# 期末文档归档

系统开发说明文件（可以在已有的产品设计方案和UI设计的基础上完善）的主要内容:

**一、2D横版跑酷游戏方案设计**

**项目实施可行性报告**

**行业市场分析：**

近年来，由于4G网络和智能手机的普及，手机游戏产业的发展成为了显而易见的趋势，并且在各年龄段群体中都体现出用户暴增的现象，这源于移动应用程序的多样性及其强大的功能性，加上使用手机上网的用户数量逐年高速增长的兜底，未来几年内手游行业的规模将继续扩大，形势呈现一片利好。

**竞争对手或同类产品分析：**

**《天天酷跑》**

天天酷跑是由腾讯旗下的的天美艺游工作室制作推出的一款移动端跑酷类游戏，它作为中国第一款月收入超1亿的手游，成功的光芒可谓是无法掩盖，是业界标杆的现象级2D横版跑酷类游戏。

除了源于微信和QQ的用户基础带来的平台效益之外，天美艺游工作室也在产品开发上下了不少功夫，精良的美术风格和流畅的画面表现，以及恰到好处的音乐音效，以及最关键的游戏性，无一不是用户留存的宝贵要素。

加上游戏内外部的运营和策划，与腾讯旗下其他产品的联动等，为它的盈利提供了保障，从而进一步改进和更新游戏内容并保证用户的消费和游戏体验。



**《神庙逃亡》**

虽然这是一款3D的，纵向的逃生类跑酷手游，但神庙逃亡仍有许多值得我们去借鉴和学习的地方。它的背景设定为一个冒险家在古老的庙宇中寻宝，却碰上怪兽的追赶，而玩家需要通过转弯、跳跃和滑铲这三种动作来帮助主角逃脱。

首先它有着引人入胜的主题背景和美术风格，古神庙和逃亡，静态与动态的微妙结合让它本身就已披上的神秘色彩更显厚重，第一时间能够抓住玩家的眼球。

其次它的操作简单，基本上失败几次就能完全领会游戏的玩法，其实就是简单的考验反应力的滑屏游戏，上滑是跳跃，左右滑动是转弯，下滑是滑铲。

然后就是它富有特点的转场设计，转弯的时候，画面会绕Z轴呈现3D立体的画面旋转，很有冲击力，参考到2D设计里，可以尝试以画面中心为圆心的整体旋转来达到类似的效果。



**自身条件分析：**

我们小组的打算是制作一款以猫和老鼠动画为背景的2D横版跑酷类游戏，以左手操控左右按钮，右手操控上下按钮为基本操作方式的跑酷玩法是基本玩法，以此可以延伸出几种不同的模式来丰富游戏性。市面上虽有以猫和老鼠为题材的多人竞技追逐类横版手游，但我们预期所采取的美术风格和玩法上与其相差甚远，因而可以避开玩法上的正面竞争。并且要是利用好我们游戏预期做到的小巧精致的特性，甚至于加入开放性的玩家编辑工具，不仅可以从另一个方向优化我们的企划思路和发展规划，还能一定程度地让玩家加入到游戏内容的开发建设中。

**产品定位及目标**

**用户群分析：**

首先最外层的大用户群可以囊括所有在手机上使用互联网的用户，这是休闲类手游都能达成的一个共识，如果能进一步细化，在更深层次上培养一批老玩家，资深玩家以至于核心玩家，从而我们便可以搜集和统计分析玩家们在游戏中的种种举措来挖掘本产品留住玩家的关键因素，除此之外也能通过构建社区以及建立官方渠道来和玩家沟通并回收意见，来明确游戏改进的大致方向。

猫和老鼠的题材首先可以吸引到年轻用户的关注，而这种轻松明快的画风同样适合多种多样的场合和不同年龄段的用户，属于非常亲民的娱乐应用类型。其次玩法简单，目标明确，但也能加入多人竞技性以及巧妙的关卡策划来给某些硬核玩家提供更具深度的游戏需求，同时我们也想提供一个编辑工具让玩家可以自己设计关卡、场景、角色甚至是情节，丰富游戏内容的同时，使得游戏生态的构建存在可能。如此，在这种快餐化与深度化交织的游戏玩家圈内，我们的游戏便能结合社交性，休闲性，竞技性和开放性，实现一个游戏，多面发展的设想。

