Programación Avanzada
IIC2233 2025-2

Cristian Ruz - Pablo Araneda - Francisca Ibarra - Tamara Vidal - Daniela Concha

Anuncios

14 de agosto de 2025

1. ¡Respondan el formulario y acepten la invitación!
2. Ya deberían tener todo instalado en su computador.
3. Hoy haremos la Actividad 1, qué nos servirá para acostumbrarnos al flujo de actividades del semestre.

Guía de estilo: PEP8

Guía de estilo

- ¿Qué es una guía de estilo?
- ¿Por qué usarla?
- Guía de estilo en Python: PEP 8.

Ejemplo de cambio de estilo ¿Qué cambia en el formato?

Antes de PEP8

```
def mifuncion(nombre,apellido):
    print(nombre+" "+apellido)
    print(time.today()       )

listaDeProfes = [["Cristian", "Ruz"],
["Pablo", "Araneda"],["Francisca",
"Ibarra"],["Tamara", "Vidal"]
["Daniela", "Concha"]]

for lista in listaDeProfes:
  mifuncion(lista[0],

              lista[1])

import time
```

Después de PEP8

```
import time

def mi_funcion(nombre, apellido):
    print(nombre + " " + apellido)
    print(time.today())

lista_de_profes = [
    ["Cristian", "Ruz"],
    ["Pablo", "Araneda"],
    ["Francisca", "Ibarra"],
    ["Tamara", "Vidal"],
    ["Daniela", "Concha"]
]

for lista in lista_de_profes:
    mi_funcion(lista[0], lista[1])
```

Modularización

Modularización

- ¿Qué es modularización?
- Cómo importar en Python
- ¿Por qué modularizar?

Ejemplo de modularización

modulo.py

```
def funcion():
    print(“Hola amiguitos de IIC2233”)

def funcion2():
    print(“Adios amiguitos de IIC2233”)
```

main.py

```
import modulo

import modulo as p

from  modulo import funcion, funcion2

from  modulo import * # Usted NO lo haga!!
```

Paths

Paths

- *Paths* absolutos y relativos
- *Paths* según el sistema operativo
- Librerías para manejo de *paths* :
  - OS
  - pathlib

Ejemplo de *paths*

Usted **NO** lo haga

```
p = “C:/Usuario/gatito/main.py” # absoluto

p = “/Usuario/gatito/main.py” # absoluto

p = “carpeta/tarea/main.py” # Puede fallar

p = “carpeta\tarea\main.py” # Puede fallar
```

Usted **SÍ** lo haga

```
import os
from pathlib import Path


p = os.path.join(“carpeta”, “tarea”, “main.py”)

p = Path("data/archivo.txt")
```

Uso de terminal

Comandos de terminal

```
comando argumentos* -opciones*
```

Windows:

- `cd` Ver y cambiar directorio actual
- `dir` Listar contenido del directorio
- `mkdir` Crear un directorio nuevo
- `echo` Mostrar mensaje en la terminal
- `>` Guardar contenido en un archivo

```
echo “contenido” > nombre.extension
```

Linux y macOS:

- `pwd` Ruta absoluta actual
- `cd` Ver y cambiar directorio actual
- `ls` Listar contenido del directorio
- `mkdir` Crear un directorio nuevo
- `touch` Crear un archivo nuevo

```
touch nombre.extension
```

Git

Flujo de git

Mi computador

Lista de cambios

Repositorio local

GitHub

`git add`

`git commit`

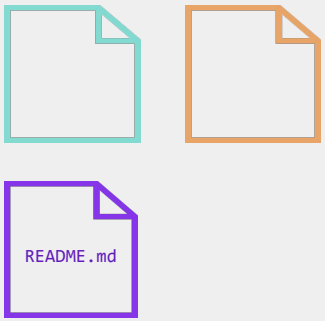
`git push`

Todo esto ocurre localmente

Internet

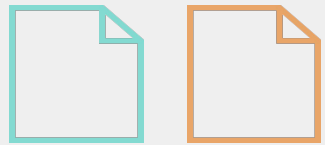
¿Qué acabamos de hacer?

