1. **Source Tree**:
   Source tree is *a free Git desktop client for developers on windows*. Say goodbye to the command line and use the full capabilities of git through source tree's beautifully simple interface.

2. **Master branch and branch**:
   A branch is essentially is a unique set of code changes with a unique name. Each repository can have one or more branches. The main branch-*the one where all changes eventually get merged back into, and is called as a master brancher.*

3. **How to properly use the command of in, commit and push:**
   1. On the computer, move the file that we would like to upload to Github into the local directory that was created when we cloned the directory.
   2. Open Git Bash by clicking any white space within the file.
   3. Enter the command of' $ git init' to initialize files needed to be uploaded onto the repository.
   4. Stage the file for commit to the local repository by using the command:

      ```
      $ git add .
      ```

   5. Commit the file that we have staged in our local repository by the command:

      ```
      $ git commit -m "Add existing file"
      ```

   6. Push the changes in our local repository to Github.

      ```
      $ git push origin your-branch
      ```

4. **Check out command:**
   Definition: The git checkout command lets you navigate between the branches created by git branch. Checking out a branch updates the files in the working directory to match the version stored in that branch and it tells Git to record all new commits on that branch.

   Existing branches:
   Assuming the repro that we are working in contains pre-existing branches, we can switch between these branches using git checkout. To find out what branches are available and what the current branch name is, execute git branch.

   ```
   $> git branch
   master
   another_branch
   feature_inprogress_branch
   $> git checkout feature_inprogress_branch
   ```

New branches:

The git branch command can be used to create a new branch. When we want to start a new feature, we can create a new branch off master using git branch new_branch. Once created we can then use git checkout new_branch to switch to that branch.

```
git checkout -b <new-branch>
```

Switching branches:

Switching branches is a straightforward operation. Executing the following will point HEAD to the tip of <branchname>:

```
git checkout <branchname>
```

Git Checkout a Remote Branch:

When collaborating with a team, it might be common to utilize remote repositories. These repositories may be hosted and shared or they may be author colleague's local copy. Each remote repository will contain its own set of branches. In order to checkout a remote branch, we have to first fetch the contents of that branch.

```
git fetch --all
```

In modern versions of Git, you can then checkout the remote branch like a local branch.

```
git checkout <remotebranch>
```

Additionally, we can check-out a new local branch and reset it to the remote branches last commit:

```
git checkout -b <branchname>
git reset --hard origin/<branchname>
```

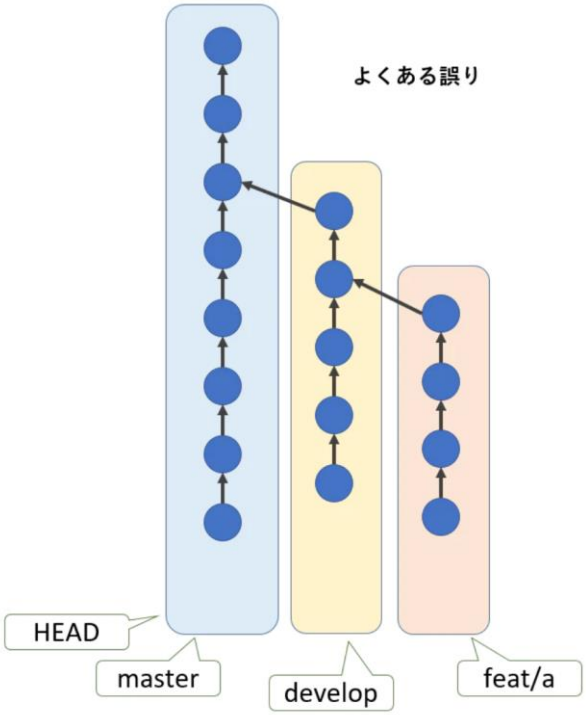
5. **Conceptions of HEAD:**

A pointer directs to the occasion that the developing is currently working on.

Detached Head:

A status that HEAD is not well located into the position of the current branch.

A sort of *misunderstanding* is that HEAD will be constantly located in the position of master for *all the series* instead of being checked-out into the new branch created by developers.



The *correct understanding* about the exact location of HEAD is the following: