

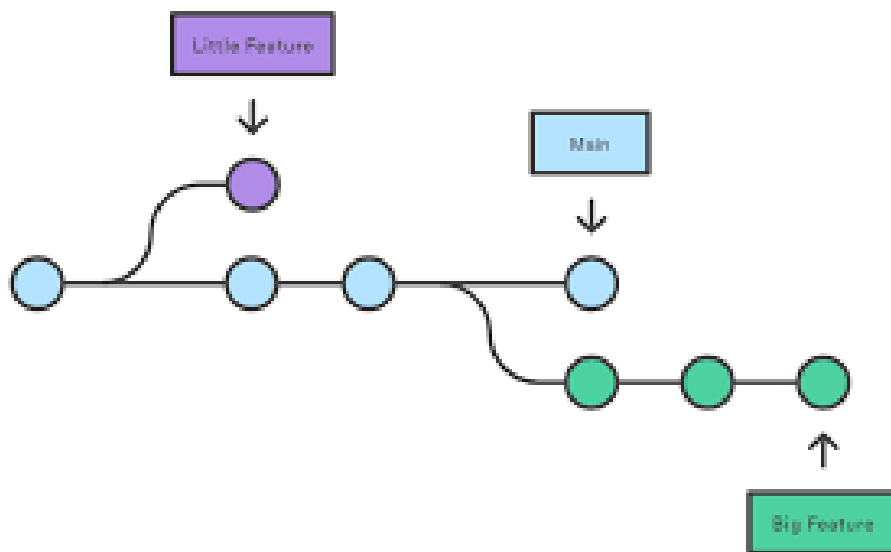
Contenidos de la unidad

- Flujo de trabajo
- Git avanzado
- Git Flow

Flujos de trabajo Github

Ramas

La creación de ramas es una función disponible en la mayoría de los sistemas de control de versiones actuales. La creación de ramas en otros sistemas de control de versiones puede tanto llevar mucho tiempo como ocupar mucho espacio de almacenamiento. En Git, las ramas son parte del proceso de desarrollo del día a día. Las ramas de Git son un puntero eficaz para las instantáneas de tus cambios. Cuando quieres añadir una nueva función o solucionar un error, independientemente de su tamaño, generas una nueva rama para alojar estos cambios. Esto hace que resulte más complicado que el código inestable se fusione con el código base principal, y te da la oportunidad de limpiar tu historial futuro antes de fusionarlo con la rama principal. Con git, la gestión de ramas se hace a través del comando `git branch`, las ramas de Git son una parte integral del flujo de trabajo diario.



Funcionamiento

Una rama representa una línea independiente de desarrollo. Las ramas sirven como una abstracción de los procesos de cambio, preparación y confirmación. Pueden generarse como una forma de solicitar un nuevo directorio de trabajo o un nuevo historial de proyecto. Las nuevas confirmaciones se registran en el historial de la rama actual, lo que crea una bifurcación en el historial del proyecto.

El comando `git branch` te permite crear, enumerar, cambiar el nombre y eliminar ramas. No te permite cambiar entre ramas o volver a unir un historial bifurcado. Por este motivo, `git branch` está estrechamente relacionado con los comandos `git checkout` y `git merge`.

Opciones comunes

El comando principal, sin parámetros, permite listar todas las ramas del repositorio `git branch`

Para crear una rama nueva se usa el comando como se muestra a continuación:

`git branch nombreRama`

Para eliminar una rama, es necesario agregar la opción -d

`git branch -d nombreRama`

Si se desea cambiar el nombre de una rama, se puede utilizar la opción -m, es importante mencionar que cambia el nombre de la rama sobre la cual se está trabajando.

`git branch -m nombreRama`

Finalmente, para listar todas las ramas en el repositorio remoto se puede utilizar el comando:

`git branch -a`

Gitflow

Gitflow es un flujo de trabajo de Git heredado que fue en origen una estrategia innovadora y revolucionaria para gestionar ramas de Git. Gitflow ha perdido popularidad en beneficio de los flujos de trabajo basados en troncos, que ahora se consideran prácticas recomendadas para el desarrollo continuo de software y las prácticas de DevOps. Además, puede ser difícil utilizar Gitflow con CI/CD. Esta entrada presenta el uso de Gitflow con fines de historial.

¿Qué es Gitflow?

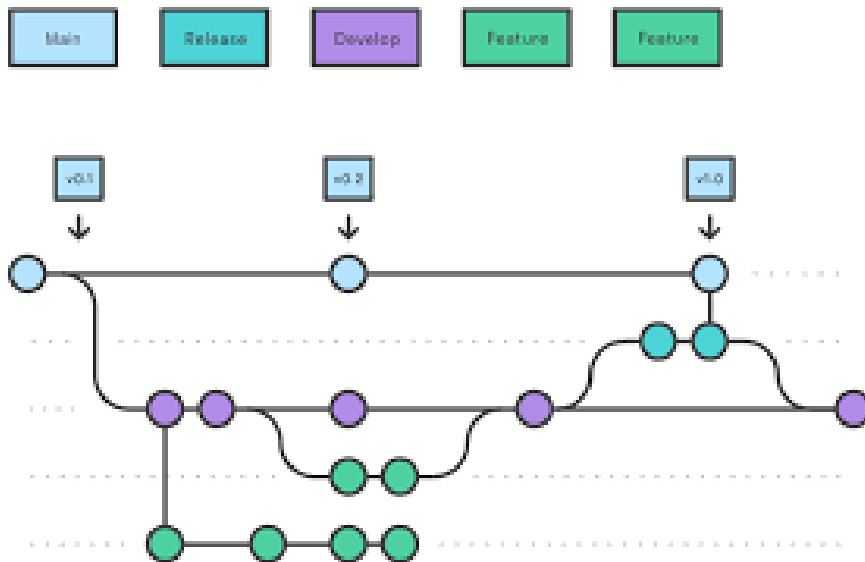
Gitflow es un modelo alternativo de creación de ramas en Git en el que se utilizan ramas de función y varias ramas principales. Fue Vincent Driessen en nvie quien lo publicó por primera vez y quien lo popularizó. En comparación con el desarrollo basado en troncos, Gitflow tiene diversas ramas de más duración y mayores confirmaciones. Según este modelo, los desarrolladores crean una rama de función y retrasan su fusión con la rama principal del tronco hasta que la función está completa. Estas ramas de función de larga duración requieren más colaboración para la fusión y tienen mayor riesgo de desviarse de la rama troncal. También pueden introducir actualizaciones conflictivas.

Gitflow puede utilizarse en proyectos que tienen un ciclo de publicación programado, así como para la práctica recomendada de DevOps de entrega continua. Este flujo de trabajo no añade ningún concepto o comando nuevo, aparte de los que se necesitan para el flujo de trabajo de ramas de función. Lo que hace es asignar funciones muy específicas a las distintas ramas y definir cómo y cuándo deben estas interactuar. Además de las ramas de función, utiliza ramas individuales para preparar, mantener y registrar publicaciones. Por supuesto, también puedes aprovechar todas las ventajas que aporta el flujo de trabajo de ramas de función: solicitudes de incorporación de cambios, experimentos aislados y una colaboración más eficaz.

Inicio

En realidad, Gitflow es una especie de idea abstracta de un flujo de trabajo de Git. Esto quiere decir que ordena qué tipo de ramas se deben configurar y cómo fusionarlas. El conjunto de herramientas de git-flow es una herramienta de línea de comandos propiamente dicha que tiene un proceso de instalación. El proceso de instalación de git-flow es sencillo. Los paquetes de git-flow están disponibles en varios sistemas operativos. En Windows, tendrás que descargar e instalar git-flow. Después de instalar git-flow, puedes utilizarlo en tu proyecto ejecutando git flow init. Git-flow es un contenedor para Git. El comando git flow init es una prolongación del comando predeterminado git init y no cambia nada de tu repositorio aparte de crear ramas.

Funcionamiento



Ramas principales y de desarrollo

En lugar de una única rama main, este flujo de trabajo utiliza dos ramas para registrar el historial del proyecto. La rama main o principal almacena el historial de publicación oficial y la rama develop o de desarrollo sirve como rama de integración para las funciones. Asimismo, conviene etiquetar todas las confirmaciones de la rama main con un número de versión.

