Hooks and Customization

- Implement a pre-commit hook to enforce specific coding standards or checks before each commit.

- Explore other Git hooks like post-commit, pre-push, etc., and understand their use cases

Pre-Commit Hook Implementation

A **pre-commit hook** is a script that Git runs before committing code. It can be used to enforce coding standards, validate code, or run tests. Below is a step-by-step guide to implementing a pre-commit hook.

Steps to Implement a Pre-Commit Hook

1. **Create a Pre-Commit Hook File**
    - Navigate to the Git repository.
    - Open the .git/hooks directory.
    - Create or edit the pre-commit file:
    - touch .git/hooks/pre-commit
2. **Write the Script**
    - Add a script to check coding standards. For example:
    - #!/bin/bash
    -
    - echo "Running pre-commit checks..."
    -
    - # Example: Check for code formatting using Prettier
    - if ! npx prettier --check .; then
    - echo "Code formatting issues detected. Please fix them before committing."
    - exit 1
    - fi
    -
    - # Example: Run tests
    - if ! npm test; then
    - echo "Tests failed. Fix the issues before committing."
    - exit 1
    - fi
    -
    - echo "All checks passed!"
    - exit 0
3. **Make the Script Executable**
    - Ensure the hook is executable:
    - chmod +x .git/hooks/pre-commit
4. **Test the Hook**
    - Try committing changes. The hook will run before the commit completes.

Exploring Other Git Hooks

1. **Post-Commit Hook**
   - Runs after a commit is made.
   - Use case: Notify the team, update logs, or trigger CI/CD pipelines.
   - #!/bin/bash
   - echo "Post-commit hook triggered!"
   - # Example: Notify team via Slack
   - curl -X POST -H 'Content-type: application/json' \
   - --data '{"text":"A new commit has been made."}' \
   - https://hooks.slack.com/services/your/webhook/url
2. **Pre-Push Hook**
   - Runs before pushing changes to a remote repository.
   - Use case: Validate code or run tests to prevent broken builds.
   - #!/bin/bash
   - echo "Running pre-push checks..."
   -
   - # Example: Run linting
   - if ! npm run lint; then
   - echo "Linting failed. Fix issues before pushing."
   - exit 1
   - fi
   -
   - echo "All checks passed!"
   - exit 0
3. **Post-Checkout Hook**
   - Runs after checking out a branch.
   - Use case: Set up environment-specific files or configurations.
   - #!/bin/bash
   - echo "Post-checkout hook triggered!"
   - # Example: Load environment variables
   - if [-f .env]; then
   - source .env
   - fi
4. **Post-Merge Hook**
   - Runs after a merge is performed.
   - Use case: Perform database migrations or update dependencies.
   - #!/bin/bash
   - echo "Post-merge hook triggered!"
   - # Example: Install dependencies
   - npm install

Use Cases and Best Practices

- **Pre-Commit**: Check code quality (e.g., linting, tests).
- **Post-Commit**: Notify teams or log commits.
- **Pre-Push**: Prevent broken builds from reaching remote repositories.
- **Post-Checkout/Merge**: Manage environment-specific configurations.

By using these hooks effectively, you can enforce coding standards, automate tasks, and improve collaboration.