

# Introducción al Desarrollo de Sistemas

[Área personal](#) / [Mis cursos](#) / [IDS](#) / [Tema 1 - Introducción y marco de trabajo colaborativo](#)  
/ [Práctica : Introducción al manejo de repositorios Git](#)

## Práctica : Introducción al manejo de repositorios Git

### Introducción

En la actualidad, de manera prácticamente unánime, en todas las empresas del sector informático que desarrollan software utilizan una herramienta de gestión de versionamiento de código fuente. Esto es esencial puesto que el trabajo en software es inherentemente grupal/por equipos, donde múltiples personas trabajan con el código al mismo tiempo. Por lo tanto es vital que los aportes de cada miembro del equipo no dañen la funcionalidad de los aportes de otro, y que a su vez, el resultado final de los aportes constituya una versión del software que sea estable y entregable para el/los clientes.

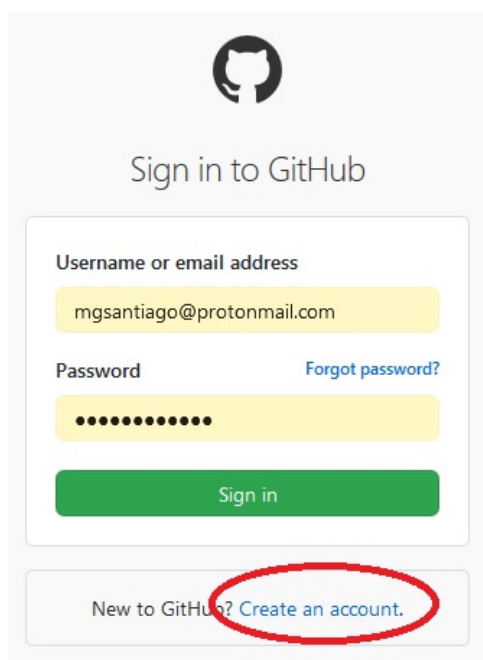
### Actividad

En esta actividad, lo que vamos a realizar es una simulación de metodología de trabajo similar a la de una empresa u organización y de aquí en adelante, todas las actividades a realizar serán gestionadas a través de repositorios.

### 1. Registración al servicio

1. Lo primero que vamos a realizar es la suscripción a un servicio de repositorios, en nuestro curso utilizaremos al menos uno de los servicios gratuitos más populares que existe: GitHub.

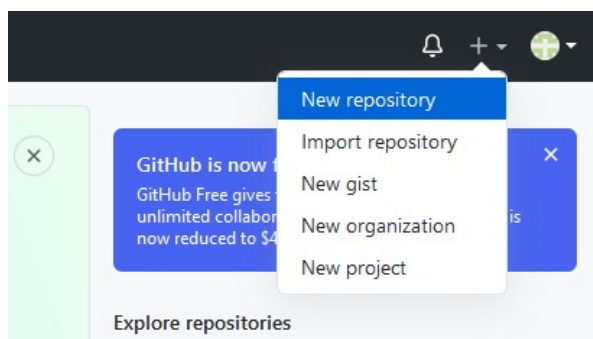
<https://github.com/>



( Si por alguna razón ya disponés de una cuenta, podés reutilizarla para realizar este curso ).

## 2. Creación del repositorio

Una vez constituida nuestra cuenta, lo que vamos a efectuar es la creación de nuestro primer repositorio. Vamos a realizarlo de manera gráfica a través del sitio oficial.



El mismo deberá contener las siguientes características:

**Nombre:** *curso-sistemas*

**Descripción:** *Repositorio del curso de Introducción al Desarrollo de Software*

**Tipo:** *Público*

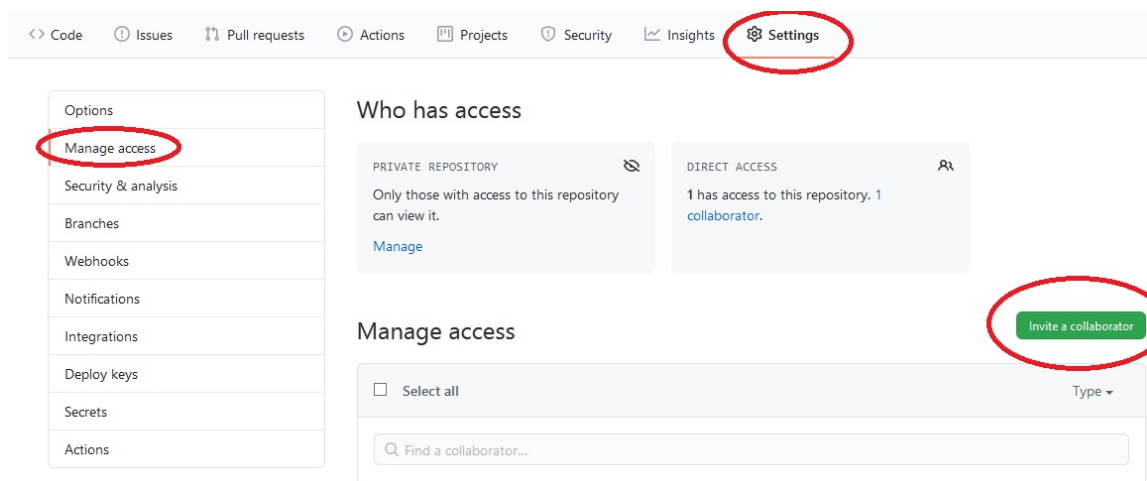
**Inicialización:** *Inicializar con archivo Readme*

