

# **MANUAL DE PRIMEROS PASOS - GIT**

INDICE

CAPITULO 01 .....3

1. GIT .....4

2. iniciando .....6

3. Adicional .....31

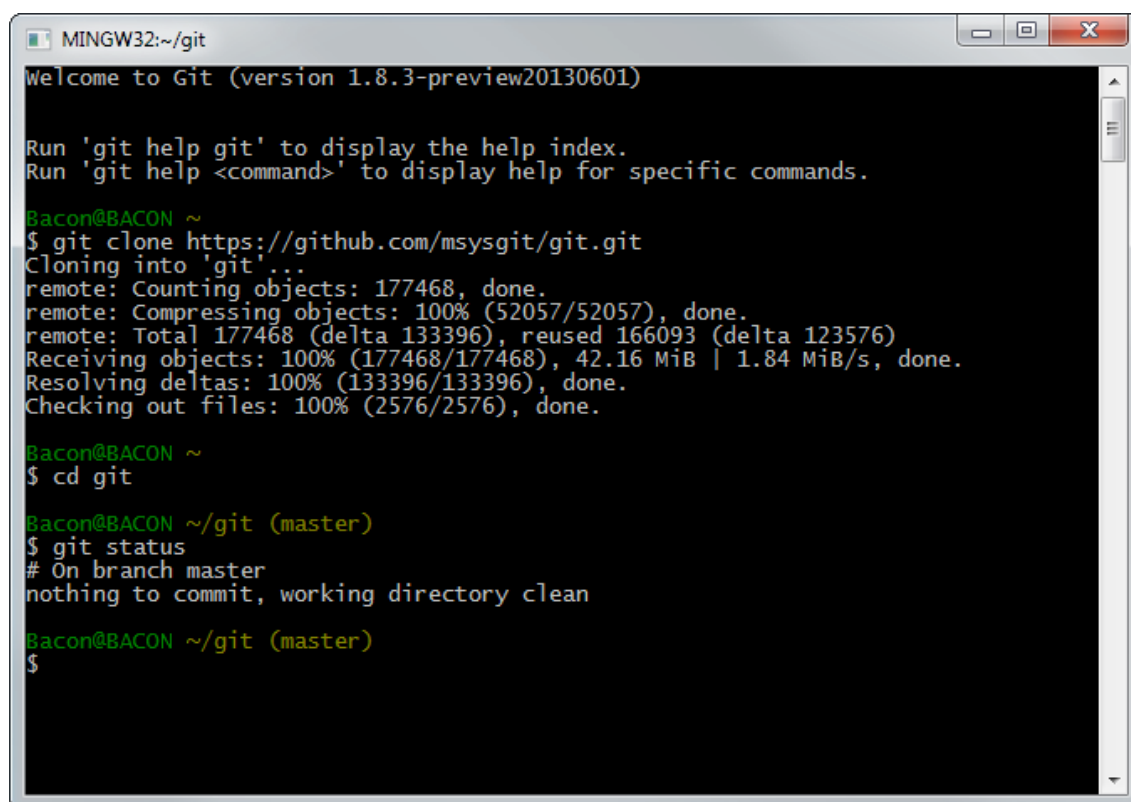
# CAPITULO 01

## 1. GIT

Git para Windows se centra en ofrecer un conjunto ligero, nativo de herramientas que traen el conjunto completo de características de la Git SMC a Windows mientras que proporciona interfaces de usuario adecuadas para los usuarios novatos y experimentados por igual Git.

### 1.1. Git BASH

Git para Windows proporciona una emulación BASH utilizado para ejecutar Git desde la línea de comandos. \* NIX usuarios deben sentirse como en casa, como la emulación BASH se comporta igual que el comando "git" en entornos Linux y Unix.



```
MINGW32:~/git
Welcome to Git (version 1.8.3-preview20130601)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Bacon@BACON ~
$ git clone https://github.com/msysgit/git.git
Cloning into 'git'...
remote: Counting objects: 177468, done.
remote: Compressing objects: 100% (52057/52057), done.
remote: Total 177468 (delta 133396), reused 166093 (delta 123576)
Receiving objects: 100% (177468/177468), 42.16 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (133396/133396), done.
Checking out files: 100% (2576/2576), done.

Bacon@BACON ~
$ cd git

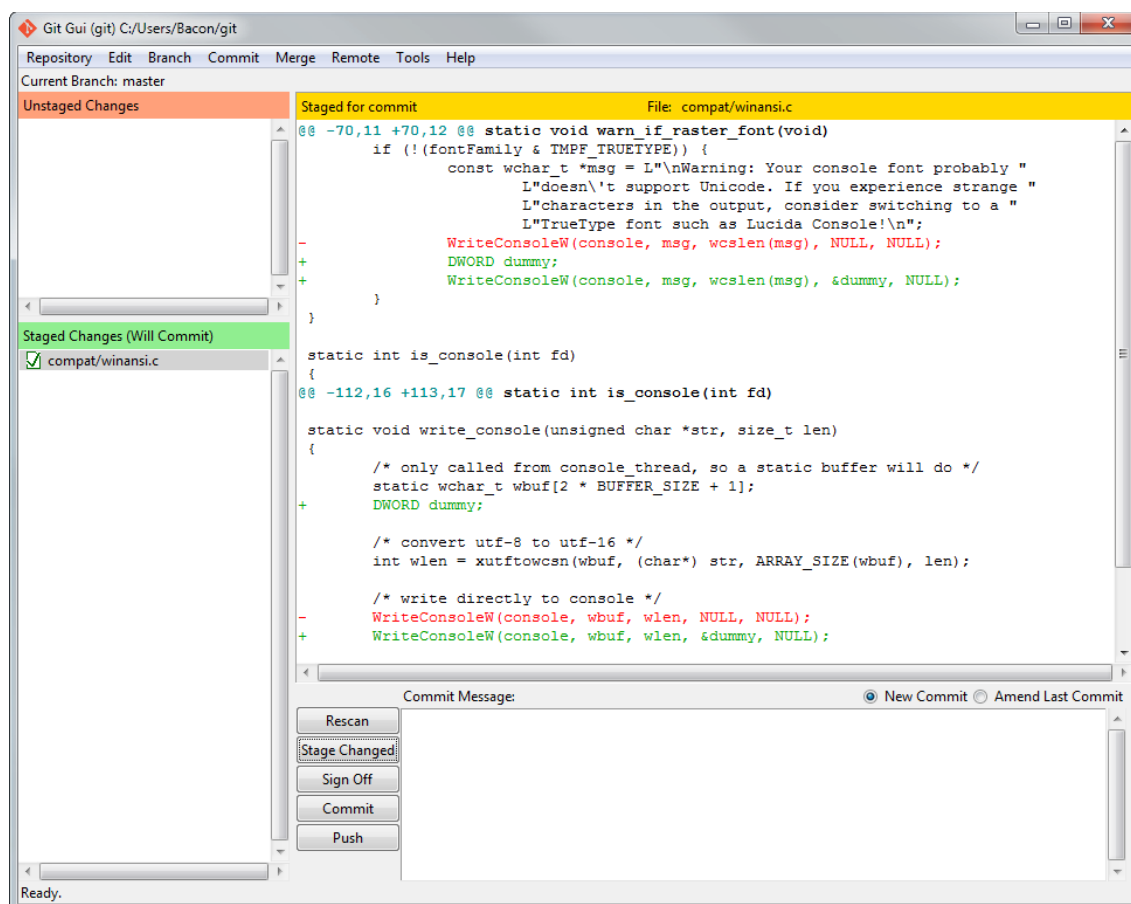
Bacon@BACON ~/git (master)
$ git status
# On branch master
nothing to commit, working directory clean

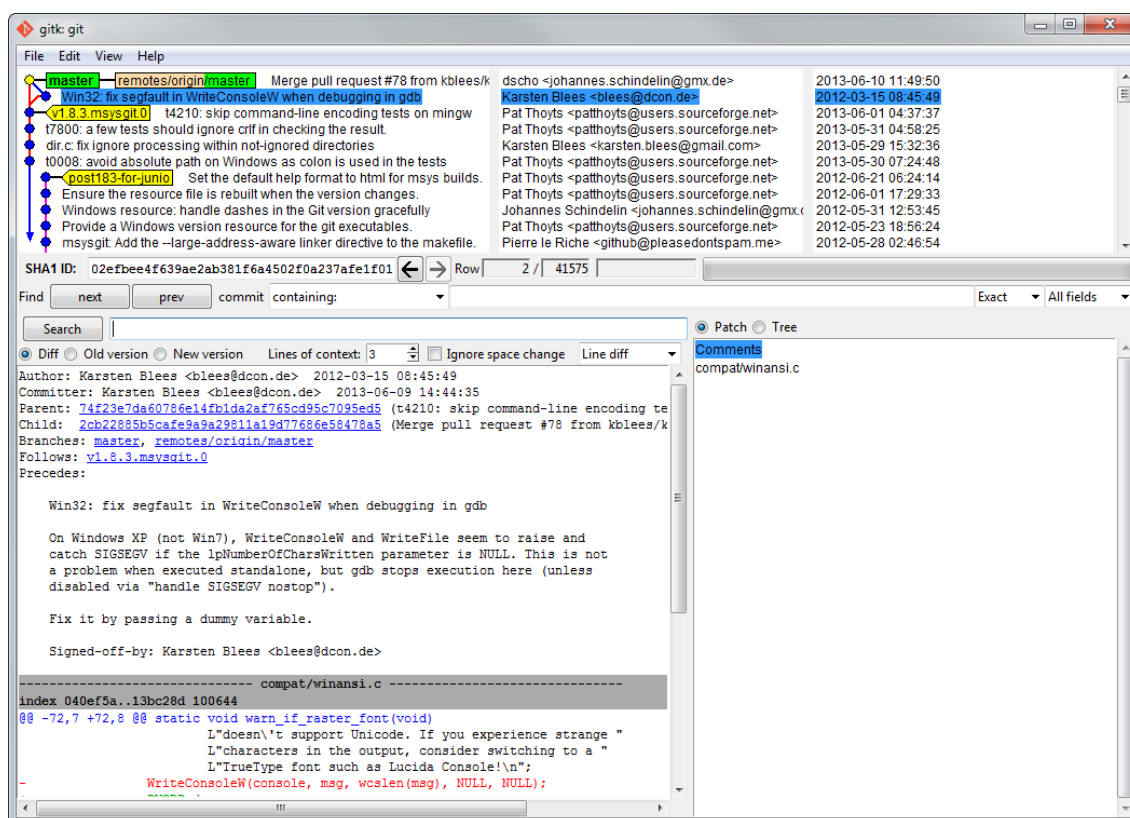
Bacon@BACON ~/git (master)
$
```

