



TECHPROED

PROFESSIONAL TECHNOLOGY EDUCATION

Jenkins Global Configuration and Manage Plugins

How to set Global Configuration of Jenkins

How to find JAVA_HOME path on macOS

- 1: Open Terminal
- 2: Type **which java** then press enter. You should see **/usr/bin/java** on terminal
- 3: Type **\$(dirname \$(readlink \$(which javac)))/java_home** press enter.

```
[apple@apples-iMac-6 ~ % which Java
/usr/bin/Java
[apple@apples-iMac-6 ~ % $(dirname $(readlink $(which javac)))/java_home
/Library/Java/JavaVirtualMachines/jdk1.8.0_241.jdk/Contents/Home
apple@apples-iMac-6 ~ %
```

← **JAVA_HOME path**

How to find JAVA_HOME path on Windows

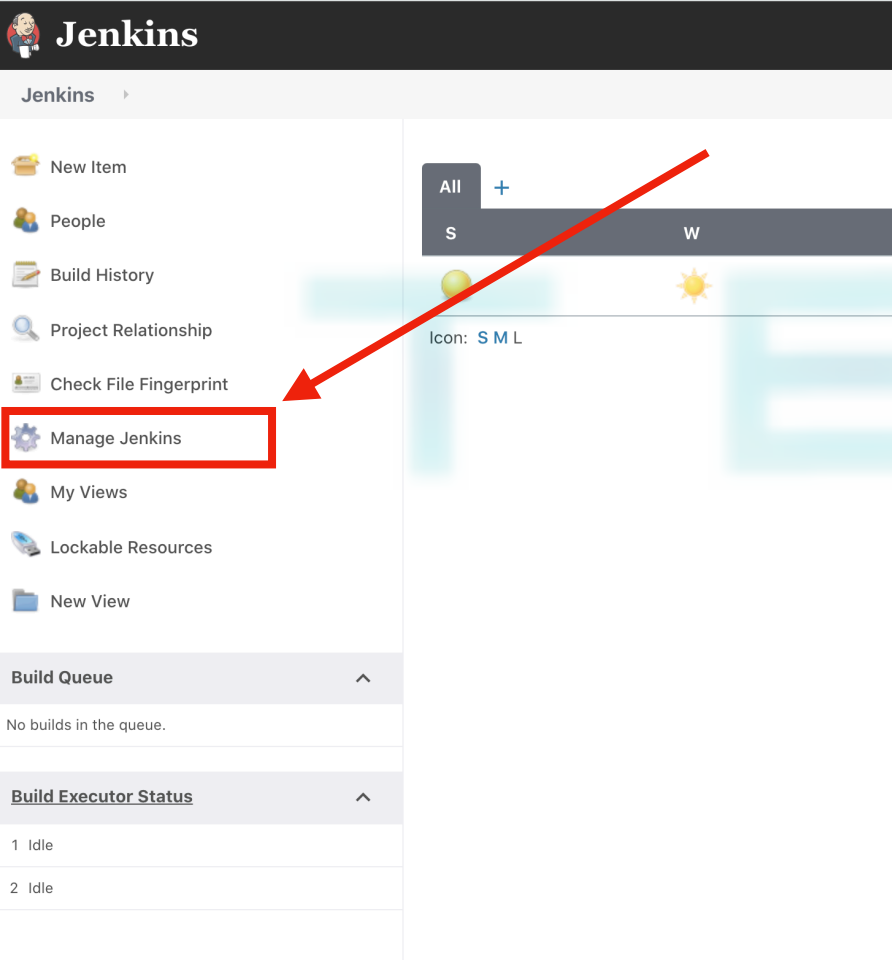
- 1: Open cmd
- 2: Type **where java** then press enter. It will show the location of the JAVA_HOME

```
C:\Users\sam>where java
C:\Program Files (x86)\Common Files\Oracle\Java\javapath\java.exe
C:\Program Files\Java\jdk-10.0.2\bin\java.exe
C:\Program Files\Java\jre-10.0.2\bin\java.exe
```

← **JAVA_HOME path**

After finding JAVA_HOME path

1: Open **Jenkins** and click on **Manage Jenkins**



2: Click on **Global Tool Configuration**

System Configuration

Configure System
Configure global settings and paths.

