

# github

May 30, 2016

## Contents

<b>1</b>	<b>Introducción</b>	<b>1</b>
1.1	Beneficios de GitHub . . . . .	2
<b>2</b>	<b>Registro</b>	<b>2</b>
<b>3</b>	<b>Crear un repositorio</b>	<b>4</b>
<b>4</b>	<b>Conectarse a GitHub</b>	<b>5</b>
4.1	Conectarse por SSH . . . . .	5
4.2	Conectarse remotamente mediante la URL . . . . .	6
<b>5</b>	<b>Subir repositorios</b>	<b>7</b>

## 1 Introducción

Es una plataforma de desarrollo colaborativo, para alojar proyectos utilizando el sistema de control de versiones Git. Todos los códigos se almacenan de forma pública, en caso de que no se desee esto, tambien se pueden subir de forma privada, pero con la diferencia de que se debe subir con una cuenta de pago. Los proyectos que esten almacenados en forma pública pueden recibir contribuciones de terceros para mejorarlos, ya que ellos pueden descargar alguna de las versiones almacenadas (forks), y manipularlas a su antojo; una vez realizadas las modificaciones, el desarrollador las puede enviar al dueño del proyecto (pull), este Ãºltimo tendrá la desición de adjuntar los cambios al proyecto original.

## 1.1 Beneficios de GitHub

1. Permite revertir archivos a un estado anterior.
2. Permite revertir el proyecto entero a un estado anterior.
3. Comparar cambios realizados a lo largo del proyecto.
4. Control total de los cambios (nombre del usuario, fecha, etc.)
5. Recuperar cambios en el proyecto

## 2 Registro

a) Primeramente en la página principal se ingresa en la opción 'Signup and Pricing' de la barra de herramientas de la página.



Figure 1: Página principal de GitHub

b) Seguidamente se creará una cuenta gratuita en la opción 'Create a free account', en caso de querer repositorios unicamente accesibles para los miembros de el proyecto, se seleccionará una cuenta de pago.

c) Por último sólo se tendrá que llenar un formulario con los datos del lider del proyecto:

- Nombre de usuario
- Correo electrónico

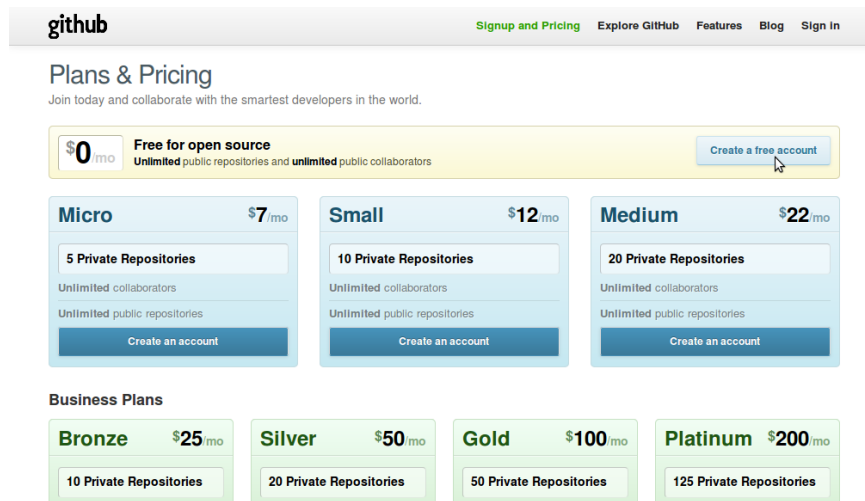


Figure 2: Creación de cuenta nueva en GitHub

- Contraseña
- Confirmación de la contraseña

---

### Create your free personal account

**Username**

**Email Address**

We promise we won't share your email with anyone.

**Password**

Must contain one lowercase letter, one number, and be at least 7 characters long.

**Confirm Password**

By clicking on "Create an account" below, you are agreeing to the [Terms of Service](#) and the [Privacy Policy](#).

Create an account

## 3 Crear un repositorio

Primeramente como requisito para crear un repositorio de algún proyecto, es necesario haber accedido ya al sitio web con algún usuario, posteriormente solo hay que seleccionar el botón "New repository", el cual se encuentra en el lado inferior derecho de la página principal.

Inmediatamente se redireccionara a una página, en donde se deberá llenar un formulario con el nombre del repositorio y una descripción del mismo (opcional). En caso de ser una cuenta de pago, se podrá escoger si el repositorio será privado o público, por otro lado si la cuenta es gratuita, el repositorio solo podrá ser público.

