Lesson : Git & GitHub
Day01
Git Intro
Git Pull & Push & Conflict

techproeducation

techproeducation

techproeducation

techproeducation

WHAT IS GIT?

- Git is a **version control** tool

WHY IS GIT USED?

1. **Save** the codes as different versions
2. **Update** the exiting codes and store them as new versions
    1. Add codes and save as new version/commit
    2. Delete codes and add as new versions
    3. Fix bugs and store as new version
3. **Collaboration** and **Team Work**
4. **Share** the codes online
5. **Track** the changes in the codes
    1. Who changed the code
    2. When the change happened
    3. What was the older codes look like
6. In short, you can **store, access, use, share** all your new or older codes, and **see their detailed information** anytime
    1. Commit 1 (version 1)
    2. Commit 2 (version 2)
    3. Commit 3 (version 3)

GIT VS GITHUB?

- Git is installed in your own computer (LOCAL REPOSITORY)
- Git is used to make local code changes and commits
- When you make local changes, it is save in **.git folder** in your local computer

- Github is remote version of git (REMOTE REPOSITORY)
- GutHub is hosted in the cloud
- Github is used to store the project in the remote repository
- Owned by Microsoft-2018

WHAT IS THE BENEFIT OF GIT-LOCAL REPOSITORY

1. Saves Time
2. Ability to work without internet offline
3. Revert and undo mistakes
4. Track history

WHAT IS THE BENEFIT OF GITHUB-REMOTE REPOSITORY

1. Collaboration. Enables to work with other people on the same projects
2. Sharable codes, Easy access from anywhere with internet
3. Security of the informations
4. Storing the new versions in the cloud
5. View the history in the remote repository
    1. Who made change
    2. When change is mage
    3. Where is the change
6. Open source

HOW WE CAN PUSH A GIT PROJECT TO GITHUB FOR THE FIRST TIME

CREATE NEW PROJECT

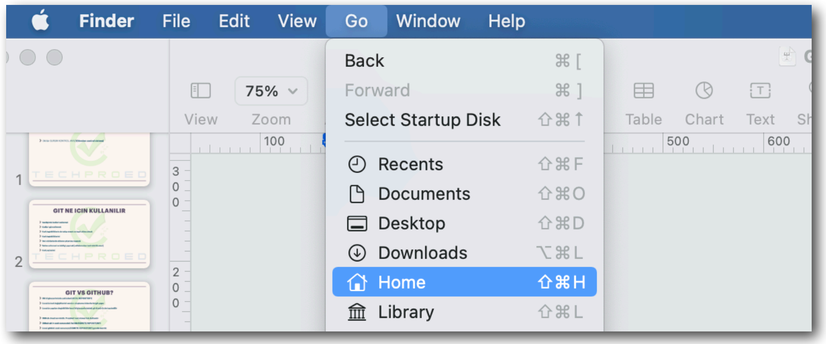
- Yeni java projesi olustur: **first_git_project**
  - File>New>Java Project> **first_git_project**>finish
- Create a new project under java package
  - src > new > package > **first_git_package**
- Create a Java class : **MyGitClass**
  - Create main method
  - System.out.println("Version 1”);
- **NOTE: SAVE if SAVE each time**

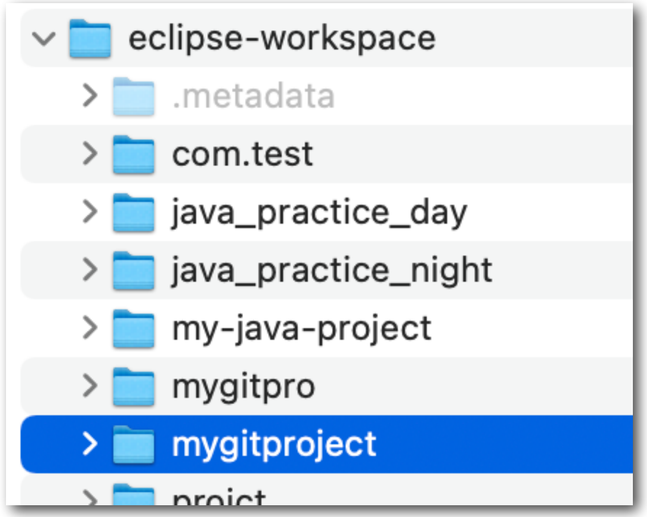
OPEN TERMINAL/GITBASH/CMD/...

- Open terminal or cmd
  - Mac: terminal, Window : CMD
- Change directory to you project: **cd yourprojectpath**
- Sample:
  - Path of my project : /Users/techproed/eclipse-workspace/mygitproject

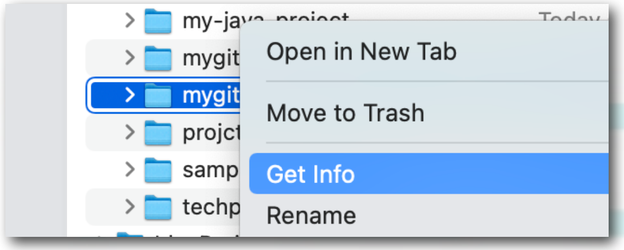
```
cd  /Users/techproed/eclipse-workspace/mygitproject

ENTER
```

SAMPLE PATH FINDER

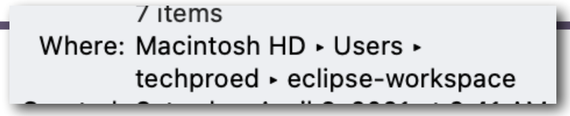





```
techproed@techproeds-iMac mygitproject % cd /Users/techproed/eclipse-workspace/
mygitproject
```




```
techproed@techproeds-iMac mygitproject %
```



CREATE GITHUB REPOSITORY

- https://github.com/ and login
- ***. click on your profile
- ***. click on your Repositories
- ***. click on New
- ***. repository name : first-repo
- ***. select public or private then create

HOW TO PUSH FIRST GITHUB CODE

- ***. Create a new project or select an existing project
- ***. Go to project directory if you are not on the project folder
- *cd PATH OF THE PROJECT
- ***. git init
- ***. git status(OPTIONAL)
- ***. git add .
- ***. git commit -m "messsage"
- ***. git branch -M main
- ***. git remote add origin https... github repository link
- ***. git push -u origin main

Github provides this information

...or create a new repository on the command line

```
echo "# testrepo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/techproed/testrepo.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/techproed/testrepo.git
git branch -M main
git push -u origin main
```

Lesson : Git & GitHub
Batch 64-67-66-67-68-69
Day02
Git Conflict
Branch
Collaboration

techproeducation

techproeducation

techproeducation

techproeducation

HOW TO PUSH NEW COMMITS AFTER FIRST COMMIT TO GITHUB

GIT PUSH COMMANDS

- *** git init** ---> Create local repository. Create .git folder. DO THIS ONE FOR EACH PROJECT
  - **NOTE:** if git is not installed, then you will see error.
- *** git status** ---> Shows changes. Our guide. Use anytime to see the current condition.
- *** git add .** —> Add ALL codes to STAGING AREA
    - *** git add “file name 1"** --->>> Add ONLY that file in the staging area
    - *** git add “file name 2"** --->>> Add ONLY that file in the staging area
- ***git commit -m “ANY MESSAGE"** —> Create a new commit(new version). Moves codes from Staging Area to Local Repository
    - After creating new version, I can PUSH codes in the Remote Repository
- ***git push** —>Push the code to Remote Repository

REVIEW OF GIT PUSH AND GIT PULL

- Create a new java class : SecondJavaClass
- Create another JAVA class: ThirdJavaClass
- Create a new commit to push all codes to your GitHub

- Create a new class in GitHub : RemoteJavaClass
- And pull the code from GitHub to your local repository

