

Capítulo 1

INTRODUCCIÓN A GIT Y GITHUB

OBTENER EL SOFTWARE

Te recomiendo que instales GIT FOR WINDOWS, la ruta es la siguiente:

<https://git-for-windows.github.io/>

El archivo que debes descargar para Windows de 64 Bits es el siguiente:

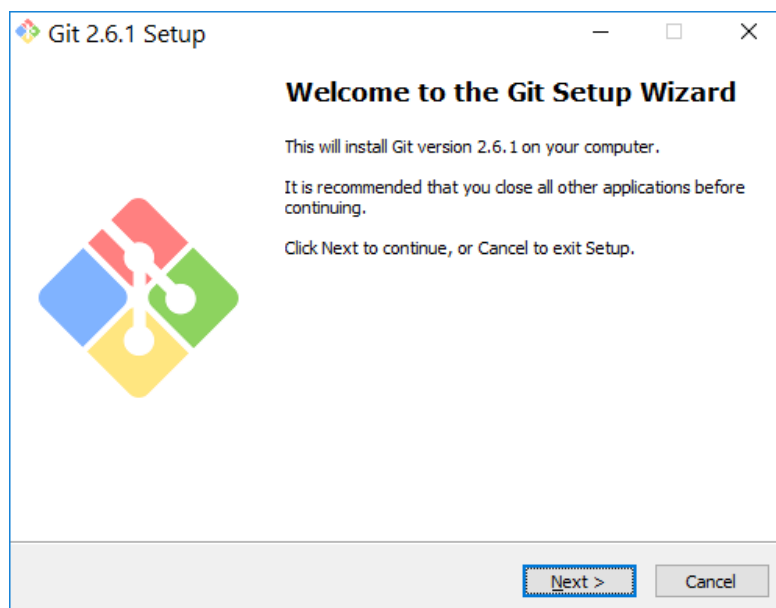
Git-2.6.1-64-bit.exe

Es posible que en este momento que estás leyendo este documento ya exista una nueva versión.

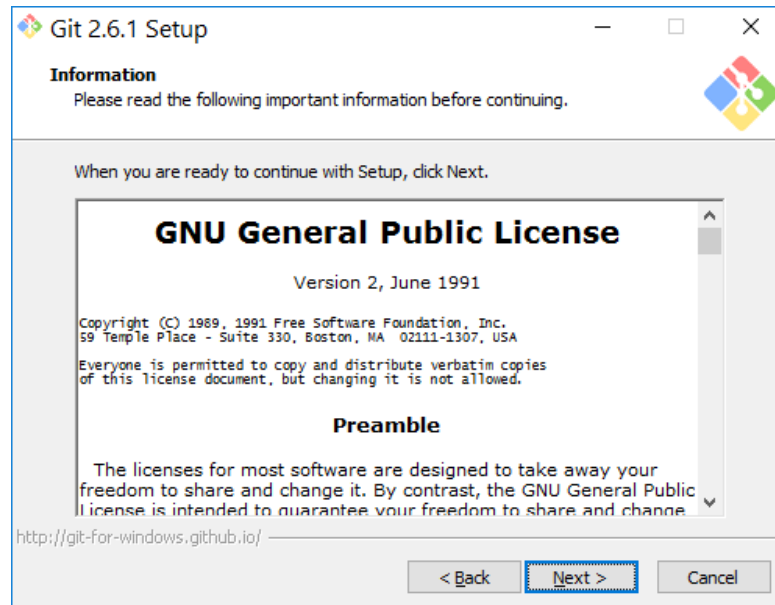
PROCESO DE INSTALACIÓN

Durante el proceso de instalación debes integrarlo con la consola de Windows.

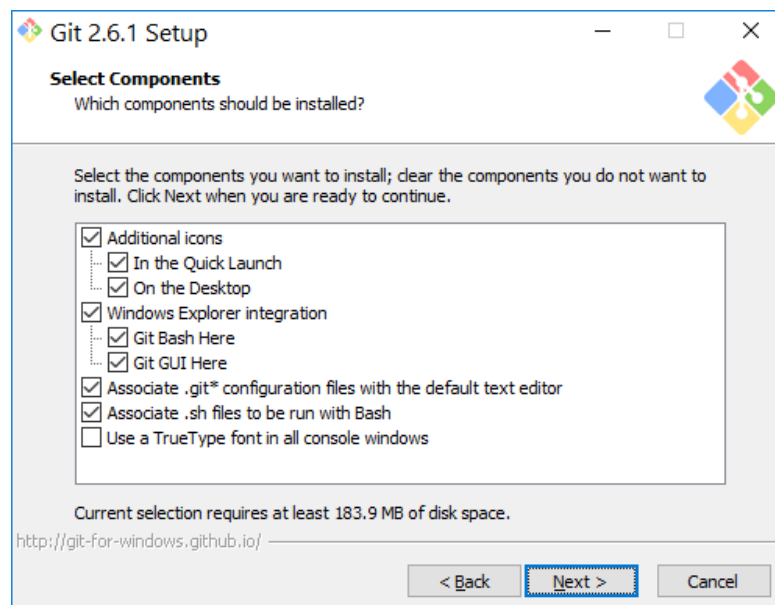
1. Ventana de bienvenida, solo debes hacer click en el botón **Next**.



