DEV.F::

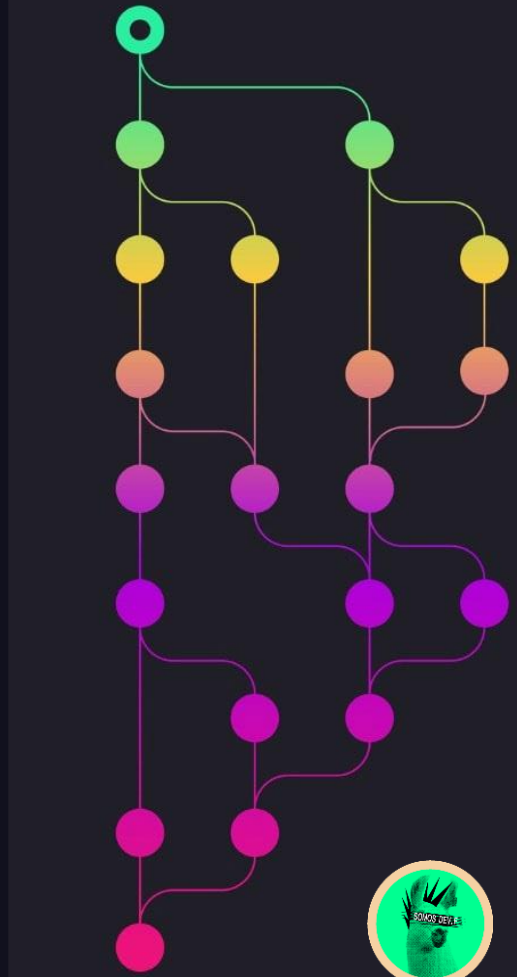
Git Flow

Enfoque para proyectos colaborativos

Fundamentos de GitHub

GitHub es una plataforma de alojamiento de repositorios Git que permite:

- Control de versiones del código.
- Colaboración en tiempo real.
- Seguimiento de problemas y gestión de proyectos.
- Automatización de tareas CI/CD.



Principales conceptos:

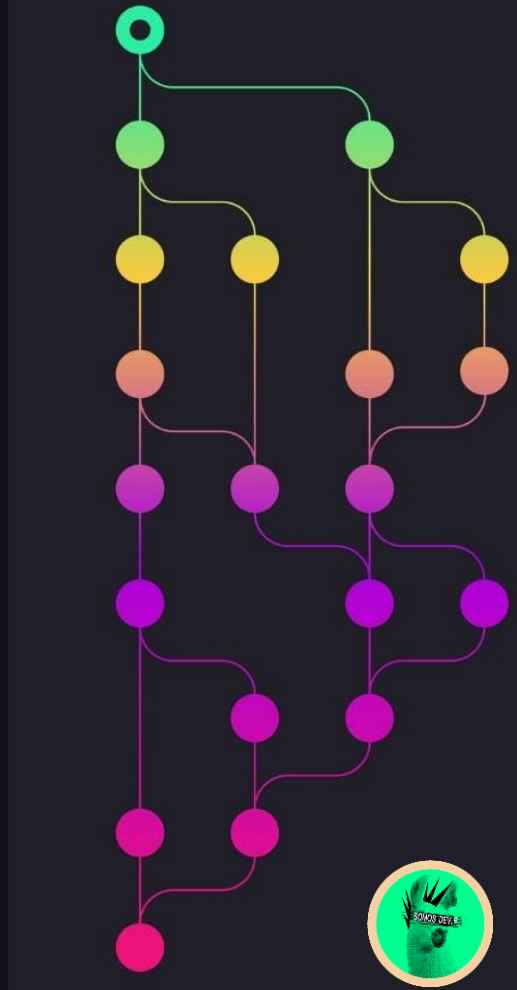
Repositorio:
Almacén de código donde se controlan cambios.

Branch:
Una rama donde se trabaja en una característica específica o corrección.

Pull Request (PR):
Solicitud para fusionar cambios de una rama a otra.

Issue:
Registro de tareas, errores o sugerencias de mejora.