El primer paso es complementar la rama main predeterminada con una rama develop. Una forma sencilla de hacerlo es que un desarrollador cree de forma local una rama develop vacía y la envíe al servidor:

```
git branch develop
```

```
git push -u origin develop
```

Esta rama contendrá el historial completo del proyecto, mientras que main contendrá una versión abreviada. A continuación, otros desarrolladores deberían clonar el repositorio central y crear una rama de seguimiento para develop.

A la hora de utilizar la biblioteca de extensiones de git-flow, ejecutar `git flow init` en un repositorio existente creará la rama develop:

```
$ git flow init
```

Initialized empty Git repository in ~/project/.git/

No branches exist yet. Base branches must be created now.

Branch name for production releases: [main]

Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?

Feature branches? [feature/]

Release branches? [release/]

Hotfix branches? [hotfix/]

Support branches? [support/]

Version tag prefix? []

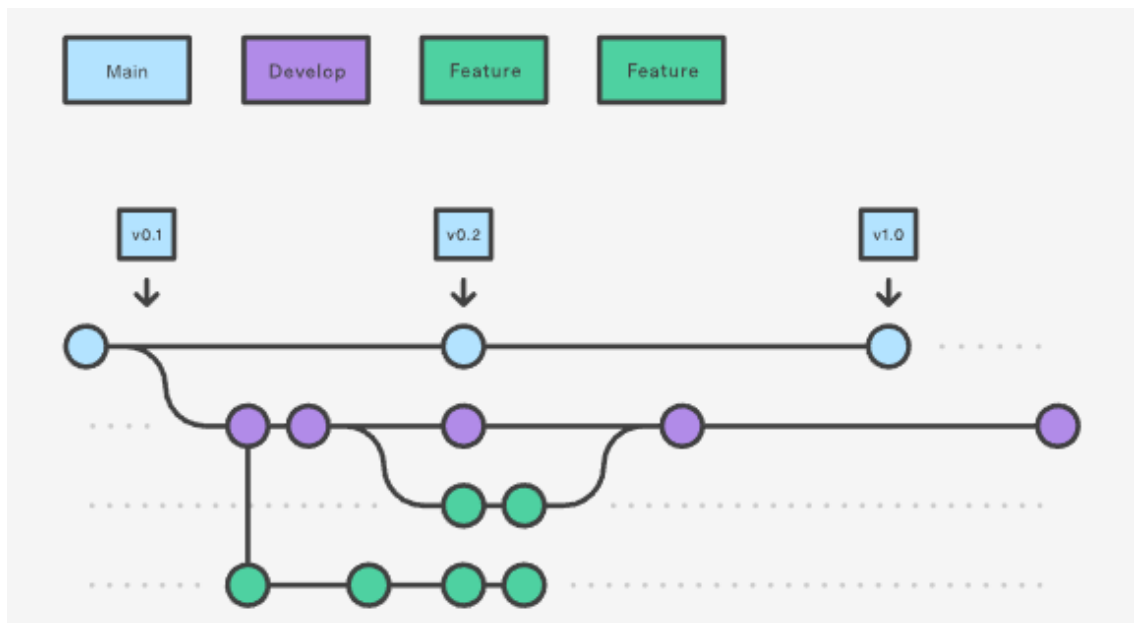
\$ git branch

* develop

main

Ramas de función

Todas las funciones nuevas deben residir en su propia rama, que se puede enviar al repositorio central para copia de seguridad/colaboración. Sin embargo, en lugar de ramificarse de main, las ramas feature utilizan la rama develop como rama primaria. Cuando una función está terminada, se vuelve a fusionar en develop. Las funciones no deben interactuar nunca directamente con main.



Ten en cuenta que las ramas feature combinadas con la rama develop conforman, a todos efectos, el flujo de trabajo de ramas de función. Sin embargo, el flujo de trabajo Gitflow no termina aquí.

Las ramas feature suelen crearse a partir de la última rama develop.

Creación de una rama de función

Sin las extensiones de git-flow:

```
git checkout develop
```

```
git checkout -b feature_branch
```

Cuando se utiliza la extensión de git-flow:

```
git flow feature start feature_branch
```

Sigue trabajando y utiliza Git como lo harías normalmente.