### 1.2. Git GUI

A medida que los usuarios de Windows comúnmente esperan interfaces gráficas de usuario, Git para Windows también proporciona la interfaz gráfica de usuario de Git, una poderosa alternativa a Git Bash, ofreciendo una versión gráfica de casi

todas las funciones de línea de comandos de Git, así como herramientas completas visual de diferencias.





### 1.3. Shell Integration

Simplemente haga clic en una carpeta en el Explorador de Windows para acceder a la interfaz gráfica de usuario o BASH.

How to Use Git Extensions

<https://www.youtube.com/watch?v=jLOeS8dZTUM>

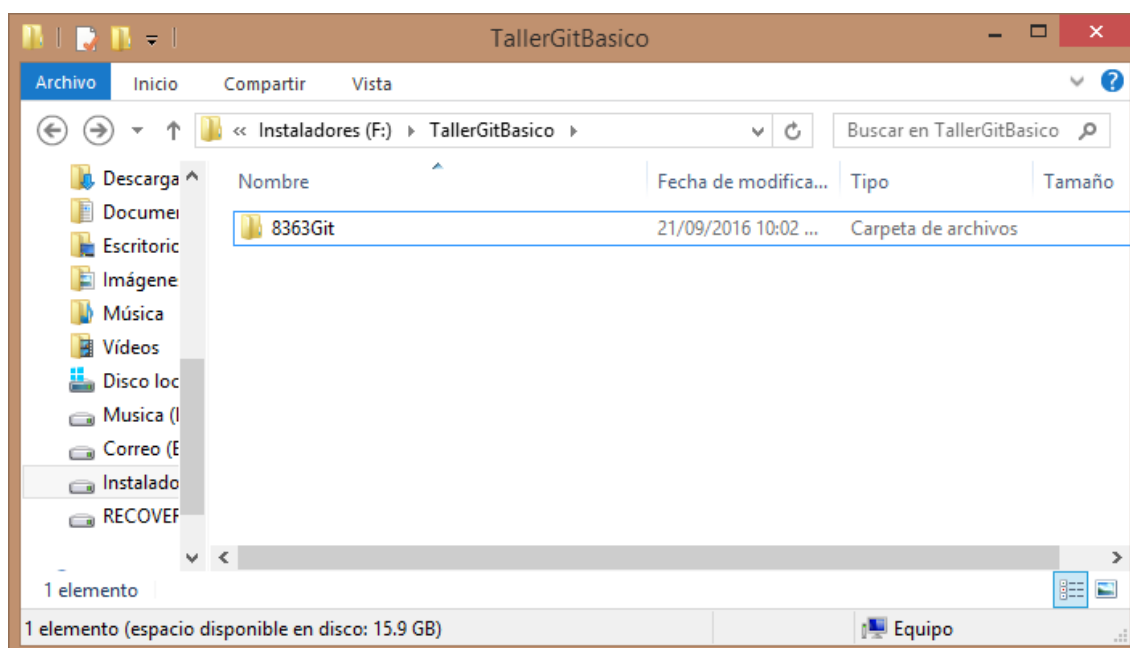
Introduction to GIT with Git Extensions on Bitbucket or Assembla (SSH Key Setup)

<https://www.youtube.com/watch?v=cFbCusX9bKs>

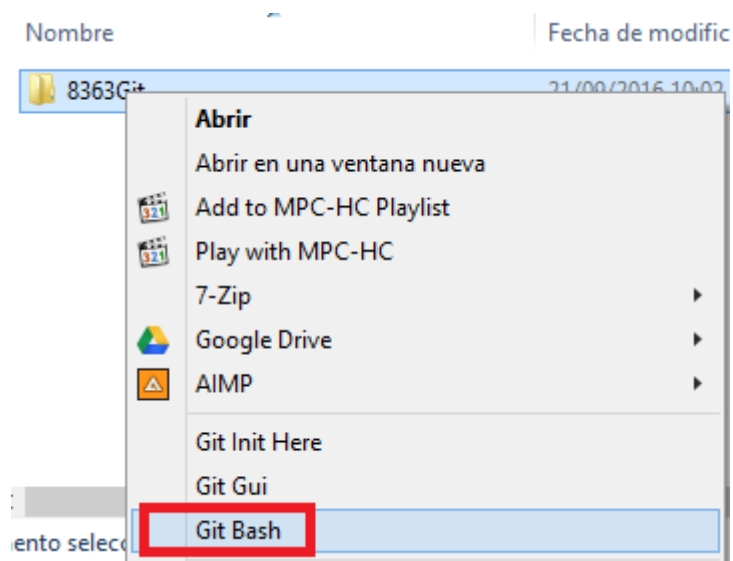
## 2. Iniciando

### 2.1. Creando un Repositorio Local

Crear una carpeta compuesta código de alumno mas la palabra GIT



Abrir el Git GUI



```

MINGW32:/F/TallerGitBasico/8363Git
Welcome to Git (version 1.9.5-preview20150319)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git
$

