

SHANGHAI JIAO TONG UNIVERSITY



游戏设计与开发：2D游戏创意改造

作业报告

学生姓名: 潘峰立

学生学号: 518021910835

授课教师: 杨旭波

学院(系): 电子信息与电气工程学院

软件工程专业

**目****录**

[**目录** 2](#_Toc68958576)

[一、游戏设计 3](#_Toc68958577)

[1.1 前期设计思路 3](#_Toc68958578)

[1.1.1 设计透镜 3](#_Toc68958579)

[1.1.2 游戏主题 3](#_Toc68958580)

[1.1.2 创意 3](#_Toc68958581)

[1.2 设计目的 4](#_Toc68958582)

[1.3 改动内容 4](#_Toc68958583)

[二、 程序实现 5](#_Toc68958584)

[2.1 主要功能 5](#_Toc68958585)

[2.1.1 弹弓弹射功能 5](#_Toc68958586)

[2.1.2 猪的受伤，死亡，以及对应的特效 8](#_Toc68958587)

[2.1.3 游戏逻辑的判定，实现多只小鸟的飞出 10](#_Toc68958588)

[2.1.4 小鸟飞出的拖尾效果 12](#_Toc68958589)

[2.1.5 显示失败/胜利界面 12](#_Toc68958590)

[2.1.6 一颗颗地显示星星并显示烟花粒子效果 14](#_Toc68958591)

[2.1.7 暂停游戏 16](#_Toc68958592)

[2.1.8 镜头跟踪 17](#_Toc68958593)

[2.1.9 播放音乐 17](#_Toc68958594)

[2.1.10 添加黄色的小鸟 18](#_Toc68958595)

[2.1.11 绿色的小鸟 19](#_Toc68958596)

[2.1.12 黑色的小鸟 20](#_Toc68958597)

[2.1.13 地图选择，关卡页面 21](#_Toc68958598)

[2.1.14 数据存储 22](#_Toc68958599)

[2.1.15 计分 23](#_Toc68958600)

[2.1.16 关卡之间的切换，加载，以及“下一关”按钮 24](#_Toc68958601)

[2.1.17 视差滚轴 25](#_Toc68958602)

[2.1.18 金蛋关卡 27](#_Toc68958603)

[2.1.19 大红鸟 28](#_Toc68958604)

[2.2 其他未归类的bug及解决办法 28](#_Toc68958605)

[2.2.1 分辨率 28](#_Toc68958606)

[2.3 优化设计 29](#_Toc68958607)

[2.3.1 不同的鸟/建筑显示的分数不一样 29](#_Toc68958608)

[2.3.2 退出游戏 30](#_Toc68958609)

[三、 参考资料 30](#_Toc68958610)

一、游戏设计

1.1 前期设计思路

在开始进行关于设计的思考之前，我先去复习了一下课上讲到的关于游戏设计的原理及各种方法，尝试利用各种游戏设计工具来指导我的思路方向。

1.1.1 设计透镜

（1）本质体验：我想要让玩家有什么体验？

——我想让玩家有休闲轻松的体验——这就要求难度适中。

（2）惊讶/趣味：什么让玩家感到惊讶/快乐/有趣？

——不断设计出新的鸟和关卡，通过不断更新来让人感兴趣。

（3）好奇心：玩家的真实动机是什么？他们玩这个游戏想要达成什么目标？如何买下疑问来激发好奇心？

——限时目标？关卡动画来揭示剧情？

（4）内生价值：在我的游戏中，什么对玩家是有价值的？比如索尼克的收集，不是单纯地收集。

——我这里的收集星星也不是单纯地星星，而是解锁隐藏关卡——或者新的鸟。

（5）解题：要求玩家解决什么问题？

——把猪干掉。隐藏问题——新问题——设计彩蛋！比如往回飞（《头号玩家》），能够有收集品，甚至还有隐藏的地下室等等隐藏关卡。

（6）四类元素MAST：

——机制基本固定了，但是仍然有很大的修改空间——比如可以自选小鸟，就像《植物大战僵尸》一样。

——美术，不是这次作业的重点…

——故事，大概的故事就是打败猪，夺回自己的家园。但是也可以设计一些彩蛋——甚至像模拟大鹅一样地类似《恐怖游轮》一样地，地下室有一个用猪去打鸟的隐藏图，致敬赛文奥特曼里面人类也是侵略者那一集。

——技术，随着需求的变化而一起变化。

但是最重要的是这四者之间的合作，也就是为了一个休闲娱乐的核心来服务，所以美术和音效等等一定要符合休闲的要求——简单不复杂，轻松欢快。

（7）全息设计：看到游戏的表和里。

1.1.2 游戏主题

就是“休闲”，为这个核心目的服务。

1.1.2 创意

获得好创意的办法，是先有一大堆创意，上面的思考已经包含了头脑风暴的内容。

进一步利用创意滤镜来筛选出创意

（1）我自己的内在感觉。美感，或者如果我作为玩家能够感受到惊喜。

（2）受众的喜好程度。比如我设计一个新鸟，就像《愤怒的小鸟》电影中的无敌神鹰的形象，那么电影的受众也自然会喜欢。

（3）体验设计是否良好：兴趣曲线——要保持难度和玩家现有的技能相匹配，保持在一定的阈值中。游戏平衡，也就是数值设计需要合理。

（4）创新性：比如助教给出的金蛋的创意在原版游戏中也有过，就不够好

（5）市场前景：本次作业由于投入有限，暂不考虑和商业游戏相比较

（6）技术可行性：如果我想要做自选小鸟的话，需要考虑到技术的难易问题——考虑到并没有前人做过，所以在创新性好的同时可能还具有一定的难度。

（7）社会目标：。

（8）测试反馈：按照迭代的思想，先做一个创意，收集反馈，然后修改等等

1.2 设计目的

稍微总结一下，愤怒的小鸟这款游戏并不能够明确地分为RTS或者RPG等等典型的类型，它的核心就是“休闲”二字，我们设计的目的便是为了休闲而服务。

所以愤怒的小鸟这款游戏的故事性的要求较弱，便舍弃了彩蛋的设计——细思极恐的设计可能在模拟大鹅这样一个本身有故事要求的游戏里面会很好，但是这里并不。

创意改造的想法大概有：

（1）就像《植物大战僵尸》一样，玩家能够自选用哪些鸟：设计成小红鸟只要一个格子，炸弹黑鸟要三个，并且随着关卡的升级，格子会开放。在给玩家自由度的同时，能够进一步提升游戏的变数，玩家可以自行组合小鸟，产生设计者都没想到的化学变化。

（2）设计新的鸟。就像《愤怒的小鸟》电影中的无敌神鹰一样。或者飞出去之后可以玩家自己操控的鸟。或者能够自动飞向最近的猪的鸟。

（3）金蛋关卡。

1.3 改动内容

考虑到实现成本，上面提到的三个创意只保留了后两者：即

（1）金蛋关卡。具体地说，是玩家在“尽可能少”地破坏金蛋的情况下获胜——如果必须全都完好无损，会有难度。所以设计成金蛋是一个奖励的设计，活下来的金蛋越多得分越高。但是如果金蛋全部损毁，那么还是判断游戏输掉。

（2）新的鸟。在原版游戏中的大红鸟没有特殊的技能，这里添加了一个能够玩家自己操控的功能——在它飞在空中的时候点击屏幕，大红鸟会改变飞行方向，飞向鼠标所处的位置，实现玩家自己可控。



图1 大红鸟

1. 程序实现

主要但是不限于功能的实现。其实还有很多unity的基础操作的小细节，这里不便延伸，仅抓住重难点的实现去说。

2.1 主要功能

2.1.1 弹弓弹射功能

1、实现方法

模拟弹弓的发射，主要依赖于SpringJoint组件。这个组件就像一个弹簧一样，连接两个刚体。

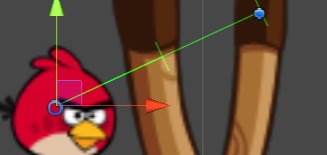


图2 Spring Joint 效果图

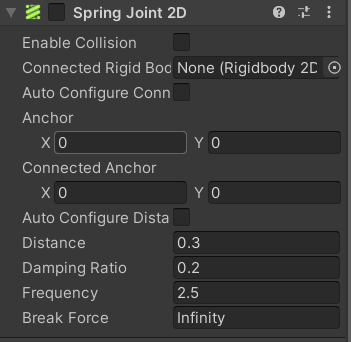


图3 Spring Joint的参数

我们可以通过修改这个组件的各个参数，来控制自然长度（distance），或者弹簧的劲度系数（damping ratio和frequency都有影响）。

但是spring joint这个组件只是一个弹簧的功能，它提供了超过一定的长度弹簧会断掉的功能（即break force），而弹弓需要的是一个最大长度的限制，所以需要我们手动实现最大距离控制。

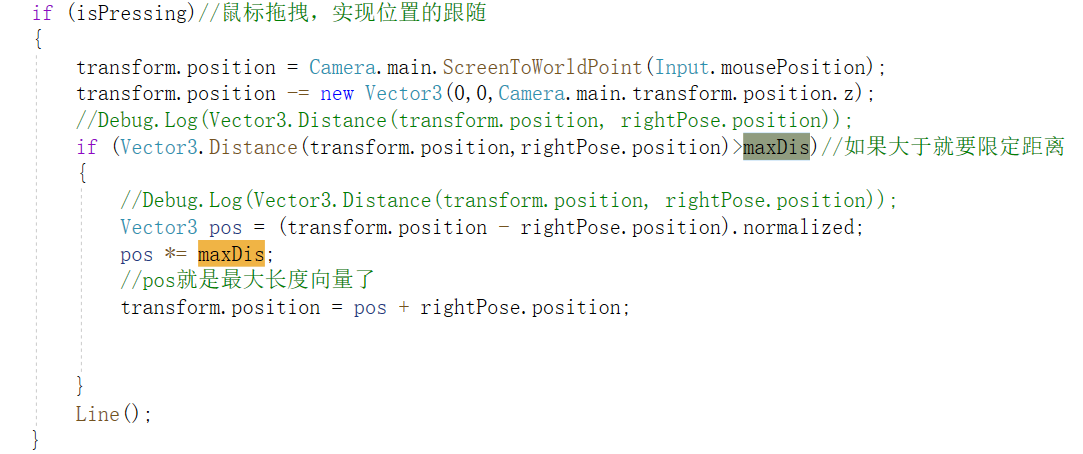


图4 实现最大距离的控制

上面的代码片段中包含了用鼠标拖拽小鸟的实现，用OnMouseDown函数来检测鼠标按下（在collider上按了鼠标的时候，就会调这个）。按下之后实现拖拽。但是实际上的鼠标坐标系和鸟的坐标系不一样，还需要利用camera进行一次转化。

有OnMouseDown那么也自然有OnMouseUp，松开鼠标之后小鸟就飞出。

这样说并不严谨，并不是松开后小鸟就飞出，目前的spring joint会一直绑定在鸟身上，所以会一直做弹簧的来回运动。

所以我们需要把spring joint给失活（disable，或者enabled=false）。那之后鸟就不受弹簧的约束，可以飞出。

实现还没有结束，目前的弹簧是一个隐形的，没有颜色的，透明的。所以如果想要模拟成弹弓一样，还需要画线。这里用到line renderer组件。

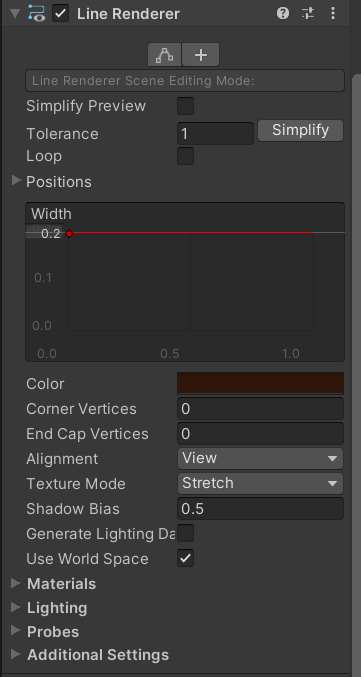


图5 Line renderer组件

当鼠标按下时开始划线，当飞出去之后划线消失。

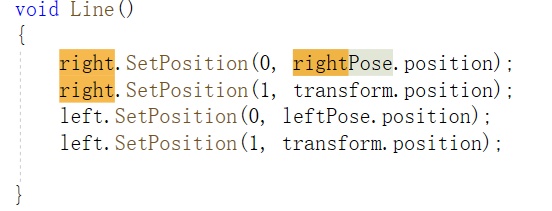


图6 划线代码实现

两点确定一条直线段。只需要把鸟和对应的两端的弹弓设置进去即可。



图7 划线效果图

2、遇到的问题及解决方案

（1）层级问题

在一开始设计弹弓的时候，为了在2D游戏中能够看出3D效果，看起来真实，要求左边的弹弓会遮挡鸟但是右边的弹弓不会，以体现出小鸟是在两个弹弓中间。这就涉及到层级遮盖问题。

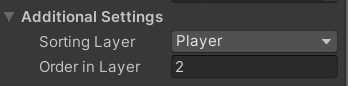


图8 Layer设置

在Sprite renderer组件中，可以设置层级。层级判断的方法大概是——先判断Sorting layer，比如这里我自己设置的player层就比default层高。然后在同一个sorting layer中，就根据order in layer来判断，2比1高。

（2）飞不出去

这是一个比较严重的问题，关于什么时候把spring joint给disable掉。我参考的教程的设计是在松手后一段固定长度的时间后失活。这种实现方法对于时间参数的选择要求很高，而且可想而知肯定有问题——不同伸长长度的弹簧到飞出去的时间肯定也是不一样的，如果时间过短，会出现加速不够就飞出去，如果时间过长则会本来已经飞出去了还要被拉回来。

所以我又采取了自己的实现方法，当鸟和弹弓之间的距离小于“松弛长度”（或者叫自然长度）之后就松开。但是还是会出现飞不出去的问题——小鸟会一直绕着弹弓转圈。

经过debug之后，我发现了这是因为距离根本就没有小于自然长度。进一步判断是因为小鸟的重力的原因，导致了有一个垂直于弹簧方向的速度，所以不一定会小于自然长度。

之后便修改成小鸟在弹弓上的时候没有重力（gravity scale设置成0），飞出去之后再把重力还给他。

那之后便好了许多，但是还是偶尔会出现飞不出去转圈的情况。

进一步思考，才发现了是用户自己的问题。在我们拉动弹弓的时候，可能鼠标会给一个旋转的速度。所以最后我在脚本里面对松手时的速度进行了处理，去掉切向的速度，只保留法向的速度。