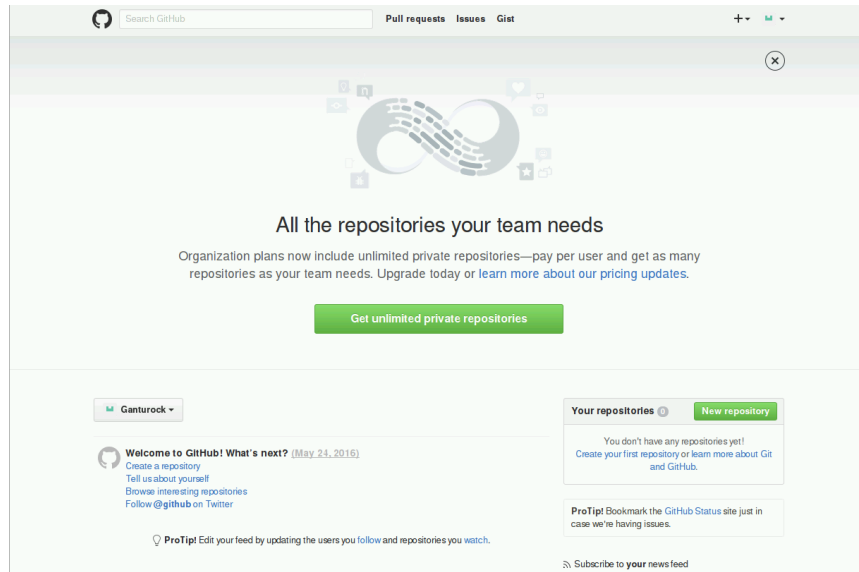


Figure 3: Opción para crear un nuevo repositorio

Una vez creado un repositorio, este estará vacío, por lo que se tiene que crear los archivos que tendrá el proyecto, pero antes de todo eso se debe instalar el git y conectarse a este mediante el protocolo ssh:

## 4 Conectarse a GitHub

Para conectarse a git es necesario que este nos reconozca, para ello se accederá a la configuración del git y se ingresará el nombre de usuario (figura 7) y el correo electrónico (figura 8), es importante que estos dos coincidan con el usuario registrado en la página de GitHub.

### 4.1 Conectarse por SSH

En caso de querer enviar los archivos por ssh, se debe ingresar una llave ssh para conectar la PC al servidor de git, para lograr eso se genera la llave ssh con el comando "ssh-keygen".

*nota: Conectarse mediante llaves SSH no es muy recomendable para proyectos que impliquen varios integrantes, se recomienda conectarse mediante la URL del proyecto.*

Una vez ingresado el comando, este pedirá una dirección para guardar la llave ssh, en caso de no ingresar ninguna dirección, la llave se guardará

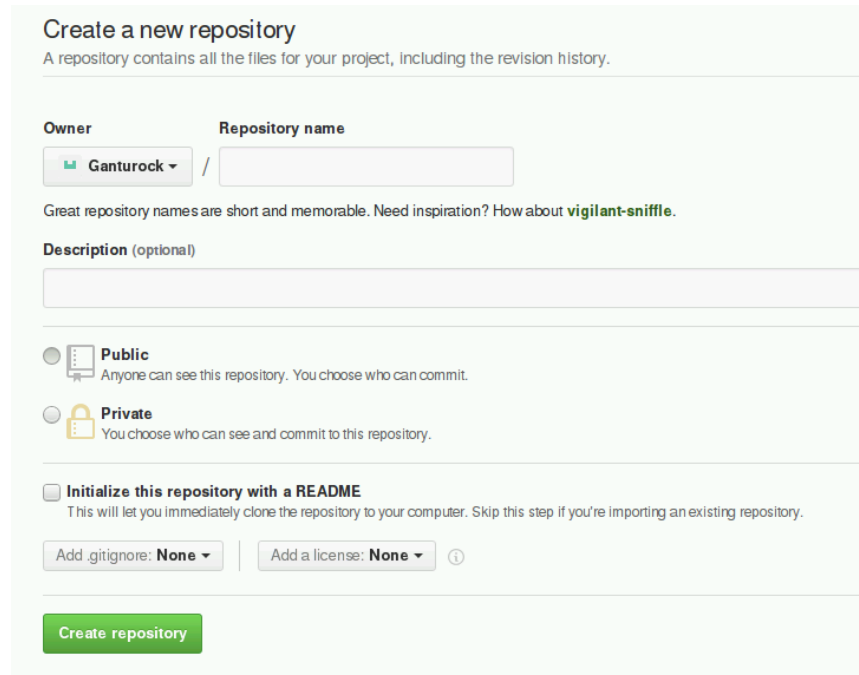
The image shows the GitHub 'Create a new repository' form. At the top, it says 'Create a new repository' and 'A repository contains all the files for your project, including the revision history.' Below this, there are two main sections: 'Owner' and 'Repository name'. The 'Owner' section has a dropdown menu showing 'Ganturock'. The 'Repository name' section has an empty text input field. Below these, there is a note: 'Great repository names are short and memorable. Need inspiration? How about **vigilant-sniffle**.' Then, there is a 'Description (optional)' section with a large text area. Below the description, there are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option has a subtext: 'Anyone can see this repository. You choose who can commit.' The 'Private' option has a subtext: 'You choose who can see and commit to this repository.' Below these, there is a checkbox labeled 'Initialize this repository with a README' and a subtext: 'This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.' At the bottom, there are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None'. Finally, there is a green 'Create repository' button.

