

Preparación previa a la entrega de proyectos



BlueJ y GitHub

GitHub y Git

- **GitHub**
 - es uno de los servicios más populares en la nube de alojamiento de repositorios
 - estos repositorios usan el sistema de control de versiones **Git**
 - no es el único, hay otros como **Bitbucket**
 - <https://github.com/>

GitHub y Git

- **Y qué es Git?**
 - es un sistema de control de versiones que nos permite
 - seguir la pista de los cambios realizados en nuestro código
 - volver a versiones anteriores si es necesario
 - crear ramas
 - trabajar en grupos más fácilmente
- Git es la herramienta y GitHub el servicio que aloja los proyectos que usan Git

BlueJ y Git - Github

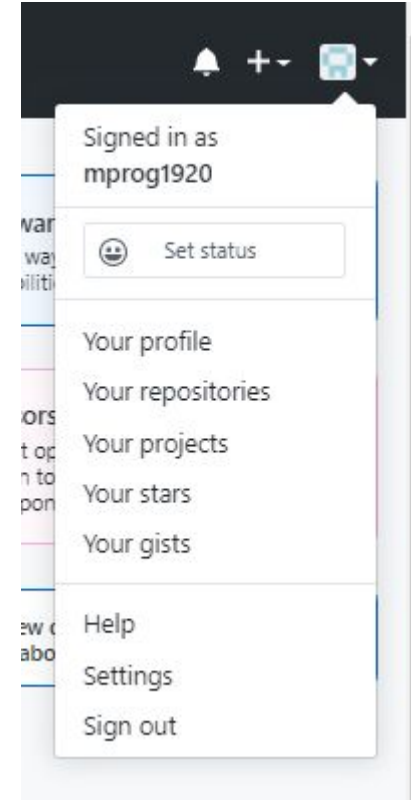
- BlueJ soporta el uso de repositorios Git aunque solo un subconjunto de su funcionalidad
- Nosotros utilizaremos BlueJ y GitHub para
 - descargar desde Github (repositorio remoto) a nuestro equipo local el proyecto de partida para completarlo
 - una vez realizadas las modificaciones al proyecto en local las subiremos a Github desde BlueJ (actualizamos el repositorio remoto)
- Debemos tener **previamente creada una cuenta en GitHub** y un nombre de usuario asociado

Proyecto a entregar (ejemplo *ensayoentrega*)

- Veremos qué hay que hacer con un proyecto de ejemplo *ensayoentrega*
- ***ensayoentrega*** es el proyecto de partida (repositorio) que se os habrá dejado en la cuenta de GitHub **<https://github.com/montsemsanz>**
- el repositorio *ensayoentrega* os lo llevaréis a vuestra cuenta de GitHub
- descargaréis dicho repositorio a vuestro PC local
- completaréis el proyecto en BlueJ en vuestro equipo
- una vez finalizado desde BlueJ lo llevaréis de nuevo a GitHub
- a continuación se indican los pasos en detalle con gráficos y alguna breve descripción de lo que significan

Paso 1 - Ejemplo *ensayoentrega*

1. Iniciar sesión en nuestra cuenta de GitHub
2. Elegimos *Settings / Repositories*
 - nos aseguramos que la rama principal es **master**
 - con ese nombre de rama principal por defecto se crearán los nuevos repositorios



Paso 1 - Ejemplo *ensayo entrega*

The screenshot shows the GitHub account settings interface. On the left is a sidebar with navigation links: Account settings, Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Sponsorship log, Emails, Notifications, Scheduled reminders, SSH and GPG keys, and Repositories. The 'Repositories' link is highlighted with a red bar. The main content area is divided into two sections. The top section, 'Repository default branch', has a heading and a paragraph: 'Choose the default branch for your new personal repositories. You can choose a different workflow, or because your integrations still require "master", you can change the default branch name on individual repositories. [Learn more](#)'. Below this is a dropdown menu with 'master' selected and highlighted by an orange box, and an 'Update' button. The bottom section, 'Repositories', has a heading and two tabs: 'Repositories' (active) and 'Deleted repositories'. It lists two repositories: 'montseaweb' and 'mprog1920'. The 'montseaweb' repository is expanded, showing two files: 'montseaweb/ENTRE-01-UT4-Pedido' (13 KB) and 'dganuzasdaw1/ENTRE-01-UT4-Pedido'. The 'mprog1920' repository is partially visible, showing 'mprog1920/cursopinfor' (7 KB) and '0 collaborators'.

