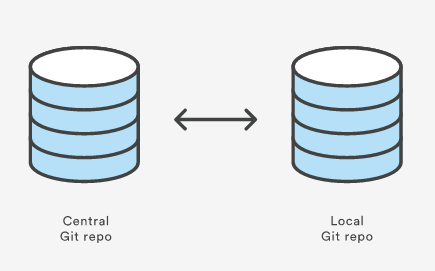
**Aprende Git**

* *Algunos datos interesantes sobre los repositorios*
  + Tienes acceso a todos los archivos de tu repositorio local, tanto se trabajas con un archivo como con varios de ellos.
  + Puedes ver los repositorios públicos sin una cuenta de Bitbucket si tienes la URL de dichos repositorios.
  + Cada repositorio pertenece a una cuenta de usuario o a un equipo. En el caso de las cuentas de usuario, ese usuario es el propietario del repositorio. En el caso del equipo, el equipo es el propietario.
  + El dueño del repositorio es la única persona que puede eliminar un repositorio.
  + Un código de proyecto puede estar formado por varios repositorios en varias cuentas, pero también puede ser un único repositorio de una única cuenta.
  + Cada repositorio tiene un límite de tamaño de 2 GB, pero recomendamos que el repositorio no supere 1 GB de tamaño.

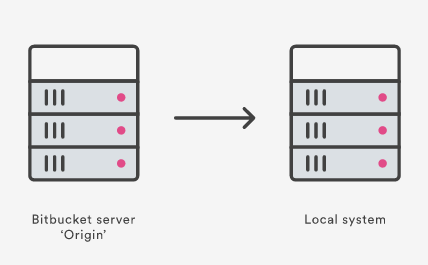
### **Paso 1. Crea el repositorio**

Inicialmente, el repositorio que creas va a estar vacío sin ningún tipo de código dentro. Este repositorio será el repositorio central para tus archivos, lo que significa que los demás pueden acceder a ese repositorio si les concedes permiso. También copiarás una versión de ese repositorio a tu sistema local: de esa forma puedes actualizarlo desde un repositorio y transferir después esos cambios al otro.



### **Paso 2. Copia tu repositorio de Git y añade los archivos**

Ahora que tienes un sitio donde añadir y compartir los archivos de tu estación espacial, necesitas una forma de acceder a este desde tu sistema local. Para ello, debes copiar el repositorio en tu PC. En Git, nos referimos a esto como "clonar" un repositorio. Cuando clonas un repositorio, creas una conexión entre el Server (especificado como origen en Git) y su sistema local.



### **Paso 2.1. Clona el repositorio a tu sistema local**

Abre un navegador y una ventana de terminal desde tu escritorio. Una vez que hayas abierto la ventana de terminal, sigue estos pasos:

1. Navega al directorio donde estará tu proyeto.

$ cd /dirmain

Cuando empieces a usar git con más frecuencia, trabajarás con varios repositorios. Por esta razón, es buena idea crear un directorio que contenga todos esos repositorios.

1. Ahí crea un directorio en el que se almacenen todos tus repositorios.

$ mkdir repos

1. Desde el terminal, actualiza el directorio en el que quieres trabajar en tu nuevo directorio de repositorios.

$ cd repos

1. Desde github, accede al repositorio **que creaste**.
2. Ve a la pestaña <> Code y ahí selecciona botón verde code y copia la url para **Clonar este repositorio**. De forma predeterminada, la clonación establece el protocolo en **HTTPS** o puedes seleccionar ahí mismo **SSH**.

Si el terminal solicita la contraseña. Si creaste una cuenta vinculada a Google, usa la contraseña de dicha cuenta.

* Si se produce un **error de contraseña de Windows**, sigue estos pasos:
  + En algunas versiones del sistema operativo Microsoft Windows y Git se puede producir un error similar al del siguiente ejemplo.

#### Ejemplo de error de contraseña de clonación de Windows

$ git clone

https://emmap1@bitbucket.org/emmap1/bitbucketstationlocations.git

Cloning into 'bitbucketspacestation'...

fatal: could not read

Password for 'https://emmap1@bitbucket.org': No such file or directory

* + Si se produce este error, introduce esta información en la línea de comandos:

$ git config --global core.askpass

1. En este momento, debería poder clonar el repositorio:

$ git clone https://github.com/EdchaSw/Language-Syntax-.git

Cloning into...

Password

warning: You appear to have cloned an empty repository.

Ya sabías que tu repositorio estaba vacío, ¿verdad? Recuerda que aún no has añadido archivos de origen que está en el servidor.

