Git을 이용한 버전 관리【Git의 기본】| 누구나 쉽게 이해할 수 있는 Git 입문~버전 관리를 완벽하게 이용해보자~ | Backlog

Git의 기본

시작하기

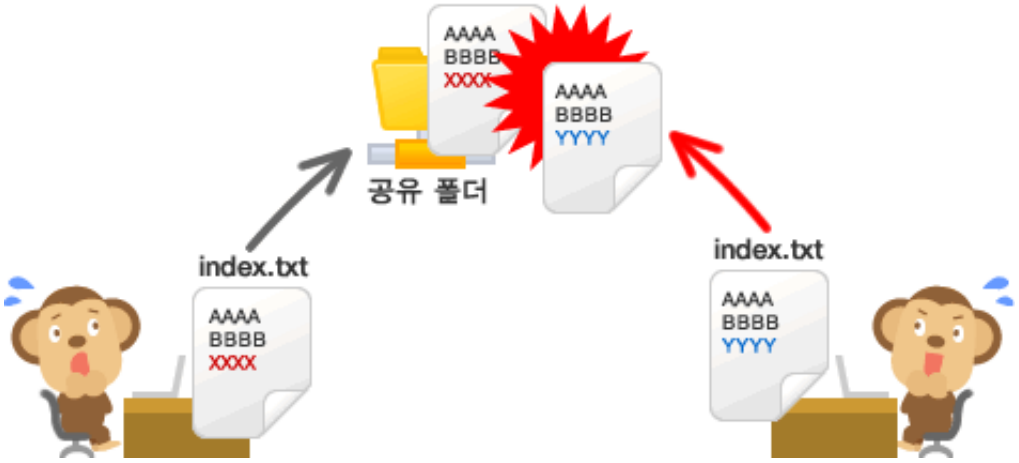
안녕, 하카타에서 태어난 원숭이 킥킥이야. 오늘은 나랑 같이 버전 관리 시스템, 'Git(깃)' 을 공부해보자.

여러분은 파일을 편집 전 상태로 되돌리고 싶을 때 어떻게 하고 있나요?

가장 간단한 방법은 편집하기 전에 파일을 미리 복사해두는 것입니다. 파일과 폴더명 뒤에 편집한 날짜를 붙여주는 방식이죠. 하지만 파일을 편집할 때마다 매번 복사하는 일은 번거롭기도 하고 실수할 가능성도 많습니다.

또한 위의 그림처럼 특별한 규칙 없이 마음대로 이름을 붙여놓는 경우 어느 파일이 최신인지, 또 파일의 어떤 부분이 변경된 것인지 파악하기 어렵습니다.

아래 그림을 보세요. 이렇게 여러 명이 공유한 파일을 동시에 편집하는 바람에 다른 사람이 먼저 변경하고 있던 내용을 지워버린 경험은 없나요?



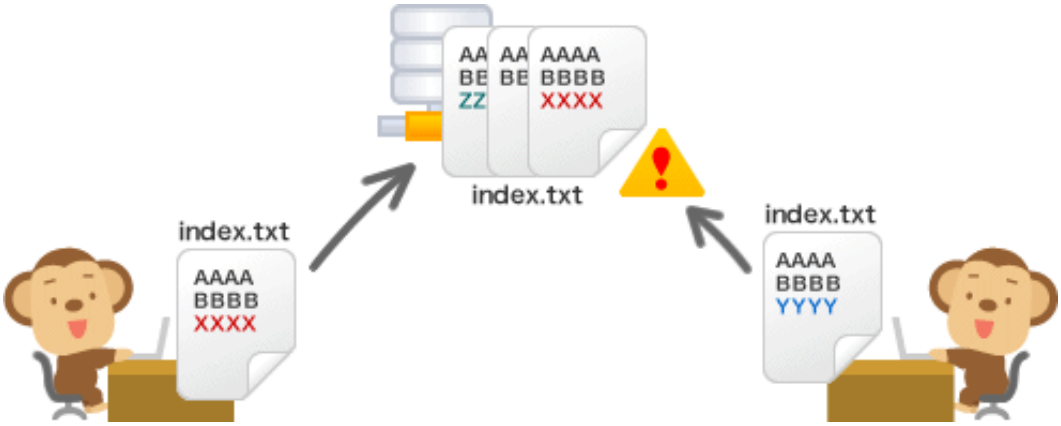
바로 이런 문제를 해결하기 위해 만들어진 것이 Git과 같은 버전 관리 시스템입니다.

