



## Guía de investigación y prácticas No 2 –

### Parte I-35%

Temática: Gestión de versiones

Objetivo: Ampliar los contenidos teóricos planteados y así mejorar sus competencias y habilidades en cuanto a gestión de versiones

#### Objetivos-

- Investigar en qué consisten los ítems de configuración de software.
- Identificar las ventajas y desventajas del software de configuración así sus elementos para mejorar la capacidad, controlar alcance y los cambios en un proyecto de software.
- Instalación de Git como herramienta de control de versiones.
- Investigar acerca de los comandos básicos del uso de Git.
- Usar GitHub como plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git.

Indicaciones: A continuación, se presentan 4 partes correspondientes a Gestión de Versiones.

Deberá seguir las indicaciones de entrega y entregar un documento PDF EXCLUSIVAMENTE con la solución de:

PARTE I Ítems de configuración: Completa (10%)

PARTE II Entender que es Git y GitHub: Completa (20%)

PARTE III Instalación y configuración de Git en Windows: Captura de pantalla en el numeral 18 (20%)

PARTE IV Creando repositorio en GitHub y haciendo push de mi primer proyecto: 1 captura del numeral 13 y 2 capturas de pantalla del numeral 16 (50%)

Siga las indicaciones y evite entregar más de lo que se le pide, las partes y numerales a entregar están resaltadas en verde.

## **PARTE I: ÍTEMES DE CONFIGURACIÓN.**

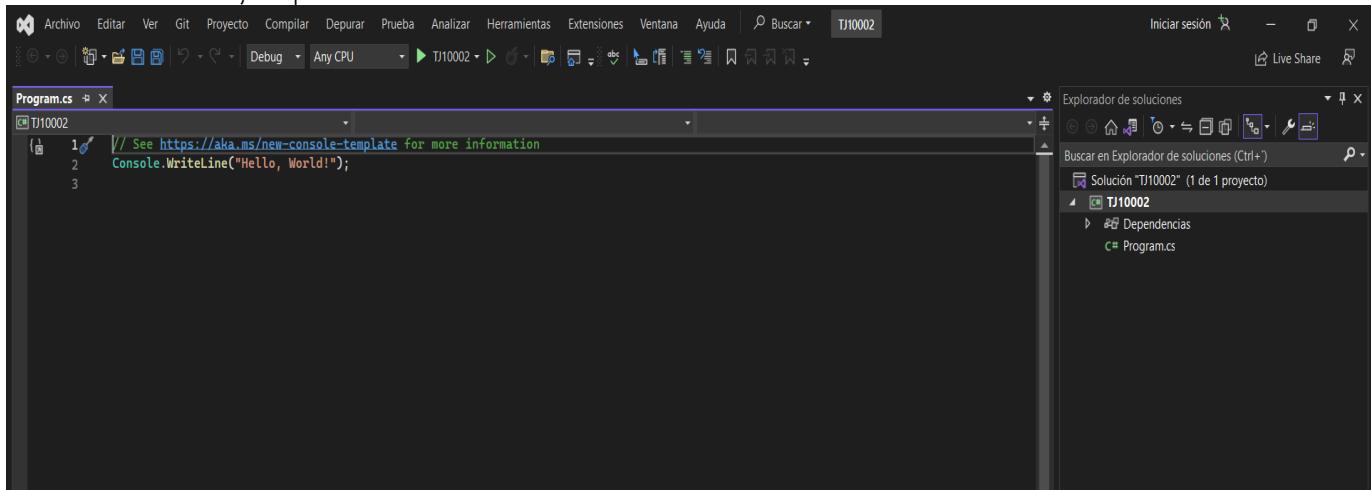
1. ¿Qué es un ítem de configuración?
2. Ejemplos de ítems de configuración son:
3. ¿Qué es un software de configuración?
4. ¿Cuáles son las ventajas y desventajas del software de configuración?
5. ¿Qué es un sistema de configuración?
6. Explique cuáles son los elementos de configuración de software
7. De ejemplos de sistemas de gestión de la configuración y explique al menos 2

## **PARTE II: Entender qué es Git y GitHub.**

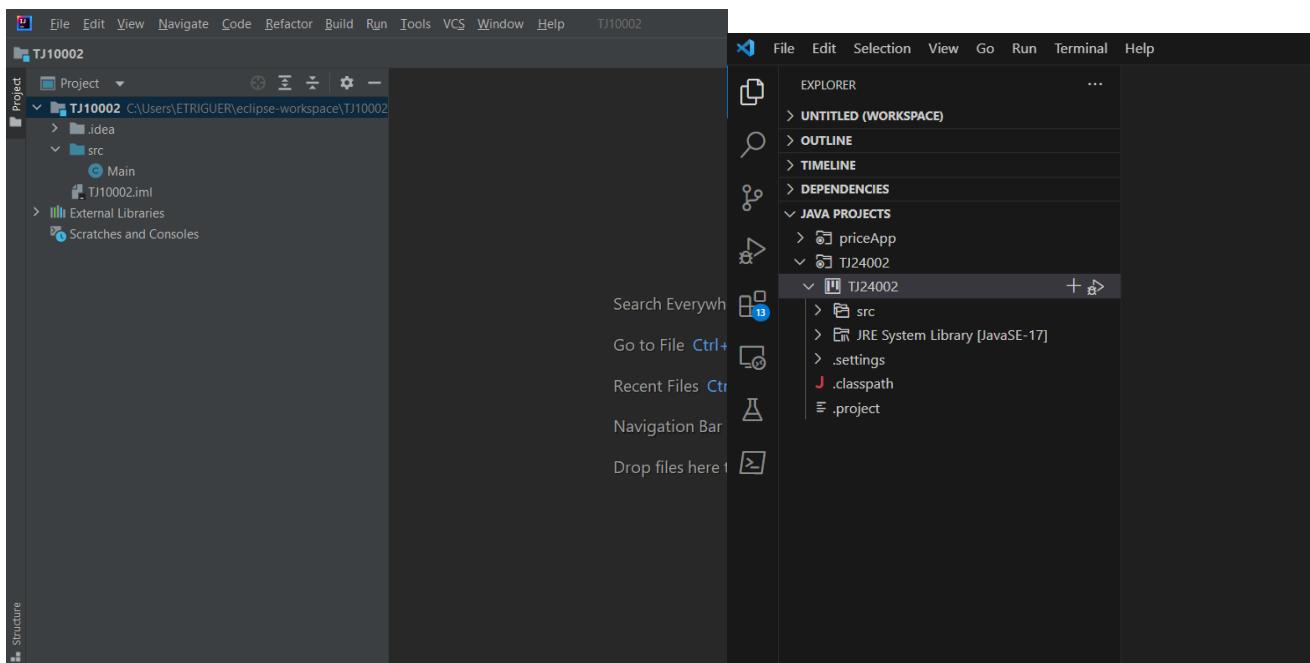
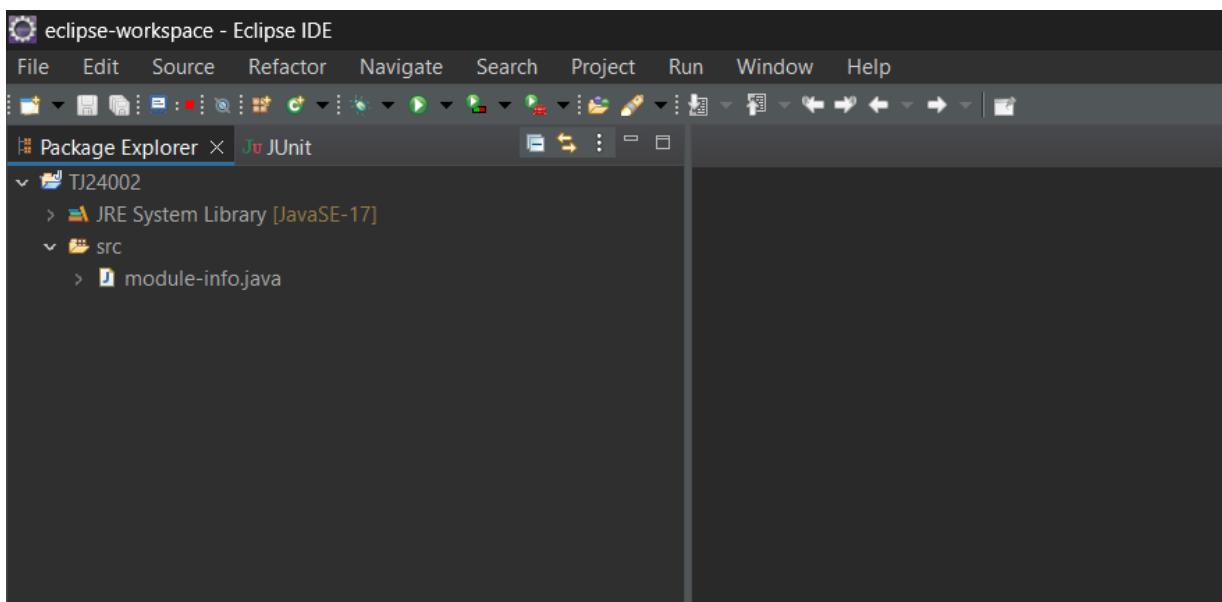
De respuesta a los siguientes enunciados:

1. ¿Qué es Git?
2. Liste algunos de los comandos básicos de Git y su respectiva descripción.
3. ¿Cuál es la diferencia entre un commit y un push?
4. Investigue los siguientes términos.
  - ¿Qué es GitHub y cuál es su función?
  - ¿Qué es un repositorio en GitHub y cuál es su función?
  - ¿Qué es un branch en GitHub y cuál es su función?
  - ¿Qué es clonar un repositorio en GitHub?
5. Visitar y leer el contenido de las siguiente URL's:  
[Hola mundo - Documentación de GitHub](#)  
[Configuración de Git - Documentación de GitHub](#)  
[Creación de un repositorio - Documentación de GitHub](#)
6. Cree un proyecto sencillo en su IDE y lenguaje de preferencia, **el proyecto debe contener su carnet estudiantil (Entregar captura de pantalla).**

Se muestran 4 ejemplos:



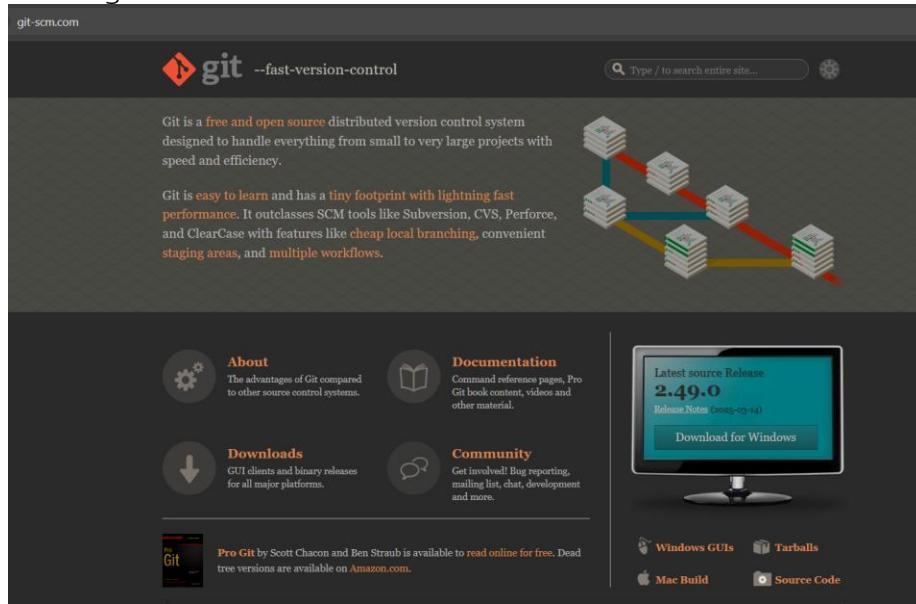
```
1 // See https://aka.ms/new-console-template for more information
2 Console.WriteLine("Hello, World!");
3
```



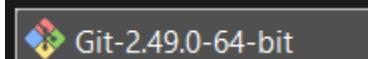
## PARTE III: Instalación y configuración de Git en Windows.

Para instalar Git en Windows necesitará descargar el instalador desde el sitio web de Git:

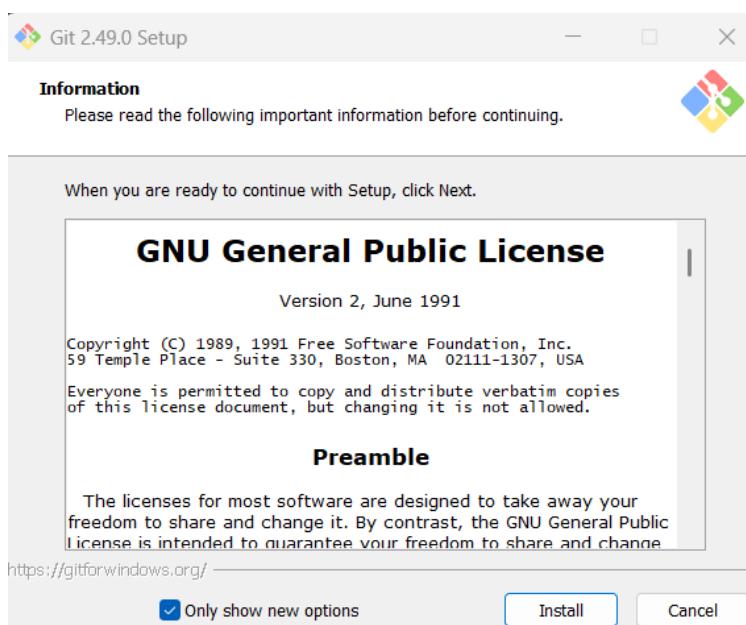
1. En el sitio web de Git seleccione Windows: la versión más reciente de git comenzará a descargarse automáticamente.



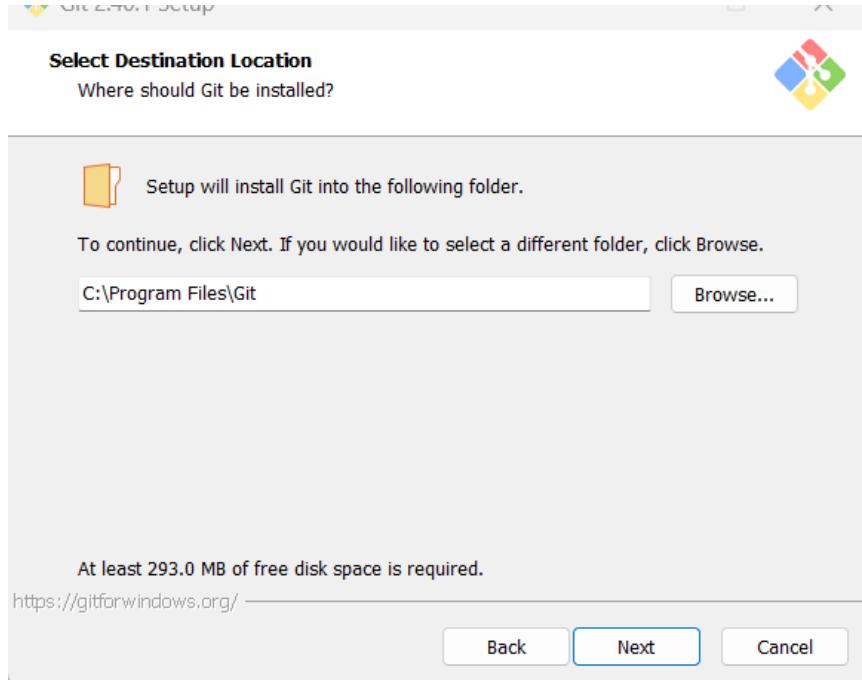
2. Abra el paquete descargado y seleccione Ejecutar:



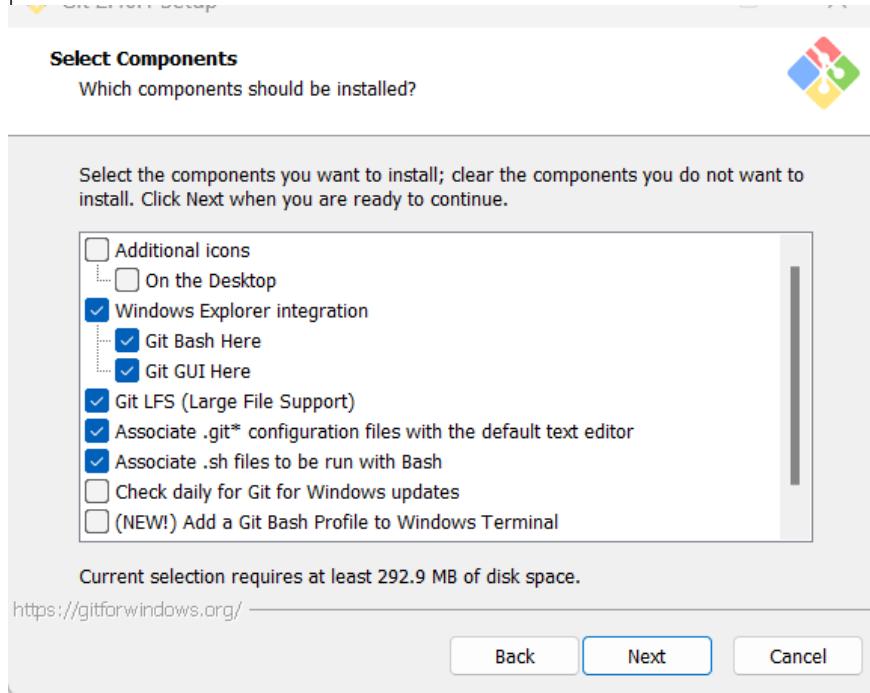
3. Haga clic en Next para continuar:



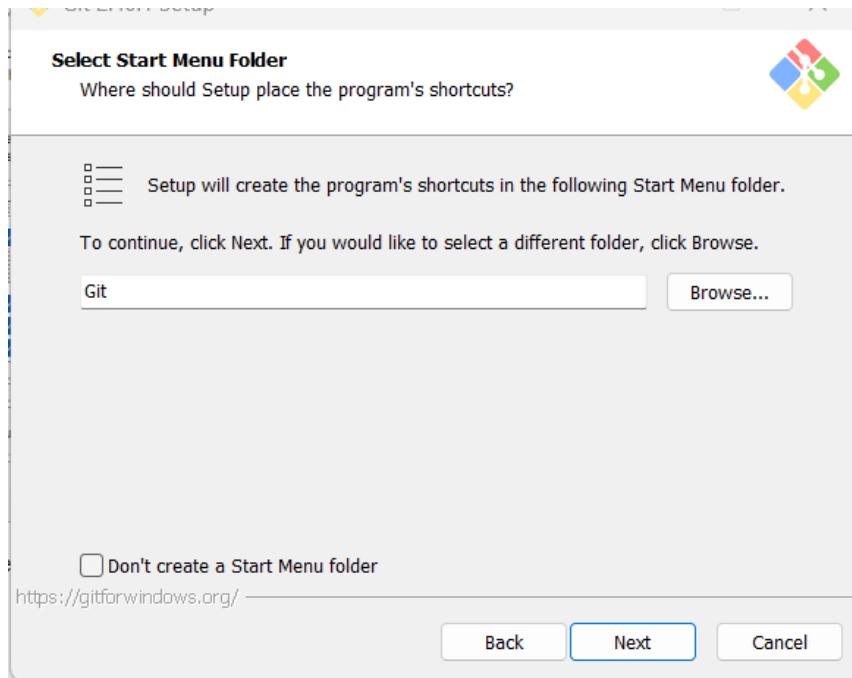
4. Si desea aceptar la ubicación de instalación predeterminada puede hacer clic en Next. Si necesita cambiar la carpeta de instalación haga clic en el botón Browse y seleccione una nueva ubicación:



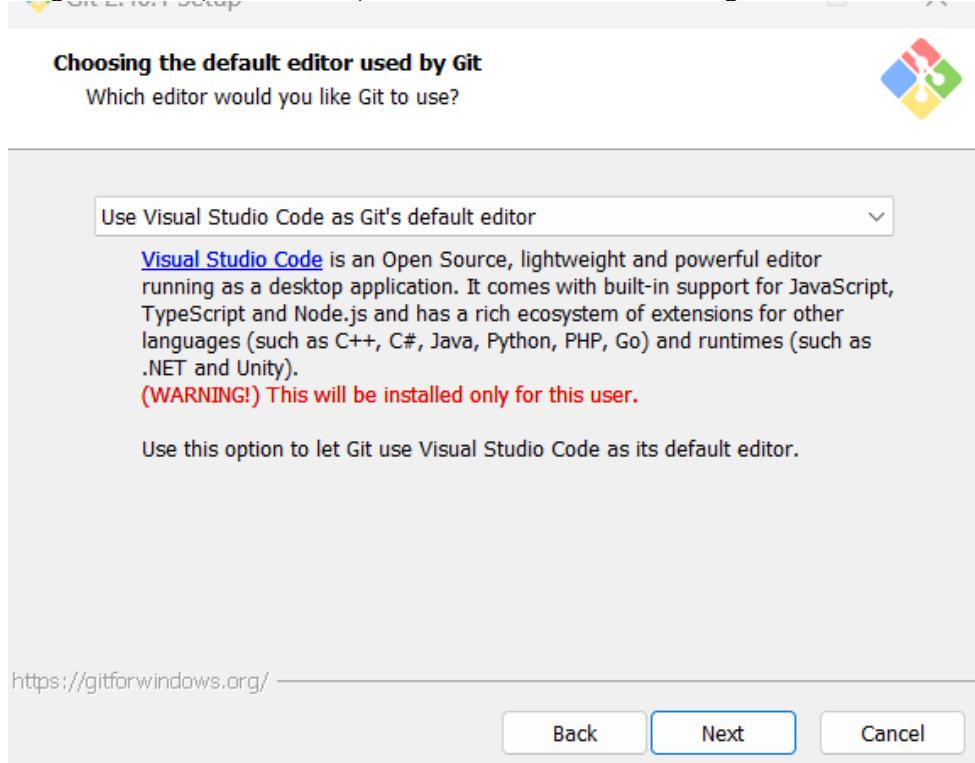
5. Puede continuar con la configuración de componentes predeterminados haciendo clic en Next. Si usted es un usuario avanzado y desea cambiar estos datos puede hacerlo y luego pulsar Next.



6. Acepte la carpeta predeterminada para el Menú de Inicio haciendo clic en Next o utilice Browse para cambiarla:



7. Elegir el editor por defecto para editar archivos de configuración de Git.



8. Elegir el branch por defecto al clonar un repositorio o que Git decida.

**Adjusting the name of the initial branch in new repositories**

What would you like Git to name the initial branch after "git init"?



**Let Git decide**

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project [intends](#) to change this default to a more inclusive name in the near future.

**Override the default branch name for new repositories**

**NEW!** Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:

main

This setting does not affect existing repositories.

<https://gitforwindows.org/>

Back Next Cancel

9. Elegir la opción recomendada para trabajar con Git desde línea de comandos.

**Adjusting your PATH environment**

How would you like to use Git from the command line?



