

4 NOVIEMBRE, 2018 4 NOVIEMBRE, 2018 / HANSEL THOMAS

Como enviar repositorio local a GitHub desde Git

Introducción:

Una de las formas de administrar nuestros proyectos hoy en día, es a través de lugares específicos llamados repositorios, estos repositorios nos ayudan a controlar el avance de nuestro proyecto de mejor manera.

Con Git es posible crear repositorios locales para realizar cambios, pero es importante contar con un repositorio remoto alojado en un sitio hosting de confianza, como por ejemplo GitLab o GitHub.

En este pequeño tutorial realizaremos un ejercicio para enviar nuestros cambios por primera vez a nuestro repositorio remoto creado, en este caso, sera en GitHub.

2.-Entorno

Sistema de control de versiones: Git 2.18.0

Sistema Operativo: Windows 7

Sitio hosting remoto: GitHub

3.-Pre-requisitos

-Tener instalado el sistema de control de versiones Git. [5 pasos para instalar Git en Windows](https://myhanoli.com/2018/08/25/5-pasos-para-instalar-git-en-windows/) (<https://myhanoli.com/2018/08/25/5-pasos-para-instalar-git-en-windows/>).

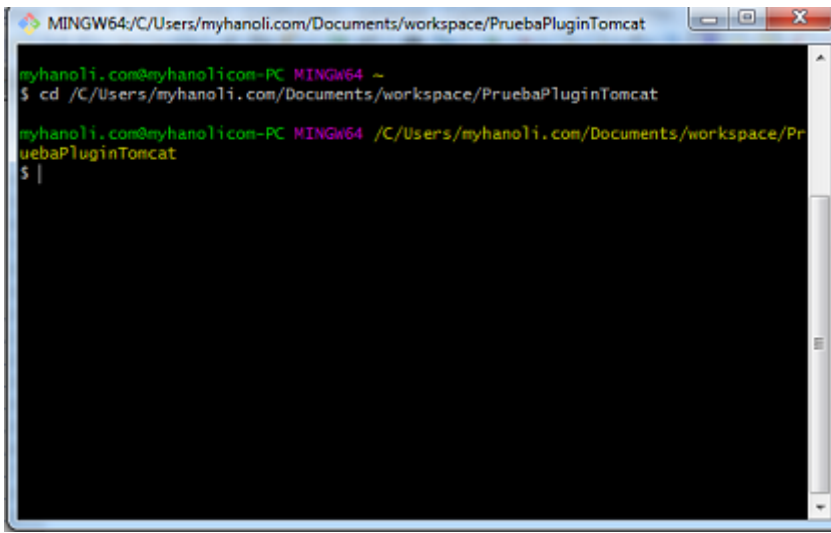
-Contar con un proyecto construido en Maven. [Como crear un proyecto web con maven desde eclipse](https://myhanoli.com/2018/08/11/como-crear-un-proyecto-web-con-maven-desde-eclipse/) (<https://myhanoli.com/2018/08/11/como-crear-un-proyecto-web-con-maven-desde-eclipse/>).

