

POLA

포트폴리오

레이턴시

플레이



-박준기-

목차

01

게임 소개

데모영상
게임설명

02

UI

플로우차트
UI 작성내용
인게임 적용

03

시스템

랭크 시스템
콤보 시스템
판정 시스템
레이턴시 시스템

04

컨텐츠

스테이지 컨텐츠
스테이지 제작 방법
별 컨텐츠

05

사운드

사운드 리소스
리소스 관리 문서

06

그래픽

캐릭터 그래픽
UI 그래픽

참고자료

별 콘텐츠 기획서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

랭크 시스템 기획서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

캐릭터 모델링 문서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

사운드 리소스

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

UI 기획서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

스테이지 비트맵

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

콤보 시스템 기획서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

인게임 그래픽 파일

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

스테이지 데이터테이블

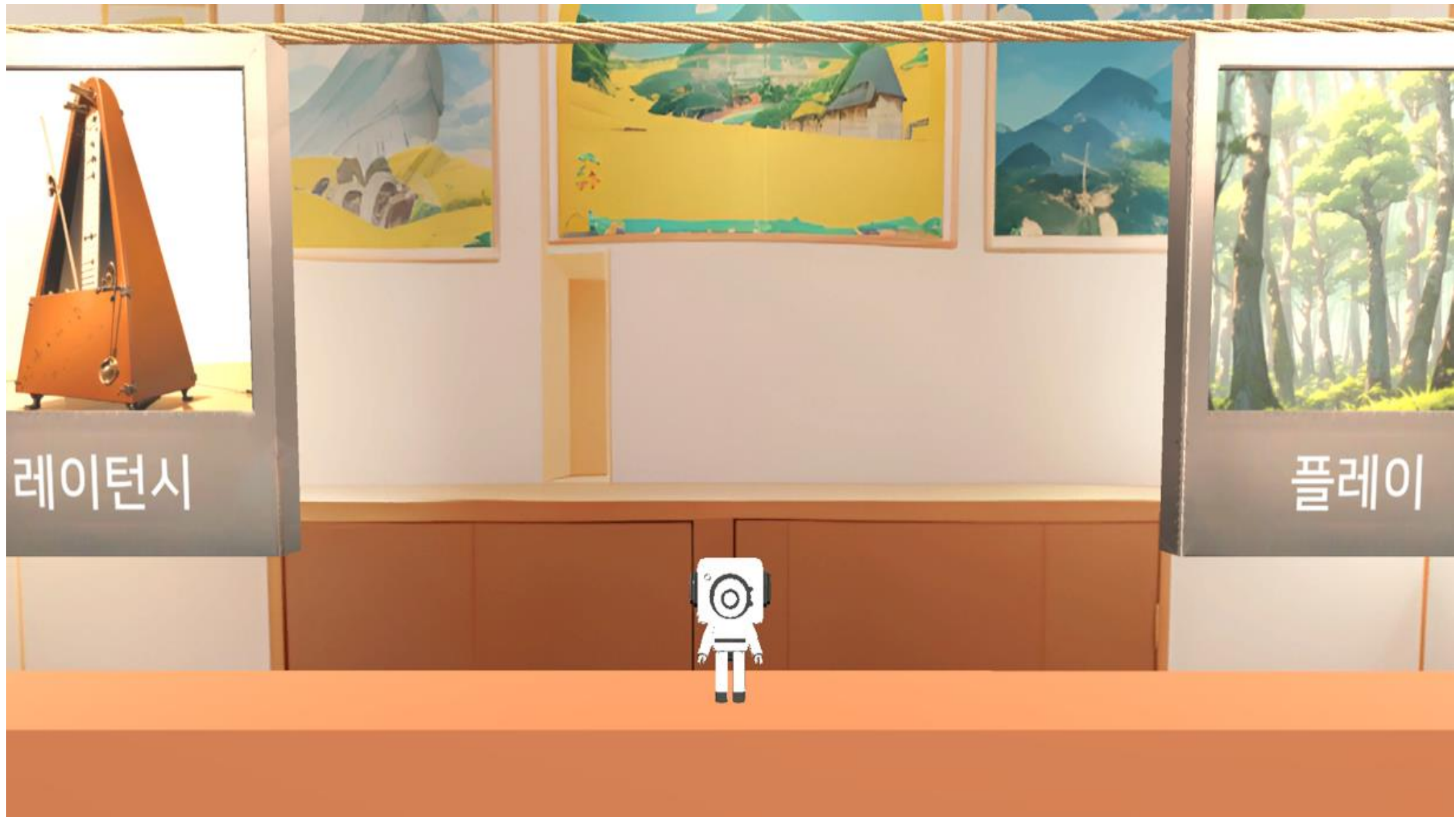
[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

판정 시스템 기획서

[-구글 드라이브 링크](#)
[-네이버 MYBOX 링크](#)

1

게임 소개



[POLA 60초 소개 영상](#)

(하이퍼링크 첨부)

POLA

장르: 캐주얼, 리듬

게임컨셉: 폴라로이드, 사진관

플랫폼: Android, iOS, PC

개발엔진: Unity-2022.3.10f1 / Stable Diffusion / Photoshop

개발기간: 2023. 10 ~ 2023.12 (2개월)

개발인원: 4인 개발(기획1, 프로그래밍3)

출시경험: 구글플레이스토어 출시 및 서비스 종료 (2024.01 ~ 2024.11)



기획 의도

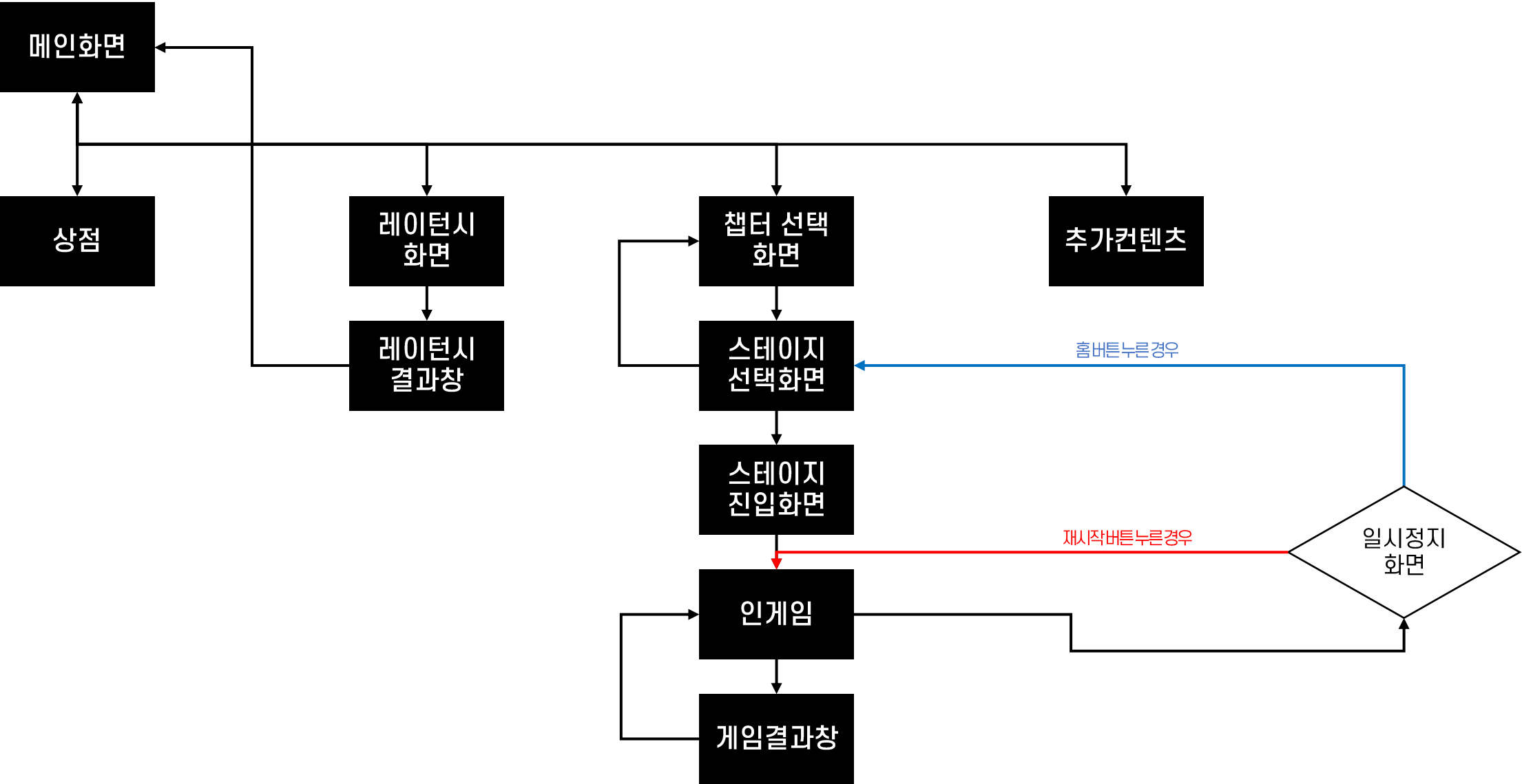
- 폴라로이드 사진에 들어가 그 사진의 추억을 경험해보자-
- 사진에 담긴 추억을 리듬게임의 스테이지로 표현해보자-

2

UI



플로우차트



UI 작성내용

1.3. UI구성

index	이름	설명	타입
1	전채배경	- 메인화면의 배경화면을 출력한다.	SkyBox
2	설정창 아이콘	- 아이콘 1회 터치(클릭)시 설정창 씬으로 이동한다.	Button
3	모드 이미지	- 해당 모드에 맞는 이미지를 출력한다.	Img
4	모드 아이콘	- 모드 아이콘 1회 터치(클릭) or 키입력 시 해당 모드 씬으로 이동한다.	Button
5	모드 텍스트	- 해당 모드의 이름을 텍스트형식으로 하단에 출력한다.	String
6	플레이어 캐릭터	- 메인화면에서 유저가 조작하는 오브젝트, 해당 오브젝트를 기준으로 하여 모드 아이콘과의 상호작용이 이루어진다. - 카메라는 플레이어를 중심으로 움직이며 플레이어는 '모드아이콘'을 기준으로 한칸씩 이동을 한다.	Object
7	바닥이미지	- 메인화면의 바닥 이미지를 출력한다.	Img

