

Introducción a Git y GitHub



Que es Git



GIT es un sistema de control de versiones:

- Guarda un historial de los cambios
- Permite abordar desarrollos colaborativos
- Permite saber que cambios se han hecho, cuando, y quién los realizó
- Permite revertir los cambios y volver a un estado previo
- Permite el control de versiones distribuido
- Los usuarios pueden trabajar en local, y mantener toda la historia de cambios.

¿Qué es GitHub?



- GitHub es el mayor servicio de hosting de repositorios git basado en web
 - Es decir se utiliza como servidor “remote” en muchos proyectos, y permite la colaboración online
- Añade funcionalidad encima de git
 - UI, documentación, bug tracking, feature requests, pull requests, etc.
- Además github se ha convertido en uno de los lugares en los que los developers muestran su trabajo.
 - Github permite construir un portfolio online

Conceptos básicos de Git

Snapshots

- Son el modo en el que Git guarda el histórico del código/versiones
- Fundamentalmente graba/almacena las características de los ficheros en un momento temporal
- El usuario decide cuando se realiza ese “snapshot” y sobre que ficheros
- Y estos snapshots se pueden “recorrer” para ver versiones anteriores, recuperarlas, etc

Commit

- Commit se usa como nombre y como verbo
 - Como verbo es el acto de crear un snapshot
 - Como nombre describe el conjunto de elementos que forman un snapshot
- Un proyecto es esencialmente una serie de commits
- Los commits contienen 3 piezas de información
 - Information about how the files changed from previously
 - A reference to the commit that came before it Called the “parent commit”
 - A [hash code](#) name .Will look something like: fb2d2ec5069fc6776c80b3ad6b7cbde3cade4e

Repositorios

A menudo llamados repos son una colección de todos los ficheros, y de la historia de los mismos. Esta formado por todos los commits, y es el lugar donde está almacenado en el disco

El repositorio puede ser local y estar en nuestra máquina, o puede estar en un servidor/servicio remoto como GitHub o BitBucket.

- Cuando copiamos un repositorio completo desde un servidor remoto estamos **clonando** el repositorio
- Cuando actualizamos con los commits que no tenemos en local, desde el servidor remoto estamos haciendo una operación de **pulling**
- Cuando añadimos cambios que hemos realizado en local al servidor remoto estamos haciendo **pushing**

Branches

Los branches o ramas, son las áreas en las que se sitúan todos los commits. La rama principal del proyecto se denomina **master**.

Las ramas sirven para que los desarrolladores trabajen en partes del proyecto o del código de manera separada, y una vez que han realizado el desarrollo lo consolidan con la rama principal, en una operación denominada **merge**

Usando GIT

Conceptos básicos de Git

Instalación de Git

Mac: instalar desde <http://sourceforge.net/projects/git-osx-installer/>

Windows: <http://msysgit.github.com/>

Conceptos básicos de Git

Configure your user

```
$ git config --global user.name "tunombre"  
$ git config --global user.email tumail@dominio.es
```

Initialize an empty repository

```
$ git init  
Initialized empty Git repository in /home/diegodl/git-tutorial/.git/
```