-Tener una cuenta creada, y un repositorio remoto en el portal de GitHub

4.- Instalando carpeta Git en nuestro proyecto

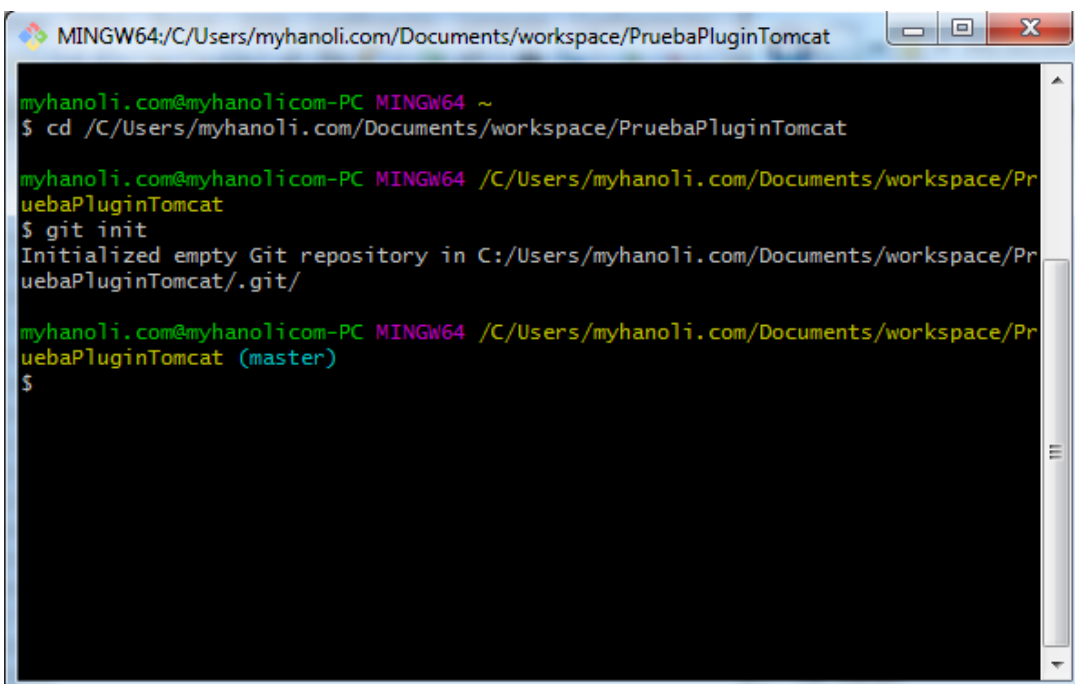
Para que Git pueda gestionar los archivos o carpetas que han tenido algún cambio, como primer paso habría que instalar la carpeta Git dentro de nuestro proyecto, para realizar esto...

Abrimos la línea de comando Git Bash y nos ubicamos en la ruta de nuestro proyecto con el comando cd, es importante ubicarnos dentro de la carpeta donde se encuentra el archivo POM.xml, como lo muestra la siguiente imagen



```
MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
myhanoli.com@myhanoli.com-PC MINGW64 ~
$ cd /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
$ |
```

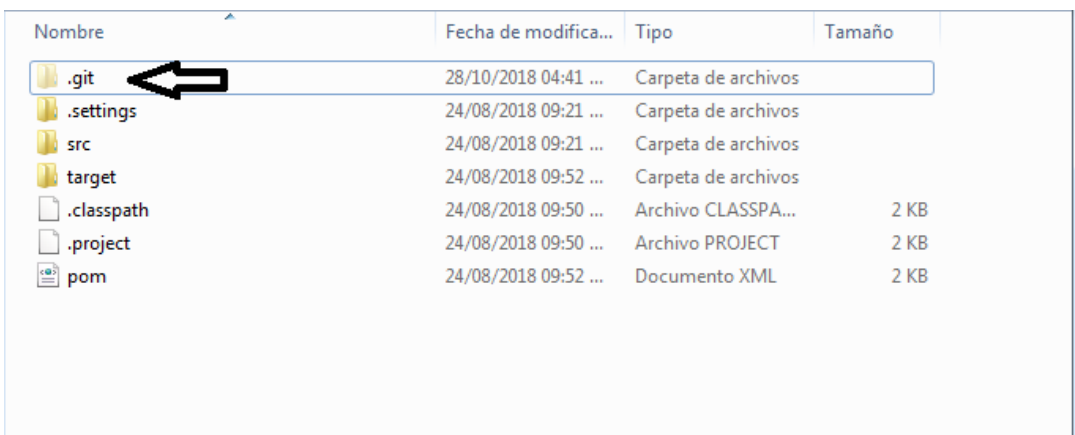
Una vez dentro lanzamos el comando git init como lo muestra la siguiente imagen



```
MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
myhanoli.com@myhanoli.com-PC MINGW64 ~
$ cd /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
$ git init
Initialized empty Git repository in C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat/.git/
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$
```