**Use Git from Git Bash only**

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

**Git from the command line and also from 3rd-party software**

**(Recommended)** This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.  
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

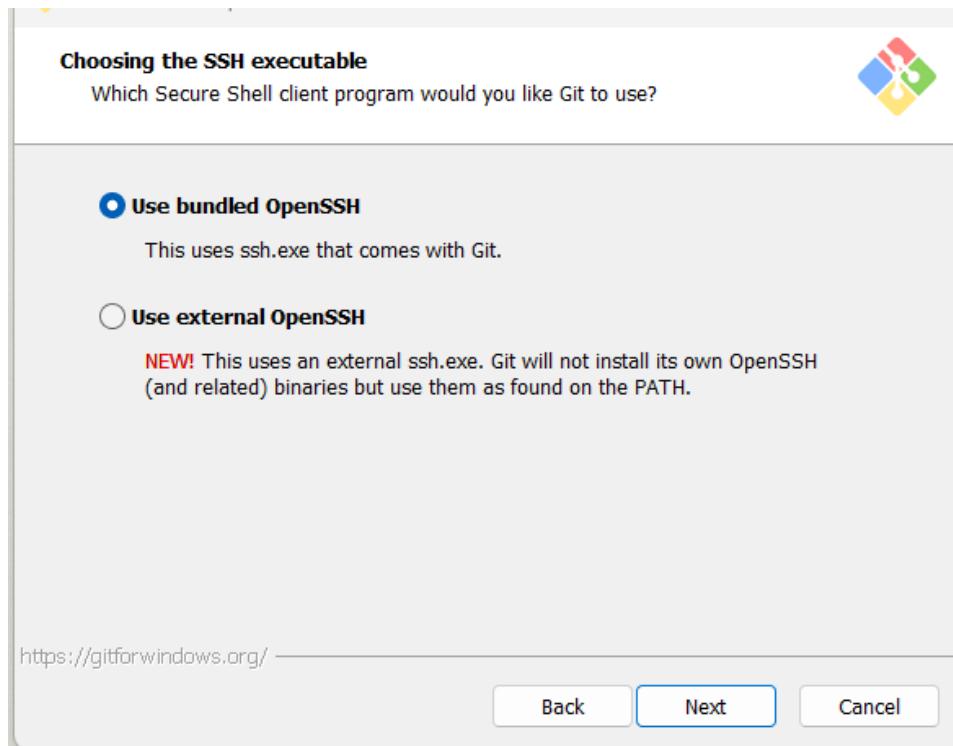
**Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.  
**Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.**

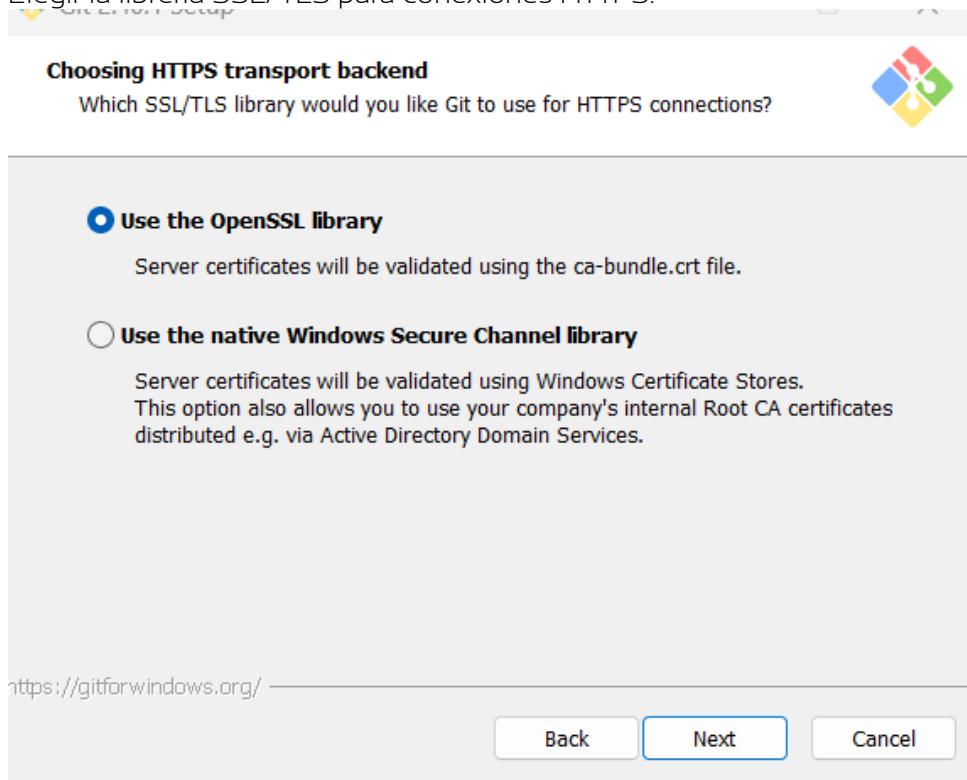
<https://gitforwindows.org/>

Back Next Cancel

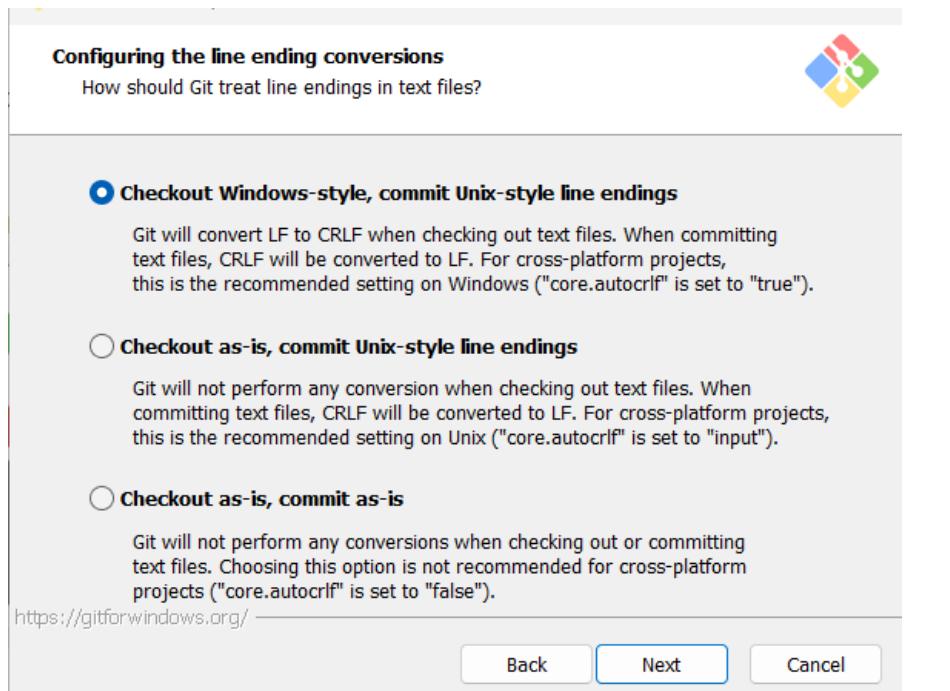
10. Elegir el Secure Shell Cliente a usar con Git.



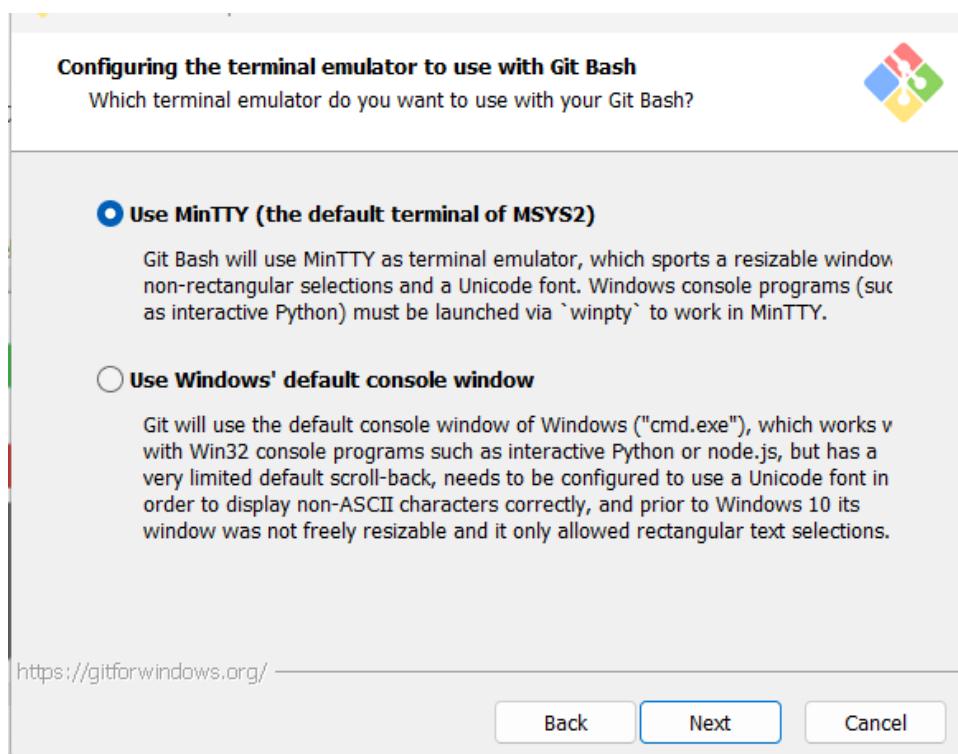
11. Elegir la librería SSL/TLS para conexiones HTTPS.



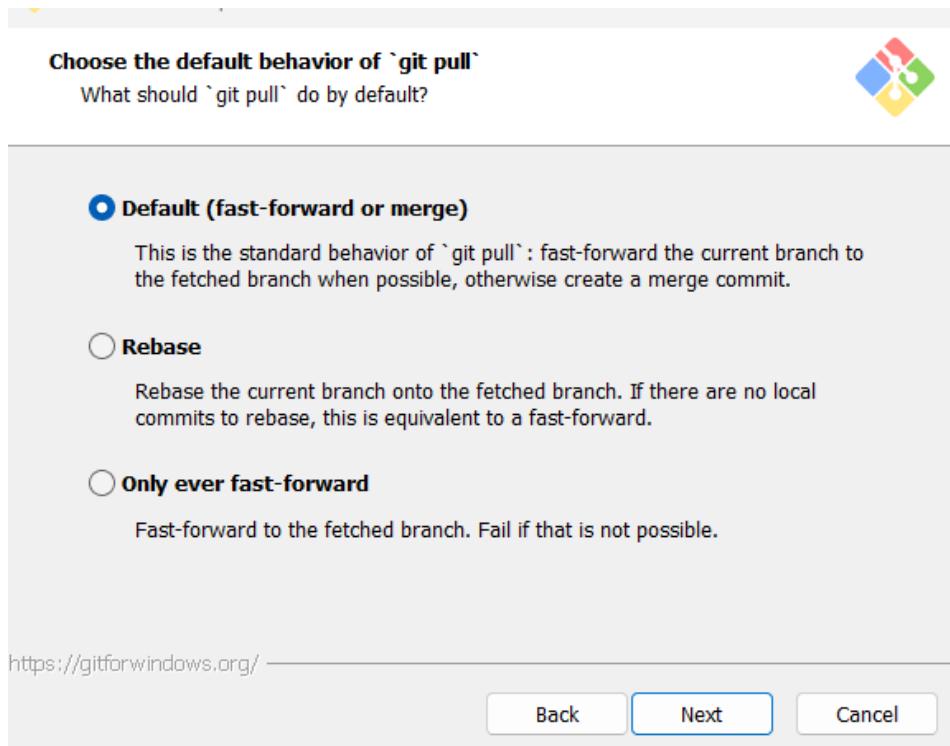
12. Dejar la opción por defecto y NEXT.



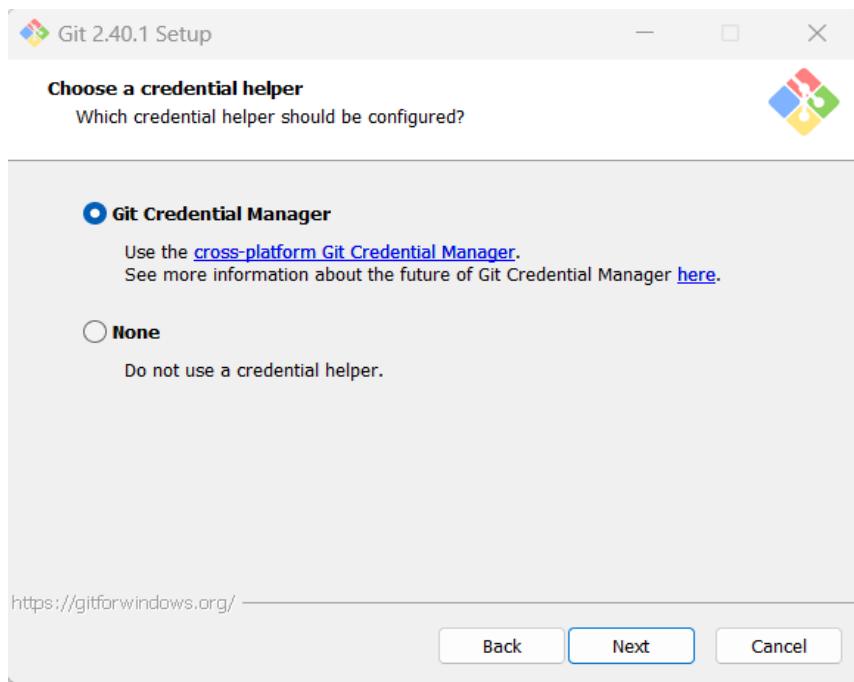
13. En la siguiente ventana puede hacer clic en Next para instalar el emulador MinTTY para el bash de git (opción predeterminada) o seleccionar la otra opción para usar el símbolo de sistema de Windows.

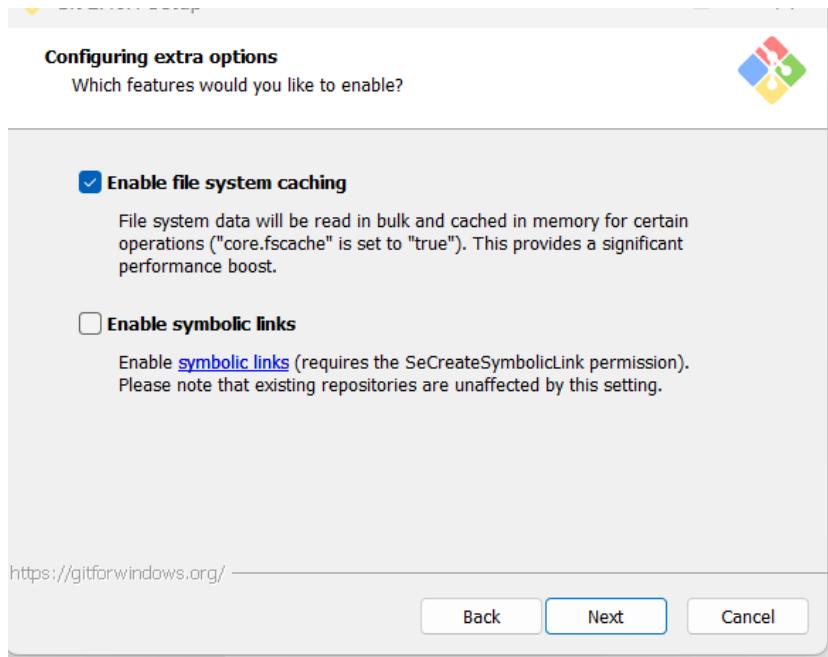


14. Elegir opción por defecto para un git pull.

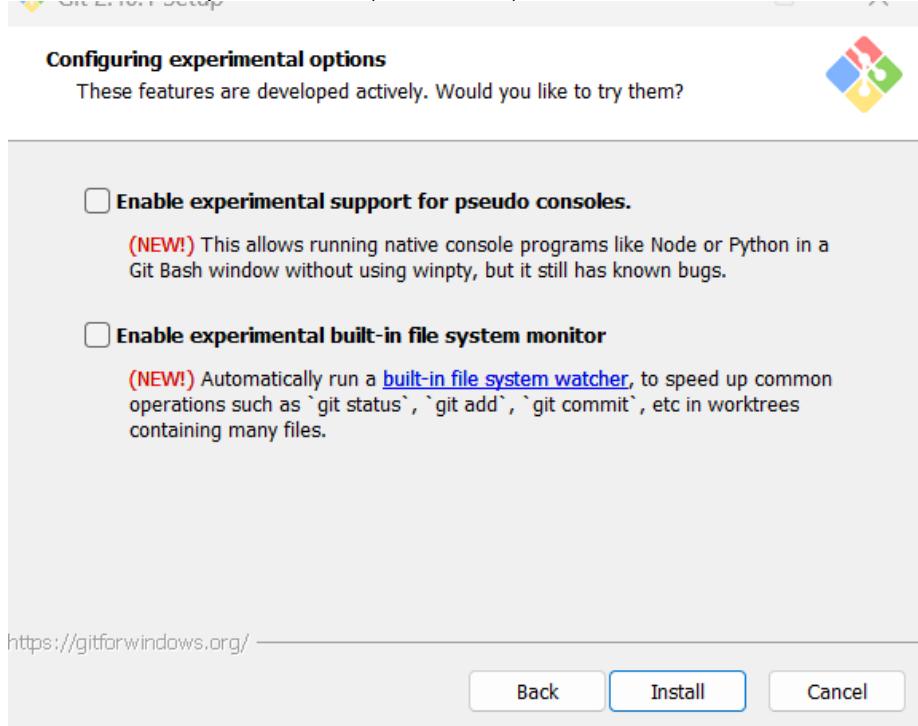


15. En los siguientes pasos se recomienda hacer clic en Next para seguir con las opciones predeterminadas.





16. Clic en instalar y omitir las opciones experimentales.



17. Ahora tiene Git instalado. Clic en Finish.

## Completing the Git Setup Wizard

Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

- Launch Git Bash
- View Release Notes



Finish

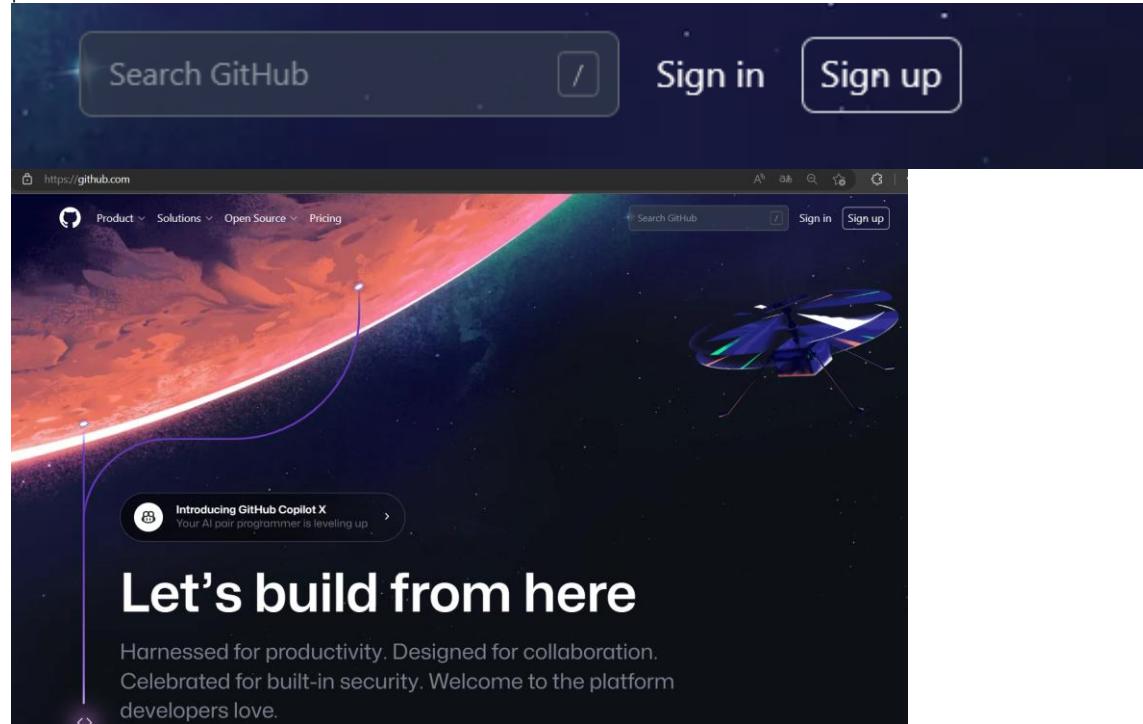
18. Verificar versión de Git con comando git -- version (Entregar captura de pantalla donde se vea lo indicado en círculos verdes de la imagen)  
-Versión Git Instalada  
- Icónico de Git en la barra de tareas  
- Fecha y hora del sistema)

```
MINGW64:/c/Users/ETRIGUER/Downloads
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Downloads
$ git --version
git version 2.49.0.windows.1

ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Downloads
$
```

## Parte IV: Creando repositorio en GitHub y haciendo push de mi primer proyecto.

1. Entrar a [GitHub: Let's build from here · GitHub](https://github.com) y hacer un sign up con su correo de preferencia.



Deberá ingresar la siguiente información.

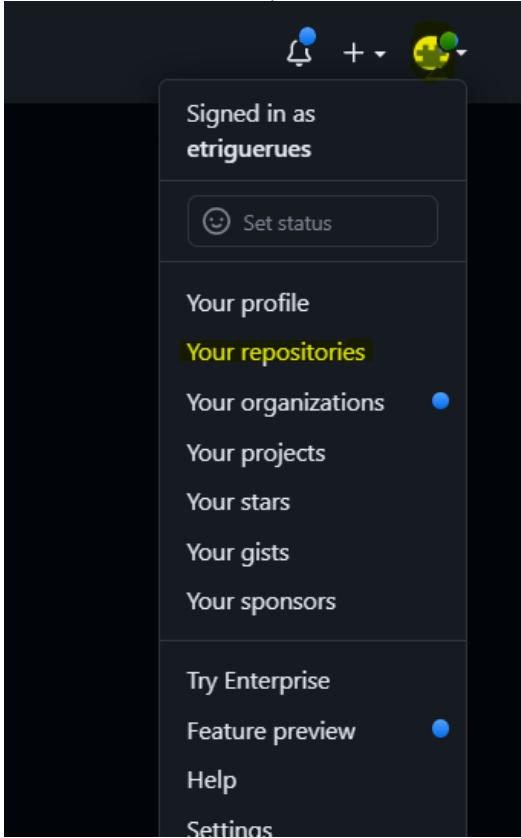
A screenshot of the GitHub sign-up form. The form fields are as follows:

- Welcome to GitHub!**  
Let's begin the adventure
- Enter your email\***  
✓ test@ues.edu.sv
- Create a password\***  
✓ ..... (redacted)
- Enter a username\***  
→ testuesocc

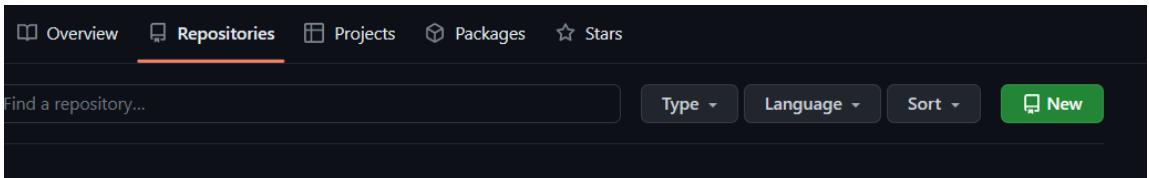
A "Continue" button is located to the right of the "Enter a username" field. Below the form, a message states "testuesocc is available.".

