## IMPORTANTE

Git por default hace el push a todas las branches existentes, lo cual es confuso y lleva a errores. Para evitar esto se puede ejecutar: **git config --global push.default upstream**

De esta forma le indicamos que por default los push sean contra el branch en el que estamos.Ojo que al usar upstream no es obligatorio que el branch local y el remoto se llamen iguales.

## Armado de un pryecto local, subirlo a github

A. Armar una carpeta llamada **PRUEBA1**, crear el archivo **index.html**. Colocamos un h1 con el texto "Titulo" y un h2 con el texto "Version 1"
B. Creamos en github un nuevo repositorio llamado **RepositorioTesting**. Debe ser publico y NO debe poseer Readme.md ni .gitgnore ni licencia
C. En la carpeta PRUEBA1 ejecutar: **git init**
**D.** En la carpeta PRUEBA1 ejecutar: **git add index.html**
E. En la carpeta PRUEBA1 ejecutar: **git commit -m "primer commit "**
F. En la carpeta PRUEBA1 ejecutar: **git branch -M main**
G. En la carpeta PRUEBA1 ejecutar: **git remote add origin <url>** (reemplazar <url> por la url del repositorio)
H. En la carpeta PRUEBA1 ejecutar: **git push -u origin main**

Con todo lo anterior:

- Ya tenemos el repositorio local en nuestra carpeta PRUEBA1 y el repositorio online en github (llamado RepositorioTesting)
- Tenemos armado el branch principal (llamado main) y el mismo existe tanto en github como de forma local
- El branch main local y el branch main de github se encuentran iguales (se puede verificar haciendo un git status)
- Actualmente estamos sobre el branch main (se puede verificar haciendo un git status)
- Si se va a github se puede ver en el branch main el archivo index.html

## Hacer un cambio directamente sobre el branch main de forma local

1) Cambiamos el texto del h2 a "Version 2". Con este cambio el archivo aparece en amarillo
2) Comparamos el repositorio local contra nuestros archivos locales ejecutando: **git status** (deberiamos ver index.html en rojo ya que no esta en el stage)
3) Grabamos el cambio realizado a index.html en el repositorio local:
    a) Pasamos al stage el archivo index.html ejecutando: **git add index.html**
    b) Verificamos que este en staged el archivo index.html ejecutando: **git status** (deberiamos verlo en verde)
    **c)** Grabamos la modificacion con el mensaje "paso a version 2" ejecutando: **git commit -a -m "paso a version 2"**
    d) Verificamos el estado actual con **git status**
        i) Como resultado vamos a ver el mensaje "Your branch is ahead of 'origin/main' by 1 commit." ya que en nuestro repositorio local tenemos el branch main que ya tiene un commit extra comparado con el branch main de github
    **e)** Grabamos en github el commit ejecutando: **git push -u origin main**
        i) Podemos verificar ahora que en github el h2 tiene el texto "version 2"

## Hacer cambios sobre un branch secundario e integrarlo al branch main

El objetivo de este branch es pasar el h2 al texto "Version ALFA”

1) Ejecutamos: **git checkout -b branch VersionAlfa**
   a) Esta sentencia hace cosas: Crea el branch y nos posiciona sobre el mismo
2) Verificamos estar sobre el branch branchVersionAlfa ejecutando: **git status**
   a) deberiamos ver el mensaje "On branch branchVersion3"
3) Creamos el branch "branchVersion3" en github ejecutando: **git push -u origin branchVersionAlfa**
   a) " * [new branch]  branchVersionAlfa -> branchVersionAlfa" nos indica que se creo en github el nuevo Branch
   b) branch 'branchVersionAlfa' set up to track 'origin/branchVersionAlfa'." indica que ahora estamos parados de forma local en el branch banchVersionAlfa
4) Si vamos a github debe estar creado el nuevo Branch
5) Modificamos el h2 colocandole el texto "Version ALFA"
6) verificamos que archivos tienen cambios ejecutando: **git status**
7) Pasamos el index.html al stage ejecutando: **git add index.html**
8) verificamos que el archivo index.html este en el stage ejecutando: **git status**
9) Grabamos todos los cambios en el stage de forma local ejecutando: **git commit -a -m "paso a version ALFA"**
10) Verificamos el status actual ejecutando: **git status**
    a) Nos va a indicar que estamos en el branch "branchVersionAlfa" en la sentencia "On branch branchVersionAlfa"
    b) Nos va a indicar que tenemos un commit extra de forma local sobre nuestro branch "branchVersionAlfa" en comparacion al mismo branch en github mediante el mensaje "Your branch is ahead of 'origin/branchVersionAlfa' by 1 commit."
11) Subimos el commit ejecutado al branch branchVersionAlfa de github ejecutando: **git push -u origin branchVersionAlfa**
12) Integramos todos los cambios del branch "branchVersionAlfa" en el branch "main" de forma local.
    a) Para esto primero nos aseguramos estar posicionados en el branch main ejecutando: **git checkout main**
    **b)** Ejecutamos un PULL solicitando integrar el branch "branchVersionAlfa" en el branch "main" ejecutando: **git merge branchVersionAlfa**
13) Verificamos que ahora nuestro branch main local esta un commit adelantado al branch main remoto ejecutando: **git status**
14) Subimos el merge realizado en el branch main local al branch main remoto ejecutando: **git push -u origin main**
    a) Si vamos ahora a github vemos que el branch main tiene en el h2 el texto "Version ALFA"

