캡스톤 디자인
중간발표
(리듬액션 게임 제작)

팀_45

목차

- 기본적인 용어
- 프로젝트 제안배경 및 목적
- 기획의도
- 레퍼런스 벤치마킹
- 개발과정
- 게임구성 (기획 및 개발내용 Game Design)
- 프로젝트의 차별성 및 강점
- 기술적 어려움들
- 차후 프로젝트 진행

들어가기 앞서 - 기본적 용어

- BPM: beat per minute 1분 동안 얼마나 많은 beat가 반복되는가
- Note: 게임 중 처리 해야 하는 요소들(적, 스테이지 기물, 발판 등)
  [이에 관련된 것을 제작하는 과정을 채보를 만든다고 합니다.]
- Beat / 박자 : 규칙적으로 반복되는 간격, 패턴
- Stage : 실제 인 게임 내에서 플레이어가 상호작용하는 요소들

들어가기 앞서 - 기본적 용어

- Scene : 게임을 구성하는 여러가지 화면들
- 타이밍바 : 플레이어에게 박자를 알 수 있도록 만든 장치
- Ex)
- 60 BPM = 1분 동안 60번 1박자(4beat 간격)가 진행된다. = 1박은 1초에 한번
- 60BPM에서 4beat(1박자) 노트가 1분 동안 60번 나올 수 있다.

프로젝트 제안배경 및 목적

- 예술에 관련된 게임
- 실제로 서비스가 가능하고 경쟁력이 있을 만 한 게임
- BM 모델이 확실한 게임

프로젝트 제안배경 및 목적

- 최종목적
  - 멀티 플랫폼으로 플레이 가능한 독특한 플레이 방법을 가진 게임의 완성

- 프로젝트 기간동안 목표
  - 플레이 가능할 정도의 게임 완성

- 프로젝트 방향성
  - 다른 직군 들과의 협업을 고려한 개발
    ex) 노트 채보(Mapper), 음악, 스토리, 그림 등의 아트
  - 다른 게임과는 차별성이 있는 게임의 플레이 방식

기획의도

- 왜 리듬게임을 고르게 되었는가

변화 대응 쉬움

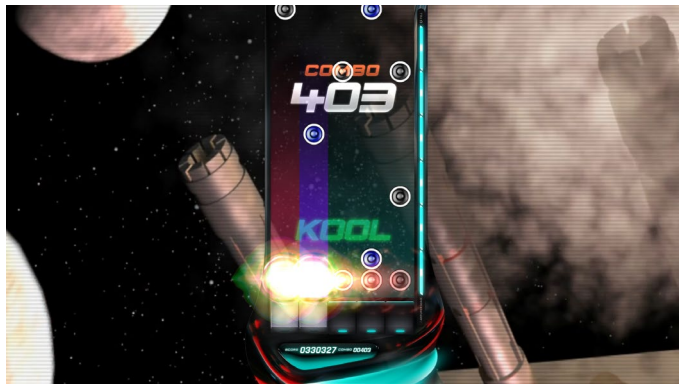
빠른 프로토타이핑
(쉬운 알파 테스트)

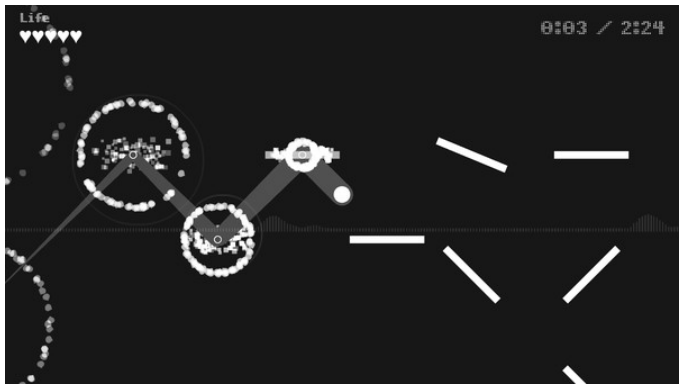
게임 심의

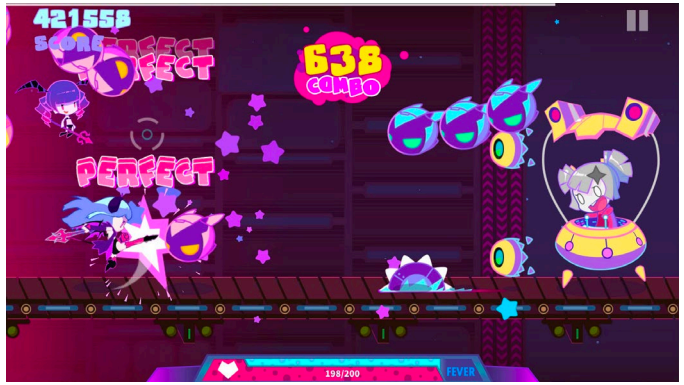
분업

명확한 이용자층

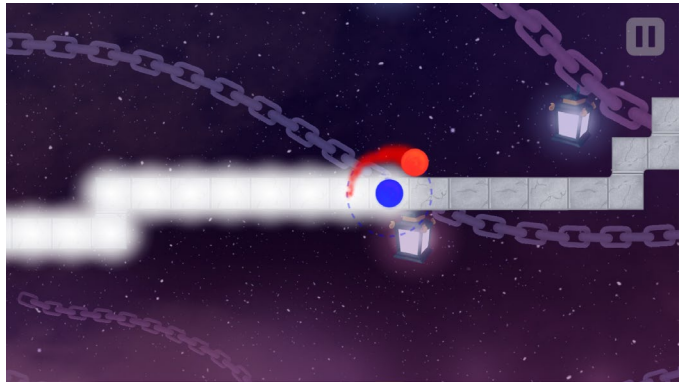
레퍼런스 벤치마킹



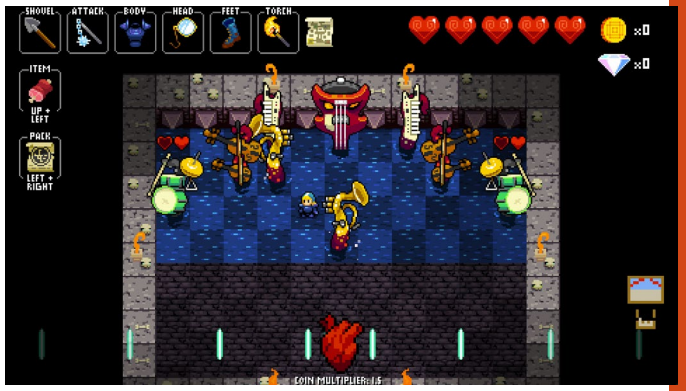




노트를 처리하는 형태

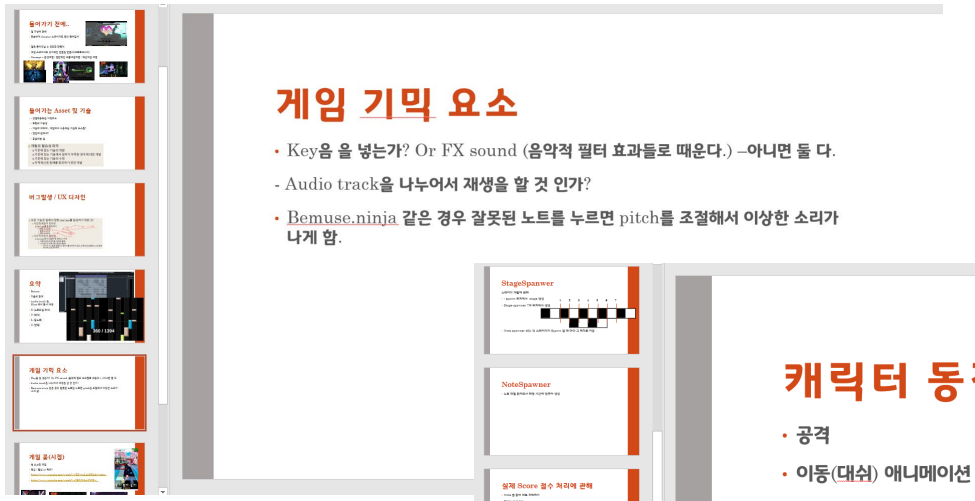


캐릭터가 목표지점으로 이동



복합형

개발과정 - 게임기획단계

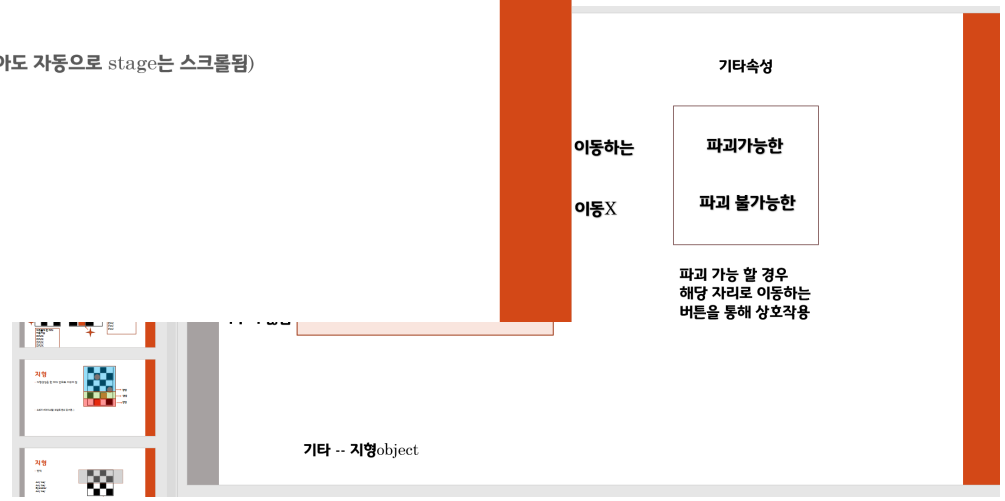


게임 기믹 요소

- Key음 을 넣는가? Or FX sound (음악적 필터 효과들로 때운다.) –아니면 둘 다.
- Audio track을 나누어서 재생을 할 것 인가?
- Bemuse.ninja 같은 경우 잘못된 노트를 누르면 pitch를 조절해서 이상한 소리가 나게 함.

캐릭터 동작

- 공격
- 이동(대쉬) 애니메이션 스피드 (60 /BPM) 이내로 (DFJK)
- 점프 (space)
- 피격 – 카메라 흔들기 (피격을 받아도 자동으로 stage는 스크롤됨)
- 통상(idle)



개발과정 - 진행단계

- 1. 게임 기본 기믹 제작

- 박자에 맞추어 점수 내보기

- 2. 메인 기믹(1)을 추가해 prototype 완성 (<- 현재의 단계)

- 스테이지의 이동, 생성 및 캐릭터 이동

- 3. 비주얼 적 요소, 부가적인 기믹, 이벤트(카메라 무브, 게임 이펙트) 들 추가

- 플레이 가능한 게임 완성

- 4. 스토리 및 컨셉, 디자인 등 게임의 외적 요소 완성

- 5. 게임의 기술적인 문제(후술할 딜레이 문제 등) 완성해 게임 완성도 높이기

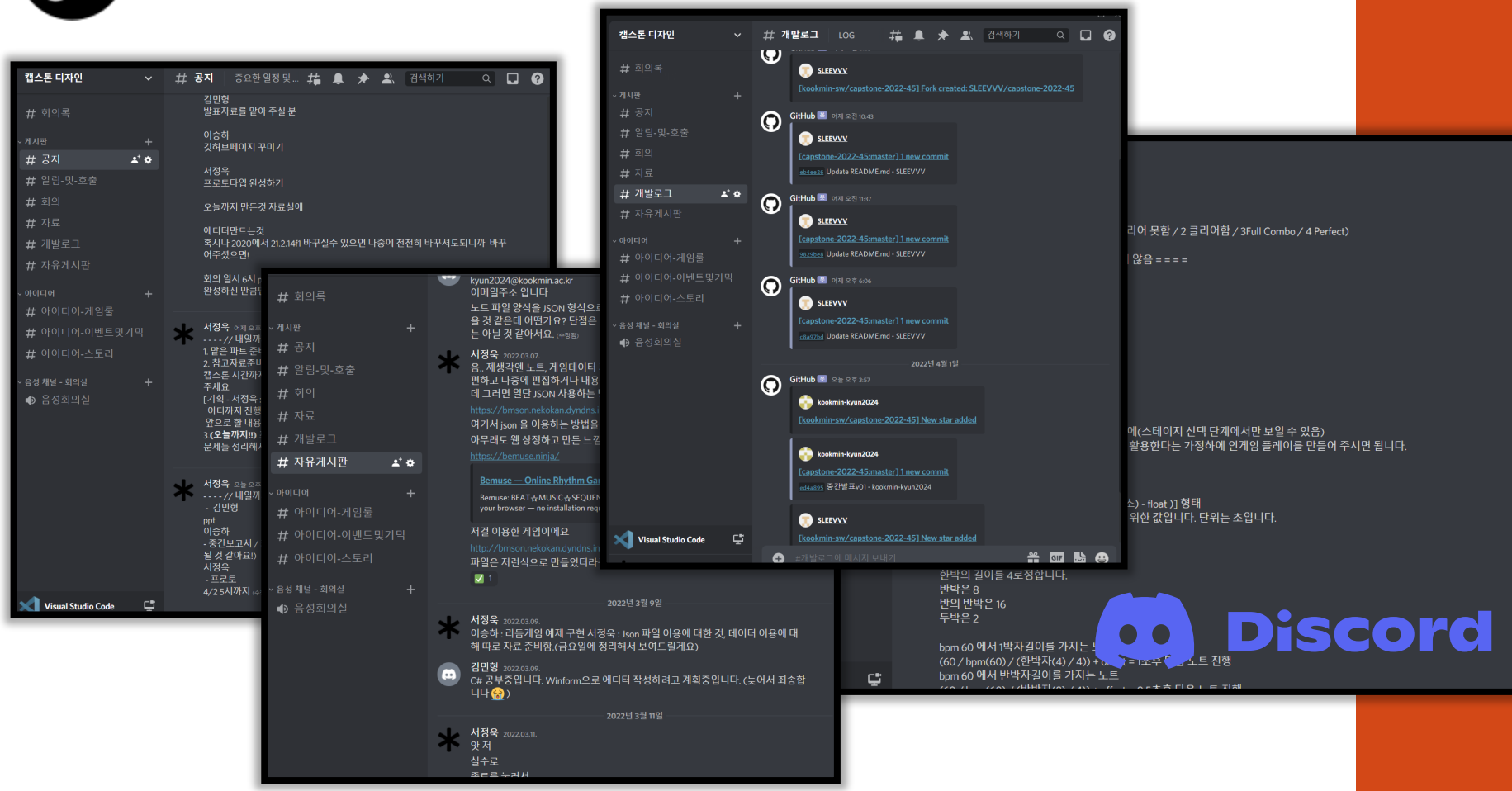
(1)기믹(Gimmick) : 게임의 몬스터, 함정, 퍼즐 리듬게임의 특이한 노트배열, 채보(Stage를 구성하는 노트파일 등)
에서 나타나는 패턴이나 진행 매커니즘

개발과정 - 사용 툴 및 환경

- PC(Steam)

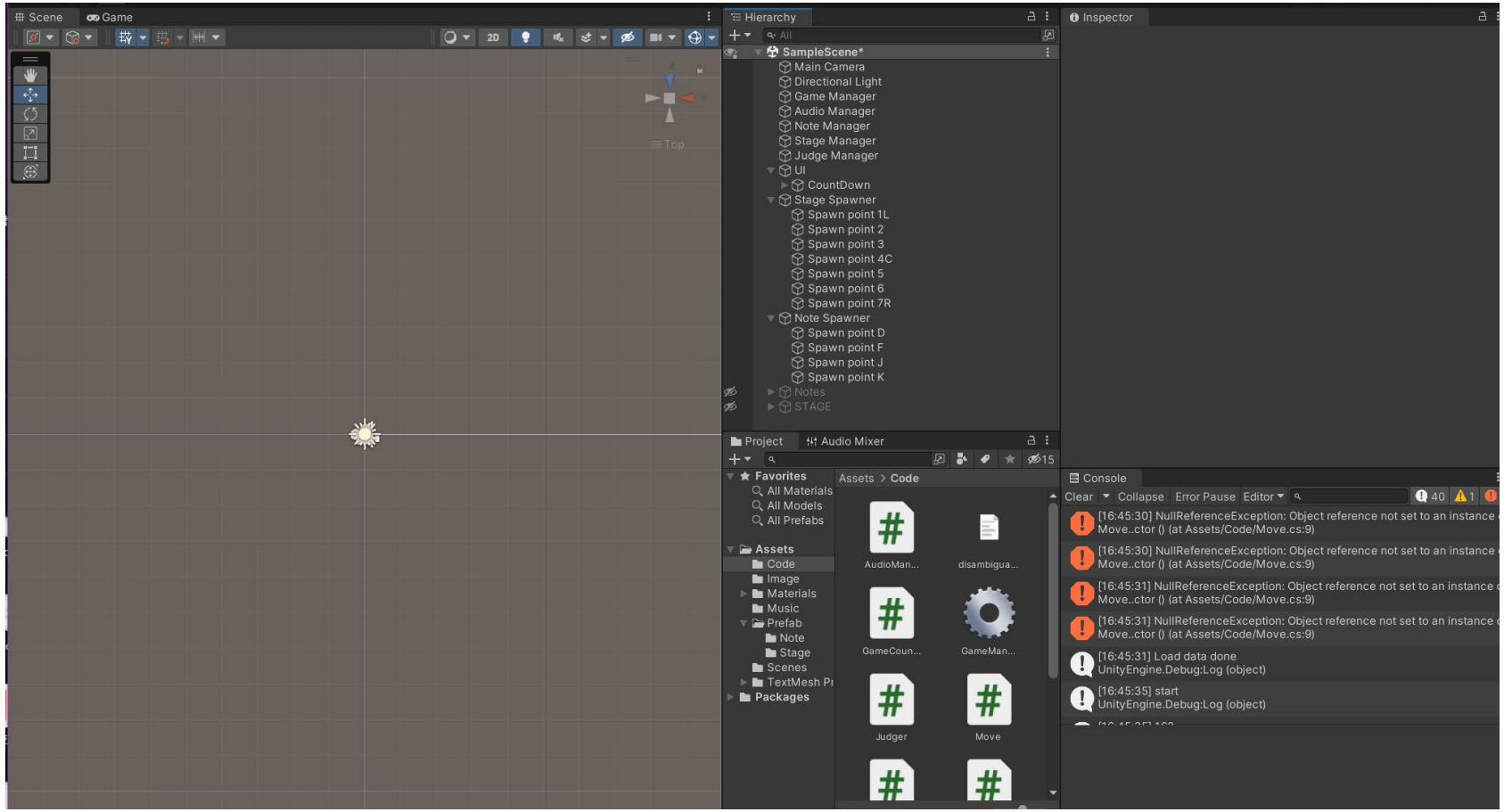
STEAM®

- 협업 툴 디스코드 (Discord)



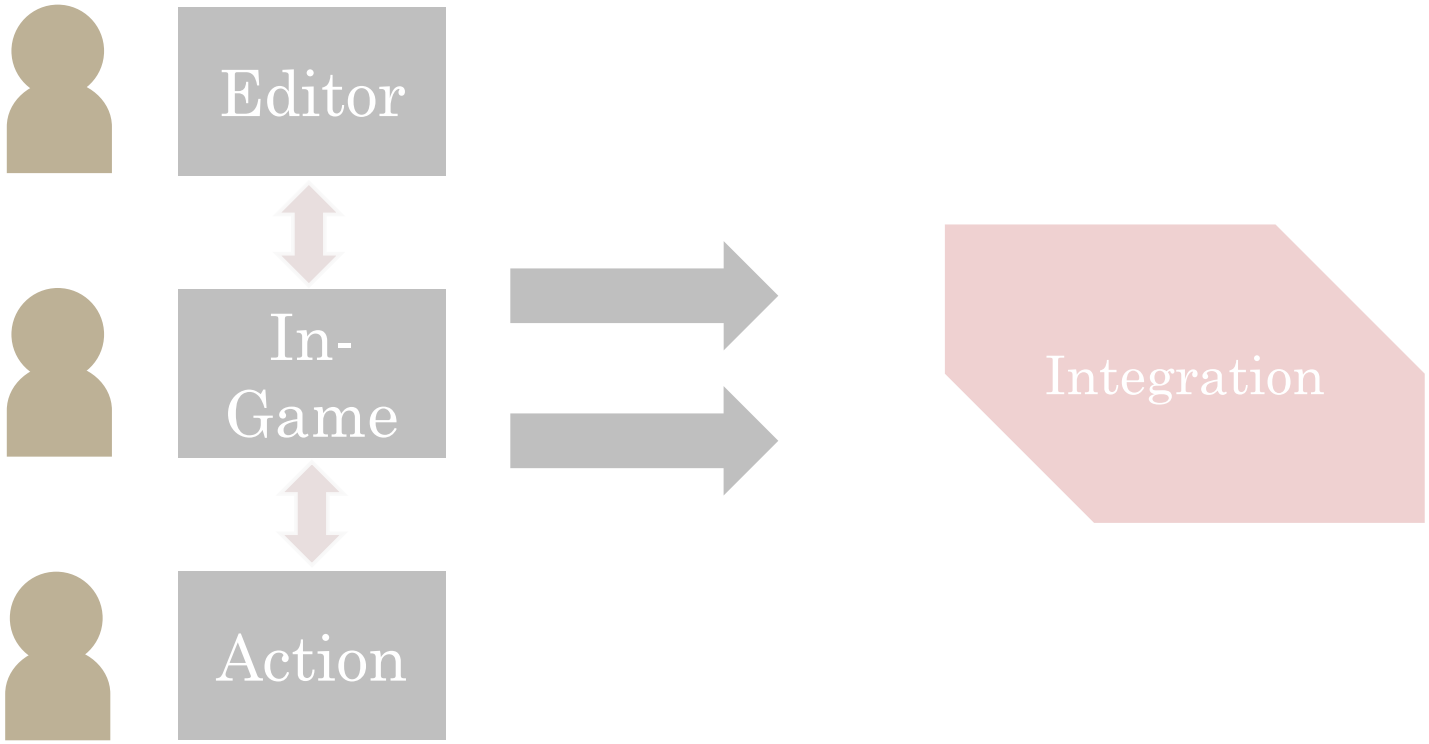
개발과정 - 사용 툴 및 환경

- Unity 21.214f1



개발과정 - 분업

- 기본적으로 3개의 파트로 나누어서 진행



개발과정 - 에디터 개발

- 에디터개발이유

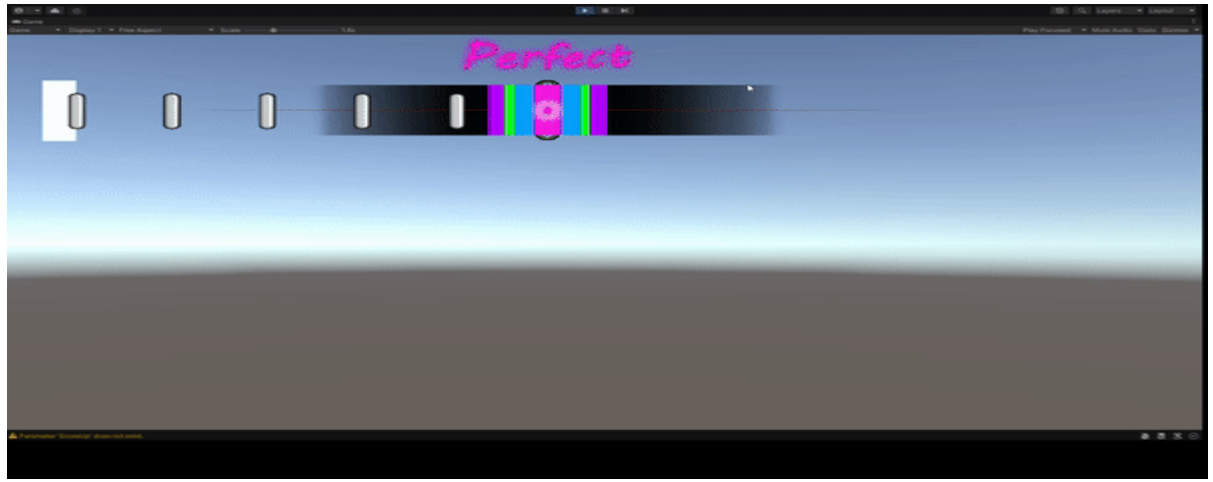
- 실제 게임을 서비스하고 운영할 경우 팀원이 늘어났을 때 혹은 서비스중의 분업을 고려해 개발분업을 하였음.

