<IV DEVS>

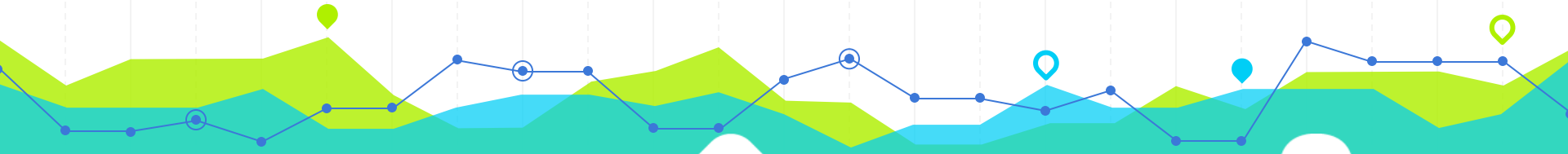
Comunidad de Programadores IV Región

Impulsando el desarrollo profesional de la Región

fb.com/groups/programadores.iv.region.chile/

<IV DEVS>

http://slack-ivdevs.herokuapp.com/



THIS IS

git

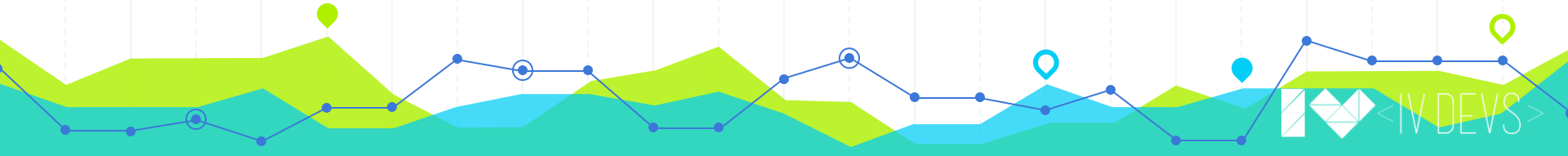
HELLO!

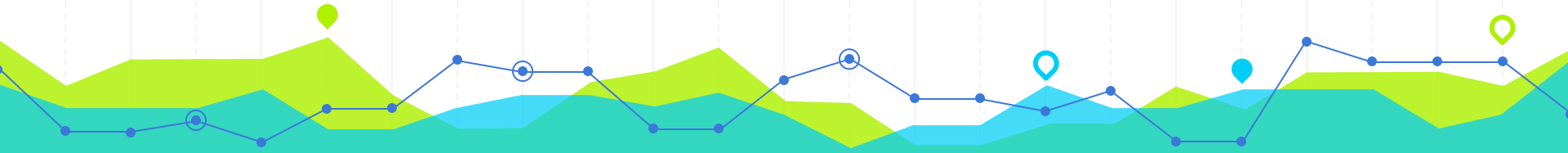
@Gerardo | @Cristofer

"

"El mejor **trabajo en equipo** proviene de hombres que **trabajan de forma independiente** para conseguir una meta al unísono."

James Cash Penny



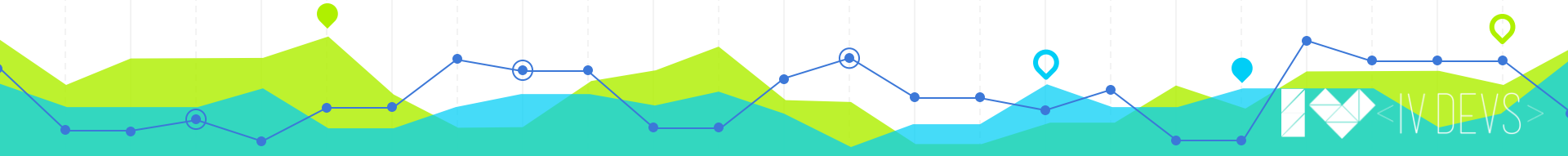


ANTES DE EMPEZAR

Algunos conceptos...

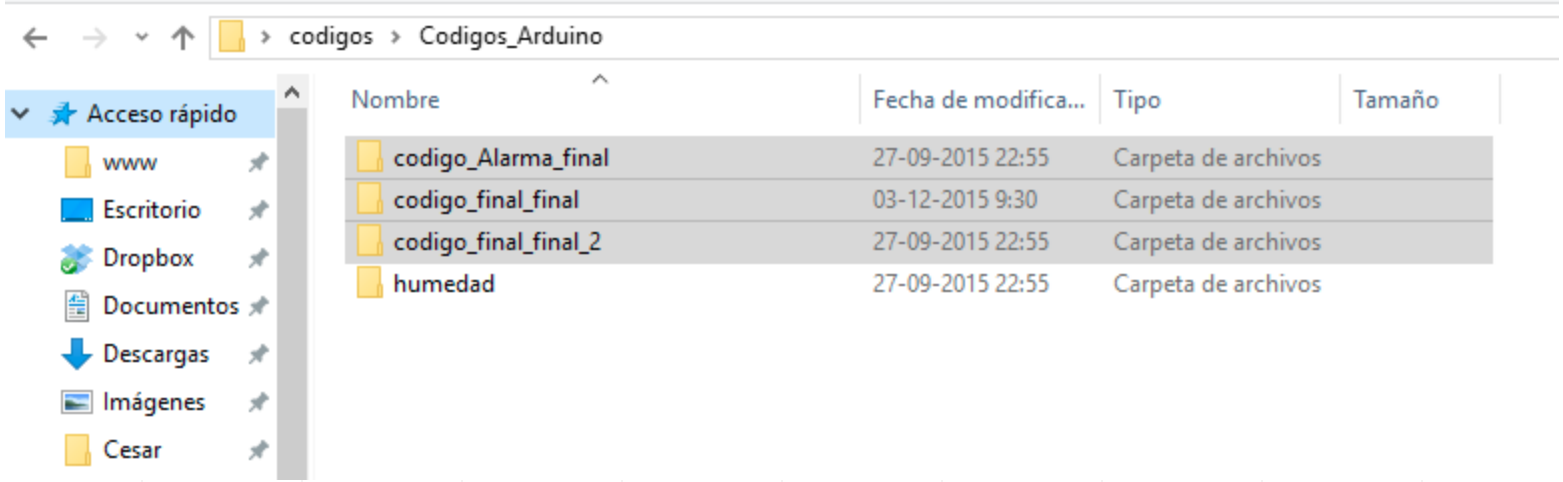
git != Bitbucket

GitLab



QUE ES GIT?

PARA QUE?



PARA QUE?

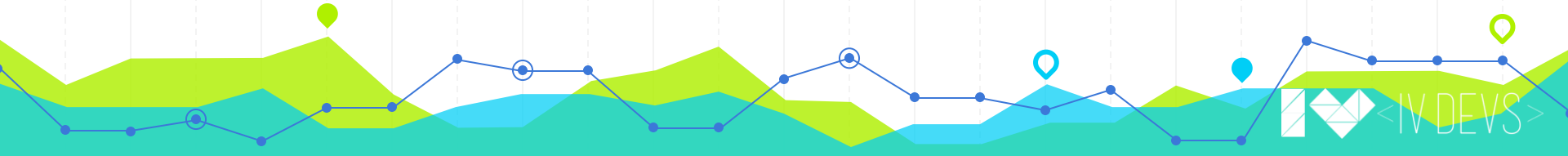
Nombre	Propietario	Modificados por última vez	Tamaño del archivo
Doc01Rev01	yo	29 oct. 2012 yo	679 KB
Doc01Rev02	yo	28 oct. 2012 yo	600 KB
Doc01Rev02	yo	21 ago. 2014 yo	–
Doc02Rev01	yo	29 oct. 2012 yo	305 KB
Doc03Rev01	yo	2 jun. 2013 yo	243 KB
Doc04Rev01	yo	29 oct. 2012 yo	1 MB
Doc05Rev01	yo	29 oct. 2012 yo	118 KB
Doc06Rev01	yo	29 oct. 2012 yo	39 KB
Doc06Rev02	yo	29 oct. 2012 yo	40 KB
Doc06Rev02	yo	21 ago. 2014 yo	–
Doc07Rev01	yo	29 oct. 2012 yo	696 KB

PARA QUE?

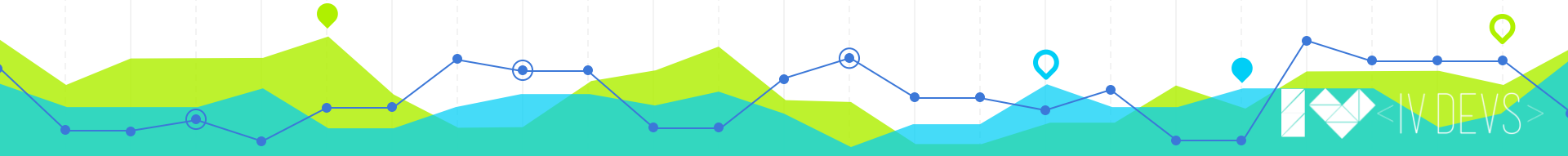
```
Cesar@EXTERMINADOR ~/Documents/iv-devs.github.io (master)
$ git log --oneline --graph
*   c9b6a93 Merge branch 'release/0.0.1'
|\
| * 2a1e34e ADD icon-slack!
| * d6fc5cb First modification to template
| * 19c98b8 CHANGE site template
|/
* ca5fdd3 FIX internal links
* 3d3da9e FIX
* 09e798a Initial Commit Under Construction
* e7eecde Initial commit
```

PARA QUE?

