중앙집중식 통합

* 개요
* 원칙
* 개발할 형상에 대하여 담당자를 정하여 충돌이 발생할 가능성을 미연에 방지한다.
* 본인 담당 로직에 대하여 통합시점 전까지 Local에서 충분히 테스트후 통합시점에서 Push를 진행한다.
* Push와 동시에 자동적으로 시스템 통합이 이루어 진다.
* 형상분리 예
* 개발자1 ☞ 계산기 로직 담당

public class Calculator {

public static double performOperation(double num1, double num2, char operator) {

double result = 0.0;

switch (operator) {

case '+':

result = num1 + num2;

break;

case '-':

result = num1 - num2;

break;

case '\*':

result = num1 \* num2;

break;

case '/':

if (num2 != 0) {

result = num1 / num2;

} else {

System.out.println("0으로 나눌 수 없습니다.");

}

break;

default:

System.out.println("올바른 연산자를 입력하세요.");

}

return result;

}

}

* 개발자2 ☞ 계산기 입출력 로직 담당

import java.util.Scanner;

public class CalculatorInput {

public static void getInputAndCalculate() {

Scanner scanner = new Scanner(System.in);

System.out.println("간단한 콘솔 계산기 프로그램");

System.out.print("첫 번째 숫자를 입력하세요: ");

double num1 = scanner.nextDouble();

System.out.print("두 번째 숫자를 입력하세요: ");

double num2 = scanner.nextDouble();

System.out.print("사용할 연산자를 입력하세요 (+, -, \*, /): ");

char operator = scanner.next().charAt(0);

double result = Calculator.performOperation(num1, num2, operator);

if (!Double.isNaN(result)) {

System.out.println("결과: " + result);

}

scanner.close();

}

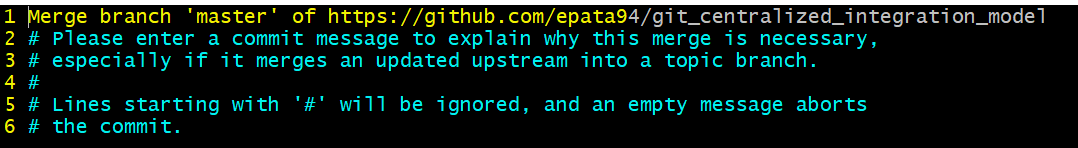
}

* 적용 시나리오
* 연습1 – 개인 연습
* GitHub에서 ‘git\_centralized\_integration\_model’ 레파지토리생성
* 로컬 워크스페이스 홈디렉토리의 서브디렉토리에 developer1, developer2 디렉토리를 만든다.
* 각각 원격 레파지토리와 연결한다.
* developer1 작업공간에서는 Calculator.java를 developer2 작업공간에서는 CalculatorInput.java를 번갈아 가면서 로컬 commit을 진행하며 version을 올린다.
* 모든 개발이 끝났을 경우에는 최종 통합 진행

Push전 최종 형상 pull

만약에 내가 담당하지 않은 형상에 대한 변경이 서버에 반영이 되어 있고 이 내역이 내 로컬에 반영시 자동 머지가 일어나고 이에 대한 자동 커밋메세지가 보인다. 별도의 메시지작성이 필요없으면

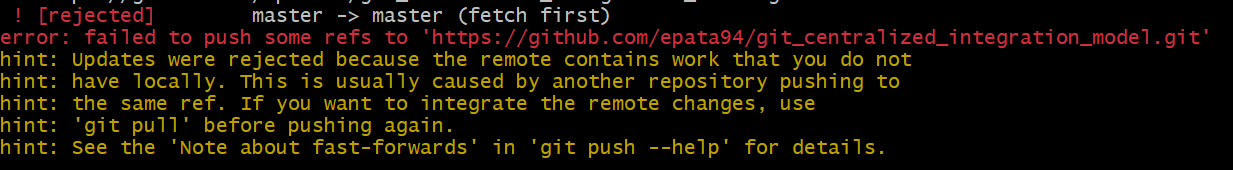
:q로 종료한다.



* developer1 pull &push
* developer2 pull
* developer2 push
* developer1 pull

Pull 하지 않고 Push 한 경우

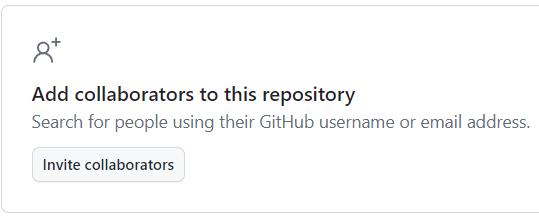
아래 에러 메시지 발생



git pull origin master 후 자동 머지 커밋 메시지 작성 후 pull 작업 완료 후

다시 push 한다.

* 연습2 – 팀연습
* 2인 1조로 진행하여 GitHub에서 ‘git\_centralized\_integration\_model\_team’ 레파지토리를 public 모드로 생성한다.
* 레파지토리 생성한 GitHub 계정 담당자가 팀원의 권한을 할당한다.



초대를 받은 팀원은 이메일 인증을 진행한다.

* 로컬 워크스페이스에서 [팀명 또는 프로젝트명] 디렉토리를 만든다.
* 각자 원격 레파지토리와 연결한다.
* 팀멤버는 Calculator.java, CalculatorInput.java 형상 담당자를 정한다.
* 각 담당자는 개인 로컬레파지토리에서 여러 번에 걸쳐 commit을 진행한다.
* 최종 개발 완료 후 팀원 합의하에 통합(push)을 진행한다.

Push 전에는 Pull 한다.

* Push가 진행되었다면 다른 개발자의 레파지토리에서 Pull 명령을 진행하여 최종 형상을 공유한다.
* 최종 형상에 대한 테스트를 진행한다.
* 실행결과물을 보고 개선 사항에 대해 논의한 후 추가작업을 진행한다.
* 추가 작업 후 최종 통합을 진행한다.