**Dimenshift**

**Git & 작업 프로세스 정의**

1. Git 기본 사용법 안내

ㆍRepository 관리법(커밋, 푸시, 풀, 브랜치 사용법)

1. 커밋

본인 작업을 완료한 후 스테이지 하여 제목을 입력 후

만들어져있는 팀 리포지토리에 커밋하여 업로드한다.

작업 프로그램에 적용을 위해 팀원 끼리의 버전을 통일하고, 충돌을 막기 위해 인게임 코드를 변경 할 때에는 순서를 정하여 커밋한다.

1. 푸시

본인 작업물을 커밋한 후 push를 눌러 리포지토리 맨 위로 올려 같은 작업자들에게 공유한다.

1. 풀

리포지토리에 공유되어있는 작업물(push상태)을 가지고 작업을 하기 위해 본인이 작업하는 프로젝트에 가장 최신의 작업물을 가져와 작업을 한다.

1. 브랜치 네이밍 규칙

ㆍfeature/[기능명] : 새로운 기능 추가하였을 때

ㆍbugfix/[이슈번호] : 버그를 수정하였을 때

ㆍtest/[기능명] : 기능을 테스트할 때

1. 커밋 메시지 규칙 정의

작업물 추가 시 : [Feat] Program / UI / Character / Map / Planning 으로

본인 파트를 구분하여 커밋 메시지를 작성하였고 처음 추가를 할 때는 무조건 맨 뒤에 “1차”를 붙여 나중에 수정할 시 n차로 이어지도록 작성 하였습니다.

작업물 수정 시 : 맨 앞 [Fix]를 붙이고 추가 시와 똑같이 파트를 구분하여 커밋 메시지를 작성하였습니다. 또한 수정한 회차에 따라 2차, 3차등의 차수를 작성해 구분하도록 하였습니다.

1. 작업 프로세스 정의

ㆍGit Flow

1. 작업 전에 main branch에서 pull 하여 가장 최신의 작업물을 불러온다.
2. 작업에 맞는 브랜치를 생성한다. ( ex: bugfix/fight )
3. 작업 종료 후 양식을 지키고 커밋 한 후 push를 한다.
4. pull request를 요청하고 확인한다.
5. 승인을 받고 main branch에 merge 한다.
6. 협업 가이드라인 및 예외상황 처리법

* 충돌이 발생할 경우

작업이 간단한 작업물을 다시 작업 후 재커밋, 다시 작업하지 않은 작업물을 선택해 우선 push한다.

* 커밋 메시지

커밋 메시지의 양식을 무조건 지키며 한눈에 알아보기 쉽게 작성한다.

* 충돌 방지

충돌을 방지하기 위해 작업 할 시에 작업 여부를 팀원에게 물어보고 없을 시 바로 작업에 들어가고, 작업 하는 인원이 있을 시 그 인원의 작업이 끝난 뒤에 작업을 수행한다.