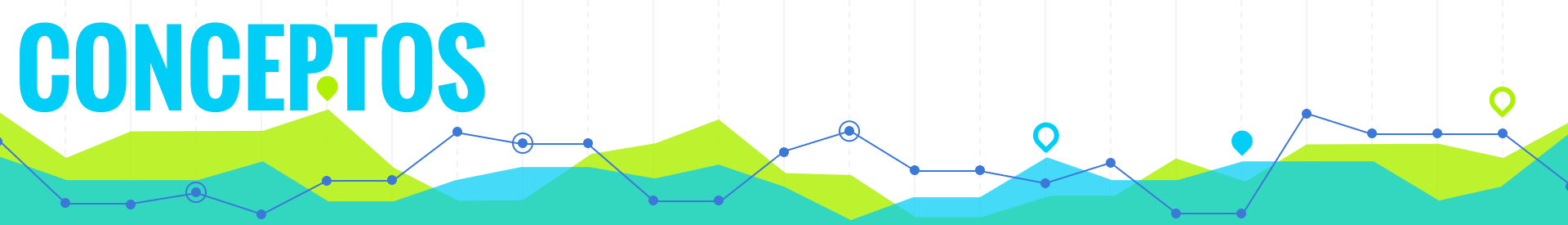
```
Cesar@EXTERMINADOR /C/Users/Cesar/Documents/iv-devs.github.io (master)
$ git blame index.html
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   1) <!doctype html>
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   2) <html lang="en">
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   3)       <head>
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   4)             <meta charset="UTF-8">
d6fc5cb7 (César Encina Leon 2015-12-05 15:34:25 -0300   5)             <meta name="description" content="Comunidad de programadores IV Región" />
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   6)             <meta name="keywords" content="Zerif, responsive, html, template, creative"/>
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   7)             <meta name="author" content="Mizanur Rahman" />
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300   8)             <meta name="viewport" content="width=device-width, initial-scale=1.0">
d6fc5cb7 (César Encina Leon 2015-12-05 15:34:25 -0300   9)             <title>IV Devs - Programadores IV Región</title>
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300  10)
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300  11)             <!-- ========================
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300  12)             FAV AND TOUCH ICONS
19c98b81 (César Encina Leon 2015-11-26 00:53:04 -0300  13)             ========================== -->
fff5e416 (César Encina Leon 2015-12-11 02:07:54 -0300  14)             <link rel="apple-touch-icon" sizes="57x57" href="/apple-icon-57x57.png">
fff5e416 (César Encina Leon 2015-12-11 02:07:54 -0300  15)             <link rel="apple-touch-icon" sizes="60x60" href="/apple-icon-60x60.png">
fff5e416 (César Encina Leon 2015-12-11 02:07:54 -0300  16)             <link rel="apple-touch-icon" sizes="72x72" href="/apple-icon-72x72.png">
```



CONCEPTOS



CONCEPTOS



COMMIT

CONCEPTOS

REPOSITORIO

REPO

<IV DEVS>

CONCEPTOS

RAMA

BRANCH

<IV DEVS>

CONCEPTOS

FLUJO

WORKFLOW

<IV DEVS>

REQUISITOS

Cuenta en Github
www.github.com

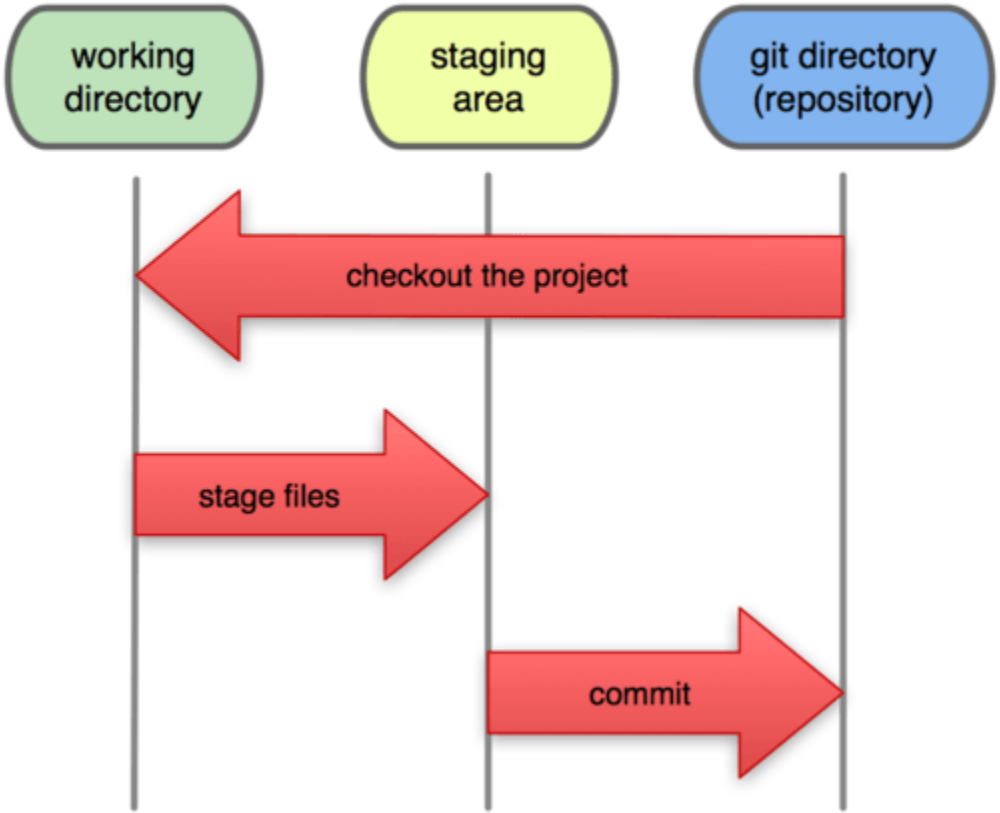
Tú laptop
Windows, Mac o GNU/Linux

<IV DEVS>

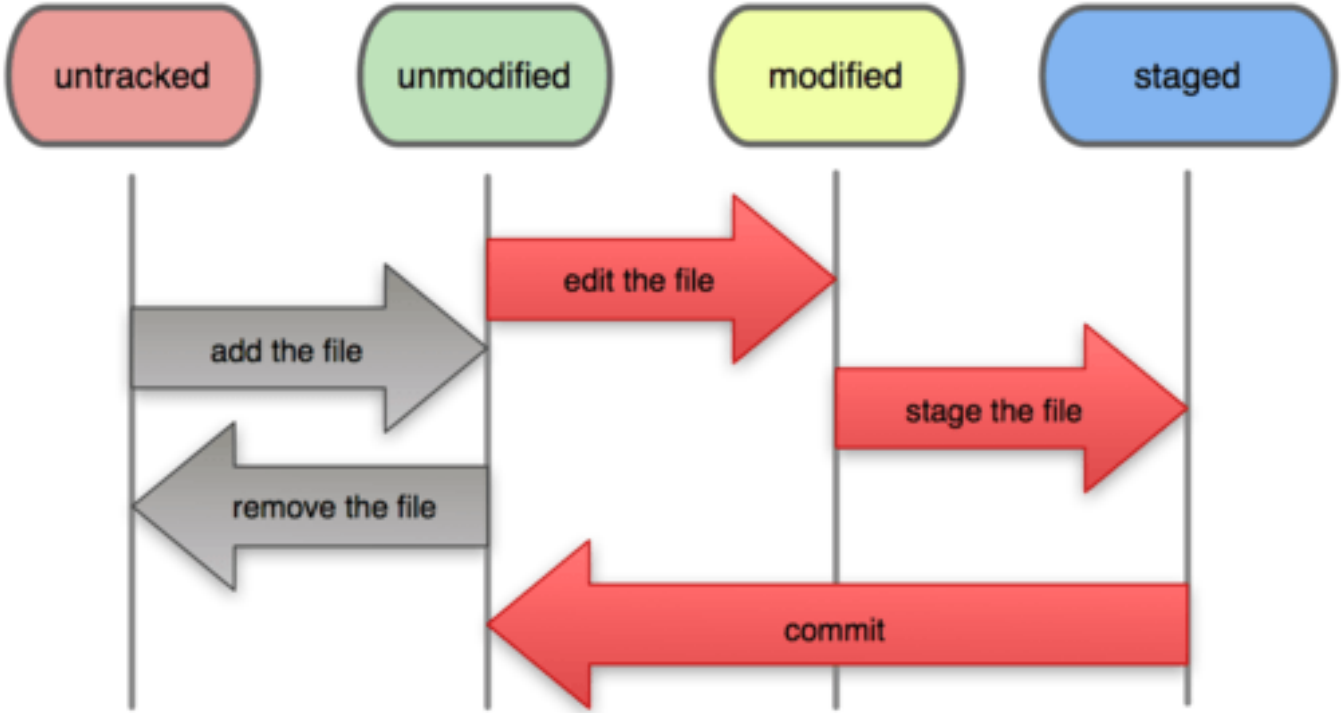
LA BRUJULA COWORK

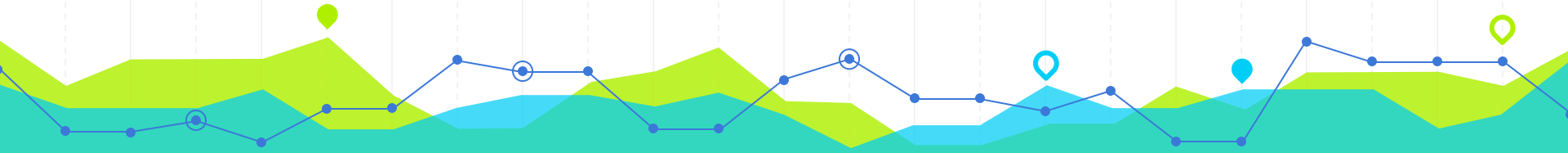
THIS IS git

Local Operations



File Status Lifecycle





1 INICIANDO

Aprendiendo a gatear

INSTALANDO

```
# Fedora
yum install git-core

# Debian
apt-get install git

# Mac
sudo port install git-core +svn +doc +bash_completion
+gitweb

# Windows
http://msysgit.github.com/
```