Global Tool Configuration
Configure tools, their locations and automatic installers.

Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
⚠ There are updates available

Manage Nodes and Clouds
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Security

Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.


Manage Credentials
Configure credentials

Configure Credential Providers
Configure the credential providers and types

Manage Users
Create/delete/modify users that can log in to this Jenkins

In-process Script Approval
Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.

3: Click on **JDK installations...**



Global Tool Configuration

Maven Configuration

Default settings provider

Use default maven settings

Default global settings provider

Use default maven global settings

JDK

JDK installations...

4: Type **JavaHome** as name, and paste the JAVA_HOME path you found in the first slide

JDK

JDK installations

Add JDK

JDK

Name

JavaHome

JAVA_HOME

/Library/Java/JavaVirtualMachines/jdk1.8.0_241.jdk/Contents/Home

☐ Install automatically

Add JDK

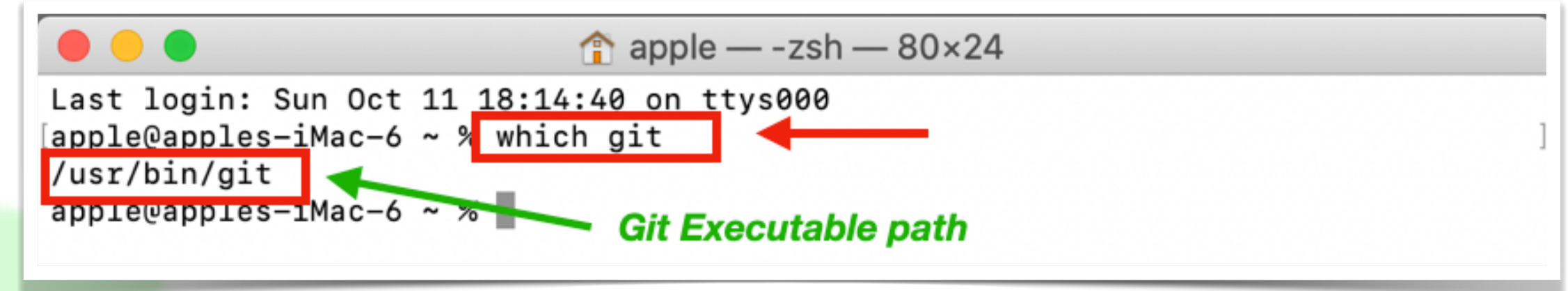
List of JDK installations on this system

Delete JDK

PROFESSIONAL TECHNOLOGY EDUCATION

How to find GIT EXECUTABLE path on macOS

- 1: Open Terminal
- 2: Type **which git** then press enter. You should see **/usr/bin/git** on terminal



```
apple — -zsh — 80x24
Last login: Sun Oct 11 18:14:40 on ttys000
apple@apples-iMac-6 ~ % which git
/usr/bin/git
```

A green arrow points from the text "Git Executable path" to the output "/usr/bin/git".

How to find GIT EXECUTABLE path on Windows

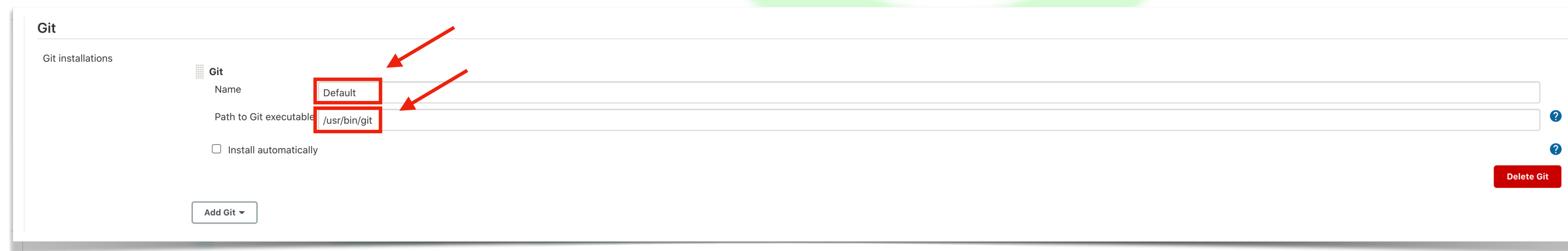
- 1: Open cmd
- 2: Type **where git** then press enter. It will show the location of the *Git Executable path*



```
C:\Users\sam>where git
C:\Program Files\Git\cmd\git.exe
C:\Program Files\Git\mingw64\bin\git.exe
```

A green arrow points from the text "Git Executable path" to the output "C:\Program Files\Git\mingw64\bin\git.exe".

