Desarrollo del lado servidor con NodeJS, Express y MongoDB

Instalación y configuración de GIT

Para comenzar, instalaremos GIT.

Instrucciones para instalar en Mac

Un instalador para Mac se encuentra disponible en:

http://git-scm.com/download/mac

Instrucciones para instalar en Windows

Un instalador para Windows se encuentra disponible en:

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

Instrucciones para instalar en Linux

El comando para instalar GIT será de acuerdo a la versión de Linux que se tenga. Para ver la lista de comandos según la versión ir a:

http://git-scm.com/download/linux

Configurar nuestra identidad en GIT

Ahora que tenemos GIT instalado, es hora de configurar nuestros datos en Git. Esto nos permitirá ser identificados en los cambios que hagamos a los archivos del proyecto.

Ejecutaremos los siguientes comandos, reemplazando "John Doe" y "johndoe@example.com" por tu nombre y tu email respectivamente.

```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

Al usar la opción *–global ,* le especificamos a Git que use ese nombre y el email para todos los repositorios que tengamos.

Si en un proyecto queremos usar otro nombre o email, podemos ejecutar nuevamente estos comandos sin la opción *--global*.

Bitbucket

¿Qué es Bitbucket?

Bitbucket es un servicio de alojamiento de repositorios. En vez de guardar un repositorio sólo localmente, tenemos la posibilidad de guardarlo en un servidor en la nube. Esto tiene varias ventajas:

- Nos permite tener una copia de respaldo remoto.
- Permite a otros usuarios acceder al proyecto sin tener que consultar nuestra copia local.

Para actualizar la copia remota, vamos a usar el comando "*git push*", para copiar los cambios que hicimos en nuestro repositorio local en el remoto.

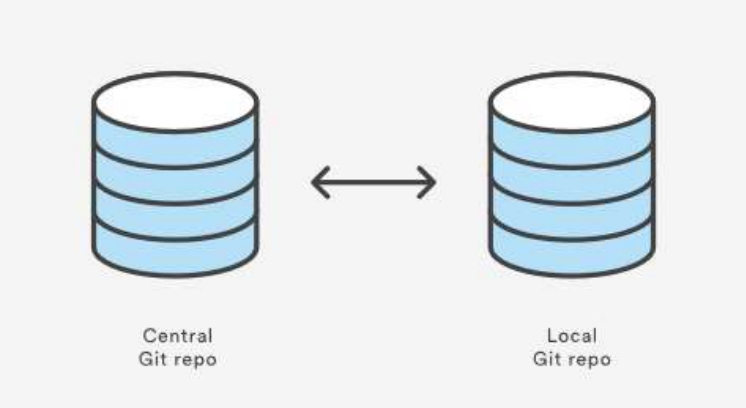
Para hacernos una cuenta en Bitbucket, tenemos que seguir los siguientes pasos:

1. Entrar a https://bitbucket.org/product
2. Tocar el botón "Get Started".
3. Seguir las instrucciones de registro para Bitbucket Cloud.

Creando nuestro primer repositorio

El repositorio que creemos con nuestra cuenta de Bitbucket va a estar inicialmente vacío, sin nada de código. Eso está bien ya que va a estar recién creado.

Este repositorio de Bitbucket va a ser el repositorio central para nuestros archivos, lo que significa que otros usuarios van a poder acceder si les damos el permiso. Después de crear el repositorio, vamos a copiar la versión de nuestro repositorio local al de Bitbucket.



De ahora en más llamaremos al repositorio de Bitbucket *"repositorio remoto"* y, al repositorio en nuestra computadora, *"repositorio local"*.

Para crear el repositorio remoto seguiremos los siguientes pasos:

1. Tocar el botón **"+"** y seleccionar "Repository".
2. Se abrirá un cuadro de diálogo con varios campos para llenar.
    a. En el campo **"Repository name"** vamos a poner: "test-git"
    b. El siguiente campo nos pide determinar el nivel de privacidad del repositorio. Si lo ponemos privado, entonces podemos determinar quién puede verlo y quién no. Por ahora lo pondremos como privado.
    c. En el campo **"Include a README?"** vamos a seleccionar que NO. Un Readme es un archivo que contiene información sobre el proyecto o las configuraciones a tener en cuenta. Lo vamos a usar más tarde para proveer una descripción general de nuestro proyecto.
    d. En el selector **Version Control** seleccionamos Git, por ser el VCS que vamos a utilizar. **Este es el único campo que no se puede modificar más tarde.**
    e. Haz clic en "Create repository" para que Bitbucket cree el repositorio y te redirija a la página general del repositorio.

Este es un buen momento para explorar todos los menues y opciones que tiene el repositorio. Si entras a la sección de “Commits”, podrás ver que no hay commits hechos. Esto está bien ya que todavía no hemos agregado ningún archivo al repositorio.

Hacer una copia local de nuestro repositorio

Ya tenemos un lugar para agregar y compartir nuestros archivos de trabajo; ahora tenemos que copiarlo a nuestro sistema local. Para hacer esto queremos copiar nuestro repositorio de Bitbucket en nuestro sistema.

El término que GIT usa para copiar un repositorio es “clonarlo”. Cuando clonamos un repositorio, creamos una conexión entre el servidor de Bitbucket (conocido por GIT como “origin”) y nuestro sistema local.

En este momento, debes crear una carpeta donde vas a clonar el reporsitorio de Bitbucket. Para este turorial, elegimos el nombre “repos”.

Para hacer una copia local de nuestro repositorio vamos a seguir los siguientes pasos:

1. Tocar el ícono **“+”** en la barra lateral y seleccionar “Clone this repository”.
2. Bitbucket mostrará un diálogo con un comando y un selector.

. a. Con el selector podemos elegir qué protocolo usar, si HTTPS o SSH. Por ahora simplemente lo dejaremos en HTTPS.

b. El comando que aparece en el cuadro es el que tenemos que usar para clonar el repositorio.

3. Copiar el comando del diálogo.
4. Con la Terminal navegamos hacia nuestra carpeta “repos”.
5. Parados con la Terminal en nuestra carpeta “repos”, pegamos el comando y damos Enter.
6. Ingresamos la contraseña de Bitbucket cuando sea pedida.

7. Chequeamos que se haya creado una carpeta con el nombre de nuestro repositorio.

Agregar archivos a nuestro repositorio

Ahora que tenemos el repositorio en nuestro sistema local, podemos empezar a agregarle archivos. Por ahora, vamos a agregar un archivo de texto (.txt) con nombres de ciudades.

1. Con la Terminal, navegamos hasta la carpeta “test-git”.
2. Ingresamos el siguiente comando para crear un archivo “lugares.txt” que contenga el texto “Buenos Aires”.

