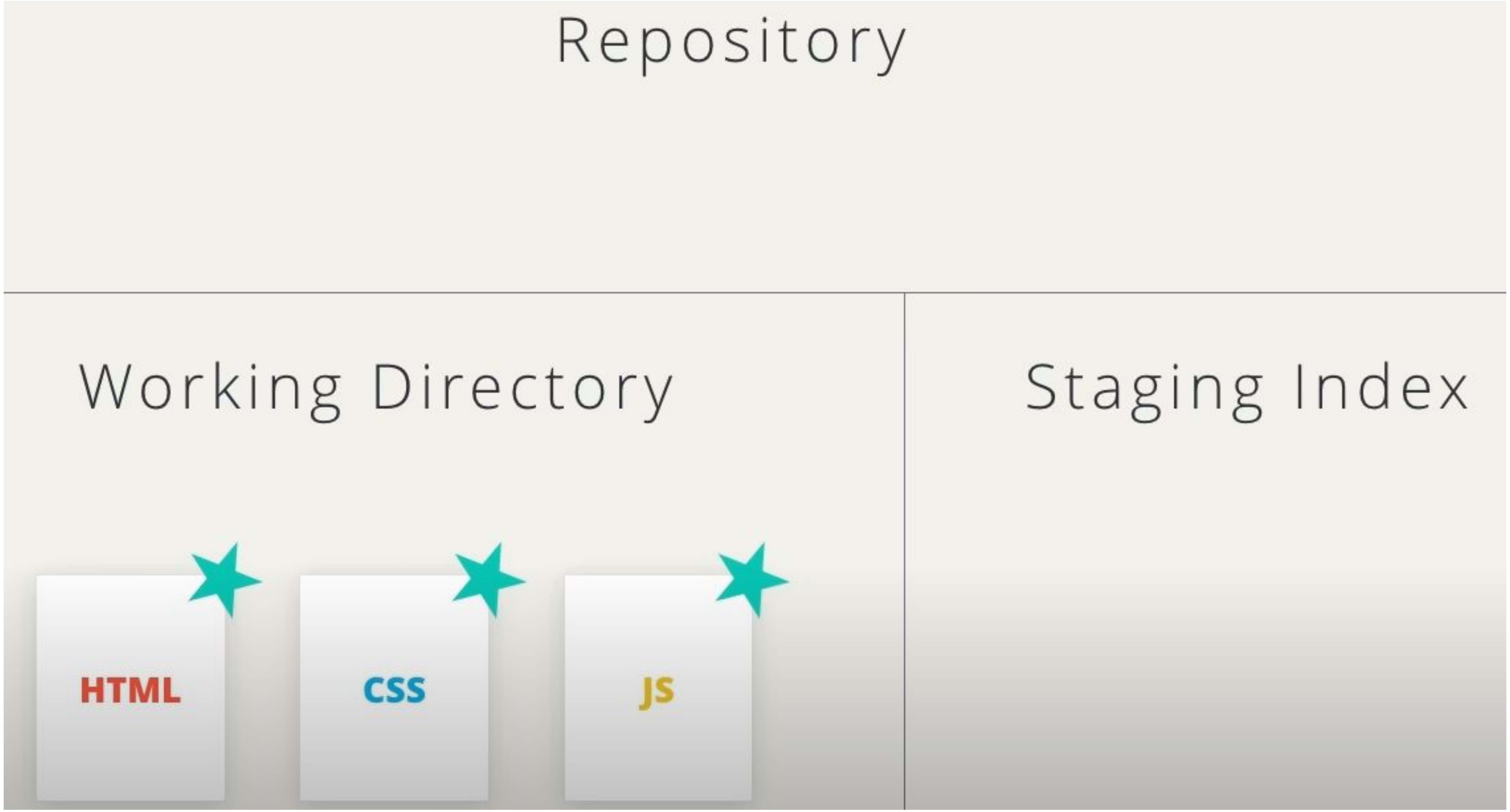
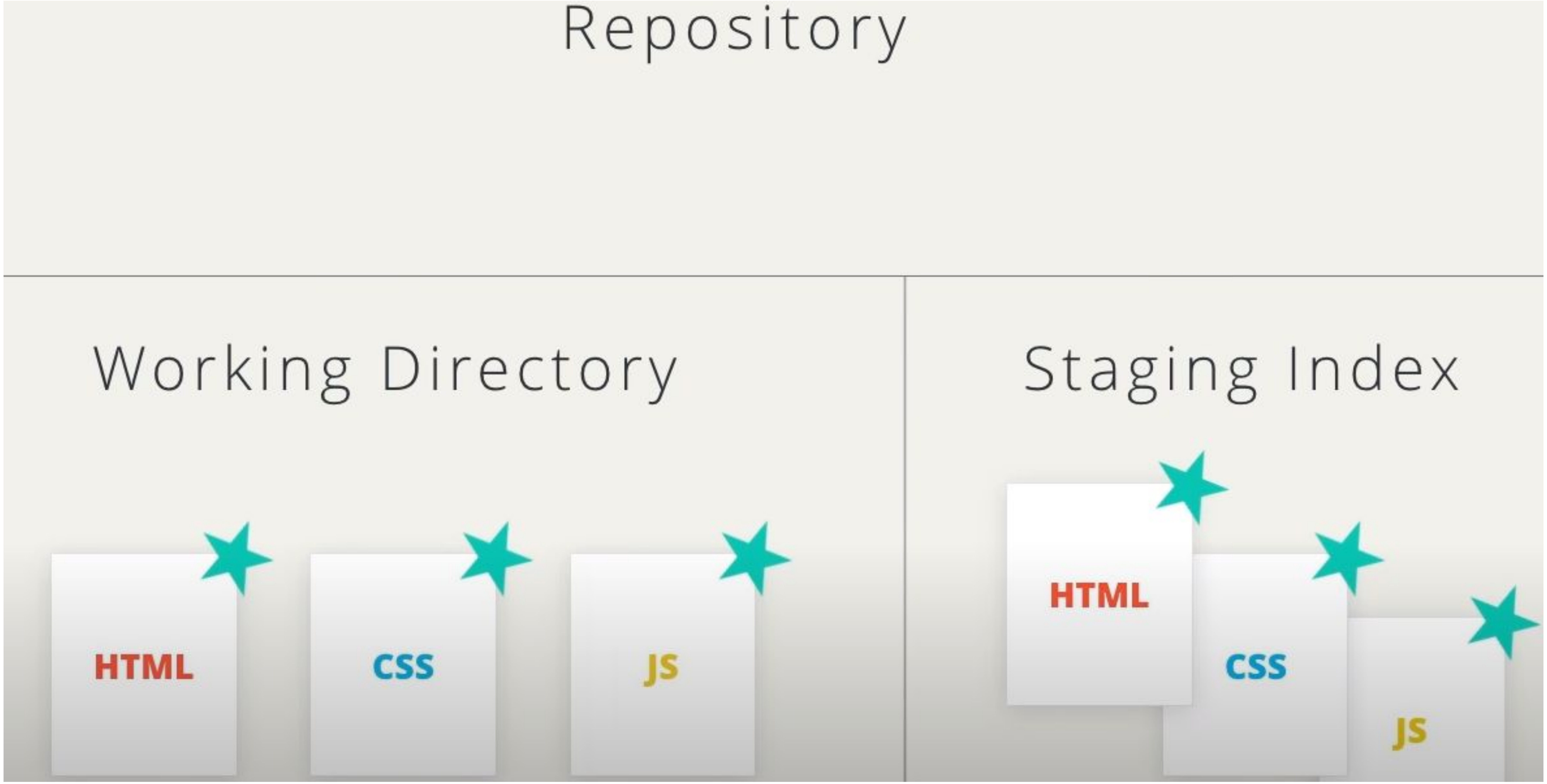
Git for Web Apps 1

