주 형탁

Mail: juht97@naver.com

한국산업기술대학교 게임공학과  
클라이언트 프로그래머 부분

포트폴리오

**목차**

[벨레스(Veles) -졸업작품 2](#_Toc89198374)

[게임 소개 2](#_Toc89198375)

[구현: 애니메이션 재생 모듈 3](#_Toc89198376)

[자료구조 4](#_Toc89198377)

[실행 흐름 4](#_Toc89198378)

[맵 제작 5](#_Toc89198379)

[Unity Terrain을 이용하여 맵 제작. 5](#_Toc89198380)

[봄버맨 모작 -네트워크 게임프로그래밍 프로젝트 6](#_Toc89198381)

[게임 소개 6](#_Toc89198382)

[클라<->서버 데이터 전송 7](#_Toc89198383)

[슈퍼마리오 월드 모작 -2D 게임프로그래밍 프로젝트 8](#_Toc89198384)

[게임 소개 8](#_Toc89198385)

[Ride & joyful (승마 정보 애플리케이션) – 스마트폰 프로그래밍 9](#_Toc89198386)

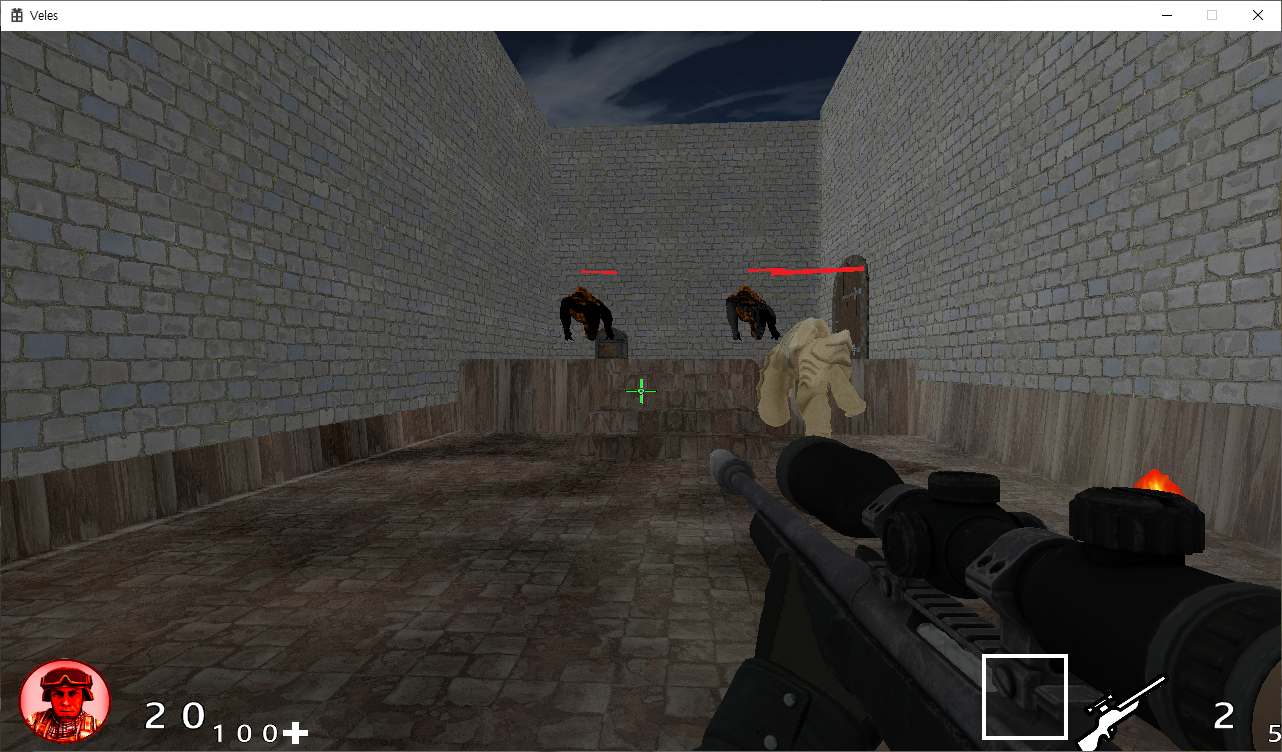
[프로그램 소개 9](#_Toc89198387)