- 실제 서비스가 시작 되면 아트 / 노트파일제작(STAGE구성)을 하는 팀원을 따로 두어서 비개발자도 원활하게 소통이 가능하도록 제작함.

- 리듬게임에서 많이 쓰이는 형태인 BMS (Be-music script) 라는 시뮬레이터와 해 해당 노트 파일 규격이 있는데 이를 벤치마킹하였음.

개발과정 - 기본 기믹 제작

(예시화면)



후에 Stage가 스크롤되는 타이밍을 알 수 있는 장치(타이밍바)로 이용됩니다.

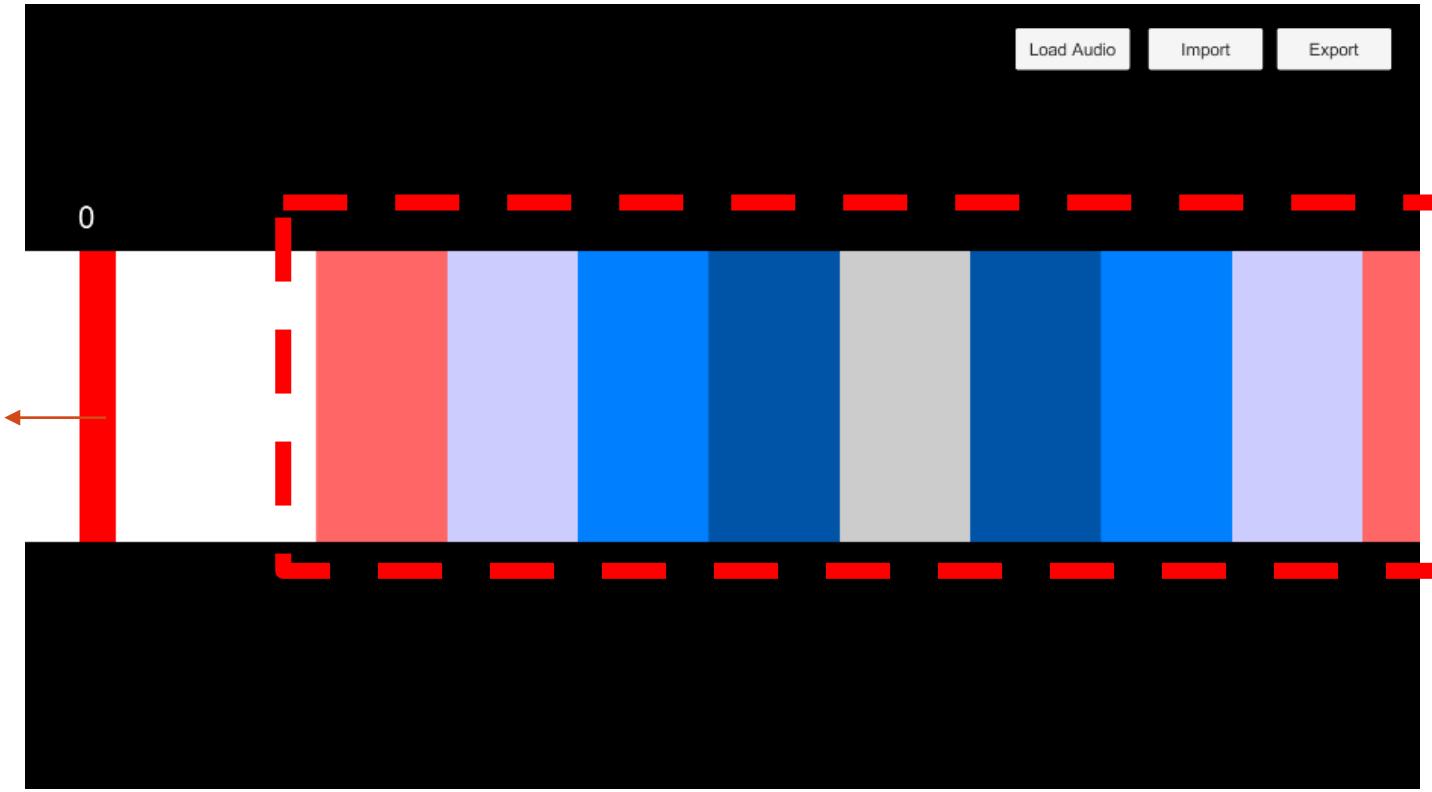
다른 레퍼런스 게임들 에서의 활용모습

개발과정 - 프로토타입 제작

- 현재 진행중인 과정 중 하나이며
  아래와 같은 플레이 장면을 가지고 있습니다.