Le debería caer un código de confirmación a su correo luego de continuar.

2. Luego de hacer el registro exitosamente, se abre nuestro perfil. Crearemos un nuevo repositorio. Para ello seleccionamos nuestro perfil en la esquina superior derecha y nos vamos a nuestros repositorios.



3. Clic en New.



4. Escribiremos el nombre de nuestro repositorio **de preferencia que contenga su carnet estudiantil**, una pequeña descripción y elegir si será público o privado, en mi caso elijo un repositorio privado, pero para efectos de la guía elegir repositorio Público por si desea validar las aprobaciones de reglas de protección de rama en la parte III, de lo contrario no serán aplicadas (en la guía no se le piden realizar aprobaciones). Como opción podemos elegir agregar un archivo README que generalmente se escriben detalles de nuestro proyecto.

**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

**Owner \*** etriguerues / **Repository name \*** HelloWorld\_IIS  
 etriguerues /  HelloWorld\_IIS is available.

Great repository names are short and memorable. Need inspiration? How about [expert-lamp](#)?

**Description (optional)**  
 Cloning my first repository.

**Public**  
 Anyone on the internet can see this repository. You choose who can commit.

**Private**  
 You choose who can see and commit to this repository.

**Initialize this repository with:**  
 **Add a README file**  
 This is where you can write a long description for your project. [Learn more about READMEs](#).

**Add .gitignore**  
 .gitignore template: **None**

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

**Choose a license**  
 License: **None**

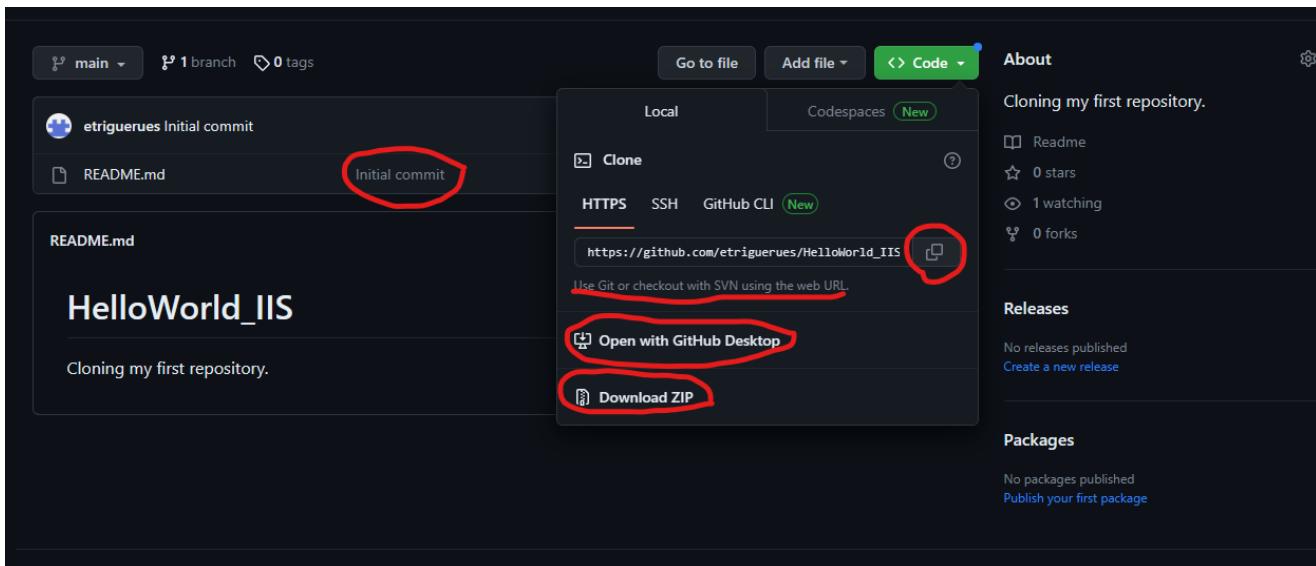
A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set **main** as the default branch. Change the default name in your [settings](#).

ⓘ You are creating a private repository in your personal account.

**Create repository**

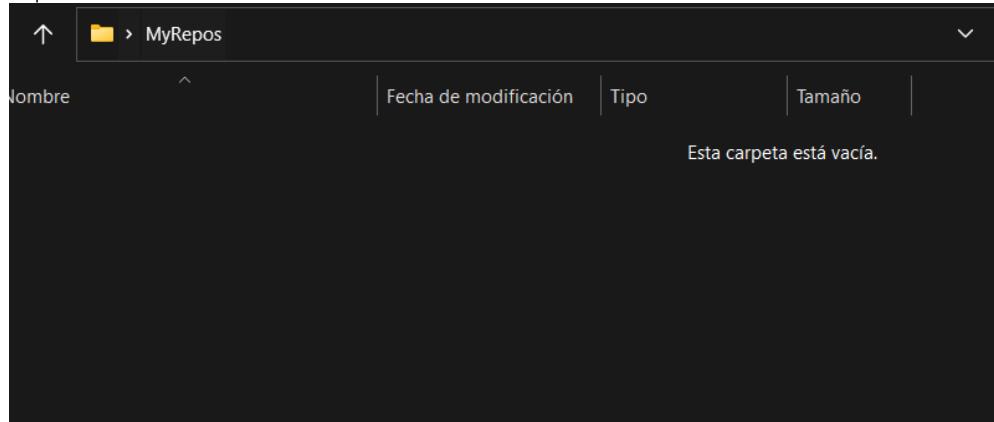
5. Ahora vamos a clonar nuestro repositorio.



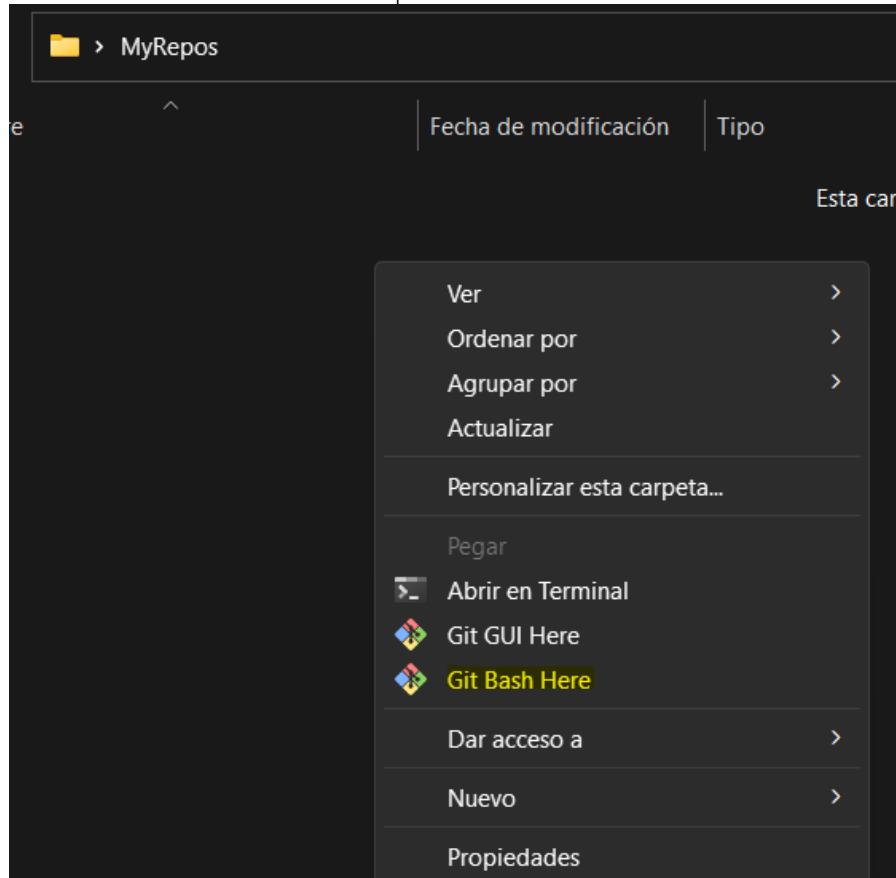
Como vemos al hacer un commit al momento de crear nuestro repositorio, se nos agregó un archivo README, nuestro repositorio ya tiene contenido y está alojado en nuestra rama main. Podemos clonar el repositorio usando la URL, abriéndolo desde Github Desktop, o podemos descargar un archivo ZIP de nuestro repositorio y montarlo localmente en el IDE de nuestra preferencia.

Vamos a elegir la clonación por URL, para ello copiamos la URL del repositorio.

- Una vez copiada la URL creamos un directorio donde vamos a alojar nuestro repositorio.



- Abrimos un Git Bash en la carpeta creada.

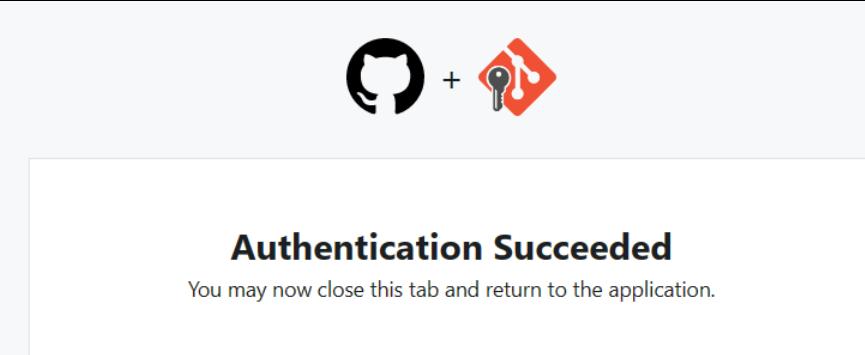


Escribimos git clone y pegamos la url de nuestro repositorio creado.

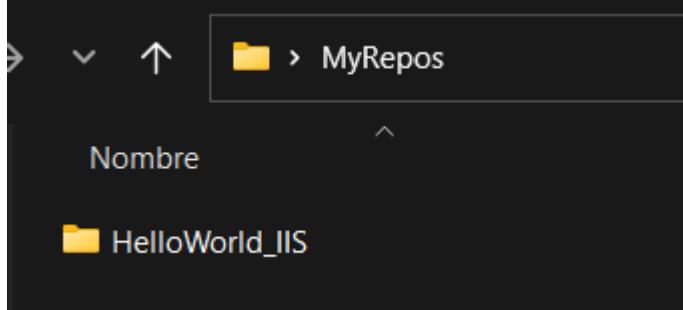
```
MINGW64:/c/Users/ETRIGUER/Desktop/MyRepos
$ git clone https://github.com/etriguerues/HelloWorld_IIS.git
```

Al dar enter nos pedirá autenticación de git credential (si lo activamos al instalar), de lo contrario se va clonar nuestro repositorio.

```
MINGW64:/c/Users/ETRIGUER/Desktop/MyRepos  
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos  
$ git clone https://github.com/etriguerues/Helloworld_IIS.git  
Cloning into 'Helloworld_IIS'...  
remote: Enumerating objects: 3, done.  
remote: Counting objects: 100% (3/3), done.  
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
Receiving objects: 100% (3/3), done.  
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos  
$
```



8. Listo, tenemos nuestro repositorio creado.



9. Ahora vamos a ingresar al directorio de nuestro repositorio.

```

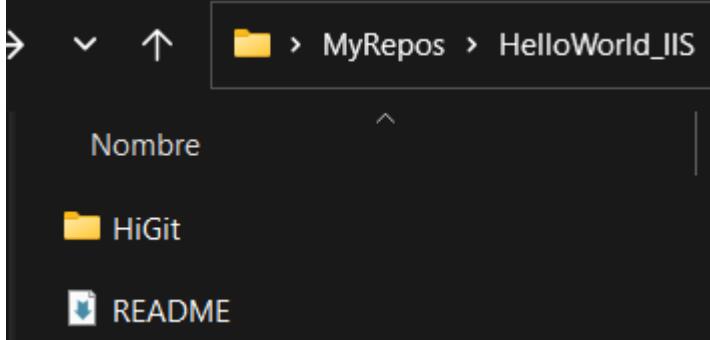
MINGW64:/c/Users/ETRIGUER/Desktop/MyRepos/HelloWorld_IIS
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos
$ git clone https://github.com/etriguerues/HelloWorld_IIS.git
Cloning into 'HelloWorld_IIS'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos
$ cd HelloWorld_IIS/
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/HelloWorld_IIS (main)
$ ...

```

Como podemos ver, ya tenemos nuestro repositorio clonado y está alojado en la rama main.

10. Agregar al repositorio nuestro proyecto java creado en la Parte I de esta temática.



11. Revisar el status del repositorio usando el comando "git status".

```

MINGW64:/c/Users/ETRIGUER/Desktop/MyRepos/HelloWorld_IIS
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos
$ git clone https://github.com/etriguerues/HelloWorld_IIS.git
Cloning into 'HelloWorld_IIS'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos
$ cd HelloWorld_IIS/
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/HelloWorld_IIS (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    HiGit/
nothing added to commit but untracked files present (use "git add" to track)

ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/HelloWorld_IIS (main)
$ 

```

Como vemos ya tenemos cargada la carpeta del proyecto a nuestro repositorio.

12. Ahora vamos a hacer un git add de todo lo que agregamos a nuestro repositorio.

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    HiGit/
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$ git add .
```

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
```

13. Al hacer un git status, vemos que tenemos cambios para hacer commit. (Entregar captura)

```
$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

Changes to be committed:

```
(use "git restore --staged <file>..." to unstage)
  new file:  HiGit/.classpath
  new file:  HiGit/.project
  new file:  HiGit/.settings/org.eclipse.core.resources.prefs
  new file:  HiGit/.settings/org.eclipse.jdt.core.prefs
  new file:  HiGit/bin/hiGit/HiGit.class
  new file:  HiGit/src/hiGit/HiGit.java
```

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
```

14. Hacemos el commit, para ello usamos git commit -m "My first commit"

Si no nos hemos registrado nos mostrará lo siguiente:

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
```

```
$ git commit -m "My first commit"
```

Author identity unknown

```
*** Please tell me who you are.
```

Run

```
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
```

to set your account's default identity.

Omit --global to set the identity only in this repository.

```
fatal: unable to auto-detect email address (got 'ETRIGUER@ETRIGUER-HP9ES64VUB.(none)')
```

Debemos ejecutar ambos comandos.

```
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
```

Luego probar de nuevo con el commit:

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$ git commit -m "My first commit"
[main 7520413] My first commit
 6 files changed, 53 insertions(+)
 create mode 100644 HiGit/.classpath
 create mode 100644 HiGit/.project
 create mode 100644 HiGit/.settings/org.eclipse.core.resources.prefs
 create mode 100644 HiGit/.settings/org.eclipse.jdt.core.prefs
 create mode 100644 HiGit/bin/higit/HiGit.class
 create mode 100644 HiGit/src/higit/HiGit.java
```

Nuestro commit ha sido exitoso.

15. Ahora al hacer un git status nos recomienda hacer un git push.

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

16. Haciendo git push (Entregar captura)

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$ git push
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (14/14), 1.78 KiB | 910.00 KiB/s, done.
Total 14 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/etriguerues/Helloworld_IIS.git
  3e19de8..7520413  main -> main
```

