**《Heroes Age 3D》**

战斗系统设计说明书

Heroes Age 3D项目组

**文档历史**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 日期 | 版本 | 作者 | 批准人 | 描述 |
| 2015-12-28 | V1.0 | 刘锐 ray@ucool.com |  | 第一版 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

《Heroes Age 3D》 1

1. 引言 7

1.1. 目的 7

1.2. 文档约定 7

1.3. 预期读者和阅读建议 7

1.3.1. 预期读者 7

1.3.2. 阅读建议 7

1.4. 文档的范围 7

2. 系统框架 8

2.1. 整体结构 8

2.2. 层次划分 8

2.2.1. 基础逻辑层 8

2.2.2. 组件连接层（Joint） 9

2.2.3. 接口层（Interface） 10

2.2.4. 游戏对象层（UnityObject） 11

2.2.5. AI扩展层 11

2.3. 模块划分 11

2.3.1. 场景管理(Gsm) 11

2.3.2. 主逻辑(Logic) 12

2.3.3. 元素(Element) 12

2.3.4. 技能 13

2.3.5. 印记(Mark) 13

2.3.6. 陷阱(Trap) 13

3. 基础规范 14

3.1. 战斗场景 14

3.2. 角色 14

3.3. 动作 14

3.4. 帧事件 14

3.4.1. PlayEffect（播放特效） 15

3.4.2. PlaySound（播放声音） 15

3.4.3. CamShake（摄像机震动） 15

3.4.4. AttackSpot（攻击帧） 16

3.4.5. DoAction（执行动作） 16

3.4.6. 执行技能Talent 19

4. 系统流程 20

4.1. 战斗初始化流程 20

4.2. 逻辑更新流程 20

4.2.1. 战斗逻辑更新 20

4.2.2. 角色更新流程 20

4.2.3. 技能更新流程 20

4.2.4. 效果更新流程 20

4.3. 技能释放流程 20

5. 模块设计 22

5.1. 战斗 22

5.1.1. 战斗场景 22

5.1.2. 战斗模式 22

5.1.3. 战斗状态 22

5.2. 角色 22

5.2.1. 基本属性 22

5.2.2. 模型定义 22

5.2.3. 角色状态 22

5.3. 技能 22

5.4. 技能组 22

5.5. Ability 22

5.6. 效果 22

5.7. 印记 22

5.8. 帧事件 22

5.8.1. PlayEffect（播放特效） 23

5.8.2. PlaySound（播放声音） 23

5.8.3. CamShake（摄像机震动） 23

5.8.4. AttackSpot（攻击帧） 23

5.8.5. DoAction（执行动作） 23

5.8.6. 执行技能Talent 24

# 引言

## 目的

1. 描述战斗系统的结构、流程
2. 为战斗系统的详细设计及开发提供指导。

## 文档约定

## 预期读者和阅读建议

### 预期读者

战斗及技能开发人员

### 阅读建议

相关人员应尽量仔细的阅读整个文档。

## 文档的范围

本文档涉及整个项目战斗相关的技术规范

# 系统框架

## 整体结构

为保证战斗系统的扩展性和可移植性，对战斗系统进行分层设计，通过接口层对逻辑和表现进行隔离，保证战斗系统本身的独立性。



## 层次划分

战斗系统可划分为5个主要层次，基础逻辑层、组件连接层、接口层、游戏对象层、AI扩展层。

### 基础逻辑层

基础逻辑层包含战斗系统的主要实现，其中主要包括战斗主逻辑模块（Logic）、角色模块（Role）、技能模块（Talent）、抛射器模块(Effect)。

基础逻辑层实现整个战斗的核心逻辑。

### 组件连接层（Joint）

组件连接层用于连接战斗中的逻辑对象与场景中的实体对象之间关联，并实现对游戏对象的精确控制。

为了性能考虑，逻辑层的主循环是基于场景管理器（Gsm）的FixedUpdate事件作为入口，以8fps的逻辑帧率进行循环，这样可以有效降低CPU的占用。而表现层需要以渲染帧率（15-30fps）进行循环，才可保证角色行为的流畅，而组件连接层（Joint）的主要目的就是为了对游戏对象在逻辑帧与渲染帧之间进行同步。