[어려웠던 점/해결 방법 10](#_Toc89198388)

[veles – 스켈레톤 메시 게임 오브젝트 깨짐 현상 10](#_Toc89198389)

[veles - 사용하기 힘든 라이브러리 Directx12 10](#_Toc89198390)

# 벨레스(Veles) -졸업작품



## 게임 소개

게임 장르: 3D 멀티 로그라이크 FPS던전 어드벤쳐

프로젝트 소스: <https://github.com/dnjsgnlwo/BOBY.git>

소개 영상: <https://youtu.be/G-9cBvwS9z8>

사용 언어: C++

개발 환경: Visual Studio 2019, DirectX12, Assimp, FMOD

제작 인원: 3명

역할: 서브 클라이언트(Assimp를 이용한 애니메이션 재생 라인 구현 및 맵 제작)

제작 기간: 2020-12-26 ~ 2021-9-1

게임 내용: 로비에서 인원을 모아 무기를 정한 후, 게임이 시작되면 몬스터를 잡아가며 방을 클리어하고 아이템을 모아 보스를 잡아 던전을 클리어하기.

## 구현: 애니메이션 재생 모듈

샘플 1 애니메이션 계산 함수의 일부

asClip\* animationClip = animationClips[animationClipNumber]; **// 1)**

Matrix4x4 nodeTransform = Matrix4x4(node->m\_nodeTransform);

asClipNode\* nodeAnim = FindNodeAnim(animationClip, node->GetName());

if (nodeAnim) {

Vector3 transform;

Quaternion rotation;

Vector3 scale;

**// 2)**

transform = CalcInterpolatedVectorFromKey(animationTime, nodeAnim->KeyframeTranslation.size(),

nodeAnim->KeyframeTranslation);

rotation = CalcInterpolatedQuaternionFromKey(animationTime, nodeAnim->KeyframeRotation.size(),

nodeAnim->KeyframeRotation);

scale = CalcInterpolatedVectorFromKey(animationTime, nodeAnim->KeyframeScale.size(),

nodeAnim->KeyframeScale);

nodeTransform = Matrix4x4(transform, rotation, scale).transpose();

node->m\_Transform = nodeTransform.transpose();

}

Matrix4x4 globalTransform = parentTransform \* nodeTransform;

**// 3)**

for (int k = 0; k < rootNode->rootBoneInfos.size(); k++)

{

if (rootNode->rootBoneInfos[k].boneName == node->GetName())

{

rootNode->rootBoneInfos[k].finalTransform = globalTransform \* track->animationWeight;

break;

}

}

1. 실행하고자 하는 애니메이션의 클립 번호를 통해 정보 찾기
2. Transform, Scale, Rotation값을 시간의 흐름에 따라 각각 계산한 후, 부모의 nodeTransform과 계산하기
3. 최종 계산된 본의 transform와 가중치를 계산하여 알맞은 애니메이션 Transform 구현