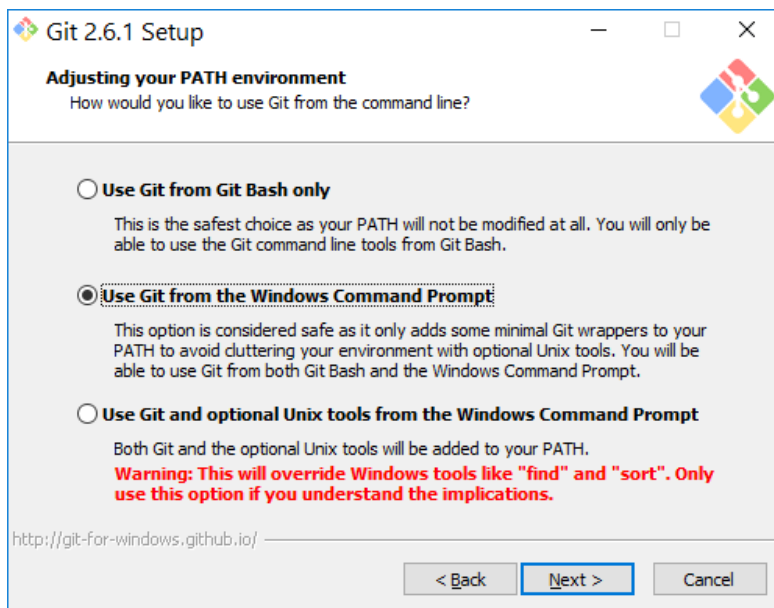
2. Ventana de licencia del software, solo debes hacer click en el botón **Next**.



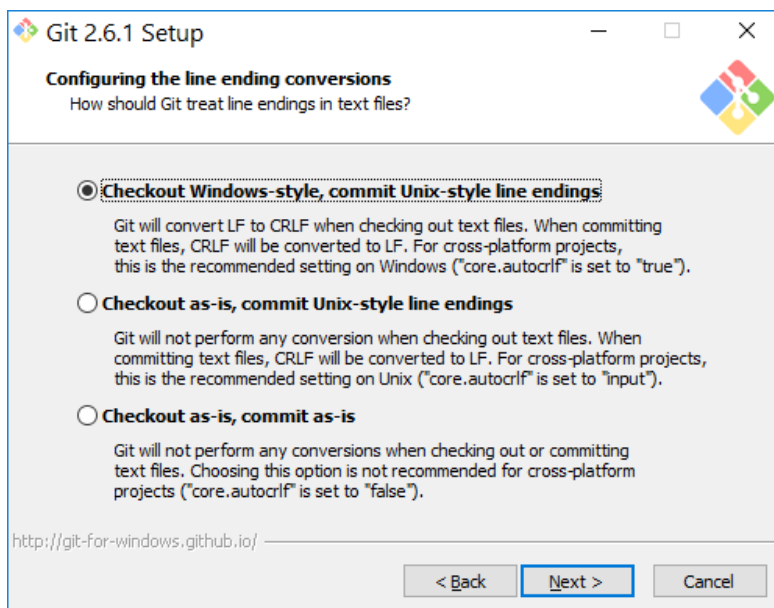
3. Ventana de componentes a instalar, seleccione los componentes tal como se ilustra en la siguiente imagen y haga click en el botón **Next**.



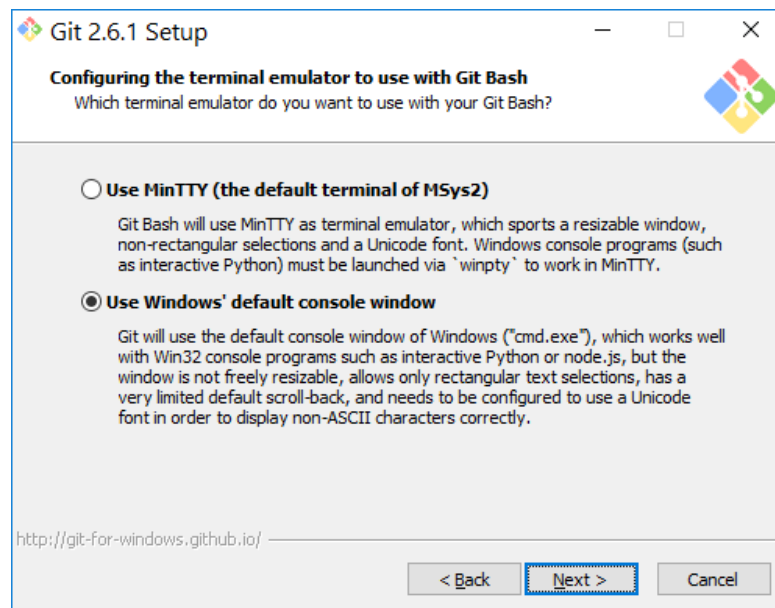
4. Ventana de configuración de la variable PATH de entorno de Windows. Seleccione la opción tal como aparece en la siguiente imagen y haga click en el botón **Next**.



