Instituto Tecnológico de Costa Rica

Ingeniería Mecatrónica

Microprocesadores y Microcontroladores

Profesor: Felipe Meza Obando

Tarea 1

Alexander Solís López

2018158707

Luis Rodríguez Mena

2018087995

I Semestre 2021

1) ¿Diferencie la herramienta Git de Github?

"Git" es un sistema de control de versión, que permite guardar progresivamente un código de programación y, de ser necesario, también permite volver a una versión anterior del mismo. Además, la herramienta Git también permite la colaboración de varios programadores en un código [1].

Github, por otro lado, es una plataforma que permite que los usuarios guarden sus códigos en la nube, a partir de la herramienta Git [1]. Además, Github facilita la participación de distintas personas en el mismo código, pues no solo guarda el avance del código en general, sino que también identifica quién realizó el cambio en cuestión [2].

2) ¿Qué es un branch?

Un Branch son los espacios o entornos independientes que pueden ser utilizados para trabajar dentro de un mismo proyecto, sin borrar o modificar el conjunto de archivos originales del proyecto o "árbol de código" [1].

3) ¿Qué es un commit?

El comando ".git commit" guarda la versión del código en el que se esté trabajando [1]. El commit registra los cambios que se hayan realizado desde el anterior commit hasta el actual [3]. Todos los commits que se hayan realizado pueden ser verificados con el comando "git log", que devolverá información útil como el autor del commit o la fecha en que se realizó [1].

4) ¿Qué es la operación cherry-pick?

Corresponde a un comando que permite utilizar una o varias funcionalidades o "commits" de una segunda rama, en la rama actual de trabajo, todo esto sin tener que hacer un "merge" completo [4].

5) ¿Qué hace el comando git stash?

El comando "git stash" sirve para guardar temporalmente un código que tal vez no está suficientemente avanzado para realizar un commit, pero que por algún motivo no se puede continuar (por ejemplo, que se esté programando un código y sea necesario empezar a trabajar con otro distinto, pero que el primero no esté completo aun). Una vez que se puede continuar con el código que se guardó en un stash, se utiliza el comando "git stash apply" o "git stash pop" para recuperar lo que se guardó en dicho stash. Varios stashes pueden crearse en una misma hoja de trabajo, y pueden ser administrados con el comando "git stash list", indicando la rama y el commit sobre el cuál se encuentran, así como cualquier etiqueta que se haya añadido por parte del programador [5].

6) ¿Compare las operaciones git fetch y git pull?

En contexto, ambos comandos se utilizan cuando se está trabajado con un repositorio clonado, es decir, un duplicado de otro repertorio y se espera mantener actualizado el clon con los cambios que se realicen en el repertorio original [6].

Con el comando git fetch se recupera la última información de los metadatos del original, esta se coloca en una rama espejo, en donde se pueden mirar los cambios en dicha rama y posteriormente hacer un "merge" a la rama local de trabajo [6].

Con el comando git pull se recuperan los cambios realizados en el repositorio original y se actualiza automáticamente la rama local de trabajo. Esto se puede ver como un "git fetch + merge" [6].

7) Asumiendo que usted está en un Branch llamado "secundario" y su Branch principal se llama "master" ¿Qué resultado espera de hacer git rebase master? ¿Qué resultado espera de hacer git rebase origin/master?

El comando "git rebase" es útil para unir dos *Branches* distintos en los que se está trabajando. Si se está en "secundario" y se realiza "git rebase master", entonces se realizan una serie de pasos: primero, se compara cuál es el último *commit* en el que ambos *Branches* coinciden, que usualmente será el *commit* sobre el que se inició "secundario". Luego, se verifica qué cambios han habido en "secundario" y se almacenan en una memoria interna temporal. Posteriormente, se verifican qué cambios se han hecho en "master". Finalmente, todos los nuevos *commits* encontrados en "master" se colocan en "secundario", y al frente de estos se colocan los cambios en "secundario" que se encontraban almacenados en la memoria temporal. En la Figura 1 se muestra un ejemplo de lo anterior, suponiendo que la rama en amarillo corresponde a "master" y la azul a "secundario" [7].

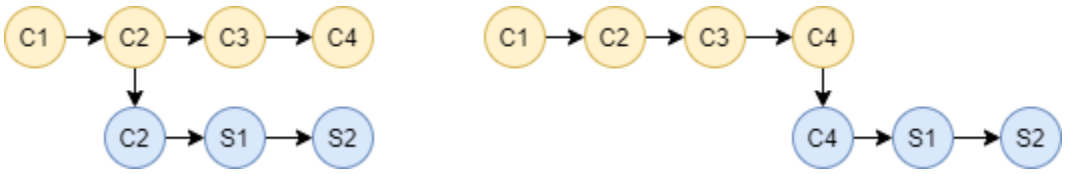

Figura 1. Ejemplo del comando "git rebase master" para dos ramas antes de ser aplicado (izquierda) y después de ser aplicado (derecha). Fuente: elaboración propia.