```
ETRIGUER@ETRIGUER-HP9ES64VUB MINGW64 ~/Desktop/MyRepos/Helloworld_IIS (main)
$
```

Ahora nuestros cambios ya están publicados en la rama de GitHub.(Entregar captura)

The screenshot shows a GitHub repository page for 'etriguerues/Helloworld\_IIS'. At the top, there are navigation links: Pull requests, Actions, Projects, Wiki, Security, Insights, Settings, and a green 'Code' button. Below the header, the repository details are shown: 'main' branch, 1 branch, 0 tags. The commit history is listed:

Commit	Author	Date
HiGit	etriguerues	5 minutes ago
README.md	etriguerues	2 hours ago

Each commit has a red box around it. To the right of the commits, there is an 'About' section with the text 'Cloning my first repository.' and a list of metrics: 0 stars, 1 watching, 0 forks. Below the commits, there is a preview of the 'README.md' file which contains the text 'HelloWorld\_IIS' and 'Cloning my first repository.'

Ahora ya podemos compartir nuestro proyecto con más personas y que puedan codificar en él.

## RECOMENDACIONES A LA HORA DE TRABAJAR CON VARIOS COLABORADORES EN UN MISMO PROYECTO:

1. Siempre se debe hacer un "git pull" antes de iniciar a codificar en nuestro proyecto si lo hemos compartido con otros colaboradores, esto es para descargar los últimos cambios que los demás han hecho.
2. Hacer push de cambios bastante significativos en nuestro proyecto. Si hemos codificado bastante durante el día
3. Tener una buena comunicación con los demás para no trabajar sobre los mismos procesos y así evitar conflictos al hacer merge de cambios.

## Parte II-35%

Temática: Gestión de la liberación del Software

Objetivos-

- Investigar y entender conceptos importantes en la liberación de software
- Usar GitHub como herramienta para la gestión de liberación de software.

Indicaciones: A continuación, se presentan 4 partes sobre la temática de Gestión de la liberación de Software.

**Deberá seguir las indicaciones de entrega y entregar un documento PDF EXCLUSIVAMENTE con la solución de:**

PARTE I: Completa (10%)

PARTE II: Captura de pantalla en el numeral 4 (20%)

PARTE III: Captura de pantalla en el numeral 3 (20%)

PARTE IV: Última captura del numeral 3 (pull request a develop) 15%

Última captura del numeral 4 (pull request a release) 15%

Última captura del numeral 5 (pull request a main) 15%

Penúltima captura del numeral 6 (publicando hotfix) 5%

Siga las indicaciones y evite entregar más de lo que se le pide, las partes y numerales a entregar están resaltadas en verde.

PARTE I: Entender que es un release y en qué momento crearlos

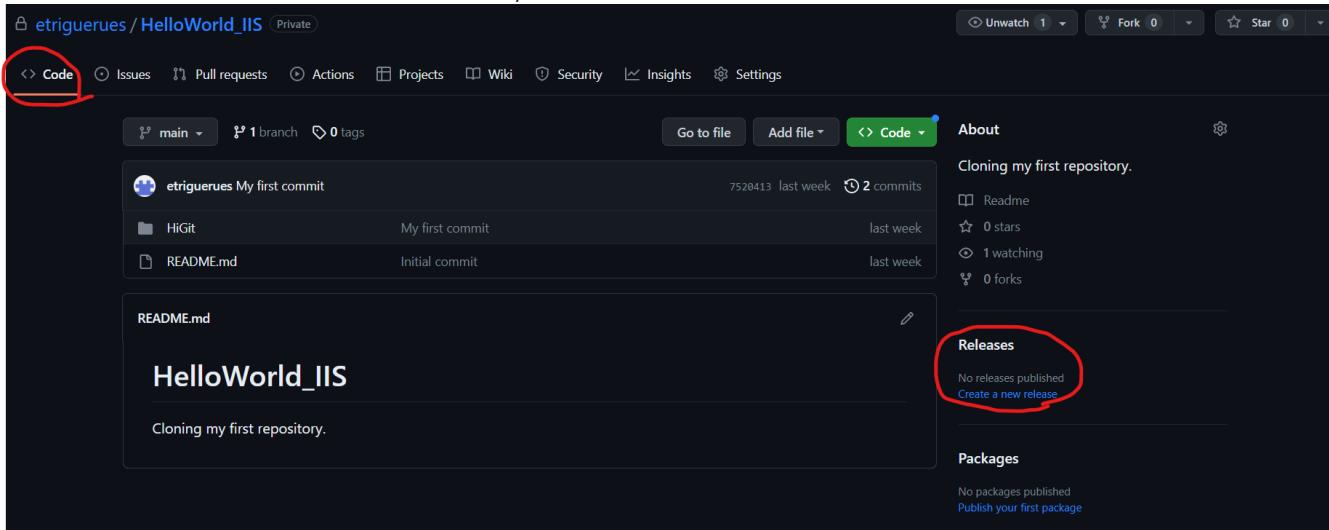
De respuesta a los siguientes enunciados:

1. ¿Qué es un release?
2. Liste al menos 4 momentos en que se puede generar un release
3. ¿Qué es un Hotfix?
4. ¿Qué es semántica de versionamiento?  
Puede leer más en: <https://semver.org/>
5. ¿Qué son los tags en github y para qué sirven?
6. ¿Qué es un pre-release en github y cuál es su objetivo?
7. ¿Qué son las reglas de protección de rama y para qué sirven?

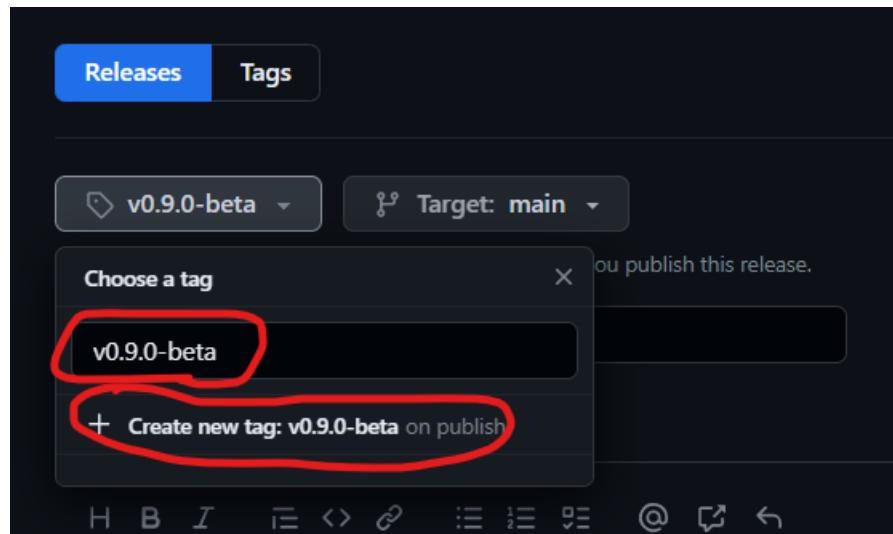
## PARTE II: Creando un Pre-release

Es importante tener un repositorio creado en github (puede usar el que creó en la parte I, de preferencia público para poder configurar las reglas de protección).

1. Dentro de la sección de Code, clic en "create a new release"



2. Crear un nuevo tag-Esc



3. Completar la descripción, check "Set as a pre-release" y "Publish release"

The screenshot shows the GitHub Releases interface. At the top, there are two tabs: 'Releases' (highlighted in blue) and 'Tags'. Below the tabs, there are dropdown menus for 'v0.9.0-beta' (with a 'Pre-release' badge) and 'Target: main'. A message says 'Excellent! This tag will be created from the target when you publish this release.' A 'Release title' input field is present. Below it, there are 'Write' and 'Preview' tabs, with 'Write' being active. The main area contains a rich text editor with various styling options like bold, italic, and code blocks. The text 'Esta release expone una funcionalidad BETA a los usuarios' is written in the editor and has a red underline. To the right of the editor, it says 'Previous tag: auto'. Below the editor, there's a note to attach files and a placeholder for binaries. At the bottom, there are two buttons: 'Set as a pre-release' (with a checked checkbox and a red circle), 'Publish release' (highlighted with a red oval and a red border), and 'Save draft'.

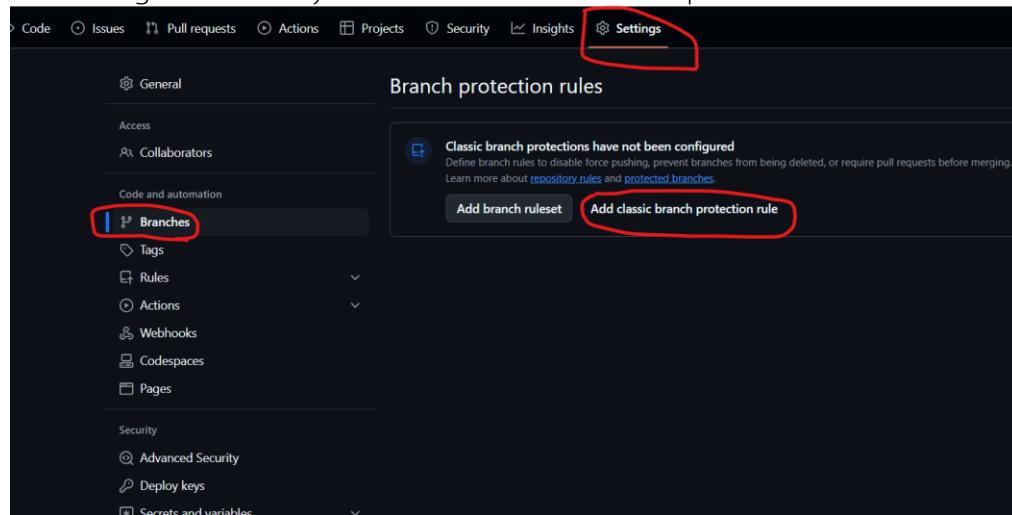
4. Ya podemos ver nuestro commit, tags y código fuente en la sección de Assets. (Entregar captura)

This screenshot shows the 'Assets' section for the 'v0.9.0-beta' release. At the top, it shows the release name 'v0.9.0-beta' with a 'Pre-release' badge, the author 'etriguerues', the date 'released this now', the tag 'v0.9.0-beta', and the commit ID '7520413'. Below the release summary, there is a description: 'Esta release expone una funcionalidad BETA a los usuarios'. The 'Assets' section lists two items: 'Source code (zip)' and 'Source code (tar.gz)', both uploaded 'last week'. Both of these items are circled with red lines.

## PARTE III: Reglas de protección de rama

Como sabemos la rama "release" y "main" o también llamada "master" deben estar protegida para que al hacer merge sobre ella se tenga al menos un control de lo que se va integrar y si la integración a dicha rama es algo planificado.

1. Para configurar reglas de protección de rama, hacemos lo siguiente:  
Ir a Settings>Branches y clic en "Add classic branch protection rule"

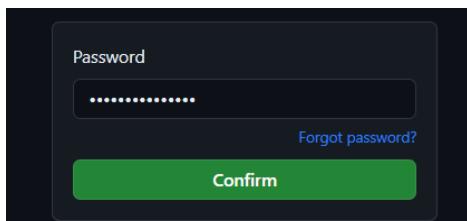
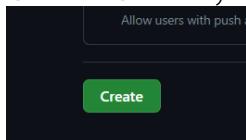


2. Vamos a escribir como pattern el nombre de la rama a la cual queremos dar protección en este ejemplo es la rama main, y para ellos vamos a agregar una regla de protección.

Con esta regla vamos a requerir siempre un Pull request antes de hacer el merge a la rama, y el número de personas que van a aprobar para efectos de ejemplo se dejará una persona, generalmente los approvals son los devops y el jefe del proyecto cuando se va pasar a una rama productiva, en caso de una rama release es un QA.

Puedes probar con más reglas de protección cuando lo deseas.

Clic en "Create" y luego confirmar la creación con nuestra contraseña.



Y podremos ver una regla de protección a nuestra rama, como estamos usando un repo privado de organización no nos muestra la advertencia de "Not enforced", caso contrario si estamos trabajando con un repo privado de organización.

3. Vamos a crear 2 reglas más, una release y una develop, le agregaremos como regla de protección sólo a la rama release.

Usaremos el siguiente pattern: release\* y le asignaremos la misma regla agregada a la rama main, con esto estamos indicando que todo lo que venga de release va ser integrado a través de un pull request. Tendrá también un approval que generalmente la aprobación siempre lo hace un QA a la hora de hacer la integración a la rama.

Importante recalcar que si eligió un repositorio privado no organizacional las reglas de firmas no van a surtir efecto si quiere validarlas mostrando la advertencia de la imagen, para evitarlo si quiere validar las reglas, muévase a un repo público o GitHub Team o Cuenta Empresarial.

Nuestras reglas van a ir quedando de la siguiente forma (Entregar captura):

The screenshot shows the 'Branch protection rules' section of a GitHub repository. It lists three branches: 'main', 'release\*', and 'develop'. Each branch has a status of 'Not enforced' with an information icon, an 'Edit' button, and a 'Delete' button. A prominent yellow warning box at the top states: '⚠ Your protected branch rules for your branch won't be enforced on this private repository until you move to a GitHub Team or Enterprise organization account.' with an 'Upgrade' button.

#### PARTE IV: Creación de Hotfix

En ocasiones necesitamos realizar correcciones porque en producción se presentó algún error y necesitamos crear una rama hotfix para poder aislar el cambio, corregirlo, probarlo e integrarlo nuevamente a la rama main.

Como ejemplo estamos usando un proyecto para impresión de un Hello World IIS, pero en nuestra rama productiva no se está mostrando el mensaje de Hello World IIS, los clientes al ver que no se muestra, reportan el error y el equipo de incidentes procede a resolverlo.

The screenshot shows a GitHub code editor for a Java file named 'HiGit.java' located in the 'src' directory of a repository named 'HelloWorld\_IIS / HiGit'. The code contains a single-line main method with a TODO comment: 'public static void main(String[] args) { // TODO Auto-generated method stub }'. The code tab is selected, showing 10 lines (6 loc) and 124 Bytes.

Para ello:

1. Creamos una nueva rama a partir de la rama productiva donde se está mostrando el error(main).

2. Ahora vamos a editar el archivo, lo podríamos hacer desde cualquier IDE de preferencia, en este ejemplo, por ser un cambio pequeño se hará desde github y clic en Commit.

3. Vamos a generar el pullRequest a la rama develop y release para que estas queden actualizadas respectivamente.

Antes que todo crear las ramas develop y release:

The screenshot shows the 'Create a branch' dialog box overlaid on the GitHub branches page. In the dialog, the 'New branch name' field contains 'develop' and the 'Source' dropdown is set to 'main'. The 'Create new branch' button at the bottom right is highlighted with a red oval.

deben quedar de la siguiente manera:

The screenshot shows the GitHub branches page after creation. Under 'Default', there is one branch named 'main'. Under 'Your branches', there are three branches: 'release' (updated 'now'), 'develop' (updated '1 minute ago'), and 'hotfix-helloworldIIS' (updated '6 minutes ago').

Ahora vamos a generar el pullRequest a la rama develop y release, para ello nos posicionamos en cualquiera de las ramas involucradas, como primer paso vemos que en la rama hotfix nos muestra el botón "Compare & pull request"

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

hotfix-helloworldIIS had recent pushes less than a minute ago

Compare & pull request

hotfix-helloworldIIS 4 branches 1 tag

Go to file Add file <> Code

This branch is 1 commit ahead of main.

Contribute

etriguerues Update HiGit.java ... 2e121b5 now 3 commits

HiGit Update HiGit.java now

README.md Initial commit last week

README.md

HelloWorld\_IIS

Cloning my first repository.

About

Cloning my first repository.

Readme 0 stars 1 watching 0 forks

Releases 1 tags Create a new release

Packages No packages published Publish your first package

Languages Java 100.0%

la flecha nos indica hacia qué rama (base) van los cambios

base: develop compare: hotfix-helloworldIIS Able to merge. These branches can be automatically merged.

Update HiGit.java

Write Preview

Fix de mensaje

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

### Update HiGit.java #1

etriguerues wants to merge 1 commit into `develop` from `hotfix-helloworldIIS`

Conversation 0    Commits 1    Checks 0    Files changed 1

etriguerues commented now

Fix de mensaje

Update HiGit.java

Verified 2e121b5

Add more commits by pushing to the `hotfix-helloworldIIS` branch on [etriguerues/HelloWorld\\_IIS](#).

Continuous integration has not been set up  
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

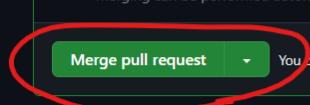
This branch has no conflicts with the base branch  
Merging can be performed automatically.

Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

Write Preview

Leave a comment



### Entregar captura de la integración desde feature a develop

### Update HiGit.java #1

Merged etriguerues merged 1 commit into `develop` from `hotfix-helloworldIIS` now

Conversation 0    Commits 1    Checks 0    Files changed 1

etriguerues commented now

Fix de mensaje

Update HiGit.java

Verified 2e121b5

etriguerues merged commit `b1ec2b` into `develop` now

Pull request successfully merged and closed

You're all set—the `hotfix-helloworldIIS` branch can be safely deleted.

Delete branch

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

4. Ya hemos integrado el cambio a la develop, ahora lo haremos a la release.

## Comparing changes

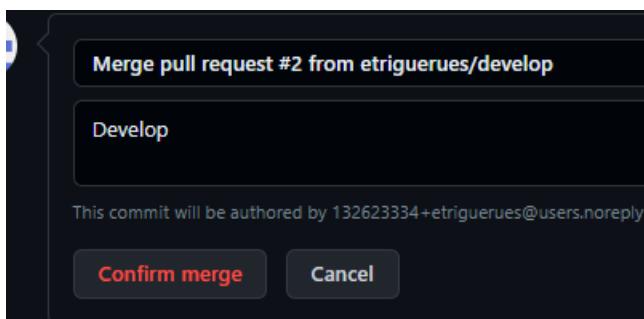
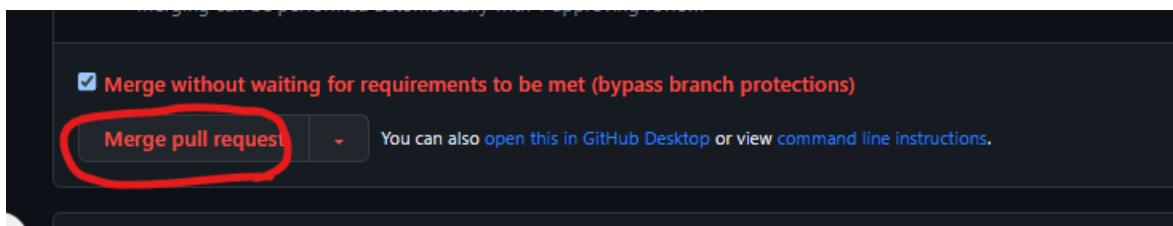
Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

The screenshot shows the GitHub interface for comparing branches. At the top, it says "base: release" and "compare: develop". It also says "✓ Able to merge. These branches can be automatically merged." Below this is a large text area with "Leave a comment" placeholder and a "Create pull request" button at the bottom right. There is a note at the bottom left: "Remember, contributions to this repository should follow our GitHub Community Guidelines."

Si notamos (Si hemos trabajado con un repo público y aplicado las reglas) en las reglas de protección necesitamos un approval para hacer la integración a la rama, generalmente lo hace un QA. Dado que en nuestro proyecto no hay más colaboradores, haremos un bypass de revisión.

Si usted trabajó con un repo privado no notará los efectos de las reglas de protección de rama y podrá hacer el merge sin ningún problema.

The screenshot shows a GitHub pull request titled "Develop #2" where "etriguerues" wants to merge 2 commits into "release" from "develop". The pull request has 0 conversations, 2 commits, 0 checks, and 1 file changed. The commit details show "etriguerues commented 3 minutes ago" with "No description provided." and "etriguerues added 2 commits 6 minutes ago" including "Update Higit.java" and "Merge pull request #1 from etriguerues/hotfix-helloworldIIS". The status bar indicates "Verified" for both commits. The pull request is closed by "etriguerues" 1 minute ago and reopened 1 minute ago. A note at the bottom says "Add more commits by pushing to the develop branch on etriguerues/HelloWorld\_IIS." The bottom section shows merge status: "Review required" (At least 1 approving review is required by reviewers with write access), "Merging is blocked" (Merging can be performed automatically with 1 approving review), and a red-circled checkbox for "Merge without waiting for requirements to be met (bypass branch protections)".

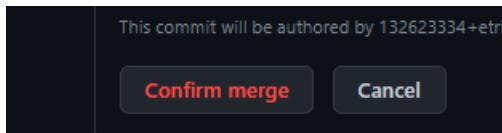


Entregar captura de la integración desde develop a release

A screenshot of the GitHub commit history for a merged pull request. The commit message is "etriguerues merged 2 commits into release from develop now". The commit history shows:

- etriguerues commented 5 minutes ago: "No description provided."
- etriguerues added 2 commits 9 minutes ago:
  - Update Higit.java ... (Verified 2e121b5)
  - Merge pull request #1 from etriguerues/hotfix-helloworldI18N ... (Verified bc1ec2b)
- etriguerues closed this 4 minutes ago (with a warning icon).
- etriguerues reopened this 3 minutes ago (with a green info icon).
- etriguerues merged commit a1f168a into release now (with a purple info icon). There is a "Revert" button next to this entry.
- A summary message at the bottom states "Pull request successfully merged and closed" and "You're all set—the develop branch can be safely deleted." with a "Delete branch" button.

5. Ahora vamos a hacer la integración a la main, y haremos un bypass de los approvals (Se hará bypass para efectos de ejemplo, en ámbito laboral no es permitido).



Entregar captura, de merge desde release a main

This commit will be authored by 132623334+etri

**Confirm merge** **Cancel**

## Release #3

Merged etrighuerues merged 3 commits into main from release now

Conversation 0 Commits 3 Checks 0 Files changed 1

etrighuerues commented 1 minute ago

Mejora de mensaje

etrighuerues added 3 commits 13 minutes ago

- Update HIGIT.java ... Verified 2e121b5
- Merge pull request #1 from etrighuerues/hotfix-helloworldIIS ... Verified bc1ec2b
- Merge pull request #2 from etrighuerues/develop ... Verified a1f168a

etrighuerues merged commit 8760f62 into main now

Revert

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Comment

6. Ahora nos vamos a releases para hacer el cambio de versión. Vamos a suponer que ya teníamos la versión 1.0.0 y como el cambio a implementar es un hotfix, vamos a modificar el tercer dígito.

Nos vamos a releases

The screenshot shows the GitHub Releases page for the repository `etriguerues / HelloWorld_IIS`. It displays two releases:

- v1.0.0: Merge pull request #3 from etriguerues/release** (Latest)  
Published now by etriguerues. Contains assets: Source code (zip) and Source code (tar.gz), both uploaded 4 minutes ago.
- v0.9.0-beta** (Pre-release)  
Published 4 hours ago by etriguerues. Contains assets: Source code (zip) and Source code (tar.gz), both uploaded 4 minutes ago. Description: Esta release expone una funcionalidad BETA a los usuarios.

A red circle highlights the "Draft a new release" button at the top right of the page.

Y creamos el tag para asignarlo.

The screenshot shows the "Choose a tag" modal in GitHub. The modal lists the tag `v1.0.1` and an option to "Create new tag: v1.0.1 on publish".

Agregamos descripción y generamos notas de release.

The screenshot shows the GitHub Releases interface for a repository. At the top, there are tabs for 'Releases' (selected) and 'Tags'. Below that, a dropdown shows 'v1.0.1' and a target dropdown shows 'main'. A message says 'Excellent! This tag will be created from the target when you publish this release.' The title 'v1.0.1 Hotfix' is in a text input field. The main area is titled 'Write' and contains a rich text editor with a red circle around the 'Generate release notes' button. The text in the editor is: 'Release para mostrar el mensaje Hello World IIS'. Below the editor is a note: 'Attach files by dragging & dropping, selecting or pasting them.' Underneath the editor is a note: '↓ Attach binaries by dropping them here or selecting them.' At the bottom, there are checkboxes: 'Set as a pre-release' (unchecked) and 'Set as the latest release' (checked). Buttons at the bottom are 'Publish release' (highlighted with a red circle) and 'Save draft'.

Por último, publicamos el release.

This screenshot shows the same GitHub Releases interface after publishing the release. The title 'v1.0.1 Hotfix' is now bolded. The release notes content is identical to the previous screenshot. The 'Set as the latest release' checkbox is checked. The 'Publish release' button at the bottom is highlighted with a red circle. The right sidebar contains general release-related information.

Podemos ver el resumen (Entregar captura):

The screenshot shows a GitHub release page for 'v1.0.1 Hotfix'. At the top, it says 'Releases / v1.0.1'. The main title is 'v1.0.1 Hotfix' with a 'Latest' button. Below the title, it shows 'etriguerues released this now' and the commit hash '8760f62'. A note says 'Release para mostrar el mensaje Hello World IIS'. The 'What's Changed' section lists three items: 'Update HiGit.java by @etriguerues in #1', 'Develop by @etriguerues in #2', and 'Release by @etriguerues in #3'. The 'New Contributors' section shows one contributor, '@etriguerues'. The 'Contributors' section lists 'etriguerues'. The 'Assets' section shows two files: 'Source code (zip)' and 'Source code (tar.gz)'. Both were uploaded 11 minutes ago.

Podemos también comparar los releases que hemos creado y tener un control de que cambios y funcionalidades se han agregado respecto al actual u otros.

The screenshot shows a GitHub release comparison interface. It has tabs for 'Releases' and 'Tags'. The 'Releases' tab is active. It shows a comparison between 'v1.0.1' (released 1 minute ago) and 'v0.9.0-beta' (released 12 minutes ago). The 'v1.0.1' release notes say 'Release para mostrar el mensaje Hello World IIS'. The 'What's Changed' section for the comparison shows the same three changes as the previous release. The 'Contributors' section shows 'etriguerues'. The 'Assets' section shows the same two files as the previous release.

## Parte III: 30%

Temática: Maven y Jenkins con Junit Test

### Objetivos-

- Entender qué es y para qué sirve Maven
- Investigar y entender que son las pruebas unitarias y JUnit con sus métodos básicos.
- Investigar y entender conceptos de Integración continua (CI) y la entrega continua (CD).
- Entender que es un pipeline y un Job
- Entender para qué sirve Jenkins y las diferencias con GitHub Actions

**Indicaciones:** A continuación, se presentan 5 partes sobre la temática de maven y JUnit a las cuales deberá dar seguimiento de sus indicaciones para dar solución.

**Deberá seguir las indicaciones de entrega y entregar un documento PDF EXCLUSIVAMENTE con la solución de:**

**PARTE I: Completa (10%)**

**PARTE II: Captura de pantalla en el numeral 6-Proyecto creado (20%)**

**PARTE III: Captura de pantalla en el numeral 4-Ejecución de prueba Unitaria (20%)**

**PARTE IV: Captura de pantalla en el numeral 6 -Panel principal de Jenkins, deberá verse su nombre (20%).**

**PARTE V: Captura de pantalla en el numeral 9 -Proyecto en repositorio github (15%)**

**Captura de pantalla en el numeral 12 -Status del proyecto en Jenkins (15%)**

**Siga las indicaciones y evite entregar más de lo que se le pide, las partes y numerales a entregar están resaltadas en verde.**

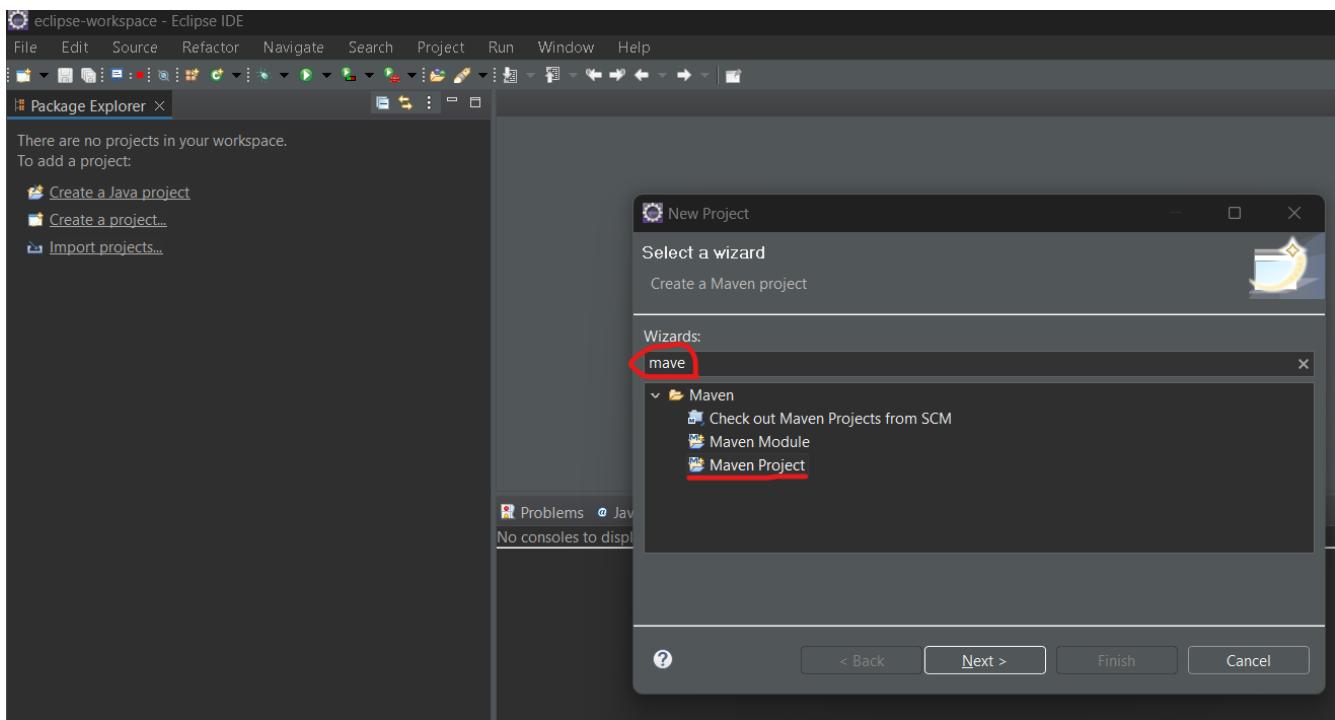
**PARTE I: Integración continua.**

De respuesta a los siguientes enunciados:

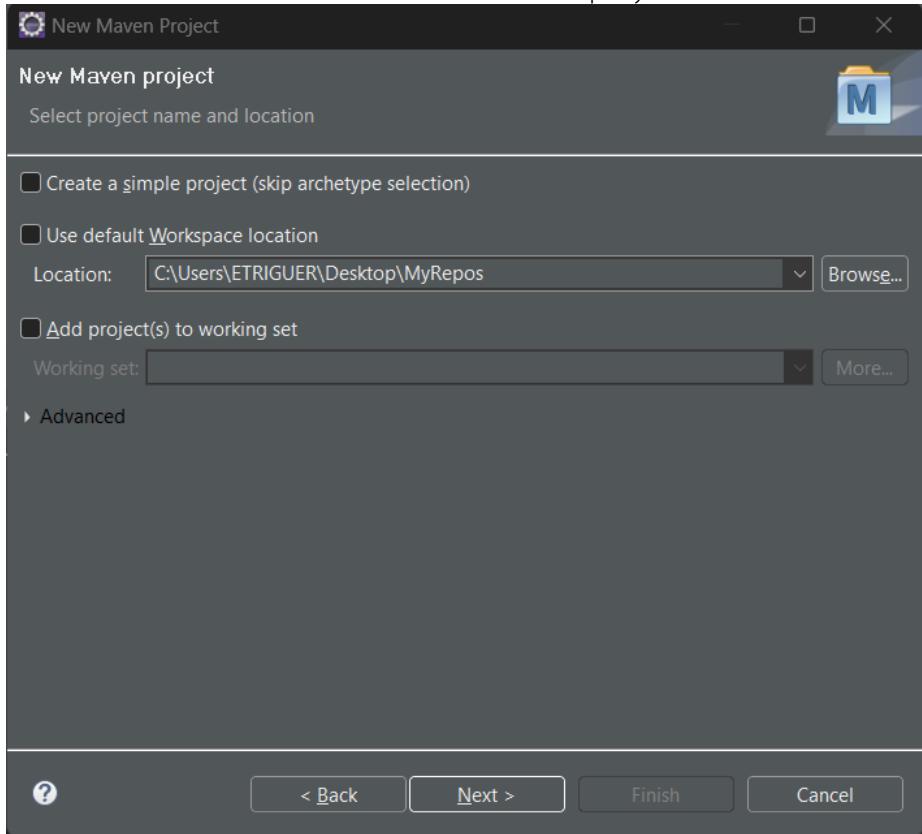
1. ¿Qué es Eclipse IDE?
2. ¿Cómo configurar el jdk y editar las variables de entorno para el uso correcto de proyectos Java?
3. ¿Qué es Maven y para qué nos sirve?
4. ¿Qué son las pruebas unitarias?
5. ¿Qué es Junit y para qué nos sirve?
6. ¿En qué consiste la integración continua (CI) y la entrega continua (CD)?
7. ¿Qué es Jenkins y para qué nos sirve?
8. ¿Qué es un pipeline y para qué nos sirve?
9. ¿Qué son las GitHub Actions y para qué nos sirve?
10. ¿Cuáles son las diferencias entre Jenkins y GitHub Actions?

**PARTE II: Creación de proyecto Maven con Eclipse (Si no tiene instalado eclipse instale y tambien configure el jdk en las variables de entorno)**

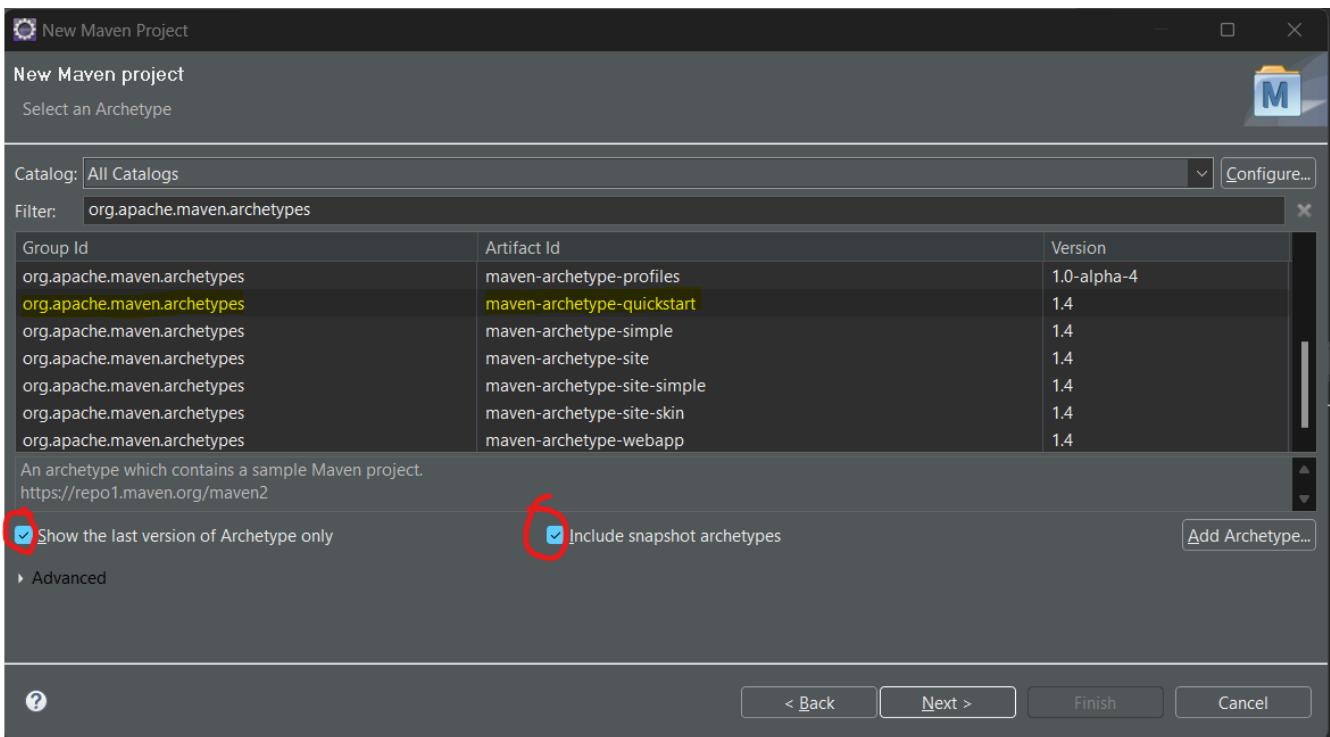
1. File>New>Project>Buscar la sección de proyectos de maven>Maven Project



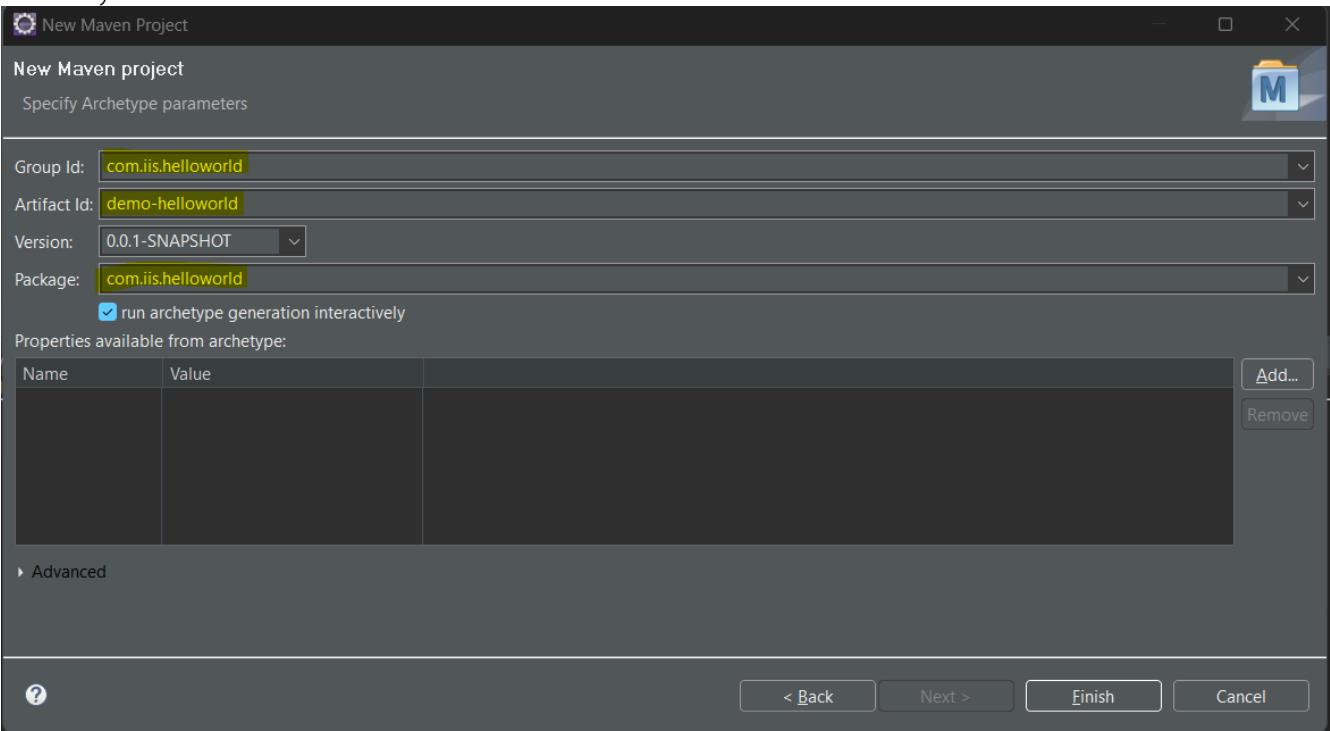
2. Seleccionar directorio de creación de proyecto.



3. Seleccionar el Archetype



4. Configurar los parámetros siguientes, **en vez de helloworld, escriba su CARNET ESTUDIANTIL** y clic en Finish:



5. Confirmar las propiedades de configuración con "Y" y presionar enter.