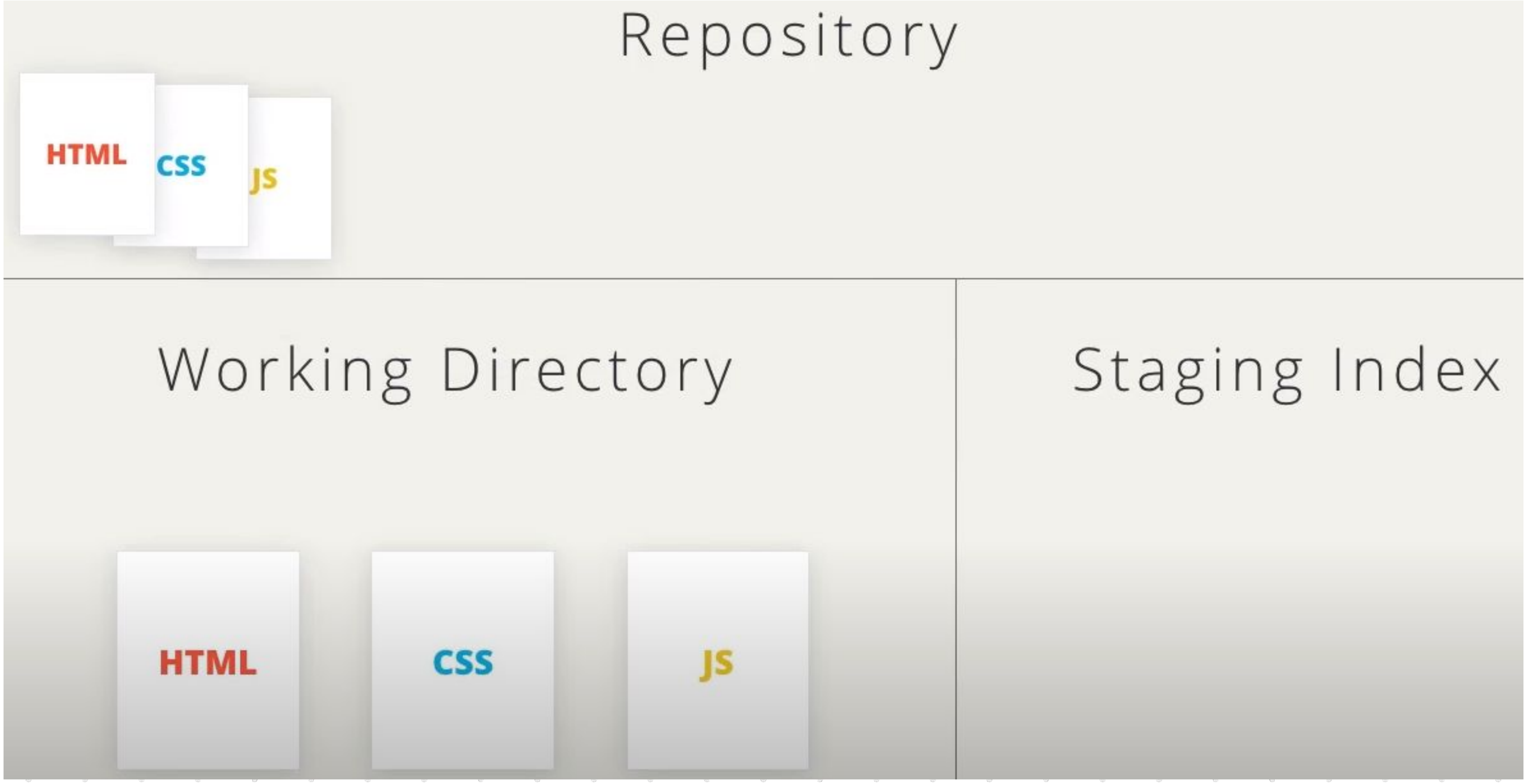
Git scenario **1: change** HTML & CSS & JS