如图:



Role为逻辑层对象，表示一个逻辑角色，RoleJoint表示一个角色连接，RoleController对应场景中的一个实际可见游戏角色。战斗系统在主循环（OnEnterFrame）中更新Role的位置和速度，而RoleJoint则根据Role中的速度数值，计算出当前渲染帧角色所在的实际位置，并通过RoleController设置给游戏对象（GameObject）。

### 接口层（Interface）

接口层的设计目的为彻底隔离战斗逻辑与Unity对象之间的关联，是战斗逻辑本身不对Unity产生直接依赖。

战斗系统主要接口有3类接口：连接接口、控制器接口、场景接口。

#### Joint接口（IJoint）

IJoint用于隔离Joint层，确保战斗逻辑不对Joint产生直接依赖，并实现对Joint的控制及约束。



#### 控制器接口（IController）

IController用于隔离角色控制器、特效控制器，确保战斗逻辑不对游戏对象产生直接依赖。

IController又抽象出 IRoleController 和 IEffectController,实现对角色控制器(RoleController)和特效控制器（EffectController）的分别隔离和约束。



#### 场景接口（IBattleScene）

IBattleScene用于隔离场景管理器，确保战斗逻辑不对Unity对象产生直接依赖，并实现对游戏场景的一些基础控制约束。



### 游戏对象层（UnityObject）

游戏对象层实现对Unity对象的直接控制，并为战斗逻辑提供必要的接口。一切对可见游戏对象的控制均在此层实现，包括角色、特效、漂浮文字、血条更新等等。

游戏对象层主要包含场景管理器(Gsm)和游戏对象控制器：角色控制器RoleController、特效控制器EffectController。

### AI扩展层

AI扩展层用于对一些非常特殊又无法通过配置解决的技能行为进行扩展，通过对 Role 或 Talent 进行重载，来实现不同的技能行为。

## 模块划分

战斗系统可分为以下主要模块：场景管理、战斗主逻辑、元素、技能。

### 场景管理(Gsm)

场景管理模块负责维护逻辑对象在场景中对应的Joint的实例，并提供逻辑层对场景表现的控制。主要职责有：

* 管理 Joint 列表，并维护Joints的帧循环
* 维护网络通讯及网络事件的分发
* 逻辑帧更新入口FixedUpdate
* 渲染帧更新入口Update
* 战斗UI的管理
* 战斗的暂停与继续
* 战斗掉落显示
* 战斗漂浮文字显示
* 战场特效管理

### 主逻辑(Logic)

战斗系统主要管理模块，主要负责：

* 战斗的状态管理
* 逻辑对象管理
* 战斗初始化
* 战斗流程
* 战斗结算

### 元素(Element)

元素是战斗系统逻辑对象的基础单位，代表一个可独立存在于战斗中、拥有自主更新逻辑的虚拟单位，如角色、效果等。

#### 角色(Role)

角色是战斗的主体，游戏中的英雄、怪物、召唤物等都属于角色。

#### 效果(Effect)

这里的效果是个侠义的概念，特指在游戏中的发射类元素，如子弹、炮弹、冲击波等。

### 技能

技能是战斗的核心元素，每个角色拥有多个技能，角色的所有攻击行为都由技能产生。普通攻击也作为一个技能实现。

而一个技能根据要实现的效果不同，又拆分为两个部分，技能(Talent)主体和Ability。

#### 技能(Talent)

Talent作为一个技能的具体实现，包含了技能的基本定义，如释放条件、目标规则、释放范围、伤害效果、对应的动作及特效、Ability集合等等。

#### Ability

Ability作为技能效果的一个子集，用于定义技能所附带的特殊效果，如属性变更（增加防御、降低防御、减少攻击、增加法术强度等等）、行为限制（禁止移动、禁止攻击）、保护效果（护盾、伤害免疫）等等。

