Git을 사용한 버전 관리

동기화 오류 모음집(pull편)

/* elice */

동기화와 관련된 오류들

pull 오류

첫 번째 pull 오류

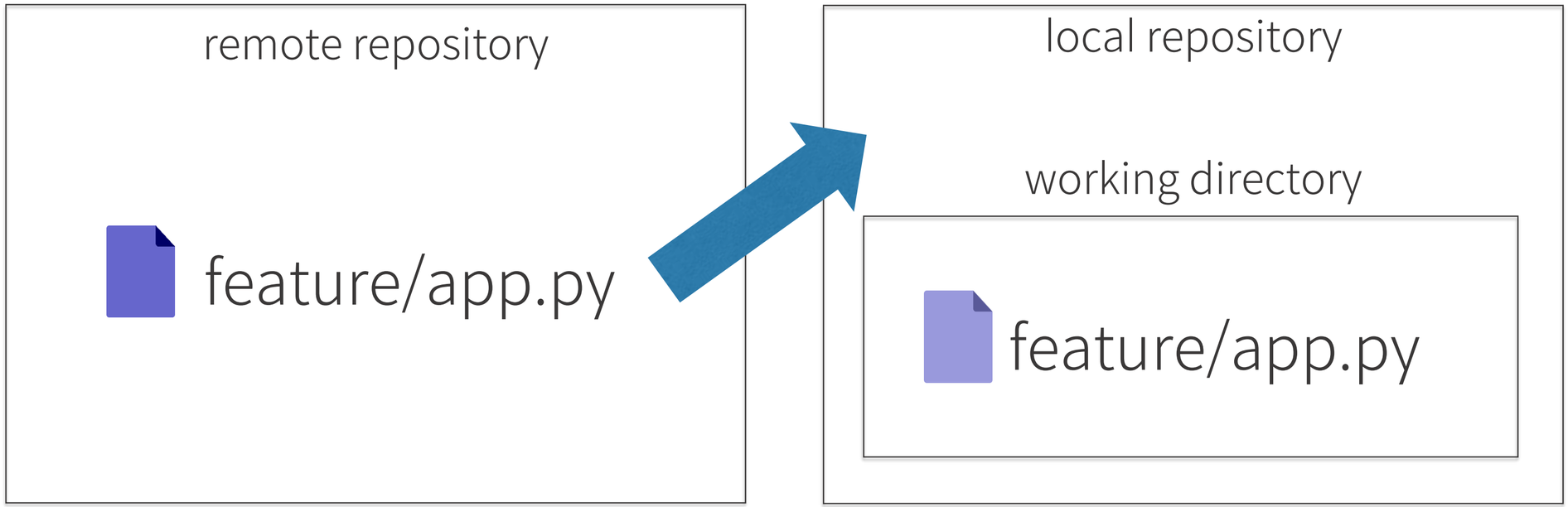
```
$ git pull
error: The following untracked working tree files would be overwritten by merge:
        <file_name>
Please move or remove them before you merge.
Aborting
```

두 번째 pull 오류

```
$ git pull
error: Your local changes to the following files would be overwritten by merge:
        <file_name>
Please move or remove them before you merge.
Aborting
```

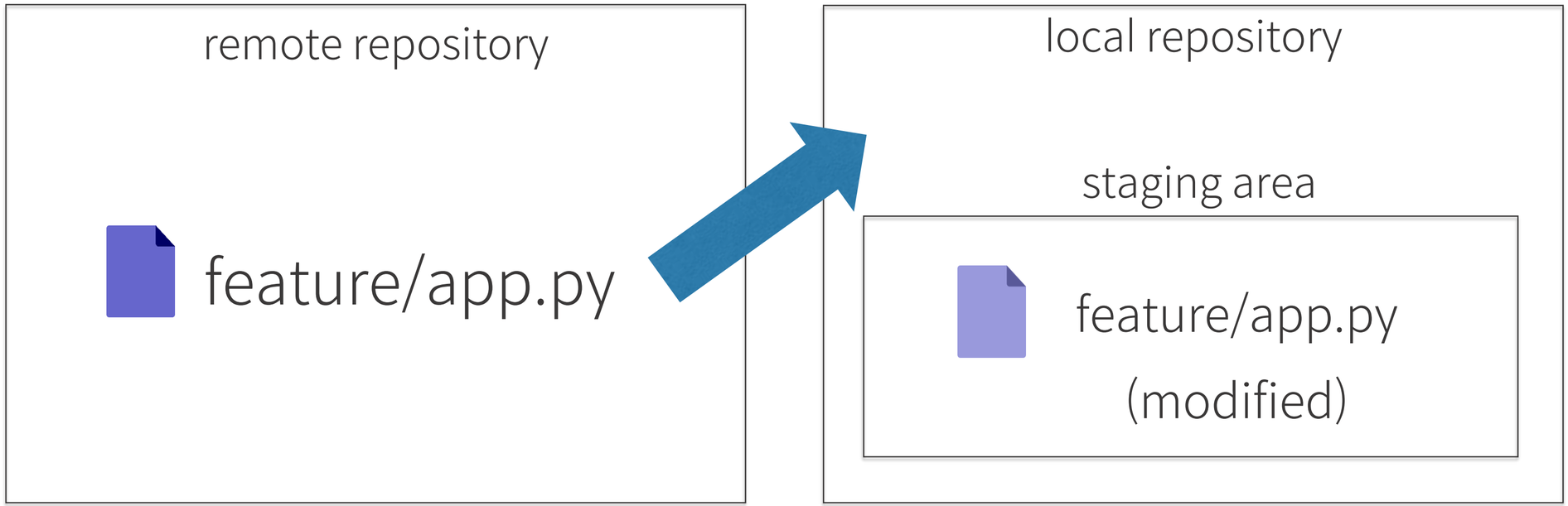
pull 오류 (1)

