

Version Control - Git 2

A Basic CS Skill, ABC Winter School

박원

Objective

- Git Commit 규칙
- GitHub issue
- Pull Request와 코드리뷰
 - fork, pull request, code review, merge

Git Commit

fix error



circle-oo committed on 20 Jan 2021

fixed



circle-oo committed on 7 Nov 2015

Git Commit


- 거의 같은 커밋 메시지이지만
- 날짜도, repository도 다릅니다.
- 무엇이 수정되었는지 알 수 없습니다.

Git Commit

Add Pollard's rho algorithm

 circle-oo committed on 26 Nov 2021


Add Miller-Rabin primality test

 circle-oo committed on 26 Nov 2021


Add Stoer-Wagner algorithm to get global min-cut

 circle-oo committed on 26 Nov 2021

Add min cost max flow algorithm

 circle-oo committed on 26 Nov 2021

Add dinic algorithm

 circle-oo committed on 26 Nov 2021

- 이 경우는 적어도 뭐가 추가되었는지는 알 수 있습니다.

Git Commit Rules

- 앞에서 보았듯이, 대충 `fix error`, `add code` 이런식으로 커밋을 작성하면
 - 뭐가 추가되고 수정된건지 전혀 알 수 없다!
- 이러면 협업을 하는데에 있어서 생산성이 떨어진다.
 - 커밋메세지만 보고 바뀐점을 알아채는 것 보다
 - 직접 코드를 뜯어보는게 시간이 더 오래걸리기 때문

Git Commit Rules

- 커밋 메시지를 규칙에 맞게 잘 작성하면 아래 3가지를 얻을 수 있습니다.
 - 더 좋은 커밋 로그 가독성
 - 더 나은 협업과 리뷰 프로세스
 - 더 쉬운 코드 유지보수

Git Commit Rules

- Git Commit 규칙은 7가지가 있어요.
 - 제목과 본문을 빈 행으로 구분
 - 제목은 최대 50자
 - 제목 첫글자는 대문자로
 - 마침표 넣지 않기
 - 제목은 항상 명령문으로
 - 본문의 각 행은 최대 72자
 - focused on **what** and **why**

Git Commit Rules

- 커밋 메시지 구조는 이렇습니다.

```
(type): title
```

```
body(생략가능)
```

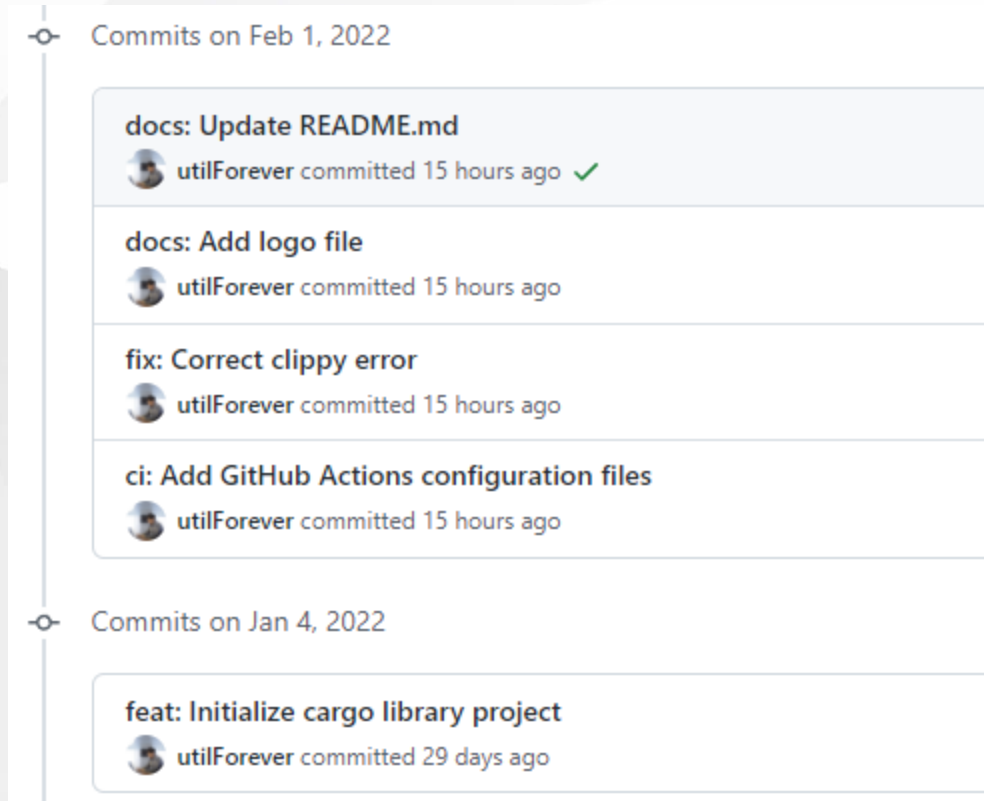
```
Resolves : #issue, ... (해결한 이슈, 생략 가능)
```

```
See also : #issue, ... (참고 이슈, 생략 가능)
```