Si no hubo ningún problema, nos lanzara el mensaje **“Initialized empty Git repository in:”**

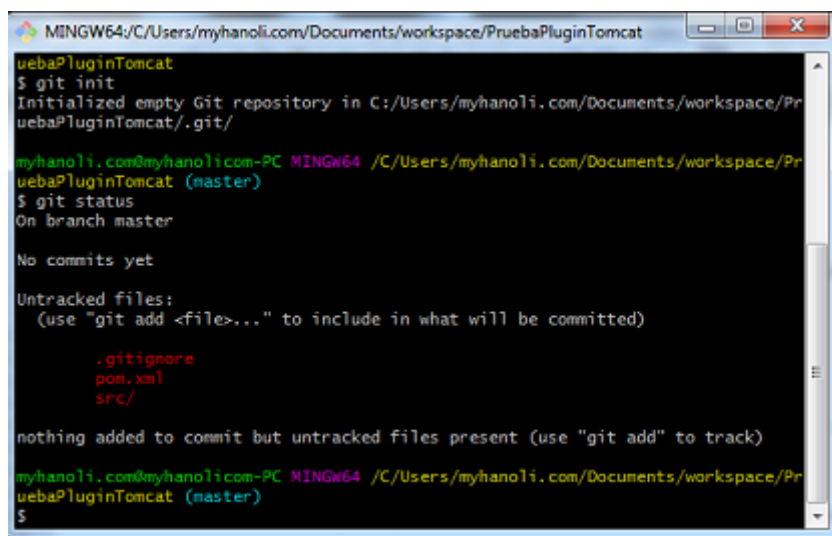
Este comando creara una carpeta con el nombre git, en el directorio de nuestro proyecto para que git, pueda gestionar los cambios (La carpeta se instala de manera oculta, para visualizarla, habría que habilitar la herramienta de carpetas ocultas).



Nombre	Fecha de modifica...	Tipo	Tamaño
.git	28/10/2018 04:41 ...	Carpeta de archivos	
.settings	24/08/2018 09:21 ...	Carpeta de archivos	
src	24/08/2018 09:21 ...	Carpeta de archivos	
target	24/08/2018 09:52 ...	Carpeta de archivos	
.classpath	24/08/2018 09:50 ...	Archivo CLASSPA...	2 KB
.project	24/08/2018 09:50 ...	Archivo PROJECT	2 KB
pom	24/08/2018 09:52 ...	Documento XML	2 KB

5.- Primer comando Git dentro de nuestro proyecto

Después de lanzar el comando `git init`, lanzamos el comando `git status` para revisar el estatus del repositorio, básicamente lo que hace este comando es revisar dentro de nuestro proyecto, si existe algún cambio o archivo nuevo que no se haya pasado de nuestro Working Directory (Directorio de Trabajo), al Staging Área (Área de preparación) o del Área de preparación al Repositorio Git .



```

MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
uebaPluginTomcat
$ git init
Initialized empty Git repository in C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat/.git/

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git status
On branch master

No commits yet

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

        .gitignore
        pom.xml
        src/

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

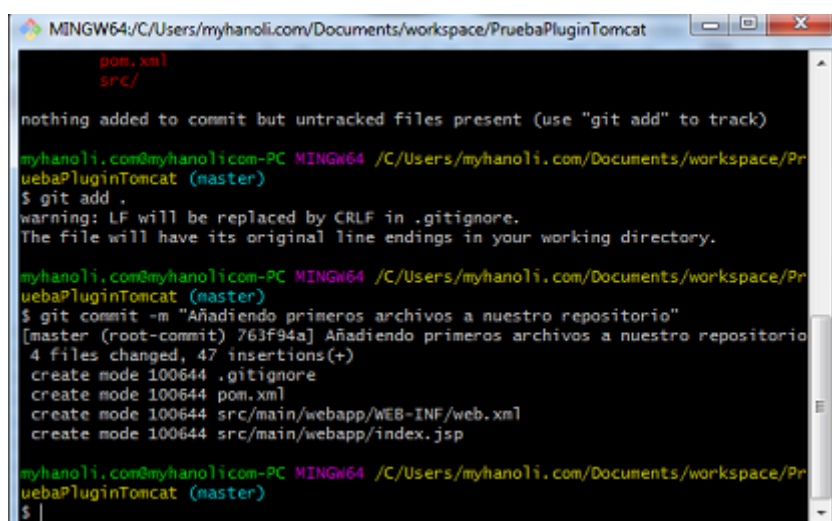
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$
  
```

Observamos que nos manda un mensaje de Untracked files (Archivos sin seguimiento), este mensaje lo muestra, porque los archivos aún no se encuentran en el Staging Area o area de preparación de Git y hay que añadirlos.

6.-Añadiendo cambios al Staging Area

El Staging Area o área de preparación de Git, es el área en el cual se encuentran los cambios que ya han sufrido modificación y que están listos para ser confirmados con un commit

Para realizar esta acción solo hay que lanzar el comando `git add` . como lo muestra la siguiente imagen



```

MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
        pom.xml
        src/

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

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git add .
warning: LF will be replaced by CRLF in .gitignore.
The file will have its original line endings in your working directory.

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git commit -m "Añadiendo primeros archivos a nuestro repositorio"
[master (root-commit) 763f94a] Añadiendo primeros archivos a nuestro repositorio
4 files changed, 47 insertions(+)
create mode 100644 .gitignore
create mode 100644 pom.xml
create mode 100644 src/main/webapp/WEB-INF/web.xml
create mode 100644 src/main/webapp/index.jsp

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$
  
```

7.-Confirmando cambios

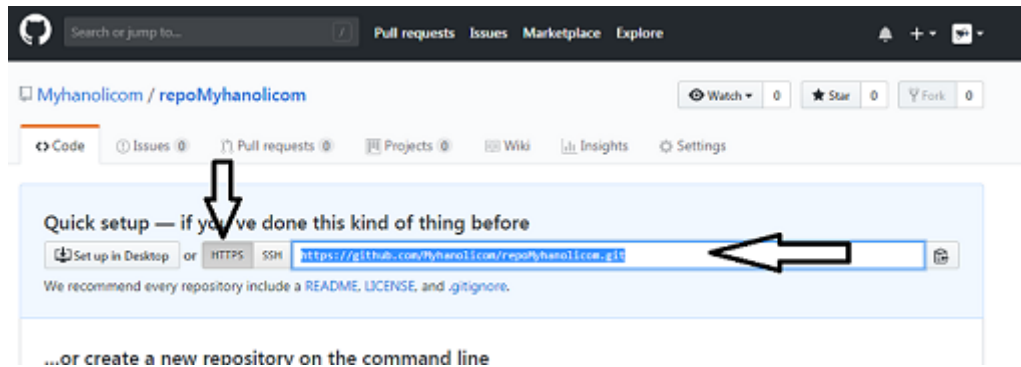
Para confirmar nuestros cambios en nuestro repositorio local, solo basta con lanzar el comando `git commit -m "Comentario"` y quedaran confirmados nuestros cambios, listos para ser enviados a nuestro repositorio remoto.