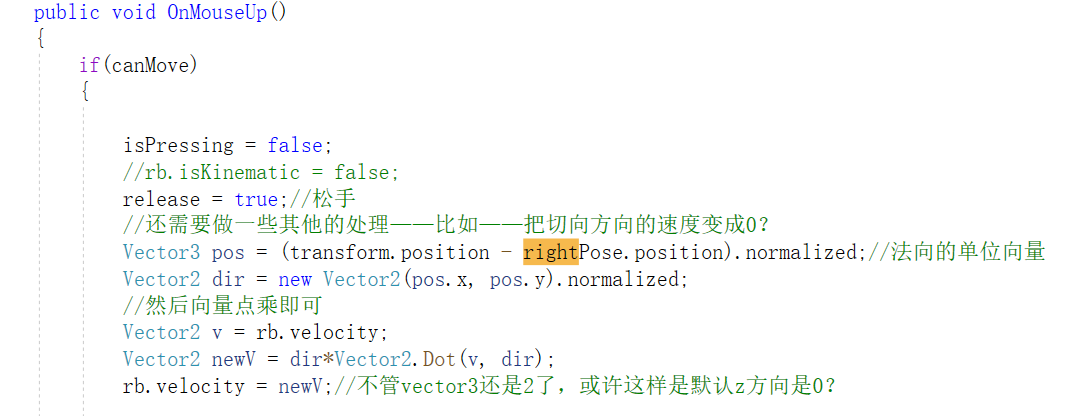


图9 抬手松开的预处理

2.1.2 猪的受伤，死亡，以及对应的特效

1、实现方法



图10 猪的脚本参数

参数的意义一点点来讲。

猪一开始是没有受伤的，在撞到建筑物或者鸟，或者地面之后会受伤，然后就会变成鼻青脸肿的样子。

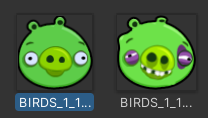


图11 Sprite的变化

所以就直接把sprite修改成后者即可。

但是有一个直接的问题是——如果一碰就碎的话，那么猪一开始落到地面上的时候就会马上受伤，这显然不是我们希望的。

所以设置了minSpeed，只有相对速度大于min Speed，才会受伤。

Max speed类似，如果相对速度大于这个速度，那么就会死亡。

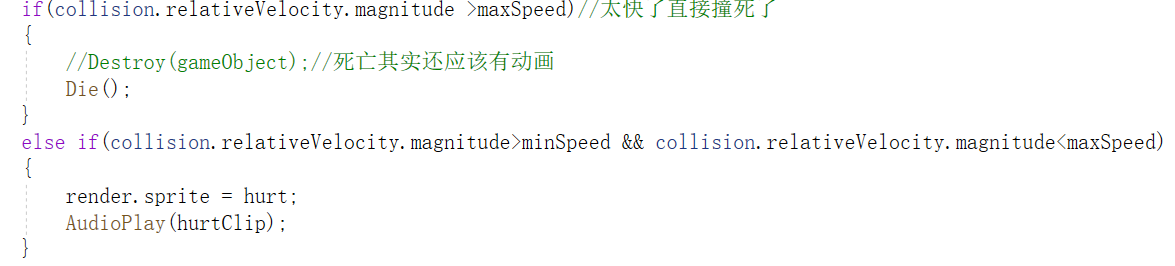


图12 Min和max speed，相对速度

特效分为两种，一个是死亡的爆炸特效，另一种是计分特效。

便用到animator。

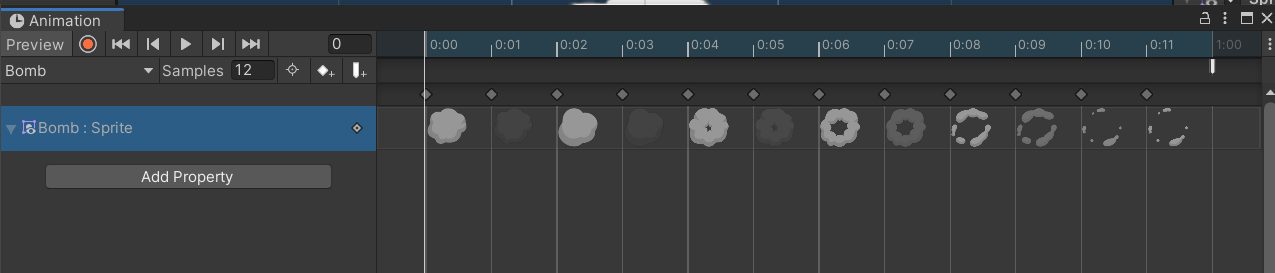


图13 爆炸的animator

只消把爆炸的各帧图片放入即可。注意动画的loop time设置取消勾选，只播放一次

计分特效更简单只需要一张图即可

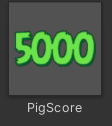


图14 计分特效



图15 死亡特效同时加分

这里的go便是加分特效。显示的位置需要注意，y坐标加了0.5，显示在物体的上方，这样才不会被遮挡。停留1.5秒之后消除掉。

2.1.3 游戏逻辑的判定，实现多只小鸟的飞出

1、实现方法

游戏的逻辑包括胜利和失败的判定——小鸟的个数大于猪的个数就胜利，如果小鸟用完了猪还有剩，那么就失败了。

自然需要用到game manager。



图16 Game manager参数

这里的list获取到小鸟，来实现一个飞出之后，能够获取到下一个鸟。

当前面的鸟死亡之后，就destroy并从list中移除。并且调用下一只鸟。

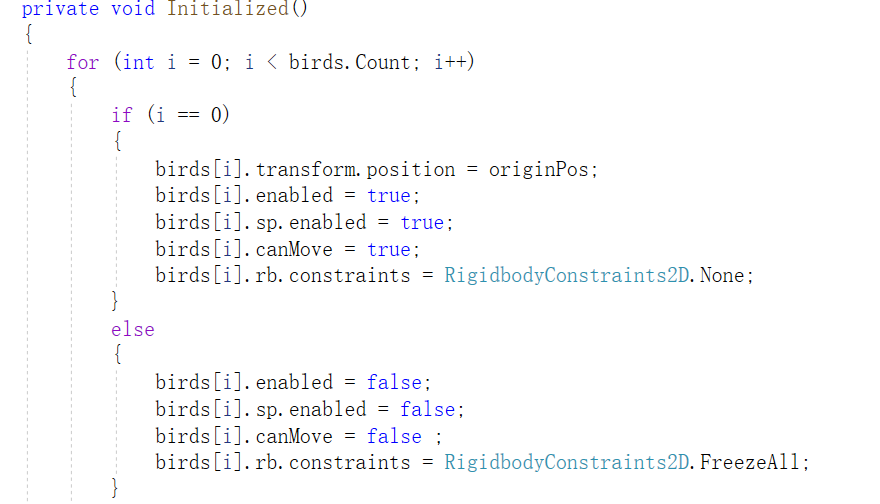


图17 初始化，即调用下一只鸟

上面的代码还包含了两个问题的解决办法：

一是位置记忆的问题，我们需要记住第一支鸟的位置，然后后面的就设置成originPos，不然的话在每次更新的时候鸟就会忽然飞向弹簧，来回摆动。

二是固定不动的问题，如果在鸟飞出去之前受到了碰撞，应该是不会有任何影响的。

2、问题及解决办法

我们需要获取到鸟的sp来做enable和disable操作，但是这本来希望是一个private的，所以：

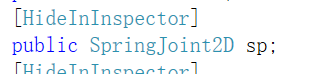


图18 HideInInspector

这样能够保证外部不可见，但是脚本可以访问到。

此外，关于胜负的判断，我找的教程里面是在每次更新鸟的时候去判断。但是在实际操作中我发现这样有问题，可能会在我们已经更新好鸟之后，在发射新的鸟之前，猪并没有停止滚动，会继续从高台落下，有死亡的可能。如果用之前的实现方法，那么必须要发射出去新鸟之后等下次更新才会判断胜负，这样是不对的。

