

**Tipo** : Guía de laboratorio  
**Capítulo** : Lenguaje de programación Java  
**Duración** : 60 minutos

---

## I. OBJETIVO

Crear un repositorio en GitHub para manejo de versiones.

## II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio:

- Netbeans 11
- Navegador internet

## III. EJECUCIÓN DEL LABORATORIO

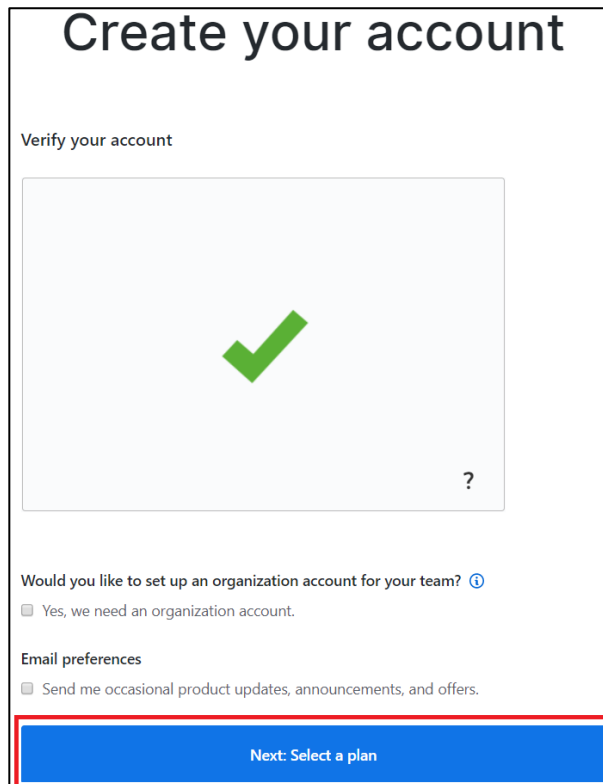
- **Ejercicio: Crear y configurar un repositorio de GitHub para un proyecto Java en Netbeans.**

1. Crea una cuenta en GitHub para poder establecer el contenedor que tendrá el proyecto. Ingresa a la siguiente página.

<https://github.com/>

2. La página solicita un nombre de usuario, un correo y una clave (cumpliendo las reglas de validación), para poder crear la cuenta. Ingresa los datos y presiona sobre el botón **“Sign up forGithub”**.

3. Lo siguiente que solicita la página, es saber si la cuenta será de tipo empresarial o no, para este caso, deja sin seleccionar ninguno de los checks y presiona sobre el botón **"Next: Select a plan"**.



Create your account

Verify your account

Would you like to set up an organization account for your team? ⓘ

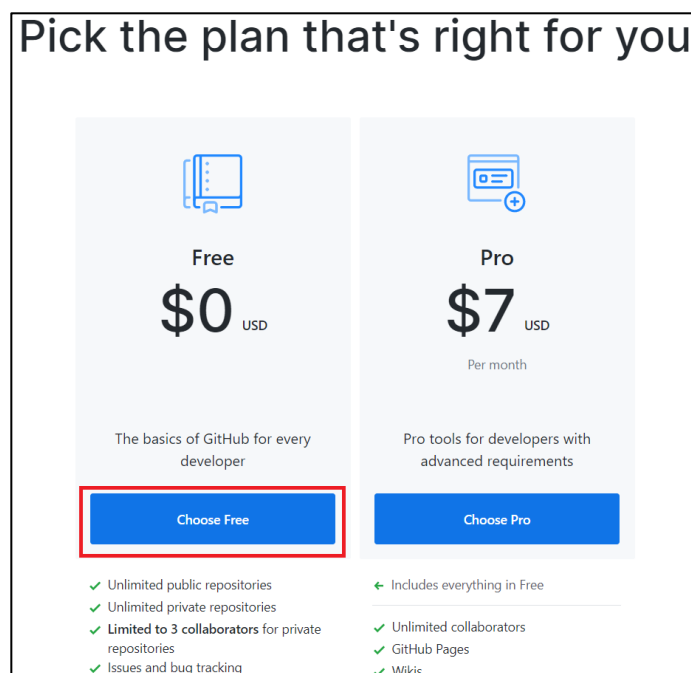
☐ Yes, we need an organization account.

Email preferences



☐ Send me occasional product updates, announcements, and offers.

Next: Select a plan

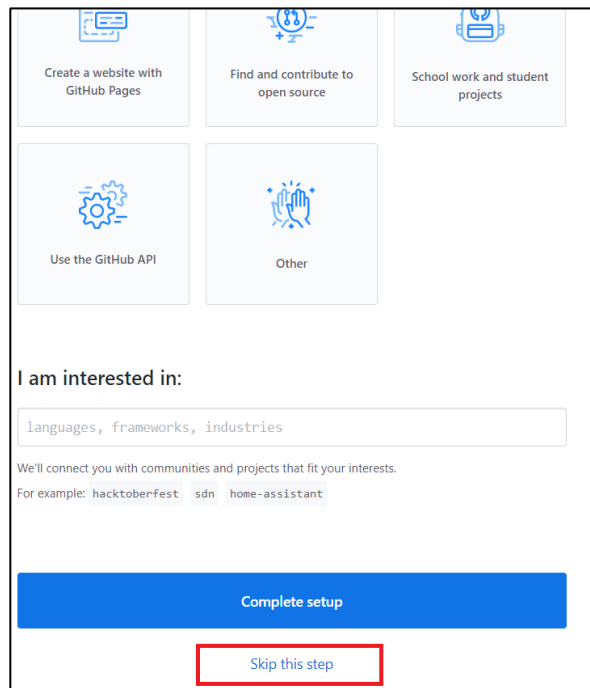
4. La siguiente pantalla solicita elegir el tipo de cuenta. Desde inicios del 2019, las cuentas gratuitas ya pueden crear repositorios privados, pero con máximo 3 colaboradores. En este caso, selecciona cuenta gratuita y presiona sobre el botón **"Choose Free"**.