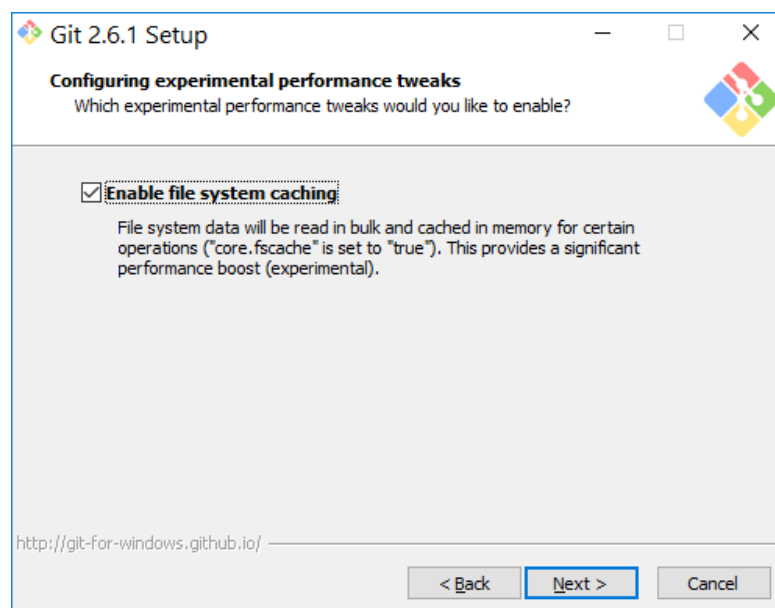
5. Ventana de configuración de conversiones de fin de línea. Deje la opción por defecto y haga click en el botón **Next**.



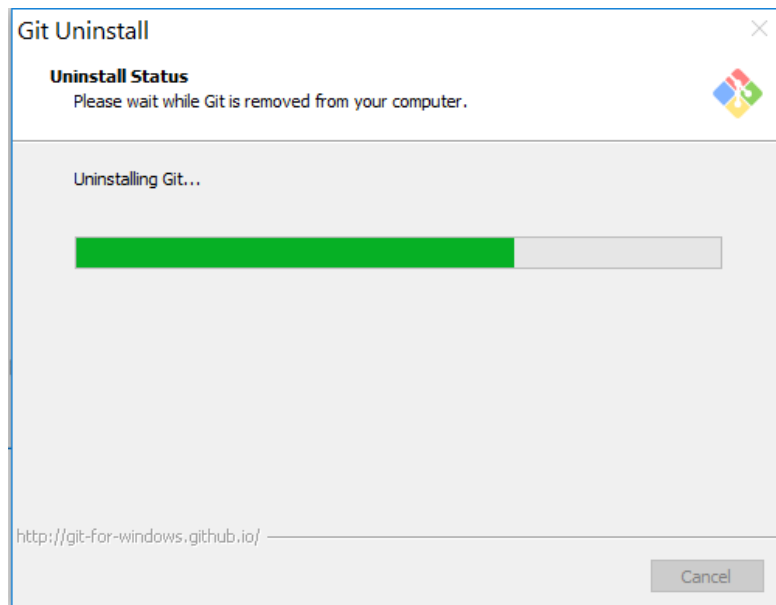
6. Ventana de configuración de terminal. Seleccione la opción que aparece en la siguiente imagen y haga click en el botón **Next**.



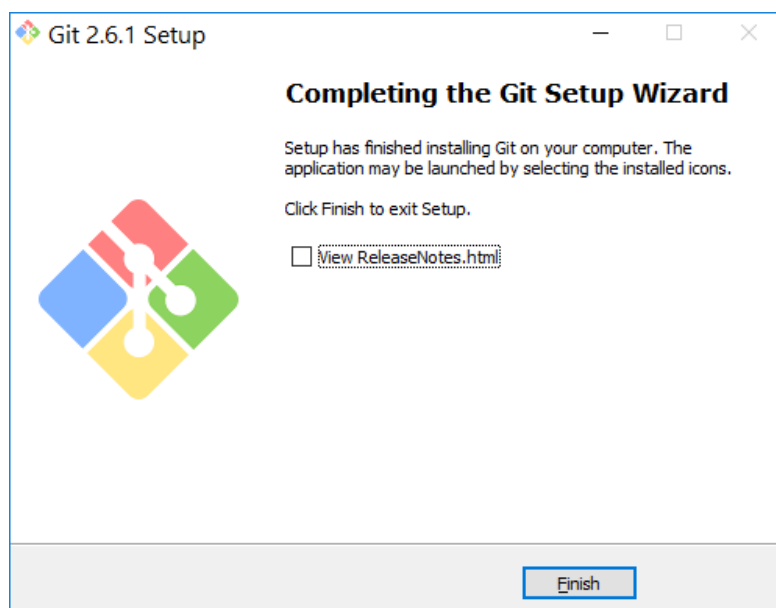
7. En la siguiente ventana configure el uso de sistema de cache y haga click en el botón **Next**.