Git을 이용하여 버전 관리하기

Git이란 소스코드를 효과적으로 관리하기 위해 개발된 '분산형 버전 관리 시스템'입니다. 원래는 Linux 소스코드를 관리할 목적으로 개발 되었습니다.

Git에서는 소스 코드가 변경된 이력을 쉽게 확인할 수 있고, 특정 시점에 저장된 버전과 비교하거나 특정 시점으로 되돌아갈 수도 있습니다.

또 내가 올리려는 파일이 누군가 편집한 내용과 충돌한다면, 서버에 업로드 할 때 경고 메시지가 발생됩니다. 누군가가 애써 편집한 내용을 덮어써버리는 실수는 이제 없겠죠!



Git으로 파일을 관리하면, 업데이트 이력이 Git에 저장되지.

매번 백업용 파일 복사본을 만들 필요가 없으니까 엄청 편하고 깔끔하다구!

Git의 기본

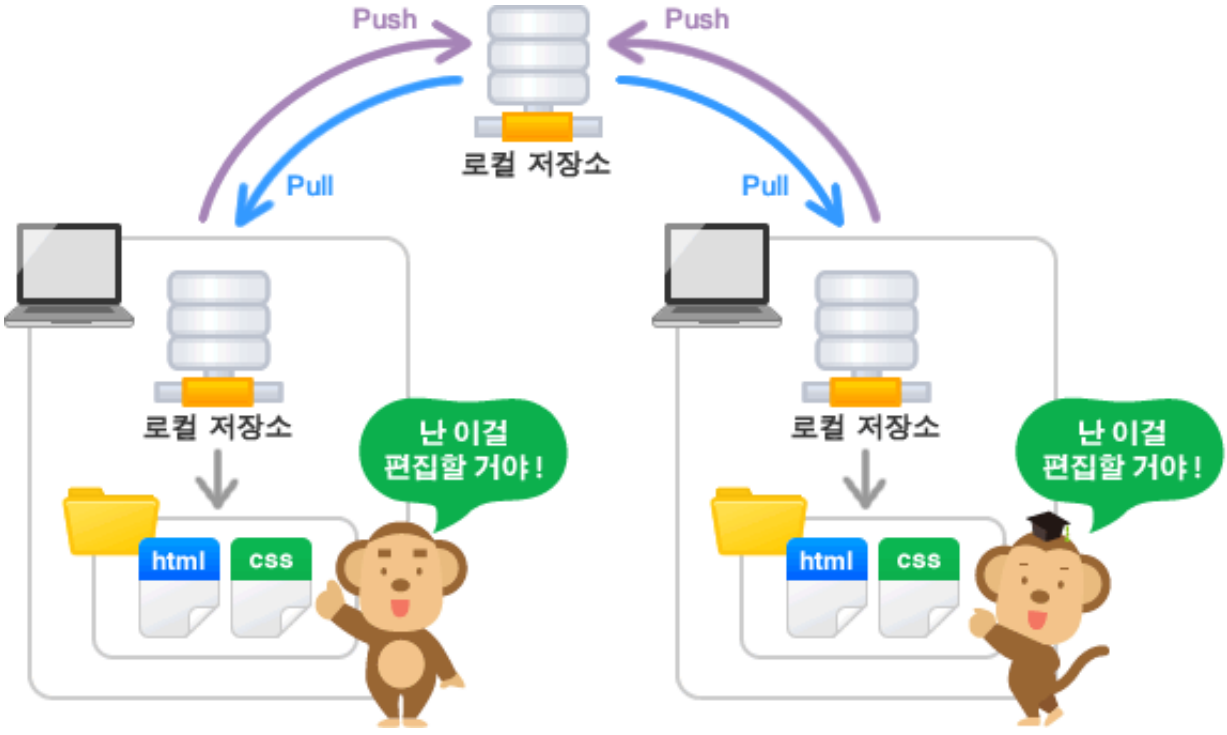
저장소(Git repository)란 말그대로 파일이나 폴더를 저장해 두는 곳입니다. 그런데 Git 저장소가 제공하는 좋은 점 중 하나는 파일이 변경 이력 별로 구분되어 저장된다는 점입니다. 비슷한 파일이라도 실제 내용 일부 문구가 서로 다르면 다른 파일로 인식하기 때문에 파일을 변경 사항 별로 구분해 저장할 수 있습니다.

