

译



梦



译梦

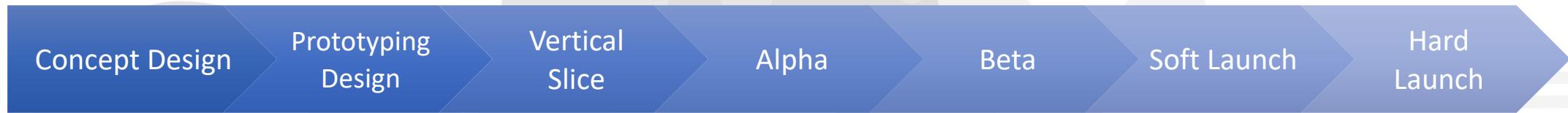


译梦机

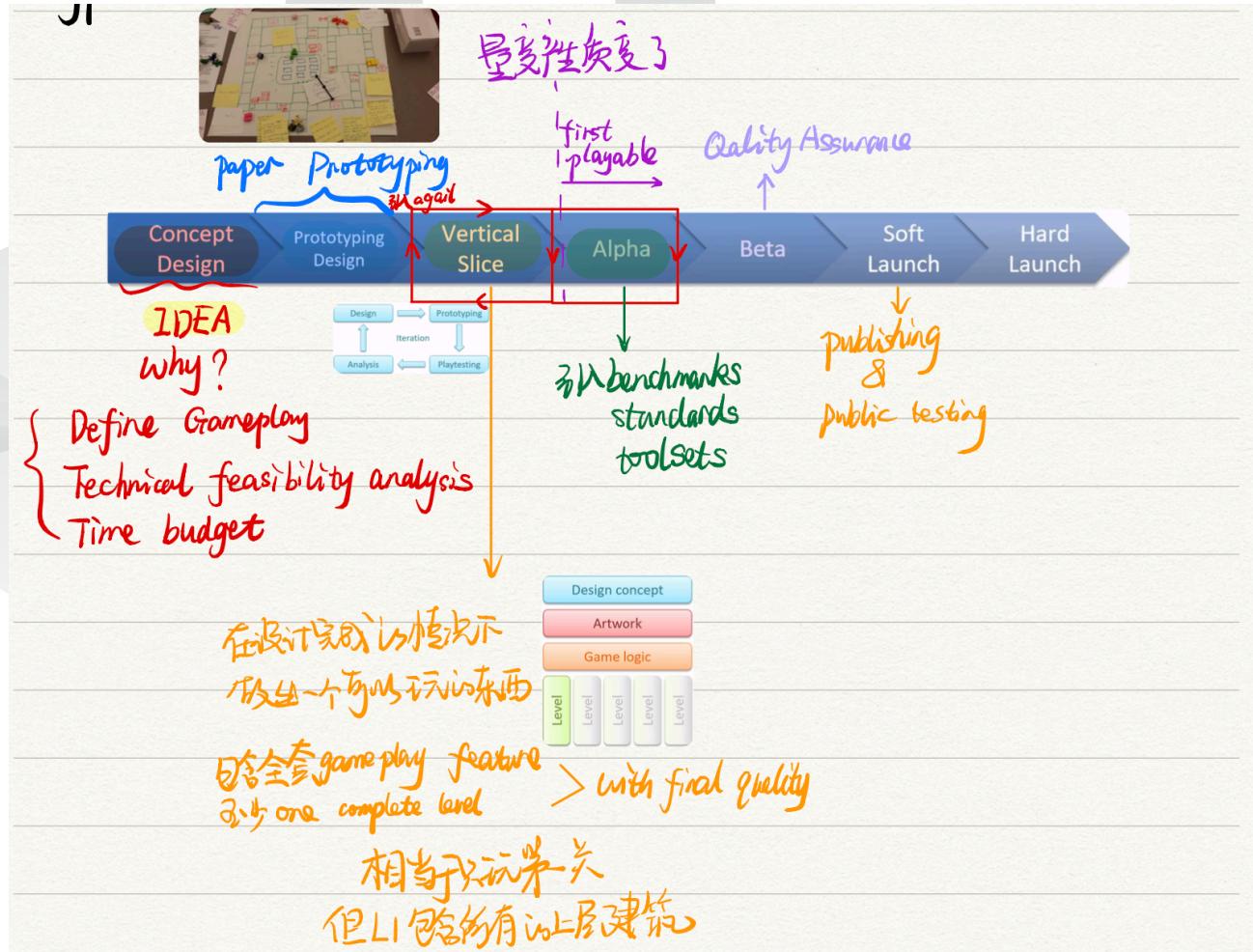
Soul Maze

第一组 张喆 陈开昕 张靖萌 李文玥

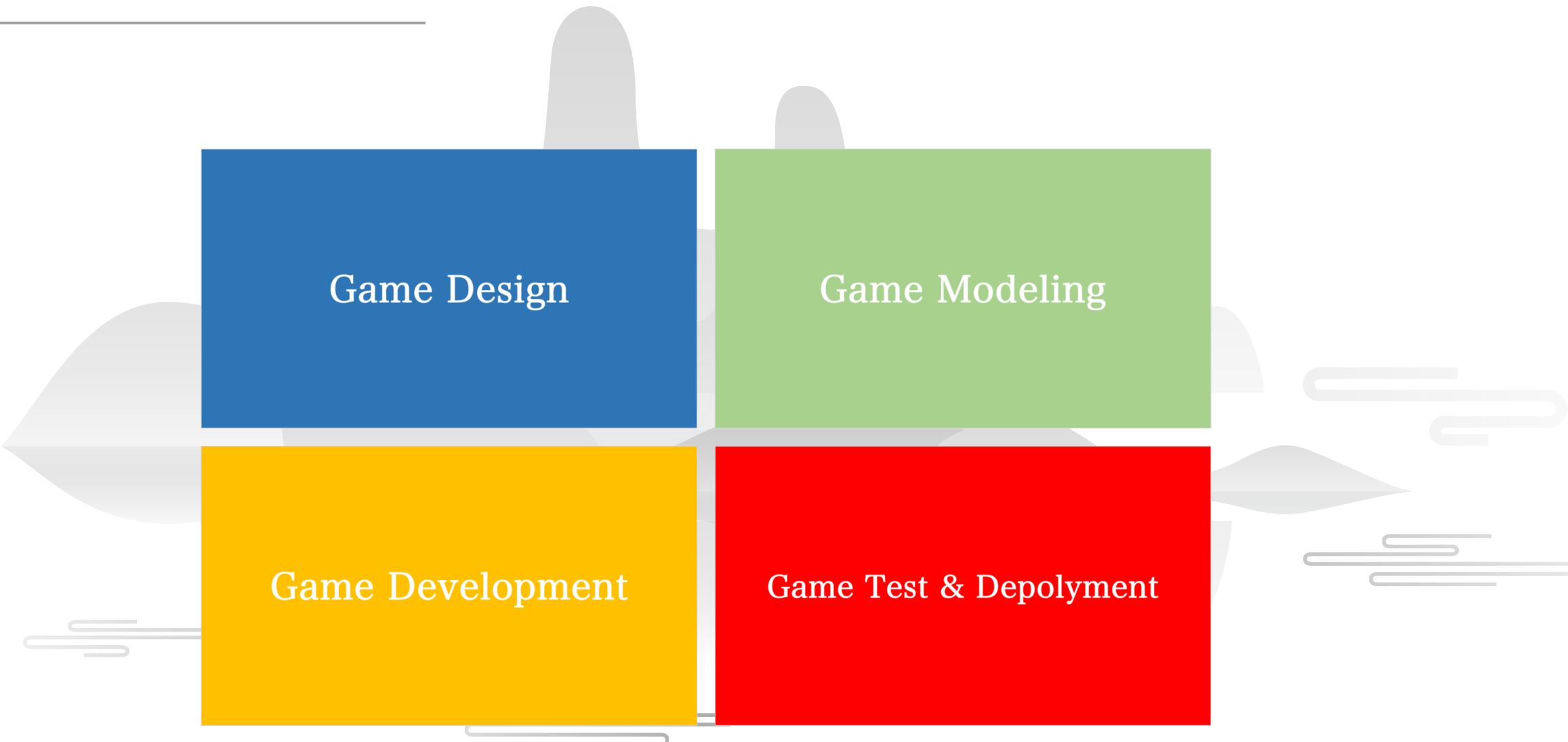
Game Development Process



Game Development Process



译梦机 | Soul-Maze



Game Design

Game Design

Background

我有一只黑狗，它名为抑郁
它总不期而至的出现我的面前
整个世界好像都在享受生活，我却只能与黑狗相伴
它蚕食掉我的记忆力和集中精神的能力
它让我烦躁不安，难于相处
它让我发现，躺倒永远比重新站起来更容易
黑狗带来的无力与羞愧
我总是担心被人知道
于是我很用力的，很用力的，
把它藏起来
藏进了我幽暗晦涩的迷宫般的心里
我感觉我，和整个世界失去了联系
...
你需要相信，你的人生也如这迷宫一般。尽管四周潜伏着黑狗，尽管行走时处处碰壁，尽管有时会迷失方向，但总会碰到生活中的“火种”，总会捡拾到抵抗困难的“碎片”，感到脆弱并不丢人，只有错过生活才是遗憾。

Game Design

Background

怪物：黑狗

火种：人快乐的小事（一句话，随机，屏幕闪现）

药片：十字标志，医生的帮助（补血）

碎片：与抑郁症的和解的建议（9个）

- 寻求专业的帮助是迈向康复的第一步。不要畏惧，无数的人都在被这条黑狗所侵扰，任何人都可能被它袭击，会有人理解你的处境与想法。
- 世界上并不存在万能灵药或魔法药丸，药物确实对部分人有帮助，其它人还需要辅以另外的手段，不要过度依赖药物。
- 向亲近的人表达出自己的真实感情能起到关键的疗效
- 不要害怕黑狗，你越是疲劳和紧张，它就叫的越凶，所以学会让自己平静下来很重要
- 临床证明，经常锻炼对于环节轻/中度抑郁的效果，不比抗抑郁药差。去走走，去跑步吧，把这条笨狗甩在后面！
- 记一份情绪日记，把想法写在纸上是种宣泄，往往有助于看清问题，也记录下来那些值得感恩的事情。
- 最重要的是记住，不管情况变得多么糟糕，只要你走向正确的方向，找适当的人交流，黑狗降临的日子一定会过去
- 黑狗也许永远是你生命的一部分，但你要相信，你们能够达成理解，通过学习知识、耐心、克制和幽默，最凶狠的黑狗也可以被制服。

Game Design

Concept Design

Prototyping Design

游戏程序与设计期末项目开题说明v0.0

游戏程序与设计期末项目开题说明v0.0
小组成员信息
游戏背景概要
游戏玩法简要说明
核心模块简要说明

小组成员信息

姓名	学号
张喆	1754060
陈开昕	1753188
李文明	1750803
张婧萌	1753498

游戏背景概要

黑暗笼罩世界，唯有光明划破沉寂，指引生命流淌的方向。该游戏定格为2.5D迷宫冒险游戏，玩家置身围墙迷宫之中，只有手里的一盏灯能带来光明，仅有的一扇逃生之门被魔法控制，唯有按照线索完成解谜才能重获自由；而在这漆黑冰冷的迷宫之中深藏狡猾可怖的（怪兽？），只有不断的探索才能有机会看到怪兽的行踪和面孔。聪明的人们啊，希望你们心中充满爱和勇气，找到通向家园的方向.....编不下去了