8. Luego se inicia el proceso de instalación.

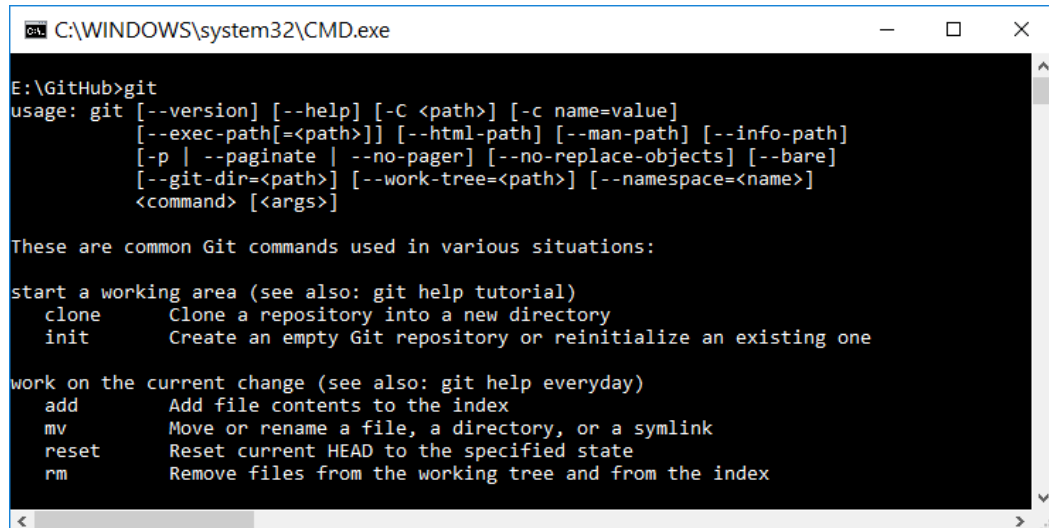


9. Finalmente, el proceso de instalación finaliza, haga click en el botón **Finish**.



COMANDOS INICIALES

Todos los comandos se deben ejecutar en la consola de Windows. Personalmente, tengo una carpeta GitHub donde tengo todos mis repositorios que mantengo en GitHub.



```
C:\WINDOWS\system32\CMD.exe

E:\GitHub>git
usage: git [--version] [--help] [-C <path>] [-c name=value]
         [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
         [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
         [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
         <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
    clone      Clone a repository into a new directory
    init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
    add        Add file contents to the index
    mv         Move or rename a file, a directory, or a symlink
    reset      Reset current HEAD to the specified state
    rm         Remove files from the working tree and from the index
```

Verificar la instalación de Git

Cuando ejecutas el comando **git** muestra su sintaxis y una ayuda sobre cada uno de los comandos que puedes ejecutar para gestionar un repositorio.

```
E:\GitHub>git
usage: git [--version] [--help] [-C <path>] [-c name=value]
         [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
         [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
         [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
         <command> [<args>]
```

Consultar la versión de Git instalada

```
E:\GitHub>git --version
git version 2.6.1.windows.1
```

Registra tu nombre

Es muy importante que registres tu nombre para saber quién o quienes estan registrando cambios en el repositorio.

La sintaxis es la siguiente:

```
git config --global user.name "Aquí escribe tu nombre y apellido"
```

Por ejemplo, para mi caso sería así:

```
E:\GitHub>git config --global user.name "Eric Gustavo Coronel Castillo"
```

Registra tu correo electrónico

Es muy importante que registres tu correo electrónico para saber quién o quienes están registrando cambios en el repositorio.

La sintaxis es la siguiente:

```
git config --global user.email "Aquí escribe tu correo electrónico"
```

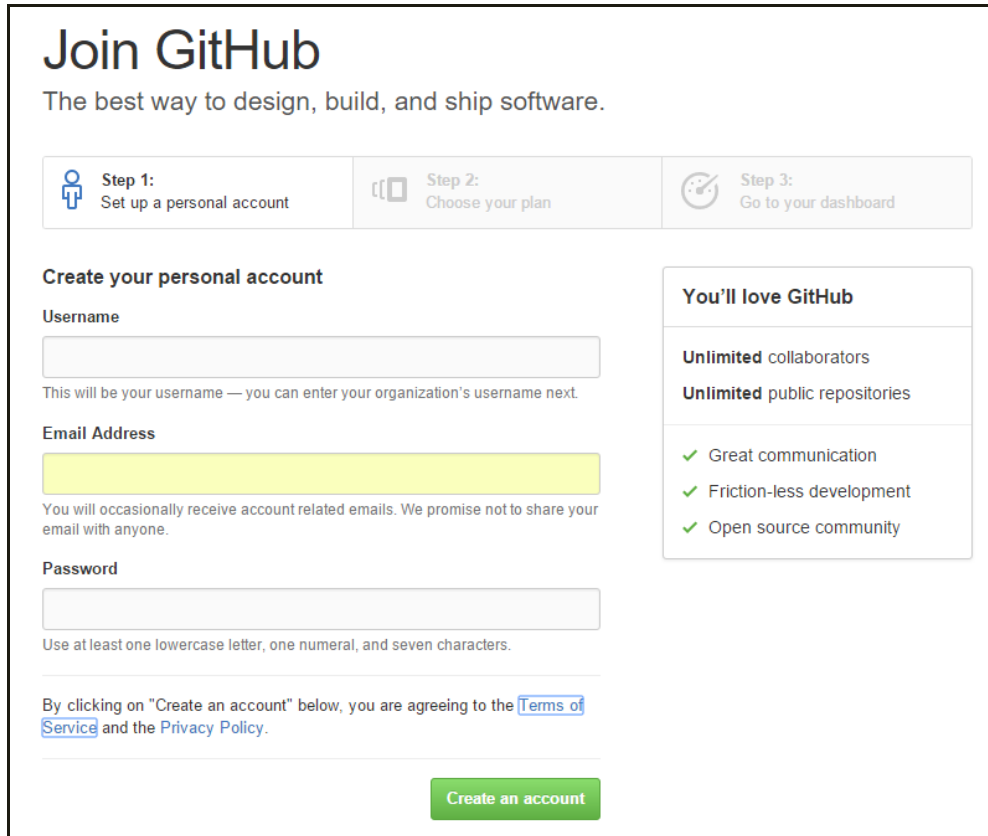
Por ejemplo, para mi caso sería así:

```
E:\GitHub>git config --global user.email "gcoronelc@gmail.com"
```