캐릭터의 이동과 피격 피드백

게임구성 - 에디터



음악 파일 불러오기
노트 파일 입출력

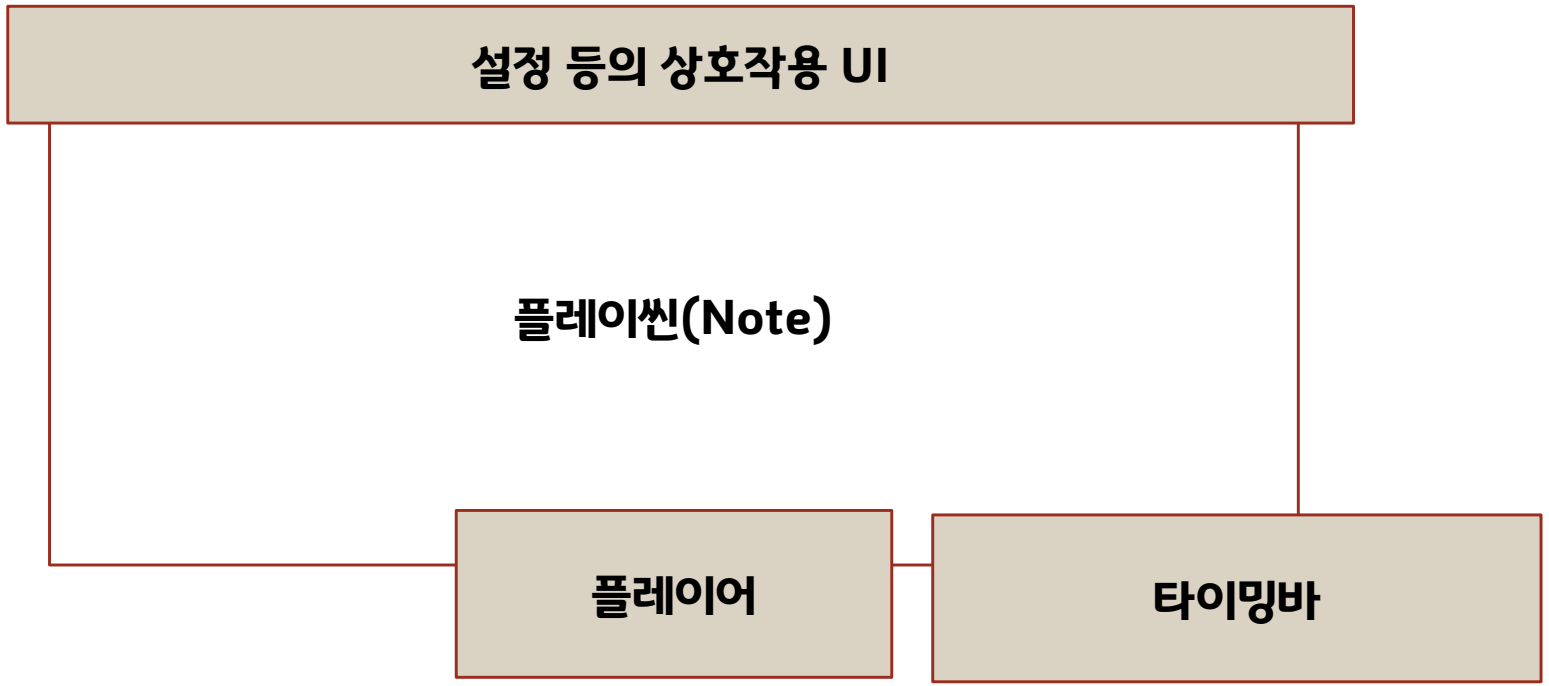
Click을 통해 타입을 정할 수 있음

판정 선

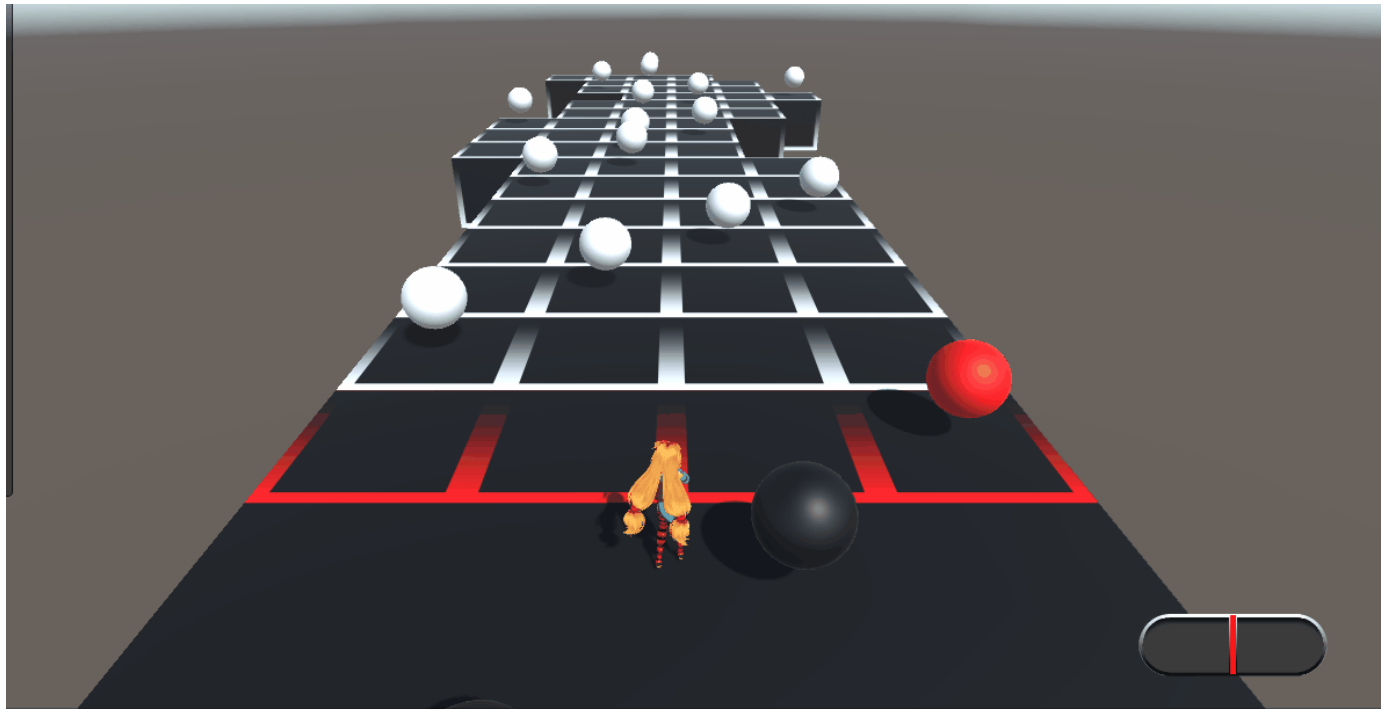
시간에 맞추어 스크롤 가능. . ..등등

게임구성 – 에디터

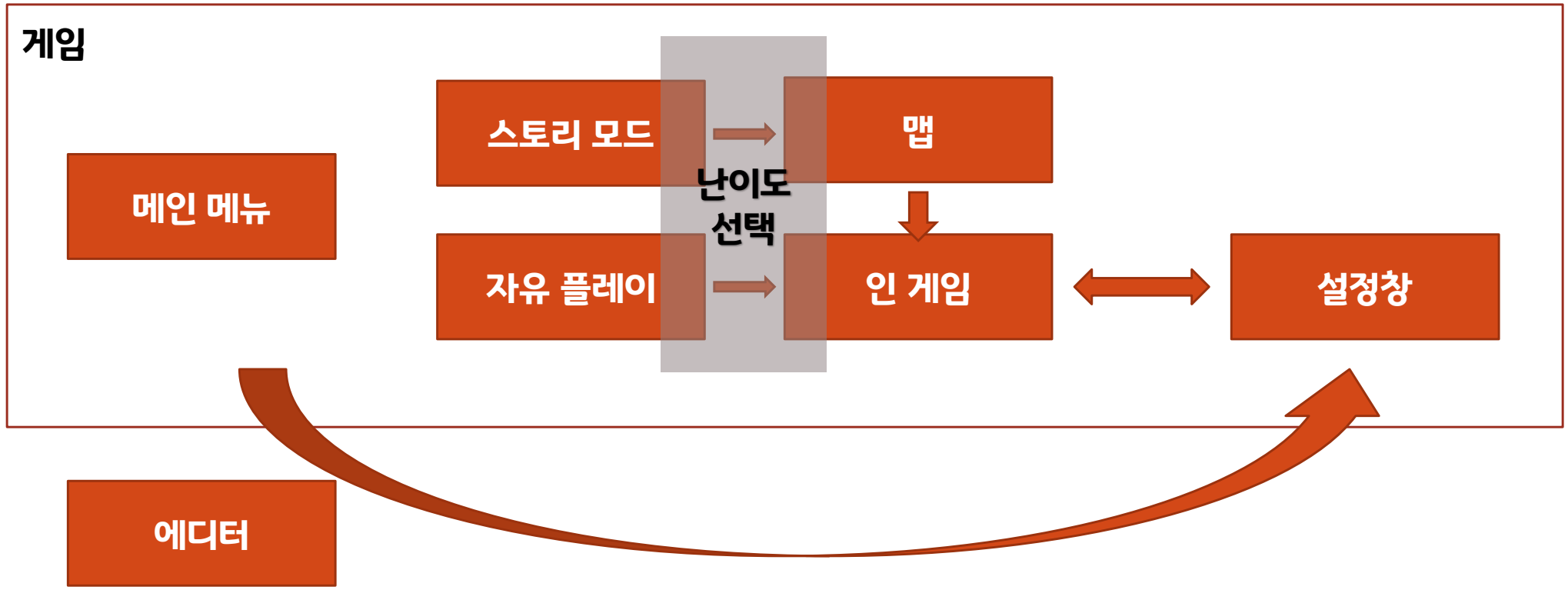
게임구성 - 인 게임 Scene



게임구성 – 인 게임 Scene



게임구성 - Scene의 흐름



게임구성 - 난이도

게임구성 - 키보드 입력

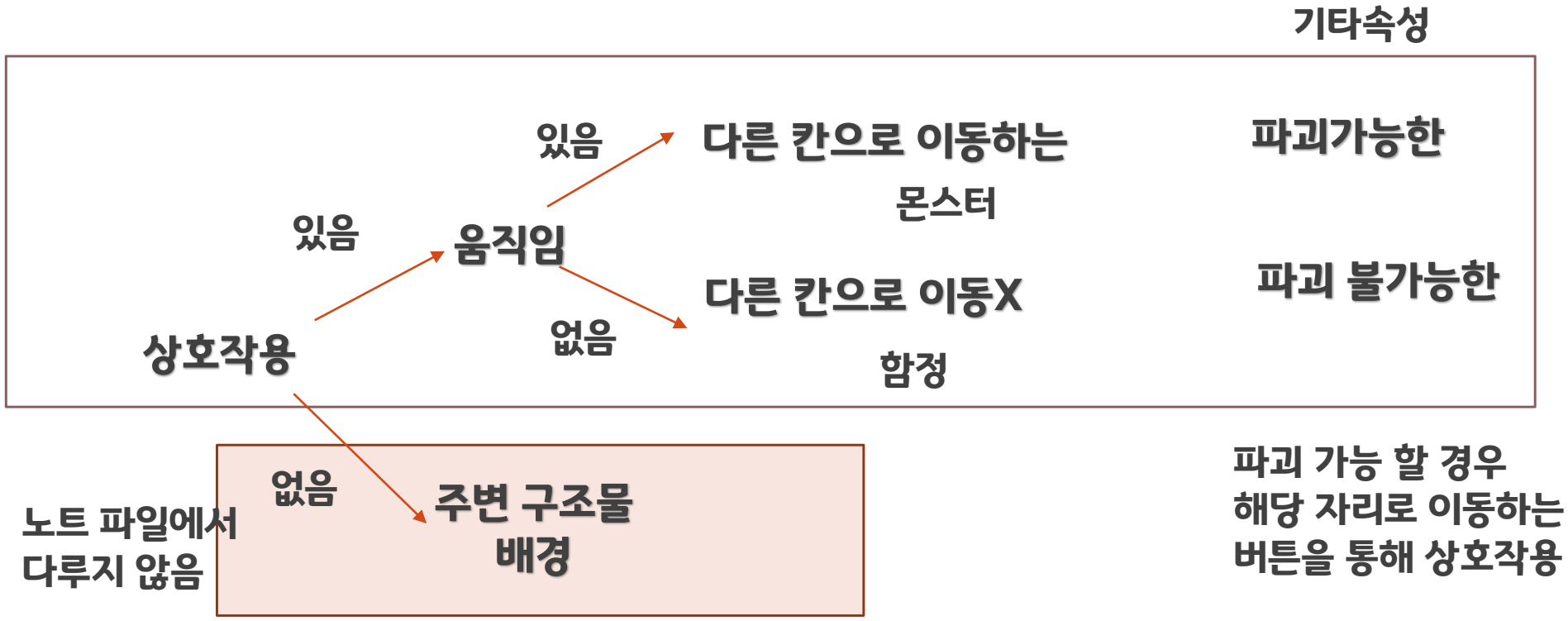
- D F J K Space bar로 게임이 진행됨
- D: 왼쪽이동
- K: 오른쪽이동
- FJ: 연속공격 및 앞으로 전진 (몬스터를 여러 번 눌러서 처리할 경우)
- Spacebar: 상호작용 및 회피, 점프 등

게임구성 - Object 종류

- (Note)
- 몬스터 – 처리해야 하는 종류의 노트
- 함정 – 캐릭터가 피해야 하는 종류의 노트
- 지형(Stage) – 캐릭터가 이동 가능한 종류의 노트

- (Note아님)
- 나머지 지형지물 및 이펙트들

게임구성 - Object 종류 (Note)



기타 -- 지형object

게임구성 - Object 종류 (Note)

속성	몬스터	함정	구조물, 배경
상호작용	O	O	X
움직임	O	X	-
파괴가능	O	X	-

* 상호작용 불가능한 Object는 노트 파일에서 다루지 않음

게임구성 - 몬스터(Note)

- 파괴가능한 – 피격 시 애니메이션, 처리시 애니메이션, 공격 애니메이션
- 다른 칸으로 이동가능 이동할 경우 보여줄 방법 필요(화살표 또는 예비동작)

한 번 이동

두 번 이동

- 이동 공격 등 모든 움직임은 정해진 박자에 한번
- 몬스터의 종류에 따라 눌러야 하는 박자, 횟수가 다름
- Ex)

한 번

세 번

8beat
네 번

셋 잇단 세번

연타(일정시간내 박자 무시하고 누르기)

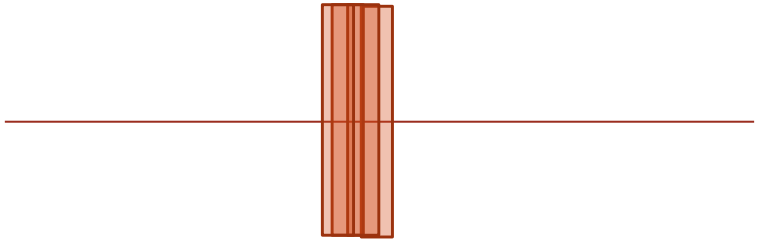
게임구성 - 몬스터(Note)

- 공격 – miss 판정을 받을 경우
- 이동(좌우) – 해당 이벤트가 있을 경우 애니메이션 스피드 (60/BPM/32beat)
- 피격 (good perfect 판정을 받을 경우)
- 회피 (bad 판정을 받을 경우)
- 통상 (idle)

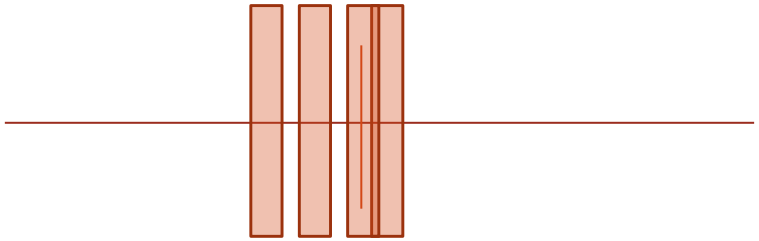
게임구성 - 몬스터(Note)

- 몬스터 판정 처리

4번 눌러야 하는 적을 4번 정확한 타이밍에 누르면 Perfect!