DESAFÍO #1: Instalar Git

GNU/Linux

```
apt-get install git
```

Windows

```
https://git-scm.com/download/win
```

OSX

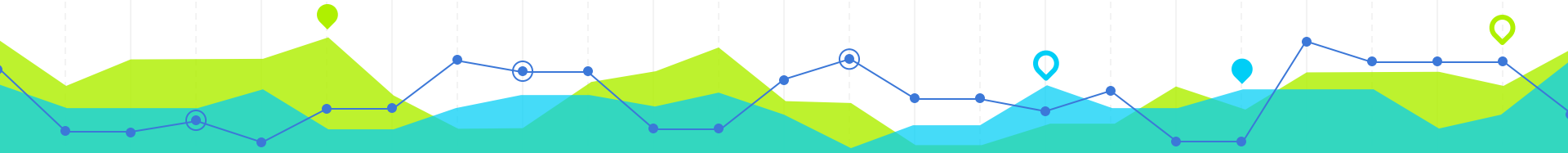
```
https://git-scm.com/download/mac
```

YOUR IDENTITY

```
# Identificandonos
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com


# Verificando
$ git config --list
```

DESAFÍO #2: Quién Eres

```
# Identificate
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com


# Verificando
$ git config --list
```



2 BASICS

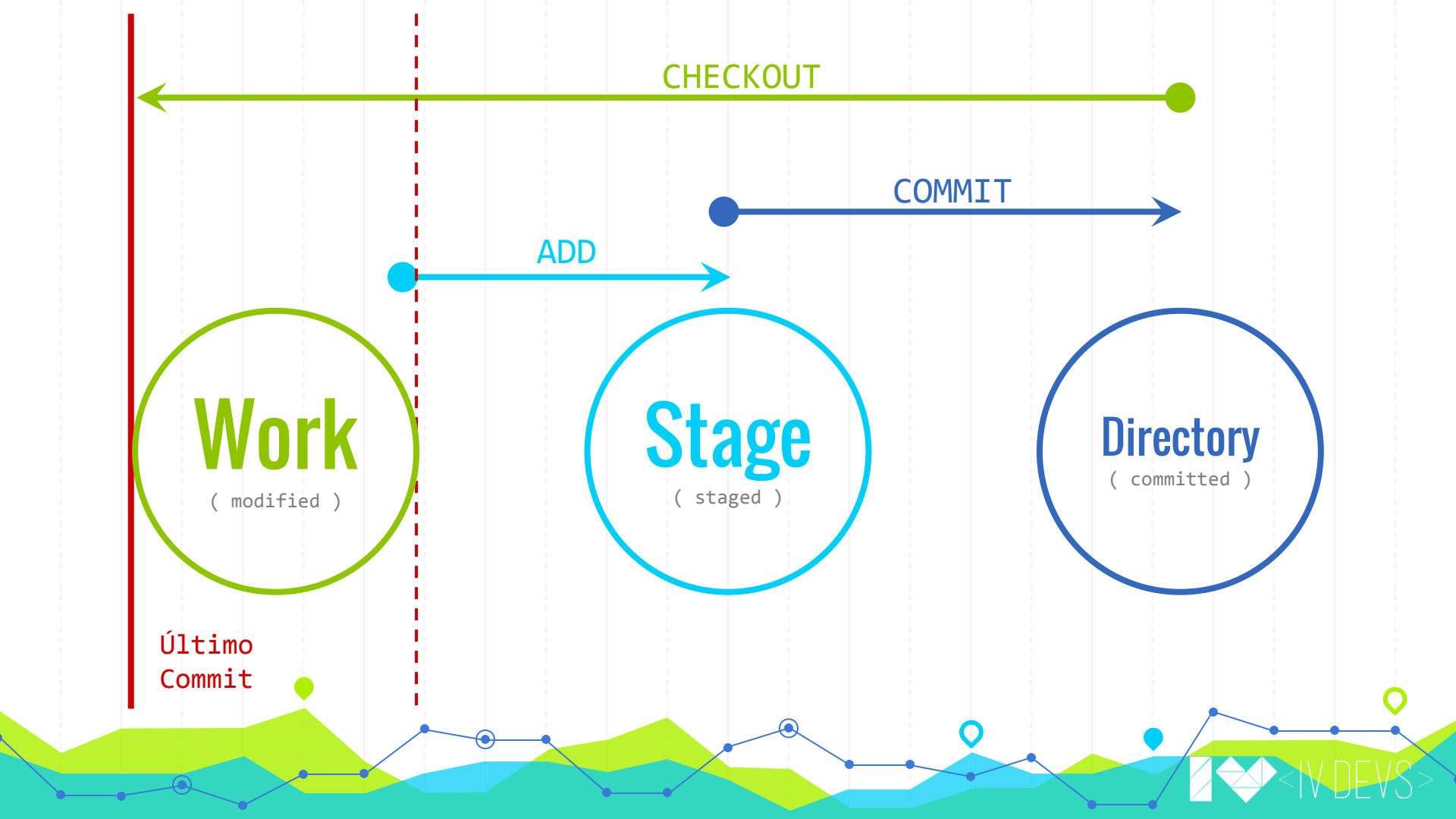
Primeros Pasos

INICIALIZANDO UN REPO

```
# Inicializar en local
$ git init

# Add
$ git add *.c
$ git add README

# 1º Commit
$ git commit –m 'initial project version'
```

DESAFÍO #3: Primer Repo

```
# Crear carpeta en el escritorio
$ mkdir tu_nombre

# Ingresa a la carpeta
$ cd tu_nombre

# Inicializar un repo
$ git init
```


CHECKOUT
COMMIT
ADD
Work
(modified)
Stage
(staged)
Directory
(committed)
Último
Commit

DESAFÍO #4: Agregando Archivos

```
# Crear un archivo dentro de la carpeta
$ git touch tu_nombre.txt

# Git status
$ git status

# Dile a git que controle los cambios futuros
$ git add <name_file>

# Git status
$ git status
```

DESAFÍO #5: 1° Commit

```
# Git status
$ git status

# Primer Commit
$ git commit

# Git status
$ git status
```

LOG

```
# ver log
$ git log
$ git log --oneline
$ git log --pretty=format:"%h - %an, %ar : %s"
```

```
ca82a6d - Scott Chacon, 11 months ago : changed the version number
085bb3b - Scott Chacon, 11 months ago : removed unnecessary test code
a11bef0 - Scott Chacon, 11 months ago : first commit
```

Ref: www

DESAFÍO #6: Consultando el Log

```
# Revisa el log de commit
$ git log
```

```
# Revisa el log de commit resumido
$ git log --oneline
```

DESAFÍO #7: Editando Archivos

```
# Abre el archivo (con su editor regalón)
tu_nombre.txt

# Agrega en la línea 1
Esta es la línea 1

# Guarda el archivo

# Pregunta el estado
$ git status
```

DESHACIENDO COSAS

```
# Modificando tu última confirmación
$ git commit --amend
```