8.-Agregando repositorio remoto a nuestro local

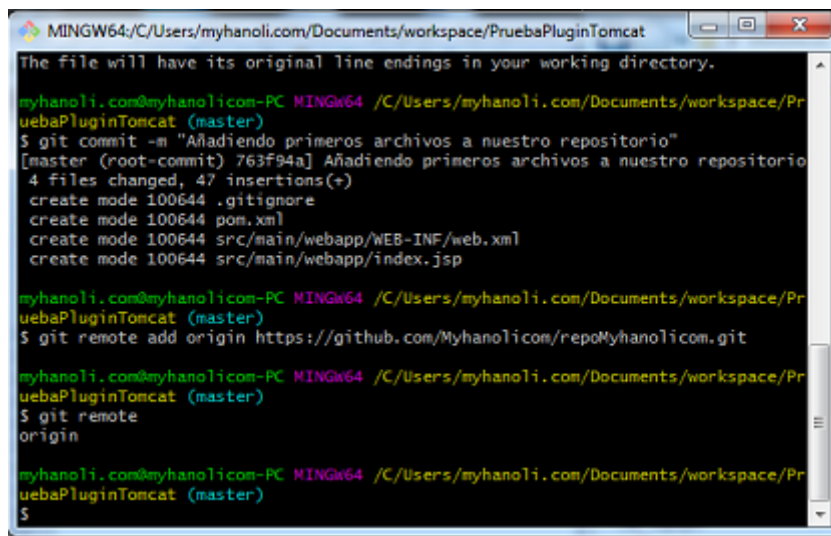
Para que podamos enviar los cambios desde un repositorio local a un repositorio remoto, es importante agregar el repositorio remoto a nuestro local (esto lo haremos solamente una vez), para ello.

Nos dirigimos a nuestro repositorio remoto en el portal de GitHub y copiamos la ruta del repositorio donde queremos agregar nuestro proyecto

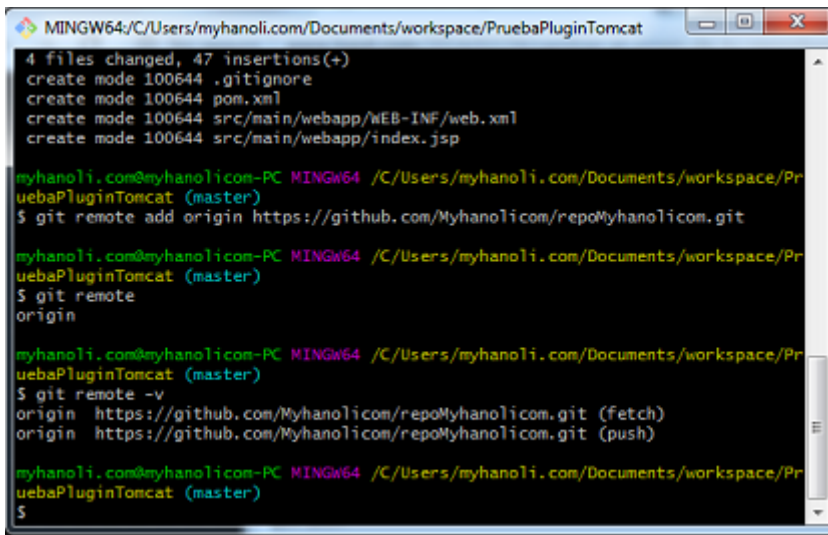
En este caso sería: <https://github.com/Myhanolicom/repoMyhanolicom.git>
(<https://github.com/Myhanolicom/repoMyhanolicom.git>).



Dentro de la línea de comandos Git Bash, y dentro de la ubicación donde instalamos la carpeta Git dentro de nuestro proyecto, agregamos el repositorio con el comando `git remote add + "NombreRepositorioRemoto"`, como se muestra en la siguiente imagen