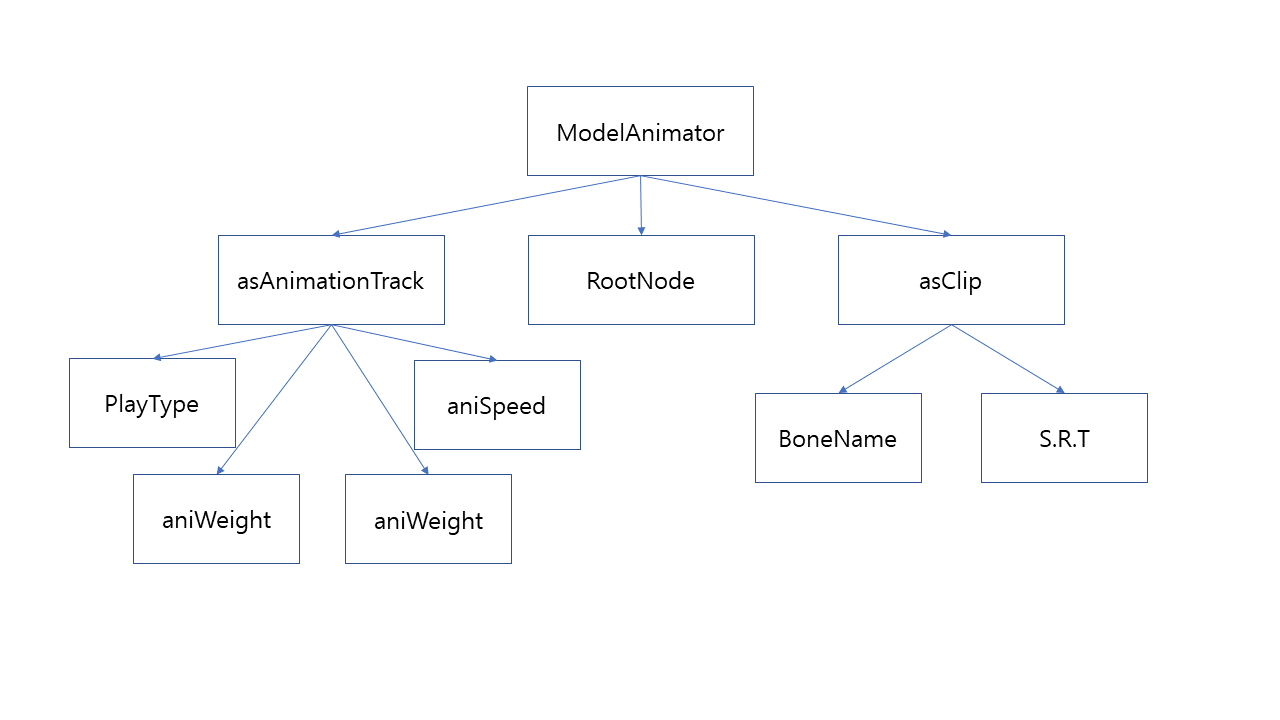


그림 1 내부 자료 구조

### 자료구조

1. **ModelAnimator 클래스**

각 본의 FinalTransform 계산 및 Update를 담당하는 클래스

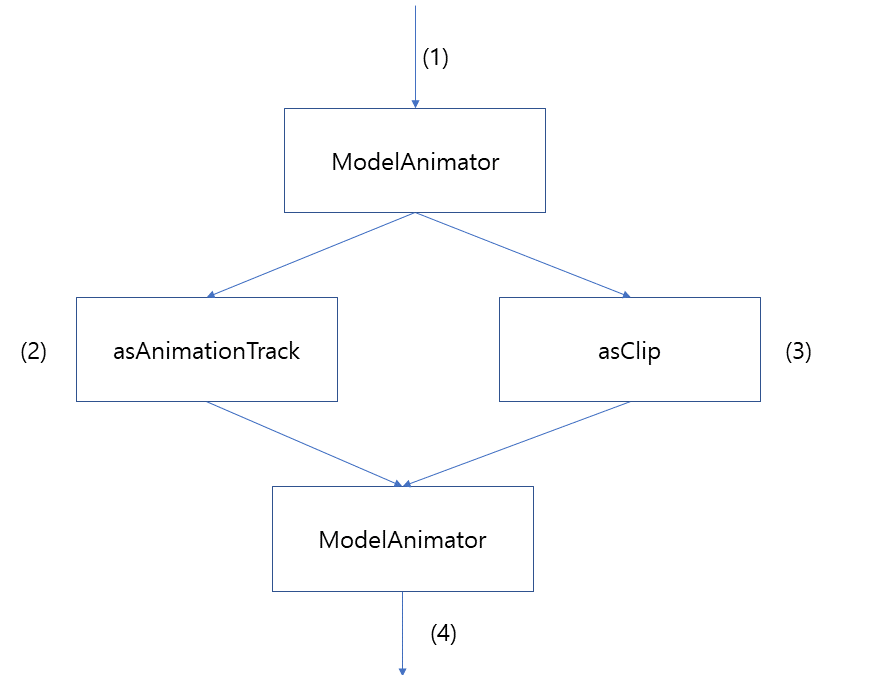
