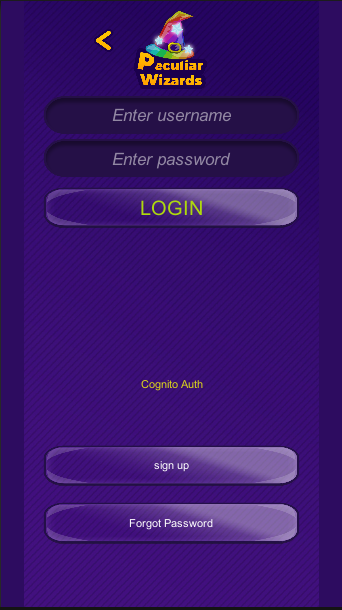
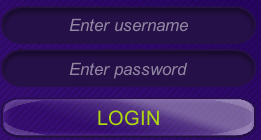
# 앱 설명서

## 로그인 씬

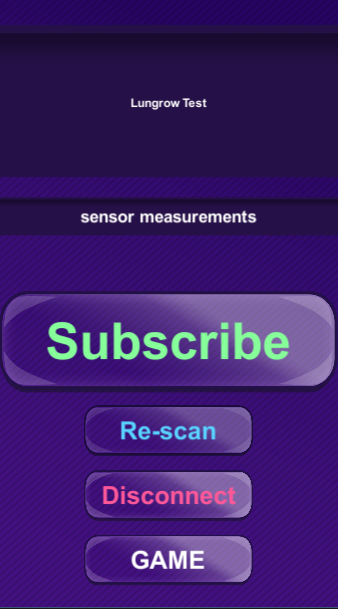


처음 어플을 실행하면 나오는 로그인 화면이다.



Enter username 부분에 가입한 이메일을 적고 Enter password 부분에 비밀번호를 적은 후 LOGIN 버튼을 누르면 로그인 하여 다음 씬으로 이동할 수 있다.

## 기기 연결 씬



로그인 하면 나오는 기기를 연결하는 씬으로 스마트폰 블루투스를 킨 상태면 씬에 진입하면 자동으로 기기를 검색하고 키지 않으면 오류가 발생하여 블루투스를 활성화 시켜주어야 한다.



블루투스를 킨 후 Re-scan 버튼을 누르면 기기를 검색한다.



기기 검색에 성공 후 Subscribe 버튼을 누르면 기기와 연결을 하고 연결 완료 후 기기를 한 번 불면 기기에서 호흡과 관련된 정보들을 받아올 수 있다.



Disconnect 버튼을 누르면 연결된 기기를 해제할 수 있다.



기기 연결 후 GAME 버튼을 누르면 게임 시작씬으로 이동할 수 있다.

## 게임 로비 씬





게임 로비 씬에서는 게임의 난이도를 선택하여 게임을 시작할 수 있다.



버튼으로 기기 연결 화면으로 돌아가서 기기 연결을 다시 할 수도 있다.

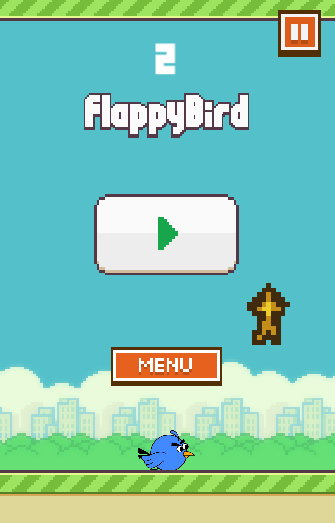
## 게임 씬



EASY모드를 선택하여 게임을 실행한 화면이다. 날숨으로 새를 위로 올라가게 할 수 있고 들숨으로 새를 아래로 내려가게 할 수 있다.

EASY모드에서는 현재 코인을 먹으면 스코어가 증가하게 되어 있고 바닥에 닿아도 게임오버가 되지 않게 되어 있다.



왼쪽 위의 일시정지 버튼을 누르면 이와 같이 일시정지 화면이 나온다.



위의 버튼을 누르면 게임을 재개하여 이어서 플레이할 수 있다.



메뉴 버튼을 누르면 게임을 종료하고 만족도 조사씬으로 갈 수 있다.

## 만족도 조사 씬

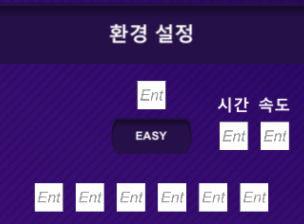


만족도 조사 씬에서는 게임의 만족도를 조사할 수 있고 해당하는 만족도 버튼을 누르면 조사가 완료된다. 완료된 후에는 게임 정보와 만족도 정보, 로그인 정보를 취합하여 POST로 DB에 보내게 된다. 그리고 다시 게임 로비 씬으로 돌아가 새로운 게임을 시작할 수 있다.