Figure 4: Formulario para la creación de un repositorio

automaticamente en la carpeta señalada; tambien requerirá de ingresar una clave de seguridad para la llave y confirmarla, en caso de que no se desee ninguna se tecleará enter sin escribir nada. Esta llave debe ingresarse en la página de GitHub para que su servidor se conecte a la PC, usando el comando "cat" a la llave creada.

Ahora se deberá copiar toda la llave y pegarla en la configuración de GitHub. Para eso se ingresa a la página del perfil del usuario y elegir la opción "Edit profile".

Posteriormente se elige la opción "SSH and GPG keys" - "New SSH", se le asigna un titulo a la llave ssh y en la opción "key" se pega la llave anteriormente consultada.

## 4.2 Conectarse remotamente mediante la URL

El conectarse mediante la URL donde se encuentre alojado el proyecto suele ser el método más usado, aunque un poco menos inseguro comparado con el método de llaves ssh. Para esto se debe añadir la URL del proyecto a la lista de repositorios remotos git y asignarle un nombre para identificar el

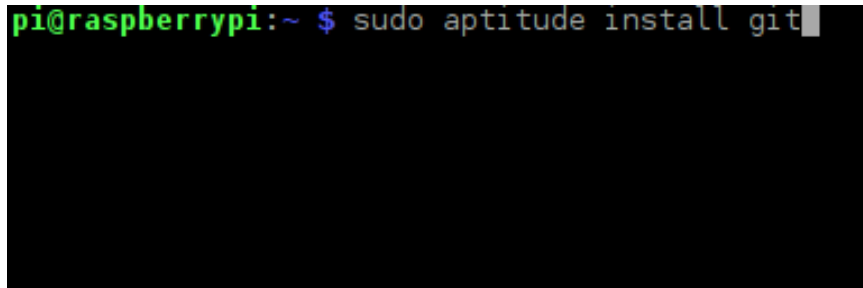


Figure 5: Instalación de git en consola

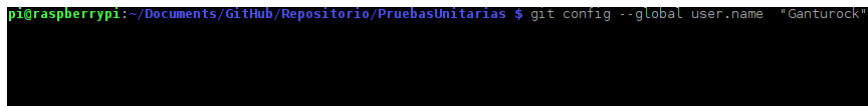


Figure 6: Configuración del nombre de usuario en git

repositorio remoto, entonces la estructura seria: *\$: git remote add [Nombre del repositorio remoto] [URL del proyecto]*.

Para confirmar que se ha añadido con éxito la URL del proyecto a la lista de repositorios remotos de git se consulta la lista con el comando *\$: git remote -v*

Sí por alguna razón se desea renombrar el nombre con el que se identifica el repositorio remoto, se ingresa el comando *\$: git remote rename [nombre actual del repositorio remoto] [Nuevo nombre del repositorio remoto]*, y para eliminar alguno de la lista de repositorios remotos seria *\$: git remote rm [nombre del repositorio remoto]*

## 5 Subir repositorios

Antes de hacer alguna cosa en Git, es necesario tener un repositorio creado, pues de otra forma Git no tendría forma de almacenar el código que se subirá a un repositorio, al igual que no podría identificar los cambios realizados en el código del proyecto. Para crear esta carpeta se debe estar localizado en

