

# Instructivo Técnico

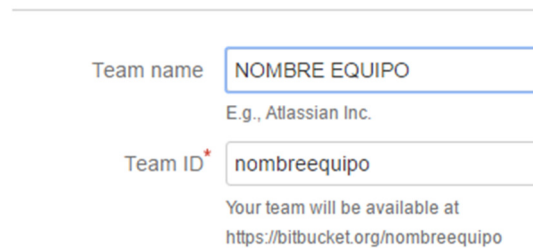
"Desarrollo colaborativo mediante software de Control de Versiones GIT"

## 1-. Crear cuenta el Servidor GIT

Existen variados servidores (pagos y no pagos) que proporcionan el servicio de control de versiones, en particular utilizaremos el proporcionado por **Bitbucket**:

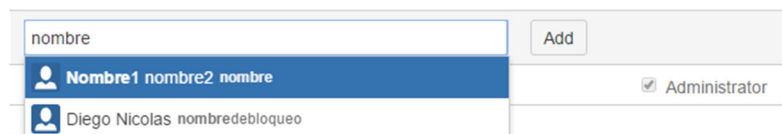
- a. Entrar a la página <https://bitbucket.org/>
- b. Crear una cuenta con su correo institucional.
- c. Cree un equipo de trabajo
  - i. Teams-> Create team
  - ii. Asigne un nombre (*Team name*) y un ID (*Team ID*) al equipo de trabajo. Ejemplo:

### Create a team



- iii. Agregue los colaboradores faltantes al equipo de trabajo en la sección *Add team members*. Asigne en cada caso los privilegios adecuados, en particular, asigne roles de administrador a cada uno de ellos.

### Add team members



- d. Cree un repositorio para su equipo de proyecto. Utilice el mismo nombre que tiene o tendrá su proyecto eclipse.



You don't have any code

Get started by putting some bits in your bucket.

Create your first repo

or

Import a repository from GitHub

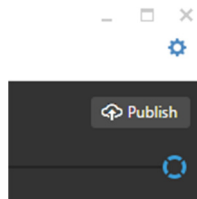
## 2-. Subir mi primer proyecto a GIT

1. El proyecto eclipse debe estar previamente creado.
  - a. Ejemplo: **`c:\workspace\nombreProyecto`**
2. Abrir una consola GIT (*git shell*) y posicionarse en la raíz del proyecto eclipse.
  - a. Ejemplo: **`cd "c:\workspace\nombreProyecto"`**
3. Convertir el proyecto eclipse en un proyecto git

- a. **git init**
4. Conectar con repositorio remoto (debe estar previamente creado)
  - a. **git remote add origin**  
***https://USUARIO@bitbucket.org/TEAM/PROYECTO.git***
5. Crear el archivo de texto **.gitignore** para evitar subir archivos indeseados de configuración, compilación y otros.
  - a. **touch .gitignore**
6. En el archivo **.gitignore** se especifica en cada línea un archivo o tipo de archivo que será ignorado al subir el proyecto. Un ejemplo del contenido es:

```
/target  
/bin  
*.bak  
*.class
```
7. Agregar todos los archivos del proyecto eclipse a git (exceptuando los ignorados)
  - a. **git add .**
8. Realizar el primer commit al repositorio local especificando una descripción para el commit.
  - a. **git commit -m "Texto que describe los cambios hechos"**
9. Subir el proyecto local al servidor remoto
  - a. **git push -u origin --all**
  - b. **git push -u origin --tags**

**NOTA:** En caso de no ser posible subir el proyecto (Problemas con Win10) es posible subir (publicar) el proyecto con GITHUB (ver punto 5).

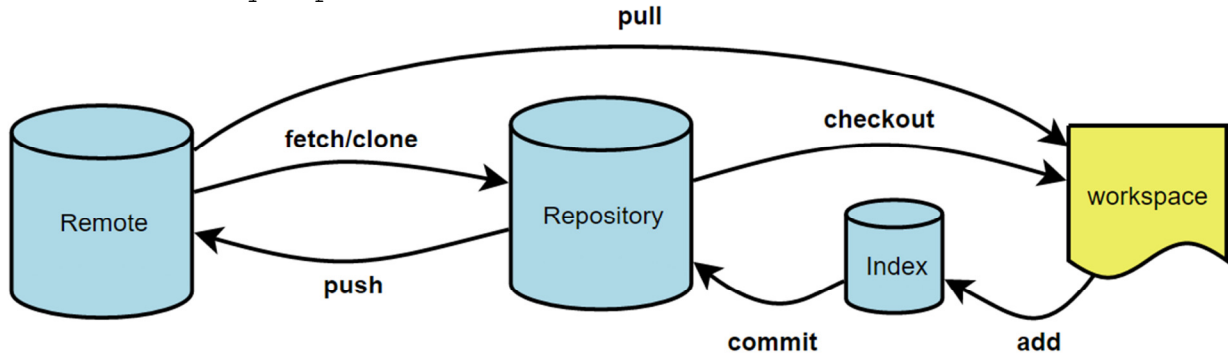


### 3-. Clonar un proyecto GIT existente

1. Se considera que los puntos 1 y 2 ya fueron ejecutados exitosamente por lo que usted se encuentra dentro de un equipo de trabajo que contiene al menos un repositorio en el servidor remoto, por lo que ahora falta descargar la última versión del proyecto:
  - a. Abrir una consola GIT (**git shell**) y posicionarse en el workspace de eclipse.
    - i. Ejemplo: **c:\workspace\**
  - b. Ejecutar el comando que permite clonar el proyecto a un entorno local
    - i. **git clone**  
***https://USUARIO@bitbucket.org/pruebaisw/pruebaiswgit.git***  
**git**
  - c. Importar el proyecto con eclipse normalmente.

## 4-. Conflictos: Actualizar y fusionar.

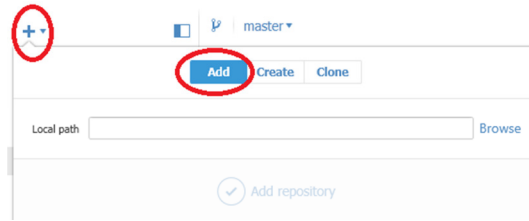
1. Para subir los cambios al repositorio remoto primero de deben indexar los archivos modificados y almacenarlos en el repositorio local. El siguiente flujo de información muestra la utilidad de los principales comandos de GIT y el flujo de información que produce cada uno:



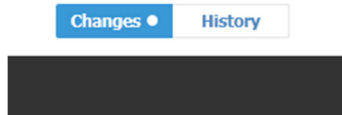
2. En primera instancia es recomendable ver el estado de nuestro proyecto GIT con el comando:  
**`git status`**
3. Posteriormente es necesario agregar los archivos modificados al índice, para ello se utiliza el comando:  
**`git add NOMBRE_ARCHIVO`**
4. Luego agregamos los cambios al repositorio local mediante un commit:  
**`git commit -m "Texto que describe los cambios hechos"`**
5. Después se descargan los cambios del repositorio remoto para que GIT realice la mezcla de contenidos de forma automática en nuestro entorno de trabajo:  
**`git pull`**
6. En caso de existir conflictos (archivos modificados en el mismo sitio por más de una persona), estos se deben resolver manualmente por medio de un editor de texto plano (puede ser eclipse). Una vez resuelto los cambios se deben incluir nuevamente al repositorio local, es decir:  
**`git add NOMBRE_ARCHIVO`**  
**`git commit -m "Texto que describe los cambios hechos"`**
7. Por último se suben los cambios al repositorio remoto para que estos sean vistos por todos los integrantes del proyecto.  
**`git push`**

## 5-. Proyecto GIT con GitHub

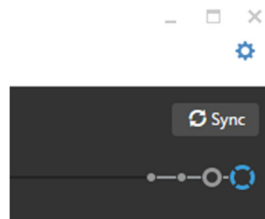
1. Una vez que disponga del proyecto de forma local es posible utilizar la aplicación **GitHub**, para facilitar el proceso de sincronización de los archivos.
  - a. Abra el programa **GitHub** y en la esquina superior izquierda presione **+** y agregue su proyecto GIT.



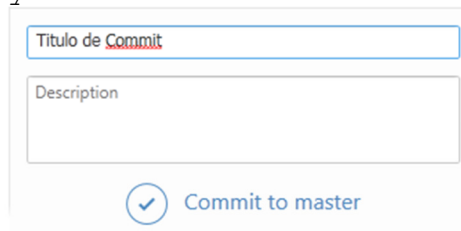
2. En el centro del programa puede ver los cambios actuales y el historial.



3. En la esquina superior derecha puede acceder a las opciones del programa y a la sincronización de su proyecto local con el proyecto remoto.



4. Para poder sincronizar su proyecto primero es necesario guardar los cambios localmente realizando un commit en el HEAD (o índice local). Cada Commit debe poseer un título y una descripción, lo cual es muy importante para que el resto de los desarrolladores del equipo sepan que archivos se modificaron y cuál fue el resultado de ello. Esto ayuda a mantener una mejor organización del proyecto.



5. En caso que se produzcan conflictos en un archivo al sincronizar (dos personas modifican el mismo trozo de código en formas diferentes), es necesario resolver dichos conflictos de forma manual, es decir:
  - a. Abra el archivo o archivos en eclipse o cualquier editor de texto plano y solucione los conflictos manualmente.
  - b. Una vez que estén solucionados los conflictos se deben agregar manualmente al repositorio local (HEAD) agregándolos al índice y luego se deben seleccionar los archivos y realizar un nuevo commit.