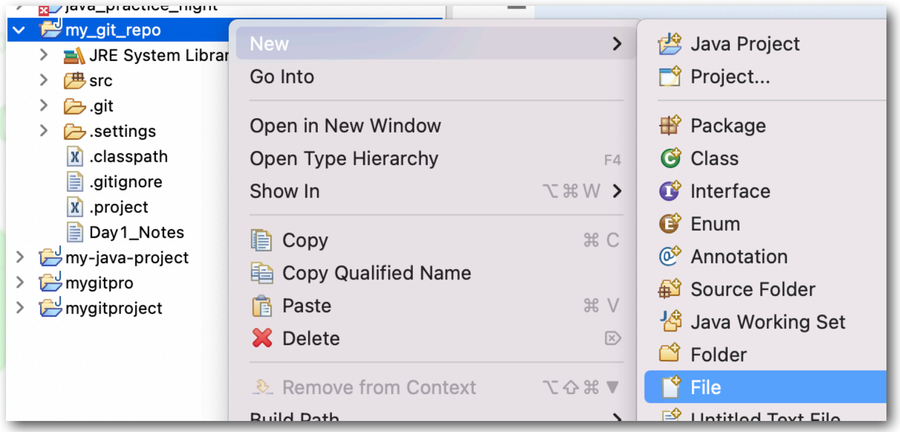
PRACTICE 1-HOW TO PUSH SPECIFIC FILES

- Create a new java class : ForthJavaClass
- Create another JAVA class: FifthJavaClass
- Create a new commit to **push only ForthJavaClass**
- Create another commit to push only FifthJavaClass

HOW TO IGNORE SPECIFIC FILES

- **.gitignore**
- We create .gitignore file in the project level.
- When we put the files path in this file, then git will ignore these files
- This file is used to ignore the files that will be committed

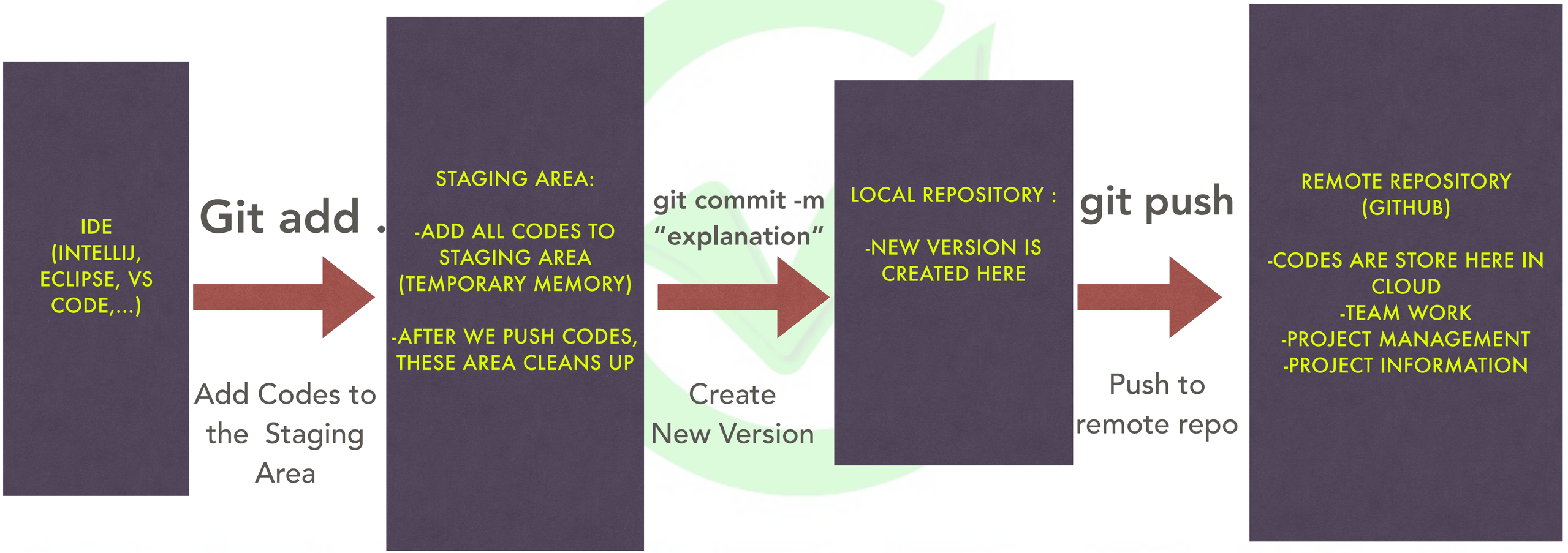
PRACTICE 2



- Create another JAVA class: SixthJavaClass
- Create another JAVA class: SeventhJavaClass
- Create a new java class : eightJavaClass
- Create a new commit to push All codes **except SixthJavaClass**

GIT STAGES

GIT STAGES AND GIT PUSH



HOW TO PULL CODES FROM GITHUB TO LOCAL REPO

WHEN TO USE GIT PULL?

- **If there is new commit in GITHUB, then we must PULL**
- We always need latest updated coded in our local repository
- If GitHub has latest codes, then always pull the latest codes
  - We need to pull codes from GitHub after other people send their code to GitHub
  - Otherwise there is a risk. We can delete other people codes by mistake!
  - GitHub will understand and won't let us push our code if GitHub has latest code

- **!!!GIT PULL BEFORE GIT PUSH**

GIT PULL COMMANDS

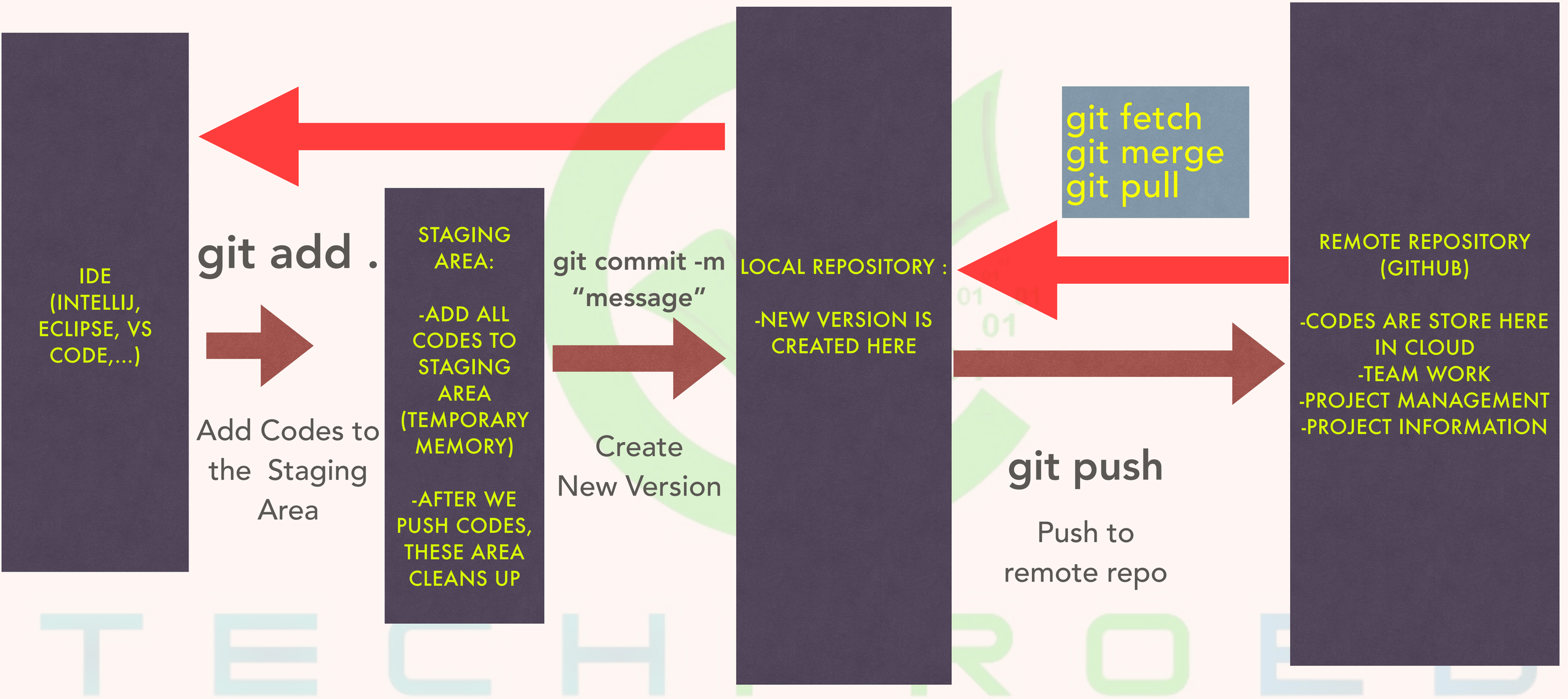
- ***git pull** —> brings and adds to your local repo. THIS IS ENOUGH TO PULL THE CODE