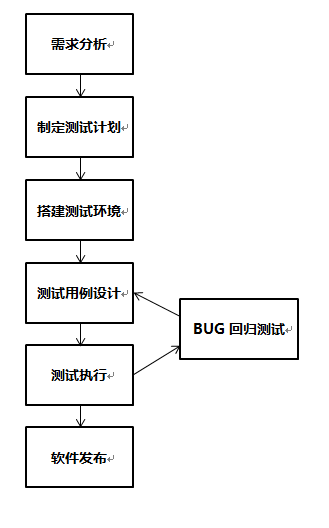
**产品内容总策划（应用流程规划，设计与测试规范，开发日程表）**

app主要内容

该App为2D横板类游戏，暂定为3种模式（开发期间可能会进行更改）。跑酷模式：玩家操纵Jerry，使Jerry躲避Tom的追捕，直到Jerry被Tom抓住，跑酷期间Jerry可以拾取道具用于加速，回血，短时无敌等。参考传统跑酷游戏如地铁跑酷，神庙逃亡等。生存模式：n只老鼠被关在房间内，需要在规定时间内躲避猫的追捕，老鼠可以拾取奶酪，一个奶酪可以加速一次。猫也可以拾取相应道具用于抓捕老鼠。竞赛模式：n只猫与n只老鼠分成两个阵营，比赛为积分制，率先到达终点的积分较多，最后总分高的阵营胜利。

**1、应用流程规划**

需求分析—>制定测试计划—>搭建测试环境—>测试用例设计—>测试执行—>BUG回归测试—>测试总结—>软件发布



**2、设计与测试规范**

该App使用安卓专用长度单位dp和安卓专用字体单位sp，尺寸稿暂定为5.0英寸即1080 x 1920 PX 来做安卓设计稿尺寸。图标对以下尺寸都支持512x512px、192x192px、144x144px、96x96px、72x72px、48x48px，且分圆角图标和直角图标两种。

该App会对市面上大部分安卓手机进行真机测试，如华为，小米，三星，VIVO，OPPO等品牌的手机。测试直到App在这些品牌部分手机上都能正常运行。

**3、开发日程表**

前期：设计人物动作形象，背景，游戏界面。完成账号注册登录等功能。

中期：实现游戏的三种模式。

后期：加入装备和装备强化增幅等系统以达到盈利的目的。

**技术解决方案**

开发环境：Android Studio

开发语言：Java

人物设计：Photoshop

**五、 推广方案**

在游戏的初期推广上我们的想法是尽可能地降低宣发成本，集中精力把游戏骨架搭好，首先要还原一个最真实的游戏产品，然后是控制圈子的规模，培育口碑，等游戏完成度够高时再做进一步推广，并在积累起一定规模的用户之后采取一定的盈利措施回收游戏成本。

初期推广可以考虑多种媒体渠道进行宣传，如微博，知乎，贴吧，论坛等网络公共社区，以及微信、qq等即时通讯应用上的圈子。可以考虑通过游戏来获取一些广告赞助，最好是和游戏主题有所关联的广告，如猫和老鼠官方授权的周边贩售，以及举办拥有相关动画播放版权的影视播放软件的会员抽奖活动等，前期不推出任何收费渠道，限额开放多人测试，以缓解服务器租用负担，尽可能地积累起初期用户群体。

待用户群体的规模以及游戏知名度都有了一定扩大和提高之后，便可以考虑盈利，与此同时想办法登录多个平台并依赖它们的用户基数来实现本产品更广泛的传播，若取得明显成效，便可以细化运营方案，在此基础上根据需要再进行下一步推广方案的设计。

**六、 运营规划书**

**1、 盈利模式：**现阶段的盈利方案主要是依靠用户充值购买或合成装备并获得强化增幅。强化增幅在非竞技模式下不设上限，用户可以根据自己的经济实力进行相应的充值，而竞技模式中使用的装备，可以在游玩一段时间游戏并完成相应任务之后领取材料并合成。