- 5: Keep the name Default, and paste the Git Executable path you found above



The screenshot shows the "Git" settings window. The "Name" field is set to "Default" and the "Path to Git executable" field is set to "/usr/bin/git". Both fields are highlighted with red boxes and red arrows. There is an "Add Git" button at the bottom left and a "Delete Git" button at the bottom right.

6: Type **MAVEN_HOME** as name, **select checkbox** for install automatically and from the dropdown select version **3.6.3**

Maven

Maven installations

Add Maven

Maven

Name

MAVEN_HOME

☒ Install automatically

Install from Apache

Version

3.6.3

Add Installer

Add Maven

List of Maven installations on this system

Delete Installer

Delete Maven

7: Click on **Apply** and **Save**

Save

Apply

Jenkins can Install Maven Automatically

1:

Dashboard → Global Tool Configuration

Gradle

Gradle installations

Add Gradle

List of Gradle installations on this system

Ant

Ant installations

Add Ant

List of Ant installations on this system

Maven

Maven installations

Add Maven

Maven

Name

maven 3.6.3

☒ Install automatically

Install from Apache

Version

3.6.3

Add Installer

Delete Installer

Delete Maven

Add Maven

List of Maven installations on this system

To Build Job in Jenkins after Clicking on “Build Now”

1:

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description

[Plain text] Preview

☐ Discard old builds

☒ GitHub project

Project url

https://github.com/salptekin/SeleniumGrid_Jenkins.git/

Advanced...

2:

Build

Invoke top-level Maven targets

Maven Version

maven 3.6.3

Goals

clean install

Advanced...

Add build step

3:

General Source Code Management Build Triggers Build Environment Build Post-build Actions

None

Git

Repositories

Repository URL

https://github.com/salptekin/SeleniumGrid_Jenkins.git

Credentials

- none - Add

Advanced...

Add Repository

Branches to build

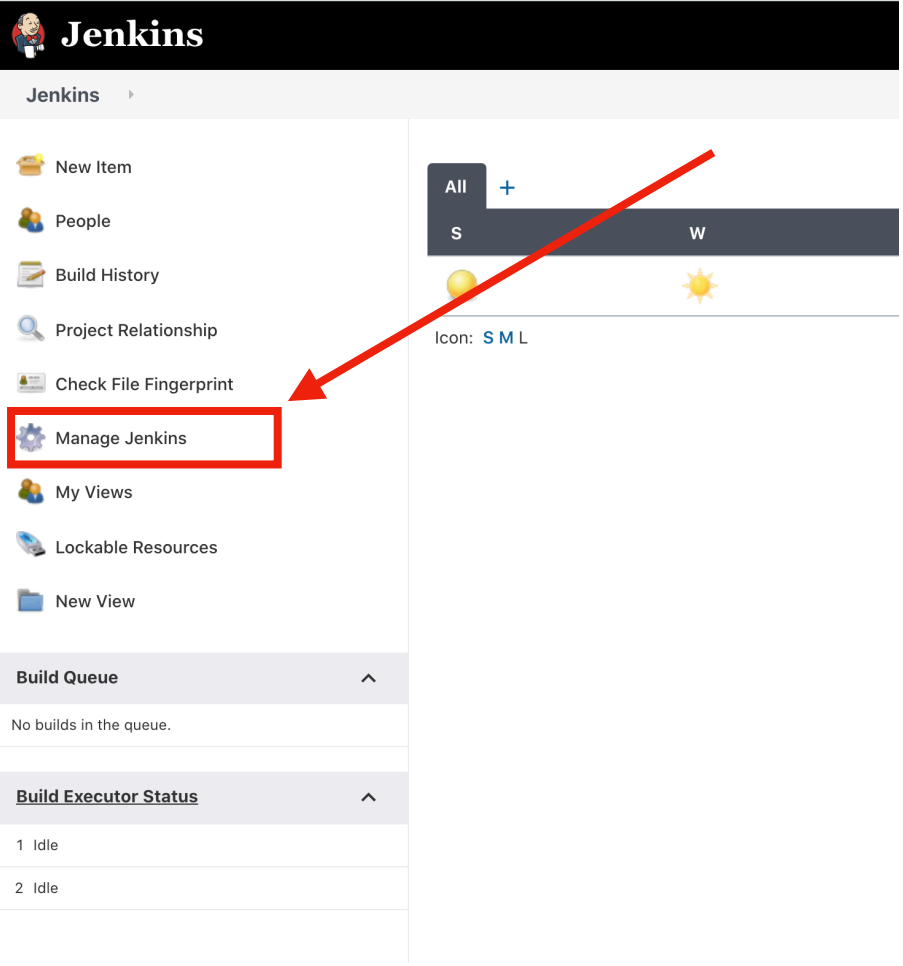
Branch Specifier (blank for 'any')