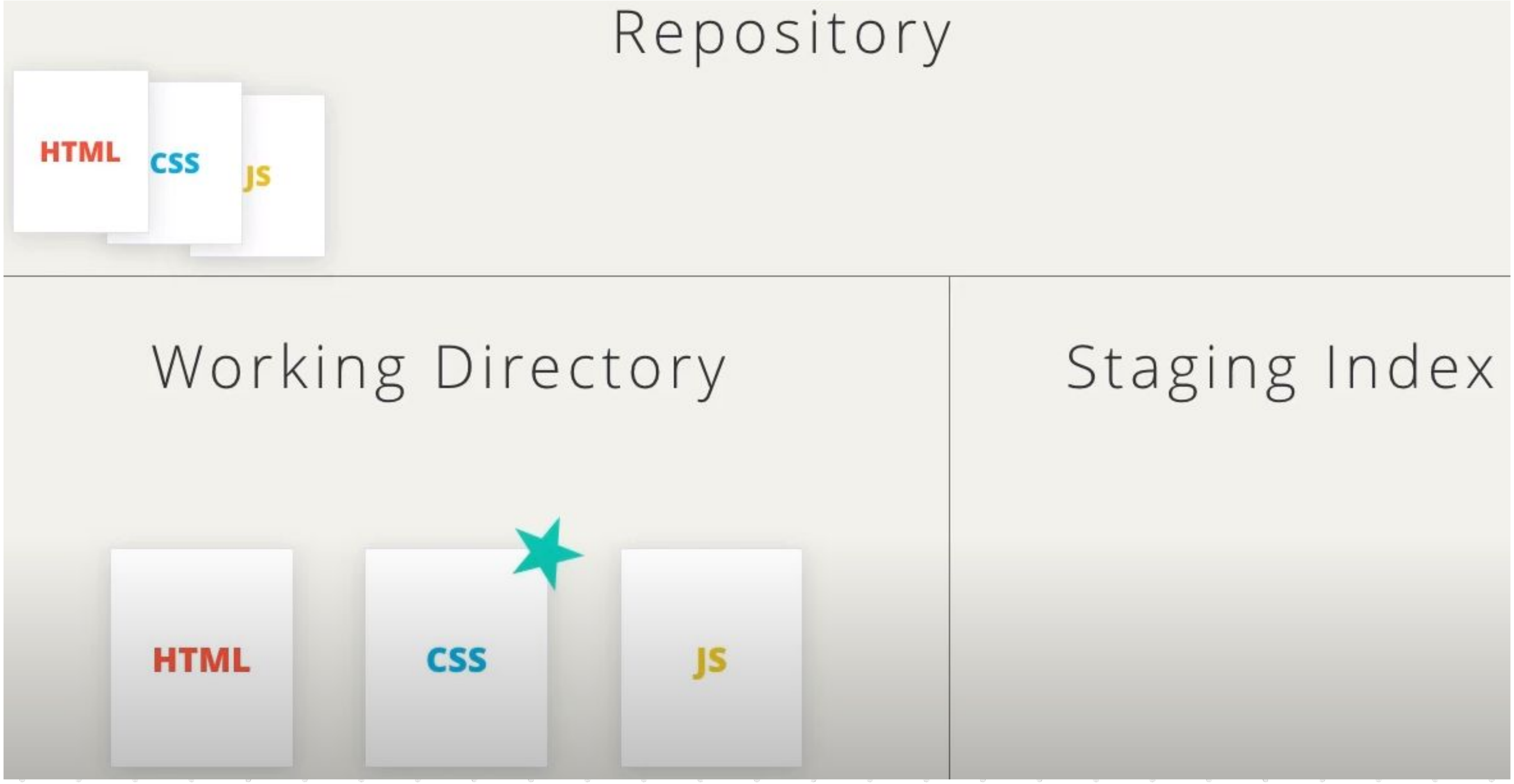
Git scenario **1: staging** changes



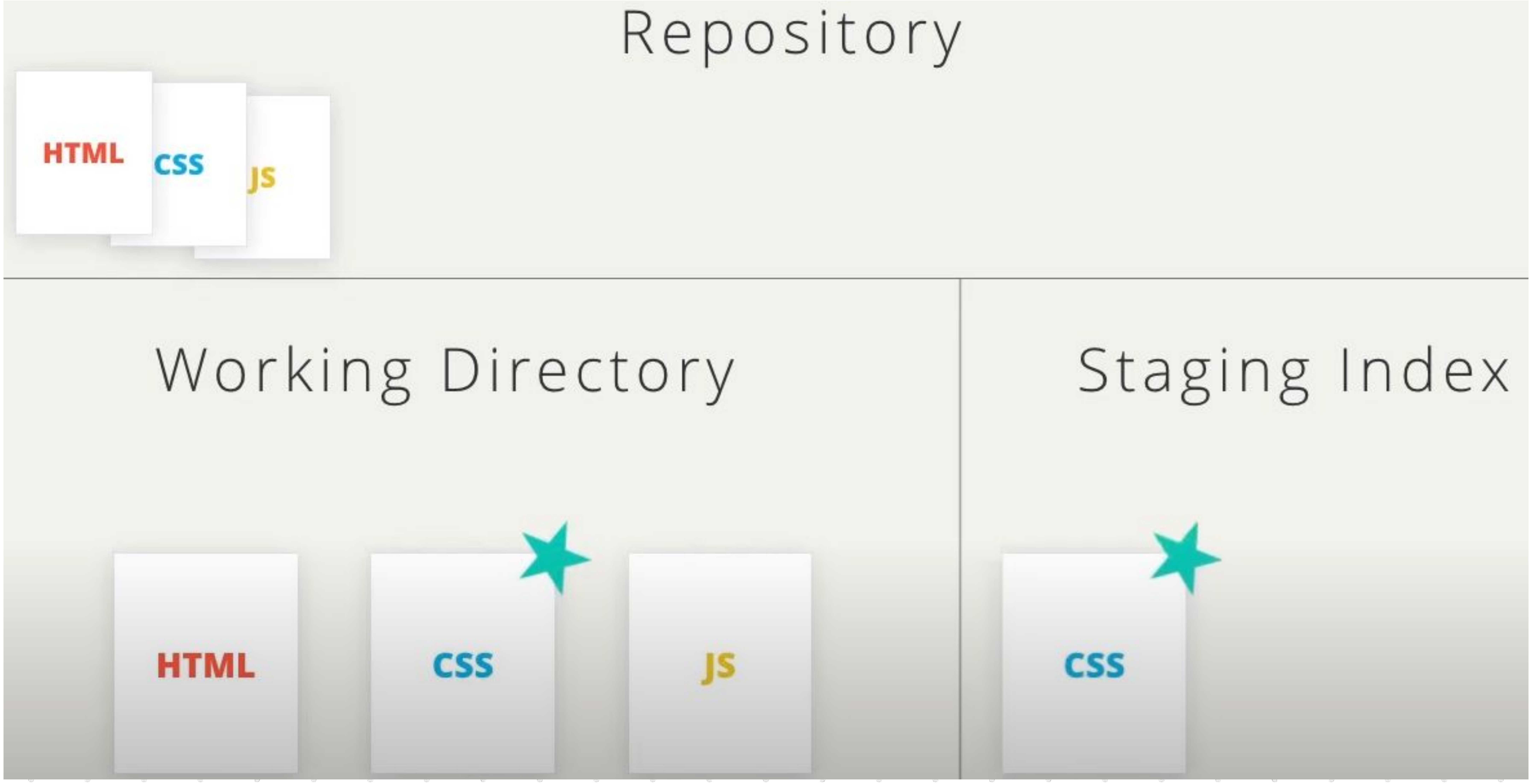
Git scenario **1: commit** changes to repo