메인화면씬 구성요소화면

- 맵 이동 설명

→ 메인화면의 맵은 무한맵으로 4번째 모드에서 오른쪽으로 움직임을 입력시 다시 첫번째 모드 아이콘 위치로 이동하게 된다. / 반대도 동일

→ 자루 콘텐츠가 추가 될 경우 콘텐츠와 상점 아이콘간의 거리가 멀어진다.

→ 상점과 거리가 가까운 콘텐츠 아이콘이면 문제가 없지만 정반대이거나 그에 준하는 거리에 배치된 아이콘일 경우 매번 한칸씩 이동하기 번거로움이 발생한다.

→ 해당 조건에 만족하는 콘텐츠에서 상점창으로 편하게 이동하기 위한 방법으로 해당 방법을 착안

→ 이때 이 기능이 가능하다는 알림을 주기 위해서 양쪽에 화살표 오브젝트를 배치

메인화면씬상호작용기획

기획서작성나선

메인화면 / 챕터 선택 / 스테이지 선택 / 스테이지 진입 / 인게임 / 결과창

U별 기획 작성내용

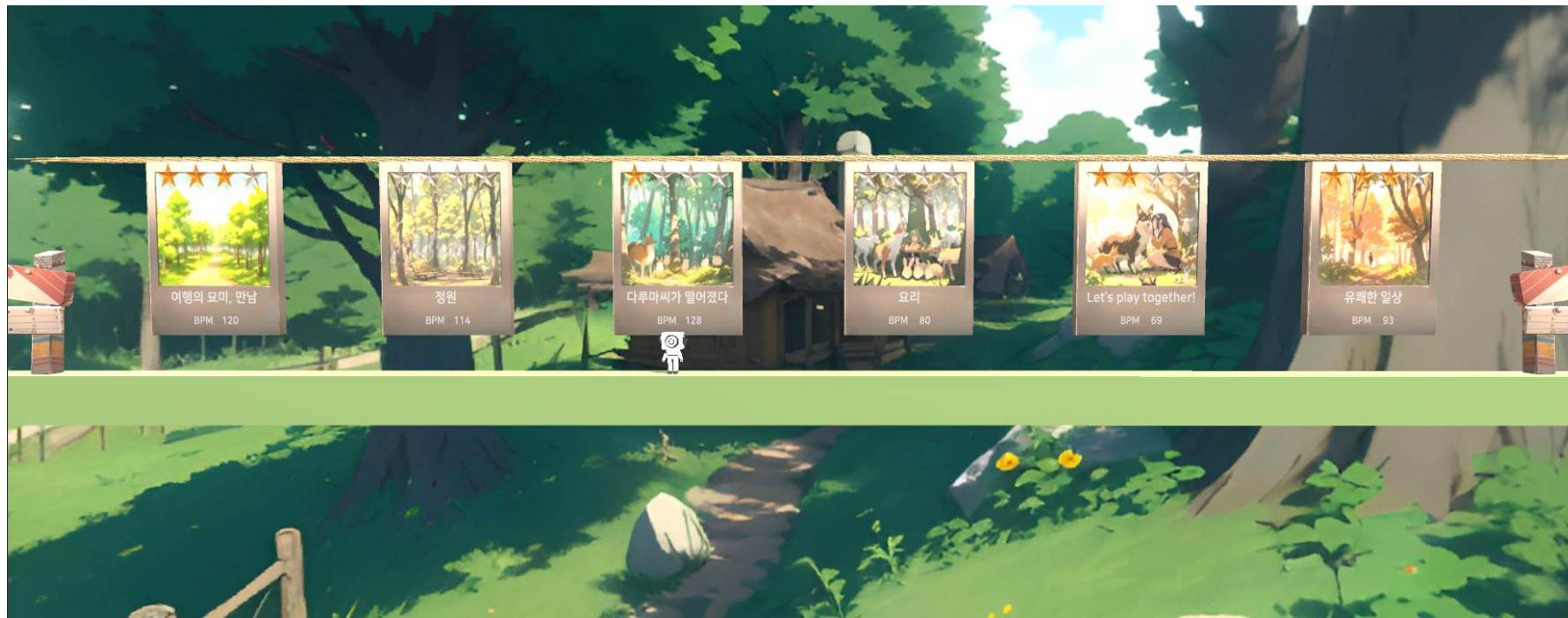
기획의도 / 레퍼런스 이미지 / U구성 / 씬 오브젝트 배치 위치 / 씬 이동 설명 / 특이사항 / 결과물 / 적용 사운드 / 플로우차트

참고자료

참고 자료 페이지 '나'기획서 링크참조'

아이디어 : 카메라 컨셉의 캐릭터를 기획 의도의 핵심 요소로 설정했는데 이를 게임 플레이에만 보여주기에는 아쉽다.

