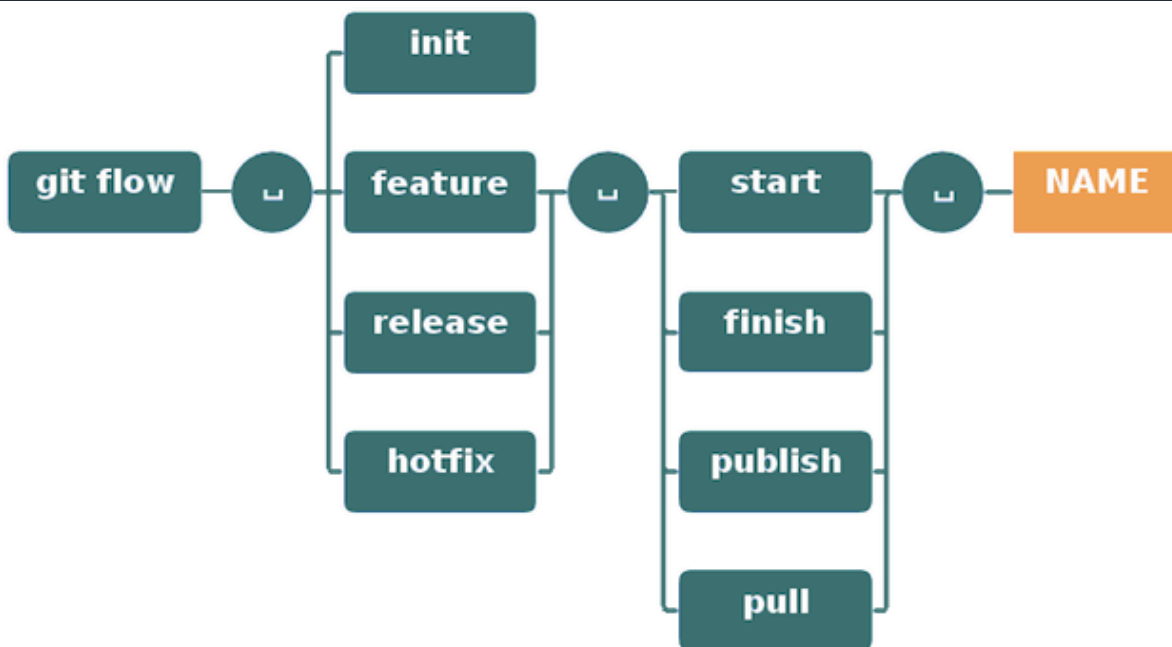




UNIVERSIDAD
TECNOLÓGICA
DE AGUASCALIENTES

Profesor: Alberto Campos Hernández



Alumno: Omar Alejandro Sánchez Cuevas

Materia: Desarrollo Web Profesional

Grupo: IDGS 8-B-11

Fecha: 24/05/2022

Como programadores, rara vez trabajamos en solitario en un proyecto de gran envergadura. Lo normal es que nos acompañen otros desarrolladores que colaborarán con nosotros mano a mano. A veces nos encontramos con situaciones como tener que trabajar en funcionalidades nuevas, arreglar bugs críticos, acabar una iteración y tener que juntar todas las funcionalidades que se han desarrollado hasta el momento, o incluso dejar lo que estábamos haciendo para seguir con otra tarea más importante o probar implementaciones.

Estas situaciones descritas anteriormente pueden llegar a ser un calvario si un equipo de desarrollo carece de algún sistema de organización en su repositorio de control de versiones.

Es ahí donde entra en juego la metodología Git flow, que como su nombre indica, es un flujo de trabajo aplicado a un repositorio Git. Vincent Driessen fue el encargado de popularizarlo, definiendo un modelo estricto de ramificación diseñado en torno a los lanzamientos del proyecto. Es ideal para proyectos que lleven una planificación de entregas iterativas. Permite la paralelización del desarrollo mediante ramas independientes para la preparación, mantenimiento y publicación de versiones del proyecto, así como soporta la reparación de errores en cualquier momento.