## 환경 설정 씬



환경 설정 씬에서는 새의 속도, 각 난이도의 동전, 파이프의 생성 위치, 생성 간격 시간, 속도를 설정할 수 있다. 각 입력창에 숫자를 입력한 후 밑에 있는 설정 버튼을 누르면 적용이 완료된다. 생성 위치의 추천 값은 -0.6 ~ 0.6 사이이다. 설정 버튼을 눌러 설정을 완료한 후 위에 있는 주황색 돌아가기 키를 누르면 다시 게임 로비 씬으로 돌아갈 수 있다.



환경 설정 밑에 있는 입력창 - 새의 속도

각 난이도 옆에 있는 첫번째 입력창 - 동전 혹은 파이프 생성 시간

각 난이도 옆에 있는 두번째 입력창 - 동전 혹은 파이프 속도

각 난이도 밑에 있는 6개의 입력창 - 동전 혹은 파이프의 생성 위치(순서대로 생성)

# 2DFlappyBird\_ver0.0.5 변경점

## 이벤트를 관리하는 EventManager 게임 오브젝트를 로그인씬에 추가

## 

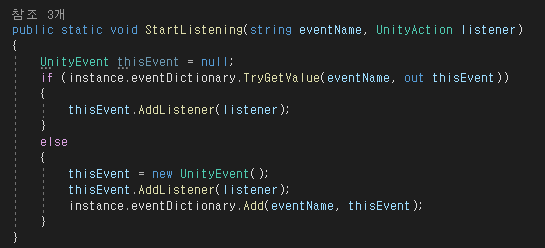
EventManager에는 cshEventManager 스크립트만 존재한다. DontDestroyOnLoad

오브젝트이므로 처음 로그인씬에서 만들어지면 그 후 씬에서도 계속 유지된다.



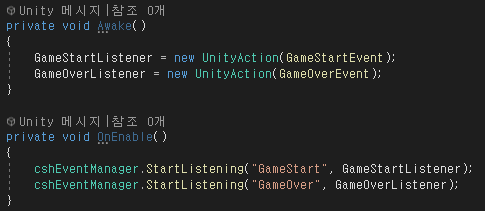
딕셔너리오브젝트인 eventDictionary에 이벤트의 이름과 유니티 이벤트 오브젝트를

저장한다.



딕셔너리에 이벤트를 등록하는 방법은 StartListening 함수를 통해 다른 스크립트에서 만든

이벤트를 등록할 수 있다.



예시로 GameManager 오브젝트의 컴포넌트인 cshGameManager 스크립트에서 이처럼

게임 시작과 게임 오버 이벤트 함수를 만든 후 유니티 액션으로 만든다. 그 후

StartListening함수로 이름을 지정하여 만든 유니티 액션을 이벤트로 등록할 수 있다.

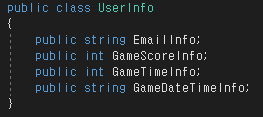




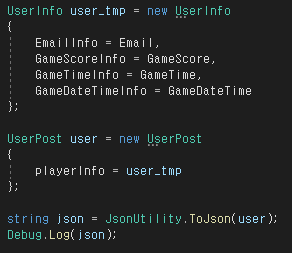
딕셔너리에 등록한 이벤트의 실행은 TriggerEvent 함수를 이용하여 할 수 있다. 이벤트 이름을

변수로 넣어주면 invoke()함수로 해당 이벤트를 실행시킨다.

## 플레이어의 Email, 게임플레이 시간, 점수, 게임실행 시각을 기록하여 json 형식으로 기록



위에서 만든 cshEventManger 스크립트에 유저의 정보를 기록할 수 있는 Userinfo라고 하는 클래스를 추가하였다. 이메일은 로그인할 때 이벤트로 로그인한 이메일 정보를 기록하고 게임과 관련된 게임 플레이시간, 점수, 플레이 시각 정보들은 게임이 종료되었을 경우 게임 오버 이벤트로 기록한다.



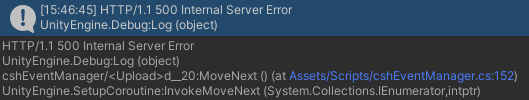
게임 오버 이벤트가 실행되면 정보를 기록한 유저 클래스를 JsonUtility.ToJson 함수를 이용하여 json 형태로 변경한다.

## 변경한 json 형식을 UnityWebRequest를 이용하여 통신 (현재 에러 존재)



유니티에서 HTTP 통신을 하기위해 사용하는 UnityWebRequest를 사용하였다.

Upload 함수에서는 보내는 URL과 json형식의 문자열을 받아 Post해준다.



현재 유저 정보를 json파일로 변경은 문제 없이 되지만 Post 했을경우 500 internal Server Error가 발생한다.

## 게임의 변경점



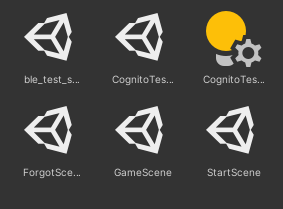
기존에 점수를 얻는 아이템으로 사용하던 동전을 별로 변경.



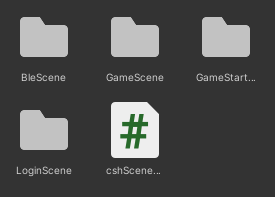
점수를 얻는 아이템 획득 애니메이션을 반짝이는 별 애니메이션으로 변경.

# 2DFlappyBird\_ver0.0.6 변경점

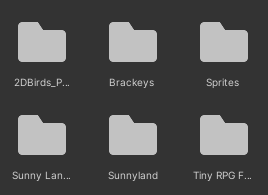
## 안쓰는 씬 및 스크립트와 에셋들 정리



씬은 Scenes 폴더 안에 현재 사용하는 씬인 CognitoTestScene과 ForgotScene(로그인 씬), ble\_test\_scence1(기기 연결 씬), StartScene(게임 로비씬), GameScene(게임 씬)으로 정리했다.

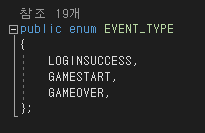


스크립트는 Scripts 폴더 안에 각 씬에 해당하는 폴더를 만들어 스크립트가 사용되는 씬에 맞는 폴더에 넣고 여러 씬에서 사용하는 스크립트만 꺼내놓아 좀 더 스크립트를 찾기 쉽게 정리했다.



에셋 스토어에서 받은 에셋들은 AssetsStore라는 폴더를 만들어 이 곳에 모아 원하는 에셋을 편하게 찾을 수 있도록 했다.

## 이벤트를 관리하는 EventManager 수정



각각의 이벤트 타입을 관리해 줄 수 있는 EVENT\_TYPE 열겨형을 추가.

이 곳에 원하는 이벤트 이름을 추가하여 이벤트를 관리할 수 있다.



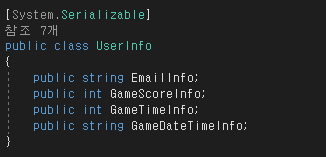


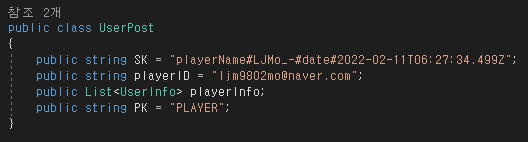
이벤트를 추가할 때 EVENT\_TYPE으로 이벤트를 구별하고 딕셔너리에 EVENT\_TYPE으로 저장하므로 이벤트 구별과 저장이 더욱 한 눈에 알아보기 쉬워졌다.



이벤트의 실행 또한 각 스크립트에서 만들고 딕셔너리에 저장한 이벤트를 이벤트 타입으로만 불러와서 실행만 시켜주면 된다.

## Json 형식 데이터 POST 성공





받을 정보들을 정리할 수 있는 UserInfo 클래스를 만들고 보내는 형식에 맞추기 위한 UserPost클래스를 생성하였다.

클래스를 list로 만들고 다시 한 번 json 형식으로 바꾸어 주기 위해 [System.Serializable]를 사용하여 list로 만들 클래스를 직렬화 시켜 주었다. 이 부분이 빠진다면 list를 사용하여 json 형식으로 변경할 때 list 부분이 변경이 되지 않아 list 부분이 빠지거나 오류가 날 것이다.



만든 이벤트로 받아온 정보들을 UserInfo 클래스로 정리하고 이걸 다시 한 번 보내야 하는 형식에 맞추어 UserPost 클래스에 list로 추가한 후 json형식으로 변경하여 주었다.