**2、 推广前：**对本校部分学生开放并填写游戏问卷，改进代码，优化性能，尽量减少bug；该期主要是对App进行内部测试，以免App投入市场后出现重大bug；完善游戏的全局设计，首要任务是把基础的跑酷玩法操作做好。

**3、 推广期：**App已经能稳定运行，开始进行低成本大面积的推广。如去投入各种AppStore，投入微博等网络公共社区，积累一定的用户资源之后想办法盈利，在此基础上加大推广力度。

**4、 盈利期：**开始提供VIP等级功能，上线首充活动等。有了用户基础后，前期可利用流量拉取广告以保证资金供应的回暖，中期要在游戏内容的更新上推陈出新，以游戏内容培植玩家群体的原则继续运营，目的在于建立起游戏的差异性竞争能力，并建立更多不同的收费方式和渠道来实现盈利，后期若能在业内站稳脚跟，有一定的地位，可以考虑接受其他公司的收购或是来自多方的投资来让产品焕发出新的活力。

**二、产品实现方案**

系统的主要功能

UI界面设计

关键技术和技术难点

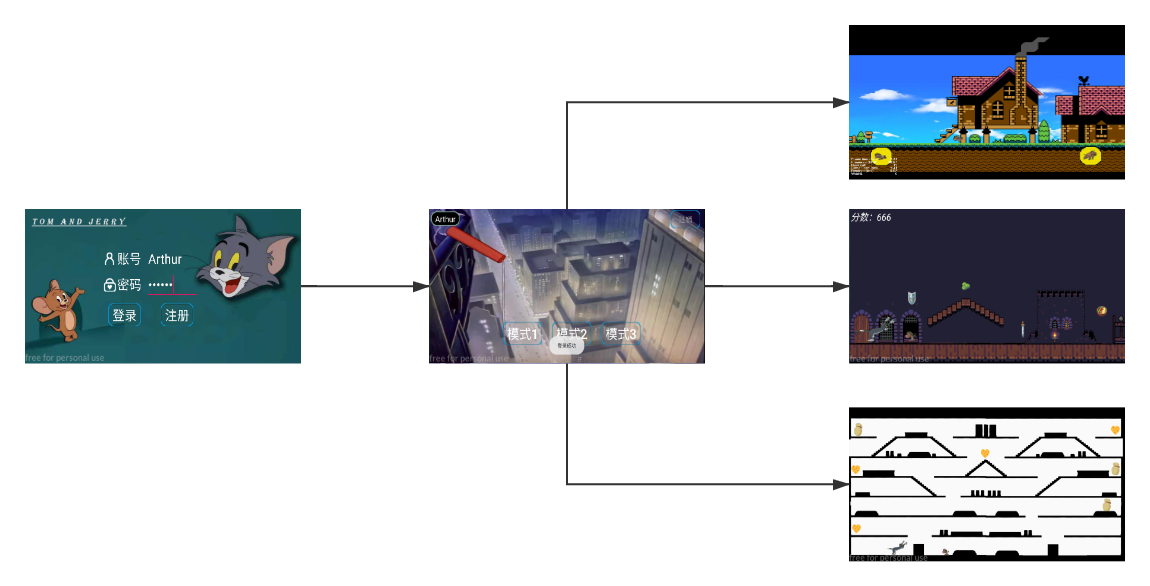
用户体验记录和分析

已完成的改进和存在的问题

**系统的主要功能：**

该app是一个以动画《猫和老鼠》为背景的跑酷游戏，目的是让快节奏生 活的人们能慢下来享受游戏的乐趣

**UI界面设计：**



**关键技术和技术难点：**

关键技术：