Account settings

Profile

Account

Appearance

Account security

Billing & plans

Security log

Security & analysis

Sponsorship log

Emails

Notifications

Scheduled reminders

SSH and GPG keys

Repositories

Repository default branch

Choose the default branch for your new personal repositories. You can choose a different workflow, or because your integrations still require "master", you can change the default branch name on individual repositories. [Learn more](#)

master Update

Repositories

Repositories Deleted repositories

montseaweb

montseaweb/ENTRE-01-UT4-Pedido 13 KB

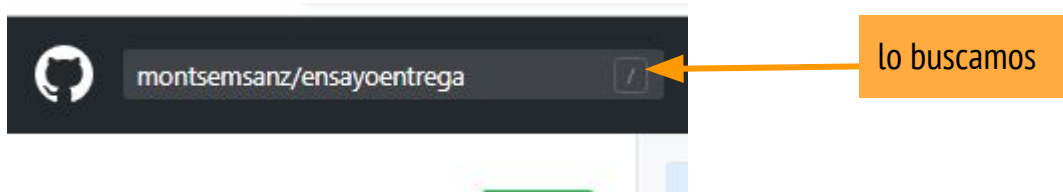
dganuzasdaw1/ENTRE-01-UT4-Pedido

mprog1920

mprog1920/cursopinfor 7 KB 0 collaborators

Paso 2 - Ejemplo *ensayoentrega* - Hacer un fork

- Hacer un **fork** del proyecto *ensayoentrega* desde **montsemsanz**



montsemsanz / *ensayoentrega*

Watch 1

Star 0

Fork 0

Code

Issues 0

Pull requests 1

Projects 0

Wiki

Security

Insights


No description, website, or topics provided.





Paso 2 - Ejemplo *ensayoentrega* - Hacer un fork

montsemsanz / **ensayoentrega** Public después del fork

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

 **montsemsanz** Primer commit ed7b714 9 minutes ago 🕒 1 commit

 .gitignore	Primer commit	9 minutes ago
 Estudiante.java	Primer commit	9 minutes ago
 TestEstudiante.java	Primer commit	9 minutes ago
 package.bluej	Primer commit	9 minutes ago

Qué es un fork

- **Hacer un fork** es
 - realizar una copia de un repositorio (en este caso de `montsemsanz/ensayoentrega`) en nuestra cuenta de GitHub

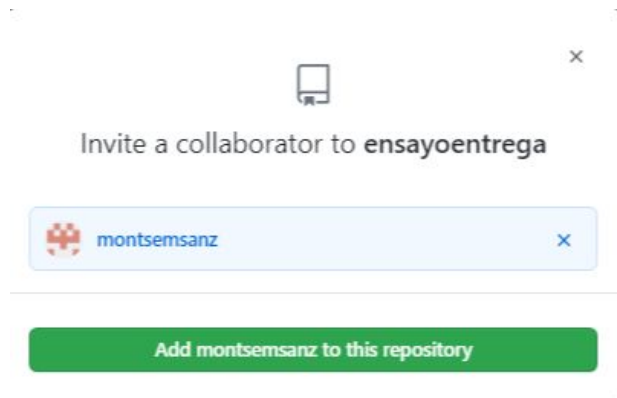
Paso 3 - Ejemplo *ensayoentrega* - Configurar el repositorio

- Hacéis a *montsemsanz* colaboradora (opcional)
 - click en el repositorio / pestaña *Settings* - *Manage Access* / *Invite a collaborator*

The screenshot shows the GitHub repository settings page for 'mprog1920/ensayoentrega', which is a fork of 'montsemsanz/ensayoentrega'. The 'Settings' tab is selected, and the 'Manage access' sub-tab is active. On the left, a sidebar lists various settings: Options, Manage access (highlighted), Security & analysis, Branches, Webhooks, Notifications, Integrations, Deploy keys, Secrets, Actions, Moderation settings, and Interaction limits. The main content area is titled 'Who has access' and shows two sections: 'PUBLIC REPOSITORY' (indicating the repository is public and visible to anyone, with a 'Manage' link) and 'DIRECT ACCESS' (showing 0 collaborators with access). Below this, the 'Manage access' section displays a message: 'You haven't invited any collaborators yet' with an icon of a person and a lock, and a green button labeled 'Invite a collaborator'.