### 印记(Mark)

印记是一个特殊的系统，通常作为技能的辅助释放条件，主要用来定义一些特殊的被动技能效果。

印记的主要特性是计数功能，可通过累加计数到设定值触发指定的技能效果。

### 陷阱(Trap)

陷阱是一个独立的游戏对象，通常由技能创建。可独立存在于场景中。

陷阱包含多种主动触发条件，开始触发、进入范围、离开范围、周期触发、结束时触发等，可结合技能实现丰富的技能效果表现。

# 基础规范

## 战斗场景

## 角色

## 动作

## 帧事件

帧事件用于定义基于动作时间触发的各类行为效果，如播放特效、音效，触发攻击行为、触发特殊动作等等。

如图：

E1-E3为动作中的3个帧事件，在技能开始后，随着动作的播放进度，会触发对应的帧事件，引发事件所定义的行为效果。

帧事件一类（通常只有攻击帧）由动作师填写到分帧脚本中，在导入Unity时自动生成，另外的则直接在Unity动画导入选项的Events部分进行设置。

所有的帧事件均需要在 RoleController中增加对应的事件处理方法。

需要在战斗逻辑中相应的帧事件，要在Talent.EventHandler 中增加对应的处理。

目前支持以下帧事件：

### PlayEffect（播放特效）

事件名称： PlayEffect

说明：播放角色身上绑定的特效，在RoleController中实现全部逻辑。

参数类型：String

参数格式：[特效名称]

帧事件示例：

播放技能特效

Function:PlayEffect

String:fx\_skill\_C1002\_atk2

### PlaySound（播放声音）

事件名称： PlaySound

说明：播放指定的音效文件

参数类型：String

参数格式：[音效路径]

帧事件示例：

播放攻击音效

Function:PlaySound

String:Sound/Combat/normal\_attack

### CamShake（摄像机震动）

事件名称： CamShake

说明：执行一次摄像机震动

参数类型：【待确认】

参数格式：【待确认】

帧事件示例：

执行摄像机震动

Function:CamShake

### AttackSpot（攻击帧）

攻击帧事件具有2个作用，一个作用用来定义攻击帧的发起时刻，另外一个作用是用来定义抛射类效果发射时的位置。

定义：void AttackSpot()

说明：攻击帧，在RoleController中实现空定义，在Talent.EventHandler()中实现具体逻辑。

事件名称： AttackSpot

说明：攻击帧，定义技能何时发起命中行为。

参数类型：无

参数格式：无

帧事件示例：

攻击帧

Function:AttackSpot

### DoAction（执行动作）

事件名称： DoAction

说明：执行特定的Action。

参数类型：String

参数格式：[action],[arg1],...,[argN]

action: 表示动作类型

Dash：冲刺

Teleport：瞬移

Ghost：残影

Invisible：隐身

arg1-n：动作参数，每种动作不同。

#### Dash 冲刺

冲刺按照恒定的速度前进，到达技能指定的目标位置或到达冲刺总时间时停止。

参数格式 ：Dash,[Time],[Speed],[Distance]

Time: 定义持续时间，0为不限时间。

Speed：定义冲刺速度。

Distance：定义以目标为起点的冲刺距离。

帧事件示例：

以2m/s的速度冲刺到目标后方1米。这里强调冲到目标后面的一段距离，由于目标的距离不确定，实际冲刺长度是可变的，所以忽略time参数，确保冲刺到目标位置。

Function:DoAction

String:Dash,0,2.0,1.0

以5m/s的速度向目标方向冲刺6米。注意，这里强调的是超某个方向冲刺固定的距离，而不关心目标的实际位置，所以这里忽略distance参数，通过time\*speed确定冲刺的总长度。