1. **asAnimationTrack 클래스**

애니메이션의 속도, 시작시간, 종료시간, 가중치, 재생 타입을 저장하여 관리하는 클래스

1. **asClip 클래스**

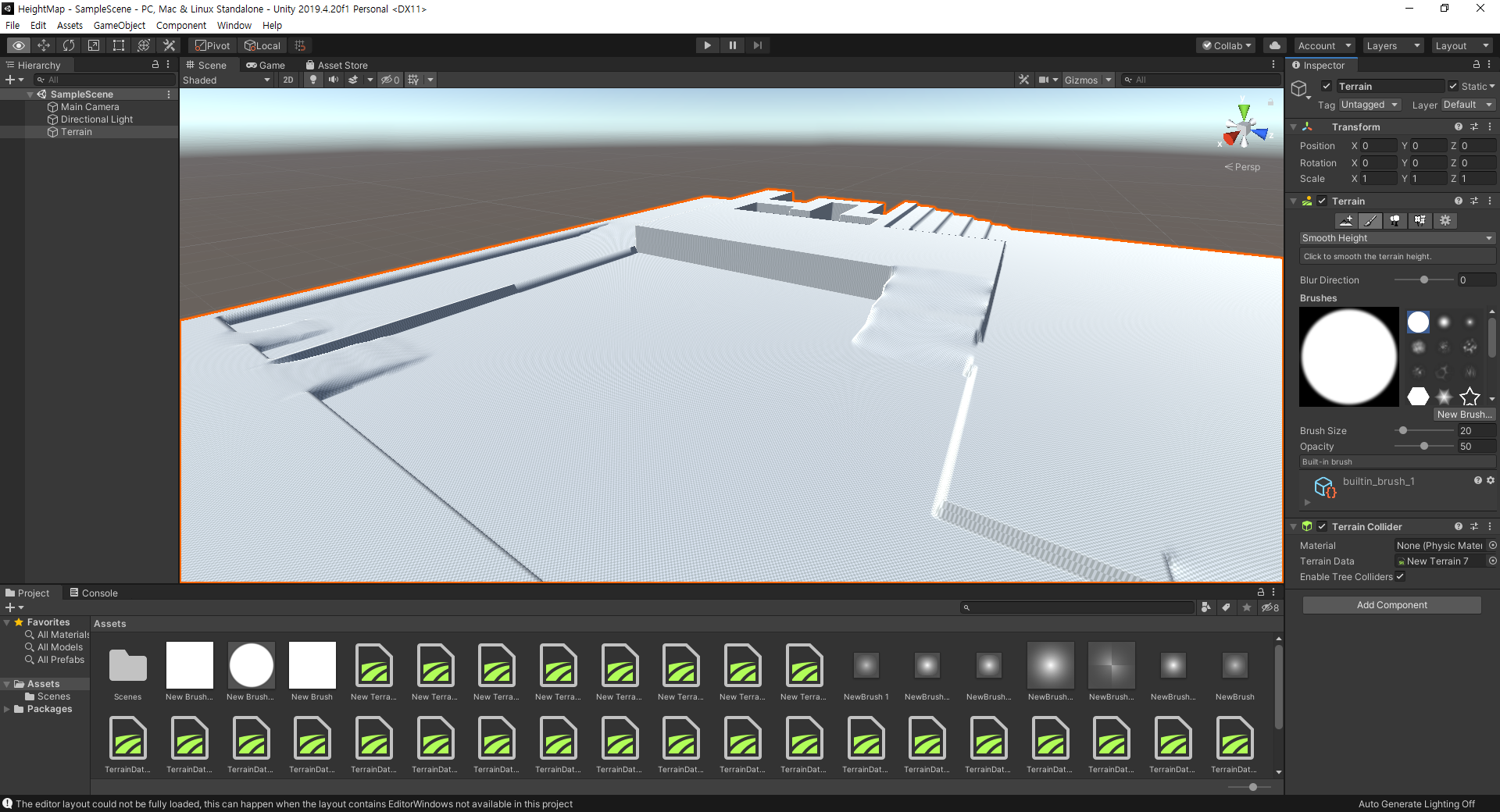
애니메이션의 이름, 전체 프레임의 수, 애니메이션의 길이, 키프레임을 저장하는 클래스