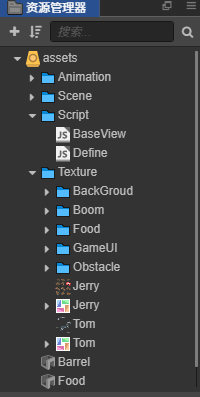
sqlite数据库存储用户信息，Intent进行页面跳转

游戏开发引擎：cocos creator

游戏脚本驱动语言：javascript

美术素材来源：网络及原创

资源使用量：

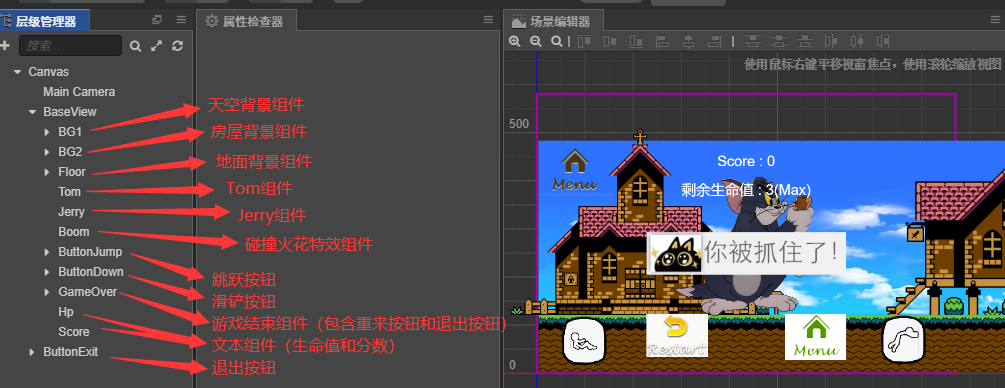


技术难点：

1. javascript作为一门脚本语言，结合游戏开发引擎Cocos Creator，要对场景中的各个组件以键值对字典的形式在脚本文件中进行注册，并且要在Cocos Creator中把已注册的组件定位到层级管理器中相应的节点。
2. 注册时为node类型的组件，在单独播放动画时要转化为animation类型，在获取坐标或执行播放移动行为序列等方法时要用.node属性引用该组件附加到的节点。
3. 角色组件和场景中其他组件的交互要在脚本文件的update函数中用判断语句检测每帧的状况，比如Jerry是否碰撞到垃圾桶，需用If语句判断两者的坐标距离是否小于一个足够且不会过于小的值。
4. 在游戏结束后必须停下画面，但不能使弹出的选项按钮失效，于是才用了stopAllActions()方法使所有组件的属性不再变化，原类型为animation的组件再执行stop()方法停止播放动画，达到了类似于停止画面渲染的效果，但不会影响后续操作。