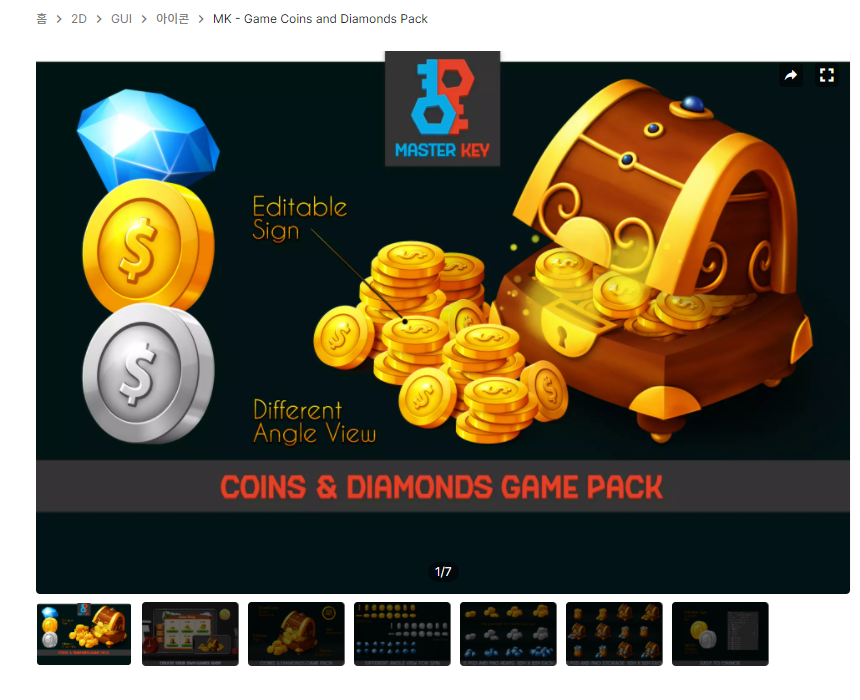
Json 형식의 데이터를 post하기 위하여 UnityWebRequest를 이용하여 통신을 했다.

이 부분에서 주의해야할 점이 json 형식의 데이터는 바로 UnityWebRequest.Post해서 url로 Json 데이터를 보내면 괄호와 같은 문자는 url에 사용할 수 없어 변경되어 보내지므로 Json 데이터가 제대로 보내지지 않는다.

그렇기에 json 파일을 byte로 변경후 UnityWebRequest로 post를 새로 만들고 uploadHandler를 이용하여 byte로 변경한 데이터를 추가하여 보내주어야 한다.

이 Upload 함수는 게임 오버 이벤트가 발생하면 실행된다. 게임 오버 했을 때 플레이한 사람 정보, 게임 정보를 전부 수집하여 보내 줄 수 있기에 이렇게 설정했다.

