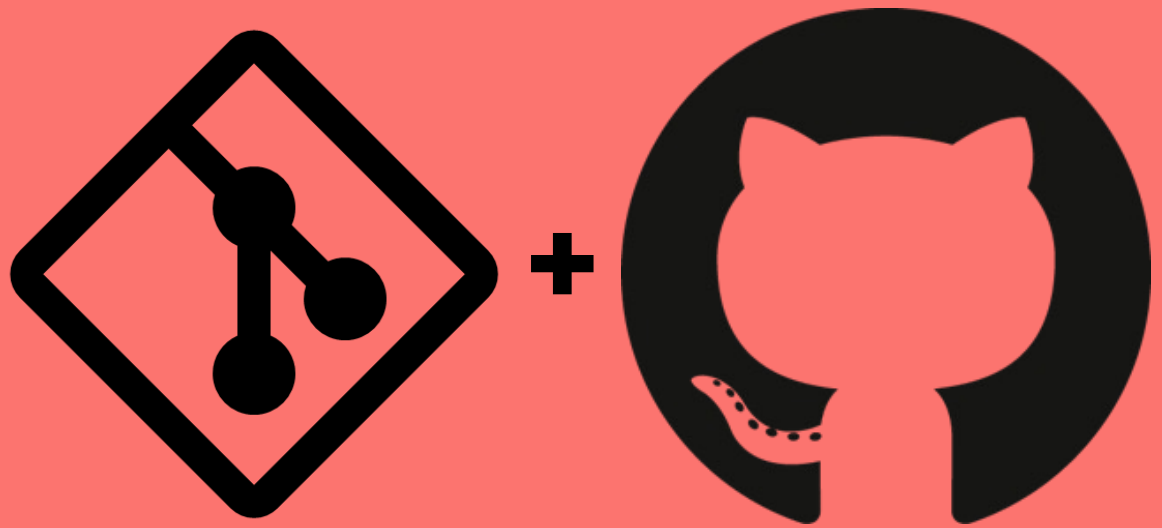
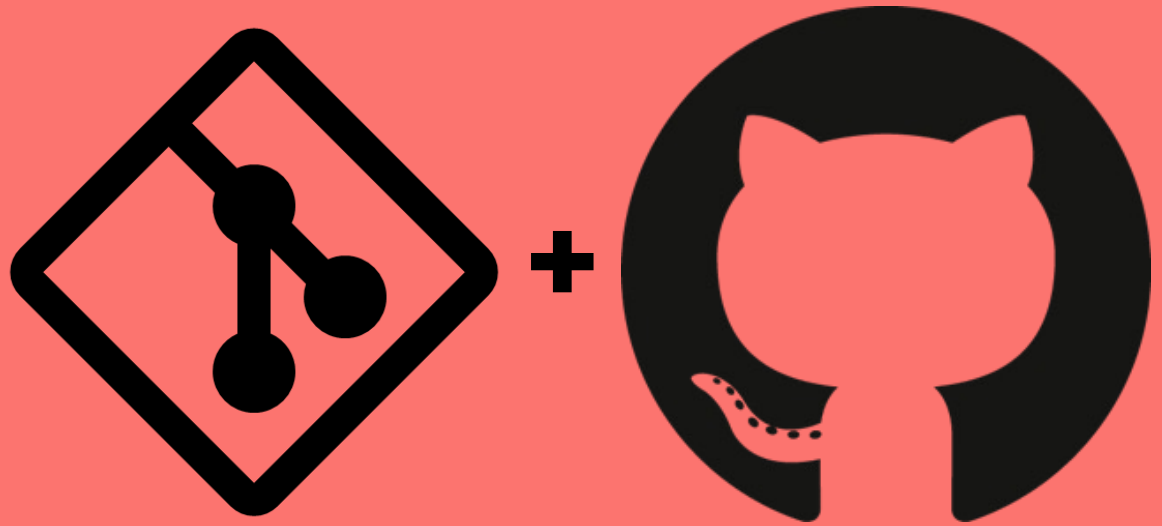
Git

In Practice



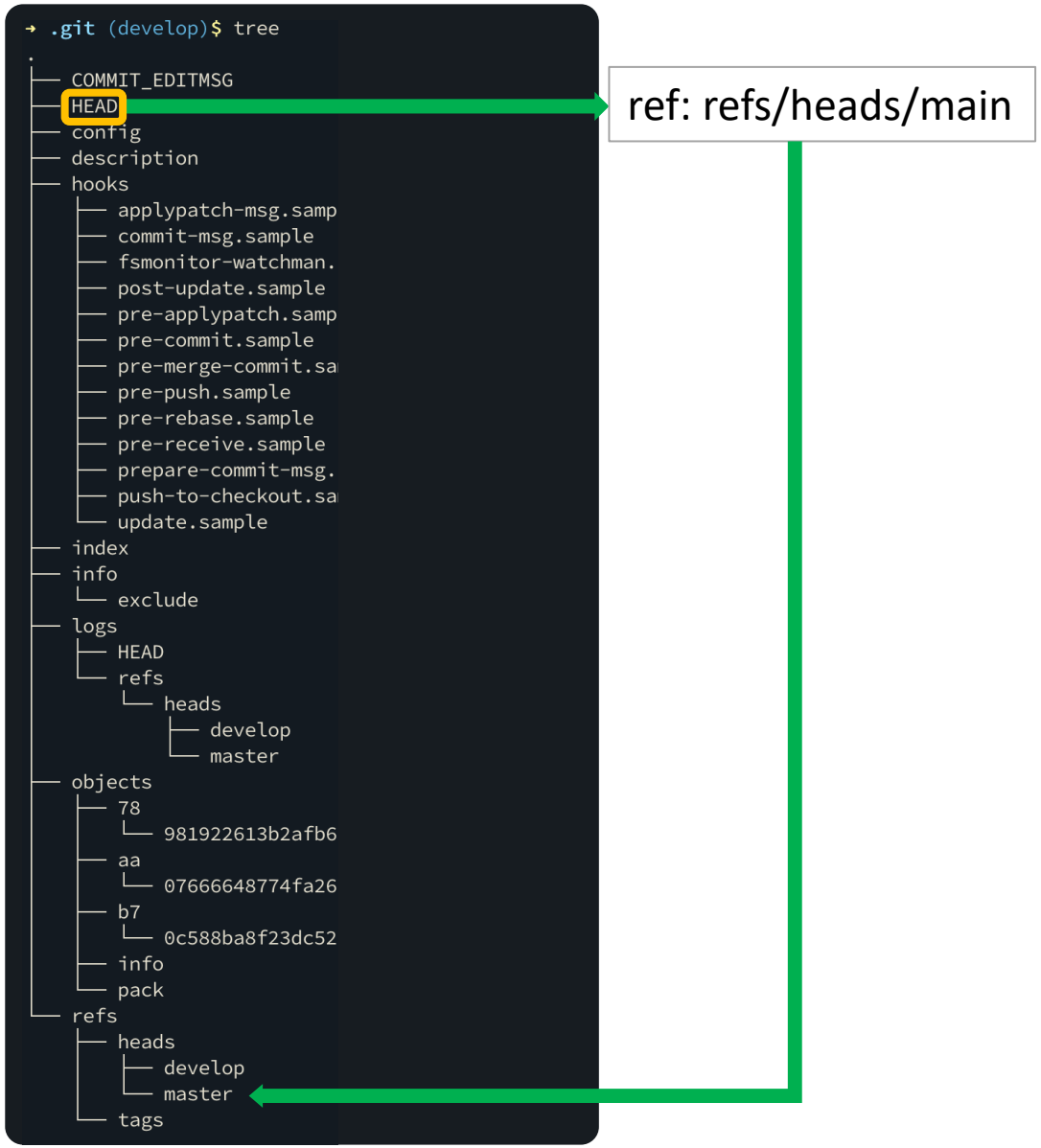
Git

In Practice



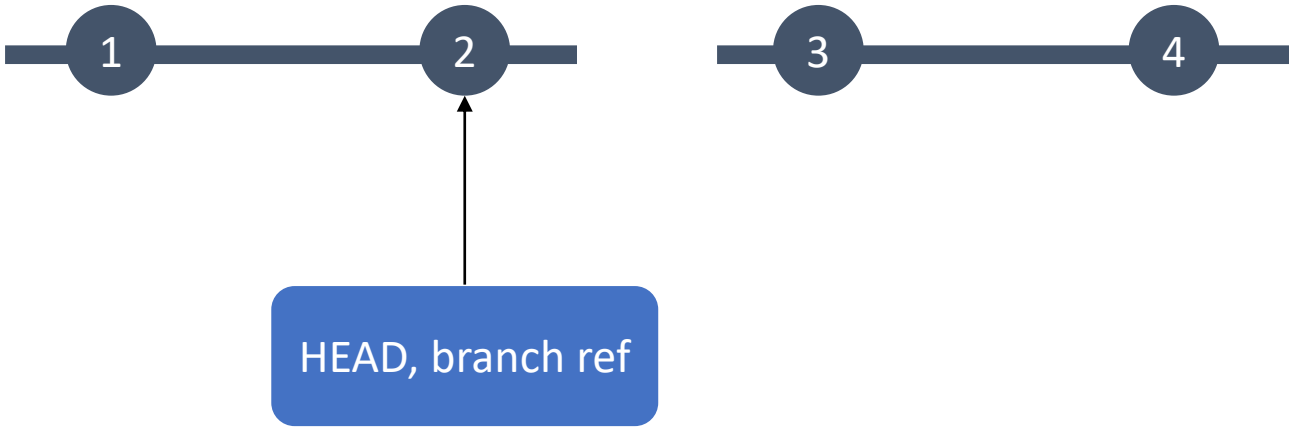
1. git reset
2. git revert
3. Comparisons
   git checkout vs reset vs revert

HEAD / branch refs

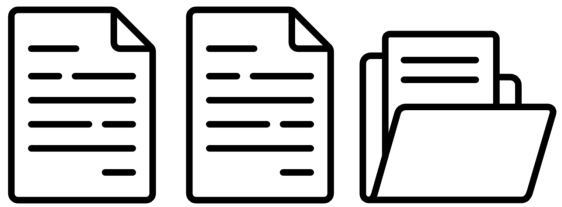


git reset


```
git reset 2 # commit ID 2
```



git reset – Types

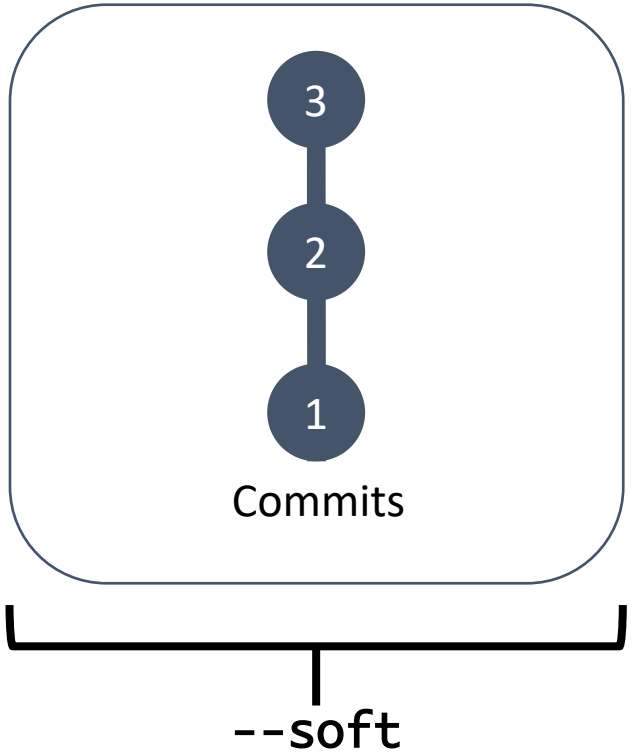


Working Tree

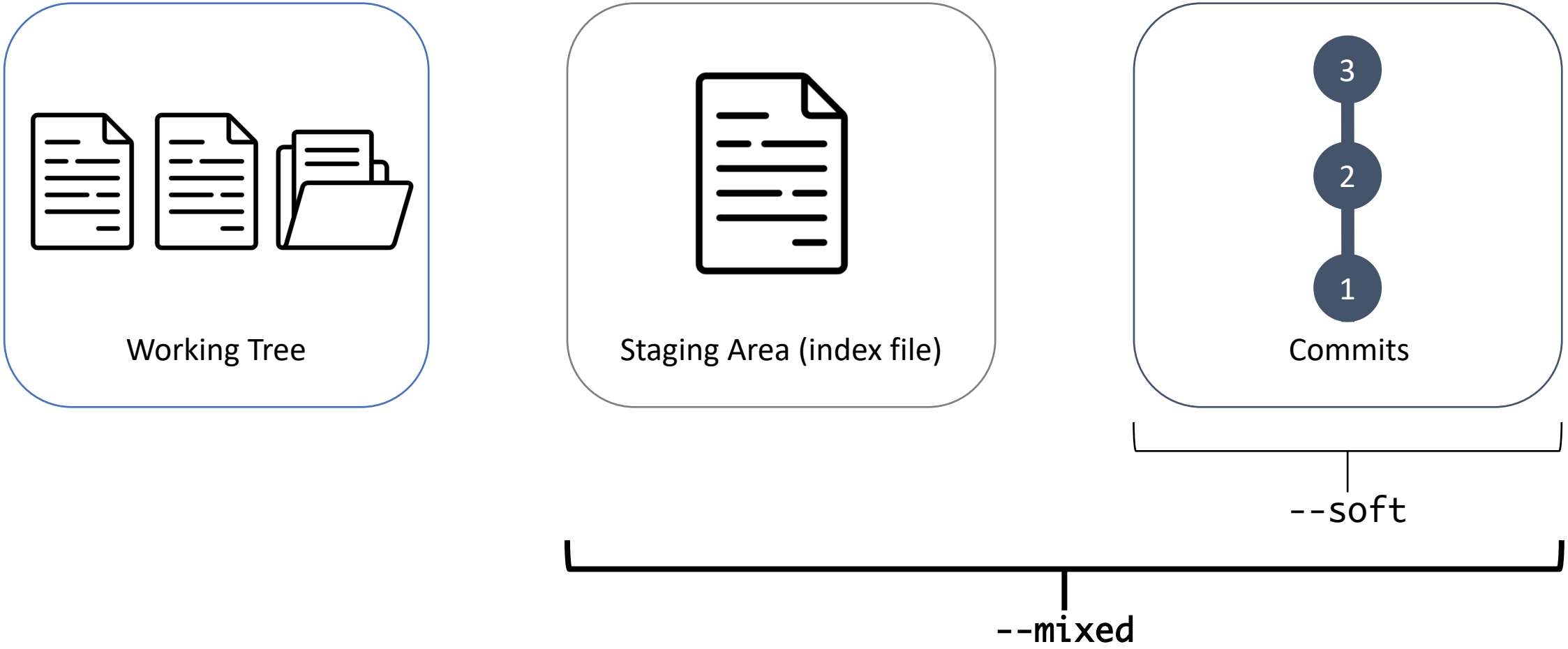
Staging Area (index file)

Commits

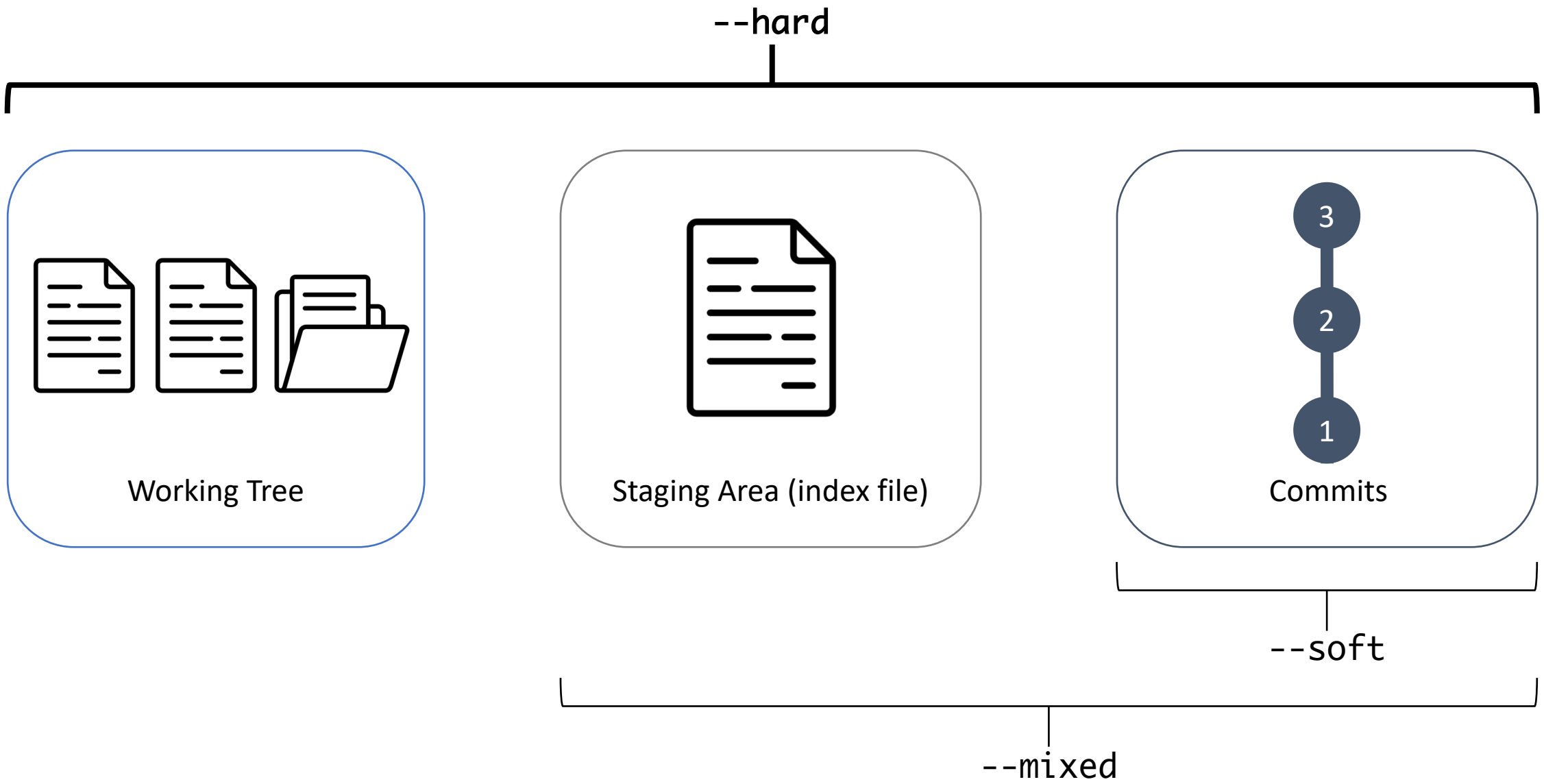
git reset – Types



git reset – Types



git reset – Types

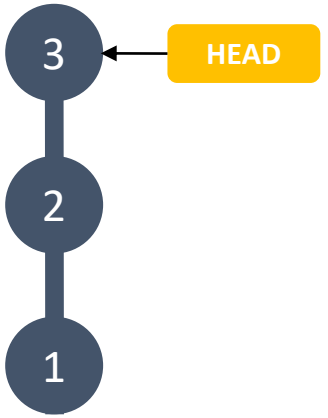


git reset – Types

--hard

Working Tree

Staging Area (index file)



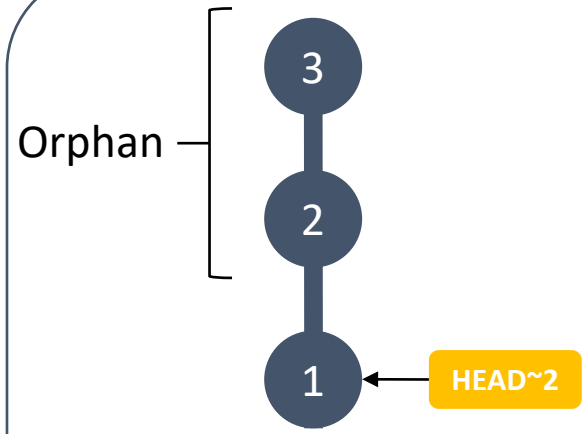
Commits

git reset – Types

--hard

Working Tree

Staging Area (index file)



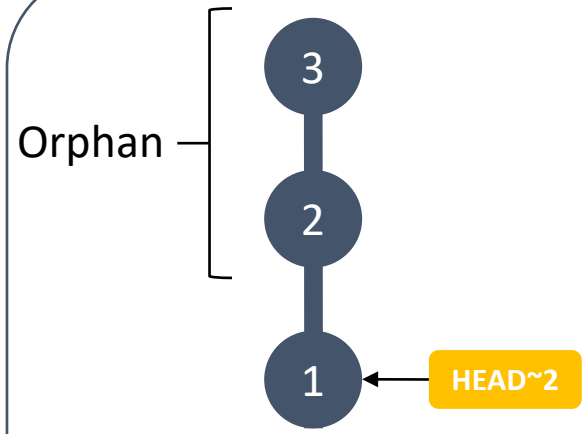
Commits

USE CASES for 3 types – TODO

--hard

Working Tree

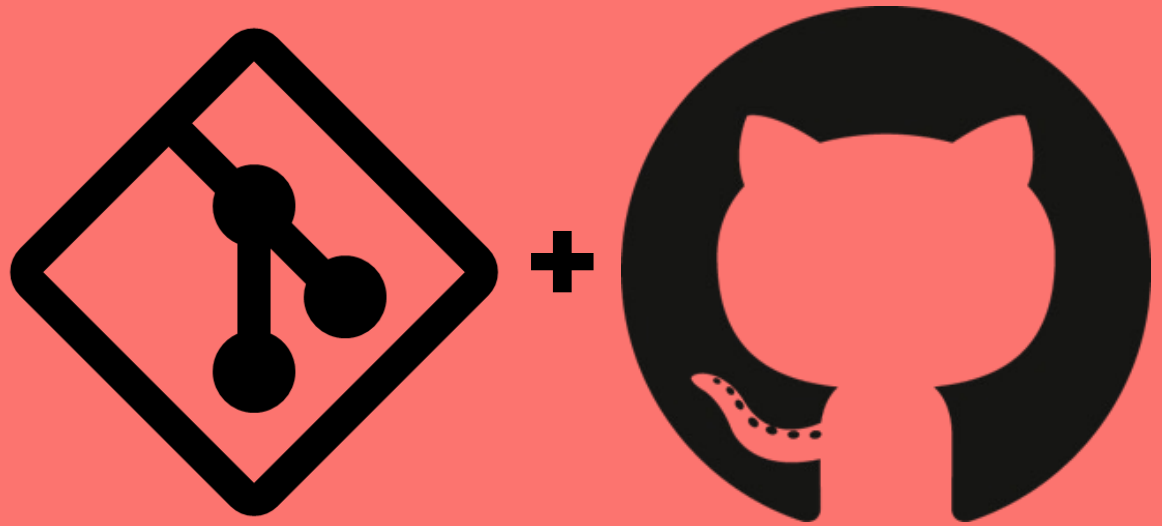
Staging Area (index file)



Commits

Git

In Practice



1. git reset
2. git revert
3. Comparisons
   git checkout vs reset vs revert

git revert

1. Revert is used to apply the inverse of a commit to the project history
2. The git revert command can be considered as an “undo” type command, however, it is not a traditional undo operation
3. Instead of removing the commit from the project history, revert inverts the changes introduced by a commit and appends a new commit for the resulting inversed content

git revert


```
git revert 2 # commit ID 2
```

Scenario 1 – Reverting a Fast-Forward commit

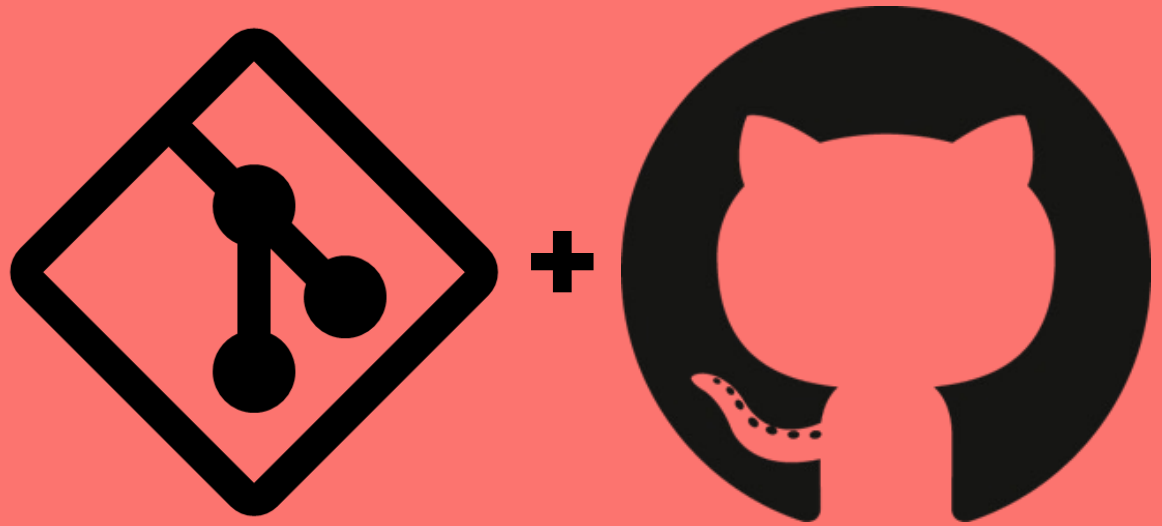
1. x

Scenario 2 – Reverting a Merge commit

1. x

Git

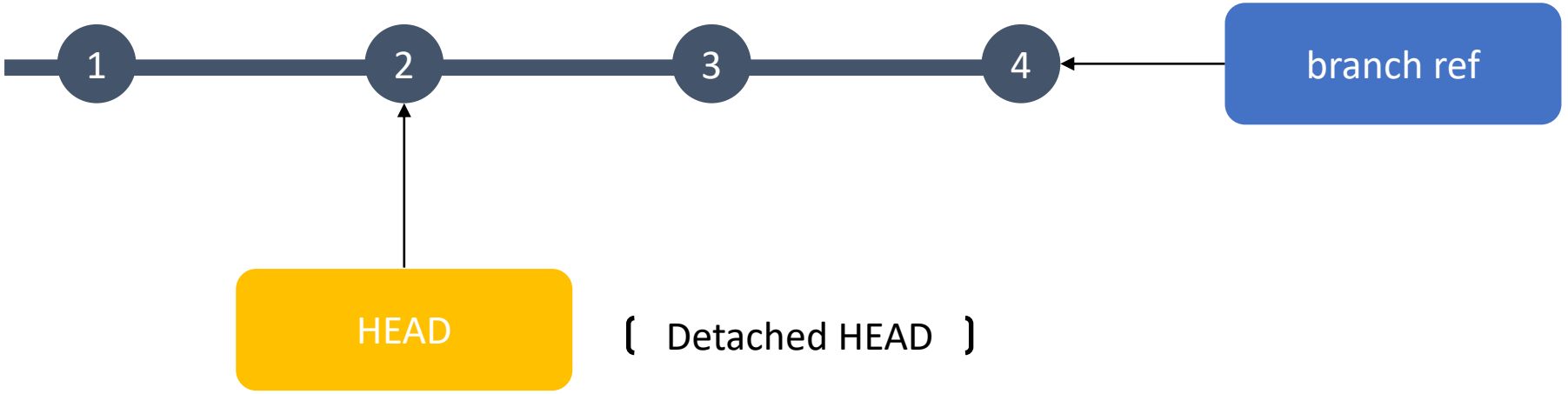
In Practice



1. git reset
2. git revert
3. Comparisons
   git checkout vs reset vs revert

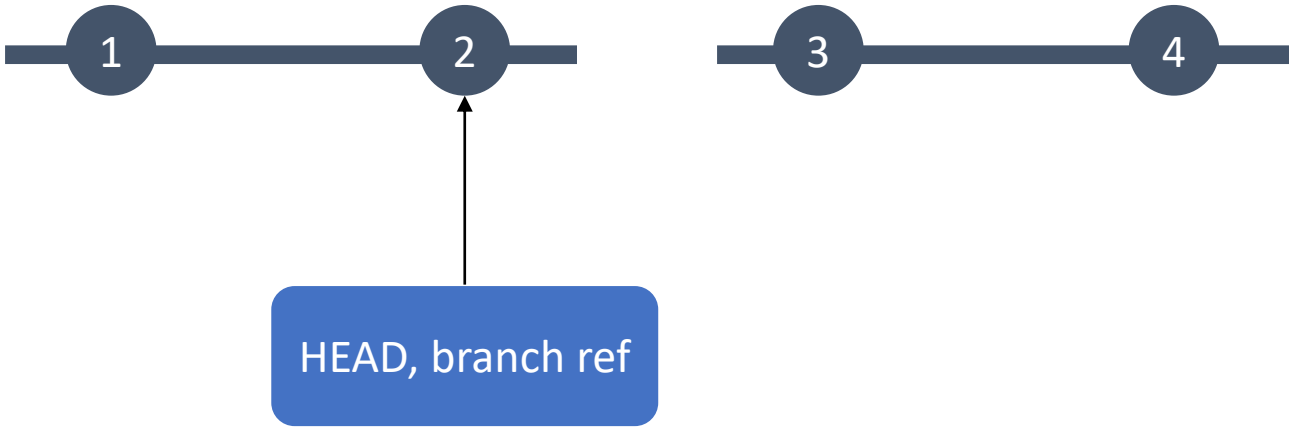
git checkout


```
git checkout 2 # commit ID 2
```



git reset


```
git reset 2 # commit ID 2
```



git revert


```
git revert 2 # commit ID 2
```

revert vs reset

revert	reset
Can revert a single branch	Can revert multiple branches
Will create a new commit	Will not