所以我修改成了每次update的时候去判断。

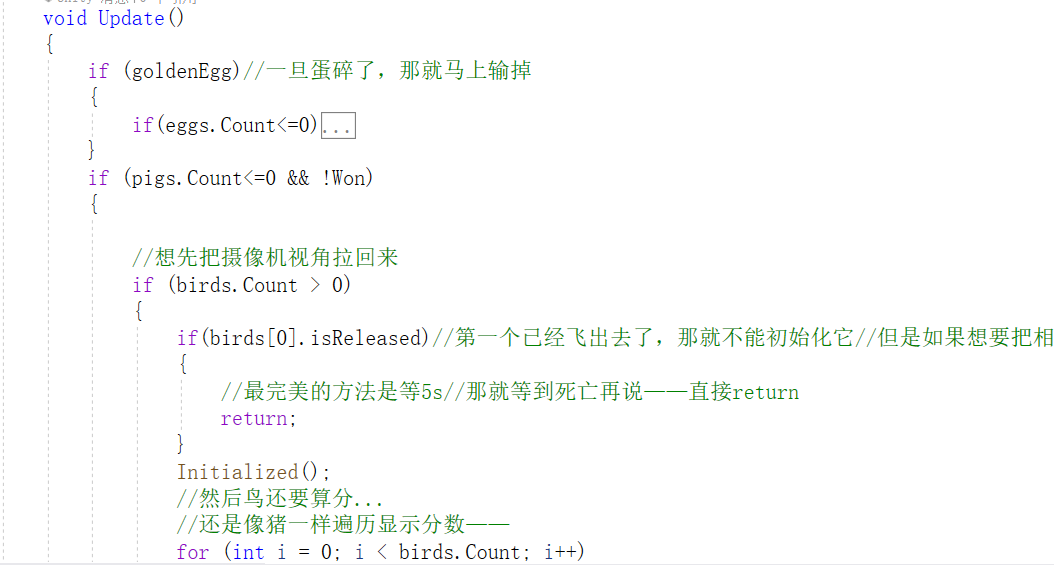


图19 判断胜负

同时加入了Won参数来表示结果是不是已经出来了。

以及一开始做的时候会遇到报错的问题，经过研究是awake和start的问题：

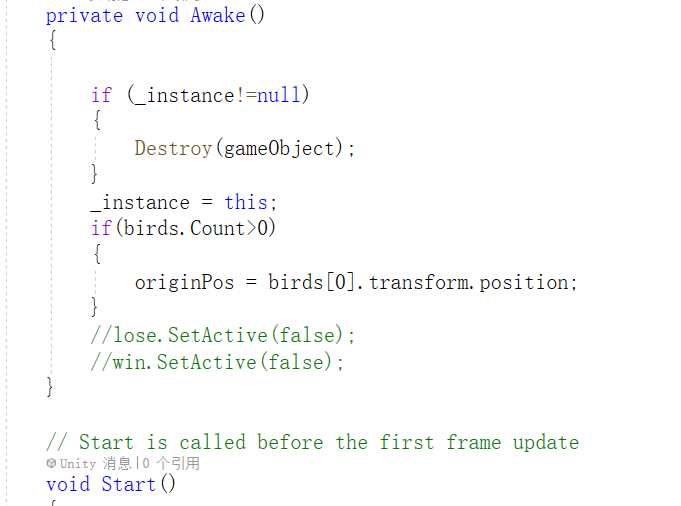


图20 单例模式

应该在awake中实现单例。毕竟awake比start要更早调用。

2.1.4 小鸟飞出的拖尾效果

1、实现方法

主要参考了一个博客的武器拖尾的实现。添加了mesh render来处理拖尾效果。



图21 Mesh renderer

并且为了小鸟只是在天上飞的时候才有拖尾效果，进一步完善bird的逻辑。



图22 拖尾效果

2、问题及解决办法

拖尾显示不出来的问题。经过研究，发现是z轴的问题，导致了背景图片遮挡了trail尾迹。

2.1.5 显示失败/胜利界面

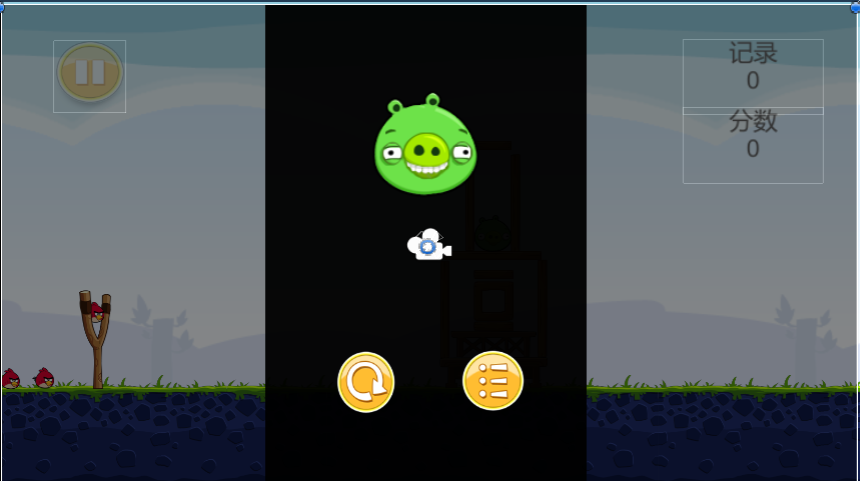


图23失败界面

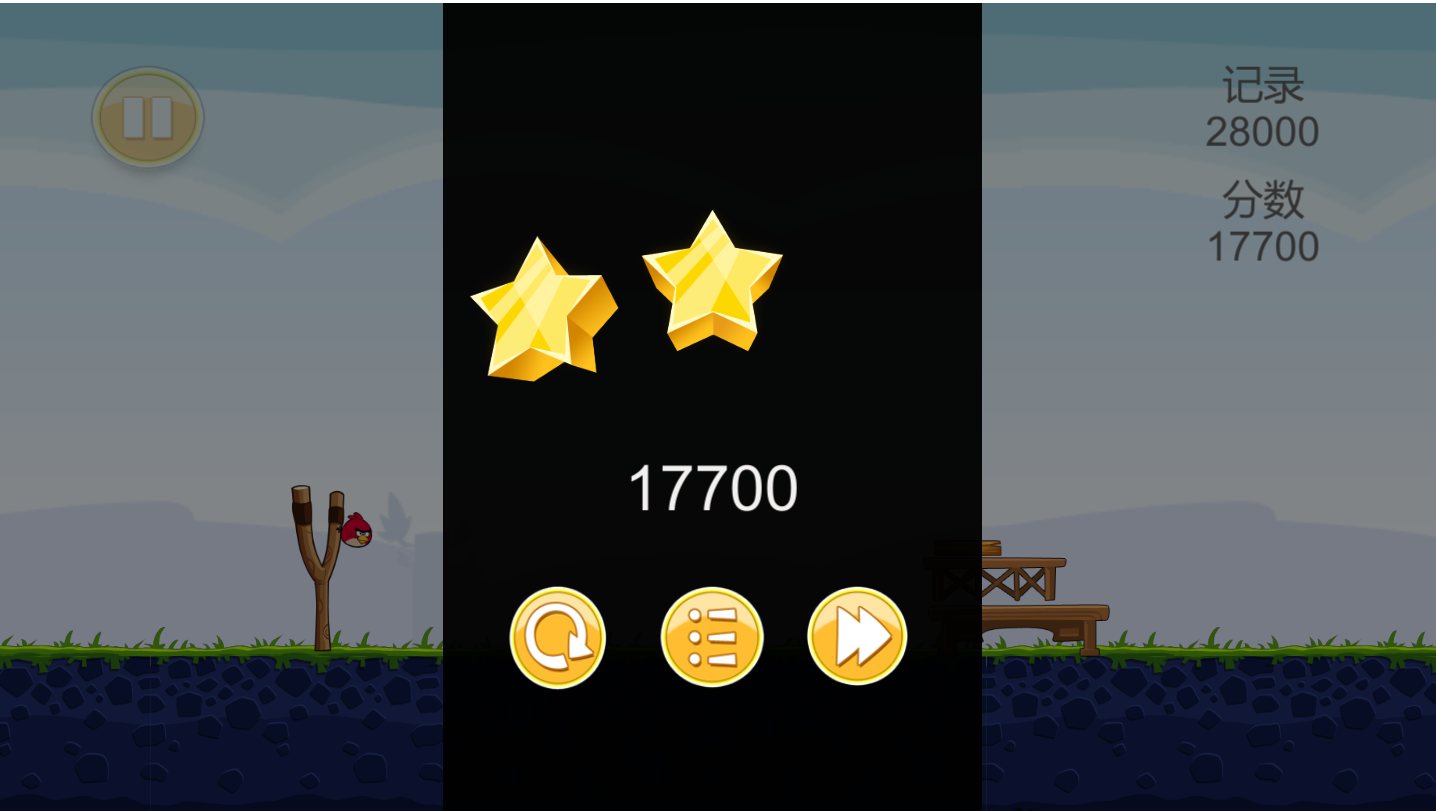


图24 胜利页面

添加UI-image。我希望它的出现是渐变的，所以添加动画



图25渐变动画

动画设置界面可以修改透明度，一开始从0变到之后的数值。

赢的界面和输的界面类似，只是要显示星星。

2.1.6 一颗颗地显示星星并显示烟花粒子效果

1、实现方法

星星一颗颗地显示，只需要把三颗星星设计成三个独立的物体，然后用脚本控制显示即可。剩两个以上鸟的时候，显示三颗星星，一个的时候显示两颗星星，都用完了就显示一颗星星。