代码设计：

使用资源



BaseView.js

1. cc.Class({
2. **extends**: cc.Component,
4. properties: { //注册
5. // 这个属性引用了Barrel预制资源
6. BarrelPrefab: {
7. **default**: **null**,
8. type: cc.Prefab
9. },
10. // 这个属性引用了Food预制资源
11. FoodPrefab: {
12. **default**: **null**,
13. type: cc.Prefab
14. },
15. m\_Tom:cc.Node,//字典，m\_Tom是键，cc.Node是值（写上m\_Tom的类型）
16. m\_Jerry:cc.Node,
17. m\_Boom:cc.Node,//碰撞效果
18. m\_BtDown:cc.Button,
19. m\_BtJump:cc.Button,
20. m\_Back1:[cc.Node],//定义为数组，可以多放Node
21. m\_Back2:[cc.Node],
22. m\_Floor:[cc.Node],
23. newBarrel:cc.Node,
24. newFood:cc.Node,
25. closerX:0,
27. //游戏结束画面组件
28. m\_BtRetry:cc.Button,
29. m\_BtExit:cc.Button,
30. GameOver:cc.Node,
32. //数值组件
33. scoreDisplay: {
34. **default**: **null**,
35. type: cc.Label
36. },
37. hpDisplay: {
38. **default**: **null**,
39. type: cc.Label
40. }
41. },
43. // LIFE-CYCLE CALLBACKS:
45. // onLoad () {},
47. start () {
48. cc.log('正常运行');
49. **this**.score = 0;
50. **this**.hp = 3;
51. **this**.m\_Tom = **this**.m\_Tom.getComponent(cc.Animation);
52. **this**.m\_Jerry = **this**.m\_Jerry.getComponent(cc.Animation);
53. **this**.m\_Boom = **this**.m\_Boom.getComponent(cc.Animation);
54. //暂停游戏结束界面节点上注册的所有节点系统事件，节点系统事件包含触摸和鼠标事件
55. **this**.m\_BtRetry.node.pauseSystemEvents();
56. **this**.m\_BtExit.node.pauseSystemEvents();
57. //隐藏游戏结束界面
58. **this**.GameOver.active = **false**;
59. //Tom和Jerry初始进行奔跑
60. **this**.m\_Tom.play('TomRun');
61. **this**.myJerryPlay('Run');
62. //Tom靠近时的距离
63. **this**.closerX = (**this**.m\_Jerry.node.x-**this**.m\_Tom.node.x)/3;
64. //滑铲动作
65. **this**.m\_BtDown.node.on(cc.Node.EventType.TOUCH\_START,**this**.touchStart,**this**);
66. **this**.m\_BtDown.node.on(cc.Node.EventType.TOUCH\_END,**this**.touchEnd,**this**);
67. **this**.m\_BtDown.node.on(cc.Node.EventType.TOUCH\_CANCEL,**this**.touchEnd,**this**);
68. //跳跃动作
69. **this**.m\_BtJump.node.on(cc.Node.EventType.TOUCH\_START,**this**.uptouchStart,**this**);
70. **this**.m\_BtJump.node.on(cc.Node.EventType.TOUCH\_END,**this**.uptouchEnd,**this**);
71. **this**.m\_BtJump.node.on(cc.Node.EventType.TOUCH\_CANCEL,**this**.uptouchEnd,**this**);
72. // 生成一个新的Barrel
73. **this**.newBarrel = **this**.spawnNewBarrel();
74. **this**.newFood = **this**.spawnNewFood();

77. //定义两背景位置再播放动作
78. **for**(**var** i = 0; i < **this**.m\_Back1.length;i++){
79. window.width = **this**.m\_Back1[i].width;//定义第一层（最里层）背景宽度
80. **this**.m\_Back1[i].setPosition(i\*(width-6),0);
82. **var** moveImgLeft = cc.moveTo((i+1)\*SkyBackMoveTime,-(width-6),0);
83. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.backMoveEnd,**this**,SkyBackMoveTime));
84. **this**.m\_Back1[i].runAction(seq);
85. }
86. **for**(**var** i = 0; i < **this**.m\_Back2.length;i++){
87. //window.width = this.m\_Back2[i].width;//定义两倍屏幕宽度的第二层背景宽度
88. **this**.m\_Back2[i].setPosition(i\*1716,0);
90. **var** moveImgLeft = cc.moveTo((i+1)\*HouseBackMoveTime,-(1716-6),0);
91. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.backMoveEnd,**this**,HouseBackMoveTime));
92. **this**.m\_Back2[i].runAction(seq);
93. }
94. **for**(**var** i = 0; i < **this**.m\_Floor.length;i++){
95. //window.width = this.m\_Floor[i].width;//定义第三层背景宽度
96. **this**.m\_Floor[i].setPosition(i\*858,0);
98. **var** moveImgLeft = cc.moveTo((i+1)\*FloorBackMoveTime,-(858-28),0);
99. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.backMoveEnd,**this**,FloorBackMoveTime));
100. **this**.m\_Floor[i].runAction(seq);
101. }
102. },


106. //一、背景和人物动画
107. //背景图左移到底后再置于最前
108. backMoveEnd:**function**(target,BackTime){
109. **var** width = target.width;
110. target.setPosition((width-6),0);
112. **var** moveImgLeft = cc.moveTo(BackTime\*2,-(width-6),0);
113. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.backMoveEnd,**this**,BackTime));
114. target.runAction(seq);
115. },

