**CLASE 6 GIT/GITHUB**

(22) 🚀GIT: TODO LO QUE TENÉS QUE SABER SOBRE BRANCHES (RAMAS) 🌳 2021 🤩 | Introducción a GIT y GITHUB #6 - YouTube

https://www.youtube.com/watch?v=gjKKtQVVCZU

Transcript:

(00:00) a todos soy luisina y bienvenidos a un nuevo vídeo en donde todo para chicos en este vídeo vamos a ver un concepto que venimos charlando hace muchos vídeos en esta serie de git que les voy a dejar acá arriba y en la descripción para que tengan el link a los vídeos anteriores que es el tema de las branches o mejor conocida en español como ramas que son las ramas habíamos trabajado habíamos hablado también de que lo que tiene de maravilloso es que podemos manejar varias versiones de un mismo proyecto al mismo tiempo y donde se guarda cada una

(00:42) de estas versiones en rama así que podemos hacer con esas ramas podemos estar cambiando entre una y otra todo el tiempo que quiere decir en lugar de tener 25 mil archivos como siempre decimos de versión una estás y diversión 2 esta nueva versión 3 y volver y saber cuál es la versión original en vez de hacer eso tenemos distintas branches donde podemos tener de un cambio después de un cambio después de haber agregado a lo que necesitábamos después que quitamos algo y ahora no funciona podemos volver a la versión anterior de forma súper sencilla

(01:16) únicamente intercalando entre las branch y también muchas veces cuando trabajamos con distintos entornos de desarrollo como un entorno productivo otro de test y otro de desarrollo en general también podemos tener distintas ramas para simular cada uno de estos ambientes o de los archivos que tenemos en cada uno de estos ambientes ahora vamos a ver un poquito de forma más gráfica qué son las ramas vamos a la definición formal 15 que son como punteros que nos permiten ver cada uno de los cambios o versiones que va a tener nuestro proyecto con el

(01:50) que vamos a estar trabajando ahora estos cambios o versiones por cada cambio versión que creemos podemos crear una nueva rama que es lo que les comentaba recién la definición correcta es que es una línea independiente de desarrollo por ejemplo si trabajamos entre varias personas la rama va a ser una línea de trabajo distinta y lo que podemos hacer después es unir todas estas ramas en una sola que es nuestra rama master con nuestra rama main que es con la que siempre trabajamos si siempre acuérdense que git cuenta con esta rama principal o mail sí

(02:23) y que también se la conoce como master como venimos trabajando y de esta parte en todas las demás fíjense esta imagen que tenemos acá si este es nuestro proyecto hasta ahora lo venimos trabajando fíjense lo que venimos haciendo siempre en nuestra rama master hasta ahora venimos haciendo esto que pasa vamos a suponer que radio bueno ahora esto que estoy trabajando quiero agregarle otras cosas si yo me abro una rama si de este proyecto y le agrego tres nuevas funcionalidades por ejemplo qué hago después lo puedo volver a

(02:57) incorporar a esa rama principal y mientras tanto alguien pudo haber estado trabajando en esta rama principal y lo que yo agregué se va a incorporar a esto que ya está trabajado acá bien además de esto otra persona en este punto pudo haber sacado la versión que había hasta este momento de la rama haber hecho su propia rama y luego después de que simplemente la mía agregar esa nueva funcionalidad de esta manera trabajan las ramas yo puedo trabajar con un montón de personas al mismo tiempo cada uno crea su rama y

(03:32) luego podemos unificar todo lo que tenemos mediante diferentes comandos ahora vamos a pasar aquí y vamos a ver cómo podemos hacer para crear cada una de nuestras ramas intercambiar y cambiar entre ellas y otras cuestiones que se los voy a mostrar pasamos a eso bien chicos acá estamos en nuestra carpetita que corresponde al repositorio con el que estuvimos trabajando en la clase anterior que habíamos hecho nuestro clon y nuestro pool tenemos los dos archivos que habíamos creado vamos a suponer que esto es un proyecto en súper sofisticado

