Git을 사용한 버전 관리

Git HEAD pointer

/* elice */

HEAD

파일들을 staging시키고 commit하여 git repository에
저장하는 방법까지 알아보았습니다.

그렇다면 git은 어떤 방식으로 commit된 내용으로
버젼을 효율적으로 관리 할 수 있을까요?

파일의 영역 라이프 사이클

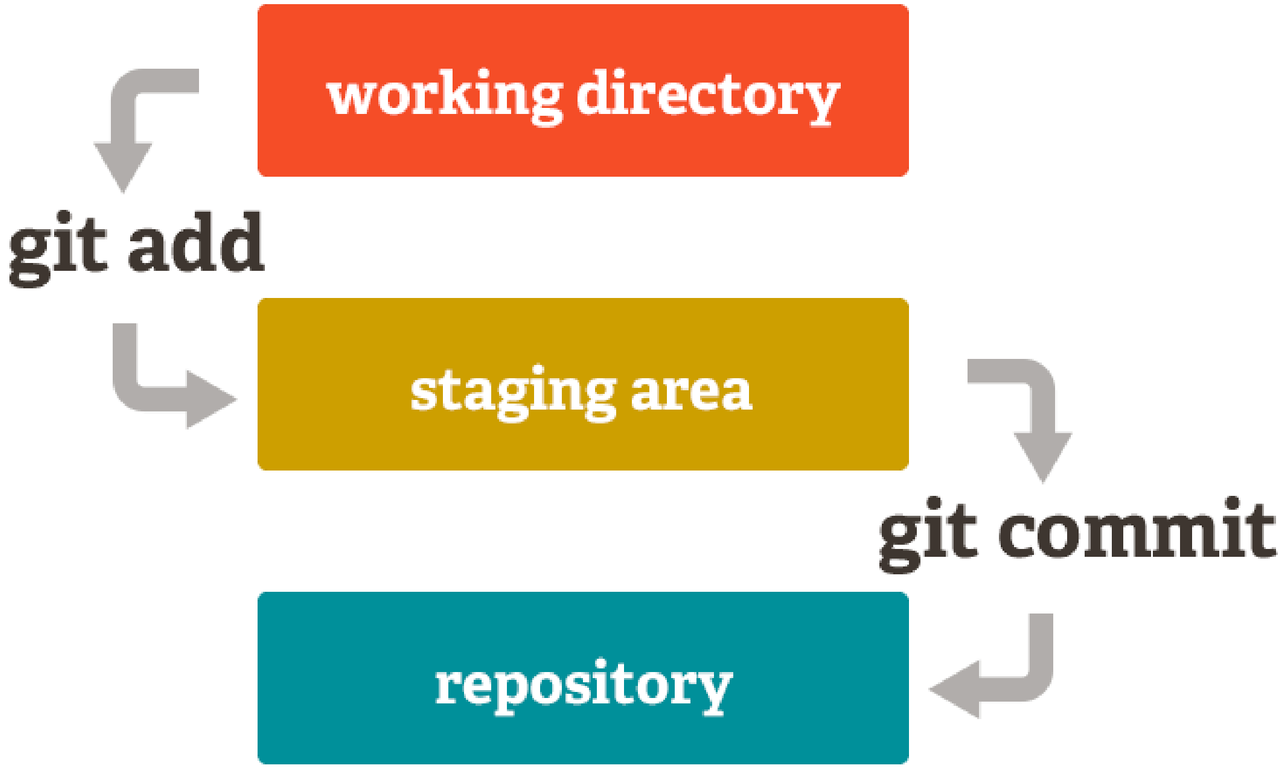


그림 git-scm.com

git 영역의 구성

HEAD

Index

working directory

파일의 워크플로

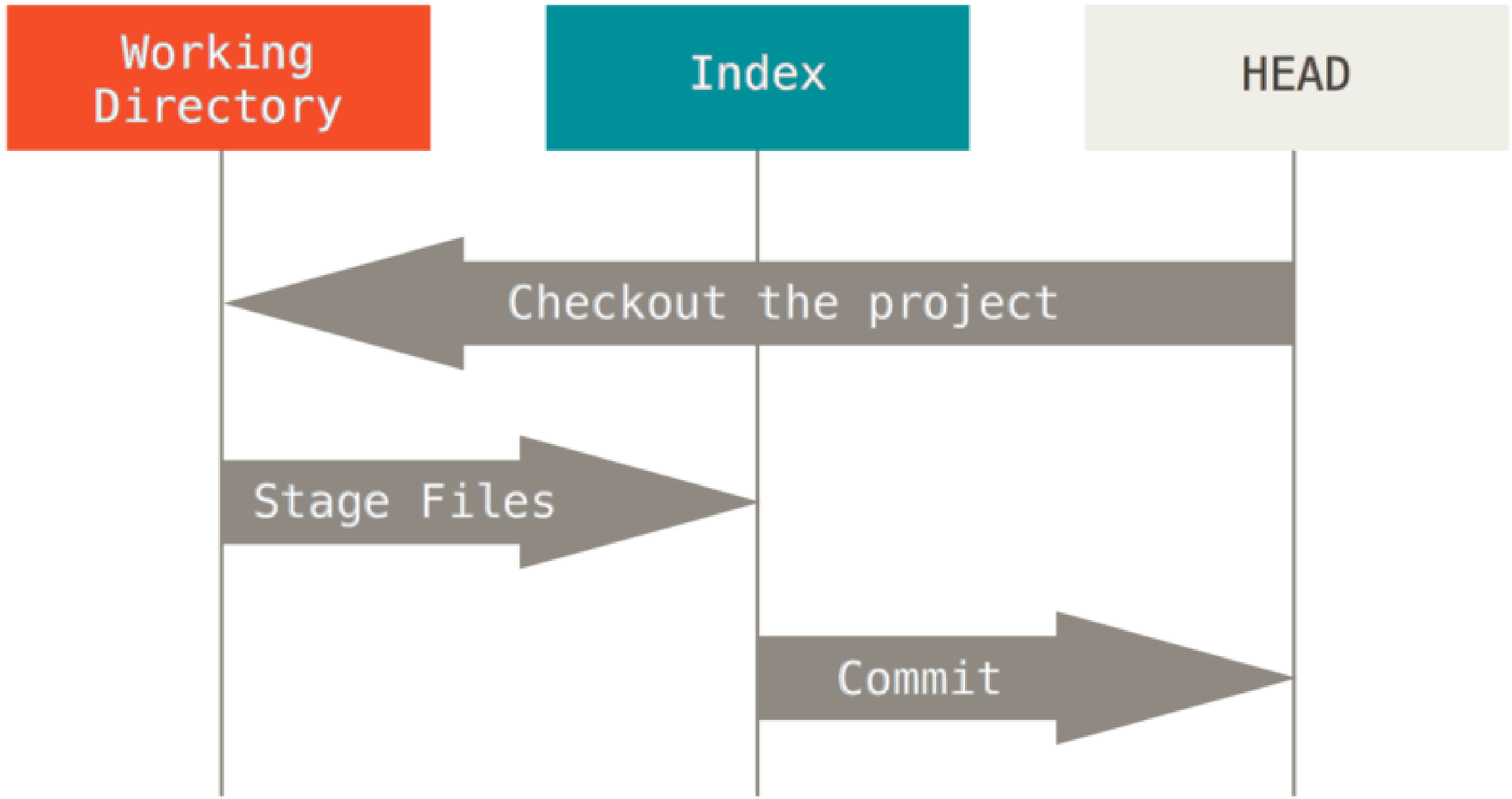


그림 git-scm.com

파일의 워크플로

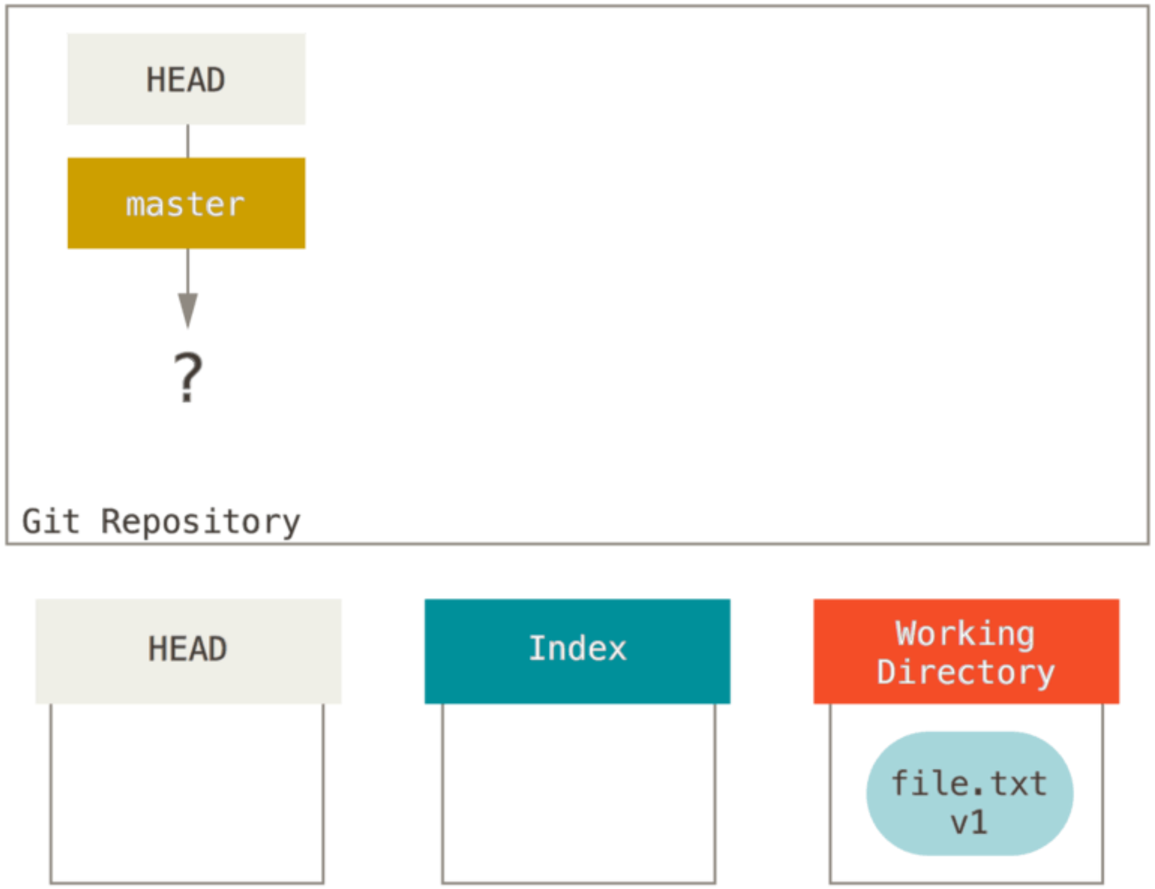


