그룹별 Git: 완전한 가이드

Git을 사용하여 그룹으로 작업할 때 명확하고 체계적인 작업 프로세스를 갖는 것이 중요합니다. 팀에서 Git을 효과적으로 사용하는 방법에 대한 단계별 가이드입니다.

기억해야 할 단계에 대해 간략히 설명하겠습니다.

제가 추천하는 접근 방식은 문제부터 시작하는 것입니다. 따라야 할 단계는 다음과 같습니다(가이드 하단에서 더 자세히 설명).

1. **문제를 선택하고** , 브랜치를 생성한 뒤 생성된 명령을 복사합니다.
2. 터미널에서 명령을 실행하여 **새로운 브랜치로 전환하세요** . 새로운 브랜치는 자동으로 최신 버전을 기반으로 합니다 `main`.
3. **변경 사항을 적용하세요.** 진행 상황을 저장하려면 스테이징에 변경 사항을 추가하고 커밋을 생성하세요. 원격 저장소에 푸시할 수도 있지만 이는 선택 사항입니다.
4. **병합할 준비가 되면** `main` **브랜치** 를 푸시하고 풀 리퀘스트를 생성하세요.
5. **풀 리퀘스트를 검토받으세요.** 동료가 검토하고 모든 것이 괜찮은 것 같으면 이를 병합합니다 `main`. 귀하의 이슈는 병합된 브랜치에 연결되어 있으므로 자동으로 닫힙니다.

각 단계는 자세히 설명되어 있습니다.

1. 기능을 선택하세요

먼저 작업할 기능이나 작업을 선택하세요. 많은 프로젝트에서는 GitHub Issues와 같은 이슈 관리 시스템을 사용하여 작업을 정리합니다.

- **프로젝트에 문제가 사용되는 경우: 작업할 열려 있는 문제를 찾아 자신에게 할당하세요.**
- 문제가 없는 경우: 수행할 수 있는 작업에 대해 팀과 소통하세요.

2. 브랜치 생성

코드를 체계적으로 정리하려면 각각의 새로운 기능이나 수정 사항을 별도의 브랜치에서 개발해야 합니다.

GitHub Issue에서 직접 브랜치를 생성합니다.

GitHub을 통해 작업하는 경우 GitHub UI에서 "브랜치 만들기" 버튼을 클릭하여 이슈에서 직접 브랜치를 만들 수도 있습니다. 받은 명령을 복사해두었다가 나중에 터미널에서 사용하세요.

터미널로 가서 명령을 실행하세요. 새로운 브랜치에 도달했는지 확인하세요. 터미널의 응답에 그렇게 나와요.

터미널을 통해 브랜치를 만듭니다.

1. 마스터 브랜치( ) 에 있는지 확인하세요 . `main`

```
git checkout main
```

2. 최신 변경 사항 받기:

```
git pull origin main
```

3. 새로운 브랜치를 생성합니다. `main:`

```
git checkout -b <feature-namn>
```

3. 기능을 개발하세요

기능 개발을 시작하세요. 코드를 변경한 후 커밋하기 전에 로컬에서 테스트합니다.

4. 커밋(스테이징에 추가)하려는 파일을 선택합니다.

변경 사항에 만족하면 커밋하기 전에 "스테이징" 영역에 추가해야 합니다.

1. 어떤 파일이 변경되었는지 확인하세요.

   ```
   git status
   ```

2. 또는 변경된 모든 파일을 추가합니다(터미널이나 VSCode의 소스 제어를 통해 개별 파일을 추가할 수도 있습니다):

   ```
   git add .
   ```

5. 커밋 메시지를 작성하세요

각 커밋에는 변경된 내용을 명확하고 구체적으로 설명하는 메시지가 있어야 합니다.

```
git commit -m "Lade till funktion för att filtrera produkter efter kategori"
```

좋은 커밋 메시지:

- 현재 시제의 동사로 시작합니다("추가", "수정", "업데이트")
- 변경 사항이 무엇을 하는지 간단하고 간결하게 설명합니다.

6. GitHub의 브랜치에 푸시하세요

변경 사항을 커밋한 후에는 나머지 팀원도 볼 수 있도록 GitHub에 푸시해야 합니다. **문제를 통해 브랜치를 생성한 경우** :

```
git push
```

이 브랜치를 처음으로 푸시하는 경우(문제를 통해서가 아니라 로컬에서 생성한 경우):

```
git push --set-upstream origin <feature-namn>
```

7. 풀 리퀘스트 만들기

기능이 완전히 개발되고 테스트되면 팀의 다른 구성원이 코드를 추가하기 전에 검토할 수 있도록 **풀 리퀘스트(PR)** 를 생성하세요 . `main`

1. GitHub의 저장소로 이동합니다. 방금 푸시를 한 경우 비교를 하고 풀 리퀘스트를 만들 것인지 묻는 메시지가 나타납니다. 그렇지 않은 경우: **풀 리퀘스트를** 클릭하고 **새 풀 리퀘스트를** 클릭합니다 . 지점을 선택하여 비교해보세요 . `main`
2. PR에 포함된 내용과 변경이 필요한 이유를 설명하세요.
3. 팀원에게 코드를 검토해 달라고 요청하세요.