(04:01) no lo es pero bueno tenemos los documentos acá vamos a suponer que lo decir que si quiero empezar a quedar ramas distintas versiones para ir probando cambios y ver si esos cambios me generan o no conflictos como lo voy a dentro de mi repositorio en este caso por david mi repositorio local puede ser clic derecho git bayern para abrir nuestra consola voy a agrandar a cada consola sí y lo que voy a hacer es tirar el siguiente comando voy a poner git branch y quedo con esto me va a decir en qué rama estoy parada actualmente acá ya

(04:39) tenemos una pista porque hice master pero si hacemos git branch no va a decir en qué rama estamos parados ahora en este caso nos dice que estamos en la rama master fíjense que nos aparece con un asterisco con el game price entonces lo que hacemos es ver dónde estamos parados ahora con eso si podemos cambiar de rama en rama de versión en versión vamos a suponer que yo quiero crear una nueva rama que se llame rama 1 vamos a inventar el nombre sí entonces para hacer esto vamos a utilizar nuevamente 10 branch pero ahora vamos a poner el

(05:10) nombre que le queremos asignar a la rama que vamos a crear entonces le va a poner rama 1 hacemos enter y automáticamente nos acaba de crear si esta rama 1 si ponemos git branch de vuelta fíjense que nos dice que tenemos dos ramas creadas master y rama 1 page vamos con los primeros conceptos básicos pero qué pasa si yo quiero cambiar el nombre de esta rama me molesta que se llame rama 1 en realidad le quiero llamar ramita supongan se si era ramita 1 y le puse rama y no me di cuenta lo puedo hacer haciendo lo siguiente git branch nuevamente pero él

(05:52) agregó guión y me pongo cuál es el nombre actual de la rama en este caso rama 1 y luego pongo el nombre por el cual quiero cambiar el nombre nuevo en este caso ramita 1 apretó enter y van a ver ahora si tiro git branch nuevamente para ver los comandos anteriores pueden mover con las flechitas del teclado para arriba y para abajo y les muestra si los comandos voy a ver a big branch para no escribir de vuelta y me dice fíjense que estoy parada en master y se me aparece el nombre nuevo que acabo de cambiar bien ahora bien ya

(06:30) cree la rama nueva me equivoque el nombre se le cambia el nombre conjugamos para cambiar de rama para pasar de donde estoy ahora que es master a ramita 1 para pasar la ramita 1 lo que voy a hacer es poner kits checkout y voy a poner el nombre de la rama a la que quiero cambiar en este caso ramita 1 aprieto enter y me dice automáticamente que cambie a la rama ramita 1 fíjense cómo hasta caben y diciendo master y ahora cambia y dice ramita 1 vamos a verificar si es así tiro git branch nuevamente y efectivamente me dice que

(07:11) tengo dos ramas master y ramita 1 pero que ahora estoy parada en ramita 1 bien podemos suponer quien me arrepentí si me arrepentí ya no quiero que este ramita 1 y la quiero eliminar vamos a suponer eso bien qué voy a hacer entonces voy a poner git guión de debilidad de eliminar le voy a poner el nombre de la rama que quiero borrar en este caso ramita 1 voy a hacer enter y automáticamente me dice que no puede si borrar estas ramas porque estoy parada en este momento en ella lo que voy a hacer entonces es cambiarme a

(07:50) master nuevamente git checkout master si ahora que estoy para el master voy a intentar borrar la mitad 1 y fíjense cómo se puede borrar correctamente si hacemos git branch de vuelta para ver qué ramita uno no existe hasta vivimos los conceptos básicos como creamos una rama como cambiamos entre una rama y otra como le cambiamos los nombres y cómo hacemos para eliminar unas ramas y ya no la queremos utilizar más bien chicos ahora lo que acabo de hacer es borrar todo mi terminal que pueden hacer local mediante la palabra

(08:28) clear si en inglés que significa limpiar les va a quedar todo limpio y ahora sí que salimos los conceptos básicos de cómo trabajar con las ramas vamos a crear una nueva rama y en esa rama vamos a crear archivos que no estén en nuestra rama principal bien vamos a ir haciéndolo de a poquito que vamos a hacer voy a crear git branch sí y vamos a poner el nombre de la rama que vamos a crear hacemos a llamarle rama 1 bien bien una vez creado si ponemos git brands sólo nos va a decir que tenemos las dos ramas sí y que estamos parados

