



ESCUELA DE
INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



Día, Fecha:	jueves, 27/07/2023
Hora de inicio:	07:10 - 08:50

0771 INTRODUCCIÓN A LA PROGRAMACION Y COMPUTACION 2 [N]

Sergio Mynor David Felipe Zapeta



Lab #2

IPC2 N

Sergio Mynor David Felipe Zapeta

Agenda

01 **Repaso**
Clase anterior

02 **Git**
Inicio de
repositorios

03 **Ejemplos**
De uso de git



GitHub
Teórico **04**

Python
Basico **05**

Python
medio **06**



GIT

Es una herramienta utilizada para
administrar el control de las versiones de
sus aplicaciones

- `git config --list`
- `git config --global user.name "mi_nombre"`
- `git config --global user.email "mi_correo"`

Comandos

Configuración

Primeros pasos

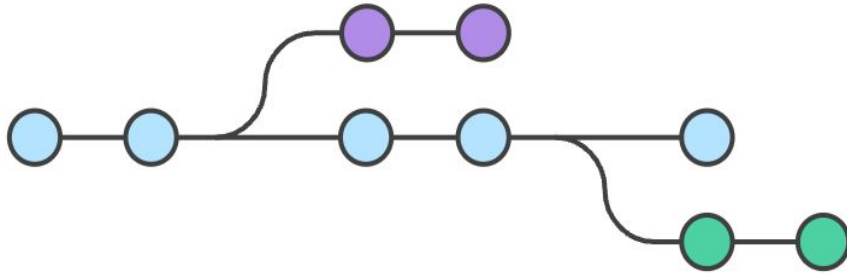


Commit

Es una instantánea de los cambios realizados en ese momento en particular.

- `git init`
- `git add .`
- `git commit -m "Mensaje de commit"`





Branch

Una rama es una bifurcación del estado del código que crea un nuevo camino para la evolución o cambios del mismo

Comandos

Branch

Ver las ramas disponibles

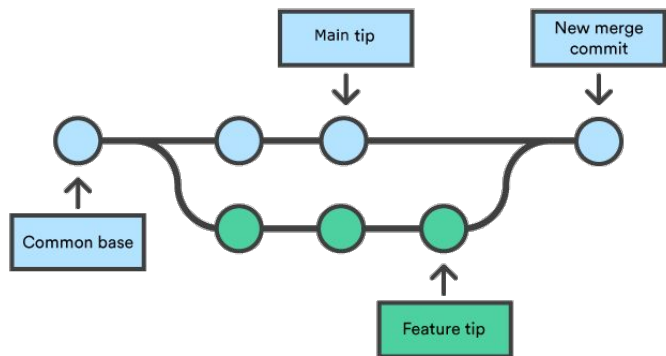
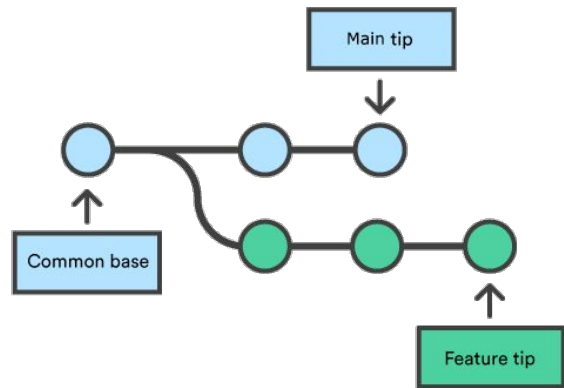
- `git branch`

Crea una branch

- `git branch [nueva_rama]`

Mueve el puntero a la rama

- `git checkout [rama]`



Merge

Es una operación que combina los cambios de dos ramas diferentes en una sola, siendo una, la rama donde nos encontramos cuando ejecutamos el comando y la segunda, la rama que indiquemos después del comando.

Comandos

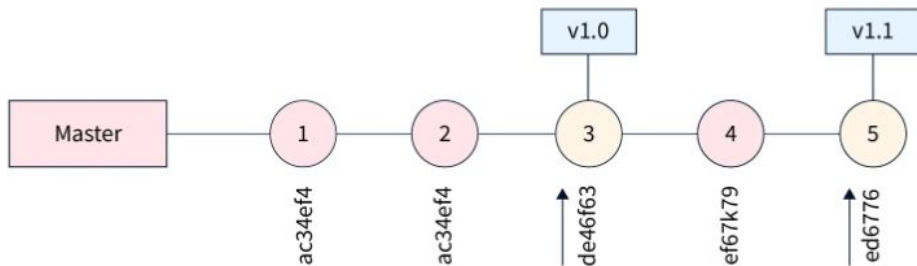
Merge

Nos movemos a la rama que aceptara los nuevos cambios

- `git checkout [rama1]`

Rama de la cual obtenemos los cambios

- `git merge [rama2]`



Tag

Es una referencia específica a un punto concreto en la historia del repositorio. Se utiliza principalmente para marcar versiones importantes, cómo lanzamientos

Comandos

Tag

Listar los tags

- `git tag`

Crea tag y lo asigna a un commit

- `git tag -a nombre id_commit`

Pública el tag en nuestro repositorio

- `git push origin -tags`



Python

lenguaje de programación de alto nivel,
interpretado. Es de sintaxis clara y legible, fácil de
aprender y usar.

```
#comentario una linea  
""""  
comentario multilinea  
""""
```

```
print("con doble")  
print('simple')
```

```
#nombre igual valor  
numero = 10
```

```
varNumero = int("9")  
varCadena = str(10)
```

Comparadores

>, >=, <, <=, ==, !=

Sintaxis Basica

Comentarios

Imprimir

Asignacion

conversion



Sintaxis Medio

If

if else

if elif else

```
if(True):  
    print("is True")
```

```
if(False):  
    print("is False")  
else:  
    print("val else")
```

```
if(False):  
    print("No")  
elif (False):  
    print("No")  
else:  
    print("Si")
```

```
fruits = ["val1", "val2", "val3"]  
for x in fruits:  
    print(x)
```

```
while(True):  
    #sentencias
```

```
n = 4  
for i in range(0,4):  
    print(i)
```

Sintaxis Medio (2)

For in

For range

while



Sintaxis Medio (3)

funciones

pass

return

```
def imprimirBienvenida():  
    nombre = "Sergio"  
    print(f'Bienvenido {nombre}')
```



```
imprimirBienvenida()
```

```
def noImplementada():  
    pass
```

```
def calcularSuma(num1, num2):  
    return num1 + num2
```

Sintaxis Medio (4)

Strings

```
conDoble = "Texto Doble"  
conSimple = 'Texto "Simple"'  
conFormato = "hoy es {} {} de febrero".format("miercoles", 1)  
conFormatoInverso = "hoy es {1} {0} de febrero".format("miercoles", 1)  
conFormatoNombre = "Ciudad: {city}, Año: {year}".format(year=2023, city="Guate")
```

```
class ClaseDemo:  
    def __init__(self):  
        self.cantidad = 0  
  
    def getValorDemo(self):  
        return self.cantidad
```

Sintaxis Medio (5)

Clases



Sintaxis Medio (6)

Importación de clases

```
from nombre_archivo import Nombre_Clase
```

```
from clasedemo import ClaseDemo
```

```
val = ClaseDemo()  
print(val.getValorDemo())
```

Classes

uso