Ramas Principales

Todo proyecto, por defecto, debería tener al menos dos ramas infinitas para su desarrollo. Esta metodología define que deben existir dos ramas principales:

- master
- develop

Rama master

Contiene cada una de las versiones estables del proyecto. Cualquier commit que subamos en esta rama debe estar preparado para que se pueda incluir en producción.

Rama develop

Contiene el código de desarrollo de la siguiente versión planificada del proyecto. En ella se incluirán cada una de las nuevas características que se desarrollen. Esta rama puede incorporarse tanto en una rama release (que veremos más adelante) como en la rama master, para su despliegue en producción.

Ramas de apoyo

Junto a las ramas master y develop, existe un conjunto de ramas de apoyo, que como hemos descrito anteriormente, su objetivo es el permitir el desarrollo en paralelo entre los miembros del equipo, la resolución de problemas en producción de forma rápida, etc. A diferencia de las ramas principales, estas están limitadas en tiempo. Serán eliminadas eventualmente. Los diferentes tipos de ramas que se usarán son:

- feature
- release
- hotfix

Ramas feature

Estas ramas tienen que surgir de la rama develop. Cada una de estas ramas almacenan código de desarrollo con nuevas características. Típicamente existen solamente en los repositorios locales de los desarrolladores y no en el repositorio origen. Una vez terminado su desarrollo, se incorporarán nuevamente a la rama develop, que contendrá la última versión de código en desarrollo.

Convención de nombres: estas ramas se pueden nombrar de cualquier forma, excepto master, develop, release-*, o hotfix-*.

Crear una rama feature

```
$ git checkout -b feature/myfetaure develop  
Switched to a new branch "feature/myfetaure"
```

Finalizar una rama feature

```
$ git checkout develop  
Switched to branch 'develop'  
  
$ git merge --no-ff feature/myfetaure  
Updating ea1b82a..05e9557  
(Summary of changes)  
  
$ git branch -d feature/myfetaure  
Deleted branch feature/myfetaure (was 05e9557).  
  
$ git push origin develop
```

Ramas release

Como las ramas feature, las ramas release también tienen que surgir de la rama develop. Contienen el código de la versión que se va a liberar próximamente. Es un paso previo y preparatorio para la versión definitiva de producción. En ella se incluye todo el código de develop necesario para el lanzamiento. Puede que contenga algún error pequeño que se debe de arreglar en este momento para no incluirlo en producción. Una vez finalizada la rama, esta se debe incluir tanto en la rama develop como en la rama master.

Convención de nombres: deben de seguir la siguiente convención: release-*, sustituyendo el * por el número de versión (1.1, 2.3, 4.7, etc)

Crear rama reléase

```
$ git checkout -b release-1.2 develop
```

```
Switched to a new branch "release-1.2"
```

(Hacer las modificaciones necesarias)

```
$ git commit -a -m "Release version 1.2 of the project"
```

```
[release-1.2 74d9424] Release version 1.2 of the project
```

```
1 files changed, 1 insertions(+), 1 deletions(-)
```

Finalizar una rama release

Primero debemos actualizar la rama master.

```
$ git checkout master
```

```
Switched to branch 'master'
```

```
$ git merge --no-ff release-1.2
```

```
Merge made by recursive.
```

(Summary of changes)

```
$ git tag -a 1.2
```

A continuación, debemos guardar esos cambios en la rama develop.

```
$ git checkout develop
```

```
Switched to branch 'develop'
```

```
$ git merge --no-ff release-1.2
```

```
Merge made by recursive.
```

```
(Summary of changes)
```

Una vez integrada la rama tanto en master como en develop eliminaremos la rama en el repositorio local.

```
$ git branch -d release-1.2
```

```
Deleted branch release-1.2 (was ff452fe).
```

Ramas hotfix

Estas ramas surgen de la rama master. Contienen una versión de producción con un error que se desea arreglar urgentemente. Una vez arreglado el error, se incluye el contenido de esta rama en las ramas master y develop para subsanar el error. Además, hay que marcar la versión arreglada de producción con un tag en la rama master.