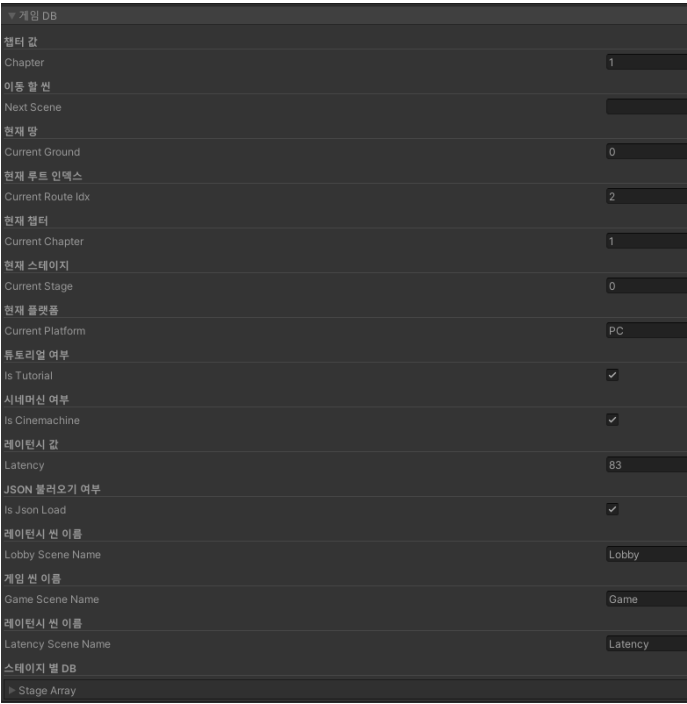
- 게임 캐릭터가 실제로 조작할 수 있는 것처럼 들리며 반응하는 인터페이스인 Diegetic UI 방식을 차용
- 설정창을 제외한 모든 상호작용을 캐릭터를 직접 조작하여 이뤄지도록 설정
- 메인 플레이, 레이턴시 콘텐츠, 상점 등을 캐릭터를 직접 이동시켜 사진과 그 사진 속 이야기라는 요소에 몰입할 수 있도록 기획



캐릭터를 직접 움직여 스테이지에 들어가는 형식의 UI 디자인

아이디어 : Diegetic UI 방식은 조작 횟수가 많아지면 유저의 편의성이 떨어지니 DB를 구성하여 UI 시스템의 편의성을 높이자.

- 게임 내부 DB를 구현하여 1차적으로 게임 데이터(최근 플레이 스테이지, 콤보, 스코어 등)를 저장한 후, Json으로 만든 외부 DB에 2차로 저장하여 해당 주소값을 통해 언제든지 게임 데이터를 불러올 수 있도록 구현
- 구현한 DB값을 이용해 각 UI별 이동 시 유저가 최근 플레이한 스테이지와 모드로 즉시 이동 할 수 있도록 구현



Game DB	
현재 값	
Chapter	1
이동 할 씬	
Next Scene	
현재 땅	
Current Ground	0
현재 루트 인덱스	
Current Route idx	2
현재 챕터	
Current Chapter	1
현재 스테이지	
Current Stage	0
현재 플랫폼	
Current Platform	PC
튜토리얼 여부	
Is Tutorial	<input checked="" type="checkbox"/>
시네마산 여부	
Is Cinemachine	<input checked="" type="checkbox"/>
레이턴시 값	
Latency	83
JSON 불러오기 여부	
Is Json Load	<input checked="" type="checkbox"/>
레이턴시 씬 이름	
Lobby Scene Name	Lobby
게임 씬 이름	
Game Scene Name	Game
레이턴시 씬 이름	
Latency Scene Name	Latency
스테이지 배열 DB	
Stage Array	

게임 플레이에 필요한 요소에 맞는 DB를 제작하여 Unity 내부에서 쉽게 관리 및 조작할 수 있게 모듈화 한 이미지

3

시스템

랭크 시스템

1.3. UI구성

전체배경

1

2

3

4

5

6

7

모드 이미지

모드 텍스트

바닥이미지

index	이름	설명	타입
1	전체배경	- 메인화면의 배경화면을 출력한다.	SkyBox
2	설정창 아이콘	- 아이콘 1회 터치(클릭)시 설정창 씬으로 이동한다.	Button
3	모드 이미지	- 해당 모드에 맞는 이미지를 출력한다.	Img
4	모드 아이콘	- 모드 아이콘 1회 터치(클릭) or 키입력 시 해당 모드 씬으로 이동한다.	Button
5	모드 텍스트	- 해당 모드의 이름을 텍스트형식으로 하단에 출력한다.	String
6	플레이어 캐릭터	- 메인화면에서 유저가 조작하는 오브젝트, 해당 오브젝트를 기준으로 하여 모드 아이콘과의 상호작용이 이루어진다. - 카메라는 플레이어를 중심으로 움직이며 플레이어는 '모드아이콘'을 기준으로 한칸씩 이동을 한다.	Object
7	바닥이미지	- 메인화면의 바닥 이미지를 출력한다.	Img