```

Iniciando por comando

Si desean escribir el comando aparte y pegar.

```

MINGW32:/F/TallerGitBasico/8363Git
Welcome to Git (version 1.9.5-preview20150319)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git
$ -

```

Generamos el repositorio

```

MINGW32:/F/TallerGitBasico/8363Git
Welcome to Git (version 1.9.5-preview20150319)

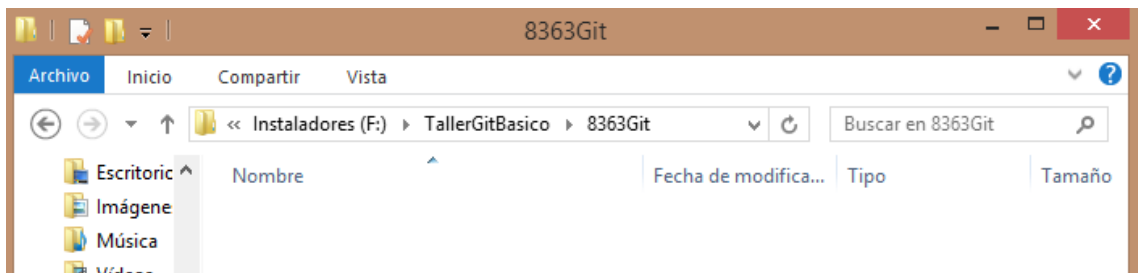
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git
$ git init
Initialized empty Git repository in f:/TallerGitBasico/8363Git/.git/

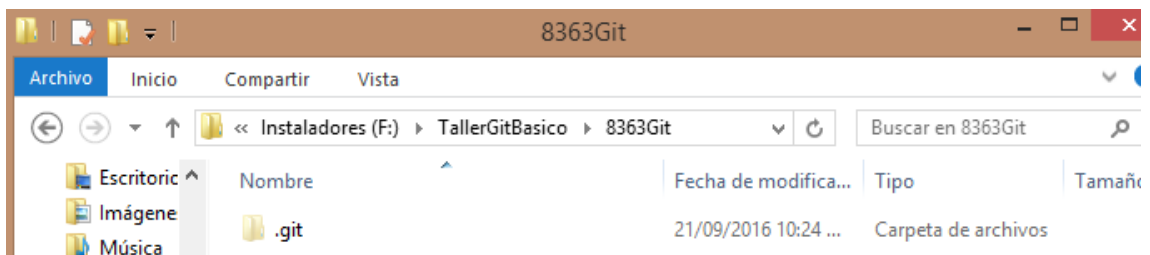
```



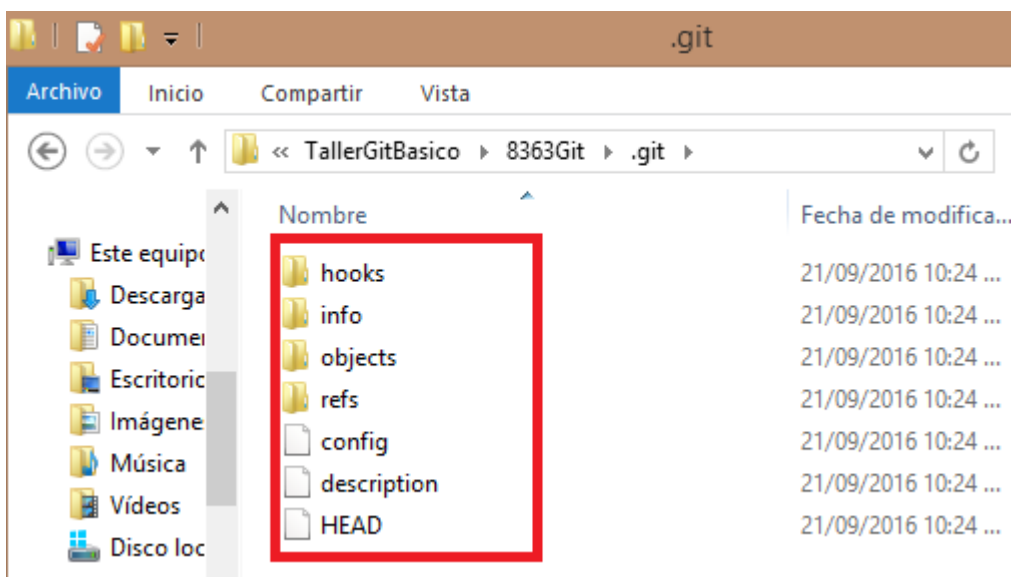
Antes



Después

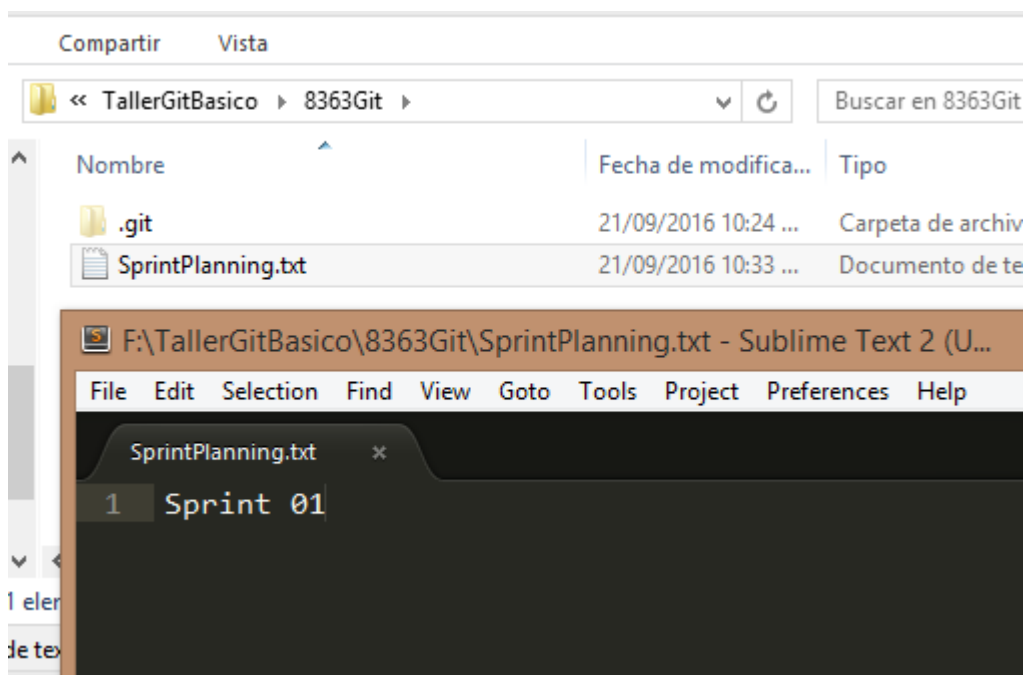


Dentro de la carpeta se genera los archivos propios del git.



