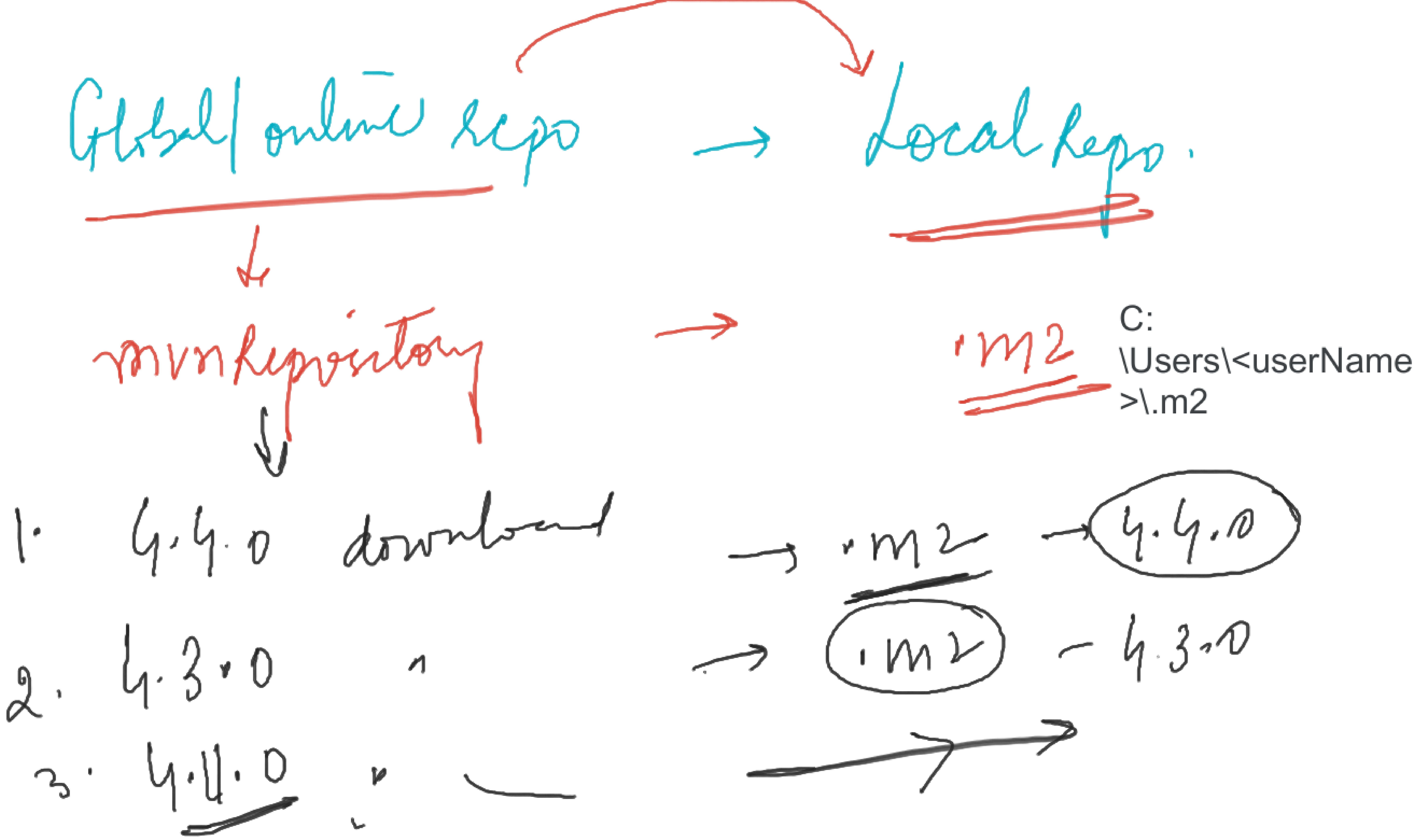
project

↳ project info.

↳ properties → compiler
 → skip

→ plugins

→ dependencies


Global/online repo
Local Repo.
mvnRepository
.m2
C:\Users\<userName>\.m2
1. 4.4.0 download
2. 4.3.0 "
3. 4.4.0 "
.m2
4.4.0
.m2
4.3.0

$$\begin{tikzcd}
\text{TestNG Classes} \arrow[r] \arrow[d] & \text{TestNG.xml} \arrow[dr] & & \\
\text{Methods} & & \text{hit/run} & \\
& & \checkmark\ \text{pom.xml} \arrow[r] & {} \\
& & \text{Maven} \arrow[u] &
\end{tikzcd}$$

Maven Surefire plugin -

https://maven.apache.org/surefire/maven-surefire-plugin/examples/testng.html

Using Suite XML Files

[plugin is used to hit/run the testng.xml through pom.xml]