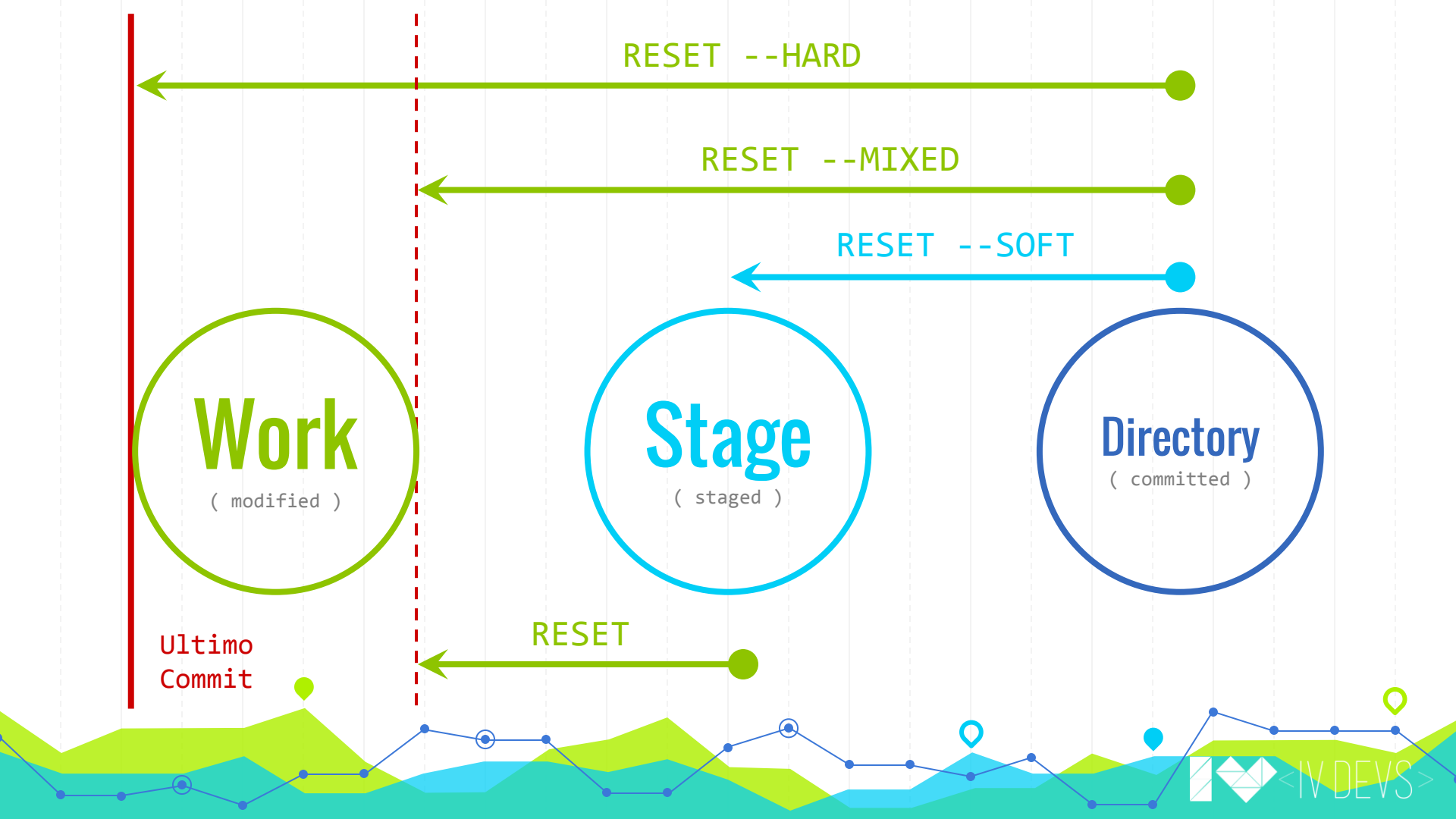
```
$ git commit -m 'initial commit'
$ git add forgotten_file
$ git commit --amend
```

DESAFÍO #8: AMEND

```
# Pregunta el estado
$ git log

# Agrega un nuevo archivo
$ touch file88.txt

# Agregarlo al mismo commit anterior
$ git commit --amend

# Pregunta el estado
$ git log
```

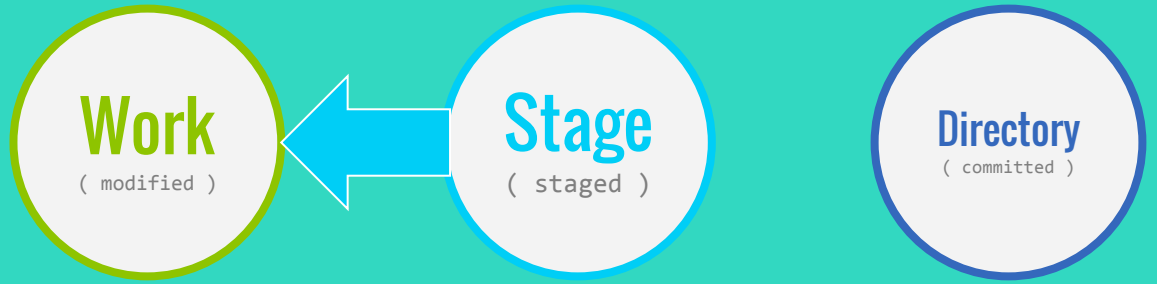
DESHACIENDO COSAS

```
# Deshaciendo la preparación de un archivo (sacarlo del stage)
$ git reset HEAD archivo

# Deshaciendo los cambios de un archivo (al commit anterior)
$ git checkout -- benchmarks.rb
```


RESET --HARD
RESET --MIXED
RESET --SOFT
Work
(modified)
Stage
(staged)
Directory
(committed)
RESET
Ultimo
Commit
<IV DEVS>

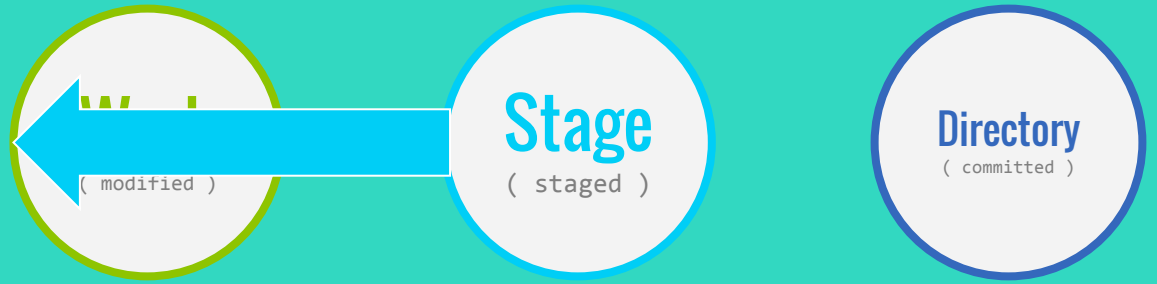
DESAFÍO #9: RESET

```
# Abre un archivo y editalo
File88.txt

# Agregalo al staging
$ git add File88.txt

# Pregunta el estado
$ git status

# Pregunta el estado
$ git reset
```



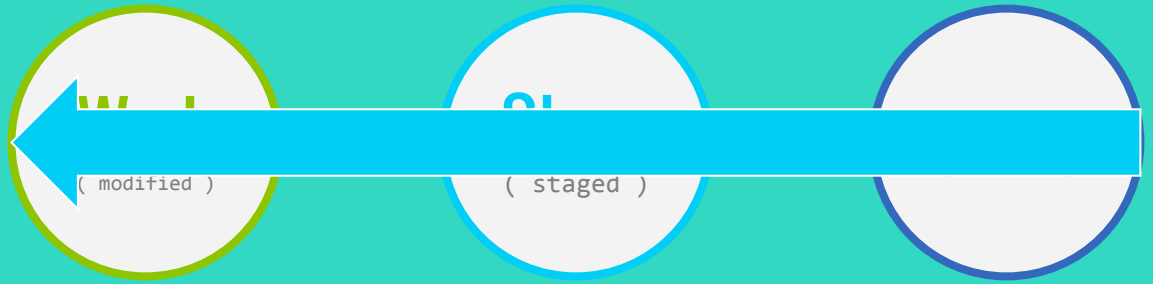
DESAFÍO #10: RESET –HARD

```
# Abre el archivo

# Agrega en la línea 1
Esta es la linea 1

# Guarda el archivo

# Pregunta el estado
$ git status
```



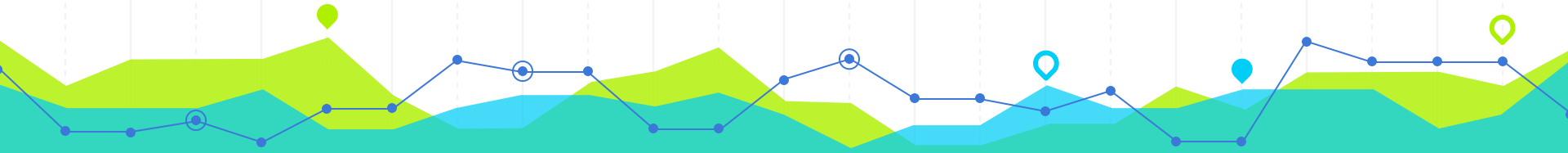
DESAFÍO #10: RESET –HARD HEAD

```
# Abre el archivo

# Agrega en la línea 1
Esta es la linea 1

# Guarda el archivo

# Pregunta el estado
$ git status
```