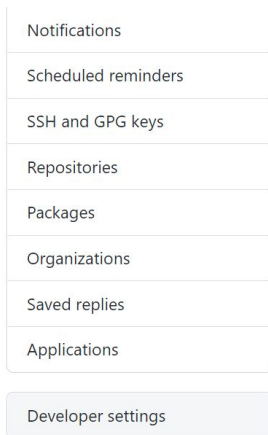
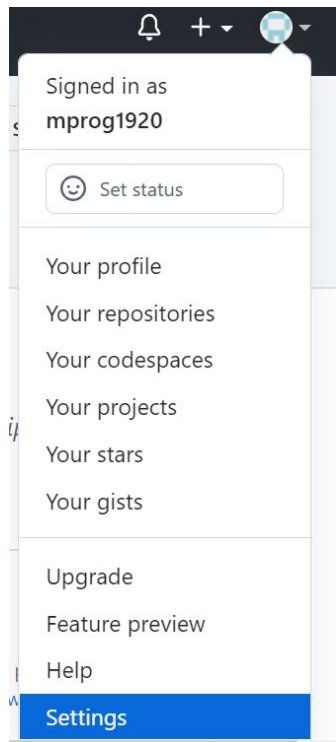
Paso 3 - Ejemplo *ensayoentrega* - Configurar el repositorio

- Hacéis a *montsemsanz* colaboradora (opcional)
 - click en el repositorio / pestaña *Settings - Manage Access / Invite a collaborator*

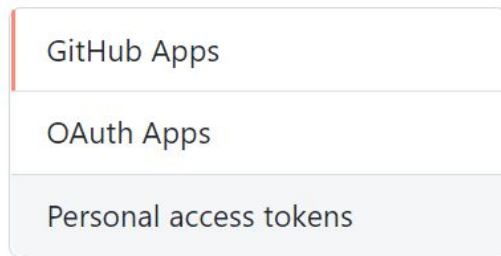


Paso 4 - Ejemplo *ensayoentrega* - Crear un token de autenticación en GitHub

- En GitHub accedemos en nuestro perfil a *Settings* – *Developer Settings* – *Personal Access Tokens*

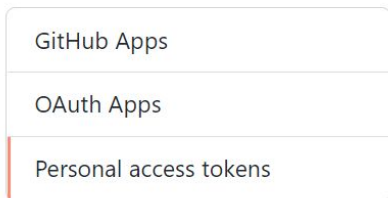


Settings / Developer settings



Paso 4 - Ejemplo *ensayo entrega* - Crear un token de autenticación en GitHub

- Pulsar el botón ***Generate new token***.
 - Podemos llamarlo por ejemplo **BlueJ**, sin fecha de expiración, y con todos los permisos seleccionados de la primera sección: *repo*.
 - Pulsar ***Generate Token***.



Personal access tokens

Need an API token for scripts or testing? [Generate a personal access token](#) for quick access to the [GitHub API](#).

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

A button labeled 'Generate new token' is highlighted with an orange rectangular border. The button has a light gray background and rounded corners.

Paso 4 - Ejemplo *ensayo entrega* - Crear un token de autenticación en GitHub

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

BlueJ

What's this token for?

Expiration *

No expiration

The token will never expire!

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

☒ repo

Full control of private repositories

☒ repo:status

Access commit status

☒ repo_deployment

Access deployment

☒ public_repo

Access public repositories

☒ repo:invite

Access repository

☒ security_events

Read and write security events

☐ read:pkg_key

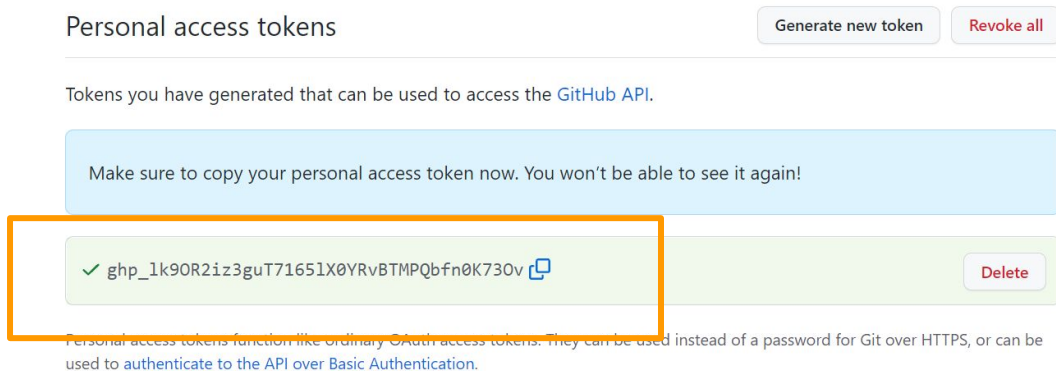
Read public user GPG keys

Generate token

Cancel

Paso 4 - Ejemplo *ensayoentrega* - Crear un token de autenticación en GitHub

- Ya tendremos generado el token de autenticación
 - lo copiamos con el botón proporcionado en el formulario. **Debemos copiarlo en ese momento, ya que no permite su consulta posterior.**
 - lo pegamos en un bloc de notas temporal y lo guardamos



