ArgentinaPrograma
YoPrograma

git y GitHub

por Leonardo Blautzik, Federico Gasior y Lucas Videla

Julio / Diciembre 2021

Ministerio de
Desarrollo Productivo
Argentina

¿Cómo trabajamos en proyectos que exceden a una persona?

- ¿Cómo **sincronizamos** el trabajo?
- ¿Cómo **dividimos** las partes a realizar?
- ¿Cómo volvemos a **juntarlas**?
- ¿Qué sucede si la **persona** que iba a juntar todo, no llega a hacerlo?
- ¿Qué hacemos si hay que volver a una **versión anterior**?
- ¿Cómo hacemos **experimentos** sobre el trabajo?

El status quo

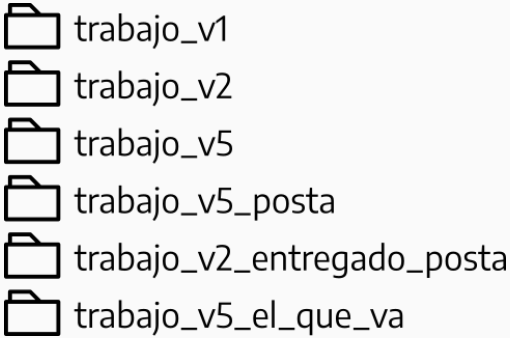

Figure 1: El horror

¿Qué es git?

git es un sistema de control de versiones.

¿Qué es un SCV?

Un **control de versiones** es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.
Fuente: git-book

Una analogía simple

Imaginemos que git es una **cámara de fotos**.

En este caso, la analogía es muy buena, ya que git es la herramienta, y no “la cosa” que haremos.

Descargo

Esta clase tiene dos partes. Durante la primera, aprenderemos todo utilizando la **línea de comandos**. Luego, repetiremos el proceso (más rápidamente) utilizando las **herramientas incluidas con Eclipse**.

A lo largo de su vida profesional, será necesario que utilicen la línea de comandos. Recomendamos enfáticamente amigarse, de a poco, con ella.

Creando el álbum de fotos

El primer paso, es la creación de un **repositorio git**. Nótese que no es **un git**, sino **un repo**.

```
$ git init
```

Viendo el panorama

Antes de comenzar a sacar fotos, queremos ver **cómo está el paisaje**.

```
$ git status
```

Posando para la foto

Cuando estemos por tomar una foto, debemos avisarle a aquellos que saldrán en la misma, **que se preparen**.

```
$ git add <archivo>
```

¡Clic!

Para sacar una foto, debemos **escribir un mensaje** (como se hacía en las viejas Polaroid)

```
$ git commit -m "<mensaje>"
```

Probando distintas tomas

Podemos crear “pequeños álbumes **experimentales**” para alojar allí otras tomas que nos pueden parecer interesantes, y luego quizás descartemos o incorporemos al álbum principal.

```
$ git switch -c <nombre_de_rama>
```

Moviéndonos entre sub-álbumes

Podemos **trasladarnos entre sub-álbumes** a gusto.

```
$ git switch <nombre_de_rama>
```

Incorporando un sub-álbum al principal

Si quisiéramos **incorporar un sub-álbum** al principal, solo debemos mezclarlo con el mismo:

```
$ git merge <nombre_de_rama>
```

Posibles conflictos

Si hubiera cambios similares en dos álbumes, git nos preguntará **qué cambio deseamos conservar**.

La resolución de conflictos “de merge” es una tarea que puede asustar, pero es simple si se trabaja en forma ordenada.

Sincronizando nuestro álbum

Para **subir nuestro contenido** a un repo remoto, debemos...

```
$ git push
```

Y si en cambio, queremos **recibir la última versión disponible**, debemos...

```
$ git pull
```

Simplificando...

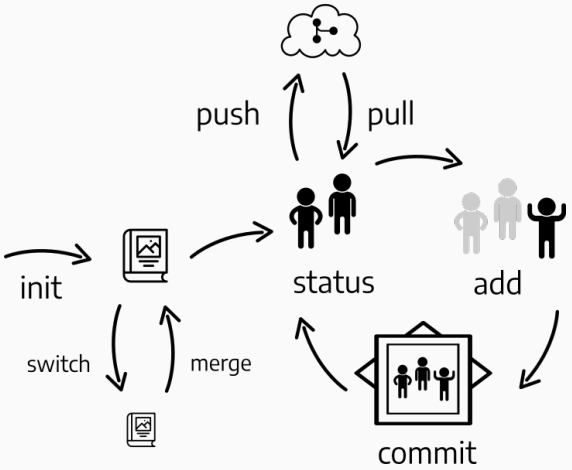

Figure 2: git, de un vistazo

¡Desde el principio!

...pero con **Eclipse**.

Para profundizar

- **Git**. Retrieved July 27, 2021, https://git-scm.com/
- **Git-Book**. Chacon, Scott. Retrieved July 27, 2021, https://git-scm.com/book/es/v2
- **git básico: Desde cero a conflictos de merge**. Sip of Code. Videla, Lucas. Retrieved July 27, 2021, https://www.youtube.com/watch?v=yEs0E3PnGuI

¡Muchas Gracias!

continuará...

Ministerio de
Desarrollo Productivo
Argentina