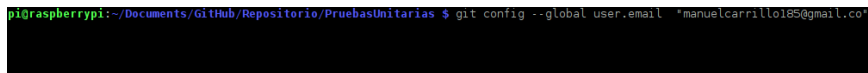


Figure 7: Configuración del correo electrónico en git

```

pi@raspberrypi:~/Documents/GitHub/Repositorio/PruebasUnitarias $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pi/.ssh/id_rsa): /home/pi/Documents/LlaveSSH
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/pi/Documents/LlaveSSH.
Your public key has been saved in /home/pi/Documents/LlaveSSH.pub.
The key fingerprint is:
c5:ea:0a:ab:3f:13:cc:1d:0a:72:5d:a2:68:54:50:6f pi@raspberrypi
The key's randomart image is:
+---[RSA 2048]----+
| .+o                |
|  o  .  .          |
| .. o E  o         |
| o.+ o .  o        |
| .o + o .S         |
|   = ..           |
| .. .             |
| oo .             |
| .ooo.            |
+-----+
pi@raspberrypi:~/Documents/GitHub/Repositorio/PruebasUnitarias $

```

Figure 8: Generar llave ssh

```

pi@raspberrypi:~/Documents/GitHub/Repositorio/PruebasUnitarias $ cat ~/Documents/LlaveSSH.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQAC/2f/3WAfZfGsiRcfrw2pZSrCForazZsjQEyyvNxSVEkv+AaCnG7Qkzz
ZMLY6jOSn1L7V2YLYYGxyYCEWYGZ0kcNVL0MFKNESaoZNCq2hA2mZSsLXj0wIuP7fwSo7E+BCXLDpzSE9/kawkbjIXv9wr
Qd3rViZhBcSkzTvDIInLVZNCKp/nYbJ2qYrBAdXltxFAQbJhmzRvANJ/0V02yR/m5aRve48ZfT0z pi@raspberrypi

```

Figure 9: Consultar llave ssh

la carpeta donde se encuentran los repositorios y escribir el comando `$: git init`.

En seguida se puede notar que hay una carpeta oculta llamada `.git`, en la cual se almacenan todos los cambios del proyecto. Lo próximo a hacer es ingresar los cambios al staging area (área de comprobación o de ensayo), en esta área se encuentran los archivos próximos a enviar como repositorio en el próximo envío; para ingresar estos cambios se ingresa `$: git add [Nombre del archivo]`, en caso de querer agregar recursivamente todos los archivos que se encuentran en una carpeta se ingresa el comando `$: git add .`

En cualquier momento se puede consultar el estado de los archivos en el directorio de trabajo y en el staging area con el comando `$: git status -s`

En este ejemplo se puede observar el estado de tres archivos, el primer archivo el cual tiene una letra "A" al inicio indica que a sido añadido y permanece en el staging area, por el contrario los otros dos archivos han sido borrados del area (delete); hay otros casos en donde el estado se identifica con las iniciales "AM" las cuales indican que se añadió el archivo pero se modifico en el disco local desde que se agrego. Una vez agregados todos los archivos en el staging area se tendrán que confirmar los cambios con el comando `$: git commit -m "[comentarios]"`, se usa la opción `/-m` para agregar un mensaje