TIPS

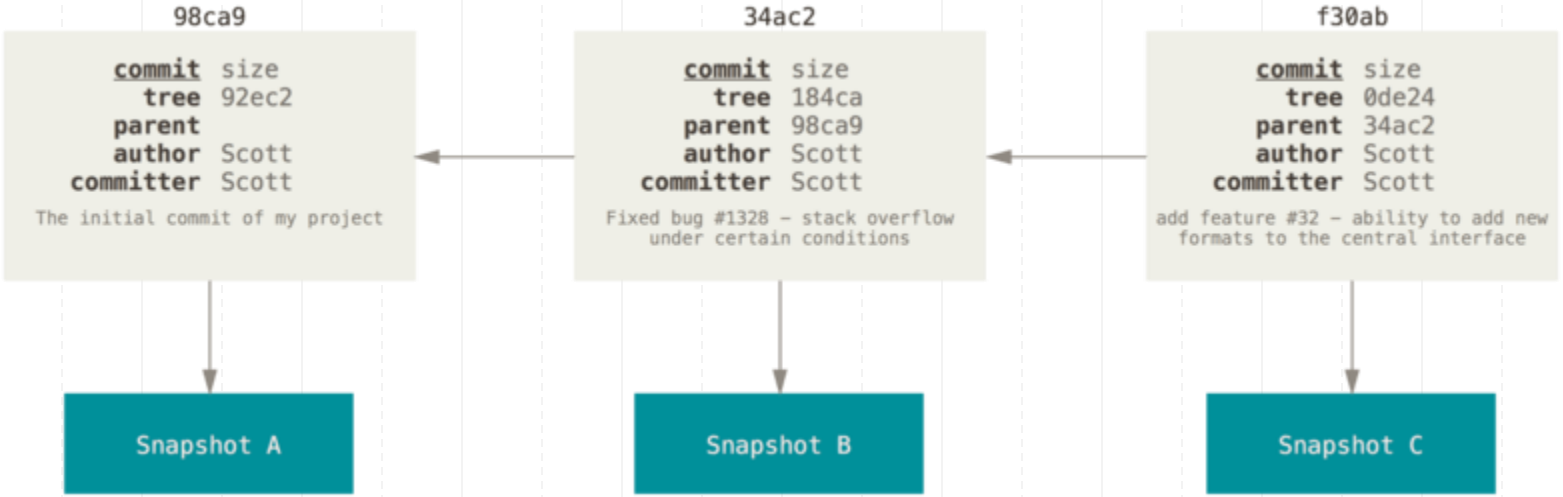
```
# Autocompletado

# Alias de git
$ git config --global alias.tu_alias instruccion

$ git config --global alias.co checkout
$ git config --global alias.br branch
$ git config --global alias.ci commit
$ git config --global alias.st status
$ git config --global alias.lg 'log --pretty=format:"%h - %an, %ar : %s"'
```



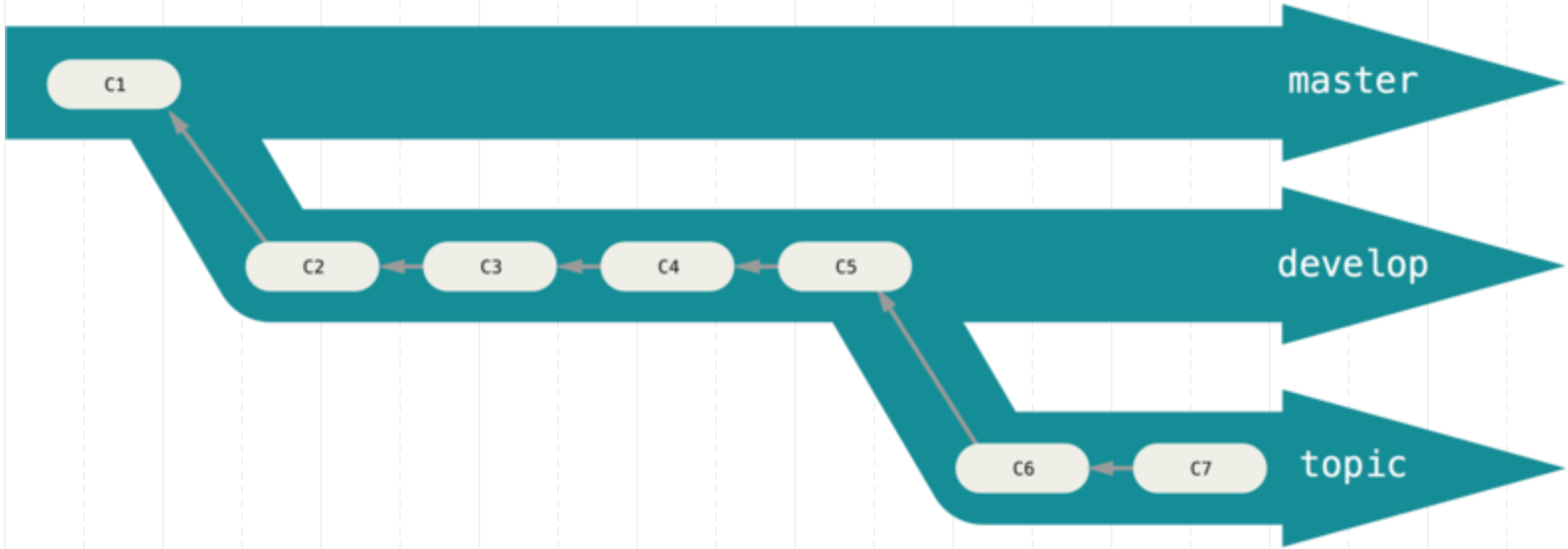
3 RAMAS

Dividiendo el <código>... unificando esfuerzos...

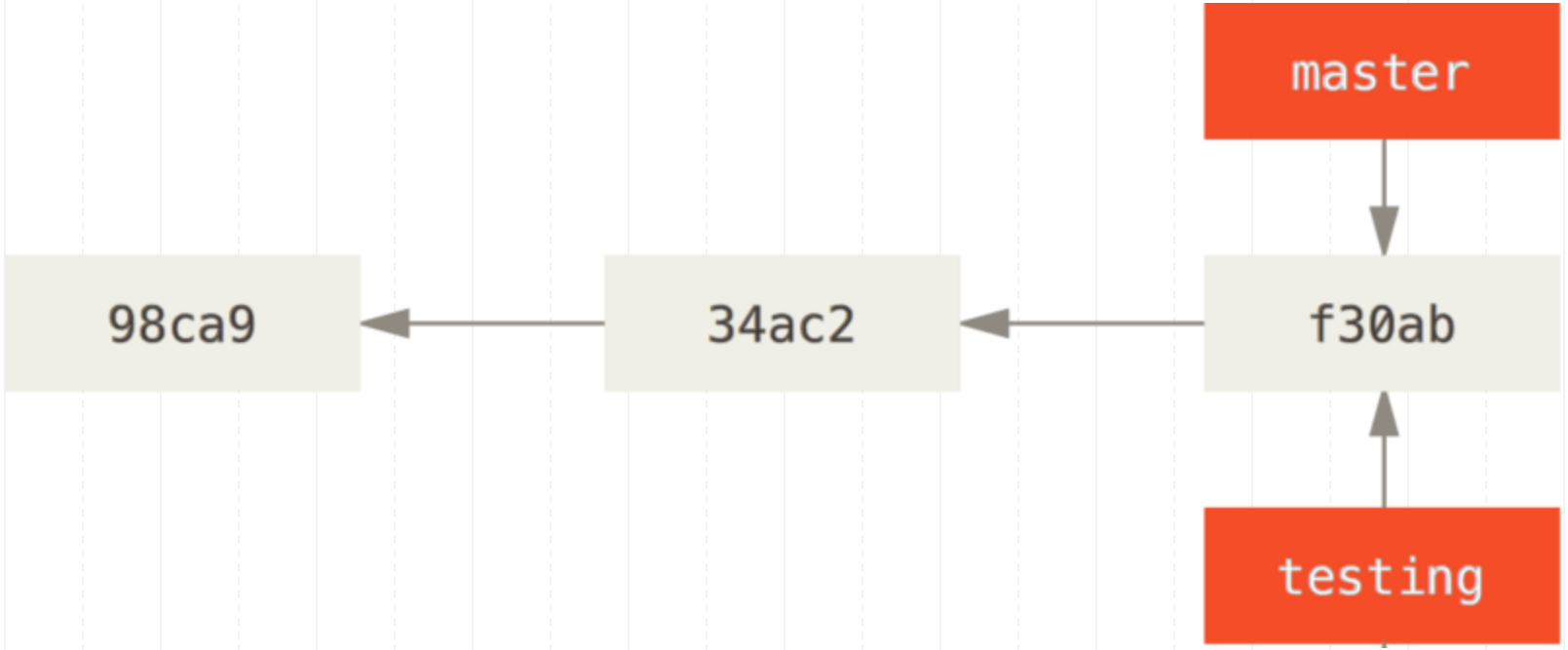
Si haces más cambios y vuelves a confirmar, la siguiente confirmación guardará un apuntador su confirmación precedente.



