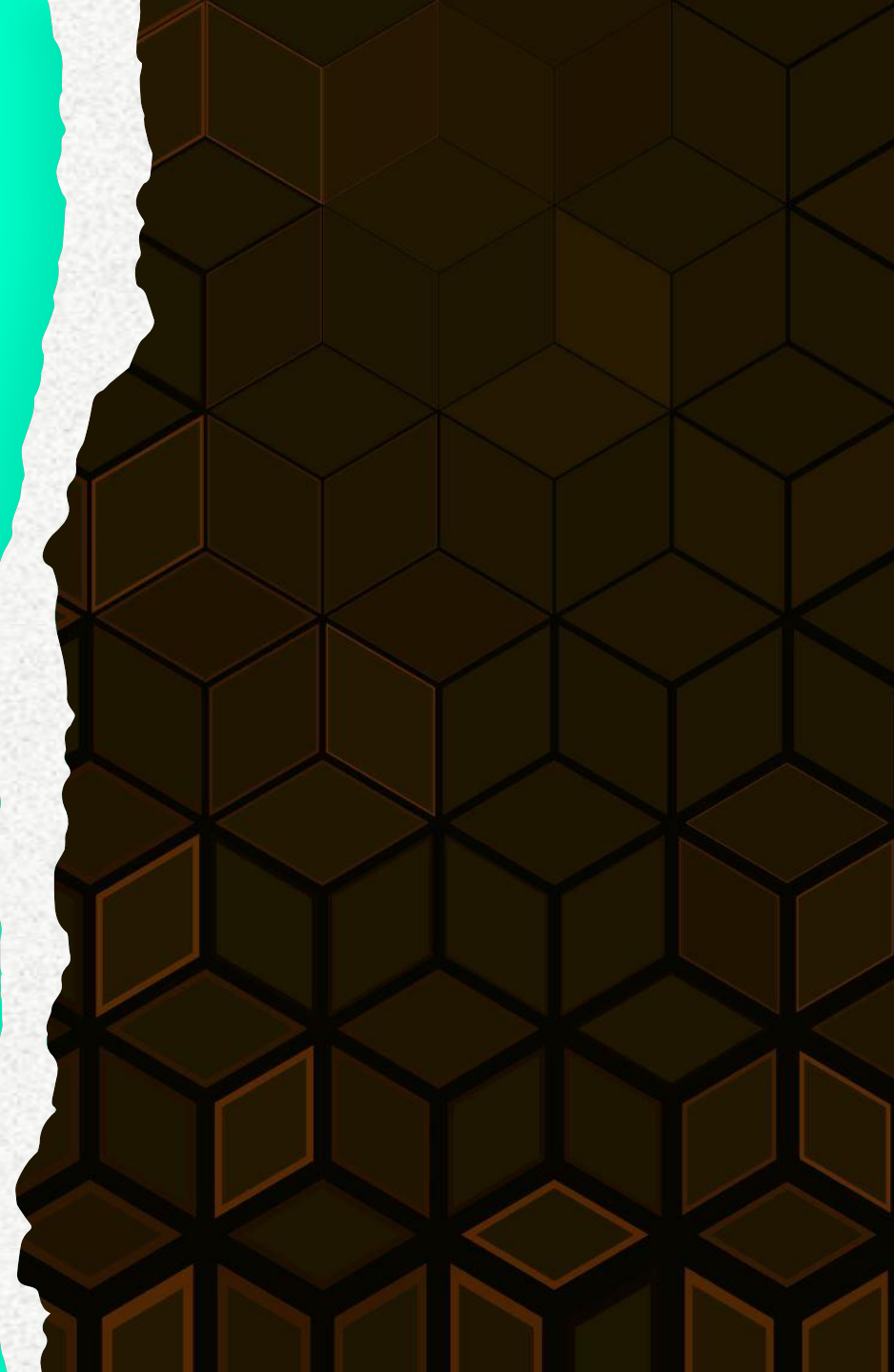


GIT

Repositorios de código



¿QUE ES GIT?