원격 저장소와 로컬 저장소

Git은 원격 저장소와 로컬 저장소 두 종류의 저장소를 제공합니다.

- 원격 저장소(Remote Repository): 파일이 원격 저장소 전용 서버에서 관리되며 여러 사람이 함께 공유하기 위한 저장소입니다.
- 로컬 저장소(Local Repository): 내 PC에 파일이 저장되는 개인 전용 저장소입니다.

평소에는 내 PC의 로컬 저장소에서 작업하다가 작업한 내용을 공개하고 싶을 때에 원격 저장소에 업로드 합니다. 물론 원격 저장소에서 다른 사람이 작업한 파일을 로컬 저장소로 가져올 수도 있습니다.



저장소 만들기

내 컴퓨터에 로컬 저장소를 만드는 방법은 두 가지가 있습니다.

첫 번째, 아예 저장소를 새로 만들거나, 두번째, 이미 만들어져 있는 원격 저장소를 로컬 저장소로 복사해 올 수 있습니다.

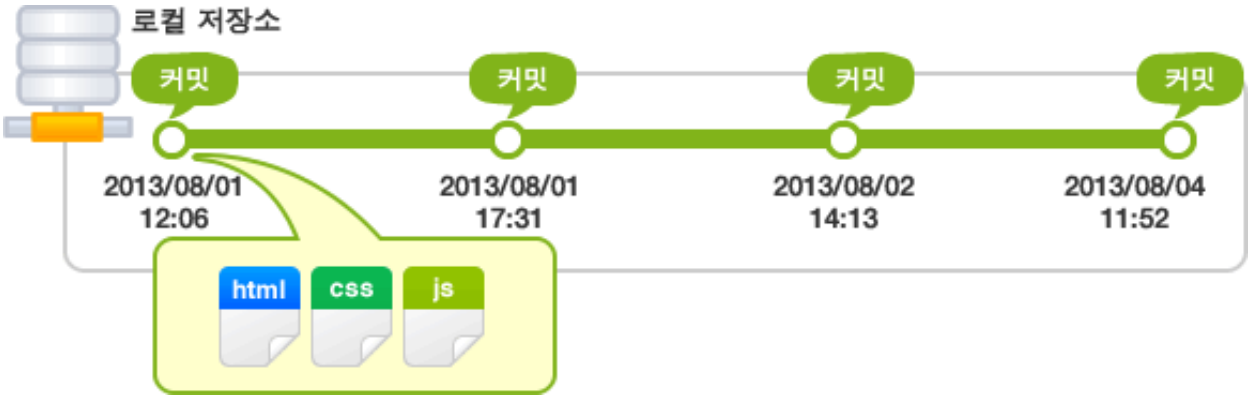
다음 페이지에서는 커밋이란걸 알려줄게!

Git의 기본

파일 및 폴더의 추가/변경 사항을 저장소에 기록하려면 '커밋'이란 버튼을 눌러줘야 합니다.

커밋 버튼을 누르면 이전 커밋 상태부터 현재 상태까지의 변경 이력이 기록된 커밋(혹은 리비전)이 만들어집니다.

커밋은 아래 그림처럼 시간순으로 저장됩니다. 최근 커밋부터 거슬러 올라가면 과거 변경 이력과 내용을 알 수 있겠죠.



각 커밋에는 영문/숫자로 이루어진 40자리 고유 이름이 붙습니다. 저장소에선 이 40자리 이름을 보고 각 커밋을 구분하고 선택합니다.

Tips

버그 수정, 기능 추가 등 특별한 의미가 있는 업데이트를 작업 별로 구분해서 각각 커밋하면, 나중에 이력을 보고 특정 변경 내용을 찾기 쉽습니다.

커밋은 이렇게 이력을 남기는 중요한 작업이기 때문에 커밋 버튼을 누를땐 커밋 메시지를 필수로 입력해야 합니다. 메시지가 없으면 커밋이 실행되지 않습니다.