*/main

Add Branch

How to Manage Plugins in Jenkins

1: Open *Jenkins* and click on *Manage Jenkins*



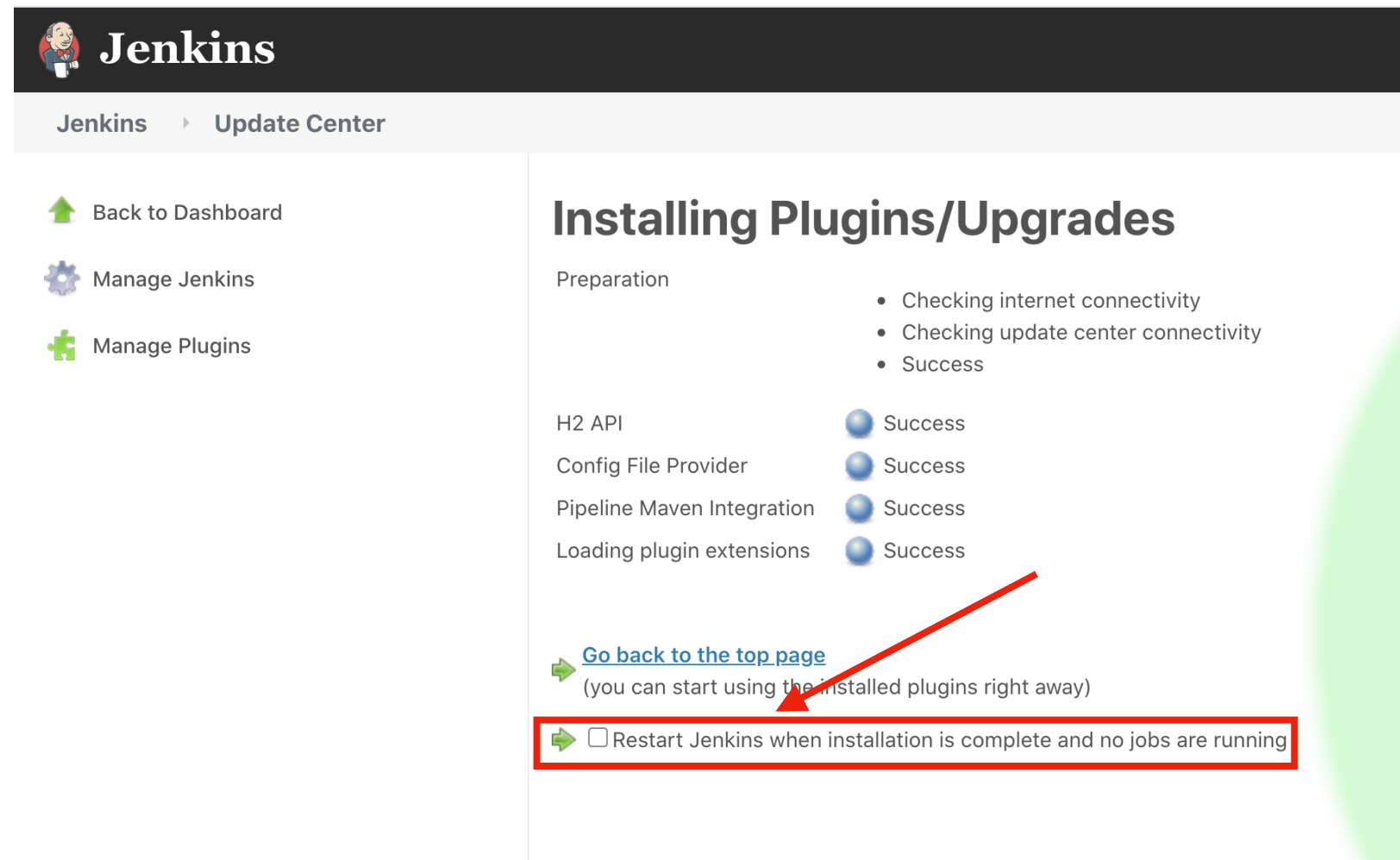
2: Click on *Manage Plugins*



3: Select *Available Tab*, in the search box type *Maven* then find *Maven Integration* in the list, *select* the checkbox, then click *Install without restart*



4: Wait until everything is installed then select the ***Restart Jenkins when installation is complete and no jobs are running***



Note: Repeat the steps 1, 2, 3, 4 to install the following plugins

- a) Git Plugin
- b) TestNg plugin
- c) Apache HttpComponents Client 4.x API
- d) Javadoc
- e) JSch Dependency
- f) JUnit
- g) Mailer
- h) Token Macro