**Licencia:** *GNU / General Public License v 3.0*

## 3. Configuración del repositorio

Cada servicio de repositorios establece una configuración inicial de fábrica para cada repositorio. Si bien la cantidad de parámetros posibles de modificación son enormes, nosotros únicamente nos vamos a concentrar en los relevantes y más importantes.

Una parte que es esencial en los repositorios es la **configuración de accesos**, es decir, determinar qué personas pueden participar en tu código. Autorizar las personas que pueden realizar modificaciones ( Generalmente esas personas son tus compañeros de trabajo y supervisores ). Aquí, en nuestro caso, yo seré un colaborador de tu repositorio, para que pueda efectuar comentarios sobre tu código y realizar devoluciones en tus trabajos y actividades.



Los colaboradores se pueden agregar a través del usuario de GitHub de otra persona.

En nuestro caso, van a utilizar mi cuenta: [mgsantiago-10g](#)

## 4. Instalación del cliente Git

Recuerden que el servicio de repositorios es un servicio que trabaja en la nube, el código se encuentra allí, pero necesitamos tener una conexión con nuestra propia computadora para poder interactuar con el servicio. Algo así como las aplicaciones de Dropbox, o Mega.

Si trabajan con sistemas operativos Windows, deberán instalar (con configuración de fábrica):

<https://git-scm.com/download/win>

En caso de trabajar con sistemas operativos Linux (Debian/Ubuntu):

```
# apt-get install git
```

## 5. Operaciones sobre el repositorio

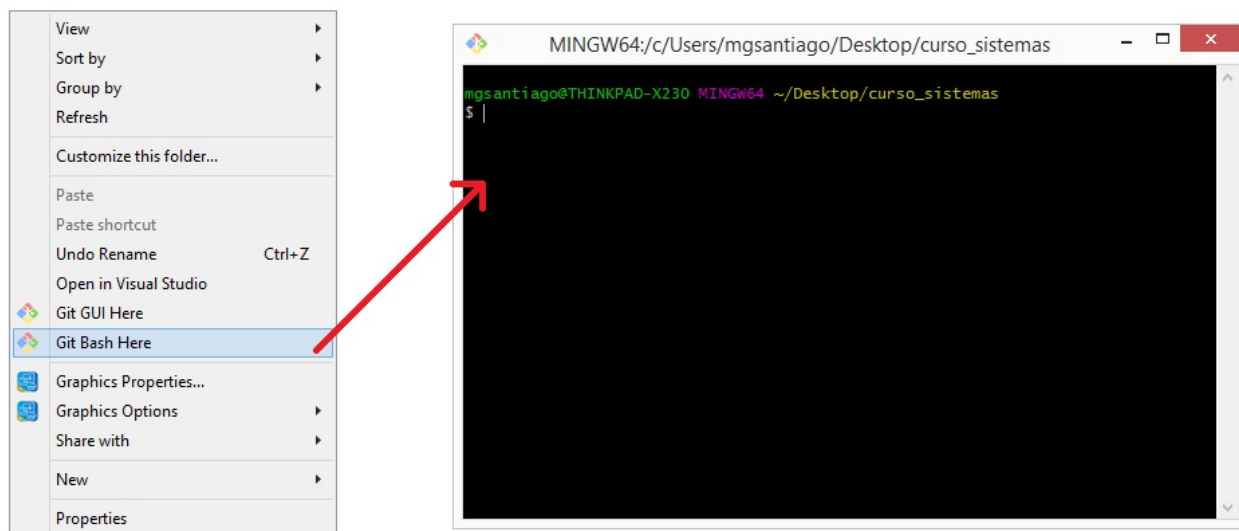
Los repositorios Git, tienen una cantidad enorme de operaciones posibles, y combinaciones. He ahí, porqué la herramienta es utilizada enormemente por empresas y organización, su inherente flexibilidad hace posible la adaptación a diversas estrategias de organización de código y su versionamiento. Nosotros, en nuestro curso, veremos las operaciones esenciales e imprescindibles para poder operar de una forma elemental.