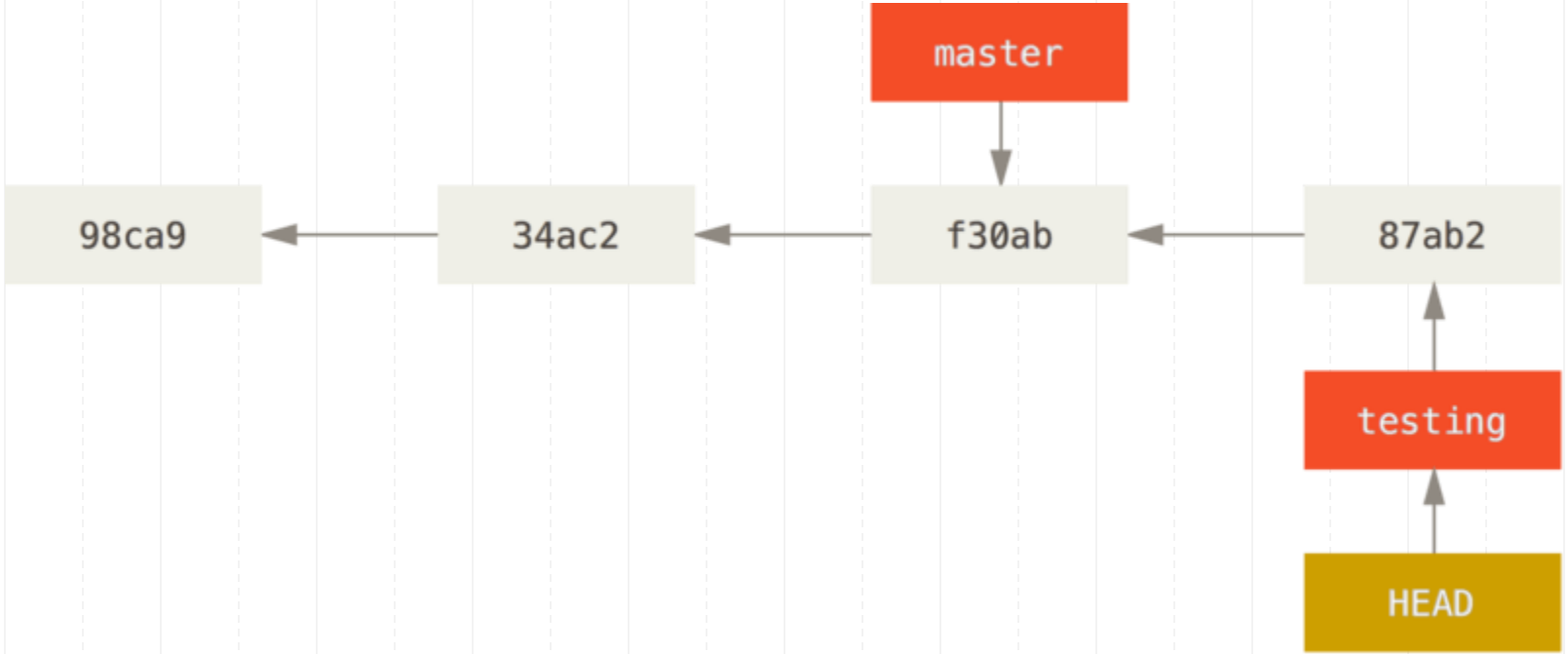
CONCEPTO



Una rama Git es simplemente un apuntador móvil apuntando a una de esas confirmaciones.



¿cómo sabe Git en qué rama estás en este momento?



BRANCH

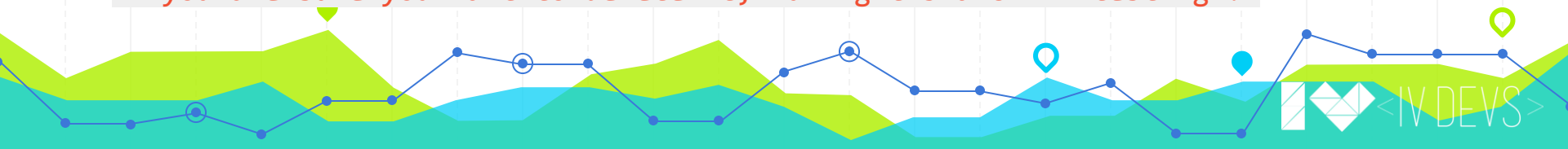
```
# Crear Rama
$ git branch [nueva_rama]

# Ver ramas existentes (Se muestra un * sobre la rama actual)
$ git branch

# Cambiar de rama
$ git checkout rama

# Crear y Cambiar de rama
$ git checkout -b rama
```

MERGE

```
# ¿Nos podemos cambiar cuando queramos de rama?
$ git ...

# Nos cambiamos a la rama de destino
$ git checkout rama_que_recibe

# Unimos
$ git merge --no-ff [rama_a_unir]
```

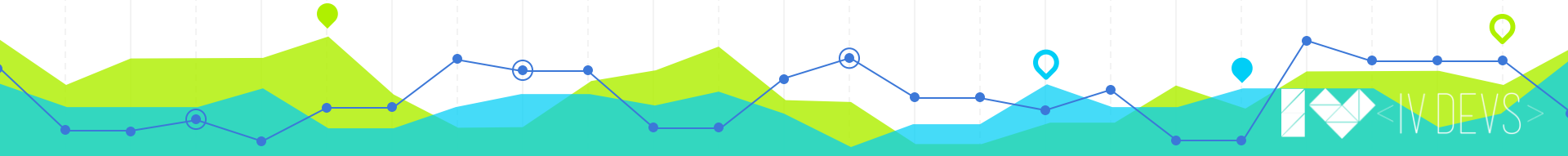
BRANCH

```
# Muestra el último commit de cada rama
$ git branch -v

# Muestra ramas ya fusionadas
(las que podemos borrar)
$ git branch --merged

# Muestra ramas que aún tienen trabajo sin fusionar
(las que NO podemos borrar)
$ git branch --no-merged

# Si tratamos de borrar una rama NO fusionada
$ git branch -d testing
error: The branch 'testing' is not an ancestor of your current HEAD.
If you are sure you want to delete it, run 'git branch -D testing'.
```



CONFLICTOS

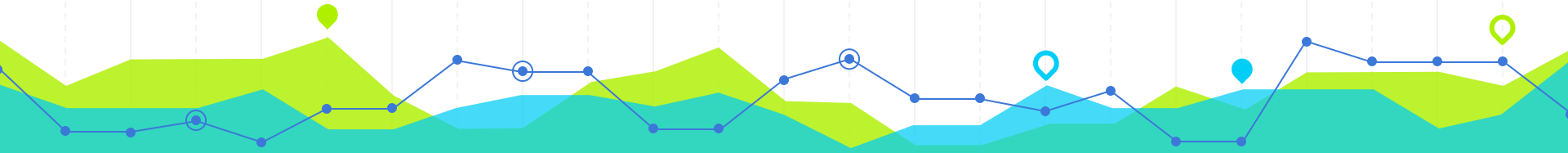
```
<<<<<<< HEAD:index.html
<div id="footer">contact :
email.support@github.com</div>
=======
<div id="footer">
  please contact us at support@github.com
</div>
>>>>>>> iss53:index.html
```



“

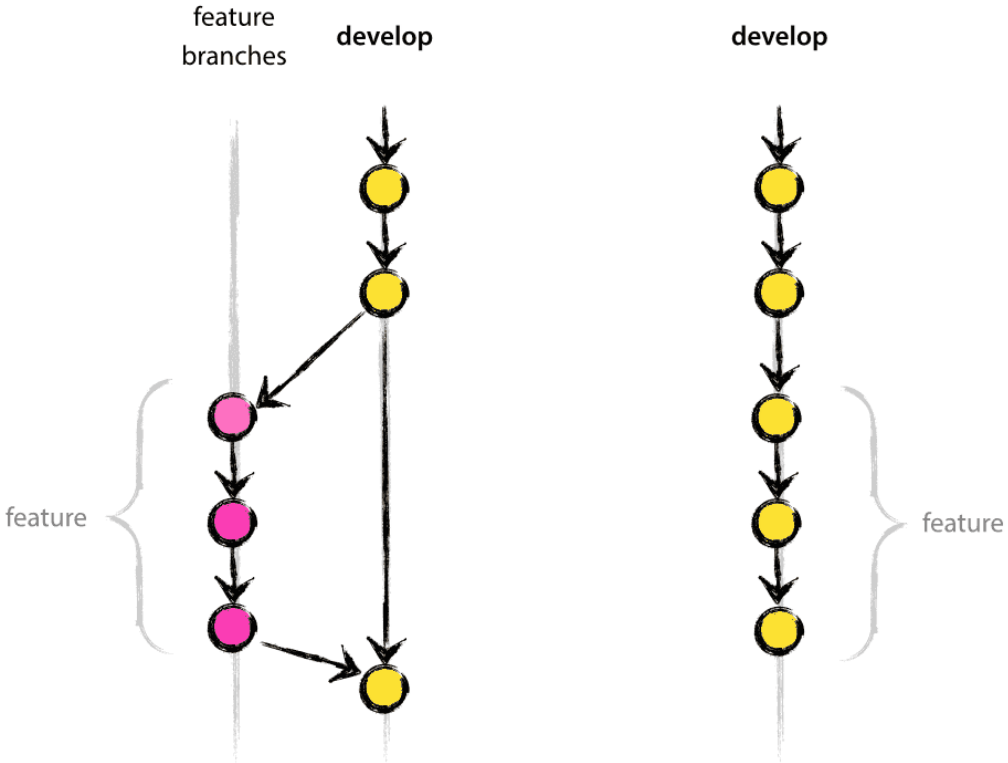
Las ramas a veces pueden convertirse en un caos…