Convención de nombres: deben de seguir la siguiente convención: hotfix-*, sustituyendo el * por el número de la revisión (1.1.5, 2.3.1, 4.7.9, etc)

Crear una rama hotfix

```
$ git checkout -b hotfix-1.2.1 master
```

```
Switched to a new branch "hotfix-1.2.1"
```

(Hacer las modificaciones necesarias)

```
$ git commit -a -m "Bumped version number to 1.2.1"
```

```
[hotfix-1.2.1 41e61bb] Bumped version number to 1.2.1
```

```
1 files changed, 1 insertions(+), 1 deletions(-)
```

Finalizar una rama hotfix

Primero debemos actualizar la rama master y etiquetarla.

```
$ git checkout master
```

```
Switched to branch 'master'
```

```
$ git merge --no-ff hotfix-1.2.1
```

```
Merge made by recursive.
```

(Summary of changes)

```
$ git tag -a 1.2.1
```

A continuación, debemos incluir el hotfix en develop también.

```
$ git checkout develop
```

```
Switched to branch 'develop'
```

```
$ git merge --no-ff hotfix-1.2.1
```

```
Merge made by recursive.
```

```
(Summary of changes)
```

Una vez integrada la rama tanto en master como en develop eliminaremos la rama en el repositorio local.

```
$ git branch -d hotfix-1.2.1
```

```
Deleted branch hotfix-1.2.1 (was abbe5d6).
```


Herramienta git-flow

No es necesario ejecutar manualmente cada uno de los comandos expuestos anteriormente para administrar las ramas. Tenemos a nuestra disposición una herramienta de línea de comandos que nos ayudará en este proceso, ya que se encarga de realizar todos los pasos intermedios necesarios para gestionar las ramas. Os dejamos la documentación online del autor de Git flow donde explica cómo instalar la herramienta para los distintos sistemas operativos.

Cómo utilizar la herramienta

Para comenzar a utilizar la metodología Git flow, debemos iniciarla dentro de un repositorio git existente. Para ello ejecutaremos el comando:

```
$ git flow init
```

Con ello nos iniciará un proceso guiado con preguntas relacionadas con las convenciones de nombres para las ramas. Para utilizar la convención que hemos descrito arriba, debemos dejar los valores por defecto, sin introducir nada.

```
PS D:\PC\Documents\Codeledge\jQuery\DVD> git flow init
Initialized empty Git repository in D:/PC/Documents/Codeledge/JQuery/DVD/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/PC/Documents/Codeledge/JQuery/DVD/.git/hooks]
PS D:\PC\Documents\Codeledge\jQuery\DVD> █
```

Gestionar features

Crear una nueva feature (en el ejemplo my-feature):

```
$ git flow feature start feature/my-feature
```

```
PS D:\PC\Documents\Codelledge\JQuery\DVD> git flow feature start feature/my-feature
Switched to a new branch 'feature/feature/my-feature'

Summary of actions:
- A new branch 'feature/feature/my-feature' was created, based on 'develop'
- You are now on branch 'feature/feature/my-feature'

Now, start committing on your feature. When done, use:

    git flow feature finish feature/my-feature

PS D:\PC\Documents\Codelledge\JQuery\DVD> █
```

```
PS D:\PC\Documents\Codelledge\JQuery\DVD> git add movimiento.png
PS D:\PC\Documents\Codelledge\JQuery\DVD> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   assets/css/styleSheet.css
    new file:   assets/js/script.js
    new file:   index.html
    new file:   movimiento.png
```

```
PS D:\PC\Documents\Codelledge\JQuery\DVD> git push -u origin main
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (11/11), 6.17 KiB | 1.54 MiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:initialshoot/gitflow.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS D:\PC\Documents\Codelledge\JQuery\DVD> █
```

```
PS D:\PC\Documents\Code\lledge\jQuery\DVD> git checkout feature/feature/my-feature
Switched to branch 'feature/feature/my-feature'
PS D:\PC\Documents\Code\lledge\jQuery\DVD> git add assets\img\movimiento.png
PS D:\PC\Documents\Code\lledge\jQuery\DVD> git commit -m "change the img file to a new folder inside assets"
[feature/feature/my-feature e9403bd] change the img file to a new folder inside assets
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 assets/img/movimiento.png
```