Paso 5 - Ejemplo *ensayoentrega* - clonar el repositorio a nuestro PC

- Clonaremos ahora este repositorio desde GitHub a nuestro PC

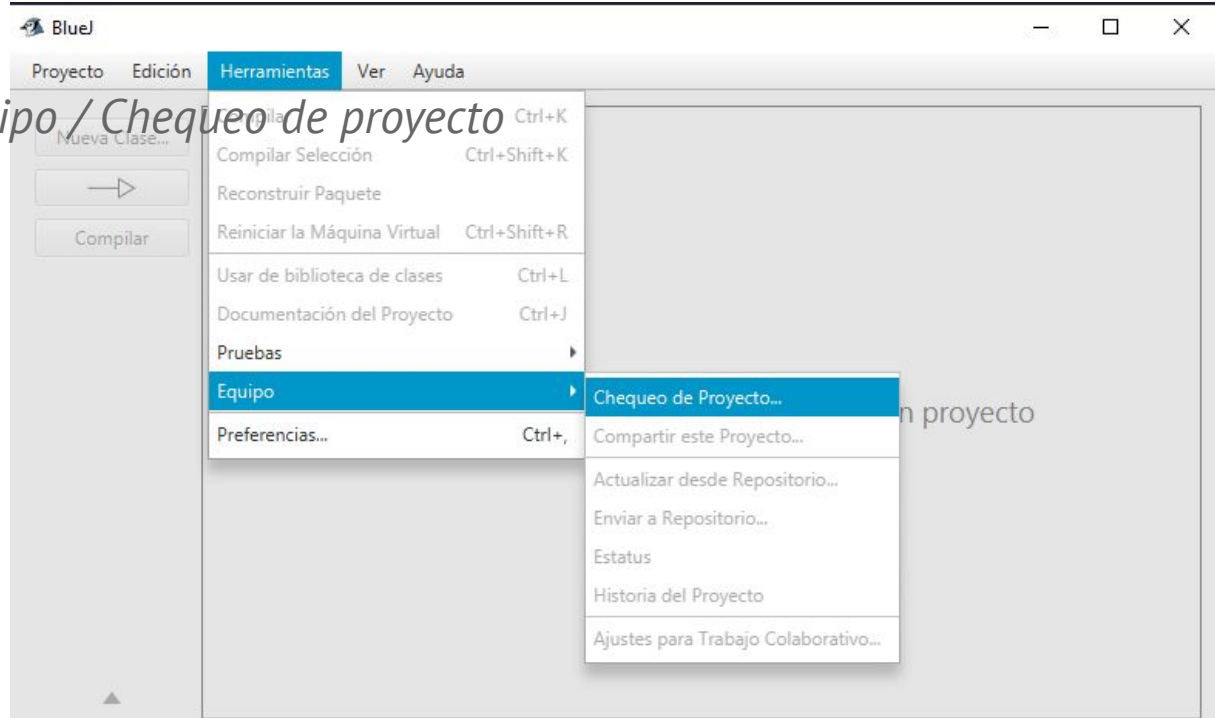
The screenshot shows a GitHub repository page for 'montsemsanz Primer commit'. At the top, there are buttons for 'Go to file', 'Add file', and 'Code'. The 'Code' button is highlighted in green. Below the repository name, there is a message: 'This branch is even with montsemsanz:master.' Below this, there is a table of files:

File	Commit
.gitignore	Primer commit
Estudiante.java	Primer commit
TestEstudiante.java	Primer commit
package.bluej	Primer commit

On the right side, the 'Code' dropdown menu is open, showing options: 'Clone', 'Open with GitHub Desktop', and 'Download ZIP'. The 'Clone' option is selected, and the URL 'https://github.com/mprog1920/ensayoentrega' is displayed. An orange box with an arrow points to the URL with the text 'copiar la URL'.

