Report No. 2.2: Git Merge Conflict

Course Code: CSE 404
Course Title: Software Engineering and ISD Laboratory

Submitted by

Suraiya Mahmuda (ID: 364)

Submitted to

Dr. Md. Mushfique Anwar,Professor
Dr. Md. Humayun Kabir,Professor

Computer Science and Engineering
Jahangirnagar University

Dhaka, Bangladesh

September 12, 2024

Contents

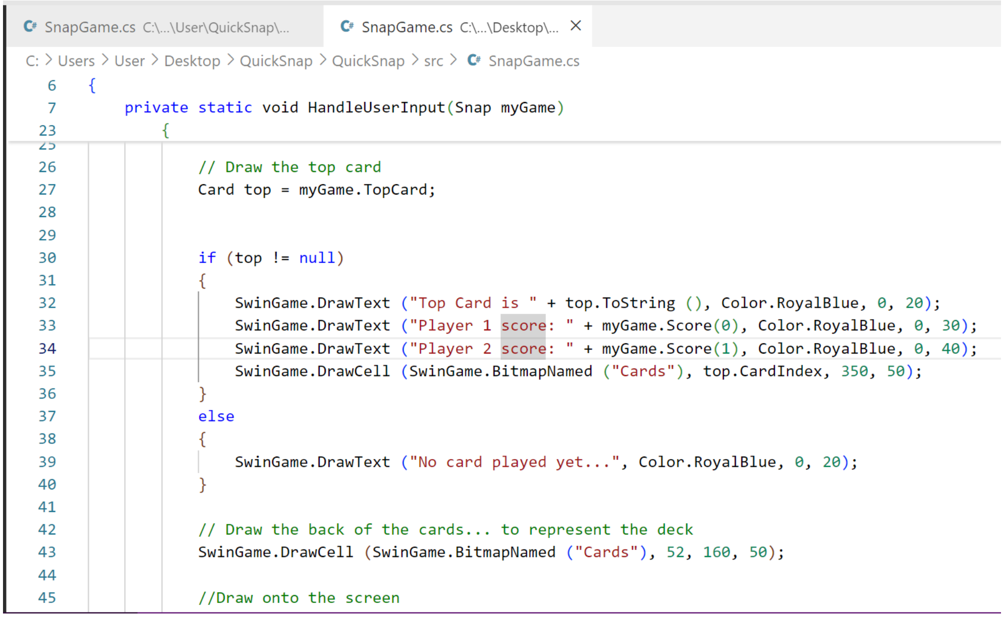
1. OBJECTIVE

The objective of resolving a Git merge conflict is to manually reconcile differences between branches when Git cannot automatically combine changes. It ensures that all contributions are integrated correctly without overwriting or losing any work.

2. GIT MERGE

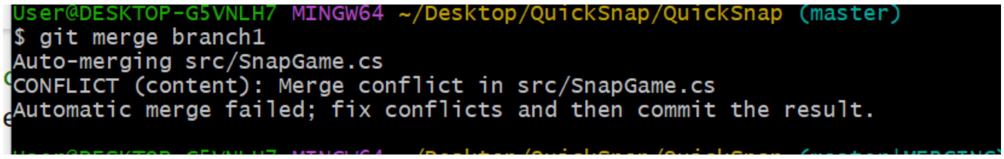
Git Merge is a command that integrates changes from one branch into another. It combines the histories of both branches, allowing developers to consolidate work from multiple lines of development without losing changes. If conflicts arise, they need to be manually resolved before completing the merge.

3. SCREENSHOT OF MY WORK


SnapGame.cs C:\...\User\QuickSnap\...
SnapGame.cs C:\...\Desktop\...
C: > Users > User > Desktop > QuickSnap > QuickSnap > src > SnapGame.cs

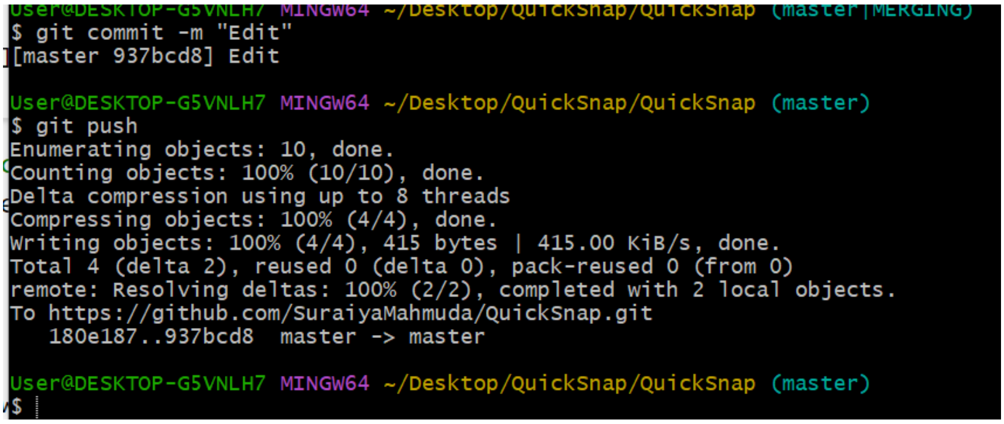
```
{
    private static void HandleUserInput(Snap myGame)
        {

            // Draw the top card
            Card top = myGame.TopCard;


            if (top != null)
            {
                SwinGame.DrawText ("Top Card is " + top.ToString (), Color.RoyalBlue, 0, 20);
                SwinGame.DrawText ("Player 1 score: " + myGame.Score(0), Color.RoyalBlue, 0, 30);
                SwinGame.DrawText ("Player 2 score: " + myGame.Score(1), Color.RoyalBlue, 0, 40);
                SwinGame.DrawCell (SwinGame.BitmapNamed ("Cards"), top.CardIndex, 350, 50);
            }
            else
            {
                SwinGame.DrawText ("No card played yet...", Color.RoyalBlue, 0, 20);
            }

            // Draw the back of the cards... to represent the deck
            SwinGame.DrawCell (SwinGame.BitmapNamed ("Cards"), 52, 160, 50);

            //Draw onto the screen
```

Figure 3.1: git merge



```
User@DESKTOP-G5VNLH7 MINGW64 ~/Desktop/QuickSnap/QuickSnap (master)
$ git merge branch1
Auto-merging src/SnapGame.cs
CONFLICT (content): Merge conflict in src/SnapGame.cs
Automatic merge failed; fix conflicts and then commit the result.
```

Figure 3.2: Merge Conflict



```
User@DESKTOP-G5VNLH7 MINGW64 ~/Desktop/QuickSnap/QuickSnap (master|MERGING)
$ git commit -m "Edit"
[master 937bcd8] Edit

User@DESKTOP-G5VNLH7 MINGW64 ~/Desktop/QuickSnap/QuickSnap (master)
$ git push
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 415 bytes | 415.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/SuraiyaMahmuda/QuickSnap.git
   180e187..937bcd8  master -> master

User@DESKTOP-G5VNLH7 MINGW64 ~/Desktop/QuickSnap/QuickSnap (master)
$ 
```

Figure 3.3: Conflict solved

4. DESCRIPTION

Here, I created a new branch called branch1 and made some changes. After pushing these changes to the remote repository, I attempted to merge branch1 into the master branch. During the merge, a conflict occurred as both branches had modified the same file. To resolve the conflict, I manually edited the file, choosing how to combine the changes and marked the conflict as resolved. Finally, I completed the merge and pushed the merged changes back to the remote repository, ensuring the project was up to date and conflict-free.