Finalizar la rama feature:

```
$ git flow feature finish feature/my-feature
```


```
PS D:\PC\Documents\Code\lledge\jQuery\DVD> git flow feature finish feature/my-feature
Switched to branch 'develop'
Updating cdb01ad..e9403bd
Fast-forward
 assets/img/movimiento.png | Bin 0 -> 3520 bytes
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 assets/img/movimiento.png
Deleted branch feature/feature/my-feature (was e9403bd).

Summary of actions:
- The feature branch 'feature/feature/my-feature' was merged into 'develop'
- Feature branch 'feature/feature/my-feature' has been locally deleted
- You are now on branch 'develop'


PS D:\PC\Documents\Code\lledge\jQuery\DVD> █
```

Publicar la rama feature en el repositorio remoto:

```
$ git flow feature publish feature/my-feature
```

 feature/feature/my-feature had recent pushes less than a minute ago
 Compare & pull request

main ▾
1 branch
0 tags
Go to file
Add file ▾
Code ▾


initialshoot add main files of the web
e32aafb 9 minutes ago 2 commits

assets	add main files of the web	9 minutes ago
index.html	add main files of the web	9 minutes ago
movimiento.png	add main files of the web	9 minutes ago

Help people interested in this repository understand your project by adding a README.

Add a README

```

PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow feature publish feature/my-feature
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 3.78 KiB | 1.26 MiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature/feature/my-feature' on GitHub by visiting:
remote:   https://github.com/initialshoot/gitflow/pull/new/feature/feature/my-feature
remote:
To github.com:initialshoot/gitflow.git
* [new branch]      feature/feature/my-feature -> feature/feature/my-feature
branch 'feature/feature/my-feature' set up to track 'origin/feature/feature/my-feature'.
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Already on 'feature/feature/my-feature'
Your branch is up to date with 'origin/feature/feature/my-feature'.

Summary of actions:
- The remote branch 'feature/feature/my-feature' was created or updated
- The local branch 'feature/feature/my-feature' was configured to track the remote branch
- You are now on branch 'feature/feature/my-feature'

PS D:\PC\Documents\Codelledge\jQuery\DVD> █

```

Obtener una rama feature del repositorio remoto:

\$ git flow feature pull origin feature/my-feature

```

PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow feature pull origin feature/my-feature
The command 'git flow feature pull' will be deprecated per version 2.0.0. Use 'git flow feature track' instead.
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Pulled origin's changes into feature/feature/my-feature.
PS D:\PC\Documents\Codelledge\jQuery\DVD> █

```

Seguir de los cambios de la feature:

\$ git flow feature track feature/my-feature

```

PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow feature track feature/my-feature
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
fatal: 'origin/feature/feature/my-feature' is not a commit and a branch 'feature/feature/my-feature' cannot be created from it
Fatal: Could not create 'feature/feature/my-feature'.
PS D:\PC\Documents\Codelledge\jQuery\DVD> █

```

Gestionar releases

Comenzar una release (en el ejemplo release-1.2):

```
$ git flow release start release-1.2
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow release start release-1.2  
Switched to a new branch 'release/release-1.2'
```

Summary of actions:

- A new branch 'release/release-1.2' was created, based on 'develop'
- You are now on branch 'release/release-1.2'

Follow-up actions:

- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

```
git flow release finish 'release-1.2'
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> █
```

Concluir una release:

```
$ git flow release finish release-1.2
```

```
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
```

```
To github.com:initialshoot/gitflow.git
```

```
- [deleted]          release/release-1.2
```

```
Deleted branch release/release-1.2 (was e9403bd).
```

Summary of actions:

- Release branch 'release/release-1.2' has been merged into 'main'
- The release was tagged 'release-1.2'
- Release tag 'release-1.2' has been back-merged into 'develop'
- Release branch 'release/release-1.2' has been locally deleted; it has been remotely deleted from 'origin'
- You are now on branch 'develop'