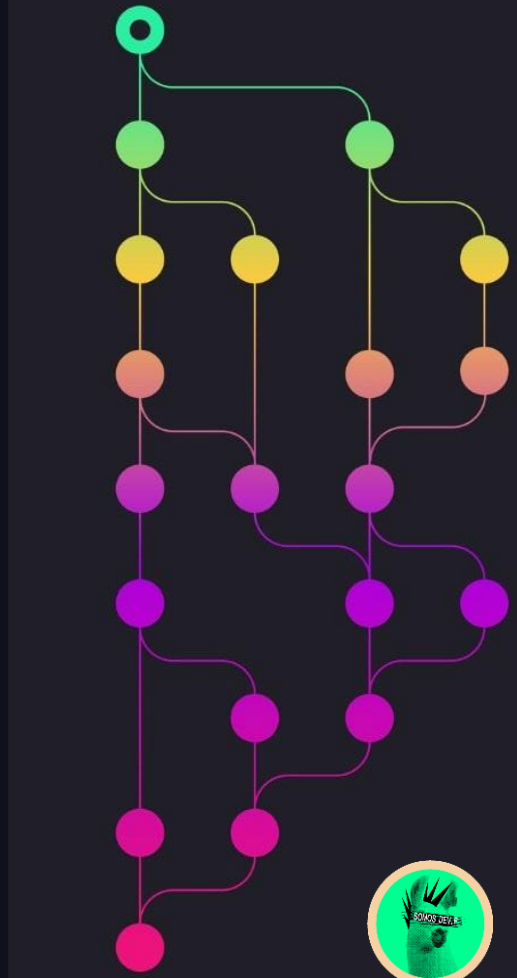
Fork:
Copia de un repositorio para trabajar en modificaciones antes de enviar cambios al original.



Comandos claves

```
# Clonar un repositorio

git clone <URL-del-repositorio>

# Ver ramas disponibles

git branch -a

# Crear y cambiar a una nueva rama

git checkout -b feature/nueva-funcionalidad

# Subir cambios a GitHub

git add .

git commit -m "Añadir nueva funcionalidad"

git push origin feature/nueva-funcionalidad
```

Visualizador git: https://git-school.github.io/visualizing-git/

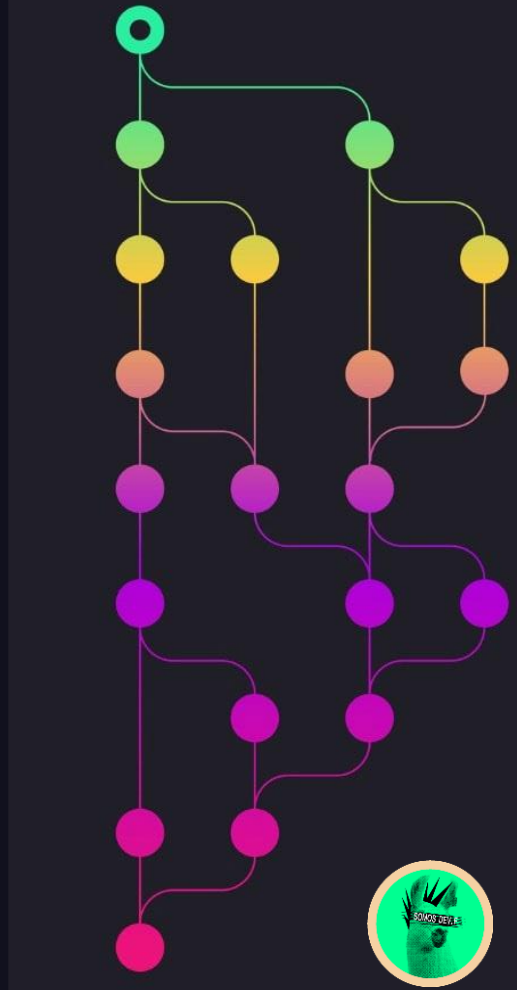


DEV.F::

¿Qué es Git Flow?

GitFlow es una estrategia de ramificación que define roles claros para cada rama y reglas de interacción.