- i) Bouncy castle API
- j) Command Agent Launcher
- k) JDK Tool

Note: If some plugins are not displayed under ***available tab*** no worries skip them.

How to send the codes to GitHub Repository

1)

A

Local Master Branch

A

B

C

Remote Master Branch

2)

A

B

C

Local MAIN Branch

Pull remote master to your local master

A

B

C

Remote Master Branch

3)

A

B

C

Local MAIN Branch

Create a new branch from your local master branch

A

B

C

Local Branch

A

B

C

Remote MAIN Branch

4)

A

B

C

Local Master Branch

A

B

C

F

G

H

Remote Master Branch

Work on your local branch do all tests

A

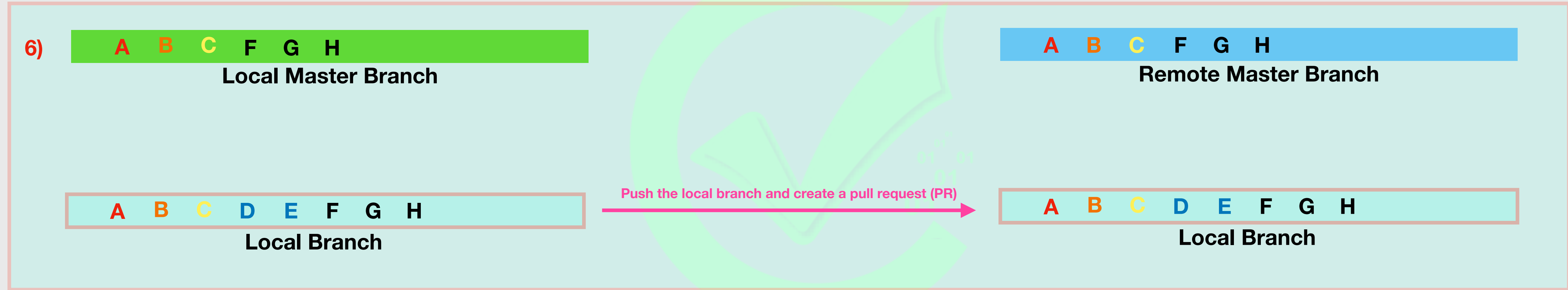
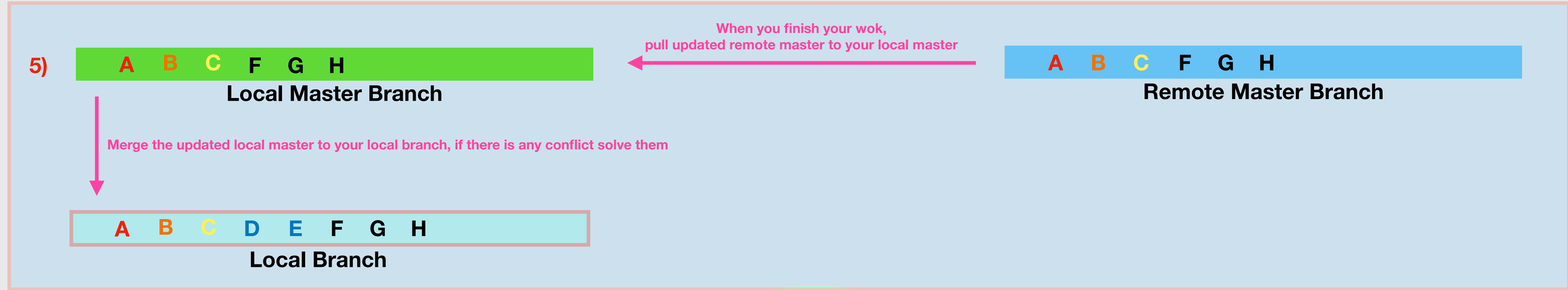
B