Pick the plan that's right for you

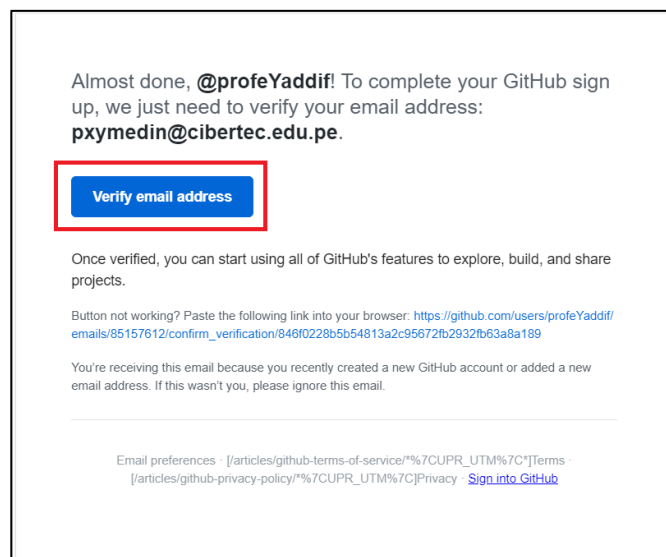
Free	Pro
	
Free	Pro
\$0 USD	\$7 USD
	Per month
The basics of GitHub for every developer	Pro tools for developers with advanced requirements
Choose Free	Choose Pro
<ul style="list-style-type: none"><li>✓ Unlimited public repositories</li><li>✓ Unlimited private repositories</li><li>✓ Limited to 3 collaborators for private repositories</li><li>✓ Issues and bug tracking</li></ul>	<ul style="list-style-type: none"><li>← Includes everything in Free</li><li>✓ Unlimited collaborators</li><li>✓ GitHub Pages</li><li>✓ Wikis</li></ul>

5. Se presenta una pequeña encuesta, la cual se deja pasar. Navega hasta el final de la página y selecciona el link “**Skipthisstep**”.



The image shows a GitHub onboarding survey form. It has five options: 'Create a website with GitHub Pages', 'Find and contribute to open source', 'School work and student projects', 'Use the GitHub API', and 'Other'. Below these is a section 'I am interested in:' with a text input field containing 'languages, frameworks, industries'. There is also a section 'We'll connect you with communities and projects that fit your interests. For example: hacktoberfest sdn home-assistant'. At the bottom, there are two buttons: 'Complete setup' and 'Skip this step' (which is highlighted with a red rectangle).

6. Github envía una confirmación al correo para poder verificar que realmente existe. Ingresa al correo y selecciona el link que se te indica.

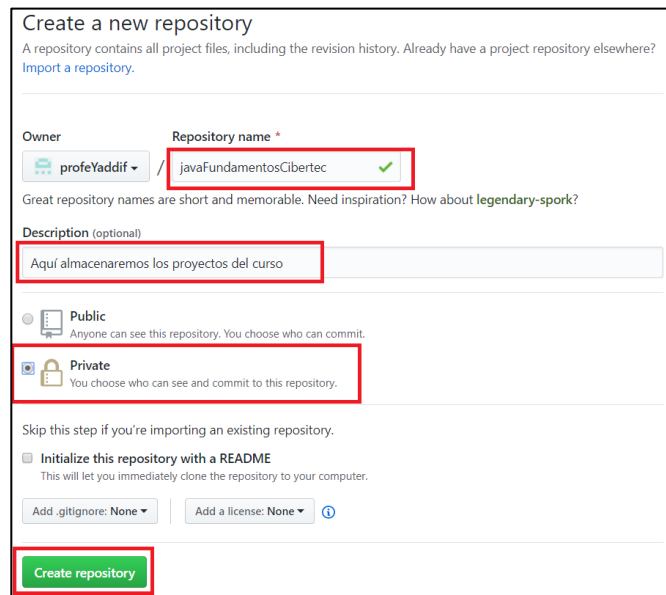


The image shows an email verification page from GitHub. It says 'Almost done, @profeYaddif! To complete your GitHub sign up, we just need to verify your email address: pxymedin@cibertec.edu.pe.' Below this is a button 'Verify email address' (highlighted with a red rectangle). The page also includes instructions on what to do after verification, a link to the verification page, and a note about receiving the email. At the bottom, there are links for 'Email preferences', 'Terms of service', 'Privacy policy', and 'Sign into GitHub'.

7. Al verificar el correo, Github se presenta la página donde se puede crear el repositorio. Repositorio, es donde se almacenarán los archivos del proyecto y que será compartido (si se quiere) con los demás colaboradores del proyecto.

Se solicita el nombre del repositorio (que no necesariamente tiene que ser el nombre del proyecto, aunque se sugiere que así sea), una descripción del proyecto, si el repositorio será público o privado y si tendrá un archivo “Readme” que sirve de instructivo para los colaboradores que se unan al proyecto.