```
echo "Buenos Aires" >> lugares.txt
```

Si el comando no muestra nada más, significa que hemos creado el archivo correctamente.

También puedes crear el archivo y guardarlo en la carpeta “test-git” utilizando el explorador de Windows.

3. Vamos a ver el estado de nuestro repositorio. El comando ***git status*** nos dice cómo está nuestro repositorio local en relación a nuestro repositorio remoto.

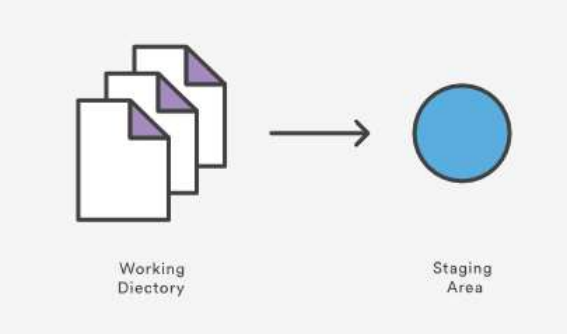
```
$ git status
On branch master
No commits yet
Untracked files:
  (use "git add <file>..." to include in what will be
committed)
      lugares.txt
nothing added to commit but untracked files present (use "git
add" to track)
```

Podemos notar que nuestro nuevo archivo nos aparece como “untracked”. Esto es porque GIT no encuentra ninguna referencia a este archivo en versiones anteriores.

4. Le vamos a indicar a GIT que empieza a trackear (rastrear) el archivo “lugares.txt”. Para esto, vamos a usar el comando “***git add*** lugares.txt”. Si lo usamos correctamente, el comando no mostrará ningún mensaje.