- Below combination can be used as well
- ***git fetch** —> brings updates on your local repo
- ***git merge** —> merges local git and remote GitHub codes
- ***git pull**

- NOTE: **git pull = git fetch + git merge**
- **git pull is enough to get the GitHub codes in our local repository**

- If there is no conflict then all GitHub codes will be in our local
- **NOTE:** We will practice when we see and how we can solve conflict

GIT STAGES/CYCLE AND GIT PULL



PRACTICE 1-UPDATING LOCAL CODES BY GIT PULL

- Commit a new code in GitHub
- Update your local code by using git pull
- Then you should see GitHub and local is matches
- **TASK STEPS:**
- Go you your GitHub repo and create a new file in the project level : MyNotes.txt
  - Add File > Create new file > name: MyNotes1.txt > commit new file
- Then go to your terminal and pull the code in your local repo
  - git pull
- Then we you see the GitHub and local codes are now the same

PRACTICE 2- ALWAYS KEEP YOUR LOCAL UPDATED BY GIT PULL

- Create a new commit in **GitHub** by adding a new file. : MyNotes2.txt
- Create a new commit in **local git** by creating a new class in local : MyNotes3.txt
- Try pushing your local code before pulling GitHub code
  - git add .
  - git commit -m “message”
  - git push
- **Then what happens???**
- **What is the solution???**
- GIT PULL BEFORE GIT PUSH!!!!!!!!!!!
  - git pull
  - git add .
  - git commit -m “message”
  - git push

```
To https://github.com/techproed/my_git_repo.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/techproed/my_git_repo.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
techproed@techproeds-iMac my_git_repo %
```

CONFLICT

WHAT IS CONFLICT

- If local and remote has different codes on the same lines, then we get conflict when we merge local and remote codes
- Conflicts usually happens when different people writes different codes on the same lines

HOW TO SOLVE CONFLICT?

- IDE can suggest to solve conflicts automatically, but we will solve them manually

- **Steps to solve conflicts manually**
  - Remove the conflicting code from local or edit the entire codes
  - Then pull GitHub codes
  - Then create the new version and push GitHub
    - git add .
    - git commit -m “”
    - git push
  - After this solution, local git and remote GitHub will have the same code

WHAT IS CONFLICT

1. LOCAL REPO: LINE 4 HAS MY CODE

```
1 package my_git_repo;
2
3 public class FifthJavaClass {
4 int x = 10;
5 }
6
```

```
techproed@techproeds-iMac my_git_repo % git add .
techproed@techproeds-iMac my_git_repo % git commit -m "committing local codes"
```

2. REMOTE REPO: LINE 4 HAS FRIEND'S CODE

5 lines (4 sloc) | 75 Bytes

```
1  package my_git_repo;
2
3  public class FifthJavaClass {
4  String str = "JAVA";
5  }
```

3. PULLING THE CODE BEFORE THE PUSH

```
techproed@techproeds-iMac my_git_repo % git pull
Auto-merging src/my_git_repo/FifthJavaClass.java
CONFLICT (content): Merge conflict in src/my_git_repo/FifthJavaClass.java
Automatic merge failed; fix conflicts and then commit the result.
techproed@techproeds-iMac my_git_repo %
```

4. CONFLICT!!!!

```
1 package my_git_repo;
2
3 public class FifthJavaClass {
4 <<<<<<< HEAD
5  int x = 10;
6 =======
7  String str = "JAVA";
8 >>>>>>> 94f2d897b8b2b08cb0a61538cdd4bbf2e
9 }
10
```

SOLUTION:

1. DELETE THE CONFLICT MESSAGES MANUALLY
2. COMMIT THE CLEAN CODE
3. PUSH THE CREAN CODE

```
1 package my_git_repo;
2
3 public class FifthJavaClass {
4
5  int x = 10;
6
7  String str = "JAVA";
8
9 }
```

```
% git add .
% git commit -m "resolved conflict"

% git push
```

FINAL CODE ON GITHUB HAS ALL CLEAN CODES

9 lines (5 sloc) | 92 Bytes

```
1  package my_git_repo;
2
3  public class FifthJavaClass {
4
5   int x = 10;
6
7   String str = "JAVA";
8
9  }
```

NON CONFLICT SCENARIOS

- When do you get conflict issue?
  - If github and intellij has DIFFERENT CODES in the SAME LINES

- How do you resolve conflict?
  - Normally IDE(intellij) has seating to resolve the conflict automatically.
  - But I prefer to solve manually because I need to see what is the conflict

NON CONFLICT SCENARIOS

- **SCENARIO 1:**
  - Local and remote repo has same codes on the **different lines**
  - I want to **pull code** from GitHub
- **Result?**
  - Do I get error message? Why?
    - No Error Message. I have to pull latest code from git hub.
  - Do I see any conflict?Why?
    - **No conflict cause codes are NOT ON THE SAME LINE**
- **Solution?**
  - No solution is required

NON CONFLICT SCENARIOS

- **SCENARIO 2:**
  - Local and remote repo has same codes on the **different lines**
  - I want to **push** my local code to GitHub
- **Result?**
  - Do I get error message? Why?
    - Yes, pull before push!
  - Do I see any conflict?Why?
    - No, not on the line
- **Solution?**

REMOTE REPO:
LINE 10 HAS FRIEND'S CODE

LOCAL REPO:
LINE 11 HAS MY CODE

CONFLICT SCENARIOS

- **SCENARIO 3:**
  - Local and remote repo has same codes on the **same lines**
  - I want to **pull** code from github
- **Result?**
  - Do I get error message? Why?
    - No error message, but we see a conflict warning message
  - Do I see any conflict?Why?
    - Yes cause same line has different codes
- **Solution?**
  - Auto merge, or manual fix(recommended)
  - git add . git commit -m “conflict resolved” git push

REMOTE REPO:
LINE 10 HAS FRIEND’S CODE

LOCAL REPO:
LINE 10 HAS MY CODE

- **PRACTICE**
  - Local : Update MyNotes3.txt **Line 1**: Local Code
  - Remote: Update MyNotes3.txt **Line 1**: Remote Code
  - Pull The code
  - What is the result?
- **Result?**
- **Solution?**