En este caso, define los siguientes valores:



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner: profeYaddif / Repository name: javaFundamentosCibertec ✓

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

Description (optional): Aquí almacenaremos los proyectos del curso

☐ Public  
Anyone can see this repository. You choose who can commit.

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

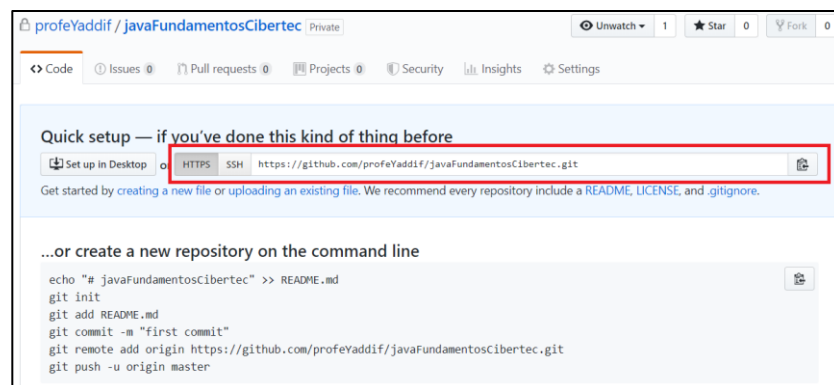
Skip this step if you're importing an existing repository.

☐ Initialize this repository with a README  
This will let you immediately clone the repository to your computer.

Add .gitignore: None Add a license: None ⓘ

Create repository

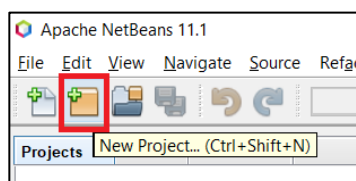
8. Listo, el repositorio está creado y listo para usarse. Es importante recordar la dirección, pues se usará para enlazar el proyecto desde el Netbeans.



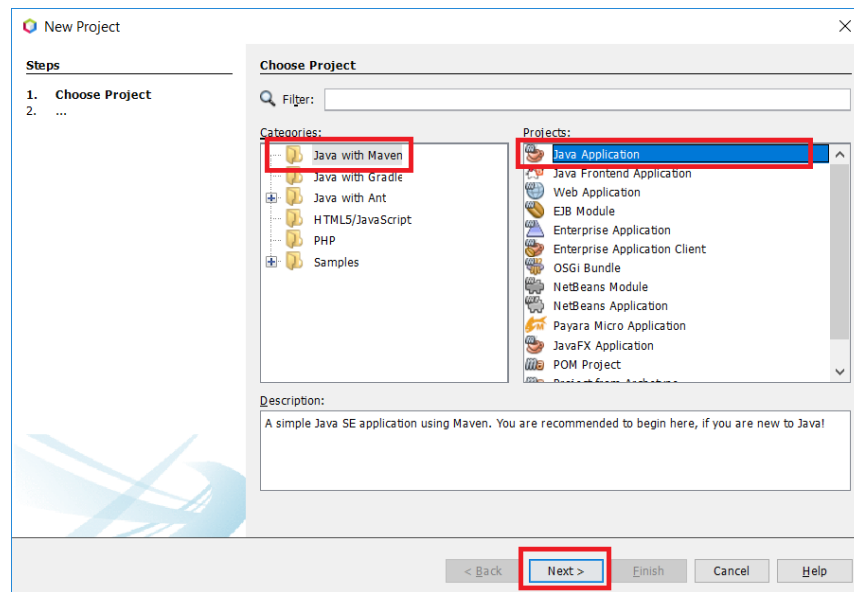
9. Procede a inscribir el repositorio recién creado en el NetBeans, para que puedas trabajarlo desde la herramienta y hacerlo de manera más fácil.

Crea un proyecto del tipo “Java **Application**” y lo trabajarás con GitHub.

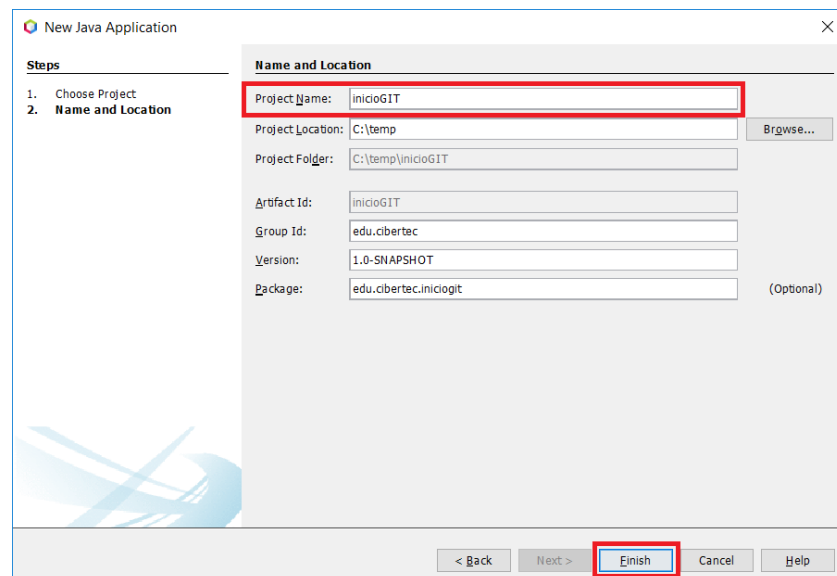
Dentro de NetBeans presiona sobre el botón “**New Project**”.



