Lab Exercise 1- Working with Git Revert

Lab Exercise: Git Revert

This exercise will guide you through reverting changes in Git. The git revert command is used to create a new commit that undoes the changes introduced by a previous commit without modifying the history.

Objective

- Learn how to use git revert to undo changes from specific commits.

- Practice handling merge conflicts during a revert.

Prerequisites

1. Install Git on your system.

2. Set up a Git repository:

```
git init git-revert-lab

cd git-revert-lab
```

```
PS E:\> git init git-revert-lab
Initialized empty Git repository in E:/git-revert-lab/.git/
PS E:\> cd git-revert-lab
```

Steps

1. Set Up the Repository

1. Create a file:

```
echo "Line 1" > file.txt
```

```
PS E:\git-revert-lab> echo "Line 1" > file.txt
```

2. Stage and commit the file:

```
git add file.txt

git commit -m "Initial commit: Add file.txt with Line 1"
```

```
PS E:\git-revert-lab> git add .
PS E:\git-revert-lab> git commit -m"first commit"
[master (root-commit) ad28c78] first commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.txt
```

3. Add more changes:

echo "Line 2" >> file.txt

git commit -am "Add Line 2 to file.txt"

```
PS E:\git-revert-lab> echo "Line 2" >> file.txt
PS E:\git-revert-lab> git commit -am"second commit"
[master 8deb578] second commit
 1 file changed, 0 insertions(+), 0 deletions(-)
```

4. Add another change:

echo "Line 3" >> file.txt

git commit -am "Add Line 3 to file.txt"

```
PS E:\git-revert-lab> echo "Line 3" >> file.txt
PS E:\git-revert-lab> git commit -am"third commit"
[master 2e6ed61] third commit
 1 file changed, 0 insertions(+), 0 deletions(-)
```

5. Verify the commit history:

git log –oneline

```
PS E:\git-revert-lab> git log --oneline
2e6ed61 (HEAD -> master) third commit
8deb578 second commit
ad28c78 first commit
```

2. Revert the Last Commit

1. Revert the most recent commit:

```
git revert HEAD

PS E:\git-revert-lab> git revert HEAD
[master 909b5ea] Revert "third commit"
 1 file changed, 0 insertions(+), 0 deletions(-)
```

2. Git will open a text editor for the commit message. Save and close the editor to complete the revert.

3. Verify the contents of file.txt:

```
cat file.txt

PS E:\git-revert-lab> cat file.txt
Line 1
Line 2
```

4. Check the commit history:

```
git log –oneline
```

```
PS E:\git-revert-lab> git log --oneline
909b5ea (HEAD -> master) Revert "third commit"
2e6ed61 third commit
8deb578 second commit
ad28c78 first commit
```

3. Handle Merge Conflicts During Revert

1. Modify file.txt:

```
echo "Conflict Line" >> file.txt

git commit -am "Add Conflict Line"
```

```
PS E:\git-revert-lab> echo "Conflict Line" >> file.txt
PS E:\git-revert-lab> git commit -am "Add Conflict Line"
[master 6c21983] Add Conflict Line
 1 file changed, 0 insertions(+), 0 deletions(-)
```

2. Revert the second commit (8b2a1c1 again) to trigger a conflict:

```
git revert 8b2a1c1
```

```
PS E:\git-revert-lab> git revert 8deb578
warning: Cannot merge binary files: file.txt (HEAD vs. parent of 8deb578 (second commit))
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not revert 8deb578... second commit
hint: After resolving the conflicts, mark them with
hint: "git add/rm <pathspec>", then run
hint: "git revert --continue".
hint: You can instead skip this commit with "git revert --skip".
hint: To abort and get back to the state before "git revert",
hint: run "git revert --abort".
hint: Disable this message with "git config advice.mergeConflict false"
```

3. Git will indicate a conflict. Resolve it:

- Open file.txt and remove conflict markers.

- Keep the desired lines.

4. Stage the resolved file:

```
git add file.txt

PS E:\git-revert-lab> git add .
```

5. Complete the revert:

```
git revert –continue

PS E:\git-revert-lab> git revert --continue
[master 4f5d2a9] Revert "second commit"
 1 file changed, 0 insertions(+), 0 deletions(-)
```