

RAICHU DEV TALKS

#2 GIT SIN MORIR EN EL INTENTO

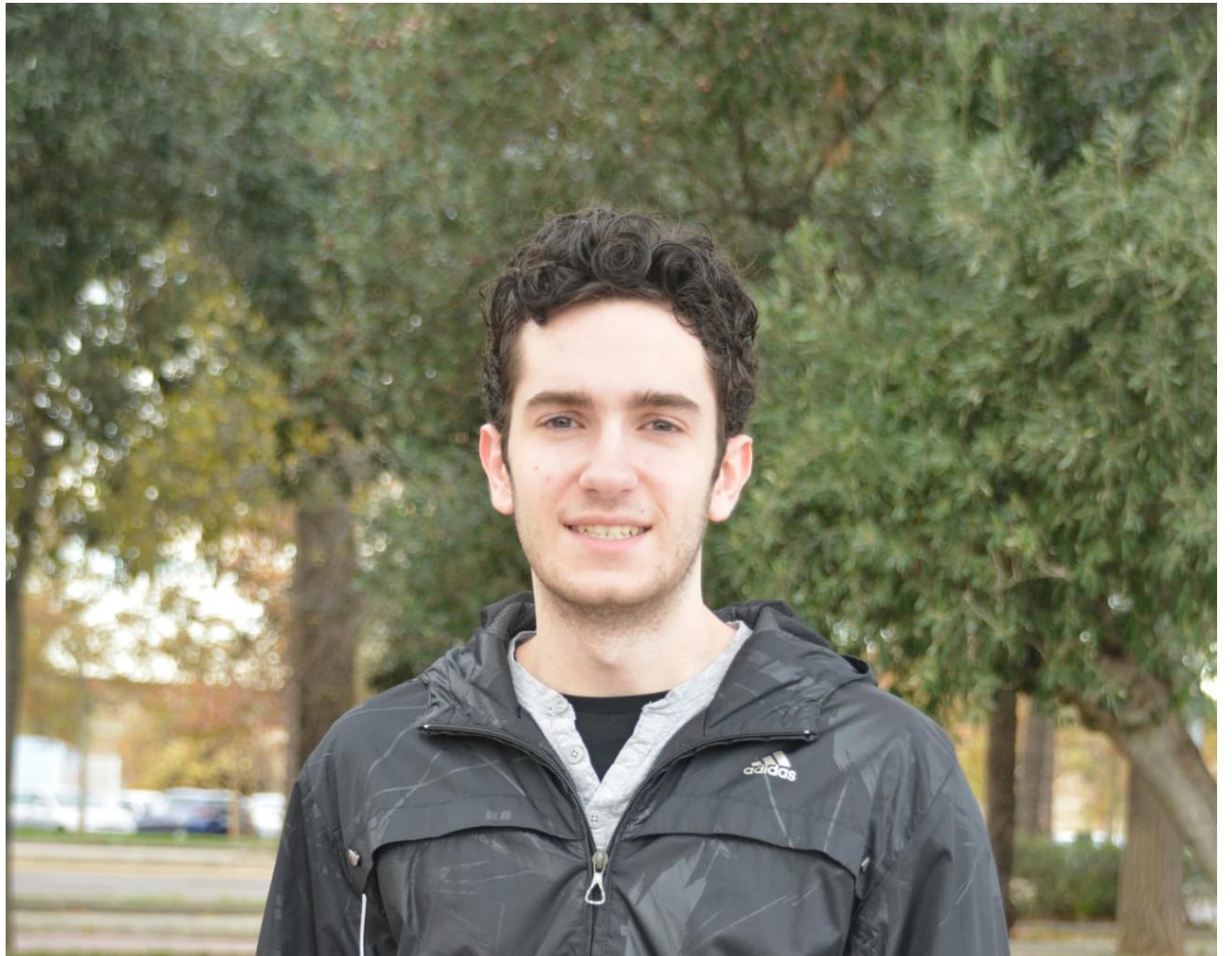
Disclaimer

- Compartiré las slides al final de la sesión
- Seguramente ya conozcáis muchos conceptos
- La sesión será grabada

Pepe Fabra Valverde

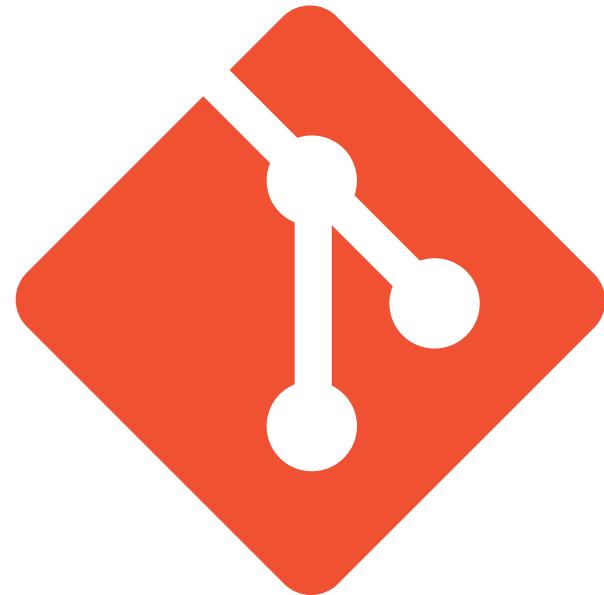
Frontend,
Valencia

Si nos esforzamos, todavía me parezco



Git

Linus Torvalds, 2005



git

Modelo mental



PRIMERA FASE









Commit

El átomo de Git

Commit

- hash
- blob (deltas y diffings)
- prev_commit_id
- message y descripción

git diff

QUÉ PUEDE CONTENER UN BLOB

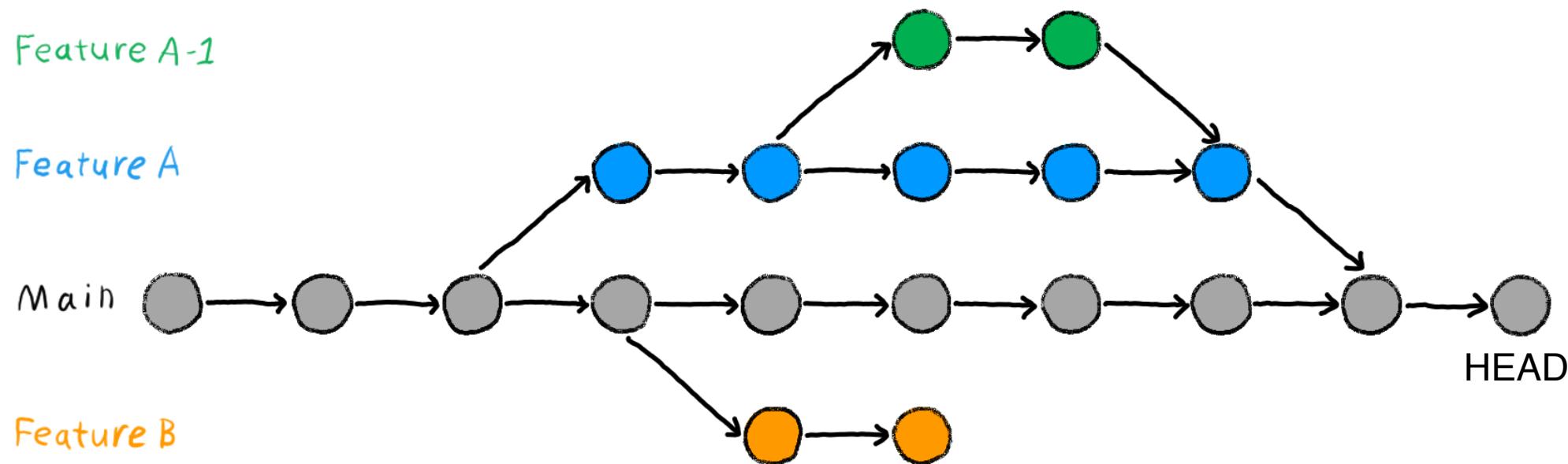
```
kendumez@kendumez-MacBook-Pro: ~ % → minikube-demo git:(main) ✘ git diff README.md
diff --git a/README.md b/README.md
index fb65379..6fc463a 100644
--- a/README.md
+++ b/README.md
@@ -1,3 +1,8 @@
"># Changelog
+
+3.21.24: Updated golang image
+
+
# Getting started with minikube
```

Un commit es commit porque
viene de otro commit

y juntos componen...