NON CONFLICT SCENARIOS

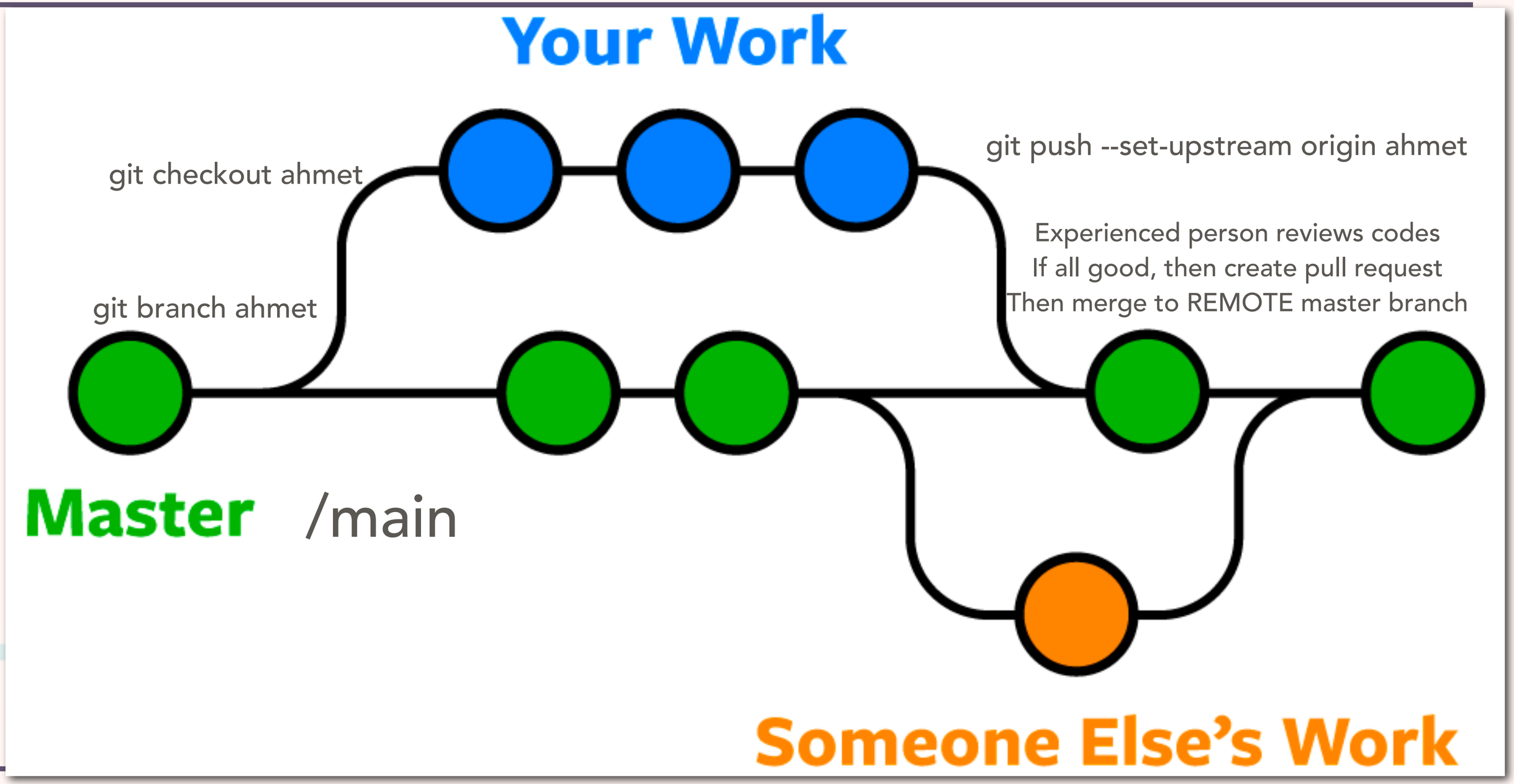
- **SCENARIO 4:**
  - Local and remote repo has same codes on the **same lines**
  - I want to **push** my local code to github
- **Result?**
  - Do I get error message? Why?
    - Yes, pull before push!
  - Do I see any conflict?Why?
    - No, not on the same line
- **Solution?**

REMOTE REPO:
LINE 10 HAS FRIEND'S CODE
COMMIT BY ORCUN

LOCAL REPO:
LINE 10 HAS MY CODE
COMMIT BE AHMET

GIT BRANCHES

WHAT IS BRANCH



WHY BRANCH

- Never write code in the master/main branch when working with team
- We are making copy of the original code and working on it without breaking original source code with branch
- Each person in the team works in their own local branch and push their own remote repo
- Then do **pull request**. Then One developer reviews the codes and approves pull request, then we merge codes goes to remote master/mail

BENEFITS OF BRANCH

- Security of the original codes
- Faster development
- Responsibility and ownership of their own code
- Less mistakes
- Quick fixes
- Organized codes
- No chaos

BRANCH COMMANDS

- *git branch OR git branch -a -> Shows all branches. Default branch is the master branch
- * git branch **feature** -> creates new branch named feature. This is the copy based on current branch.
- * git checkout **feature** -> switch to feature branch
- * git branch **-d feature** -> delete **feature** branch
- ***** git checkout **master** -> **switch to master branch**
- * git merge **feature** -> If you run this when you are on master branch, then feature branch codes merges **into master branch**
- ***git merge **master** -> If you run this when you are on feature branch, then master branch codes merges **into feature branch**
- **git push --set-upstream origin feature** -> use when you push for the FIRST TIME. After the first time, **git push** is enough

CREATING BRANCH AND PUSHING FIRST CODE

I. **Create a new branch** : git branch ahmet

II. **Check the branches**: git branch -> you should see ahmet branch

III. **Switch to the feature branch** : git checkout ahmet

    I. Now I'm on the feature branch. All changes I make will be only on this branch

IV. Create a new class and write some code on this branch

    I. MyLocalClass1

V. Create a new version on the branch

    I. git add .

    II. git commit -m "my local commit 1"

    III. git push --->>> NOTE : Gives error. First push must be done using below code

    IV. git push --set-upstream origin ahmet

    V. Check if your codes are in remote repository

VI. Question 1: Does ahmet branch have the codes we just pushed? YESSSSSSS

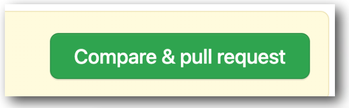
VII. Question 2: Does master have the codes we just pushed? NOOOOO. WE HAVE TO MERGE ALL CODES TO MASTER.NEXT SLIDE

MERGING THE CODES TO REMOTE MASTER BRANCH

- The codes we just pushed are not in remote master.
- Do the followings to move codes from local branch to master branch

I. Select ahmet branch from **project level**

II. Click on pull request

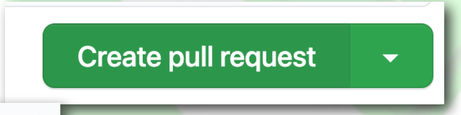


III. Click on new pull request

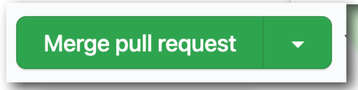
IV. Compare master and ahmet branch

   I. base:master, compare: ahmet

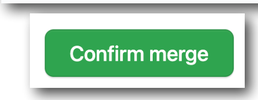
V. Review code and click create pull request



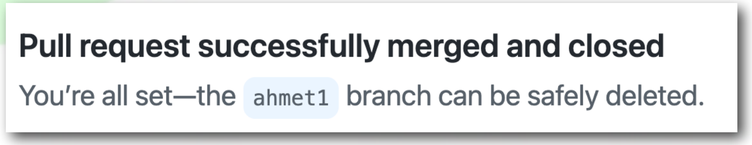
VI. Click Merge pull request



VII. Click Confirm merge



VIII. DONE!!! After these steps, all local branch codes will be in remote master



   I. Compare the master and local branch, then you will see they are same

IX. QUESTION: Does my ahmet branch and LOCAL master branch are same.(Is my local master updated?) NOOOO

X. Since remote master branch has the latest updated code, all team should update they local master(switch to local master then git pull) NEXT SLIDE

UPDATING LOCAL MASTER

- As of now, ahmet branch and remote master up to date, but my local master does not have latest codes

- Do the followings to update local master

  I. git checkout main

     I. you will see master has some missing codes

  II. git pull

- DONE! ALL BRANCHED UP TO DATE!

UPDATING LOCAL MASTER

- QUESTION: Should I use local master to write new codes? Next Slide.

CHECKING OUT LOCAL BRANCH AND CREATING NEW COMMITS

I. Again, we should not use master branch to write codes with team

II. Do the following to switch back to local branch and push the code

    I. git checkout ahmet

    II. Add code, commit changes, and push remote repo

        I. git add .

        II. git commit -m "feature 1 second commit"

        III. git push --->>> NOTE : This is not our first push. This will work !

            I. Note we had to use this for the first push : git push --set-upstream origin ahmet

            II. Now git push is enough

SUMMARY IN ONE PAGE : HOW TO SWITCH TO BRANCHES AND PUSH THE CODE

I. Switch to the local branch : git checkout ahmet

II. Create a new version on the local branch

I. git add . git commit -m “my local commit 2" git push

II. Check if your codes are in remote repository

III.QUESTION: Does master have the codes we just pushed? PULL REQUEST

I. Select ahmet branch from project level. Click on pull request. Click on new pull request

II. Compare master and ahmet branch

I. base:master, compare: ahmet

III. Review code and click create pull request. Click Merge pull request. Click Confirm merge

IV. DONE!!! After these steps, all local branch codes will be in remote master

I. Compare the master and local branch, then you will see they are same

V. Since remote master branch has the latest updated code, all team should update they local master(git pull)

IV.QUESTION: Does my ahmet branch and LOCAL master branch are same.(Is my local master updated?)