Mi computador

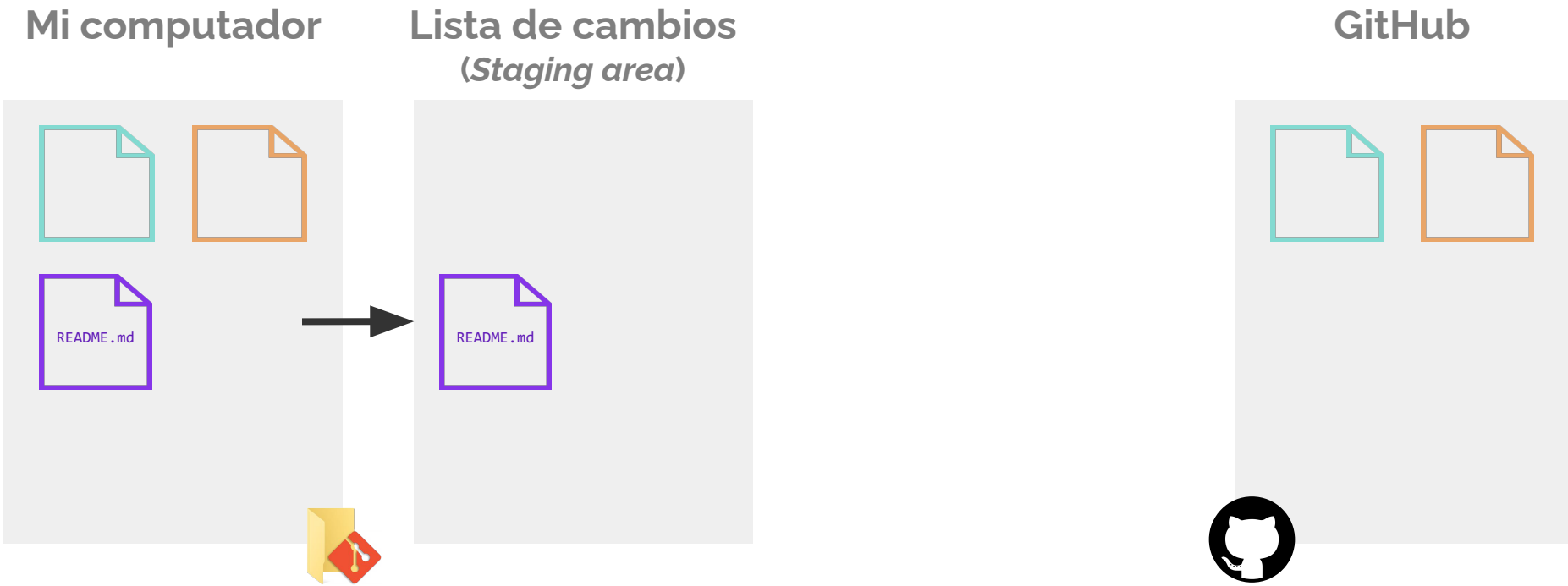



```
git add README.md
```

