



# Git 101

# ¿Qué es git?

- Es un **sistema de control de versiones (VCS)** utilizado para seguir cambios en archivos y coordinar el trabajo en dichos archivos entre varias personas.
- Se usa principalmente para el **manejo y respaldo del código fuente**, pero puede ser utilizado para seguir cambios en **cualquier conjunto de archivos**.

# ¿Qué gano con usar git?

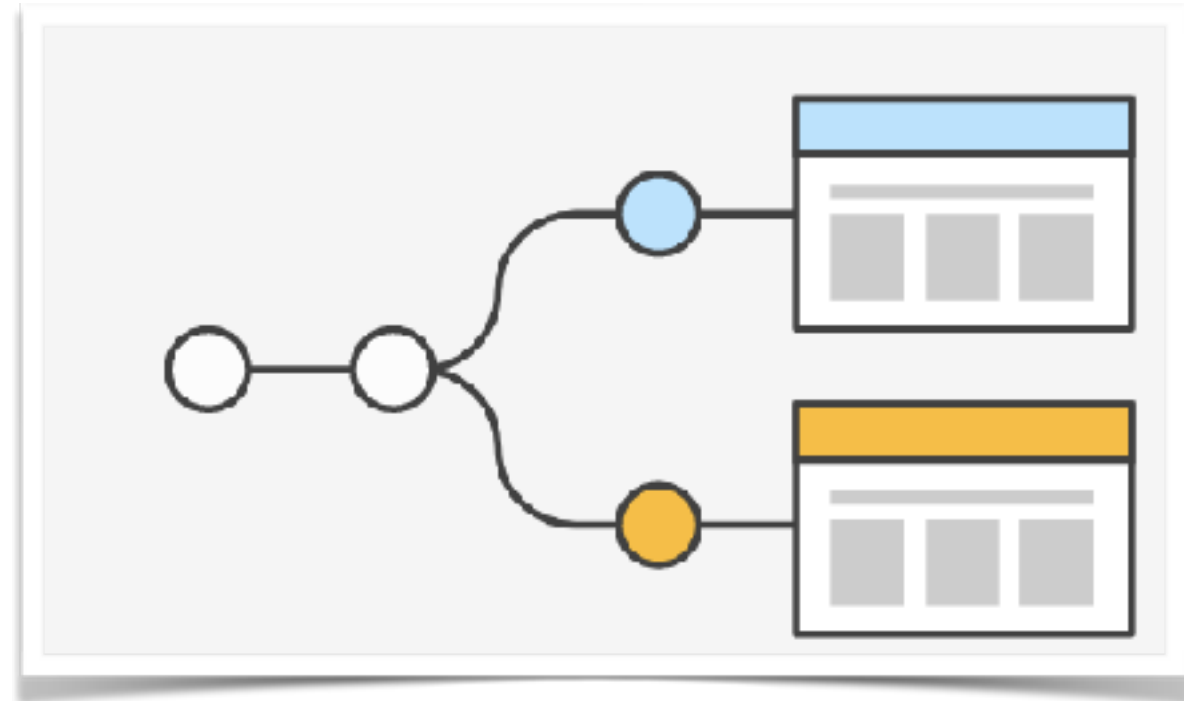
## *Developers*

- ***Desarrollo paralelo de features:*** Un desarrollador puede estar desarrollando un modulo mientras que otra persona esta desarrollando otro, o inclusive el mismo modulo, sin estarse pisando los talones
- ***Pruebas rápida de conceptos sin alterar el código fuente:*** En cuestión de segundos puedes empezar a desarrollar algo con motivo de prueba y descartarlo, sin alterar el código fuente original
- ***Manejo sencillo de versiones:*** En aplicaciones móviles por ejemplo, puedes etiquetar el código de una versión y cuando tengas que subir fixes a la App/Play Store, solo es cuestión de alterar esa version, sin preocuparte por el código nuevo que se ha escrito


# ¿Qué gano con usar git?

## *Diseñadores*

- ***Fácil prototipado:*** Cuando se quiera modificar alguna cosa de producción como reemplazar iconos o cambiar alguna ventana, puedes cambiar de prototipos fácilmente. De esta manera, cualquiera del equipo puede ver como esta quedando el diseño de algo sin tener que acudir personalmente con el desarrollador en cargo para ver la nueva interfaz.



