Lecture Note 3.
Git

March 26, 2025

Kwanghee Lee
Dept. of Software
Dankook University

kh-lee@dankook.ac.kr

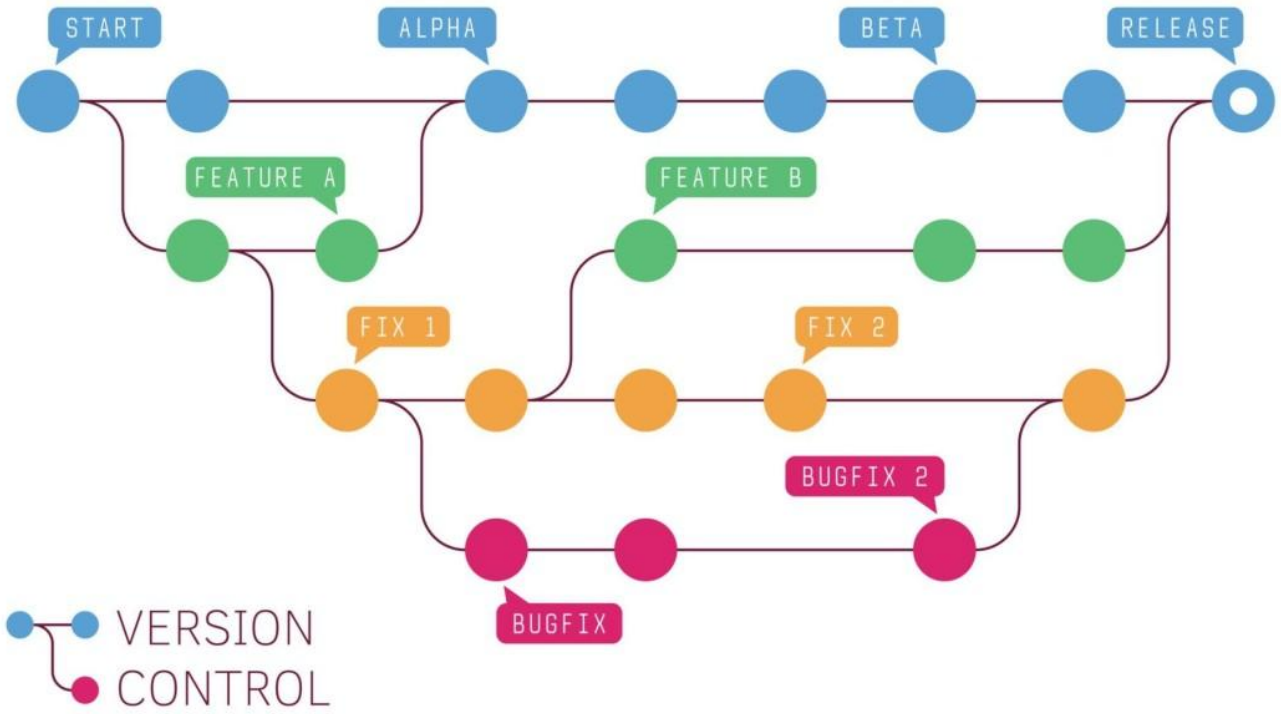
(본 교재는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 'SW중심대학사업' 지원을 받아 제작 되었습니다.)

Contents

- What is version control?
- Version Control System(VCS)
  - Centralized
  - Distributed
- How to use Git?
- Practice

What is version control? (1/4)

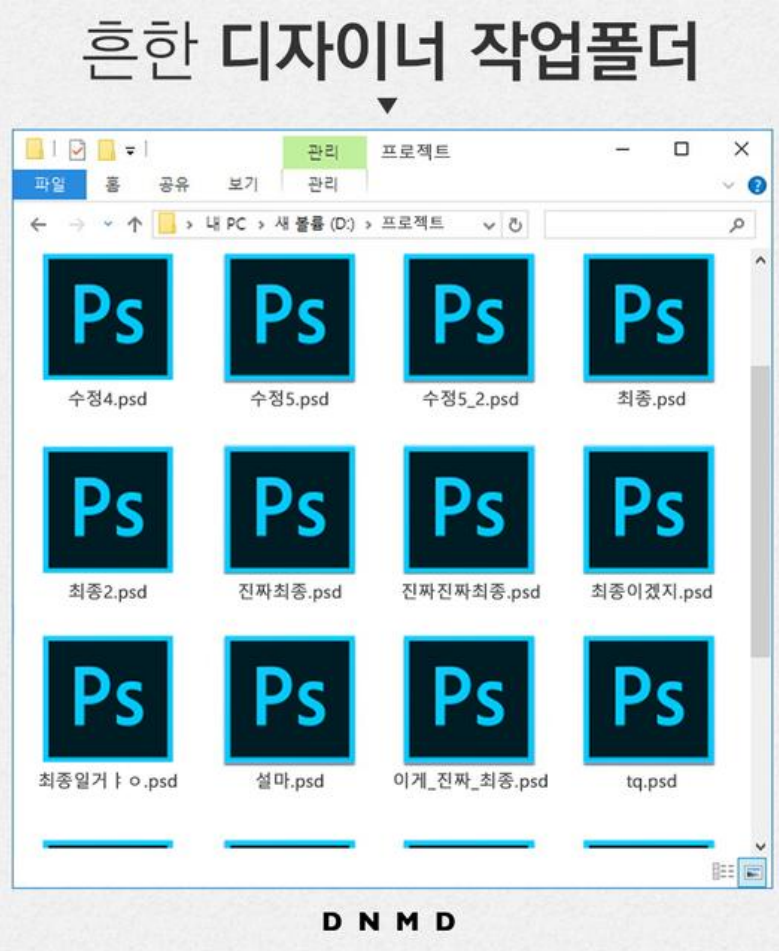
- **Version control**
  - Version control is a system that records the **history of changes to files**, allowing you to track modifications and revert to a **specific point** in time if needed.



Ref: google image

What is version control? (2/4)

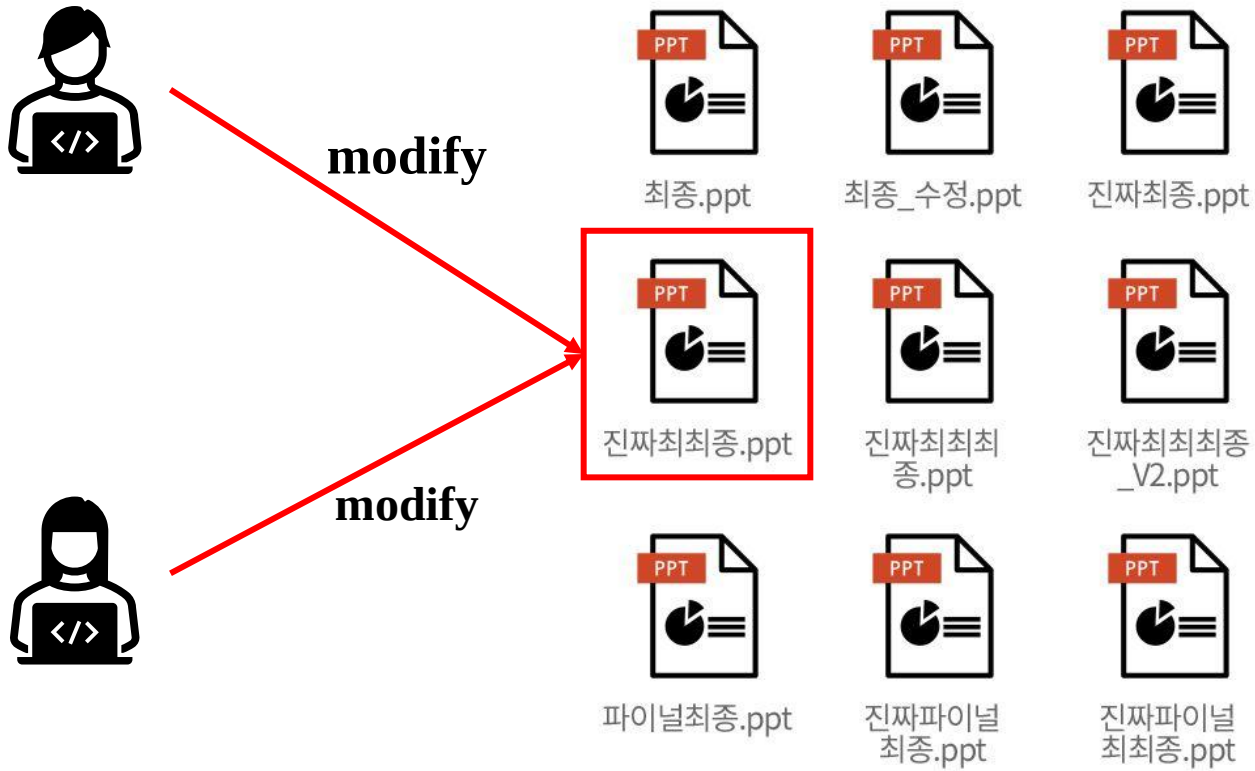
- **Why do we need version control?**
  - File version control



Ref: google image

What is version control? (3/4)

- **Why do we need version control?**
  - Cooperation – Shared file version control



Ref: google image

What is version control? (4/4)

- **Why is a version control system?**
  - A tool that automatically records all changes to a file and lets you go back to any point in time.

- **Version control system**
  - Centralized
    - SVN(SubVersion), CVS
  - Distributed
    - Git

Ref: google image

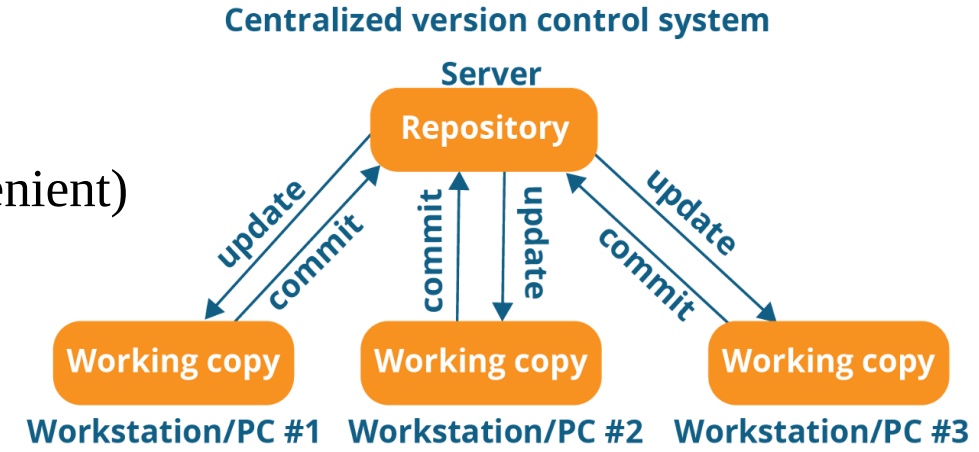
Version Control System(VCS) (1/2)

- **Centralized VCS**
  - SVN(SubVersion), CVS

  - **All version** information is stored on **one central server (center repository)**
  - **Users receive data from the server** and work with it

  - **Pros**
    - Management is simple
    - Easy access control

  - **Cons**
    - Internet connection required (offline operation is inconvenient)
    - If the central server fails, all work stops
    - Server load is high and bottlenecks are likely to occur