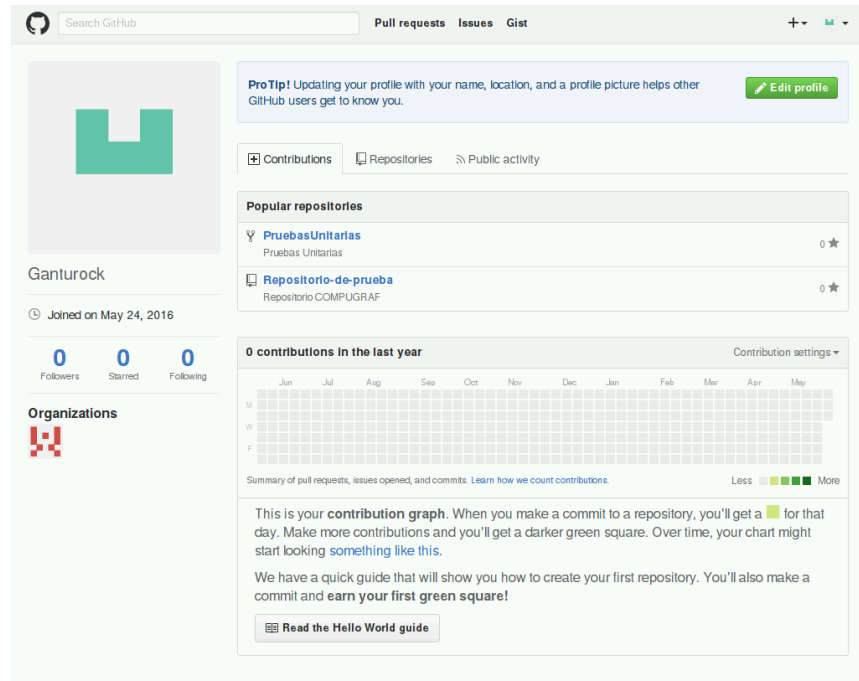


Figure 10: Editar perfil del usuario git

al mismo tiempo que se realizan los cambios, sí no se elige la opción `-m`, git automaticamente abraira un editor de texto para ingresar el mensaje:

En caso de que se desee deshacer el último cambio de confirmación en el commit y sacar del staging area algunos archivos (archivos a los que se les hizo un git add), se deberá usar el comando `/ $: git reset HEAD -[nombre del archivo]`, ejemplo:

Con git reset lo que en realidad se hace es resetear las entragas del archivo en el index, para que sean iguales a las del commit anterior

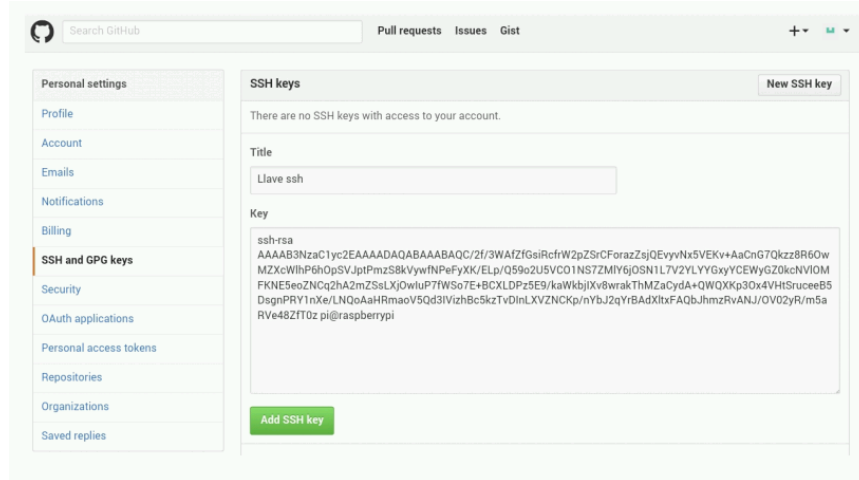


Figure 11: Ingresar llave ssh

```
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas/RepoBajar $ git remote add RepoPrueba https://github.com/ganturock/Repositorio-de-prueba.git
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas/RepoBajar $
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas/RepoBajar $ git remote -v
RepoManuel      https://github.com/Ganturock/Repositorio-de-prueba.git (fetch)
RepoManuel      https://github.com/Ganturock/Repositorio-de-prueba.git (push)
RepoPrueba      https://github.com/Ganturock/Repositorio-de-prueba.git (fetch)
RepoPrueba      https://github.com/Ganturock/Repositorio-de-prueba.git (push)
```

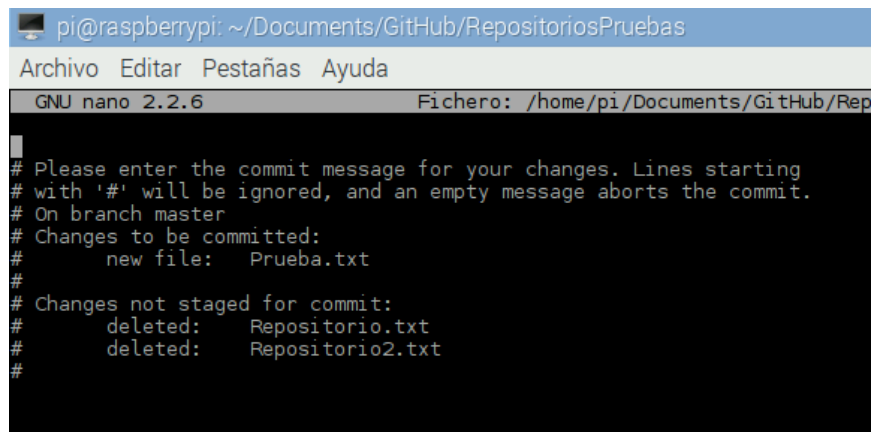
Figure 12: Ingresar URL del repositorio remoto

```
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ echo "Esto es una prueba" > Prueba.txt
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ ls
Prueba.txt
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ git init
Reinitialized existing Git repository in /home/pi/Documents/GitHub/RepositoriosPruebas/.git/
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ ls
Prueba.txt
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ ls -la
total 16
drwxr-xr-x 3 pi pi 4096 may 26 17:47 .
drwxr-xr-x 5 pi pi 4096 may 26 17:32 ..
drwxr-xr-x 8 pi pi 4096 may 26 17:48 .git
-rw-r--r-- 1 pi pi   19 may 26 17:47 Prueba.txt
```