# Requisitos

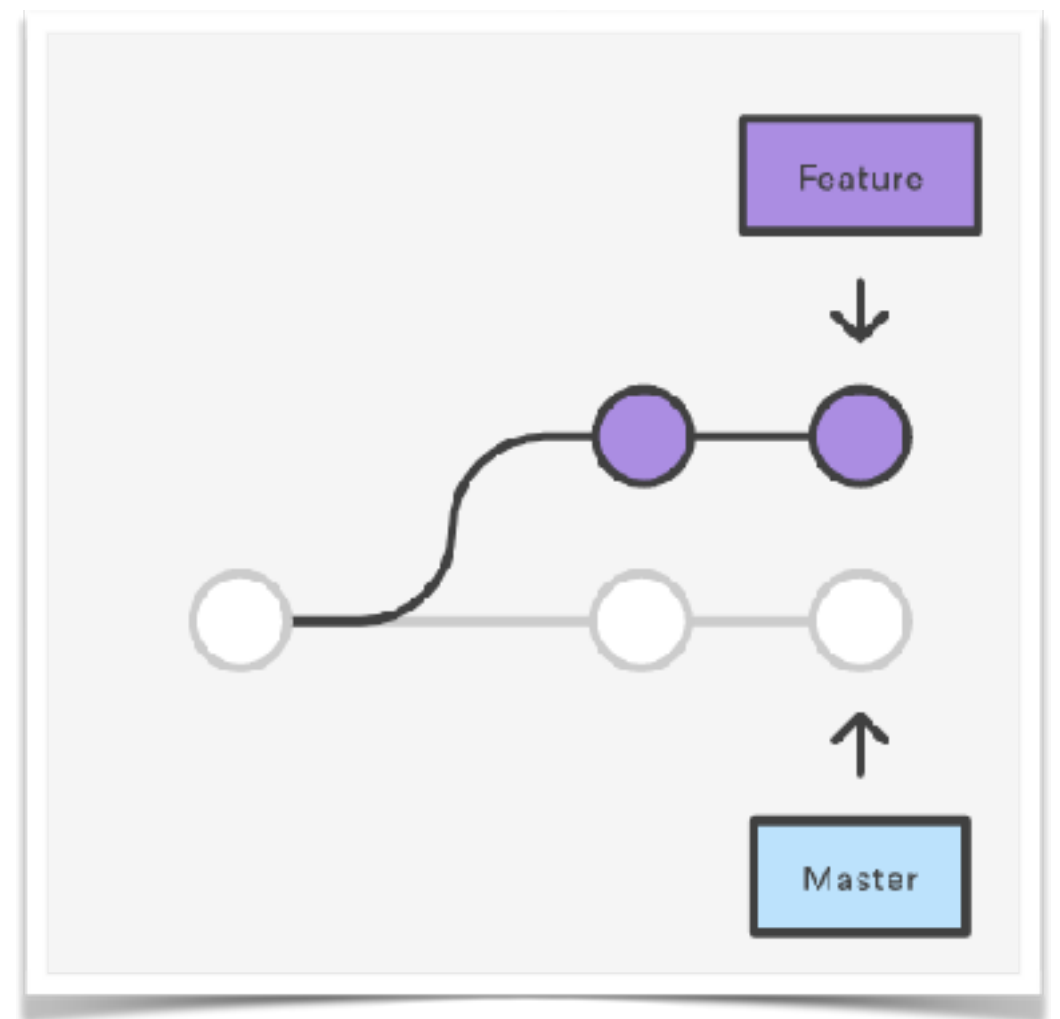
- Computadora con Windows / Unix
- Git instalado (EN LA TERMINAL)
- Ganas de aprender 

# Configuración de Usuario de Git para tu PC

- \$ **git** config —global user.name <nombre>
- \$ **git** config —global user.email <email>

# Creando un repositorio

- Crear una carpeta para el repositorio
- \$ **git** init
- Este comando inicializa un repositorio git y una rama **master**.
- Solo tiene que correrse una vez
- Crear .gitignore



# Creando un repositorio

- .gitignore ejemplo laravel

```
1 vendor/
2 node_modules/
3 npm-debug.log
4
5 # Laravel 4 specific
6 bootstrap/compiled.php
7 app/storage/
8
9 # Laravel 5 & Lumen specific
10 public/storage
11 public/hot
12 storage/*.key
13 .env/*.php
14 .env.php
15 .env
16 Homestead.yaml
17 Homestead.json
18
19 # Rocketeer PHP task runner and deployment package. https://github.com/rocketeers/rocketeer
20 .rocketeer/
```



# Guardando cambios

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)

# Guardando cambios

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)
2. \$ **git** status

# Guardando cambios

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)
2. \$ **git** status
3. \$ **git** add <nombredearchivo>

# Guardando cambios

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)
2. \$ **git** status
3. \$ **git** add <nombredearchivo>
4. \$ **git** status

# Guardando cambios

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)
2. \$ **git** status
3. \$ **git** add <nombredearchivo>
4. \$ **git** status
5. \$ **git** commit -m "Texto del commit"

# **Subir cambios (repositorio remoto no agregado)**

- Crear un repositorio remoto (Github, gitlab, bitbucket, etc)

# Subir cambios

## (repositorio remoto no agregado)

- Crear un repositorio remoto (Github, gitlab, bitbucket, etc)
- \$ **git** remote add <nombre del repo remoto>  
<url\_repo\_remoto>

# Subir cambios (repositorio remoto no agregado)

- Crear un repositorio remoto (Github, gitlab, bitbucket, etc)
- \$ **git remote add** <nombre repo remoto> <url repo remoto>
- \$ **git push** -u <nombre del repo remoto> <nombre de la rama local>



# Subir cambios

## (repositorio remoto no agregado)

- Crear un repositorio remoto (Github, gitlab, bitbucket, etc)
- \$ **git remote add** <nombre repo remoto> <url repo remoto>
- \$ **git push** -u <nombre del repo remoto> <nombre de la rama local>
- Checar GitHub

# Subir cambios (repositorio remoto si agregado)

1. Creamos un archivo con el contenido que deseemos (código, texto, imagen, etc.)
2. \$ **git** status
3. \$ **git** add <nombredearchivo>
4. \$ **git** status
5. \$ **git** commit -m "Texto del commit"

# Subir cambios

## (repositorio remoto si agregado)

- \$ **git push** -u <nombre del repo remoto> <nombre de la rama local>
- \$ git push origin master

# Subir cambios (repositorio remoto si agregado)

- \$ **git push** -u <nombre del repo remoto> <nombre de la rama local>
- Checar GitHub

# Trabajando en un repositorio

- \$ **git** clone <url de repo remoto>

# Trabajando en un repositorio

- \$ **git** clone <url de repo remoto>
- Todo lo demás aplica igual

# Crear rama

- \$ **git** checkout -b <nombre de rama>
- Al usar el -b, la rama creada es una copia de la rama actual

# Cambiar de rama

- \$ **git** checkout <nombre de rama>
- No deben de existir cambios en archivos compartidos para permitir el cambio libre entre ramas



# Escenario de colaboración real

- Juanito tiene una proyecto que esta alojado en un repositorio de git

# Escenario de colaboración real

- Juanito tiene una proyecto que esta alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas
- Antes de levantar el proyecto, Juanito crea una rama que va a fungir como ambiente de pruebas, además de master (producción)

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas
- Antes de levantar el proyecto, Juanito crea una rama que va a fungir como ambiente de pruebas, además de master (producción)
- Pancho termina su parte y antes de pasarlo a producción, lo pasan al ambiente de pruebas

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas
- Antes de levantar el proyecto, Juanito crea una rama que va a fungir como ambiente de pruebas, además de master (producción)
- Pancho termina su parte y antes de pasarlo a producción, lo pasan al ambiente de pruebas
- Cuando ven que todo está bien, Pancho hace un *pull request* a Juanito (*que tiene permiso de administrador*)

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas
- Antes de levantar el proyecto, Juanito crea una rama que va a fungir como ambiente de pruebas, además de master (producción)
- Pancho termina su parte y antes de pasarlo a producción, lo pasan al ambiente de pruebas
- Cuando ven que todo está bien, Pancho hace un *pull request* a Juanito (*que tiene permiso de administrador*)
- Resuelven los conflictos

# Escenario de colaboración real

- Juanito tiene un proyecto que está alojado en un repositorio de git
- Para acelerar el desarrollo del proyecto, Juanito va a agregar a Pancho al proyecto, para que ambos puedan colaborar
- Mientras Pancho trabaja en la parte de reportes, Juanito puede seguir trabajando en la parte de cobranzas
- Antes de levantar el proyecto, Juanito crea una rama que va a fungir como ambiente de pruebas, además de master (producción)
- Pancho termina su parte y antes de pasarlo a producción, lo pasan al ambiente de pruebas
- Cuando ven que todo está bien, Pancho hace un *pull request* a Juanito (*que tiene permiso de administrador*)
- Resuelven los conflictos
- Pasan el código pruebas a producción



# Mejores practicas