Git: mooc_git-entrega2_rebase

Objetivo

Continuar practicando con repositorios locales y remotos, commits y ramas, pero integrando con rebase.

Resumen de la práctica

Reutilizar la cuenta y el repositorio my_calculator en GitHub tal y como quedó en la entrega 1. Crear una nueva rama que parta del primer commit de master "x^3 button" de nombre ops. Crear dos commits en la nueva rama ops, el primero añade el botón x^2 y el segundo con el botón 1/x. Integrar la nueva rama ops en la rama master utilizando "git rebase .....".

Para terminar se deben subir los nuevos commits integrados en la rama master a un nuevo repositorio en su cuenta de GitHub, denominado my_calculator_2.

El grafo de commits es más fácil de seguir cuando se integra con rebase, ya que todos los commits de la rama lateral pasan a master y todo queda en master. En cambio se pierde algo de la historia del proyecto, porque desaparecen las ramas laterales donde se suelen desarrollar las nuevas funcionalidades. Aunque no lo vamos a ver hasta la entrega 3, rebase tiene además la opción interactiva que permite rehacer la rama que se integra y sus commits. Por esta razón hay personas que prefieren el rebase frente al merge para integrar desarrollos.

¡Cuidado! Las ramas que se hayan compartido con terceros no deben modificarse con rebase, porque si un tercero ha descargado la rama, deberá repetir el rebase en su repositorio local. Lo recomendado habría sido hacer un rebase de **master** a **ops**, para luego hacer un merge, de tipo fast forward, a **master**.

Prueba de la práctica

Para comprobar que la práctica ha sido realizada correctamente hay que utilizar el validador de este repositorio

https://github.com/practicas-ging/mooc_git-entrega2_rebase1

Recuerde que para utilizar el validador se debe tener node.js (y npm) (https://nodejs.org/es/) y Git instalados. El proyecto se descarga, instala y ejecuta en el ordenador local con estos comandos:

```
$                                                ## El proyecto debe clonarse en el ordenador local
$ git clone https://github.com/practicas-ging/mooc_git-entrega2_rebase1
$
$ cd  mooc_git-entrega2_rebase1                  ## Entrar en el directorio de trabajo
$
$ npm install                                    ## Instala el programa de test
$
$ npm run checks  [nombre_de_la_cuenta]          ## Pasa los tests sobre el repositorio en github
.....                                            ## indicando que partes  están correctamente
.....                                            ## implementadas y cuales no.
... (resultado de los tests)
$
```

Debe cambiar [nombre_de_la_cuenta] por el nombre de su cuenta en GitHub.

Instrucciones para la Entrega y Evaluación

Una vez finalizado el desarrollo debe entregar en MiriadaX solo el **nombre de su cuenta** en GitHub donde ha subido el repositorio my_calculator_2.

¡Cuidado! Compruebe que el nombre de la cuenta subido es el correcto y que el repositorio esta actualizado con la última versión.

El evaluador debe comprobar que la entrega es correcta buscando en GitHub el nombre de cuenta entregado y comprobando que contiene el repositorio pedido con las características solicitadas.

RÚBRICA: La resolución de cada uno de estos puntos dará un el % indicado de la nota total:
- 10%: Existe el repositorio my_calculator
- 30%: Los tres primeros commits de master son los originales: "x^3 button", "x^4 button" y "sin(x) button"
- 30%: El cuarto commit de la rama master es "x^2 button" y contiene lo pedido
- 30%: El quinto commit de la rama master es "1/x button" y contiene lo pedido

El objetivo de este curso es sacar el máximo provecho al trabajo que están dedicando, por lo que les recomendamos que utilicen la evaluación para ayudar a sus compañeros enviando comentarios sobre la corrección del código, su claridad, legibilidad, estructuración y documentación.

Dado que es un curso para principiantes, ante la duda les pedimos que sean benevolentes con sus compañeros, porque muchos participantes están empezando y los primeros pasos siempre son difíciles.

¡Cuidado! Una vez enviada la evaluación, esta no se puede cambiar. Piensen bien su evaluación antes de enviarla.

Pasos a seguir en el desarrollo de la práctica

Paso 1) Clonar en un repositorio local el repositorio my_calculator de GitHub en un directorio my_calculator_2. Clonar cuando my_calculator contenga el desarrollo completo de la entrega 1.

Sugerencia de comandos a utilizar

```
$ git clone  https://github.com/<su _cuenta>/my_calculator    my_calculator_2
$
```

Paso 2) Crear una rama de nombre **ops** que comience después del primer commit (con mensaje "x^3 button") de la rama **master** y restaurarla en el directorio de trabajo, para poder trabajar sobre ella.

Sugerencia de comandos a utilizar:

```
$ git branch -v                          # muestra las ramas existentes en el repositorio
$ git checkout -b ops <id_de_commit>  # Crea rama en commit indicado y realiza checkout a rama
```

Paso 3) Crear un commit en la rama **ops**, que añada a la calculadora del fichero calculator.html el botón x^2 que eleve un número al cuadrado.

Una vez añadido el código del nuevo nuevo botón a la calculadora, comprobar que funciona correctamente, registrar los cambios en el índice y crear el nuevo commit.

Sugerencia de comandos a utilizar:
$ git branch -v # muestra las ramas existentes en el repositorio
$ git status -s # muestra el estado de los ficheros respecto al índice
$ git add <fichero1> <fichero2> … # añadir ficheros al índice
$ git commit -m "x^2 button" # cierra el commit, añadiéndole el mensaje indicado

También se pueden utilizar
$ git add . # registra todos los fich. nuevos o mod. en índice, Peligroso! usar con cuidado
$ git log --oneline # muestra la historia de commits en formato corto
$ git log --oneline --all # Para mostrar la historia de todas las ramas en formato corto
$ git diff …. # muestra las diferencias con el commit anterior antes y después de cerrarlo

Paso 4) Crear otro commit más en la rama **ops**, que añada a la calculadora del fichero calculator.html el botón 1/x que calcule el inverso.

Una vez añadido el código del nuevo nuevo botón a la calculadora y después de comprobar que funciona correctamente, registrar los cambios en el índice y crear el nuevo commit.

Sugerencia de comandos a utilizar:
$ git branch -v # muestra las ramas existentes en el repositorio
$ git status -s # muestra el estado de los ficheros respecto al índice
$ git add <fichero1> <fichero2> … # añadir ficheros al índice
$ git commit -m "1/x button" # cierra el commit, añadiéndole el mensaje indicado

También se pueden utilizar
$ git add . # registra todos los fich. nuevos o mod. en índice, Peligroso! usar con cuidado
$ git log --oneline # muestra la historia de commits en formato corto
$ git log --oneline --all # Para mostrar la historia de todas las ramas en formato corto
$ git diff …. # muestra las diferencias con el commit anterior antes y después de cerrarlo

Paso 5) Integrar la rama **ops** en la rama **master** con "git rebase" para crear una calculadora con cuatro botones: x^2, x^3, x^4, sin(x) y 1/x.

"git rebase …" realiza la integración ejecutando un bucle, donde cada iteración traslada un commit de la rama origen a su nueva base. El traslado implica integrar el código del commit con el de su nueva base. Si la integración tiene conflictos, git indica el error y finaliza.

"git status" muestra los ficheros con conflictos. Los conflictos deben resolverse entonces con el editor.

Una vez resueltos, se debe comprobar primero que la integración funciona correctamente. Después se debe continuar la integración (rebase) añadiendo los cambios al índice y continuando el rebase con "git rebase --continue".

Una vez generado un commit, git pasa a intentar integrar el siguiente de la rama origen. Y así hasta el último de la rama origen.

Al acabar el rebase la rama origen habrá desaparecido.

Sugerencia de comandos a utilizar:
$ git rebase <commit_id> # rehace la rama en la que estamos a partir de <commit_id>
$ git status -s # permite ver los ficheros con conflictos
$ git add <fichero1> <fichero2> … # añadir ficheros al índice
$ git rebase --continue # continua el re base después de resolver un conflicto
  # antes resolverse con el editor los conflictos y añadir los cambios al índice

También se pueden utilizar
$ git branch -v # muestra las ramas existentes en el repositorio
$ git add . # registra todos los fich. nuevos o mod. en índice, Peligroso! usar con cuidado
$ git log --oneline # muestra la historia de commits en formato corto
$ git log --oneline --all # Para mostrar la historia de todas las ramas en formato corto
$ git diff …. # muestra las diferencias con el commit anterior antes y después de cerrarlo

Paso 8) Crear en su cuenta de GitHub un repositorio vacío de nombre my_calculator_2.

Paso 9) Subir la rama master del repositorio local al nuevo repositorio en GitHub.

Sugerencia de comandos a utilizar:
$ git push https://github.com/<su_cuenta>/my_calculator_2 master