Figure 13: Comando git init

```
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ git add Prueba.txt
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $ git status -s
A Prueba.txt
D Repositorio.txt
D Repositorio2.txt
pi@raspberrypi:~/Documents/GitHub/RepositoriosPruebas $
```

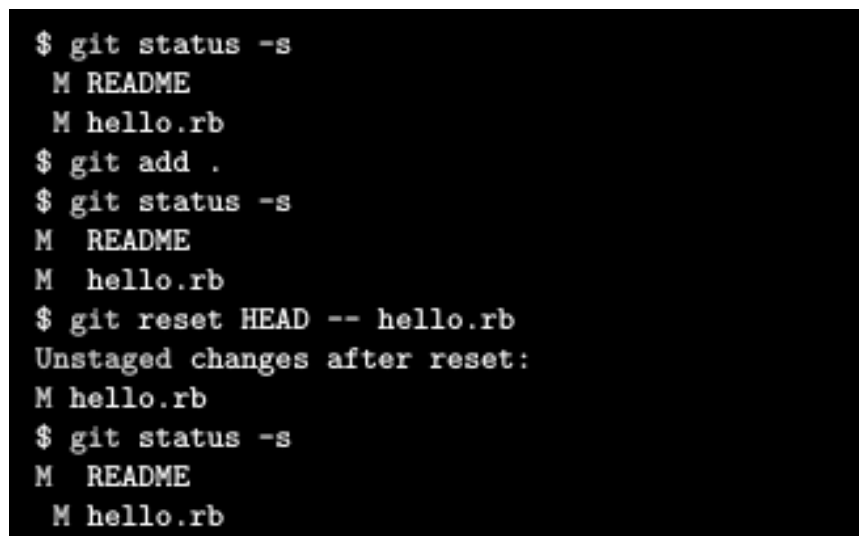
Figure 14: Comando git status y git add



The screenshot shows a terminal window with the nano text editor open. The title bar indicates the user is 'pi' on a 'raspberrypi' machine, in the directory '~/Documents/GitHub/RepositoriosPruebas'. The menu bar includes 'Archivo', 'Editar', 'Pestañas', and 'Ayuda'. The status bar shows 'GNU nano 2.2.6' and the file path '/home/pi/Documents/GitHub/Rep'. The editor content displays the standard git commit prompt, including instructions on commit messages and a summary of changes: 'new file: Prueba.txt' and 'deleted: Repositorio.txt', 'deleted: Repositorio2.txt'.

```
pi@raspberrypi: ~/Documents/GitHub/RepositoriosPruebas
Archivo Editar Pestañas Ayuda
GNU nano 2.2.6 Fichero: /home/pi/Documents/GitHub/Rep
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Changes to be committed:
#   new file:   Prueba.txt
#
# Changes not staged for commit:
#   deleted:    Repositorio.txt
#   deleted:    Repositorio2.txt
#
```

Figure 15: Comando git commit



The screenshot shows a terminal window with the following sequence of commands and their outputs: 'git status -s' showing 'M README' and 'M hello.rb'; 'git add .' adding the changes; 'git status -s' showing the same status; 'git reset HEAD -- hello.rb' resetting the file; and a subsequent 'git status -s' showing 'M README' and 'M hello.rb', indicating that 'hello.rb' remains staged for commit.

```
$ git status -s
M README
M hello.rb
$ git add .
$ git status -s
M README
M hello.rb
$ git reset HEAD -- hello.rb
Unstaged changes after reset:
M hello.rb
$ git status -s
M README
M hello.rb
```

Figure 16: Comando git reset HEAD