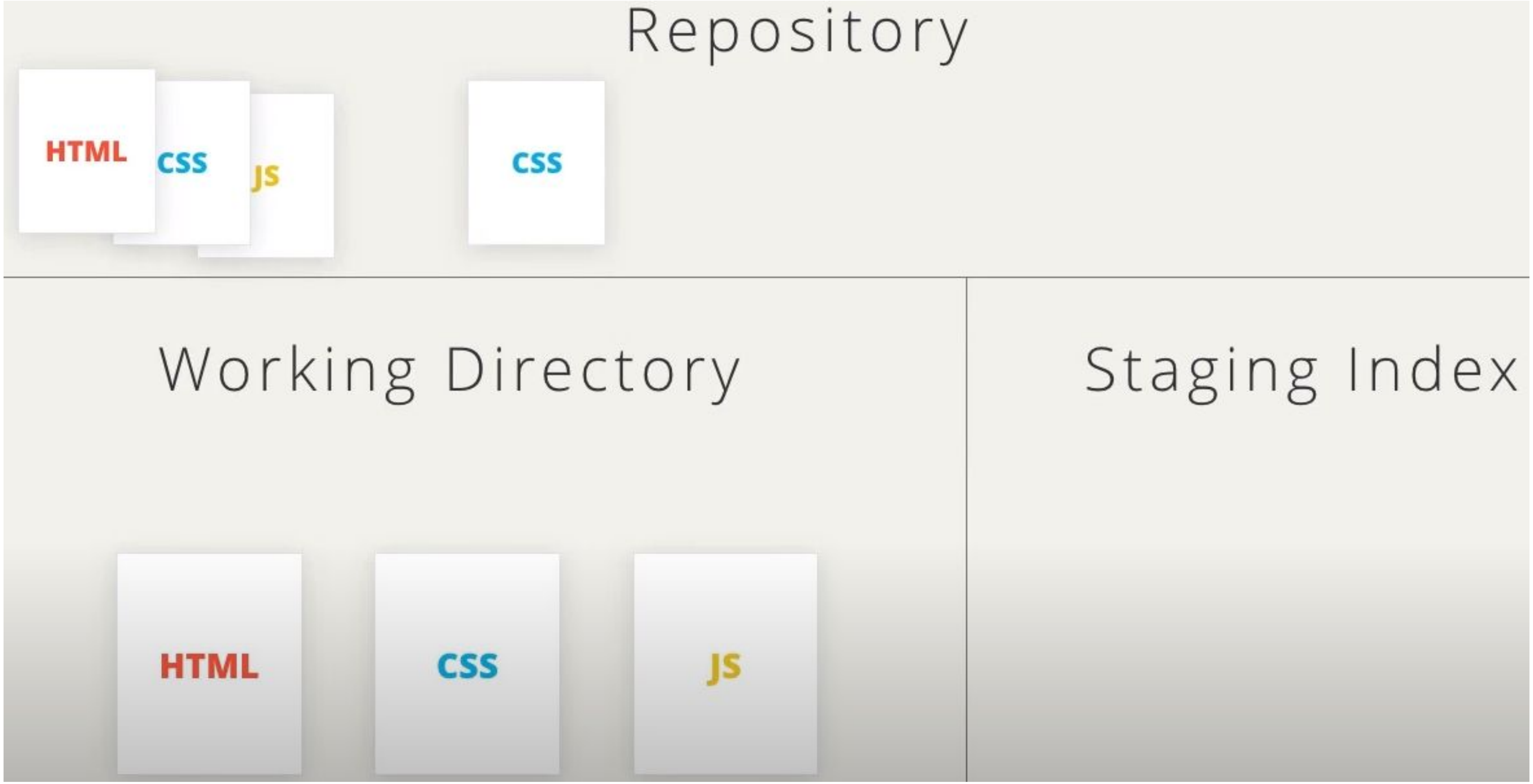
Git scenario **2: change** CSS



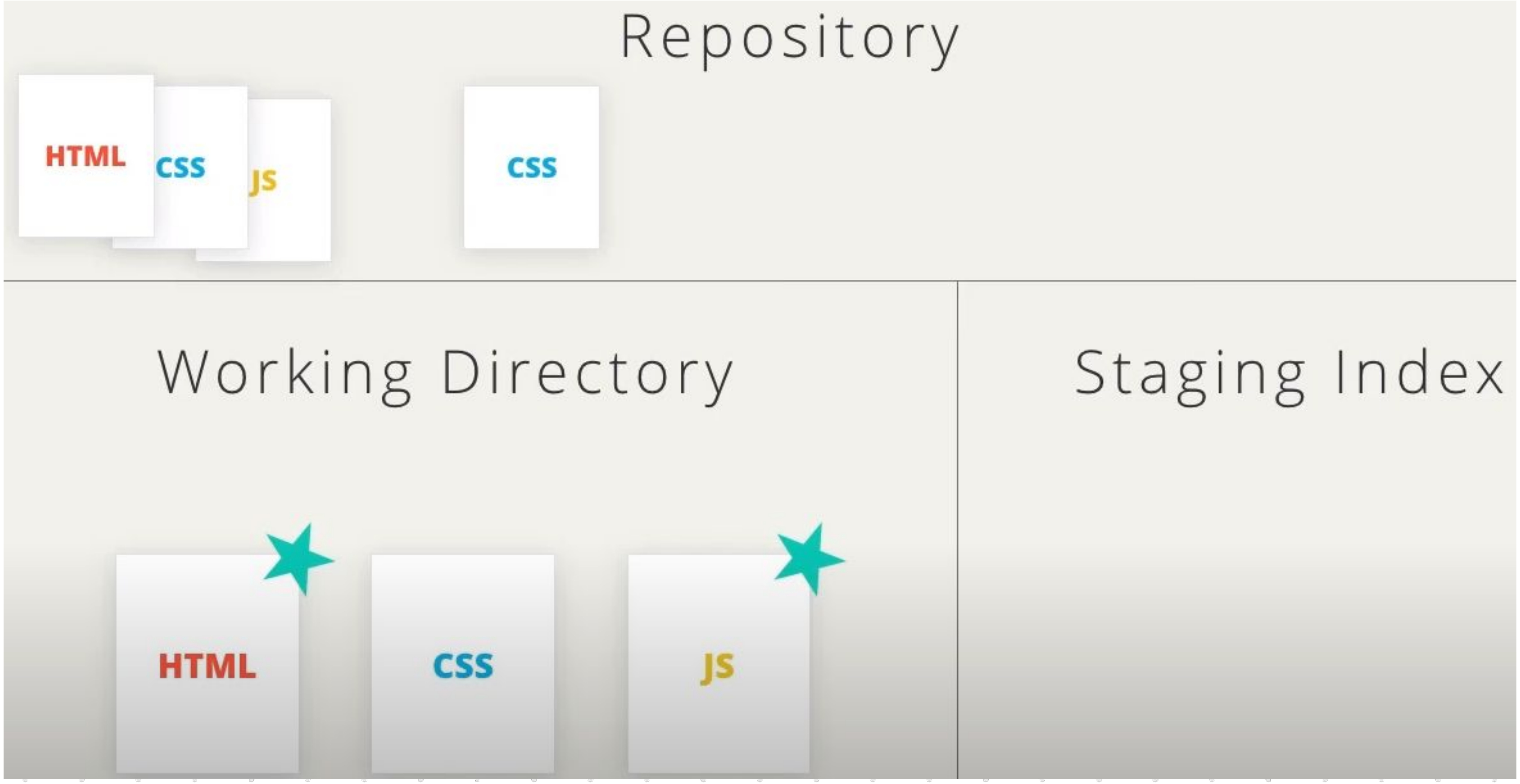
Git scenario **2: staging** changes



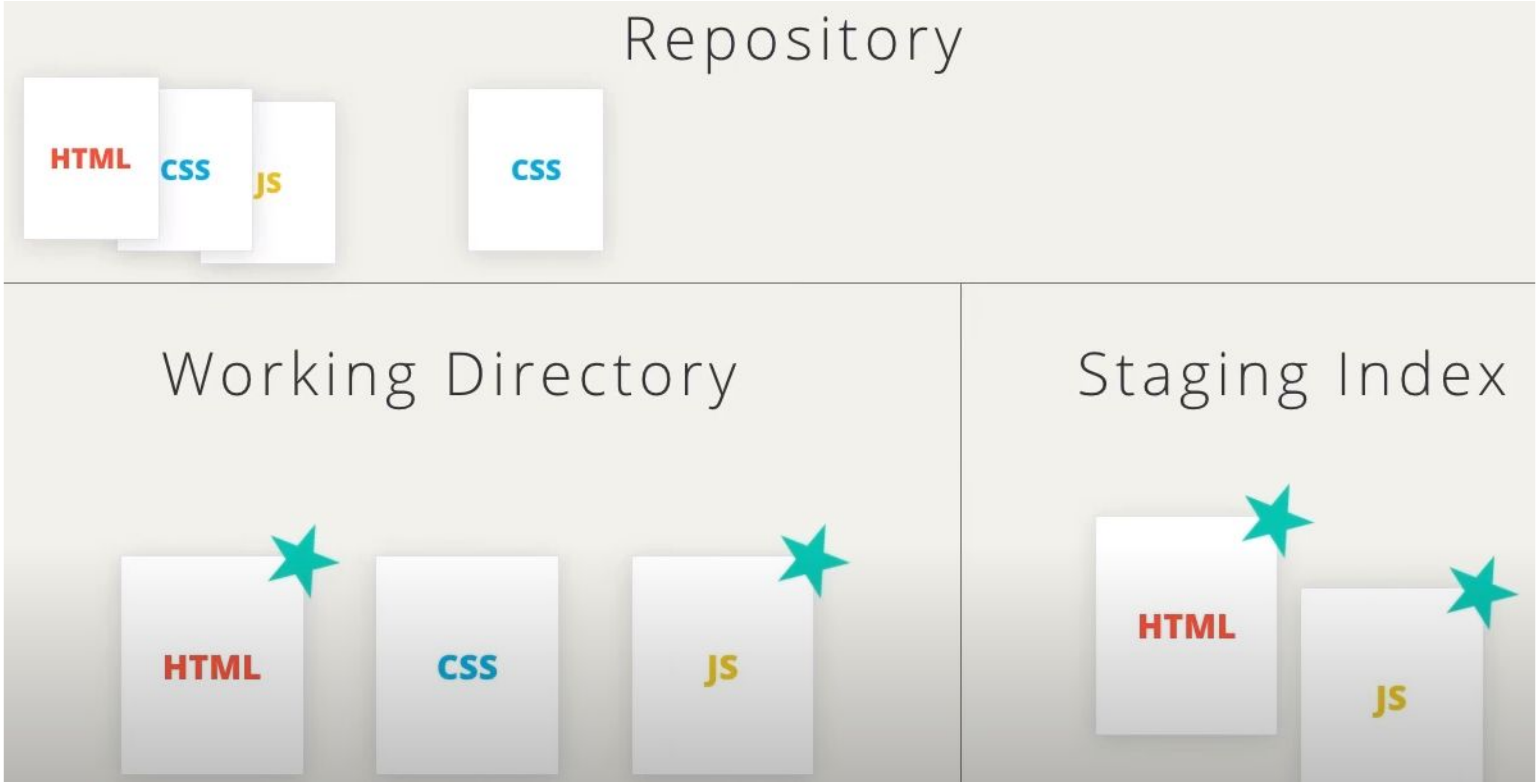
Git scenario **2: commit** changes to repo