游戏玩法简要说明

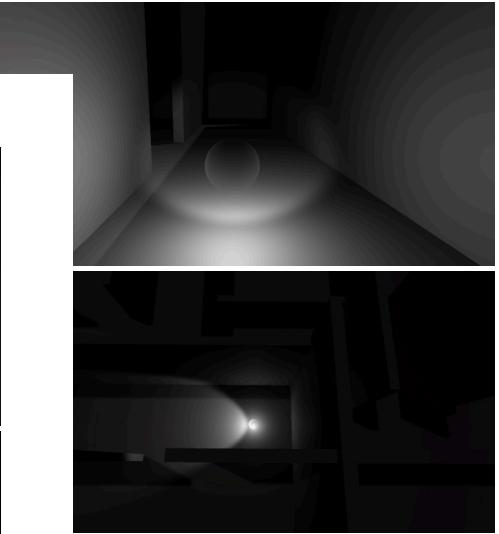
- 该游戏为2.5D（参考「弓箭手大作战」），玩家在第三人称视角下（相见摄像机）只能看到自己周围很小区域的环境（注：本游戏作为迷宫冒险类游戏不额外提供全局小地图）
 - 游戏场景为迷宫（参考电影「移动迷宫」），通过高墙等建筑作为迷宫的布局（注：视进展，后期考虑游戏中按照某种规律动态改变迷宫）
- 示意图如下

模块简要说明

摄像机（视角）



该游戏提供两种视角，简要记为第一人称视角和第三人称视角，示意图见下



进行解谜或搜集烛火可点亮地图中固定位置的灯

可能随着玩家的灯光而来，相见AI部分

于简单的AI完成
的领地，当玩家进入自己的领地之后，AI会根据玩家的位置信息进行追赶
内时，玩家位置坐标对于AI来说是公开信息，但是这里可能采取某些机制，如
才能获取一次玩家位置信息)
定算法进行不同的增益（如 火、冰等特殊效果对玩家造成影响）

下之后玩家手里的一盏灯能照亮周围情况，除此之外场景中所有地方均为黑暗

下

中有唯一通关的门，但是要根据线索搜集相应物品、完成相应任务才能获得通关钥匙
中有多个怪兽，每个怪兽有自己的领地，当玩家接近领地时，首领会对玩家进行追击，玩家
通过迷宫地形及自身智慧躲避怪兽（注：怪兽可以攻击玩家，玩家不可以攻击怪兽）
迷宫中信号不稳定，玩家只有收集到一定数量的（yyy）才能像卫星发送信号，以观察自己所
位置和周围情况的俯视图，其他时间需要通过逻辑分析躲避怪兽
UI提示玩家关键搜寻物品的大致方向；同时UI和音效部分也会根据怪兽的位置进行相应的变
以帮助玩家在第一人称视角下分辨怪兽
被怪兽攻击后会自身属性产生相应影响（包括生命值、手电筒的范围、移动速度等等），同时
上也会收到相应的影响（例如，屏幕产生污点效果等等），玩家生命值为0时宣告失败

Game Design

Concept Design

Pro

游戏程序与设计期末项目开题说明v0.0

游戏程序与设计期末项目开题说明v0.0

小组成员信息
游戏背景概要
游戏玩法简要说明
核心模块简要说明

小组成员信息

姓名	学号
张喆	1754060
陈开昕	1753188
李文明	1750803
张婧萌	1753498

游戏背景概要

黑暗笼罩世界，唯有光明打破沉寂，指引生命流淌的方向。该游戏定标为2.5D迷宫冒险游戏，玩家置身围墙迷宫之中，只有手里的一盏灯能带来光明，仅有的一扇逃生之门被魔法控制，唯有按照线索完成解谜才能重获自由；而在这漆黑冰冷的迷宫中深藏狡猾可怕的「怪兽」，只有不断的探索才能有机看到怪兽的行踪和面孔。聪明的人们啊，希望你们心中充满爱和勇气，找到通向家园的方向.....编不下去了

游戏玩法简要说明

- 该游戏为2.5D（参考「弓箭手大作战」），玩家在第三人称视角下（相视摄像机）只能看到自己周围很小区域的地图（注：本游戏作为迷宫冒险类游戏不额外提供全局小地图）
- 游戏场景为迷宫（参考电影「移动迷宫」），通过高墙等建筑作为迷宫的布局（注：视进展，后期考虑游戏中按照某种规律动态改变迷宫）

示意图如下



Game Program and Design Final Project Brief Introduction v0.0

Game Program and Design Final Project Brief Introduction v0.0

Group member information
Game background summary
Brief description of Gameplay
Brief description of core modules

Group member information

Name	Student ID
Zhe Zhang	1754060
Kaixin Chen	1753188
Wenyue Li	1750803
Jingmeng Zhang	1753498

Game background summary

Darkness envelopes the world, only light can pierce the silence and guide the direction of life flowing. The game is calibrated as a 2.5D labyrinth adventure game. The player is placed in the maze of the wall, a lamp in his hand can bring light. The only door to escape is controlled by magic, only by completing the clue to solve the puzzle can the player regain his freedom. In this dark and cold labyrinth, there are hidden sly and terrible monsters. Be alert and listen! Smart people, always keep in mind: filling your heart with love and courage. Only in this way can you find the way to the homeland.

Brief description of Gameplay

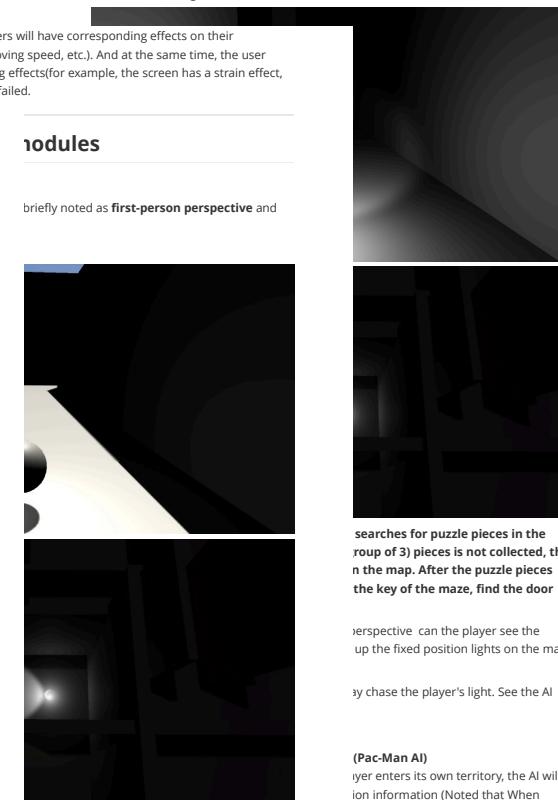
- The game is 2.5D (refer to "Arrow.io"). In the third-person perspective, the player can only see a map of a small area around him. (Note: This game doesn't provide the global mini map as a labyrinth adventure game.)
- The game scene is a maze (refer to the movie "Maze Runner"). Buildings such as high walls are used as the layout of the maze. **The maze design considers the establishment of a maze library. The maze map in a scene is formed by combining the elements in the maze library.**

The schematic diagram is as follows.



- In the initial situation, a lamp in the player's hand can illuminate the surrounding situation. In the third perspective, the circle with the player as the center and a radius of 1.5 maze path width is set to low brightness. All the other places are dark.

The schematic diagram is as follows.



; you must collect the corresponding to the clues.

' has its own territory. When the layer. The player needs to avoid the wisdom. (Note: Monsters can attack

only send a signal to the satellite after op view of his location and avoid the monster with logical

at the player about the general sound effects will be changed yer distinguish the monster in the

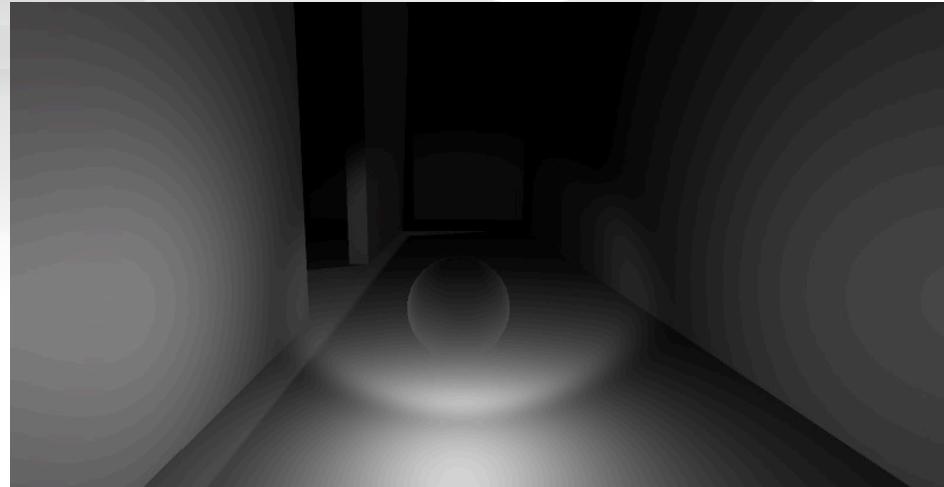
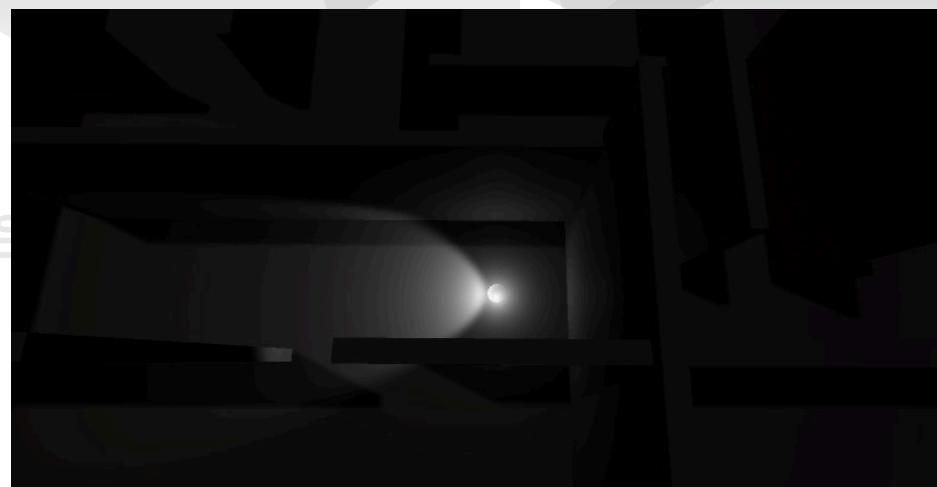
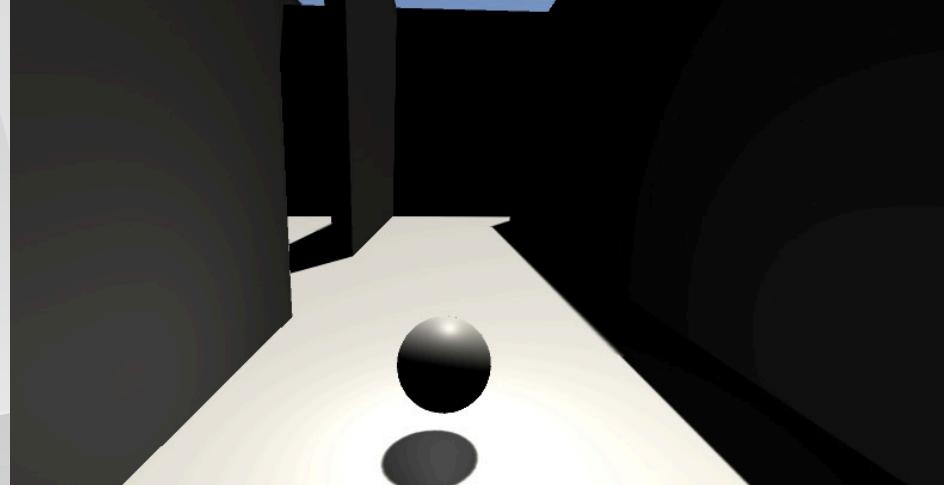
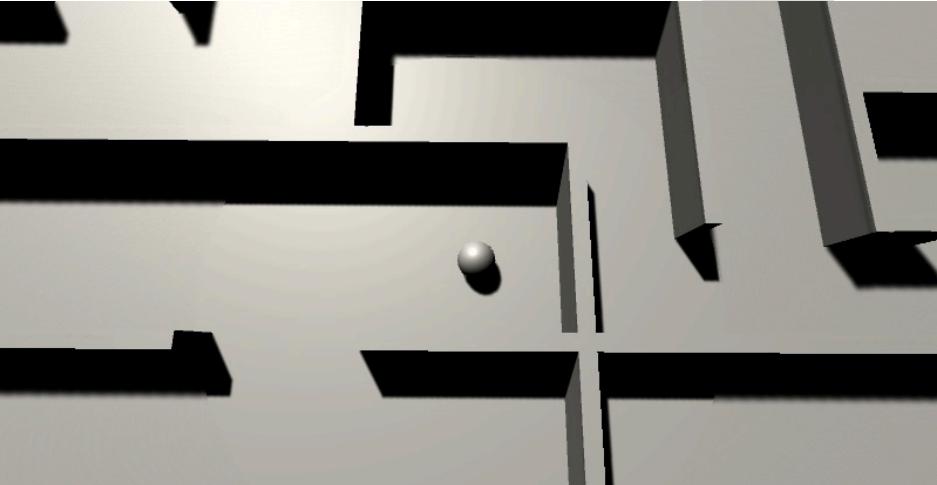
(Pac-Man AI)
player enters its own territory, the AI will ion information in a small local area. In this lens to contact with the satellite

