# 综合

## Shade在哪里用

## 场景

## 角色动作

## UI贴图

## 场景加载

Application.LoadLevel("mainscene");

## 管理器的存在形式

全局对象(管理器、单件)生成的时候一般 会创建一个**空的GameObject**,然后把脚本(对象、Component)挂在空Object下，比如：

public static NetworkManager GetInstance()

{

if (!Instance)

{

Container = **new GameObject()**;

Container.name = "NetworkManager";

Instance = Container.AddComponent(typeof(NetworkManager)) as NetworkManager;

}

return Instance;

}

全局的对象需要有全局的GameObject与之关联：

public class BulletManager : MonoBehaviour {

public static BulletManager Instance;

public static GameObject Container;

对象生成时机：

在代码中调用生成，一般初始化完成后各个管理器对象已经生成并建立好相应的关联

## 第三方库

using Lidgren.Network;

库代码路径：Unity\Assets\3rdParty\Lidgren.Network

## Prefab

Object PlayerBigPrefab;

Object PlayerBulletPrefab

PlayerShot()

{

GameObject.Instantiate(PlayerBigPrefab, location, rotation \* BulletRotation);

}

void Start ()

{

PlayerBigPrefab = Resources.Load("Bullets/PlayerBigShot");

PlayerBulletPrefab = Resources.Load("Bullets/PlayerBulletPrefab");

}

## 名字空间

CrabBattleServer.CrabBehavior cb;

这里CrabBattleServer是名字空间

## 各种成员初始化方式

**方式1** 隐藏得好深

public class NetworkManager : MonoBehaviour {

public GameObject Enemy;

public CrabManager EnemyManager;

。。。

**Enemy** = GameObject.Instantiate(Resources.Load("battlecrab"),..as GameObject;

Enemy.animation.Play("laying");

**// CrabManager竟然藏在prefab:battlecrab里**

**EnemyManager** = Enemy.GetComponent<CrabManager>();

}

**方式2**

Netman = NetworkManager.GetInstance();

**方式3** 直接加载资源

**//** Object BulletSmallShot;

BulletSmallShot = Resources.Load("Bullets/EnemySmallShot");

**方式4** 克隆

projectile = GameObject.Instantiate(BulletSmallShot,。。。

## 血条在哪里

血条是Healthbar(Prefab)里的一张纹理

healthbarobject = GameObject.Instantiate(Resources.Load("Healthbar")) as GameObject;

healthbar = healthbarobject.GetComponent<GUITexture>();