Git Commit Rules

- 커밋 유형에는
 - feat: 새로운 기능의 추가
 - fix: 버그 수정
 - docs: 문서 수정
 - style: 스타일 관련 (코드 포매팅 등, 코드내용의 변경은 없는경우)
 - refactor: 코드 리팩토링
 - test: 테스트 코드, 리팩토링 테스트 코드 추가
 - chore: 빌드 업무 수정, 패키지 매니저 수정(ex. .gitignore)

Git Commit Rules



GitHub Issue

- Issue?
 - 프로젝트를 진행하면서 발생하는 모든 이슈
 - 버그, 풀 리퀘스트, 기능 추가 요구 등

Fork

- 브랜치 vs 포크
- 브랜치
 - 하나의 원격 저장소에서 분기를 나눔
 - 하나의 원본 저장소에서 코드 커밋 이력을 볼 수 있으나
 - 다수의 사용자가 다수의 브랜치를 만들면 관리가 힘들
- 포크
 - 여러 원격저장소를 만들고
 - 코드 수정이 더 자유롭다.

Pull Request and code review

Pull Request

- 여러 브랜치 또는 forked 저장소가 있다.
- 당연히 수정한 것을 서비스에 적용하려면 메인브랜치에 합쳐줘야한다.
- 이때 여러 브랜치를 합쳐달라고 요청하는 것을 Pull Request라고 한다.
 - 대상 브랜치에서, merge 될 브랜치를 pull하는 것을 요청
- 함부로 합쳤다가 수정한 내용이 겹쳐서 충돌이 일어나거나, 무분별한 merge를 막는다.

Pull Request

Convert OOP-based code to ECS #18

Edit <> Code

Merged utilForever merged 68 commits into `main` from `ecs` on 18 Jul 2021

Conversation 45 Commits 68 Checks 22 Files changed 43

+48,573 -903



utilForever commented on 17 Jul 2021

Owner

This revision includes:

- Convert OOP-based code to ECS (✓ [Convert OOP-based code to ECS #17](#))
 - Add library `entt`
 - Convert classes to components using `struct`
 - Refactor related code

utilForever and others added 30 commits 7 months ago

- feat(ecs): Add library 'entt'
- build: Add library 'entt' to 'include_directories'
- feat(ecs): Create struct 'Name'
- feat(ecs): Create class 'Game'
- feat(ecs): Delete class 'ItemLoader'
- feat(ecs): Delete class 'ItemManager'
- feat(ecs): Delete class 'Item'

6d01ba1

✓ 816b4ce

94bf0c4

8fb52be

ef84cf9

3f06236

h06e0e8

Reviewers

circle-oo

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked issues

Successfully merging this pull request may close these issues.

✓ [Convert OOP-based code to ECS](#)

Code Review

- Pull Request를 처리하는 과정에서 남길 수 있다.
- Pull Request가 올라오면
 - 리퀘스트를 놓지 않은 다른 사람이 PR의 모든 코드를 확인하고
 - 피드백을 남기거나, 의문점 등의 코멘트를 남긴다.
 - 그리고 피드백이 반영되고 의문점이 모두 해결되면 merge!



circle-oo reviewed on 18 Jul 2021

[View changes](#)

Includes/PokeMaster/Helpers/PokemonHelpers.hpp

Hide resolved

```
75 + /// \param registry A registry of the entity-component system.  
76 + /// \param entity A pokemon entity.  
77 + /// \return The types of the pokemon.  
78 + std::tuple<Type, Type> GetTypes(entt::registry& registry, entt::entity entity);
```



circle-oo on 18 Jul 2021

Collaborator



Is `std::tuple<>` better than `std::pair<>` ?



utilForever on 18 Jul 2021

Owner

Author



`std::tuple<>` is generalized type of `std::pair<>` . `std::pair<>` stores two types only. However, `std::tuple<>` can store arbitrary number of types as you wish.



Reply...

Unresolve conversation

utilForever marked this conversation as resolved.

Questions?

Next

LaTeX

Install LaTeX

- <https://www.tug.org/texlive/>
- <https://marketplace.visualstudio.com/items?itemName=James-Yu.latex-workshop>