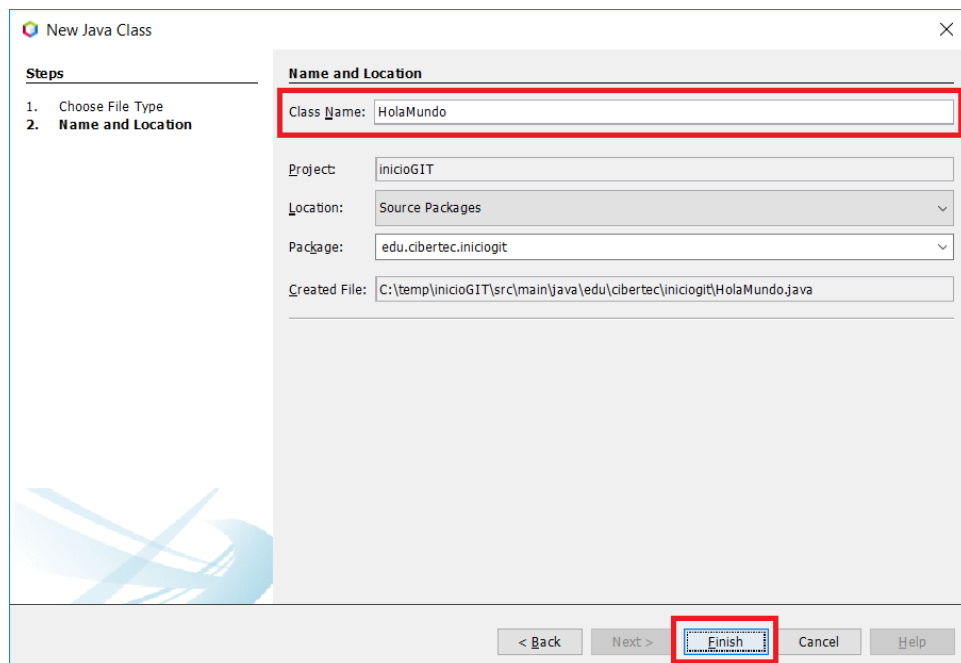
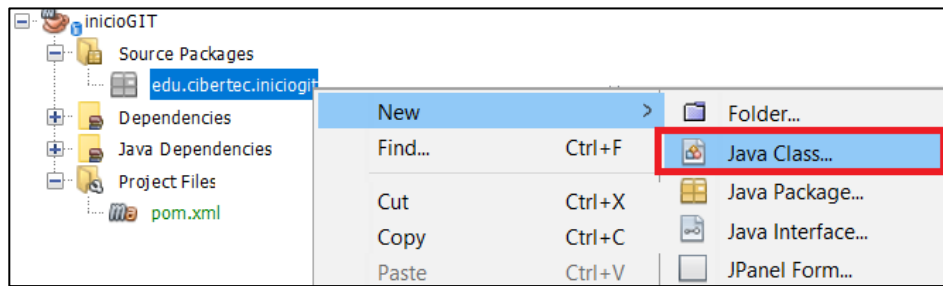
Selecciona la categoría de “Java con Maven” y el tipo de proyecto “Java Application”.



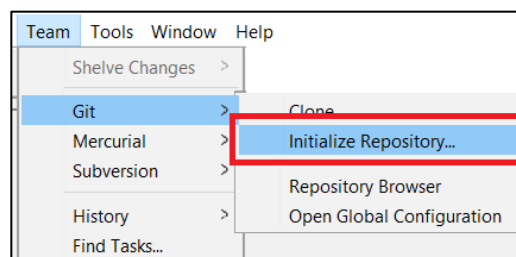
Luego, coloca el nombre del proyecto a “inicioGIT” dentro de la carpeta de trabajo del alumno.



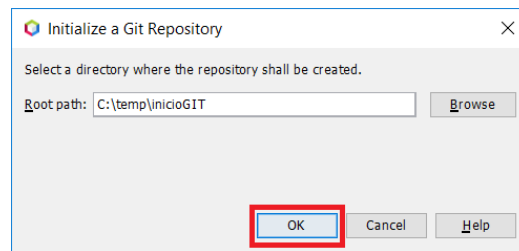
Crea una clase para comprobar cómo se suben los archivos al repositorio. La clase se llamará **“HolaMundo”** y no tendrá funcionalidad.



Procede a inscribir el GIT para que trabaje con el proyecto. Selecciona el menú **“Team” -> “Git” -> “InitializeRepository...”**.

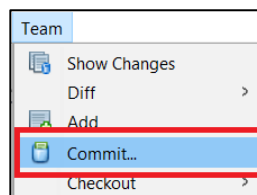


Solicita el directorio del repositorio local donde se trabajará. Como usualmente es el directorio donde están las fuentes, simplemente acepta y haz clic al botón “Ok”.