## 2.2. Generando un archivo en el repositorio iniciado

Creamos un archivo y su contenido



Revisamos el status de dicho repositorio

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master

Initial commit

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

    SprintPlanning.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Adicionaremos al repositorio el archivo pues a pesar de estar ahí, no lo hemos adicionado al seguimiento del GIT.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add SprintPlanning.txt

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$
```

Volemos a revisar el estatus

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add SprintPlanning.txt

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

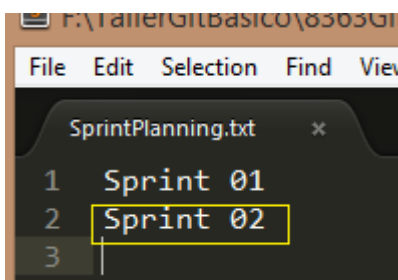
    new file:   SprintPlanning.txt

```

Nos indica que dicho documento ahora si esta puesto bajo el GIT.

### 2.3. Modificación del archivo

Modificamos el archivo colocando el sprint 02



```

SprintPlanning.txt
1 Sprint 01
2 Sprint 02
3

```

Revisemos el estatus

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   SprintPlanning.txt

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:   SprintPlanning.txt

```

Notamos que las modificaciones realizadas al archivo sin el editor propio del GIT no están en el estado staged (unstaged).

Adicionaremos estos cambios al estado Staged

```
Juan@LAPTOP-JTINOC01 /F/TallerGitBasico/8363Git (master)
$ git add SprintPlanning.txt

Juan@LAPTOP-JTINOC01 /F/TallerGitBasico/8363Git (master)
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   SprintPlanning.txt
```

## Procedemos a hacer nuestro primer commit

```
Juan@LAPT-JTIN0C01 /F/TallerGitBasico/8363Git (master)
$ git commit
```

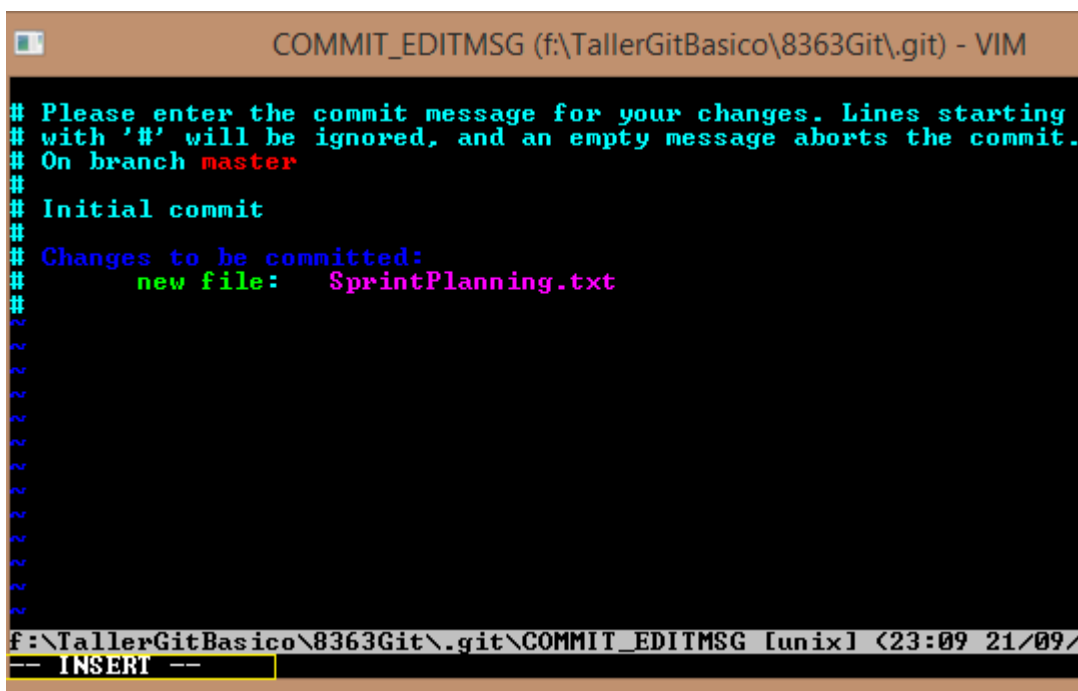
Por defecto se abrirá el

Revisemos el estatus del archivo que estamos modificando

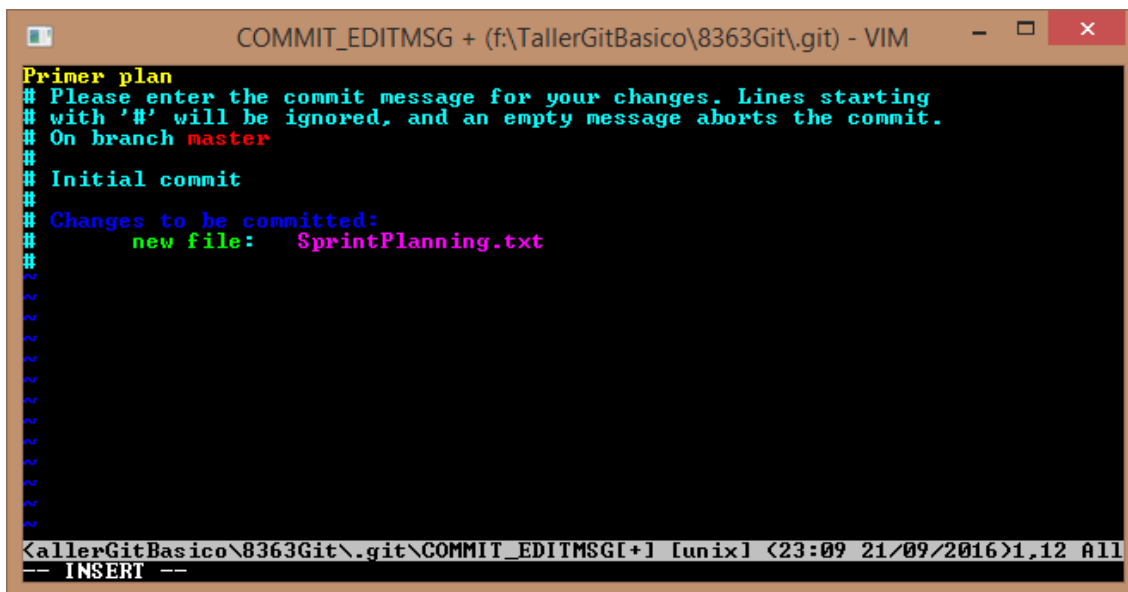
```
COMMIT_EDITMSG (f:\TallerGitBasico\8363Git\.git) - VIM
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
#
# Initial commit
#
Changes to be committed:
#   new file:   SprintPlanning.txt
~
~
~
~
~
~
~
~
~
~
~
~
```

<\TallerGitBasico\8363Git\.git\COMMIT\_EDITMSG [unix] <23:09 21/09/20

Debemos escribir digitar la tecla insertar para empezar la edición



Debemos escribir el mensaje que identifique el commit.



Debemos teclear ESC y luego :WQ para terminar el editor VIM.

[illegible]

Volveran a la pantalla anterior donde se nota el cambio realizado

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git (master)
$ git commit
[master (root-commit) f59acb4] Primer plan
1 file changed, 2 insertions(+)
create mode 100644 SprintPlanning.txt

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git (master)
$
```

Revisamos el status para comprobar que ya estamos sin modificaciones

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit
[master (root-commit) f59acb4] Primer plan
1 file changed, 2 insertions(+)
create mode 100644 SprintPlanning.txt