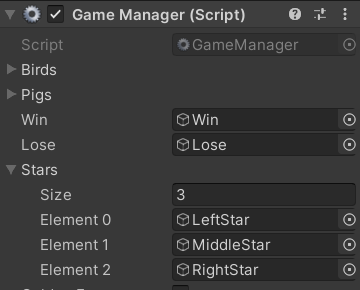


图26 Game manager里面获取到三个星星

同时为了一颗颗地有顺序地显示，设计了显示的时间间隔。

——用到了协程的方法，和wait函数配合来实现控制时间间隔。类似的方法后面也有

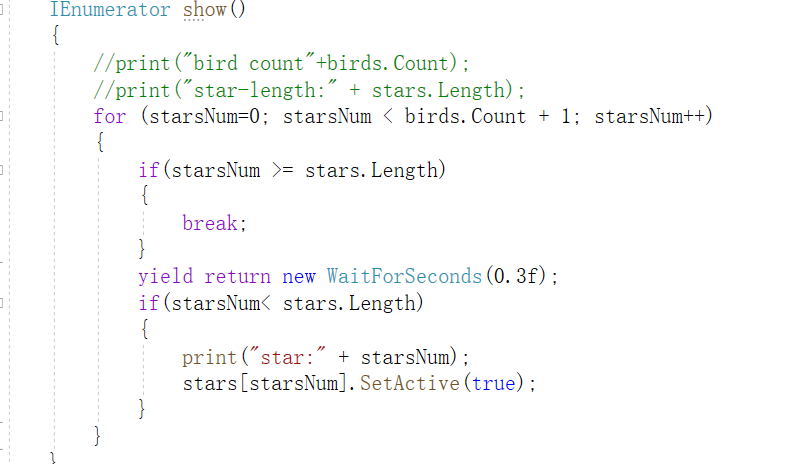


图27 协程

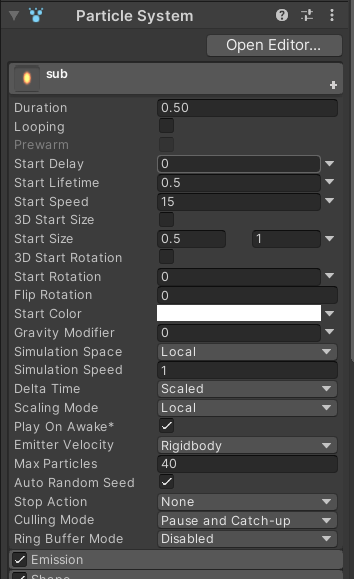


图28 粒子系统

粒子系统我是参考的教程，和自己的项目进行配饰，并没有自己编写底层代码。所以之后还自己实现了视差滚动…

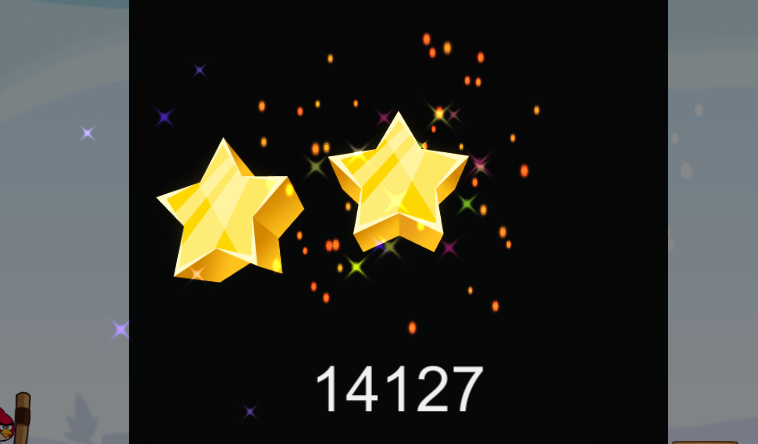


图29 效果图

2、问题及解决方法

烟花爆炸效果的显示问题，层级的问题。

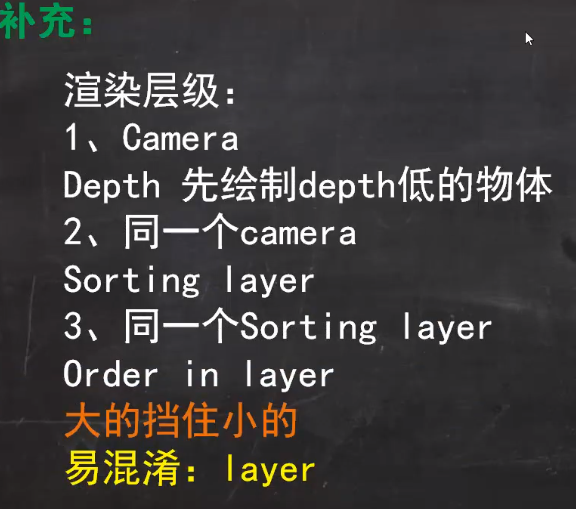


图30 渲染层级的先后

所以我新建了一个相机去拍摄UI的物体，并且修改了culling mask，让它只拍摄到UI界面的东西

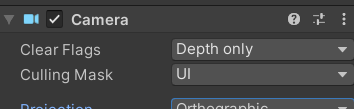


图31 UI camera

2.1.7 暂停游戏

设计一个侧边栏，按了暂停按钮之后就会弹出侧边栏。



图32 侧边栏

这个侧边栏本来是藏在摄像机之外的。是弹出动画让他能够弹出，而且能够弹回去。

和之前的改透明度类似，这里修改坐标。

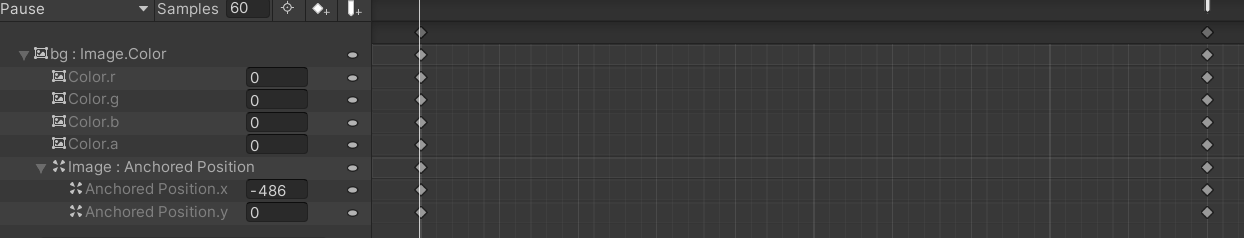


图33 弹出动画

从图中可以看到，我在动画完成的最后一帧加了一个事件。

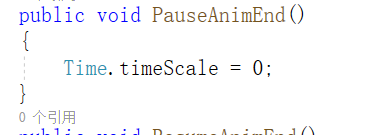


图34 暂停事件

这个事件中把timeScale设置成0，来实现暂停的效果。

然后回到游戏的时候需要把它设置成1

暂停的弹出和resume的弹回实际上是两个动画，需要有一个转化的过程，所以用到animator，通过改变参数来改变状态。

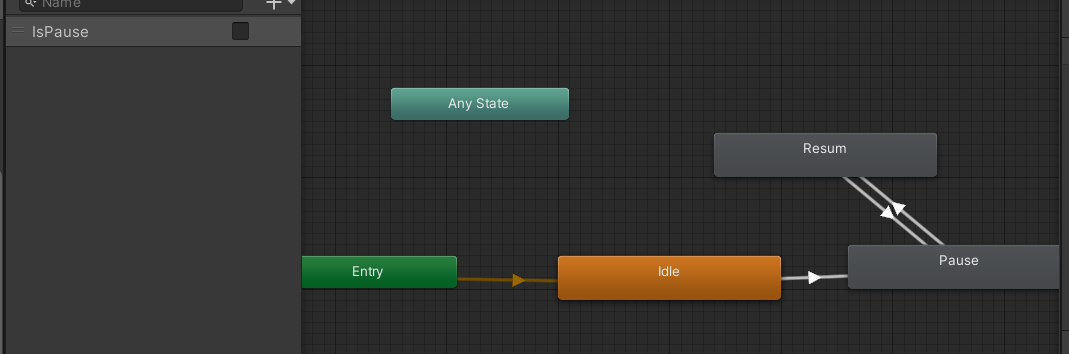


图35 Animator

2、问题及解决方法

TimeScale一旦设置成了0，那么动画也播放不了，所以必须要等动画播放完之后，游戏才会完全暂停。

遇到的常见的问题是，暂停之后点击重新玩游戏之后，就动不了了，没有反应。

原因就是时间设置成0了之后没有还成1

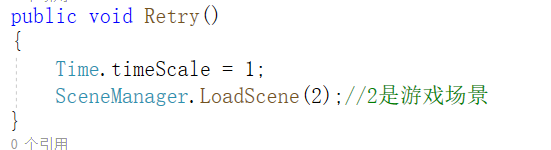


图36 Retry，重新开始本关

所以只需要在每个按钮对应的函数里面确保时间还回去即可。

2.1.8 镜头跟踪

我们的摄像机能够拍到的范围是有限的，但是小鸟可以飞很远，这个时候就需要镜头移动，实现跟随。

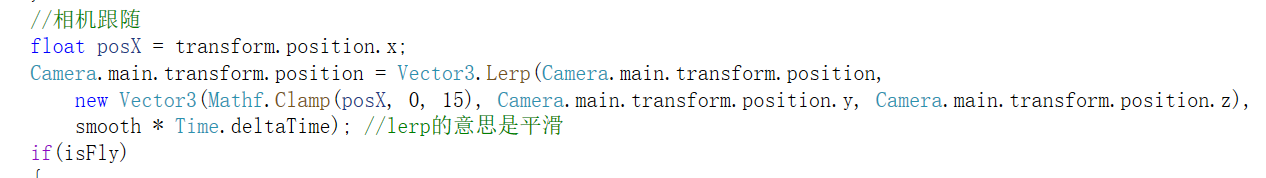
1、实现方法

图37 相机跟随

Clamp函数的意思是，把posX限定在0到15之间，这样让相机有一个最远距离，如果鸟飞得超过了这个距离就不再跟随。

Smooth是平滑程度的意思。

这种实现方法在之后的视差滚动中有参考。

2.1.9 播放音乐

一方面是背景音乐，给main camera添加audio source组件

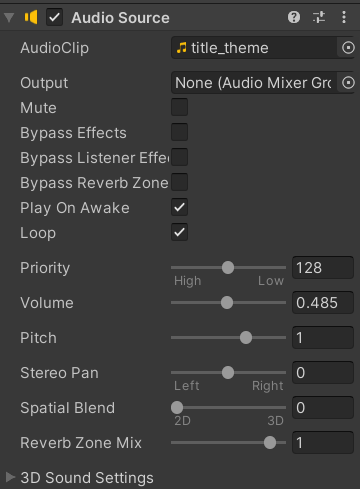


图38 Audio source

这样就会一直循环播放。

另一方面是音效，比如小鸟被选择会叫，飞出去会wryyy等等。添加一个函数，来播放外部设定的audioclip即可

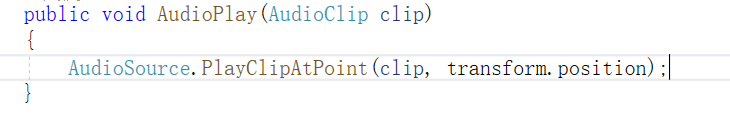


图39 音效

猪也同理

2.1.10 添加黄色的小鸟

黄色小鸟和普通小鸟的区别是在于，飞出的过程中点击鼠标的话，它的速度会加快，而且一旦碰撞之后就不能再点击加速了。

由于变化不大，所以采用继承的方法，继承已经写好的bird的脚本，给一些virtual的方法来给子类重写



继承

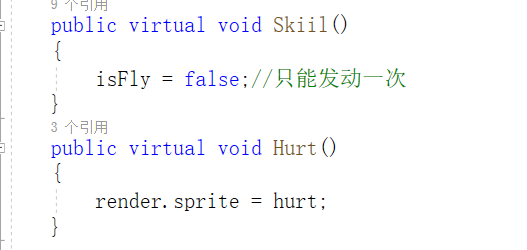


图40 虚方法

黄色小鸟的技能，速度变快，就直接修改速度为两倍

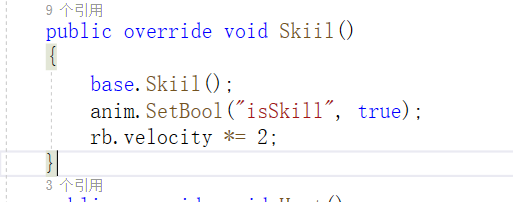


图41 速度变快

关键问题在于，如何判断在空中，如何使得撞击了之后便不再能放技能。这里通过各种布尔变量来联合判断，比如isFly表示能不能发动技能，在飞出之后是true，撞到了或者发动过之后就是false等等。

