



Práctica Git

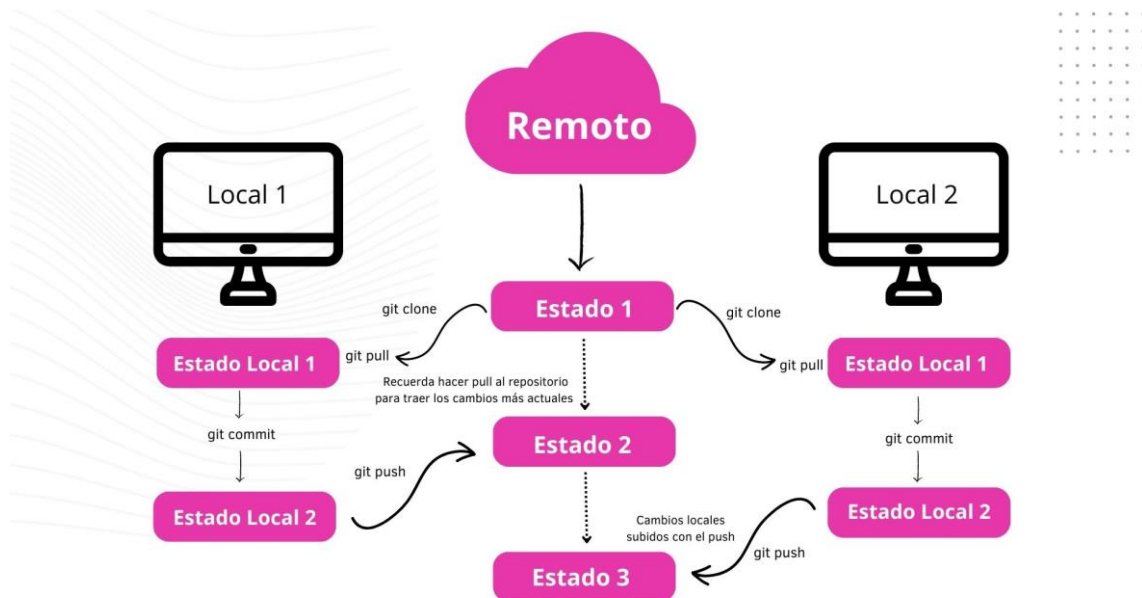


Objetivos

Familiarizarnos con las herramientas de control de versionado de código, Git y gestionar un proyecto. (descarga en <https://git-scm.com/>)

Git es un software de control de versiones, pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Tiene como propósito:

- Llevar registro de los cambios en archivos
- Coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.



Existen varias empresas que brindan el servicio de Git en la nube, las más conocidas son:

- GitHub: github.com
- GitLab: gitlab.com
- Bitbucket: bitbucket.org



Herramientas / Preparación

- Cuenta en Github



Carrera: INGENIERÍA EN SISTEMAS
Materia: PROGRAMACIÓN ORIENTADA A LA WEB
TRABAJO PRÁCTICO N° 1



- <https://github.com/>



Organización

Individual



Fecha de entrega / presentación

no hay fecha de entrega, el trabajo no debe entregarse



Ejercicios

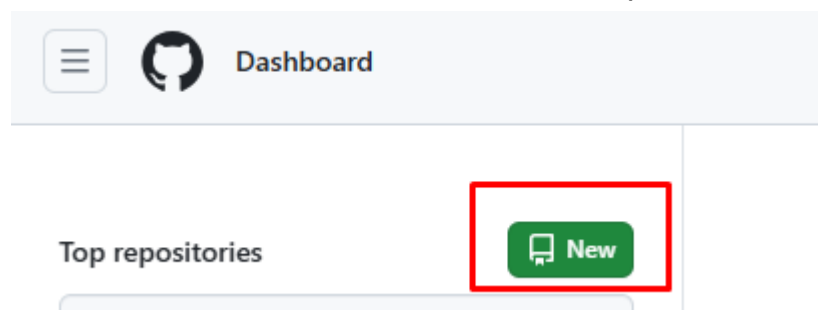
Algunas guías interesantes del uso de git

<https://www.hostinger.com.ar/tutoriales/comandos-de-git>

<https://aulasoftwarelibre.github.io/taller-de-git/usobasico/#diferencias-entre-workdir-y-staging>