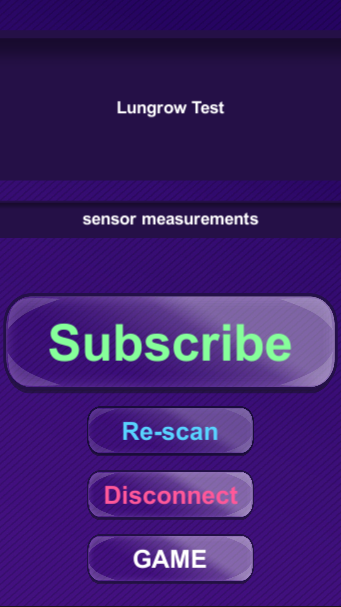
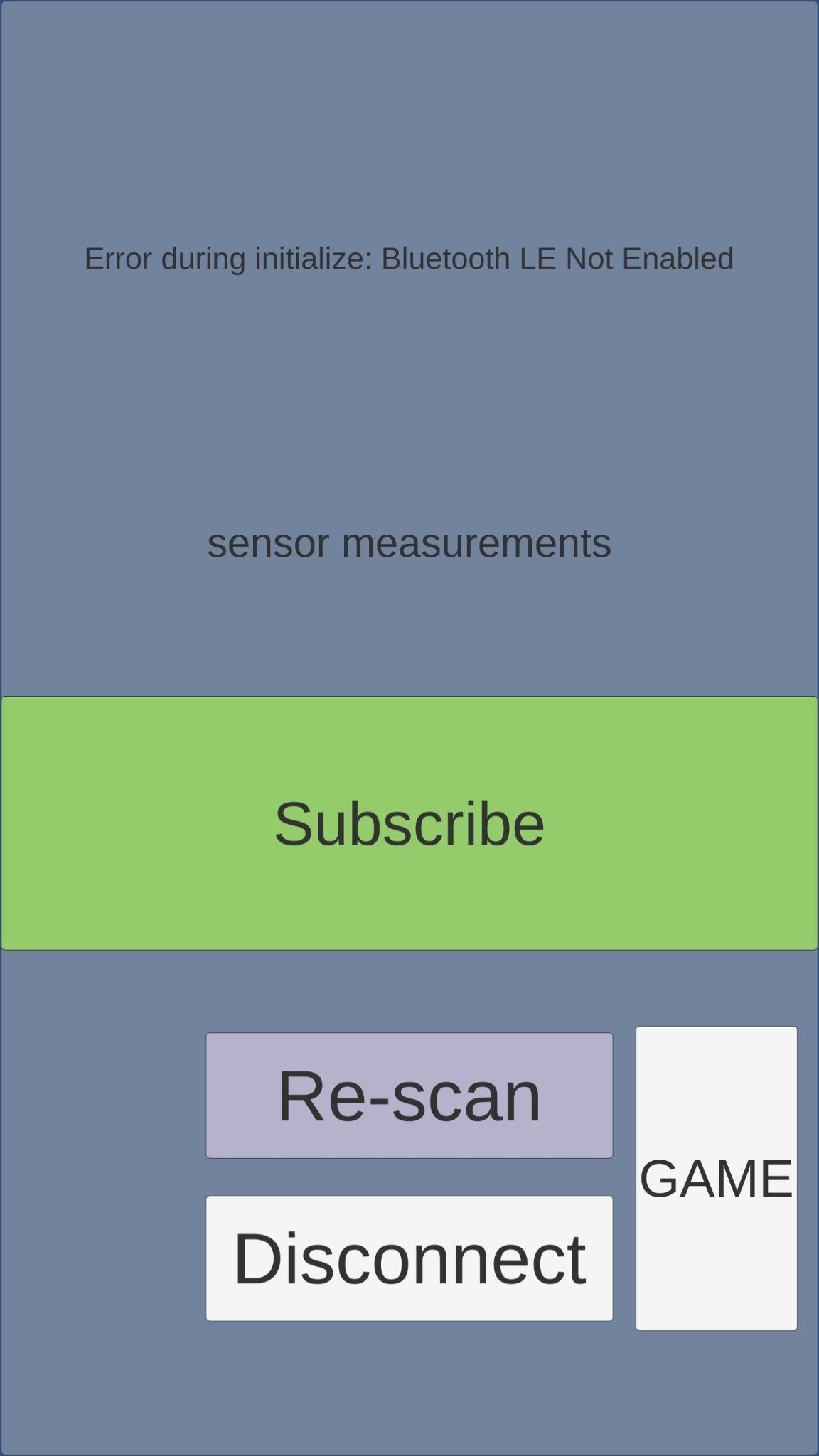
## 점수 아이템의 에셋 변경



현재 사용하고 있던  에셋에서 MK - Game Coins and Diamonds Pack(https://assetstore.unity.com/packages/2d/gui/icons/mk-game-coins-and-diamonds-pack-107491) 새로운 코인 에셋으로 변경했다.

 바꾼 코인의 애니메이션

## 게임 로비 씬 UI 변경



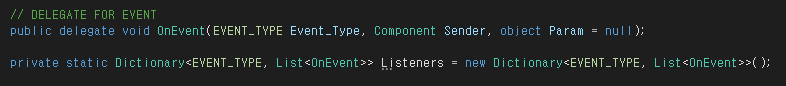
기존 로비씬인 좌측에서 로그인 씬 UI에 맞추어 우측과 같은 UI로 게임 로비씬의 UI를 변경했다.

기기 연결 상태를 나타내는 텍스트가 나타나는 부분은 배경을 두고 글자를 흰색으로 변경하여 더욱 보기 편리하게 변경했다.

각 버튼은 Subscribe는 초록색으로 그대로 유지하여 보기 편하게 변경했고 Re-scan은 하늘색으로 Disconnect는 연결을 끊는 것이므로 색을 맞추어 붉은색으로 변경하여 사람들이 쉽게 인식할 수 있도록 했다.

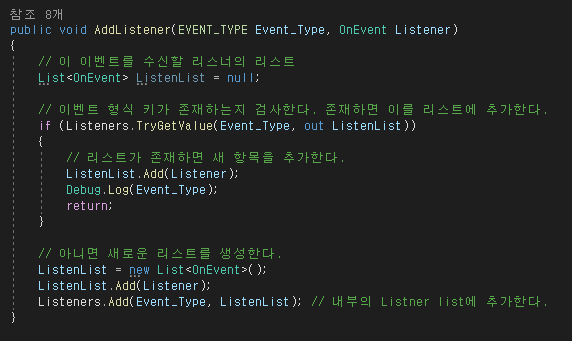
Game버튼은 기존 UI에서는 우측에 있어 일관성이 없었지만 바뀐 UI에서는 버튼을 통일하여 수직으로 나열함으로 UI의 통일성을 맞추어 주었다.