Clone a remote repository

```
$ git clone <repo> [<directory>]
```

Check the status of your repository

```
(base) DiegoMac:DS-Bootcamp ddl$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

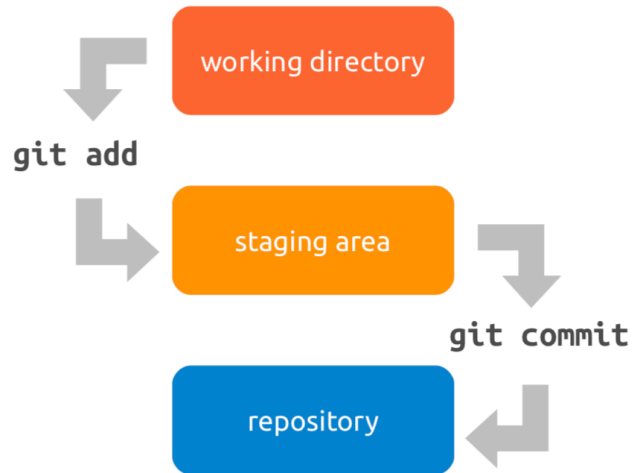
        modified:   .DS_Store
        modified:   Semana1/2.1.WarmUp_Introduccion_a_Python.ipynb
        modified:   Semana1/2.2.WarmUp_Secuencias_en_python.ipynb
        modified:   Semana1/3.1.WorkOutFunciones.ipynb
        modified:   Semana1/3.1.warmUp_Funciones.ipynb
        modified:   Semana1/3.2.WarmUpDecisionStructures.ipynb
        modified:   Semana1/3.2.WorkOutDecisionStructures.ipynb
        deleted:    Semana1/4.1.WarmUpColecciones.ipynb
        deleted:    Semana1/4.1.WorkOutColecciones.ipynb
        deleted:    Semana1/4.2.WarmUpProgramacion_orientada_objetos.ipynb
        deleted:    Semana1/4.2.WorkOutPOO.ipynb
        deleted:    Semana1/5.1.UnbrokenSpaceInvaders.ipynb
        modified:   Semana2/W2-4.1.Warmup_SQL.ipynb

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

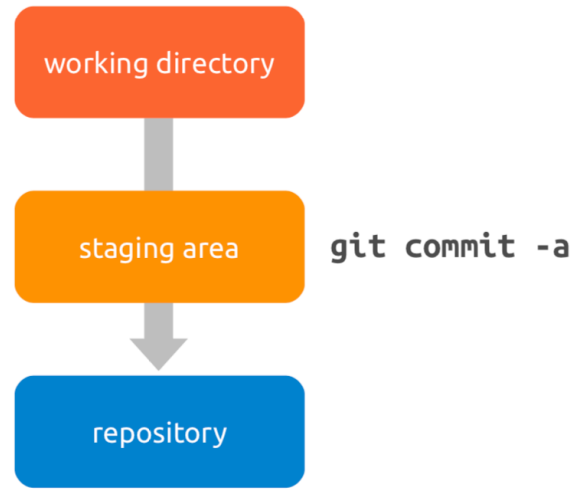
        .ipynb_checkpoints/
        Semana1/.DS_Store
        Semana1/.ipynb_checkpoints/2.1.WarmUp_Introduccion_a_Python-checkpoint.ipynb
        "Semana1/.ipynb_checkpoints/2.1.Workout_Introducci\303\263n_a_Python-checkpoint.ipynb"
        Semana1/.ipynb_checkpoints/2.2.WarmUp_Secuencias_en_python-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/2.2.WorkOutSecuencias-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/3.1.WorkOutFunciones-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/3.2.WarmUpDecisionStructures-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/3.2.WorkOutDecisionStructures-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/4.1.WarmUpColecciones-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/4.2.WarmUpProgramacion_orientada_objetos-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/4.2.WorkOutPOO-checkpoint.ipynb
        Semana1/.ipynb_checkpoints/5.1.UnbrokenSpaceInvaders-checkpoint.ipynb
```

Git Stages

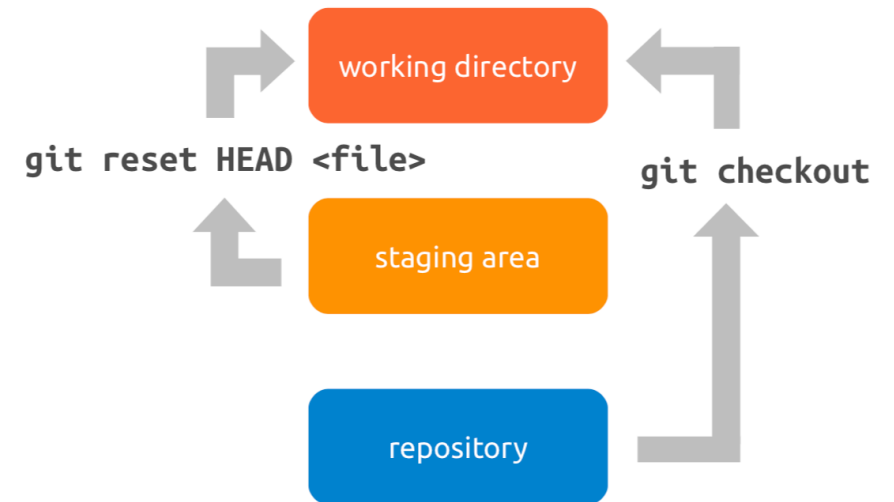
El repositorio local esta compuesto por tres "árboles" administrados por git. El primero es tu **Directorio de trabajo** que contiene los archivos, el segundo es el **Index/staging area** que actua como una zona intermedia, y el último es el **HEAD** que apunta al último commit realizado.