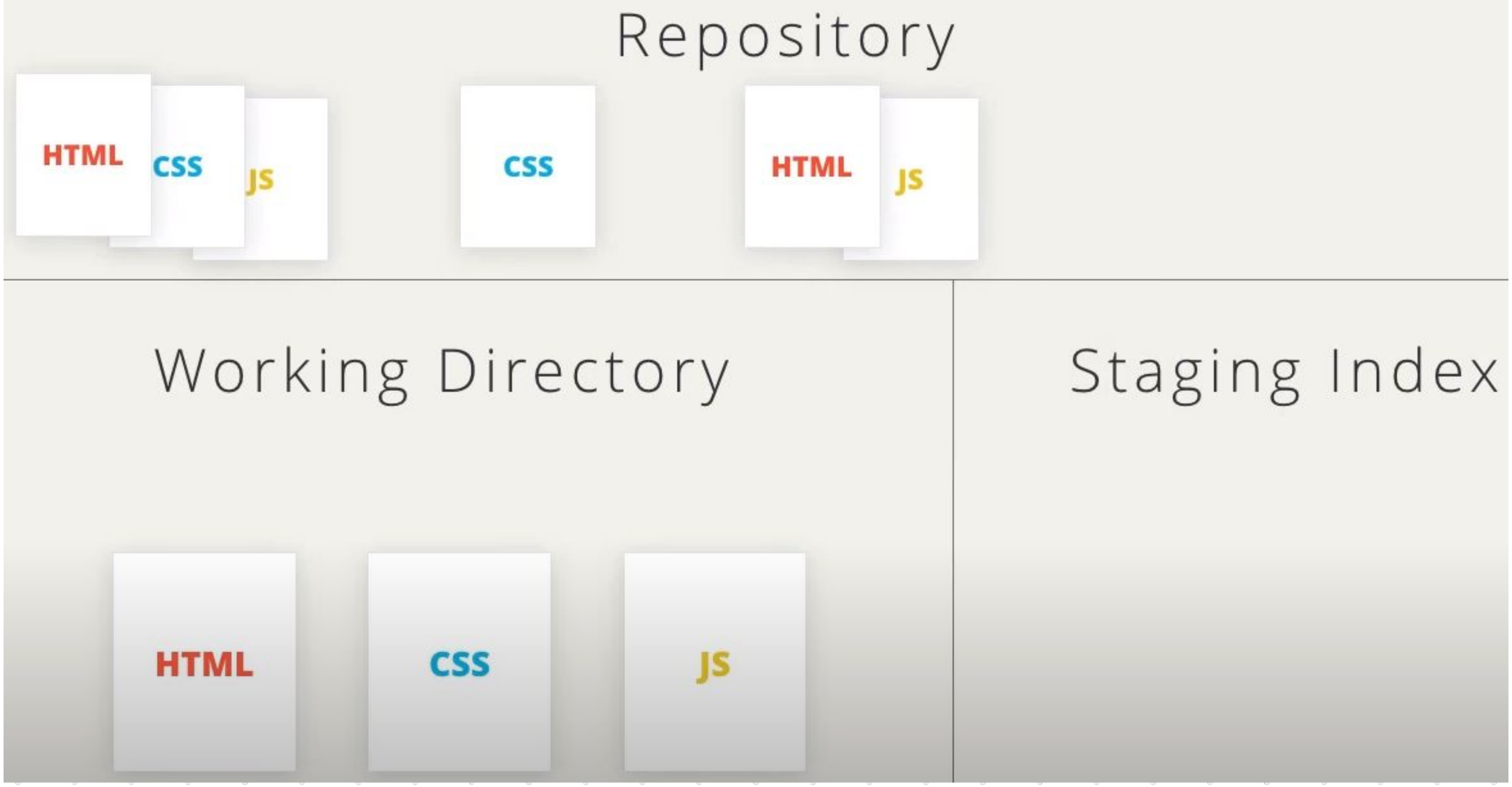
Git scenario **3: change** HTML & JS



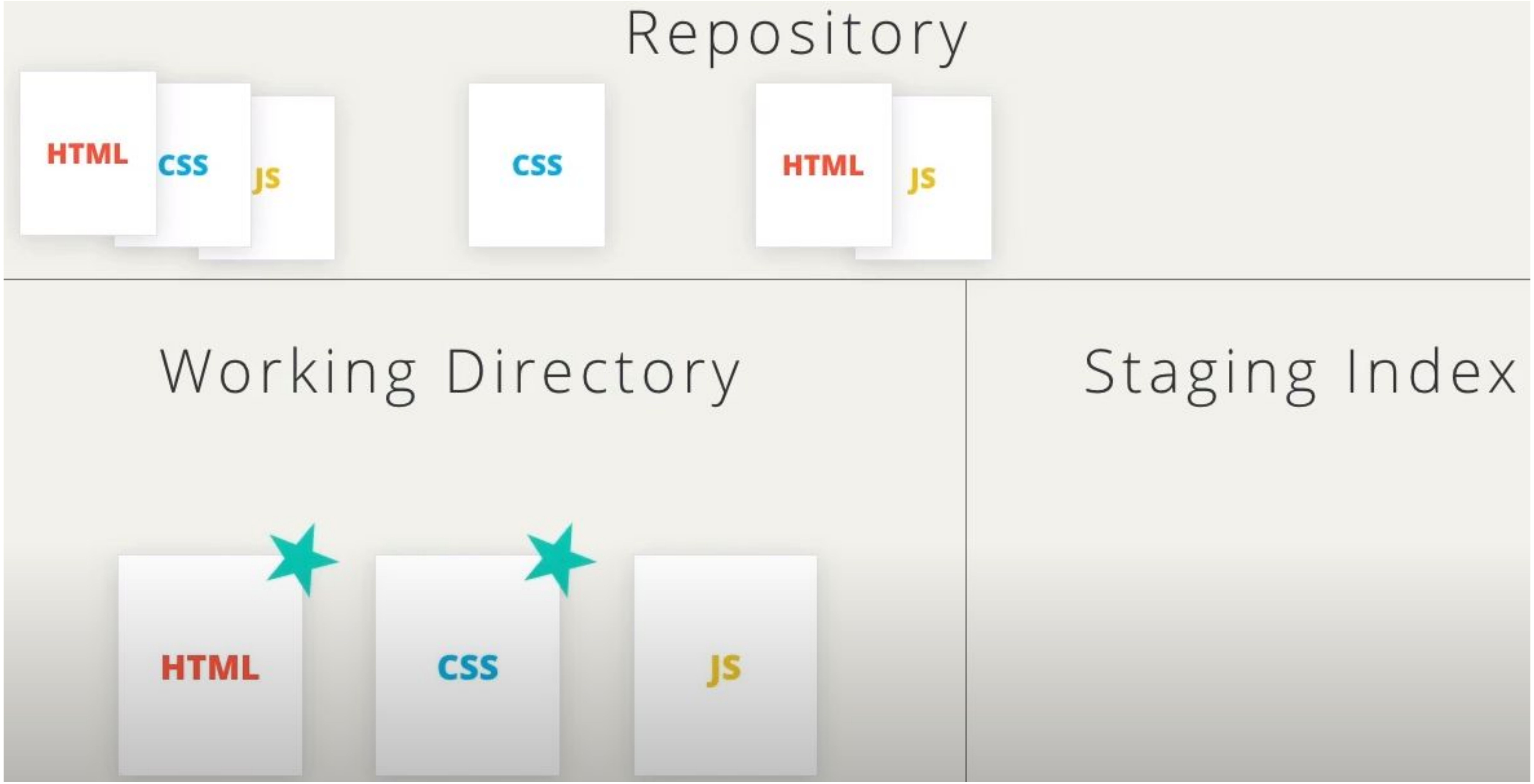
Git scenario **3: staging** changes



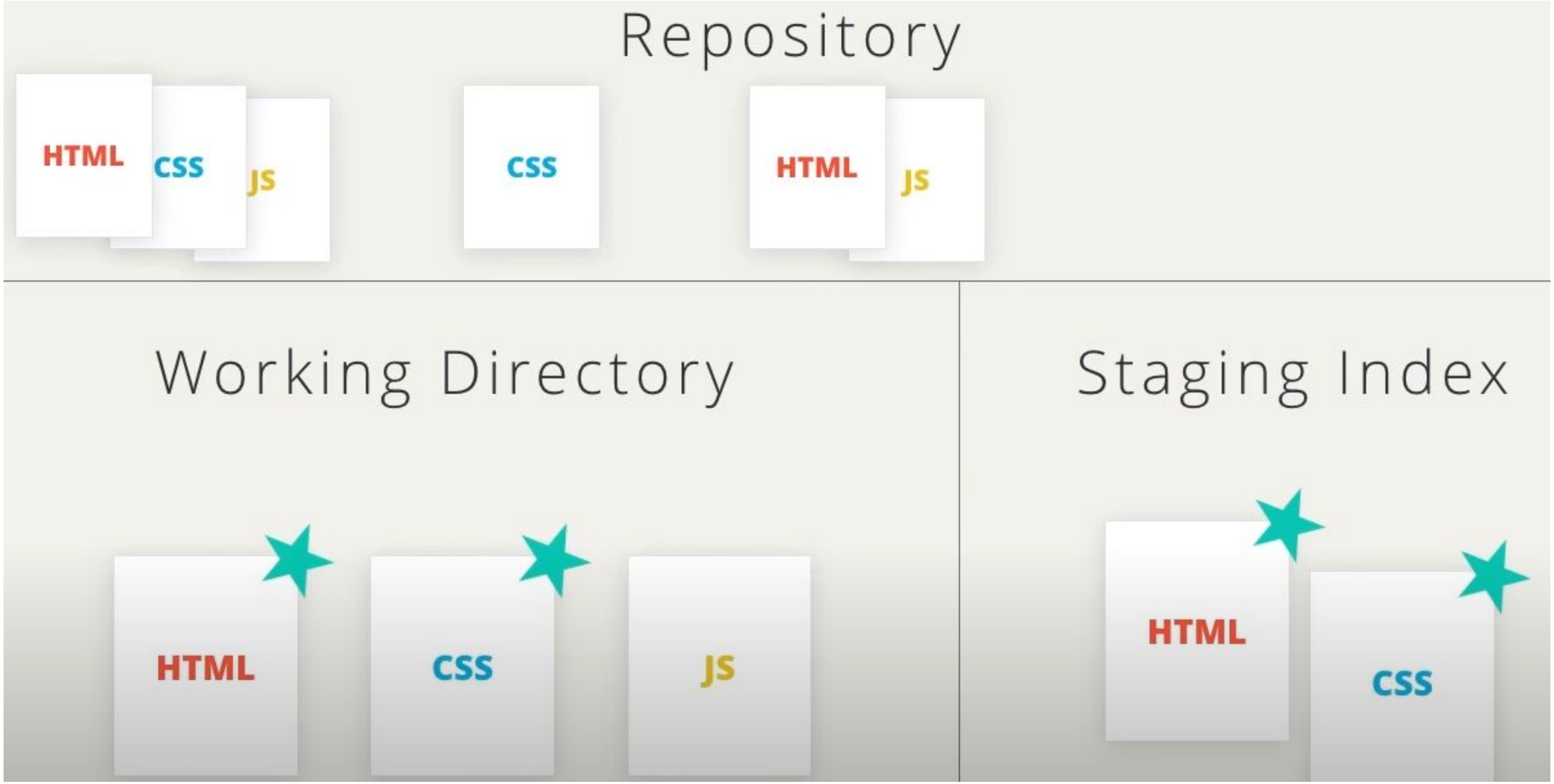
Git scenario **3: commit** changes to repo