### 실행 흐름

****

1. CFbxModelObject에서 실행하고자 하는 애니메이션의 트랙 번호를 animationManual에서 찾아 ModelAnimator에 넘겨줌.
2. ModelAnimator에 넘어온 번호를 통해 asCilp내의 애니메이션이 저장된 위치, 애니메이션의 가중치, 재생 속도, 시작시간, 종료시간, 재생 방식 등의 정보를 찾아서 ModelAnimator에 넘겨줌.
3. asClip에서 애니메이션의 각 본의 Transform, Rotation, Scale값을 찾아 ModelAnimator에 넘겨줌.
4. asCilp과 asAnimationTrack에서 전달한 값들로 현재 각 본들의 FinalTranform을 계산하여 CFbxModelObject로 전달함.

## 맵 제작



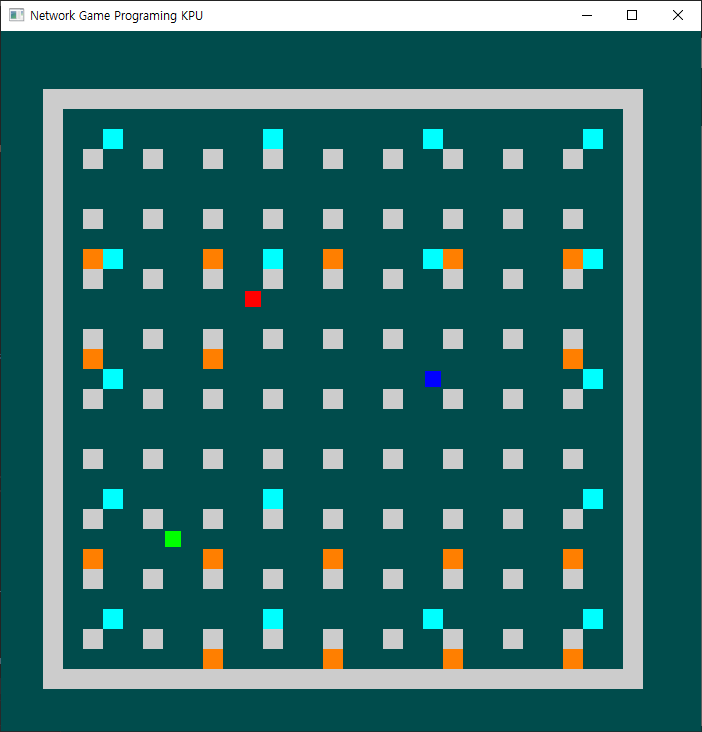
### Unity Terrain을 이용하여 맵 제작.

맵의 Raw이미지를 export한 후 클라이언트에서 파일을 읽어와서 적용함.

1. Raw이미지를 사용하는 이유는?

JPG형식을 사용할 경우 용량을 줄일 수 있지만 저장하는 과정에 있어 압축으로 인해 Terrain에 적용했을 때 지형이 부자연스럽게 나올 수 있고, 투명한 배경이 필요치 않기에 PNG대신 Raw이미지를 사용함.

# 봄버맨 모작 -네트워크 게임프로그래밍 프로젝트



## 게임 소개

게임 장르: 아케이드 액션

프로젝트 소스: <https://github.com/hyungtak/NGP_2020_2.git>

사용 언어: C++, OpenGL, TCP/IP

개발 환경: Visual Studio 2019

제작 인원: 3명

담당 부분: 로비 구현, 게임 기믹(폭탄) 구현, 클라쪽 서버연결 구현

제작 기간: 2020-11-26 ~ 2020-12-10

게임 내용: 봄버맨 모작

## 클라<->서버 데이터 전송

샘플 2 SimpleGame.cpp의 일부

if (gameStart)

{

//SendToServer()

retval = send(sock, (const char\*)(&g\_inputs), sizeof(g\_inputs), 0);

if (retval == SOCKET\_ERROR) {

err\_display("GameScene send()");

}

//RecvFromServer()

MapData mapData[MAP\_SIZE][MAP\_SIZE];

retval = recvn(sock, reinterpret\_cast<char\*>(&mapData), sizeof(mapData), 0);

if (retval == SOCKET\_ERROR) {

err\_display("MapData recv()");

}

g\_game->SetMapData(mapData);

g\_game->RendererScene();

}

else

{

g\_lobby->RendererScene();

//RecvFromServer()

retval = recvn(sock, reinterpret\_cast<char\*>(&gameStart), sizeof(gameStart), 0);

if (retval == SOCKET\_ERROR) {

err\_display("gameStart recv()");

}

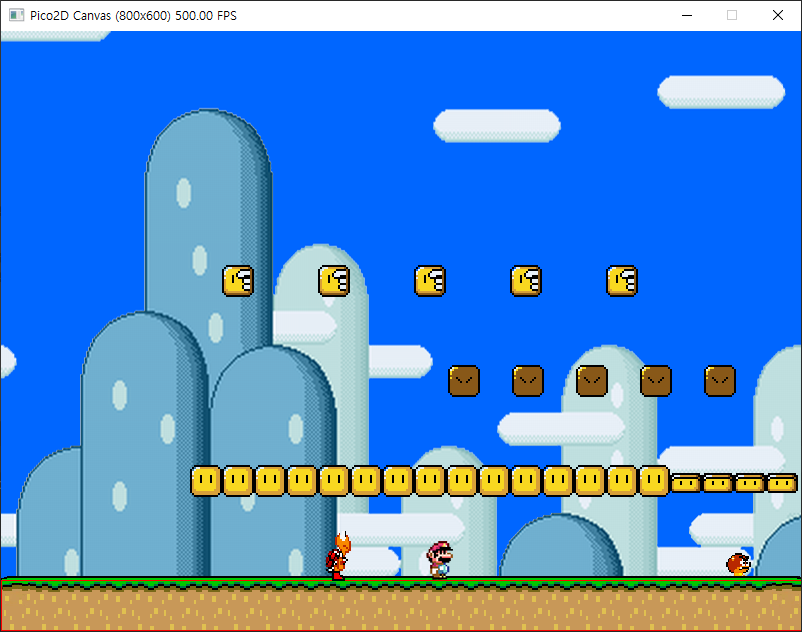
}

플레이어 3명이 로비에 접속하면 자동으로 게임이 시작되도록 구현했기에, 게임을 실행하게 되면 로비 씬을 출력하고, 서버에서 3명의 플레이어가 로비에 접속했음을 알리는 패킷을 기다립니다.

패킷 값이 달라지게 되면, 게임 씬으로 변경하고, 플레이어가 입력한 키 패킷을 서버로 전달하며,

서버에서 보내주는 맵 데이터 패킷을 받아와 패킷 값에 맞춰 화면을 클라이언트에 출력해줍니다.

# 슈퍼마리오 월드 모작 -2D 게임프로그래밍 프로젝트



## 게임 소개

게임 장르: 플랫폼 게임

프로젝트 소스: <https://github.com/hyungtak/SuperMarioWorld2016180040.git>

사용 언어: Python 3.9 64bit, pico2d

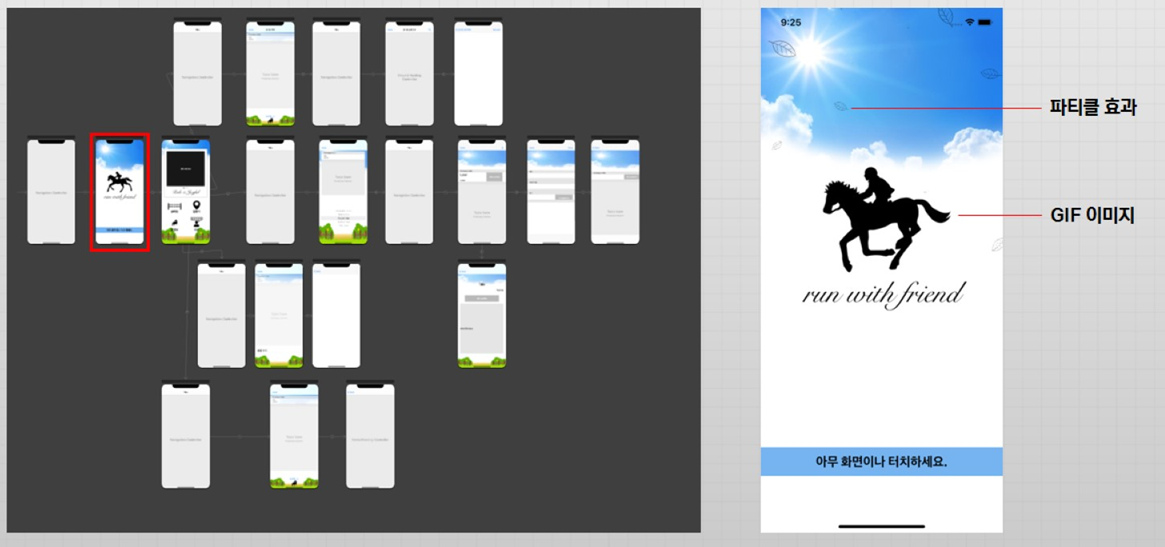
개발 환경: Pycharm, GitHub Desktop

제작 인원: 1인

제작 기간: 2021-10-22 ~

게임 내용: 슈퍼마리오월드 모작

# Ride & joyful (승마 정보 애플리케이션) – 스마트폰 프로그래밍



## 프로그램 소개

프로그램 내용: XCODE를 사용하여 공공데이터포털의 OpenAPI중 한국 마사회의 OpenAPI를 이용하여 승마 스포츠에 대한 사람들의 접근성을 완화하고자 만든 승마관련 정보 검색 애플리케이션입니다.

프로젝트 소스: <https://gitlab.com/asdssp/smartphoneproject.git>

사용 언어: IOS XCODE

개발 환경: IOS, XCODE, GitLab

제작 인원: 2인

제작 기간: 2021-02-16 ~ 2021-06-13

최종 발표: <https://youtu.be/6jRTnmY_Kgw>

프로젝트 파일: <https://drive.google.com/file/d/1xPvaR4BGYdf0lCuBuZjwWdjDcKp8cMqR/view?usp=sharing>

담당 부분: 메인화면, 리뷰, 길찾기 페이지. 앱 아이콘, 버튼 사운드 추가.

# 어려웠던 점/해결 방법

## veles – 스켈레톤 메시 게임 오브젝트 깨짐 현상

애니메이션을 오브젝트에 적용하기 위해 본의 가중치를 추가하여 vertex의 좌표를 계산을 하니 일부 모델이 깨지는 문제가 있었습니다. 본의 가중치를 제외한 T-Base자세에서는 문제가 발견되지 않고, 일부 모델에서만 문제가 발견되어 원인 파악이 매우 힘들었습니다. 문제의 원인은 애니메이션 계산을 할 때 한 vertex당 최대 4개의 본만 가중치를 받도록 했었는데, 일부 모델에서는 최대 개수를 넘는 20, 32개 이상의 가중치를 가지고 있어서 생긴 문제였습니다.

이후 본의 가중치를 들어오는 순서대로 4개를 받는 게 아닌, 가장 높은 가중치부터 8개를 받아 애니메이션을 계산하게 하여 문제를 해결하였습니다.