I. Do the followings to update local master

I. git checkout master

I. you will see master has some missing codes

II. git pull

V. QUESTION: Should I use local master to write new codes?

I. No. Checkout your local branch and repeat the steps again.

VI.This git workflow cycle keeps repeating

COLLABORATION

GIT CLONE

- **git clone**
- This is used to clone the github projects. We don’t use git clone to bring only the latest codes, but the entire code
- git clone https://github.com/techproed/my_git_repo.git

GIT COLLABORATION CYCLE

i. *Go to GitHub project *Settings *Manage Access *Invite a collaborator *Enter email

ii. *Accept invitation *Clone the project (git clone url) *git clone https://github.com/techproed/first-repo

iii. *COLLABERATOR: Collaberator creates his/her own branch and write in that branch

    i. *git branch collabatorbranch *git checkout collabatorbranch *create a new class *Write your code

    ii. *Push your code

        i. *git add . *git commit -m "collabarotor branch" *git push --set-upstream origin collaberatorsbranch. *check your github

    iii. *As responsible from git merge, I'm merging from local branch to master branch

        i. *pull requests *new pull request *compare codes *base: master, compare: COLLABARORS BRANCH *create pull request *merge pull request *confirm merge *git hub master branch is now updated.

        ii. Collabatators code is now on remote master

iv. *MY TURN: I write my code, and push to remote, then merge to remote master. Write code and push

    i. *git add . *git commit -m "message"

    ii. *Update local master

        i. *git checkout master *git pull -> local master is updated

    iii. *Update Local Branch

        i. *git checkout feature ***git merge master**

        ii. *In case of conflict keep, resolve the conflict and push the code to remote

        iii. *git add . *git commit -m "collabarotor branch" *git push

v. *codes are now in GitHub

vi. *As responsible from git merge, I'm merging from local branch to master branch

vii. *pull requests *new pull request *compare codes *base: master, compare: COLLABARORS BRANCH *create pull request *merge pull request *confirm merge *git hub master branch is now updated.

viii. *Everyone updates local master

    i. *git checkout master *git pull

ix. Now local master is updated *Collaberator must update the local master as well

HARD RESET

HARD RESET

- git reset --hard origin/master : resets your code and COPY REMOTE MASTER to local master
- git reset --hard origin/main
  - Deleted my local commits that is not pushed
  - Brings the remote master codes
- —>Be careful when you use this cause this will delete your commits that is not pushed