Ramas

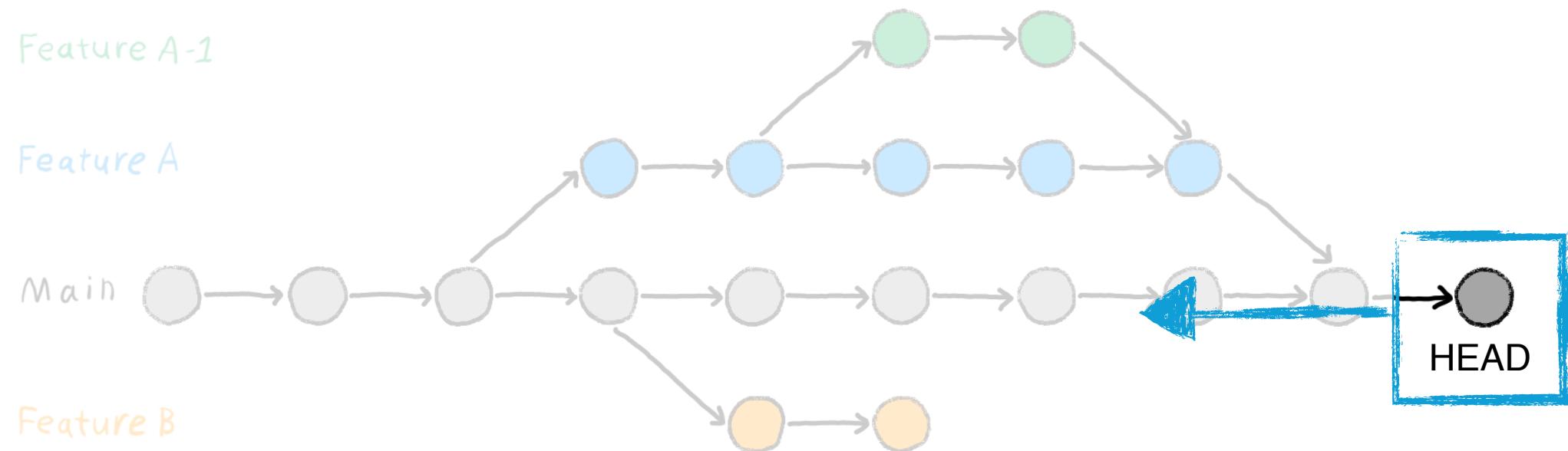
workingCopy, workingTree, HEAD



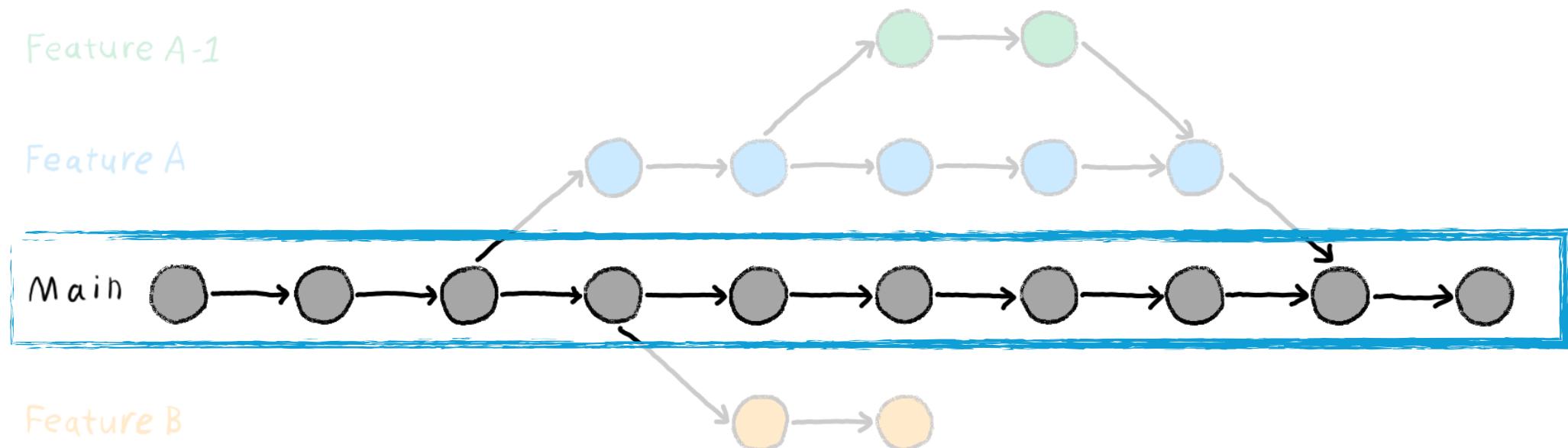
workingCopy, workingTree

HEAD

workingCopy



workingTree



Extra

Ramas huérfanas

- main
 - master esta deprecado por referencias a exclavitud
- No hay límite, tantas como quieras

forman...

Repositorios

Conjunto de ramas y commits... y más detalles

que se alojan en...

Remotos

origin, upstream y referencias

origin y upstream

- Estándar de facto
- Fluyen y convergen a un punto (upstream, corriente)

retomando los commits...

La chicha de un commit

Hashes

- Dado un contenido genera un uid
- No es crítico comprenderlo, solo saber que se usan

Deltas y diffing

- Un delta es la distancia entre dos puntos
 - Edit difference y Levenshtein distance

Deltas y diffing

- Diffing es encontrar los deltas de un mismo fichero para distintas referencias (que no commits... spoiler)

Deltas y diffing

- Son la base de los conflictos (**LXCY**)

Deltas y diffing

- Línea X, columna Y -> delta
- se eliminan dos caracteres
- se añade una “Z”

Deltas y diffing

- Git almacena cambios y guarda un resultado
- El commit es el cambio aplicado, no la instancia del fichero

Inmutabilidad

- Consistencia y trazabilidad

Inmutabilidad

- Alterar un commit crea uno nuevo
- Si un commit dependía de este, se crea otra copia (habrá cambiado su prev_commit_id)

