**အသုံးများတဲ့ Basic command များ အကြောင်း**

ကိုယ့်ရဲ့ လက်ရှိ git version ကိုကြည့်ချင်တယ်ဆိုရင် ဒီလို command သုံပြီး ကြည့်နိုင်ပါတယ်။

git --version

ကိုယ်က project အသစ်ထဲမှာ git သုံပြီး github ပေါ်က repository နဲ့ ကိုယ့်ရဲ့ local နဲ့ Link လုပ်တော့မယ်ဆိုရင်

git init  
git remote add origin <https://github.com/kelvinkyaw/git_tutorial.git>  
#origin ရဲ့နောက်မှာ github ပေါ်မှာ create လုပ်လိုက်တဲ့ repo URL

ပြီးတာနဲ့ git add ဆိုပြီး files တွေ folders တွေ ထပ်ထည့်သွားလို့ရပါပြီ။ ဥပမာ

git add src/main.js  
git add index.php  
git add app/Controller  
git add app/Model  
git add tmp

စသည်ဖြင့် ထည့်သွားလို့ရပါတယ်။ ကိုယ်က ဘယ် file တွေ folder တွေ ချိန်းထားလဲ သိချင်ရင်။

git status

ဆိုပါစို့ ကိုယ်က တစ်ချို့ file တွေ၊ folder တွေကို github ပေါ်မှာ မတင်ချင်ဘူး ဆိုတဲ့ ကိစ္စမျိုးမှာဆိုရင် .gitignore ဆိုတဲ့ hidden ဖြစ်နေတဲ့ file ကိုပြင်ပြီး ignore လုပ်လို့ရပါသေးတယ်။ အပေါ်က example မှာဆိုရင် tmp ဆိုတဲ့ folder ကို ignore လုပ်ချင်တယ်ဆိုရင်၊ နှစ်သက်ရာ editor သုံးပြီး .gitignore ကိုဖွင့်လိုက်ပါ။ ကျနော်က vim သုံးပါမယ်။ မရှိရင်လည်း vim .gitignore လို့ရိုက်ပြီးရင် keyboard က a key ကိုနှိပ်။ ပြီးတာနဲ့ tmp/လို့ရိုက်။ Esc key ပြန်နှိပ်ပြီးရင် :wq ဆိုပြီး file ကိုသိမ်းပြီးထွက်လိုက်ပါ။ vim ကိုလုံး၀မသုံးတက်ရင် ကြိုက်နှစ်သက်ရာ editor သုံးပြီး ပြင်လို့ရပါတယ်။ . dot files တွေက hidden ဖြစ်နေတက်ပါတယ်။ hidden ဖြစ်နေတဲ့ file တွေကို ဖော်ပေးတဲ့ command တွေက

Mac: Cmd + Shift + . (dot)

Ubuntu: Ctr + H

Windows သမားတွေဆို folder ကို right click -> Properties tab ထဲက hidden ဆိုတဲ့ checkbox ကို ရွေးပြီး OK နှိပ်။

Command line ကနေဆို ls -al

vim .gitignore

git status ခေါ်ကြည့်ရင် tmp folder အောက်မှာ ဘာချိန်းချိန်း၊ file အသစ်ပဲထည့်ထည့် ပေါ်မလာတော့ပါဘူး။

git add လုပ်ပြီးသွားတဲ့ file ကို reset ပြန်လုပ်ချင်ရင်။ အဲ့ဒါဆို commit လုပ်တဲ့အချိန်မှာ ပါမသွားတော့ပါဘူး။

git reset src/main.js #file or folder name

လိုချင်တဲ့ files တွေလည်း ထည့်ပြီးပြီဆိုရင် commit message လေးထည့်ဖို့ လိုပါတယ်။ ဒါမှလည်း github မှာ file တွေရောက်လာတဲ့အခါ ဘာအတွက် တင်ခဲ့တာလည်း သိရမှာပါ။ command က

git commit -m "first commit"

github ပေါ် push မလုပ်ရသေးဘူး။ commit လုပ်လိုက်တဲ့ message လေးကို ပြင်ချင်ရင်။ (push လုပ်ပြီးသွားရင်တော့ အောက်က command နဲ့ ပြင်လို့မရပါ။ တစ်ခြား command သုံးရပါမယ်။

git commit --amend

ဒီလိုပေါ်လာပါလိမ့်မယ်။ ပြီးတာနဲ့ message အသစ်ပြန်ပြင်ရေးလို့ရပါပြီ။

github ပေါ် တင်တော့မယ်ဆိုရင်

git push origin branchname

တကယ်လို့ ကိုယ့် branch က github လုံး၀မရှိသေးတဲ့ branch ဆိုရင်

git push -u origin branchname  
git push origin branch #branchname ကရှိပြီးသာဆို

ကိုယ့်ရဲ့ local မှာ branch အသစ်ယူချင်တယ်ဆို

git checkout -b branchname

နောက်ထပ်အရေးကြီးတဲ့ basic command တစ်ခုက stash ဆိုတဲ့ command ပါ။ ဘယ်နေရာမှာအသုံး၀င်လဲဆိုရင်, လုပ်နေရင်းတန်းလန်း commit လုပ်လို့လည်းမရသေးဘူး။ ရှေ့က လုပ်ထားတဲ့ တစ်ခြား branch ရဲ့ commit က file မှာ error တက်လို့ fix အရင်လုပ်ရမယ်။ လုပ်လက်စကို commit လုပ်လိုက်ရင်လည်း မလိုအပ်ပဲ commit တွေ ထပ်နေအုံးမယ်။ ဒီ command ကိုရိုက်တာနဲ့ လက်ရှိ လုပ်နေတဲ့ အလုပ်တွေ အရင် original stage ပြောင်းသွားပါပြီ။ ပြီးတာနဲ့ ကိုယ်သွားချင်တဲ့ branch ကိုသွား၊ fix လုပ်။ Command က

git stash

fixed လုပ်ပြီးသွားပြီးဆိုတာနဲ့ ကိုယ့် လုပ်နေတဲ့ branch ကိုပြန်လာ။ ပြီးရင်

git stash pop

ဆိုတာနဲ့ အရင် stash လုပ်ထားတဲ့ file တွေ ပြန်ပေါ်လာပြီး အလုပ်ဆက်လို့ရပါပြီ။ မိုက်တယ်နော်။ 😍