메인화면씬구성요소화면

- 맵 이동 설명

맵의 마지막 콘텐츠 아이콘

추가 콘텐츠

방향 이동

상점창 진입 아이콘

추가 콘텐츠

방향 이동 시 기존 콘텐츠 ui 등장

→ 메인화면의 맵은 무한맵으로 4번째 모드에서 오른쪽으로 움직임을 입력시 다시 첫번째 모드 아이콘 위치로 이동하게 된다. / 반대도 동일

→ 자후 콘텐츠가 추가 될 경우 콘텐츠와 상점 아이콘간의 거리가 멀어진다.

→ 상점과 거리가 가까운 콘텐츠 아이콘이면 문제가 없지만 정반대이거나 그에 준하는 거리에 배치된 아이콘일 경우 매번 한칸씩 이동하기 번거로움이 발생한다.

→ 해당 조건에 만족하는 콘텐츠에서 상점창으로 편하게 이동하기 위한 방법으로 해당 방법을 착안

→ 이때 이 기능이 가능하다는 알림을 주기 위해서 양쪽에 화살표 오브젝트를 배치

메인화면씬상호작용기획

기획서작성나선

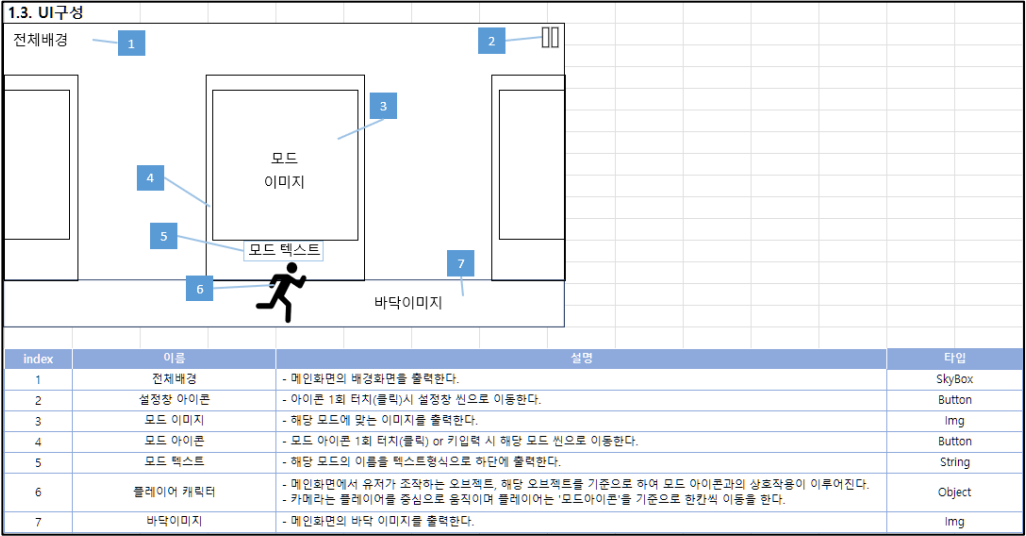
메인화면 / 챕터 선택 / 스테이지 선택 / 스테이지 진입 / 인게임 / 결과창

U별 기획 작성내용

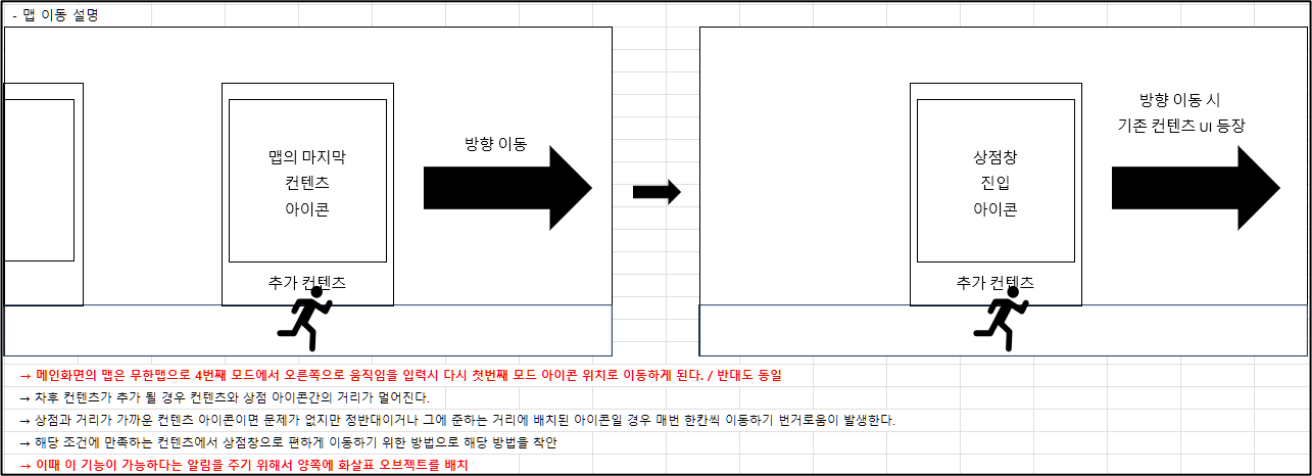
기획의도 / 레퍼런스 이미지 / U구성 / 씬 오브젝트 배치 위치 / 씬 이동 설명 / 특이사항 / 결과물 / 적용 사운드 / 플로우차트