```

Progress (3): 4.8/37 kB | 3.2/20 kB | 0/1.5 MB
Progress (3): 4.8/37 kB | 3.2/20 kB | 0/1.5 MB
Progress (3): 4.8/37 kB | 3.2/20 kB | 0/1.5 MB
Progress (3): 4.8/37 kB | 3.2/20 kB | 0/1.5 MB
Progress (3): 4.8/37 kB | 3.2/20 kB | 0/1.5 MB

Downloaded from : https://repo.maven.apache.org/maven2/o
Downloaded from : https://repo.maven.apache.org/maven2/o
Downloaded from : https://repo.maven.apache.org/maven2/o
[INFO] Generating project in Interactive mode
[INFO] Archetype repository not defined. Using the one f
[INFO] Using property: groupId = com.iis.helloworld
[INFO] Using property: artifactId = demo-helloworld
[INFO] Using property: version = 0.0.1-SNAPSHOT
[INFO] Using property: package = com.iis.helloworld
Confirm properties configuration:
groupId: com.iis.helloworld
artifactId: demo-helloworld
version: 0.0.1-SNAPSHOT
package: com.iis.helloworld
Y: : Y

```

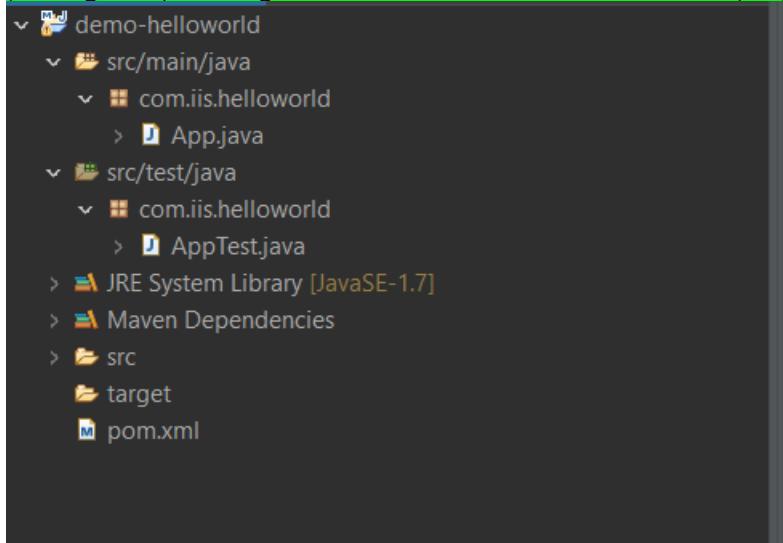
6. El proyecto se ha creado de forma satisfactoria.

```

package: com.iis.helloworld
Y: : Y
[INFO] -----
[INFO] Using following parameters for creating project from Archetype: maven-archetype-quickstart:1.4
[INFO] -----
[INFO] Parameter: groupId, Value: com.iis.helloworld
[INFO] Parameter: artifactId, Value: demo-helloworld
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] Parameter: package, Value: com.iis.helloworld
[INFO] Parameter: packageInPathFormat, Value: com/iis/helloworld
[INFO] Parameter: package, Value: com.iis.helloworld
[INFO] Parameter: groupId, Value: com.iis.helloworld
[INFO] Parameter: artifactId, Value: demo-helloworld
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] Project created from Archetype in dir: C:\Users\ETRIGUER\Desktop\MyRepos\demo-helloworld
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 02:14 min

```

(Entregar captura siguiente, donde se vea su nombre de proyecto (Que contenga su Carnet estudiantil))



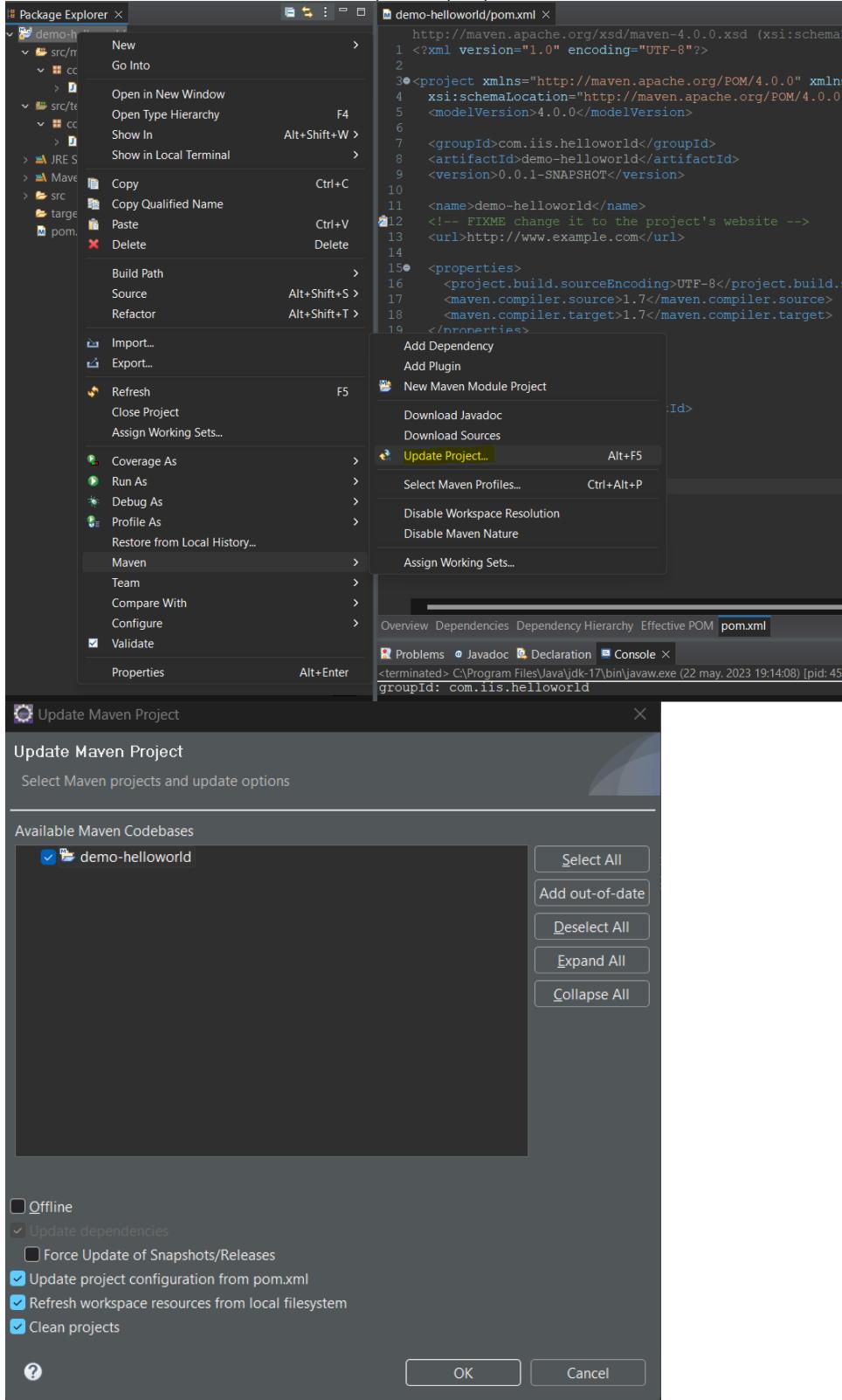
7. Vamos a borrar el tag correspondiente a <build> del archivo pom.xml

```
<version>4.11</version>
<scope>test</scope>
</dependency>
</dependencies>
<build>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to
        &lt;plugins&gt;
            &lt;!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean-plugin--&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-clean-plugin&lt;/artifactId&gt;
                &lt;version&gt;3.1.0&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;!-- default lifecycle, jar packaging: see https://maven.apache.org/ref/current/maven-core/lifecycles.html#jar-plugin--&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-resources-plugin&lt;/artifactId&gt;
                &lt;version&gt;3.0.2&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-compiler-plugin&lt;/artifactId&gt;
                &lt;version&gt;3.8.0&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-surefire-plugin&lt;/artifactId&gt;
                &lt;version&gt;2.22.1&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-jar-plugin&lt;/artifactId&gt;
                &lt;version&gt;3.0.2&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-install-plugin&lt;/artifactId&gt;
                &lt;version&gt;2.5.2&lt;/version&gt;
            &lt;/plugin&gt;
            &lt;plugin&gt;
                &lt;artifactId&gt;maven-deploy-plugin&lt;/artifactId&gt;
                &lt;version&gt;2.8.2&lt;/version&gt;
            &lt;/plugin&gt;
        &lt;/plugins&gt;
    &lt;/pluginManagement&gt;</pre>
```

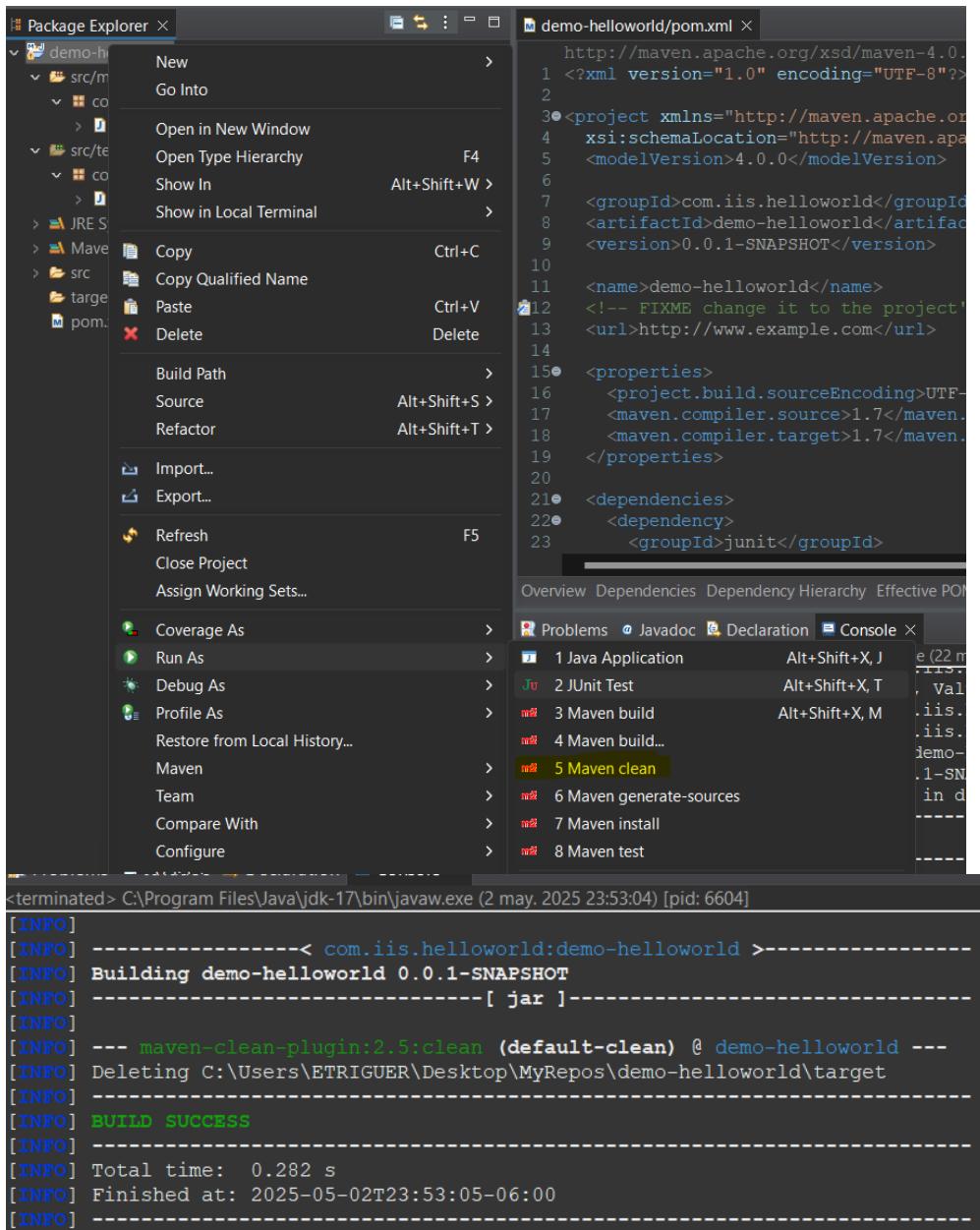
Quedando de la siguiente forma:

```
M demo-helloworld/pom.xml ×
http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
1 <?xml version="1.0" encoding="UTF-8"?>
2
3<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instantiation-schema"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.iis.helloworld</groupId>
8   <artifactId>demo-helloworld</artifactId>
9   <version>0.0.1-SNAPSHOT</version>
10
11  <name>demo-helloworld</name>
12  <!-- FIXME change it to the project's website -->
13  <url>http://www.example.com</url>
14
15<properties>
16   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17   <maven.compiler.source>1.7</maven.compiler.source>
18   <maven.compiler.target>1.7</maven.compiler.target>
19 </properties>
20
21<dependencies>
22  <dependency>
23    <groupId>junit</groupId>
24    <artifactId>junit</artifactId>
25    <version>4.13.2</version>
26    <scope>test</scope>
27  </dependency>
28 </dependencies>
29
30</project>
31
```

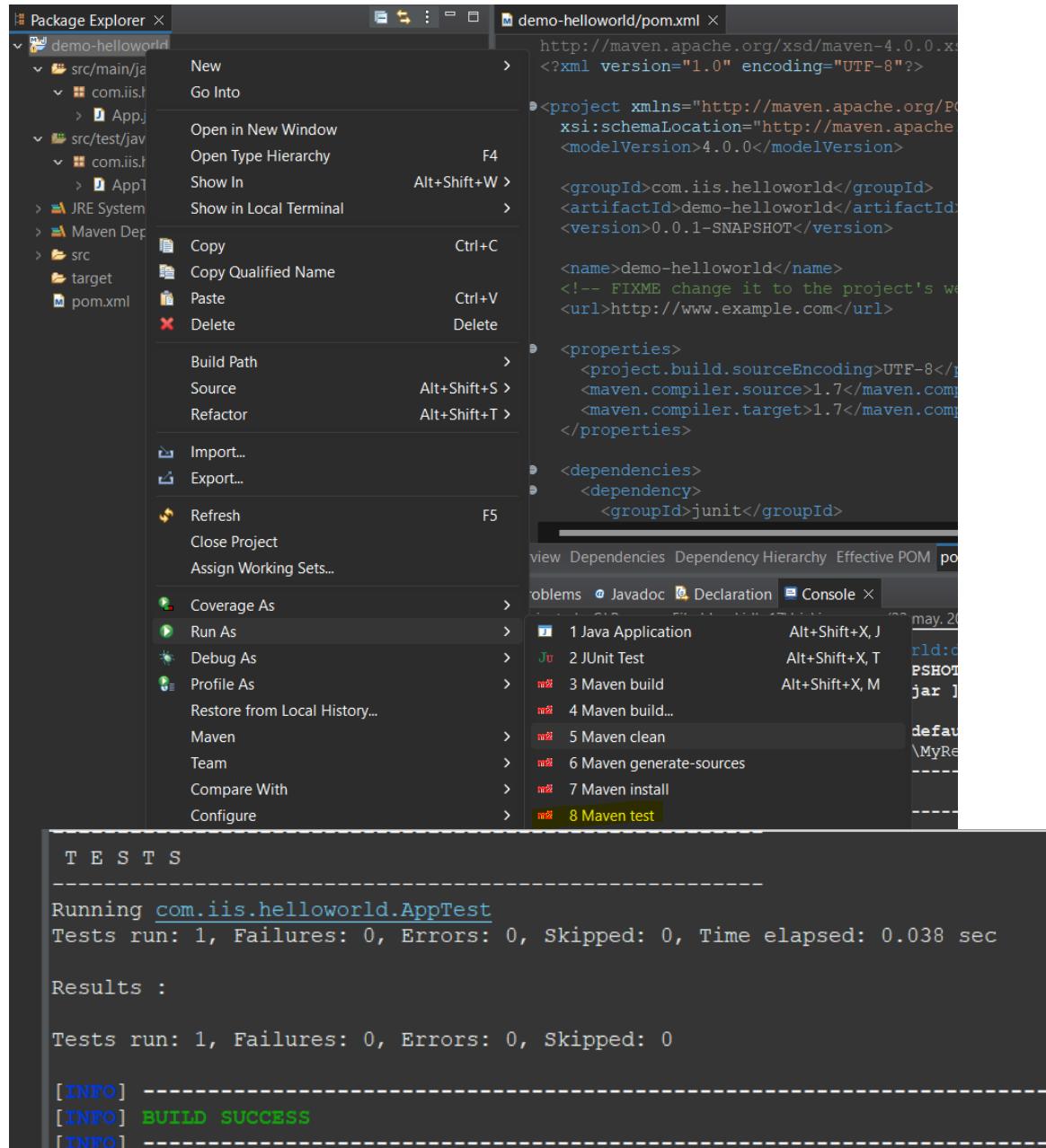
8. Click derecho> Maven>Update project



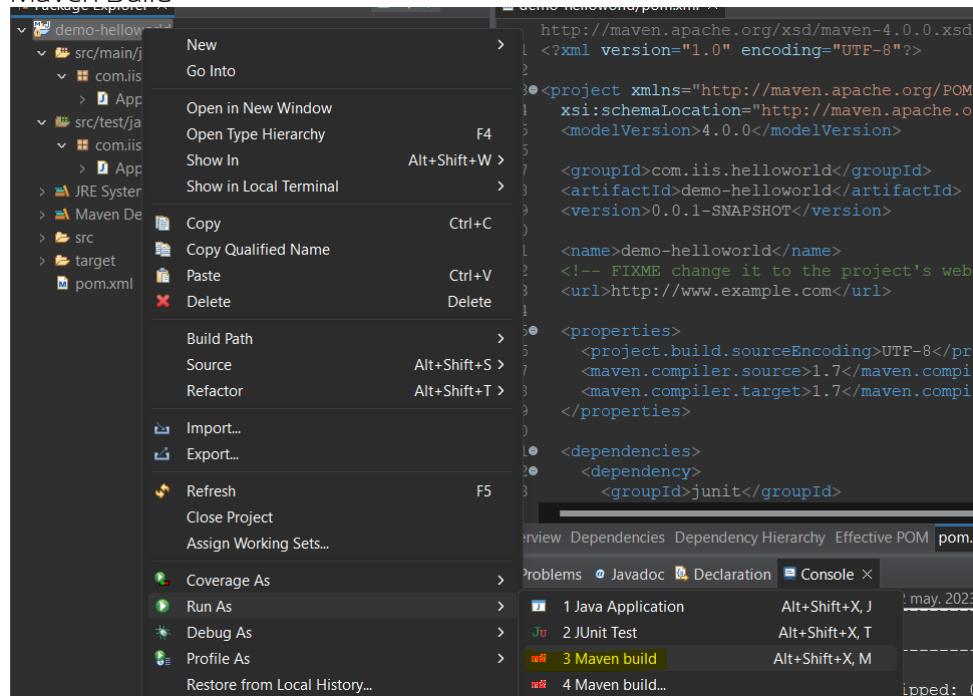
9. Click derecho> Maven Clean, Luego un Maven Test y finalmente Maven Build



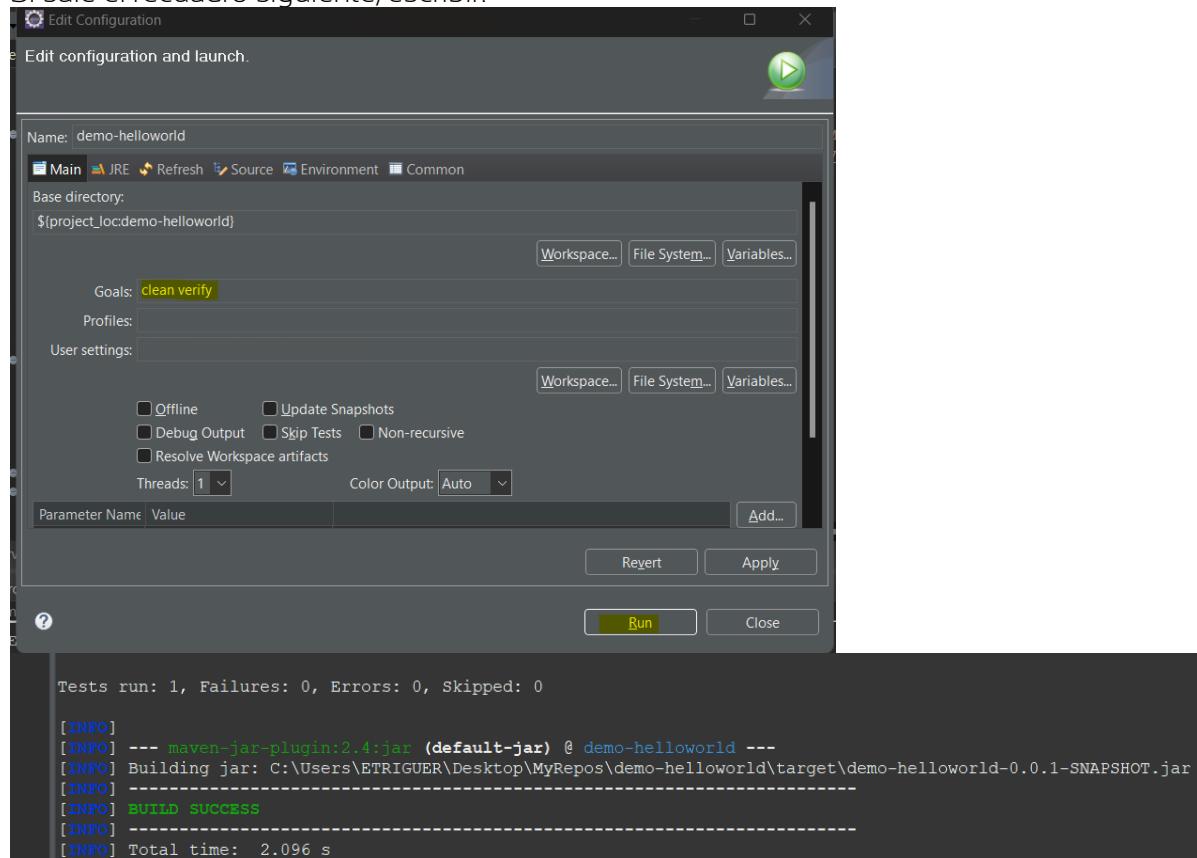
## Maven Test



## Maven Build

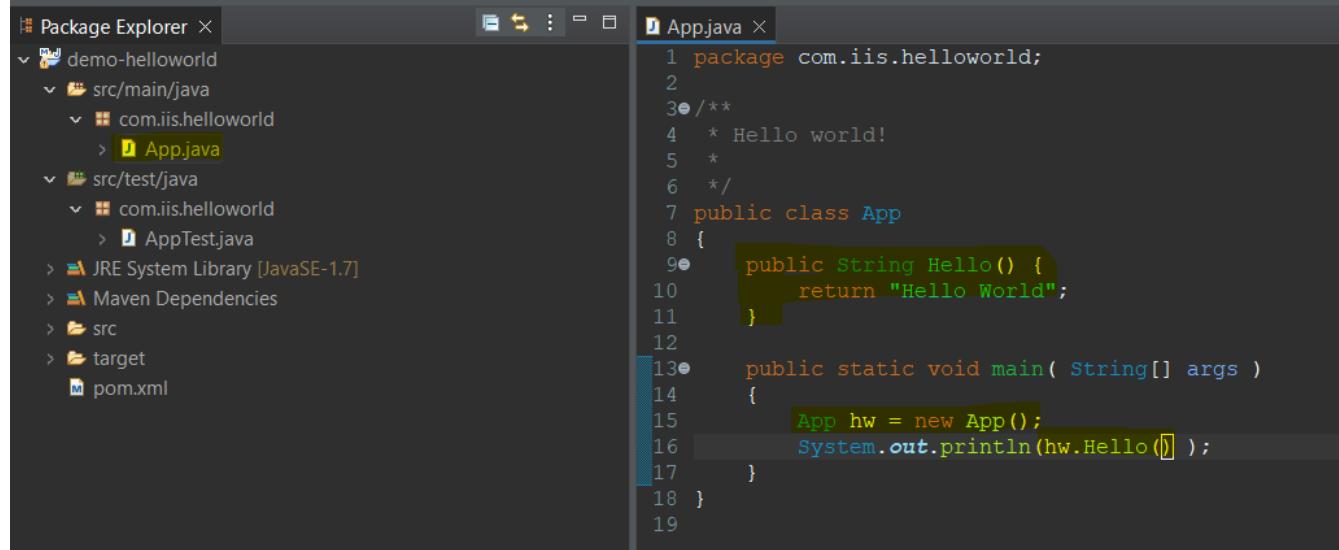


Si sale el recuadro siguiente, escribir:



## PARTE III: Pruebas Unitarias

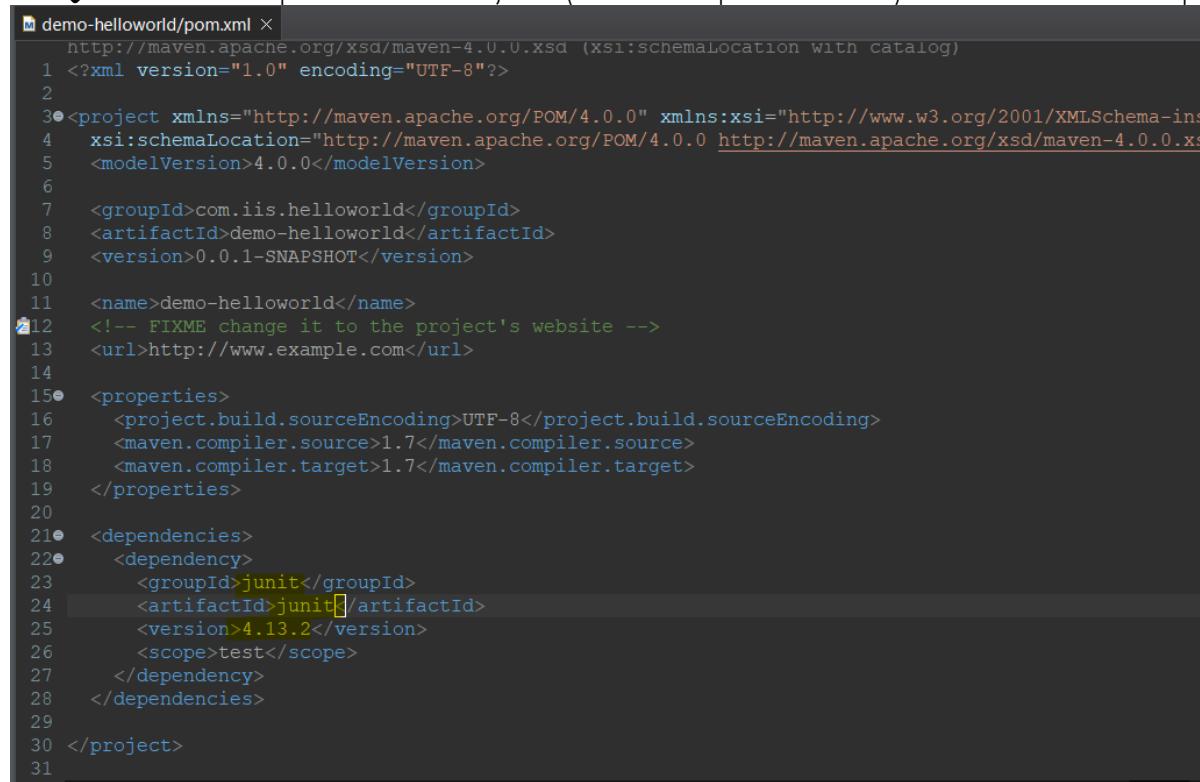
1. Agregar el siguiente método e instanciar en la clase principal.



```
1 package com.iis.helloworld;
2
3 /**
4  * Hello world!
5  */
6
7 public class App
8 {
9     public String Hello() {
10         return "Hello World";
11     }
12
13     public static void main( String[] args )
14     {
15         App hw = new App();
16         System.out.println(hw.Hello());
17     }
18 }
19
```

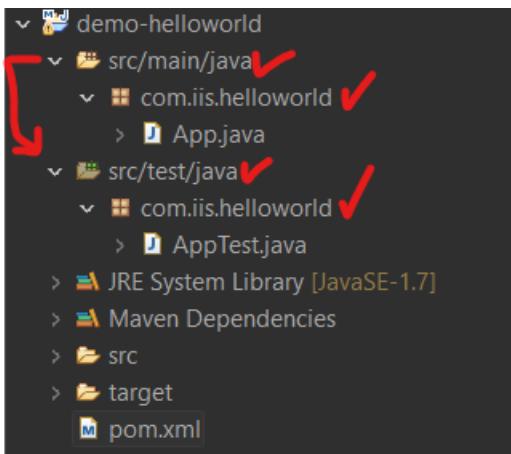
2. Para implementar pruebas unitarias asegurarnos de:

- ✓ Tener las dependencias de junit (la versión puede variar) en nuestro archivo pom.xml

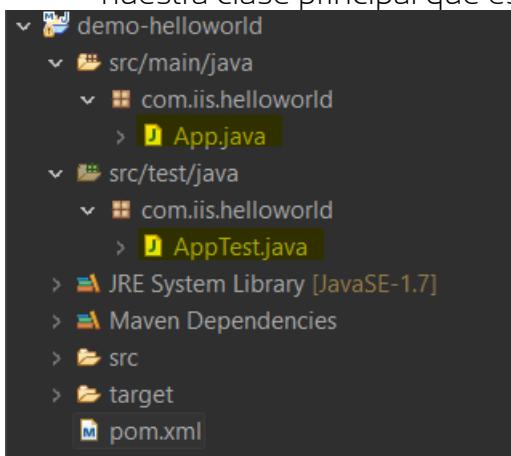


```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.iis.helloworld</groupId>
8   <artifactId>demo-helloworld</artifactId>
9   <version>0.0.1-SNAPSHOT</version>
10
11   <name>demo-helloworld</name>
12   <!-- FIXME change it to the project's website -->
13   <url>http://www.example.com</url>
14
15   <properties>
16     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17     <maven.compiler.source>1.7</maven.compiler.source>
18     <maven.compiler.target>1.7</maven.compiler.target>
19   </properties>
20
21   <dependencies>
22     <dependency>
23       <groupId>junit</groupId>
24       <artifactId>junit</artifactId>
25       <version>4.13.2</version>
26       <scope>test</scope>
27     </dependency>
28   </dependencies>
29
30 </project>
31
```

- ✓ Tener una estructura de carpetas test idéntica a la de mi proyecto principal



- ✓ Nombrar las clases idénticas, y agregar la palabra Test al final. En el ejemplo tenemos nuestra clase principal que es App.java y nuestra clase a testear es la AppTest.java



- ✓ Codificar nuestra clase

En la clase es importante importar la librería de junit :

```
import org.junit.Test;
```

y los métodos necesarios de junit a utilizar:

```
import static org.junit.Assert.assertTrue;
```

También es importante usar la anotación @Test para que a la hora de correr las pruebas unitarias se pueda diferenciar qué clases son de prueba y cuáles no.

3. Para probar el método de nuestra clase principal, codificamos nuestra clase de prueba como se muestra:

The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer view displays a project structure for 'demo-helloworld' containing 'src/main/java' and 'src/test/java' packages, each with an 'App.java' file. On the right, the code editor shows the 'AppTest.java' file:

```

1 package com.iis.helloworld;
2
3 import static org.junit.Assert.assertTrue;
4
5 /**
6  * Unit test for simple App.
7 */
8
9
10 public class AppTest
11 {
12     /**
13      * Rigorous Test :-)
14     */
15     @Test
16     public void testHello()
17     {
18         assertTrue( true );
19     }
20 }

```

Antes de probar nuestra clase vamos a hacer el build de nuestra prueba:

The screenshot shows the Eclipse IDE interface. The context menu for 'AppTest.java' is open, with 'Run As' selected. The 'Run As' submenu shows several options: Java Application, JUnit Test, Maven build, Maven clean, Maven generate-sources, Maven install, and Maven test. The 'Maven test' option is highlighted. In the bottom right corner, the 'Console' view shows the output of the Maven build:

```

22 may. 2023 20:33:4
1 Java Application Alt+Shift+X, J
2 JUnit Test Alt+Shift+X, T
3 Maven build Alt+Shift+X, M
4 Maven build... Alt+Shift+X, M
5 Maven clean Alt+Shift+X, M
6 Maven generate-sources Alt+Shift+X, M
7 Maven install Alt+Shift+X, M
8 Maven test Alt+Shift+X, M

Running com.iis.helloworld.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.035 sec

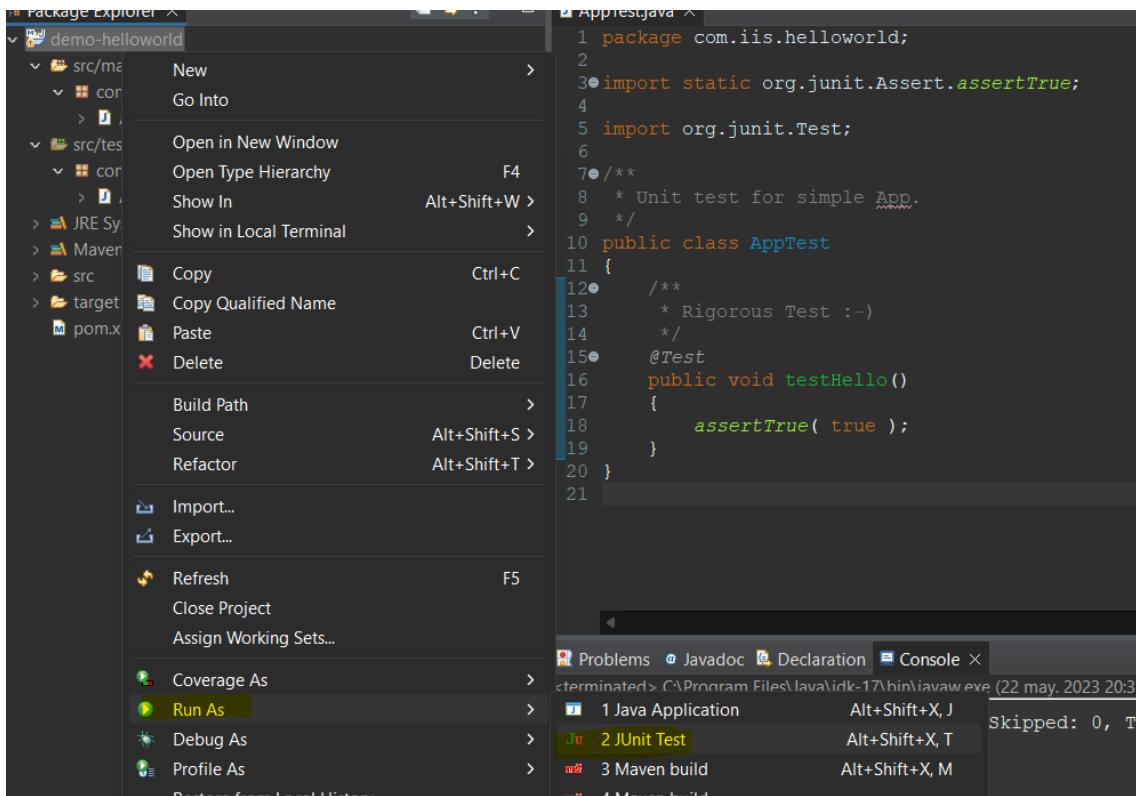
Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