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master
nothing to commit, working directory clean
```

Verificamos en el log

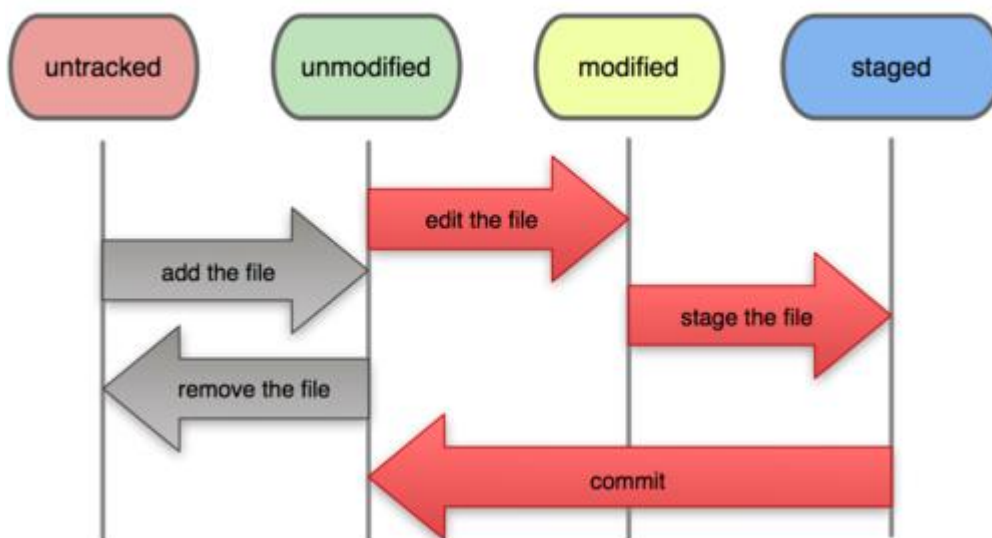
```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git log
commit f59acb413d2865a3a8b15478940cf360c33f1200
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date:   Wed Sep 21 23:09:17 2016 -0500

    Primer plan

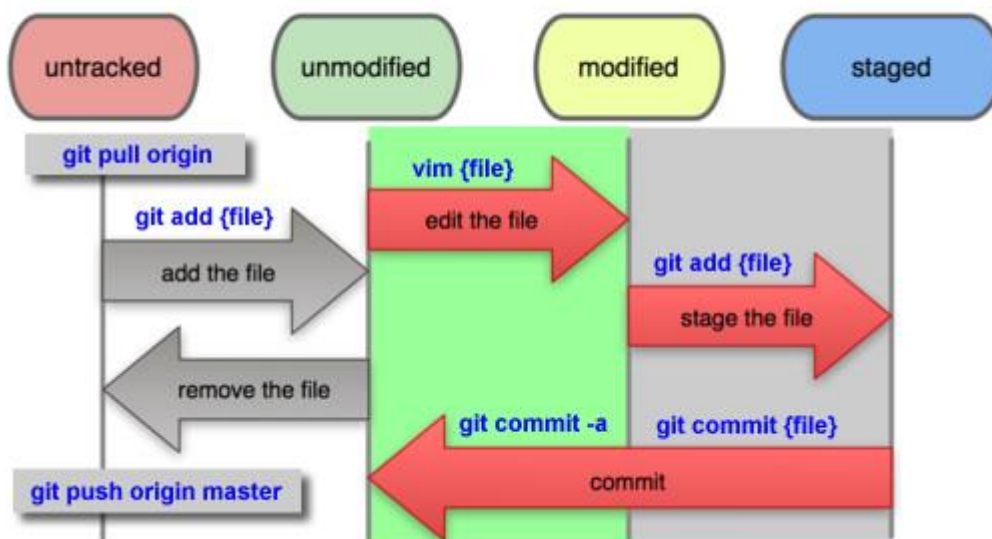
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$
```

## 2.4. REVISION DE ESTADOS.

## File Status Lifecycle

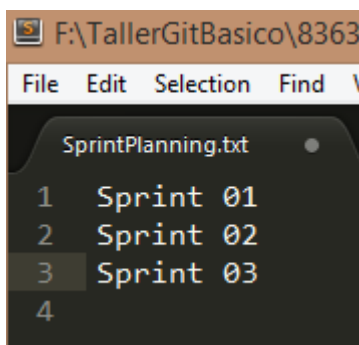


## File Status Lifecycle



## 2.5. Versión reducida del cambio en el archivo.

Modifico el archivo



Luego adicionamos el archivo y hacemos el commit directo.

Evitamos el paso de ingresar al editor en donde debemos colocar el comentario para el commit.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add SprintPlanning.txt

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit -m 'segundo commit'
[master 92c0bf8] segundo commit
1 file changed, 1 insertion(+)
```

Para verificar los commit via el log.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit -m 'segundo commit'
[master 92c0bf8] segundo commit
1 file changed, 1 insertion(+)
```

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git log
commit 92c0bf8bc75dbeddbda720709cc33047c0a9c87d
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:28:46 2016 -0500

    segundo commit

commit f59acb413d2865a3a8b15478940cf360c33f1200
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:09:17 2016 -0500

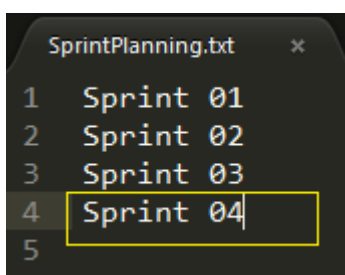
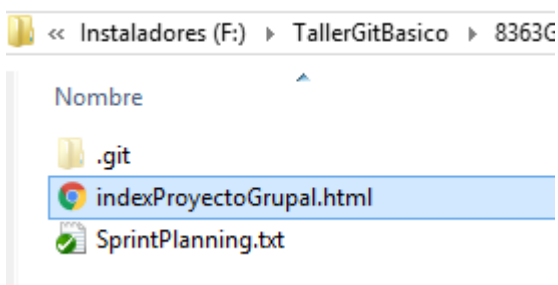
    Primer plan
```

## 2.6. Adicionando varios cambios a la vez.

Creamos un archivo indexProyectoGrupal.html y modificamos el archivo

SprintPlanning.txt





Revisamos el status de mi repositorio y nos mostrara que hemos modificado dos archivos y que están pendientes de ingresarlos al estado STAGE.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch 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:   SprintPlanning.txt

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

        indexProyectoGrupal.html

no changes added to commit (use "git add" and/or "git commit -a")
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$
```

Debemos adicionar todos los cambios y verificamos que se encuentran previo al commit.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add .
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

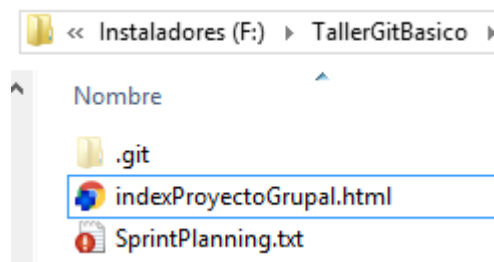
        modified:   SprintPlanning.txt
        new file:   indexProyectoGrupal.html
```

Estos archivos quedaran en este estado para poder aprender a usar el .gitignore

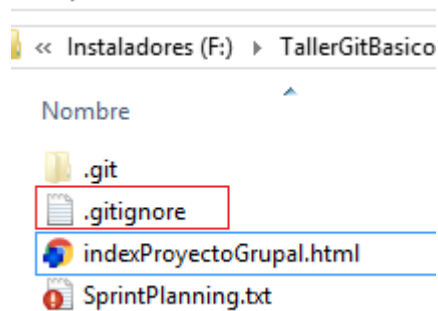
## 2.7. Generar el archivo .gitignore

En windows no se puede generar un archivo sin nombre y con extensión .gitignore por lo cual se usara un comando para la generación de dicho archivo.