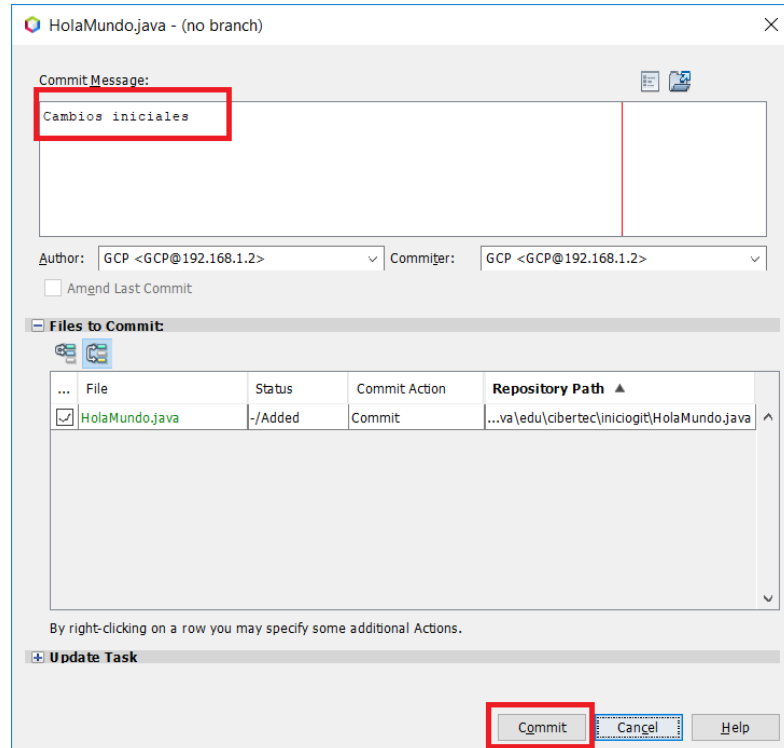


El siguiente paso es realizar un “commit”, para enviar los archivos que se han modificado del proyecto al repositorio local. Como es la primera vez que se realiza el commit, se enviarán todos los archivos creados hasta el momento.

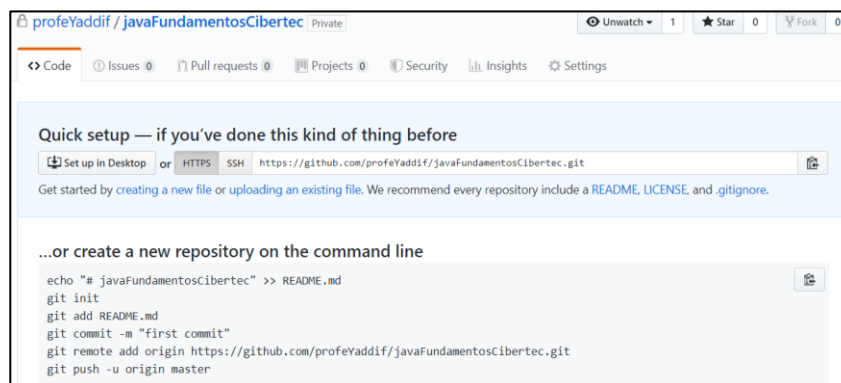
Para realizar el commit ingresa a la opción de menú “Team” -> “Commit”.



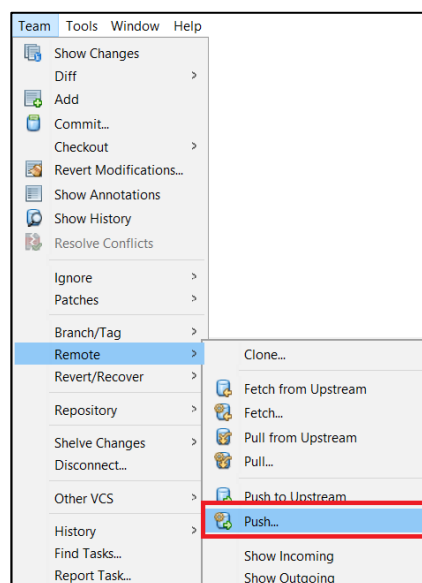
La ventana solicita un comentario de los cambios realizados, pero es opcional.



Observa que si se regresa a la página de GitHub y se actualiza (presionando [F5]) aún no se encuentra el proyecto.

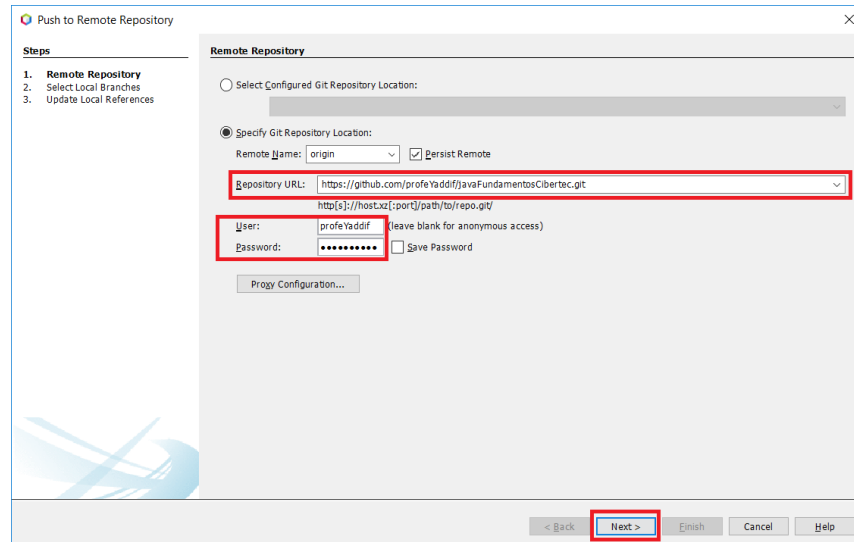


Para enviar los cambios al repositorio remoto (del GitHub), realiza un “push”. Para hacer esto, ingresa al menú **“Team” -> “Remote” -> “Push”**.

