Git 스터디 1주차

23.01.12

스터디 방식??

1. 깃 공부 & 고급 레벨까지 ?
   - https://git-scm.com/book/ko/v2
   - https://www.udemy.com/course/best-git-github/

2. 깃허브 기능 써보기
   a. Git 공식 홈페이지

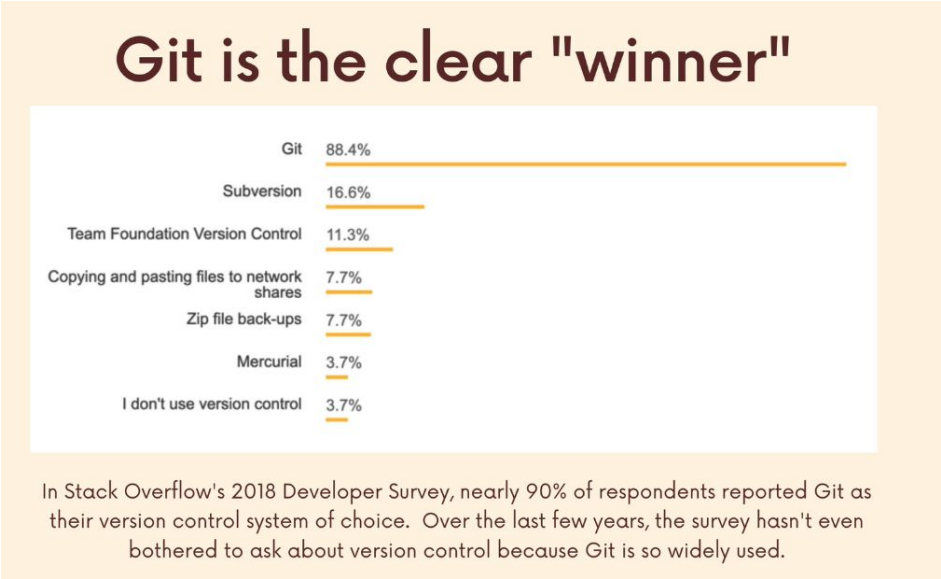
3. 깃 관련 아티클 선정해서 같이 보기

Git이란 무엇일까요?

전세계에서 가장 인기있는 VCS

Subversion, CVS, Mercurial 2018년 Stackoverflow 자료 조사

2019년부터는 95%이상이 사용



Git Core


Intro to Git
Merging
Fetching & Pulling
Git Tags
Installation
Diffing
Github Odds & Ends
Git Behind The Scenes
Git Basics
Stashing
Collaborative Workflows
Reflogs
Committing In Detail
Undoing Changes
Rebasing
Custom Aliases
Branching
Github Intro
Interactive Rebasing
???

Next Level Git

Github / Collab Core

Git other part

4: Git의 기초: 추가하기와 커밋하기
| 51분

5: 커밋과 관련 주제 자세히 알아보기
| 51분

6: 브랜치(branch)로 작업하기
| 1시간

7: 브랜치 병합하기, 맙소사!
| 56분

8: Git Diff로 변경사항 비교하기
| 53분

9: 스태시(Stash)의 모든 것
| 35분

10: 변경사항 취소하기 및 시간 여행
| 59분

11: Github의 기초
| 1시간 33분

12: Fetch와 Pull
| 54분

13: Github의 이모저모: 잡다한 지식
| 1시간 2분

14: Git 협업 워크플로우
| 1시간 44분

15: 리베이스(Rebase)는 가장 까다로운 Git 명령어일까?
| 40분

16: Interactive Rebase를 사용하여 히스토리 삭제하기
| 26분

17: Git tag: 히스토리상의 중요한 순간에 표시하기
| 38분

18: Git의 이면 – 해싱(Hashing)과 객체
| 1시간 19분

19: Reflog의 힘 – '사라진' 작업 복구하기
| 44분

아무도 알려주지 않는 커밋 명령어

하나의 명령으로 추가와 커밋을 한번에

git에서 가장 많이 사용되는 명령 중 하나는 git add 다음으로 git commit 입니다.

```
git add .
```

```
git commit -m "커밋 메시지"
```

대부분의 경우 변경된 모든 파일을 추가하려고 합니다.
두 명령을 모두 작성하는 대신 -am 플래그 를 사용하여 두 명령을 하나로 결합할 수 있습니다 .
이 플래그는 h git add . 와 git commit을 모두 수행합니다.

```
git commit -am "커밋 메시지"
```

출처: https://medium.com/design-bootcamp/git-commands-nobody-has-told-you-cd7025bea8db

다른 지점에서 변경 사항 복사

두 개 이상의 브랜치에 변경 사항을 추가해야 하는 몇 가지 시나리오가 있습니다.
예를 들어 두 가지 버전이 있고 두 버전을 모두 지원하는 경우
두 브랜치에 대한 변경 사항을 커밋해야 합니다.

branchA 와 branchB라는 두 가지 branch를 가집니다.
두 브랜치에서 수동으로 커밋하는 대신 git rebase 명령을 사용할 수 있습니다.

```
git checkout branchA
```

```
git rebase branchB
```

일어날 일은 branchA 가 branchB 에서 분기된 것처럼 보일 것 입니다.

마지막 커밋에 변경 사항 추가

우리 모두는 작은 변경사항 하나를 잊고
새로운 커밋을 해야 하는 상황에 처해 있습니다.
이 변경 사항이 그다지 크지 않은 경우 —amend 플래그 를 사용하여 마지막 커밋에 추가할 수 있습니다 .

```
git add .
```

```
git commit --amend --no-edit
```

— no-edit 플래그를 사용 하면
커밋 메시지를 수정하지 않고 마지막 커밋에 변경 사항을 적용할 수 있습니다.