Las operaciones elementales son: **Clone, Pull, Commit, Add, Push, Status, Checkout**

### 5.1 Clone

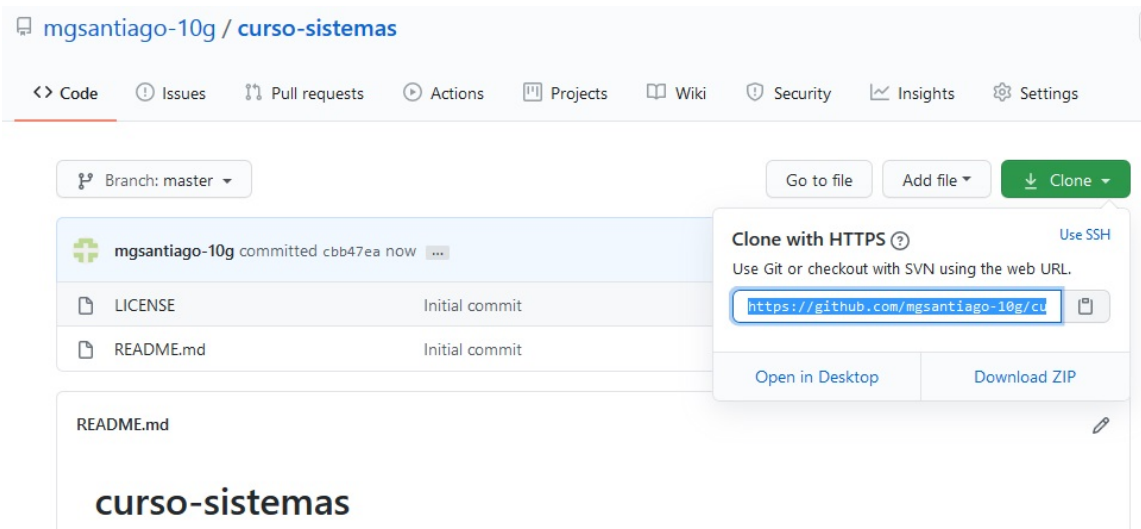
Lo primero que vamos a hacer, es sincronizar el repositorio en la nube en una carpeta/directorio local de nuestra PC por primera vez. Para ello, lo que vamos a hacer es una operación de "Clonado" <-> **clone**. Creamos una carpeta en algún lugar de acceso conveniente en nuestra PC, *entramos allí* y abrimos el cliente Git.

Si estás en Windows y ya instalaste correctamente el cliente Git, podrás ejecutarlo de esta manera:



En Linux, la operación es similar, solo que deberás abrir una terminal en el directorio en cuestión.

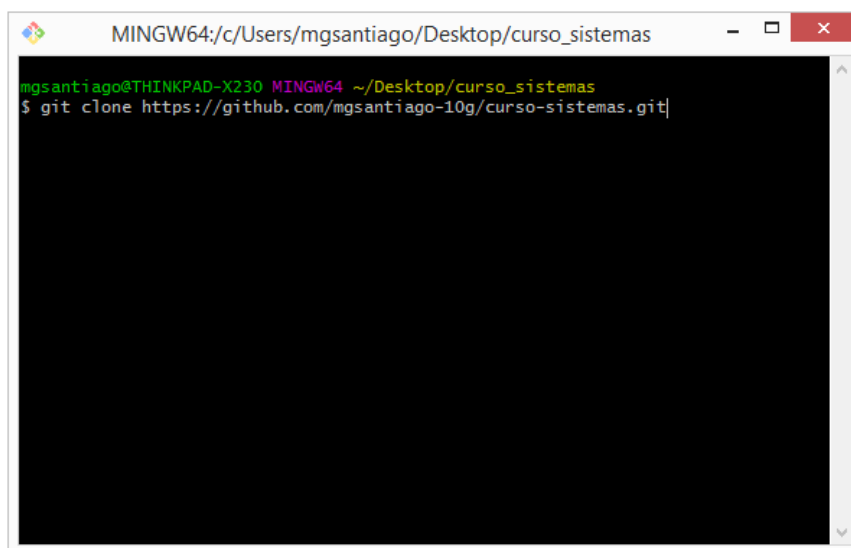
En cualquier caso, una vez que tengas la línea de comandos abierta, deberás determinar la dirección remota (web) del repositorio:



Y posteriormente, con esa dirección, escribir el siguiente comando:

***git clone dirección***

Quedando algo así:



Introducimos el comando, lo ejecutamos. Probablemente se les va a pedir que ingresen una información adicional, como un nombre de usuario y contraseña, y también los datos de la cuenta de GitHub. De ser necesario, ejecuten el comando nuevamente, hasta que termine correctamente y vean en su carpeta el contenido mínimo del repositorio que es un archivo README.

## 5.2 Status, Commit, Add y Push

Para comenzar a efectuar las operaciones, lo primero que vamos a hacer, es generar la estructura de carpetas/directorios para cada encuentro del curso. El curso aproximadamente tiene 11 encuentros, cada uno con diversas actividades, por lo tanto, vamos a crear 11 carpetas, con nombres del estilo:

***clase-01, clase-02, clase-03, etc...***

Terminado esto, abriremos el cliente Git como hicimos en el punto anterior y escribiremos:

***git status***

Observaremos que el resultado en pantalla de este comando, nos devuelve una lista de las modificaciones LOCALES (En nuestra PC) que tenemos realizadas hasta el momento y que aún no están ACTUALIZADAS en la nube.

Para agregar los cambios que muestra el comando *git status*, emplearemos el comando:

#### ***git add .***

(Para agregar todos los cambios realizados sobre todo el repositorio. Aquí se sincroniza el 100% del repositorio local con el remoto)

#### ***git add ruta-del-archivo-especifico***

(Para agregar sólo los cambios puntuales sobre archivos específicos. Aquí sólo se sincroniza lo que realmente queremos puntualmente)

Posteriormente, para registrar los cambios a sincronizar utilizamos el comando:

#### ***git commit -m "mensaje-personalizado"***

(De esta manera, dejamos registrado que todos los cambios de agregamos, serán publicados y con una descripción de texto que indique la razón de los mismos. Por ejemplo, en nuestro caso sería algo así como: "git commit -m "Preparación de las carpetas del curso"")

Finalmente, para subir los cambios a la nube utilizamos el comando:

#### ***git push***

Si ingresan a la página web de GitHub, y específicamente en su repositorio, verán reflejados todos los cambios en la nube.

### 5.3 Pull

El último comando que veremos aquí (por ahora) es el que nos permite traernos los cambios que realizaron todos los colaboradores del repositorio a nuestra PC. Si fueran los únicos colaboradores de sus repositorios, absolutamente todos los cambios de archivos posibles se generan en sus computadoras, y son subidos desde allí a la nube. En cambio, cuando se trabaja con colaboradores, ellos también realizan cambios, entonces uno debe SIEMPRE, antes de subir sus propios cambios, DESCARGAR los ya existentes para evitar problemas de coordinación en lo posible. Para ello, se ejecuta el comando:

#### ***git pull***

## Resumen final

La metodología de trabajo es rutinaria y casi siempre bastante similar:

Clono una vez sola el repositorio para traerme del servidor el estado inicial del mismo con ***git clone***, luego, todo cambio que realizo allí, debo visualizarlo con ***git status***, agregarlo con el comando ***git add***, registrarlo con ***git commit*** y subirlo con ***git push***.

Si trabajo con colaboradores (lo más usual), antes de subirlos (de ejecutar git push), se descargan los cambios con ***git pull***.

## Entrega

En esta actividad, después de efectuar cada punto, es entregar el link/dirección del repositorio web de c/u. (Recuerden que el mismo termina con .git)

## Sumario de calificaciones

No mostrado a los estudiantes

No

Participantes

23

Enviados

21

Pendientes por calificar

21



En batan.coop educación consideramos primordial la agilidad del proceso comunicativo, cualquier inquietud, duda y aporte que quieras hacernos llegar será motivo de crecimiento para nosotros y mejorará el modo en que podamos ayudarte a través de este servicio.

VER/CALIFICAR TODAS LAS ENTREGAS

Calificación

## INFO

[Batan.Coop](#)[Ecolan Soluciones](#)[Tecnológicas](#)

Ir a...

[Servicio Educativo Caraludmé](#)

## CONTÁCTANOS

Julián Ríos 4215, Batán, Buenos Aires, Argentina. Julián Ríos 4215. Código Postal: 7601

Phone : (0223) 464-3000

E-mail :

[info@cooperativabatan.com.ar](mailto:info@cooperativabatan.com.ar)

## GET SOCIAL

[Foro de consultas ►](#)

Copyright © 2020 - Desarrollado por Batan.Coop powered by Ecolan

[Resumen de conservación de datos](#)[Descargar la app para dispositivos móviles](#)