Paso 5 - Ejemplo *ensayo entrega* - clonar el repositorio a nuestro PC

- Abrir BlueJ
 - *Herramientas / Equipo / Chequeo de proyecto*



Paso 5 - Ejemplo *ensayoentrega* - clonar el repositorio a nuestro PC

Ajustes del Equipo

Tipo de servidor: Subversion ☐ Git ☒

Localización

Repository URI:

Personal

Your name:

Your e-mail:

Usuario:

Contraseña:

☒ Recuerde ajustes para futuros proyectos

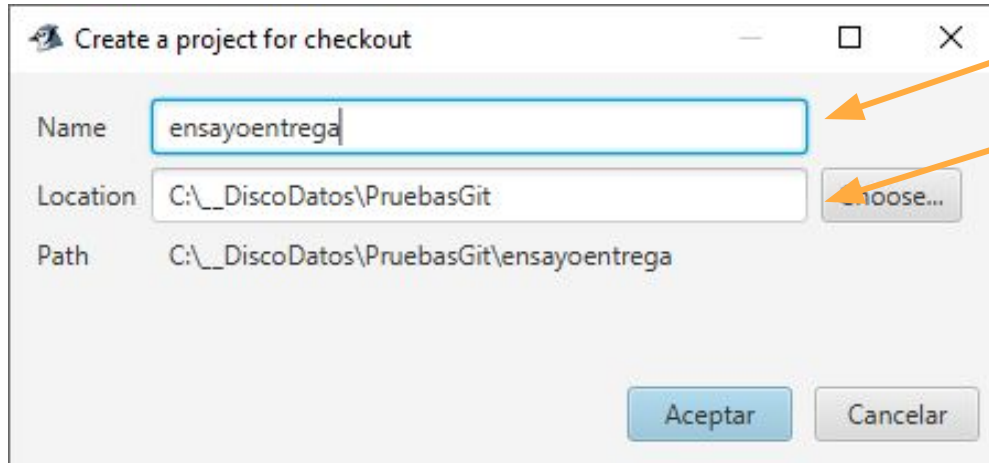
pegar la URL seleccionada
anteriormente

vuestro nombre

vuestra cta gmail

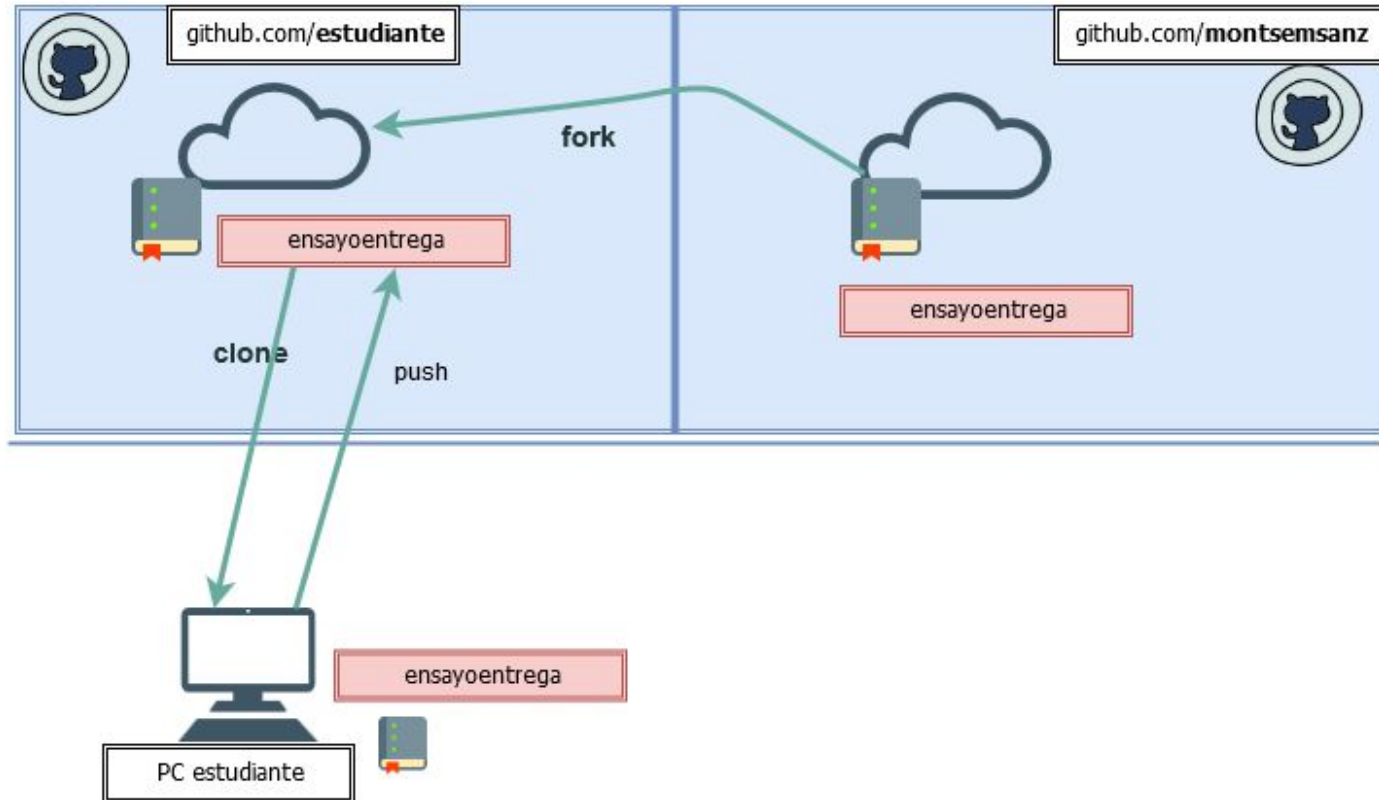
vuestro usuario y contraseña
el token generado

