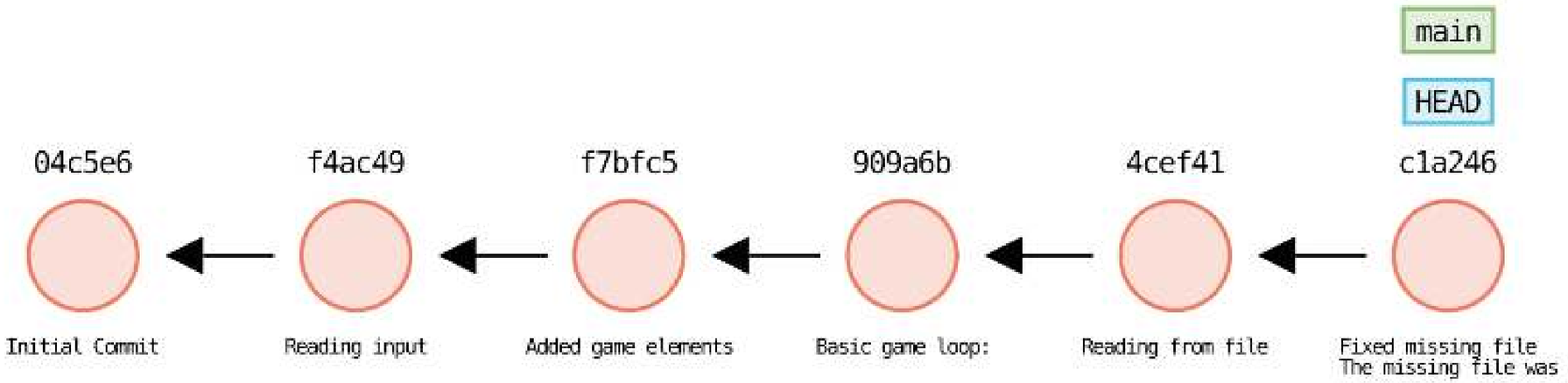
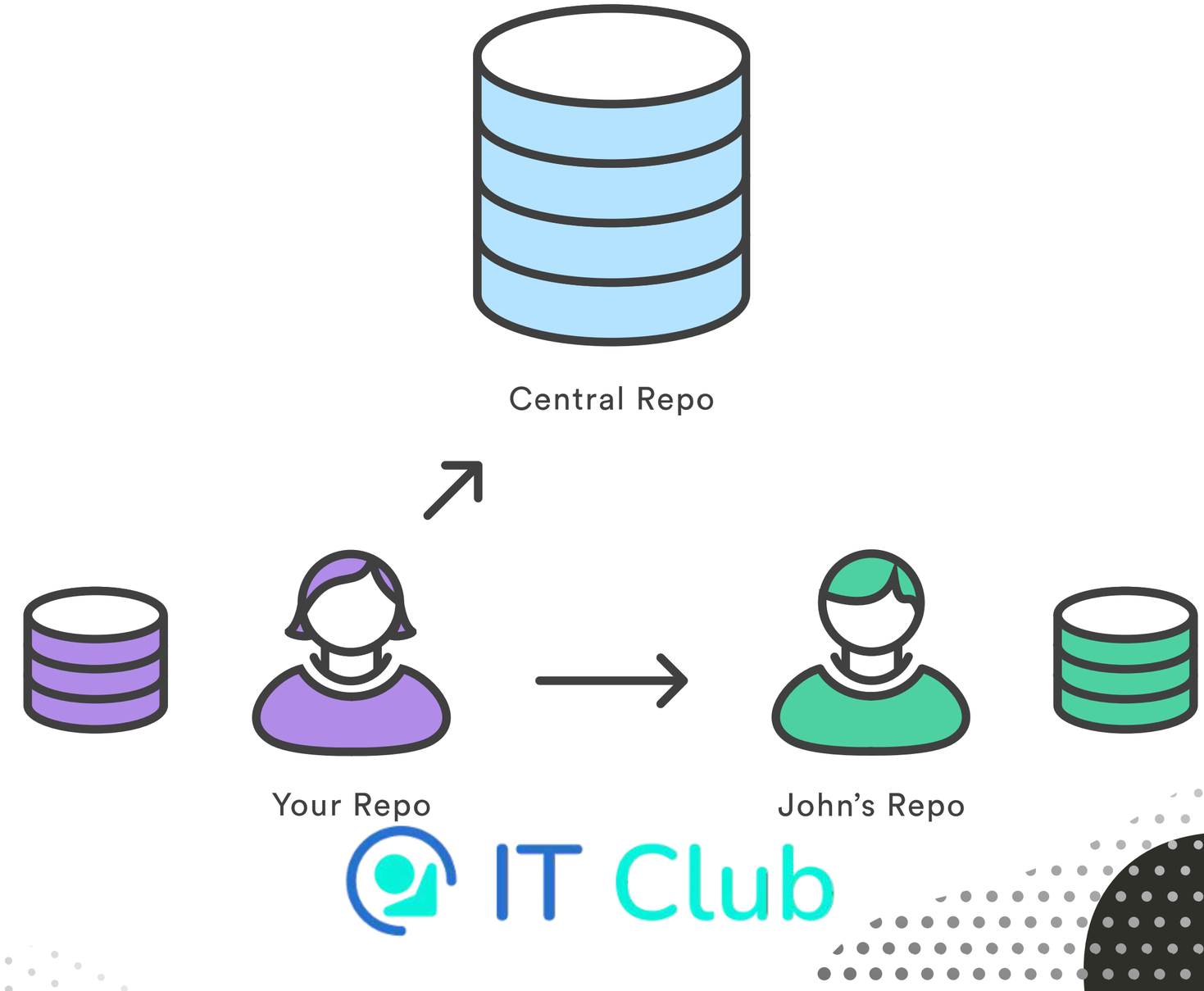
BRANCHING AND MERGING

Up until now...


main
HEAD
04c5e6
f4ac49
f7bfc5
909a6b
4cef41
c1a246
Initial Commit
Reading input
Added game elements
Basic game loop:
Reading from file
Fixed missing file
The missing file was

Recap...



