



Git

Sistema de Control de Versiones  
Víctor Manuel Barceló Nieves  
DACB UJAT

# ¿Qué es Git?



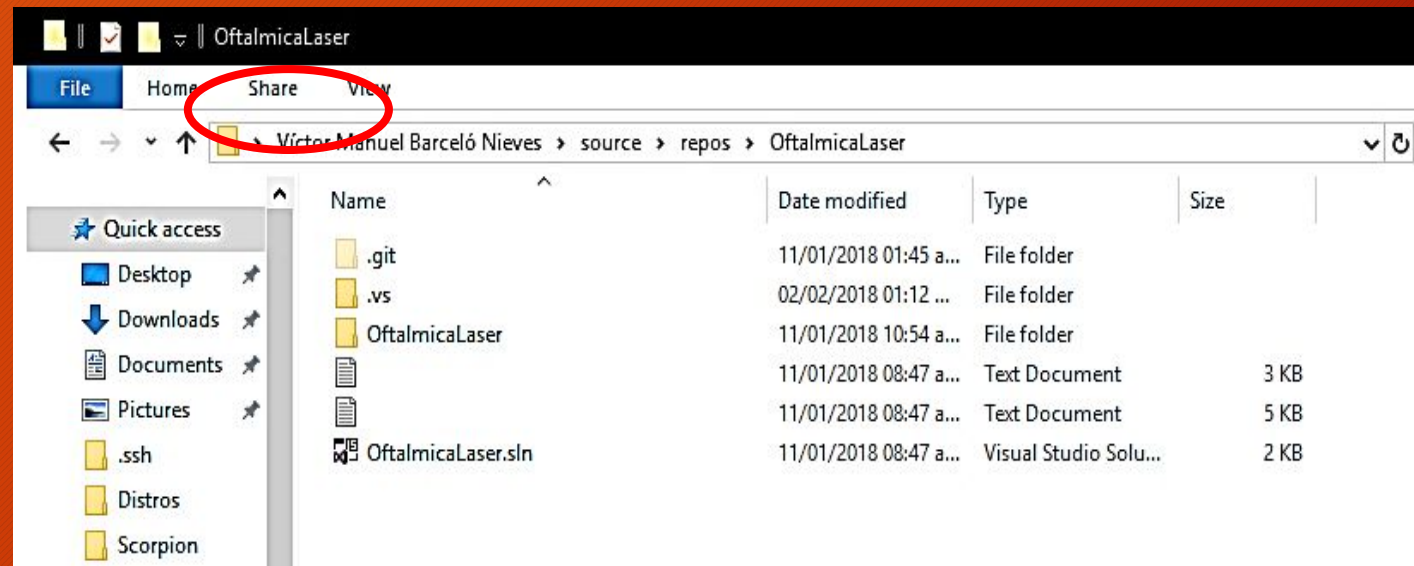
- Sistema de Control de Versiones
- ¿Qué es un SCV?
  - Un esquema de seguimiento de cambios de proyectos, carpetas y/o archivos
- Fue creado por Linus Torvalds en 2005 para el desarrollo del kernel de Linux



# ¿Cómo funciona Git?



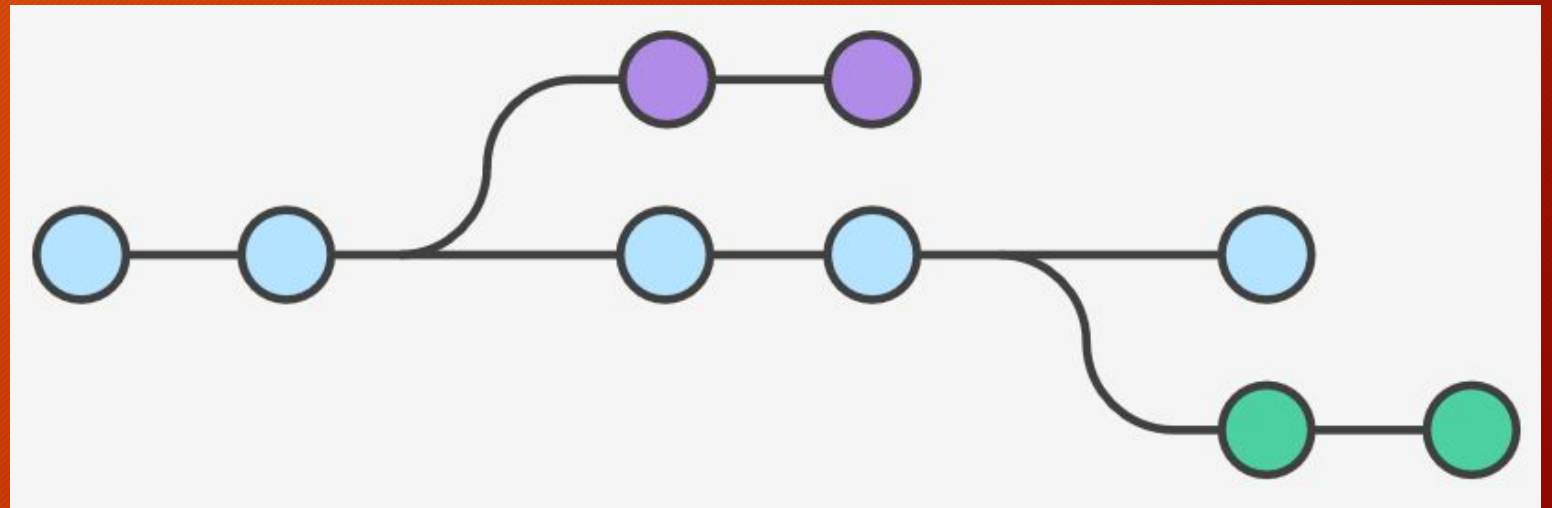
- Fundamentalmente, con una estructura de datos llamada **Repositorio**
- Todos los proyectos contienen una carpeta llamada **.git** en la raíz
- Esta carpeta contiene toda la información sobre la historia del proyecto



# Commit

- Son los objetos cuyos cambios se guardan en el historial del repositorio
- Todo dentro del proyecto de Git está basado en commits que se manejan en el repositorio

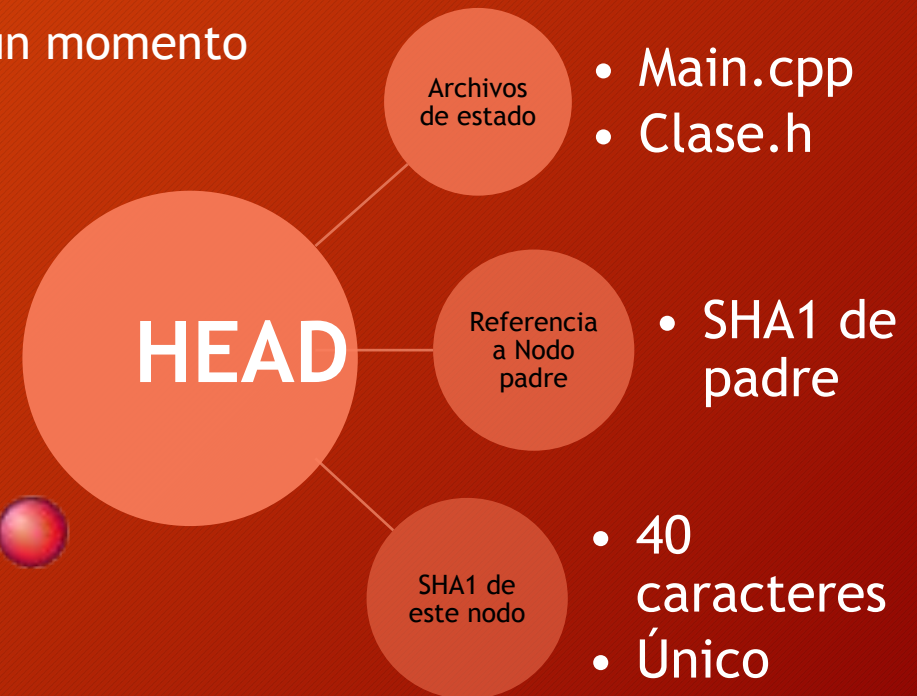
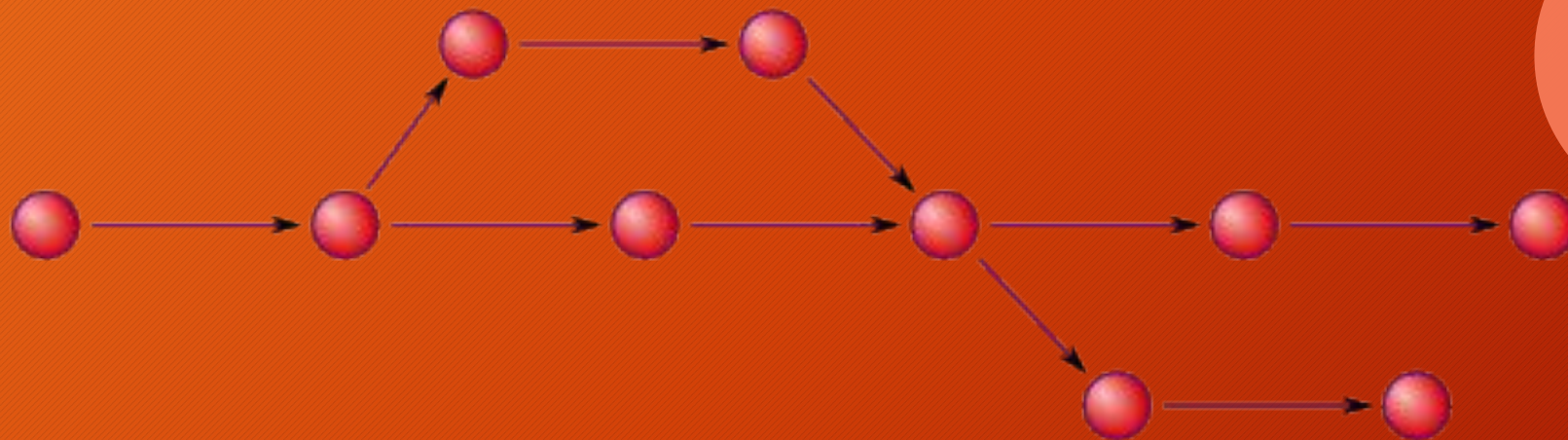
**Initial  
commit**