最后就是飞出的时候，黄色小鸟的sprite也要改变，改成冲刺的样子。



图42 冲刺效果

这个我是通过动画来实现——因为当他撞击之后还会受伤。

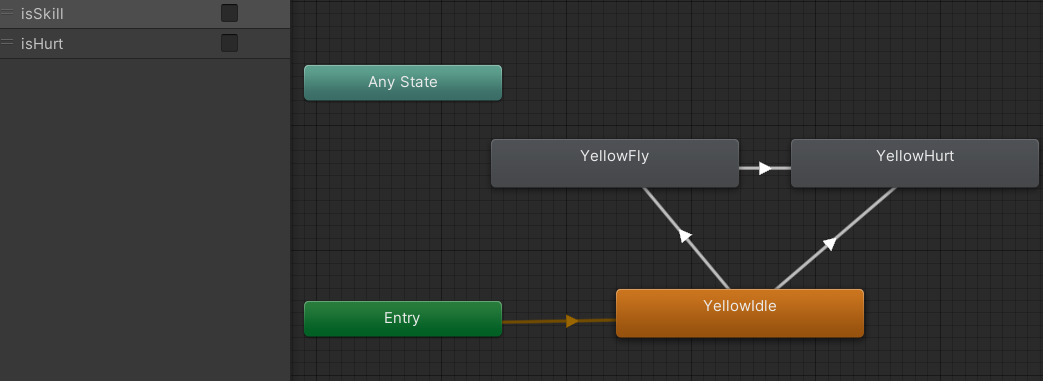


图43 Animator来控制状态

2.1.11 绿色的小鸟

绿色的鸟的功能是回旋，不过这里的实现比较简单，只是单纯地x方向速度改变而已

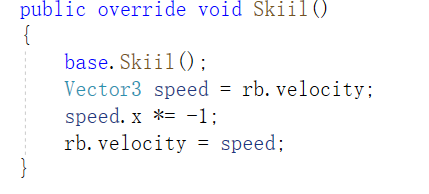


图44 “回旋”的实现

2.1.12 黑色的小鸟

1、实现方法

黑色小鸟的爆炸效果简化成为destroy它周围的敌人和建筑

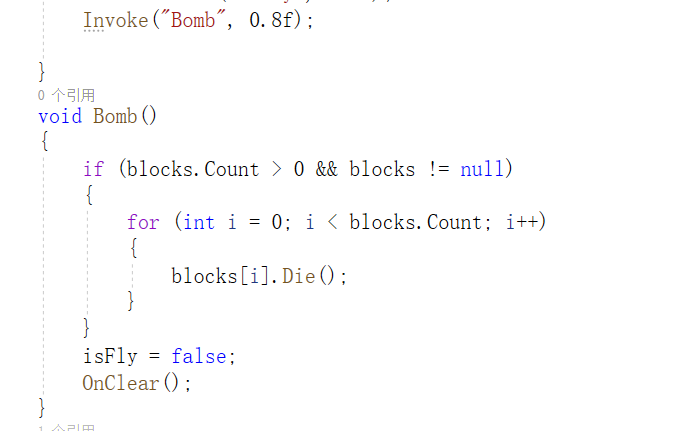


图45 爆炸效果

上面的block这个数组边是存储它周围的敌人和建筑的一个数组。我用trigger来实现判断和存储。

建立一个更宽一点的circle collider，并设置成trigger

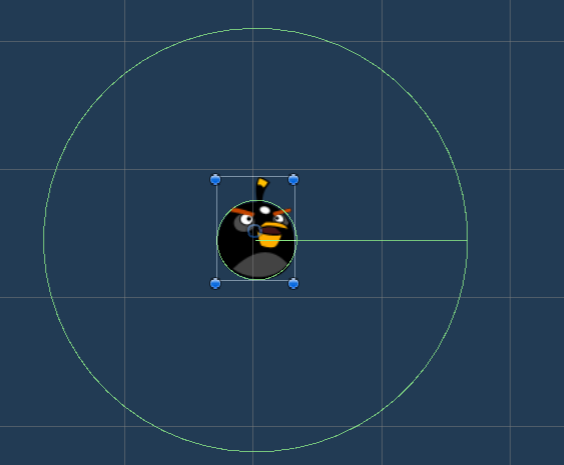


图46 可以看到有两个绿色的圈

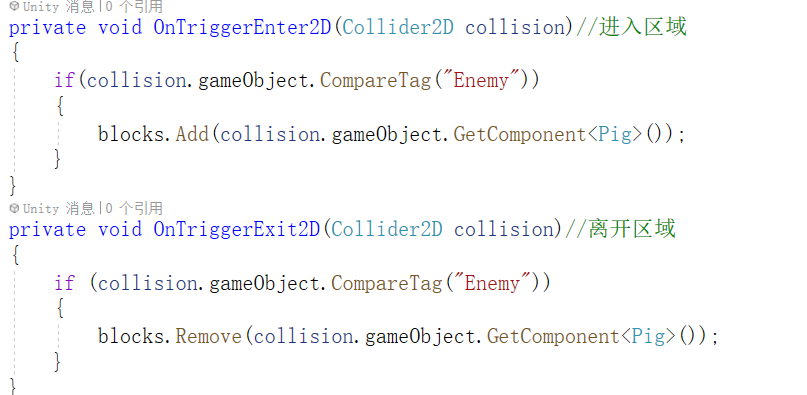


图47 实现存储

补充一下，这里getcomponent都是pig类型，是因为建筑和pig都是用的一个脚本，只是用一个bool值来区分到底是建筑还是pig



图48 IsPig

此外，黑鸟还有一点特别的是，它应该是在撞击之后也可以炸。所以就需要在bird脚本里面多整几个virtual的函数来让黑鸟去重写。

2、问题及解决办法

炸掉之后尾迹还在，以及特效播放两次等等的问题。因为我们的destroy是在飞出去5s后才调用，所以在炸掉之后需要专门用一个函数来处理后事。

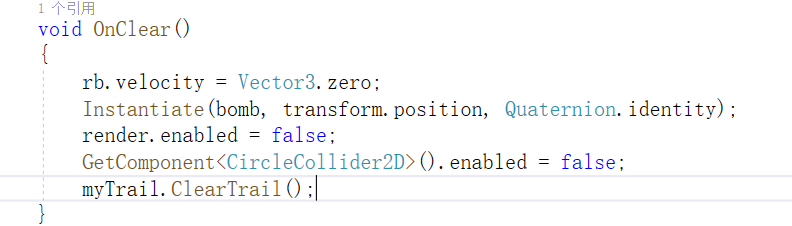


图49 爆炸后处理函数

还遇到过提前爆炸的问题。重写awake函数来保证初始化的正确性。

2.1.13 地图选择，关卡页面

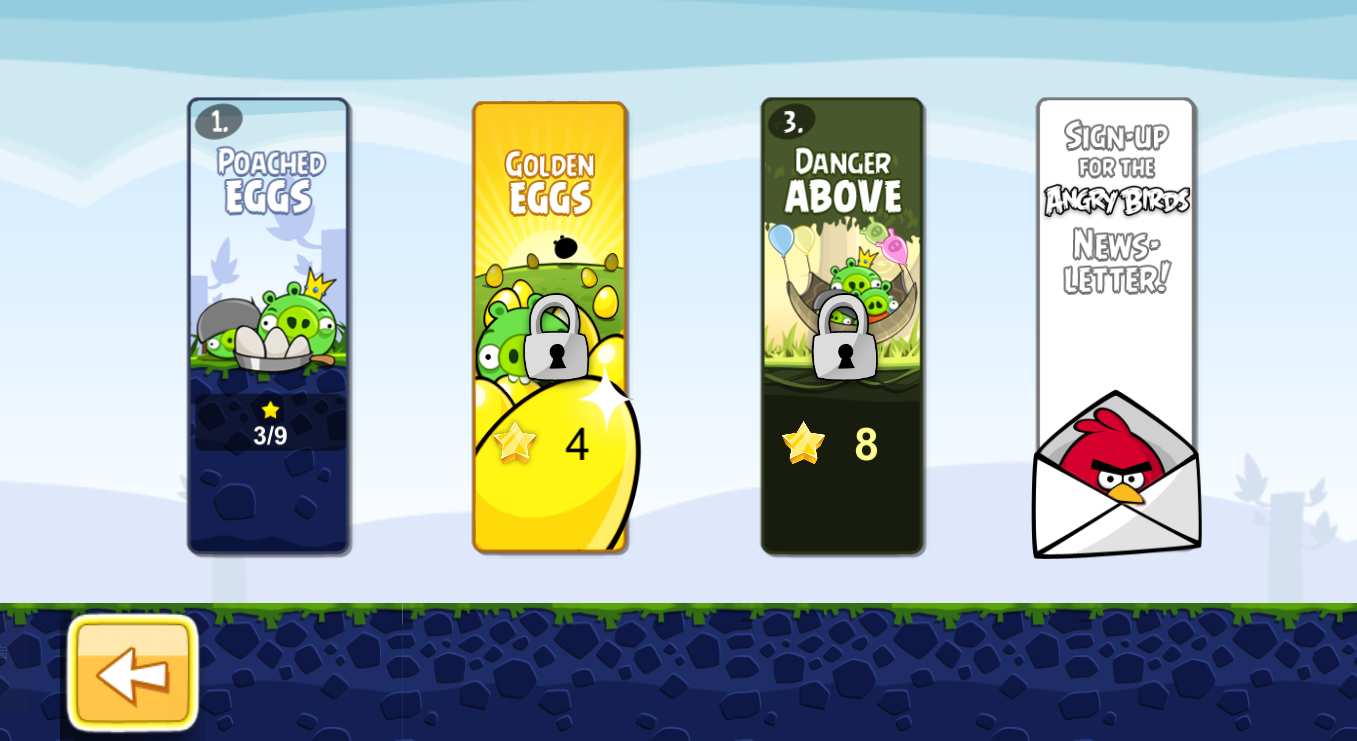


图50 地图的选择

地图选择的主要问题就是解锁的问题，需要达到一定的个数的星星才能够解锁，在解锁之前是不能点击的。这就涉及到数据存储的问题，之后细说。

然后选择一个地图进去之后，就会显示里面对应的各个关卡

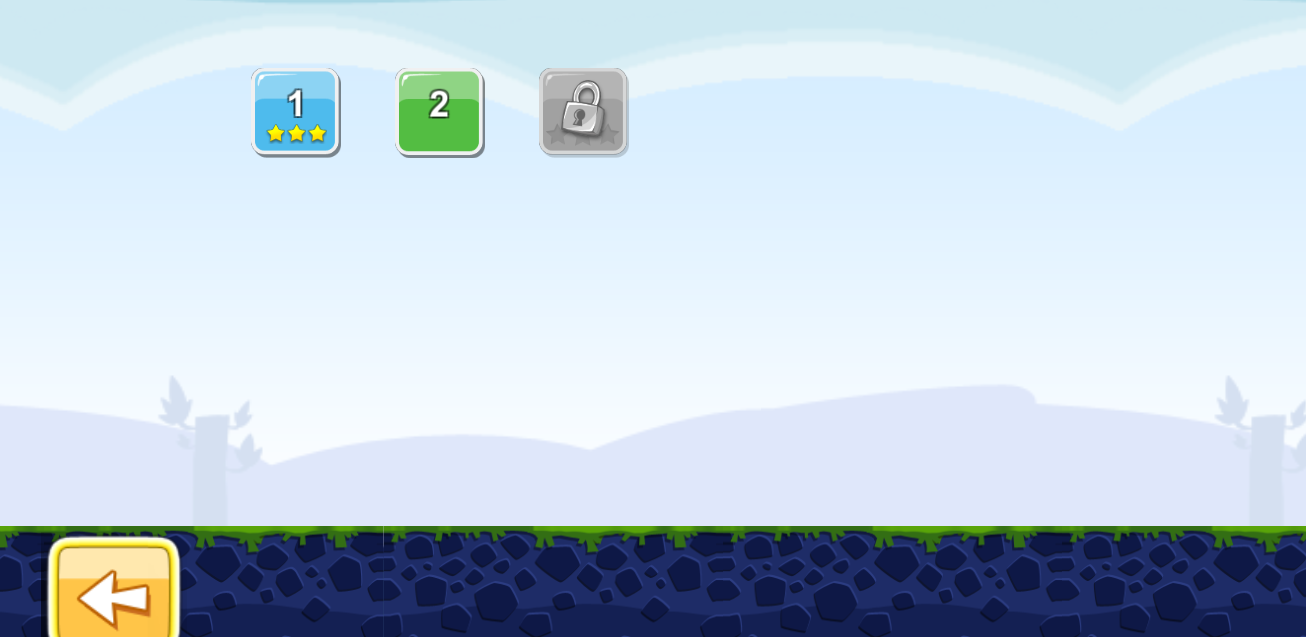


图51 关卡选择界面

这也涉及到一个解锁的问题，需要前一关已经打过了之后才能够解锁后一关，同时，每个地图的第一关是不用解锁的。

这里利用了gird layout group来方便管理

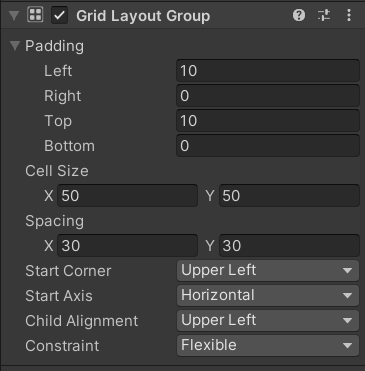


图52 Grid Layout Group

这里可以设置单元的大小，行间距等等，可以自动对齐，就不用手动去移动。

2、问题及解决办法

一开始会遇到这样一个问题，当我点击关卡1的时候，可能会点到关卡2上面去.经过检查，发现了是因为在2里面的text太宽了，延伸到了1的位置上去，所以才会出现误点的情况。