Paso 5 - Ejemplo *ensayoentrega* - clonar el repositorio a nuestro PC













nombre del proyecto y ubicación
en vuestro PC

Resumen hasta el momento



Paso 6 - Ejemplo *ensayoentrega* - completar el proyecto en el PC local

Disco local (C:) > __DiscoDatos > PruebasGit > ensayoentrega

Nombre	Fecha de modificación
 .git	28/09/2021 17:55
 .gitignore	28/09/2021 17:55
 Estudiante.class	28/09/2021 17:56
 Estudiante.ctxt	28/09/2021 17:56
 Estudiante.java	28/09/2021 17:55
 package.bluej	28/09/2021 17:55
 team.defs	28/09/2021 17:55
 TestEstudiante.class	28/09/2021 17:56
 TestEstudiante.ctxt	28/09/2021 17:56
 TestEstudiante.java	28/09/2021 17:55

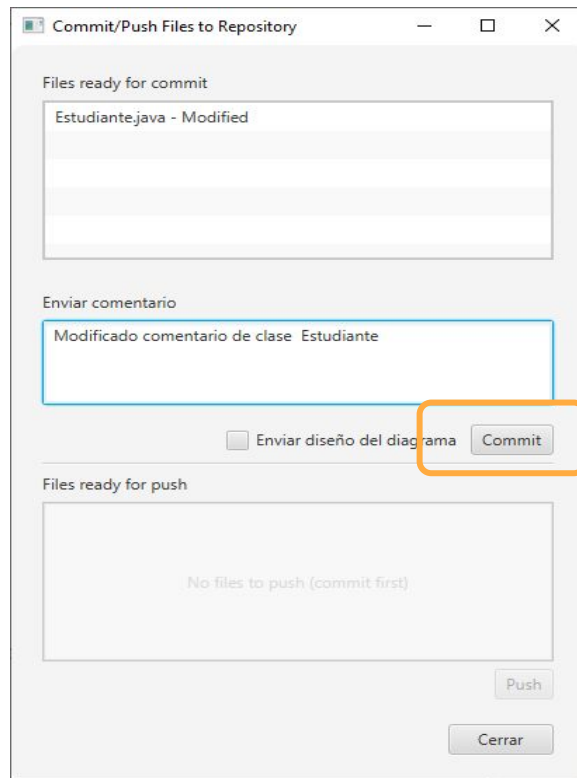
repositorio local. En la carpeta **.git** es donde el sistema de control de versiones guarda nuestro código junto con el historial de todos sus cambios

NO BORRAR esta carpeta

Es una carpeta oculta. Si no la veis desde el explorador *Vista* / activar *Elementos ocultos*