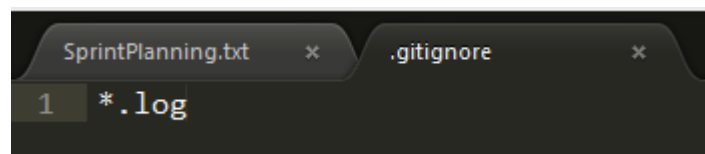
Antes



Después



Modificamos el archivo .gitignore



Adicionaremos los cambios al estado stage y luego el commit de los cambios.

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ touch .gitignore

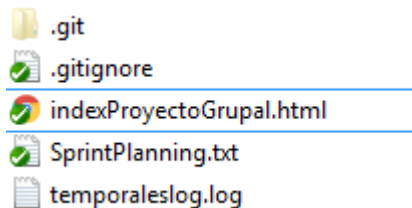
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add .

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit -m 'tercer commit'
[master 87deed1] tercer commit
3 files changed, 2 insertions(+)
create mode 100644 .gitignore
create mode 100644 indexProyectoGrupal.html

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$

```

Creamos un archivo con extensión .log (temporaleslog.log) y modificamos el html



Intentaremos add temporales.log y luego hacer un commit

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git add temporaleslog.log
The following paths are ignored by one of your .gitignore files:
temporaleslog.log
Use -f if you really want to add them.
fatal: no files added

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit -m 'cuarto commit'
On branch master
nothing to commit, working directory clean

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$

```

Todo archivo con extensión .log es ignorado para sea la adición o hacer un commit.

## 2.8. Generamos Branch

Generaremos un branch que contendrá lo que tenemos generado hasta ahora.

Generamos el primer branch

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git branch primerBranch

```

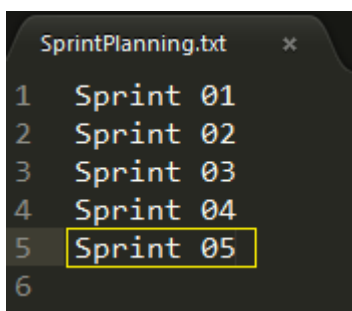
Realizamos el checkout

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git checkout primerBranch
Switched to branch 'primerBranch'
```

Creamos el archivo index.css

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ touch index.css
```

Adicionalmente modificaremos el archivo SprintPlanning.txt



```
SprintPlanning.txt x
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5 Sprint 05
6
```

Lo adicionamos

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git add .
```

Generamos el commit en este branch

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git commit -m 'cuarto commit'
[primerBranch 17d29ce] cuarto commit
2 files changed, 1 insertion(+)
create mode 100644 index.css
```

Verificamos el log y se nota hasta el cuarto commit

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git log
commit 17d29cebfc2c941ab776a8b77322cdbca5709b8a
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Thu Sep 22 00:12:13 2016 -0500

    cuarto commit

commit 87deede94675408d6b720f6604cd015f79d84d4
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:46:08 2016 -0500

    tercer commit

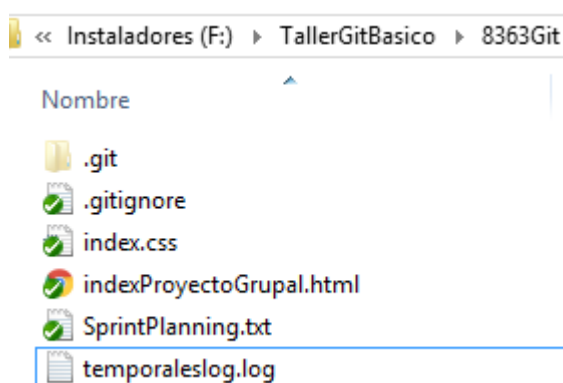
commit 92c0bf8bc75dbeddbda720709cc33047c0a9c87d
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:28:46 2016 -0500

    segundo commit

commit f59acb413d2865a3a8b15478940cf360c33f1200
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:09:17 2016 -0500

    Primer plan
```

Podemos observar en la carpeta los archivos incluido el index.css

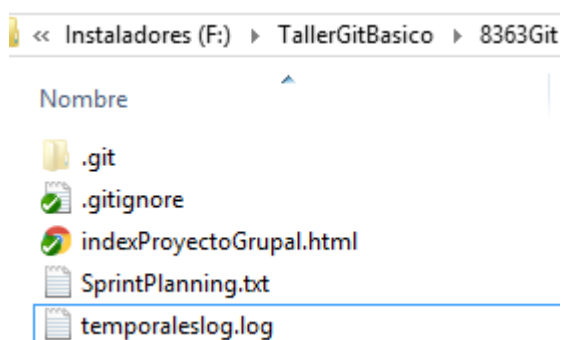


## 2.9. Intercambios entre branch

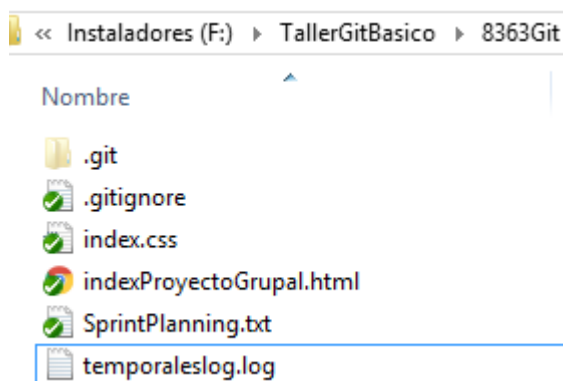
```
Juan@LAPT-ITINOC01 /E/TallerGitBasico/8363Git <primerBranch>  
$ git checkout master  
Switched to branch 'master'
```

Notemos los cambios al realizar un checkout, cambia los archivos del repositorio para poder manejarlos.

Antes con primerbranch



Después con el master



Empezaremos a usar el merge.

## 2.10. Realizaremos un merge sin conflictos.

El archivo SprintPlanning.txt del master tiene el planing hasta el spring 04.

```
SprintPlanning.txt x
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5
```

Mientras el archivo SprintPlanning.txt de mi primer branch tiene hasta el spring 05.

```
SprintPlanning.txt x
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5 Sprint 05
6
```

Adicionalmente el master no tiene el archivo index.css

Nos situamos en el master para realizar el merge que fusionara todos los cambios

Cambiamos al master en caso este en otro branch

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git checkout primerBranch
Switched to branch 'primerBranch'
```

Realizamos el checkout

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git checkout master
Switched to branch 'master'
```

Verificamos donde estamos

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master
nothing to commit, working directory clean
```

Realizamos el merge

Como no hay conflictos por defecto genera las modificaciones indicándonos en resumen dichos cambios.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git merge primerBranch
Updating 87deed..17d29ce
Fast-forward
 SprintPlanning.txt | 1 +
  index.css         | 0
 2 files changed, 1 insertion(+)
 create mode 100644 index.css
```

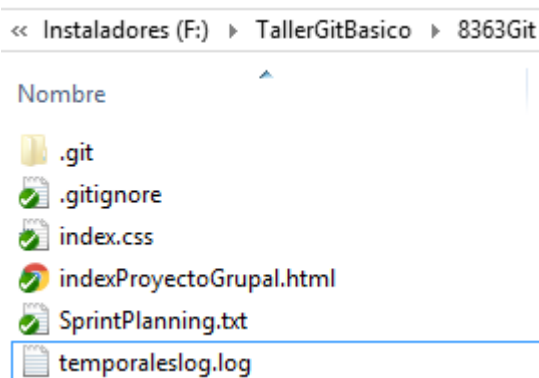
```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git log
commit 17d29cebf2c941ab776a8b77322cdbca5709b8a
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Thu Sep 22 00:12:13 2016 -0500

    cuarto commit

commit 87deede94675408d6b720f6604cd015f79d84d4
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Wed Sep 21 23:46:08 2016 -0500

    tercer commit
```

Notamos en el repositorio como ahora si se incluyó el index.css



## 2.11. Realizaremos un merge con conflictos.

Generamos cambios en cada archivo SprintPlanning.txt de cada branch.