# Commit

- Grafo acíclico dirigido
- Cada commit guarda:
  - Una serie de archivos reflejando el estado del proyecto es un momento dado
  - Una referencia a un commit padre
  - Un nombre único SHA1 de 40 caracteres





# En la práctica ...

- Crear un repositorio

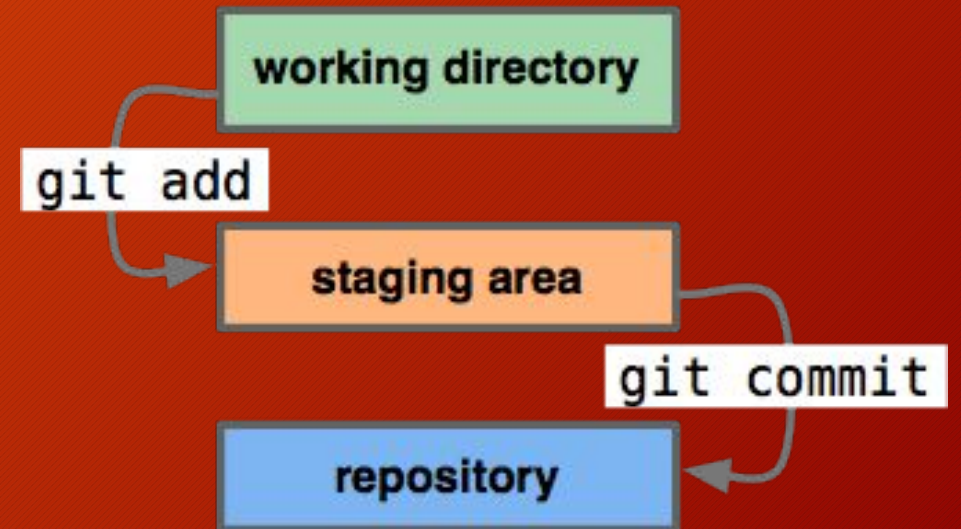
```
mkdir "proyecto"  
cd "proyecto"  
git init
```

- Crear un commit

```
git commit -a ó git add [archivos]  
"Mensaje de commit" (OBLIGATORIO)
```

- Típico flujo de trabajo:

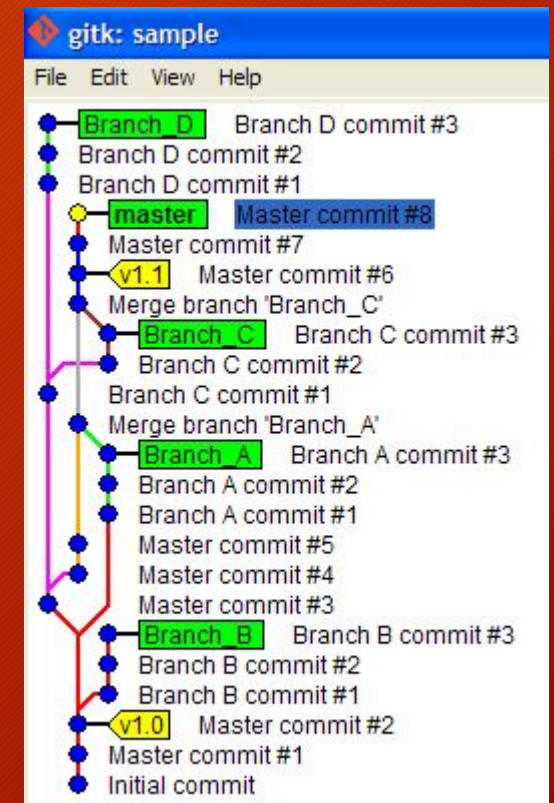
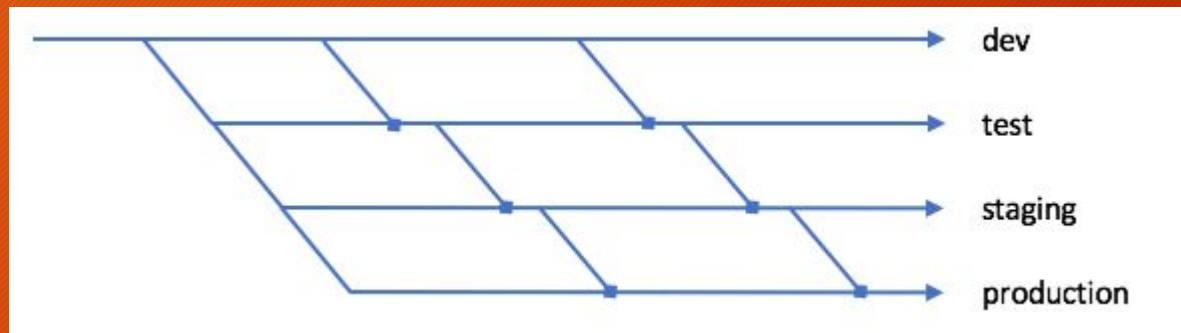
- Escribir código
- `git status` (ver cambios)
- `git diff [archivo]` (ver cambio)
- `git commit -a -m "Mensaje"`





# Branching

- Es posible crear crear “ramas” del proyecto en cualquier momento de la historia del proyecto
- Flexibilidad
- Trabajo en paralelo