Al incluir un fichero en el directorio o cambiarlo, Git nos advertirá, pero tendremos que realizar las acciones de "add" y "commit" para incluirlo

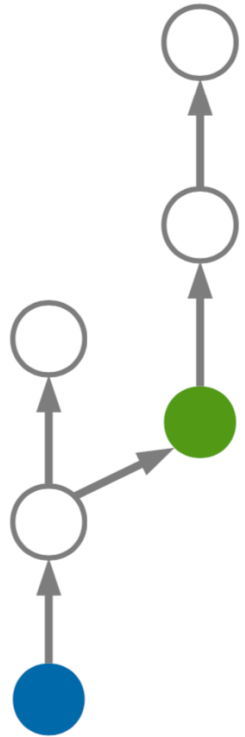


También puede no hacerse el paso intermedio



Con reset HEAD, eliminamos el fichero del index/staging. Con checkout estamos haciendo un cambio y eliminando el commit del HEAD

Git Branches



List branches

```
$ git branch  
develop  
* master  
new-feature
```

Change into a branch

```
$ git checkout develop  
Switched to branch 'develop'
```

Manage branches

Create a new branch from the one you have currently checked out

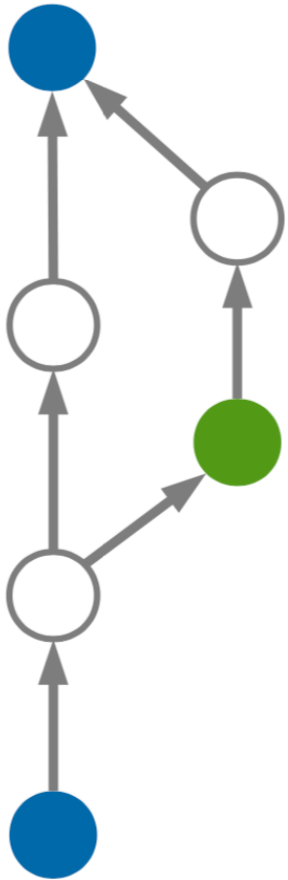
```
$ git branch <branch>
```

Rename a branch

```
$ git branch -m <oldbranch> <newbranch>
```

Delete a branch

```
$ git branch -D <branch>
```



Merge branches

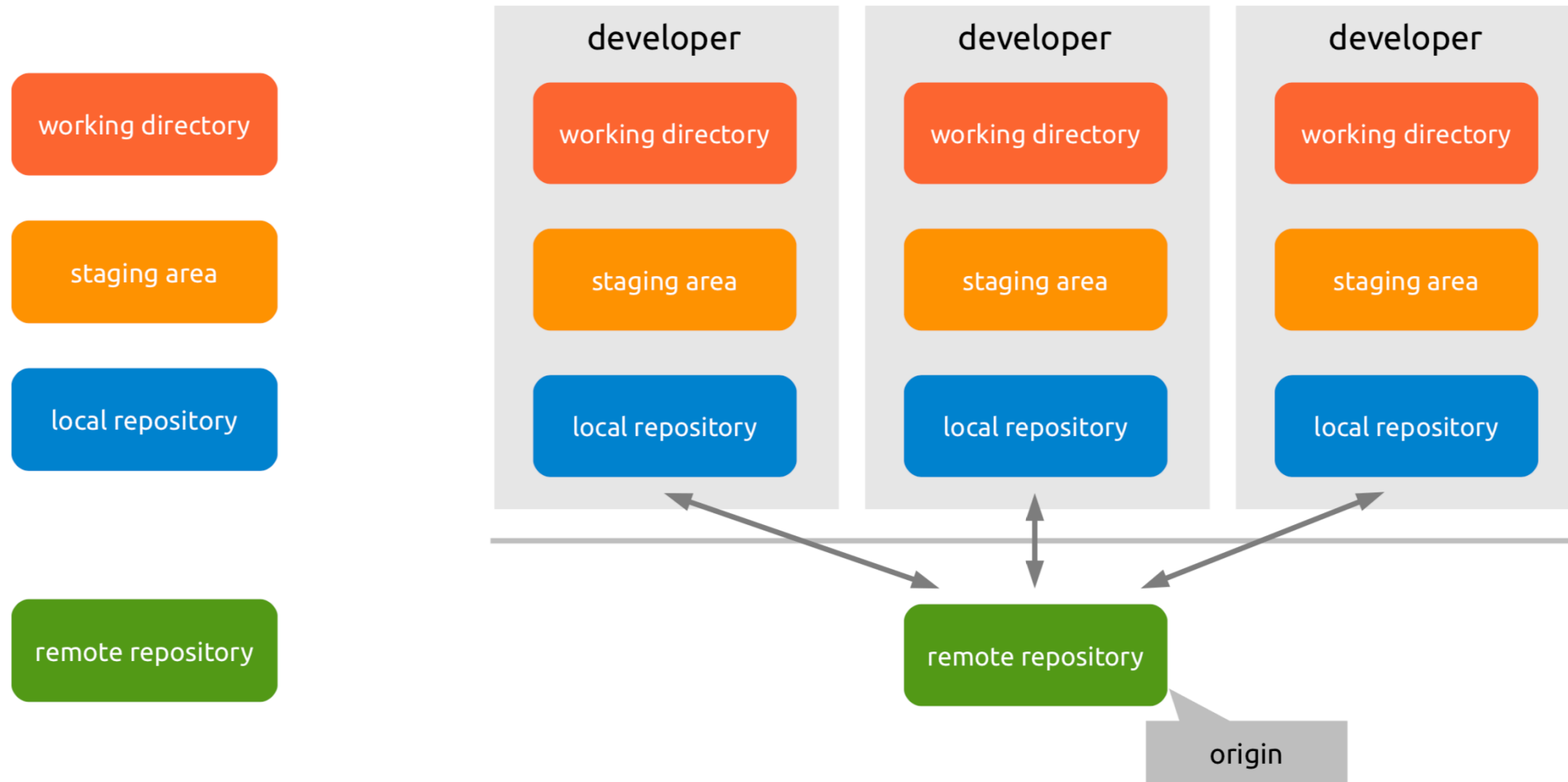
Merge the specified branch into the current branch (the one you have checked out)

