GIT & CI/CD

SYD 문민식

목차

1. 들어가기전

1. 혼자 쓰는 git / 같이 쓰는 git

   => 같이 쓰는 git 경험은 1회

2. 업무 외 svn 다른 기능은 활용해 본 적 없음.

3. git 명령어, 실습 등을 배제하고, 경험적으로 좋았던 점 소개.

   => '상황중심의 git'

git의 강점 & 장점 한마디로?

브랜치 & 원복

지옥에서 온 Git

공동공

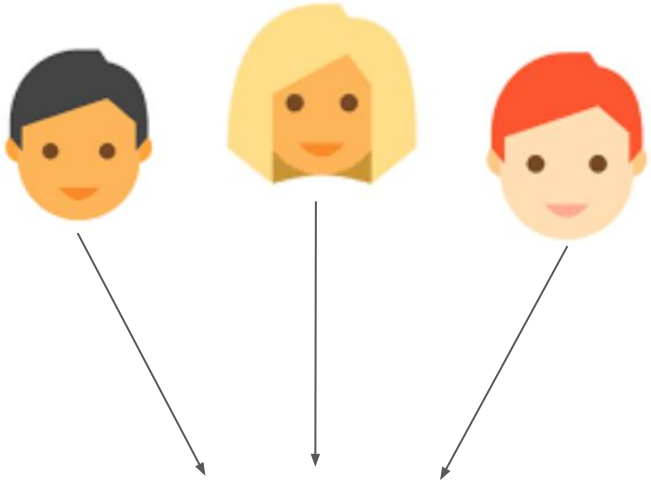
git의 혁신 - branch

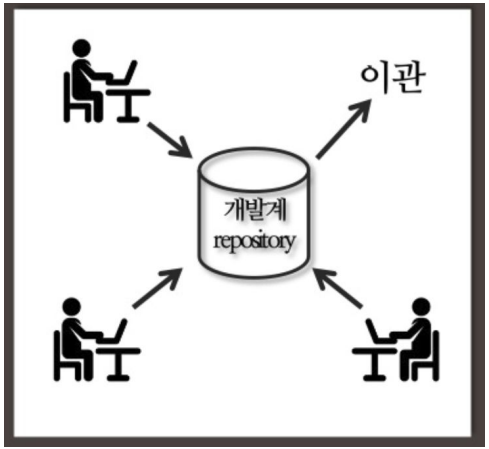
2017-02-14 23:22:10

수업내용

여기서는 git이 가져온 혁신인 branch를 다룹니다. branch는 가지라는 뜻으로 작업을 분기해서 처리하는
우에 대한 시적인 표현입니다. git이 가져온 혁신은 그 동안 개념적으로는 존재했지만 실제로는 잘 사용하
않던 branch를 사용할만한 상태까지 끌어올린 점에 있습니다. 본 수업의 하위에서는 branch에 대해서 다
니다.

2. 당시 상황





1. 동시다발적인 qna, 개발요구
2. 출발지는 같으나 도착지가 다른 개발을 동시에 진행
3. 최종 버전의 형태도 불분명(요구사항 변경, 요구사항 추가, 개발 취소)
4. 하나의 개발psr에서 많은 소스, 라인들이 변경됨 (개인 기록이나 copy로 관리 불가능)

1. 커밋 자유롭게 못함
2. 개발자들끼리 개발소스 겹침

2. 상황 요약 & git을 써서

-> 출발지가 같고(현재 운영계 버전)

-> 서로의 순서 관계를 모르는 불명확한 버전을 목표로,

-> 동시에 개발을 진행해야 하는 상황.

-> 최대한 많은 경우의 수(branch)를,
-> 기록해(commit) 선택후(checkout, no 원복노력),

-> 조립(merge)한다. (최종 컨펌 버전과 순서들이 정해졌기에)

git의 강점 & 장점 한마디로?