he aquí una alternativa para evitarlo



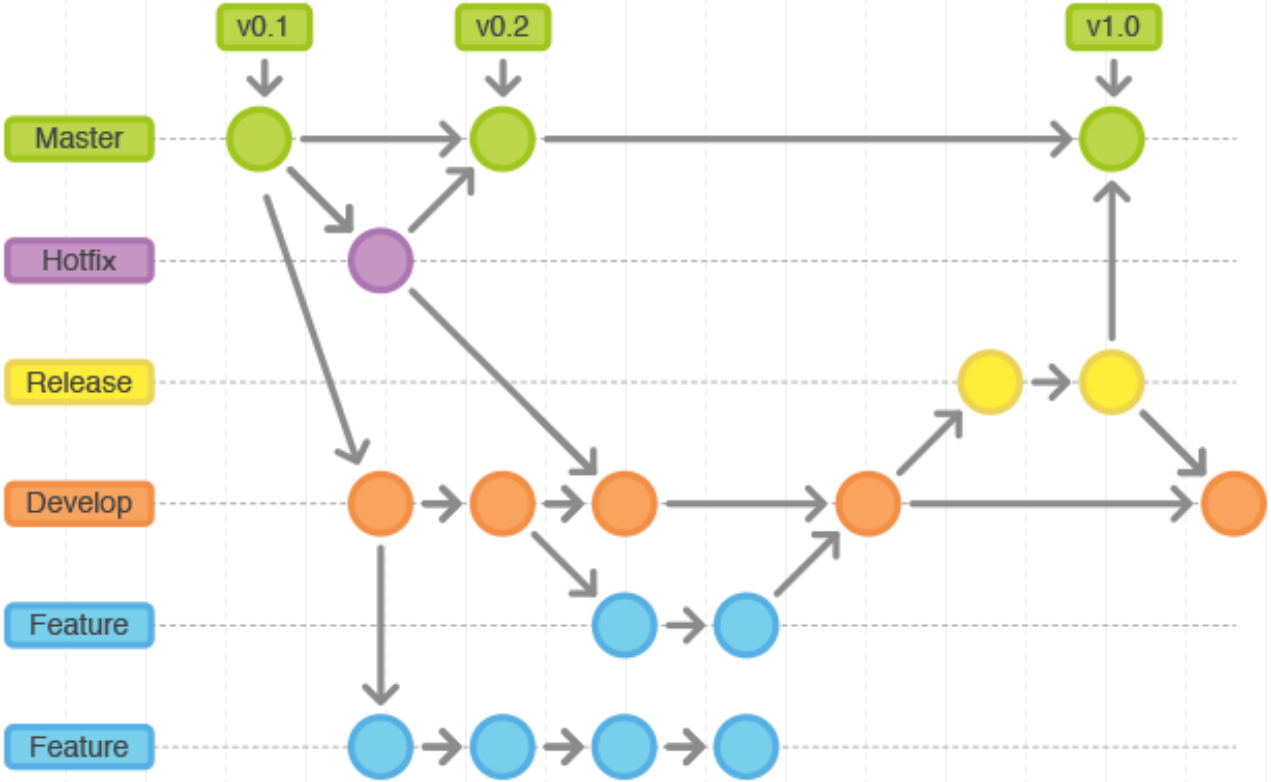
4 GIT FLOW

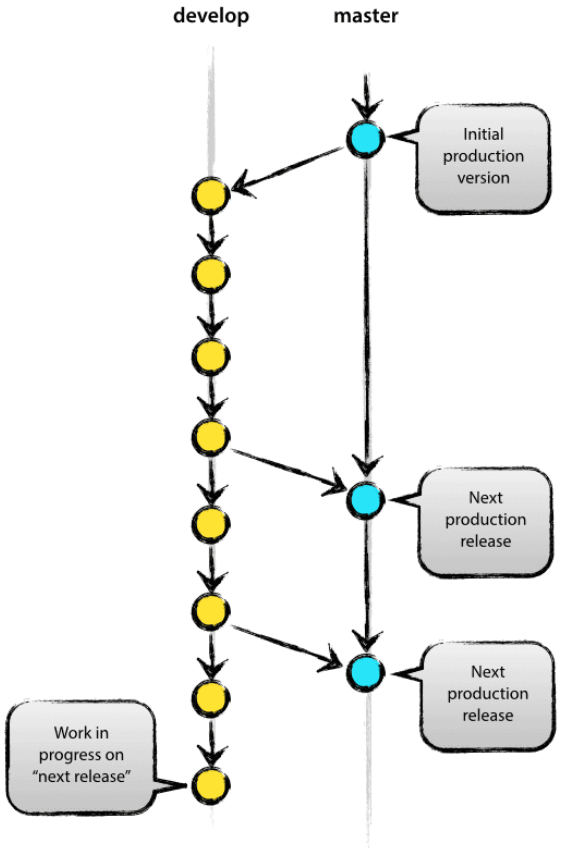
Ordenando Ramas



`git merge --no-ff`

`git merge`
(plain)



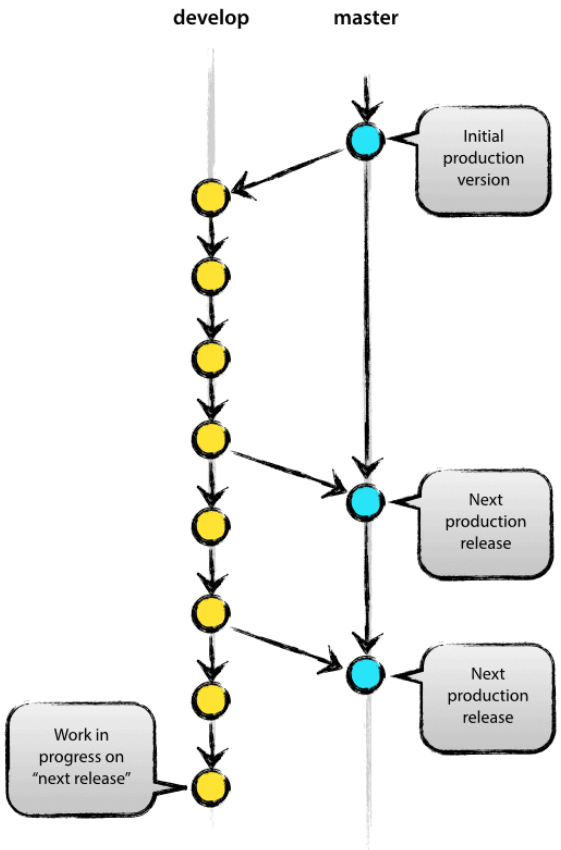


RAMA MASTER

```
# new repo
$ git init

# Clonando repo
$ git clone user@server:repo.git
```

- ★ Nombre: **master**
- ★ Perdura en el tiempo
- ★ No recibe commits
- ★ Sólo recibe **merge** de **release** y **hotfix**

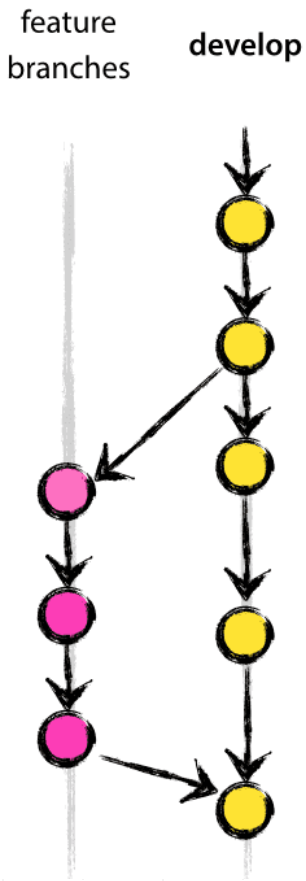
RAMA DEVELOP




```
# start develop branch
$ git checkout -b develop
```

- ★ Nombre: **develop**
- ★ Perdura en el tiempo
- ★ No recibe commits
- ★ recibe **merge** de **feature**
- ★ Hace **merge** a **release**

