IRON
HACK

BRANCHES

COLLABORATING IN THE SAME PROJECT: BRANCHES

- A branch is simply **a different named version of the project files**.

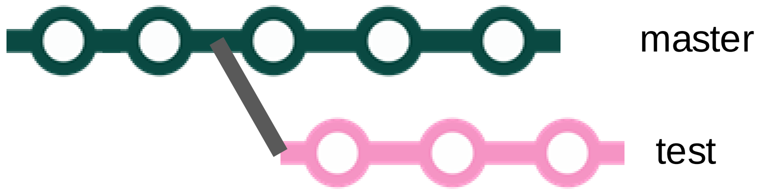
- When several developers want to add their changes to a GitHub repository, the way to do it is to create **one branch for each developer**.

- Then at the beginning of the day **each developer** pulls the latest changes from the GitHub repository.

- Creates a new branch with his/her name or topic to fix with:
  - *git branch branch_name*

- Switches to his/her own branch, :
  - *git switch branch_name*

  and start working as usuall creating commits.:



- At the end of the day, pushes his/her branch to the GitHub repo using:
  - *git push origin branch_name*

PROJECT BRANCHES:

- Usually, a project will have the following branches:
  - main/master
  - develop
  - developer_1
  - developer_2
  - …

- The “main/master” branch is expected to be the “gold true” and should only contain working code. It’s like a “backup” version.

- Experimental code should be in the “develop” branch. All the developers contributions will be added to the “develop” branch.

- Once the “develop” is fully functional, it will be added to the “main/master”

GIT MERGE

MERGING BRANCHES:

- From time to time, once the developers have a working code in their own branches, they would like to incorporate their changes in the “develop” branch. This process is called “merge”.

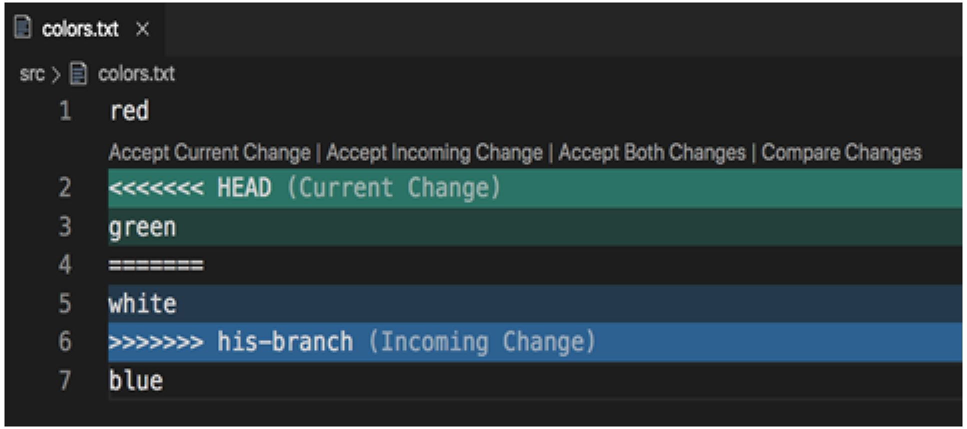
- To merge two branches together, we need to do the following steps:
  - switch to the “destination branch” in which we want to add the changes:
    - *git **switch** branch_name*
  - Then combine the changes of both branches using:
    - *git **merge** developer_branch*

- When we merge two branches, we can incidentally create a “**conflict**”, because the same section of one or more files has been modified in a different way by two developers.

- In such cases, git will report us that there is a conflict on one or more files and it will edit the corresponding files adding some marks.

Branches: solving con

- To solve this problem we need to:
    - Edit the file and remove the changes that we don’t want alongside with the marks <<<<<<HEAD ====== >>>>>>branch_name and save it.
    - git add modified_file
    - git commit -m “Merged branches and solving conflict”


colors.txt
src > colors.txt
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

```
red
<<<<<<< HEAD (Current Change)
green
=======
white
>>>>>>> his-branch (Incoming Change)
blue
```