Ref: google image

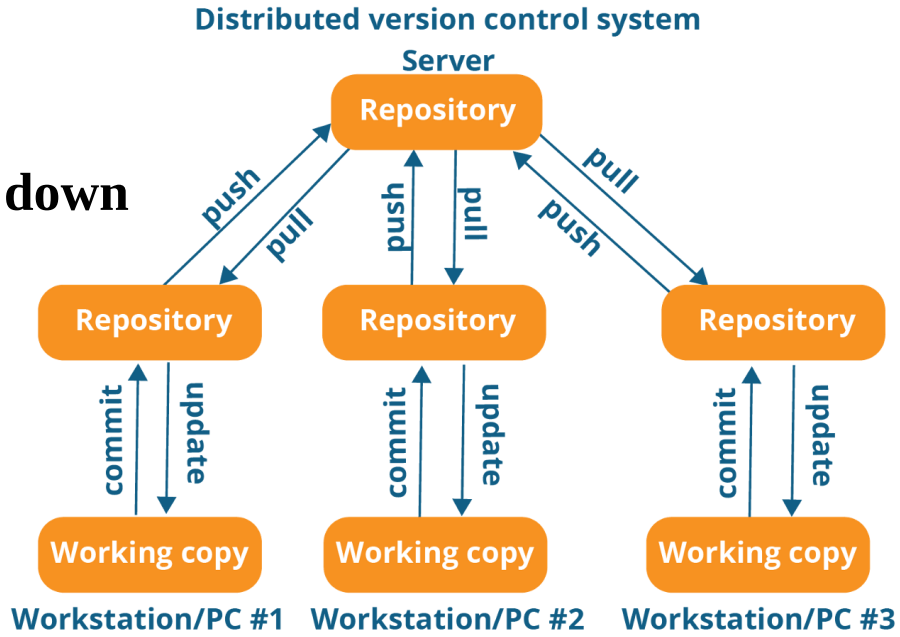
Version Control System(VCS) (2/2)

- **Distributed VCS**
  - Git

  - All users **copy** the entire **project history to their local**
  - Can perform operations such as viewing history, committing, and branching **locally**

  - **Pros**
    - Most operations can be performed **offline**
    - Fast performance (**local**)
    - **Local work remains** unaffected even if the **central server is down**

  - **Cons**
    - May take some time to understand the structure at first
    - Repository size can be large



Ref: google image

How to use Git? (1/16)

- **Install Git**
  - Windows: https://git-scm.com
  - MacOS: *brew install git*
  - Linux: *sudo apt install git*

- **Config user information**
  - *git config --global user.name "honggildong"*
  - *git config --global user.email "hgd@dankook.ac.kr"*

*--global: All git project

*local: One git project

How to use Git? (2/16)

- **Git local commands**
  - *git init*
    - Initialize a new Git repository in the current directory.
  - *git status*
    - Show the current state of the working directory and staging area.
  - *git log*
    - Display a list of previous commits in the repository.
  - *git diff*
    - Show the differences between changes in files (unstaged or staged).

How to use Git? (3/16)

- **Git local commands**
  - *git add [file or .]*
    - Add file(s) to the staging area for the next commit.
  - *git commit –m “commit message”*
    - Save the staged changes to the repository with a message.
  - *git reset [file or commit]*
    - Unstage files or move the current HEAD to a specific commit.
  - *git revert [commit]*
    - Create a new commit that undoes the changes of a previous commit.

How to use Git? (4/16)

- **Git local commands**
  - *git branch*
    - List, create, or delete branches.
  - *git checkout [branch or file]*
    - Switch branches or restore files to a previous state.
  - *git switch [branch]*
    - Switch branches more safely (recommended over checkout).
  - *git merge [branch]*
    - *Merge another branch into the current branch.*

How to use Git? (5/16)

- **Create local repository**

```
~/Lecture/git
khlee$ mkdir repo_dir
~/Lecture/git
khlee$ cd repo_dir/
~/Lecture/git/repo_dir
khlee$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/khlee/Lecture/git/repo_dir/.git/
~/Lecture/git/repo_dir
khlee$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

Working directory [repo_dir]

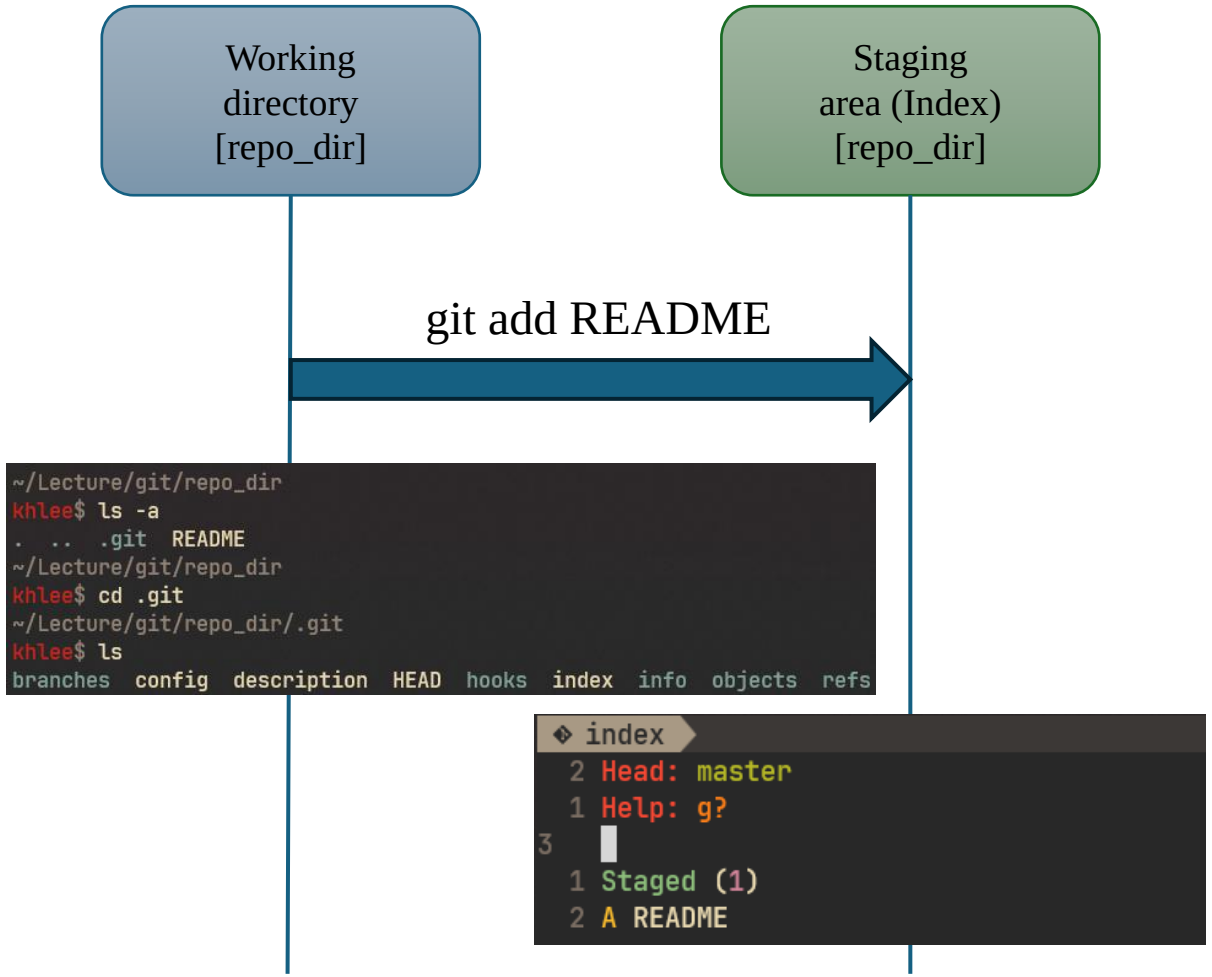
Staging area [repo_dir]

Local repository [repo_dir]

How to use Git? (6/16)

- **Add file**

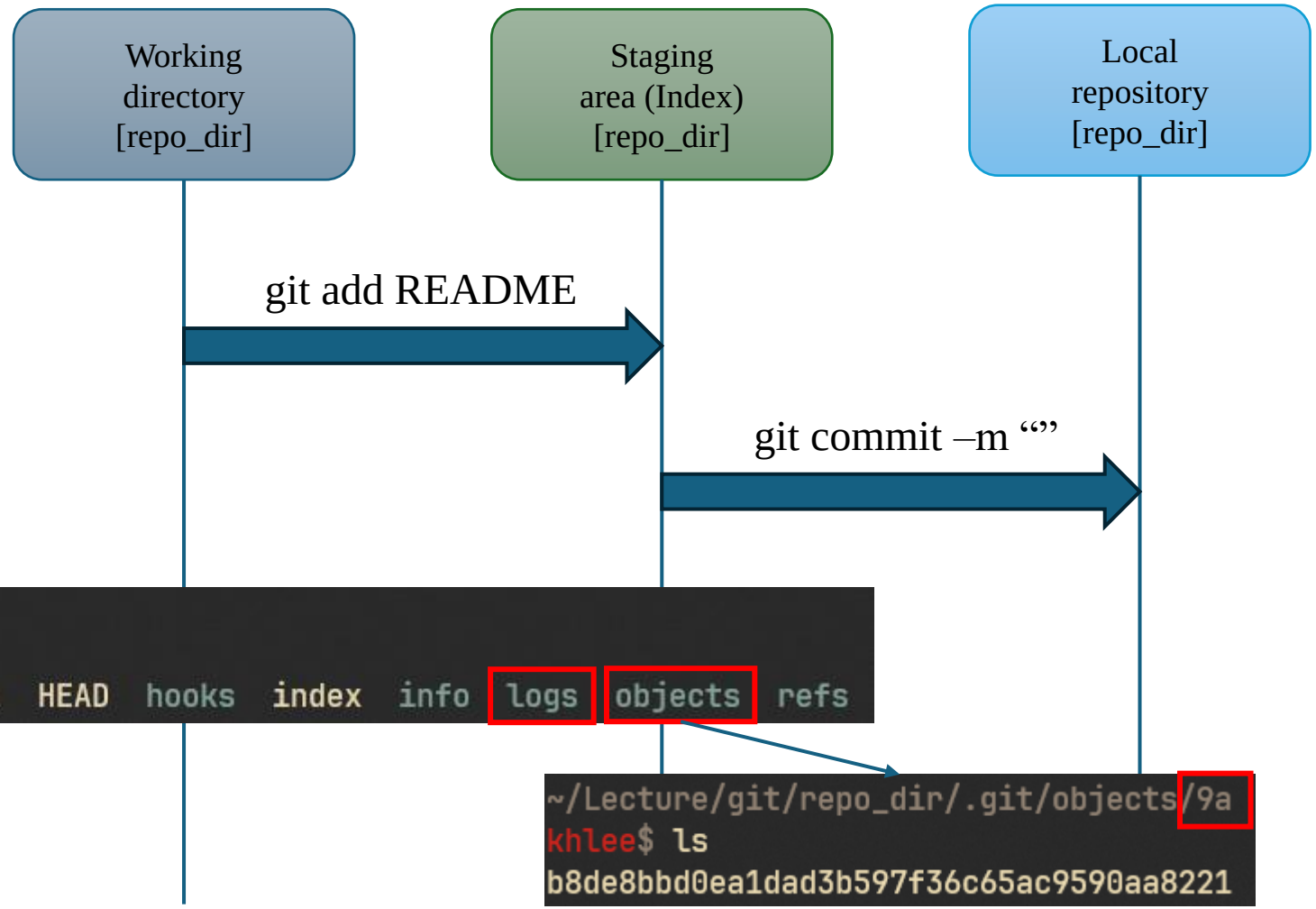
```
~/Lecture/git/repo_dir
khlee$ ls
README
~/Lecture/git/repo_dir
khlee$ git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
~/Lecture/git/repo_dir
khlee$ git add README
~/Lecture/git/repo_dir
khlee$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README
```



How to use Git? (7/16)

- **Commit file**

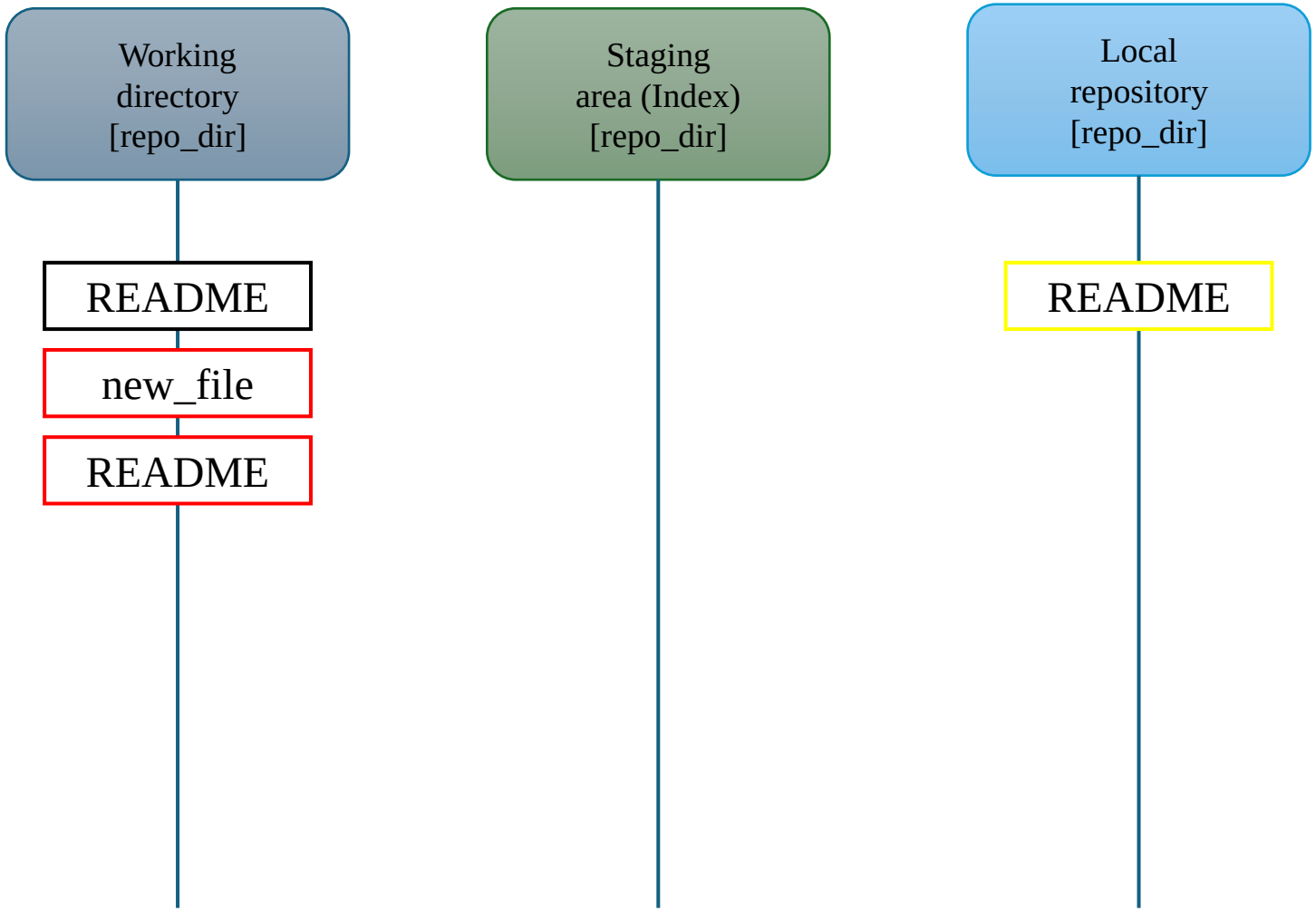
```
~/Lecture/git/repo_dir
khlee$ ls
README
~/Lecture/git/repo_dir
khlee$ git log
fatal: your current branch 'master' does not have any commits yet
~/Lecture/git/repo_dir
khlee$ git commit -m "Create README"
[master (root-commit) 9ab8de8] Create README
 1 file changed, 1 insertion(+)
 create mode 100644 README
~/Lecture/git/repo_dir
khlee$ git log
commit 9ab8de8bbd0ea1dad3b597f36c65ac9590aa8221 (HEAD -> master)
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 17:18:27 2025 +0000