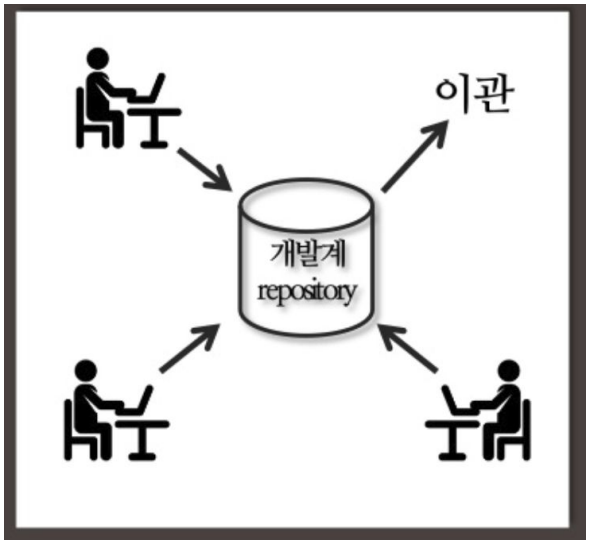
브랜치 & 원복

3. git과 인생

-> 마스터 브랜치밖에 없는 대표적인 예가 인생이라고 생각.

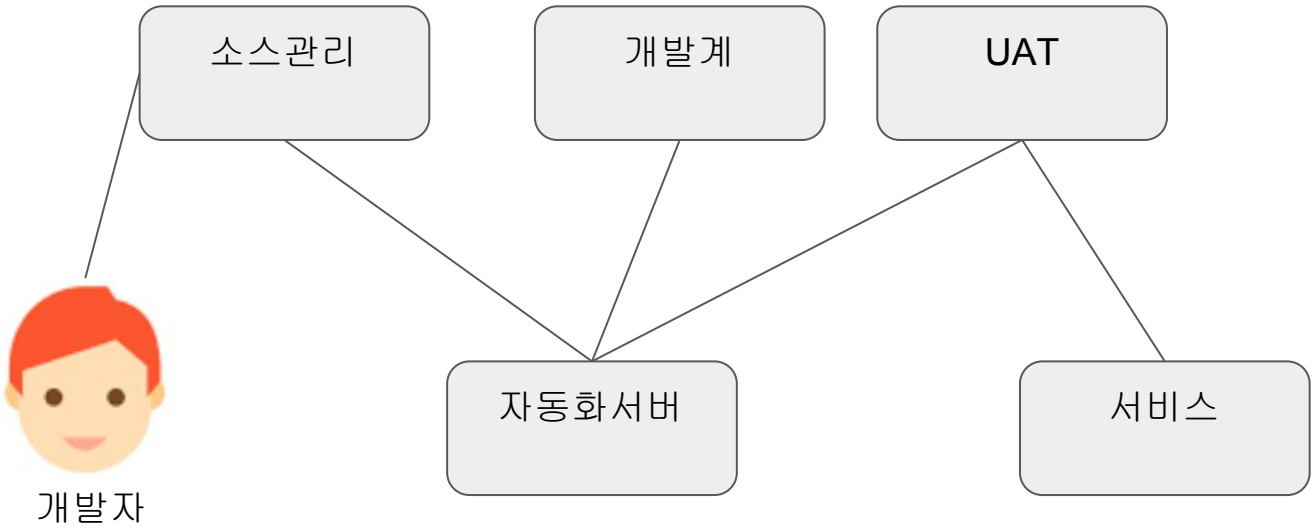
-> 인생 브랜치를 따고 싶은 욕구 = 타임머신에 대한 갈망.

4. CI / CD



1. 가장 단순한 형태의 개발 모습
2. 느린 개발
3. 적은 release 횟수
4. 사람 손(인간의 노동)에 의지하는 모습.

4. CI / CD



더 빠르게, 더 많이, 자동화 **=>** 더 높은 품질의 프로그램과 고객 서비스가 목표