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> █
```

Publicar la release en el repositorio remoto:

```
$ git flow release publish release-1.2
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow release publish release-1.2
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 3.78 KiB | 1.26 MiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'release/release-1.2' on GitHub by visiting:
remote:   https://github.com/initialshoot/gitflow/pull/new/release/release-1.2
remote:
To github.com:initialshoot/gitflow.git
* [new branch]      release/release-1.2 -> release/release-1.2
branch 'release/release-1.2' set up to track 'origin/release/release-1.2'.
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Already on 'release/release-1.2'
Your branch is up to date with 'origin/release/release-1.2'.

Summary of actions:
- The remote branch 'release/release-1.2' was created or updated
- The local branch 'release/release-1.2' was configured to track the remote branch
- You are now on branch 'release/release-1.2'

PS D:\PC\Documents\Codelledge\jQuery\DVD>
```

Debemos, también, publicar los tags en el repositorio remoto:

```
$ git push --tags
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git push --tags
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Everything up-to-date
PS D:\PC\Documents\Codelledge\jQuery\DVD>
```

Seguir los cambios de la release:

```
$ git flow release track release-1.2
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow release track release-1.2
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
fatal: 'origin/release/release-1.2' is not a commit and a branch 'release/release-1.2' cannot be created from it
Fatal: Could not create branch 'release/release-1.2'.
PS D:\PC\Documents\Codelledge\jQuery\DVD>
```

Gestionar hotfixes

Crear un hotfix (en este ejemplo hotfix-1.2.1):

```
$ git flow hotfix start hotfix-1.2.1
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow hotfix start hotfix-1.2.1
Branches 'main' and 'origin/main' have diverged.
And local branch 'main' is ahead of 'origin/main'.
Switched to a new branch 'hotfix/hotfix-1.2.1'
```

Summary of actions:

- A new branch 'hotfix/hotfix-1.2.1' was created, based on 'main'
- You are now on branch 'hotfix/hotfix-1.2.1'

Follow-up actions:

- Start committing your hot fixes
- Bump the version number now!
- When done, run:

```
git flow hotfix finish 'hotfix-1.2.1'
```

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> █
```

Publicar el hotfix en el repositorio remoto:

```
PS D:\PC\Documents\Codelledge\jQuery\DVD> git flow hotfix publish hotfix-1.2.1
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (14/14), 1.41 KiB | 239.00 KiB/s, done.
Total 14 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
remote:
remote: Create a pull request for 'hotfix/hotfix-1.2.1' on GitHub by visiting:
remote:   https://github.com/initialshoot/gitflow/pull/new/hotfix/hotfix-1.2.1
remote:
To github.com:initialshoot/gitflow.git
* [new branch]      hotfix/hotfix-1.2.1 -> hotfix/hotfix-1.2.1
branch 'hotfix/hotfix-1.2.1' set up to track 'origin/hotfix/hotfix-1.2.1'.
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519': █
```

Conclure un hotfix:

```
$ git flow hotfix finish hotfix-1.2.1
```

```
Switched to branch 'develop'
Merge made by the 'ort' strategy.
 assets/js/script.js | 2 +-
 movimiento.png      | Bin 3520 -> 0 bytes
 2 files changed, 1 insertion(+), 1 deletion(-)
 delete mode 100644 movimiento.png
Enter passphrase for key '/c/Users/initi/.ssh/id_ed25519':
To github.com:initialshoot/gitflow.git
- [deleted]          hotfix/hotfix-1.2.1
Deleted branch hotfix/hotfix-1.2.1 (was 76c1d17).

Summary of actions:
- Hotfix branch 'hotfix/hotfix-1.2.1' has been merged into 'main'
- The hotfix was tagged 'hotfix-1.2.1'
- Hotfix tag 'hotfix-1.2.1' has been back-merged into 'develop'
- Hotfix branch 'hotfix/hotfix-1.2.1' has been locally deleted; it has been remotely deleted from 'origin'
- You are now on branch 'develop'

PS D:\PC\Documents\Code\lledge\jQuery\DVD> |
```