그림 git-scm.com

Index

`git add` 로 Staging 되어 있는 데이터들

HEAD

마지막 커밋을 가리키는 snapshot

snapshot

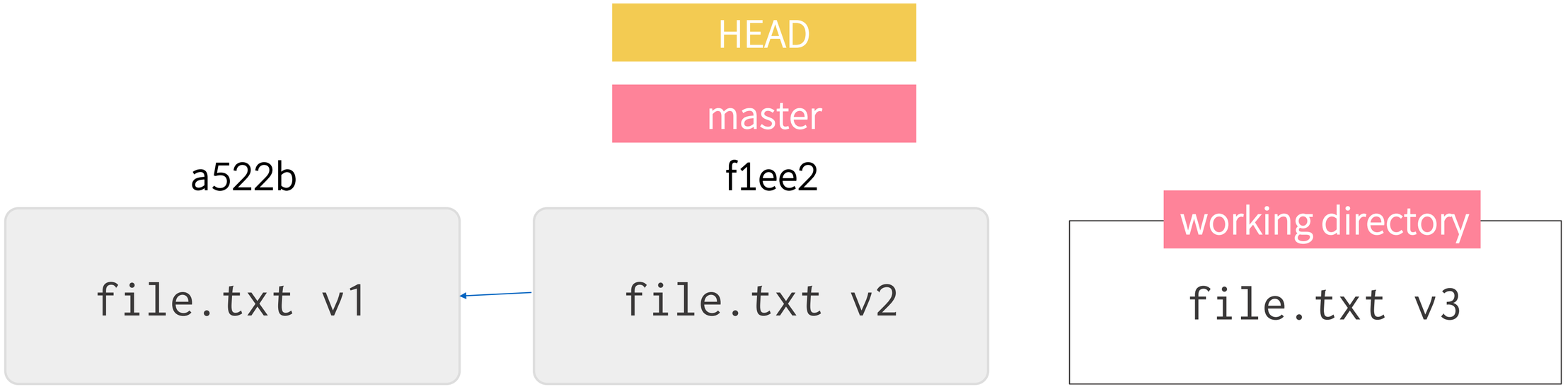
staging 되어진 데이터들(Index)을
commit 명령으로 영구히 저장한 것을
snapshot 이라 부릅니다.