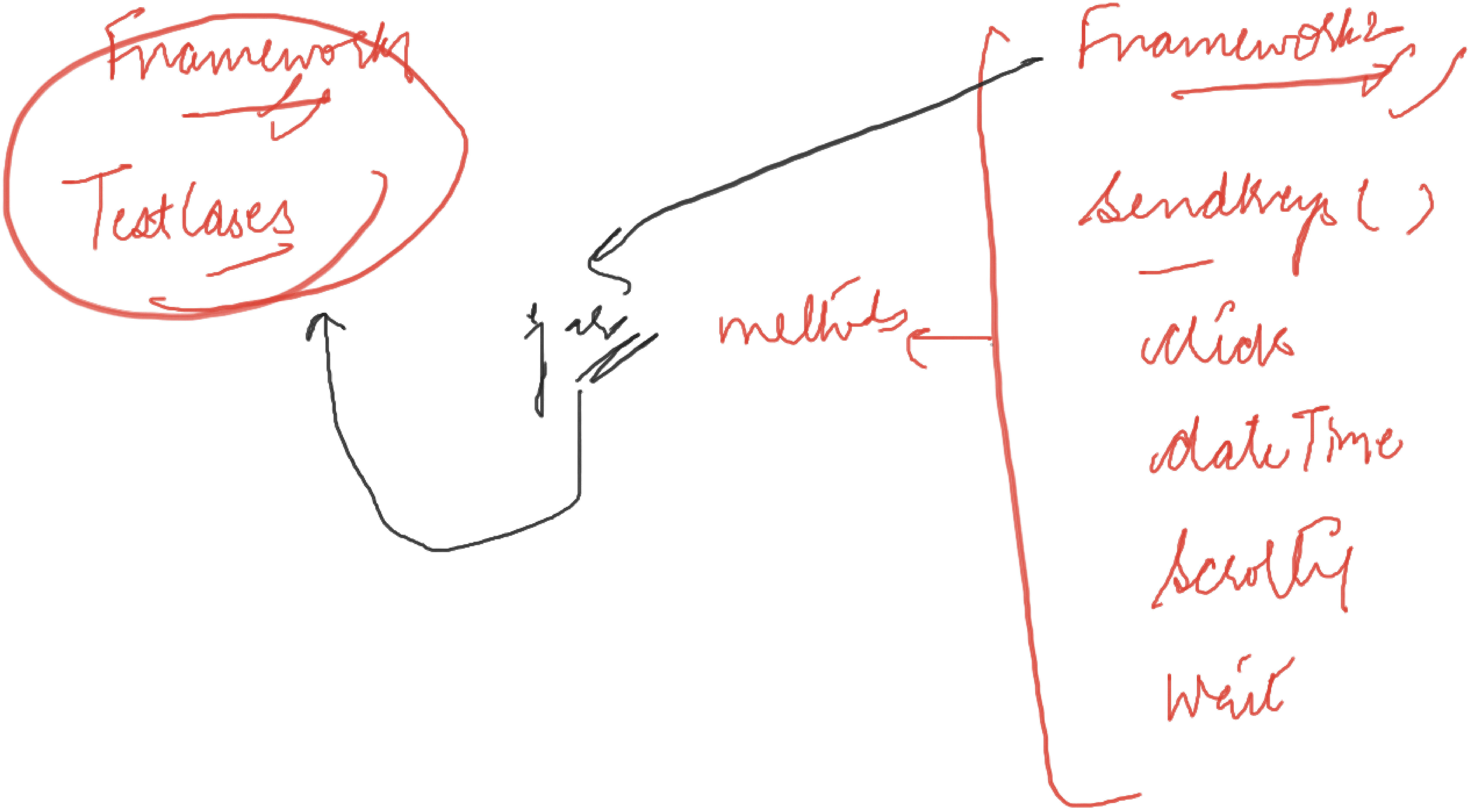
$$\begin{tikzcd}
\text{Project} \arrow[r] & \text{run} \arrow[r] \arrow[d] & \underline{\text{Build}} \arrow[dd] & \text{Generate} \;\; \textcircled{1} \arrow[d] \\
& \text{Classes} \arrow[r] & {} & \text{Pass} \;\; \textcircled{1} \\
& & \underline{\text{FAIL}} \;\; \textcircled{2} & \\
& & \text{Pass} \;\; \textcircled{3} &
\end{tikzcd}$$

(1) mvn clean → remove the Builds / target folder

(2) mvn test
↓
execute the testcases

(3) mvn install → it will generate the build .jar

Cmdline mvn → download
Env. variable
path set


Framework
Testcases
jar
methods
Framework2
SendKeys( )
click
dateTime
Scrolling
Wait

A Build Lifecycle is Made Up of Phases

Each of these build lifecycles is defined by a different list of build phases, wherein a build phase represents a stage in the lifecycle.

For example, the default lifecycle comprises of the following phases (for a complete list of the lifecycle phases, refer to the Lifecycle Reference):

validate - validate the project is correct and all necessary information is available
compile - compile the source code of the project
test - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
package - take the compiled code and package it in its distributable format, such as a JAR.
verify - run any checks on results of integration tests to ensure quality criteria are met
install - install the package into the local repository, for use as a dependency in other projects locally
deploy - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

Part 2: Maven project creation via cmd

(1) Maven Download (binary.zip)

https://maven.apache.org/download.cgi?Preferred=ftp://ftp.osuosl.org/pub/apache/

(2) Set the Env. variable

→ Edit path variable
→ Create New variable
→ set Maven path till bin folder
→ verify maven version — [mvn -version]

(4) Create Maven project

→ mvn archetype: generate

→ mvn -version

(3) → mvn clean

→ mvn test → mvn clean test

→ mvn install → mvn clean install