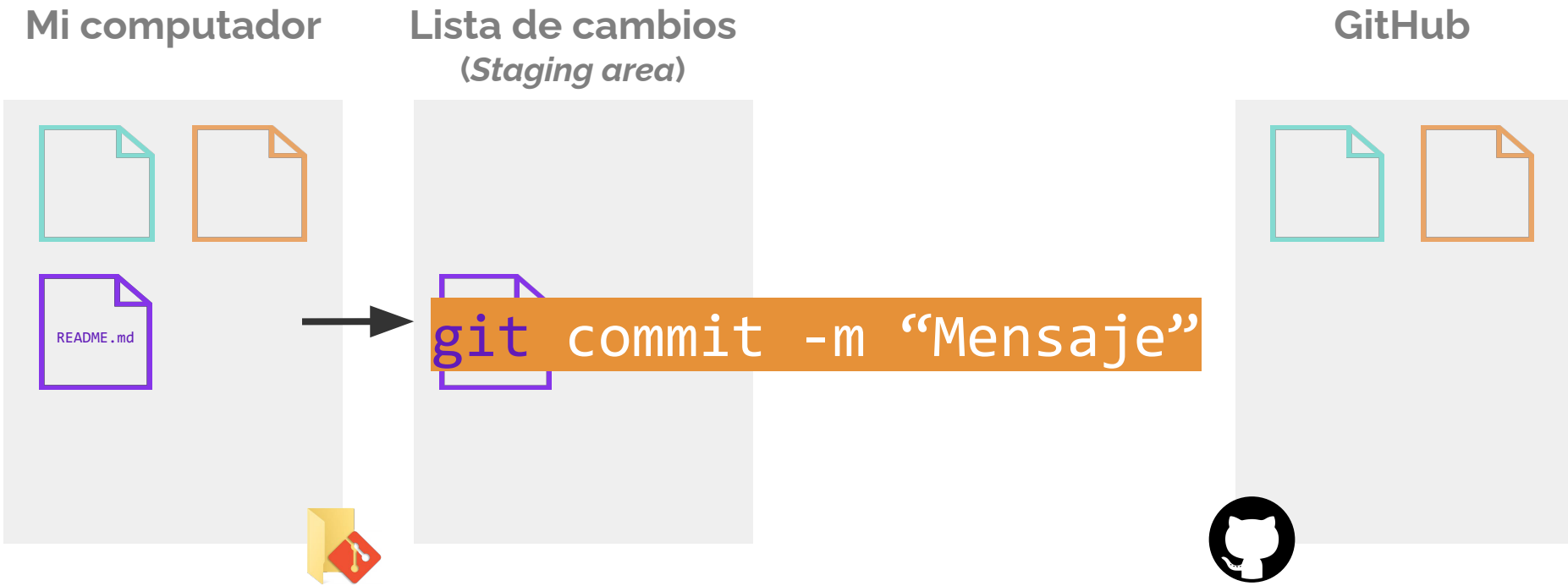
GitHub



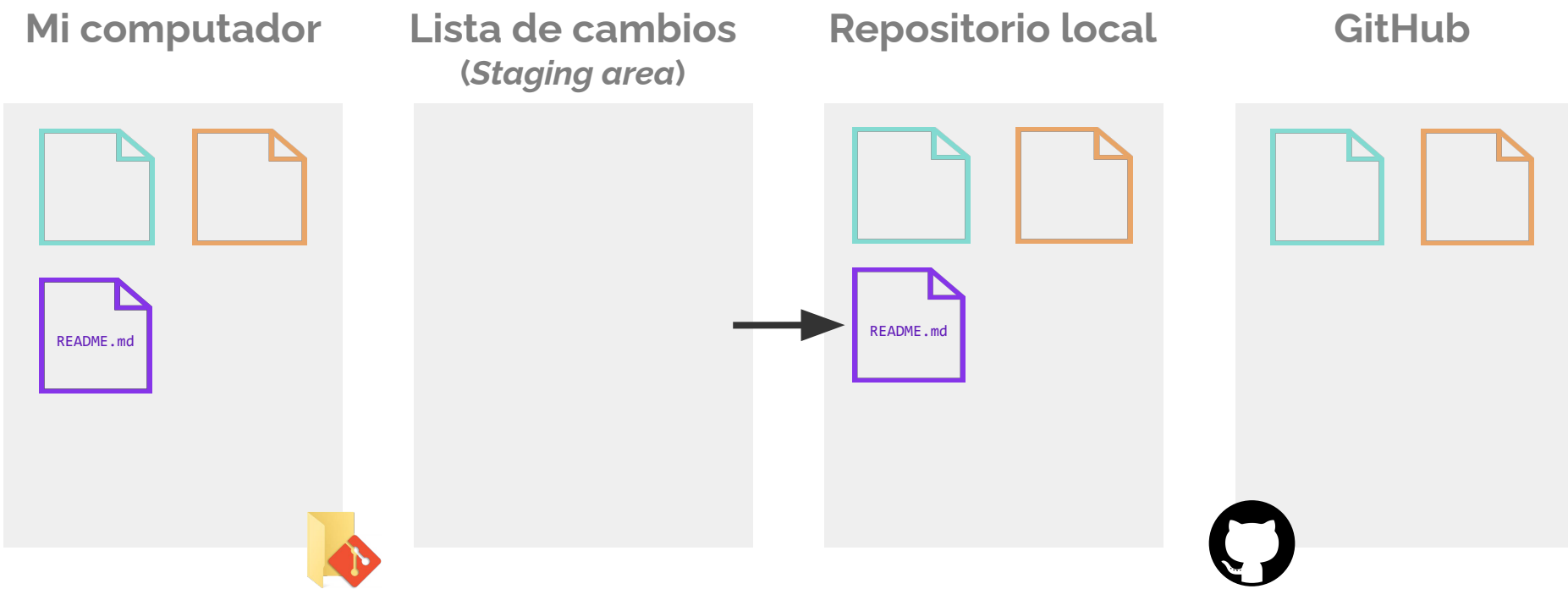
¿Qué acabamos de hacer?