¡Enhorabuena! Has clonado tu repositorio en tu sistema local.

* Si estas en una capeta vacía y no vas a clonar sino a iniciar un proyecto en local primero

$ git init

Esto inicializa el repositorio creando una carpeta oculta .git donde se almacenara todo los archivos.

* Ahora se debe configurar algunas cosas como usuario y correo

$ git config –global user.name “nombre usuario”

$ git config –global user.email mail@gmail.com

Esto configura el nombre de usuario y correo y el flag –global le indica que ese nombre o correo será aplicado en la PC solo para el usuario actual si no se pone ese flag en ese caso la configuración será local al repositorio creado.

Para confirmar los datos se debe usar, y mostrara las dos configuraciones establecidas

$ git config –list

Para revisar solo una configuración se debe usar

$ git config user.name

O

$ git config user. email

* Ahora en vez de clonar se puede agregar el repositorio de GitHub

$ git remote add origin https://github.com/EdchaSw/Language-Syntax-.git

Ahora puedes ver a donde apunta Origin y revisar que se agrego correctamente

$ git remote -v

origin https://github.com/EdchaSw/Language-Syntax-.git (fetch)

origin https://github.com/EdchaSw/Language-Syntax-.git (push)

Fetch = Baja cambios sin hacer merge

Push = Sube los cambios

### **Paso 2.2 Añade un archivo a tu repositorio local y agrégalo a GitHub**

Cuando tengas el repositorio en tu sistema local, es hora de ponerse en marcha. Deberás comenzar a trabajar con el repositorio Origin. Para ello, crea el archivo “locations.txt” en la carpeta “repos”.

 Obtén el estado de tu repositorio local. El comando [git status](https://www.atlassian.com/git/tutorials/inspecting-a-repository/git-status) te indica el progreso del proyecto en comparación con el repositorio de GitHub.

En este punto, Git ya sabe que has creado un archivo nuevo, y verás algo parecido a esto:

$ git status

On branch master

Initial commit

Untracked files:

(use "git add <file>..." to include in what will be committed)

locations.txt

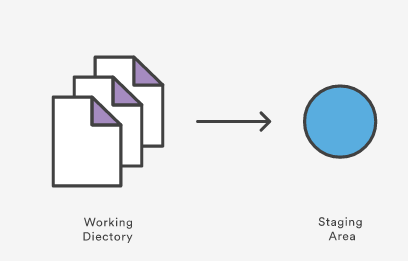
nothing added to commit but untracked files present (use "git add" to track)

El archivo esta untracked, lo que quiere decir que Git ve un archivo que no pertenece a una confirmación anterior. El resultado del estado también te muestra el siguiente paso que debes seguir: añadir el archivo.

 Indícale a Git que realice el seguimiento de tu nuevo archivo locations.txt mediante el comando [git add](https://www.atlassian.com/git/tutorials/saving-changes/git-add). el comando git add no devuelve nada cuando lo introduces correctamente.

$ git add locations.txt

El comando git add envía los cambios del directorio en curso al entorno de staging de Git. El entorno de staging es la zona en la que preparas una instantánea de un conjunto de cambios antes de confirmarlos en el historial oficial.



 Comprueba el estado del archivo.

$ git status

On branch master

Initial commit

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: locations.txt

Ahora puedes ver que se ha añadido el archivo nuevo (staging). El comando git status muestra el estado del directorio en curso y de la instantánea a la que se ha preparado.

 Introduce el comando [git commit](https://www.atlassian.com/git/tutorials/saving-changes/git-commit) con un mensaje de confirmación, como se muestra en la siguiente línea. La –m indica que incluye un mensaje de confirmación.

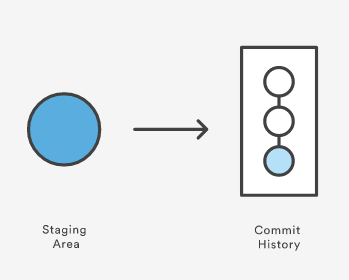
$ git commit -m 'Initial commit'

[master (root-commit) fedc3d3] Initial commit

1 file changed, 1 insertion(+)

create mode 100644 locations.txt

El comando git commit confirma la instantánea preparada en el historial del proyecto (Commit History). En combinación con git add, este proceso define el flujo de trabajo básico de todos los usuarios de Git.



Hasta este punto, todos los pasos que has seguido los has realizado en tu sistema local y de forma que no se pueden ver en el repositorio de GitHub hasta que envíes los cambios.