Siga los siguientes pasos

1. Registrar una cuenta en GitHub
 - a. ingresar a github.com y crear una cuenta, si ya tiene una la puede utilizar
2. Crear un nuevo repositorio
 - a. Para crear un nuevo repositorio ir a



- b. debe asignar un nombre, (si es público, abierto a todos o privado, solo lo verán las personas que invite a su proyecto)
- c. agregar un readme file (archivo necesario para explicar de qué se trata tu proyecto)




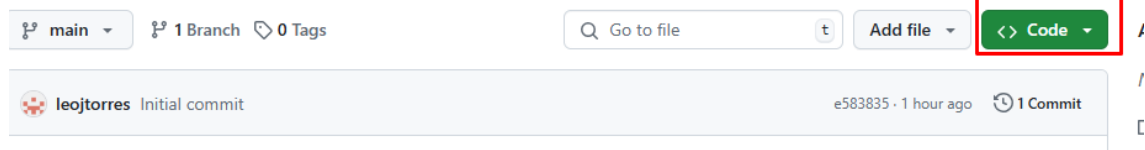
Carrera: INGENIERÍA EN SISTEMAS
Materia: PROGRAMACIÓN ORIENTADA A LA WEB
TRABAJO PRÁCTICO N° 1



3. Invitar a los profesores a tu proyecto,



- para ello deben ir a  y luego a Collaborators e ingresar los mails o cuenta indicada por los profesores
 - Paso opcional: Una vez que los profesores acepten la invitación ustedes pueden asignar roles a los mismos: Administrador (puede gestionar el proyecto), Desarrollador (puede modificar el proyecto), Auditor (solo lectura).
4. Preparando nuestro entorno de trabajo
- instalamos git, si usamos windows git-scm.com, si usamos linux <https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Instalaci%C3%B3n-de-Git>
 - una vez instalado ya está listo para usar por línea de comando, aunque deben saber que existen herramientas que permiten un uso más amigable de git, algunos ejemplos son: tortoisegit.org, www.sourcetreeapp.com o bien como extensión de visual code.
Todas estas herramientas se pueden combinar.
5. Bajemos nuestro primer proyecto
- seleccionamos una carpeta donde será nuestro lugar de trabajo
 - clonamos el proyecto con el comando git clone <<url del proyecto>> (la pueden conseguir en el repo, tocando este botón



), la primera vez que hacemos git clone, es normal que pida accesos, es decir mail y pass.

- si todo salió bien deberían ver algo como esto

Nombre	Fecha de modificación	Tipo	Tamaño
.git	7/8/2024 11:32 a. m.	Carpeta de archivos	
README.md	7/8/2024 11:32 a. m.	Archivo MD	1 KB

6. Ahora vamos a ingresar un archivo de prueba

- creamos un archivo y lo llamaremos index.html
- dentro con un block de notas o un IDE de programación le ingresamos el siguiente contenido:
<html>



```
<head>
</head>
<body>
    <h1>hola mundo</h1>
</body>
</html>
```

7. ahora es momento de agregar mi archivo a nuestro repo
 - a. Para ello utilizamos el siguiente comando `git add .` (este comando agrega los archivos que se le indican al repositorio), al poner "el punto" agregamos todos los archivos nuevos, si queremos especificar alguno en especial ponemos el nombre.
 - b. luego verificamos si se agrego con el comando `git status`
8. el próximo paso es subir nuestros cambios al repo
 - a. utilizaremos el comando `git commit -m "mensaje de cambios incluidos"`, el mismo se encarga de confirmar los cambios a subir, pero ojo todavía no se subirán al repo en la nube, también es necesario ingresar un mensaje indicando los cambios.
 - b. para ello hay que pucharlo, con el comando `git push`, ahora si subimos nuestros cambios a nuestro repo
 - c. podemos ingresar a nuestro repo y verificar si nuestros cambios ya se encuentran en el
9. Por último y solo para chequear ya que en este momento estamos trabajando en el repositorios solos, utilizaremos el comando `git pull`, el mismo se encarga de traer cambios desde el repo, si hubiera más personas trabajando sobre el proyecto, este comando se encarga de sincronizar y traer los cambios del resto del equipo.

Para finalizar

Intente los siguientes ejercicios:

1. Modifique el archivo README.md y luego impacte los cambios con los comandos commit y push
2. Comparta su código con un compañero e intenten realizar cambios y sincronizar los proyectos, para esto es necesario conocer otro comando más llamado pull