- Es un repositorio de código descentralizado
- Creado por Linus Torval.
- Nos permite descentralizar el control del software
- Cada nodo, tiene una copia completa del repositorio.
- Ejemplos de uso: github, bitbucket...

3 REPOSITORIOS PARA CONTROLARLOS A TODOS

Directorio de trabajo -> nuestra carpeta en la que trabajamos.

Index (stage area)-> Sitio donde añadimos nuestros cambios, espacio intermedio (`git add -A`)

Head -> ultima versión de nuestro repositorio. Donde van los commits. (`git commit ...`)



DEFINIR QUIENES SOMOS EN GIT

Cuando trabajamos en equipo, es interesante identificar quien ha hecho que.

```
git config user.name "mi nombre"  
git config user.email "mi email"
```

EMPEZAR EN GIT

Creando un repositorio nuevo local

- `git init`

Clonando uno existente

- `git clone urlrepo`

TRABAJANDO CON LOS 3 REPOSITORIOS LOCALES

Working dir -> Stagge area

- `git add -A` (sube todos los ficheros)
- Podemos especificar un fichero solamente con `git add nombrefichero`
- Fichero `.gitignore` para aquellos ficheros que no queramos subir a git

Stagge area -> Head

- `Git commit -m "mensaje del commit"`

AÑADIR REPOSITORIO REMOTO

- Si hemos creado un repositorio nuevo con `git init` y queremos conectarlo con un repositorio remoto, hacemos:
 - `git remote add origin [urlrepo]`

ENVIAR MI REPO A UN REPO REMOTO

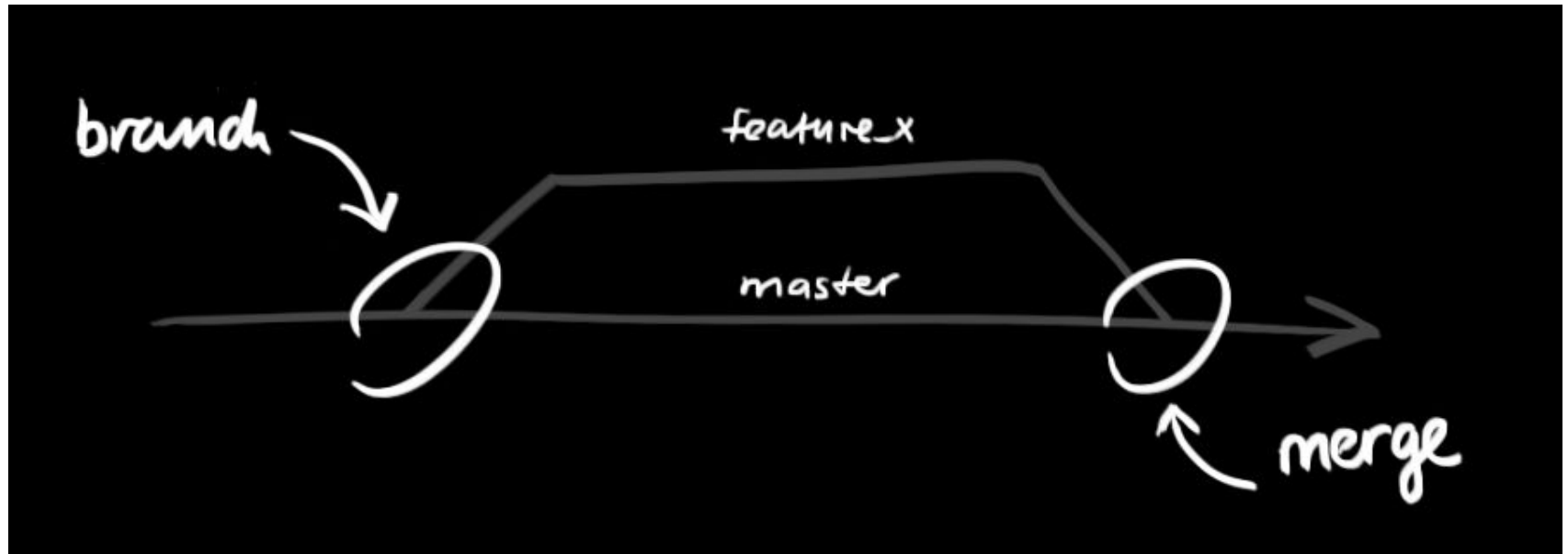
Enviar los cambios

- `git push -u origin master` (la primera vez)
- `git push origin master`

Recibir los cambios

- `git pull -u` (la primera vez)