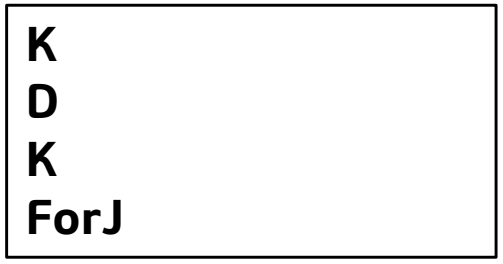


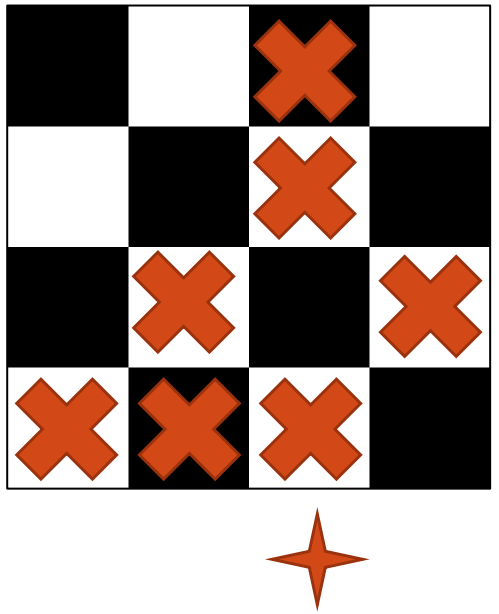
4번다 누르긴 눌렀는데 정확한 타이밍에 전부다 누르지 못했을 경우 good

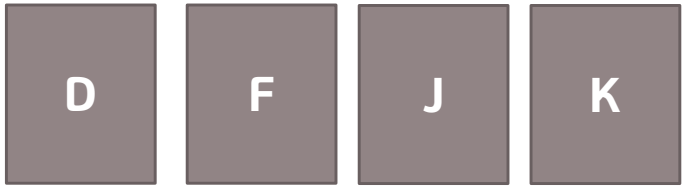


게임구성 - 함정(Note)

- 파괴 불가능한 (벽, 장애물, 구덩이 등)
- 몬스터의 경우 해당 자리로 이동해야 하지만 함정의 경우 피해야함

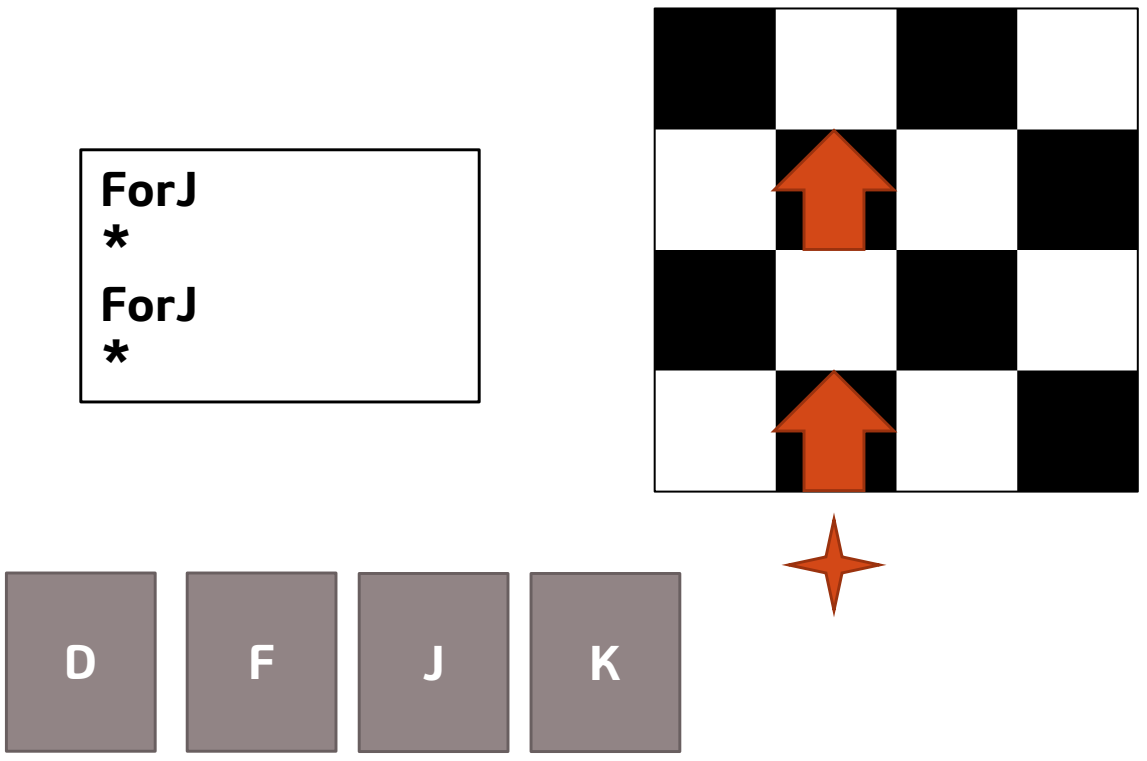






게임구성 - 함정(Note)

- STOP Note
- (아무 행동을 하지 않아야 하는 칸)



게임구성 - 함정(Note)

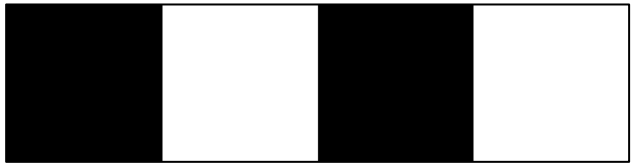
- 함정을 따로 둔 이유

- 점수에 연관되는 것은 몬스터를 잡는 것(하드코어 유저)

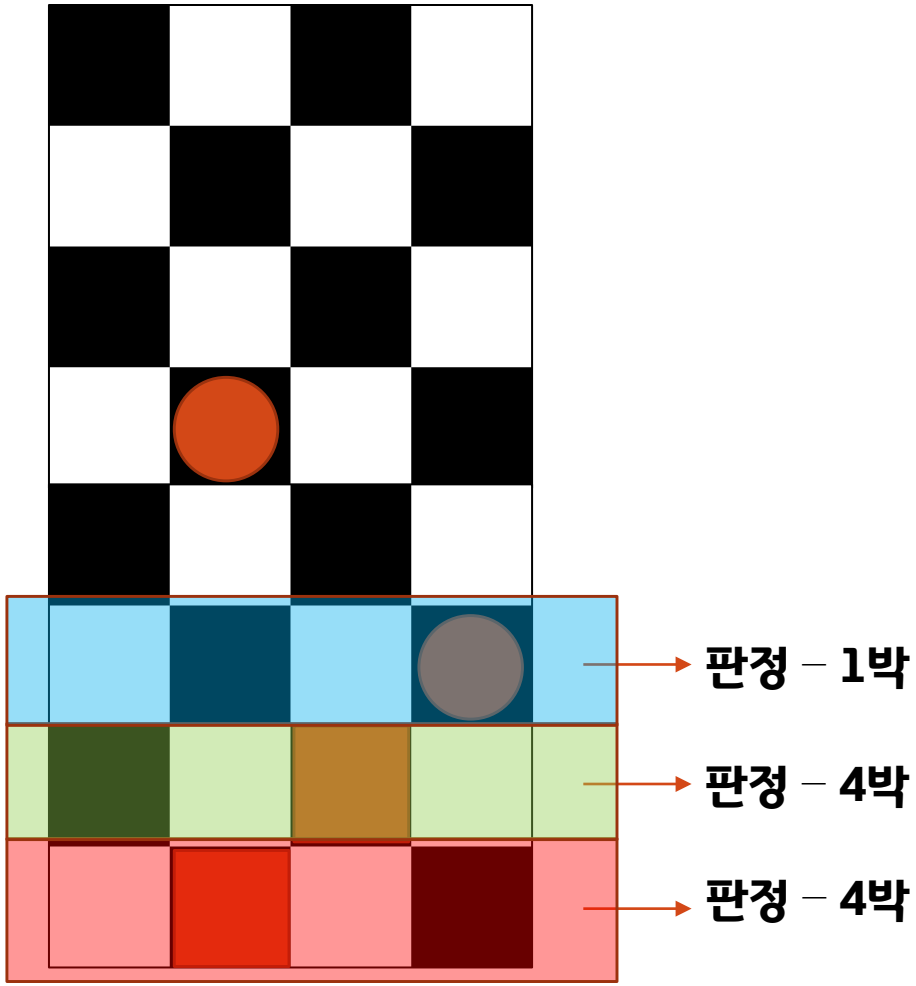
- 함정은 점수에 연관되지 않지만 클리어에 관련이 깊게

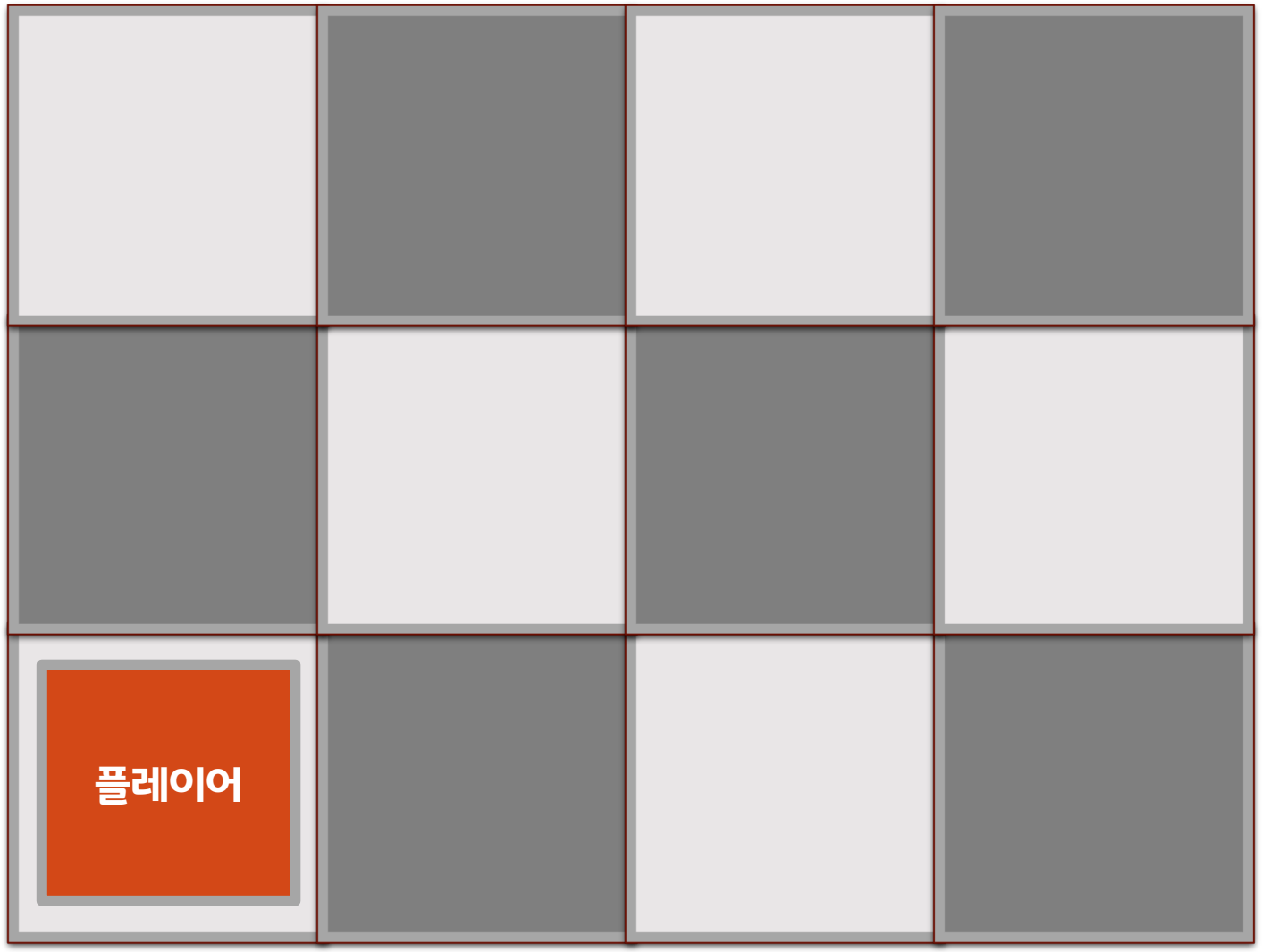
게임구성 - 지형(Note)

- 4 가로 네 칸으로 생성됨



- 판정은 스테이지가 스크롤 되는 시점에서
- 지형의 스크롤은 기본 4beat(1박)으로
  고정되지만 몬스터 및 다른 노트에 따라 변화 가능
- [더 빠른 스크롤 속도를 원할 경우 Bpm을 두 배로]
  ex) 120bpm의 4박은 60bpm의 8박

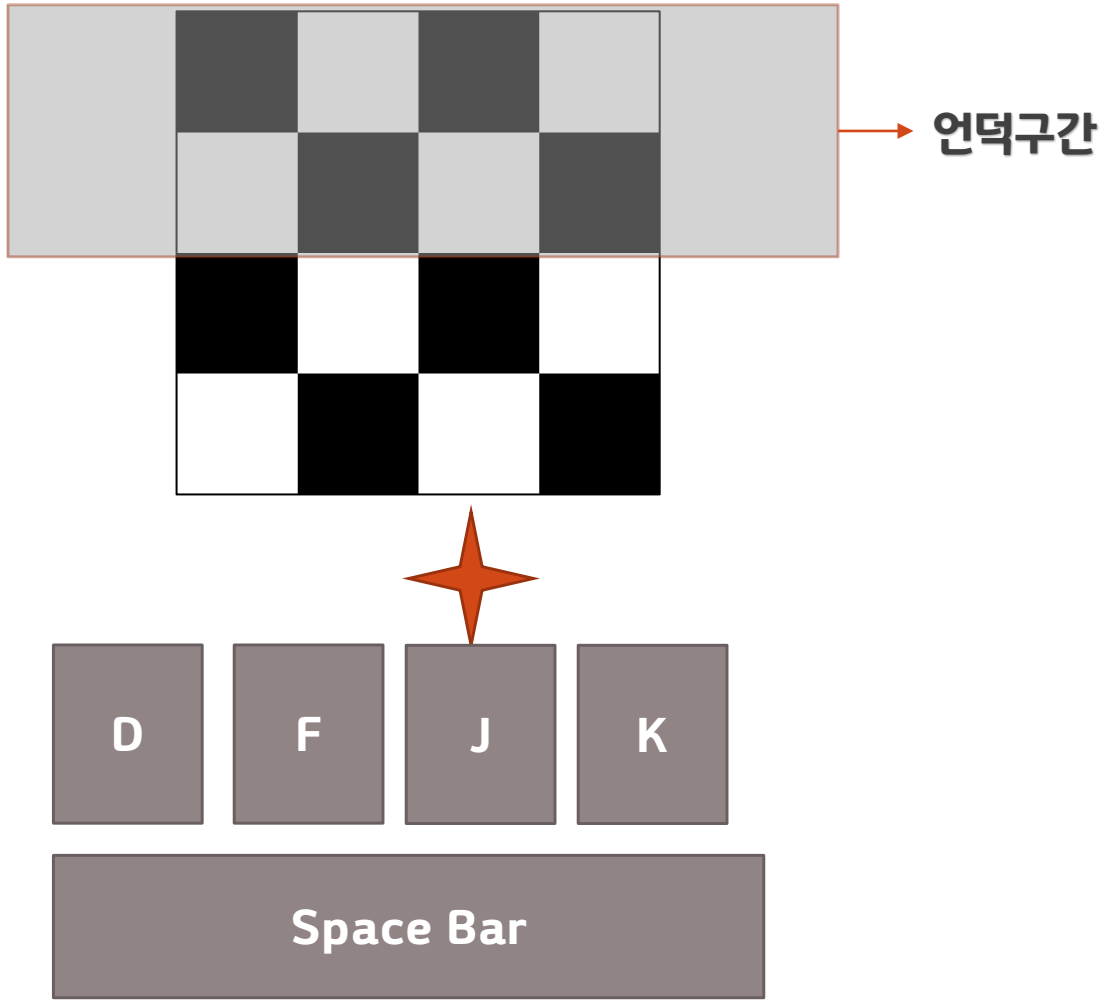




게임에서 플레이어가 느끼게
되는 화면 움직임
(1박 스크롤)

게임구성 - 지형(Note)

- 언덕



Any key
Any key
Spacebar
Any key

게임구성 - 캐릭터

- 상태와 움직임

- 공격

- 이동(대쉬) 애니메이션 스피드 (60 /BPM/32beat) 이내로 (DFJK)

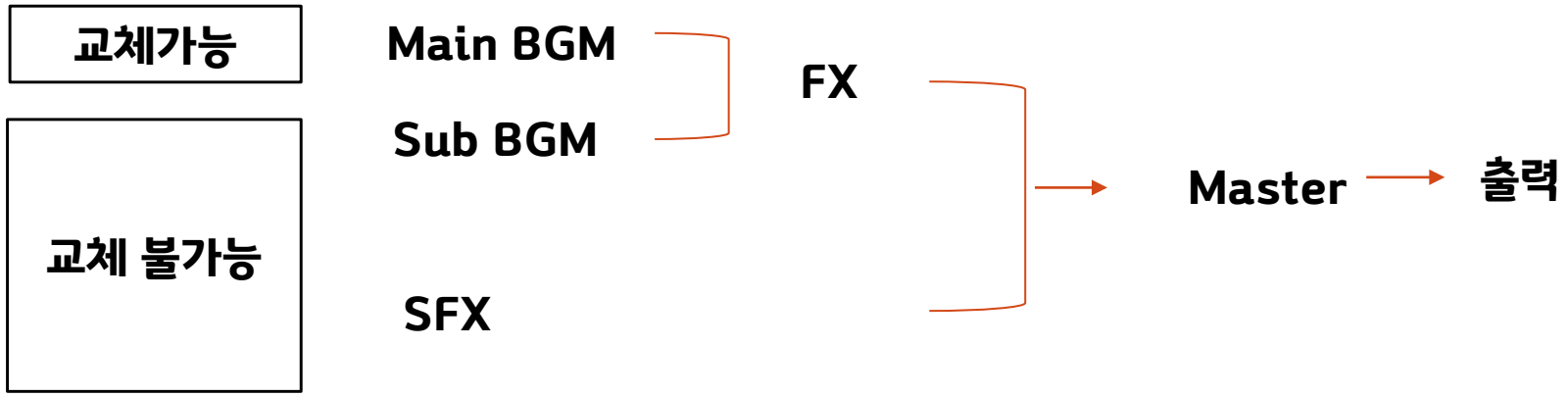
- 점프 (space)

- 피격 – 카메라 흔들기 (피격을 받아도 자동으로 stage는 스크롤됨)

- 통상(idle)

게임구성 - Audio Multi Track

- Main BGM 교체가능한 메인 악기음악 : 실제 연주할 때 Note 따라가는 소리
- Sub BGM 서브 배경음 : 이외의 배경음들
- SFX 효과음 : 타격시 타격음이나 버튼을 눌렀을 때 소리 등 특수효과음들을 담당.
- FX : 음의 Pitch 조절, Distortion(왜곡된 소리 및 노이즈) 등 직접적인 특수효과



게임구성 - Audio Multi Track

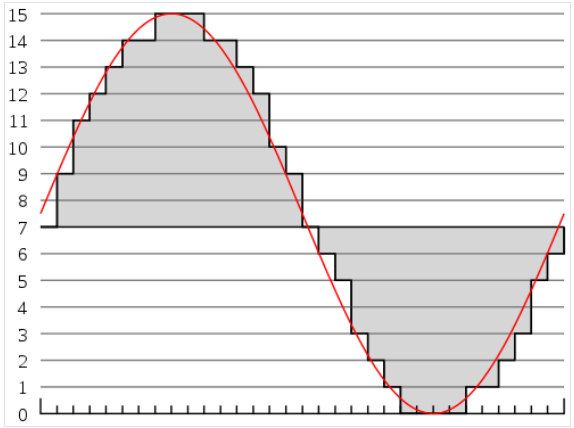
- 같은 음악이지만 Main Bgm 에서 다른 악기로 선택이 가능!

선택!

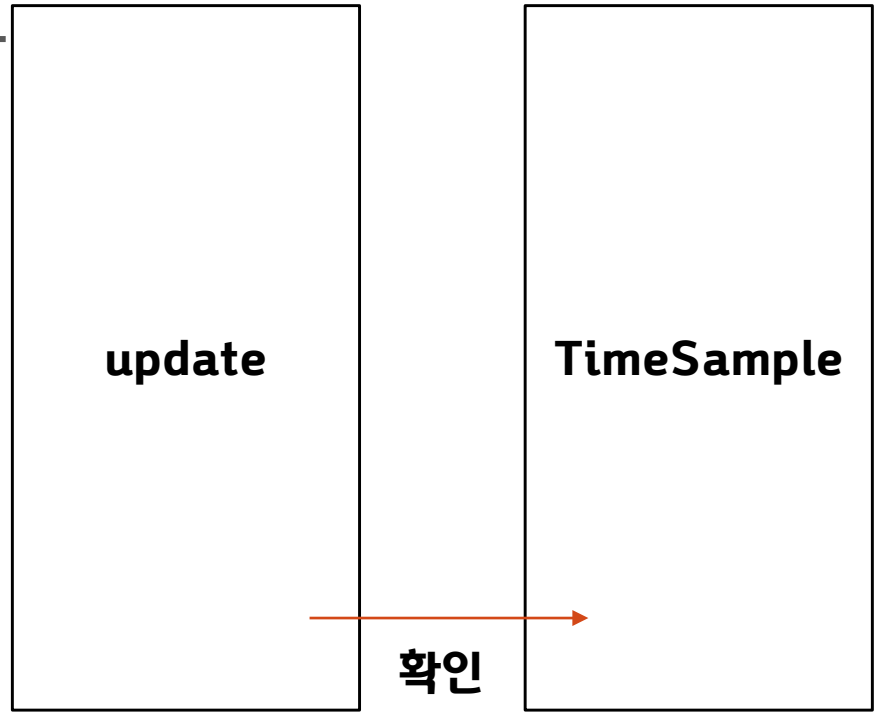
게임구성 - 재생 위치(시간) 확인

- PCM(pulse code modulation)
  샘플 단위로 재생위치를 파악 가능한
  유니티 내의 기능 TimeSamples를 이용함

- Bpm을 통한 박자의 길이
  등을 구할 수 있음.