커밋에서 파일 제거

브랜치에 커밋된 특정 파일을 제거하려면 git reset 명령을 사용할 수 있습니다.

```
git reset --soft HEAD^
```

이렇게 하면 커밋된 파일이 스테이징 영역으로 이동한 다음 제거할 파일을 정확히 지정할 수 있습니다.

```
git reset HEAD <filename>
```

버그가 있는 커밋 찾기

버그가 생겨서 이 버그가 발생하기 정확히 언제, 무엇이 변경되었는지 검색해야 하는 상황에 처한 적이 있습니까?
이 명령을 알고 있다면 이 과정이 더 빠르고 쉬울 것입니다.
git bisect를 사용하면 먼저 버그가 있는 "나쁜" 커밋과
버그가 없는 "좋은" 커밋을 알려줌으로써 버그를 생성하는 커밋을 검색할 수 있습니다.

```
git bisect start
```

```
git bisect bad
```

```
git bisect good v.11.0.1-rc2
```

완료되면 git bisect reset 을 사용 하여 상태를 정리하고 원래 HEAD로 돌아가야 합니다.

```
git bisect reset
```

그래프처럼 기록 보기

git 기록을 그래프처럼 보고 싶다면 명령 하나로 쉽게 수행할 수 있습니다.

```
git log --all --decorate --oneline --grap
```

git log를 매일 사용하지는 않겠지만 명령을 기억하는 데 사용할 수 있는 쉬운 약어가 있습니다.

“A DOG” = git log — All — Decorate — Oneline — Graph

특정 파일이 변경된 커밋 표시

특정 파일에 대한 모든 변경 사항을 확인하려는 경우가 있으며

이는 git log with — follow 플래그를 사용하여 수행할 수 있습니다.

```
git log --follow -- <filename>
```

Git 관련 다른 아티클들

- Github 흐름을 개선하기 위한 15가지 팁
- 궁극의 Github 협업 가이드
- 소프트웨어 개발자로서 Git/GitHub와 효과적으로 협업하는 방법
- ....

Github 서비스 살펴보기

GitHub Docs

GitHub 여정 중 어디에 있든 도움이 됩니다.

Get started

- Get started
- Account and profile
- Authentication
- Billing and payments
- Site policy

Collaborative coding

- GitHub Codespaces
- Repositories
- Pull requests
- GitHub Discussions
- GitHub Copilot

CI/CD and DevOps

- GitHub Actions
- GitHub Packages
- GitHub Pages

Security

- Code security
- Supply chain security
- Security advisories
- Dependabot
- Code scanning
- Secret scanning

Client apps

- GitHub CLI
- GitHub Desktop

Project management

- GitHub Issues
- Search on GitHub

Developers

- Developers
- REST API
- GraphQL API

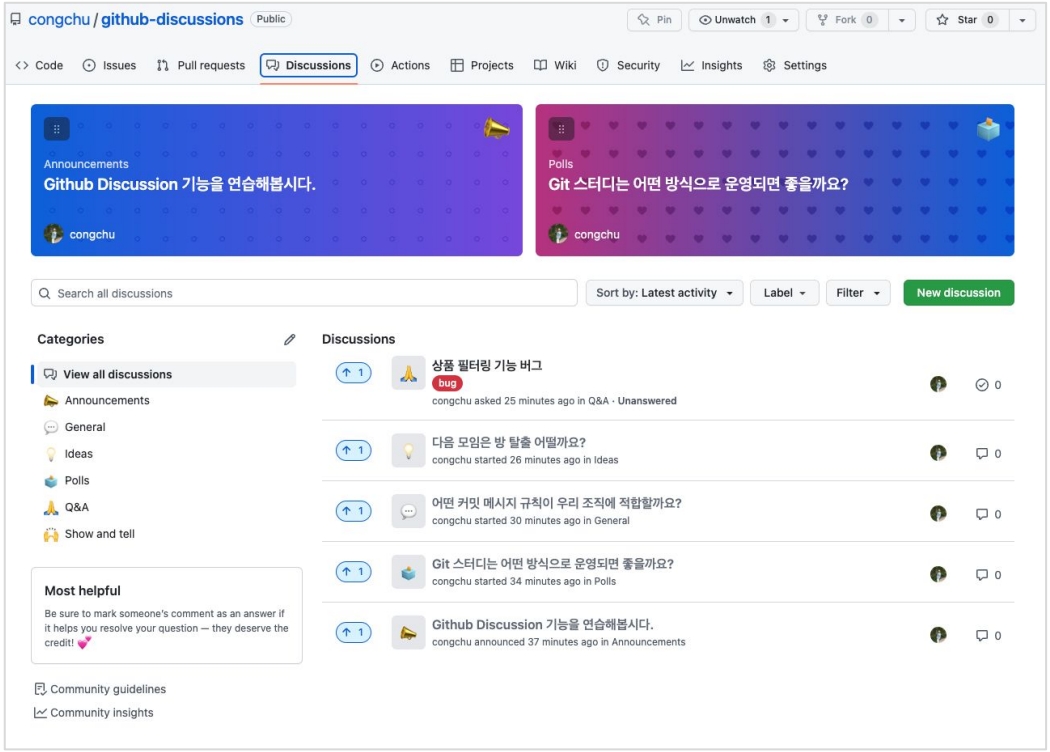
Enterprise and Teams

- Organizations
- Enterprise administrators

Community

- Building communities
- GitHub Sponsors
- Education
- GitHub Support

Github Discussions



GitHub 토론은
GitHub의 리포지토리 또는 조직에 대한
유지 관리자와 커뮤니티 간의
대화를 위한 공개 포럼입니다.

https://github.com/congchu/github-discussions/discussions