참고자료

참고 자료 페이지 '나'기획서 링크참조'



메인화면씬구성요소화면



메인화면씬상호작용기획

기획서작성나선

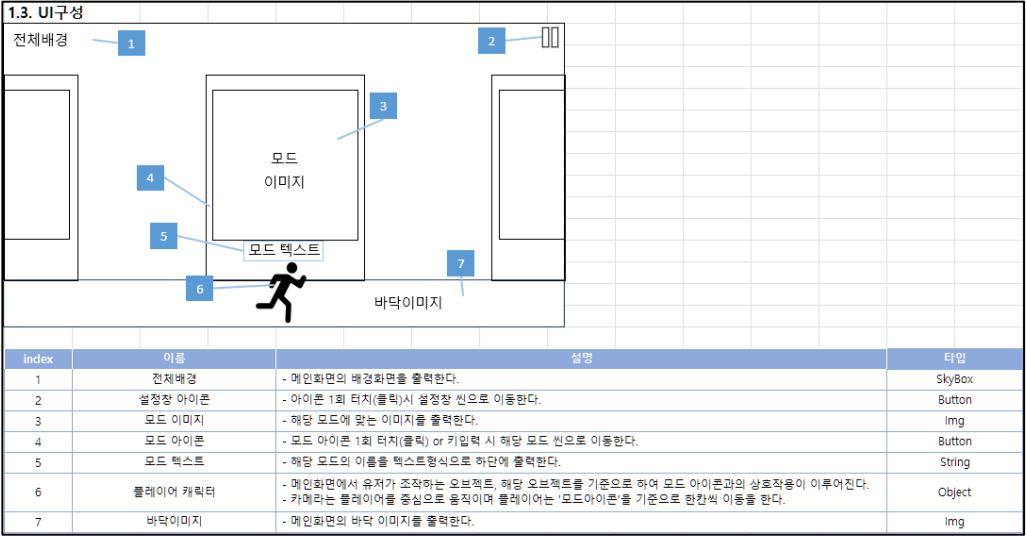
메인화면 / 챕터 선택 / 스테이지 선택 / 스테이지 진입 / 인게임 / 결과창

U별 기획 작성내용

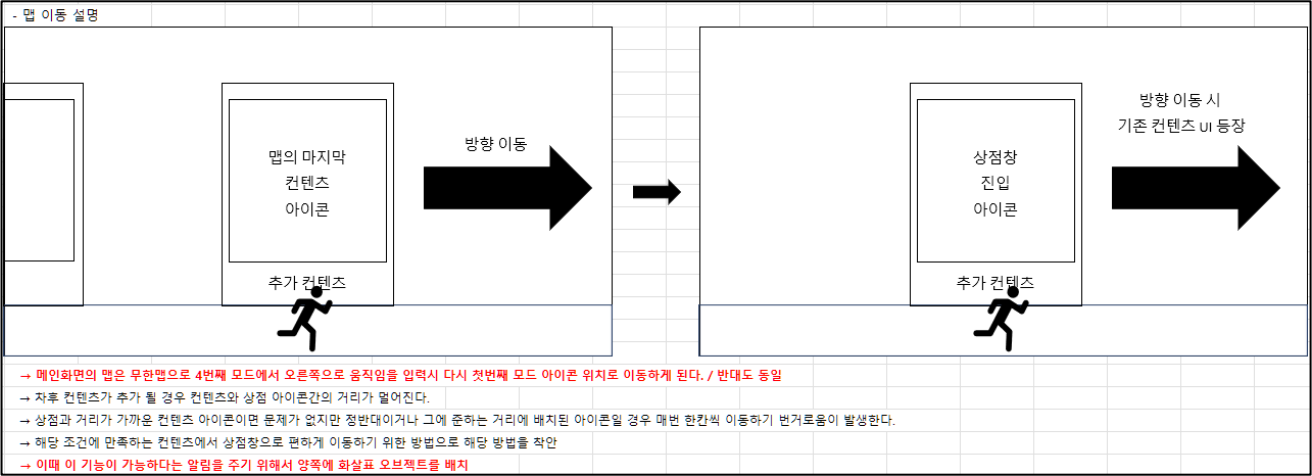
기획의도 / 레퍼런스 이미지 / UI구성 / 씬 오브젝트 배치 위치 / 씬 이동 설명 / 특이사항 / 결과물 / 적용 사운드 / 플로우차트