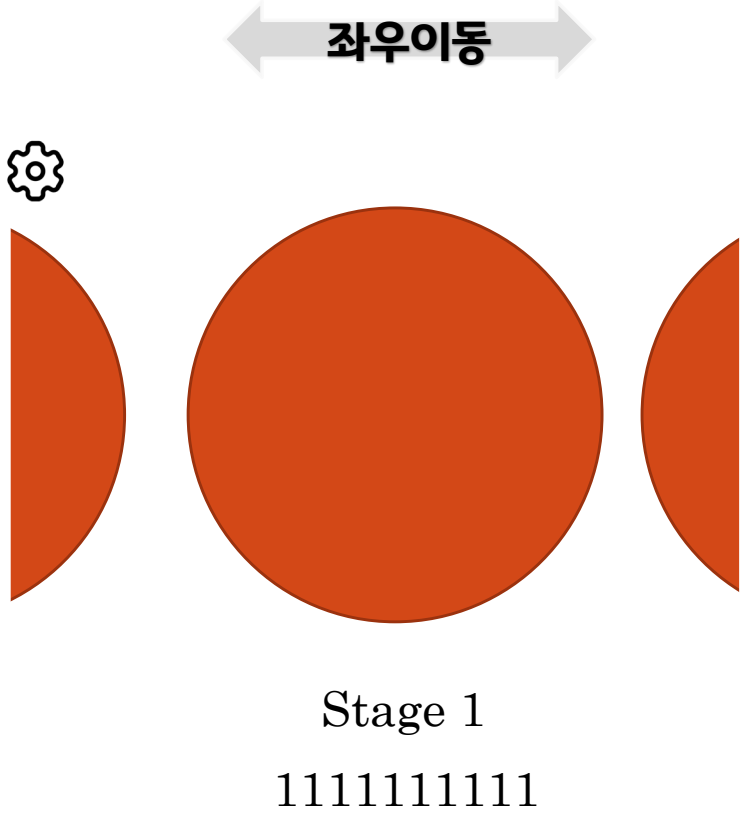
출처 http://en.wikipedia.org/wiki/Pulse-code_modulation



게임구성 - UI(prototype)

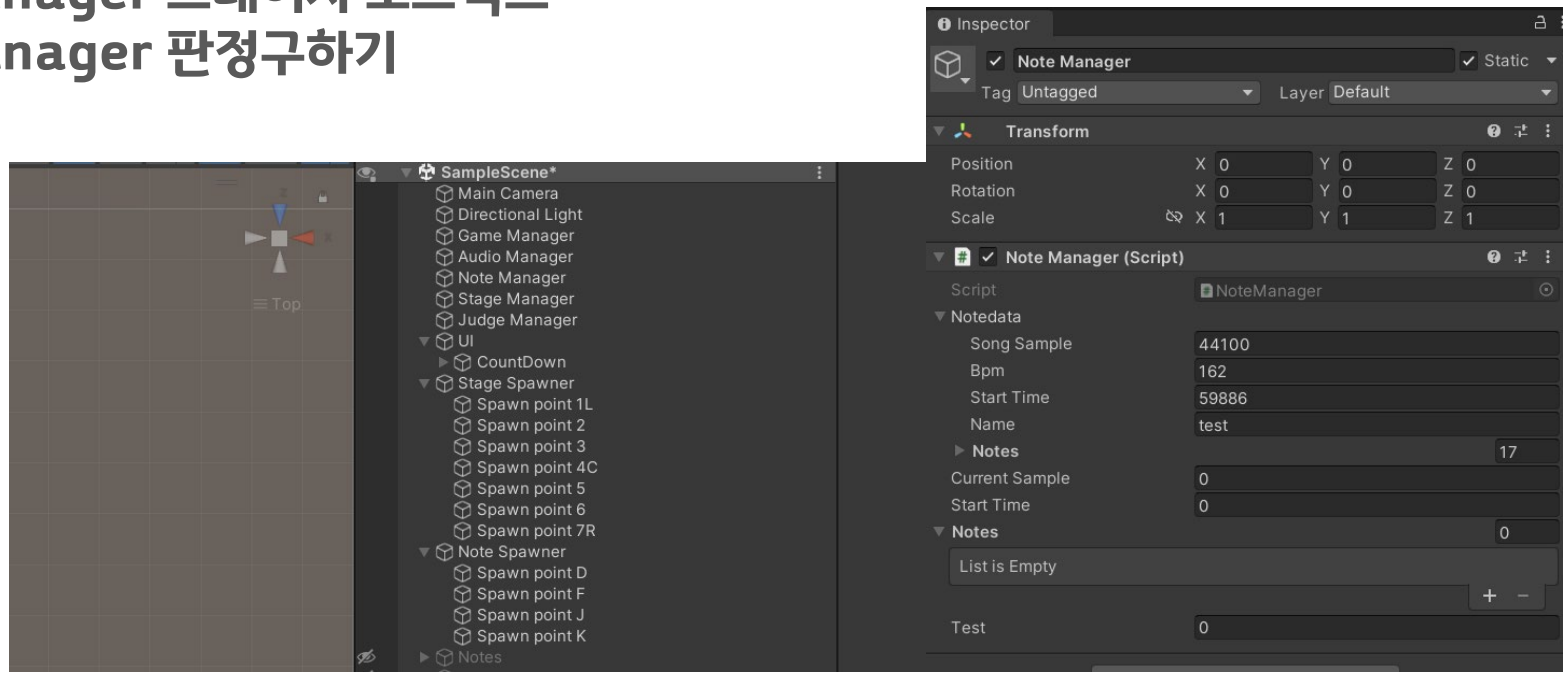
-필요요소

- 들어가기(Space 를 통해 들어가도 됨)
- 설정
- Stage 선택
- 점수
- Stage 이름
- Scene 진입
- 프로젝트 후반기 외부 Map (바깥게임)으로 교체예정



게임구성 - Manager

- 유니티를 쓰는 만큼 Singleton 패턴으로 여러 게임 매니저를 사용 중
  - GameManager 게임실행관련
  - AudioManager 오디오 정보관련
  - NoteManager 노트 정보 및 읽기 및 쓰기 / BPM 관리 이벤트
  - StageManager 스테이지 오브젝트
  - JudgeManager 판정구하기



게임구성 - Note Data

- JSON 으로 노트데이터를 입출력 하고 있음.
- E T S 는 각각 스테이지를 구성하는 노트의 정보 (Enemy Trap Stage)

```
{
    "songSample" :44100,
    "bpm": 162 ,
    "name": "test",
    "startTime": 59886,
    "notes": [
        {
            "E": 1,
            "T": 0,
            "S": 4
        },
        {
            "E": 2,
            "T": 0,
            "S": 4
        },
        {
            "E": 3,
            "T": 0,
            "S": 4
        }
        ,
        {
            "E": 4,
            "T": 0,
            "S": 4
        },
        {
            "E": 1,
            "T": 0,
            "S": 4
        },
        {
            "E": 2,
            "T": 0,
            "S": 4
```

게임구성 - BM 및 특징적 요소

- Steam 에서 의 판매가 일차 목적이며 유니티 기능을 최대한 사용하여 다른 플랫폼으로의 포팅 가능성을 염두 하며 개발중임. (외부 라이브러리 및 Asset 을 최대한 자제하고 있음)
- BM은 리듬게임에서 가장 많이 사용되고 있는 패키지 형태의 게임이 될 것이며 아이템 등을 통한 시즌 패스의 도입 또한 생각해두고 있음.

- 패키지 (완성된 본편과 확장팩(DLC 다운로드컨텐츠))의 판매형태
- 시즌패스 (일정 기간동안 발매된 DLC, 업데이트를 판매하거나 추가 보상을 주는 형태)

게임구성

- 이외에도 여러가지 키조합에 따른 커맨드
- 게임점수스코어 계산
- 피격시의 판정처리

프로젝트의 차별성 및 강점

- 이전에 없던 방식의 플레이 방법(박자에 따른 이동 및 몬스터의 처리 방법등)
- 악기를 선택해 플레이를 할 수 있음
- 게임 씬 의 이동중 플레이 가능한 맵의 존재(바깥게임)
- 플레이어의 숙련도를 고려한 Note구성(함정 몬스터의 구분)
- 하드코어 유저들이 많은 만큼 매니아도 많은 게임 장르
- 경쟁 작품이 많이 없는 장르
- 게임 심의에 유리함

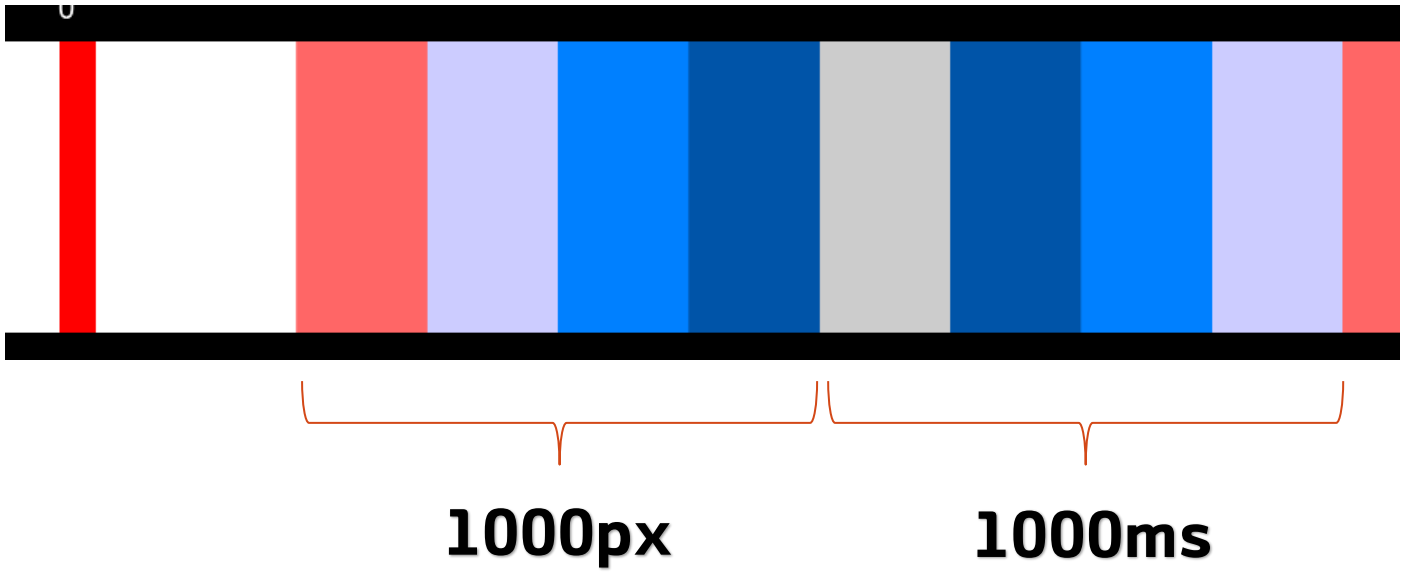
기술적 어려움들

- 소통 및 룰에 대한 문제
  비슷한 기능을 쓰거나 같은 기능을 구현하더라도 전혀 다른 방식을 쓸 수 있어 게임의 규칙과 구현 방법에 대해서 많은 회의를 진행중

- 3D 관련 기능을 많이 쓸 수 없음
  성능에 많은 영향을 주고 매 프레임 입력을 제대로 보장받지 못하는 기능들이 많아 제대로 사용할 수 없음