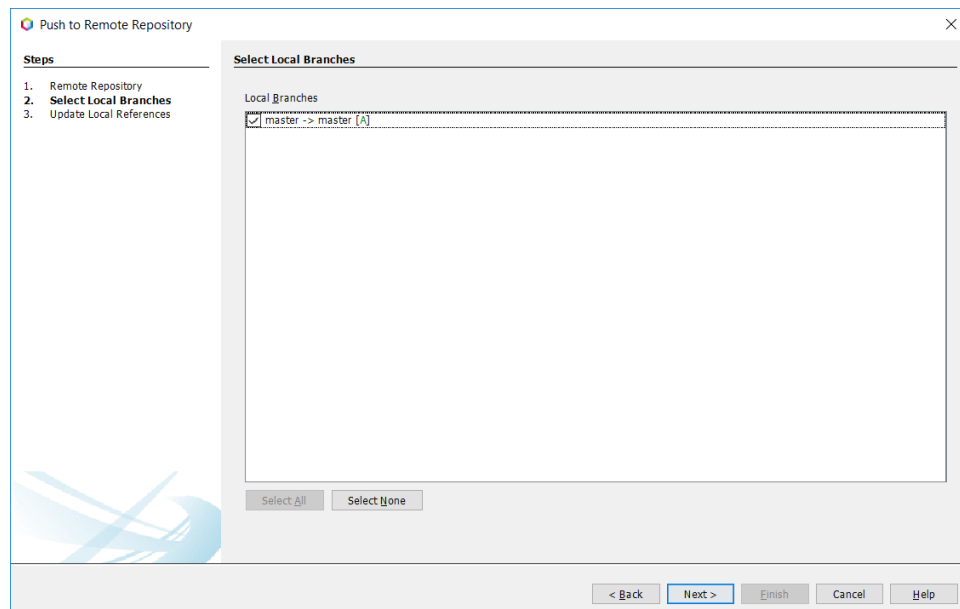




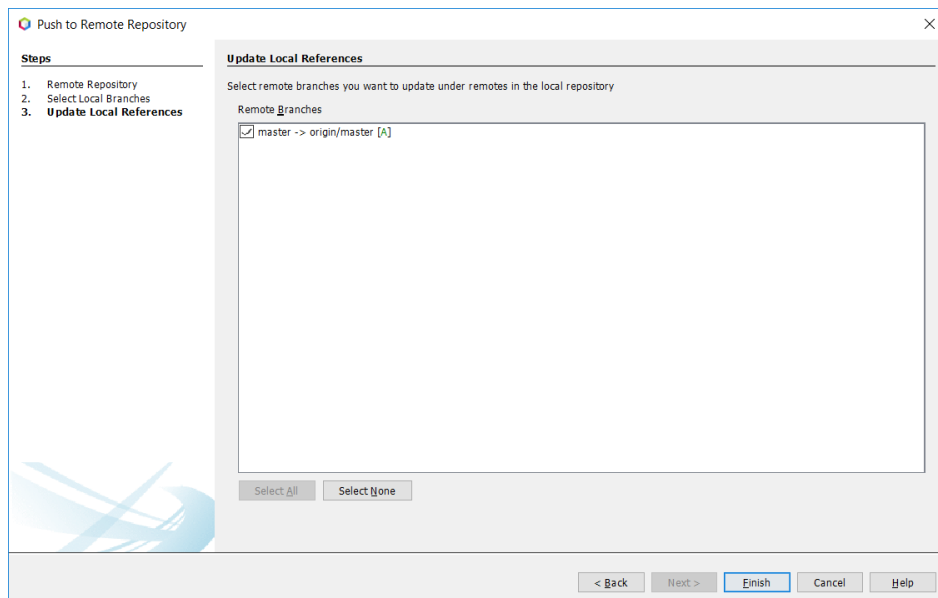
Solicita el URL, el usuario y la clave; por ello, coloca lo que se ha creado en el GitHub.



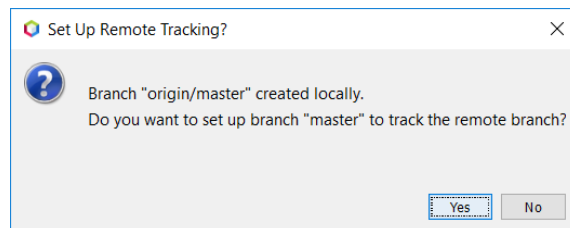
Seguidamente selecciona el branch local (un branch es una rama, eso quiere decir que se puede tener varias copias del repositorio en local para realizar cambios que puede afectar o no al proyecto y subir solo las que estamos seguros que funcionan). En este caso solo se tiene una, el “master”.



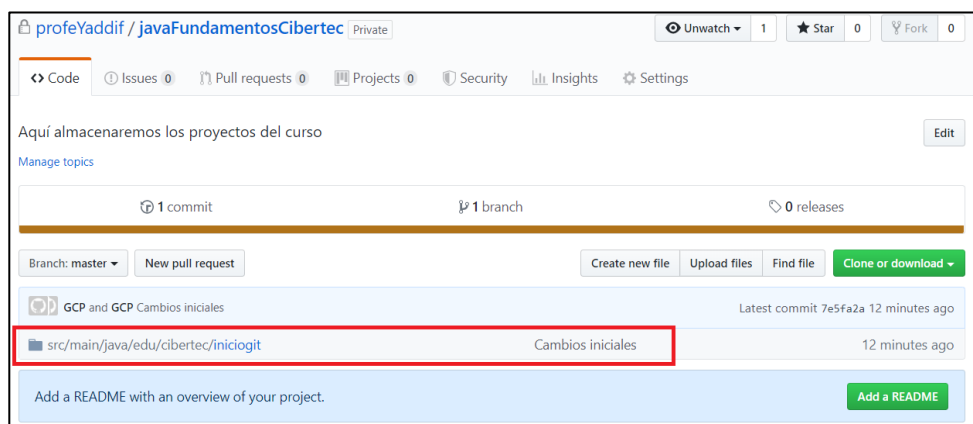
Luego, solicita el branch remoto. Selecciona el que tiene por defecto y presiona sobre el botón **“Finish”**.