Function:DoAction

String:Dash,1.2,5,0

使用Dash时需注意，如果冲刺的距离可变，或者中途可能停止，则在制作动作时需要拆分独立的冲刺动作和攻击动作，如果冲刺阶段需要停止可调用talent.OnAnimationEnd()停止当前冲刺动作，进入攻击动作，开始执行伤害。

冲刺终止条件：处在冲刺中并且技能的抛射器产生了命中行为

冲刺按照恒定的速度前进，到达技能指定的目标位置或到达冲刺总时间时停止。

#### Teleport 瞬移

瞬间移动角色到目标位置后指定距离.

参数格式：Teleport,[TargetType],[Distance]

TargetType：参考目标，1：自己， 2：目标

Distance：相对参考目标的距离

帧事件示例：

瞬移到自己前方5米

Function:DoAction

String:Teleport,1,5

瞬移到目标后方1米

Function:DoAction

String:Teleport,2,1

#### Ghost 残影

在指定时间内自动生成一系列的残影效果，并持续一段时间

参数格式：Ghost,[interval],[totaltime],[duration]

interval：生成每个残影的间隔时间

totaltime：残影效果的总时间

duration：单个残影的持续时间，0或不填则默认与间隔时间相同。

帧事件示例：

每个0.2秒生成一个残影，持续2秒，每个残影持续0.5秒

Function:DoAction

String:Ghost,0.2,2,0.5

#### Invisible 隐身

事件名称： Invisible

说明：隐藏当前游戏对象

参数类型：Int

参数格式：1：隐藏，0：恢复

帧事件示例：

隐藏当前角色

Function:Invisible

Int:1

恢复当前角色

Function:Invisible

Int:0

### 执行技能Talent

执行指定的瞬发类技能

# 系统流程

## 战斗初始化流程

## 逻辑更新流程

### 战斗逻辑更新

### 角色更新流程

### 技能更新流程

### 效果更新流程

## 技能释放流程



# 模块设计

## 战斗

### 战斗场景

### 战斗模式

### 战斗状态

## 角色

### 基本属性

### 模型定义

### 角色状态

## 技能

## 技能组

## Ability

## 效果

## 印记

## 帧事件

所有帧事件需要在 RoleController中增加对应的事件处理方法，单据需要去实现具体逻辑。

与逻辑相关的帧事件需要在Talent.EventHandler 中增加对应的事件处理。

### PlayEffect（播放特效）

定义：void PlayEffect(string name)

说明：播放角色身上绑定的特效，在RoleController中实现全部逻辑。

### PlaySound（播放声音）

定义：void PlaySounf(string name)

说明：播放指定的音效文件，在RoleController中实现全部逻辑。

### CamShake（摄像机震动）

定义：void CamShake()

说明：、执行一次摄像机震动，在RoleController中实现全部逻辑

### AttackSpot（攻击帧）

攻击帧事件具有2个作用，一个作用用来定义攻击帧的发起时刻，另外一个作用是用来定义抛射类效果发射时的位置。

定义：void AttackSpot()

说明：攻击帧，在RoleController中实现空定义，在Talent.EventHandler()中实现具体逻辑。

### DoAction（执行动作）

定义：void DoAction(string args)

说明：执行特定的Action。

参数格式：[arg0],[arg1],...,[argN]

arg0: 表示动作类型

Dash：冲刺

Teleport：瞬移

Ghost：残影

Invisible：隐身

arg1-n：动作参数，每种动作不同。

#### Dash 冲刺

冲刺按照恒定的速度前进，到达技能指定的目标位置或到达冲刺总时间时停止。

#### Teleport 瞬移

定义：void Teleport()

说明：瞬间移动角色到目标位置。

实现模块：在RoleController中实现空定义，在Talent.EventHandler()中实现具体逻辑。

#### Ghost 残影

#### Invisible 隐身

隐藏当前游戏对象（激活所有Renderer）

### 执行技能Talent