첫 번째는 pull을 했을 때 받아온 저장소의 파일이 디렉토리 내의 untracked 파일 중 같은 이름을 가질 때 발생하는 오류입니다.



pull 오류 (2)

두 번째는 pull을 했을 때 받아온 저장소의 파일이 로컬 git 저장소에서 수정된 파일과 같은 이름을 가질 때 발생하는 오류입니다.



pull 오류 해결(1)

1. 충돌되는 파일을 삭제 또는 다른 임시 디렉토리로 이동시킨다.
2. 워킹 디렉토리를 초기화 시킨다.

git clean -f -d -x

오류가 발생되는 파일이 한 두개 라면 충돌되는 파일을 다른 디렉토리로 이동시키면 됩니다. 그러나..

충돌이 발생되는 파일이 너무 많아 일일이

이동시킬 수 없을 경우가 발생할 수 있습니다.

git clean -f -d -x

```
$ git pull origin master
error: The following untracked working tree files would be overwritten by merge:
        .gradle/4.6/fileChanges/last-build.bin
        .gradle/vcsworkingDirs/gc.properties
        app/build/generated/not_namespaced_r_class_soruces1/…
        app/build/generated/not_namespaced_r_class_soruces2/…
        app/build/generated/not_namespaced_r_class_soruces3/…
  ...
Please move or remove them before you merge.
Aborting
```

git clean -f -d -x

`git clean -f -d -x` 명령어는

git이 추적 중이지 않은 워킹 디렉토리에 존재하는 파일들과

.gitignore로 git의 관리를 받지 않는 파일들까지

모두 깨끗하게 지워버리는 명령어 입니다.

git clean -f -d -x

따라서 수정 중이던 내용 중 필요한 부분이 있다면

커밋으로 수정된 상태를 저장하거나

아예 수정하고 있는 파일을 다른 디렉토리에 옮겨 저장하는 것이

pull 오류를 해결하는 가장 확실한 방법입니다.

pull 오류 해결(2)

1. git stash를 사용하여 작업한 내용을 커밋하지 않고 임시저장 한다.
2. 충돌되는 파일을 삭제 또는 다른 임시 디렉토리로 이동시킨다.

git stash

다음과 같이 작업된 내용이 아직 커밋되지 않고 있다고 가정하겠습니다.

```
$ git status
Changes to be committed:
      (use "git reset HEAD <file>..." to unstage)
      modified: app.py

Changes not staged for commit:
      (use “git add <file>…” to update what will be committed)
      (use “git checkout -- <file>…” to discard changes in working directory)
      modified: lib/myproject.py
```

git stash

`git stash` 명령어를 이용하여 내용을 임시저장 해보겠습니다.

```
$ git stash
Saved working directory and index state
     "WIP on master: 049d078 added the index file"
HEAD is now at 049d078 added the index file

(To restore them type "git stash apply")
```

git stash

working directory가 비워진 것을 확인 할 수 있습니다.

```
$ git status
On branch master
nothing to commit, working directory clean
```

git stash

`git stash list` 로 임시저장한 내역을 확인 할 수 있습니다.

```
$ git status list
stash@{0}: WIP on master: 049d078 added the index file
```

git stash

pull을 완료하고 `git stash apply` 로

가장 최근에 저장한 stash를 불러 올 수 있습니다.

```
$ git stash apply
On branch master
Changes not staged for commit:
        (use “git add <file>…” to update what will be committed)
        (use “git checkout -- <file>…” to discard changes in working directory)
        modified: app.py
        modified: lib/myproject.py
```

pull 오류 해결

한 가지 주의할 점은 stash는 staging되었던 상태의 파일들을
불러올 때 다시 staging시켜 주지 않습니다.
staging을 유지시켜 주기 위해서는 `--index` 옵션을 주면 됩니다.


/*elice*/
contact@elice.io