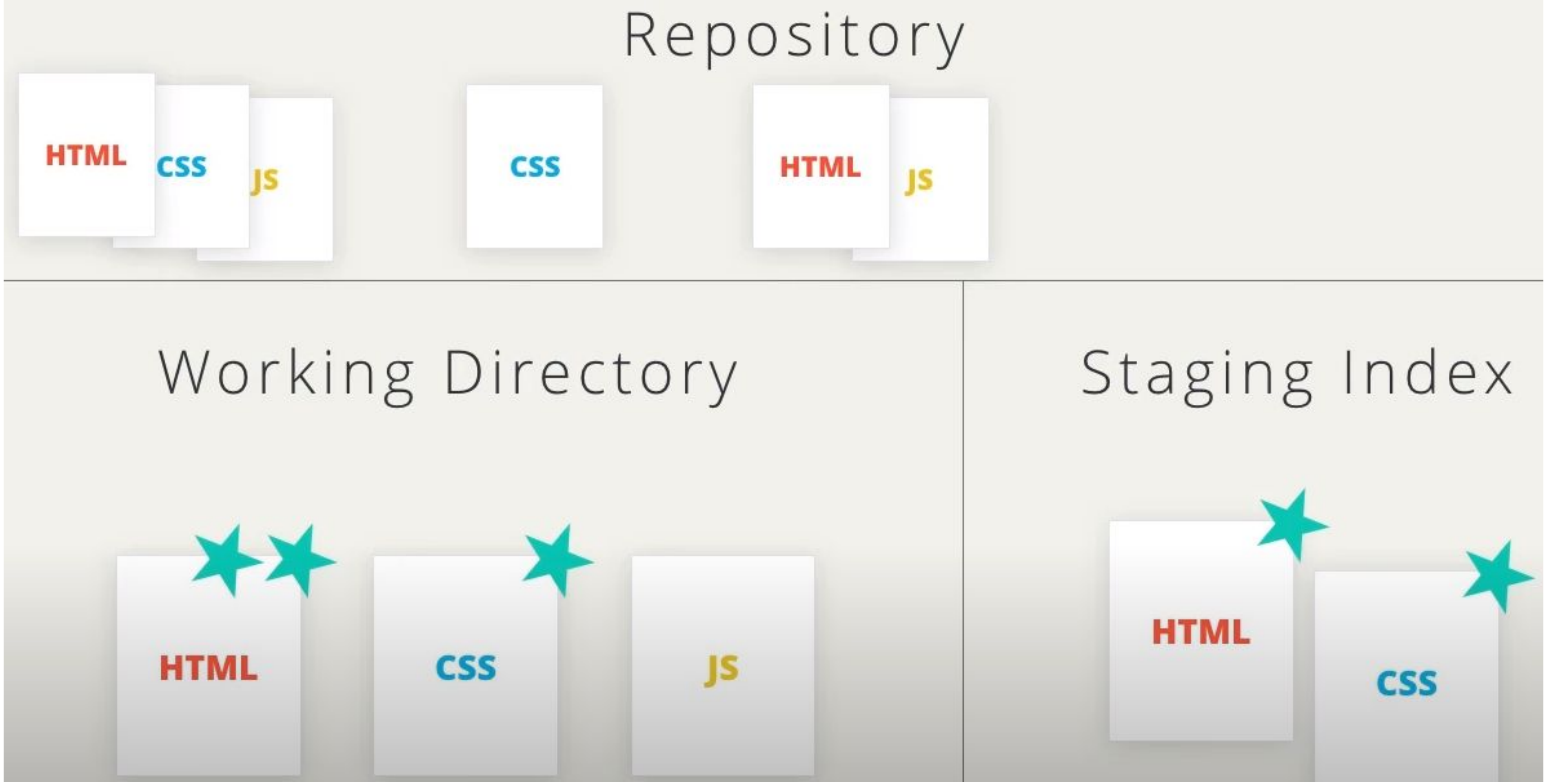
Git scenario **4: change** HTML & CSS



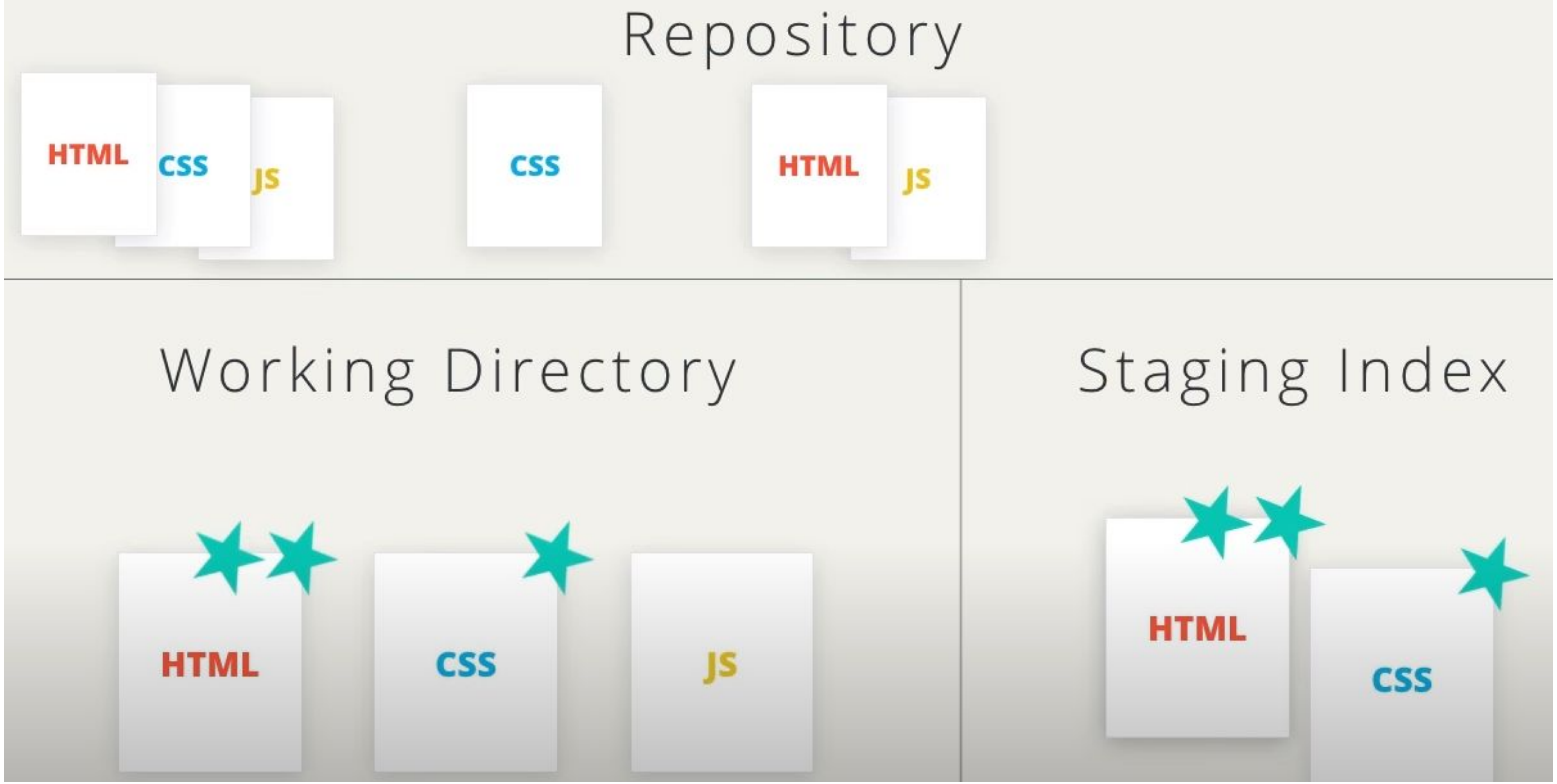
Git scenario **4: staging** changes