HEAD 역시 snapshot이며 후에 배울 branch에서 현재 우리가 위치해 있는 branch를 가리키는 포인터의 역할을 같이 수행하게 됩니다.

HEAD를 이용한 버전 관리

`git reset --<option> HEAD~` 을 이용하여

HEAD를 전의 snapshot으로 이동시킬 수 있습니다.



--soft(default)

`git reset --soft HEAD~` 을 이용하여
예전 버전으로 돌아가 commit을 수정할 수 있습니다.

--soft(default)

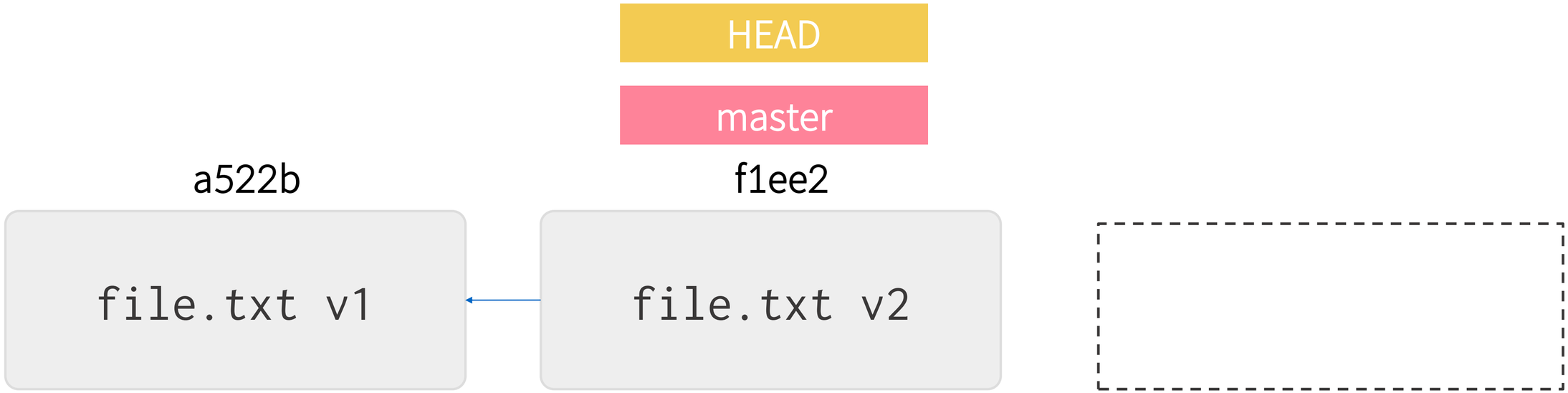
```
$ git log --pretty=oneline
eba263 (HEAD->master) add change
d3c3d init commit
```

```
$ git reset --soft HEAD~
$ git log --pretty=oneline
d3c3d (HEAD->master) init commit
$ git commit -m “modified message”
$ git log --pretty=oneline
39ae2 (HEAD->master) modifies me..
d3c3d init commit
```

--hard

`git reset --hard HEAD~` 을 이용하면

working directory에 존재하는 파일도 사라집니다.




/*elice*/
contact@elice.io