En el master

Para ello usaremos parámetros en el commit para adicionar y generar el mensaje de manera más rápida.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git commit -a -m 'Quinto commit sprint 06'
[master cb444e4] Quinto commit sprint 06
1 file changed, 1 insertion(+)

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git log
commit cb444e4c6377a653a4f80202bafef1f20876d79f9
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Thu Sep 22 00:42:01 2016 -0500

    Quinto commit sprint 06
```

Realizamos lo mismo para primerbranch

Cambios al branch

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git checkout primerBranch
Switched to branch 'primerBranch'
```

Modificamos el archivo

```
SprintPlanning.txt
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5 Sprint 05
6 Sprint 07
7
```

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git commit -a -m 'Sexto commit sprint 07'
[primerBranch 96ef9ea] Sexto commit sprint 07
1 file changed, 1 insertion(+)
```

Verificamos en el log

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git log
commit 96ef9ea2e2d0f3e5371f4f4e2f8df874b1746c55
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Thu Sep 22 00:45:26 2016 -0500
    Sexto commit sprint 07

commit 17d29cebfc2c941ab776a8b77322cdbca5709b8a
Author: PCSIJTIN <PCSIJTIN@gmail.com>
Date: Thu Sep 22 00:12:13 2016 -0500
    cuarto commit
```

Ahora intentamos realizar el merge

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git merge master
Auto-merging SprintPlanning.txt
CONFLICT (content): Merge conflict in SprintPlanning.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Te indica que hubo conflictos y fue fallido el merge.

```
Auto-merging SprintPlanning.txt
CONFLICT (content): Merge conflict in SprintPlanning.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Podemos revisar el status



```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch!MERGING>
$ git status
On branch primerBranch
You have unmerged paths.
  (fix conflicts and run "git commit")

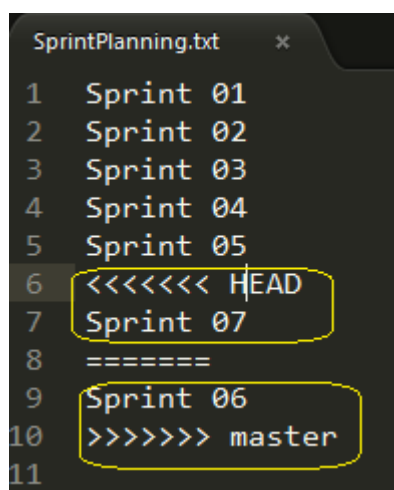
Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   SprintPlanning.txt

no changes added to commit (use "git add" and/or "git commit -a")
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch!MERGING>
$

```

Revisando el archivo en conflicto notamos lo que en el archivo en conflicto se adiciono unos marcadores indicando las líneas en conflicto y su origen.

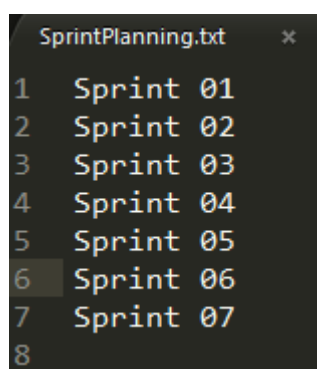


```

SprintPlanning.txt
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5 Sprint 05
6 <<<<<< HEAD
7 Sprint 07
8 =====
9 Sprint 06
10 >>>>>> master
11

```

Para este caso específico modificamos el archivo según como debería realizarse el merge.



```

SprintPlanning.txt
1 Sprint 01
2 Sprint 02
3 Sprint 03
4 Sprint 04
5 Sprint 05
6 Sprint 06
7 Sprint 07
8

```

Realizamos un commit y verificamos el status

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch!MERGING>
$ git commit -a -m 'Septimo commit merge desde el master'
[primerBranch 398dd19] Septimo commit merge desde el master
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$

```

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git status
On branch primerBranch
nothing to commit, working directory clean
```

Lo que logramos es que el branch primerBranch contenga todos los cambios del master ahora que solucionamos el conflicto.

Estos cambios deberíamos usar una interface gráfica pues no muy fácil realizarlo como se hizo vía consola.

Si desean verificar que no tenemos configurado una herramienta para este trabajo usar el comando mergetool

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
tortoisemerge emerge vimdiff
No files need merging
```

## 2.12. Usaremos el estado STAGE

Cuando generamos un archivo en el repositorio este mientras no es adicionado es un archivo que puede ser modificado y que no lo considera durante el checkout que se realiza.

Ejemplo creamos un archivo llamado archivoAdicional.txt

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ touch archivoAdicional.txt
```

Realizamos el checkout al master

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git checkout master
Switched to branch 'master'
```

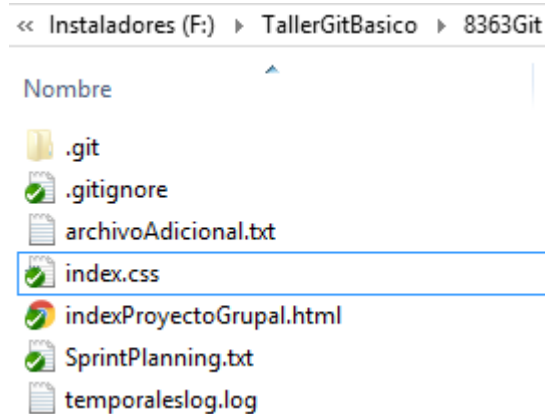
Revisamos el status

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    archivoAdicional.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Notamos que habiendo cambiado del branch al master el archivo llamado archivoAdicional.txt no desapareció, sino persiste en la carpeta al no ser manejada por el git



Realizamos el cambio de branch y procedemos a colocar el archivo adicionado y generando el stash.

Regresamos al branch

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git checkout primerBranch
Switched to branch 'primerBranch'
```

Adicionamos el archivo para su control

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git add .
```

Verificamos el estatus

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git status
On branch primerBranch
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   archivoAdicional.txt
```

Generamos el stash y notaran que desaparece del escritorio

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git stash
Saved working directory and index state WIP on primerBranch: 398dd19 Septimo com
mit merge desde el master
HEAD is now at 398dd19 Septimo commit merge desde el master
```

Ahora para poder volver al stash que estábamos trabajando se le adiciona un parámetro.