Para revisar que si lo haya agregado podemos lanzar el comando `git remote -v` y observaremos el nombre de los repositorios remotos que hayamos agregado



```

MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
4 files changed, 47 insertions(+)
create mode 100644 .gitignore
create mode 100644 pom.xml
create mode 100644 src/main/webapp/WEB-INF/web.xml
create mode 100644 src/main/webapp/index.jsp

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git remote add origin https://github.com/Myhanolicom/repoMyhanolicom.git

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git remote
origin

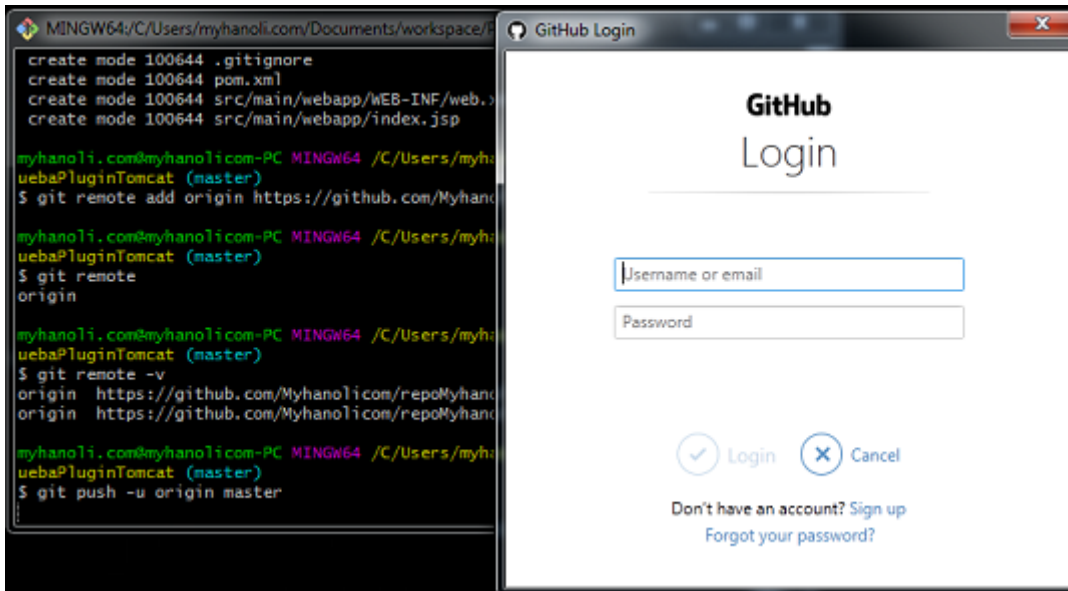
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git remote -v
origin https://github.com/Myhanolicom/repoMyhanolicom.git (fetch)
origin https://github.com/Myhanolicom/repoMyhanolicom.git (push)

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$

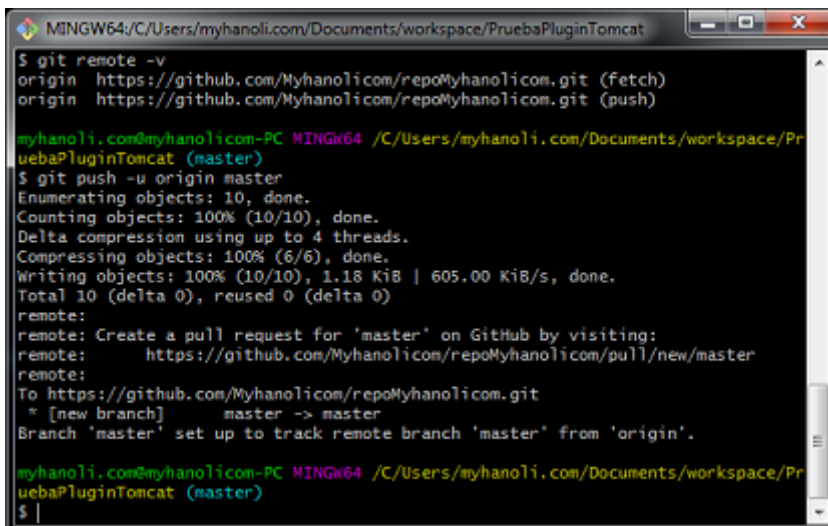
```

