GIT

1. Stage.

* Commit 전 각 변경 사항들의 이력 들은 index 파일에 저장
* 변경 사항을 기록하는 것 Staging
* Stage(Commit에 반영) UnStage(Commit에 미반영).

1. HEAD

* 현재 작업 BRANCH가 어디인지 표시한다.

1. Commit

* 변경 사항들의 집합

1. Branch

* 특정 Commit으로부터 분기되는 포인터. 생성 개수의 제한 X

1. CheckOut

* A Commit에서 B Commit으로 갈아타기.
* 이전 시점으로 돌아 갈수도, 다른 Branch의 Commit으로 가기도 가능.

1. Merge

* Branch를 병합.
* 주의사항. Main Branch로 CheckOut한 후 수정된 Branch를 Merge해야 오류가 나지 않음
* Fast-forward 방식. Conflict 없이 자동 merge.
* Non-fast-forward 방식. Conflict 발생하여 오류를 해결 후 Merge 하는 것.   
  (추가 Commit 필요)

1. Clone.

* Remote Repo -> Local Repo Download

1. Push

* Local Repo -> Remote Repo Upload

1. Pull

* Remote Repo -> Local Repo Download(변경 사항만)
* 기존 코드와 수정된 코드가 다를 경우 Merge 진행.

1. Stash

* 변경 사항 임시 저장. Commit과 다르게 현재 수정 중인 것들을 임시 백업 시 사용
* <https://gmlwjd9405.github.io/2018/05/18/git-stash.html> 참고

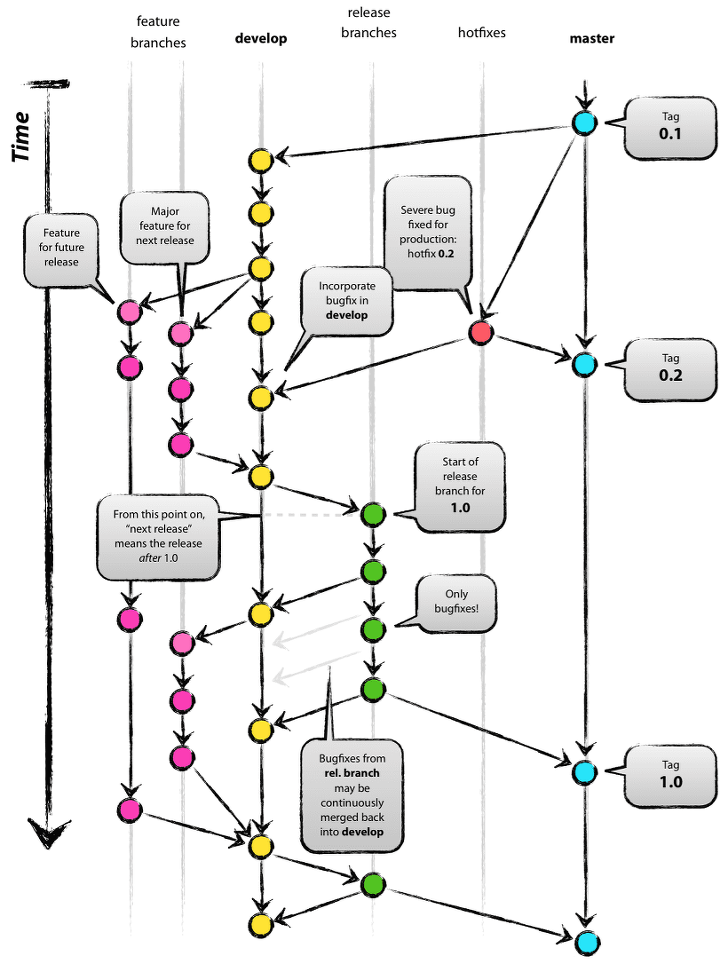
1. Pull Request(PR)

* Push 하고 나면 Compare & pull request 버튼이 활성화 됨.
* 무엇을 수정 했는지? 수정 전 코드가 남아있진 않은지. 주석은 작성했는지.
* Create Pull Request
* Merge OR Cancel
* 이후 Branch 삭제

1. Fork.

* 타겟 Proj의 Repo를 자신의 Repo로 가져온다

Git Flow



1. Master. Client가 실제 사용하는 Branch
2. Develop. 현재 개발중인 Main Branch.
3. HotFix. 긴급 오류 수정용 Branch
4. Feature. 추가 기능 개발중인 Sub Branch.
5. Release. 배포 전 테스트용 Branch. Develop이나 Master 병합용

출처 : https://cupjoo.tistory.com/6