참고자료

참고 자료 페이지 '나'기획서 링크참조'



메인화면씬구성요소화면



메인화면씬상호작용기획

기획서작성시

메인화면 / 챕터 선택 / 스테이지 선택 / 스테이지 진입 / 인게임 / 결과창

U별 기획 작성내용

기획의도 / 레퍼런스 이미지 / UI구성 / 씬 오브젝트 배치 위치 / 씬 이동 설명 / 특이사항 / 결과물 / 적용 사운드 / 플로우차트

참고자료

참고 자료 페이지 'U기획서 링크참조'

아이디어 : 맵 생성이 끝났으니 플레이어 캐릭터를 이용한 게임 플레이를 해당 노래의 박자에 맞출 수 있게 구현하자

→ 리듬게임의 특성상 0.1ms 차이에도 어색함이 발생하는 점을 감안하여 제작

→ 딜레이의 발생 원인

1. 컴퓨터가 사운드를 내보내더라도 내보낸 소리가 스피커에서 나올 때 까지 걸리는 시간
2. 인간이 음악을 듣고 키를 누르는데 까지 걸리는 시간
3. 입력장치에서 보낸 신호가 컴퓨터에서 해당 키를 인식하는데 걸리는 시간

→ 유저가 버튼을 눌러 컴퓨터가 인식한 시간에서 컴퓨터가 일정 시간마다 비트를 내보낸 시간의 차를 지연시간으로 함



해당 시스템을 통해 구현한 '레이턴시 콘텐츠'

4

컨텐츠

아이디어 : 음정과 박자에 대한 지식이 부족하니까 BMS파일을 게임 내에서 만들지 말고 외부 프로그램을 이용하여 제작하자.

- 'osu!'라는 게임에서 노트 타이밍과 시간이 적힌 BMS 파일 제작 후 .osu 확장자 파일을 추출
- 추출한 파일에서 본 게임에 필요한 요소들을 최적화 하여 추출 할 수 있는 프로그램을 구현 하여 .txt 확장자 BMS파일 추출
- 파일 안의 TimingPoints 부분으로 BPM을 계산, HitObjects 부분으로 위치 및 노트 타입의 데이터를 읽음
- BPM을 기반으로 위치와 시간 데이터를 바탕으로 시간을 비트로 변환

[HitObjects]

384,192,15,5,0,0:0:0:0:
384,192,515,1,0,0:0:0:0:
384,192,1015,1,0,0:0:0:0:
384,192,2015,1,0,0:0:0:0:
384,192,2265,1,0,0:0:0:0:
384,192,2515,1,0,0:0:0:0:
384,192,3515,1,0,0:0:0:0:
384,192,3765,1,0,0:0:0:0:
384,192,4015,1,0,0:0:0:0:

osu!를 이용하여 제작한 비트맵 형식

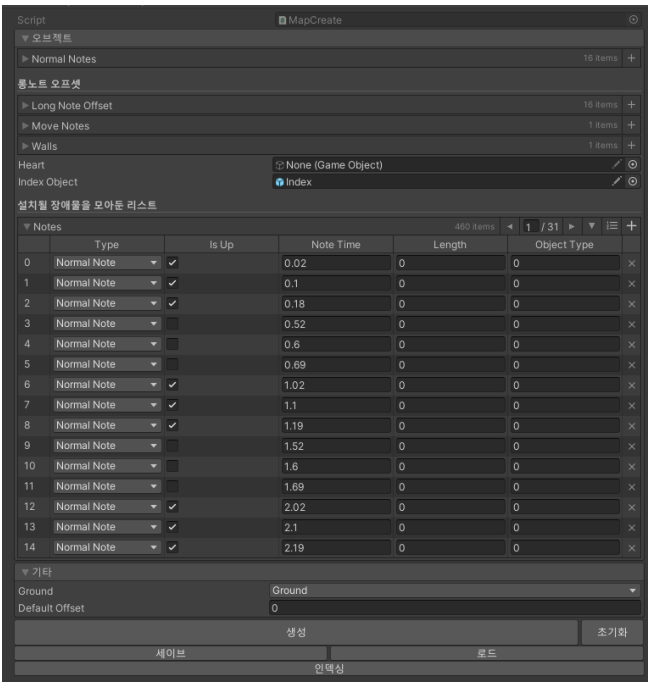


0.03
1.03
2.03
4.03
4.53
5.03
7.03
7.53
8.03
9.03
10.03
11.03
12.03
13.03
14.03
15.03

변환 프로그램을 통해 추출 된 비트맵 형식

아이디어 : BMS 파일을 본 게임에 적용 시 차후 업데이트로 인한 요소 변화를 위해 관리하기 쉽도록 시스템을 구성하자.