IT Club

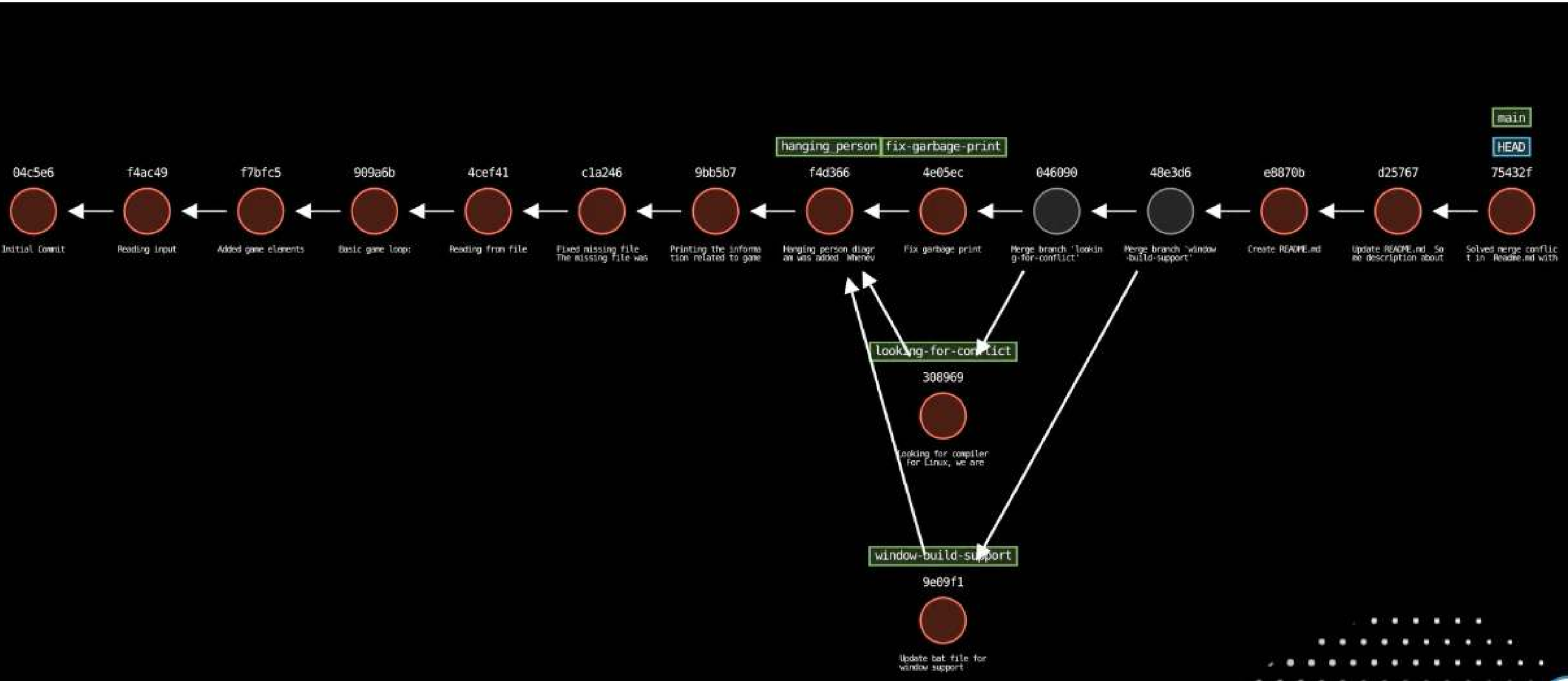
Setting up

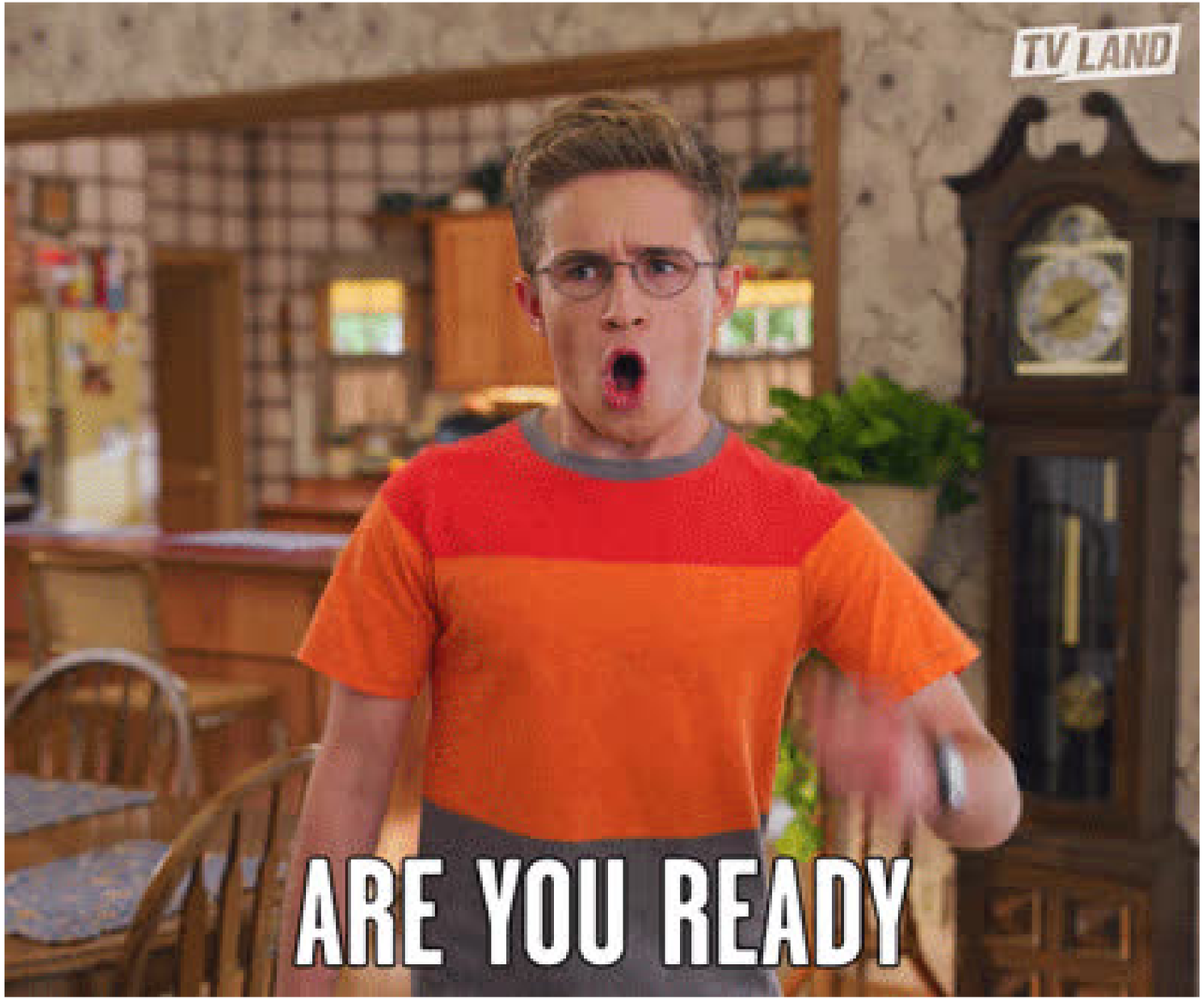
Objectives

- Understanding Branching
- Create and switch Branches
- Changing and committing files
- Merge and Resolve Conflicts

IT Club

What will we be doing?

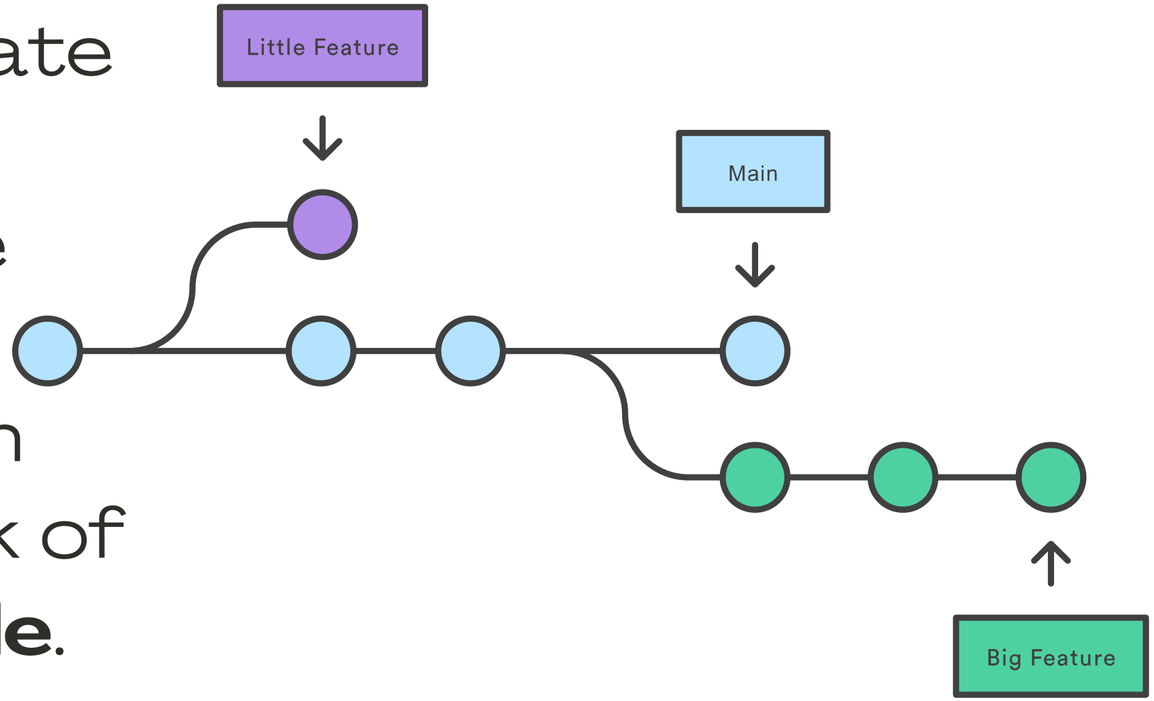




IT Club

Branch

Branching in Git is like creating alternate copies of your project to try out different ideas without affecting the main project. It allows you to work on features separately and merge them back later when they are ready. Think of it as **parallel universes for your code**.



IT Club

Creating Branch

To create a branch in Git, you can use the following command:

```
git branch <branch_name>
```

```
git branch hanging_man
```

IT Club

Rename Branch

To rename a branch in Git, you can use the following command:

```
git branch -m <old_branch_name> <new_branch_name>
```

```
git branch -m hanging_man hanging_person
```

IT Club

Switch Branch

To switch a branch in Git, you can use the following command:

```
git switch <branch_name>
```

```
git switch hanging_person
```

IT Club

Publish Branch

To publish a branch in Git, you can use the following command:

```
git push <remote_name> <branch_name>
```

```
git push origin hanging_person
```

IT Club

Deleting Branch

To delete a branch in Git, you can use the following command:

```
git branch -d <branch_name>
```

```
git branch -d hanging_person
```

IT Club

Why Branching?

- Isolate Changes
- Parallel Development:
- Experiment Freely
- Easy Collaboration

IT Club

IT Club

Merging

Merging in Git combines changes from one branch into another to create a unified codebase.

Types of Merging:

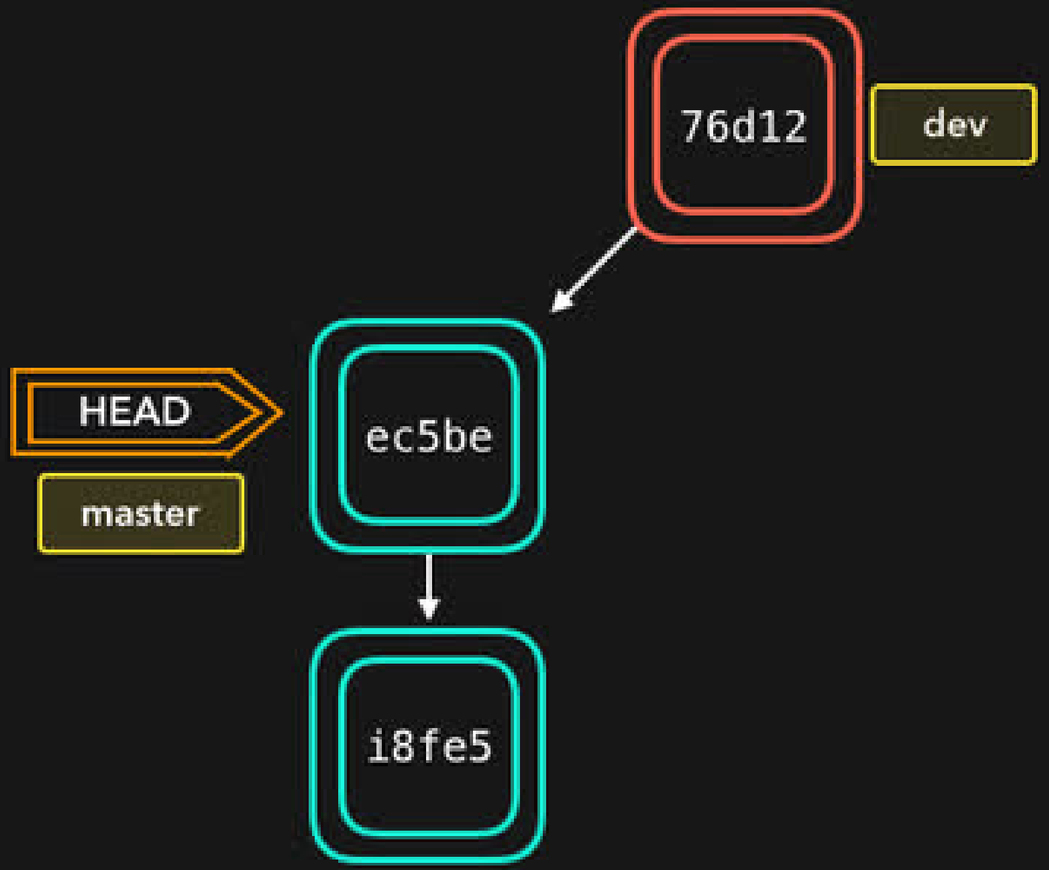
1. Fast-Forward Merge: Simple move of branch pointer.
2. Automatic Merge: Git combines changes automatically.
3. ORT Merging: Manual intervention needed for conflicting changes.

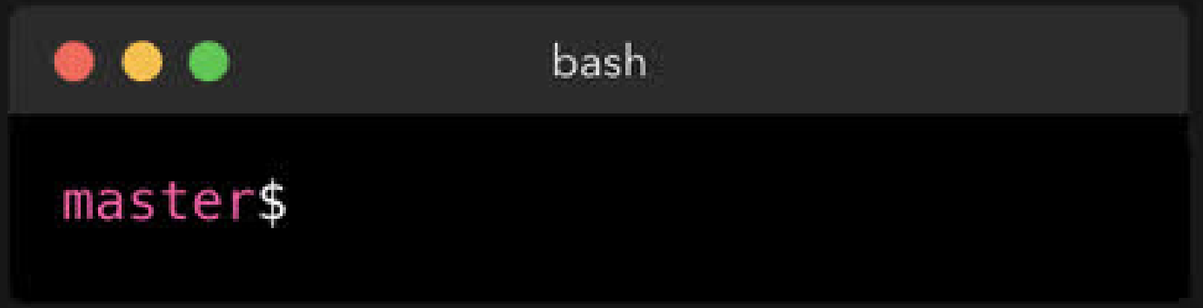
IT Club

Auto - Merging

Auto-merging is like having two friends write different parts of a story without knowing what the other wrote. When you want to combine their work, an auto merge is when the computer figures out how to put their parts together without any conflicts.

Auto - Merging





Git | Merging (fast-forward)

Default behavior when the merging branch has all of the current branch's commits

Doesn't create a new commit, thus doesn't modify existing branches

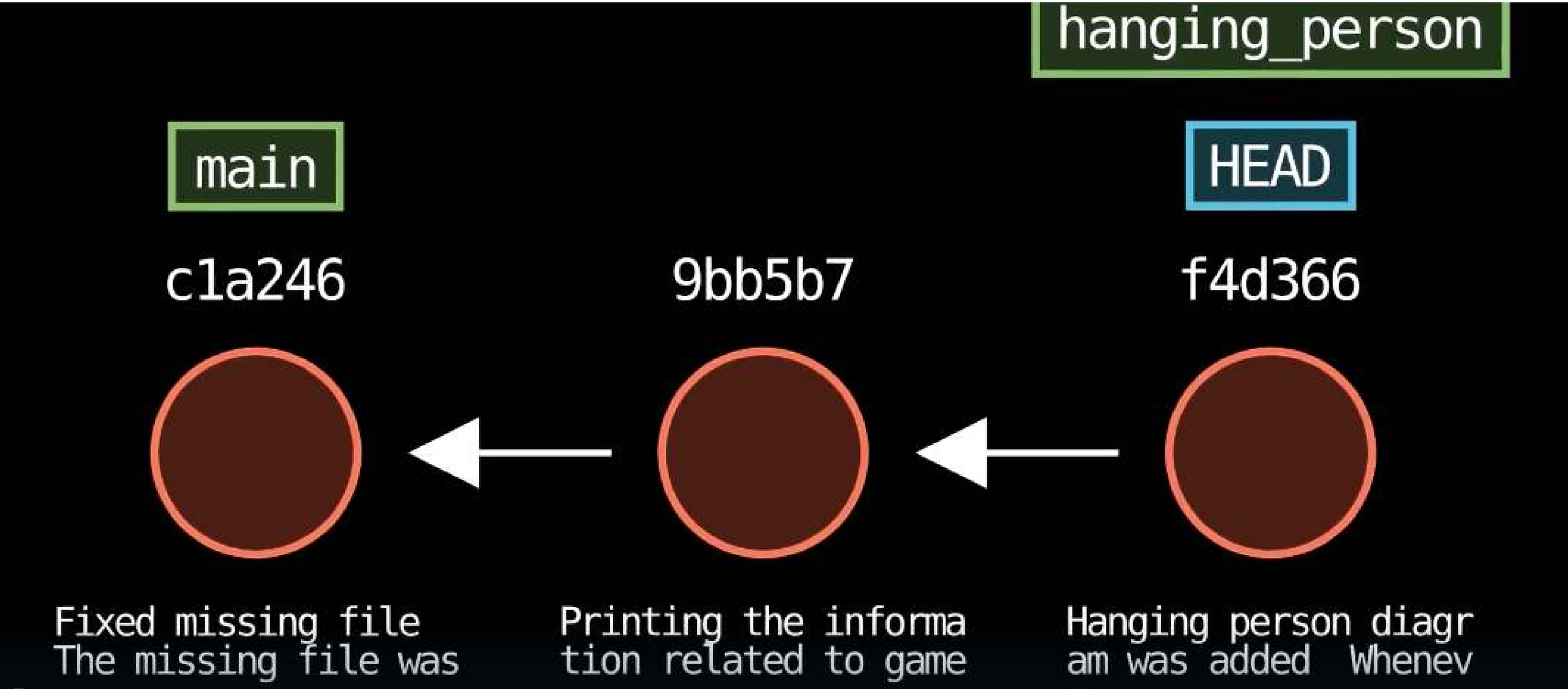
Auto - Merging

```
git switch hanging_person
```

```
git commit -m "Printing Information seperately"
```

```
git commit -m "Hanging Person Diagram was added"
```

Auto-merging



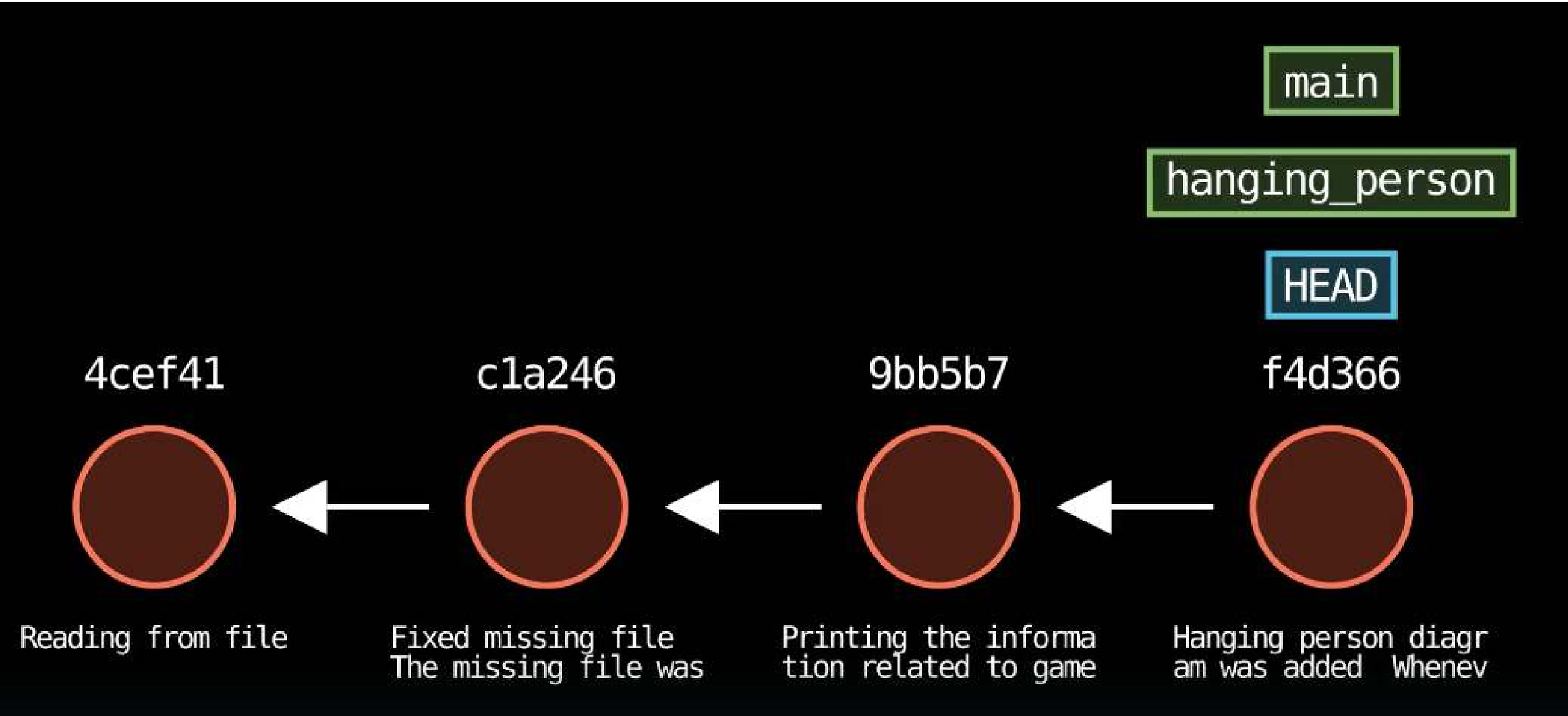
Auto - Merging

```
git switch main
```

```
git merge hanging_person
```

IT Club

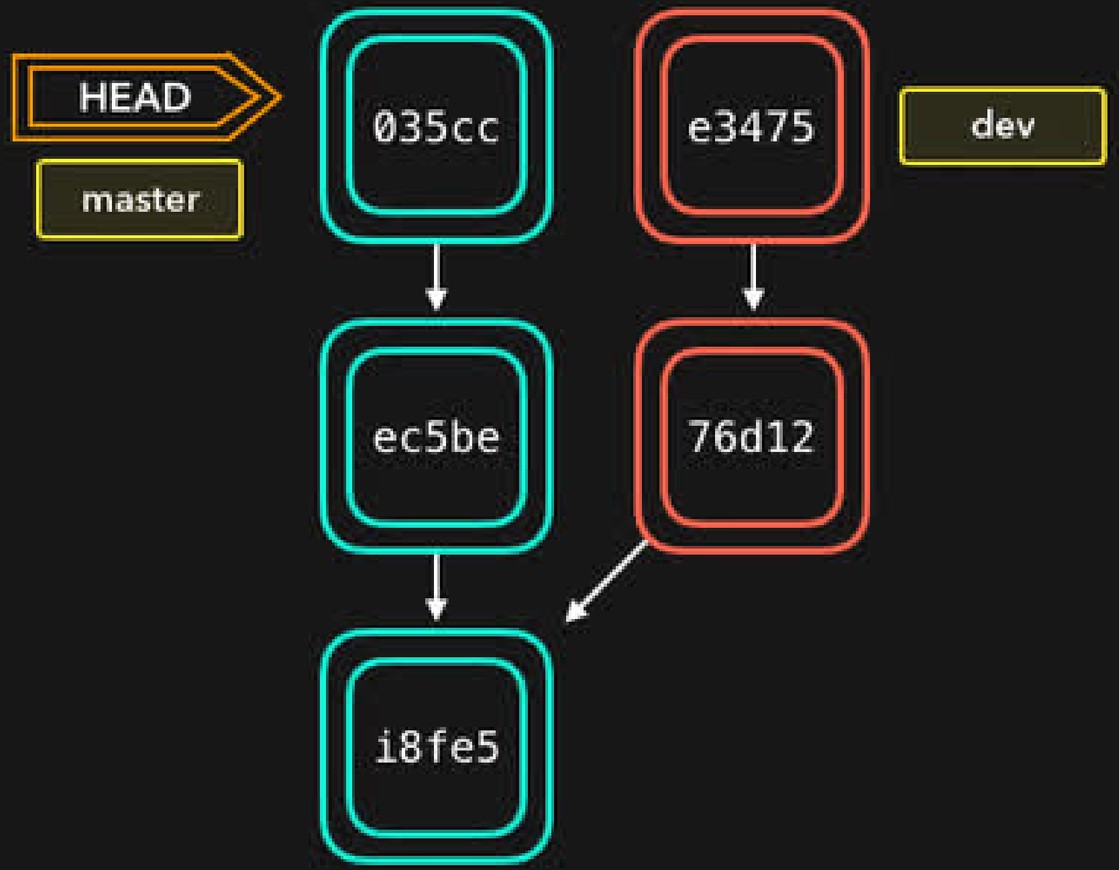
Auto - Merging

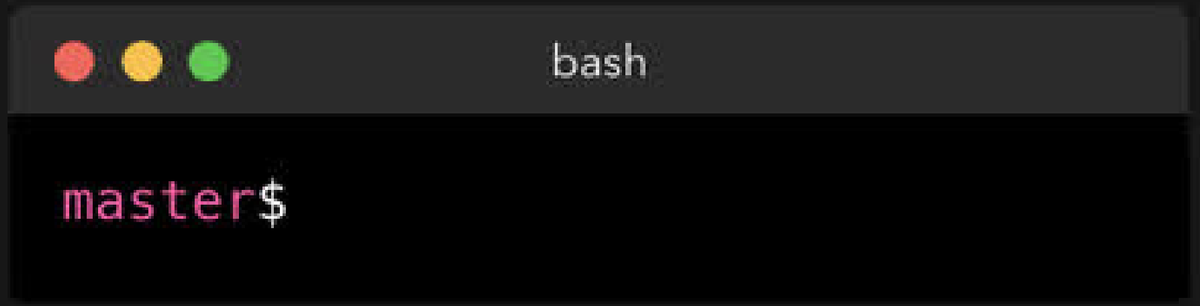


ORT - Merging

Merge-ort is a relatively new Git merge strategy that is significantly faster (for example, complex merge commits that previously took 5 or more seconds to create are now created in less than 200 milliseconds) and addresses subtle correctness issues found in the merge-recursive strategy.

ORT - Merging





Git | Merging (no-fast-forward)

Default behavior when current branch contains commits that the merging branch doesn't have

Creates a new commit which merges two branches together without modifying existing branches

ORT - Merging

```
git branch fix-garbage-print
```

```
git branch looking-for-compiler
```

```
git branch window-build_support
```

IT Club

ORT - Merging

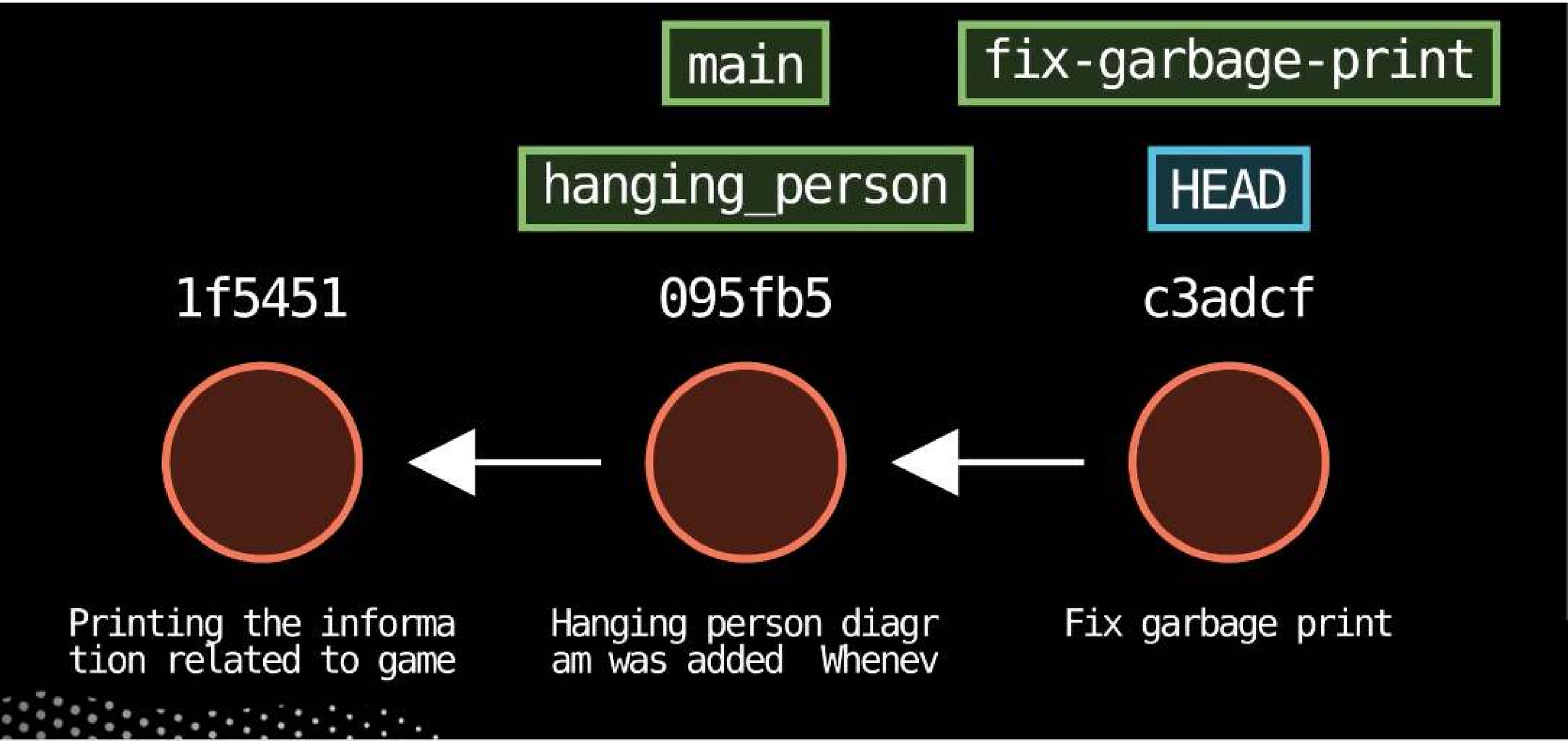
```
git switch fix-garbage-print
```

```
git commit -m "Fix garbage print"
```

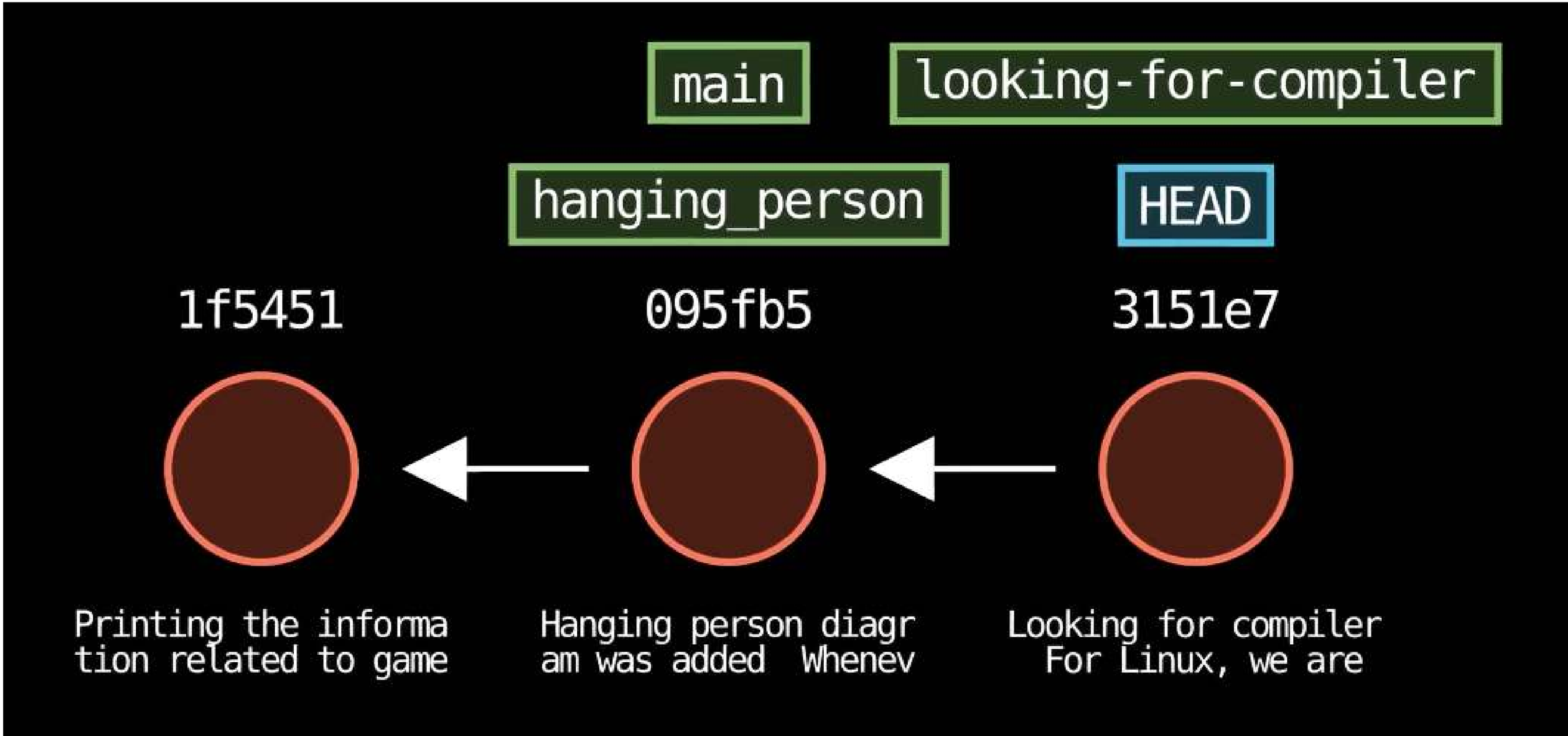
```
git switch looking-for-complier
```

```
git commit -m "Looking for complier"
```

ORT Merging


main
fix-garbage-print
hanging_person
HEAD
1f5451
095fb5
c3adcf
Printing the informa
tion related to game
Hanging person diagr
am was added Whenev
Fix garbage print

ORT Merging


main
looking-for-compiler
hanging_person
HEAD
1f5451
095fb5
3151e7
Printing the informa
tion related to game
Hanging person diagr
am was added Whenev
Looking for compiler
For Linux, we are

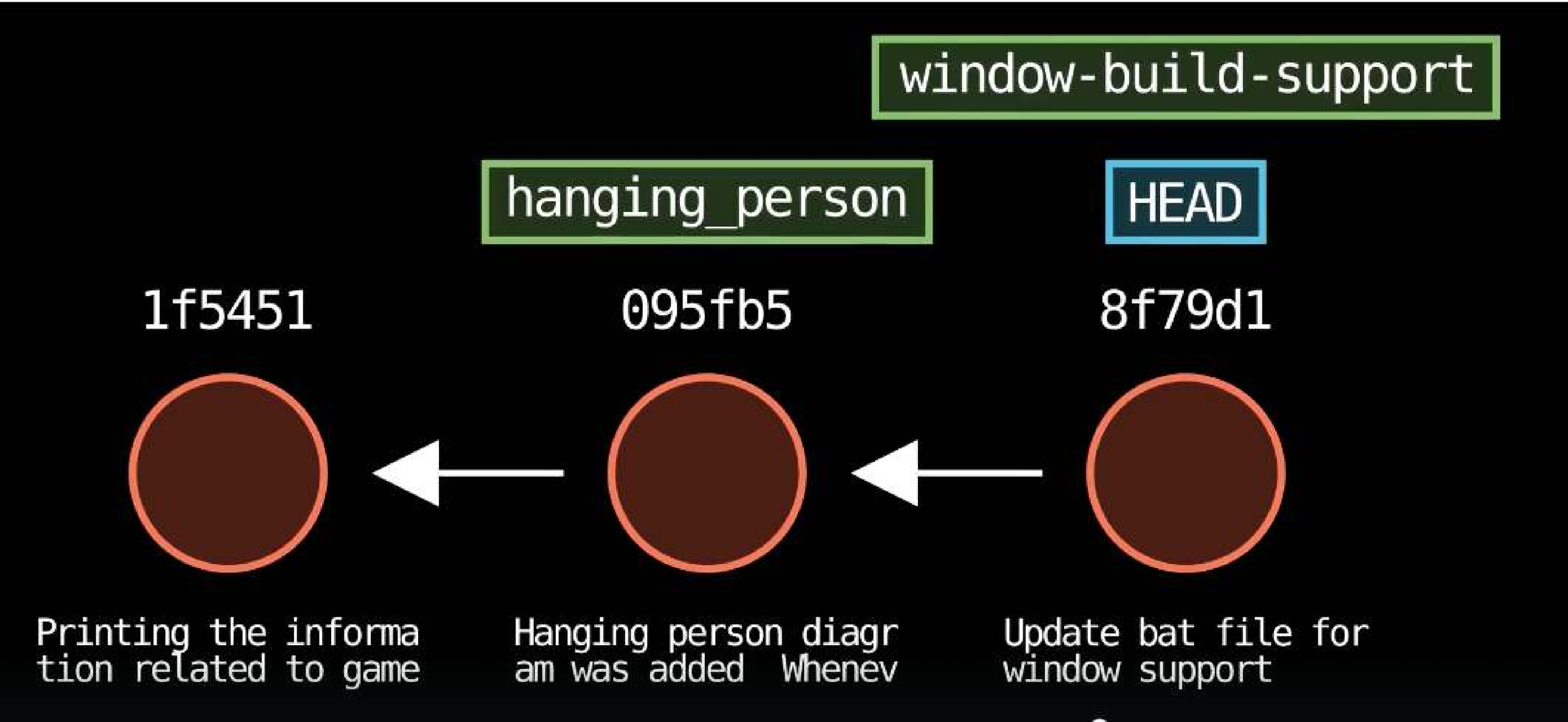
ORT Merging

```
git switch window-build_support
```

```
git commit -m "Window build support"
```

IT Club

ORT Merging


window-build-support
hanging_person
HEAD
1f5451
095fb5
8f79d1
Printing the informa
tion related to game
Hanging person diagr
am was added Whenev
Update bat file for
window support

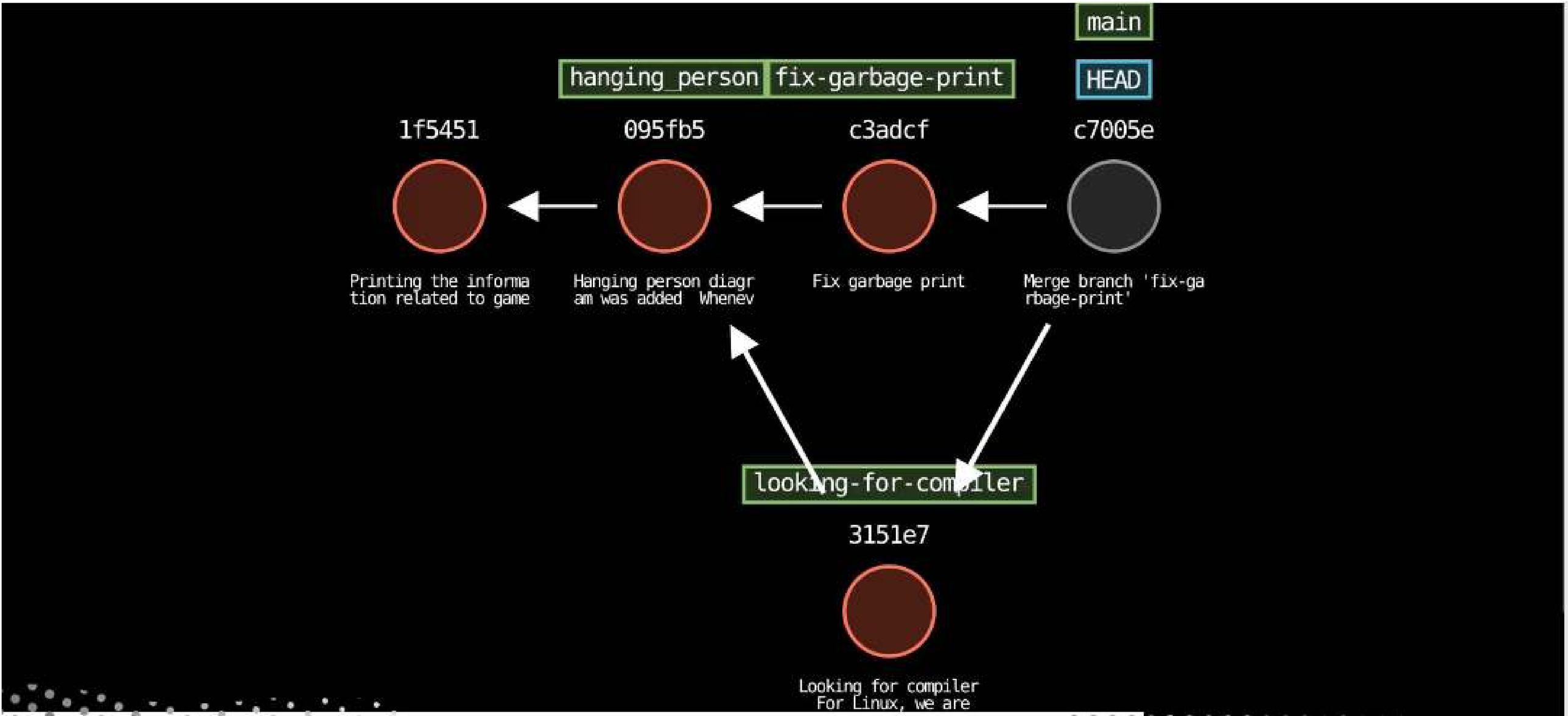
ORT - Merging

```
git switch main
```

```
git merge fix-garbage-print
```

```
git merge looking-for-complier
```

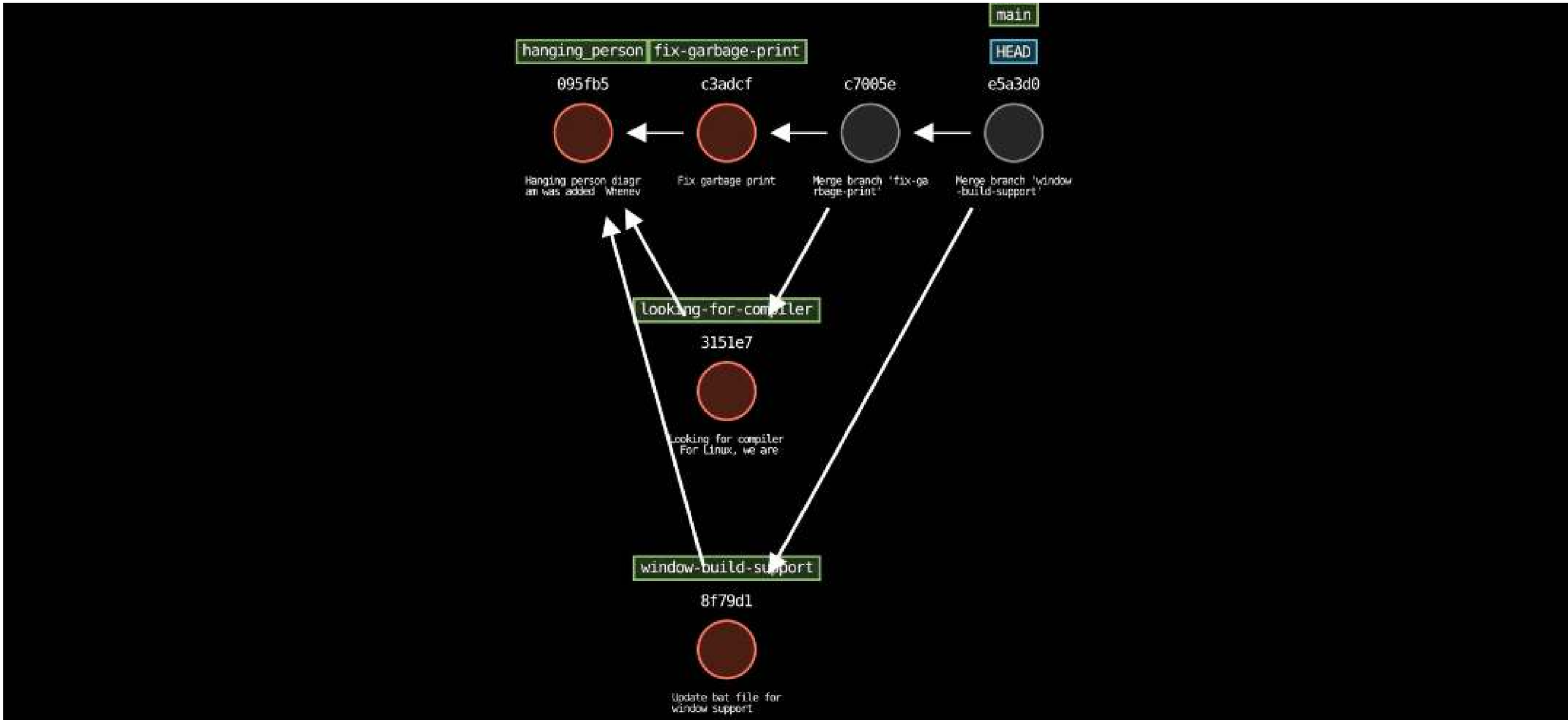
ORT Merging


main
hanging_person
fix-garbage-print
HEAD
1f5451
095fb5
c3adcf
c7005e
Printing the informa
tion related to game
Hanging person diagr
am was added Whenev
Fix garbage print
Merge branch 'fix-ga
rbage-print'
looking-for-compiler
3151e7
Looking for compiler
For Linux, we are

ORT - Merging

```
git merge window-build-support
```

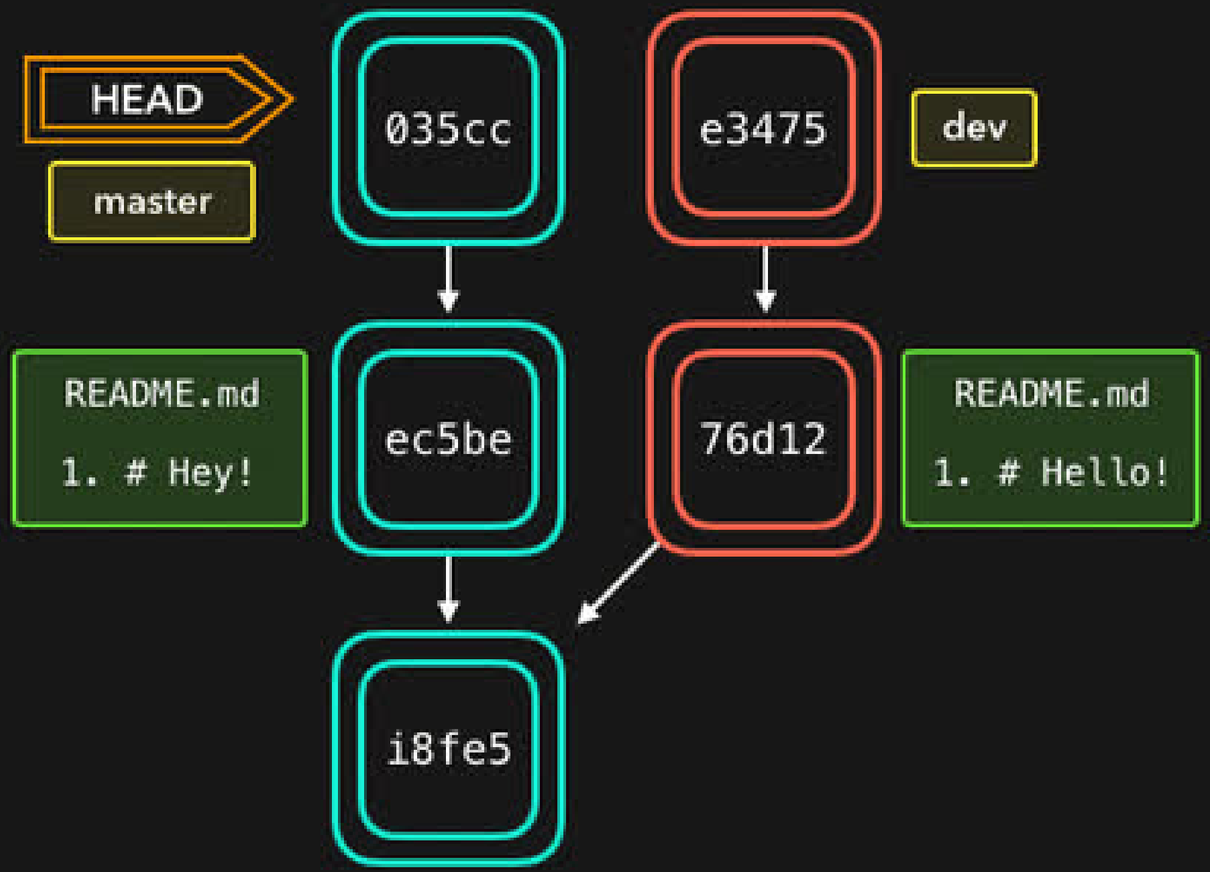
ORT Merging

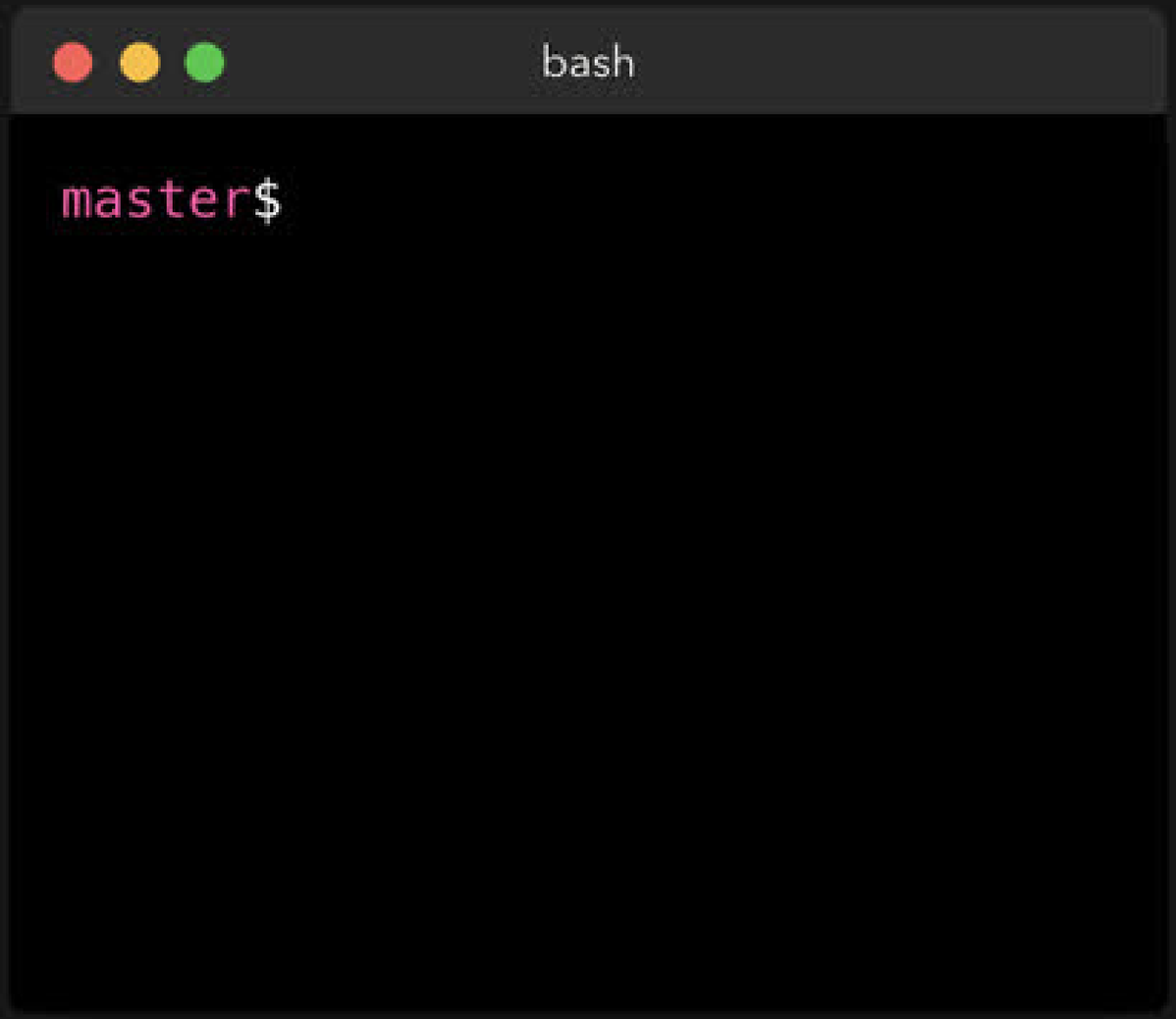

main
HEAD
hanging_person
fix-garbage-print
095fb5
c3adcf
c7005e
e5a3d0
Hanging person diagr
am was added Whenev
Fix garbage print
Merge branch 'fix-ga
rbage-print'
Merge branch 'window
-build-support'
looking-for-compiler
3151e7
Looking for compiler
For Linux, we are
window-build-support
8f79d1
Update bat file for
window support

Merge Conflict

A merge conflict in Git is like having two people try to make different changes to the same part of a document. Imagine you and your friend are editing a story together. You both decide to work on different sections. However, when you try to combine your changes, you realize that you both made different edits to the same paragraph.

Merge Conflict





Merge Conflict

Rebasing

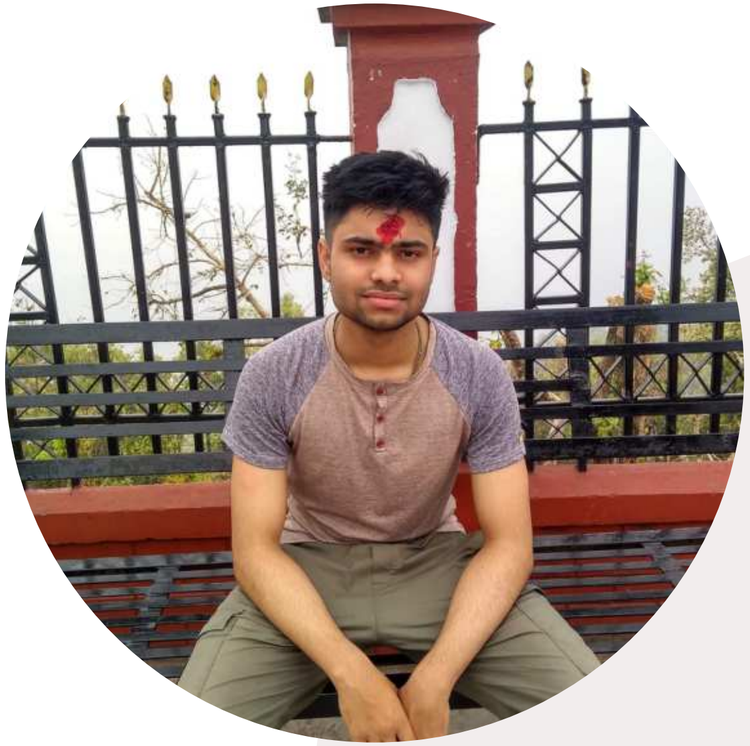
Rebasing in Git is like taking your current work and placing it on top of the latest updates. Imagine you're working on a project with others, and they make some changes while you're still working on your part. When you rebase, you bring in those latest changes, so your work is based on the most up-to-date version of the project.

```
git rebase main
```

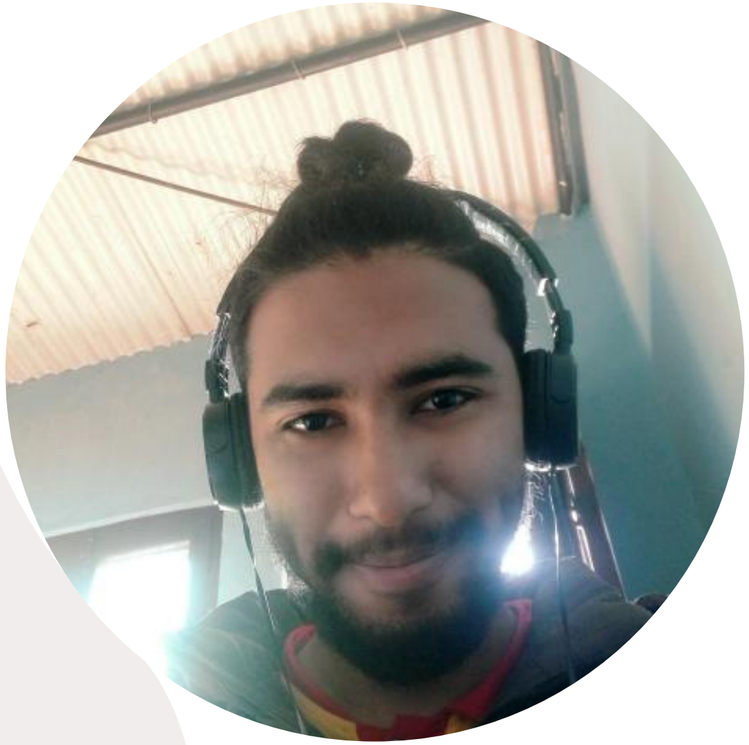

Are we clear?

Assignment time!

DEMO SOLUTION



Ujjwal Jha
077bct092
ujjwaljha1225@gmail.com
https://github.com/Ujj1225



Susheel Thapa
077bct090
077bct090.susheel@pcampus.edu.np
https://github.com/SusheelThapa

THANK YOU