    Create README
```



How to use Git? (8/16)

- **New file**

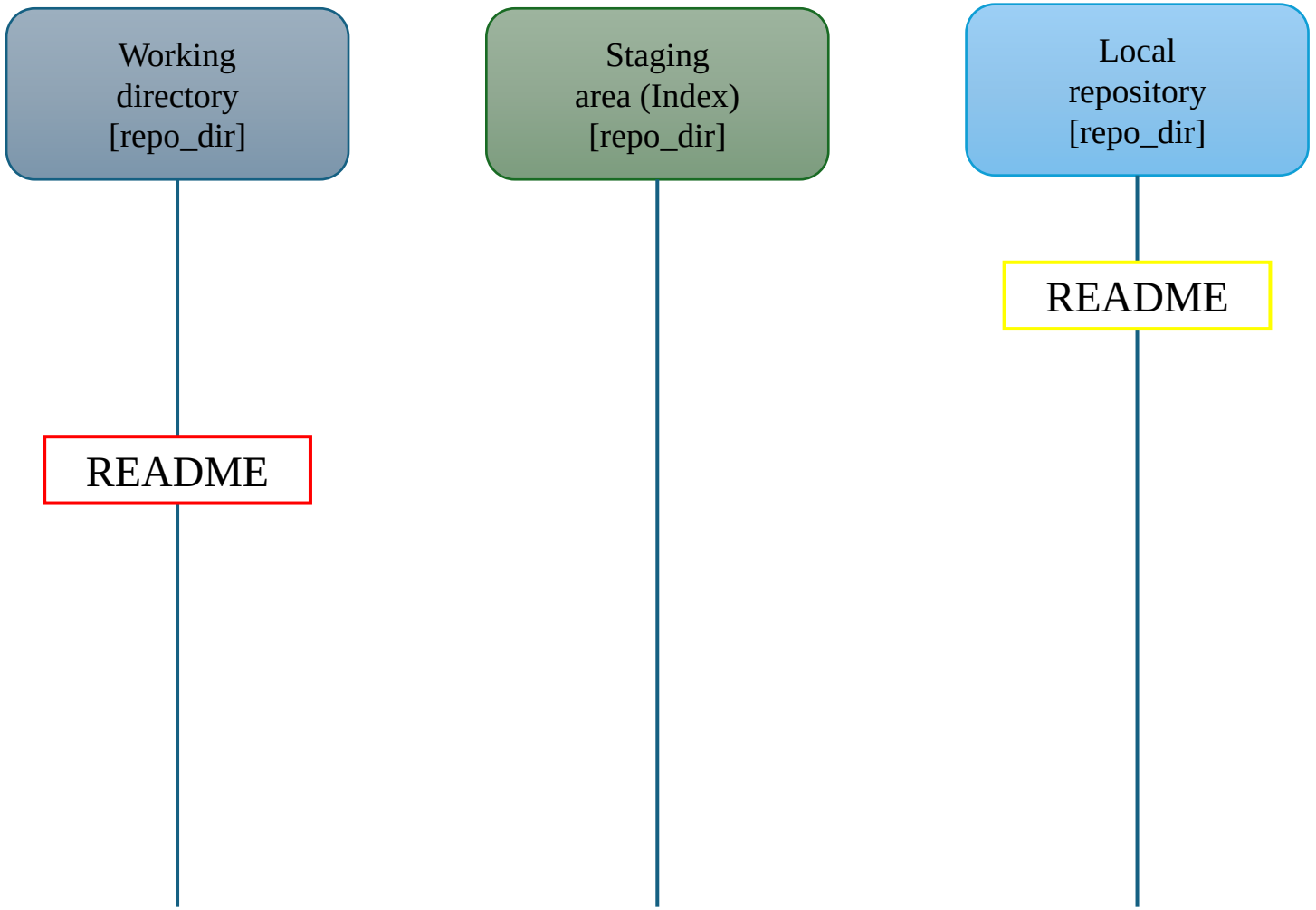
```
~/Lecture/git/repo_dir
khlee$ ls
README
~/Lecture/git/repo_dir
khlee$ touch new_file
~/Lecture/git/repo_dir
khlee$ ls
new_file  README
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        new_file

nothing added to commit but untracked files present (use "git add" to track)
~/Lecture/git/repo_dir
khlee$ v README
~/Lecture/git/repo_dir
khlee$ git status
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

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

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



How to use Git? (9/16)

- **diff**

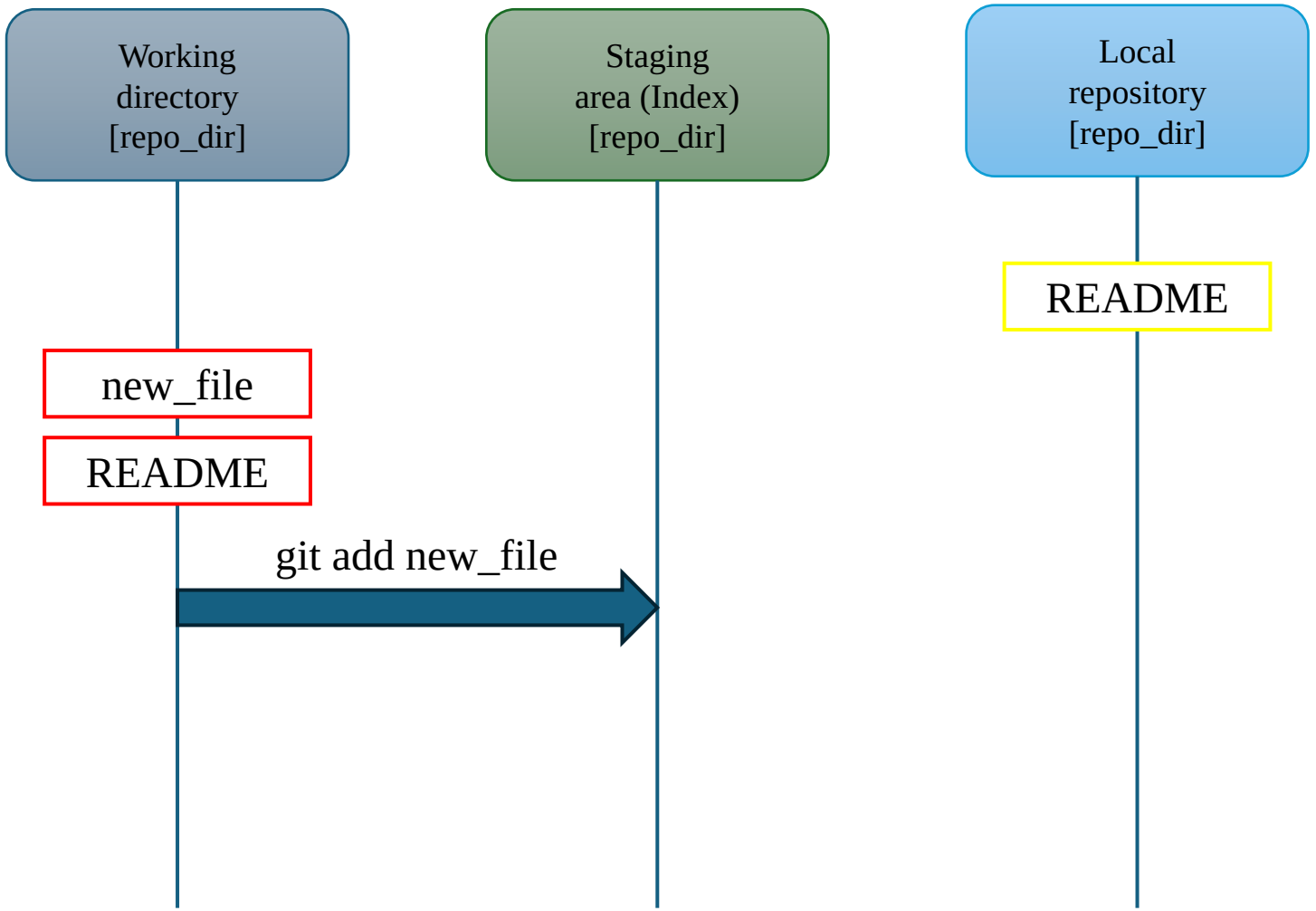
```
khlee$ git diff
diff --git a/README b/README
index 2a02d41..c5fa609 100644
--- a/README
+++ b/README
@@ -1 +1,2 @@
 TEST
+Add TEST2
```



How to use Git? (10/16)

- **add and commit**

```
~/Lecture/git/repo_dir
khlee$ git add new_file
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new_file

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
```