CREAR UN REPOSITORIO EN GITHUB

Crear una cuenta en GitHub

En GitHub (<https://github.com>) procede a crear tu cuenta de usuario.



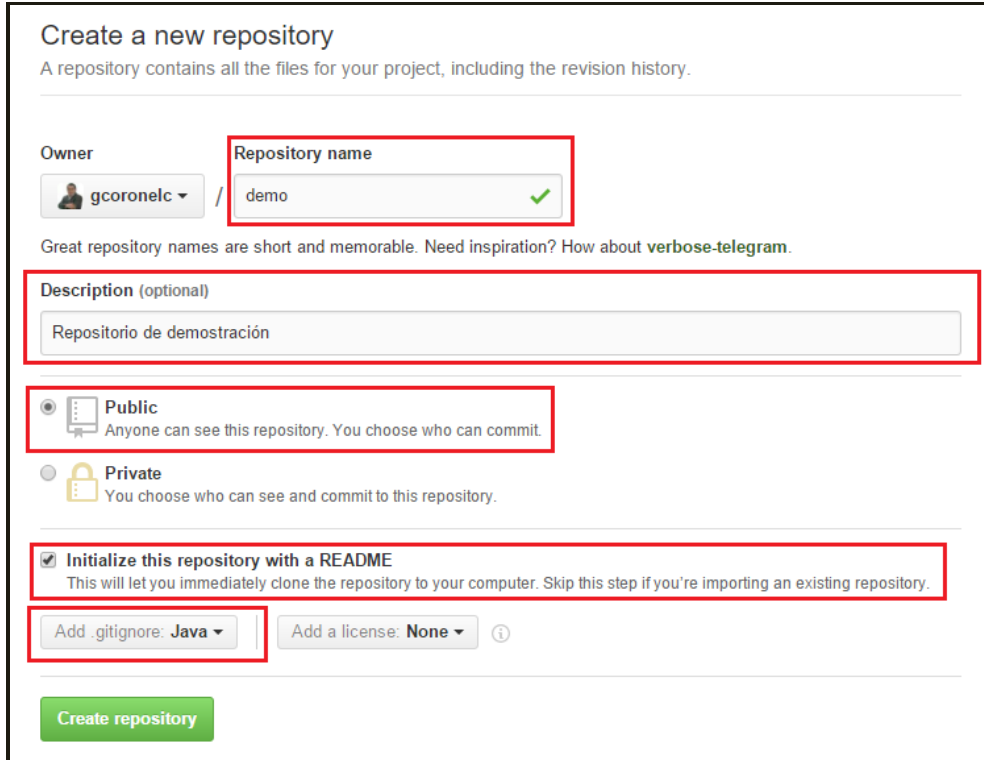
The screenshot shows the GitHub registration page. At the top, it says 'Join GitHub' and 'The best way to design, build, and ship software.' Below this is a progress bar with three steps: 'Step 1: Set up a personal account' (active), 'Step 2: Choose your plan', and 'Step 3: Go to your dashboard'. The main section is 'Create your personal account'. It has three input fields: 'Username' (with a note that it will be the user's name), 'Email Address' (highlighted in yellow, with a note about receiving emails), and 'Password' (with a note about password requirements). Below these fields is a link to 'Terms of Service' and 'Privacy Policy'. At the bottom right is a green 'Create an account' button. On the right side, there is a box titled 'You'll love GitHub' listing benefits: 'Unlimited collaborators', 'Unlimited public repositories', 'Great communication', 'Friction-less development', and 'Open source community'.

GitHub te enviará un correo electrónico para que confirmes la creación de la cuenta.

Procede a crear un repositorio

Procede a crear un repositorio de demostración de nombre **demo**.

La siguiente imagen se ilustra la creación de repositorio **demo** con mi cuenta.



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: gcoronelc / Repository name: demo

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

Description (optional): Repositorio de demostración

☒ Public: Anyone can see this repository. You choose who can commit.