## Retomar un Branch inconcluso creado por otra persona, terminarlo e integrarlo al Branch main

Para este ejemplo vamos a suponer que el empleado Diego debe cambiar el h1 para que diga "Proyecto Escarlata" y el h2 para que diga "Dirigido por Christopher Nolan". Sin embargo, vamos a simular que a Diego no le alcanza el tiempo para hacer el segundo cambio por lo cual otra persona externa (Laura) tendra que realizar el mismo

1) El primer dia Diego modifica el h1 para que diga "Proyecto Escarlata"

    a) Creamos el nuevo branch: **git checkout -b branchVersionEscarlata**
    b) Subimos el branch actual (brancVersionEscarlata) a github ejecutando: **git push -u origin branchVersionEscarlata**
    c) Modificar el index.html para que en el h1 diga "Proyecto Escarlata"
    d) Agregamos el index.html al stage ejecutando: **git add index.html**
    e) Grabamos los cambios en el stage de forma local ejecutando: **git commit -a -m "paso a version Escarlata"**

2) Al terminar su dia de trabajo Diego graba lo realizado de forma local en su branch del repositorio remoto

    a) subimos el commit al branch remoto "branchVersionEscarlata" ejecutando: **git push -u origin branchVersionEscarlata**

3) Al otro dia a Diego le piden que ayuede con otro proyecto por lo cual queda Laura a cargo de terminar estas modificaciones. Lo primero que hace Laura es traerse el proyecto de github de forma local por lo cual:

    a) Abre con el visual studio code su carpeta de proyectos (donde tiene todos sus proyectos)
    b) Abre un terminal en visual studio code
    c) En el terminal ejecuta: **git clone <url>** (donde url es la url del proyecto a traerse de forma local)
    d) Abre con el visual studio code la carpeta creada
    e) Ejecuta **git status** y ve que inicia en el branch main.
    **f)** Diego ya le indicó en un mail que los cambios los debe segir sobre el branch "branchVersionEscarlata" por lo caul se mueve a ese branch ejecutando: **git switch branchVersionEscarlata**
    g) Cambia el h2 para que diga "Dirigido por Christopher Nolan"
    h) Pasa el index.html al stage ejecutando : **git add index.html**
    i) Realiza el commit local ejecutando: **git commit -a -m "cambio en subtitulo"**
    j) Actualiza el branch branchVersionEscarlata del repositorio remoto ejecutando: **git push -u origin branchVersionEscarlata**
    k) Terminados todos los cambios decide integrar el branch branchVersionEscarlata al branch main. Para esto:

        i) Nos colocamos en el branch main ejecutando: **git checkout main**

**ii)** Ejecutamos un PULL solicitando integrar el branch "branchVersionEscarlata" en el branch "main" ejecutando: **git merge branchVersionEscarlata**

**iii)** Subimos al branch main los cambios ejecutando: **git push -u origin main**

## Lista de comandos basicos

**git init**

Crea un repositorio local en la carpeta actual

**git add <archivo>**

Agrega el <archivo> modificado al stage

**git commit -a -m "<mensaje>"**

Realiza el commit de todos los archivos (-a) en el Branch actual de forma local. Asocia a ese commit el mensaje especificado en <mensaje>

**git remote add origin <url>**

Asocia el git local al git remoto especificado en <url>

**git push -u origin <branch>**

Lleva los cambios comiteados del Branch local actual al Branch remoto especificado en <branch>. Si el Branch remoto no existe lo crea.

**git checkout -b <Branch-name>**

Crea el Branch <Branch-name> (si no existe) y te mueve localmente a trabajar con ese Branch

**git switch <Branch-name>**

Hace que pasemos a trabajar con el Branch especificado en <Branch-name>

**git merge <Branch-name>**

Une el Branch actual con el Branch especificado en <Branch-main>. Esto implica comparar el Branch actual con el Branch <Branch-name> y modificar el Branch actual con las diferencias encontradas.

**git status**

Nos muestra el Branch actual, los archivos con cambios en el stage, los archivos con cambios que no están en el stage y los archivos a los cuales no les estamos haciendo un seguimiento.

**git clone <url>**

Carga el repositorio externo indicado en <url> en el repositorio actual local

**git Branch -d <Branch-name>**

Elimina el branch <Branch-name> local

**git push origin --delete <Branch-name>**

Elimina el branch <Branch-name> remoto