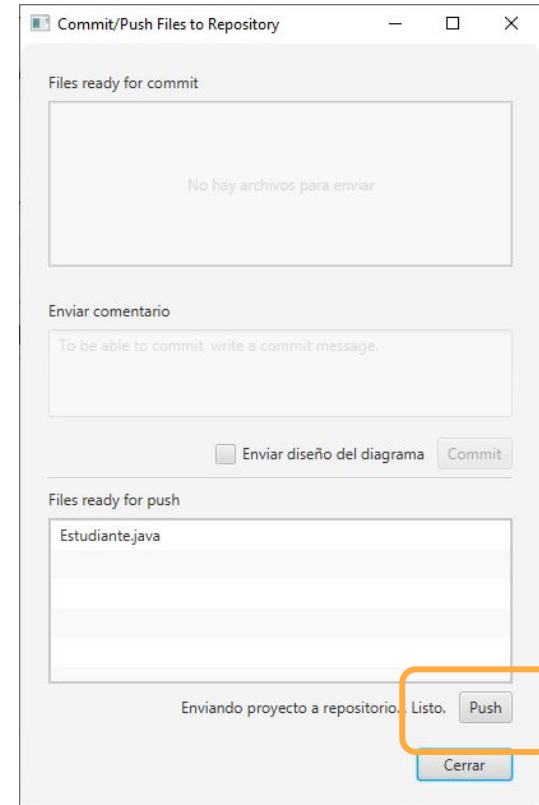
Paso 7 - Ejemplo *ensayo entrega* - Commit Push

- Desde BlueJ completamos nuestro proyecto
- Cada poco tiempo (por ejemplo al acabar un método) registramos esos cambios y los llevamos a GitHub
 - se traduce en hacer *Herramientas / Equipo / Commit / Push to repository*




Paso 7 - Ejemplo *ensayo entrega* - Commit Push

- Desde BlueJ completamos nuestro proyecto
- Cada poco tiempo (por ejemplo al acabar un método) registramos esos cambios y los llevamos a GitHub
 - se traduce en hacer
Herramientas / Equipo / Commit / Push to repository





Paso 7 - Ejemplo *ensayoentrega* - Commit Push


 **mprog1920 / ensayoentrega** Public

forked from [montsemsanz/ensayoentrega](#)

[Code](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)


 **master** ▾





 **1 branch**

 **0 tags**

[Go to file](#) [Add file ▾](#) [Code ▾](#)

This branch is 1 commit ahead of montsemsanz:master. [Contribute ▾](#) [Fetch upstream ▾](#)

 **mprog1920** Modificado comentario de clase Estudiante 0b8a235 12 minutes ago [2 commits](#)

 .gitignore	Primer commit	41 minutes ago
 Estudiante.java	Modificado comentario de clase Estudiante	12 minutes ago
 TestEstudiante.java	Primer commit	41 minutes ago
 package.bluej	Primer commit	41 minutes ago

Algo de vocabulario Git -

- **Commit**

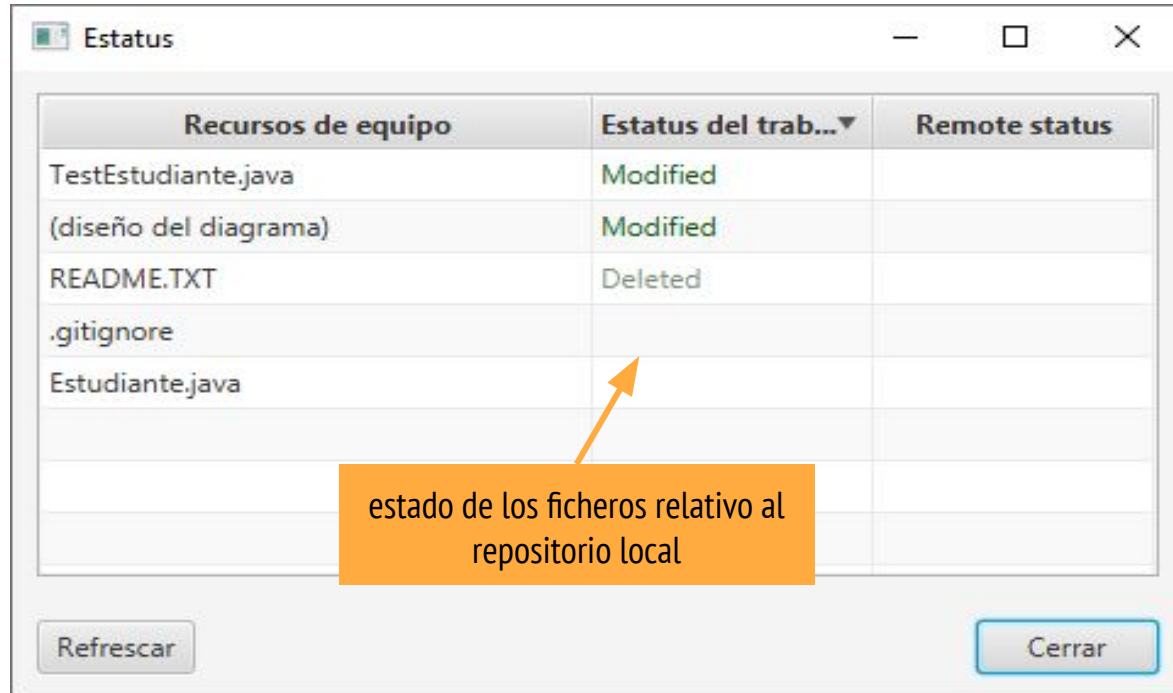
- al hacer un commit registramos los cambios de nuestro proyecto en el repositorio local
- todo commit incluye un comentario que indica el cambio realizado