⚠️ နောက်ဆုံး basic command က အရေးကြီးပြီး သတိထားသုံးသင့်တဲ့ command ပါ။ ဒီ command ကို နောက်ပိုင်းမှာ topic တစ်ခုအနေနဲ့ တင်ပေးပါမယ်။

git reset --hard or --soft #commithash

လက်ရှိရှိနေပြီးသား အလုပ်တွေက အမှားကြီး လုံးဝ history တောင် မလိုချင်တော့ဘူးဆိုရင်။ hard ဆိုတဲ့အတိုင်း history တောင် လုံး၀မကျန်ပါဘူး။ အားလုံး rollback ချိန်ချင်တဲ့ commit နောက်ပိုင်း လုပ်သမျှ lost ဖြစ်သွားပါမယ်။

git reset --hard yourcommithash #e.g 65a81740ff8eeee98d8e04df2f5ae8805377fe12

နောက်တစ်ခုက soft reset ပါ။ သူက changes ထားတဲ့ file တွေ အကုန် သိမ်းထားပါတယ်။ မဖျက်ပစ်ပါဘူး။ command က

git reset --soft yourcommithash #e.g 65a81740ff8eeee98d8e04df2f5ae8805377fe12

နောက်တစ်ခုက reset mixed ပါ။ command က

git reset --mixed yourcommithash #e.g 65a81740ff8eeee98d8e04df2f5ae8805377fe12working index HEAD target working index HEAD  
----------------------------------------------------  
 A B C D --soft A B D  
 --mixed A D D  
 --hard D D D

ဒီ table ရဲ့သဘောဆိုရင် ဒါလေးတွေ သိထားဖို့ လိုပါတယ်။

🔢 working: လက်ရှိလုပ်ပြင်နေဆဲ files တွေ

🔢index: commit မလုပ်ခင် git ရဲ့ cache ထဲမှာ git add ဆိုပြီးလုပ်ထားခဲ့တဲ့ files တွေ

🔢HEAD: ဆိုတာက လက်ရှိကိုယ့်ရဲ့ branch နဲ့ active ဖြစ်နေတဲ့ hash value (e.g d6f03d6382527878552933a5067ef9d46f37e8f5)

🔢 target: ဆိုတာက ကိုယ်လက်ရှိ commit hash ကို target လုပ်မဲ့ hash နဲ့ override လုပ်မဲ့ hash value

soft option ဆိုရင် working file တွေ ကျန်နေမယ်၊ index တွေကျန်နေအုံးမယ်။ ဒါပေမဲ့ ကိုယ့်ရဲ့ လက်ရှိ branch ကတော့ override လုပ်လိုက်တဲ့ commit နဲ့ သူ့ commit မှာပါတဲ့ changes တွေ ရောက်လာမယ်။

mixed option ဆိုရင် working file တွေကျန်မယ်၊ index က override လုပ်တဲ့ commit ရဲ့ index ဖြစ်သွားသလို ကိုယ့်ရဲ့ လက်ရှိ branch မှာလည်း override လုပ်လိုက်တဲ့ commit နဲ့ သူ့ commit မှာပါတဲ့ changes တွေ ရောက်လာမယ်။

hard option ကတော့ အန္တရာယ်ရှိတဲ့ option ပါ။ ဘာ histories မှ ပြန်ရစရာ မရှိပါဘူး။ All lost ပါ။ သေချာတယ်ဆို proceed with caution ပါ။

နောက်ဆုံး command ကတော့ merge ဆိုတဲ့ command ဖြစ်ပါတယ်။ ကျနော့်အတွေ့အကြုံဆိုရင် ကိုယ့်ကိုလာပေါင်းတာလား၊ သူ့ကိုသွားပေါင်းတာလားဆိုတဲ့ ဒွိဟတွေ ရှိခဲ့ဘူးပါတယ်။ အဲ့အတွက် ကျနော် ဥပမာတစ်ခုပြောပြပါမယ်။

ကိုယ့်မှာ ပိုက်ဆံ ၁၀ ရှိတယ်။ အောင်အောင်ဆီက ပိုက်ဆံ ၁၀ ချေးမယ်။ ကိုယ့်မှာ ၂၀ ဖြစ်သွားမယ်။ ကိုယ့်လက်ရှိ branch က ၁၀ ပါ။ အောင်အောင်က အချေးခံရမဲ့ branch ၁၀ ပါ။ ကိုယ်က develop branch မှာ active ဖြစ်နေတယ်၊ feature1 ဆိုတဲ့ branch ကို ကိုယ်ဆီကို ပေါင်းထည့်ချင်မယ်။ Command က

git merge feature01

ဒါဆို develop branch ထဲမှာ feature1 က changes လုပ်ထားတဲ့ files တွေ၊ folder တွေ ရောက်လာပါပြီ။

ဒီလောက်ဆို git basic ကို မွှေနောက်နိုင်ပြီလို့ ယူဆပါတယ်။ စာအုပ်ထွက်ရင် အားပေးမယ်ဆို ထပ်ရေးရကျိုး နပ်မယ်ထင်ပါတယ်။ ဟီးဟီး 🙌