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git stash apply
On branch primerBranch
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   archivoAdicional.txt

```

El archivo aparecerá de nuevo en su carpeta

### 2.13. TRABAJANDO CON UN REMOTO

En el bitbucket tenemos un repositorio llamado ProcedimientoSupervision y debemos copiar el link HTTPS

Comenzamos la verificación si tenemos un repositorio remoto para el proyecto

```


Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git remote

```

Obtenemos la dirección de otro proyecto y lo clonamos

FISESUPERVISION / Modelo de Procedimiento / ProcedimientoSupervision


Resumen

 HTTPS <https://PCSIJTIN@bitbucket.org/fisesi>

Última actuali...	2016-08-30	1	0
Idioma	—	Branch	Tags
Nivel de acc...	Administrador (revoke)	0	2
		Forks	Watchers

Invite users to this re

[Send invitation](#)

Actividad reciente 

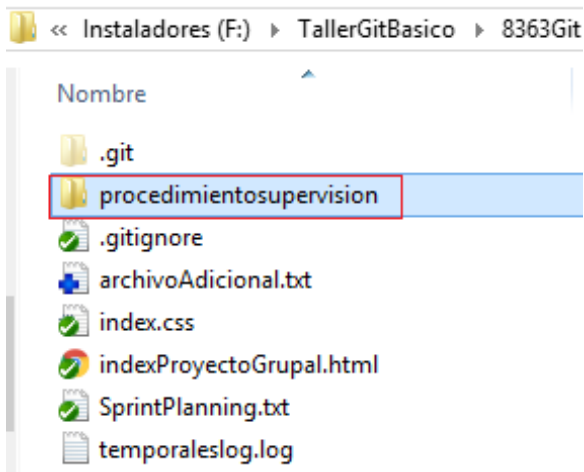
Generamos la clonación

```

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git clone https://PCSIJTIN@bitbucket.org/fisesupervision/procedimientosupervi
sion.git
Cloning into 'procedimientosupervision'...
Password for 'https://PCSIJTIN@bitbucket.org':
remote: Counting objects: 30, done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 30 (delta 18), reused 0 (delta 0)
Unpacking objects: 100% (30/30), done.
Checking connectivity... done.

```

Esta clonación trae hasta el historia de donde clonamos. Notar que se genera las carpetas del repositorio origen



Hasta este momento todavía no tenemos ningún remoto definido.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git remote
```

Pero si nos vamos a la carpeta clonada y hacemos el mismo comando veremos que si tiene un repositorio remoto.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ cd procedimientosupervision
```

Usamos el comando remote y ya hay una fuente lejana

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git remote
origin
```

Si deseamos ver a detalle el lugar remoto.

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git remote -v
origin https://PCSIJTINE@bitbucket.org/fisesupervision/procedimientosupervision.git <fetch>
origin https://PCSIJTINE@bitbucket.org/fisesupervision/procedimientosupervision.git <push>
```

Indicando las dos manera de interactúa.

Traer los cambios para luego realizar el merger

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git fetch origin
Password for 'https://PCSIJTINE@bitbucket.org':
```

Traer los cambios y que se realice el merge en un solo instante

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git pull origin
Password for 'https://PCSIJTINE@bitbucket.org':
Already up-to-date.
```

De la documentación:

`git pull` is shorthand for `git fetch` followed by `git merge FETCH_HEAD`.

Haciendo una traducción libre:

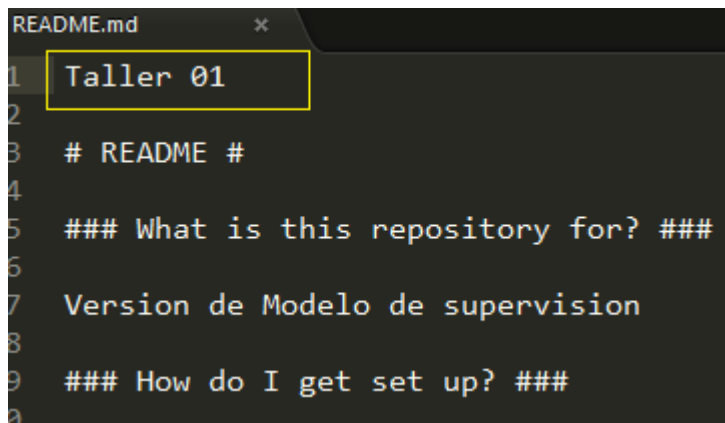
`git pull` es una abreviación de `git fetch` seguido de `git merge FETCH_HEAD`.

Es decir, `git fetch` trae los cambios, pero los deja en otro *branch*, hasta que se hace el `git merge` para traerlos al *branch* local.

<http://es.stackoverflow.com/questions/245/cu%C3%A1l-es-la-diferencia-entre-pull-y-fetch-en-git>

Modificaremos un archivo y lo subiremos al repositorio remoto.

El archivo a modificar es el README.MD



```

1  Taller 01
2
3  # README #
4
5  ### What is this repository for? ###
6
7  Version de Modelo de supervision
8
9  ### How do I get set up? ###
10
  
```

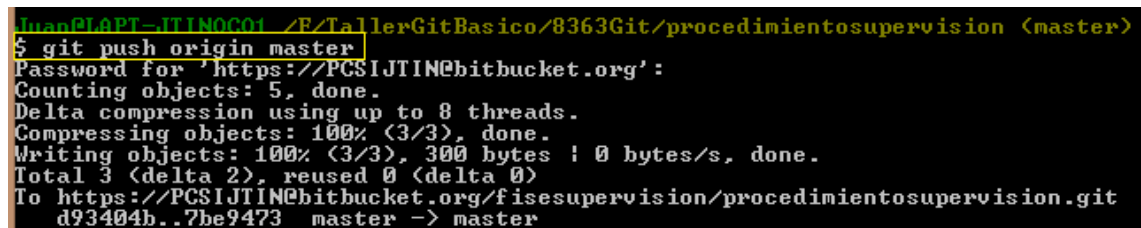
Procedemos a guardar este cambio.



```

Juan@LAPT-JTINOCO1 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git commit -a -m 'cambios al readme'
[master 7be9473] cambios al readme
1 file changed, 2 insertions(+)
  
```

Ahora enviamos los cambios al repositorio



```



Juan@LAPT-JTINOCO1 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git push origin master
Password for 'https://PCSIJTIN@bitbucket.org':
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 300 bytes | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
To https://PCSIJTIN@bitbucket.org/fisesupervision/procedimientosupervision.git
d93404b..7be9473  master -> master
  
```

Revisamos en el remote y notaremos los cambios realizados

FISESUPERVISION / Modelo de Procedimiento / ProcedimientoSupervision

## Commits

Todas las ramas ▾

Autor	Commit	Mensaje
 PCSI JTIN	7be9473	cambios al readme
 PCSI JTIN	d93404b	Septima Versión

Adicionar un nuevo repositorio a nuestro remote del repositorio

Para esto de usa el comando git remote add

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git remote add miRepo https://PCSIJTIN@bitbucket.org/desarrollowebupc/webfeyalegria.git
```

Verificamos los repositorios remotos que tenemos

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git remote
miRepo
origin

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git/procedimientosupervision <master>
$ git remote -v
miRepo https://PCSIJTIN@bitbucket.org/desarrollowebupc/webfeyalegria.git (fetch)
miRepo https://PCSIJTIN@bitbucket.org/desarrollowebupc/webfeyalegria.git (push)
origin https://PCSIJTIN@bitbucket.org/fisesupervision/procedimientosupervision.git (fetch)
origin https://PCSIJTIN@bitbucket.org/fisesupervision/procedimientosupervision.git (push)
```

### 3. Adicional

Borrar un branch perdiendo todos los cambios

```
Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <primerBranch>
$ git checkout master
Switched to branch 'master'

Juan@LAPT-JTINOC01 /F/TallerGitBasico/8363Git <master>
$ git branch -D primerBranch
Deleted branch primerBranch (was 27c5b38).
```