118. callBackDownOver:**function**(){
119. **this**.myJerryPlay('Run');
120. },
122. touchStart:**function**(){//按下（滑铲键）
123. **if**(**this**.m\_Jerry.currentClip.name == 'Jump' ){//判断落地才能继续
124. **return**;
125. }
126. **this**.myJerryPlay('Down');
127. },
128. touchEnd:**function**(){//松开（滑铲键）
129. **if**(**this**.m\_Jerry.currentClip.name == 'Jump' ){//判断落地才能继续
130. **return**;
131. }
132. **this**.myJerryPlay('Run');
133. },
134. uptouchStart:**function**(){//按下（跳跃键）
135. **if**(**this**.m\_Jerry.currentClip.name == 'Jump' ){//判断落地才能继续
136. **return**;
137. }
138. **this**.onAnimationChange('Jump');
139. },
140. uptouchEnd:**function**(){//松开（跳跃键）
141. **if**(**this**.m\_Jerry.currentClip.name == 'Jump' ){//判断落地才能继续
142. **return**;
143. }
144. **this**.myJerryPlay('Run');
145. },
147. onAnimationChange: **function** (data) {
148. **if**(**this**.m\_Jerry.currentClip.name == 'Jump' ){//判断落地才能继续
149. **return**;
150. }
151. **if**(**this**.m\_Jerry.currentClip.name == 'Down' ){//判断落地才能继续
152. **if** (data == 'Jump') {
153. **var** moveUp = cc.moveTo(0.4, -101.968, -51.87).easing(cc.easeCubicActionOut());
154. **var** moveDown = cc.moveTo(0.4, -101.968, -151.87).easing(cc.easeCubicActionOut());
155. **var** callBack = cc.callFunc(**this**.callBackDownOver.bind(**this**), **this**.m\_Jerry.node, **this**);
156. **var** seq = cc.sequence(moveUp, moveDown, callBack);
157. **this**.m\_Jerry.node.runAction(seq);
158. cc.log('Jerry的位置在'+**this**.m\_Jerry.node.x+','+**this**.m\_Jerry.node.y);
159. }
160. }
161. **if** (data == 'Jump') {
162. **var** moveUp = cc.moveTo(0.3, -101.968, -81.87).easing(cc.easeCubicActionOut());
163. **var** moveDown = cc.moveTo(0.3, -101.968, -151.87).easing(cc.easeCubicActionOut());
164. **var** callBack = cc.callFunc(**this**.callBackDownOver.bind(**this**), **this**.m\_Jerry.node, **this**);
165. **var** seq = cc.sequence(moveUp, moveDown, callBack);
166. **this**.m\_Jerry.node.runAction(seq);
167. cc.log('Jerry的位置在'+**this**.m\_Jerry.node.x+','+**this**.m\_Jerry.node.y);
168. }
169. **this**.myJerryPlay(data);
170. },
171. myJerryPlay:**function**(playName)//滑铲时移动位置
172. {
173. **if**( playName == 'Down' )
174. {
175. **this**.m\_Jerry.node.setPosition(-101.968,-154.592);
176. }
177. **else** **if**( playName == 'Run')
178. {
179. **this**.m\_Jerry.node.setPosition(-101.968,-151.87);
180. }
181. **this**.m\_Jerry.play(playName);
182. },