EXTRAS

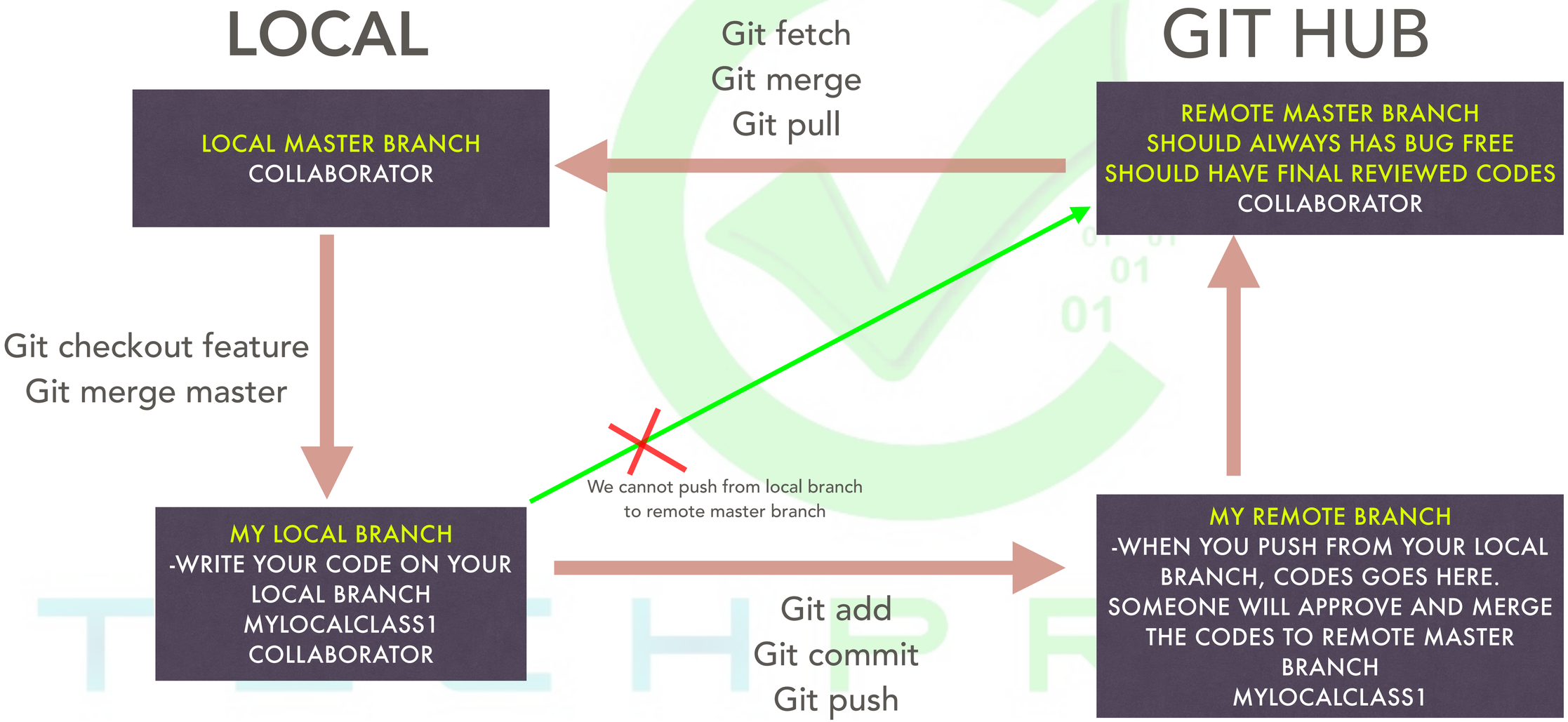
techproeducation

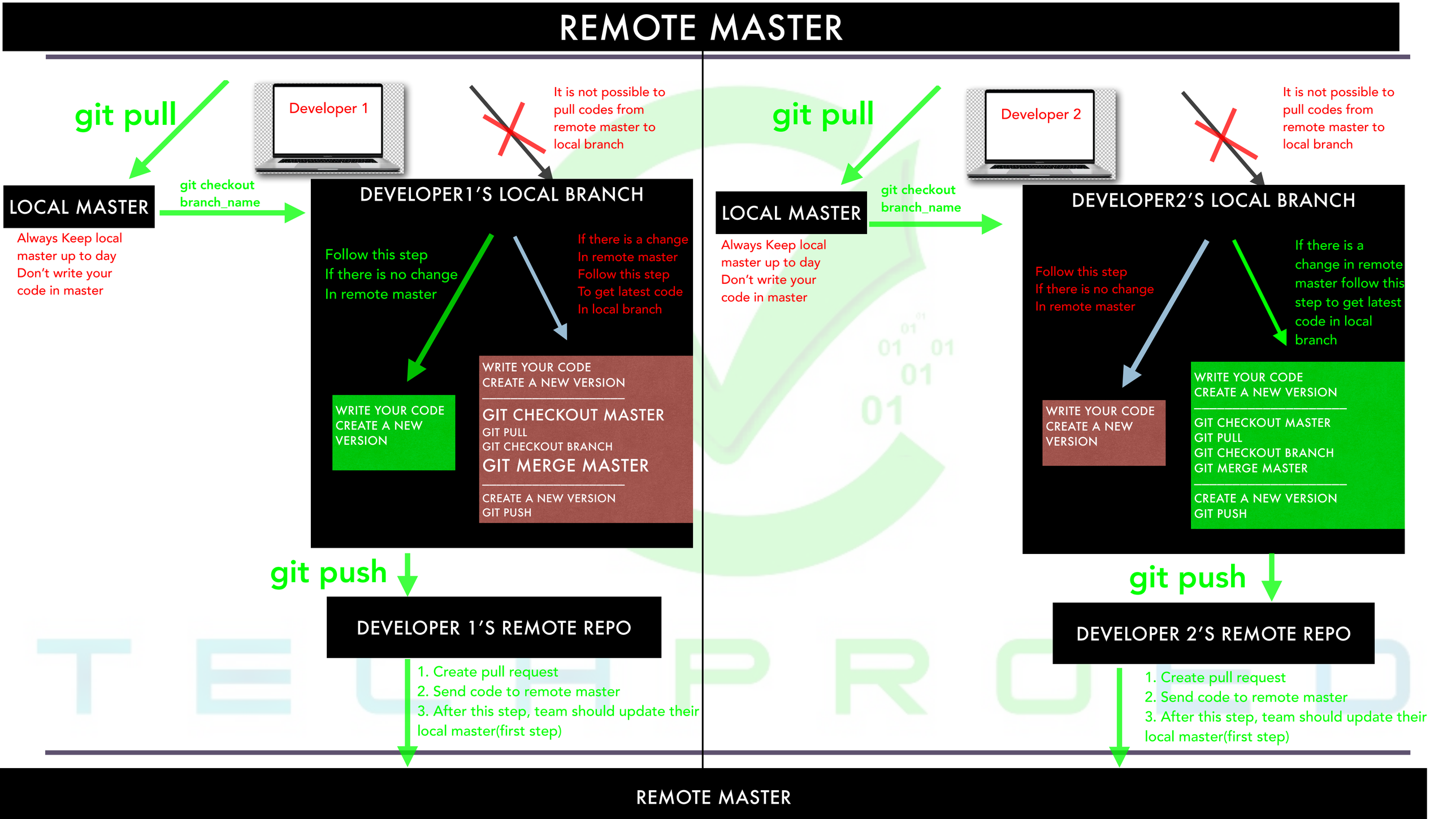
techproeducation

techproeducation

techproeducation

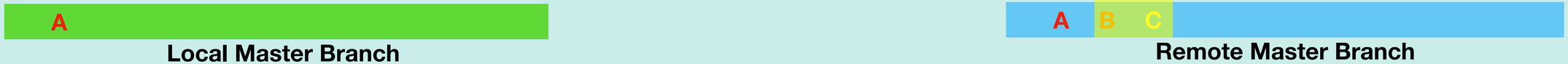
GIT CYCLE WITH BRANCH



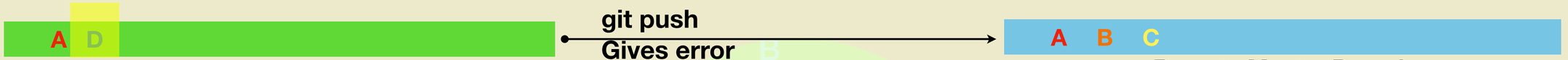

REMOTE MASTER
git pull
Developer 1
It is not possible to pull codes from remote master to local branch
LOCAL MASTER
git checkout branch_name
DEVELOPER1'S LOCAL BRANCH
Always Keep local master up to day Don't write your code in master
Follow this step If there is no change In remote master
If there is a change In remote master Follow this step To get latest code In local branch
WRITE YOUR CODE CREATE A NEW VERSION
WRITE YOUR CODE CREATE A NEW VERSION
GIT CHECKOUT MASTER
GIT PULL
GIT CHECKOUT BRANCH
GIT MERGE MASTER
CREATE A NEW VERSION
GIT PUSH
git push
DEVELOPER 1'S REMOTE REPO
1. Create pull request
2. Send code to remote master
3. After this step, team should update their local master(first step)
git pull
Developer 2
It is not possible to pull codes from remote master to local branch
LOCAL MASTER
git checkout branch_name
DEVELOPER2'S LOCAL BRANCH
Always Keep local master up to day Don't write your code in master
Follow this step If there is no change In remote master
If there is a change in remote master follow this step to get latest code in local branch
WRITE YOUR CODE CREATE A NEW VERSION
WRITE YOUR CODE CREATE A NEW VERSION
GIT CHECKOUT MASTER
GIT PULL
GIT CHECKOUT BRANCH
GIT MERGE MASTER
CREATE A NEW VERSION
GIT PUSH
git push
DEVELOPER 2'S REMOTE REPO
1. Create pull request
2. Send code to remote master
3. After this step, team should update their local master(first step)
REMOTE MASTER

Basic Level: GitHub WorkFlow

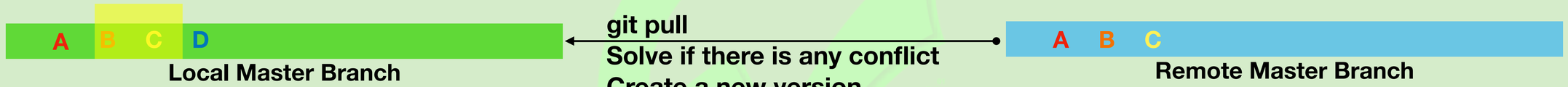
1)



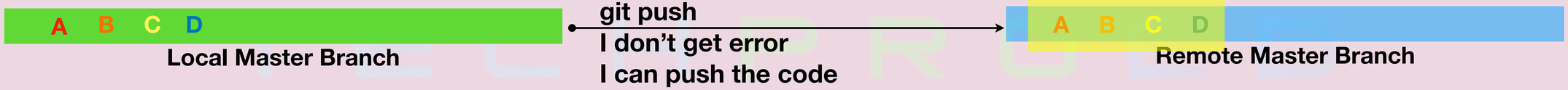
2)



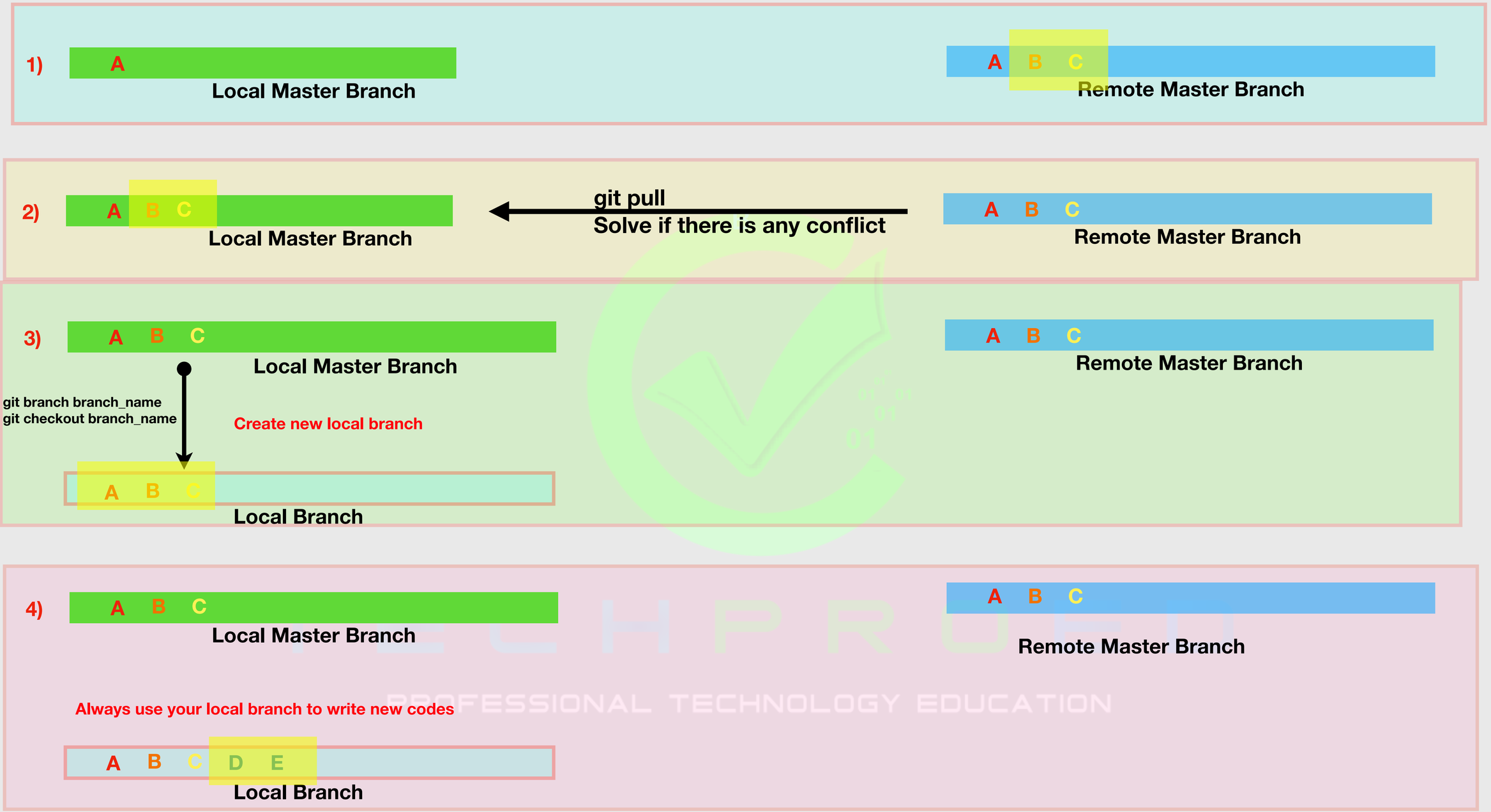
3)



