**1. 설정과 초기화**

전역 사용자명/이메일 구성하기

**git config - -global user.name “papa”**

**git config - -global user.email “papawon01@gmail.com”**

저장소별 사용자명/이메일 구성하기 (해당 저장소 디렉터리로 이동후)

**git config user.name “Your name”**

**git config user.email “Your email address”**

참고로 user 설정이 되어 있지 않으면 Github에 있는 repository에 변경사항을 푸시 한다고 해도 commit count 집계도 안되고 해당 커밋의 작성자 프로필 아이콘도 ? 로 표시되기 때문에 웬만하면 name과 email 주소를 설정하길 추천한다.

전역 설정 정보 조회 **git config - -global - -list**

저장소별 설정 정보 조회 **git config - -list**

Git의 출력결과 색상 활성화하기 **git config - -global color.ui “auto”**

새로운 저장소 초기화하기 **mkdir /path/newDir**

**cd /path/newDir**

**git init**

저장소 복제하기 **git clone <저장소 url>**

새로운 원격 저장소 추가하기 **git remote add <원격 저장소> <저장소 url>**

**2. 기본적인 사용법**

아래 명령어에서 [ ]는 선택적인 매개변수를 의미한다.

새로운 파일을 추가하거나 존재하는 파일 스테이징하고 커밋하기 **git add <파일>**

**git commit -m “<메시지>”**

파일의 일부를 스테이징하기 **git add -p [<파일> [<파일> [기타 파일들…]]]**

add 명령에서 Git 대화 모드를 사용하여 파일 추가하기 **git add -i**

수정되고 추적되는 파일의 변경 사항 스테이징하기 **git add -u [<경로> [<경로>]]**

수정되고 추적되는 모든 파일의 변경 사항 커밋하기 **git commit -m “<메시지>” -a**

작업 트리의 변경 사항 돌려놓기 **git checkout HEAD <파일> [<파일>]**

커밋되지 않고 스테이징된 변경 사항 재설정하기 **git reset HEAD <파일> [<파일>]**

마지막 커밋 고치기 **git commit -m “<메시지>” - -amend**

이전 커밋을 수정하고 커밋 메시지를 재사용하기 **git commit -C HEAD - -amend**

**3. 브랜치**

지역 브랜치 목록 보기 **git branch**

원격 브랜치 목록 보기 **git branch -r**

지역과 원격을 포함한 모든 브랜치 목록 보기 **git branch -a**

현재 브랜치에서 새로운 브랜치 생성하기 **git branch <새로운 브랜치>**

다른 브랜치 체크아웃하기 **git checkout <브랜치>**

현재 브랜치에서 새로운 브랜치 생성하고 체크아웃하기 **git checkout -b <새로운 브랜치>**

다른 시작 지점에서 브랜치 생성하기 **git branch <새로운 브랜치> <브랜치를 생성할 위치>**

기존의 브랜치를 새로운 브랜치로 덮어쓰기 **git branch -f <기존 브랜치> [<브랜치를 생성할 위치>]**

브랜치를 옮기거나 브랜치명 변경하기 **git checkout -m <기존 브랜치> <새로운 브랜치>**

* <새로운 브랜치>가 존재하지 않을 경우   
  **git checkout -M <기존 브랜치> <새로운 브랜치>**
* 무조건 덮어쓰기

다른 브랜치를 현재 브랜치로 합치기 **git merge <브랜치>**

커밋하지 않고 합치기 **git merge - -no-commit <브랜치>**

선택하여 합치기 **git cherry-pick <커밋명>**

커밋하지 않고 선택하여 합치기 **git cherry-pick -n <커밋명>**

브랜치의 이력을 다른 브랜치에 합치기 **git merge - -squash <브랜치>**