- 각 파트별로 서로 모르는 부분이 많이 생김
  맡은 부분에 관련해 깊은 공부를 하며 다른 각 파트별로 모르는 부분 발생

기술적 어려움들

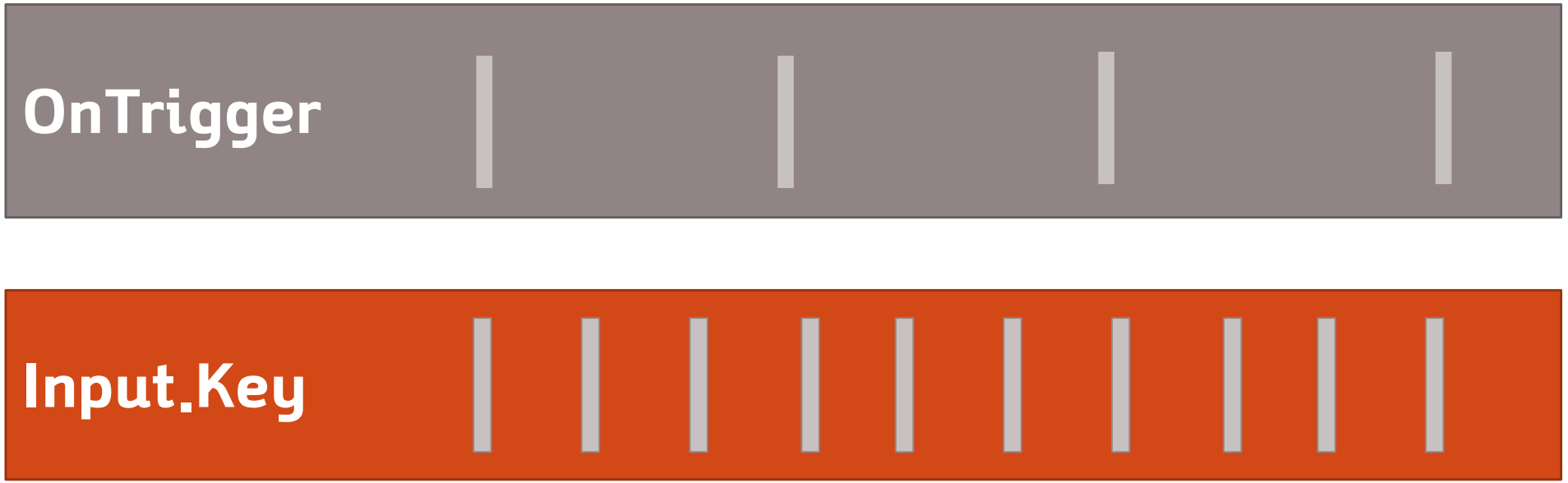


시간을 화면에 매칭시키는 법

1000ms = 1000px

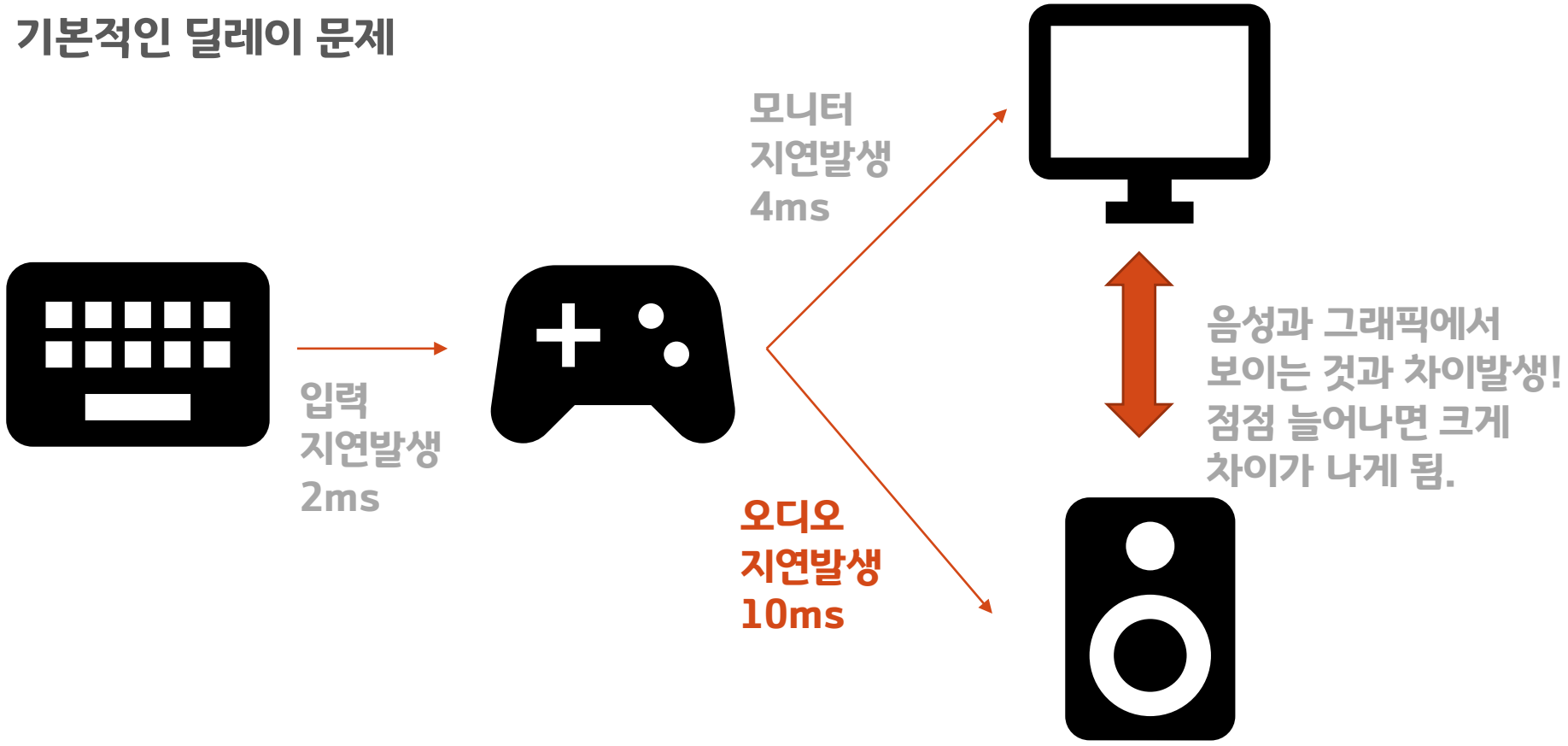
기술적 어려움들 - Action

Input시 충돌감지가 원활이 작동하지 않는 문제



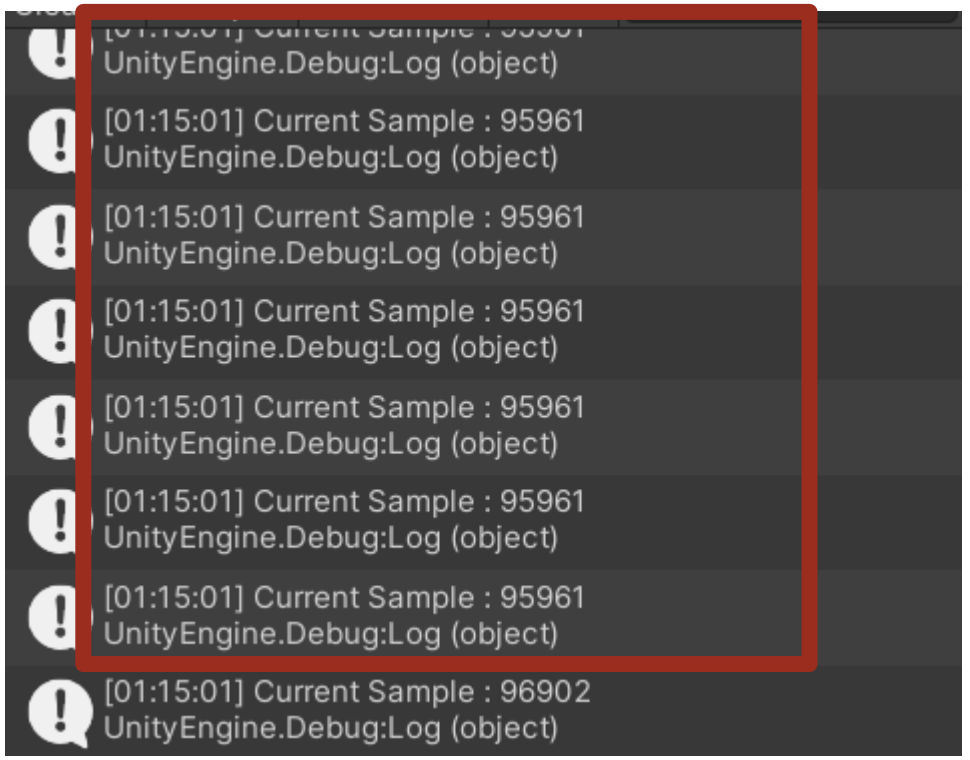
기술적 어려움들 - 인 게임

- 기본적인 딜레이 문제



기술적 어려움들 - 인 게임

- 딜레이(latency) 문제
  - 사운드 출력은 유니티의 고질적인 문제
  - 기기의 성능에 따라 천차만별
  - 게임 특성상 40ms내에 여러가지 판정이 이루어져야 해서 시간적 정확도에 민감함
  - 매 프레임 정확한 샘플을 보장받지 못함
    (샘플에 해당되는 화면 움직임을 만들 시 비연속적 움직임을 갖게 됨)



매프레임 같은 샘플이 찍히고 있다.

기술적 어려움들 - 인 게임

- 딜레이(latency) 문제 해결방안

-1 미들웨어의 사용

(아래 방법은 오디오 딜레이는 막지 못함)

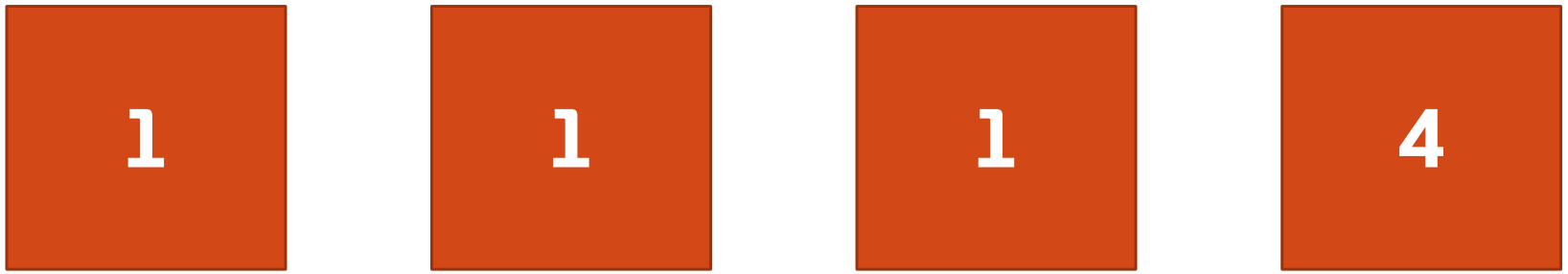
-2 미리 Offset 설정

-3 일정 시간마다 Sample 시간과 게임내부 시간을 동기화

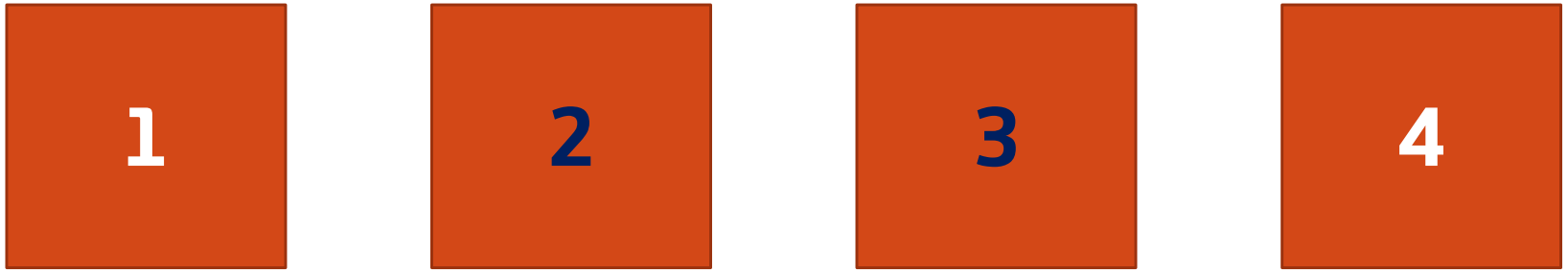
-4 미리 Sample 호출 간격을 예상해서 보간 하기

기술적 어려움들 - 인 게임

Sample 차이3 발생



사이 값 보간



차후 프로젝트 진행

UI	그래픽	음악
기믹 추가	맵 제작 (바깥게임)	채보 제작 (Note)