Finalización de una rama de función

Cuando hayas terminado con el trabajo de desarrollo en la función, el siguiente paso es fusionar feature_branch en develop.

Sin las extensiones de git-flow:

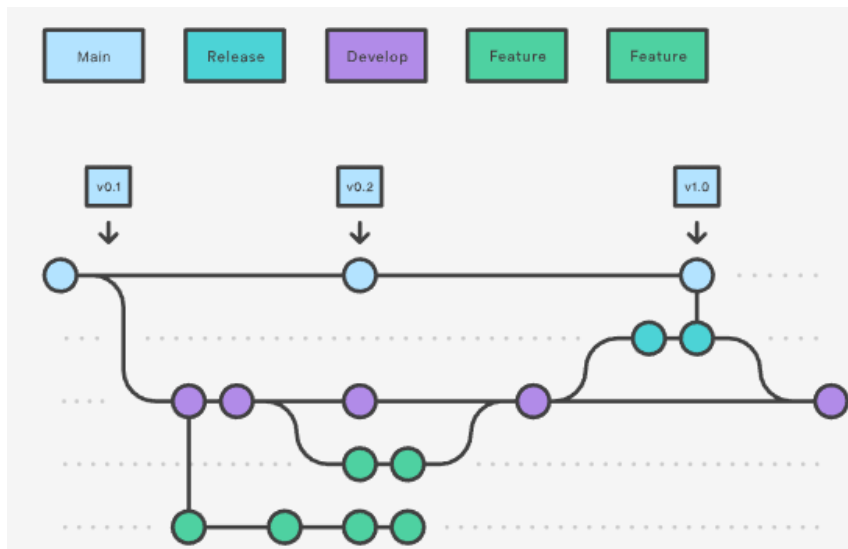
```
git checkout develop
```

```
git merge feature_branch
```

Con las extensiones de git-flow:

```
git flow feature finish feature_branch
```

Ramas de publicación



Cuando develop haya adquirido suficientes funciones para una publicación (o se acerque una fecha de publicación predeterminada), debes bifurcar una rama release (o de publicación) a partir de develop. Al crear esta rama, se inicia el siguiente ciclo de publicación, por lo que no pueden añadirse nuevas funciones una vez pasado este punto (en esta rama solo deben producirse las soluciones de errores, la generación de documentación y otras tareas orientadas a la publicación). Cuando está lista para el lanzamiento, la rama release se fusiona en main y se etiqueta con un número de versión. Además, debería volver a fusionarse en develop, ya que esta podría haber progresado desde que se iniciara la publicación.

Utilizar una rama específica para preparar publicaciones hace posible que un equipo perfeccione la publicación actual mientras otro equipo sigue trabajando en las funciones para la siguiente publicación. Asimismo, crea fases de desarrollo bien definidas (por ejemplo, es fácil decir: "Esta semana nos estamos preparando para la versión 4.0" y verlo escrito en la estructura del repositorio).

Crear ramas release es otra operación de ramificación sencilla. Al igual que las ramas feature, las ramas release se basan en la rama develop. Se puede crear una nueva rama release utilizando los siguientes métodos.

Sin las extensiones de git-flow:

```
git checkout develop
```

```
git checkout -b release/0.1.0
```

Cuando se utilizan las extensiones de git-flow:

```
$ git flow release start 0.1.0
```

```
Switched to a new branch 'release/0.1.0'
```


En cuanto la publicación esté lista para su lanzamiento, se fusionará en las ramas main y develop, y luego se eliminará la rama release. Es importante volver a fusionarla en la rama develop porque podrían haberse añadido actualizaciones importantes a la rama release, y las funciones nuevas tienen que poder acceder a ellas. Si en tu organización se da mucha importancia a la revisión de código, este sería el lugar ideal para una solicitud de incorporación de cambios.