- 추출된 .txt파일의 내용을 토대로 장애물을 적절한 위치에 배치하는 시스템 제작
- 각 테이블 리스트에 저장된 수치에 따라 노트의 종류와 배치 위치가 다름
- Type / Length / Note Time / Object Type 변수들을 통해 장애물의 길이, 위치, 형태 등을 다양하게 지정 및 관리 가능
- 초기화 버튼: 비트맵 txt의 비트맵을 리스트 초기상태로 지정 / 생성 버튼: 리스트에 저장된 데이터로 오브젝트를 생성 / 세이브, 로드 버튼: 작업한 테이블 리스트의 내용을 .csv 파일로 저장 및 로드



BMS파일 적용 및 관리를 위한 시스템 Unity 모듈화

0.03	NormalNote	TRUE	0	0
1.03	NormalNote	TRUE	0	0
2.03	NormalNote	TRUE	0	0
4.03	NormalNote	TRUE	0	0
4.53	NormalNote	TRUE	0	0
5.03	NormalNote	TRUE	0	0
7.03	NormalNote	TRUE	0	0
7.53	NormalNote	TRUE	0	0
8.03	NormalNote	TRUE	0	0
9.03	NormalNote	TRUE	0	0
10.03	NormalNote	FALSE	0	0
11.03	NormalNote	FALSE	0	0
12.03	NormalNote	FALSE	0	0
13.03	NormalNote	FALSE	0	0
14.03	NormalNote	FALSE	0	0
15.03	NormalNote	FALSE	0	0
16.03	NormalNote	FALSE	0	0
17.03	NormalNote	TRUE	0	0

해당 시스템을 통해 .csv 확장자로 저장된 BMS파일

5

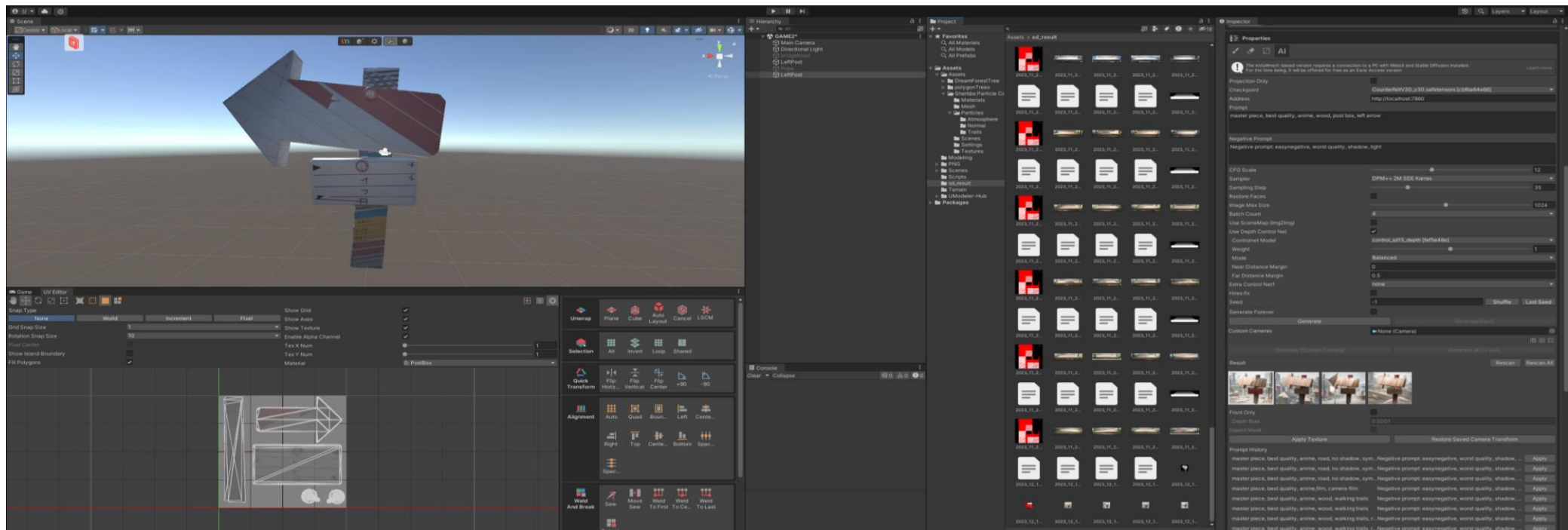
사운드

6

그래픽

아이디어 : 3D그래픽이기에 컨셉에 맞는 스타일의 모델 텍스처 리소스를 구하기 어려우니 AI를 활용하여 texture를 제작해보자

- 'Umodeler'을 활용하여 Unity를 3D modeling을 가능하게 하는 환경으로 변환
- 게임 내 필요 3D model을 3ds Max를 통해 수정 및 변형
- 변형된 model을 Unity에 불러와 UV를 펼침
- Unity에 Stable Diffusion를 연동하여 입력된 스타일(ex. Anime풍)의 terxture 파일을 펼친 UV에 맞춰 생성
- 완성된 texture를 Export를 하여 Material로 제작 후 본 게임의 3D model에 적용



Unity에서 AI를 활용하여 texture파일을 제작하는 이미지

감사합니다

nocash1234@naver.com/010-2950-3134