C

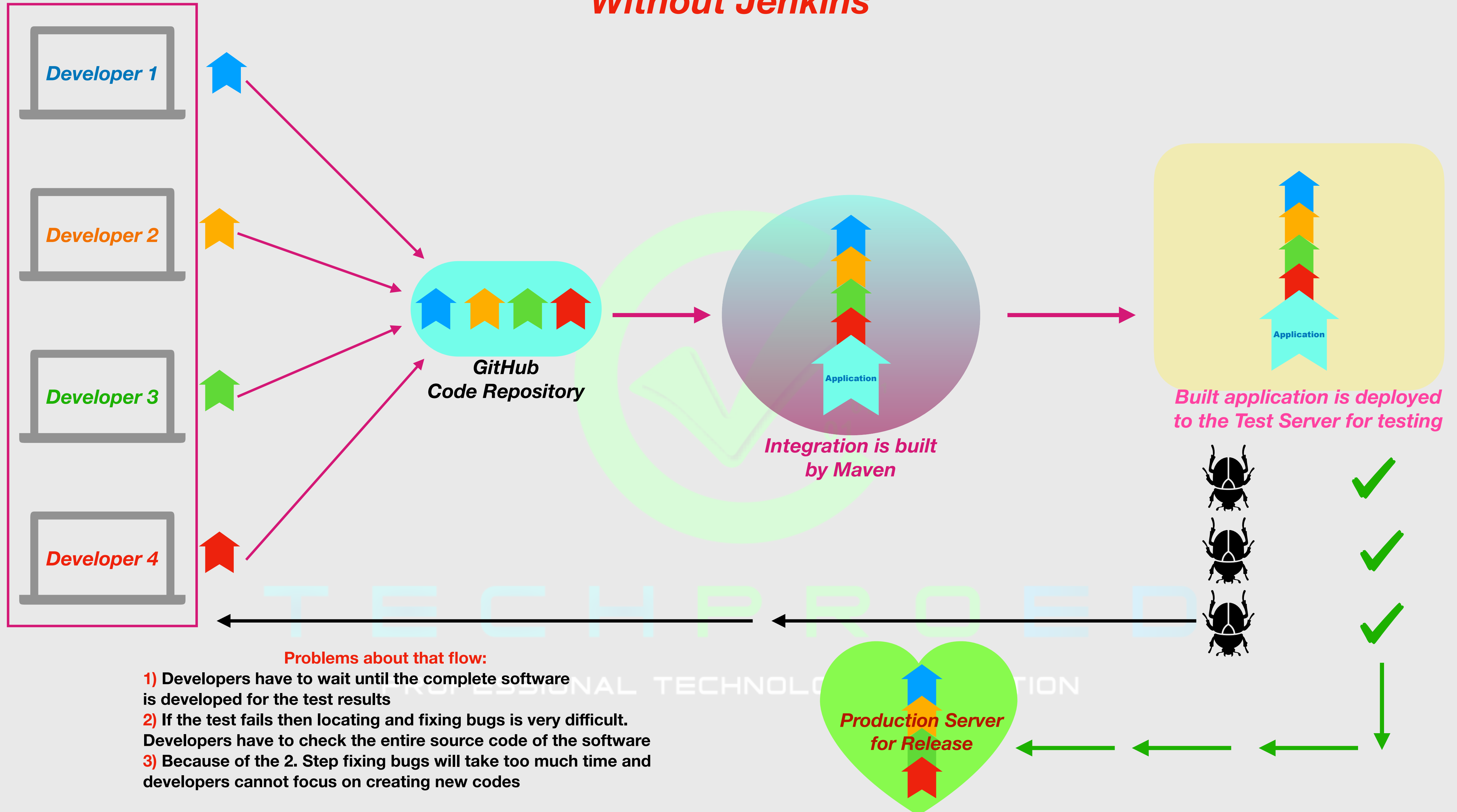
D

E

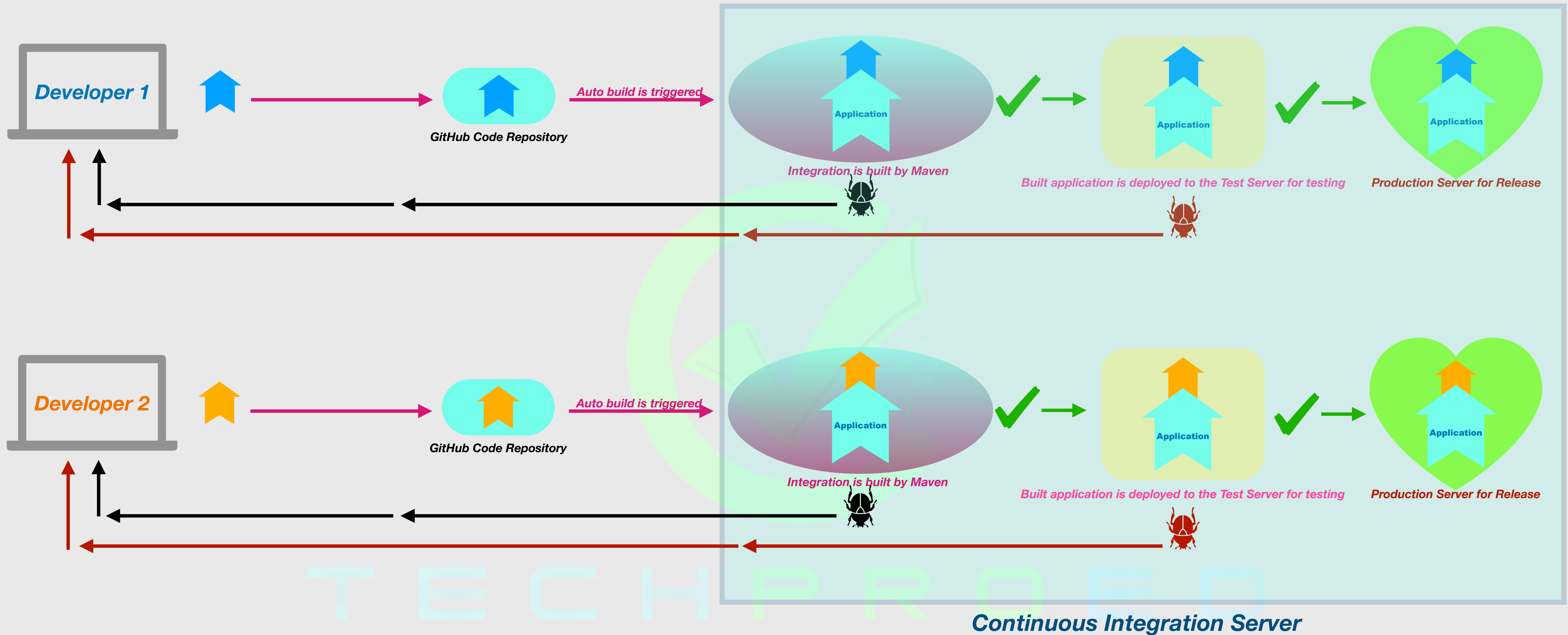
Local Branch



Without Jenkins



With Jenkins



Benefits of that flow:

- 1) If there is a bug in the code developers will be informed immediately
- 2) If there is bug in the code developers check the last commit
- 3) It increases the frequency of new releases