\$ git checkout master

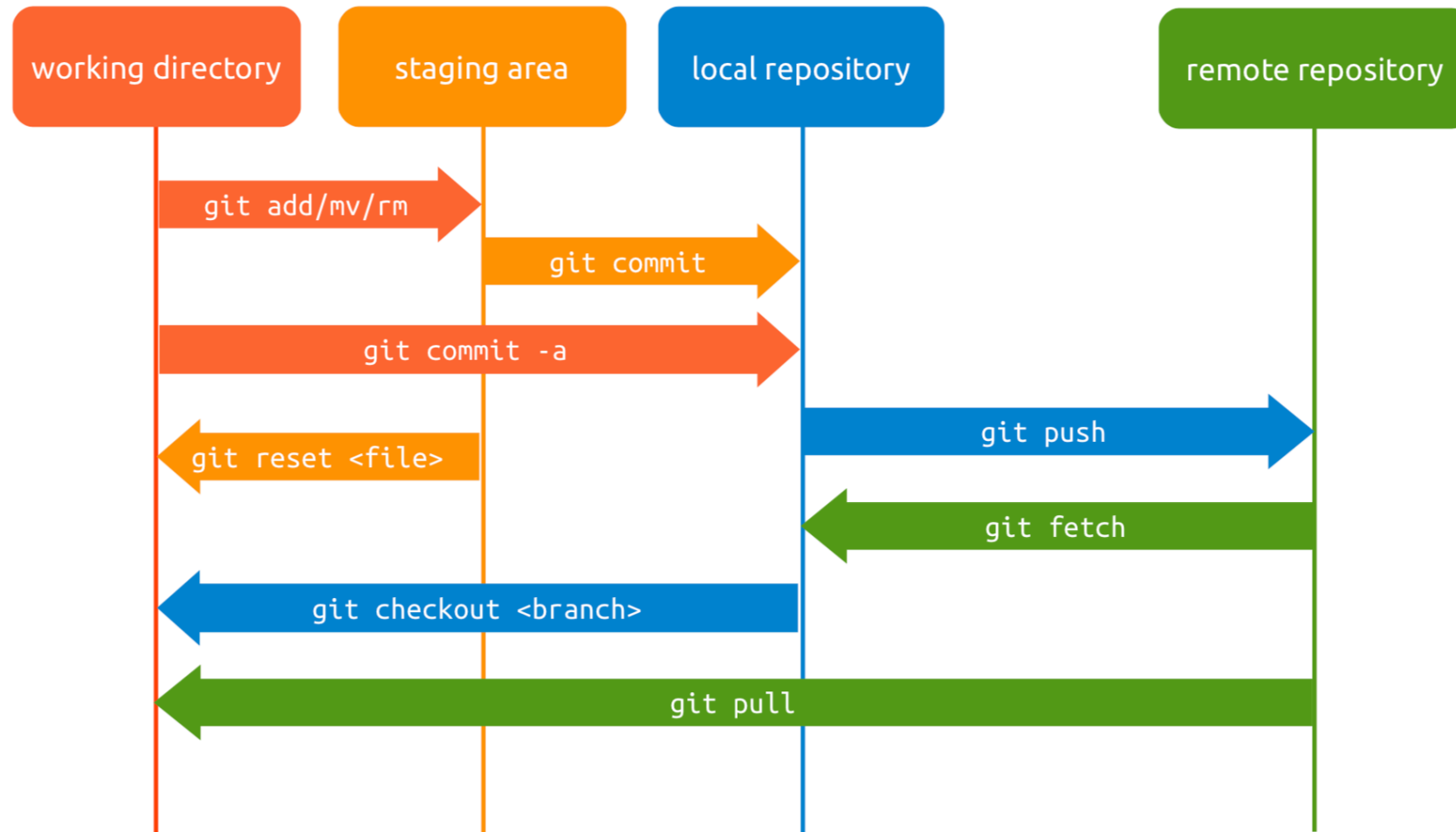
\$ git merge topic

Merges topic
into master

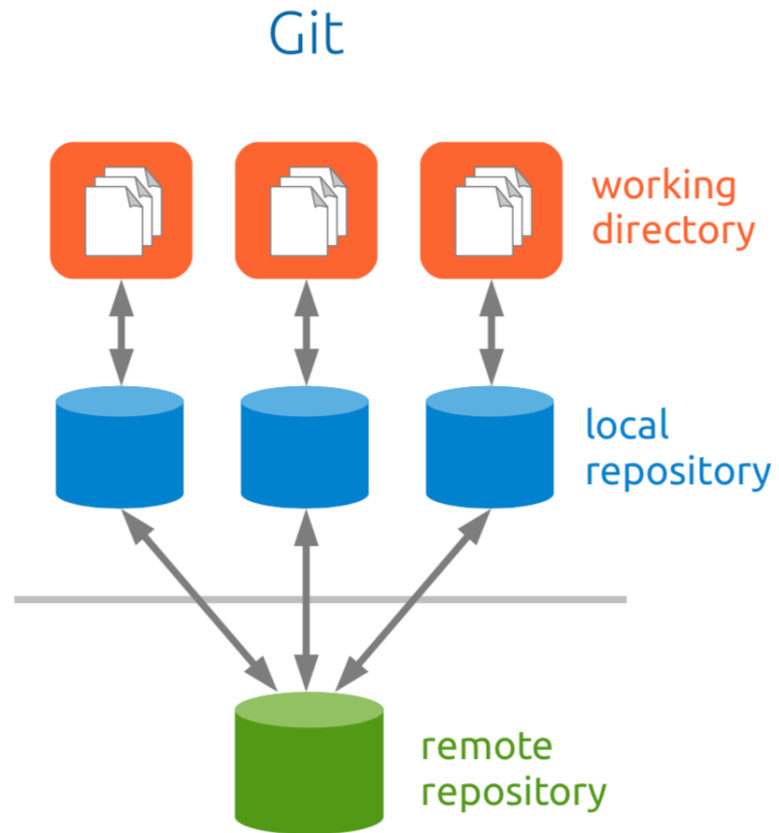
Git Remotes



Git Remotes. Esquema de funcionamiento

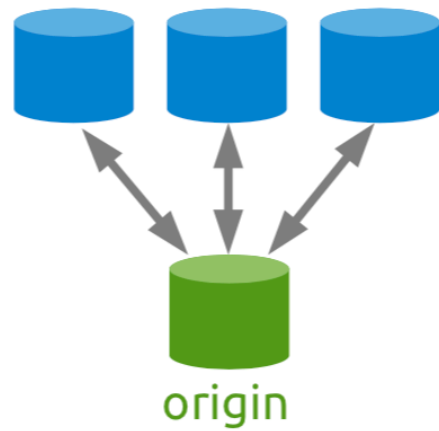