Inmutabilidad

- **SABER MÁS:** Programación Funcional, Spark y RDD

SEGUNDA FASE

Revisões o referencias

Commit, rama, tag

Revisões o referencias

- Término paraguas
- Usado por merge, rebase, commit, pull

git merge

Propio Base Mergeado



git rebase

Rebase puede darnos co-autoría de los commits, se reescriben

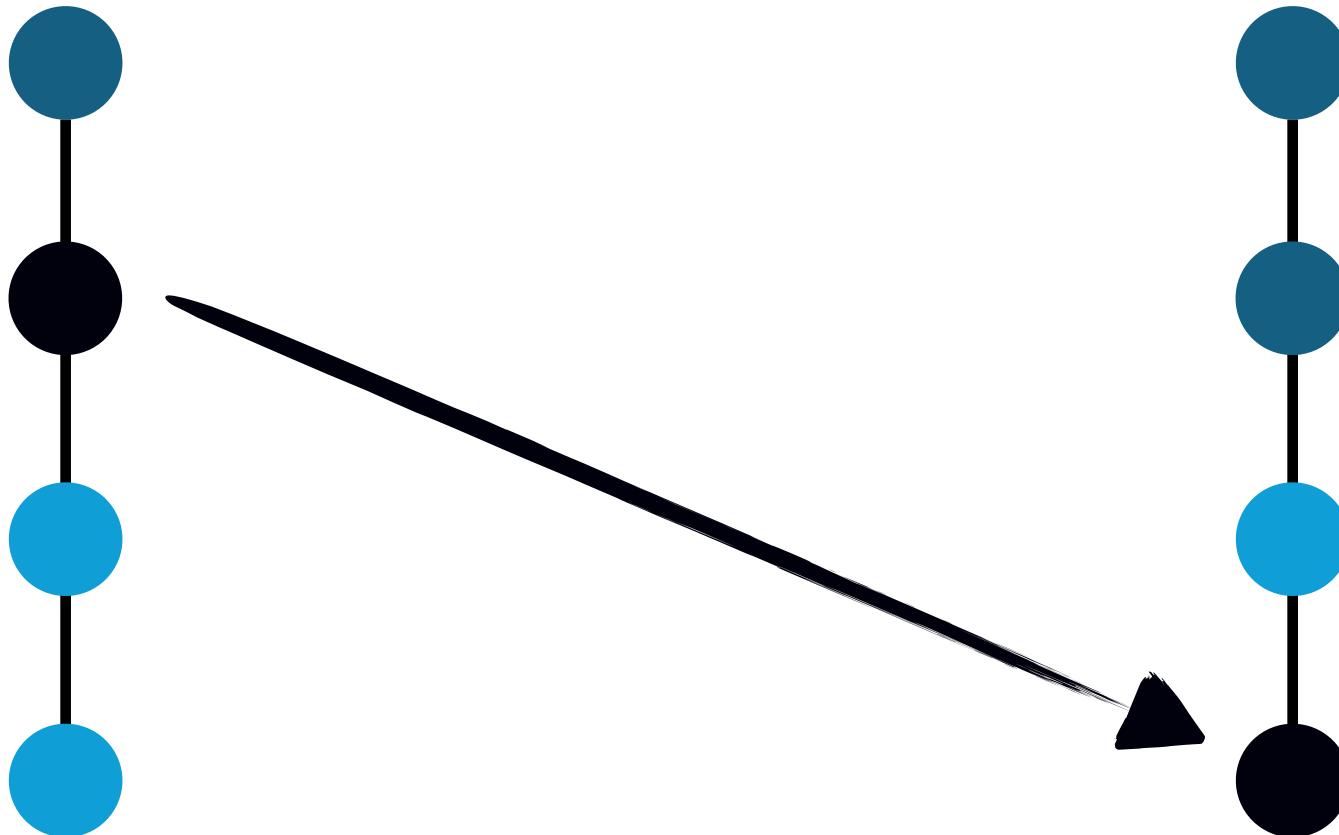
■ Propio ■ Base ■ Rebaseado



por debajo usa

git cherry-pick

Propio Base Cherry-pick



rev-parse y log

Recupera el hash del commit que quieras

otra manera de verlos

git merge

TENEMOS DOS CUERDAS (RAMAS)



git merge

HACEMOS UN NUDO, MOVERNOS POR LA CUERDA YA NO FLUYE TANTO



Más cómodo, menos riesgo,
menos natural

git rebase

TENEMOS DOS CUERDAS (RAMAS)



git rebase

REESCRIBIMOS LA HISTORIA, SIEMPRE HA HABIDO UNA (1) CUERDA



Menos cómodo, más riesgo,
más natural

**Los nudos impactan al mirar
hacia atrás**

Merge, rebase y cherry-pick
generan...

Conflictos

Diffing, --strategy-options

TERCERA FASE



s más vendidos

Amazon Basics

Ofertas

Música

Prime ▾

Últimas Novedades

Hogar y cocina

Informática

Tarjetas regalo ▾

Audible

Libros

Juguetes y juegos

La Tie

MEDINA EL ESTAFADOR DE FAMOSOS

Ver ahora



Seguir comprando ofertas

[Ver todas las ofertas](#)

Oferta Top



-30% Oferta flash



-23% Oferta flash



-25% Oferta flash



-18% Oferta flash

Ofertas para ti

[Explora ahora](#)

Identifícate para una mejor experiencia

[Inicia sesión de manera segura](#)

Portátiles con Inteligencia Artificial

Descubre la próxima
generación de
dispositivos

[Comprar ahora >](#)