4)



Middle Level: GitHub WorkFlow Slide 1

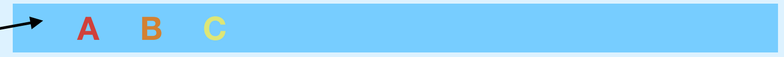


Basic Level: GitHub WorkFlow Slide 2

5)

Local Master Branch

We can't send directly from local branch to remote master

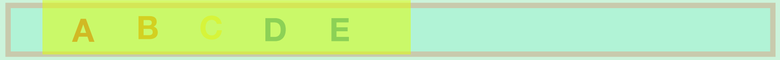


Remote Master Branch

Local Branch

PULL REQUEST:
git push

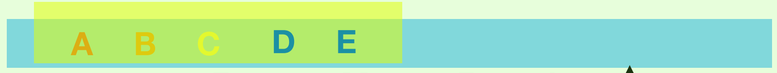
Push to your own remote repository



Local Branch

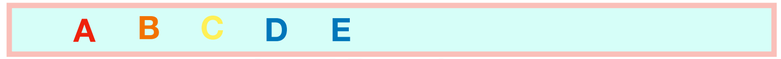
6)

Local Master Branch



Remote Master Branch

Create pull request
Merge pull request

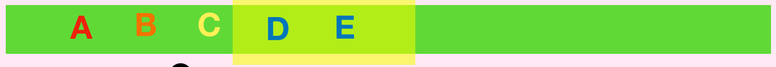


Local Branch

Responsible dev creates new pull request and checks the new codes
Merges the codes to master from local. This is called "Merge pull request"
They all happened in remote github

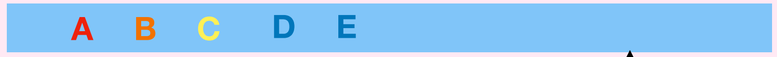
Keep your local master always up to date

7)



Local Master Branch

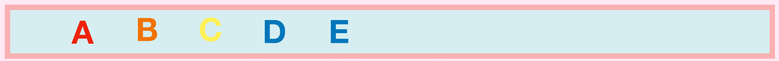
git checkout master
git pull
conflict olusursa coz