Tips

메시지는 명료하고 이해하기 쉽게 남겨야 본인 뿐만 아니라 다른 사람이 커밋 이력을 확인하기 쉽습니다. Git 에서 권장하는 메시지 형식을 따르는 것도 좋습니다.

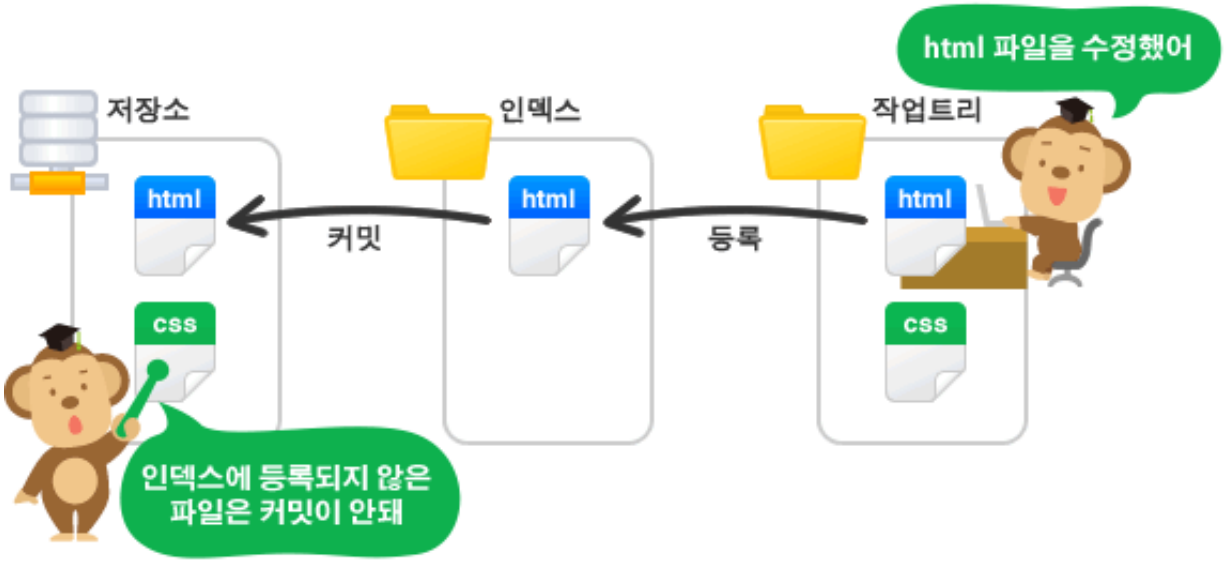
```
1번째 줄 : 커밋 내의 변경 내용을 요약
2번째 줄 : 빈 칸
3번째 줄 : 변경한 이유
```

주로 위 형식으로 메시지를 작성합니다.

Git의 기본 개념 알기

Git 에서는 우리가 흔히 말하는 폴더를 '작업 트리'(Work Tree)라고 부릅니다.

그리고 커밋을 실행하기 전의 저장소와 작업 트리 사이에 존재하는 공간을 '인덱스'라고 합니다.



지금까지 배운 Git 용어를 총정리 해 볼까요?

Git의 '커밋' 작업은 '작업 트리'에 있는 변경 내용을 저장소에 바로 기록하는 것이 아니라 그 사이 공간인 '인덱스'에 파일 상태를 기록(stage - 스테이징 한다고 표현하기도 합니다)하게 되어 있습니다. 따라서 저장소에 변경 사항을 기록하기 위해서는, 기록하고자 하는 모든 변경 사항들이 '인덱스'에 존재해야 합니다.

예를 들어, 10개의 파일을 수정했지만 그 중에 7개만 저장소에 공개하고 싶을 때를 생각해 보세요. 변경한 10개의 파일 중 7개를 선택하는 작업이 바로 '인덱스에 등록' 또는 '스테이징(stage)'이라 표현하는 작업 입니다.

이렇게 인덱스란 공간(가상이지만요!)이 중간에 있는 덕분에 작업 트리 안에 있는 커밋이 필요 없는 파일들을 커밋에 포함하지 않을 수 있고, 파일에서 내가 원하는 일부 변경 사항만 인덱스에 등록해 커밋할 수 있습니다.

자, 다음 페이지에선 지금까지 배운 내용을 직접 실습해보자! 실제로 Git을 설치해서 프로젝트에 참여해 보는거야!