Patrocinado





Click N Ship®

P



PRIORITY MAIL 3-DAY™

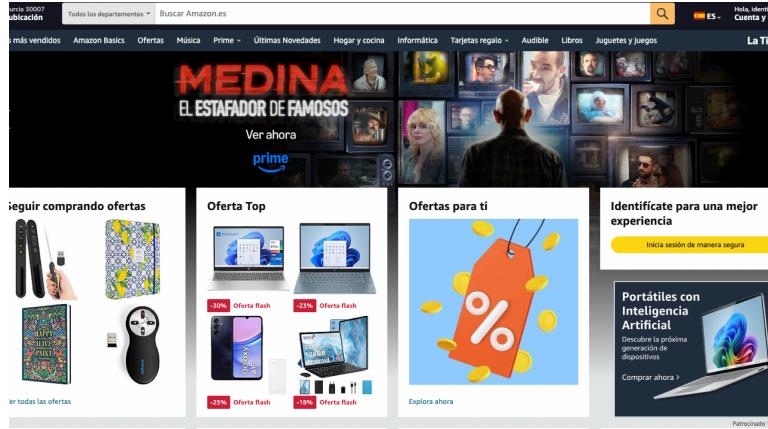
Delivery confirmation number:

00000000000000000000
00000000000000000000
00000000000000000000



Delivery confirmation number:
00000000000000000000

Carga y descarga



git fetch



git pull



git push

git fetch [remoto]

- Update status, no afecta a commits
- Usado por pull y push por debajo

git pull [remoto]

- Query, afecta a commits
 - Principalmente a la base
- Por defecto merge, puede ser rebase

git push [remoto]

- Update, no afecta a commits
 - No funcionará si no tiene los últimos cambios del remote
- --force, implicaciones y casos de uso

git pull --rebase && git push

- Para repos con mucha concurrencia

Y ahora con tags

- git fetch --tags
- git pull --tags
- git push --tags

tags

Referencia, se pueden mergear, target de checkouts

Únicas en todo el repo

COMMITs

git add y git rm

Entornos y estados

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Staged

- Git hace seguimiento del fichero

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Staged

- Git hace seguimiento del fichero

Stashed

- Git lo tiene apartado de la zona de trabajo

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Staged

- Git hace seguimiento del fichero

Stashed

- Git lo tiene apartado de la zona de trabajo

Committed

- Git tiene agrupado el cambio en una referencia

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Staged

- Git hace seguimiento del fichero

Stashed

- Git lo tiene apartado de la zona de trabajo

Committed

- Git tiene agrupado el cambio en una referencia

Remote

- Git tiene ya en el remote, como base, el cambio

Entornos y estados

Unstaged

- Git no hace seguimiento del fichero

Staged

- Git hace seguimiento del fichero

Stashed

- Git lo tiene apartado de la zona de trabajo

Committed

- Git tiene agrupado el cambio en una referencia

Remote

- Git tiene ya en el remote, como base, el cambio

Commits convencionales

semántica(scope!): JIRA-XXX mensaje

<https://www.conventionalcommits.org/en/v1.0.0/>

Eliminar de staged y remote

`git rm --cached`

Relanzar una build

`git commit -m “ci: Trigger build” --allow-empty`

¿Atómicos o en batch?

Corregir un commit

`git commit --amend`

Me pide hacer pull y push

REBASE

Reescribir la historia

Reescribir commits

Push force o merge innecesario

Rebase interactivo

El mundo oculto de rebase

git rebase --i HEAD~N

- P - Pick
- E - Edit
- S - Squash*
- D - Drop

cherry-pick y rev-parse

git cherry-pick commit

Checkout, HEAD y ^ o ~

Movimiento y puntero

HEAD

- ^
 - Padre directo
- ~
 - Padre relativo (generaciones)

Docs: <https://git-scm.com/docs/gitrevisions>

Co-autoría de commits

Inmutabilidad y rebases

CONFIGURACIONES

git config [scope]

1. Repositorio

- Por defecto, solo afectará al proyecto

git config [scope]

1. Repositorio

- Por defecto, solo afectará al proyecto

2. System

- Se configura a nivel de usuario

`git config [scope]`

1. Repositorio

- Por defecto, solo afectará al proyecto

2. System

- Se configura a nivel de usuario

3. Global

- Se configura a nivel de ordenador, todos los users

Usar VSCode para git

`git config --global core.editor "code —wait"`

*Comprueba que “code --help” te funciona

`Si quieres seguir usando VIM... supongo que puedes`

rerere

Reuse recorded resolution

`git config [scope] rerere.enabled true`

Peligro de rerere

git fetch con poderes

Alias que haga fetch, pull --rebase y push

git sync (por ejemplo)

Headless

SourceTree y GH Desktop

PRS Y CODE REVIEW

Nuevo formato de PRs

Título

type(raichu): JIRA-XXX desc

Descripción

What's the focus of this PR?