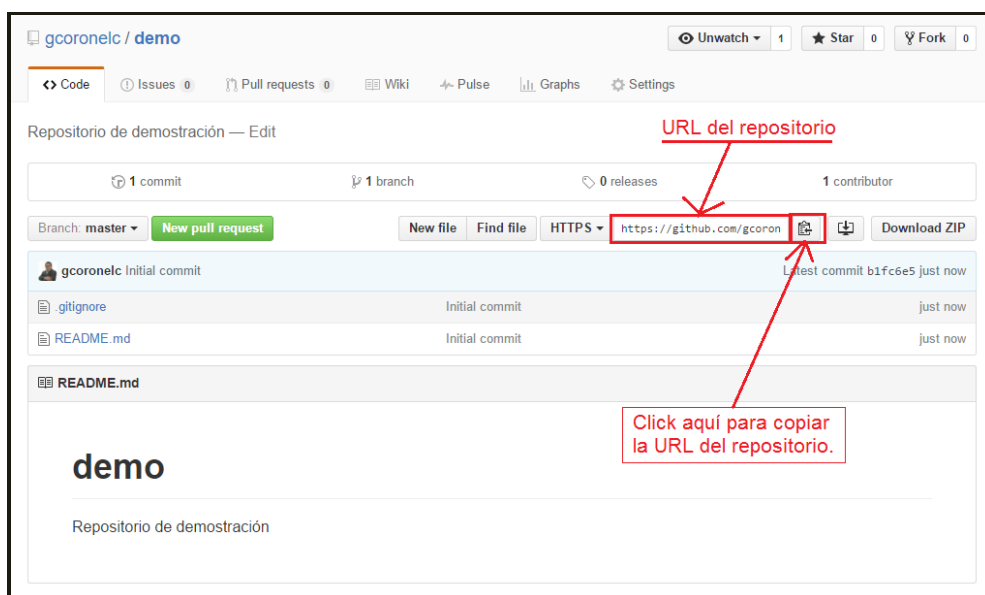
☐ Private: You choose who can see and commit to this repository.

☒ Initialize this repository with a README: This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

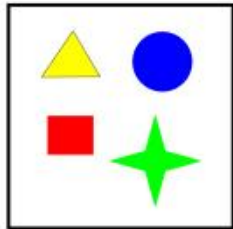
Add .gitignore: Java Add a license: None

Create repository

A continuación tienes la imagen que confirma la creación del repositorio **demo**.



CLONAR REPOSITORIO DEMO



Ahora vas a clonar el repositorio **demo** creado en el ítem anterior.

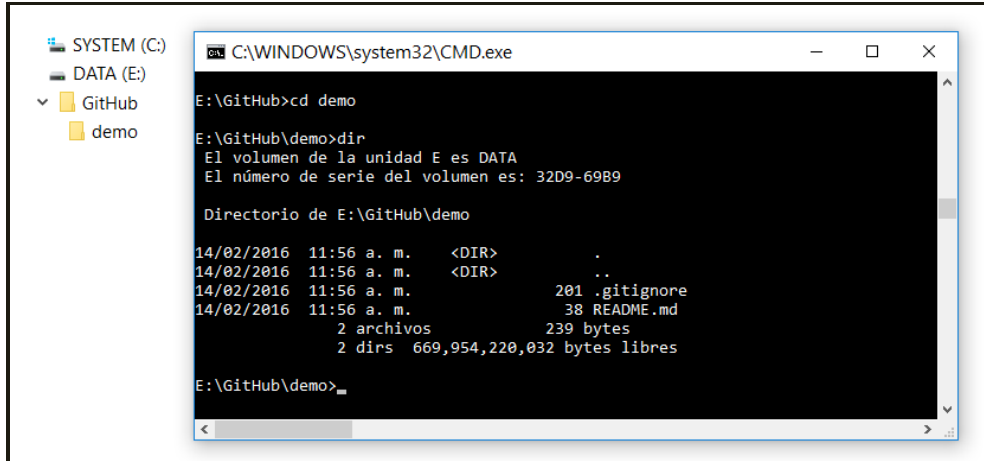
Para clonar un repositorio la sintaxis es:

```
git clone <URL del repositorio>
```

Para el caso del repositorio **demo**, sería así:

```
E:\GitHub>git clone https://github.com/gcoronelc/demo.git
Cloning into 'demo'...
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), done.
Checking connectivity... done.
```

El repositorio ha sido clonado correctamente, no presenta ningún mensaje de error. En la unidad E: puedes verificar que se ha creado la carpeta demo correspondiente a repositorio, tal como se ilustra en la siguiente imagen:



```
C:\WINDOWS\system32\CMD.exe

E:\GitHub>cd demo

E:\GitHub\demo>dir
El volumen de la unidad E es DATA
El número de serie del volumen es: 32D9-69B9

Directorio de E:\GitHub\demo


14/02/2016  11:56 a. m.    <DIR>          .
14/02/2016  11:56 a. m.    <DIR>          ..
14/02/2016  11:56 a. m.                201 .gitignore
14/02/2016  11:56 a. m.                38 README.md
                2 archivos            239 bytes
                2 dirs  669,954,220,032 bytes libres

E:\GitHub\demo>
```