(09:03) en master qué vamos a hacer vamos a cambiar y vamos a ponernos en rama uno bien vamos a hacer guita check out y vamos a poner rama 1 bien si ahora ponemos big branch nos va a decir que estamos parados fíjense que aparece en verde si en la rama 1 activamos 10 puntos que vamos a hacer en la rama 1 vamos a empezar a crear archivos si quien no van a estar en master fíjense vamos a poner con el comando touch puedo poner un nuevo archivo en este caso le voy a llamar texto punto txt vamos a verlo así y ese comando fíjense que me está

(09:48) creando acá atrás un nuevo archivo para acá me voy a crear otro más que diga texto dos puntos x t y cree dos archivos fíjense este detalle yo todavía no confirme este cambio como no lo confirme que es un cambio exclusivo de la rama 1 si yo pongo checkout y vuelvo a master fíjense también estoy en master ahora siguen apareciendo todos los archivos acá porque porque también son muy confirme que es exclusivo de la rama 1 bien voy a volver a la rama 1 voy a poner get check in check out casi 7 rama 1 ahora estamos parados acá y lo que vamos

(10:35) a hacer es agregar y luego comité ar que estos cambios que acabamos de hacer son exclusivos de la rama 1 bien como lo vamos a hacer con comandos que ya vimos antes que te voy a dejar los links a carreras si no sabes cómo se utilizaba pri vamos a hacer get up y un punto para que agregue todos los archivos que estamos trabajando en esta rama que en este caso son dos los nuevos y luego vamos a poner git commit para confirmar inm y vamos a poner un mensaje estoy cometiendo un comité cuando los archivos bien dejarlos hard aprieto dentro

(11:16) y fíjense que me está diciendo que dos archivos han cambiado si tengo texto y texto 2 ahora que acabamos de comité ar estos dos archivos y acabamos de decir que son exclusivos de la rama 1 fíjense qué va a pasar en mi carpeta que atrás si si su cambio de rama voy a volver a master sí que master no tiene estos cambios que yo acabo de comité ar vuelvo a master vamos a poner checkout master y fíjense lo que pasa acá atrás en mi carpeta se acaba de borrar texto y texto dos meses pero porque profe porque estoy volviendo a máster y máster tiene

(11:57) una versión anterior rama tiene estos dos nuevos archivos que yo acabo de crear mi rama 1 pero master todavía no tiene estos cambios porque yo no los agregue bien como ahora para volver a mi rama dónde estoy trabajando con nuevos cambios pongo mi checkout rama 1 y fíjense lo que va a pasar mi carpeta se actualiza y si aparecen estos dos archivos nuevos que tengo fíjense y lo fácil que es manejar ramas manejar versiones sin tener archivos 1 2 3 4 5 siempre en caso de que mi modificación que estoy haciendo en rama 1 falle yo puedo volver

(12:35) tranquilamente a master a mi versión anterior borrar esa rama y crear una nueva y probar de manera diferente para esto sirven las ramas chicos obviamente que si confirmamos después que esta funcionalidad es correcta podemos agregar la también a la rama cómo se hace eso mediante un comando que se llama merge que lo vamos a ver en un próximo vídeo en conjunto con unas herramientas finales que vamos a utilizar en git para complementar con toda esta parte introductoria para chicos eso fue todo por hoy espero que

(13:07) les haya gustado no se olviden de darle a me gusta dejar comentarios para que nos podamos posicionar en youtube antes suscribirse a lógica la campanita para no perderse nada al igual que seguimos en las redes sociales en la tocó academia el próximo vídeo vamos a ver un poquito lo que es el mars y algunos otros comandos extra que tiene ir para que podamos ir perfeccionando nos un poquito vimos el uso de las ramas lo cual es súper importante a la hora de manejar diferentes versiones de un mismo proyecto nos vemos entonces en un próximo vídeo

(13:35) no te lo pierdas en donde en toco cha cha no