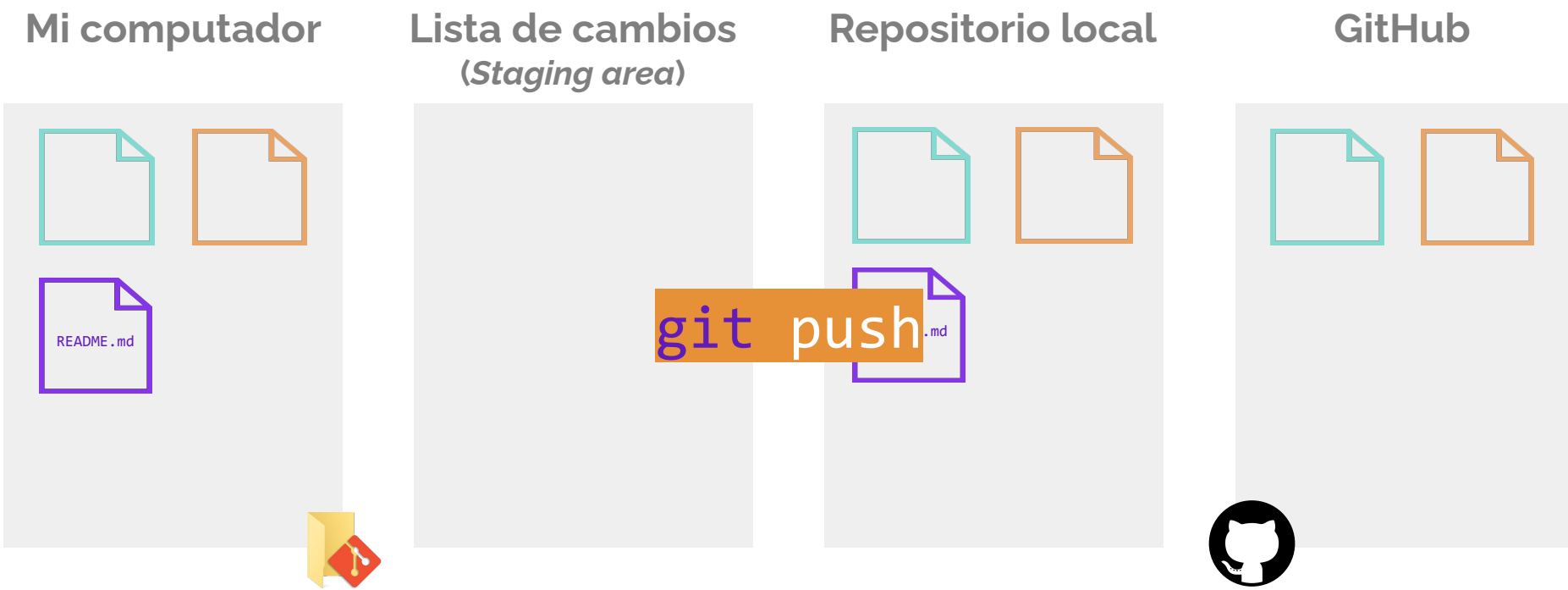
¿Qué acabamos de hacer?



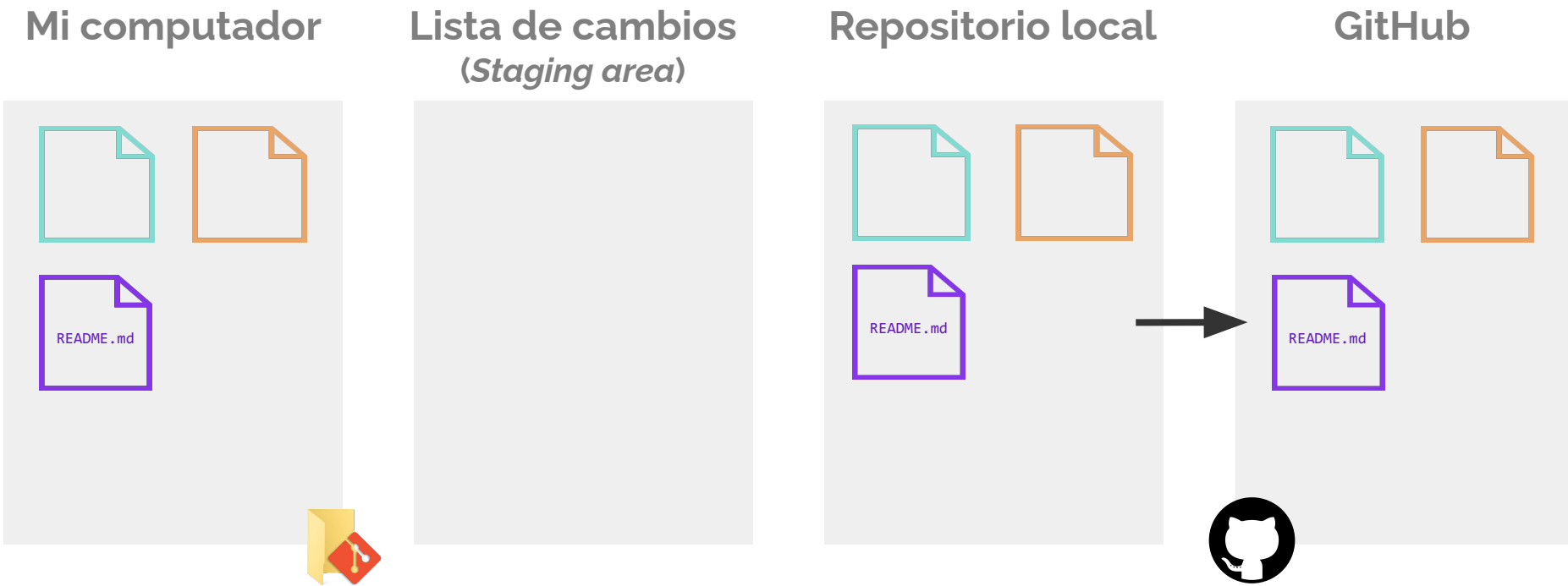
¿Qué acabamos de hacer?