perspective. When the player get a certain the satellite to observe the surrounding maze stellite communication time is over, the perspective.

Game Design

Concept Design

Prototyping Design



Game Modeling

Game Modeling

AI

1. 巡逻小怪
 - 随机出生
 - 随机溜达
 - 当 距离<x 通知场景, 切bgm
 - 速度始终小于人 (梯度变化)
 - 当距离> x, 再次随机巡逻, 切bgm
2. 中间boss
 - 只在中间4*4 / 5*5范围移动
 - 平时待在正中间一动不动
 - 人进入领地 (切bgm2)
 - 速度一直大于人
 - 不能超出自己的领地
 - 当人成功逃离领地的时候, 再次走回到正中间 (切bgm2)
3. ai不能吃道具, 穿过道具

人

- 属性
 - 生命值
 - 生命上限值
 - san值
 - 碎片
 - 文字 (一段话)
 - prefab(kind)
 - 速度multiplier
- 与道具
 - 吃❤ - 回生命 5点

```
    ❤碰撞
    血量 + 1
    if 血量 > 血量上限
        血量 = 上限
```

更新UI

- 吃🔥 - 回san +10
- 吃🍀 - 左上角的UI显示这个碎片
- 受伤判定
 - 小怪
 - 当碰撞的时候, 小怪scale放大3倍, 慢慢的fade, 然后destroy
 - 再随机生成一个新的
 - 生命值 -1
 - boss
 - 当距离小于y时, 视为受伤
 - 生命值 -2
 - 蒙版掉血 (动画)
 - 冷却10s (呆在原地不动, 取消所有判定)
 - 当血量<=3时
 - UI加上干扰视线内容
 - 速度 * 3/4
- 脚步声
- san判定
 - 一直下降 1/s
 - 当san为0的时候
 - 血量上限减少到3
 - 保持san为0

地图

- 墙顶端设置不同材质(或者考虑两个cube构建一面墙)
<https://blog.csdn.net/nanggong/article/details/54969867>
https://blog.csdn.net/n_moling/article/details/88823634
- 小地图 - 添加一个天空镜头
 - 人
 - AI
 - 碎片
 - 墙

<https://blog.csdn.net/l773575310/article/details/73100522>

游戏判定

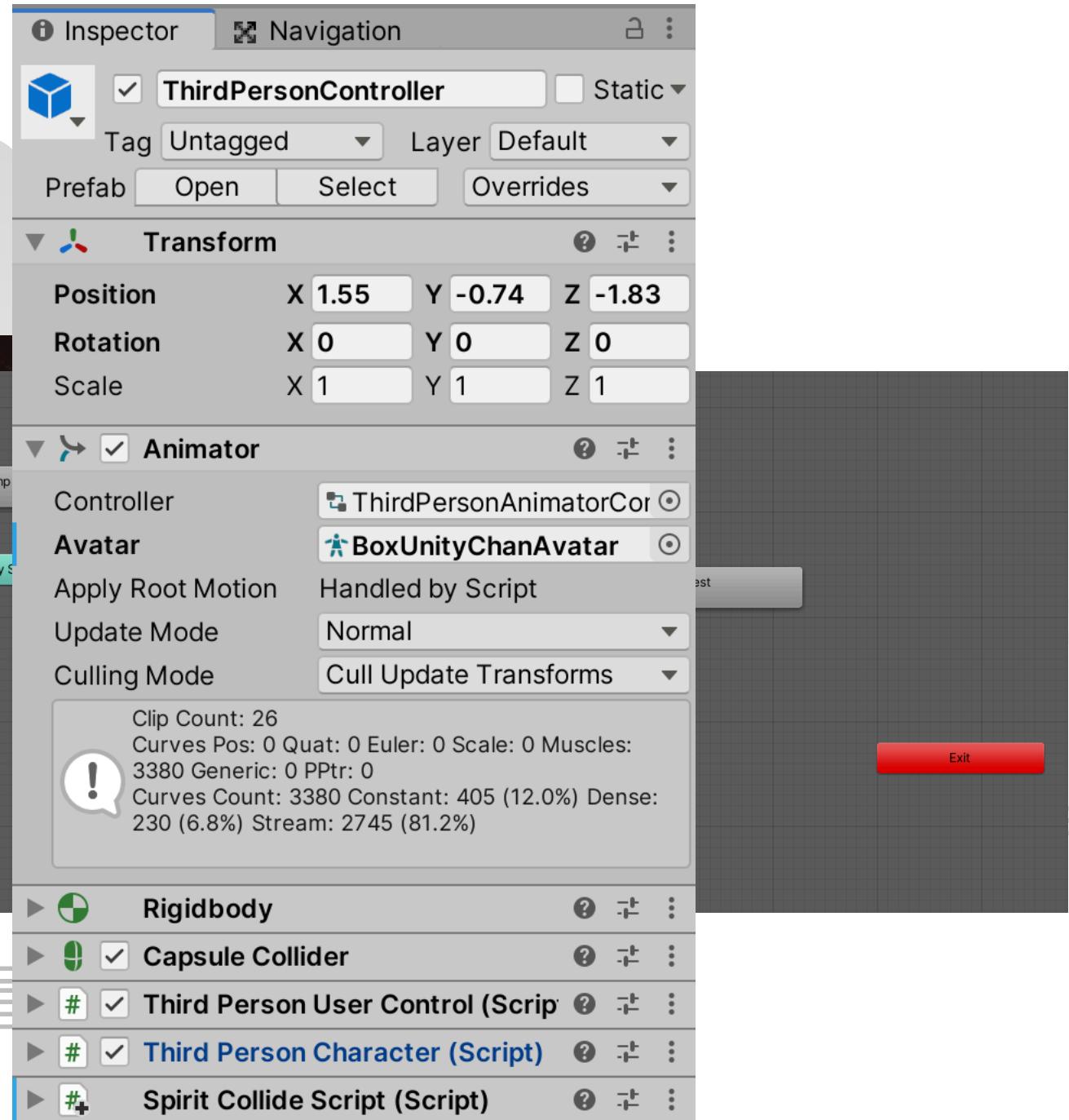
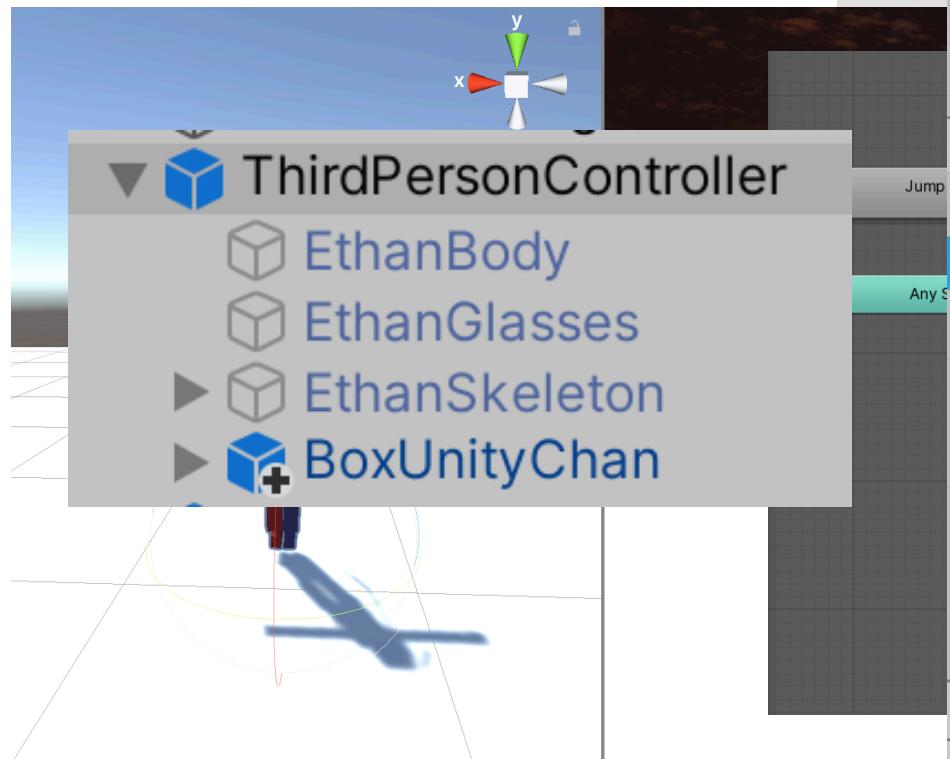
- 开始

- 出生地在场景中固定
- boss在场景中固定
- 血量为5
- san为100
- 随机产生巡逻小怪
- 暂停 (玩家自己点了暂停按钮)
 - 人不动
 - AI所有逻辑停掉
 - 寻路
 - 冷却
 - 移动
 - . . .
 - san的定时器取消
- 继续游戏
- 结束
 - 人的血量为0, 蒙版动画, 失败
 - 集齐9个碎片, 蒙版动画, 成功
 - 跳到主界面