Modelo de distribución/colaboración

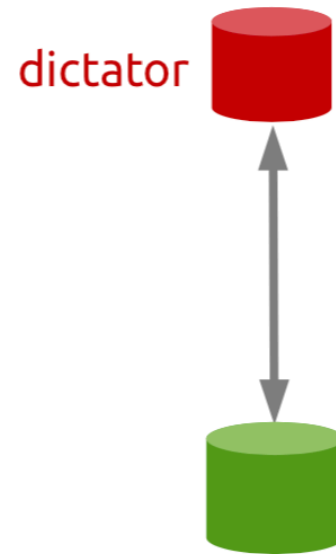


Modelo de distribución/colaboración

Centralized workflow

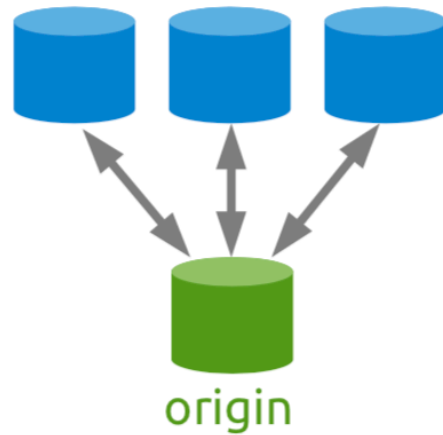


GitHub Forking/Pull Request