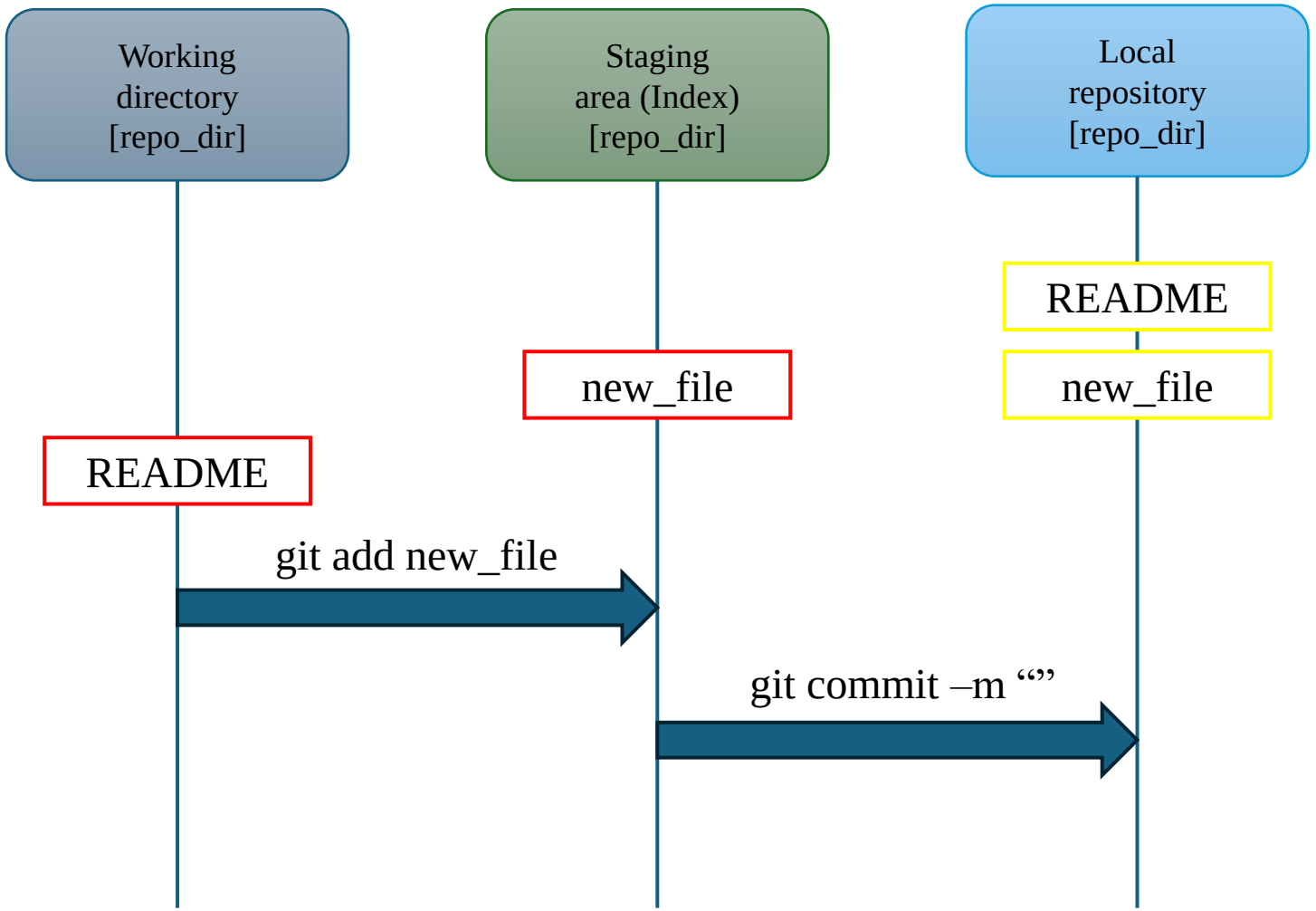


How to use Git? (11/16)

- **add and commit**

```
~/Lecture/git/repo_dir
khlee$ git add new_file
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new_file

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
```

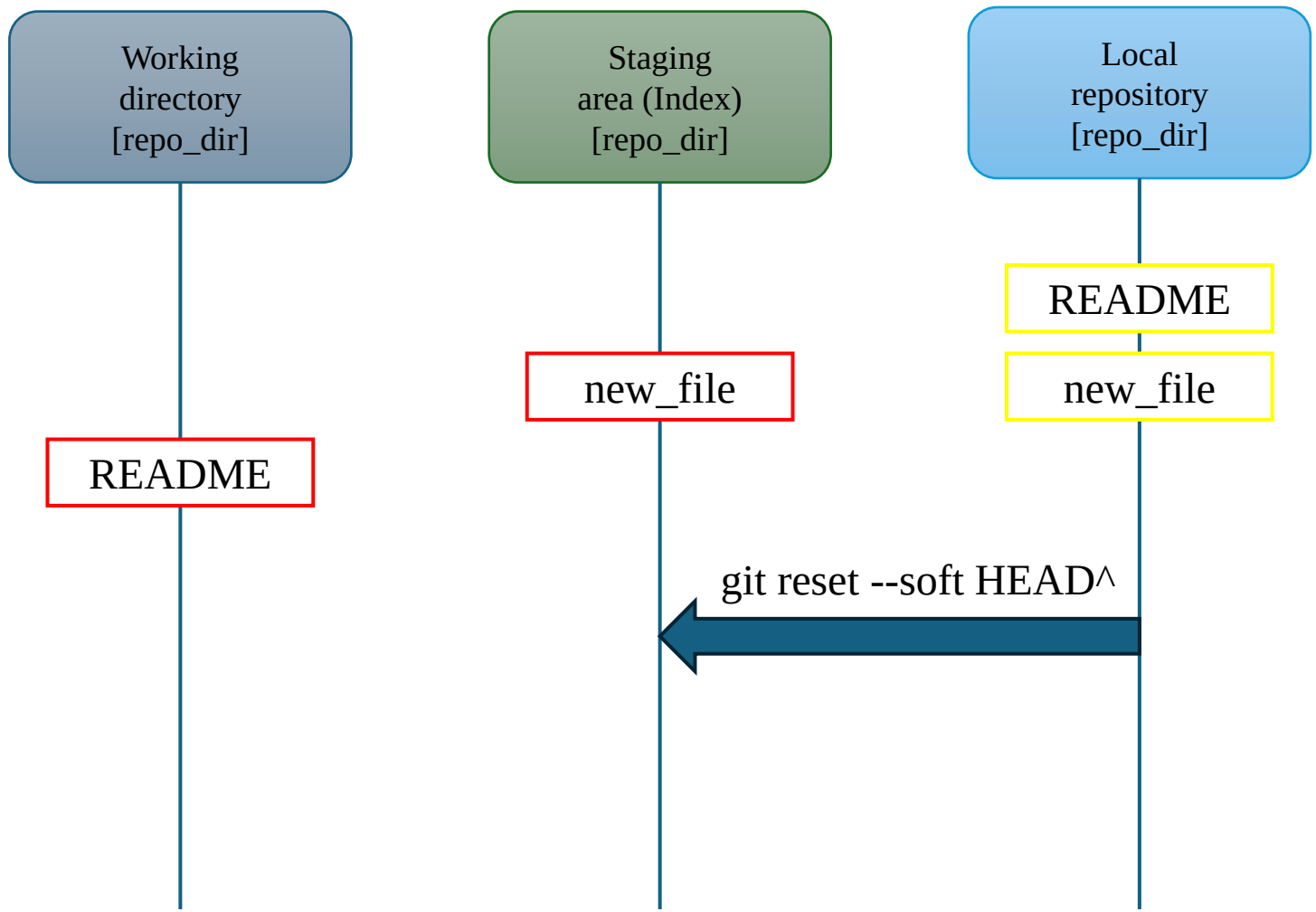
```
~/Lecture/git/repo_dir
khlee$ git commit -m "v0.2"
[master 0a1e86f] v0.2
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 new_file
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README

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



How to use Git? (12/16)

- **reset**

```
~/Lecture/git/repo_dir
khlee$ git reset --soft HEAD^
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new_file

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
```



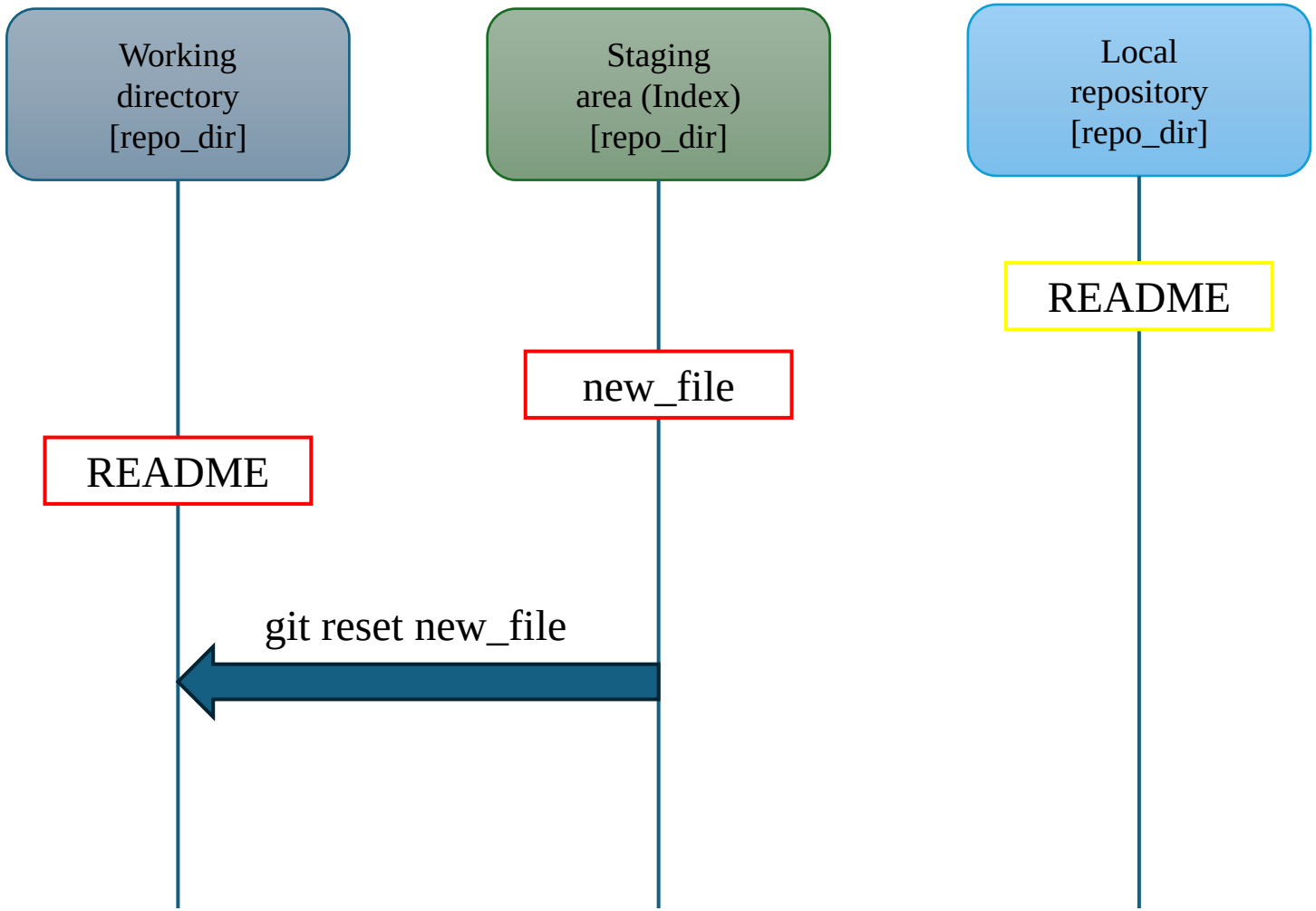
How to use Git? (13/16)

- **reset**

```
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new_file

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
```

```
~/Lecture/git/repo_dir
khlee$ git reset new_file
Unstaged changes after reset:
M       README
```

```
~/Lecture/git/repo_dir
khlee$ git status
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