How to review this PR?

Related Issues

Type of changes

Checklist

Descripción

What's the focus of this PR?

How to review this PR?

Related Issues

Type of changes

Checklist

Descripción

What's the focus of this PR?

How to review this PR?

Related Issues

Type of changes

Checklist

Descripción

What's the focus of this PR?

How to review this PR?

Related Issues

Type of changes

Checklist

Descripción

What's the focus of this PR?

How to review this PR?

Related Issues

Type of changes

Checklist

Nos falta

Definition of Done y Ready

Labels

deploy, front, back, raichu

Github <3 Markdown

<https://github.com/orgs/community/discussions/16925>

 Note	Highlights information that users should take into account, even when skimming.
 Tip	Optional information to help a user be more successful.
 Important	Crucial information necessary for users to succeed.
 Warning	Critical content demanding immediate user attention due to potential risks.
 Caution	Negative potential consequences of an action.

Github <3 Markdown

```lang

Código más visual

```

Estados de una PR

Open, Closed, Draft, Merged

Closed by #PR_ID

Enlazar la PR a JIRA

Como authors

- PRs no muy grandes
- Describir con claridad cómo probar la PR en local
- Aclarar el estado de la PR (WIP, Draft, ¿se puede megear?)

Como reviewers

- X no es build rota, tal vez le falten approvals
- Comentario al final de la revisión, destaca lo bueno y una visión general
- **Start review** y así subes todos los comentarios de golpe



I Am Devloper

@iamdevloper · [Follow](#)



10 lines of code = 10 issues.

500 lines of code = "looks fine."

Code reviews.

10:58 AM · Nov 5, 2013



6.6K



Reply



Share

[Read 109 replies](#)

Proposal

Definition of Checked Definition of Mergeable

RELEASE

¿Mergeos a ramas
intermedias? ¿o flags internas?