9.-Enviando cambios a GitHub

Una vez teniendo nuestro repositorio remoto configurado en nuestro local, solo basta con lanzar el comando `git push -u origin master`, y una vez lanzado el comando, Git nos pedirá los accesos del repositorio remoto como lo muestra la siguiente imagen



Y una vez ingresando las credenciales de acceso al repositorio remoto, se enviarán los archivos de nuestro repositorio local al remoto



```

MINGW64/C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat
$ git remote -v
origin https://github.com/Myhanolicom/repoMyhanolicom.git (fetch)
origin https://github.com/Myhanolicom/repoMyhanolicom.git (push)

myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$ git push -u origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (10/10), 1.18 KiB | 605.00 KiB/s, done.
Total 10 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Myhanolicom/repoMyhanolicom/pull/new/master
remote:
To https://github.com/Myhanolicom/repoMyhanolicom.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

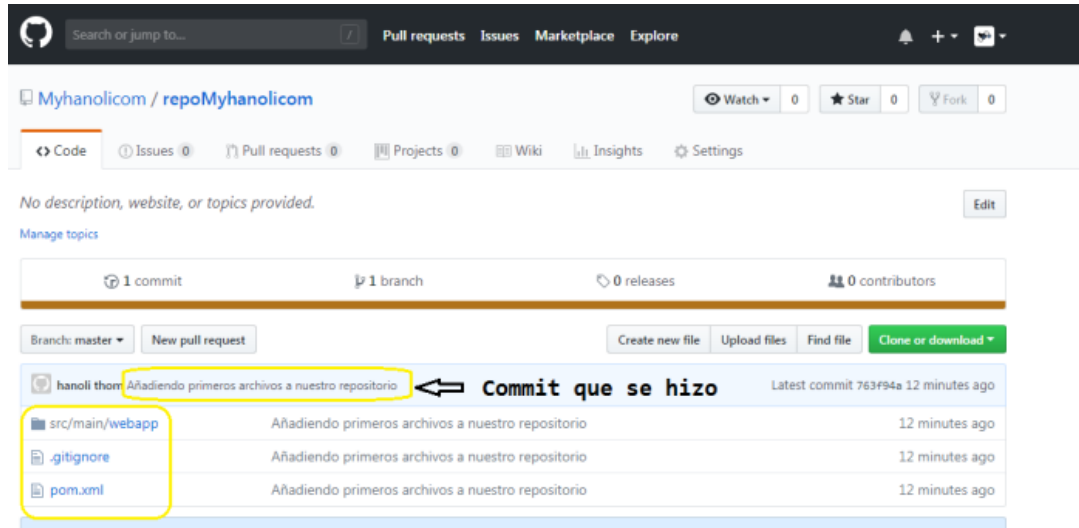
myhanoli.com@myhanoli.com-PC MINGW64 /C:/Users/myhanoli.com/Documents/workspace/PruebaPluginTomcat (master)
$

```

Como vemos en la imagen anterior, se han enviado los cambios al repositorio remoto, creando un nuevo branch en nuestro repositorio remoto llamado master, este branch, será nuestro branch principal, en el que nosotros como programadores y parte de nuestro equipo trabajara.

10.-Revisando que se encuentren los cambios subidos en el repositorio remoto en GitHub

Para ello solo basta con dirigirnos a nuestro repositorio remoto y observaremos que ahí se encuentran los archivos y el commit previamente lanzado.



Hemos llegado al final de este pequeño tutorial, esperando les sea de su agrado. No olviden dejar sus comentarios para seguir aportando al conocimiento compartido.

Referencias:

<https://git-scm.com/book/en/v2/Getting-Started-Git-Basics> (<https://git-scm.com/book/en/v2/Getting-Started-Git-Basics>)

Publicado en [Git/Deja un comentario](#)



