10차시. 브랜치와 태그

Day 2 · 13:00–13:45 (수업 45분) / 13:45–14:00 (휴식 15분)
저장소: `portfolio-web`
선수: 점심시간 전 Issue 실습 완료, `main` 브랜치 최신 상태

이 차시에서 하는 일

- 메인 코드를 보호하기 위해 브랜치로 작업을 격리한다.
- `git switch -c`로 브랜치를 만들고 전환한다.
- `checkout`의 역사적 역할을 이해한다.
- 태그로 릴리스 지점을 기록한다.
- 브랜치 전환 시 파일 내용이 바뀌는 것을 직접 확인한다.

왜 필요한가

배포 중인 `main` 브랜치에서 직접 코드를 개발하다가 코드가 망가지면 배포 사이트도 즉시 마비된다. 마치 평행우주를 만들듯 독립된 복사본 가지(Branch)를 뻗어서 작업하면 마음껏 실험하다가 실패하면 가지를 버리고 성공했을 때만 합칠 수 있어 안전하다.

핵심 개념

- **브랜치**: 작업 흐름을 나누는 독립된 가지다.
- **`git switch -c`**: 새 브랜치를 만들면서 동시에 그 브랜치로 이동한다.
- **`git switch`**: 기존 브랜치로 전환한다.
- **`git checkout`**: 과거 Git에서 브랜치 전환에 쓰이던 구 명령어다. 새 코드에서는 `switch`를 우선 사용한다.
- **태그**: 특정 커밋에 이름을 붙인 릴리스 지점이다.
- **경량 태그**: `git tag v0.1.0`처럼 단순히 이름만 붙인다.
- **주석 태그**: `git tag -a v0.2.0 -m "Release v0.2.0"`처럼 작성자와 설명을 포함한다.

따라하기

1. 터미널에서 현재 브랜치를 확인한다.
2. `feature/profile-design` 브랜치를 만들고 이동한다.
3. OpenCode에 명사형 프롬프트를 입력해 스타일 수정 초안을 받는다.
4. 변경을 확인하고 stage 한다.
5. 커밋한다.
6. `main`으로 돌아온다.
7. 브랜치 격리 현상, 즉 코드가 원래대로 돌아오는 것을 확인한다.
8. 현재 `main`에 `v0.1.0` 태그를 부여한다.
9. 태그 목록을 확인한다.
10. 브랜치 삭제는 아직 하지 않는다. 병합 뒤 11차시에서 수행한다.

Git 명령

```
git branch
git switch -c feature/profile-design
git add .
```

Git 실습 실행 결과

```
$ git branch
* feature/profile-design
  main

$ git log --oneline -1
0dde98a feat: add portfolio page
```

격리된 실습 저장소의 실제 실행 출력 (원격 배포 화면 아님)

Git 명령 · 계속

전체 예제 코드 · git-commands-1.sh

선택 참고 명령

```
git checkout feature/profile-design
git tag -a v0.2.0 -m "Release v0.2.0"
```

병합 전에는 `git branch -d feature/profile-design`이 `not fully merged`로 실패한다. 브랜치 삭제는 11차시 병합 뒤에 수행한다.

에이전트 요청 예시

> 프로필 영역 부드러운 그림자 스타일 추가
> 브랜치와 태그 차이 설명

확인법

- ☐ `git switch`로 두 브랜치를 오간다.

확인법 · 계속

- ☐ 브랜치 전환 시 파일 내용이 바뀌는 것을 확인한다.
- ☐ `git tag` 실행 시 `v0.1.0`이 표시된다.
- ☐ `git status`가 깨끗하다.

오류 복구

증상	원인	대응
파일이 저장되지 않음	커밋 전 전환	커밋 후 전환한다
브랜치 이름 중복	이미 존재하는 브랜치	다른 이름을 사용한다
브랜치 삭제 실패	병합 전 삭제 시도	11차시 병합 뒤에 삭제한다
태그가 보이지 않음	태그 생성 실패	`git tag v0.1.0`을 다시 실행한다

안전 주의

- `git branch -D`로 브랜치를 강제 삭제하지 않는다.
- 미커밋 변경이 있는 상태에서 브랜치를 전환하지 않는다.
- 태그는 릴리스 지점에만 붙인다.
- 추가 심화는 보충안 심화, 복습 퀴즈는 보충안 퀴즈, 에이전트 주도 Git은 보충안 에이전트에서 다룬다.