Trabajo de Investigación

**Alumna**: Analia Noemi Dressler Lang

**Actividad Académica**

Investigar sobre acciones que se realiza con un versionador de código:  
Explica ¿En qué consiste un...?

1. **Commit** El comando git commit captura una instantánea de los cambios preparados en ese momento del proyecto. Las instantáneas confirmadas pueden considerarse como versiones "seguras" de un proyecto: Git no las cambiará nunca a no ser que se lo pidas expresamente. Antes de ejecutar git commit, se utiliza el comando “[git add](https://www.atlassian.com/es/git/tutorials/saving-changes)” para pasar o "preparar" los cambios en el proyecto que se almacenarán en una confirmación. Estos dos comandos, git commit y git add, son dos de los que se utilizan más frecuentemente.

A grandes rasgos, a Git se puede concebir como una utilidad de gestión de cronogramas. Las confirmaciones son las unidades básicas más importantes de un cronograma de proyecto de Git. Las confirmaciones se pueden concebir como instantáneas o hitos en el cronograma de un proyecto de Git. Las confirmaciones se crean con el comando git commit para capturar el estado de un proyecto en ese determinado momento. Las instantáneas de Git siempre se confirman en el repositorio local. Este enfoque es diametralmente opuesto al de SVN, donde la copia de trabajo se confirma en el repositorio central. Por el contrario, Git no te obliga a interactuar con el repositorio central hasta que estés listo. Del mismo modo que el entorno de ensayo es una zona intermedia entre el directorio de trabajo y el historial del proyecto, el repositorio local de cada desarrollador es una zona intermedia entre sus contribuciones y el repositorio central.

1. **Push**

El **push()**método agrega uno o más elementos al final de una matriz y devuelve la nueva longitud de la matriz.

El pushmétodo agrega valores a una matriz.

pushes intencionalmente genérico. Este método se puede utilizar con [call()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function/call)o [apply()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function/apply)sobre objetos que se asemejen a matrices. El pushmétodo se basa en una length propiedad para determinar dónde comenzar a insertar los valores dados. Si la lengthpropiedad no se puede convertir en un número, el índice utilizado es 0. Esto incluye la posibilidad de lengthque no exista, en cuyo caso lengthtambién se creará.

Aunque las [cadenas](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String) son objetos nativos de tipo Array, no son adecuados en aplicaciones de este método, ya que las cadenas son inmutables. De manera similar para los [argumentos](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/arguments) nativos de objetos tipo Array

1. **Pull**

Las pull requests son una funcionalidad que facilita la colaboración entre desarrolladores que usan [Bitbucket](http://www.bitbucket.org/). Ofrecen una interfaz web intuitiva para debatir los cambios propuestos antes de integrarlos en el proyecto oficial.

En su forma más simple, las pull requests son un mecanismo para que los desarrolladores notifiquen a los miembros de su equipo que han terminado una funcionalidad. Una vez la rama de funcionalidades está lista, el desarrollador realiza la pull request mediante su cuenta de Bitbucket. Así, todas las personas involucradas saben que deben revisar el código y fusionarlo con la rama master.

Las pull requests se pueden usar junto con [Feature Branch Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow) (Workflow de ramas de funcionalidades), [Gitflow Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow) (Workflow de Gitflow) o [Forking Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow) (Workflow de bifurcación). Sin embargo, como las pull requests requieren o bien dos ramas distintas, o bien dos repositorios distintos, no funcionan con [Centralized Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows/centralized-workflow) (Workflow centralizado). El uso de pull requests con cada uno de estos workflows es ligeramente distinto, pero el proceso general es el siguiente:

1. Un desarrollador crea una función en una rama especializada de su repositorio local.
2. El desarrollador envía la rama a un repositorio de Bitbucket público.
3. El desarrollador envía una solicitud de incorporación de cambios mediante Bitbucket.
4. El resto del equipo revisa el código, debate sobre él y aplica modificaciones.
5. El responsable del mantenimiento del proyecto fusiona la función con el repositorio oficial y cierra la solicitud de incorporación de cambios.

El resto de esta sección describe cómo se puede sacar partido de las solicitudes de incorporación de cambios en flujos de trabajo de colaboración diferentes.

1. **Branch**

Este documento constituye una revisión en profundidad del comando git branch y una exposición general del modelo de creación de ramas de Git. La creación de ramas es una función disponible en la mayoría de los sistemas de control de versiones modernos. 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 diario. 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.

Funcionamiento

Una rama representa una línea independiente de desarrollo. Las ramas sirven como una abstracción de los procesos de cambio, preparación y confirmación. Puedes concebirlas como una forma de solicitar un nuevo directorio de trabajo, un nuevo entorno de ensayo 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](https://www.atlassian.com/git/tutorials/using-branches/git-checkout) y [git merge](https://www.atlassian.com/git/tutorials/using-branches/git-merge).

1. **Fetch**

La API Fetch proporciona una interfaz JavaScript para acceder y manipular partes del canal HTTP, tales como peticiones y respuestas. También provee un método global fetch() (en-US) que proporciona una forma fácil y lógica de obtener recursos de forma asíncrona por la red.

Este tipo de funcionalidad se conseguía previamente haciendo uso de XMLHttpRequest. Fetch proporciona una alternativa mejor que puede ser empleada fácilmente por otras tecnologías como Service Workers (en-US). Fetch también aporta un único lugar lógico en el que definir otros conceptos relacionados con HTTP como CORS y extensiones para HTTP.

1. **Merge**

Incorpora cambios de las confirmaciones nombradas (desde el momento en que sus historiales divergieron de la rama actual) a la rama actual. Este comando lo usa git pull para incorporar cambios de otro repositorio y se puede usar manualmente para fusionar cambios de una rama a otra.

Suponga que existe el siguiente historial y que la rama actual es " master":

Luego, " git merge topic" reproducirá los cambios realizados en la topicrama desde que divergió desde master(es decir, E) hasta su confirmación actual ( C) mastery registrará el resultado en una nueva confirmación junto con los nombres de las dos confirmaciones principales y un mensaje de registro. del usuario que describe los cambios.

Tema A --- B --- C

/ \

D --- E --- F --- G --- H maestro

Bibliografía

1. <https://www.atlassian.com/es/git/tutorials/saving-changes/git-commit>
2. <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/push>
3. <https://www.atlassian.com/es/git/tutorials/making-a-pull-request>
4. <https://www.atlassian.com/es/git/tutorials/using-branches>
5. <https://developer.mozilla.org/es/docs/Web/API/Fetch_API/Using_Fetch>
6. <https://git-scm.com/docs/git-merge>