Game Development

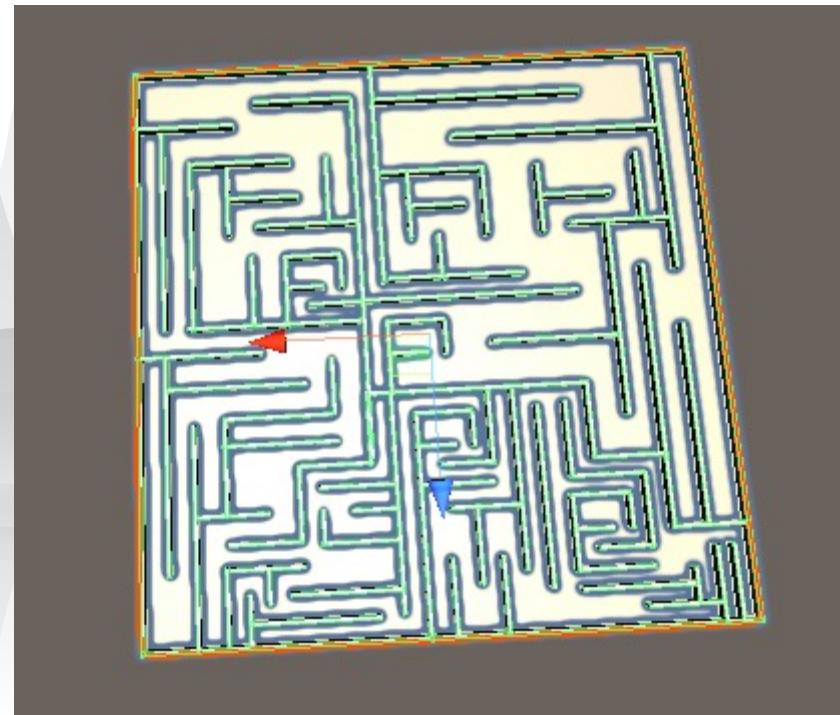
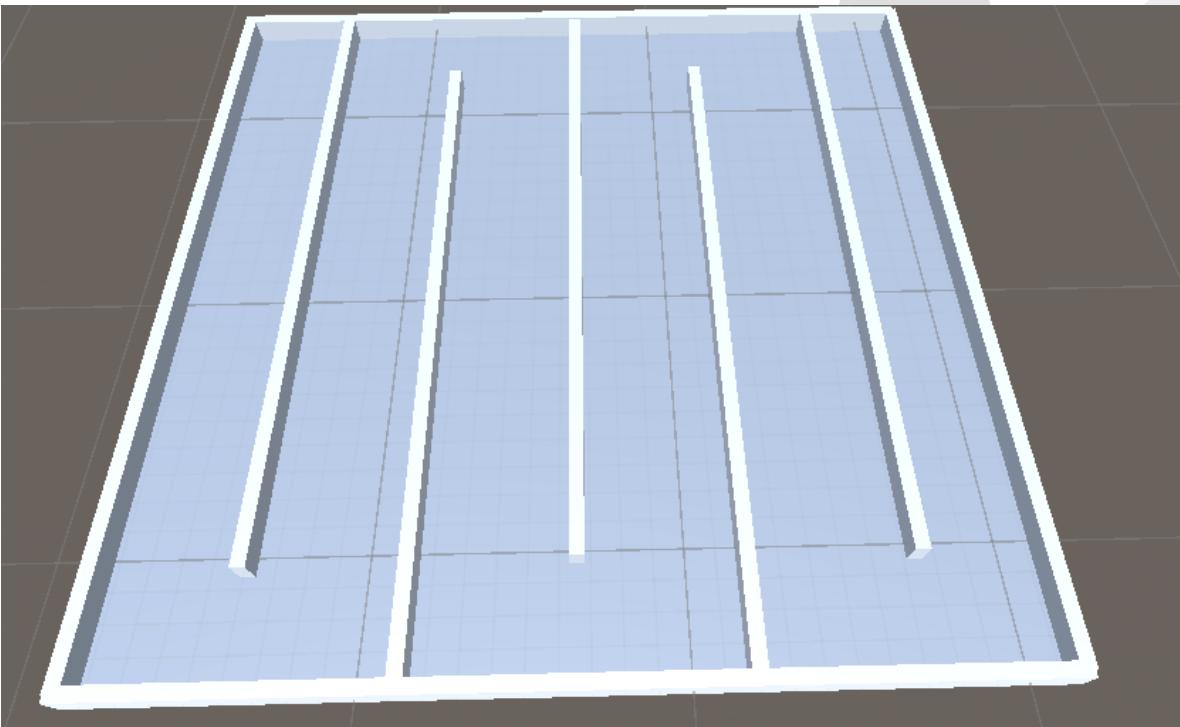
Game Development

Spirit & Hero



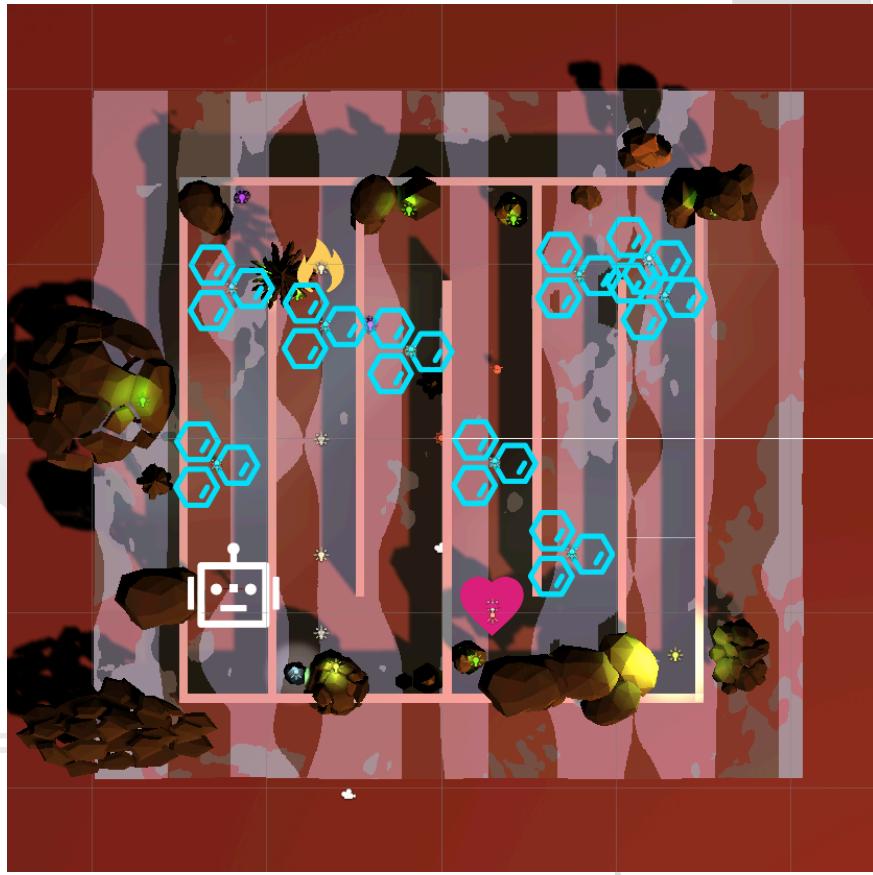
Game Development

Map



Game Development

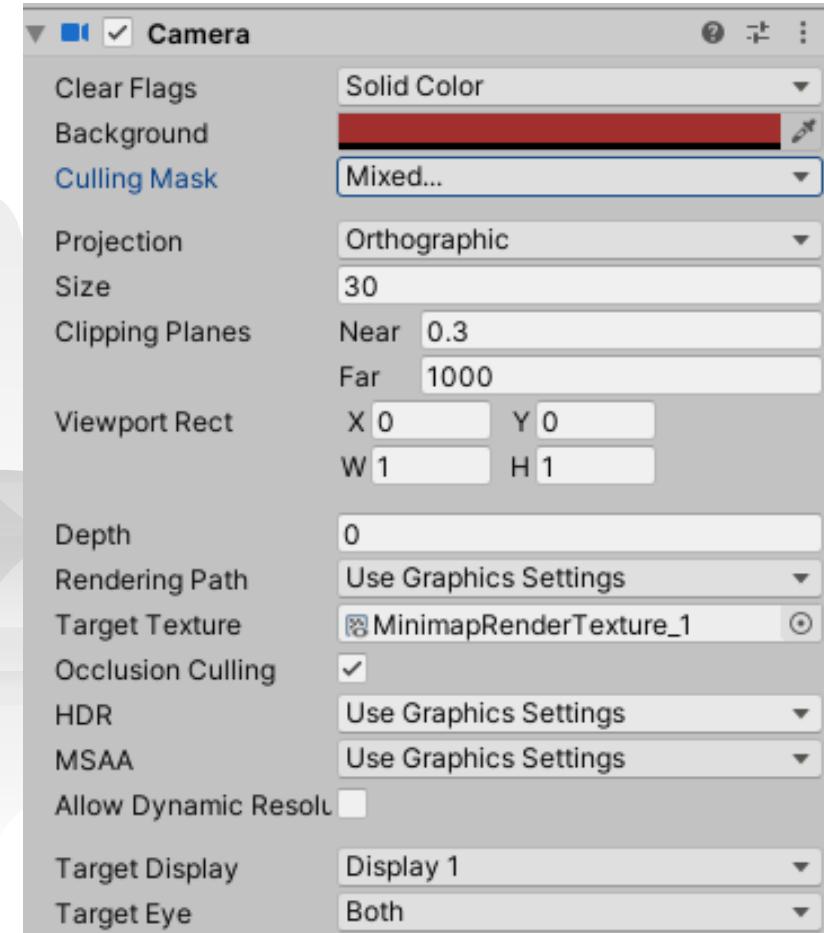
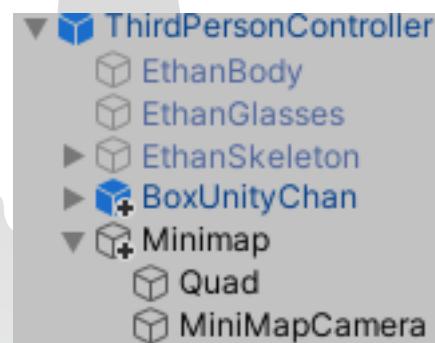
Map



Game Development

Camera

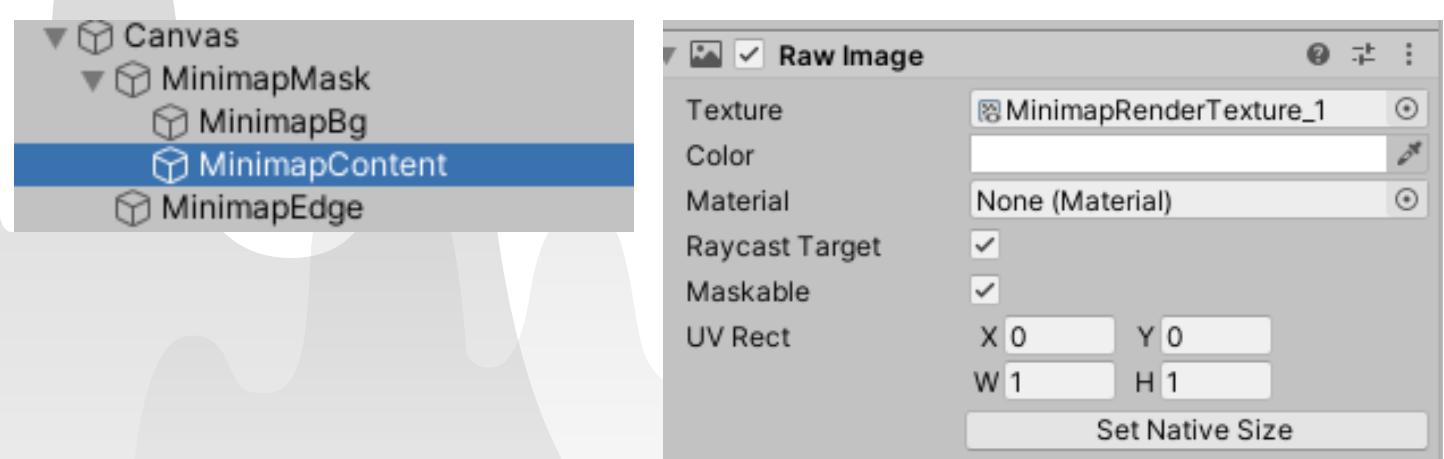
1. 添加小地图俯瞰视角相机
2. 俯瞰相机和UI Raw Image映射
3. 调整场景内物体的Layer



