**브랜치 (branch) #1**

# **브랜치(branch)**

## **브랜치란**

각각의 구현 작업 마다 독립된 브랜치를 만들어 구현 작업을 진행한다.

구현 작업이 성공적으로 마무리되면 그 작업 브랜치를 마스터(master) 브랜치에 병합한다.

구현 성공하고 테스트를 통과한 브랜치만 마스터 브랜치에 병합한다.

따라서 마스터 브랜치는 언제나 실행 가능하고, 안정적인 버전임이 보장된다.

어떤 새로운 기능의 구현을 시작할 때, 마스터 브랜치의 코드를 새 브랜치로 복사해서 작업을 시작하면 된다.

지금 구현하던 작업을 잠시 중단하고, 새 작업을 시작해야 한다면,

지금 작업하던 브랜치를 잠시 그대로 놔두고, 새 브랜치를 만들어 구현 작업을 시작하면 된다.

새 브랜치의 작업이 성공하면, 그 브랜치를 마스터 브랜치에 병합하고,

잠시 중단했던 브랜치로 돌아와서 작업을 계속하면 된다.

## **마스터 브랜치**

git repository에서 master 브랜치가 디폴트 브랜치이다.

master 브랜치는 git repository를 만들 때 자동 생성된다.

## **현재 작업 브랜치**

소스코드를 수정하고 커밋을 하면, 이 커밋은 현재 작업 브랜치에 소속된다.

git repository 에는 HEAD 이름의 참조가 있는데, 이 참조는 현재 작업 브랜치를 가르킨다.

## **브랜치 명령**

### **브랜치 만들기**

git branch 브랜치\_이름

### **브랜치 목록 보기**

git branch

위 명령에 의해서 브랜치 이름 목록이 출력된다.

브랜치 이름 앞에 \* 문자가 표시된 것이 현재 작업 브랜치이다.

### **브랜치 전환하기**

git checkout 브랜치\_이름

작업할 브랜치를 지정하는 명령이다.

이 브랜치가 현재 작업 브랜치가 된다.

git branch 명령은 새 브랜치를 만들기만 하고, 그 브랜치로 전환하지는 않는다.

새 브랜치를 만드는 것과 동시에 그 브랜치로 전환하는 명령은 다음과 같다.

git branch -b 브랜치\_이름

### **브랜치 병합**

예를 들어 issue1 브랜치에서 작업한 내용을 master 브랜치에 병합하려면,

다음 명령을 순서대로 실행해야 한다.

1. master 브랜치로 전환하기

git checkout master

1. issue1 브랜치에서 작업한 내용을 현재 브랜치(master)에 병합

git merge issue1

### **브랜치 삭제**

필요 없어진 브랜치를 삭제하는 명령은 다음과 같다.

git branch -d 브랜치\_이름

위 명령에 의해서 삭제되는 것은 브랜치 이름뿐이다.

그 브랜치에서 수행했던 커밋들은 삭제되지 않는다.

# **공동 작업 절차**

원활한 공동 작업을 위해서, 팀원들이 지켜야할 작업 절차는 다음과 같다.

## **작업 브랜치(branch) 생성**

master 브랜치에서 개발 작업을 하는 것은 여러가지 이유로 바람직하지 않다.

기능 구현, 버그 수정 등 각각의 개발 작업마다 브랜치를 따로 만들어서 작업한다.

## **master 브랜치에서 pull**

팀 동료가 구현 작업을 완료하고, remote repository의 master 브랜치에 새 commit을 push 하면,내 local repository의 master 브랜치에서 pull 하여, 팀 동료가 만든 새 commit을 받아온다.

팀 동료가 구현한 코드를 읽고 분석한다.

master 브랜치에서 pull 하면, master 브랜치의 내용만 버전 업 될 뿐,

다른 작업 브랜치의 내용은 아무 변화 없다.

## **테스트**

작업 브랜치에서 구현 작업을 완료했으면, 꼼꼼히 테스트해서 버그 없음을 확인해야 한다.

Junit 단위 테스트를 구현해서 테스트 하는 것이 필수다.

## **master 브랜치를 작업 브랜치에 merge**

작업 브랜치에서 구현 작업을 완료했고, 꼼꼼히 테스트해서 문제 없음을 확인했다면,

내가 구현한 작업 브랜치의 소스 코드와, 팀 동료가 구현한 master 브랜치의 최신 commit을 병합해야한다.

이 병합 작업이 생각보다 오래 걸릴 수도 있으므로, master 브랜치를 건드리지 말고,

작업 브랜치에서 병합 작업을 하는 것이 좋다.

즉 master 브랜치를 작업 브랜치에 merge하면 된다.

1. 먼저 내 local repository의 master 브랜치에서 pull 해서 최신 commit을 받아온다.
2. master 브랜치를 내 작업 브랜치에 merge 한다.

혹시 merge conflict가 있다면 해결한다.

1. merge 과정에서 버그가 생겼을 수도 있으니, 테스트해서 버그없음을 확인해야 한다.
2. 문제없는 것이 확인했다면, merge된 소스코드를 commit 한다.

## **작업 브랜치를 master 브랜치에 merge**

작업 브랜치에서 병합도 성공 했으므로, 내가 작업 브랜치에서 구현한 소스 코드를 팀 동료들에게 공개하는 절차만 남았다.

1. 작업 브랜치를 master 브랜치에 merge 한다.
2. master 브래치를 remote repository에 push 한다.