Universidad del Rosario

Git branches

y un poquito más . . .

Alejandro Uribe

Matemáticas aplicadas y ciencias de la computación

February 10, 2020

Qué es git?

Git es un software de control de versiones diseñado por Linus Torvalds.

Github

GitHub es una compañía global que proporciona alojamiento para el control de versiones de desarrollo de software usando Git

Almacenamiento en git

Lo que hace Git es incluír una copia de todos los archivos para cada commit, excepto para el contenido presente en el repositorio de Git, la copia simplemente señalará el contenido en vez de duplicarlo. Esto tambien nos dice que varios archivos con el mismo contenido solo se almacenan una vez.

Una copia basicamente es un commit(Confirmación).

Estados de git

Modified (Modificado)

Los archivos fueron modificados pero aún no se han almacenado en la base de datos

Ejemplos:

- Modificar un archivo de texto.
- Cambiar una linea de código.

Estados de git

Staget (Preparado)

Los archivos que han sido marcados en su versión actual para que vayan en tu próxima confirmación.

En Git por lo general usamos el comando *git add nombreArchivo* o reemplazamos el ”nombreArchivo” por un punto (.) para agregar todos los archivos.

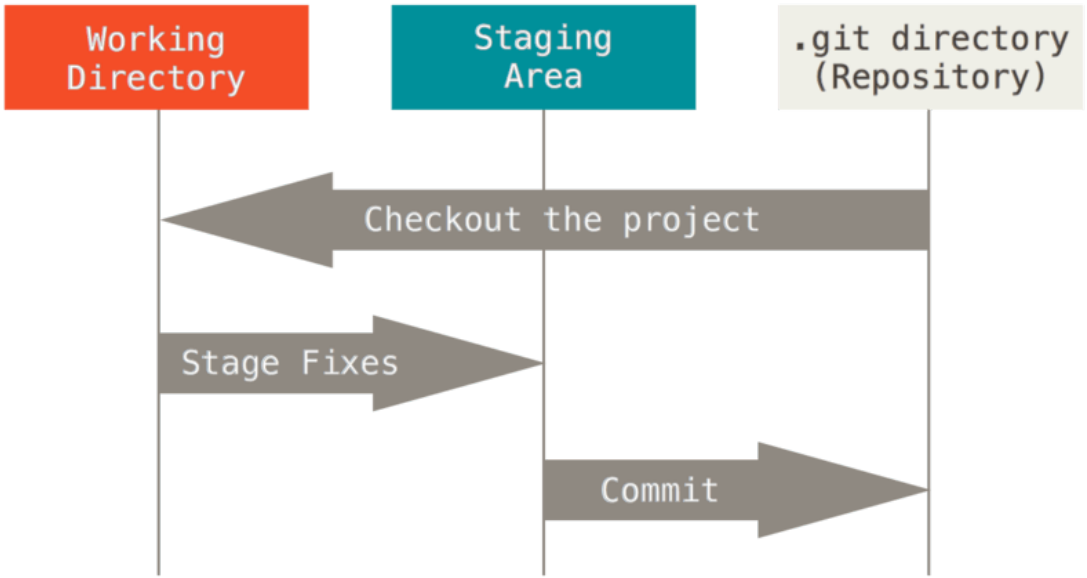
Estados de git

Commited (Confirmado)

Los datos se almacenan de manera segura en la base de datos local

git commit -m "Cambio realizado"
posteriormente usamos el *git push* para agregarlo a la base de datos remota.

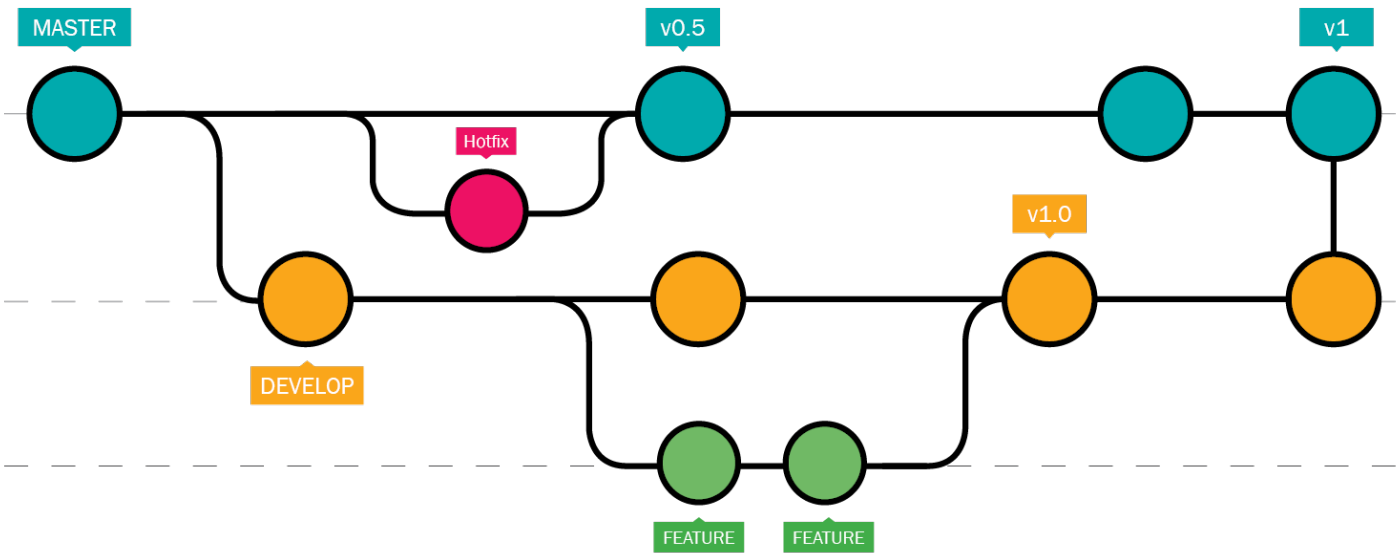
Estados de git



El directorio de trabajo es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que los puedas usar o modificar.

Ramificaciones

Las ramificaciones son importantes en git ya que permiten al usuario crear ”nodos” a parte del principal(master) para trabajar sobre ellos sin ir a dañar nada del ”nodo” master.



Creando ramificaciones

git branch nombreRama
git checkout nombreRama
git checkout -b nombreRama (cambiar y crear)
git push –set-upstream origin nombreRama (ya creada)
git checkout master
git merge nombreRama

Ejemplos de ramificaciones

```python

# Rama1
def function1(params):
    return params*params

def function2(params):
    return params+params

def hola():
    return 2

# Prueba de branches en git
```

Figure: Código rama2

```python

# Rama2
def function1(params):
    var = 0
    for i in range(0,10):
        var = var + params
    return params*var

def function2(params):
    return 10+params

def adios():
    return 2

# Prueba de branches en git
```

Figure: Código rama3

```
<<<<<<< HEAD
# Rama1
=======
# Rama2
>>>>>>> rama3
def function1(params):
    var = 0
    for i in range(0,10):
        var = var + params
    return params*var

def function2(params):
    return 10+params

def adios():
    return 2

# Prueba de branches en git
```

```
Python   Tab Width: 3   Ln 2, Col 13   INS
Switched to branch 'rama2'
alejo@Alejo:~/MergeEx$ git merge rama3
Auto-merging branches1.py
CONFLICT (content): Merge conflict in branches1.py
Automatic merge failed; fix conflicts and then commit the result.
```

Figure: Código merge

Ejemplos de ramificaciones

```python
# Rama1

def function1(params):
    return params*params

#Prueba ramas git
```

Figure: Código rama2

```python
# Rama2

def function2(params):
    return params + params

#Prueba ramas gitgit ad
```

Figure: Código rama3

```
<<<<<<< HEAD
# Rama1

def function1(params):
    return params*params
=======
# Rama2

def function2(params):
    return params + params
>>>>>>> ramas2

#Prueba ramas git
```

```
hipatia@CNA94621:~/Desktop/Prueba/Prueba$ git merge ramas2
Auto-merging rama1.py
CONFLICT (add/add): Merge conflict in rama1.py
Automatic merge failed; fix conflicts and then commit the result.
hipatia@CNA94621:~/Desktop/Prueba/Prueba$
```

Figure: Código merge

Ejemplos de ramificaciones

```
1
2 def function1(params):
3     return params*params
4
5 def function2(params):
6     return params + params
7
8
```

Figure: Código rama2

```
1
2 def function1(params):
3     return params*params
4
5 def function2(params):
6     return params + params
7
8 def another(params):
9     return 20*params
10
11 def anotherx2(otherparams):
12     return otherparams
```

Figure: Código rama3

```
1
2 def function1(params):
3     return params*params
4
5 def function2(params):
6     return params + params
7
8 <<<<<<< HEAD
9
10 =======
11 def another(params):
12     return 20*params
13
14 def anotherx2(otherparams):
15     return otherparams
16 >>>>>>> ramas2
```

Python

```
hipatia@CNA94621:~/Desktop/Prueba/Prueba$ git merge ramas2
Auto-merging rama1.py
CONFLICT (content): Merge conflict in rama1.py
Automatic merge failed; fix conflicts and then commit the result.
hipatia@CNA94621:~/Desktop/Prueba/Prueba$
```

Figure: Código merge

Ejemplos de ramificaciones

```python
def function1(params):
    return params*params

def function2(params):
    return params + params

def another(params):
    return 20*params

def anotherx2(otherparams):
    return otherparams
```

Figure: Código rama2

```python
def function1(params):
    #Cambios acá
    return 50*params

def function2(params):
    #Y acá
    print(params)
    return params + params

def another(params):
    return 20*params

def anotherx2(otherparams):
    return otherparams
```

Figure: Código rama3

```python
def function1(params):
    #Cambios acá
    return 50*params

def function2(params):
    #Y acá
    print(params)
    return params + params

def another(params):
    return 20*params

def anotherx2(otherparams):
    return otherparams
```

```
hipatia@CNA94621:~/Desktop/Prueba/Prueba$ git merge ramas2
Auto-merging rama1.py
Merge made by the 'recursive' strategy.
 rama1.py | 5 ++++-
 1 file changed, 4 insertions(+), 1 deletion(-)
hipatia@CNA94621:~/Desktop/Prueba/Prueba$
```

Figure: Código merge

Ejemplos de ramificaciones

```python
def function1(params):
    return 50*params

def function2(params):
    print(params)
    return params + params

def another(params):
    #acá cambia 200
    return 200*params

def anotherx2(otherparams):
    return otherparams
```

Figure: Código rama2

```python
def function1(params):
    return 50*params

def function2(params):
    print(params)
    return params + params

def another(params):
<<<<<<< HEAD
    #acá cambia 200
    return 200*params
=======
    #Acá no tiene el 200
    return 20*params
>>>>>>> ramas2

def anotherx2(otherparams):
    return otherparams
#Otra función

def func():
    print("hola")
```

Python

```
hipatia@CNA94621:~/Desktop/Prueba/Prueba$ git merge ramas2
Auto-merging rama1.py
CONFLICT (content): Merge conflict in rama1.py
Automatic merge failed; fix conflicts and then commit the result.
hipatia@CNA94621:~/Desktop/Prueba/Prueba$
```

Figure: Código merge