EDITAR REPOSITORIO

Modificar el archivo .gitignore

En el editor de texto carga el archivo `.gitignore` y elimina la línea que tenga `"*.jar"`, debe quedar como se muestra en la siguiente imagen:



```
.gitignore: Bloc de notas
Archivo Edición Formato Ver Ayuda

*.class

# Mobile Tools for Java (J2ME)
.mtj.tmp/

# Package Files #
*.war
*.ear

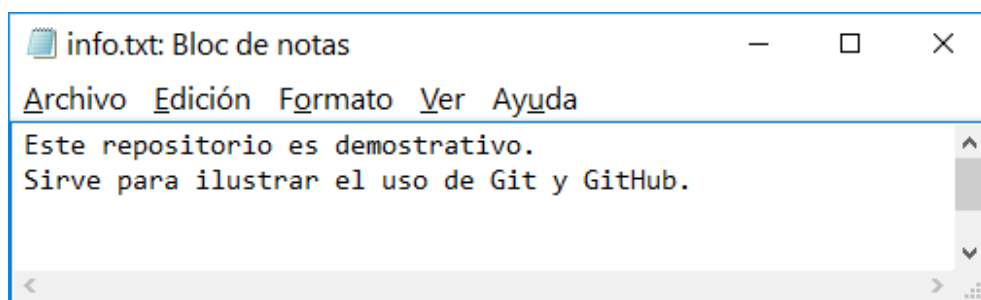
# virtual machine crash logs, see http://www.java.com/en/download/help/error_hotspot.xml
hs_err_pid*
|
```

Graba y cierra el archivo.

Crea un archivo

En la carpeta demo del repositorio procede a crear un archivo nuevo de nombre `info.txt`, y registra información sobre información que registraras en tu repositorio.

La siguiente imagen muestra lo que podría ser el archivo `info.txt`.



```
info.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda

Este repositorio es demostrativo.
Sirve para ilustrar el uso de Git y GitHub.
```

ACTUALIZAR EL REPOSITORIO EN GITHUB

Verificar el estado del repositorio

El comando a ejecutar es:

```
git status
```

Para mi caso sería así:

```
E:\GitHub\demo>git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   .gitignore

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

        info.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

El resultado nos indica que existen un archivo modificado y uno nuevo.

Confirmar los cambios en el repositorio local

Para confirmar los cambios, primero debes preparar la lista de archivos a confirmar con el comando **git add**.

Para mi caso, si quiero agregar todos los archivos a la lista:

```
E:\GitHub\demo>git add .
```

Para confirmar la lista de archivos preparada con **git add**, se debe utilizar el comando **git commit**.

Para mi caso, sería así:

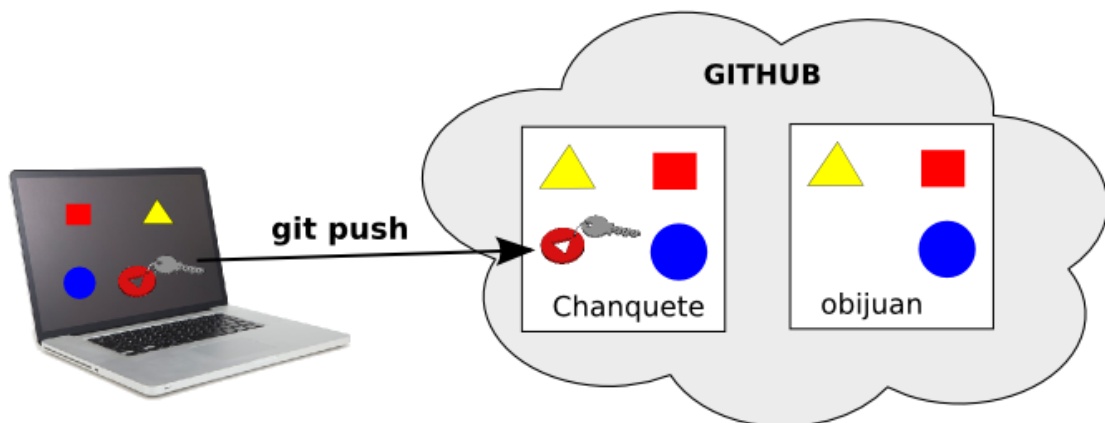
```
E:\GitHub\demo>git commit -m "Probando el repositorio."  
[master 5f781ff] Probando el repositorio.  
2 files changed, 1 deletion(-)  
create mode 100644 info.txt
```

Puedes utilizar nuevamente el comando `git status` para verificar el estado de tu repositorio.

```
E:\GitHub\demo>git status  
On branch master  
Your branch is ahead of 'origin/master' by 1 commit.  
  (use "git push" to publish your local commits)  
nothing to commit, working directory clean
```

En este caso indica que el repositorio esta adelantado 1 commit con respecto a repositorio origen, y se debe utilizar `git push` para subir los cambios.

Subir los cambios a GitHub



Para subir los cambios a GitHub se utiliza el comando `git push`.

