Git Branches and Workflow

Many features, many developers; How can Git help?

1. Git Basics
   (Good for ~1 person)

2. Merging Conflicts
   (Needed for 2+ people)

3. Using GitLab
   (Managing a team)
Issues in GitLab

- GitLab tracks Issues:
  - Demo: Show issues in an active repo

- Value of Issues
  - Use as product’s backlog
  - Assign issue to a dev to show who's working on it
  - Update issue with extra info as needed
Branches

- Main source code *branch* in a Git repo.
- Latest code on master.

- Chaotic Commits
  - Too chaotic to have many teammates constantly committing code to master.
  - Solution:..

- Branch (Feature Branch)
  - Do work on separate track (the branch) from Master
  - Commit changes to your branch
  - When feature is ready,
Issue and Branching Overview

GL = done in GitLab
AS = done in Android Studio

- GL: Pick an issue to implement & create branch.
- AS: Checkout branch, make changes, commit & push changes to the branch.

When feature is ready

- AS: Merge Master to Feature branch (resolving conflicts); commit/push changes.
- GL: Create merge request to merge branch to Master.
- GL: Branch on GitLab is “deleted” when merge request is accepted (manually remove merged local branch)
Issues and Branching

1. Create issue for bug/feature
   - Implementing a feature or fixing a bug should start with a GitLab issue.
   - Ex: Issue 14: "Add help button to game activity"

2. Assign issue to yourself
3. Create feature branch in GitLab
   - GitLab names the branch..
     - Ex: 14-game-help-button
   - In Android Studio
     a) fetch to get new branch names
        VCS --> Git --> Fetch
     b) checkout the branch
        - Bottom-right “Git” button
        - Under remote branches, select the new one
        - On sub-menu, select checkout
4. **Work on your branch**
   - Do your work changing files
   - Check-in your changes via Git:
     - **add** changes ready to be committed
     - **commit** put changes into local repo on branch
     - **push** push to remote repo on branch
6. ..

- Get latest from master’s HEAD
  In Android Studio:
  VCS --> Git --> Merge Changes...
- Resolve merge conflicts;
  test;
  add/commit/push any changes

If box is empty then nothing new on master to merge.
7. Submit a... via GitLab

- Create request to merge your branch back to master
- Since you already merged Master to Feature Branch, there should be no conflicts.
- GitLab will close issues associate with merge request;
Managing Merge Request

- Team members see merge requests and:
  - **Code review:** Comment on problems they see in the code (possibly leading to new commits to fix)
  - Thumbs-up/down for voting
- **Repo Manager accepts merge request**
  - Accepting merge requests will:
    - merge code to master (should be no conflicts)
    - delete the source branch [optional; good practice to clean up]
GitLab Feature Branch Demo

- **Setup**
  - Create new Android Studio project
  - Create GitLab project; commit/push

- **Create GitLab issue**
  - “Generate random number”

- **Solve an issue**
  - In GitLab: Assign to you & create feature branch
  - In A.S.: Update or fetch; Checkout a “remote branch” (bottom-right)
  - Code; Commit; Push
  - In A.S.: Merge **master to feature** (VCS-->Git-->Merge Changes...)
  - In GitLab: Create merge request
  - In GitLab: Accept merge request.

- **Cleanup in Android Studio**
  - Switch to master, pull, view log
  - Delete local branch (if needed)
GitLab Workflow
Feature Branch, Merging Changes, Merge Request

1. Create GitLab issue.
2. Assign issue to self
3. Create feature branch
4. Pull to update.
5. Checkout local feature branch.
6. Change files, commit, push.
7. Merging Changes.
8. Merge Master to feature branch.
9. Resolve conflicts.
11. Commit/push changes.
12. Switch to Master branch.

Legend
In GitLab
In Android Studio

Teammate Feature Branches
Master
My Feature Branch

Create & accept Merge Request
1. Pull.
2. Remove local feature branch.

Sequence of Events
Review Exercise

What is the ordering of the following steps:

a) In GitLab: Assign to you & create feature branch
b) In A.S.: Merge master to feature branch (VCS-->Git-->Merge Changes…)
c) Create GitLab issue
d) In A.S.: Checkout feature branch (bottom-right)
e) In GitLab: Accept merge request.
f) Code; Commit; Push
g) In GitLab: Create merge request
Summary

• **Branches and Workflow**
  - Create GitLab *issues*.
  - Do work on a *feature branch*.
  - GitLab *merge request* to merge branch to master.
  - Git merge command