Si se utiliza "git rebase origin/master", se está haciendo un *rebase* a partir de "master". Esto quiere decir que se van a colocar los cambios hechos en la rama "secundaria" al frente del último *commit* en común que existiese en "master", y luego se añaden los nuevos cambios hechos en "master" [7].

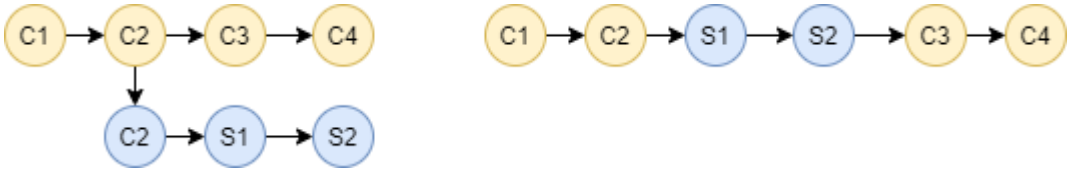

Figura 2. Ejemplo del comando “git rebase origin/master” para dos ramas antes de ser aplicado (izquierda) y después de ser aplicado (derecha). Fuente: elaboración propia.

8) ¿Qué es una Prueba Unitaria o Unittest en el contexto de desarrollo de software?

Las pruebas unitarias son una manera de probar software, consisten en aislar una parte del código y con ello comprobar que este funcione de la manera esperada. Estas partes pueden ser, por ejemplo, funciones individuales, métodos, procedimientos, módulos u objetos [8].

9) ¿Bajo el contexto de pytest. ¿Qué es un “assert”?

Es un proceso que permite crear condiciones para verificar el correcto funcionamiento de un código. Es una herramienta útil para realizar procesos de Debugging, y encontrar posibles errores a lo largo del programa [9].

10) ¿Qué es Flake 8?

Es una librería de Python que funciona como una herramienta para encontrar posibles errores cometido en la programación realizada, tales como una librería importada sin utilizar y para revisar la complejidad ciclomática [10].

Referencias Bibliográficas

[1] A. Wilkie. (2020) What is GitHub? What is Git? And How to Use These Developer Tools [En línea]. Disponible en: https://www.freecodecamp.org/news/what-is-github-what-is-git-and-how-to-use-these-developer-tools/

[2] Github, *What is Github?* 2016. Accesado en: 18 de Febrero, 2021. [Archivo de video]. Disponible en: https://www.youtube.com/watch?v=w3jLJU7DT5E&ab_channel=GitHub

[3] HubSpot Product Team (2020) An Intro to Git and GitHub for Beginners (Tutorial) [En línea]. Disponible en: https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners

[4] "Git Cherry Pick | Atlassian Git Tutorial", Atlassian Bitbucket. [En línea]. Recuperado de: https://www.atlassian.com/git/tutorials/cherry-pick.

[5] Runroomer. (2018) El comando Stash para cambios rápidos en un proyecto en Git [En línea]. Disponible en: https://www.runroom.com/realworld/git-stash

[6] J. Carrillo, "Git Fetch vs Pull: ¿Cuál es la diferencia entre los comandos Git Fetch y Git Pull?", freeCodeCamp, 2021. [En línea]. Recuperado de: https://www.freecodecamp.org/espanol/news/git-fetch-vs-pull-cual-es-la-diferencia-entre-los-comandos-git-fetch-y-git-pull/.

[7] Academind, *Git MERGE vs REBASE,* 2018. Accesado en: 18 de Febrero, 2021. [Archivo de video]. Disponible en: https://www.youtube.com/watch?v=CRlGDDprdOQ&ab_channel=Academind

[8] "Unit Testing Tutorial: What is, Types, Tools & Test EXAMPLE", Guru99, 2021. [En línea]. Recuperado de: https://www.guru99.com/unit-testing-guide.html.

[9] D. Hillard, (2020) Effective Python Testing With Pytest [En línea]. Disponible en: https://realpython.com/pytest-python-testing/

[10] V. Freitas, "How to Use Flake8", Simple is Better Than Complex, 2016. [En línea]. Recuperado de: https://simpleisbetterthancomplex.com/packages/2016/08/05/flake8.html#:~:text=Flake8%20is%20a%20Python%20library,and%20to%20check%20cyclomatic%20complexity