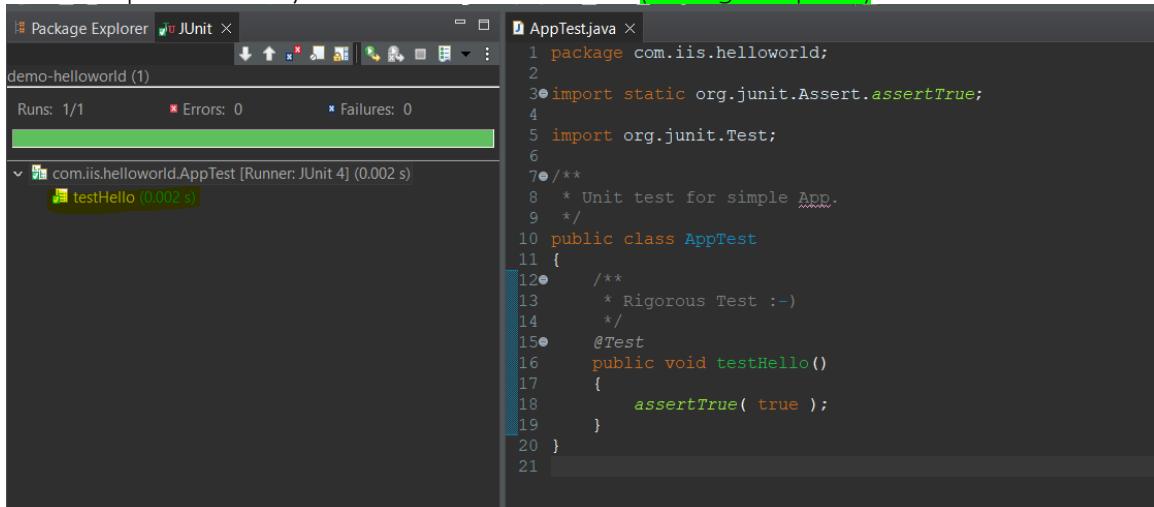
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 1.188 s

```

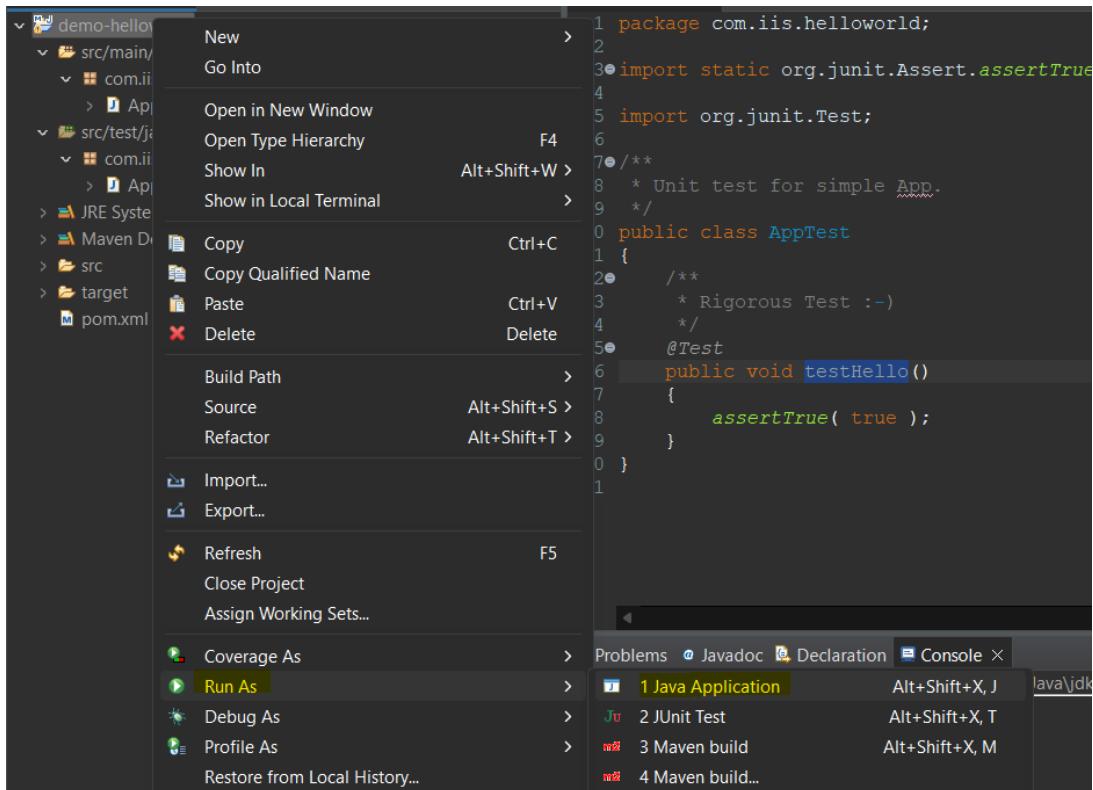
#### 4. Correr nuestra prueba con Junit



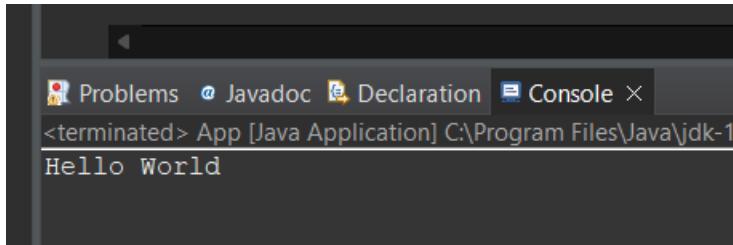
Nuestra prueba se ejecutó de forma exitosa. [Entregar captura]



Ahora que ya probamos nuestro código con una prueba unitaria, podemos ejecutar nuestro programa con la certeza de que funcionará correctamente.



Resultado exitoso.



5. Como paso final debemos subir nuestro proyecto a un repositorio de github, si no recuerda cómo hacerlo, repasar la parte de versionamiento con git.

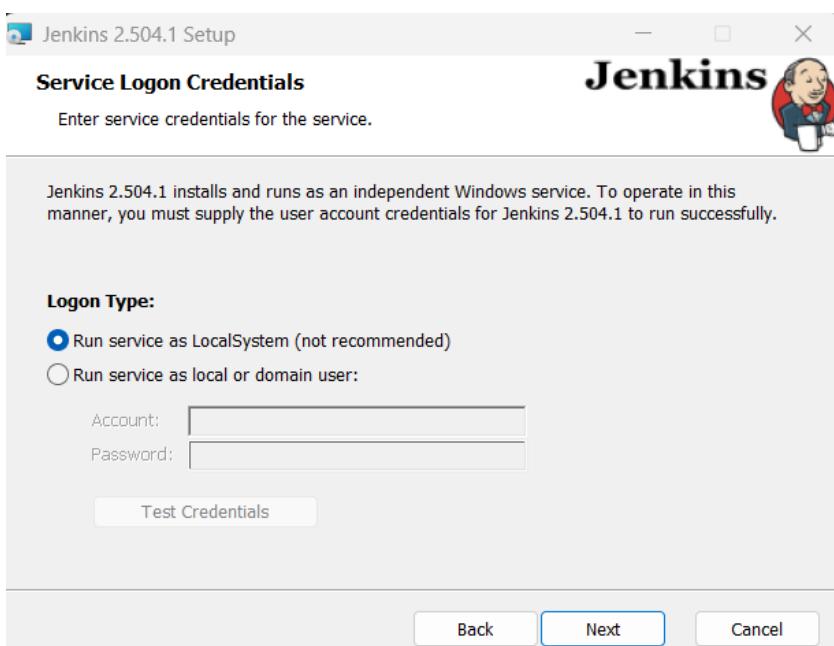
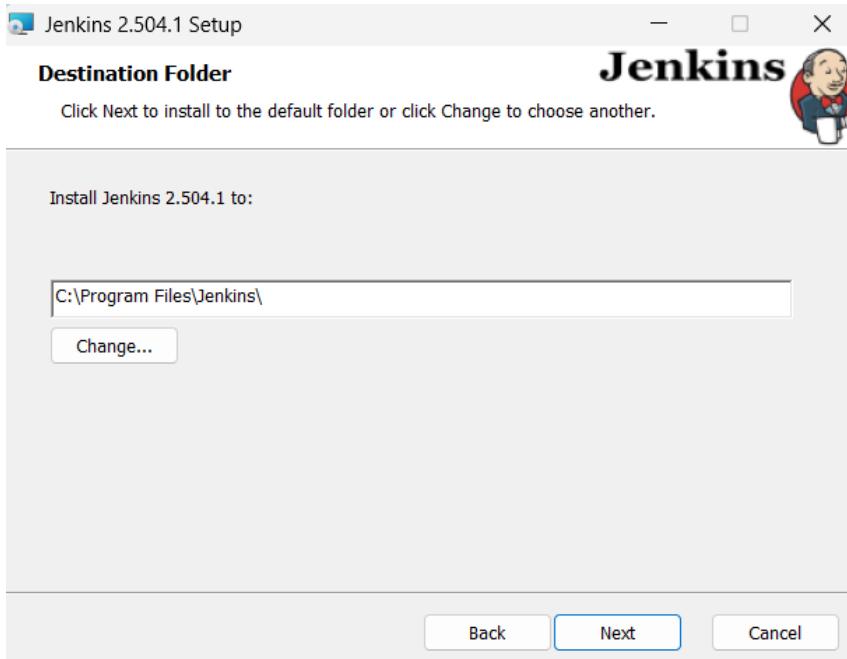
## PARTE IV: Instalando Jenkins

1. Descargar desde <https://www.jenkins.io/download/#downloading-jenkins>

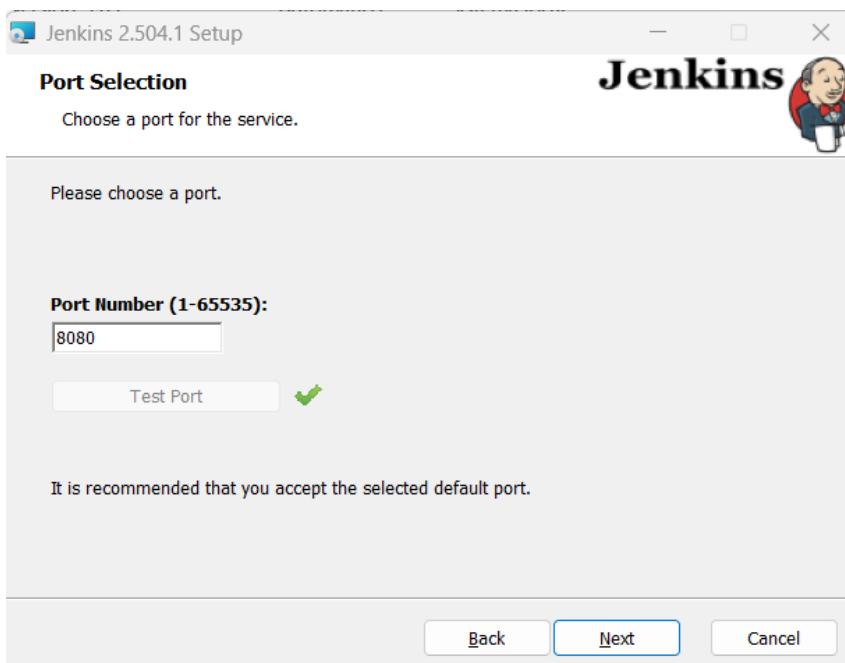
The screenshot shows the Jenkins download page with two main sections: "Download Jenkins 2.504.1 LTS for:" and "Download Jenkins 2.508 for:". Both sections offer a "Generic Java package (.war)" option. The left section includes links for Docker, Kubernetes, Ubuntu/Debian, Red Hat Enterprise Linux and derivatives, Fedora, and Windows (which is circled in red). The right section includes links for Docker, Ubuntu/Debian, Red Hat Enterprise Linux and derivatives, Fedora, Windows, openSUSE, Arch Linux (labeled as "Third party"), FreeBSD (labeled as "Third party"), Gentoo (labeled as "Third party"), and macOS (labeled as "Third party").

2. Instalar Jenkins

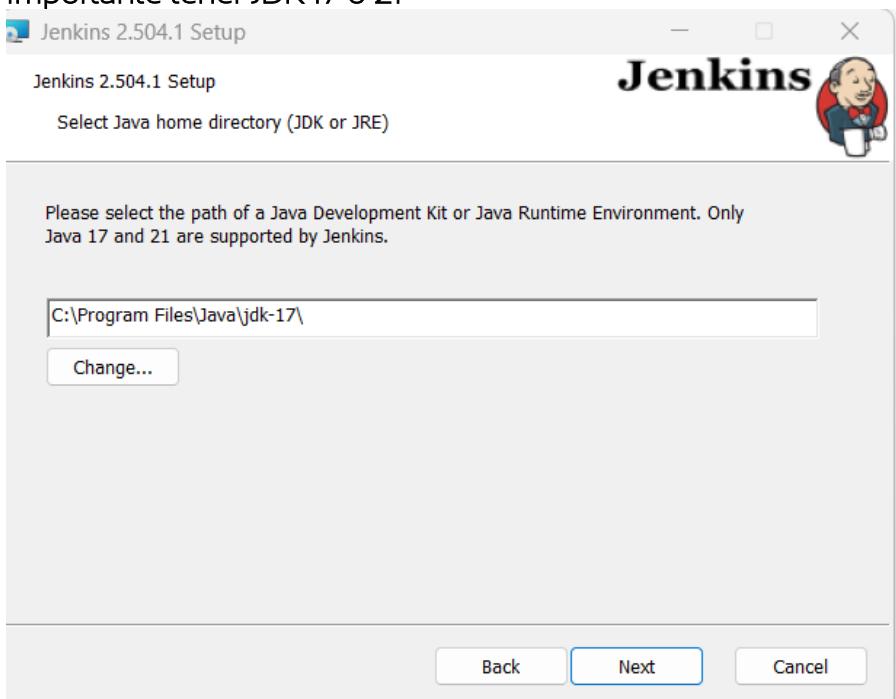


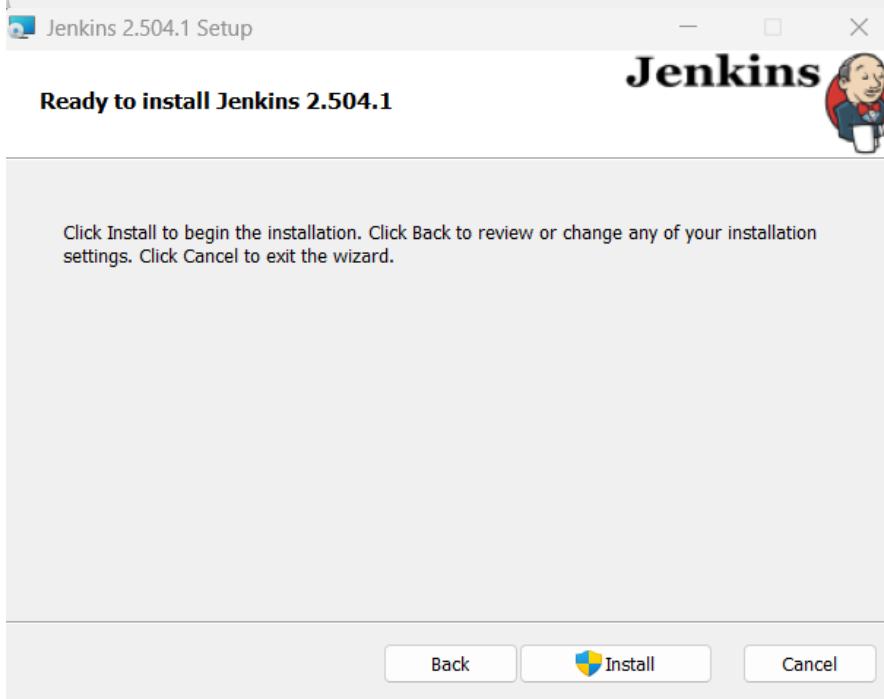
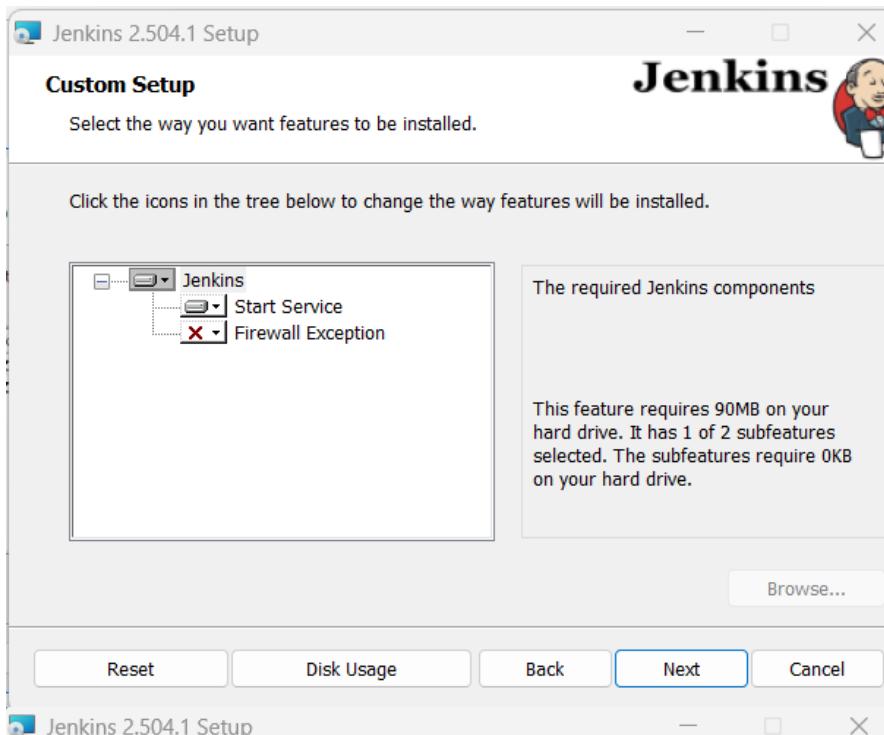


Testeo de puerto, si está ocupado, probar con un puerto diferente



Importante tener JDK 17 o 21





## Completed the Jenkins 2.440.3 Setup Wizard

Click the Finish button to exit the Setup Wizard.



Back      **Finish**      Cancel

Ir a los servicios de Windows e iniciar el servicio de Jenkins si no está iniciado

The screenshot shows the Windows Services snap-in. The left pane lists services by category: 'Servicios (locales)'. The right pane displays a list of services with Jenkins selected. Jenkins is listed with the following details:

Nombre	Descripción	Estado	Tipo de inicio	Iniciar ses...
Jenkins	Jenkins Auto...	En ejecución	Automático	Sistema loc...
Juegos guardados en Xbox L...	Este servicio ...		Manual (desen...	Sistema loc...
KTMRM para DTC (Coordina...	Coordina tra...		Manual (desen...	Servicio d...

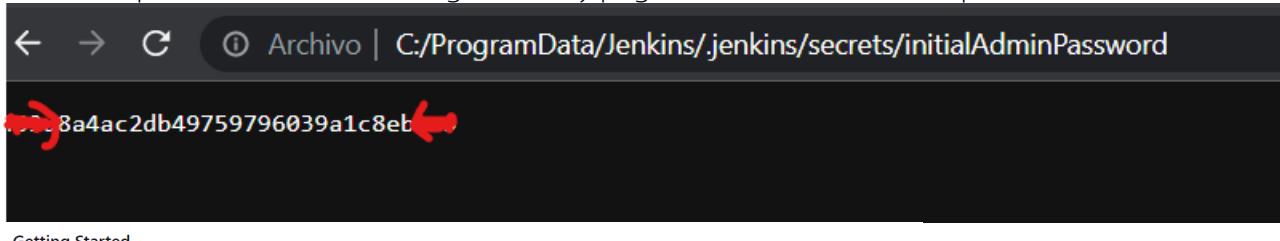
Below the table, there are two buttons: 'Detener el servicio' and 'Reiniciar el servicio'.

3. Iniciar desde navegador: <http://localhost:8080/login?from=%2F>, si configuro un puerto diferente, cambiarlo.

Si ya tenemos una cuenta, simplemente ingresar usuario y contraseña, de lo contrario seguir los pasos siguientes:

The screenshot shows the 'Getting Started' screen of the Jenkins setup wizard. The title is 'Unlock Jenkins'. It instructs the user to ensure Jenkins is securely set up by an administrator, with a password written to the log and a file on the server. The path 'C:\ProgramData\Jenkins\.jenkins\secrets\initialAdminPassword' is highlighted in red. Below this, it says 'Please copy the password from either location and paste it below.' A text input field labeled 'Administrator password' is shown, with a placeholder 'Enter password here'. At the bottom right is a 'Continue' button.

4. Copiar el AdminPassword generado y pegarlo en el Administrator password.