Remote Master Branch

Create pull request
Merge pull request

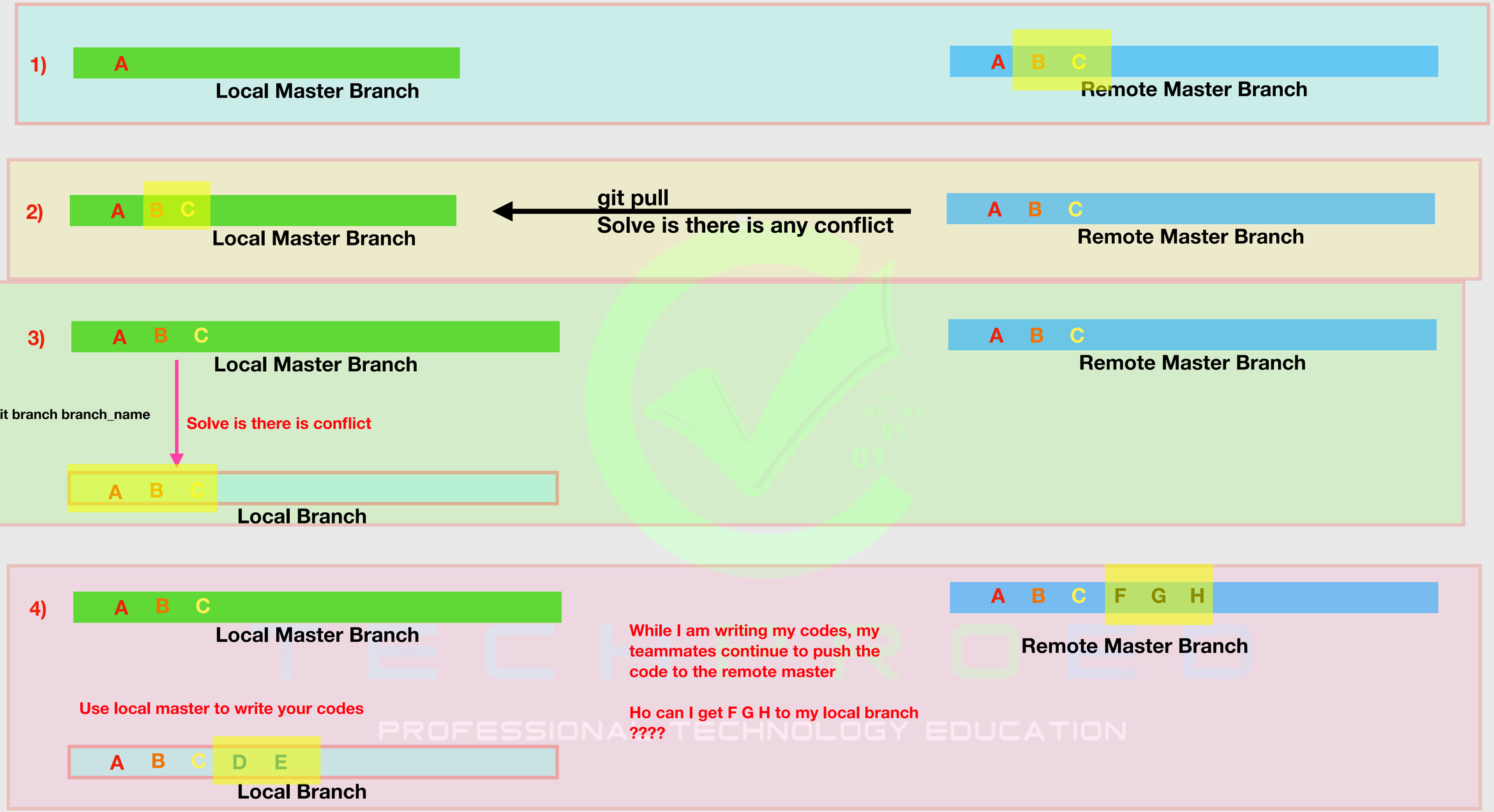
git branch branchismi
git merge master



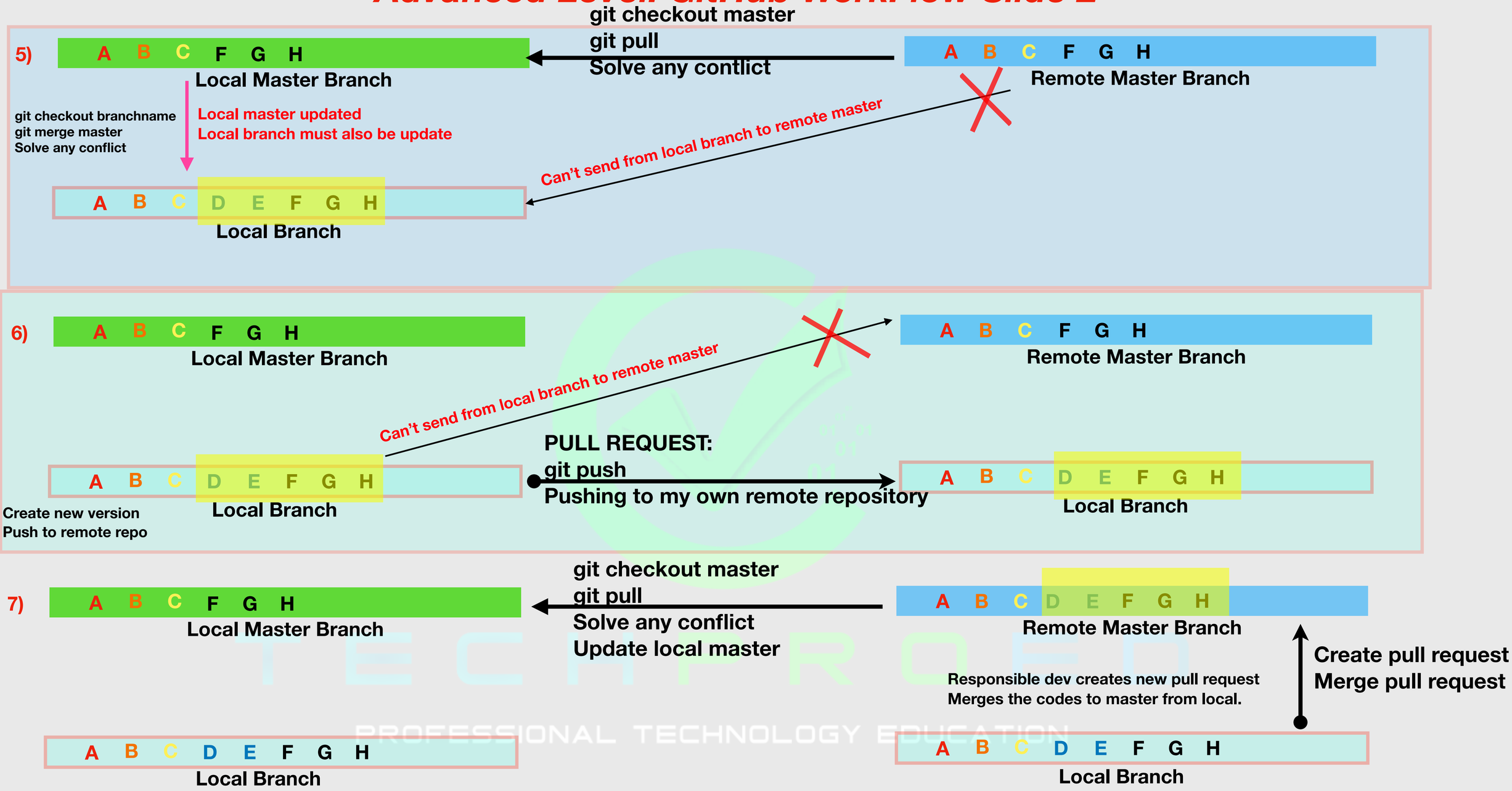
Local Branch

Local Branch

Advanced Level: GitHub WorkFlow Slide 1

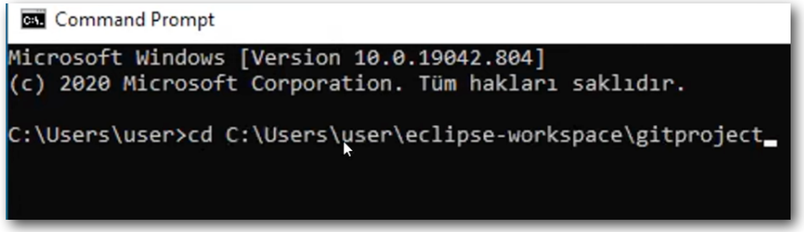


Advanced Level: GitHub WorkFlow Slide 2



EXTRA SCREENSHOT FOR SENDING GITHUB

1



2

```
C:\Users\user\eclipse-workspace\gitproject>git init
Initialized empty Git repository in C:/Users/user/eclipse-workspace/gitproject/.git/
```

3

```
C:\Users\user\eclipse-workspace\gitproject>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .classpath
        .project
        .settings/
        bin/
        src/
```

4

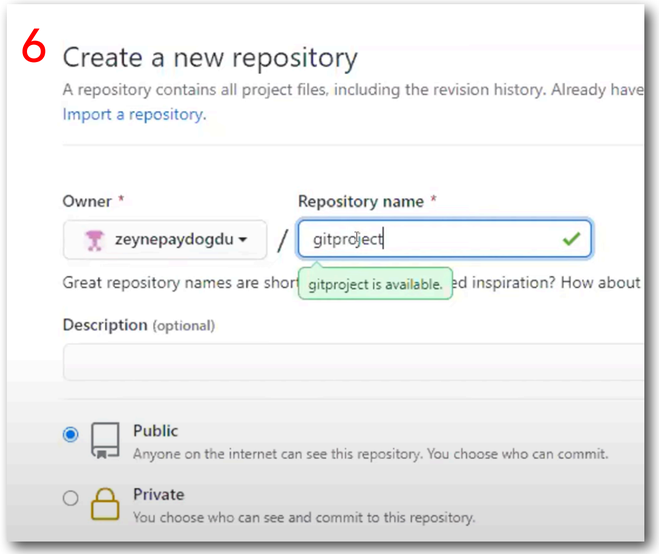
```
C:\Users\user\eclipse-workspace\gitproject>git add .
```

5

```
C:\Users\user\eclipse-workspace\gitproject>git commit -m "version 1"
[master (root-commit) 17ff831] version 1
 5 files changed, 51 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.jdt.core.prefs
 create mode 100644 bin/gitpackage/GitClass.class
 create mode 100644 src/gitpackage/GitClass.java
```

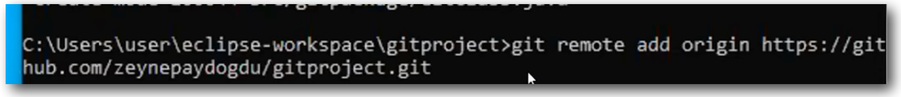
EXTRA SCREENSHOT FOR SENDING GITHUB

6



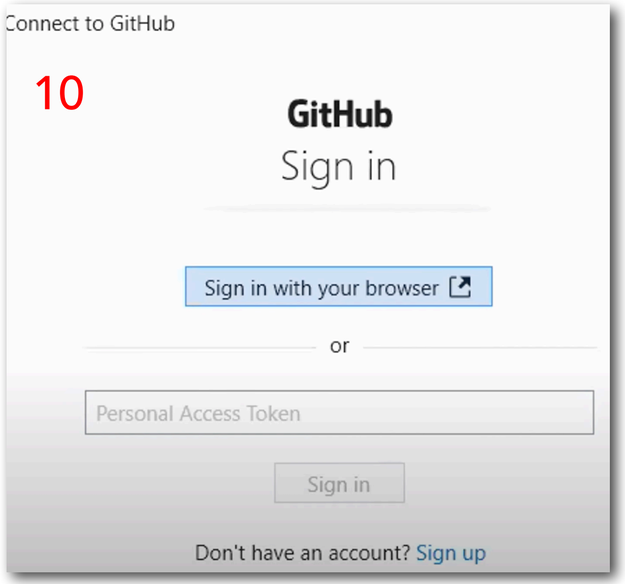
7

8

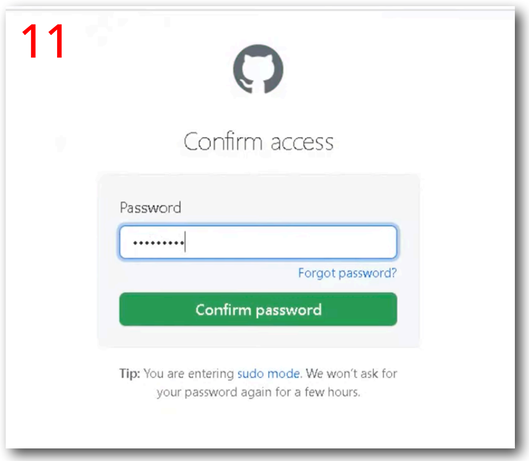


9

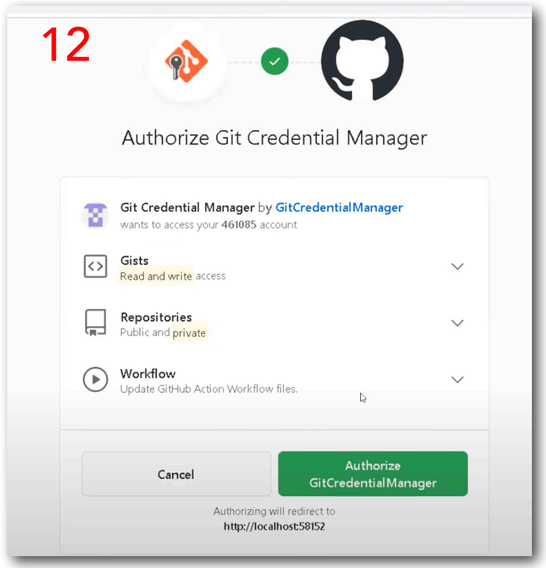
10



11



12



13