Game Development

Camera

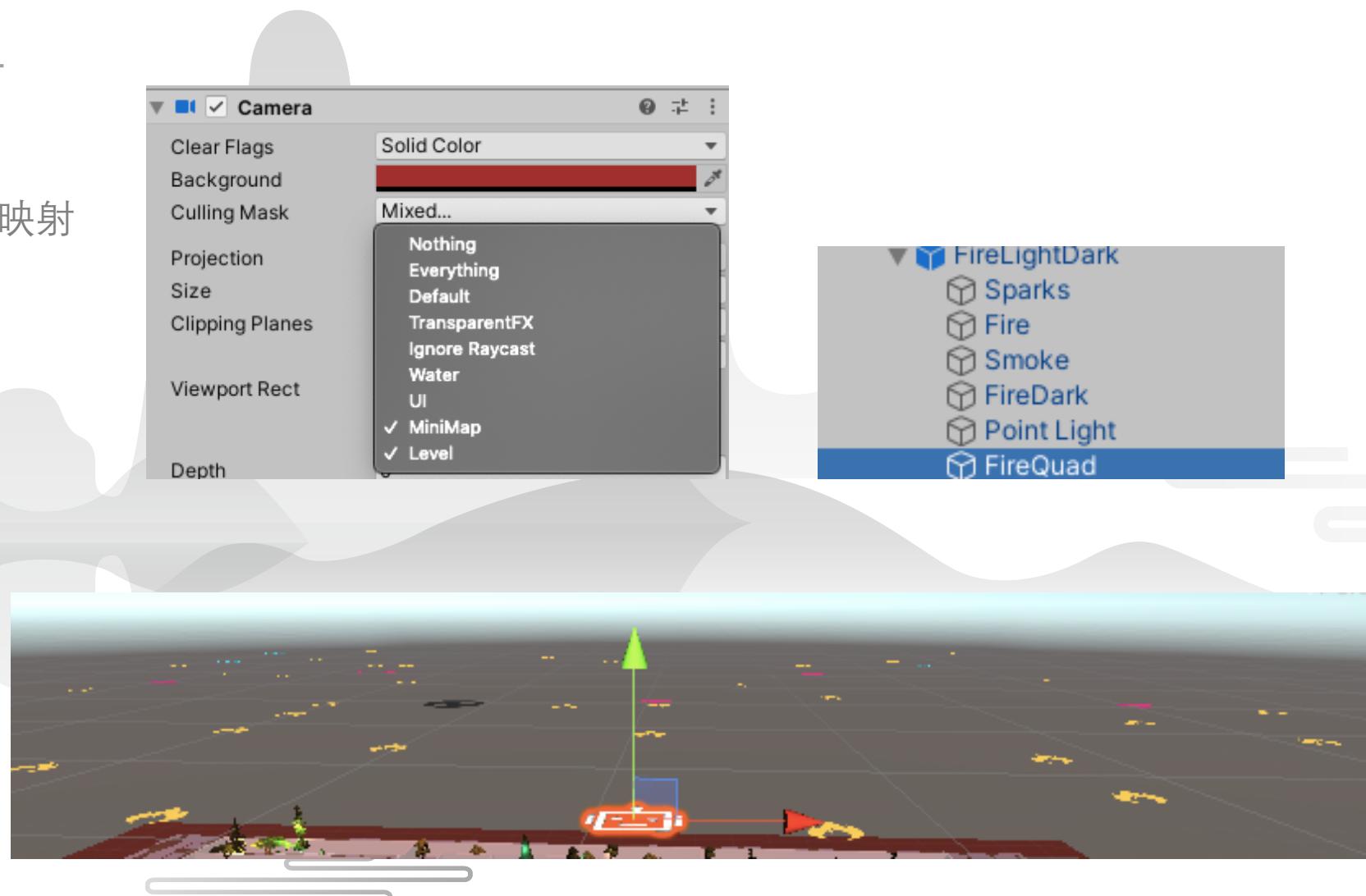
1. 添加小地图俯瞰视角相机
2. 俯瞰相机和UI Raw Image映射
3. 调整场景内物体的Layer



Game Development

Camera

1. 添加小地图俯瞰视角相机
2. 俯瞰相机和UI Raw Image映射
3. 调整场景内物体的Layer



Game Development

AI



Game Development

AI – Path Finding



FloodFill

```
void Grid::_compiteFloodFill(const Grid& collision, int x, int y, int value)
{
    static const int DirX[] = {-1, 0, 1, 0};
    static const int DirY[] = {0, -1, 0, -1};

    int nextValue = value + 1;
    // Flood
    for(int i=0;i<4;i++)
    {
        int nextX = x + DirX[i];
        int nextY = y + DirY[i];

        if(nextX >=0 && nextX < WIDTH &&
           nextY >=0 && nextY < HEIGHT &&
           collision.getValue(nextX, nextY) != 1 && //墙
           (getValue(nextX, nextY) == -1 || getValue(nextX, nextY) > nextValue))
        {
            setValue(nextX, nextY, nextValue);
        }
    }

    for(int i=0;i<4;i++)
    {
        int nextX = x + DirX[i];
        int nextY = y + DirY[i];
        if(getValue(nextX, nextY) == nextValue)
        {
            _compiteFloodFill(collision, nextX, nextY, nextValue);
        }
    }
}
```

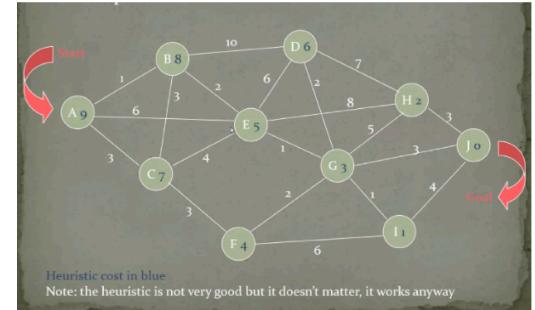
Game Development

AI – Path Finding

FloodFill A* Algo

A*

- **Basic idea:** Complete exploration of all the possible paths, but by looking in priority to the more promising paths
- 最坏情况下要遍历所有路径，而且还要维护很多其他东西
- 我已经走了多少路 + 预测到下一个点需要花的路程
- 构架
 - valued graph
 - Heuristic = $\sqrt{(\text{DestX} - X)^2 + (\text{DestY} - Y)^2}$ 启发值 (估算下一个点到目标的距离)
 - start and end node
- **数据结构**
 - OPEN node list: 将要探索的点 (白)
 - CLOSED node list: 已经探索过的点 (蓝)



蓝色相当于从某个点直线跑到目标的距离

白色是两个能直接走的距离

目标: 从A -> J

$$1. \text{ A} \rightarrow \text{B} \rightarrow \text{J} = 1 + 8 = 9$$

$$\text{A} \rightarrow \text{C} \rightarrow \text{J} = 3 + 7 = 10$$

$$\text{A} \rightarrow \text{E} \rightarrow \text{J} = 6 + 5 = 11$$

第一步去B

$$2. \text{ A} \rightarrow \text{B} \rightarrow \text{A} \rightarrow \text{J}$$

$$\text{A} \rightarrow \text{B} \rightarrow \text{C} \rightarrow \text{J}$$

A->B->E->J 最小

3.

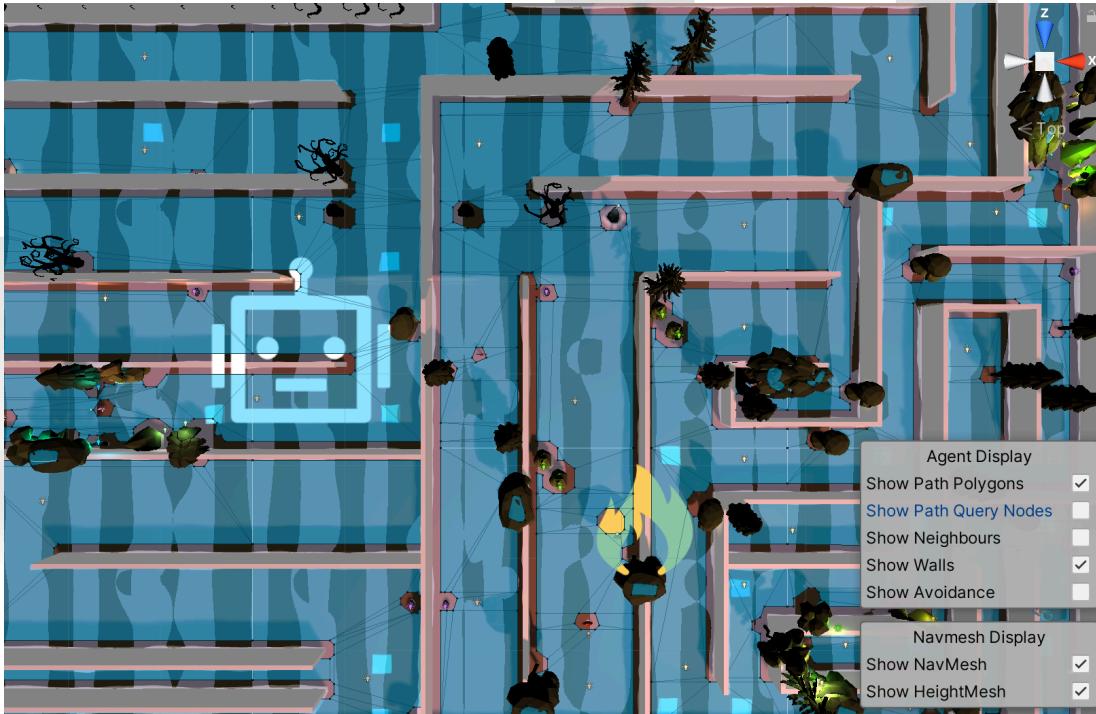
Game Development

AI – Path Finding

FloodFill

A*

Navmesh



▼ **Nav Mesh Agent** ?

Agent Type	Humanoid
Base Offset	0

Steering

Speed	50
Angular Speed	120
Acceleration	100
Stopping Distance	0
Auto Braking	<input checked="" type="checkbox"/>

Obstacle Avoidance

Radius	0.5
Height	2
Quality	High Quality
Priority	50

Path Finding

Auto Traverse Off Me	<input checked="" type="checkbox"/>
Auto Repath	<input checked="" type="checkbox"/>
Area Mask	Mixed...

AI Character Control (Script) (Ren)

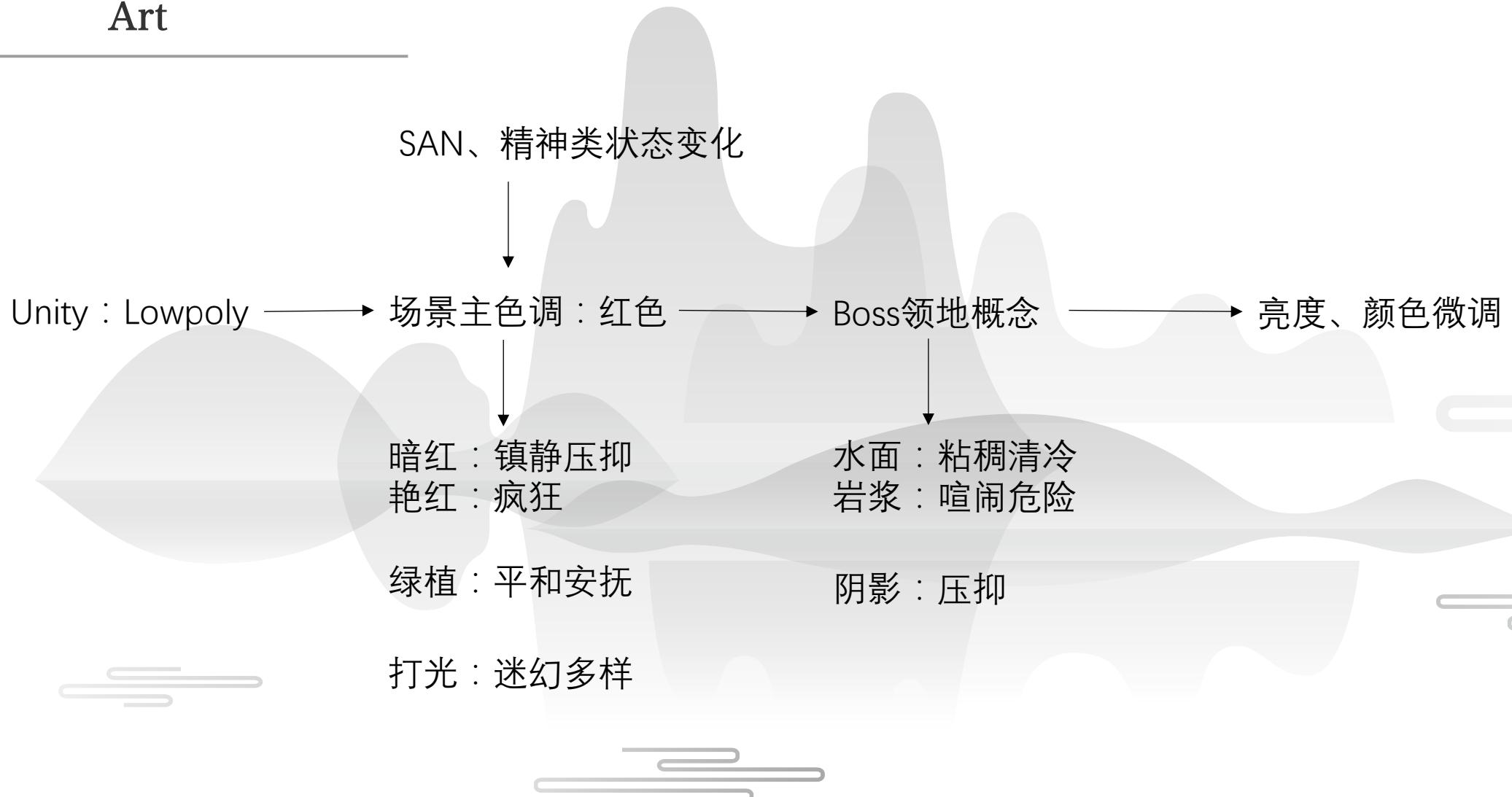
► # **Third Person Character (Script)** ?

▼ # **AI Boss Character Control (Script)** ?

Script	# AIBossCharacterContr
Target	ThirdPersonController

Game Development

Art



Game Development

Algorithm – Backface Culling



移除没有面对这镜头的三角形，只考虑三角形与摄像机的相对位置而不依赖与摄像机朝向。依靠三角形 顶点顺序直接判断法向量方向(左手法则)。只需要将物体的normal与相机的朝向进行一个计算，即可以得到是否为backface，也就是是否需要裁剪

在unity中默认就是背面剔除，例如一个平面，添加了标准材质后他只有正面是可以看到的，背面是观察 不到的;对于立方体来说，立方体的每个面都是有正面和背面的，而背面是无法观察到的。

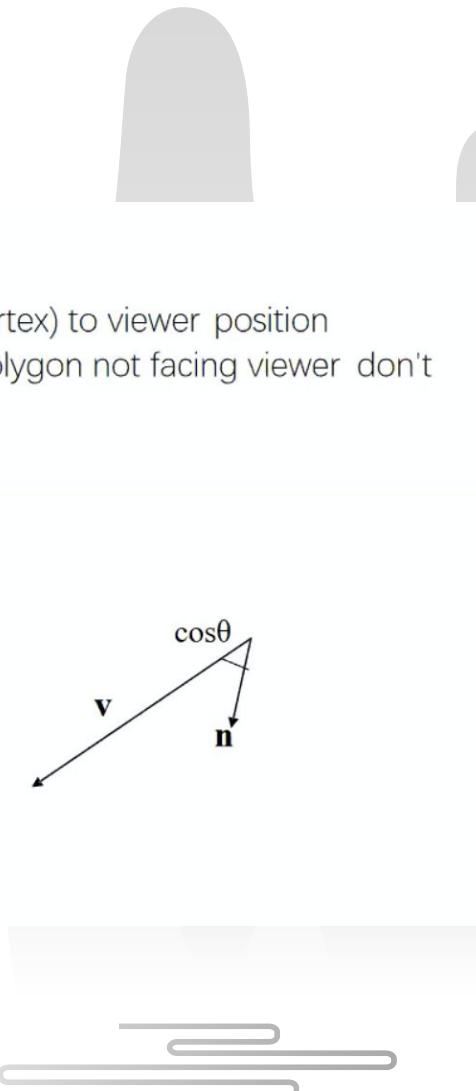
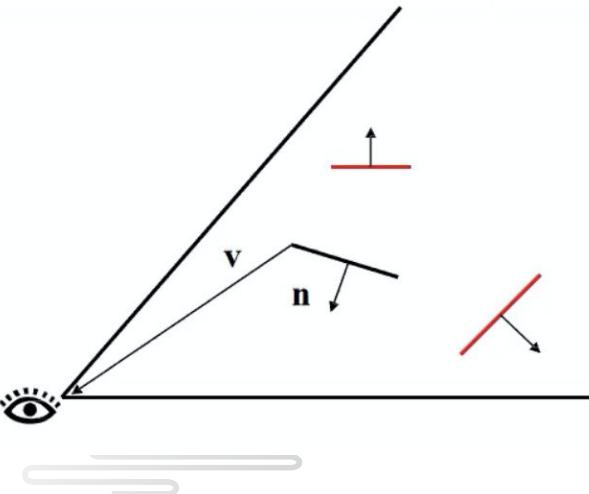
Culling操作是位于渲染管线中的光栅化阶段，在顶点着色器之后，片元着色器之前，这样是为了提高渲染性能，避免做一些无用片元的渲染计算。例如一个立方体，在unity中最多只能看到三个面。开启了背 面裁剪后其他面在背对摄像机时它对于摄像机来说就是一个背面，导致其他三个面被裁减掉。

对于一个物体而言，背面剔除会剔除掉几乎所有不需要渲染的面片，而不需要用到深度测试;深度测试 多用于物体之间，保证各个物体间正确的排列顺序，而背面剔除用于剔除自身背面。

Game Development

Algorithm – Backface Culling

- For each polygon
 - n – is surface normal
 - v – vector from point at polygon (vertex) to viewer position
 - If $\text{dot}(n,v) < 0$, then angle is > 90 , polygon not facing viewer don't render it.
 - (remember $n \cdot v = \cos \theta$)



```
Shader "Custom/NewSurfaceShader"
{
    Properties
    {
        _Color ("Color", Color) = (34,45,1,1)
        _MainTex ("Albedo (RGB)", 2D) = "white" {}
        _Glossiness ("Smoothness", Range(0,1)) = 0.5
        _Metallic ("Metallic", Range(0,1)) = 0.0
    }
    SubShader
    {
        Cull Front
        Tags { "RenderType"="Opaque" }
        LOD 200

        CGPROGRAM
        // Physically based Standard lighting model, and enable shad
#pragma surface surf Standard fullforwardshadows

        // Use shader model 3.0 target, to get nicer looking lightin
#pragma target 3.0

        sampler2D _MainTex;

        struct Input
        {
            float2 uv_MainTex;
        };

        half _Glossiness;
        half _Metallic;
        fixed4 _Color;

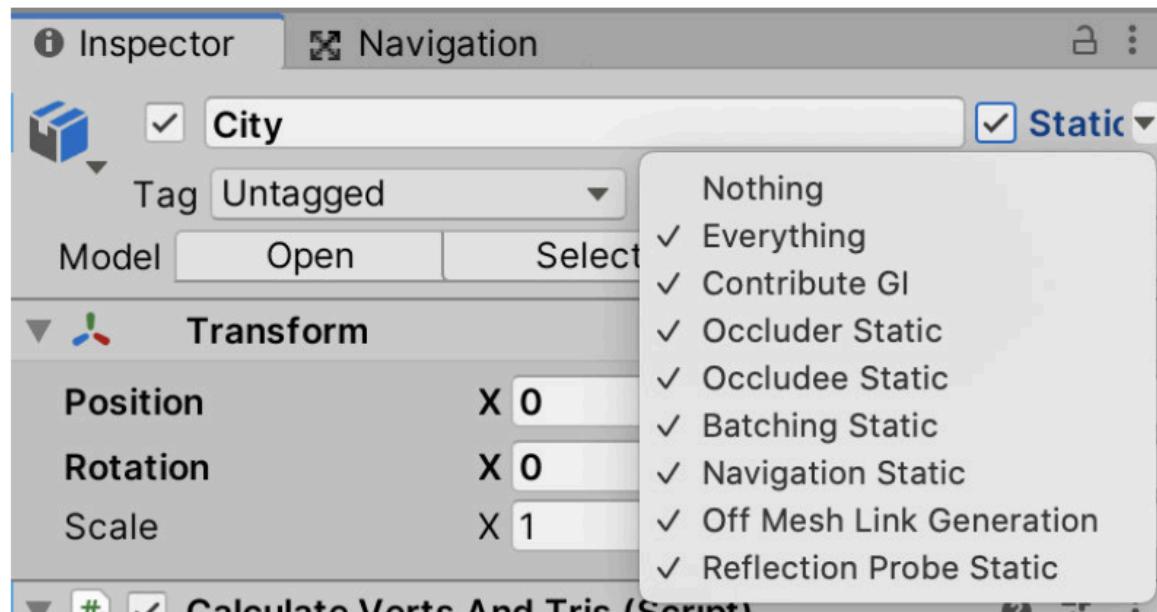
        // Add instancing support for this shader. You need to check
        // See https://docs.unity3d.com/Manual/GPUInstancing.html fc
        // #pragma instancing_options assumeuniformscaling
UNITY_INSTANCING_BUFFER_START(Props)
        // put more per-instance properties here
UNITY_INSTANCING_BUFFER_END(Props)

        void surf (Input IN, inout SurfaceOutputStandard o)
        {
            // Albedo comes from a texture tinted by color
            fixed4 c = tex2D (_MainTex, IN.uv_MainTex) * _Color;
            o.Albedo = c.rgb;
            // Metallic and smoothness come from slider variables
            o.Metallic = _Metallic;
            o.Smoothness = _Glossiness;
            o.Alpha = c.a;
        }
    }
}
```

