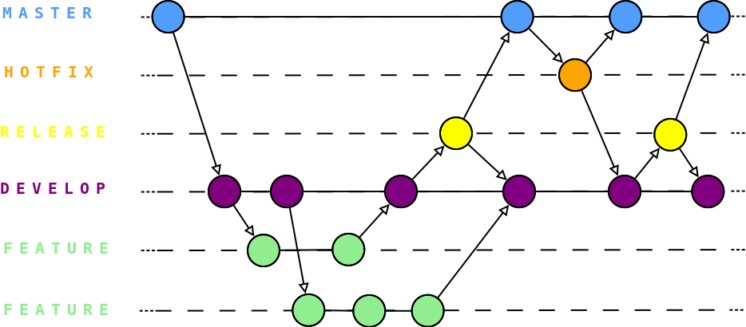
GIT FUNDAMENTAL HOMEWORK

# Git flow & branching model

The Git flow Workflow defines a strict branching model designed around the project release and provides a robust framework for managing larger projects. This flow is described as below:



There are basically 5 different branches:

* + master – branch without direct access for the developers. The purpose of this single branch is to keep the most stable deployable version of the code.
  + develop – this is the focus point for the developers where all feature branches are created from and later merged to. Same as with master, no one operates directly on this branch.
  + feature – a set of branches. Here you develop new functionalities! Always derived from develop and always merged with it.
  + release – a set of branches. When your development feature reaches the form, you want to release a new version of the app, this is release.
  + hotfix – a set of branches. When a new bug is discovered on production you create a new hotfix branch from master and after fixing it you merge it back to master as well as develop.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Derived from | Merge to (use PRs) | Life cycle |
| master | - | - | Initiate |
| develop | - | - | Initiate |
| feature | develop | develop | Deleted after merge |
| release | develop | master, develop | Deleted after release |
| hotfix | master | master, develop | Deleted after merge |

# Questions

Scenario: the team has 5 developers (Team leader – TL, DevA, DevB, DevC, DevD), following Scrum/Agile methodology and **uses basic Git flow (section 1)**.

**Q1**. For source code management, at sprint 0, TL want to create a private Git repository named “basic\_git\_flow” on Github with default “master/main” branch. (hints: go to Github and do somethings then take a screen shot with browser url).

Answer:

Create new repository

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, website

Description automatically generated

Default branch:

Graphical user interface, text, application, email

Description automatically generated

**Q2**. At sprint 0, TL want to create “develop” branch, branch contains “src” folder and file “commonBehavior” in “src” folder. Write Git commands to the requirement.

(hints: clone repo, checkout master, create new branch, add some files, stage changes, commit, push)

Answer:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Q3**. At Sprint 0, DevA must do task which is “feat1”, then DevA creates new branch named “feature/feat1” based on “develop” branch and add file “feat1” to “src” folder. After finishing this task, DevA creates a pull request to merge code from “feature/feat1” into “develop” branch. Write Git commands to do the requirement then go to Github merge the PR.

(hints: as same as Q2 hints and what is git pull request command) Answer:

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Q4**. At Sprint 0, DevC and DevD must do task which is “feat2”. DevC creates new branch named “feature/feat2” based on “develop” branch and add file “feat2” to “src” folder. DevC and DevD work on the same file “feat2”. After finishing this task, DevD creates a pull request to merge code from “feature/feat2” into “develop” branch. Write Git commands to do the requirement (include commands to use merge tools, assumes that conflicts are automatically merge).

(hints: how to decrease the conflicts when DevC/DevD commit changes on the same file/branch or what they do before each committing changes)

Answer:

Text

Description automatically generated

Edit the file feat2file.txt:

Graphical user interface, text, application, email

Description automatically generated

Solve conflict (Still unable to solve conflict):

Text

Description automatically generated

**Q5**. At sprint 0, after DevD create a pull request to merge code from “feature/feat2” into “develop”. TL review the PR and see that there are many commits on “feature/feat2”. TL wants to do “rebase” commits on this branch before merging PR. Write Git commands to do the requirement then go to Github to merge the PR. (hints: make 5 - 8 commit changes, git rebase interactive/squash on one branch, force push after finishing rebase process).

Answer:

Text

Description automatically generated

**Q6**. At sprint 1, DevA must do task which is “feat3”. DevA creates new branch named “feature/feat3” based on “develop” branch and add file “feat3” to “src” folder. At the same time, DevB do task to add some extra behavior to project. DevB creates new branch named “feature/ex\_behavior” based on “develop” branch and add file “extraBehavior” to “src” folder. After some days, DevA and Dev realize that

“feature/feat3” can re-use code of “feature/ex\_behavior”. They accept merge code from “feature/ex\_behavior” into “feature/feat3” and only make PR to merge code from “feature/feat3” to “develop”. Write Git commands to do the requirement and at the part of [merge code from “feature/ex\_behavior” into “feature/feat3”], provide 2 ways of merging branches (git merge and git rebase).

(hints: git merge branches, git rebase onto, force push after finishing rebase process)

Answer:

**Q7**. After n-1 sprint, team must create a release branch to go live some features. TL create release branch named “release/r1” based on “develop”. For uat/production environments, team must update some log message, then team adds

“extraLogMessage” file to “src” folder. As your understanding of Git flow (section 1),

write Git commands to ensure that release branch and its changes are merged into develop/master.

Answer: