

Git Workshop IV

Git Under the Hood

TYH lab git workshop IV
by Paul Z Cheng

So many command so little time!!

- git mv
- git reflog
- git reref
- git clear
- git hlep
- git clone
- git merge
- git reset
- git blame
- git push
- git filter-branch
- git commit
- git add
- git bisect
- git grep
- git status
- git mv
- git fetch
- git diff
- git log
- git pull
- git revert
- git remote
- git init
- git branch
- git cherry-pick
- git config
- git checkout
- git reset
- git rm

Git "way of thinking"

Stretching Metaphor



Course Overview

- The Four Areas of Git
- Demonstrate basic Git commands relation to four area changes.
 - add, commit, checkout, rm --cached, git reset
- Play with history

The Four Areas of GIT

Stash

Working Area

Index

Repository

Two Questions to ask before using a git command

- How does this command move information between the four areas?
- How does this command change the repository?






The Three main area

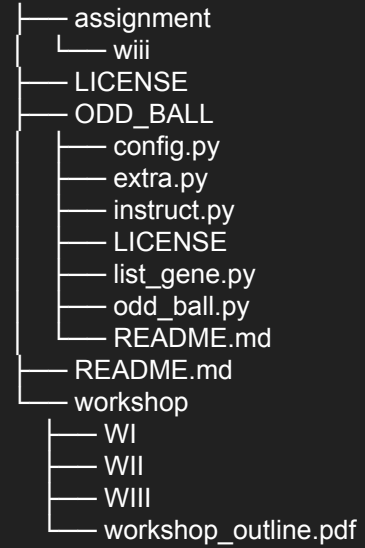
Working Area

Index

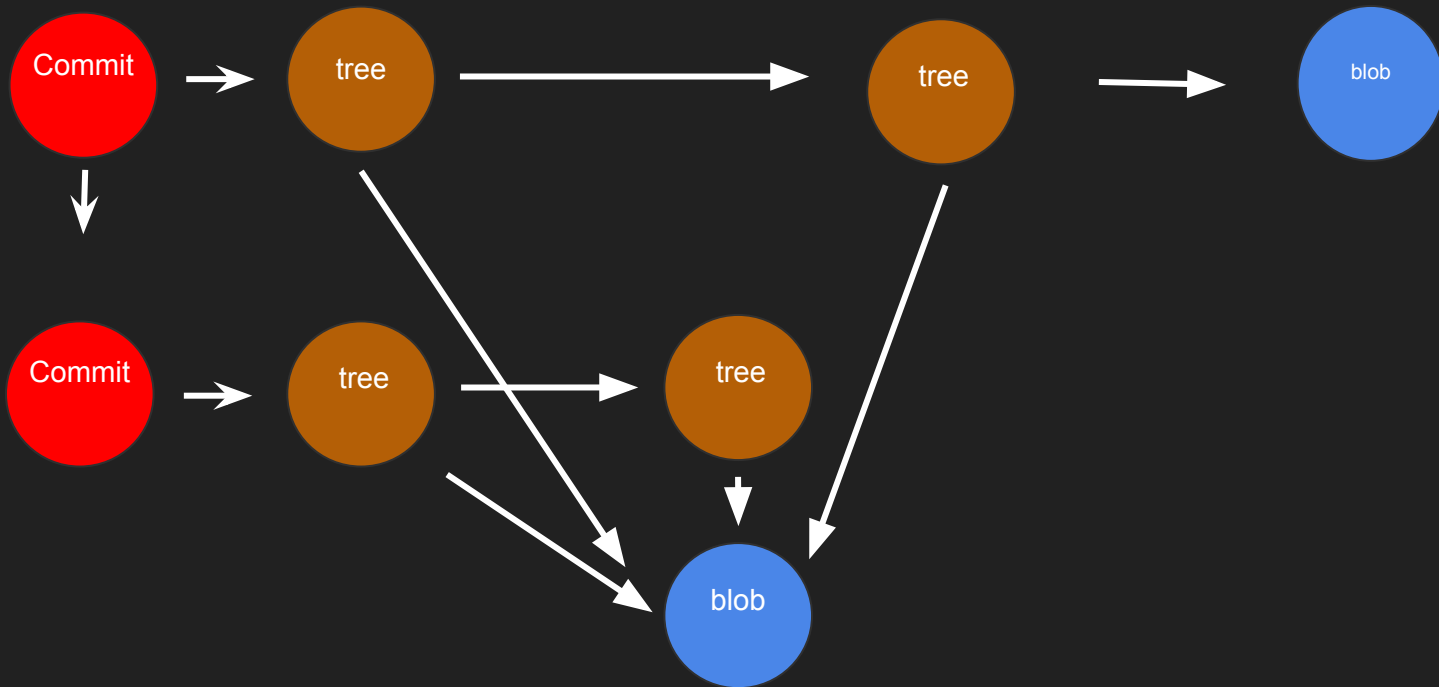
Repository

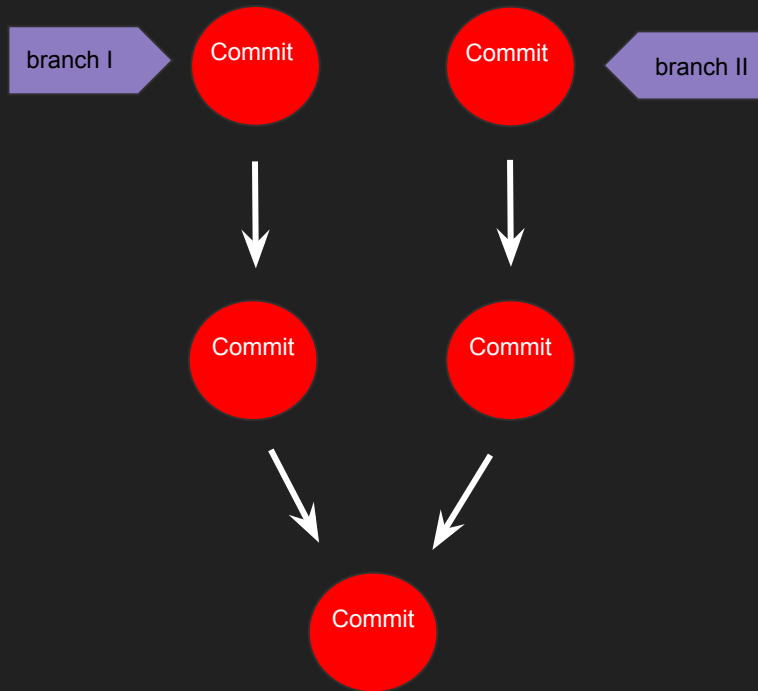
Working Area

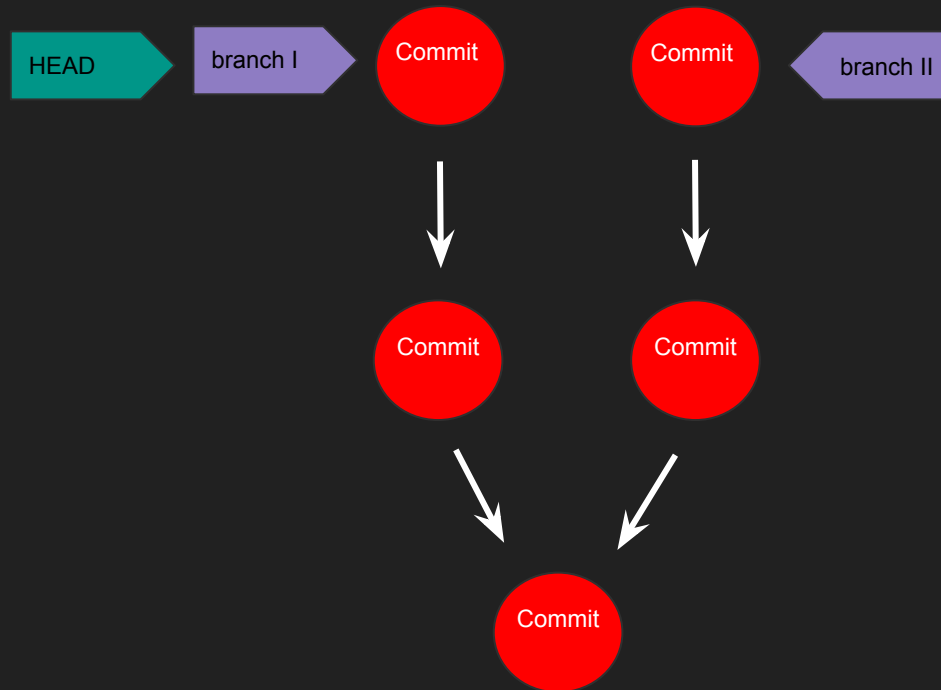
- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

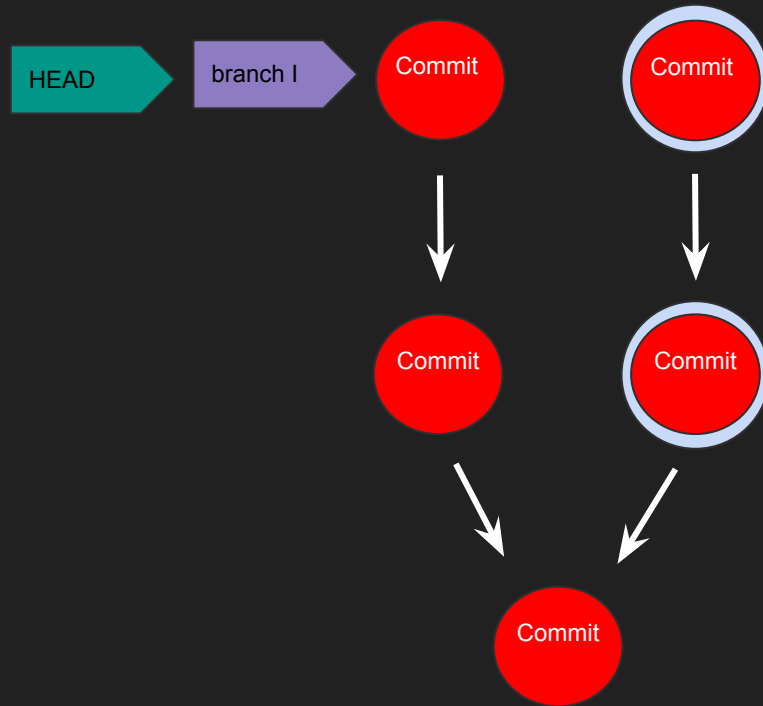


Repository



















Index

- Index == staging area
- Essentially a jumpy board
- holds all committed and add information.

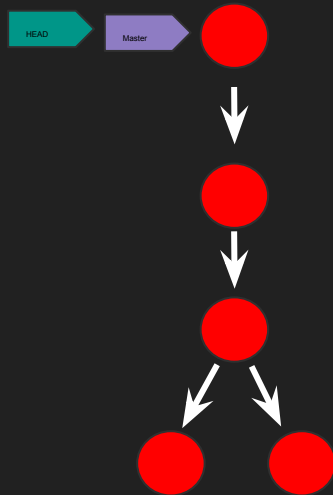
Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Repository



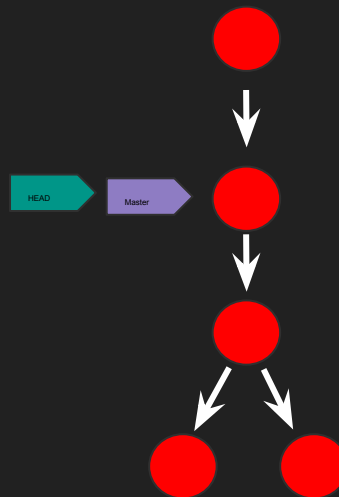
Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Repository



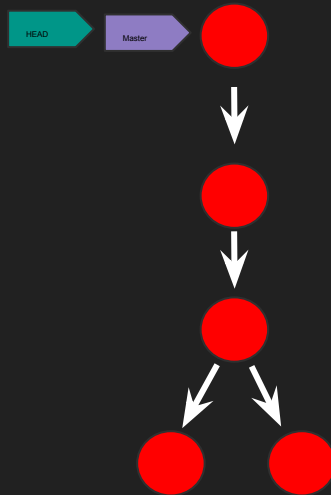
Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Repository








Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

With this in mind






- Understand how each basic commands works
 - status
 - add
 - diff
 - git commit
 - git checkout
 - git rm








On branch master
nothing to commit, working tree clean

Edit something in README.md






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

On branch master

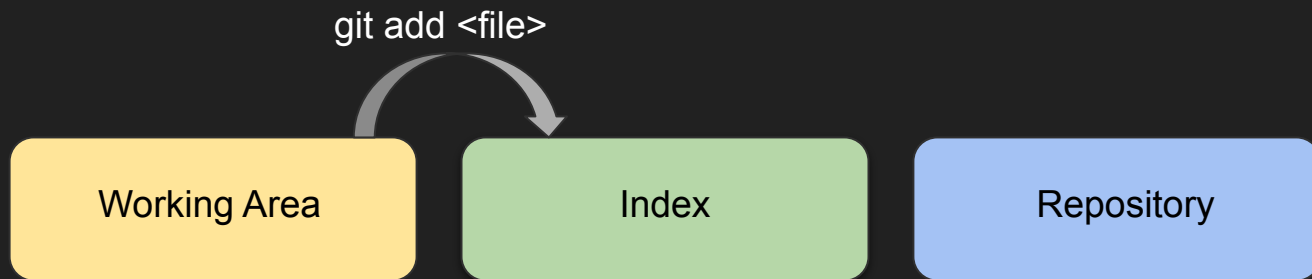
Changes not staged for commit:

(use "git add <file>..." to update what will be committed)






(use "git restore <file>..." to discard changes in working directory)

modified: README.md






no changes added to commit (use "git add" and/or "git commit -a")








Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

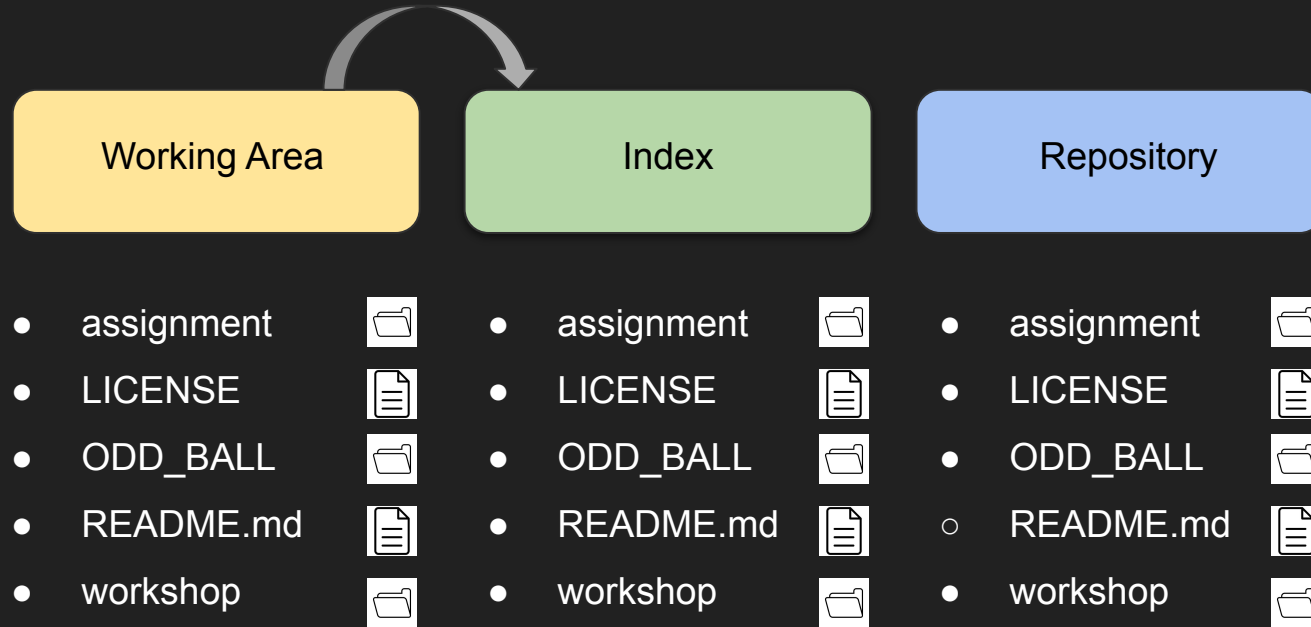
Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git add README.md



On branch master

Changes to be committed:






(use "git restore --staged <file>..." to unstage)

modified: README.md






git commit








Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

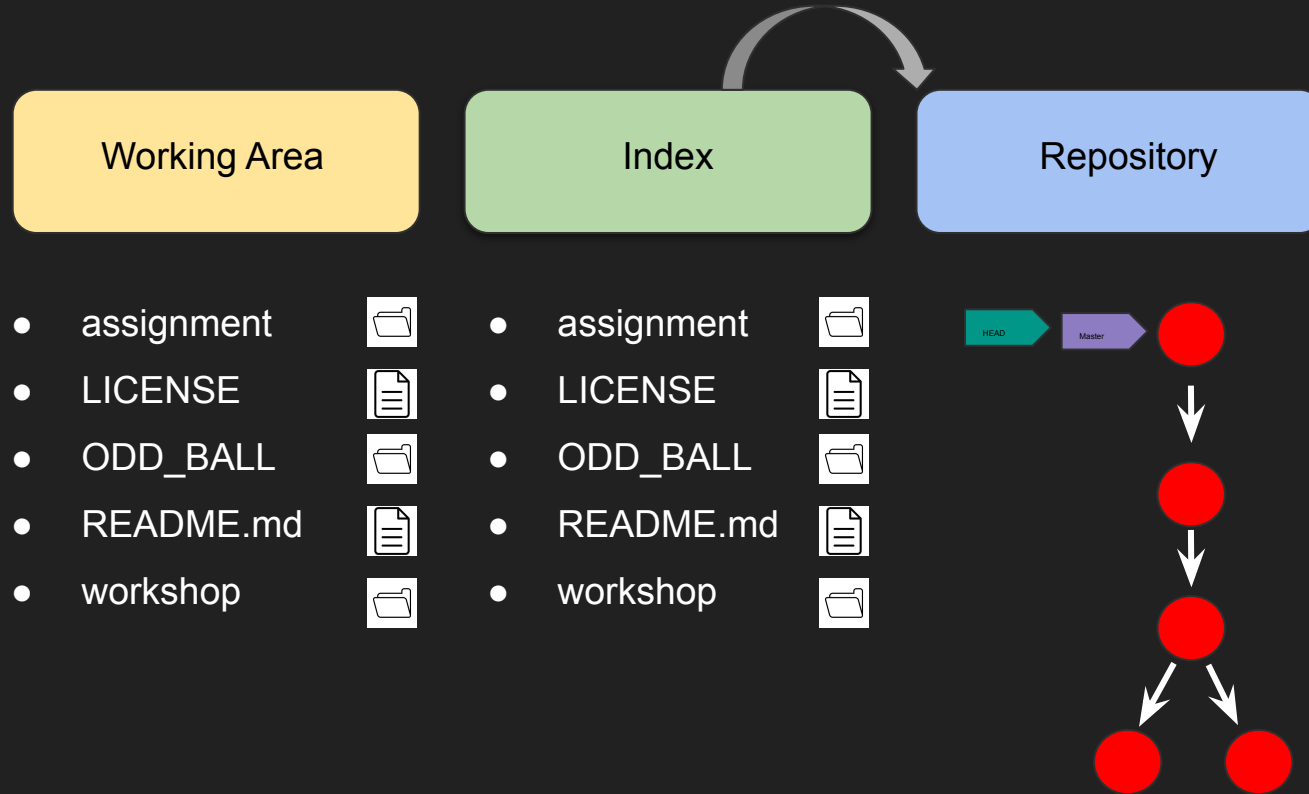
Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

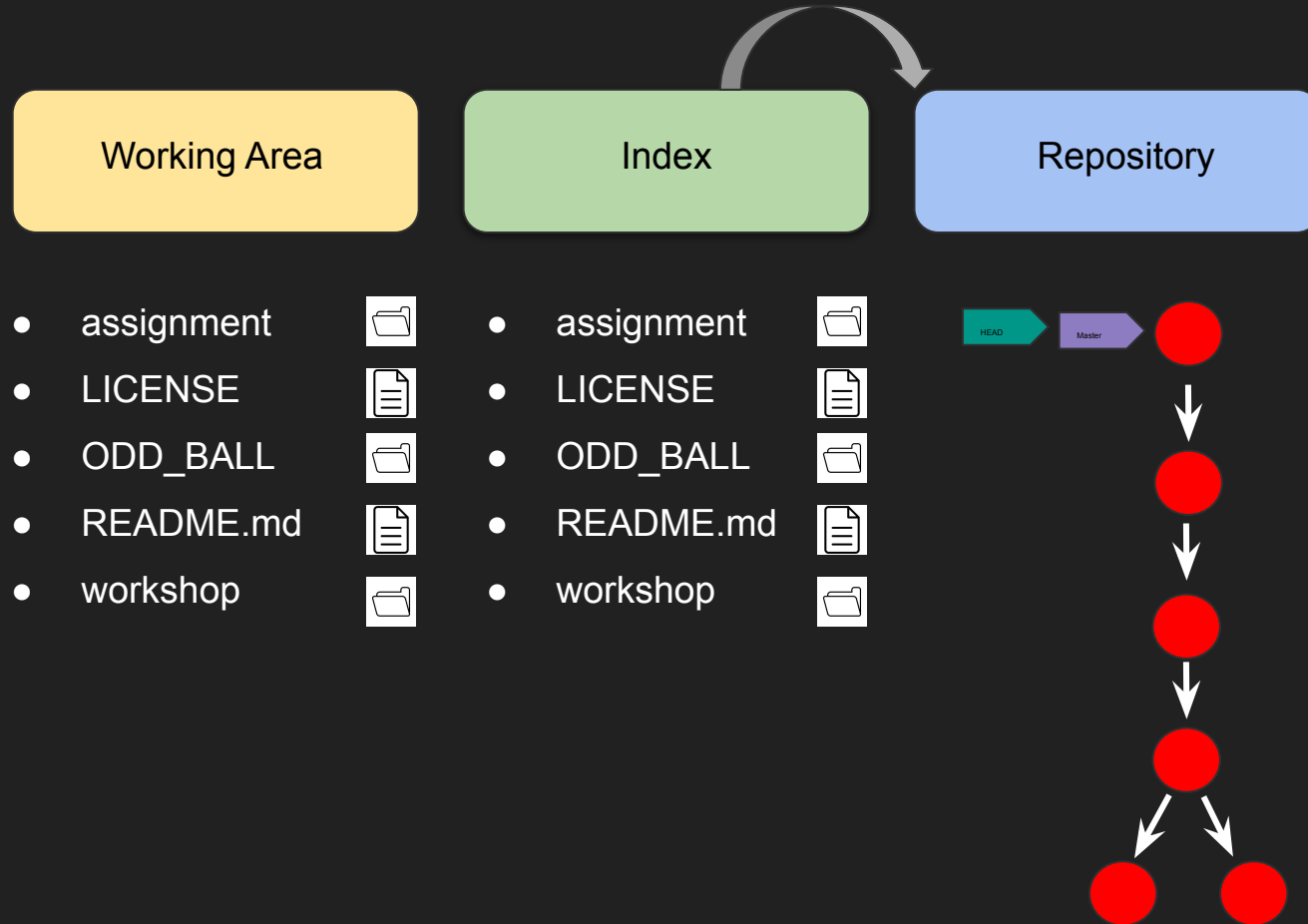
Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

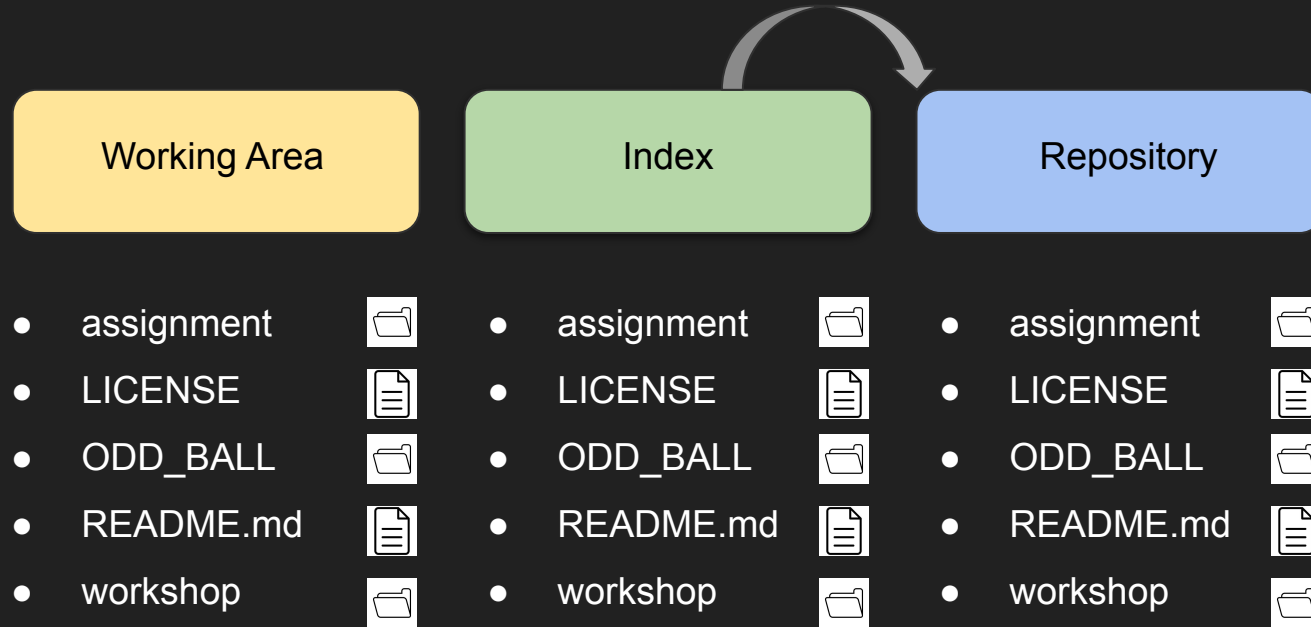
git commit -m "Edited README.md file"








git commit -m "Edited README.md file"








git commit -m "Edited README.md file"








Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

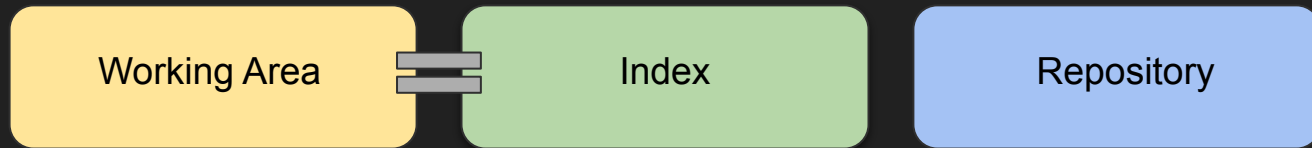
Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

```
On branch master
nothing to commit, working tree clean
```

What does git diff do?

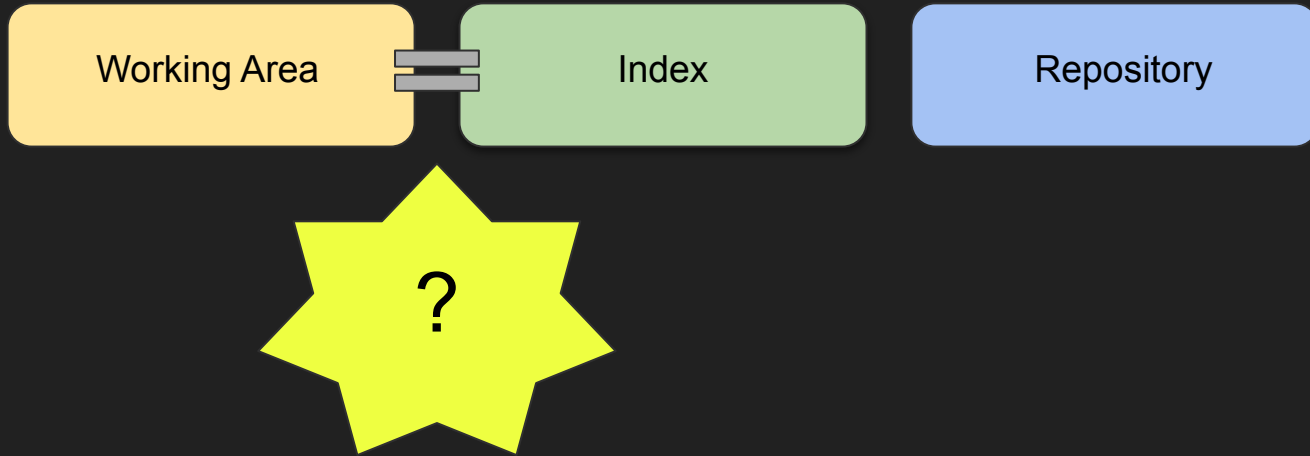
git diff



git diff --cached



Demo: git diff



edit some thing in Oddball.py

- change eeg from True to false
- Capitalized some comments of python functions

Demo: git diff



```
diff --git a/ODD_BALL/odd_ball.py b/ODD_BALL/odd_ball.py
index 0c84488..324e7c5 100644
--- a/ODD_BALL/odd_ball.py
+++ b/ODD_BALL/odd_ball.py
@@ -19,7 +19,7 @@ If Set true it will run it, this helps with debug.
     instruction = True
     demo_gui = True
     beh = True
-    eeg = True
+    eeg = False
     debug = False
```

```

diff --git a/ODD_BALL/odd_ball.py b/ODD_BALL/odd_ball.py
index 0c84488..324e7c5 100644
--- a/ODD_BALL/odd_ball.py
+++ b/ODD_BALL/odd_ball.py
@@ -19,7 +19,7 @@ If Set true it will run it, this helps with debug.
     instruction = True
     demo_gui = True
     beh = True
-    eeg = True
+    eeg = False
     debug = False

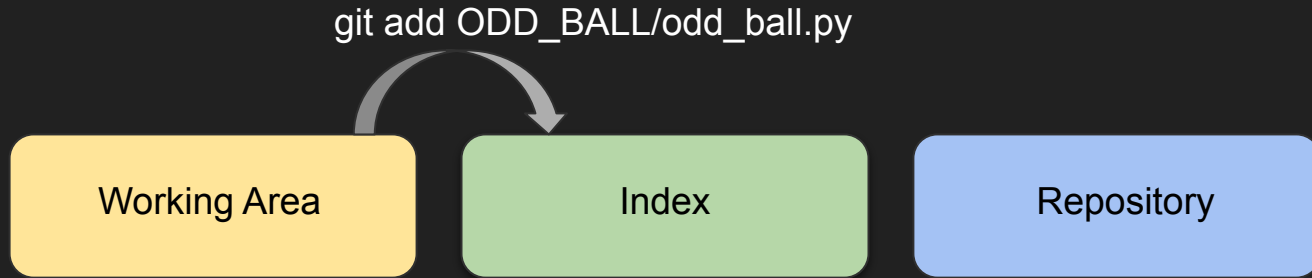
@@ -122,7 +122,7 @@ def pp_reset(eeg):
    pp.setData(0)

def keypress(config):
-    """Keypress for the experiment
+    """Keypress For The Experiment
    Accepts only responds and quit keys set in the configuration file
    and if quit key is accepted, close out the window.
    input:
@@ -149,7 +149,7 @@ def keypress(config):
    return key

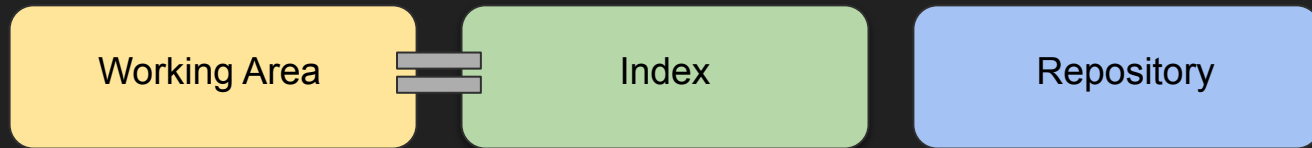
def prompt(text = "", keyList=None):
-    """Present instruction prompt.
+    """Present Instruction Prompt.
    Shows instruction and returns answer (keypress) and reaction time.
    Defaults is no text and accept all keys.
    input:

```

add the changes



git diff

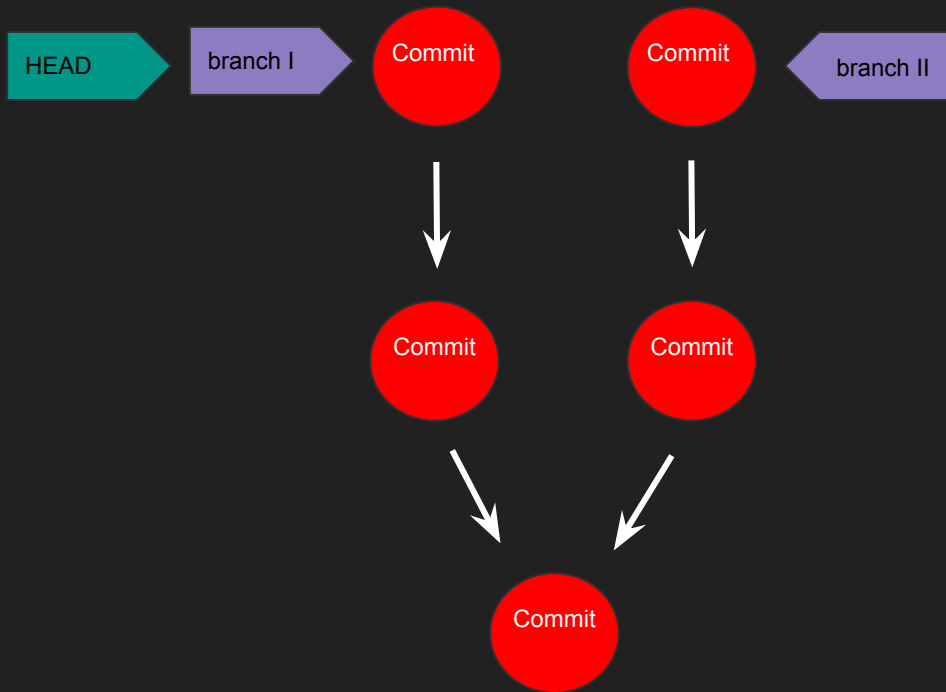


git diff --cached

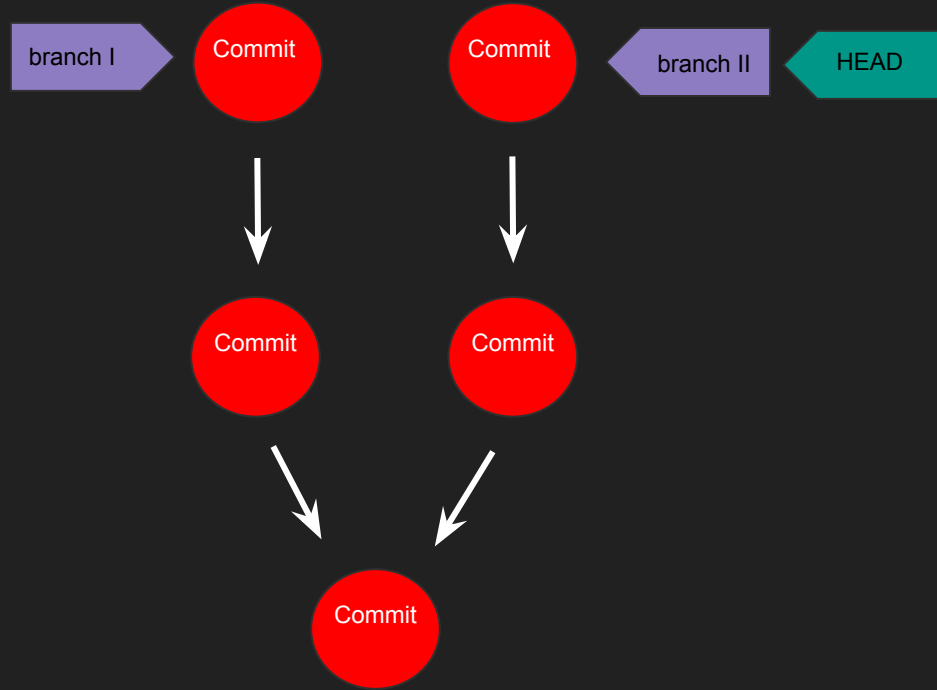


```
diff --git a/ODD_BALL/odd_ball.py b/ODD_BALL/odd_ball.py
index 0c84488..324e7c5 100644
--- a/ODD_BALL/odd_ball.py
+++ b/ODD_BALL/odd_ball.py
@@ -19,7 +19,7 @@ If Set true it will run it, this helps with debug.
     instruction = True
     demo_gui = True
     beh = True
-    eeg = True
+    eeg = False
     debug = False
```

git checkout







git checkout branch II







branch I





Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 

Index






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 

Repository






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 

branch II






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

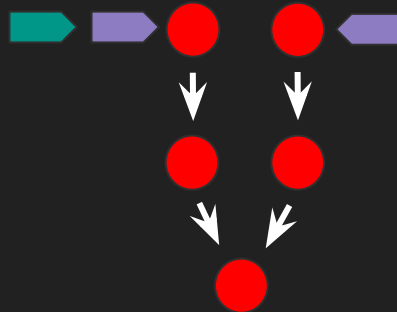
Repository

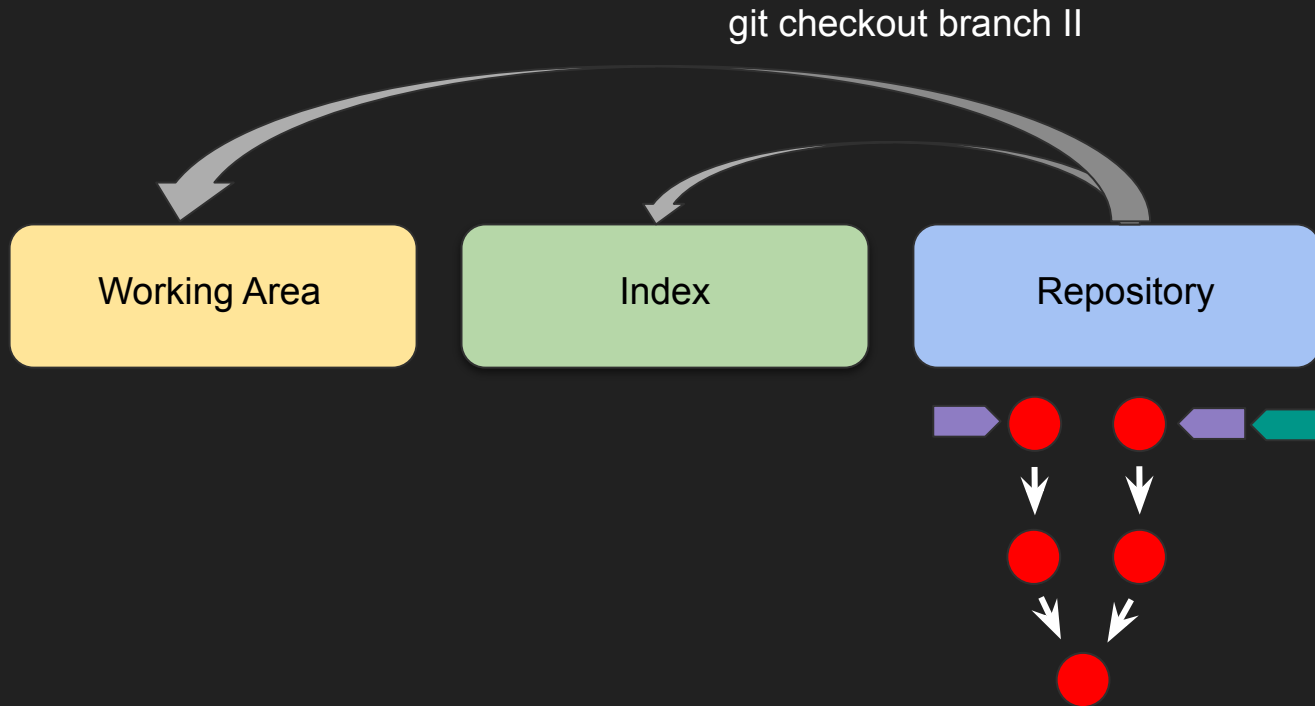
- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Working Area

Index





Repository









git checkout branch II






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 

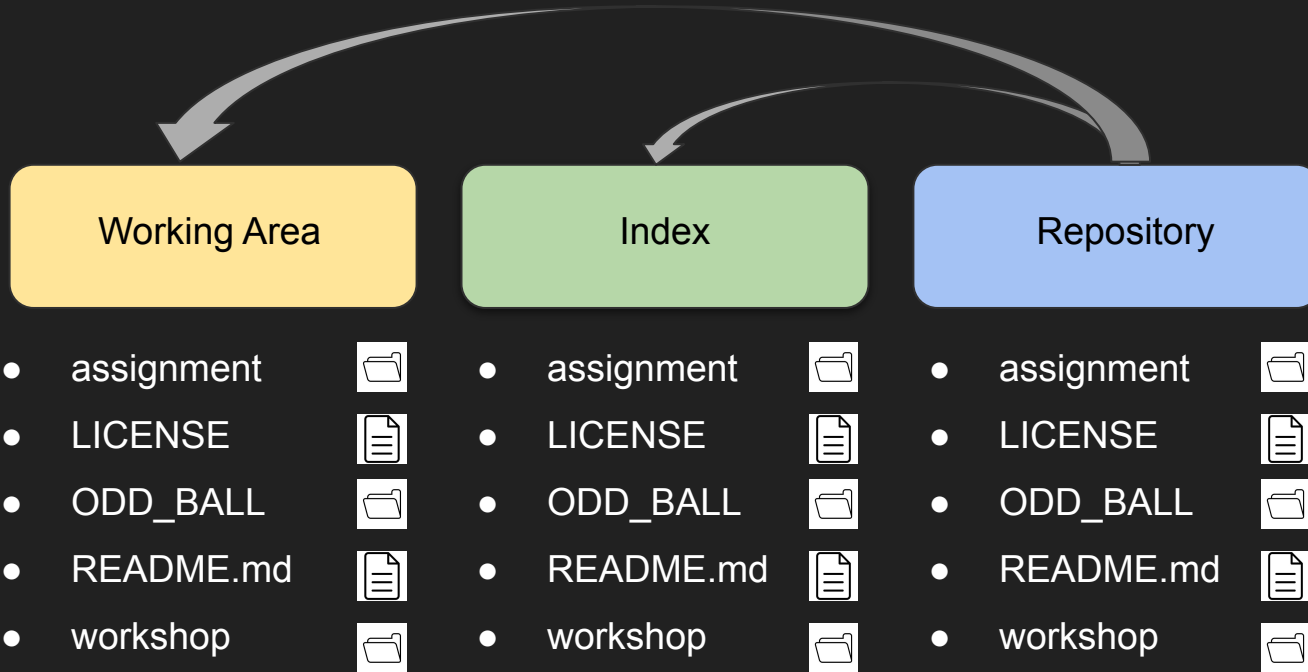
Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git checkout branch II








git checkout branch II








How to remove files?






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index





- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

rm README.md






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

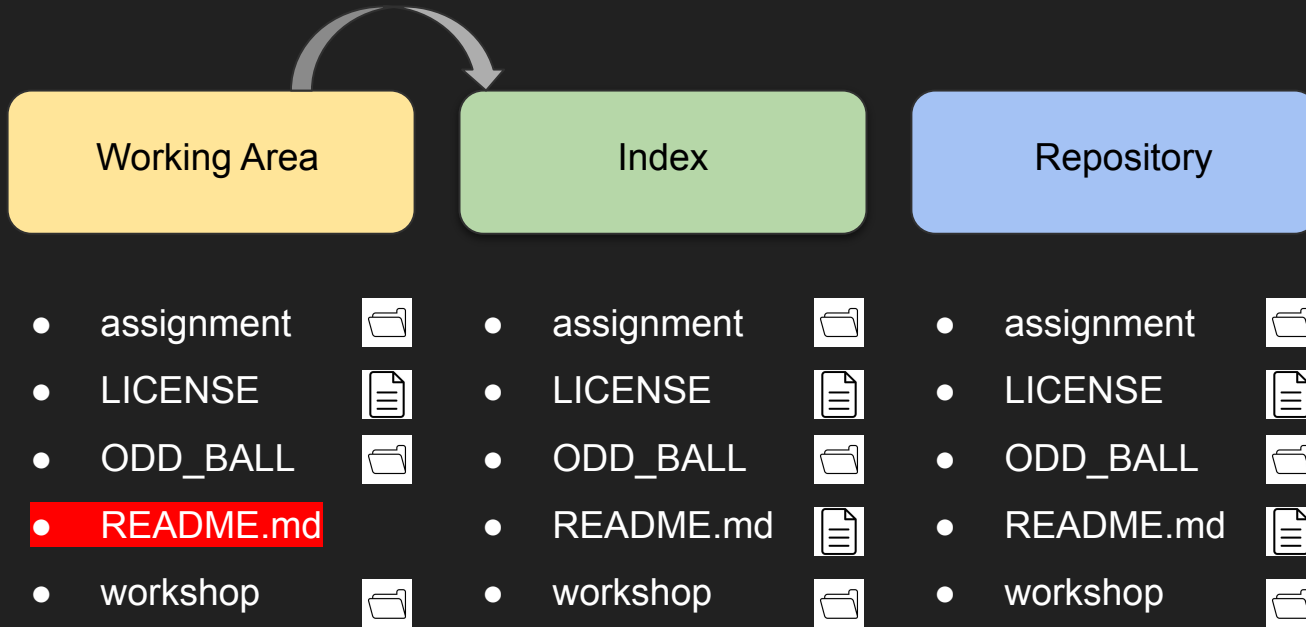
Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

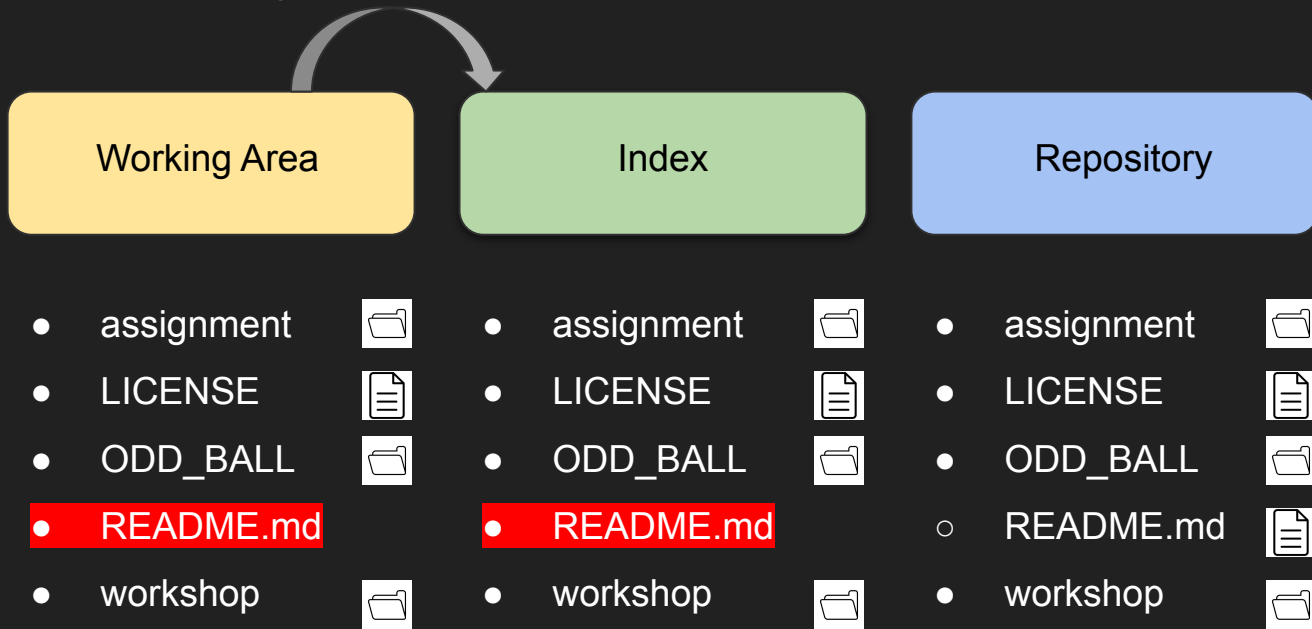
Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

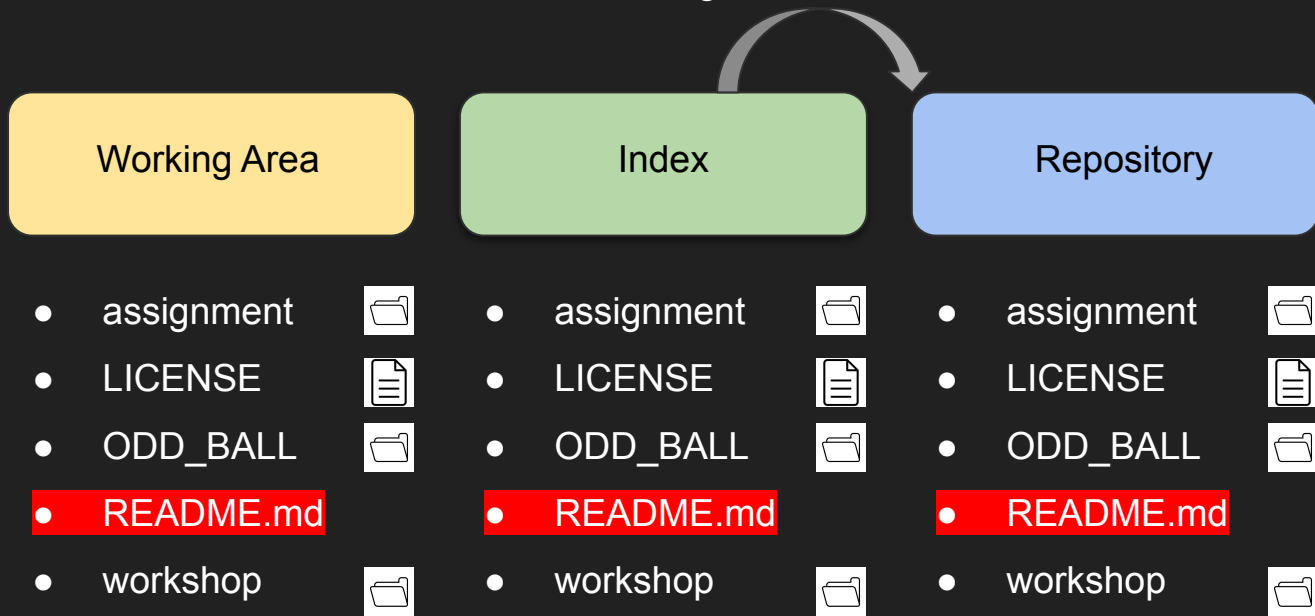
git add README.md



git add README.md







git commit -m "I don't like README.md file"







Seem like too many steps

git rm README.md






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

Index





- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

Repository





- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git status check






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

On branch master

Changes to be committed:






(use "git restore --staged <file>..." to unstage)

deleted: README.md






git rm is not the opposite of "git add"

I just want to remove a stage






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository





- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git rm --cached






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index





- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 

Repository




- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git status check






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- works

Index

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- 

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

deleted: README.md






Untracked files:

(use "git add <file>..." to include in what will be committed)





README.md

ls






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- **README.md**
- workshop 






Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






```
zen@pop-os:~/gitasktic$ ls
assignment LICENSE ODD_BALL README.md workshop
```

Renaming files






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository






- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

mv workshop WorkShop






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- WorkShop 

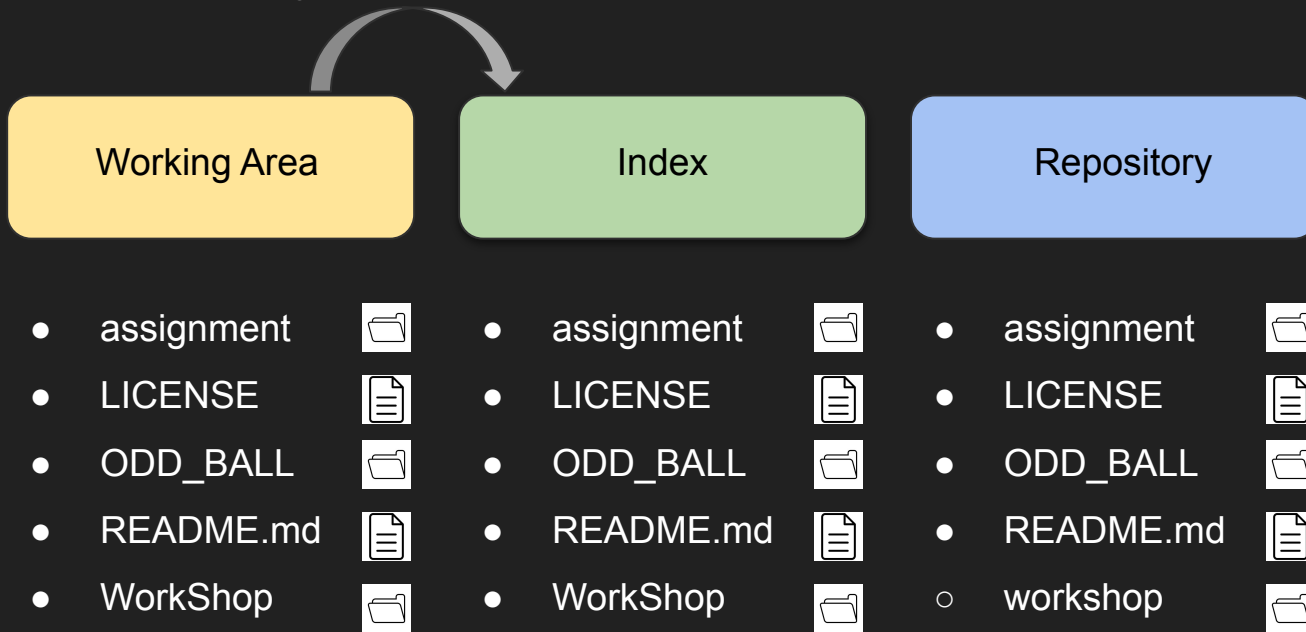
Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git add WorkShop



git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

```
deleted:    README.md
new file:   WorkShop/WI/wsi_codes.bash
new file:   WorkShop/WI/wsi_slides.pdf
new file:   WorkShop/WII/wsii_slides.pdf
new file:   WorkShop/WIII/wsiii_slides.pdf
new file:   WorkShop/workshop_outline.pdf
```

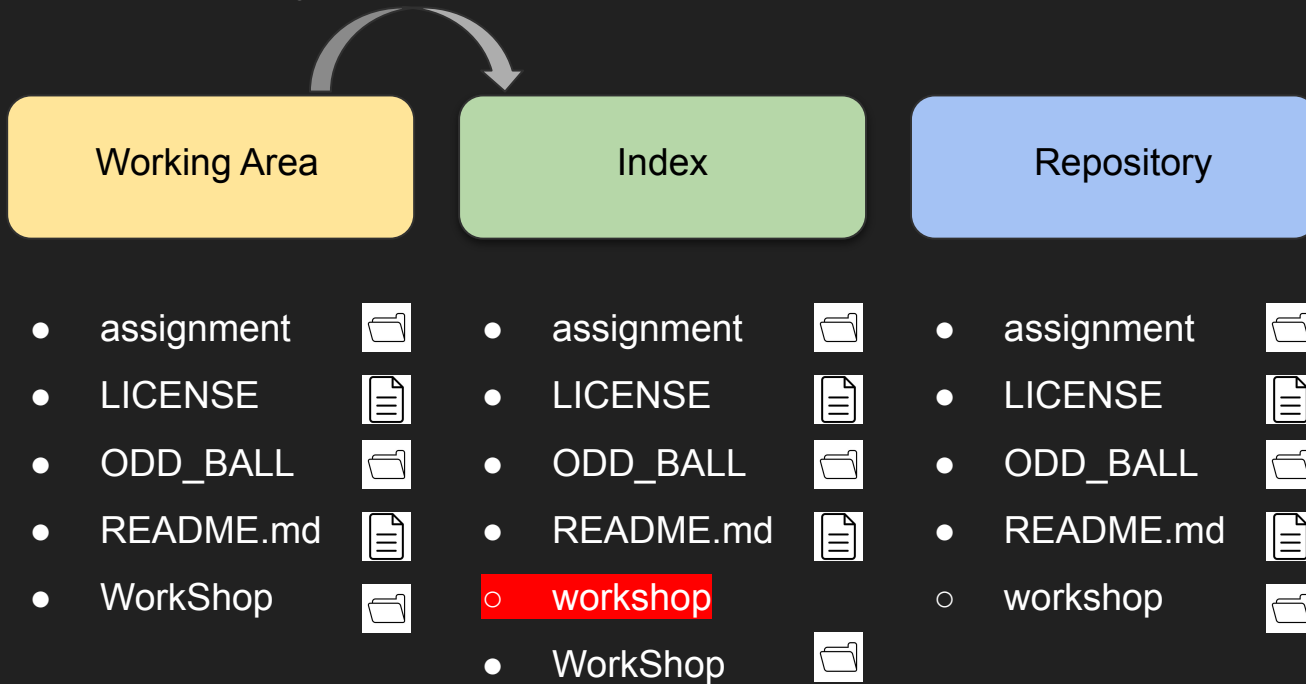
Changes not staged for commit:

(use "git add/rm <file>..." to update what will be committed)

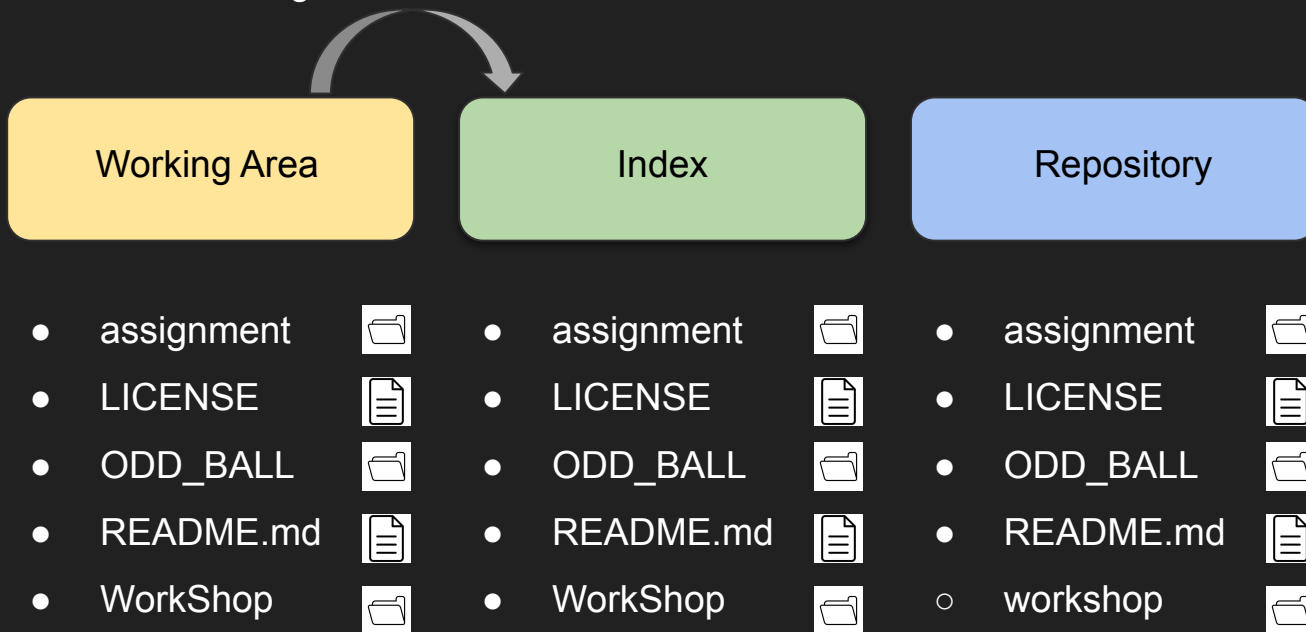
(use "git restore <file>..." to discard changes in working directory)

```
deleted:    workshop/WI/wsi_codes.bash
deleted:    workshop/WI/wsi_slides.pdf
deleted:    workshop/WII/wsii_slides.pdf
deleted:    workshop/WIII/wsiii_slides.pdf
deleted:    workshop/workshop_outline.pdf
```

git add WorkShop



git add .



git status

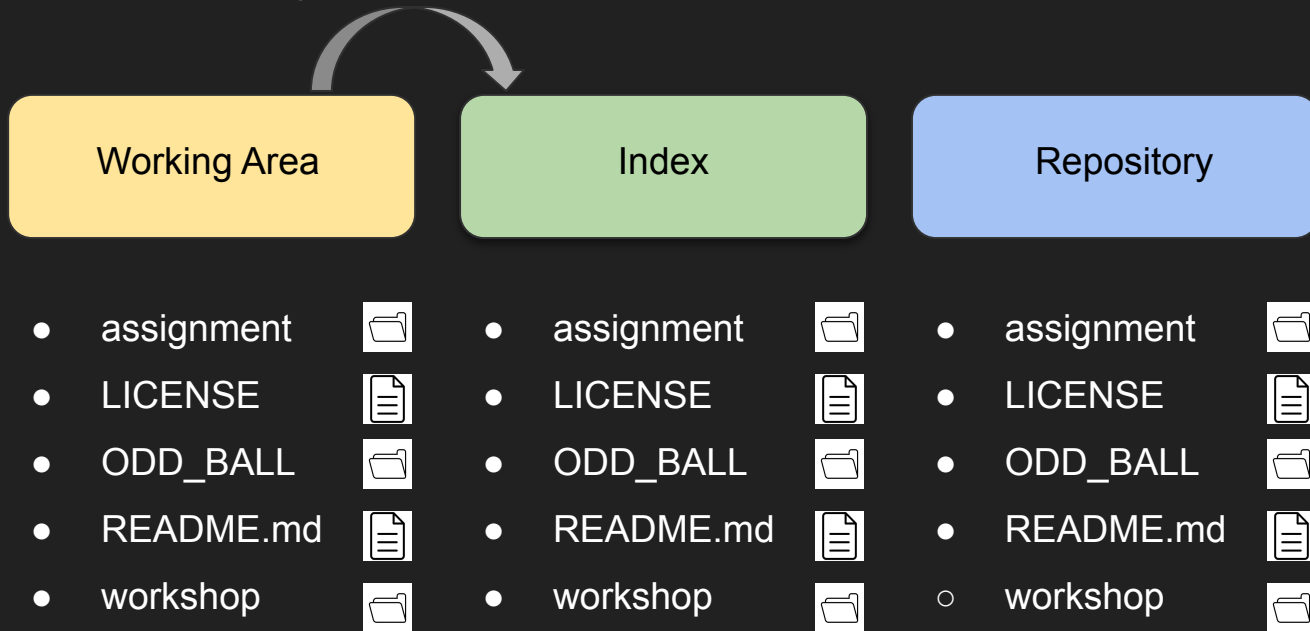
On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

```
renamed:    workshop/WI/wsi_codes.bash -> Workshop/WI/wsi_codes.bash
renamed:    workshop/WI/wsi_slides.pdf -> Workshop/WI/wsi_slides.pdf
renamed:    workshop/WII/wsii_slides.pdf -> Workshop/WII/wsii_slides.pdf
renamed:    workshop/WIII/wsiii_slides.pdf -> Workshop/WIII/wsiii_slides.pdf
renamed:    workshop/workshop_outline.pdf -> Workshop/workshop_outline.pdf
```

git mv WorkShop workshop



```
On branch master
nothing to commit, working tree clean
```

So far What have you learned so far?

The Four Areas of GIT

Stash

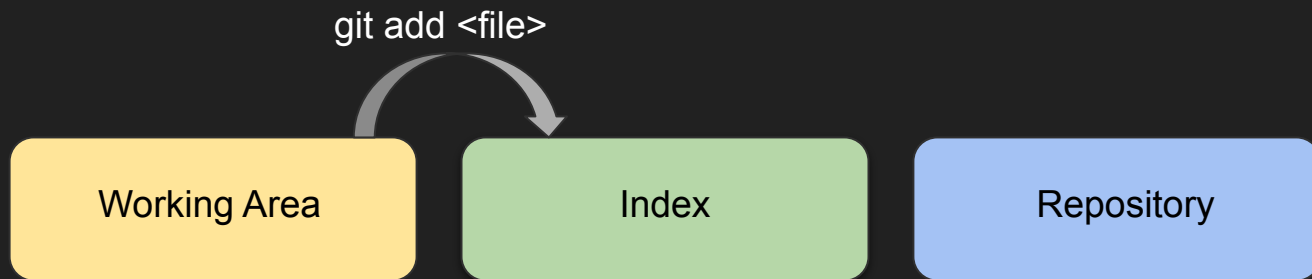
Working Area

Index

Repository

git status

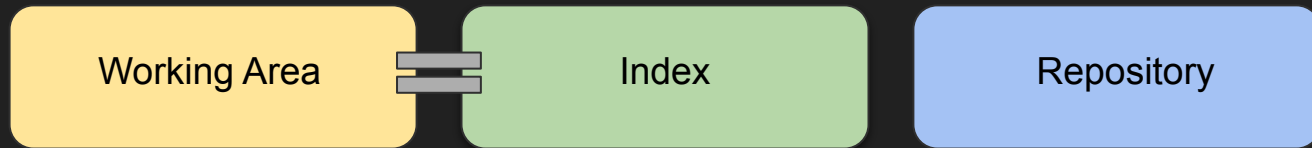




git commit

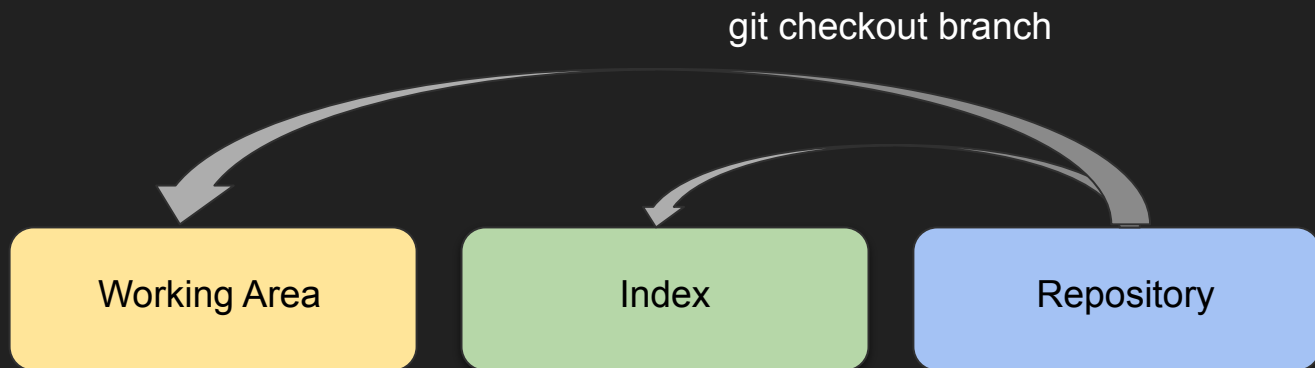


git diff

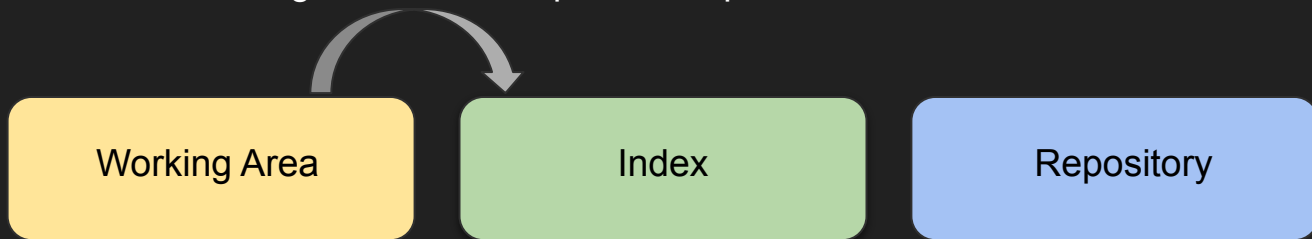


git diff --cached





`git mv WorkShop workshop`



Congrats you are one step closer to git mastery!!

Advance Git Plays

How many commands you know can move a branches?

- commit
- merge
- rebase
- pull

reset

- can moves the current branch to any position of repository
- additionally, reset can optionally copies data from repository to other areas.

Why i have not introduce git reset until now

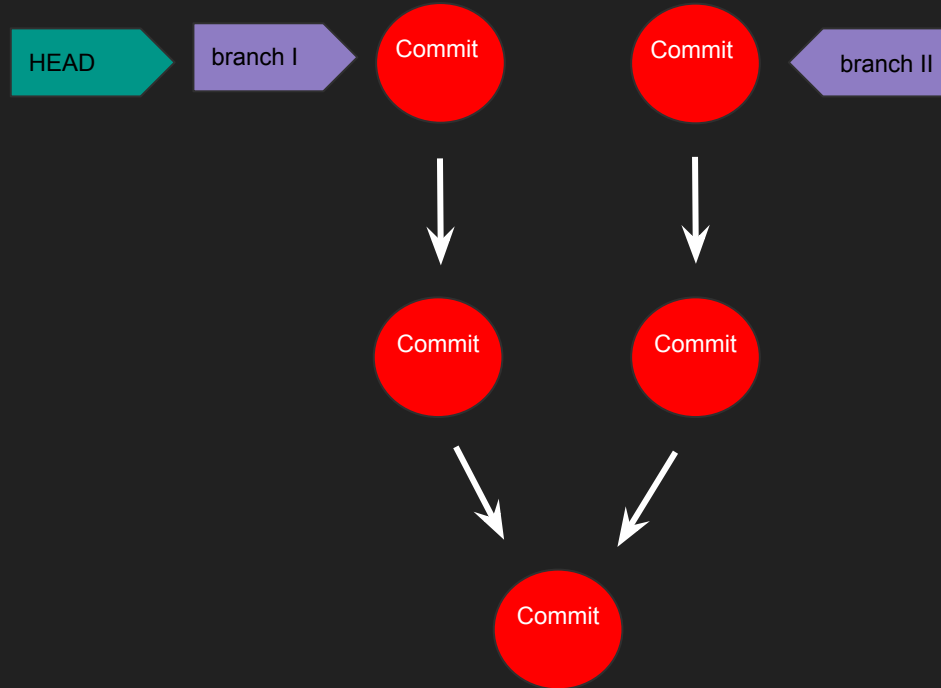


Why I have not introduce git reset until now

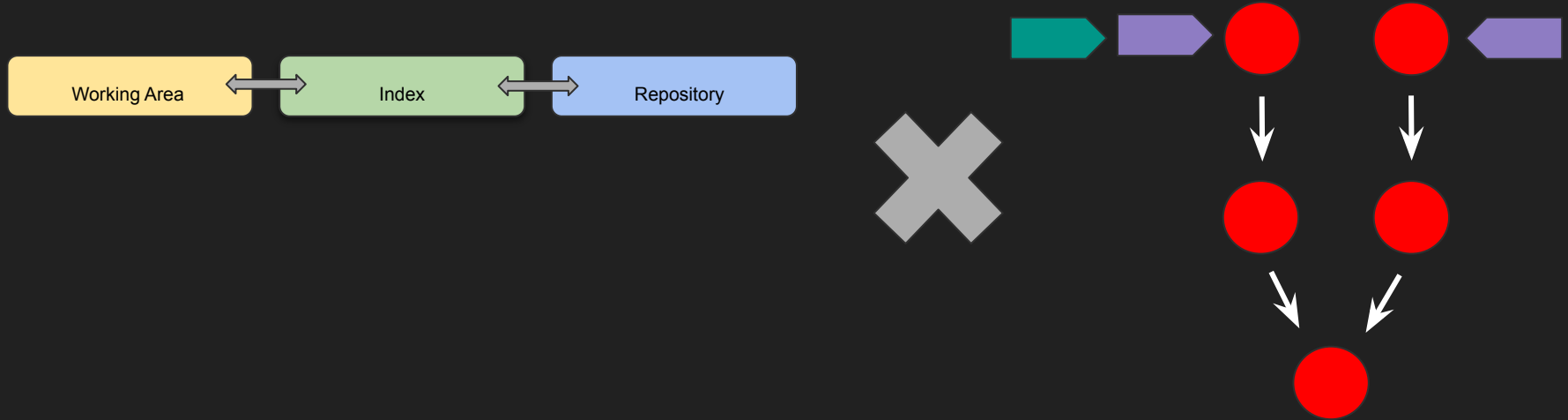
Why i have not introduce git reset until now



Why i have not introduce git reset until now

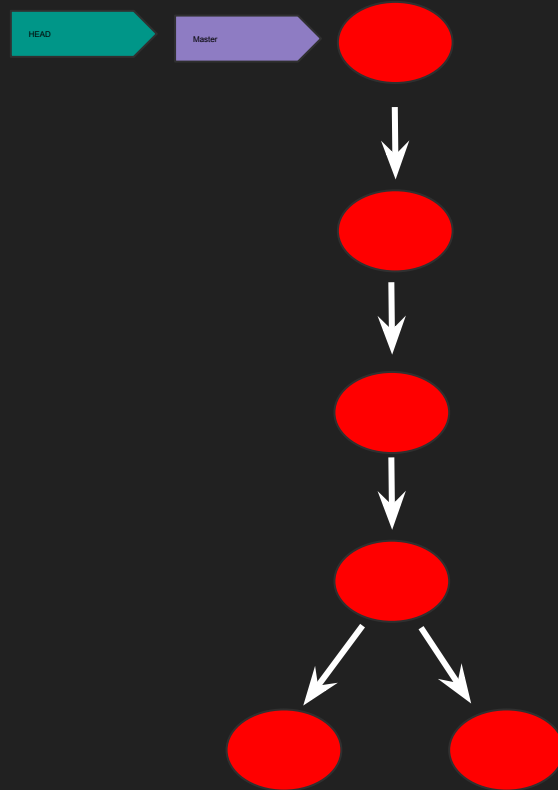


Why i have not introduce git reset until now

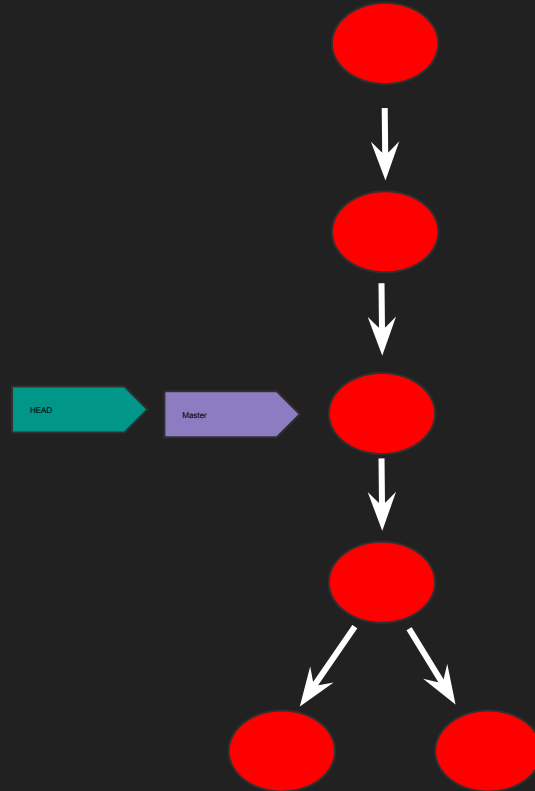


Two step process of git reset

Step I : move current branch to pointed location

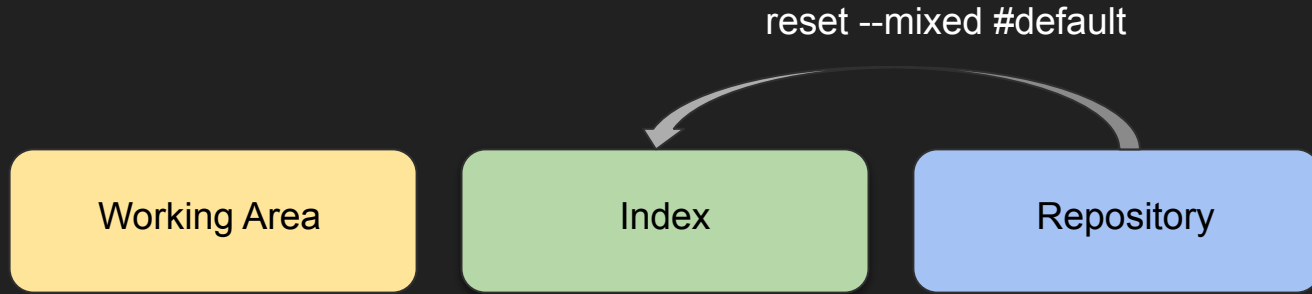


Step I : move current branch to pointed location

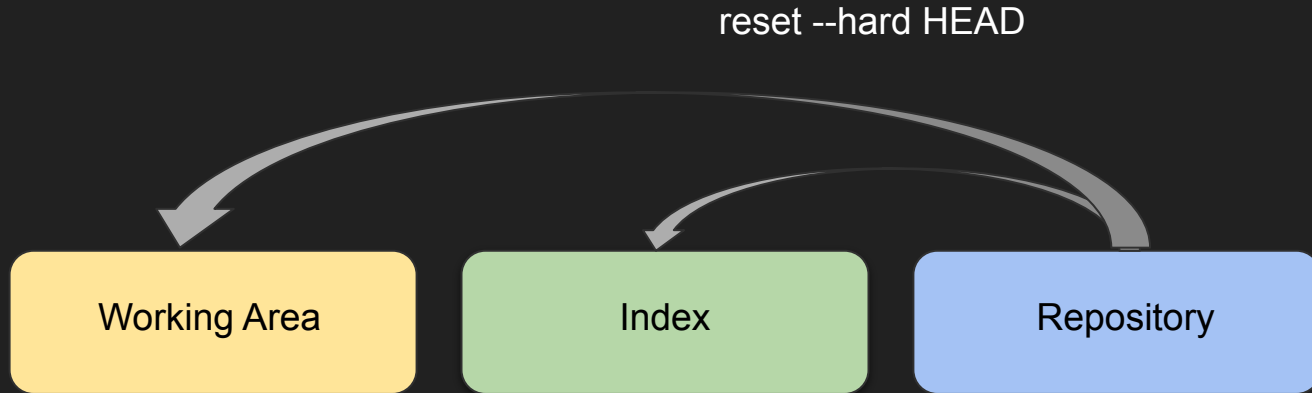


Step II: Move across the Areas

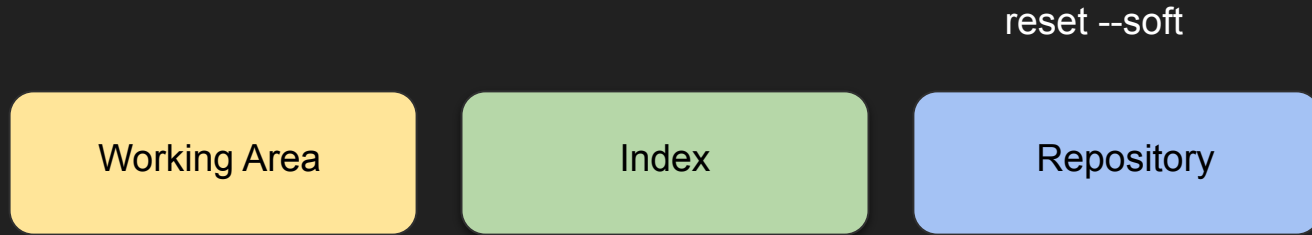
Step II: Move across the Areas



Step II: Move across the Areas



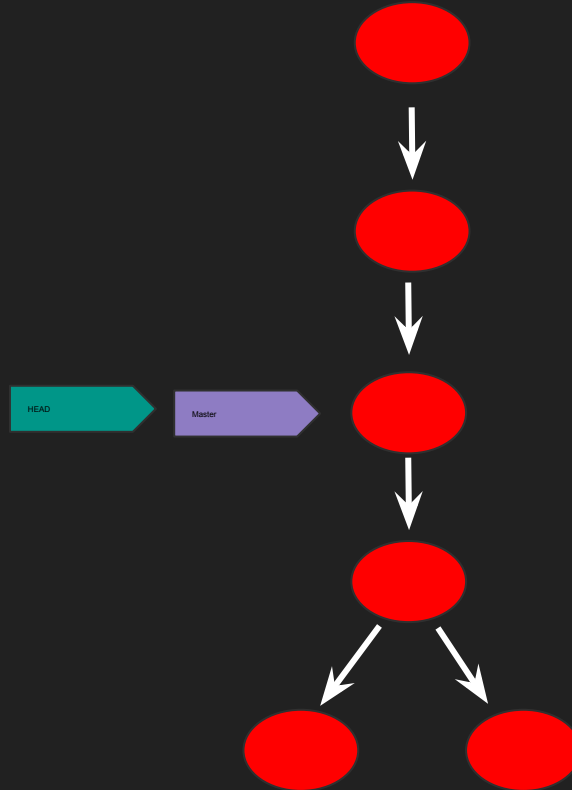
Step II: Move across the Areas



Practical:

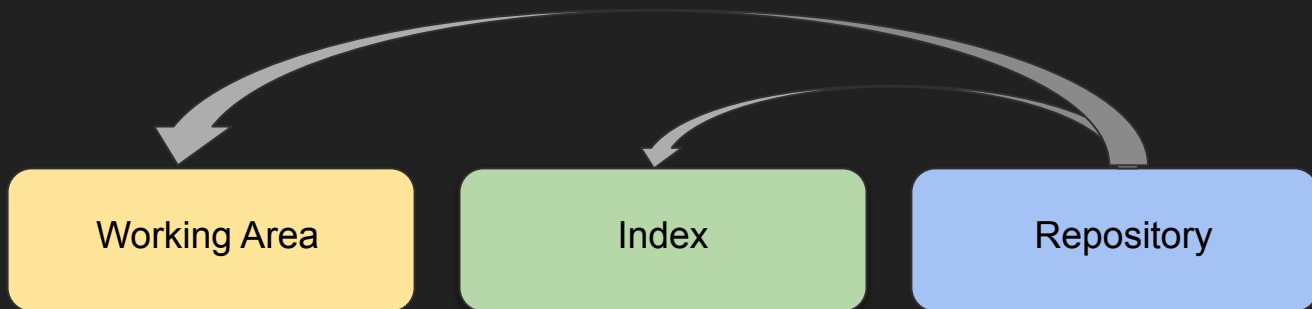
Go back to earlier work

```
reset --hard <shia>
```

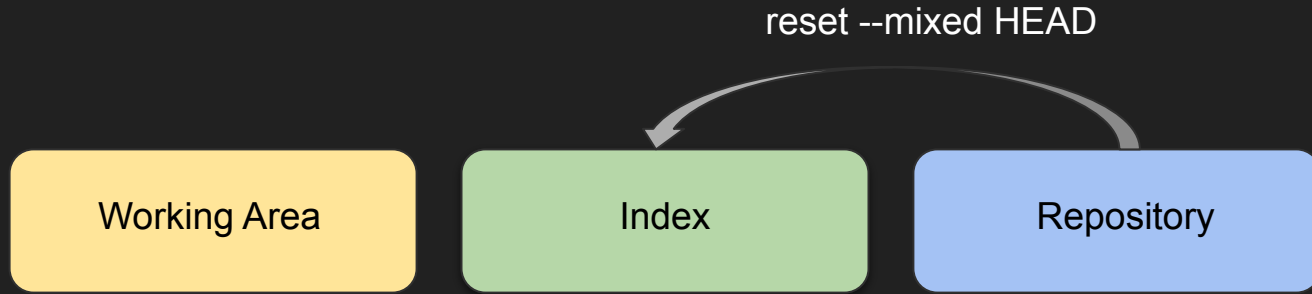


Don't like I what have done so far, git reset HEAD

reset --hard HEAD



Unstage work so far

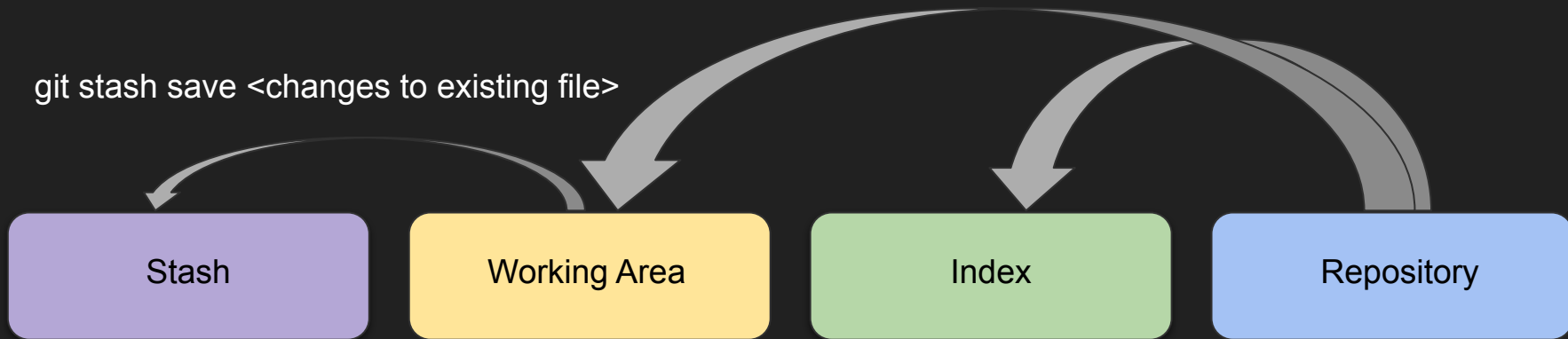


Fourth area

Stash


- Stash is like a clipboard
 - merge conflict
 - experiment with coding

git stash save <changes to existing file>









git statsh --include-untracked






Stash

- readme_2.md 






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 
- readme_2.md 


Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 






Stash

- readme_2.md 






Working Area

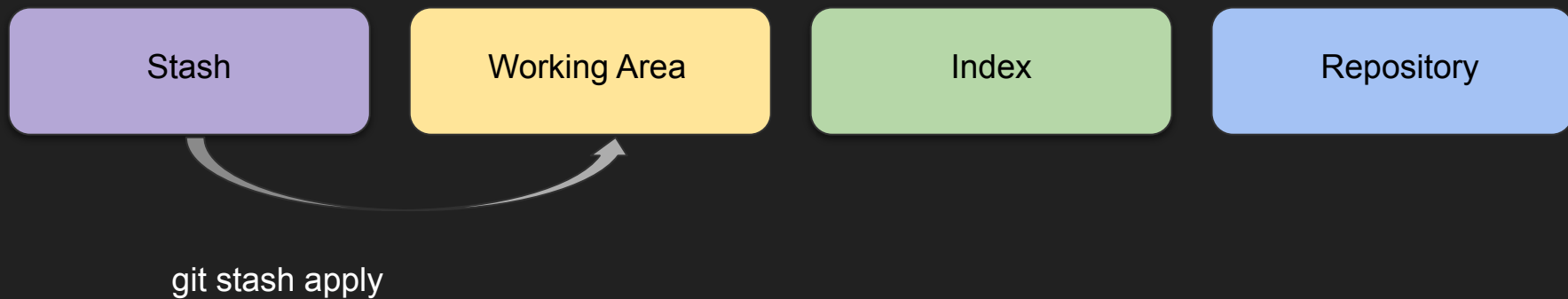
- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Index


- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository







- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 








Stash

- readme_2.md 






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 
- readme_2.md 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository







- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

git statsh clear






Stash

○ readme_2.md






Working Area

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 
- readme_2.md 

Index

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 

Repository

- assignment 
- LICENSE 
- ODD_BALL 
- README.md 
- workshop 



Git can operate on things that are smaller than a file

Hunk by hunk

git add --patch ODD_BALL/odd_ball.py

(1/6) Stage this hunk [y,n,q,a,d,j,J,g/,e,?]? ?

y - stage this hunk

n - do not stage this hunk

q - quit; do not stage this hunk or any of the remaining ones

a - stage this hunk and all later hunks in the file

d - do not stage this hunk or any of the later hunks in the file

g - select a hunk to go to

/ - search for a hunk matching the given regex

j - leave this hunk undecided, see next undecided hunk

J - leave this hunk undecided, see next hunk

e - manually edit the current hunk

? - print help

```
git checkout "shia" <file>
```

with understand of git so far,
you will no long have problem to learn new git functions.

Git New commands

- switch
- restore

What does checkout

Move to a different branch

- switch

Recovering an earlier commit

- restore

Two new functions from git

git restore recovering an earlier commit

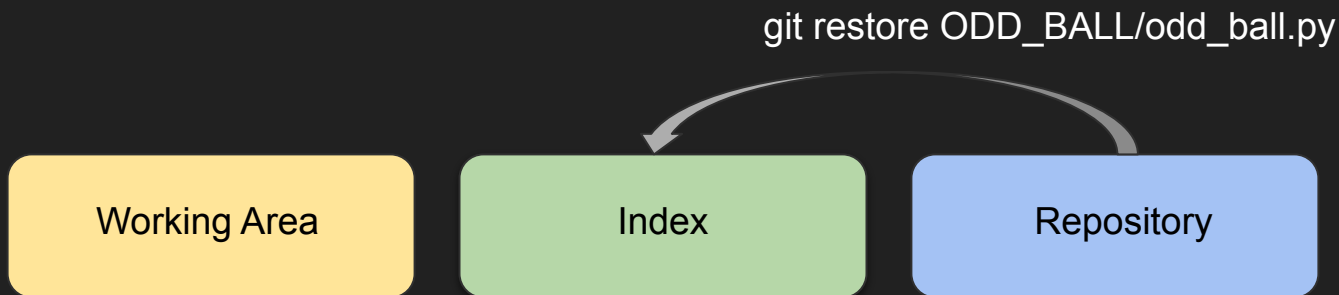
```
On branch master
```

```
Changes to be committed:
```

```
(use "git restore --staged <file>..." to unstage)
```

```
    modified:   ODD_BALL/odd_ball.py
```

git restore



On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore --staged <file>..." to discard changes in working directory)

modified: ODD_BALL/odd_ball.py

Git is toolbox

- reset and checkout
- unstage file, move late
- git add modified file, solve merge conflict
- to unstage file you need to how the index work, and once you do, you can use git reset, git rm --cached, or restore..etc There is not one command to use.
- git is flexibility.

History

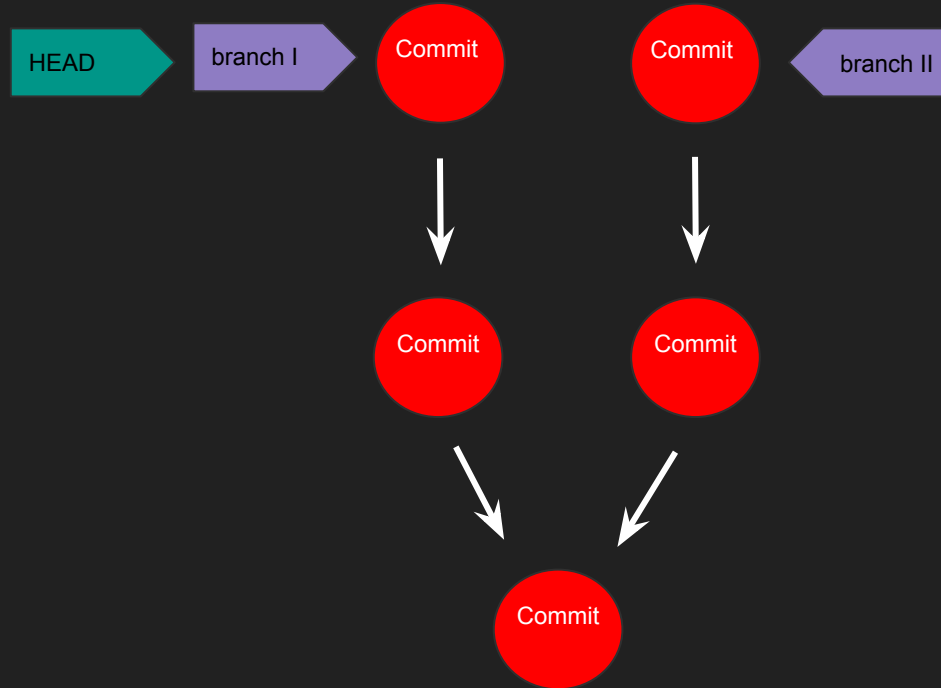
git status



git log



Tracking changes in history



git show Head^

git show Head~2

git diff Head..Head^^

git blame

git log --graph --decorate --oneline

git show HEAD

Workshop IV: Final project

1. Find partner in group of 3.
2. Pull color switching task data from paul's github
3. Discuss among yourself how to divided work for this analysis
 - a. Research questions may be asked?
 - i. Which conditions RT is significant faster?
 - ii. How to visualize the data?
 - iii. Is there a correlation between rumination and RT?
 -etc
4. Create a group folder inside assignment folder, and collaborate with each other to divided work inside that folder
5. Send a pull requested with your results to paul's github before 10 am on friday August 23.

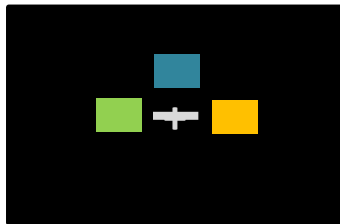
Preference judgment



- † If you see the vertical line is longer than horizontal line in the fixation cross, please do preference judgment.

Top color patch is the one you have to compare with. What you have to do is pressing left or right bottom to indicate which pair, the color patch on the left or right paired with the top one, you prefer.

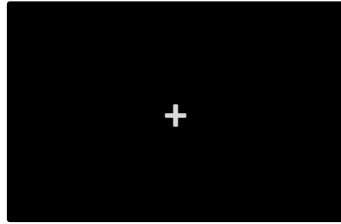
Similarity judgment



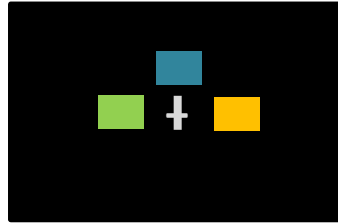
+ If you see the horizontal line is longer than vertical line in the fixation cross, please do similarity judgment.

Top color patch is the one you have to compare with. What you have to do is pressing left or right bottom to indicate which color patch on the left or right is physically similar to the top one.

Procedure



1500-2000 ms



2000 ms



1500 ms

This experiment will have two parts:

1. Practice
2. Formal experiment