Para finalizar una rama release, utiliza los siguientes métodos:

Sin las extensiones de git-flow:

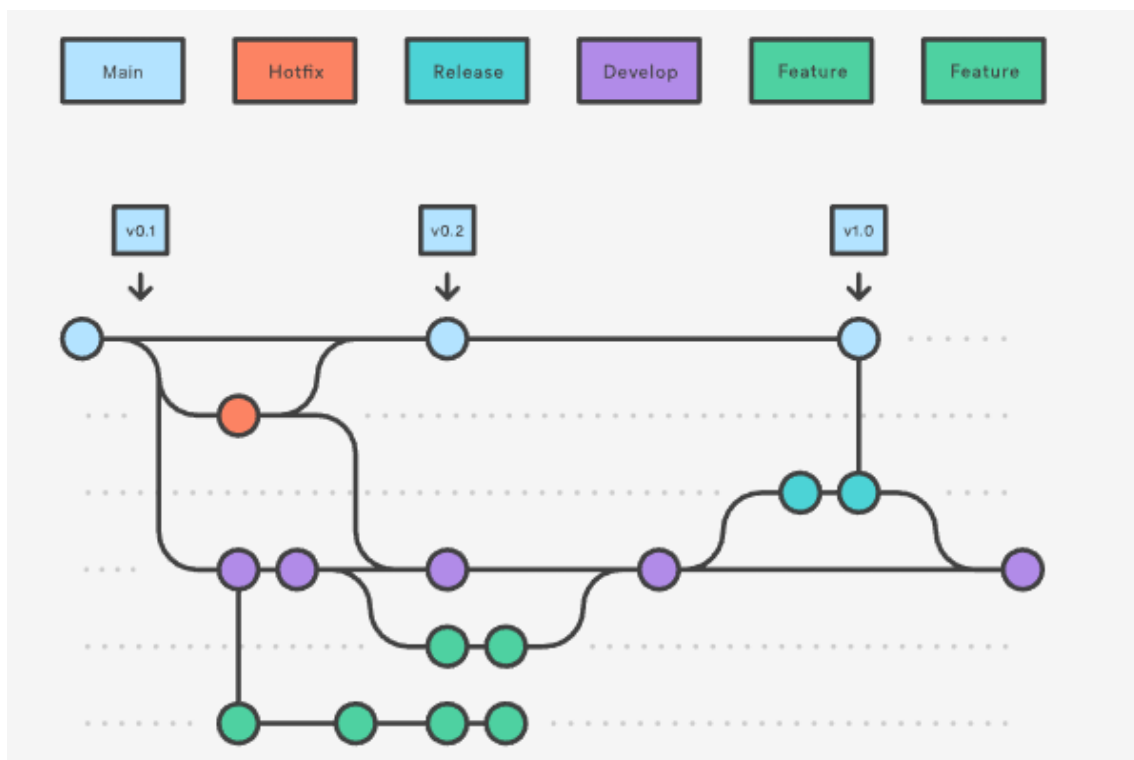
```
git checkout main
```

```
git merge release/0.1.0
```

O con la extensión de git-flow:

```
git flow release finish '0.1.0'
```

Ramas de corrección



Las ramas de mantenimiento, de corrección o de hotfix sirven para reparar rápidamente las publicaciones de producción. Las ramas hotfix son muy similares a las ramas release y feature, salvo por el hecho de que se basan en la rama main y no en la develop. Esta es la única rama que debería bifurcarse directamente a partir de main.

Cuando se haya terminado de aplicar la corrección, debería fusionarse en main y develop (o la rama release actual), y main debería etiquetarse con un número de versión actualizado.

Tener una línea de desarrollo específica para la corrección de errores permite que tu equipo aborde las incidencias sin interrumpir el resto del flujo de trabajo ni esperar al siguiente ciclo de publicación. Puedes concebir las ramas de mantenimiento como ramas release ad hoc que trabajan directamente con la rama main. Se puede crear una nueva rama hotfix utilizando los siguientes métodos:

Sin las extensiones de git-flow:

```
git checkout main
```

```
git checkout -b hotfix_branch
```

Cuando se utilizan las extensiones de git-flow:

```
$ git flow hotfix start hotfix_branch
```

Al igual que al finalizar una rama release, una rama hotfix se fusiona tanto en main como en develop.

```
git checkout main
```

```
git merge hotfix_branch
```

```
git checkout develop
```

```
git merge hotfix_branch
```

```
git branch -D hotfix_branch
```

```
$ git flow hotfix finish hotfix_branch
```