RAMAS



En git funcionamos con ramas (branch)

De la rama principal (master), pueden salir n ramas, añadir nuevas funcionalidades, y luego fusionarse con la rama master.

Una rama, puede, a su vez, tener n ramas.

¿COMO TRABAJAMOS CON LAS RAMAS?

Para crear una
rama:

- `git branch
feature_x`

Para cambiar de
rama

- `git checkout -b
feature_x`

Para crear una rama
y cambiar a esa
rama:

- `git checkout -b
feature_x`

¿COMO FUSIONAMOS LAS RAMAS?

Vamos a la rama a la que queremos incorporar los cambios.

- `git checkout <branch>`

Fusionamos con la rama que queremos incorporar

- `git merge <branch>`

OTROS COMANDOS INTERESANTES

Para ver el estado de git

```
git status
```

Para ver los commits:

```
git log
```

Para ver las diferencias entre stagge y head

```
git diff
```

PULL REQUEST

Peticiones de cambio.

Se suelen hacer desde github o bitbucket.

Creamos una rama y luego pedimos fusionarla con master.

Ese cambio, suele ir precedido de una aprobación del jefe del proyecto.

Desde bitbucket o github, podemos asignar permisos para eso.

FLUJO DE TRABAJO EN UNA EMPRESA CUALQUIERA

Tenemos 2 ramas, master y dev y dos entornos, un servidor de dev y un servidor de producción.

Sprints de 3 semanas. Solo se sube a producción, al finalizar un sprint.

Cada tarea en jira, implica una rama nueva desde dev en bitbucket.

El desarrollador hace tantos commits como considere.

Cuando acaba la tarea, hace pull request a dev. Esta se aprueba por otro desarrollador.

Al finalizar el sprint, se hace pull request de dev a master. Este pull request debe aprobarlo el jefe de proyectos y otro desarrollador.

Todo pull request debe tener tareas de jira asociado. Todo commit debe tener una tarea de jira asociada.

Antes de hacer una rama, debes sincronizar tu repositorio con el remoto para traerte los cambios de los otros desarrolladores.

Están prohibidos los commits a master y a dev.

¿COMO RECUPERAR UN CAMBIO?

Me he cargado un fichero, quiero recuperar el de la versión del stagge area:

- `git checkout -- <filename>`

Volviendo a una branch anterior

Deshaciendo un commit

- `git reset hashdelcomit (--hard --soft)`
- `git revert hashdelcomit` (crea otro comit que revierte el comit para poder hacer un push)

MI AGENDA

- Proyecto en bitbucket de una app web en PHP con acceso a datos con MySQL donde usamos Docker y bootstrap para el entorno gráfico
- <https://bitbucket.org/entorno-desarrollocesur/miagenda>

In case of fire



1. git commit



2. git push



3. exit building

TODO ESTO CON UNA PRESENTACIÓN MEJOR MAS BONITO

Todo esto con una presentación mejor:

<https://rogerdudler.github.io/git-guide/index.es.html>

Chuleta comandos git:

<https://victorhckinthefreeworld.com/2016/02/24/chuleta-de-comandos-mas-usuales-en-git/>

Curso interactivo:

<http://try.github.io/>


Un libro barato:


Aprende Git: ... y, de camino, GitHub, de JJ Merelo (1€ en version kindle)

<https://github.com/JJ/aprende-git>

@torvalds' 2020 GitHub Skyline



Print on Shapeways 

Download STL file 

[HTTPS://SKYLINE.GITHUB.COM/](https://skyline.github.com/)

VR Ready