```
← → ⌂ Archivo | C:/ProgramData/Jenkins/jenkins/secrets/initialAdminPassword
...8a4ac2db49759796039a1c8et...
```

Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

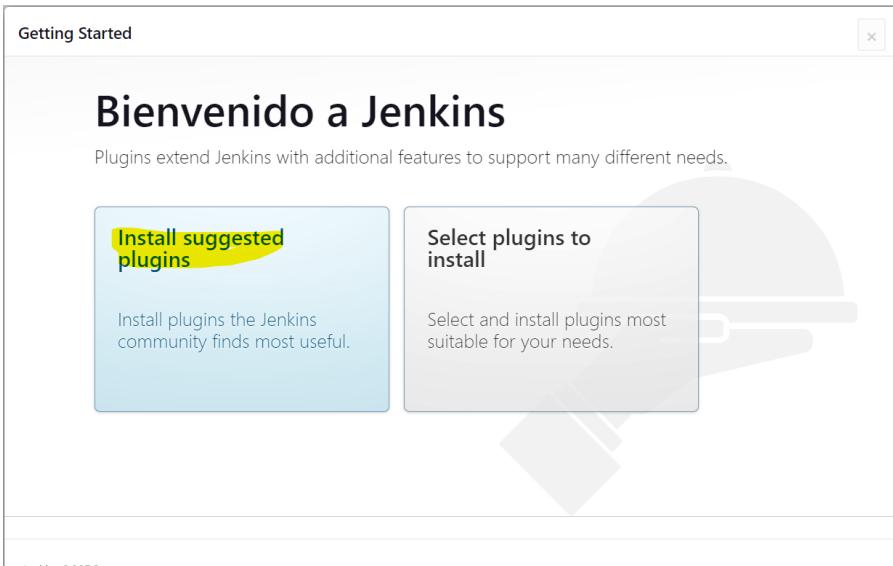
C:\ProgramData\Jenkins\.jenkins\secrets\initialAdminPassword

Please copy the password from either location and paste it below.

Administrator password

.....

Continue



The screenshot shows the Jenkins 'Getting Started' page. At the top, there's a green header bar with the title 'Getting Started'. Below it is a large section titled 'Available Plugins' with a sub-section 'Folders' highlighted in green. A table lists various Jenkins plugins:

✓ Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding	** Ionicons API Folders ** bouncycastle API
Timestamper	Workspace Cleanup	Ant	Gradle	
Pipeline	Github Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	
LDAP	Email Extension	Mailer		

At the bottom right of the table area, there's a note: '\*\* - required dependency'.

5. Se puede crear un usuario y contraseña para loguearse cada vez que iniciemos sesión o seguir como administrador.

The screenshot shows the 'Create First Admin User' form. It has several input fields:

- Usuario:** etrigner
- Contraseña:** (redacted)
- Confirma la contraseña:** (redacted)
- Nombre completo:** Erick Adiel Trigueros Jerez
- Dirección de email:** (redacted)

At the bottom, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

## Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.387.3

Not now

**Save and Finish**

### Getting Started

## Jenkins is ready!

Your Jenkins setup is complete.

**Start using Jenkins**

6. Podemos ver el panel principal de Jenkins, listo para configurar un proyecto.

(Entregar captura donde se vea su nombre)

The screenshot shows the Jenkins dashboard at <http://localhost:8080>. The top navigation bar includes the Jenkins logo, a search bar, and user information for 'Erick Adiel Trigueros Jerez'. The main content area displays the 'My.mavenproject' job card, which has a green status icon, a yellow sun icon, and the name 'My.mavenproject'. Below the card, there are links for 'Último Éxito' (30 Min), 'Último Fallo' (N/D), and 'Última Duración' (23 Seg). On the left sidebar, there are links for 'Nueva Tarea', 'Personas', 'Historial de trabajos', 'Relacion entre proyectos', 'Comprobar firma de archivos', 'Administrar Jenkins', and 'Mis vistas'. A 'Trabajos en la cola' section indicates 'No hay trabajos en la cola'. Another section titled 'Estado del ejecutor de construcciones' shows '1 Inactivo' and '2 Inactivo'. At the bottom right, there are links for 'REST API' and 'Jenkins 2.440.3'.

## PARTE V: Maven and Jenkins integration.

### 1. Configurar el plugin de maven

The screenshot shows the Jenkins administration interface at `localhost:8080/manage/`. The left sidebar includes links for 'Nueva Tarea', 'Personas', 'Historial de trabajos', and 'Administrador Jenkins' (which is highlighted with a yellow box). Under 'Trabajos en la cola', it says 'No hay trabajos en la cola'. A dropdown for 'Estado del ejecutor de construcciones' shows '1 Inactivo' and '2 Inactivo'. The main content area is titled 'Administrador Jenkins' and contains a warning about building on the built-in node. It features two sections: 'System Configuration' and 'Security'. In 'System Configuration', there are three items: 'Configurar el Sistema' (Configure System), 'Global Tool Configuration' (Configure tools, their locations and automatic installers), and 'Administrador Plugins' (Manage Plugins) which is also highlighted with a yellow box. In 'Security', there are two items: 'Configuración global de la seguridad' (Global Security Configuration) and 'Credentials' (Configure credentials). A search bar at the top right says 'búsqueda (CTRL+K)'.

localhost:8080/manage/pluginManager/available

## Plugins

Available plugins

maven

Install	Name
<input checked="" type="checkbox"/>	Maven Integration 3.22 Plugins relacionados con la forma de ejecutar trabajos This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic the automated configuration of various Jenkins publishers such as Junit.
<input type="checkbox"/>	Config File Provider 938.ve2b_8a_591c596 Groovy-related Plugins de integración con sitios y herramientas externas Plugins para Maven Ability to provide configuration files (e.g. settings.xml for maven, XML, groovy, custom files,...) loac
<input type="checkbox"/>	Jira 3.9 Plugins de integración con sitios y herramientas externas Plugins para Maven jira This plugin integrates Jenkins to Atlassian Jira.
<input type="checkbox"/>	Pipeline Maven Integration 1298.v43b_82f220a_e9 pipeline Plugins para Maven This plugin provides integration with Pipeline, configures maven environment to use within a pipel installation will be configured and prepended to the path.
<input type="checkbox"/>	Cobertura 1.17 Javadoc JSch dependency Maven Integration Loading plugin extensions  Install without restart Download now and install after restart Update information obtained: 32 Min ago  <input checked="" type="checkbox"/> Actualizado <input checked="" type="checkbox"/> Actualizado <input checked="" type="checkbox"/> Actualizado <input checked="" type="checkbox"/> Success

→ [Volver al inicio de la página](#)  
(puedes empezar a usar los plugins instalados inmediatamente)

→  Reiniciar Jenkins cuando termine la instalación y no queden trabajos en ejecución

Ya tenemos el plugin de maven...después de reiniciar jenkins nos logueamos.



## Welcome to Jenkins!

etriguer

.....

Keep me signed in

**Sign in**

2. Ahora vamos a configurar la ruta de maven, para ello tenemos que tener instalado maven de forma local, podemos hacer la descarga desde <https://maven.apache.org/download.cgi> su versión puede variar respecto a la de la guía

maven.apache.org/download.cgi

## Downloading Apache Maven 3.9.6

Apache Maven 3.9.6 is the latest release: it is the recommended version for all users.

### System Requirements

Java Development Kit (JDK)	Maven 3.9+ requires JDK 8 or above to execute. It still allows you to build against 1.3 and other JDK versions by using toolchains.
Memory	No minimum requirement
Disk	Approximately 10MB is required for the Maven installation itself. In addition to that, disk space will be used for your local Maven repository. The size of your local repository will vary depending on usage but expect at least 500MB.
Operating System	No minimum requirement. Start up scripts are included as shell scripts (tested on many Unix flavors) and Windows batch files.

### Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [Installation instructions](#). Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public [KEYS](#) used by the Apache Maven developers.

Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.6-bin.tar.gz	apache-maven-3.9.6-bin.tar.gz.asc
<b>Binary zip archive</b>	<a href="#">apache-maven-3.9.6-bin.zip</a>	apache-maven-3.9.6-bin.zip.asc
Source tar.gz archive	apache-maven-3.9.6-src.tar.gz	apache-maven-3.9.6-src.tar.gz.asc
Source zip archive	apache-maven-3.9.6-src.zip	apache-maven-3.9.6-src.zip.asc

3. Vamos a descomprimir el binario y copiar la ruta.

Este equipo > Windows (C:) > apache-maven-3.9.6 >

Nombre	Fecha de modificación	Tipo	Tamaño
bin	28/4/2024 23:37	Carpeta de archivos	
boot	28/4/2024 23:37	Carpeta de archivos	
conf	28/4/2024 23:37	Carpeta de archivos	
lib	28/4/2024 23:37	Carpeta de archivos	
LICENSE	28/4/2024 23:36	Archivo	19 KB
NOTICE	28/4/2024 23:36	Archivo	5 KB
README	28/4/2024 23:36	Documento de tex...	3 KB

4. Volvemos a Jenkins e instalamos y configururamos el plugin de maven(su versión puede variar respecto a la de la guía):

Stored XSS vulnerability  
A fix for this issue is available. Go to the [plugin manager](#) to update the plugin.

Folders Plugin 6.815.v0dd5a\_cb\_40e0e:  
CSRF vulnerability  
CSRF vulnerability may approve unsandboxed scripts  
Information disclosure  
Fixes for all of these issues are available. Go to the [plugin manager](#) to update the plugin.

**System Configuration**

**System**  
Configurar variables globales y rutas.

**Tools**  
Configure tools, their locations and automatic installers.

Panel de Control > Administrar Jenkins > Tools

instalaciones de Maven

Añadir Maven

**Maven**

Nombre: Maven

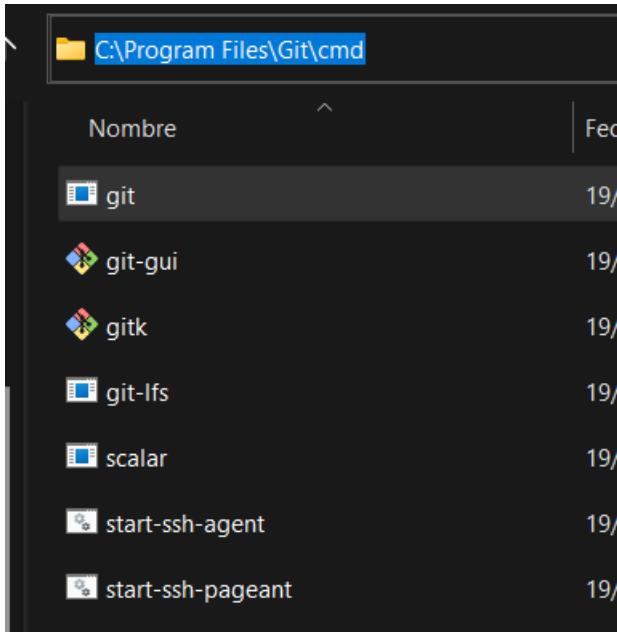
MAVEN\_HOME: C:\apache-maven-3.9.0

Instalar automáticamente

**Añadir Maven**

**Save** **Apply**

5. Ahora vamos a configurar la ruta de git, para ello vamos a donde tenemos instalado git y los copiamos.



6. Volvemos a Jenkins y configuramos el plugin de git:

localhost:8080/manage/

## Jenkins

Panel de Control > Administrar Jenkins

+ Nueva Tarea

Personas

Historial de trabajos

Administrador Jenkins

Mis vistas

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

Administrador Jenkins

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).

System Configuration

Configurar el Sistema

Configurar variables globales y rutas.

Global Tool Configuration

Configure tools, their locations and automatic installers.

para ello pegamos la ruta de instalación de git.exe

The screenshot shows the Jenkins Global Tool Configuration page at [localhost:8080/manage/configureTools/](http://localhost:8080/manage/configureTools/). The 'Git installations' section is highlighted. A 'Name' field is set to 'Default'. The 'Path to Git executable' field contains 'C:\Program Files\Git\cmd\git.exe'. A checkbox for 'Instalar automáticamente' is unchecked. Two buttons at the bottom, 'Save' and 'Apply', are circled in red.

7. Ahora vamos a crear la tarea de integración.



**Nueva Tarea**

Personas

Historial de trabajos

Relacion entre proyectos

Comprobar firma de archivos

Administrar Jenkins

Mis vistas

To

Tc

C

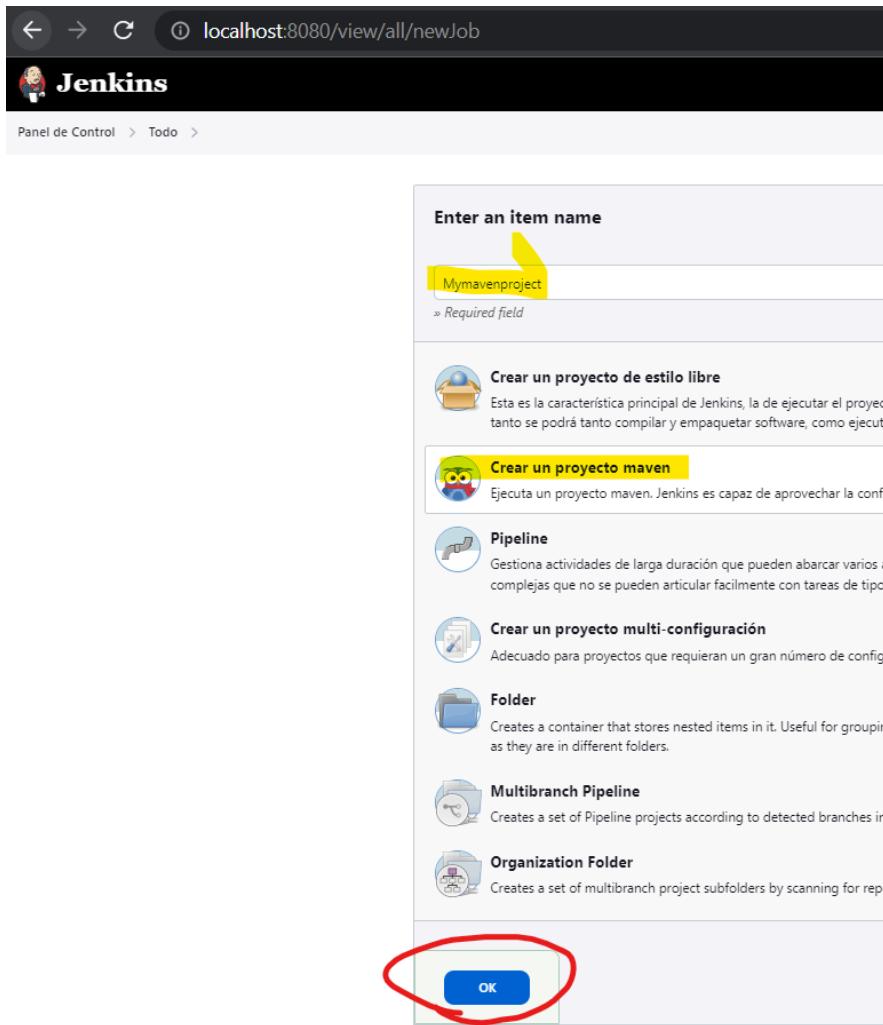
C

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

1 Inactivo



8. Ahora vamos a configurar nuestro Job de maven, entre las configuraciones más importantes tenemos:

## General

Descripción

[Plain text] [Visualizar](#)

Desechar ejecuciones antiguas [?](#)

Strategy

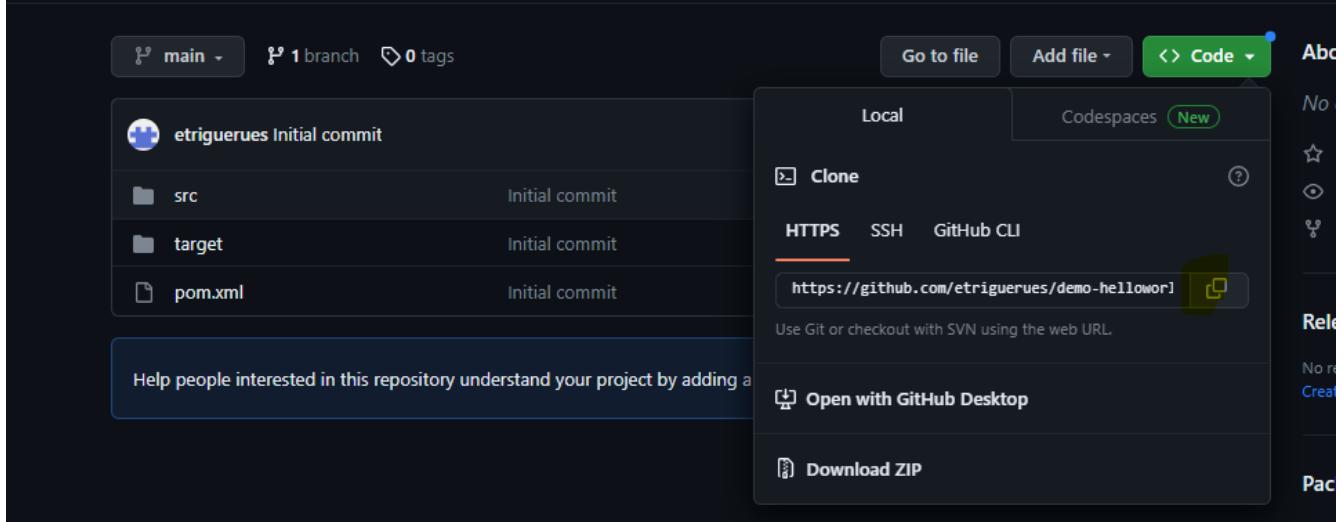
Log Rotation

Numero de días para mantener ejecuciones de proyectos  
si no está vacío, sólo se mantendrán las ejecuciones con una edad inferior a este número

Número máximo de ejecuciones para guardar  
si no está vacío, sólo se guardarán un número de ejecuciones inferior a este valor

[Avanzado ▾](#)

### 9. Configurar la ruta del proyecto git ([Entregar captura](#))



Git ?

Repositories ?

Repository URL ?  
`https://github.com/etriguerues/demo-helloworld.git`

Credentials ?  
- none -

Add ▾

Avanzado ▾

Add Repository

Branches to build ?

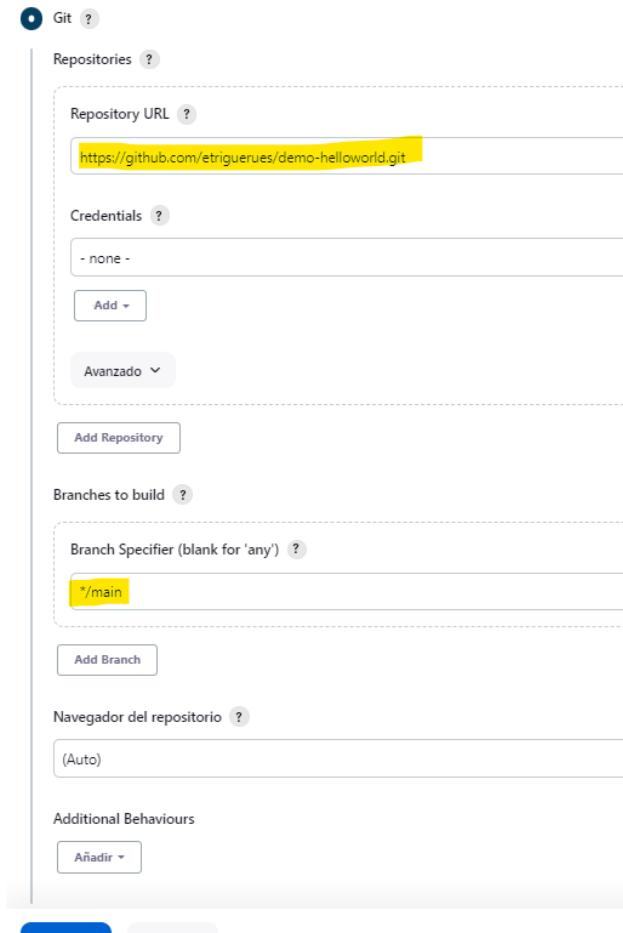
Branch Specifier (blank for 'any') ?  
`*/main`

Add Branch

Navegador del repositorio ?  
(Auto)

Additional Behaviours  
Añadir ▾

Guardar Apply



Si nos da error para acceder a la URL, la solución puede ser configurar un Token de acceso desde git.

## Proyecto

Fichero POM raíz ?  
`pom.xml`

10. Esas serían todas las configuraciones básicas, ahora guardamos y luego construimos el proyecto.

The screenshot shows the Jenkins dashboard for the 'Mymavenproject' job. The top navigation bar includes back, forward, and search icons, and the URL 'localhost:8080/job/Mymavenproject/'. The main header 'Jenkins' has a small icon next to it. Below the header, the breadcrumb navigation shows 'Panel de Control > Mymavenproject >'. A horizontal menu bar contains several items: 'Estado Actual' (selected), 'Cambios', 'Zona de Trabajo', 'Construir ahora' (highlighted with a yellow box), 'Configurar', 'Borrar Maven project', 'Módulos', and 'Rename'. The title 'Maven project Mymavenproj' is displayed.

### Maven project Mymavenproj

</> Cambios

📁 Zona de Trabajo

▷ Construir ahora

#### Enlaces permanentes

⚙️ Configurar

🗑️ Borrar Maven project

📄 Módulos

✍️ Rename

☀️ Historia de tareas

Tendencia ▾

11. Podemos ver que se comenzó la construcción de nuestro proyecto.

The screenshot shows the 'Historia de tareas' (History of tasks) section for the 'Mymavenproject' job. The top navigation bar is identical to the previous screenshot. The main content area displays a table of build history:

Nº	Estatus	Fecha/Hora
1	En ejecución	3 may 2025 21:52

☰ Estado Actual

</> Cambios

📁 Zona de Trabajo

▷ Construir ahora

⚙️ Configurar

🗑️ Borrar Maven project

📄 Módulos

✍️ Rename

☀️

Historia de tareas

Tendencia ▾

🔍 Filter...

/

🕒 #5

3 may 2025 21:52

12. Ver el estatus de nuestro proyecto. (Click en el número de la construcción)

The screenshots show the Jenkins dashboard and a specific job named "Mymavenproject".

- Screenshot 1:** Jenkins dashboard with "Nueva Tarea" selected. A table shows the last execution of "Mymavenproject" was 10 minutes ago.
- Screenshot 2:** Job details for "Mymavenproject". It shows the last build was successful (#5, 3 may 2025 21:52:00). The build log indicates no changes were made.
- Screenshot 3:** Build details for #5. It shows the build was initiated by "Erick Adiel Trigueros Jerez" and used a Git repository at <https://github.com/etriguerues/demo-helloworld.git>. The build results show no test failures. A module named "demo-helloworld" is highlighted with a red circle.

### 13. Y lo podemos monitorear en consola.

The screenshot shows the Jenkins console output for build #3 of the "demo-helloworld" project. The output displays the Maven build process, including dependency resolution and plugin execution. A portion of the output is highlighted in yellow.

```

Established TCP socket on 49930
<====[JENKINS REMOTING CAPACITY]==>channel started
Executing Maven: -B -f C:\ProgramData\Jenkins\jenkins\workspace\Mymavenproject\pom.xml install
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] < com.lis.helloworld:demo-helloworld >
[INFO] Building demo-helloworld 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] -----
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.pom (8.2 kB at 12 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom (8.1 kB at 58 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.jar (31 kB at 89 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.11.0/maven-compiler-plugin-3.11.0.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.11.0/maven-compiler-plugin-3.11.0.pom (9.8 kB at 75 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.11.0/maven-compiler-plugin-3.11.0.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.11.0/maven-compiler-plugin-3.11.0.jar (66 kB at 413 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-surefire-plugin/3.2.2/maven-surefire-plugin-3.2.2.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-surefire-plugin/3.2.2/maven-surefire-plugin-3.2.2.pom (5.1 kB at 44 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire/3.2.2/surefire-3.2.2.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire/3.2.2/surefire-3.2.2.pom (21 kB at 147 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/40/maven-parent-40.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/40/maven-parent-40.pom (49 kB at 347 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/30/apache-30.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/30/apache-30.pom (23 kB at 171 kB/s)

```

Los test se ejecutaron de forma correcta

```
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.iis.helloworld.AppTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.034 s - in com.iis.helloworld.AppTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[JENKINS] Guardando informes de test
[INFO]
```

Y la creación de nuestro .jar también

```
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.iis.helloworld.AppTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.036 s -- in com.iis.helloworld.AppTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[JENKINS] Guardando informes de test
[INFO]
[INFO] --- jar:3.3.0:jar (default-jar) @ demo-helloworld ---
[INFO] Building jar: C:\ProgramData\Jenkins\workspace\My.mavenproject\target\demo-helloworld-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- install:3.1.1:install (default-install) @ demo-helloworld ---
[INFO] Installing C:\ProgramData\Jenkins\workspace\My.mavenproject\pom.xml to C:\WINDOWS\system32\config\systemprofile\.m2\repository\com\iis\helloworld\demo-helloworld\0.0.1-SNAPSHOT\demo-helloworld-0.0.1-SNAPSHOT.pom
[INFO] Installing C:\ProgramData\Jenkins\workspace\My.mavenproject\target\demo-helloworld-0.0.1-SNAPSHOT.jar to C:\WINDOWS\system32\config\systemprofile\.m2\repository\com\iis\helloworld\demo-helloworld\0.0.1-SNAPSHOT\demo-helloworld-0.0.1-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 16.358 s
```

Y podemos ver nuestro workspace:

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with links like 'Nueva Tarea', 'Personas', 'Historial de trabajos', 'Relacion entre proyectos', 'Comprobar firma de archivos', 'Administrar Jenkins', and 'Mis vistas'. Below that is a section for 'Trabajos en la cola' which says 'No hay trabajos en la cola'. Under 'Estado del ejecutor de construcciones', it shows '1 Inactivo' and '2 Inactivo'. On the right, a context menu is open for the 'Mymavenproject' job, listing options: 'Cambios', 'Zona de Trabajo' (which is highlighted), 'Construir ahora', 'Configurar', 'Borrar Maven project', 'Módulos', and 'Rename'.

The screenshot shows the Jenkins workspace for the 'Mymavenproject' job. The URL in the browser is 'localhost:8080/job/Mymavenproject/ws/target/'. The left sidebar has links for 'Estado Actual', 'Cambios', 'Zona de Trabajo', 'Construir ahora', 'Configurar', 'Borrar Maven project', 'Módulos', and 'Rename'. The main area is titled 'Workspace of Mymavenpr...' and shows a file tree for the 'target' directory. The tree includes 'classes', 'generated-sources/annotations', 'generated-test-sources/test-annotations', 'maven-archiver', 'maven-status/maven-compiler-plugin', 'surefire-reports', 'test-classes/com/iis/helloworld', and 'demo-helloworld-0.0.1-SNAPSHOT.jar'. At the bottom, there's a link '(Descargar arch...)' and a download icon. A footer bar at the bottom has 'Historia de tareas', 'Tendencia', a search bar with 'Filter...', and a '/' button.