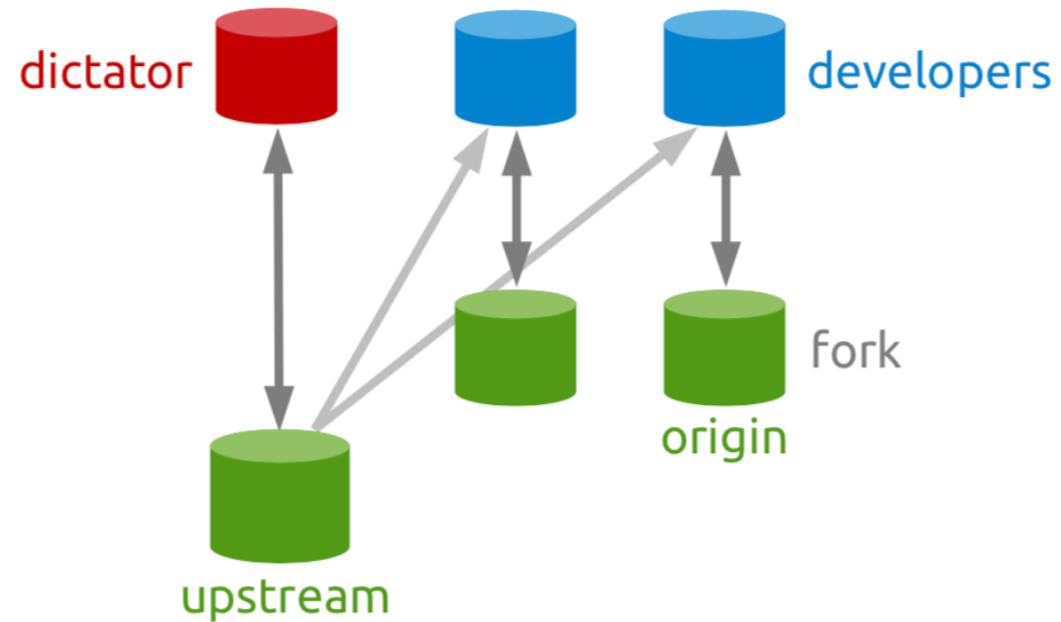


Modelo de distribución

Centralized workflow

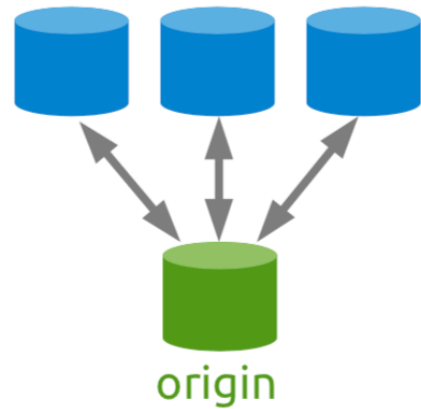


GitHub Forking/Pull Request



Modelo de distribución

Centralized workflow



GitHub Forking/Pull Request