2.1.14 数据存储

数据存储有一个通用的方法：PlayerPrefs

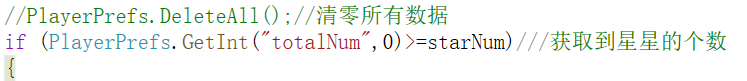


图53 存储/读取/清楚数据

数据存储的原理就是存储键值对，key是一个string，value可以是各种数据，int，string，bool等等，然后读取的时候就是根据对应的key去get相应的值即可。和web开发里面用到的session storage类似。

此外，还有一些有趣的存储方法。

比如在选择关卡的时候，把当前关卡的名字也传输进去——这里获取到了gameObject的name属性。用于之后根据名字去判断对应的是哪个关卡。

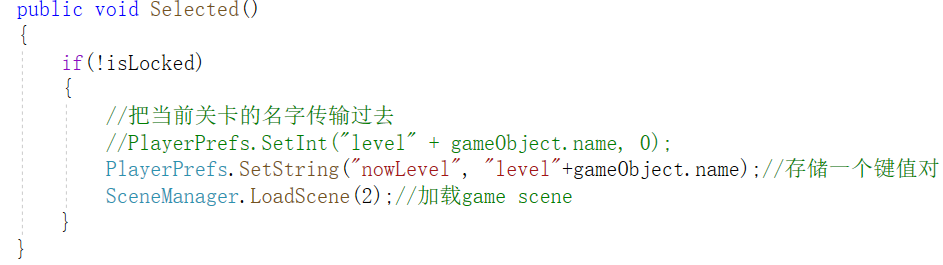


图54 存储关卡名字

2.1.15 计分

计分的涉及参考了一个网页版的愤怒的小鸟。



图55 参考对象

我们需要记录的分数有两部分，一部分是关卡对应的最高分，一部分是这次打出的目前的分数。

在加载每一个关卡开始的时候把当前的分数清零

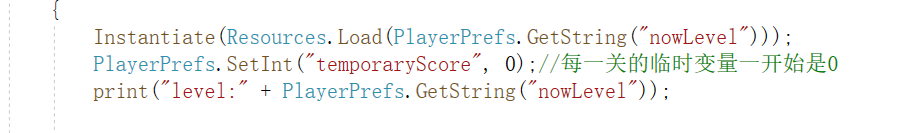


图56 这里的temporaryScore就是当前的分数

然后在每次猪死亡或者建筑损坏的时候去更新它。并且在最后结算的时候幸存的小鸟会算分，进一步更新

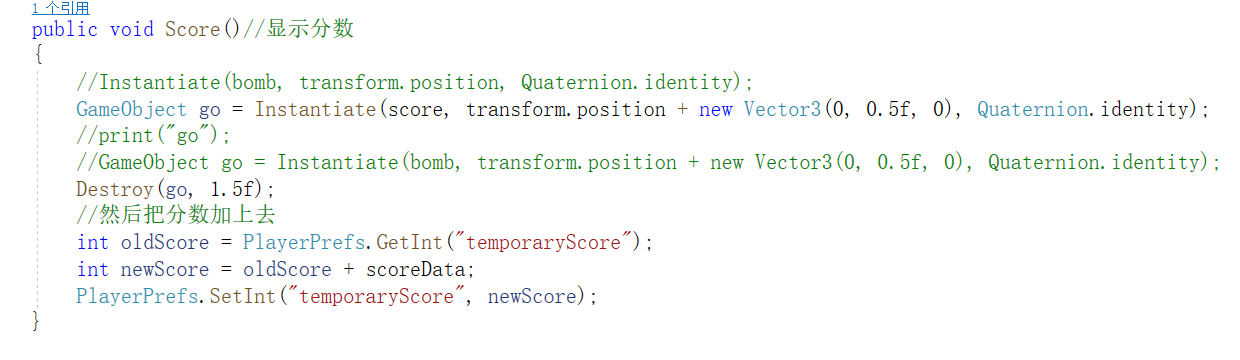


图57 Bird.Score()

此外，还有最高分的记录。

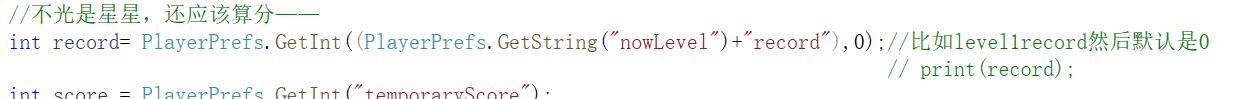


图58 Win的时候的记录更新

利用我们已经保存的当前关卡的名字（即上述的GetString(“nowLevel”)所对应的value）。我们能够保存对应的record记录。当当前的temporary的分数比之前的记录还要高的时候，就会提示“新纪录”，并更新记录。

此外，在现实分数的时候，我还实现了分数跳动的动画。

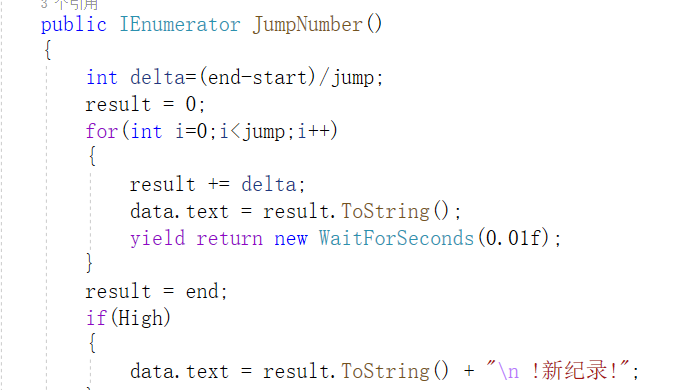


图59 协程实现分数跳动

每0.01秒更新一次分数，jump表示跳动次数，这样就可以实现跳动的效果。

2.1.16 关卡之间的切换，加载，以及“下一关”按钮

我们每一关的内容都不一样，所以它不是固定的，需要我们在每次加载游戏界面的时候去加载对应的关卡的数据

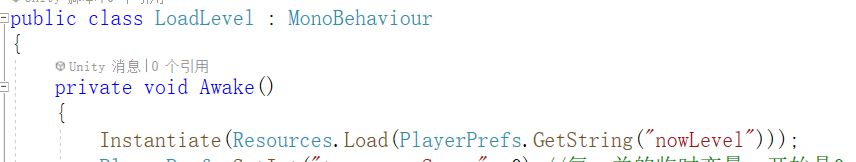


图60 加载关卡的时候回去实例化对应的resource

我们所有预先做好的关卡都会放在一个resource文件夹里面，只需要对应的名字就可以load出对应的场景。



图61 Resource

需要说明的是，这里的名字是单增的，比如level1，level2…level9。其实也可以设计成level1-1，level2-1之类的，那样或许更灵活。这个名字的取法，就和接下来的“下一关”的实现方法有关。

我这里涉及的下一关，是如果在当前的地图不是最后一关，那就跳到下一关，如果是当前地图的最后一关，就回到地图选择界面——即，并不做“最后一关不显示‘下一关’按钮”这样的处理。在写报告的时候我才想起来，其实可以直接提到第二张地图的下一关，那样甚至更简单…



图62 NextGame()进入下一关

上面的实现方法，需要获取到当前选择的是哪张地图，不同的map对应的最后一关的名字也不一样，根据和当前关卡名字的对比结果来进行判断。

2.1.17 视差滚轴

1、实现方法

视差滚轴的意思就是，不同层面的物体的移动速度不一样，这样看起来能够更有层次感。这里大概分成两层，一层天空的背景，另一层是包括地面，建筑，小鸟，弹弓等等在内的前景。



图63 Front和BG

具体代码实现思路，就是背景的移动速度要慢一些，最前面的应该和相机移动的速度一样——或者换句话说，背景要和相机一样的方向（同向），以一定的速度移动——那么就和相机的速度有关，所以我们需要获取到相机的信息——所以在每次update的时候取一个Δ，然后背景的Δ会逐渐增大，也就是移动速度逐渐加快。

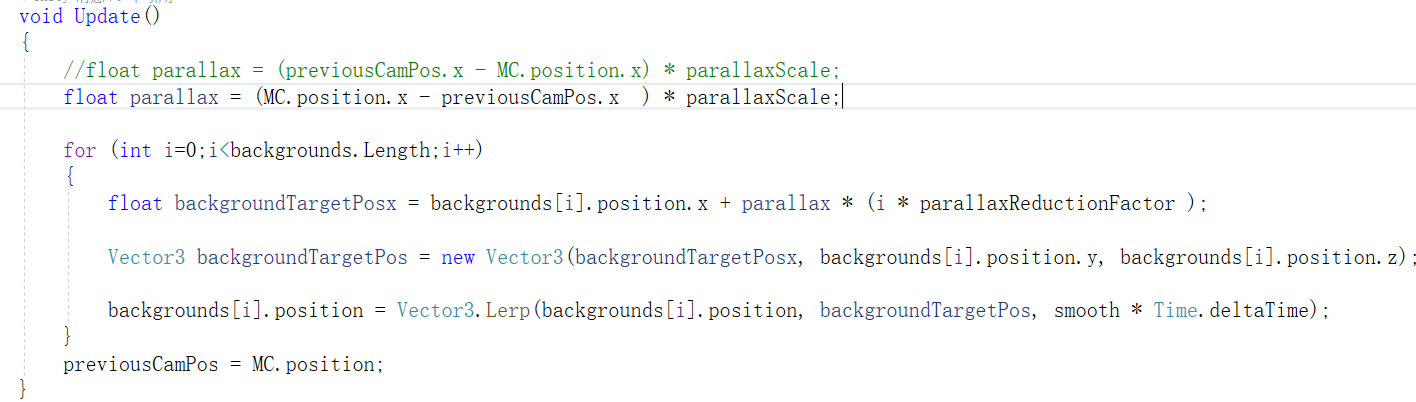


图64 具体代码

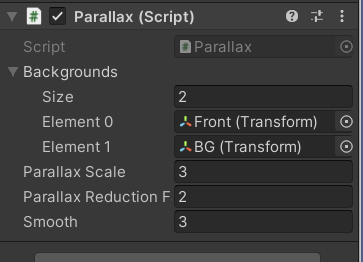


图65 传入参数

这里的parallax视差就是每一帧的相机移动的距离。

当i等于0的时候（即最前景），物体移动的距离和相机移动的距离一样（绝对位置不动）。后面的层次根据一个参数factor逐渐速度加快（相对速度越来越慢）

2、问题及解决方法

说来很奇怪，相机移动的距离和前景背景的距离有出入。换句话说，如果按照上面的参数，我第一层的背景移动的速度本来应该比相机还要快，但是实际上它和smooth也有关，因为lerp是一个插值函数…

Mathf.Lerp(float a, float b, float t)就相当于

return=a+(b-a)t

所以在调节参数的时候花了一些时间。期间还出现了相机回不到正常位置，以及拍摄到了边界之外的东西等等情况…

2.1.18 金蛋关卡

金蛋和非金蛋关卡的区别并不大，只对game manager进行了一些修改：添加了GoldenEgg布尔变量来修改胜利条件的判断，并且加上了eggs的一个list来获取到金蛋的信息。



图66 金蛋关卡的GM

金蛋的设计和猪区别不大，但是更简单，不需要受伤，只需要死亡的maxspeed即可。

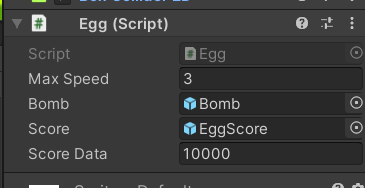


图67 Egg

最后的胜负判断也要有变化



图68 金蛋关卡的胜负判断

2.1.19 大红鸟

大红鸟除了技能不同之外，我还对它的质量进行了修改，更符合物理现实。

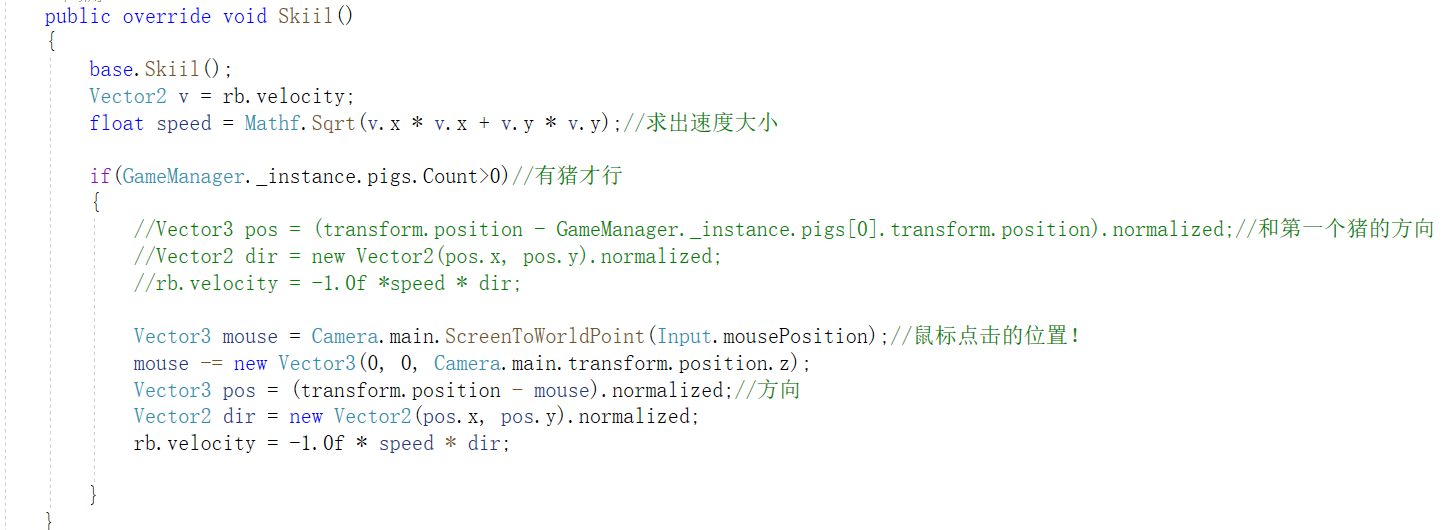


图69 大红鸟的技能

其中获取鼠标位置的方法和一开始的小鸟拉弓的时候是一样的。

2.2 其他未归类的bug及解决办法

2.2.1 分辨率

一方面，canvas需要跟着屏幕的大小来变化。

另一方面，加载页面的图片为了避免压缩变得难看，也需要选择合适的分辨率



图70 载入界面

但是如果分辨率选择的不对的话，或出现各种问题：



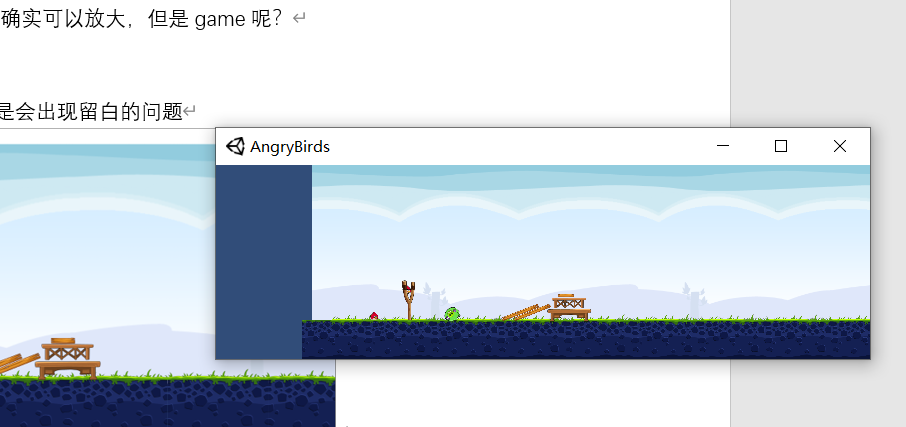


图71 各种问题

所以最后决定的是固定比例，并且不可以手动缩放，并且禁止全屏和最大化。

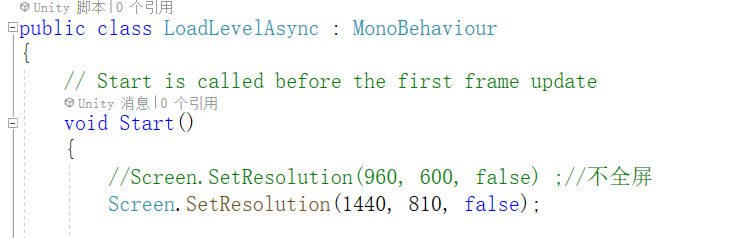


图72 手动固定分辨率

2.3 优化设计

2.3.1 不同的鸟/建筑显示的分数不一样

主要是根据颜色不同来显示

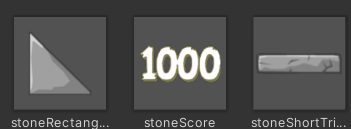
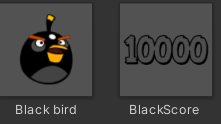


图73 分数的显示

2.3.2 退出游戏

在开始界面点击返回按钮就可以退出游戏

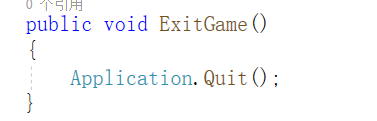


图74 退出游戏

1. 参考资料

【1】教程：http://www.sikiedu.com/cloud/search?q=愤怒的小鸟&type=course

【2】不同的update类型：<https://blog.csdn.net/qq_42351033/article/details/83047657>

【3】 网页flash版《愤怒的小鸟》：<http://tools.jb51.net/games/angry_birds>

【4】计分跳动：<http://www.tuansbook.com/newsinfo.php?id=290>

【5】协程：<https://blog.csdn.net/qq_15020543/article/details/82701551>

【6】视差滚动：https://baike.baidu.com/item/视差滚动/2290867?fr=aladdin

<https://indienova.com/indie-game-development/parallax-scrolling-in-2d-platformer/>

【7】插值函数：<https://zhidao.baidu.com/question/925248738234706379.html>

【8】武器拖尾：https://blog.csdn.net/qq\_27489007/article/details/89881515