188. //二、障碍物和食物
189. //产生新Barrel
190. spawnNewBarrel: **function**() {
191. // 使用给定的模板在场景中生成一个新节点
192. **var** newBarrel = cc.instantiate(**this**.BarrelPrefab);
193. // 将新增的节点添加到 BaseView 节点下面
194. **this**.node.addChild(newBarrel);
196. // 为Barrel设置一个初始位置
197. newBarrel.setPosition(**this**.getNewBarrelPosition());
198. // 设定Barrel动画
199. **var** moveImgLeft = cc.moveTo(BarrelBackMoveTime,-(858-14),-152.131);
200. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.barrelBackMoveEnd,**this**,BarrelBackMoveTime));
201. newBarrel.runAction(seq);
202. **return** newBarrel;
203. },
204. //获取Barrel位置
205. getNewBarrelPosition: **function** () {
206. **var** randX = 0;
207. // 根据地平面位置，得到一个Barrel的 y 坐标
208. **var** randY = -152.131;
209. // 根据屏幕宽度，得到一个Barrel x 坐标
210. **var** maxX = 858;
211. randX = maxX + 858;
212. // 返回Barrel坐标
213. **return** cc.v2(randX, randY);
214. },
215. //Barrel左移到底后再置于最前
216. barrelBackMoveEnd:**function**(target,BackTime){
217. target.setPosition(**this**.getNewBarrelPosition());
219. **var** moveImgLeft = cc.moveTo(BackTime,-(858-14),-152.131);
220. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.barrelBackMoveEnd,**this**,BackTime));
221. target.runAction(seq);
222. },
224. //产生新Food
225. spawnNewFood: **function**() {
226. // 使用给定的模板在场景中生成一个新节点
227. **var** newFood = cc.instantiate(**this**.FoodPrefab);
228. // 将新增的节点添加到 BaseView 节点下面
229. **this**.node.addChild(newFood);
231. // 为Food设置一个初始位置
232. newFood.setPosition(**this**.getNewFoodPosition());
233. // 设定Food动画
234. **var** moveImgLeft = cc.moveTo(BarrelBackMoveTime + 2,-(858-14),-81.87);
235. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.FoodBackMoveEnd,**this**,BarrelBackMoveTime + 2));
236. newFood.runAction(seq);
237. **return** newFood;
238. },
239. //获取Food位置
240. getNewFoodPosition: **function** () {
241. **var** randX = 0;
242. // 根据地平面位置，得到一个Food的 y 坐标
243. **var** randY = -81.87;
244. // 根据屏幕宽度，得到一个Food x 坐标
245. **var** maxX = 1058;
246. randX = maxX + 1058;
247. // 返回Food坐标
248. **return** cc.v2(randX, randY);
249. },
250. //Food左移到底后再置于最前
251. FoodBackMoveEnd:**function**(target,BackTime){
252. target.setPosition(**this**.getNewFoodPosition());
254. **var** moveImgLeft = cc.moveTo(BackTime,-(858-14),-81.87);
255. **var** seq = cc.sequence(moveImgLeft,cc.callFunc(**this**.FoodBackMoveEnd,**this**,BackTime));
256. target.runAction(seq);
257. },
259. //2.1碰撞检测
260. getJerryDistance: **function** () {
261. // 根据 Jerry 节点位置判断距离
262. **var** JerryPos = **this**.m\_Jerry.node.getPosition();
263. // 根据两点位置计算两点之间距离
264. **var** dist = **this**.newBarrel.position.sub(JerryPos).mag();
265. **return** dist;
266. },
267. ontouched:**function**() {
268. // 播放碰撞效果
269. **this**.m\_Boom.play('Boom');
270. // Tom靠近，生命值减少
271. **var** getClose = cc.moveTo(0.6,**this**.m\_Tom.node.x+**this**.closerX,**this**.m\_Tom.node.y);
272. **this**.m\_Tom.node.runAction(getClose);
273. **this**.loseHp();
274. // 然后销毁当前Barrel节点
275. **this**.newBarrel.destroy();
276. // 当Barrel被碰撞时，生成一个新的Barrel
277. **this**.newBarrel = **this**.spawnNewBarrel();
278. },
280. //2.2被捕后的方法
281. getCaught:**function**(){
282. // 停止所有动画的播放
283. **this**.m\_Tom.node.stopAllActions();
284. **this**.m\_Tom.stop();
285. **this**.m\_Jerry.node.stopAllActions();
286. **this**.m\_Jerry.stop();
287. **this**.m\_Boom.node.stopAllActions();
288. **this**.newBarrel.stopAllActions();
289. **this**.newFood.stopAllActions();
290. **for**(**var** i = 0; i < **this**.m\_Back1.length;i++){
291. **this**.m\_Back1[i].stopAllActions();
292. }
293. **for**(**var** i = 0; i < **this**.m\_Back2.length;i++){
294. **this**.m\_Back2[i].stopAllActions();
295. }
296. **for**(**var** i = 0; i < **this**.m\_Floor.length;i++){
297. **this**.m\_Floor[i].stopAllActions();
298. }
299. // 弹出结束画面
300. // 显示游戏结束界面
301. **this**.GameOver.active = **true**;
302. // 恢复注册
303. **this**.m\_BtRetry.node.resumeSystemEvents();
304. **this**.m\_BtExit.node.resumeSystemEvents();
305. },
306. //游戏结束后的两个按钮
307. onSceneRetry:**function** (data) {
308. cc.log('重试');
309. cc.director.loadScene('GameScene');
310. **return**;
311. },
312. onSceneExit:**function** (data) {
313. cc.log('结束');
314. cc.game.end();
315. cc.director.end();
316. **return**;
317. },