브랜치 삭제하기 **git branch -d <삭제할 브랜치>**

* 삭제할 브랜치가 현재 브랜치에 합쳐졌을 경우에만

**git branch -D <삭제할 브랜치>**

* 삭제할 브랜치가 현재 브랜치에 합쳐지지 않았어도

**4. Git 이력**

모든 이력 보기 **git log**

변경 사항을 보여주는 패치와 함께 로그 표시하기 **git log -p**

1개의 항목만 보이도록 로그 개수 제한하기 **git log -1**

20개의 항목과 패치만 보이도록 로그 제한하기 **git log -20 -p**

6개월 동안의 커밋 로그 보기 **git log - -since=”6 hours”**

이틀 전까지의 커밋 로그 보기 **git log - -before=”2 days”**

HEAD보다 세 개 이전의 커밋 로그 보기 **git log -1 HEAD-3**

**git log -1 HEAD^^^**

**git log -1 HEAD~1^^**

두 지점 사이의 커밋 로그 보기

**git log <시작 지점>…<끝 지점>**

* 시작 지점이나 끝 지점은 커밋명, 브랜치명, 혹은 태그명이 될 수 있고 조합하여 사용 가능하다.

각 항목의 로그 이력 한 줄씩 보기 **git log - -pretty=oneline**

각 항목마다 영향 받은 줄의 통계 보기 **git log - -stat**

커밋할 시점의 파일 상태 보기 **git log - -name-status**

현재 작업 트리와 인덱스의 차이점 보기 **git diff**

인덱스와 저장소의 차이점 보기 **git diff - -cached**

작업 트리와 저장소의 차이점 보기 **git diff HEAD**

작업 트리와 특정 위치 간의 차이점 보기 **git diff <시작 지점>**

* 시작 지점은 커밋명 or 브랜치명 or 태그명이다.

저장소의 두 지점 사이의 차이점 보기 **git diff <시작 지점> <끝 지점>**

차이점의 통계 보기 **git diff - -stat <시작 지점> [<끝 지점>]**

파일의 커밋 정보 줄 단위로 보기 **git blame <파일>**

파일의 줄 단위의 복사, 붙여 넣기, 이동 정보 보기 **git blame -M <파일>**

파일의 줄 단위의 이동과 원본 파일 정보 보기 **git blame -C -C <파일>**

로그에서 복사와 붙여 넣은 정보 보기 **git log -C -C -p -1 <특정 지점>**

**5. 원격 저장소**

저장소 복제하기 **git clone <저장소>**

마지막 200개의 커밋만 포함하여 저장소 복제하기 **git clone - -depth 200 <저장소>**

새로운 원격 저장소 추가하기 **git remote add <원격 저장소> <저장소 url>**

모든 원격 브랜치 목록 보기 **git branch -r**

원격 브랜치에서 지역 브랜치 생성하기 **git branch <새로운 브랜치> <원격 브랜치>**

원격 태그에서 지역 브랜치 생성하기 **git branch <새로운 브랜치> <원격 태그>**

origin 저장소에서 합치지 않고 지역 브랜치로 변경 사항 가져오기 **git fetch**

원격 저장소에서 합치지 않고 지역 브랜치로 변경 사항 가져오기 **git fetch <원격 저장소>**

원격 저장소에서 변경 사항을 가져와 현재 브랜치에 합치기 **git pull <원격 저장소>**

origin 저장소에서 변경 사항을 가져와 현재 브랜치에 합치기 **git pull**

지역 브랜치를 원격 브랜치에 푸싱하기 **git push <원격 저장소> <지역 브랜치>:<원격 브랜치>**

지역 브랜치를 동일한 이름의 원격 브랜치에 푸싱하기 **git push <원격 저장소> <지역 브랜치>**

새로운 로컬 브랜치를 원격 저장소에 푸싱하기 **git push <원격 저장소> <지역 브랜치>**

원격 저장소에서 쓸모가 없어진 원격 브랜치 제거하기 **git remote prune <원격 저장소>**

원격 저장소를 제거하고 관련된 브랜치도 제거하기 **git remote rm <원격 저장소>**