Git 저장소 만들기

Git 저장소를 만드는 방법은 두 가지다. 기존 **프로젝트를 Git 저장소로 만드는 방법이 있고 다른 서버에 있는 저장소를 Clone 하는 방법이 있다.**

기존 디렉토리를 Git 저장소로 만들기

기존 프로젝트를 Git으로 관리하고 싶을 때, 프로젝트의 디렉토리로 이동해서 아래와 같은 명령을 실행한다.

$ git init

이 명령은 .git이라는 하위 디렉토리를 만든다. .git 디렉토리에는 저장소에 필요한 뼈대 파일(Skeleton)이 들어 있다.(.git 디렉토리가 막 만들어진 직후에 어떤 파일이 있는지에 대한 내용은 9장에서 다룬다.) 이 명령만으로는 아직 프로젝트의 어떤 파일도 관리하지 않는다.

Git이 파일을 관리하게 하려면 저장소에 파일을 추가하고 커밋해야 한다. git add 명령으로 파일을 추가하고 커밋한다.

$ git add \*.c

$ git add README

$ git commit -m 'initial project version'

매우 짧은 시간에 명령어를 몇 개 실행해서 Git 저장소를 만들고 파일이 관리되게 했다.

기존 저장소를 Clone 하기

**다른 프로젝트에 참여하거나(Contribute) Git 저장소를 복사하고 싶을 때 git clone 명령을 사용한다.** 이미 Subversion같은 VCS에 익숙한 사용자에게는 checkout이 아니라 clone 이라는 점이 도드라져 보일 것이다. Git이 Subversion과 다른 가장 큰 차이점은 서버에 있는 모든 데이터를 복사한다는 것 이다. git clone을 실행하면 프로젝트 히스토리를 전부 받아온다. 실제로 서버의 디스크가 망가져도 클라이언트 저장소 중에서 아무거나 하나 가져다가 복구하면 된다. (서버에만 적용했던 설정은 복구하지 못하지만 모든 데이터는 복구된다.) 4장에서 좀 더 자세히 다룬다.)

git clone [url] 명령으로 저장소를 Clone 한다. Ruby 용 Git 라이브러리인 Grit을 Clone 하려면 아래와 같이 실행한다.

$ git clone git://github.com/schacon/grit.git

디렉토리 이름이 mygrit이라는 것만 뻬면 이 명령의 결과와 앞선 명령의 결과는 같다.

Git은 다양한 프로토콜을 지원한다. 이제까지는 git:// 프로토콜을 사용했지만 http(s):// 를 사용할 수 도 있고 [user@server:/path.git](mailto:user@server:/path.git) 처럼 SSH 프로토콜을 사용할 수도 있다. 자세한 내용은 4장에서 다룬다. 4장에서는 각 프로토콜의 장단점과 Git 저장소에 접근하는 방법을 설명한다.