322. //三、得分与落命
323. gainScore:**function** () {
324. **this**.score += 1;
325. // 更新 scoreDisplay Label 的文字
326. **this**.scoreDisplay.string = 'Score : ' + **this**.score;
327. },
328. loseHp:**function** () {
329. **this**.hp -= 1;
330. // 更新 hpDisplay Label 的文字
331. **this**.hpDisplay.string = '剩余生命值 : ' + **this**.hp;
332. },
333. //3.1拾取行为
334. getFood:**function** () {
335. // 若生命值小于3,Tom远离,生命值+1
336. **if**(**this**.hp < 3){
337. **var** getFarther = cc.moveTo(0.8,**this**.m\_Tom.node.x-**this**.closerX,**this**.m\_Tom.node.y);
338. **this**.m\_Tom.node.runAction(getFarther);
339. **this**.hp += 1;
340. // 更新 hpDisplay Label 的文字
341. **if**(**this**.hp == 3)
342. **this**.hpDisplay.string = '剩余生命值 : ' + **this**.hp + '(Max)';
343. **else**
344. **this**.hpDisplay.string = '剩余生命值 : ' + **this**.hp;
345. }
346. // 分数+1
347. **this**.gainScore();
348. // 销毁食物
349. **this**.newFood.destroy();
350. // 生成新食物
351. **this**.newFood = **this**.spawnNewFood();
352. },
353. getJerryFoodDistance: **function** () {
354. // 根据 Jerry 节点位置判断距离
355. **var** JerryPos = **this**.m\_Jerry.node.getPosition();
356. // 根据两点位置计算两点之间距离
357. **var** dist = **this**.newFood.position.sub(JerryPos).mag();
358. **return** dist;
359. },
361. update: **function** (dt) {
362. // 每帧判断Barrel和主角之间的距离是否小于碰撞距离
363. **if** (**this**.getJerryDistance() < 10) {
364. // 调用碰撞行为
365. **this**.ontouched();
366. cc.log('碰撞成功');
367. **return**;
368. }
369. // 每帧判断Food和主角之间的距离是否小于拾取距离
370. **if** (**this**.getJerryFoodDistance() < 15) {
371. // 调用拾取行为
372. **this**.getFood();
373. cc.log('拾取成功');
374. **return**;
375. }
376. // 每帧判断是否跨越成功
377. **if** (Math.abs(**this**.m\_Jerry.node.x - **this**.newBarrel.x) < 2.3) {
378. // 调用碰撞行为
379. **this**.gainScore();
380. cc.log('跨越成功');
381. **return**;
382. }
383. // 每帧判断Tom是否接触到Jerry
384. **if** (**this**.m\_Jerry.node.x-**this**.m\_Tom.node.x < 5) {
385. // 游戏结束
386. **this**.getCaught();
387. **return**;
388. }
389. },
391. });

Define.js

1. **var** SkyBackMoveTime = 30;
2. **var** HouseBackMoveTime = 10;
3. **var** FloorBackMoveTime = 3;
4. **var** BarrelBackMoveTime = 9;

**用户体验记录和分析：**

反馈问题1：人物动作不连贯

解决方法1：通过录屏截取视频，再逐帧截取人物动作

反馈问题2：图片背景衔接不好

解决方法2：将背景图片放入ps处理，使背景图片左端和右端无缝衔接

已完成的改进和存在的问题：

人物动作连贯，背景图片衔接流畅

1. **测试大纲和测试报告**

UI测试:采用了python里面的airtest库，进行ui测试。airtest是一个采用图像识别功能进行ai测试的库。







**四、产品安装和使用说明**

将apk安装到手机后，打开直接开始进行游戏。