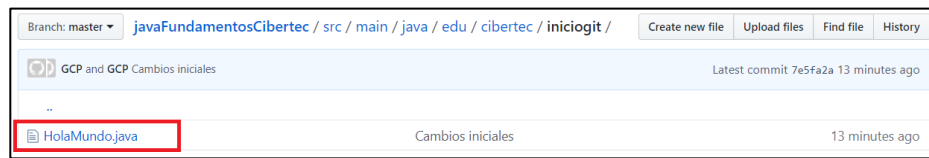
NetBeans advierte de la creación de la rama en local y pregunta si deseas que esta rama quede asociada al repositorio remoto, para no solicitar los datos de conexión cada vez que se desee hacer un push.



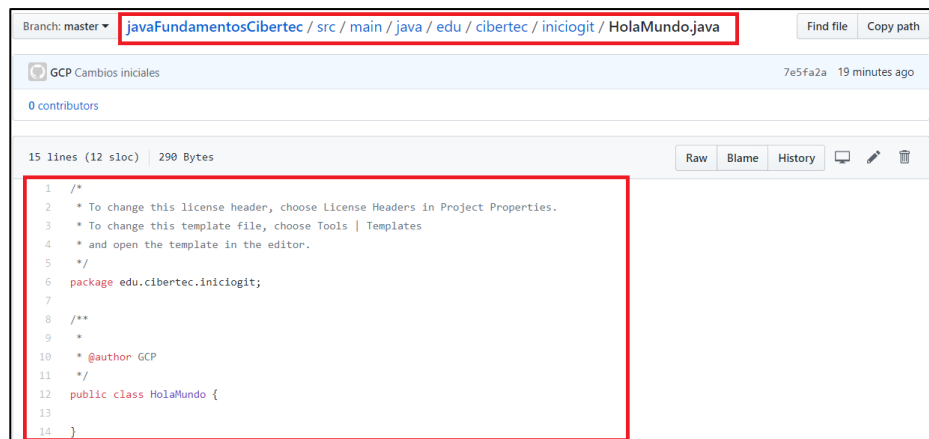
Regresa a la página del GitHub y actualiza [F5]. Se muestra el proyecto que se ha subido y los comentarios que se pusieron en el Commit.



Se puede navegar en el proyecto hasta alcanzar la clase deseada.

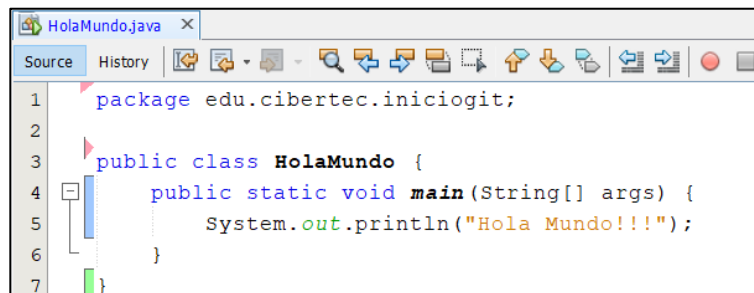


Al dar clic a la **clase**, GitHub muestra su visor con el contenido de la clase.

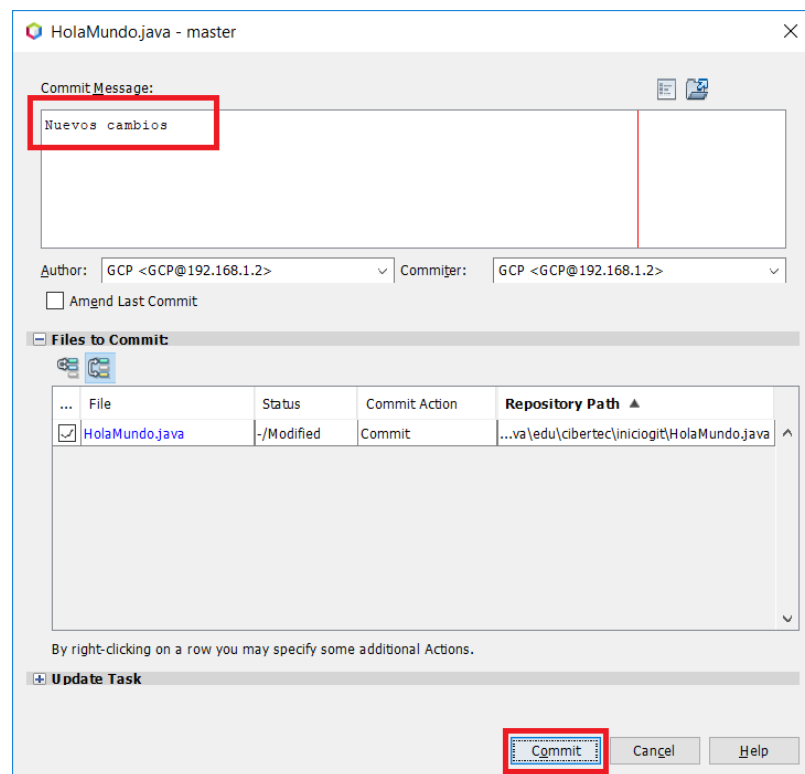


Cada vez que se quiera enviar los cambios al GitHub, se debe realizar un “commit” al repositorio local y un “pull” al repositorio remoto. Esto se realiza usualmente cuando los cambios ya se encuentran probados en local.