- **Push**

- mediante push los cambios registrados con el commit en el repositorio local se reflejan en el repositorio remoto (GitHub)

Git y BlueJ

Herramientas / Equipo / Status



Git y BlueJ

Herramientas / Equipo / Status



The screenshot shows the 'Estatus' window in BlueJ, which displays the status of files in the current project relative to the remote repository. The window has a title bar with standard window controls and a table with three columns: 'Recursos de equipo', 'Estatus del trabajo co...', and 'Remote status'. The table lists several files: '(diseño del diagrama)' with status 'Modified', 'README.TXT' with status 'Needs push (deleted)', 'TestEstudiante.java' with status 'Needs push (modified)', '.gitignore', and 'Estudiante.java'. An orange arrow points from a text box at the bottom to the 'Needs push (modified)' status for 'TestEstudiante.java'. At the bottom of the window, there are two buttons: 'Refrescar' and 'Cerrar'.

Recursos de equipo	Estatus del trabajo co...	Remote status
(diseño del diagrama)	Modified	
README.TXT		Needs push (deleted)
TestEstudiante.java		Needs push (modified)
.gitignore		
Estudiante.java		

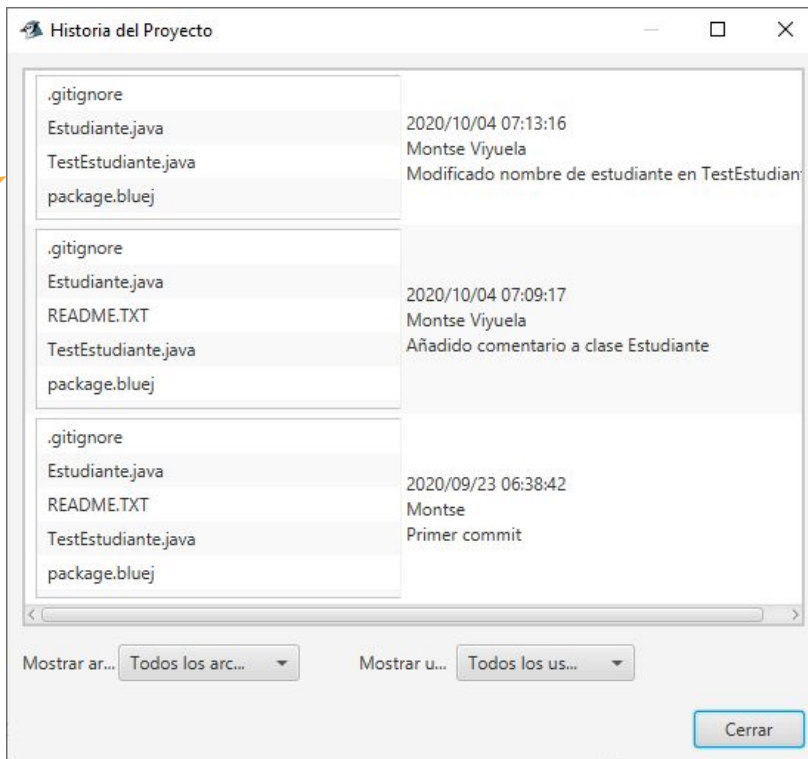
estado de los ficheros relativo al repositorio remoto

Refrescar Cerrar

Git y BlueJ

Herramientas / Equipo / Historia del proyecto

registro o log con todos los mensajes de commit del repositorio y el usuario que los ha realizado



Enlaces

- <https://www.bluej.org/tutorial/git>
- <https://guides.github.com/introduction/git-handbook/>
- <https://github.com/flowsta/github>