

TODO 본격적인 구현에 앞서 무엇이 포함되어야 하는지

1. Input - Input을 컨트롤러에서 관리
2. Scene -씬로딩에 포함사항
3. Data - Json 파일 읽기
4. Sound - BGM, 효과음(oneshot)
5. UI - 팝업, 코딩 나눠서 관리

I. Input

- 기능 게임내 모든 입력을 컨트롤러에서 처리.

제일리게이트 이용, InputManager의 Update에서
일괄실행.

```
public class InputManager
{
    public Action KeyAction = null;
    public void UpdateInput()
    {
        if (Input.anyKey == false) return;
        if (KeyAction != null)
            KeyAction.Invoke();
    }
}
```

2. Scene

I -씬로딩기초

상속받아서 사용

enum의 목적지 이름

```
public abstract class BaseScene : MonoBehaviour
{
    public Define.Scene _sceneType = Define.Scene.Unknown;
    private void Awake()
    {
        Init();
    }
    protected virtual void Init()
    {
        // 씬 초기화 및 설정
    }
    public abstract void Clear();
}
```

```
public enum Scene
{
    Unknown, // 씬
    Login, // 로그인
    Lobby, // 로비
    Game // 게임
}
```

이제일

II -씬 넘기기.

씬 레지스터를 enum 참조해서 넘기기.

```
public class SceneManagerEx
{
    // 현재 씬
    public BaseScene CurrentScene { get { return
    GameObject.FindObjectOfType<BaseScene>(); } }

    // Define.Scene에 있는 씬 로드하기
    public void LoadScene(Define.Scene type)
    {
        Managers.Clear(); // 클리어!
        SceneManager.LoadScene(GetSceneName(type));
    }

    // Define.Scene으로 이름찾기
    string GetSceneName(Define.Scene type)
    {
        string name = System.Enum.GetName(typeof(Define.Scene), type);
        return name;
    }

    // 초기화
    public void Clear()
    {
        CurrentScene.Clear();
    }
}
```

3. Data

Json 읽어와서 딕셔너리로 변환

Data Class 만들고

ILoader 상속받아

Make Dict 실행.

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

// 요거 상속받아서 만들기
public interface ILoader<Key, Value>
{
    Dictionary<Key, Value> MakeDict();
}

public class DataManager
{
    //EX)
    //public Dictionary<int, Data.Skill> SkillDict { get; private set; } = new Dictionary<int, Data.Skill>();
    public void Init()
    {
        //SkillDict = LoadJson<Data.SkillData, int, Data.Skill>("StatData").MakeDict();
    }

    Loader LoadJson<Loader, Key, Value>(string path) where Loader : ILoader<Key, Value>
    {
        TextAsset textAsset = Managers.Resource.Load<TextAsset>($"Data/{path}");
        return JsonUtility.FromJson<Loader>(textAsset.text);
    }
}
```

4. Sound (Audio)

BGM, OnShot

3요소

AudioSource - BGM 읽, 이펙트음

AudioClip - <name, clip> 딕셔너리로 정리

AudioListener - 카메라

Init - BGM 들 오디오소스 만들고 실행

play (clip, type, pitch)

clip을 type 오디오소스에, pitch로 실행

Get or Add AudioClip - 오디오클립을 들고있거나
없다면 새로 캐싱하기

Clear - 캐싱한 클립 날려주기.

TODO - 3D Sound..

play/clip At point

5. UI - 자싱라

UI - 자싱라해서 바인딩해 주고

자싱의 텍스트나든가 속성 찾아주기

자싱 붙이기, 소버라이딩

이벤트 핸들러 만들어서 드래그/클릭구분

확장인자 처리 팝업

popup, scene UI 구분

제어바. 처리 생생유지