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

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

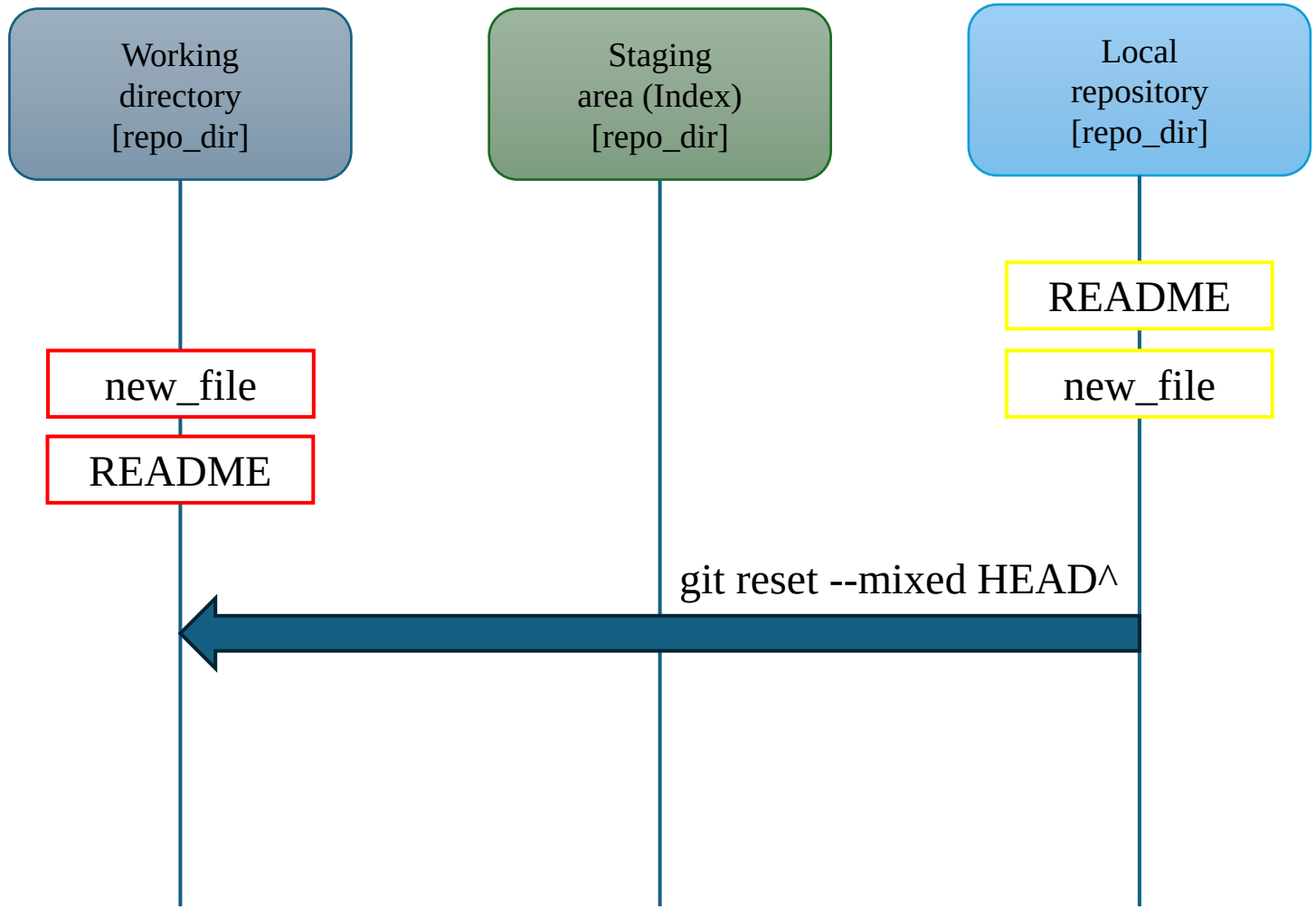


How to use Git? (14/16)

- **reset**

```
~/Lecture/git/repo_dir
khlee$ git reset --soft HEAD^
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new_file

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
```

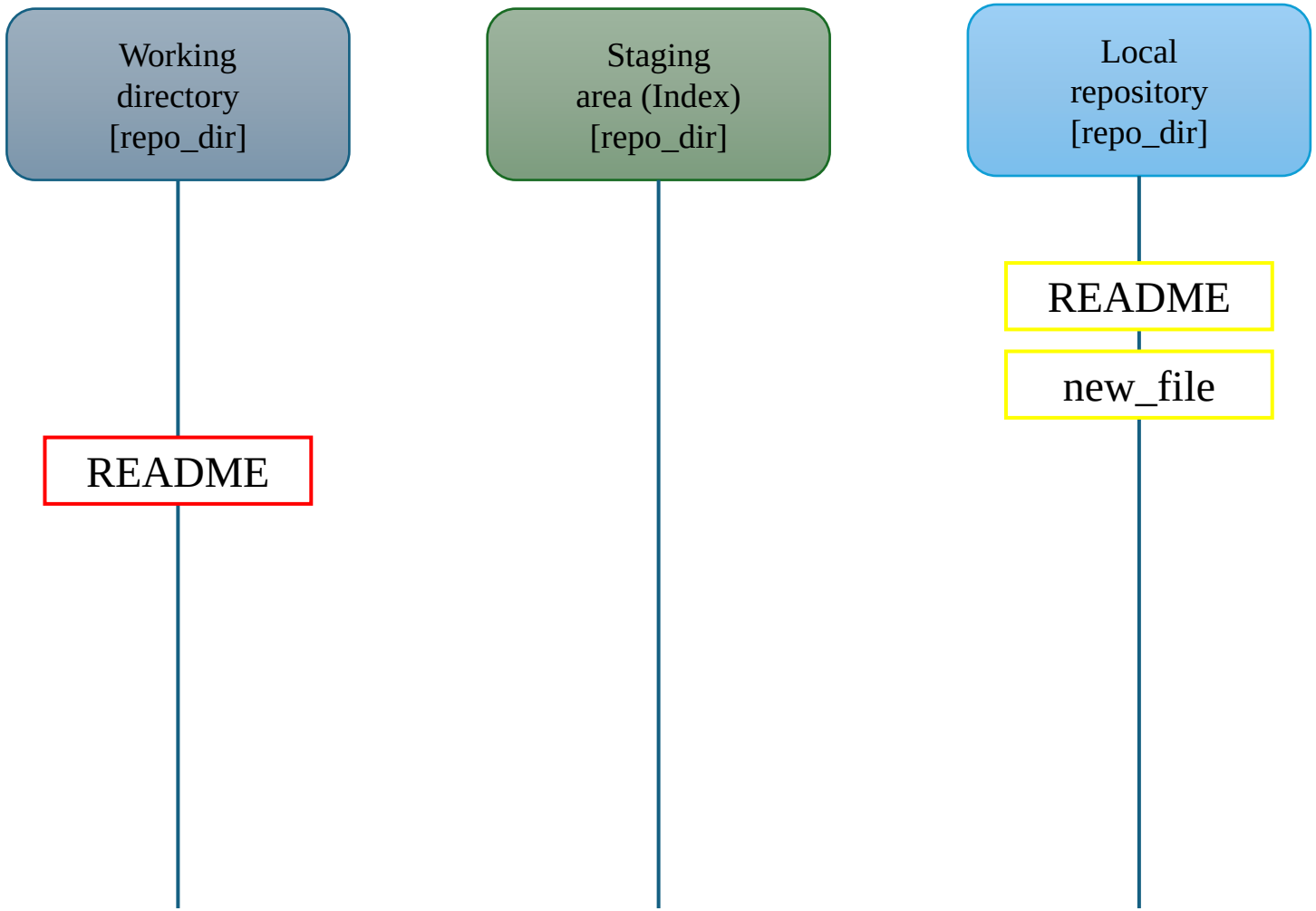
```
~/Lecture/git/repo_dir
khlee$ git reset --mixed HEAD^
Unstaged changes after reset:
M       README
~/Lecture/git/repo_dir
khlee$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README

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

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



How to use Git? (15/16)

- **reset**

```
khlee$ ls
new_file  README
~/Lecture/git/repo_dir
khlee$ git reset --hard HEAD^
HEAD is now at 9ab8de8 Create README
~/Lecture/git/repo_dir
khlee$ git status
On branch master
nothing to commit, working tree clean
~/Lecture/git/repo_dir
khlee$ ls
README
```



How to use Git? (16/16)

- **revert**

```
~/Lecture/git/repo_dir
khlee$ git log
commit 66ee59fe3284794e9fce97a16288a4e8db8ce07d (HEAD -> master)
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 18:45:51 2025 +0000

    new_file

commit 9ab8de8bbd0ea1dad3b597f36c65ac9590aa8221
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 17:18:27 2025 +0000

    Create README
```

```
khlee$ git revert 66ee59fe3284794e9fce97a16288a4e8db8ce07d
[master 68260b9] Revert "new_file"
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 new_file
~/Lecture/git/repo_dir
khlee$ git log
commit 68260b96b1ba882924f1059ed3dd34d6ade0b523 (HEAD -> master)
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 18:48:36 2025 +0000

    Revert "new_file"

    This reverts commit 66ee59fe3284794e9fce97a16288a4e8db8ce07d.

commit 66ee59fe3284794e9fce97a16288a4e8db8ce07d
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 18:45:51 2025 +0000

    new_file

commit 9ab8de8bbd0ea1dad3b597f36c65ac9590aa8221
Author: khlee <kh-lee@dankook.ac.kr>
Date:   Tue Mar 25 17:18:27 2025 +0000

    Create README
~/Lecture/git/repo_dir
khlee$ git status
On branch master
nothing to commit, working tree clean
~/Lecture/git/repo_dir
khlee$ ls
README
```

Practice (1/6)

- **Practice 1**
  - “my_repo”의 이름을 가진 로컬 저장소(local repository)를 생성하세요.
  - “README”의 이름을 가진 파일을 생성하세요.
    - Command: touch README
  - “Hello Git”의 내용을 README 파일에 추가하세요.
    - Command: echo “Hello Git” >> README

Practice (2/6)

- **Practice 2**
  - README을 Staging area(index)에 추가하세요.
  - README을 Local repository에 저장하세요.
    - 메시지는 “Add README”
  - “Change README”의 내용을 README 파일에 추가하세요.
  - Git 상태를 확인하세요.
  - 기존 내용과 바뀐 내용을 비교하세요.

Practice (3/6)

- **Practice 3**
  - “Change README”의 내용을 README 파일에 추가하세요.
  - Git 상태를 확인하세요.
  - 기존 내용과 바뀐 내용을 비교하세요.

Practice (4/6)

- **Practice 4**
  - “Change README”의 내용을 README 파일에 추가하세요.
  - Git 상태를 확인하세요.
  - 기존 내용과 바뀐 내용을 비교하세요.

Practice (5/6)

- **Practice 5**
  - 수정된 README를 Staging area에 추가하세요.
  - git 상태를 확인하세요.
  - README 내용을 다시 수정 해야해서 Staging을 취소하세요.
  - git 상태를 확인하세요.

Practice (6/6)

- **Practice 6**
  - README 파일을 Local repository에 저장하세요.
    - 메시지는 “first”
  - Local repository의 README 파일을 Staging area로 내렸다가 Local repository에 다시 저장하세요.
    - 메시지는 “seconds”
  - git의 로그를 확인하세요.

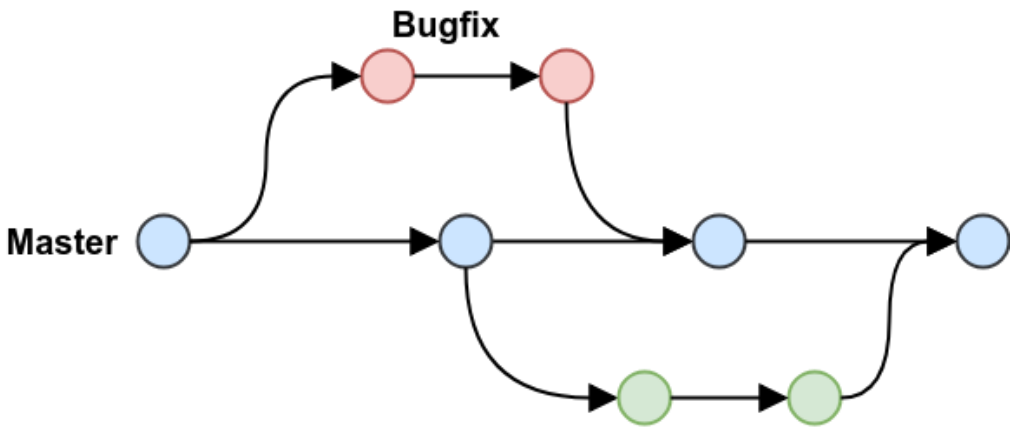
Practice (6/6)

- **Practice 6**
  - README 파일을 Local repository에 저장하세요.
    - 메시지는 “first”
  - Local repository의 README 파일을 Staging area로 내렸다가 Local repository에 다시 저장하세요.
    - 메시지는 “seconds”
  - git의 로그를 확인하세요.

What is a Branch?

- **Branch**
  - Ability to separate workflows in Git
  - Enables independent development of separate tasks.

- **Why use?**
  - To safely separate **feature additions** or **bug fixes** from the main code
  - To allow **multiple people** to work simultaneously
  - To keep the **main code (master/main) safe** even in case of mistakes



Ref: google image

How to use Git? (1/10)

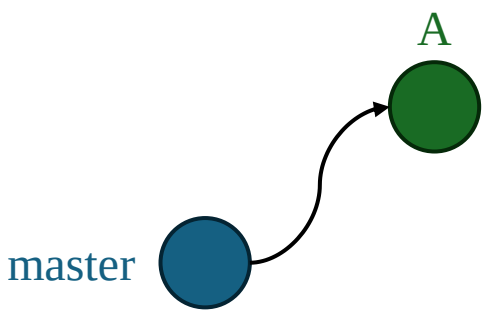
- **Branch**
  - List, create, or delete branches

```
~/Lecture/git/repo_dir
khlee$ git branch
* master
```
List

```
~/Lecture/git/repo_dir
khlee$ git branch A
~/Lecture/git/repo_dir
khlee$ git branch
  A
* master
```
Create

```
~/Lecture/git/repo_dir
khlee$ git branch -d A
Deleted branch A (was 68260b9).
~/Lecture/git/repo_dir
khlee$ git branch
* master
```
Delete

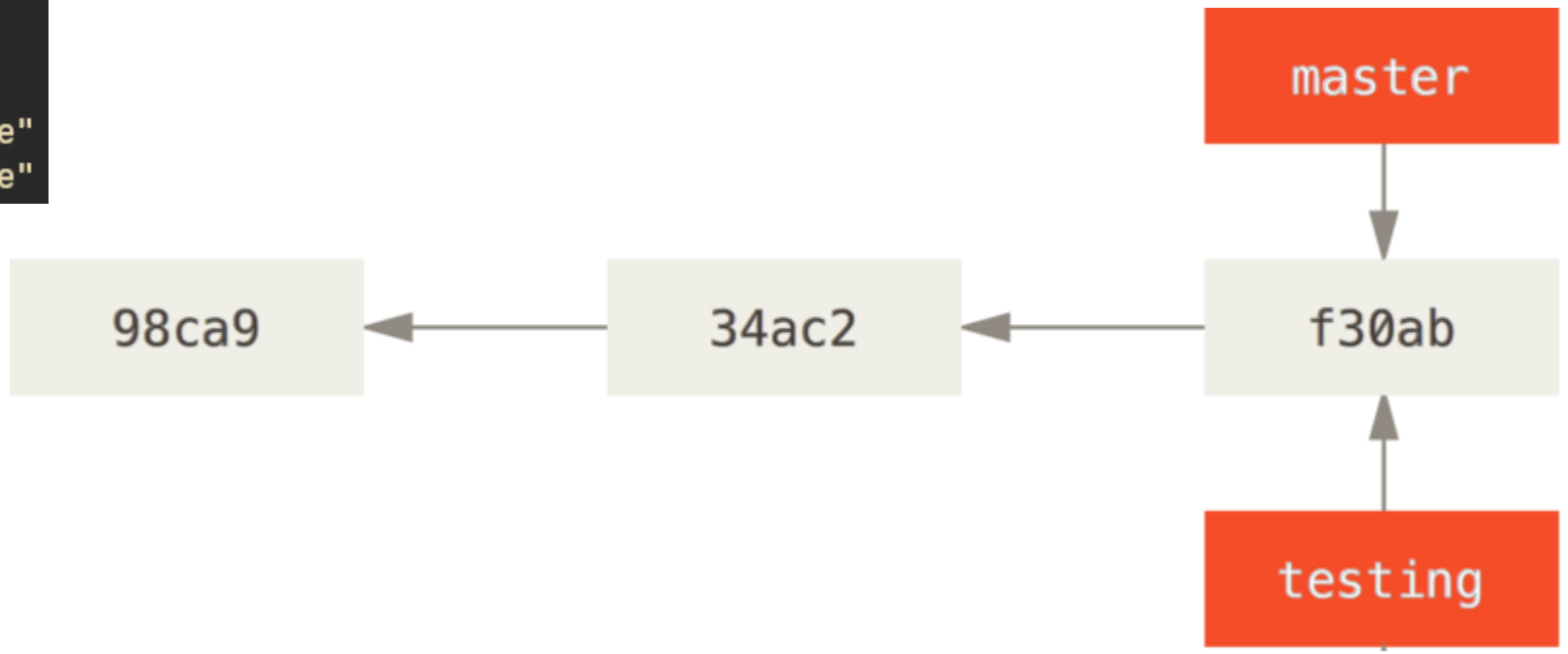
```
khlee$ git branch -v
* master 68260b9 Revert "new_file"
```



How to use Git? (2/10)

- **Branch and HEAD**
  - **branch -v**

```
~/Lecture/git/repo_dir
khlee$ git branch
* master
~/Lecture/git/repo_dir
khlee$ git branch testing
~/Lecture/git/repo_dir
khlee$ git branch -v
* master  68260b9 Revert "new_file"
  testing 68260b9 Revert "new_file"
```

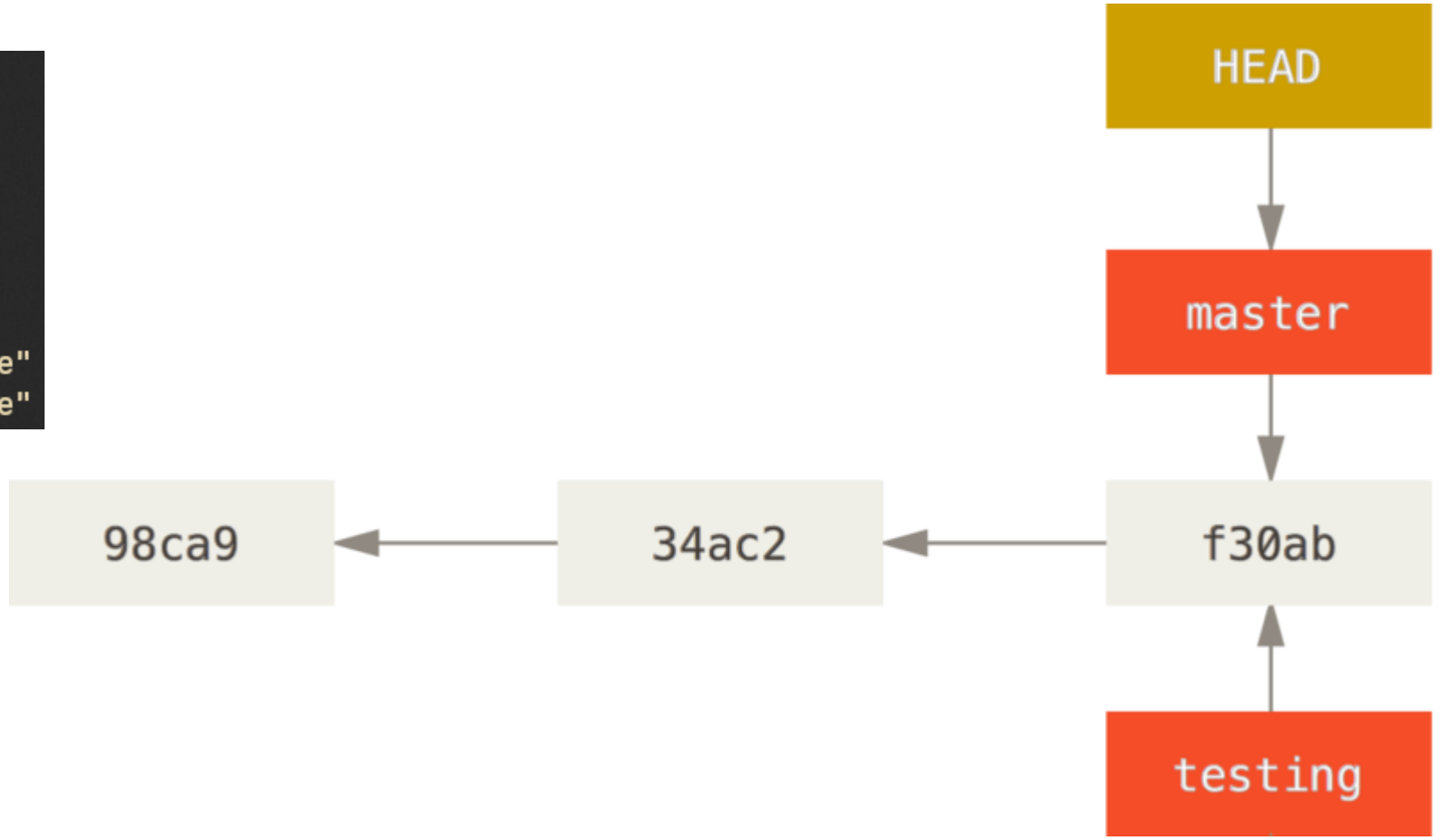


Ref: git-scm.com

How to use Git? (3/10)

- **Branch and HEAD**
  - **branch -v**

```
~/Lecture/git/repo_dir
khlee$ git branch
* master
~/Lecture/git/repo_dir
khlee$ git branch testing
~/Lecture/git/repo_dir
khlee$ git branch -v
* master  68260b9 Revert "new_file"
  testing 68260b9 Revert "new_file"
```



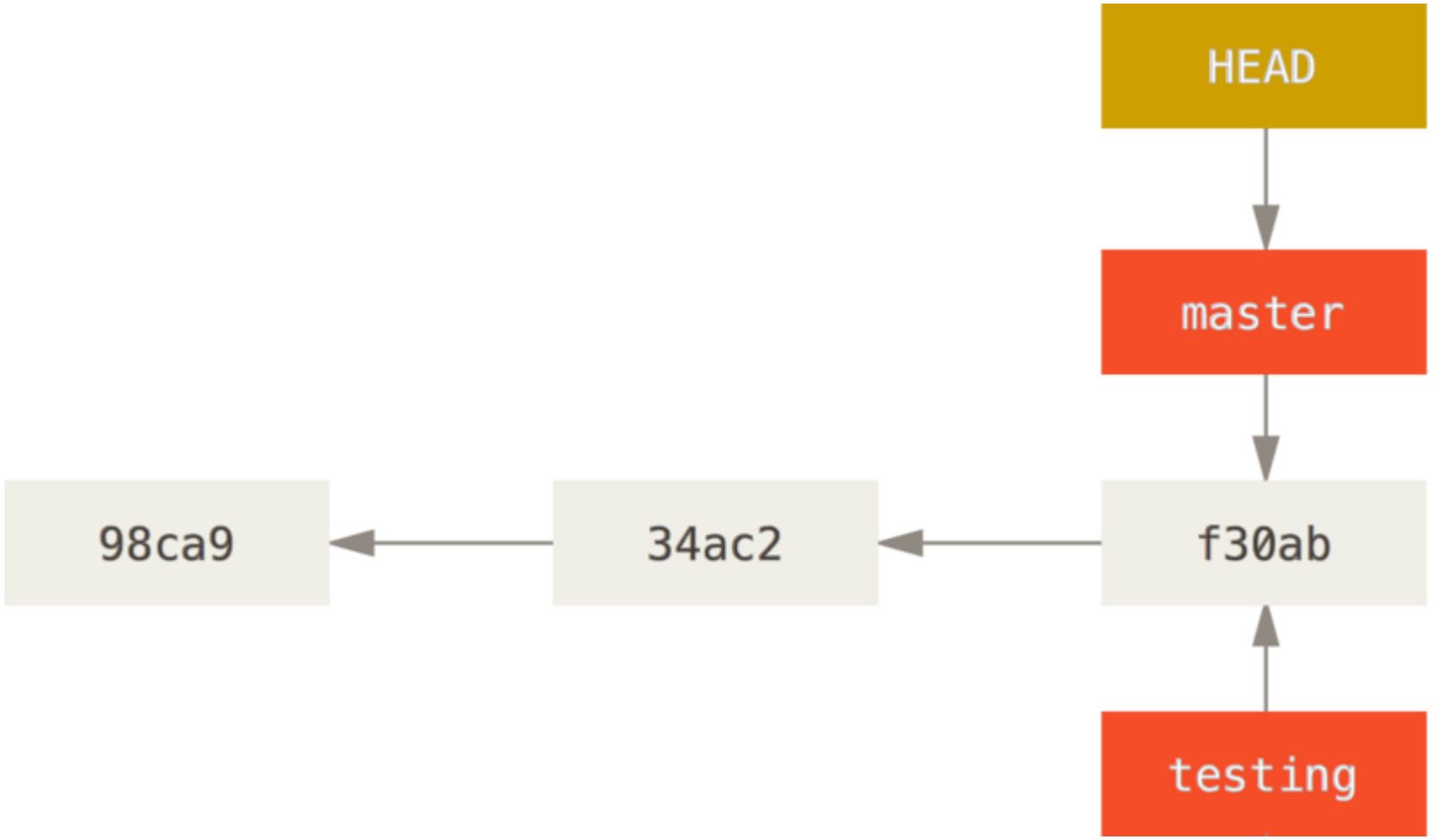
Ref: git-scm.com

How to use Git? (4/10)

- **Branch and HEAD**
  - checkout or switch

```
~/Lecture/git/repo_dir
khlee$ git checkout testing
M       README
Switched to branch 'testing'
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
```

```
~/Lecture/git/repo_dir
khlee$ git switch master
M       README
Switched to branch 'master'
~/Lecture/git/repo_dir
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ git switch testing
M       README
Switched to branch 'testing'
```



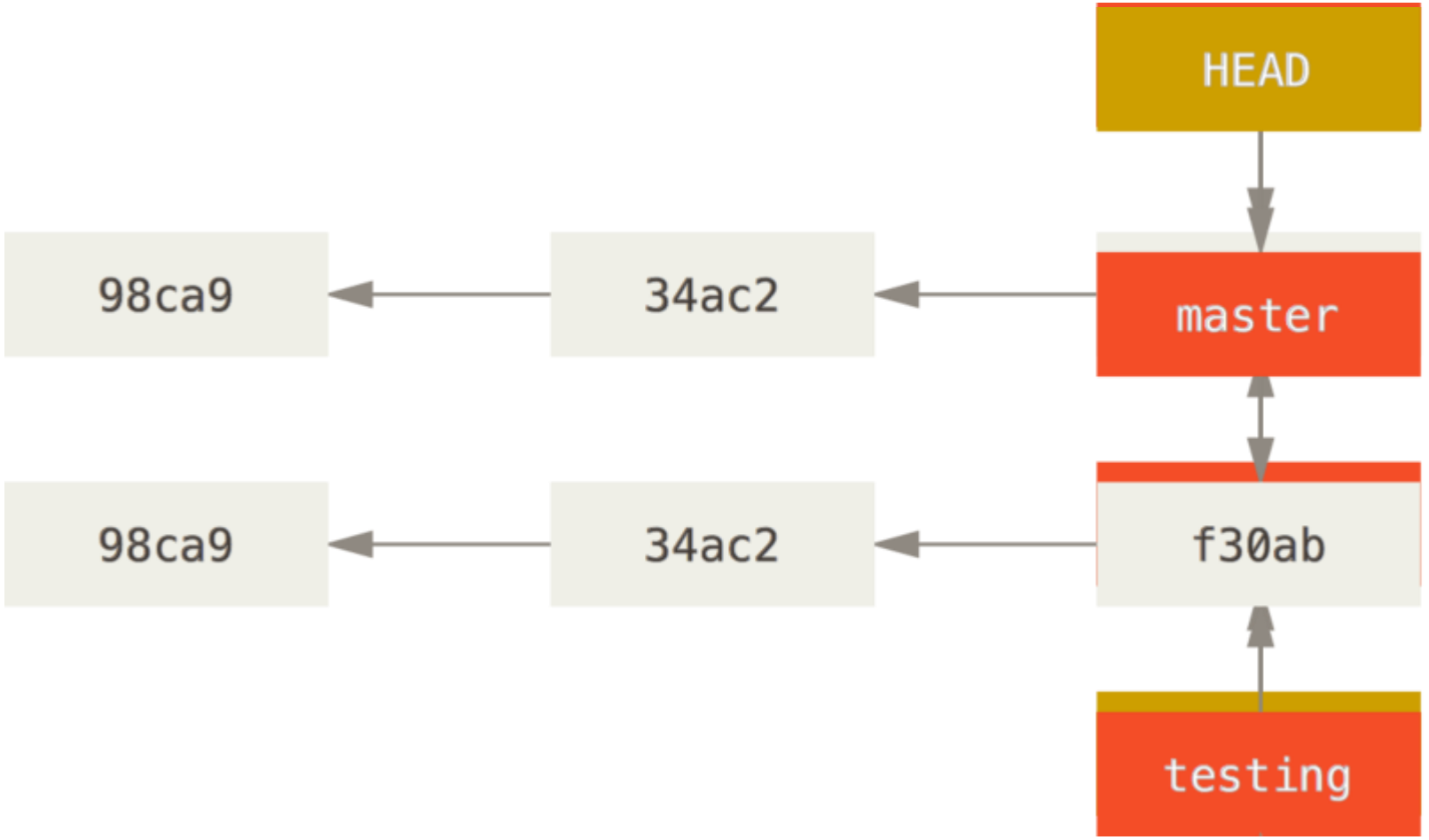
Ref: git-scm.com

How to use Git? (5/10)

- **Branch and HEAD**
  - checkout or switch

```
~/Lecture/git/repo_dir
khlee$ git checkout testing
M       README
Switched to branch 'testing'
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
```

```
~/Lecture/git/repo_dir
khlee$ git switch master
M       README
Switched to branch 'master'
~/Lecture/git/repo_dir
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ git switch testing
M       README
Switched to branch 'testing'
```



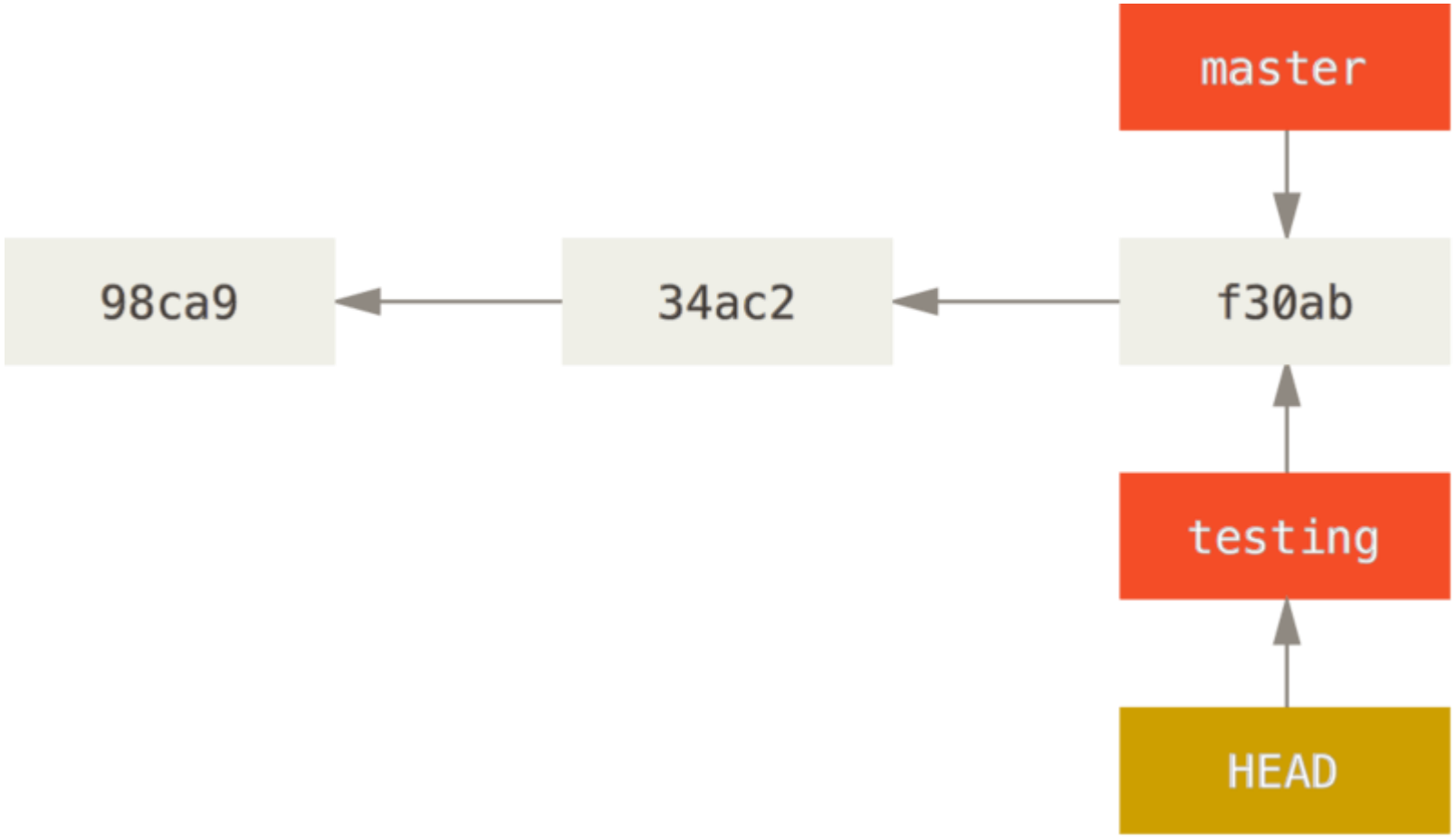
Ref: git-scm.com

How to use Git? (6/10)

- **Branch and HEAD**
  - checkout or switch

```
~/Lecture/git/repo_dir
khlee$ git checkout testing
M       README
Switched to branch 'testing'
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
```

```
~/Lecture/git/repo_dir
khlee$ git switch master
M       README
Switched to branch 'master'
~/Lecture/git/repo_dir
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ git switch testing
M       README
Switched to branch 'testing'
```



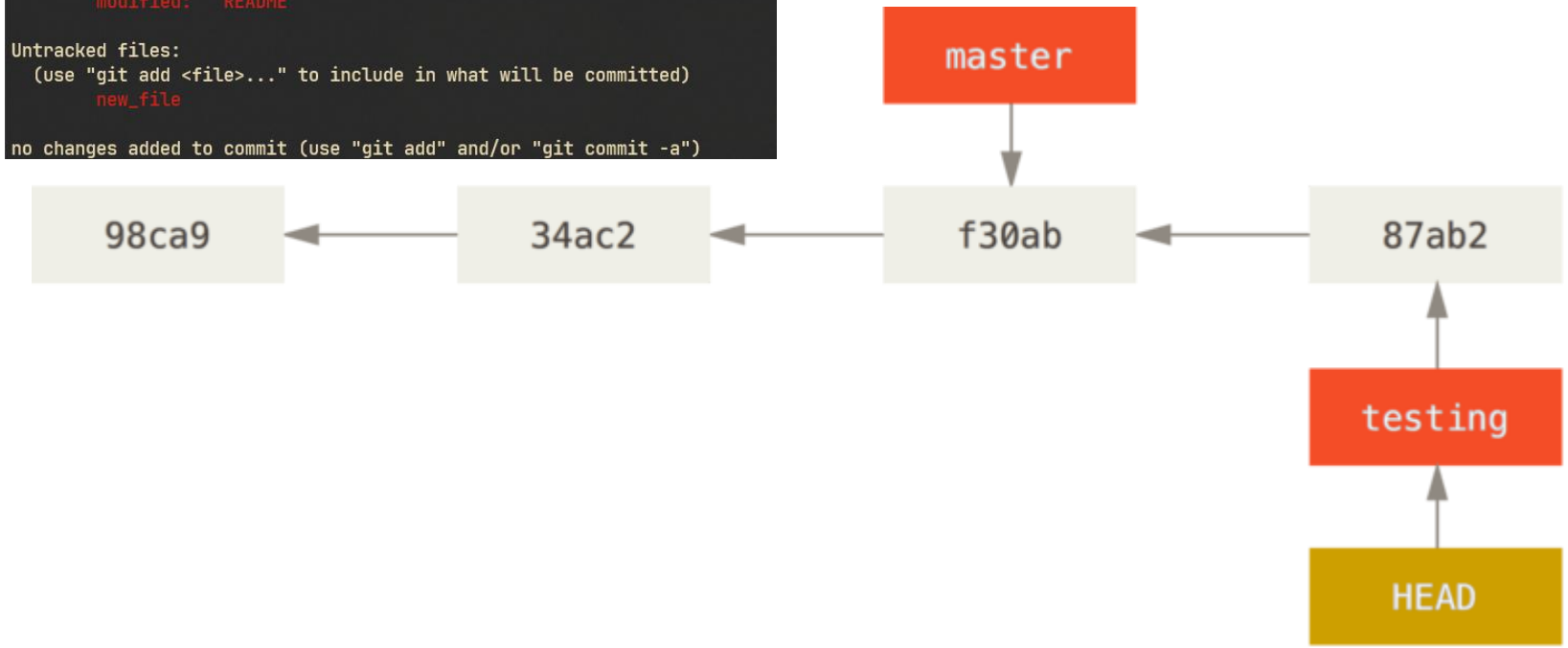
Ref: git-scm.com

How to use Git? (7/10)

- **Branch and HEAD**
  - testing branch new file commit

```
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
~/Lecture/git/repo_dir
khlee$ ls
new_file  README
~/Lecture/git/repo_dir
khlee$ touch test_file
~/Lecture/git/repo_dir
khlee$ ls
new_file  README  test_file
~/Lecture/git/repo_dir
khlee$ git add test_file
~/Lecture/git/repo_dir
khlee$ git commit -m "testing branch"
[testing c26ae6b] testing branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_file
```

```
~/Lecture/git/repo_dir
khlee$ ls
new_file  README  test_file
~/Lecture/git/repo_dir
khlee$ git status
On branch testing
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

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

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



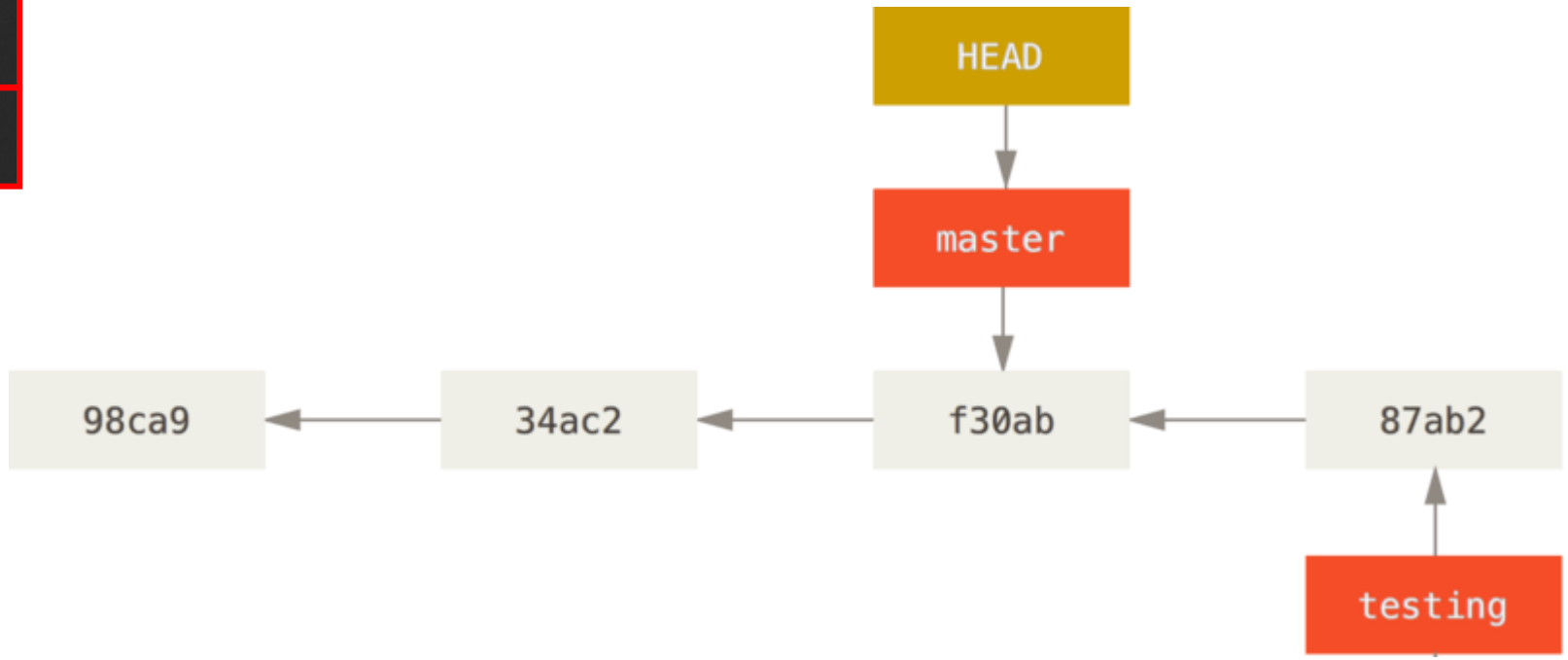
Ref: git-scm.com

How to use Git? (8/10)

- **Branch and HEAD**
  - switch master branch

```
~/Lecture/git/repo_dir
khlee$ git switch master
M       README
Switched to branch 'master'
~/Lecture/git/repo_dir
khlee$ ls
new_file  README
```

```
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
~/Lecture/git/repo_dir
khlee$ ls
new_file  README
~/Lecture/git/repo_dir
khlee$ touch test_file
~/Lecture/git/repo_dir
khlee$ ls
new_file  README  test_file
```



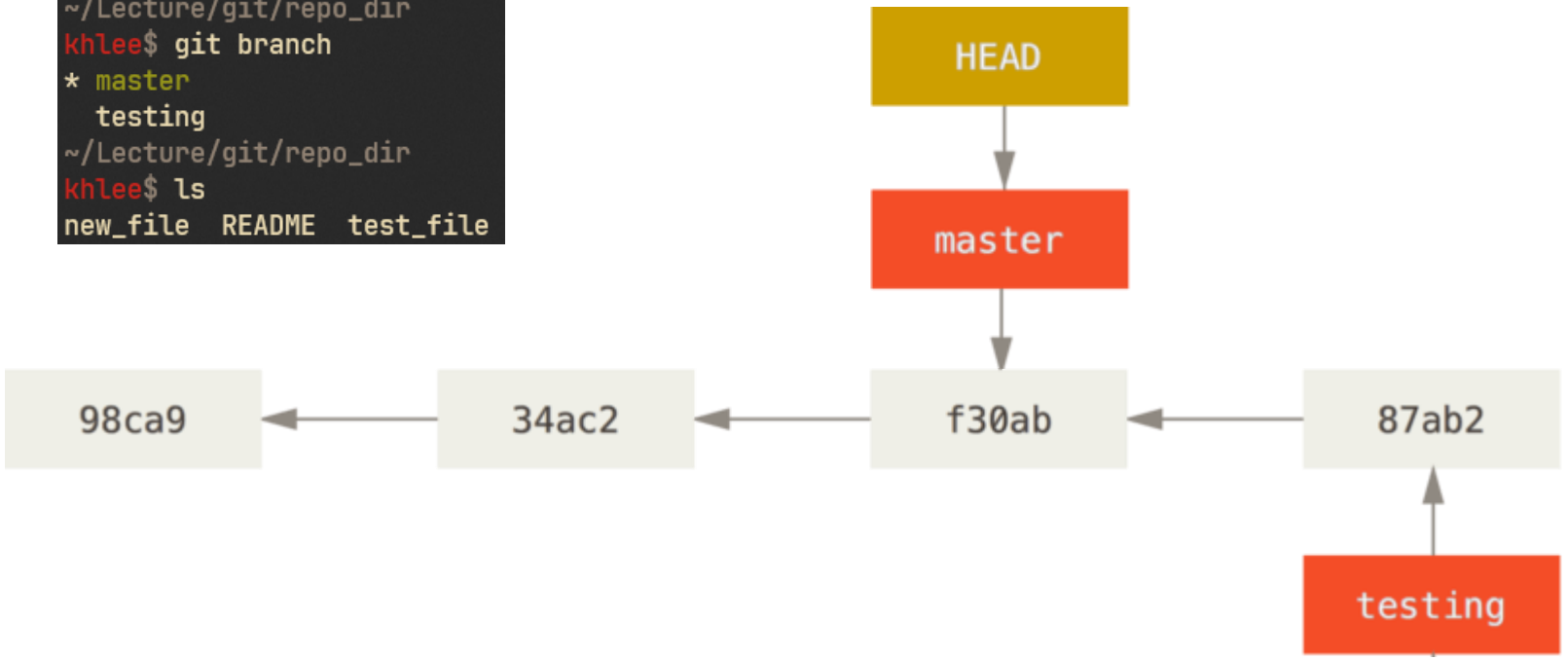
Ref: git-scm.com

How to use Git? (9/10)

- **Merge**
  - Fast-forward
    - Master branch가 뒤처졌을 때 앞으로만 이동
  - 3-way merge
    - 3개의 branch가 서로 다른 작업을 했을 때 commit 3개를 비교해 새 merge commit 생성

```
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ git merge testing
Updating 68260b9..c26ae6b
Fast-forward
 test_file | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_file
```

```
~/Lecture/git/repo_dir
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ ls
new_file  README  test_file
```



Ref: git-scm.com

How to use Git? (10/10)

- **Merge Conflict**

```
~/Lecture/git/repo_dir
khlee$ git branch
* master
  testing
~/Lecture/git/repo_dir
khlee$ echo "Master branch" > test_file
~/Lecture/git/repo_dir
khlee$ cat test_file
Master branch
~/Lecture/git/repo_dir
khlee$ git add .
~/Lecture/git/repo_dir
khlee$ git commit -m "master"
[master c2942c2] master
 3 files changed, 2 insertions(+)
 create mode 100644 new_file
```

Merge

```
~/Lecture/git/repo_dir
khlee$ git branch
  master
* testing
~/Lecture/git/repo_dir
khlee$ echo "Testing branch" > test_file
~/Lecture/git/repo_dir
khlee$ cat test_file
Testing branch
~/Lecture/git/repo_dir
khlee$ git add .
~/Lecture/git/repo_dir
khlee$ git commit -m "testing"
[testing d9b8830] testing
 1 file changed, 1 insertion(+)
```

```
~/Lecture/git/repo_dir
khlee$ git merge testing
Auto-merging test_file
CONFLICT (content): Merge conflict in test_file
Automatic merge failed; fix conflicts and then commit the result.
```

Summary

- **Version Control System(VCS)**

- **Git**

Assignment 2

1. **Practice 1~6**

2. **Explain 3-way merge and example.**

- **제출 요건**
  - Include student ID and date (using whoami, date)
  - 기한: 일주일
  - 양식: 포맷 없음, 장수 제한 없음, pdf (파일명: 오픈소스SW기초_{분반}_{이름}_{학번}.pdf)
  - 제출: e-Campus => 과제

Aknowledgement

- 본 교재는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 ‘SW중심대학사업’ 지원을 받아 제작 되었습니다.

- 본 결과물의 내용을 전재할 수 없으며, 인용(재사용)할 때에는 반드시 과학기술정보통신부와 정보통신기획평가원이 지원한 ‘SW중심대학’의 결과물이라는 출처를 밝혀야 합니다.