¿Qué acabamos de hacer?



¿Qué acabamos de hacer?



¿Qué acabamos de hacer?

Mi computador

README.md

Lista de cambios
(*Staging area*)

Repositorio local

README.md

GitHub
(Repositorio remoto)

README.md

Siempre hagan *add*, *commit* y *push*

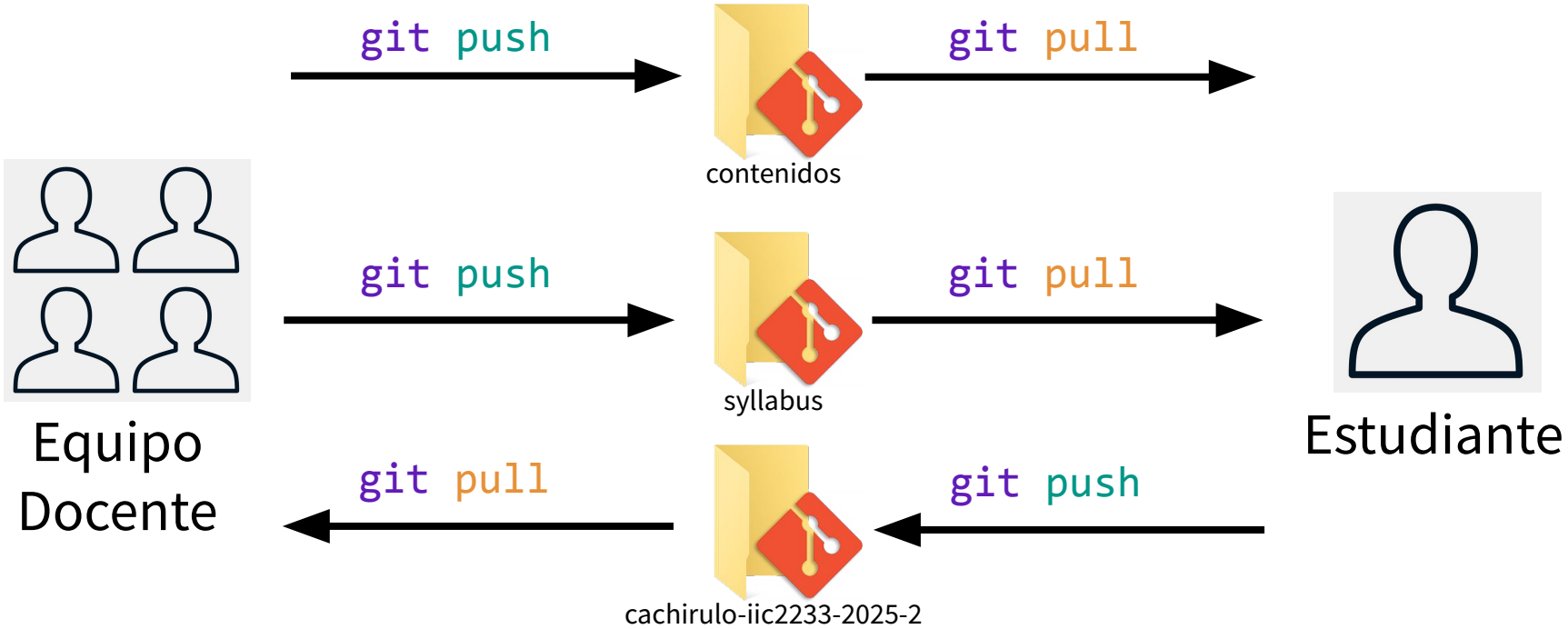
¿**Cuándo** hacerlo?