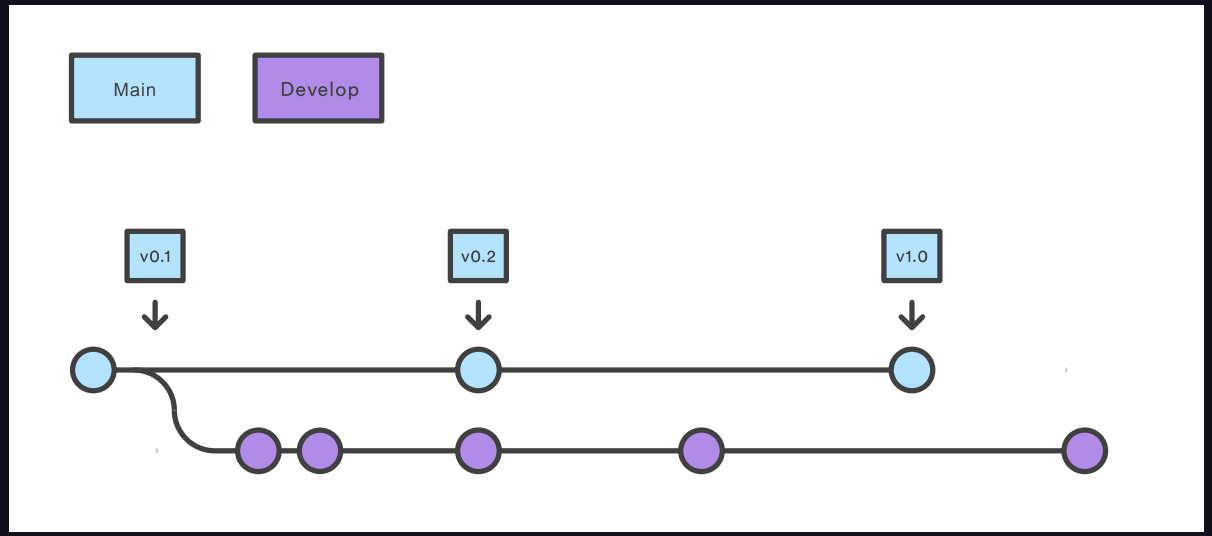
Objetivo: Facilitar el trabajo en equipo en proyectos complejos.



Ramas principales en GitFlow

Main (o master): Código en producción.

Develop: Código estable en desarrollo.



Flujo básico: main ← develop

DEV.F::

Ramas de soporte

Feature branches:

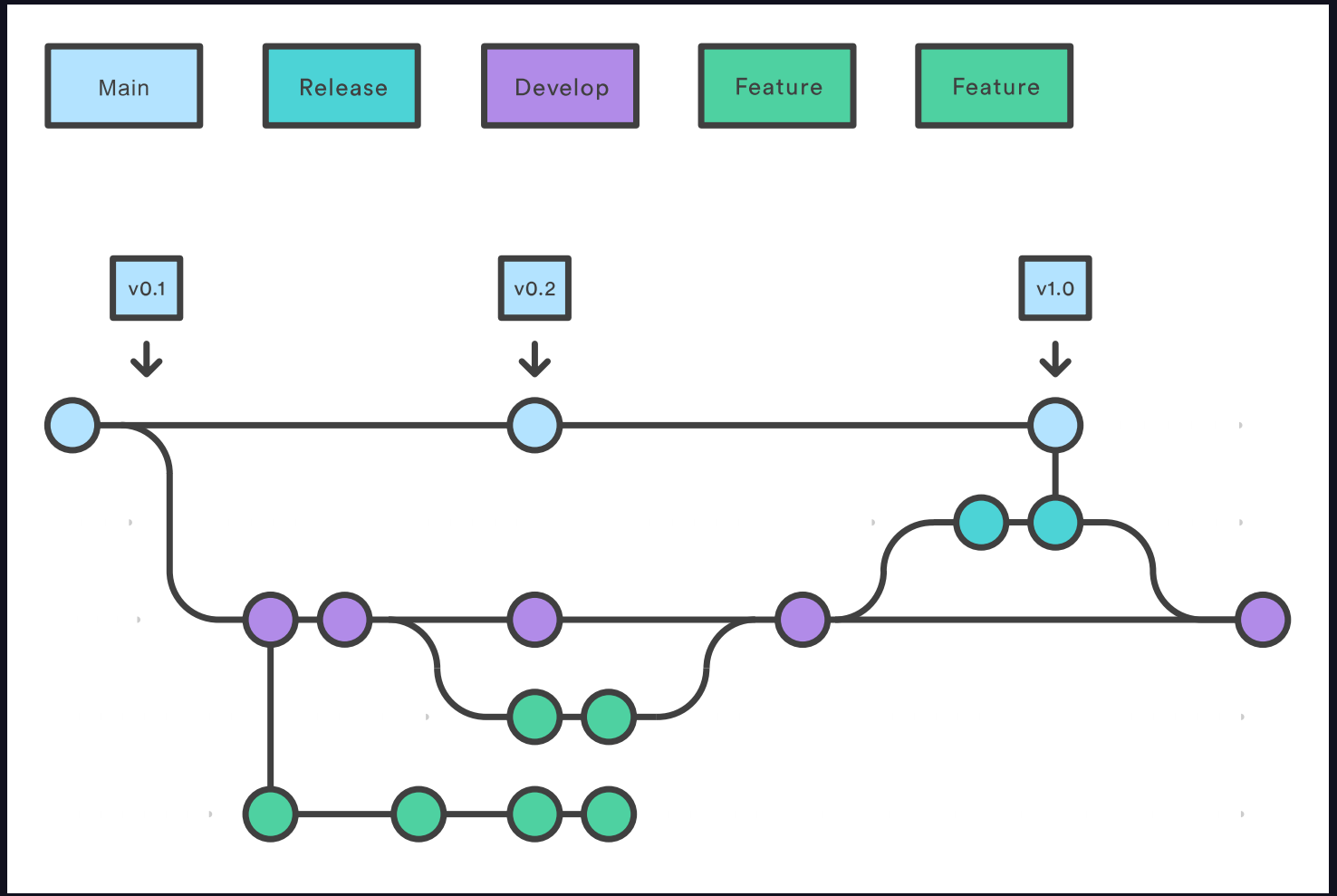
Para desarrollar nuevas funciones.

Release branches:

Preparar lanzamientos.

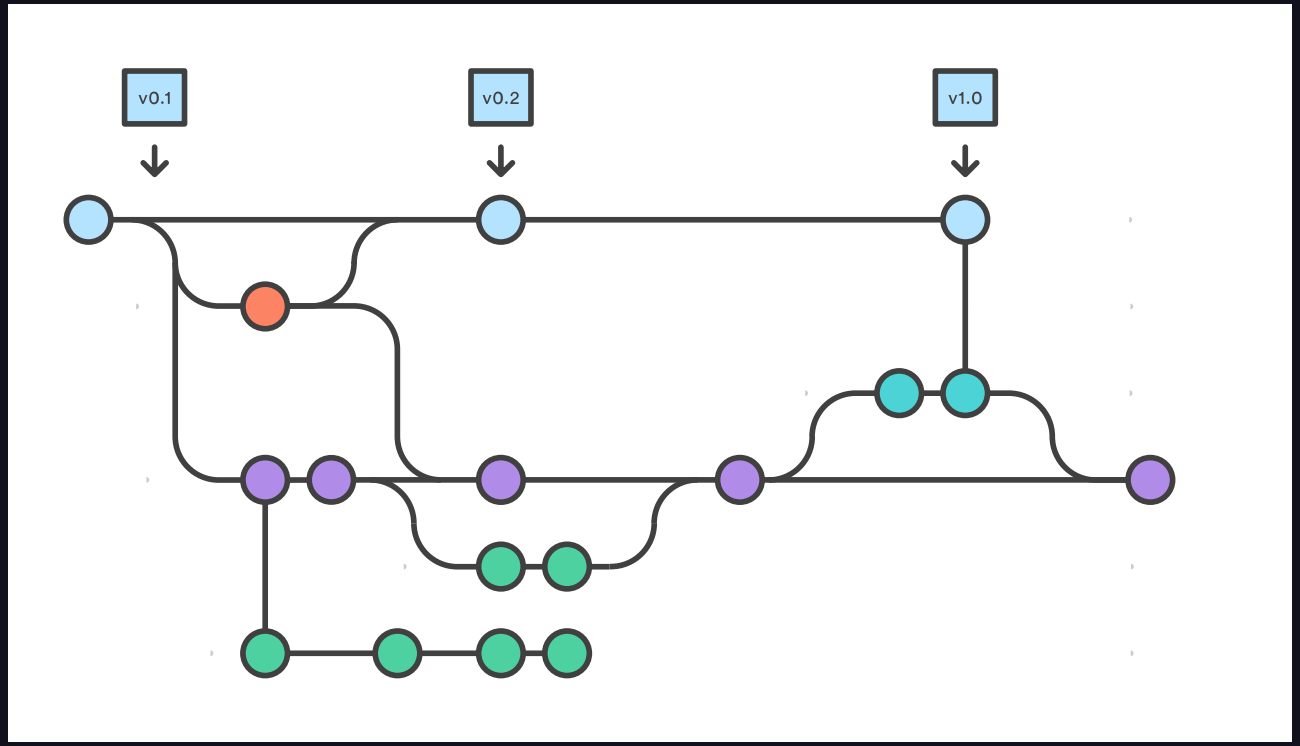
Hotfix branches:

Corregir errores críticos en producción.



DEV.F::

Flujo visual



Flujo de trabajo básico:

Crear una rama feature:

```
git checkout develop
git checkout -b feature/nueva-funcionalidad
```

Desarrollar y confirmar cambios:. Fusionar en develop:

```
git add .
git commit -m "Añadir nueva funcionalidad"
```

Desarrollar y confirmar cambios:

```
git checkout develop
git merge feature/nueva-funcionalidad
```

DEV.F::

Flujo de trabajo básico:

Crear una rama release para preparar versión:

```
git checkout -b release/v1.0.0 develop
```

Fusionar cambios de release en main:

```
git checkout main
git merge release/v1.0.0
git tag -a v1.0.0 -m "Versión 1.0.0"
```

Eliminar rama de release:

```
git branch -d release/v1.0.0
```

DEV.F::

Ventajas de GitFlow

- Organización clara del trabajo.
- Control en lanzamientos.
- Facilita trabajo en paralelo.
- Ideal para proyectos grandes con versiones planeadas.

Desventajas de GitFlow

- Más ramas → más complejidad.
- No es ideal para despliegues continuos.
- Más pasos para integrar cambios.

Comparativa con otros flujos

Característica	GitFlow	GitHub Flow	Trunk-based Development
Complejidad	Alta	Baja	Baja
Ramas principales	main, develop	main	main
Ciclo de release	Lento-planificado	Rápido-continuo	Muy rápido
Ideal para	Proyectos grandes	Startups, despliegue continuo	CI/CD extremo

Cuándo usar GitFlow

Úsalo cuando:

- El proyecto tiene lanzamientos planificados.
- Hay varios equipos trabajando en paralelo.
- La estabilidad del código es prioritaria.

Evítalo cuando:

- Hay despliegues continuos.
- El equipo es pequeño y necesita velocidad.

DEV.F::

Práctica

Colaborar en repositorio de github

La siguiente clase: Vamos a crear un repo y lo vamos a manipular desde nuestra computadora

Proyecto Final - Parte 2

En esta segunda parte deberás comenzar a definir la estructura del proyecto en el repositorio de GitHub. La estructura de carpetas y archivos es importante para que el proyecto sea fácil de mantener y escalar. Usa git y github para poder ir actualizando el proyecto de forma incremental. En esta parte del proyecto, deberás crear un archivo README.md (si aún no lo haz hecho) y con la descripción del proyecto y los acuerdos de la forma de trabajo. Además, deberás crear un archivo .gitignore para evitar subir archivos innecesarios al repositorio (vite ayuda con esta parte) y por último crear tu aplicación de react con la ayuda de vite.

Instrucciones para el entregable del proyecto:

1. Crear el proyecto con ayuda de la herramienta vite.
2. Definan la dinámica de trabajo con el repositorio si es que trabajan en equipo.
3. Practiquen la modificación y agregación de archivos del proyecto recién creado con ayuda de git y la actualización del repositorio remoto en Github.
4. Actualicen el repositorio con los nuevos commits y actualicen el repositorio remoto en github, esta última versión será la entrega de esta parte del proyecto.

DEV.F::