Game Development

Algorithm – Occlusion Culling

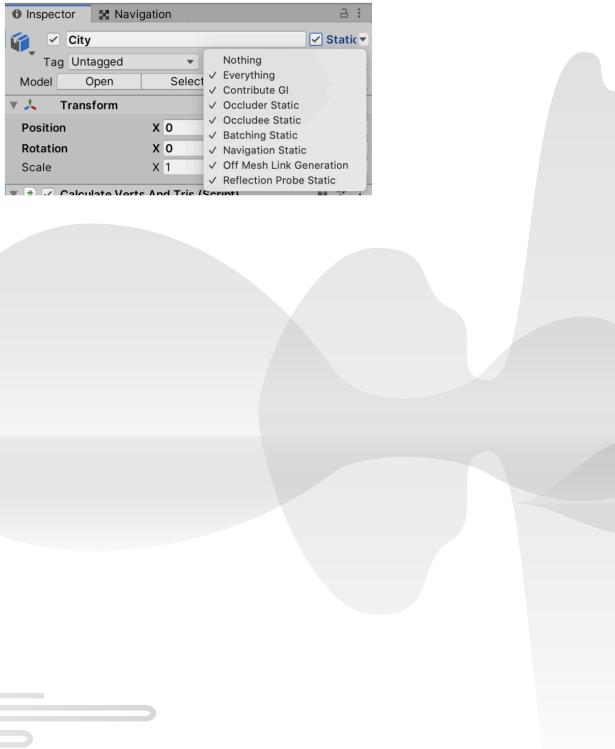
1. 设置遮挡静态 Occluder Static / Occludee Static，并且将子物体同样设置为遮挡静态



Game Development

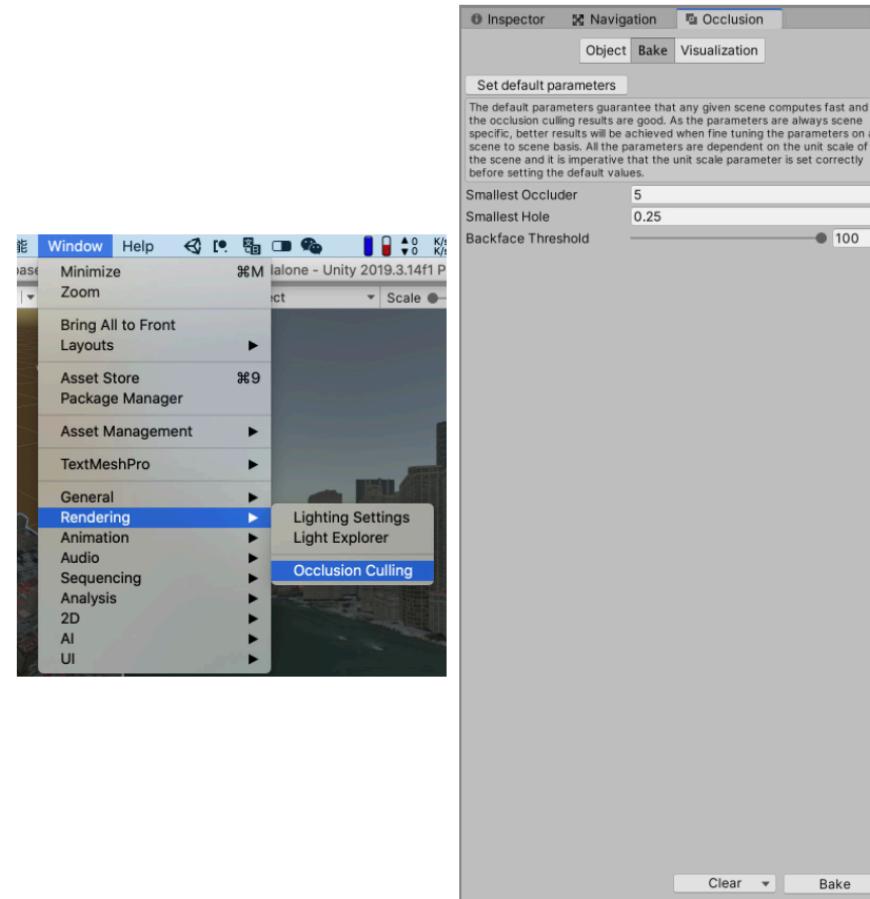
Algorithm – Occlusion Culling

1. 设置遮挡静态 Occluder Static / Occludee Static，并且将子物体同样设置为遮挡静态



2. 烘焙遮挡提出 Window -> Rendering -> Occlusion Culling

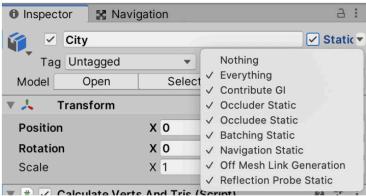
- Smallest Occluder: 最小遮挡物的尺寸，当遮挡物的长度或宽度大于设定值时，该物体才能够遮挡住后面的物体
- Smallest Hole: 最小孔的尺寸，当穿过物体内部的孔或者多个物体堆叠形成的孔大于设定的值时，遮挡剔除烘焙将忽略孔的存在
- Backface Threshold: 设置背面已处于之，用于优化场景



Game Development

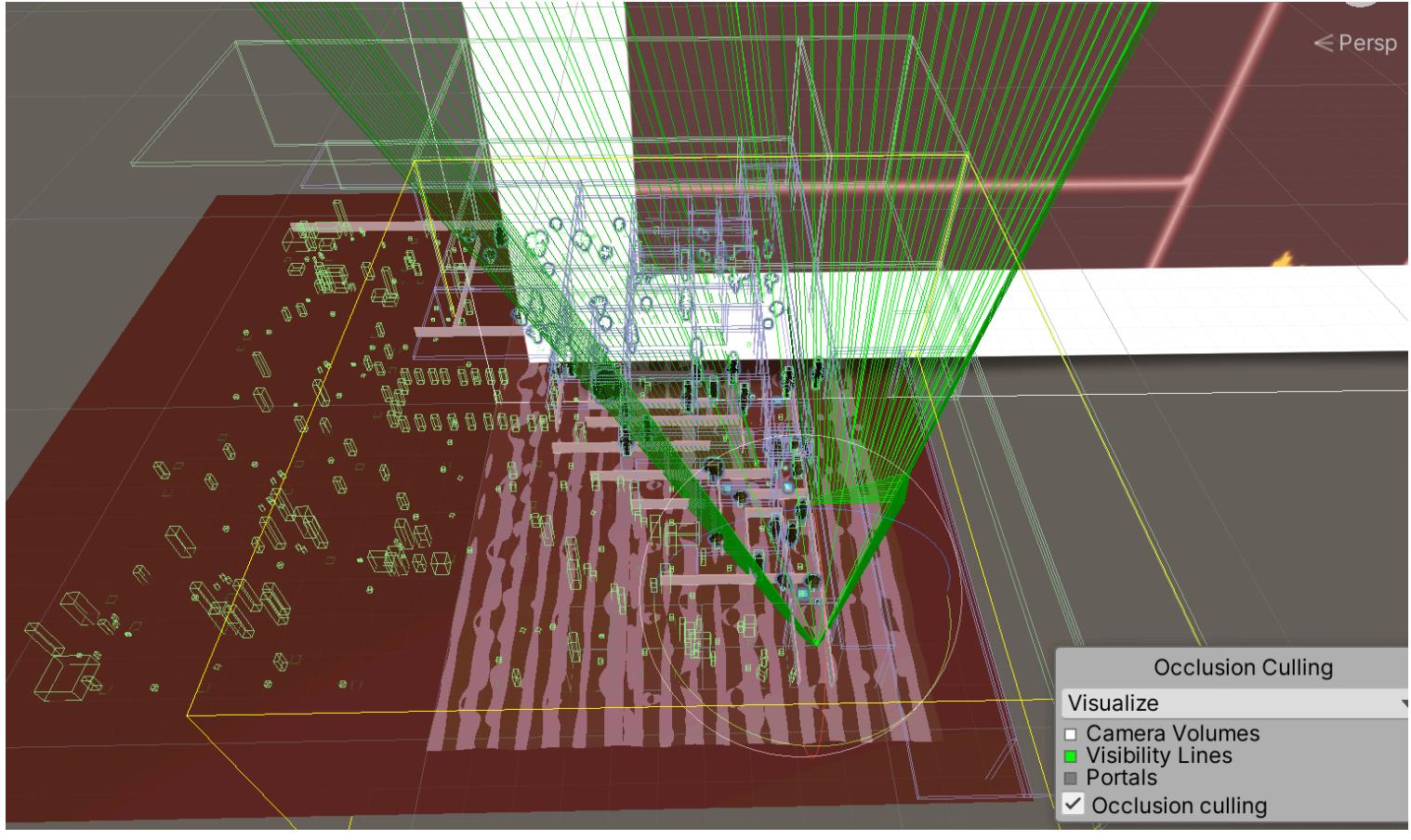
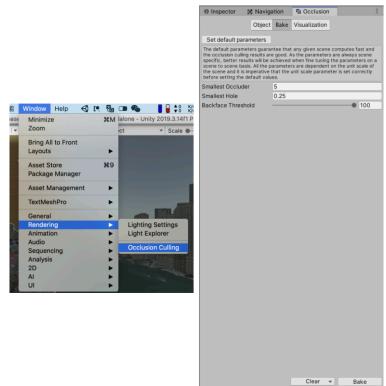
Algorithm – Occlusion Culling

1. 设置遮挡静态 Occluder Static / Occludee Static, 并且将子物体同样设置为遮挡静态



2. 烘焙遮挡提出 Window -> Rendering -> Occlusion Culling

- Smallest Occluder: 最小遮挡物的尺寸, 当遮挡物的长度或宽度大于设定值时, 该物体才能够遮挡住后面的物体
- Smallest Hole: 最小孔的尺寸, 当穿过物体内部的孔或者多个物体堆叠形成的孔大于设定的值时, 遮挡剔除烘焙将忽略孔的存在
- Backface Threshold: 设置背面已处于之, 用于优化场景



Game Development

Algorithm – State Machine

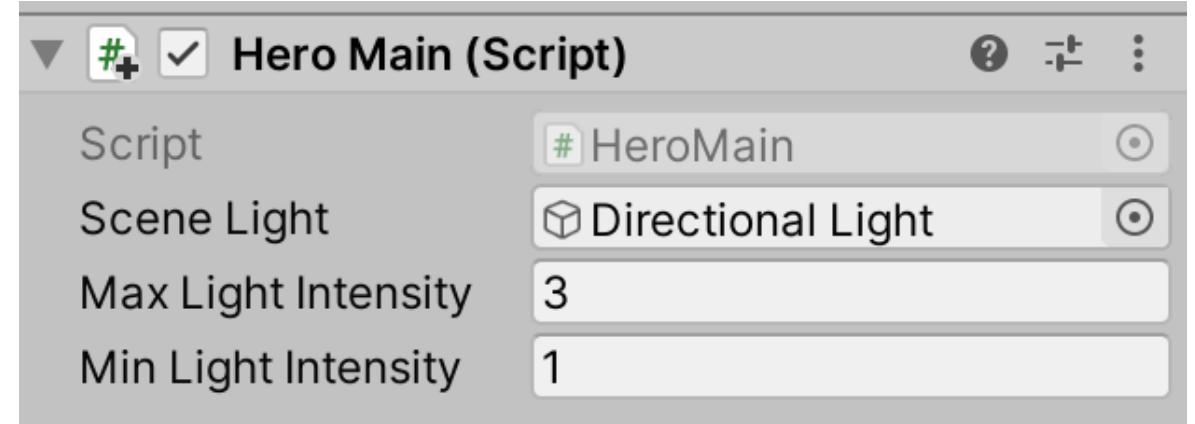
```
/**  
 * 玩家没进入领域时 (未发现 && 位置出界)  
 *      站在原地不动  
 *      状态为「未发现」玩家  
 * 当玩家进入领域时 (位置进入)  
 *      状态为「发现」玩家  
 *      开始追玩家  
 * 当玩家离开领域时 (发现 && 位置出界)  
 *      状态为「未发现」玩家  
 *      回到原点  
 **/
```

```
if (!isEatHero)  
{  
    if (isHeroInTerritory())  
    {  
        findHero = true;  
        agent.destination = target.transform.position;  
        character.Move(agent.desiredVelocity, false, false);  
    }  
    else  
    {  
        findHero = false;  
        agent.destination = startPos;  
        character.Move(agent.desiredVelocity, false, false);  
    }  
    else  
    {  
        //5s 之后  
        coolingCountDown -= Time.deltaTime;  
        if (coolingCountDown < 0)  
        {  
            isEatHero = false;  
            findHero = false;  
            agent.destination = startPos;  
            character.Move(agent.desiredVelocity, false, false);  
            coolingCountDown = 15.0f;  
        }  
        else  
        {  
            Debug.Log("cooling...");  
            agent.destination = character.transform.position;  
            character.Move(Vector3.zero, false, false);  
        }  
    }  
}
```

Game Development

Algorithm – IEnumerator

```
IEnumerator HeroSanTimer()
{
    while (true)
    {
        yield return new WaitForSeconds(1.0f);
        Hero._san -= 1;
    }
}
```



```
// 根据san动态调整speed multiplier
this.GetComponent<ThirdPersonCharacter>().m_MoveSpeedMultiplier = (float)(0.8 * Hero._san / 100 + 0.4);

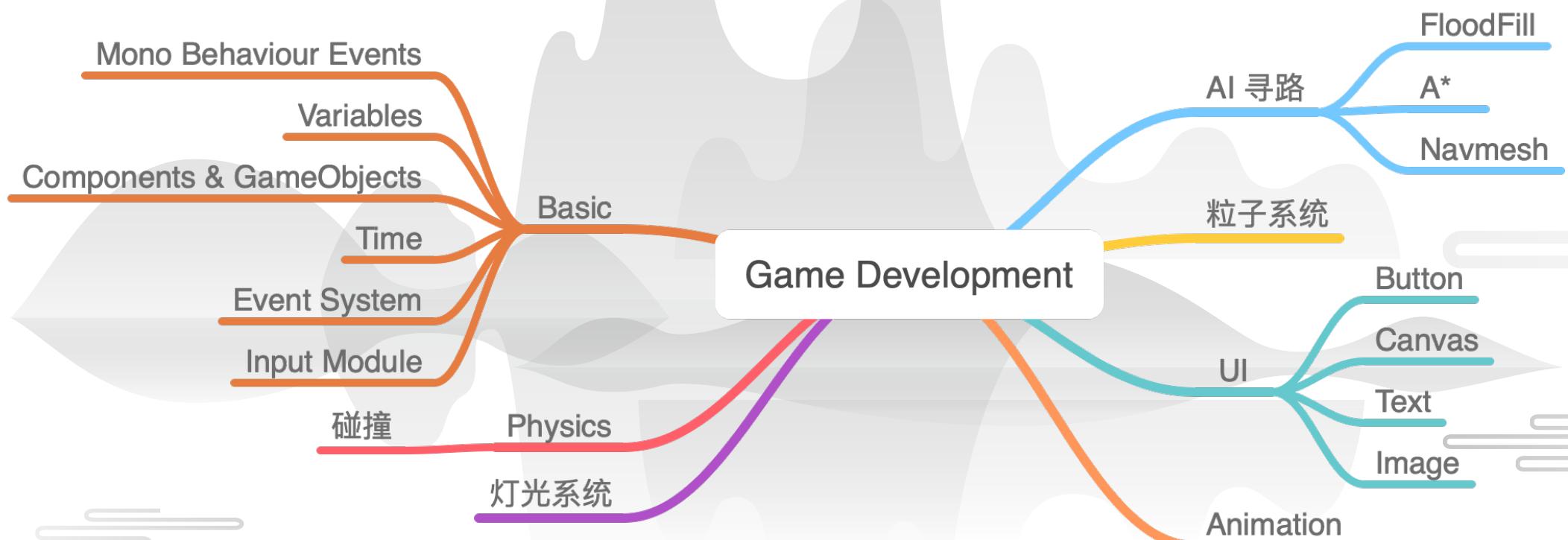
// 根据san动态调整 light 二选一
// san越低 场景越黑
// sceneLight.GetComponent<Light>().intensity = (float)(1.0 * (maxLightIntensity - minLightIntensity) / 100 * Hero._san + minLightIntensity);

// san越低 场景越亮
sceneLight.GetComponent<Light>().intensity = (float)(1.0 * (minLightIntensity - maxLightIntensity) / 100 * Hero._san + maxLightIntensity);
```

Game Test & Deployment

Game Test & Deployment

Course



Game Test & Deployment

Versioin Control

doubleZ0108 / Soul-Maze

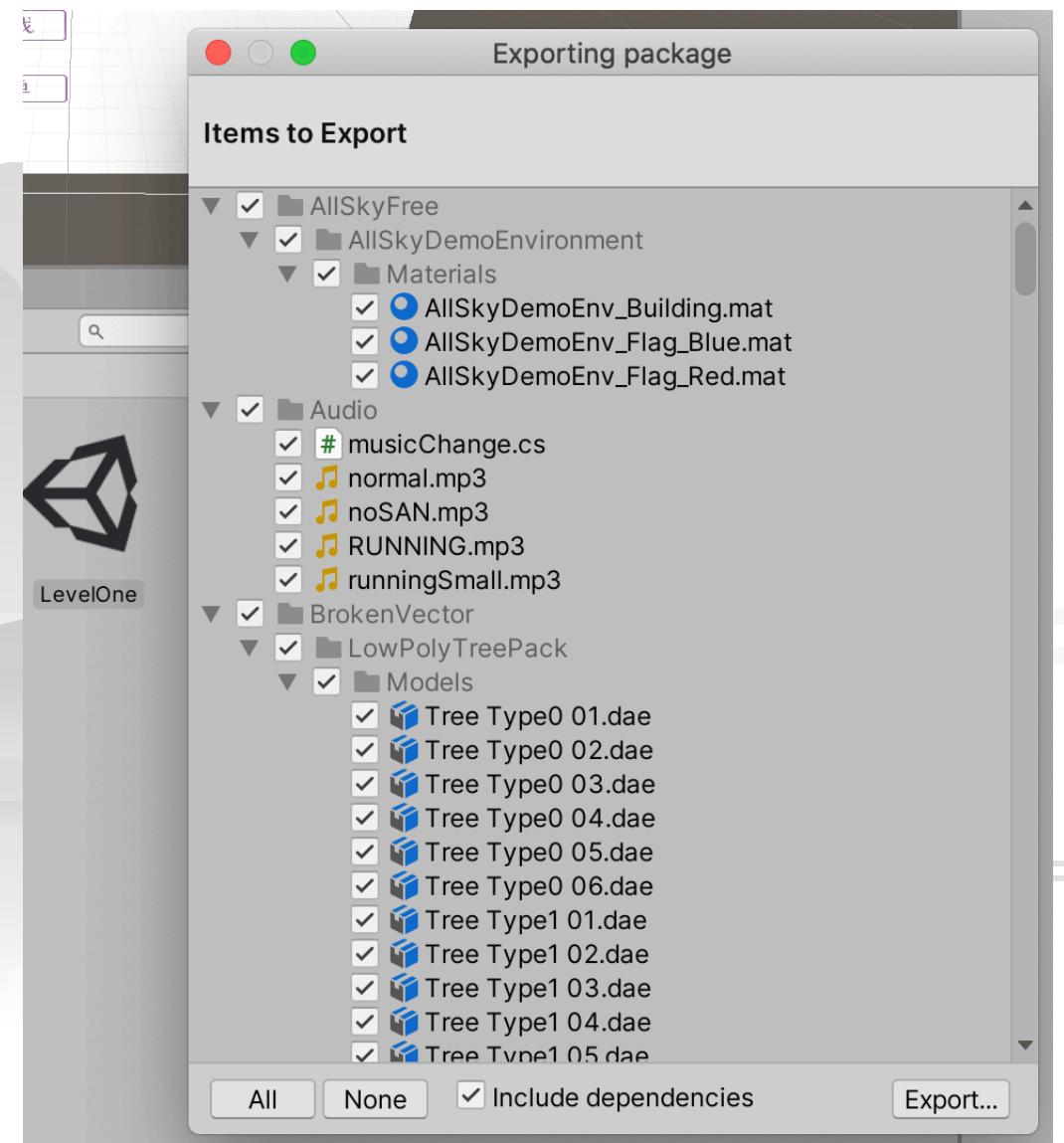
Unwatch

<> Code Issues Pull requests Actions Projects Wiki Security Insights Set

Branch: master ▾ Go to file Add file ▾ Clone ▾

ST-ern committed 66143c3 12 hours ago ... 18 commits 1 branch 0 tags

Assets	change LevelOne	12 hours ago
Packages	clean history	21 days ago
ProjectSettings	finish ai logic	2 days ago
.gitignore	finish backend main logic	16 hours ago
README.md	clean history	21 days ago



Game Test & Deployment

Team Work

张喆：游戏确立 + 人物 + AI + 游戏拼接

陈开昕：游戏确立 + 相机 + 美术 + 场景

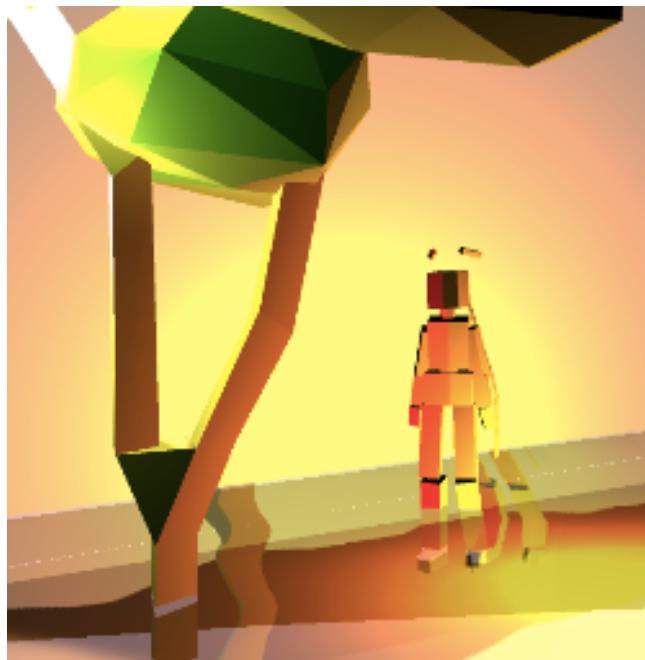
李文玥：游戏确立 + 地图 + 游戏背景 + UI

张靖萌：游戏确立 + 地图 + 游戏背景 + UI

—— MV取自吴青峰《译梦机》



译



梦



译梦



译梦机

Thanks for Listening

第一组 张喆 陈开昕 张靖萌 李文玥