- Cada vez que avancen en algo importante de su actividad o tarea.
- Si llevan programando más de media hora.
- Cuando paren para hacer otra cosa.

¿**Por qué** hacerlo?

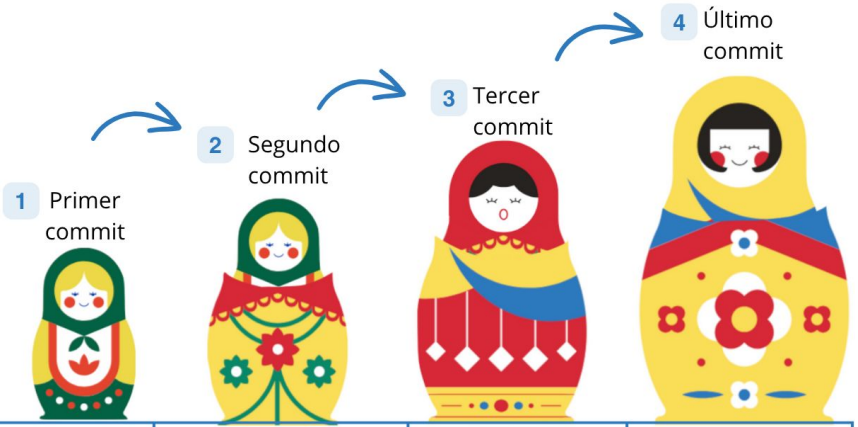
- Tener su trabajo en GitHub es una copia de seguridad.
- *Shit happens*:
  - Accidentes con líquidos.
  - Robos en Deportes.
  - Fallas de *hardware* o *software*.
  - Cortes de internet.
  - Echar a perder la tarea.
  - Y muchas otras cosas.

El flujo durante el semestre



Entendiendo un *commit*

- Cada *commit* **guarda una foto completa del proyecto**, incluyendo todos los cambios anteriores.
- Es como una muñeca rusa: el **último *commit* contiene todo lo anterior**.



Commit	Primer	Segundo	Tercer	Cuarto
Cambios realizados	Se agrega el archivo main.py	• Se agrega el archivo utils.py • Se importan las funciones de utils.py en el archivo main.py	Se corrige un error de una función en utils.py	Se agrega README.md

Veamos una pregunta de Evaluación Escrita

Tema: Entorno trabajo y git (Midterm 2024-1)

13. Con respecto al uso de Git, se presenta esta situación:

> *Al momento de subir cambios desde tu repositorio local a tu repositorio remoto, notas que hay modificaciones que no se encuentran presentes en tu computador, pero que sí están presentes en GitHub.*

Asumiendo que no hay conflictos, ¿cuál es el flujo de comandos que debes ejecutar para descargar los cambios remotos y después subir las modificaciones locales?:

```
 I. git commit -m "Mensaje descriptivo"    III. git pull
II. git add                                 IV. git push
```

A) III, II, I y IV
B) II, I y IV
C) II, I y III
D) IV, II, I y III
E) No es posible realizar lo pedido.

Veamos una pregunta de Evaluación Escrita

Tema: Entorno trabajo y git (Midterm 2024-1)

13. Con respecto al uso de Git, se presenta esta situación:

> *Al momento de subir cambios desde tu repositorio local a tu repositorio remoto, notas que hay modificaciones que no se encuentran presentes en tu computador, pero que sí están presentes en GitHub.*

Asumiendo que no hay conflictos, ¿cuál es el flujo de comandos que debes ejecutar para descargar los cambios remotos y después subir las modificaciones locales?:

```
 I. git commit -m "Mensaje descriptivo"    III. git pull
II. git add                                 IV. git push
```

A) III, II, I y IV
B) II, I y IV
C) II, I y III
D) IV, II, I y III
E) No es posible realizar lo pedido.

Programación Avanzada
IIC2233 2025-2

Cristian Ruz - Pablo Araneda - Francisca Ibarra - Tamara Vidal - Daniela Concha

Comentarios AC1

- Recuerden copiar la AC1 desde el Syllabus hasta su repo personal.