## 이벤트 매니저 재변경

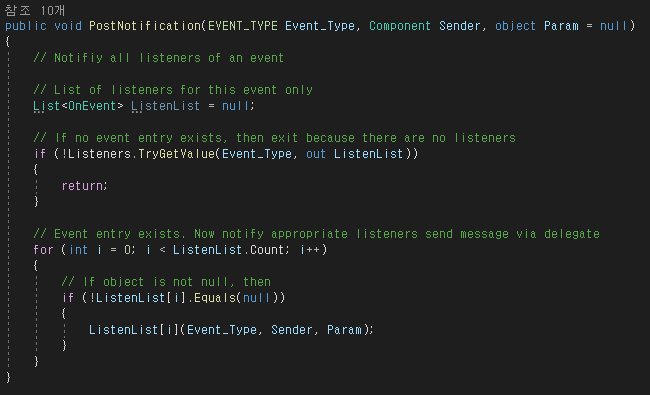


이벤트 타입을 받는 대리자 OnEvent를 생성하여 메소드를 대신 호출해 줄 수 있게 해주었다.

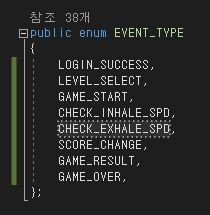
딕셔너리에는 이벤트 타입과 OnEvent를 저장한다.



AddListener는 이벤트 타입과 OnEvent를 받고 이것을 위에서 만든 딕셔너리에 저장한다.



PostNotification은 OnEvent로 참조할 수 있는 함수로 AddListener로 저장한 이벤트를 실행시켜 줄 수 있다.



현재 가지고 있는 이벤트 목록이다.

LOGIN\_SUCCESS는 로그인 성공했을시 실행하며 로그인한 이메일을 기록한다.

LEVEL\_SELECT는 게임 시작시 선택한 레벨을 게임씬에 있는 게임 매니저에 알려주어 어떤 레벨의 게임 오브젝트를 활성화 할지를 결정한다.

GAME\_START는 게임 시작시 실행하는 이벤트로 게임 시작 시간을 기록한다.

CHECK\_INHALE\_SPD, CHECK\_EXHALE\_SPD는 들숨 날숨의 세기가 일정량 이상 높아지면 실행하는 이벤트로 일정량 이상의 세기로 바람을 불면 스코어 주변에 애니메이션을 추가해준다.

 과거에 썼던 별 애니메이션을 점수 주변에 배치하여 세게 바람을 불수록 화려하게 보일수 있도록 해주었다.

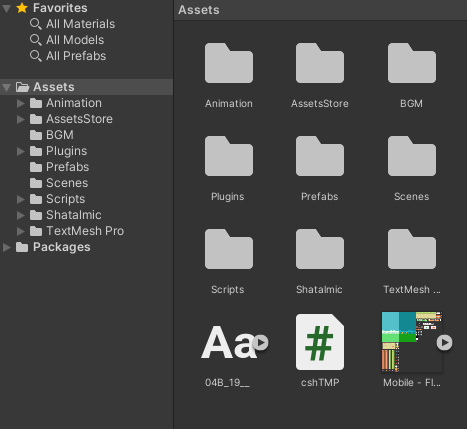
SCORE\_CHANGE는 코인과 새가 접촉하였을시 실행되고 점수를 획득하는 계산을 해주는 이벤트이다.

GAME\_RESULT는 게임 종료시 게임에 걸린 시간, 게임 플레이 시간, 게임 점수를 정리하여 user 정보에 저장해주는 이벤트이다.

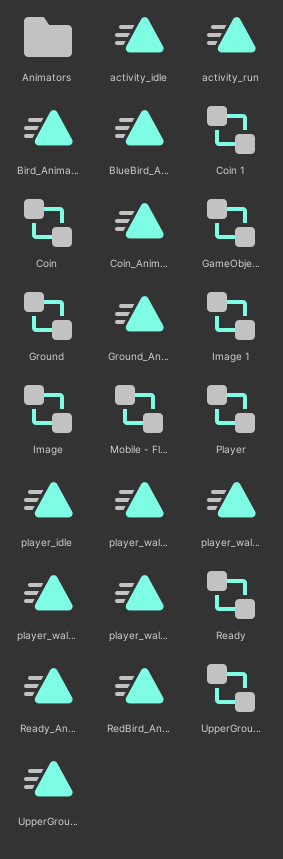
GAME\_OVER는 게임 종료시 게임 종료시간을 기록하고 게임을 일시정지 시키며 GAME\_RESULT이벤트를 실행시켜 게임 정보를 기록해주는 이벤트이다.

# 2DFlappyBird\_ver0.0.7 변경점

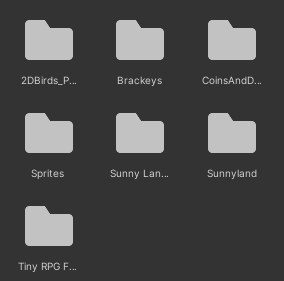
## 유니티 프로젝트의 각 폴더 설명



(프로젝트의 전체 폴더)



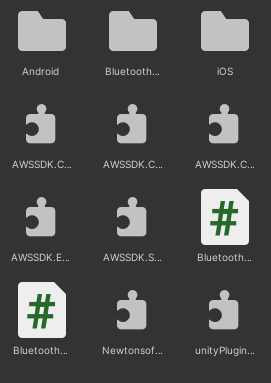
애니메이션 폴더 내부에는 게임에서 사용되는 각 애니메이션과 애니메이터가 들어있다.



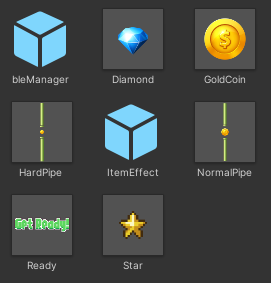
에셋 스토어 폴더에는 에셋 스토어에서 받은 에셋들을 각각의 폴더에 넣어 정리하고 있다.



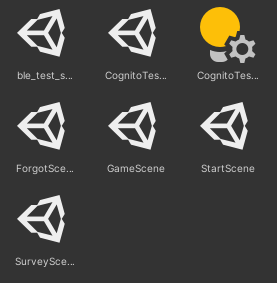
BGM 폴더에는 게임에서 사용하는 BGM과 동전 먹을 때 나오는 효과음이 들어있다.



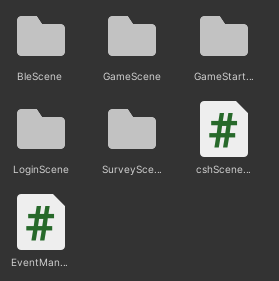
플러그인 폴더에는 프로젝트에서 사용하는 각 플러그인이 들어있고 특히 Newtonsoft플러그인은 로그인 하기 위하여 .NET 4 이상의 버전이 필요하고 프로젝트 세팅에서도 .NET의 버전을 변경해 주어야 한다.



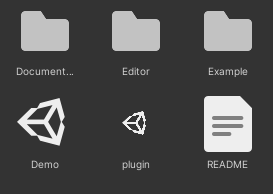
프리팹 폴더에는 프로젝트에서 사용하는 각 프리팹이 들어있다.



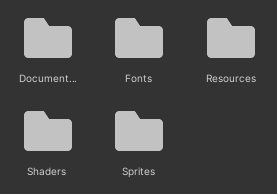
씬 폴더에는 사용하는 각 씬이 들어있다.



스크립트 폴더에는 사용하는 스크립트를 해당씬에 맞게 폴더에 정리하여 저장하고 있다.



Shatalmic 폴더에는 블루투스 통신을 하기위한 것들이 들어있다.



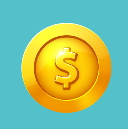
TextMesh Pro 폴더에는 사용하는 폰트와 관련된 것들이 들어있다.

## 각 난이도 표

| 난이도 | cshStart스크립트의 level 변수의 값 | cshGameManager스크립트에서 실행되는 함수 | 새의 색깔 |
| --- | --- | --- | --- |
| EASY | 0 | Easy() | 파란색 |
| NORMAL | 1 | Normal() | 빨간색 |
| HARD | 2 | Hard() | 분홍색 |

| 난이도 | 활성화 되는 스포너 | 각 스포너가 생성하는 프리팹 | 활성화 되는 새 애니메이션 |
| --- | --- | --- | --- |
| EASY | Coin Spawner | GoldCoin | Bird1 Blue |
| NORMAL | Normal Pipe Spawner | NormalPipe | Bird1 Red |
| HARD | Hard Pipe Spawner | HardPipe | Bird1 Pink |

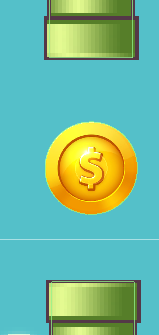
| 난이도 | 코인의 scale | 파이프의 존재 여부 | 파이프 사이의 간격 |
| --- | --- | --- | --- |
| EASY | 0.1 | 없음 | X |
| NORMAL | 0.2 | 있음 | 0.4 |
| HARD | 0.1 | 있음 | 0.2 |



Coin Spawner는 4초마다 위의 GoldCoin을 위아래로 0.55의 랜덤한 높이에서 생성한다.

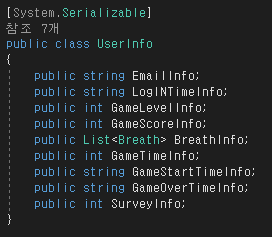


Normal Pipe Spawner는 4초마다 위의 NormalPipe를 위아래로 0.55의 랜덤한 높이에서 생성한다. 동전의 크기는 Easy보다 2배 더 크다. 파이프에 닿는다면 점수가 1점 줄어들게 된다. 점수의 최소치는 0점이다.



Hard Pipe Spawner는 4초마다 위의 HardPipe를 위아래로 0.6의 랜덤한 높이에서 생성한다. 동전의 크기는 Easy와 같다. 파이프 사이의 간격은 Normal의 절반이다. 파이프에 닿는다면 점수가 1점 줄어들게 된다. 점수의 최소치는 0점이다.

## POST를 이용해 보내는 데이터



보내는 데이터들은 cshUserInfo에 저장이 된다.

EmailInfo는 이메일 정보

LogINTimeInfo는 로그인 시간 정보

GameLevelInfo는 선택한 게임 레벨 정보

GameScoreInfo는 게임 스코어 정보

BreathInfo는 5초마다 저장한 들숨 날숨과 관련된 정보

GameTimeInfo 게임 플레이에 걸린 시간 정보

GameStartTimeInfo 게임 시작 시간 정보

GameOverTimeInfo 게임 끝난 시간 정보

SurveyInfo 게임 만족도 조사 정보

이 정보들은 게임 종료 후 만족도 조사를 끝내면 보내지게 된다.

## 만족도 조사를 위한 SurveyScene



게임을 종료하면 만족도 조사씬으로 보내지게 되고 이곳에서 만족도 조사를 실시할 수 있다.

각 버튼을 누르면 cshUserInfo에서 Survey의 변수가 매우그렇다 4점, 그렇다 3점, 보통이다 2점, 그렇지 않다 1점, 매우 그렇지 않다 0점으로 설정된다. 그 후 cshSurveyButton 스크립트의 MakeUserInfo() 함수를 이용하여 cshUserInfo에서 저장한 정보들을 POST해서 DB에 보내게 된다.

## 게임의 변수들을 조절할 수 있는 환경설정 씬

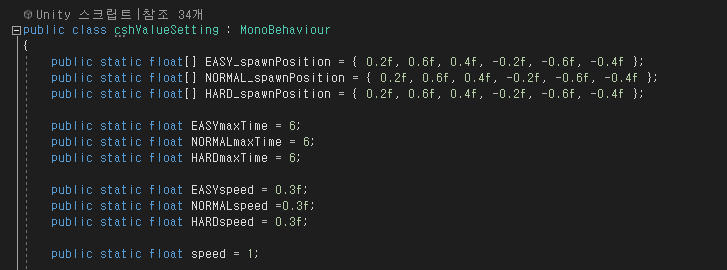


환경 설정 씬에서는 새의 속도, 각 난이도의 동전, 파이프의 생성 위치, 생성 간격 시간, 속도를 설정할 수 있다. 가장 위에 있는 입력창이 새의 속도, 각 난이도 이름 옆에 있는 두 입력창이 각각 동전과 파이프의 생성 시간과 속도 조절, 각 난이도 이름 밑에 있는 6개의 칸이 동전과 파이프의 생성 위치 순서이다. 처음에 진입하면 각 입력창에는 각 변수들의 기본값이 들어가 있다.

각 변수들의 기본값

| 새의 속도 | | | 1 | | |
| --- | --- | --- | --- | --- | --- |
| 이지모드 동전 생성 위치 | {0.2, 0.6, 0.4, -0.2, -0.6, -0.4} | 노멀모드 파이프 생성 위치 | {0.2, 0.6, 0.4, -0.2, -0.6, -0.4} | 하드모드 파이프 생성 위치 | {0.2, 0.6, 0.4, -0.2, -0.6, -0.4} |
| 이지모드 동전 생성 시간 | 6 | 노멀모드 파이프 생성 시간 | 6 | 하드모드 파이프 생성 시간 | 6 |
| 이지모드 동전 속도 | 3 | 노멀모드 파이프 속도 | 3 | 하드모드 파이프 속도 | 3 |

입력창에 실수나 정수를 입력한 후 밑에 있는 설정 버튼을 누르면 바꾼 변수값들이 적용 된다. 그 후 위에 있는 주황색 돌아가기 버튼을 누르면 게임 로비 씬으로 돌아갈 수 있다.



각 변수들은 Scripts/GameStartScene 폴더에 들어 있는 cshValueSetting이라는 스크립트에 static으로 선언되어 있다.