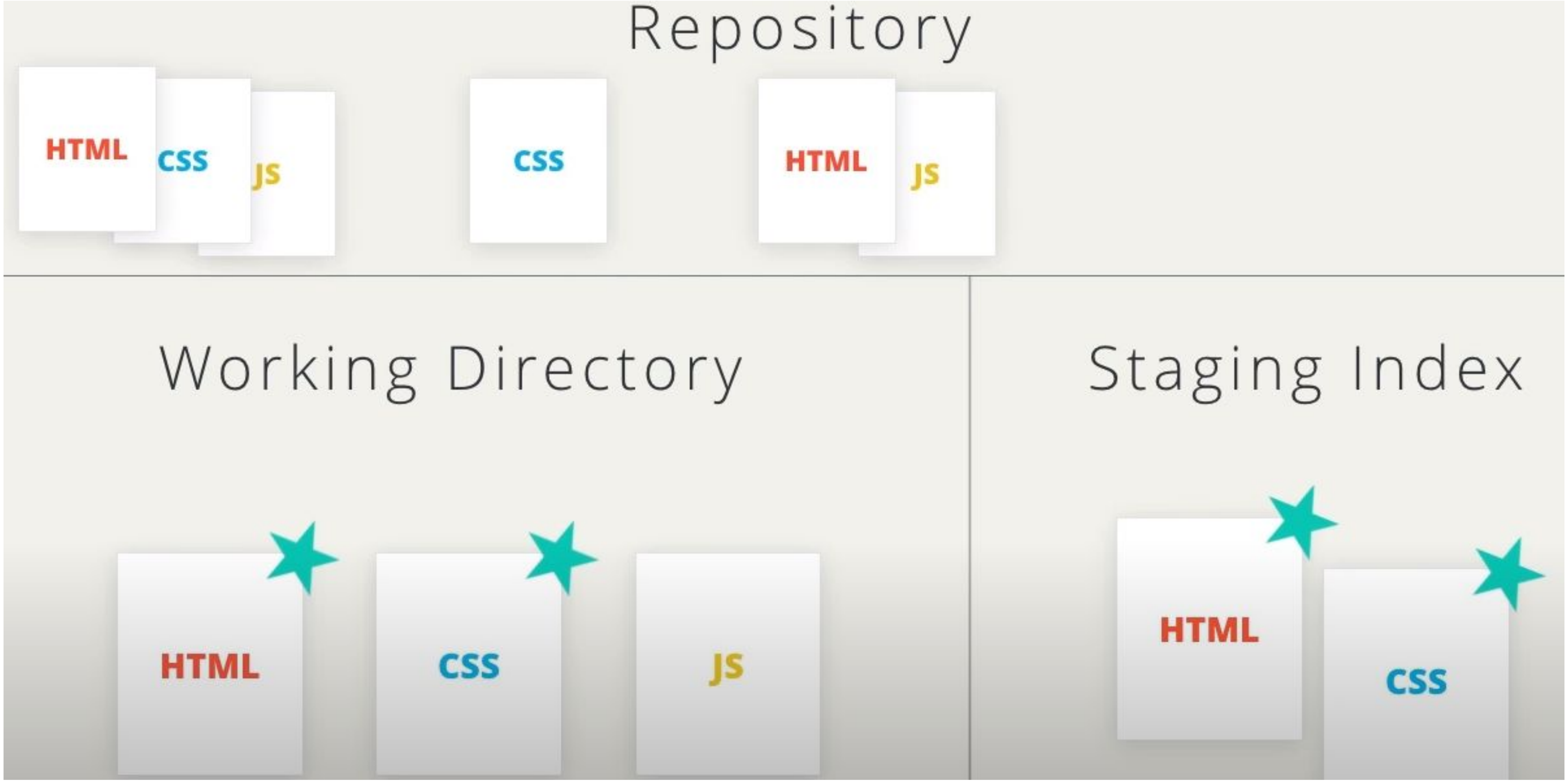
Git scenario **4: change** HTML **again**



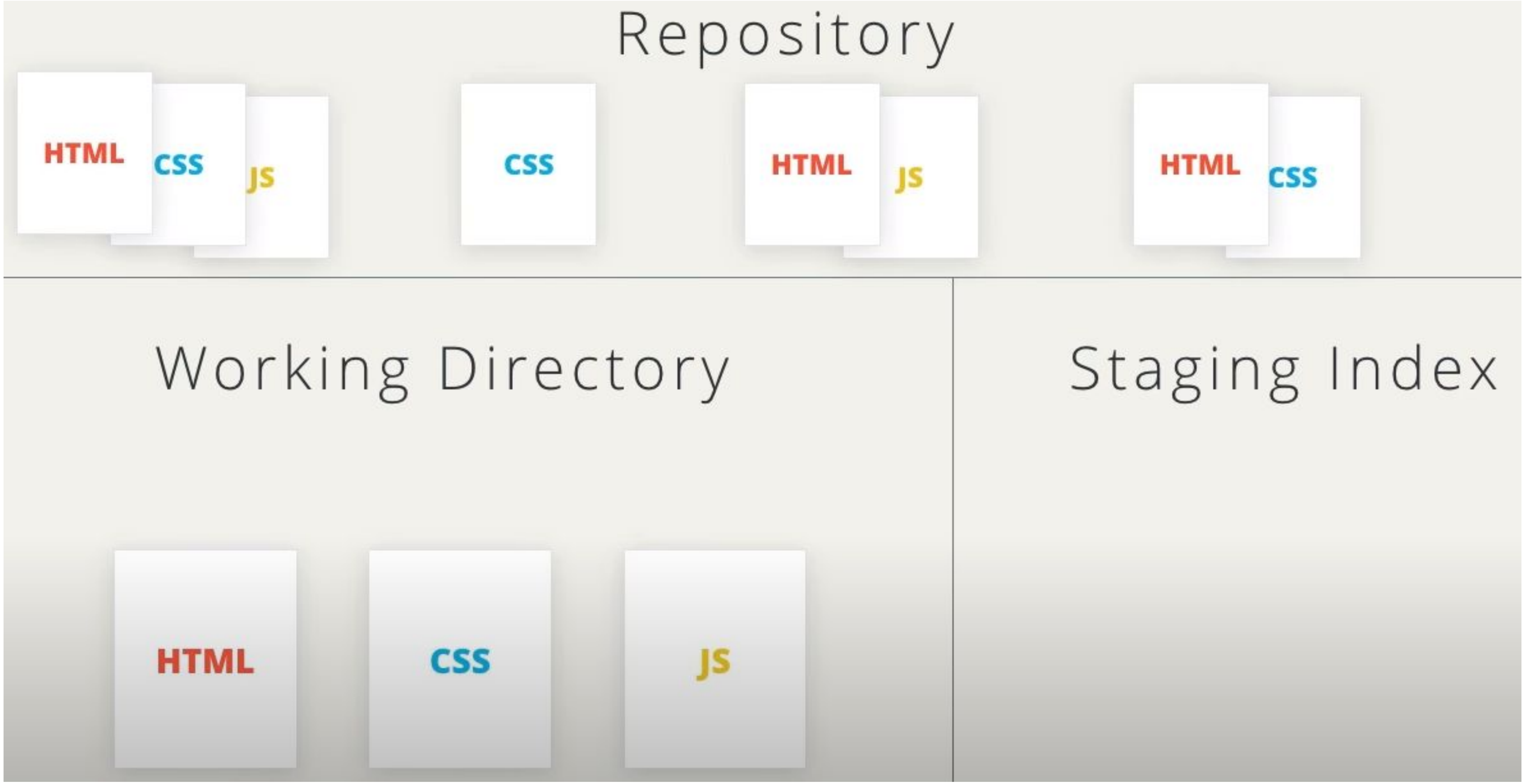
Git scenario **4: staging** changes **again**