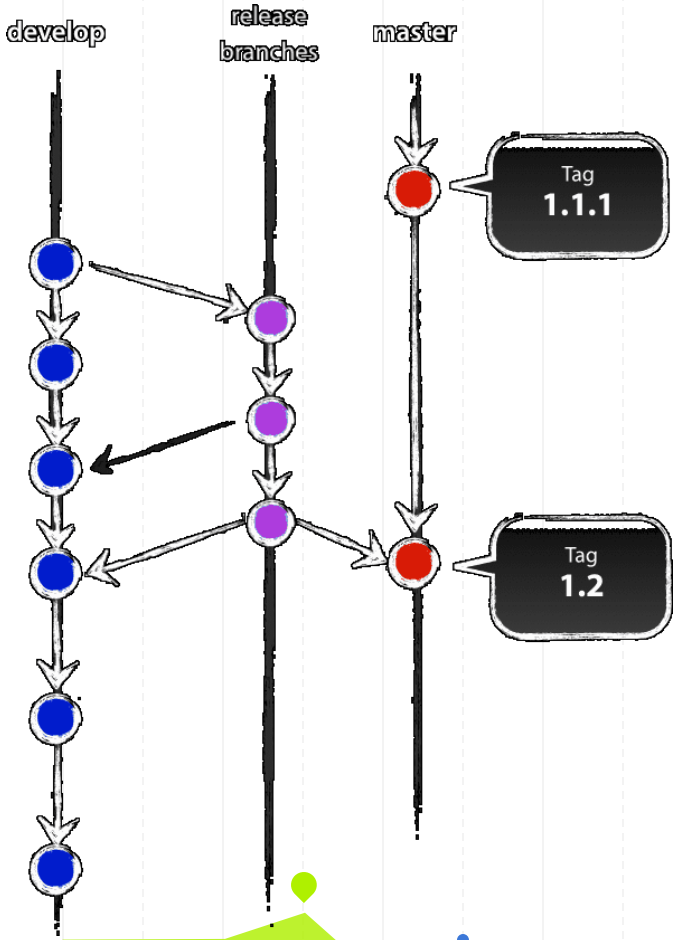
RAMA FEATURE




```
# Creación
git checkout -b feature-algo develop

# Finalización
git checkout develop
git merge --no-ff feature-algo
git branch -d feature-algo
git push origin develop
```

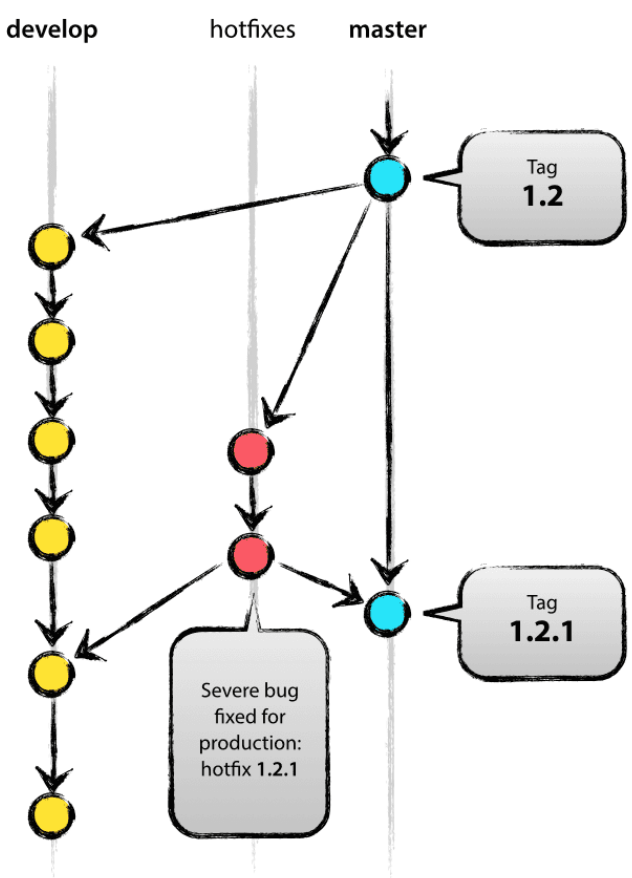
- ★ Nombre: **feature/xxx**
- ★ NO Perdura en el tiempo
- ★ recibe commits
- ★ hace **merge** a **develop**

RAMA RELEASE




```
# Creación
git checkout -b release/v2.3.0 develop

# Finalización
git checkout master
git merge --no-ff release/v2.3.0
git tag -a v2.3.0

git checkout develop
git merge --no-ff release/v2.3.0

git branch -d release/v2.3.0
```

- ★ Nombre: **release/xxx**
- ★ Perdura en el tiempo
- ★ No recibe commits (salvo pequeños fix)
- ★ hace **merge** a **master**



RAMA HOTFIXES

```
# Creación
git checkout -b hotfix/v2.3.7 master

# Finalización
git checkout master
git merge --no-ff hotfix/v2.3.7
git tag -a v2.3.7

git checkout develop
git merge --no-ff hotfix/v2.3.7

git branch -d hotfix/v2.3.7
```

- ★ Nombre: **hotfixes/xxx**
- ★ NO Perdura en el tiempo
- ★ Se origina de **master**
- ★ hace **merge** a **master** y **develop**

DESAFÍO #11: CLONE

```
# Abre tu cuenta de Github

# Crea un repositorio nuevo

# Copia la ruta

# Clona
$ git clone https://github.com/user/repo.git

# Revisa
$ git branch --all
```

REPOSITORIOS REMOTOS

```
# Recibiendo (bajando) repos
$ git pull [nombre_remoto][rama]
```

Recupera y une (merge) automáticamente la rama remota con tu rama actual (la rama en la cual te encuentras al momento de solicitar el pull).

```
$ git pull origin develop
```

```
# Enviando (subiendo) repos
$ git push [nombre_remoto] [rama]
```

```
$ git push origin rama
```

DESAFÍO #12: PUSH

```
# crea una rama feature/agregarIndex
$ git  checkout -b feature/agregarIndex

# Agrega un archivo
$ touch index.html (edítalo)

# confirma cambios
$ git add -A ,luego git commit -m “Mensaje”

# Haz checkout,merge y push
$ te quiero ver!
```

DESAFÍO #13: PULL

```
# Abre tu cuenta de Github

# En tu repo agrega un cambio directamente por GitHub

# Rescata los cambios
$ git pull origin tu_rama
```

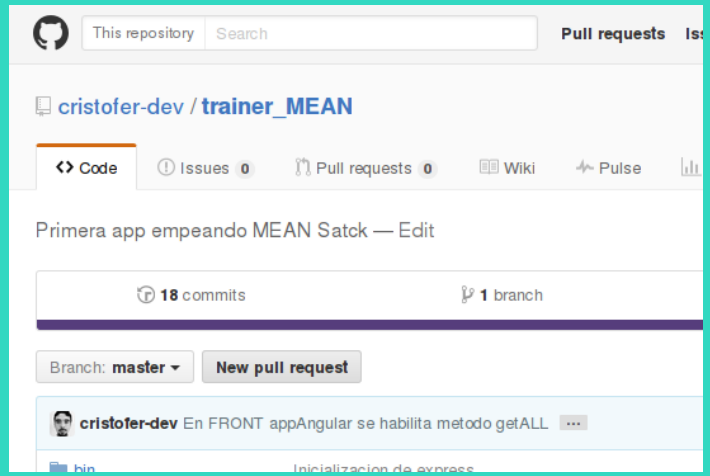
DESAFÍO #14: FEATURE/

```
# Clona el repo
$ git clone https://github.com/user/repo.git

# Crea una rama feature
$ git branch feature/tu_nombre

# Crea un archivo
$ touch tu_nombre

# Agregalo al Stage
$ git add nombre_archivo
```

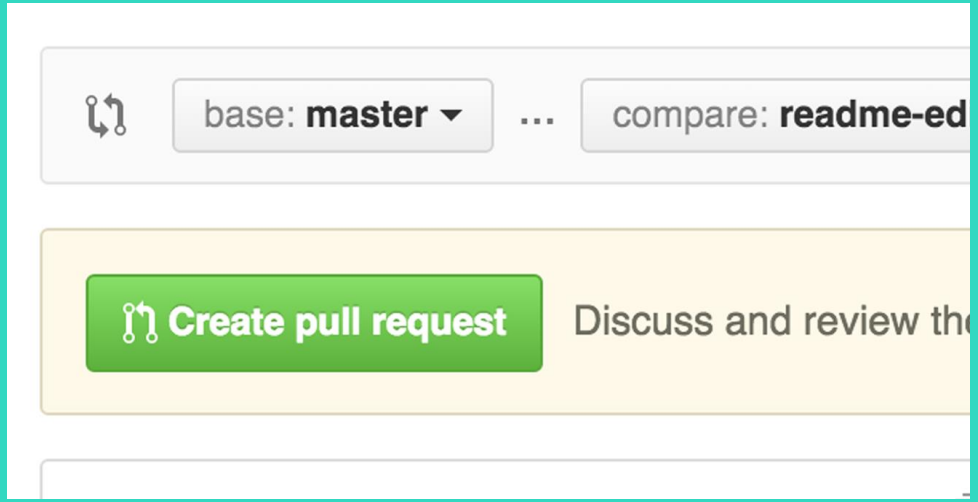
DESAFÍO #15: FEATURE/

```
# Revisa
$ git status

# Commit
$ git commit -m “ADD nombre_archivo”

# Subiendo
$ git push origin feature/tu_nombre
```

DESAFÍO #16: MERGE



DESAFÍO #17: MERGE



DESAFÍO #18: COLABORA

```
# Ve al repo de IVDEVS

# Busca workshop_git2

# Haz un fork

# … HOTFIX!! ISSUE #1

# Soluciona el problema como los dioses (GitFlow)

# Haz pull request
```

<IV DEVS>

LA BRUJULA
COWORK

fb.com/groups/programadores.iv.region.chile/

<IV DEVS>

http://slack-ivdevs.herokuapp.com/

Súmate

SITE www.**ivdevs**.com

@iv_devs

Programadores IV Región Chile

joinslack.**ivdevs**.com

fb.com/groups/programadores.iv.region.chile/

<IV DEVS>

http://slack-ivdevs.herokuapp.com/