Class Notes: Distributed Version Control - Session 3 (Git & GitHub - TL3)

Session Objectives

By the end of this session, students will be able to:

- Identify how to create a new branch.
- Outline steps to develop code on multiple branches.
- Outline steps to modify, rename, and delete a branch.
- Identify how to merge branches and resolve basic merge conflicts.
- Describe how to create a remote branch.
- Describe how to pull from a remote connection.

1. Basic Branching and Merging

What is Branching?
Branching allows parallel development in a Git repository. Each branch is an independent line of work, enabling experimentation without affecting the main codebase (typically the `main` or `master` branch).

Why Use Branches?

- Isolate features, bug fixes, or experiments.
- Enable team collaboration without conflicts.
- Facilitate code reviews before merging.

Merging:
Merging combines changes from one branch into another, integrating work into the main codebase.

Example Workflow:

- Create a branch for a new feature.
- Develop and commit changes on the branch.
- Merge the branch into `main`.

2. Creating a New Branch

Command:

```
git branch <branch-name>
git checkout <branch-name>
```

Or combine both:

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

Example:

```
# Create and switch to a new branch
git checkout -b feature/login
# Create a file
echo "Login page" > login.html
git add login.html
git commit -m "Add login page"
```

Notes:

- The new branch starts from the current commit.
- `git branch` lists all branches, with `*` indicating the current branch.

3. Delete, Rename, and Merge Branches

Deleting a Branch:

- Delete a branch after merging (if no longer needed):

  ```
  git branch -d <branch-name>
  ```

- Force delete an unmerged branch:

```
git branch -D <branch-name>
```

Renaming a Branch:

- Rename the current branch:

```
git branch -m <new-name>
```

- Rename another branch:

```
git branch -m <old-name> <new-name>
```

Merging Branches:

- Switch to the target branch (e.g., `main`):

```
git checkout main
```

- Merge the source branch:

```
git merge <branch-name>
```

Example:

```
# Create and work on a branch
git checkout -b bugfix/issue-123
echo "Fix bug" > app.js
git add app.js
git commit -m "Fix issue 123"
# Merge into main
git checkout main
git merge bugfix/issue-123
# Delete the branch
git branch -d bugfix/issue-123
# Rename a branch
git checkout -b old-name
git branch -m new-name
```

Merge Conflicts:

- Occur when Git can't automatically merge changes (e.g., same line edited in both branches).
- Resolve manually by editing the conflicting files, then:

```
git add <file>
git commit
```

4. Basic Merging to Main Branch

Workflow:

1. Ensure the feature branch is complete and tested.
2. Switch to `main`: `git checkout main`.
3. Pull latest changes: `git pull origin main`.
4. Merge the feature branch: `git merge feature-branch`.
5. Resolve conflicts if any.
6. Delete the feature branch: `git branch -d feature-branch`.

Example:

```
# Complete work on feature branch
git checkout feature/login
echo "Add form" >> login.html
git add login.html
git commit -m "Add login form"
# Merge into main
git checkout main
git merge feature/login
# Delete branch
git branch -d feature/login
```

5. Remote Branches in Git

What are Remote Branches?

Remote branches are branches hosted on a remote repository (e.g., GitHub). They are pointers to the state of branches in the remote repo.

Creating a Remote Branch:

- Push a local branch to the remote:

```
git push origin <branch-name>
```

Tracking a Remote Branch:

- Set up a local branch to track a remote branch:

```
git checkout --track origin/<branch-name>
```

Example:

```
# Create and push a branch
git checkout -b feature/signup
echo "Signup page" > signup.html
git add signup.html
git commit -m "Add signup page"
git push origin feature/signup
# List remote branches
git branch -r
```

6. Pulling in Git

What is Pulling?
Pulling fetches changes from a remote repository and merges them into the current branch.

Command:

```
git pull origin <branch-name>
```

- Equivalent to `git fetch` + `git merge`.

Example:

```
# Pull changes from main
git checkout main
git pull origin main
```

Notes:

- Resolve merge conflicts if changes overlap.
- Regularly pull to stay updated with team changes.

Classwork: Branching and Remote Operations

Objective: Create branches, merge changes, push to a remote repository, and pull updates.

Steps:

1. **Set Up a Local Repository**:
   - Create a new repository:
     ```
     mkdir team-project
     cd team-project
     git init
     echo "# Team Project" > README.md
     git add README.md
     git commit -m "Initial commit"
     ```
2. **Create and Merge a Branch**:
   - Create a branch `feature/homepage`:
     ```
     git checkout -b feature/homepage
     ```
   - Add a file `index.html` with content: `<h1>Welcome</h1>`.
   - Commit changes:
     ```
     git add index.html
     git commit -m "Add homepage"
     ```
   - Merge into `main`:

```
git checkout main
git merge feature/homepage
git branch -d feature/homepage
```

3. **Work with Remote Branches** (use a GitHub repo provided by instructor or create one):
   - Add a remote:
     ```
     git remote add origin <repository-url>
     ```
   - Push `main`:
     ```
     git push -u origin main
     ```
   - Create and push a new branch `feature/contact`:
     ```
     git checkout -b feature/contact
     echo "<h1>Contact Us</h1>" > contact.html
     git add contact.html
     git commit -m "Add contact page"
     git push origin feature/contact
     ```
4. **Pull Changes**:
   - Simulate a teammate's update by pulling from `main`:
     ```
     git checkout main
     git pull origin main
     ```

Deliverable:
Submit screenshots of:

- `git log --oneline` after merging `feature/homepage`.
- `git branch -r` showing remote branches.
- `git status` after pulling from `main`.

Session Test

Instructions: Answer the questions or complete the tasks. Submit via OnlineVarsity.

Multiple-Choice Questions:

1. What does `git checkout -b feature` do?
   a) Deletes a branch
   b) Creates and switches to a new branch
   c) Merges a branch
   d) Pushes a branch
   Answer: b
2. What command pushes a local branch to a remote repository?
   a) `git pull origin branch`
   b) `git push origin branch`
   c) `git merge branch`
   d) `git clone branch`
   Answer: b

Practical Task:

1. Create a repository named `test-branching`.
2. Initialize it and commit a `README.md` with content: `# Test Branching`.
3. Create a branch `feature/about`.
4. Add a file `about.html` with content: `<h1>About Us</h1>`.
5. Commit with message: "Add about page".
6. Merge `feature/about` into `main`.
7. Delete the `feature/about` branch.
8. Create a branch `feature/footer` and push it to a remote repository (use instructor-provided URL or skip if unavailable).
9. Submit the output of:
   - `git log --oneline`
   - `git branch` (after deleting `feature/about`)
   - `cat about.html`

Expected Output:

- `git log --oneline`: Shows commits for `README.md` and `about.html`.
- `git branch`: Shows `main` and `feature/footer`.
- `about.html` content: `<h1>About Us</h1>`.

Self-Study (S1-S2)

- **Read**: Refer to Sessions 3 and 4 of "Version Control with Git and GitHub" on OnlineVarsity.
- **Practice**: Complete "Practice 4 Me" exercises for Sessions 3 and 4.
- **Assignment**: Solve scenario-based assignments on OnlineVarsity (e.g., simulate branching for a team project).
- **Explore**: Review Glossary and References on OnlineVarsity for terms like "merge conflict" and "remote branch".

References

- "Version Control with Git" by Jon Loeliger, Matthew McCullough (Library Reference).
- OnlineVarsity: Learner’s Guide (eBook), Glossary, FAQ, and References for Sessions 3 and 4.