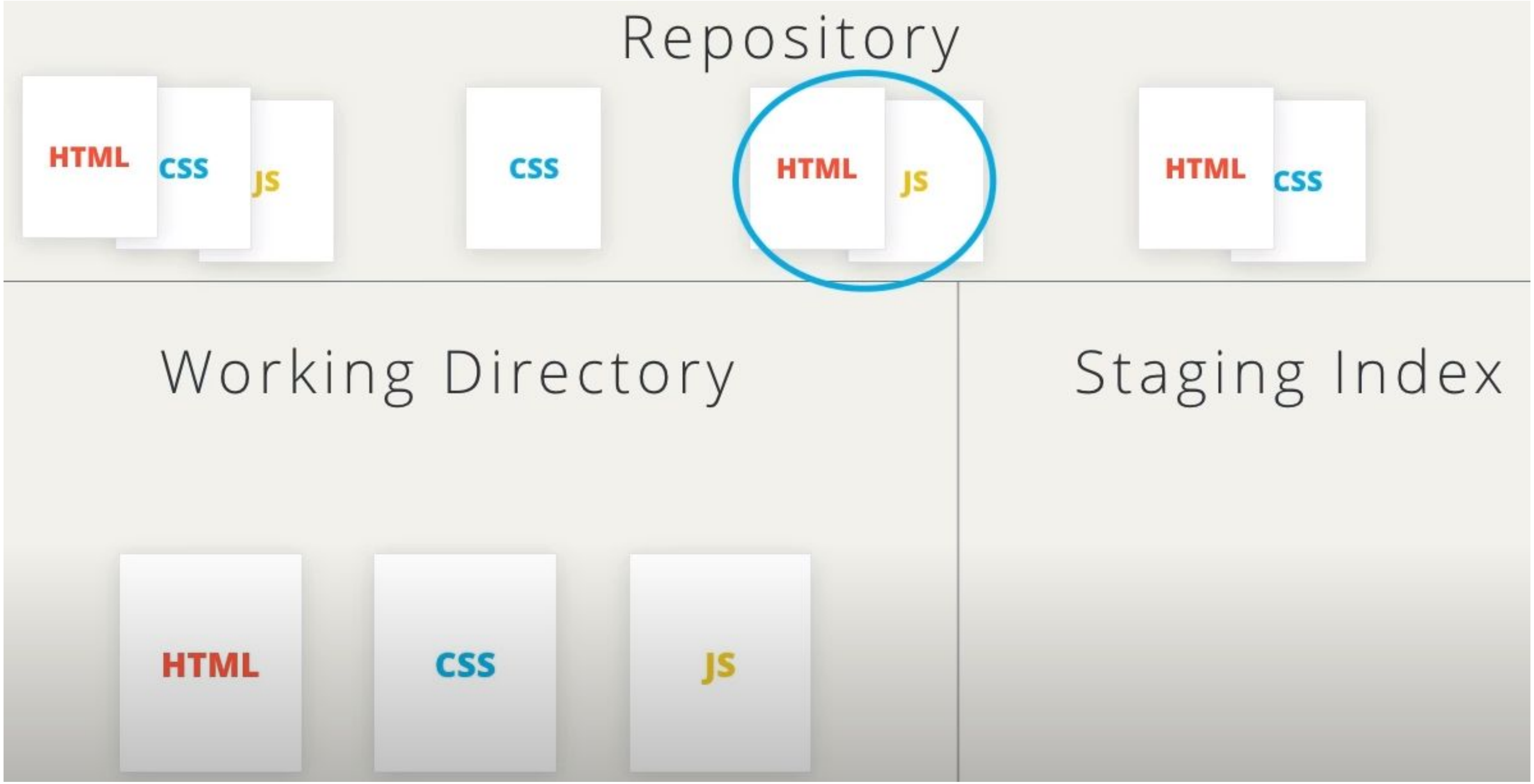
Git scenario **4: combine** changes



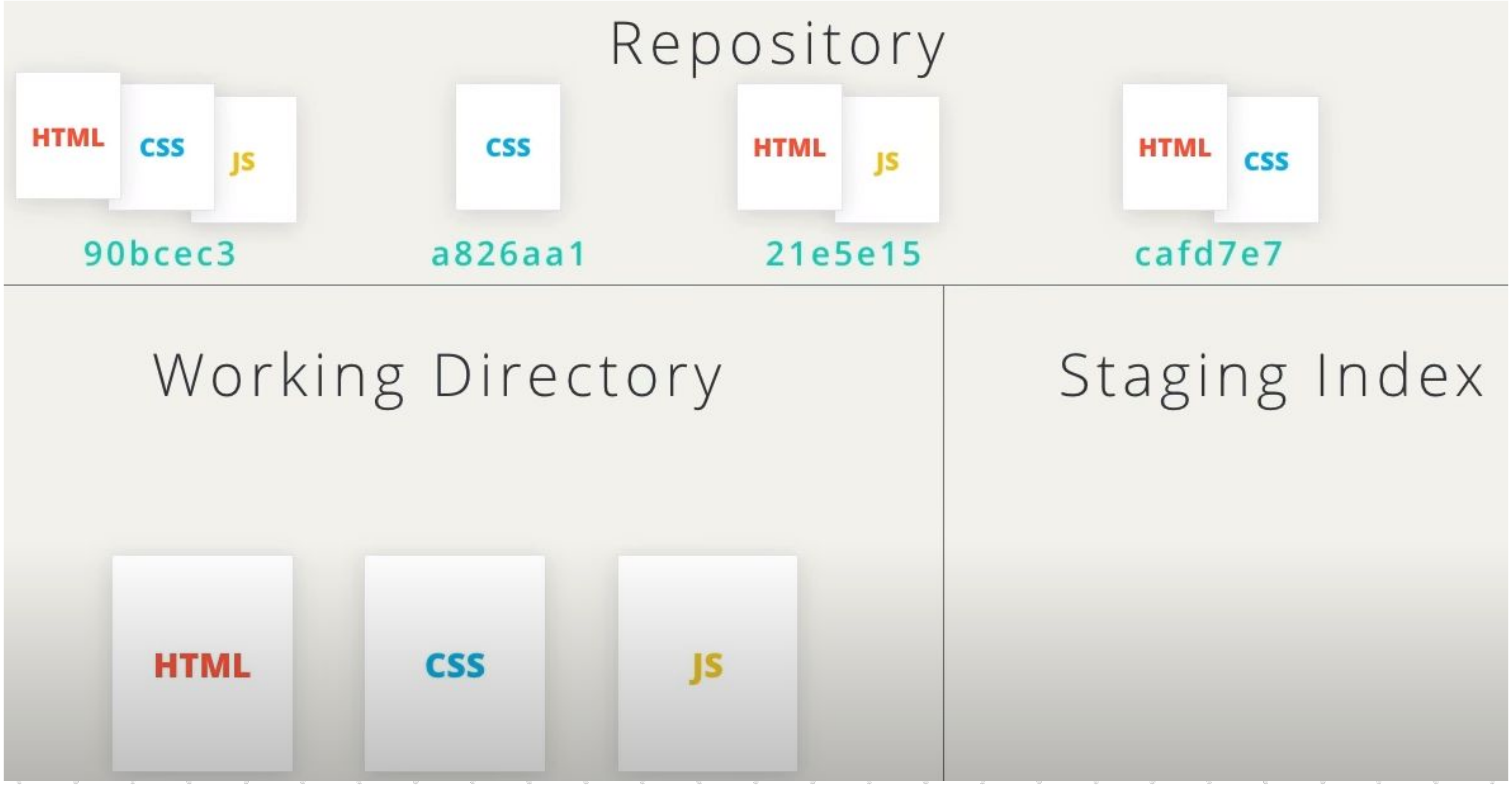
Git scenario **4: commit** changes to repo



Git: **specific commit**



Git: **specific commit hash**



Git: commands (1)

- **`git init`**
  - create new repo from scratch on your computer.
- **`git status`**
  - check (know) the status of a repo.
- **`git clone`**
  - clone an existing repo from somewhere to local computer.

Git: commands (2)

- **`git log`**
  - display info about the existing commits
  - is extremely powerful
- **`git show`**
  - displays info about the given commit
  - you provide the commit ID (aka SHA) to displays info about just that one commit.

git log (1)

```
git log --oneline
git log --stat
```

git log (2)

```
git log --patch (git log -p)
git log -p -w
git log -p <COMMIT-ID>
```

git show

```
git show
git show <COMMIT-ID>
```

Questions

Study List

1. https://www.udacity.com/course/version-control-with-git--ud123