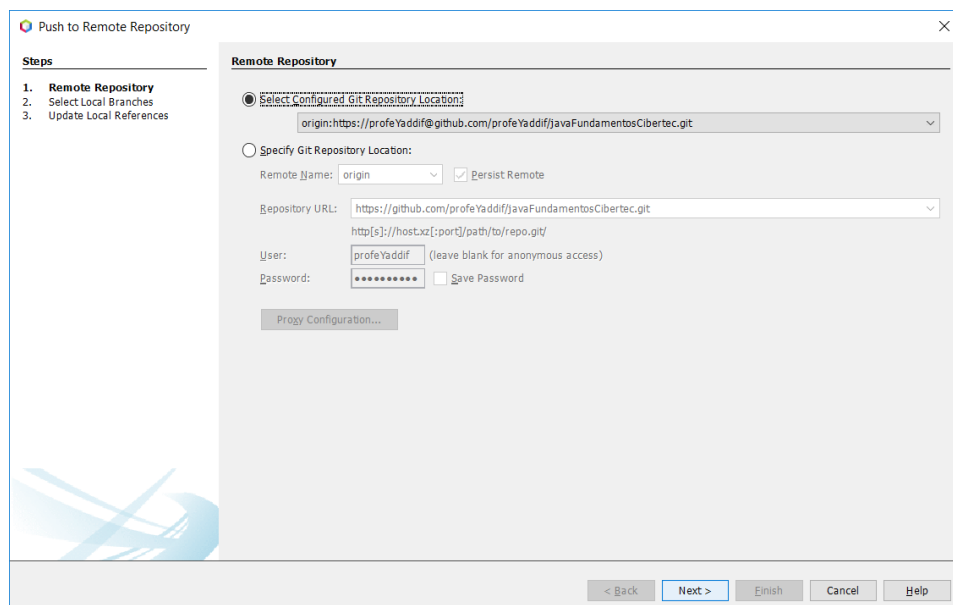
Como ejemplo, modificamos la clase HolaMundo con un código funcional (que no es importante entender en este momento).



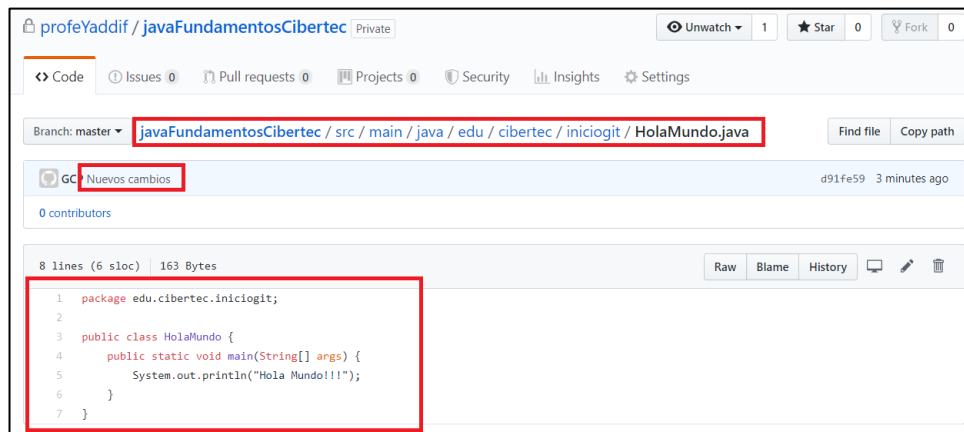
Realiza el commit con los siguientes datos.



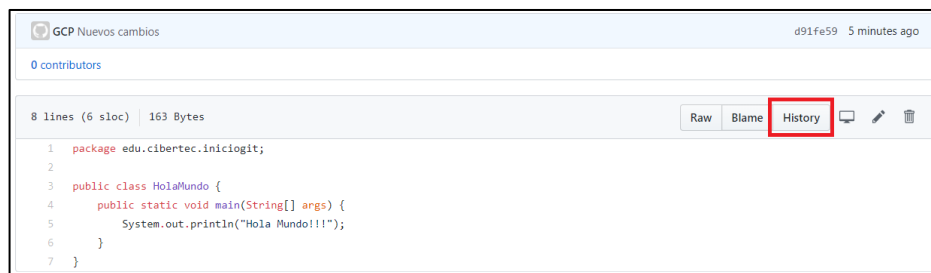
Realiza el push.



Al refrescar la clase en el GitHub, se observa la diferencia.



Si se quisiera ver las versiones que ha tenido la clase, presiona sobre el botón **"History"**.



Al presionar sobre el nombre del cambio se puede observar la clase en cada uno de esos estados.



#### IV. EVALUACIÓN

1. ¿Cuáles son las ventajas de usar un manejador de versiones?

La principal ventaja es que se cuenta con la historia de cómo ha ido evolucionando la codificación en la clase, de esta manera se puede regresar a versiones anteriores si se necesita.

2. ¿Cuáles son las ventajas de usar una herramienta de trabajo colaborativo?

Las ventajas son:

- Que todos los colaboradores trabajan sobre código centralizado y pueden realizar cambios donde todos los involucrados se enteren.
- Puede haber colaboradores remotos.
- Los cambios podrían ser aprobados antes de integrarse a la solución general.