```
$ git add lugares.txt
```

El comando ***git add*** mueve los cambios de nuestro “Working directory” a nuestra “Staging Area”.



5. Verifiquemos que hayamos hecho todo correctamente. Para eso chequeamos el estado de nuestro repositorio local nuevamente, mediante el comando ***git status***.

```
$ git status
On branch master
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
      new file:   lugares.txt
```

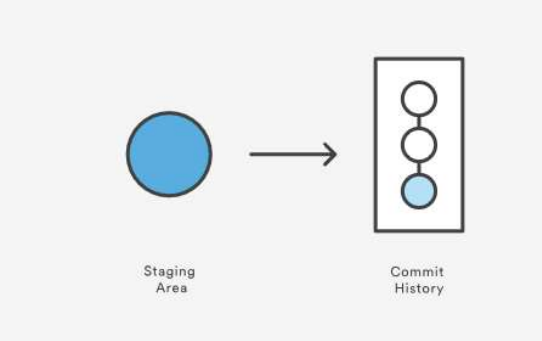
Podemos ver que nuestro archivo nuevo ha sido agregado (staged) y está listo para ser committeado cuando queramos.

6. Para committear el archivo, ejecutamos el comando ***git commit*** con un mensaje de commit. La opción ‘-m’ indica que un mensaje de commit le sigue. Es importante elegir un nombre de commit suficientemente descriptivo de los cambios que se están versionado para dar claridad en la lectura de las versiones generadas.

```
$ git commit -m "Commit inicial"
[master (root-commit) cdb6d5a] Commit inicial
```

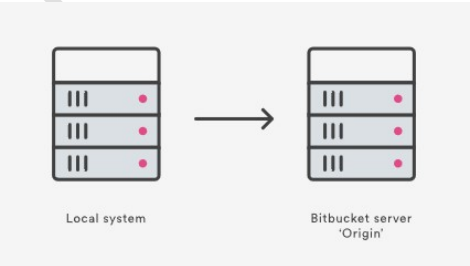
```
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 lugares.txt
```

7. El comando ***git commit*** toma todos los archivos del área de staging y los committea a la historia del proyecto.



Ya pudimos agregar una versión de nuestro proyecto en nuestro repositorio local. Para enviarlos a nuestro repositorio remoto, usaremos el comando ***git push origin master***. Este comando especifica que se envíen los cambios al branch 'master' (el branch en Bitbucket) en 'origin' (el repositorio en Bitbucket). Todo lo referido a branches puedes verlo en la documentación de Git. Para el resto de los ejemplos utilizaremos un único branch: "master". Es interesante y muy práctico el uso de diferentes branches para trabajar. Te animo a que los investigues y los pruebes.

```
$ git push
Counting objects: 3, done.
Writing objects: 100% (3/3), 269 bytes | 269.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://bitbucket.org/ejemplo/test-git.git
* [new branch]      master -> master
```



Si ahora entramos a nuestro repositorio en Bitbucket, podremos ver que ya no es igual que antes. Si entramos a la sección de **"Source",** podremos ver el nombre del archivo que subimos con la última fecha y mensaje con el que fue committeado.

Si entramos a la sección de "Commits", podremos ver el commit que acabamos de hacer.