```
git push origin master
```

Cuando ejecutas este comando te solicita tu cuenta de usuario y clave, salvo que ya se encuentre configurado, como es mi caso.

A continuación se tiene el resultado para mi caso:

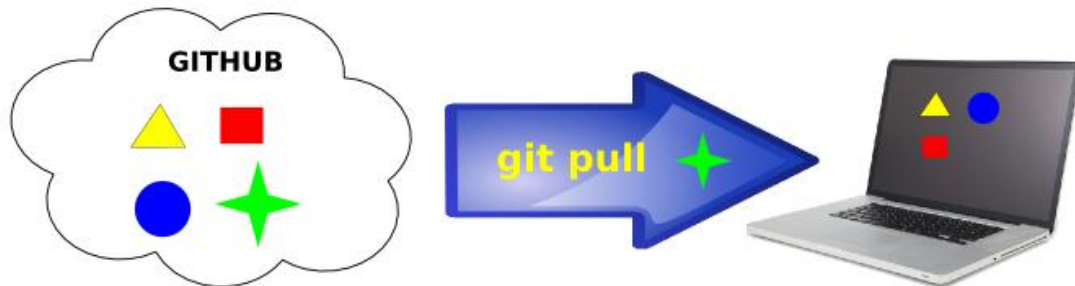
```
E:\GitHub\demo>git push origin master
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 371 bytes | 0 bytes/s, done.
Total 4 (delta 1), reused 0 (delta 0)
To https://github.com/gcoronelc/demo.git
    b1fc6e5..5f781ff  master -> master
```

Ahora puedes nuevamente verificar el estado de tu repositorio.

A continuación tienes el resultado para mi repositorio.

```
E:\GitHub\demo>git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
```


ACTUALIZAR EL REPOSITORIO LOCAL



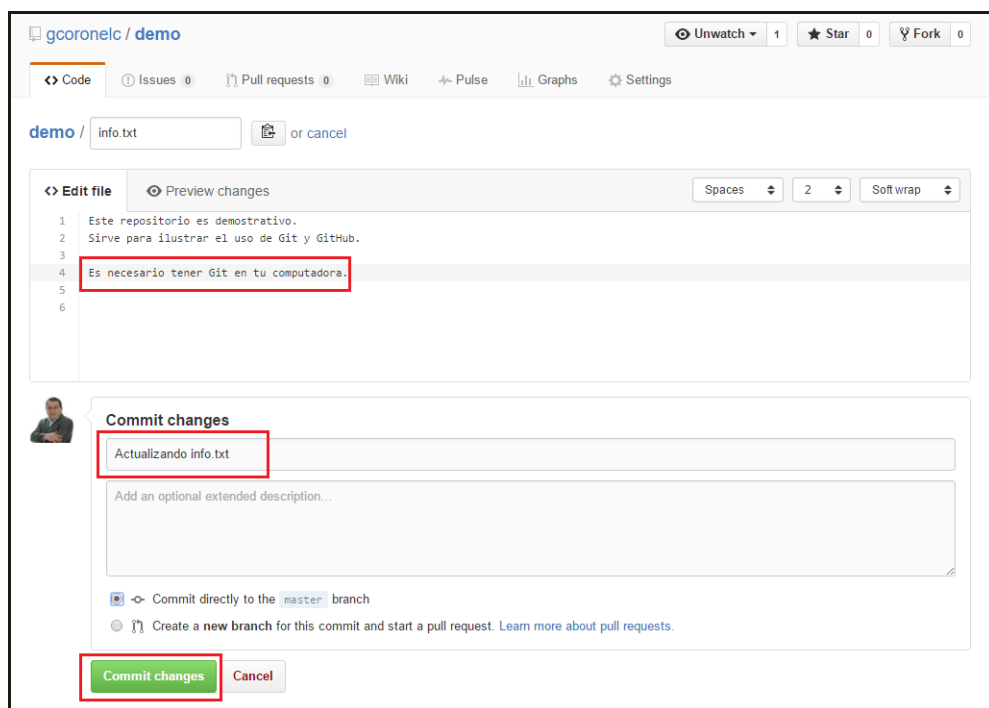
Puede suceder que el repositorio lo clones en la universidad y has realizado varias actualizaciones, luego quieres actualizar el repositorio de tú computadora en tu casa.

En estas situaciones debes usar el comando `git pull` para actualizar tu repositorio local.

Modifica el archivo `into.txt` en GitHub

Procede a editar el archivo `into.txt` en GitHub y agrégale una línea, y luego hazle `commit`.

A continuación tienes una imagen de como podrías realizarlo:



Actualiza tu repositorio local

Antes de que realices cualquier cambio en tu repositorio local, se recomienda que lo actualices con los últimos cambios, para eso debe utilizar el comando **git pull**.

Aquí tienes su sintaxis:

```
git pull origin master
```

Aquí tienes un ejemplo con mi repositorio:

```
E:\GitHub\demo>git pull origin master
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/gcoronelc/demo
* branch          master      -> FETCH_HEAD
   2a5a48e..2317f84 master      -> origin/master
Updating 2a5a48e..2317f84
Fast-forward
 info.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

El proceso se ha ejecutado correctamente, un archivo ha cambiado.

El archivo que ha cambiado es **info.txt**, he indica que una fila se ha insertado, pero, también una fila se ha eliminado, a continuación tienes el script para consultar el contenido del archivo **info.txt**:

```
E:\GitHub\demo>type info.txt
Este repositorio es demostrativo.
Sirve para ilustrar el uso de Git y GitHub.

Es necesario tener Git en tu computadora.
```

Esto ha sido una introducción a **Git** y **GitHub**, suficiente para utilizarlo como repositorio en los cursos de programación, pero si lo que necesitas es usarlo para control de versiones te recomiendo que consultes el material oficial de Git.