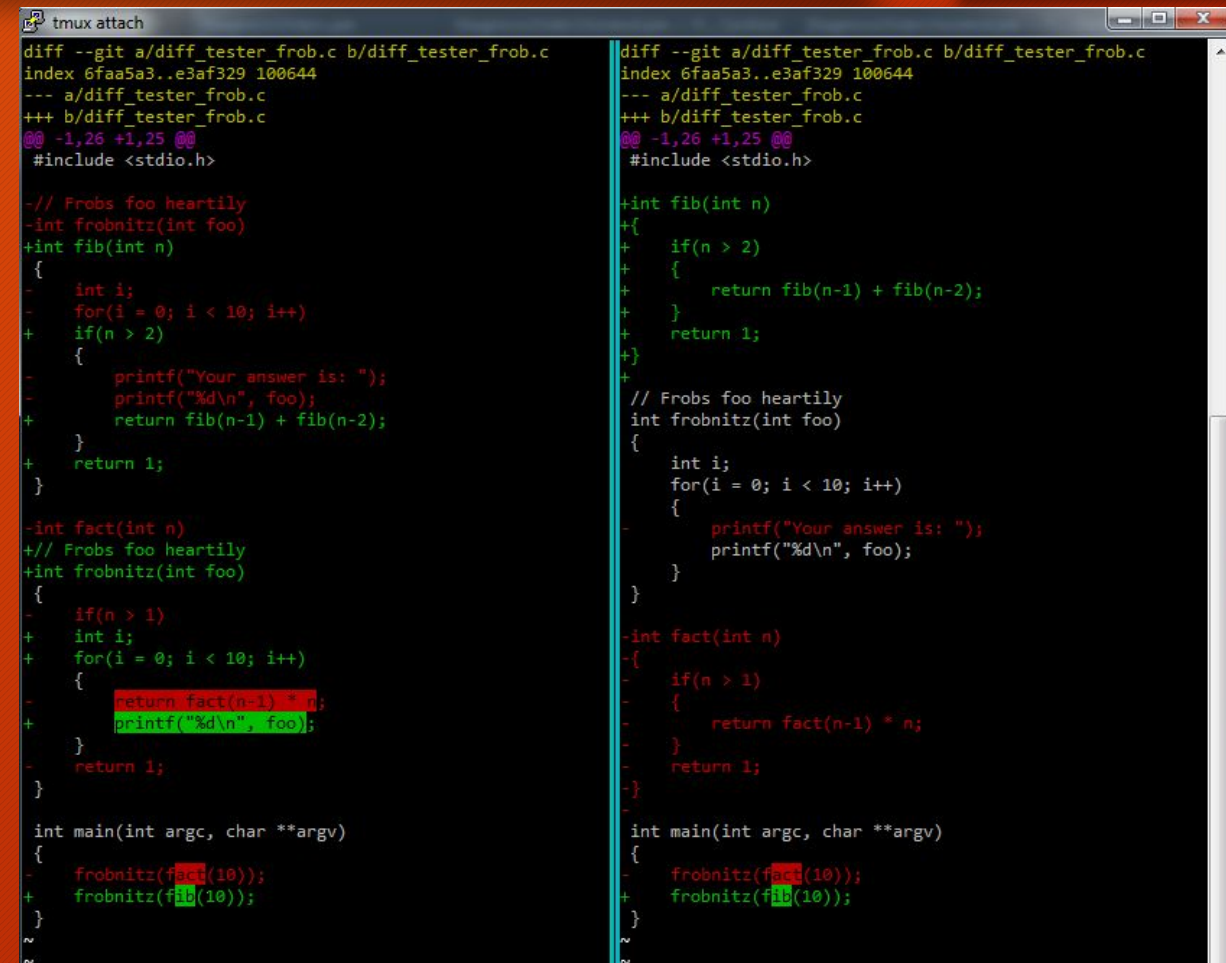
# Branching

- Cambiar de rama

`git checkout [rama]`

- Diff entre ramas

`git diff [rama1] [rama2]`



```
tmux attach
diff --git a/diff_tester_frob.c b/diff_tester_frob.c
index 6faa5a3..e3af329 100644
--- a/diff_tester_frob.c
+++ b/diff_tester_frob.c
@@ -1,26 +1,25 @@
#include <stdio.h>

-// Frobs foo heartily
-int frobnitz(int foo)
+int fib(int n)
+{
+    if(n > 2)
+    {
+        return fib(n-1) + fib(n-2);
+    }
+    return 1;
+}
+
+// Frobs foo heartily
+int frobnitz(int foo)
+{
+    int i;
+    for(i = 0; i < 10; i++)
+    {
+        printf("Your answer is: ");
+        printf("%d\n", foo);
+        return fib(n-1) + fib(n-2);
+    }
+    return 1;
+}

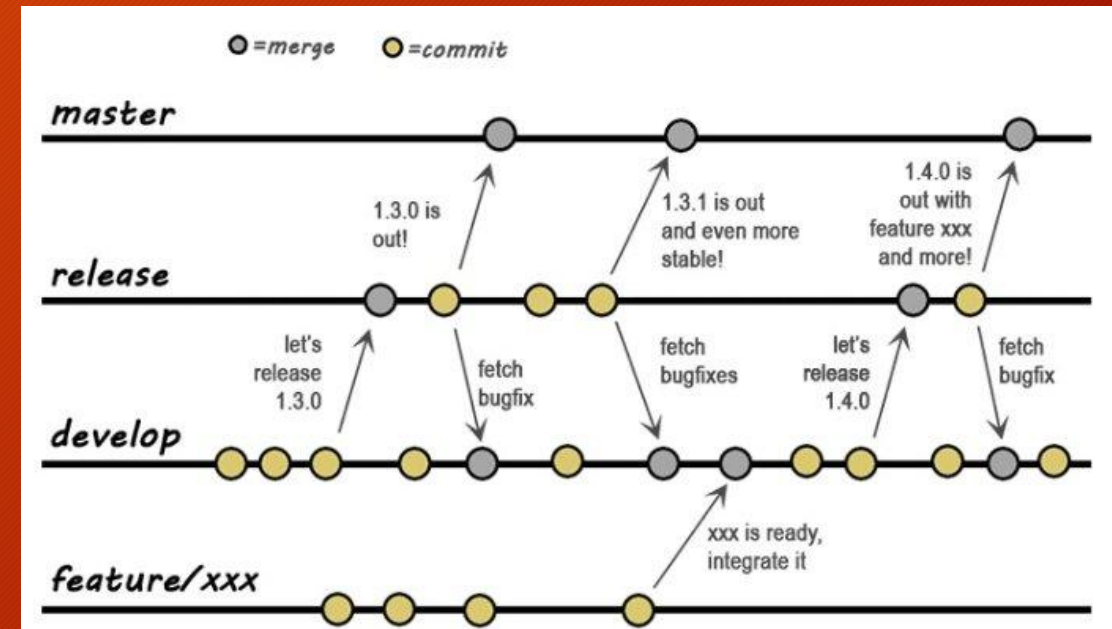
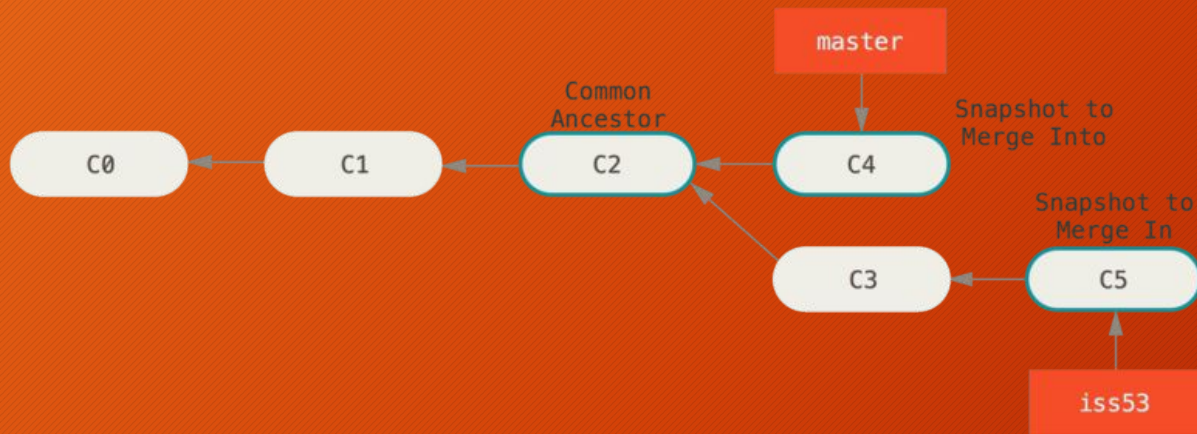
-int fact(int n)
+int main(int argc, char **argv)
+{
+    frobnitz(fact(10));
+    frobnitz(fib(10));
+}

~
```



# Merging

- “Unir” “Juntar”
- Implementar cambios de la rama  
`git merge [rama-destino]`





# Colaboración (Modelo distribuido)

- Para clonar un repositorio a tu sistema:

```
git clone [www.direcciondelrepositorio.com/...]
```

- Para enviar cambios que quieres que tus colaboradores vean\*:

```
git push [repositorio-remoto]
```

- Para recibir cambios remotos\*:

```
git fetch ó pull[repositorio-remoto]
```

\*Para poder hacer push y pull debes tener comprometidos (committed) tus cambios



# Servicios para resguardo de repositorios



# Tutoriales en línea

- <https://git-scm.com/book/es/v1/Empezando>
- <http://rogerdudler.github.io/git-guide/index.es.html>
- <https://www.youtube.com/watch?v=xNYNpkUe9Uc>
- <http://blog.santiagobasulto.com.ar/programacion/2011/11/27/tutorial-de-git-en-espanol.html>