Cómo generar una versión

Cómo deployear una versión

major.minor.patch

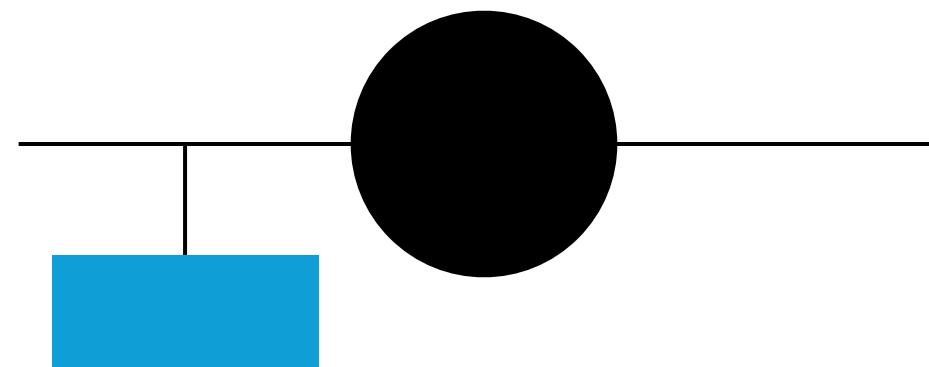
major.**minor**.patch

major.minor.patch

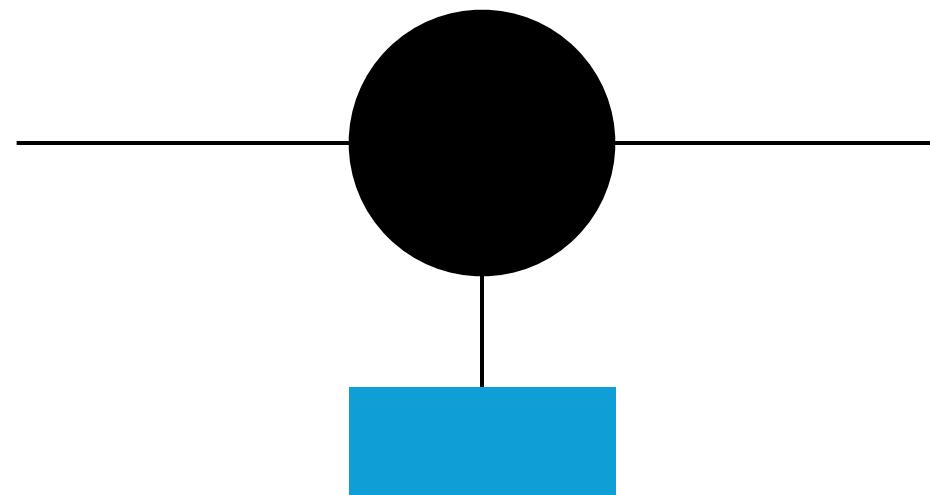
major.minor.patch

GIT HOOKS

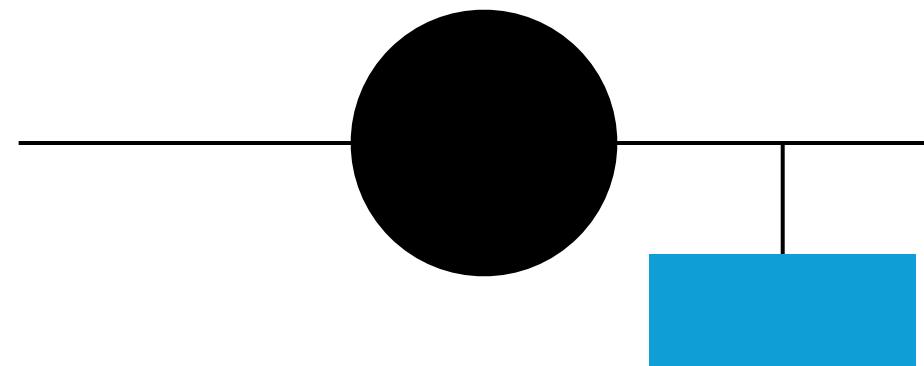
pre-commit



commit-msg



post-commit



Commit sin hooks

--no-verify

Opiniones

Castigo por commit atómico, pipeline en local

Extra

husky postinstall

TRUNK BASED DEVELOPMENT

Trunk Based Development

Short-lived branches, integración continua, feature flags

Git gestiona código, no despliegues

Git gestiona código, no despliegues

Gestionar código en prod se puede hacer con flags

Filosofías

- Mergeos a master cada poco
- Subidas a prod a diario
- Tests de integración en main, siempre

Trunk Based Development

<https://trunkbaseddevelopment.com/>

Ship / show / ask

Ship

- Subo sin PR

Show

- Pipeline de la PR y mergeo sin revisión

Ask

- Pido code review

Ship / show / ask

<https://martinfowler.com/articles/ship-show-ask.html>

RAICHU DEV TALKS

#2 GIT SIN MORIR EN EL INTENTO

¡¡GRACIAS!!

BONUS

Gracias por tomarte el tiempo de leerlo

Learn Git Branching

<https://learngitbranching.js.org/>

Adquiere soltura con los movimientos de Git

GitLens

Extensión de VSCode

Visualización de grafos, PRs y commits

Token Verified

Github Access Tokens

Máquina del tiempo

git checkout rama~N

Puedes viajar N commits al pasado, útil para aislar problemas

Stash

El alijo temporal, ideal para cuando te cambias a otras ramas en mitad de una faena

Blame

De qué commit hecho por quién es esta línea

GitLens, JetBrains lo usan internamente

Bisect

Trazar en el tiempo un cambio, pero de manera profesional 😎

Reflog

Caché de movimientos locales

- Un checkout
- Donde estaba HEAD hace X pasos

Format-patch y apply-patch

Trabajo colaborativo en esteroides, llévate cambios concretos

Internamente usa git diff y lo guarda en un .patch

Squash merge casi siempre

No limitas número de commits
Ni fuerzas estándares de los mensajes