# 技能系统需求

**玩家控制主角打怪，**

怪受伤，受伤后可能死亡，死亡后有的可能掉落物品，有的不会掉落

主角打死怪后可以得到杀怪经验，怪不同经验不同

**打怪**：按下技能按钮》调用**技能系统**释放技能》

调用动画系统播放动画》处理动画事件》

调用小怪受伤》可能调用死亡》

调用主角收集经验 》升级

**技能系统**

# 技能系统的策划

## (需求细化：确定需求的功能，列出*功能清单*)

以下为该**技能系统**中的三个技能描述，该系统中可能还有更多的技能

**普攻：**

冷却：0秒

消耗：无

攻击距离3米范围内的离主角最近的单个敌人，造成100%伤害，可连击

**六脉神剑:**

冷却：8秒

消耗：50法力

对5米范围内所有敌人每1秒造成300%伤害，持续4秒，并对敌人造成眩晕效果，持续5秒,对BOSS无效

**划地为牢:**

冷却：3秒

消耗：20法力

为主角前方120度角扇形区域内所有敌人每1.5秒造成200%伤害，持续3秒

 释放

# 技能系统的分析

**所有技能存在共性，攻击距离，范围，伤害的敌人类型，单攻或群攻，**对自身影响，对敌人影响…..，所以将技能所涉及的所有数据封装成一个技能数据类。

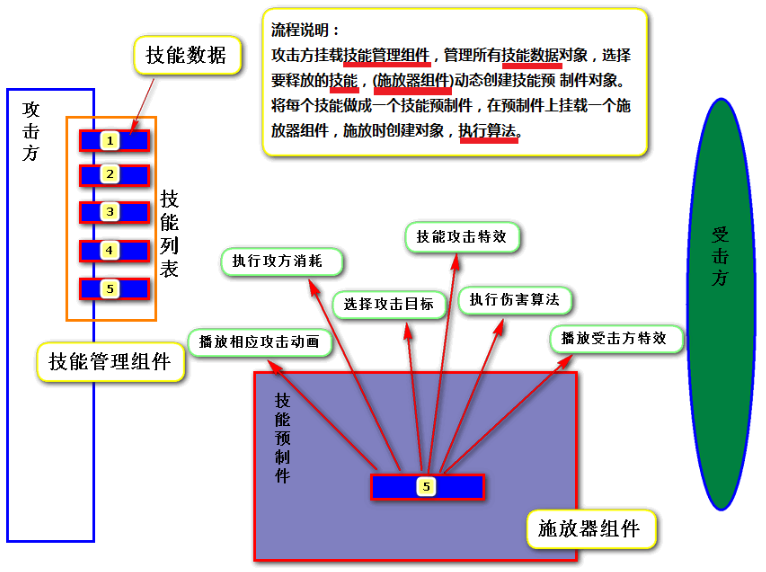
每一个角色所拥有多个技能的管理以及这些技能数据的加载，技能准备，冷却....等这些操作需要专门的对象来管理，因此提出一个技能管理对象负责。

每一个技能对象施放后可能得到不同的效果，如攻击算法的不同，特殊效果的不同….所以将技能施放后的算法由**施放器**来处理。如果施放方式不同，可以通过对施放器作派生来解决

* 技能数据
* 技能管理组件
* **施放器**组件
* **攻击方**：挂技能管理组件，管理所有技能数据对象，选择要释放的技能，(释放器组件)动态创建技能预制件对象。
* **受击方**
* 技能管理组件
  + 技能数据
  + 技能列表
* 释放器组件：挂在技能预制件，释放时创建对象，执行算法

# 技能系统设计

### 技能系统框架图

****

## 类设计

### 技能数据类图

所有技能共性，存在攻击距离，范围，伤害的敌人类型，单攻或群攻，对自身影响，对敌人影响

1．技能数值可调节

2. 不同技能释放的特效，动画，敌人受击特效都不同，可调节

3. 玩家可选择随身技能。

**技能数据类 SkillData**

**类成员:**

技能ID，(skillID )

技能名称 (name)

技能描述(description)

冷却时间(coolTime)

冷却剩余(coolRemain)

魔法消耗(costSP)

攻击距离(attackDistance)

攻击角度(attackAngle)

**攻击目标tags[] , (attackTargetTags)**

**攻击目标对象数组 (attackTargets)**

连击的下一个技能编号(nextBatterId)

伤害比率 (damage)

持续时间 (durationTime)

伤害间隔 (damageInterval)

**技能所属（Onwer），**

技能预制件名称 (perfabName)

**预制件对象 (skillPerfab)**

动画名称 (animationName)

受击特效名称 (hitFxName)

**受击特效对象(hitFxPerfab)**

等级(level)

是否激活（activated）

技能类型 (SkillAttackType attackType ) ：单攻Single，群攻Group

伤害模式 (DamageMode damageMode) ：圆形Circle，扇形Sector矩形Rectangle

### 技能管理类图

**技能管理类**

**类成员:**

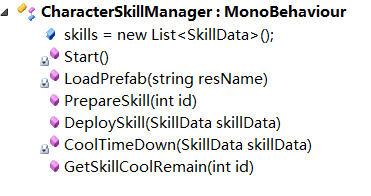
**数据：**

**管理多个技能数据对象---容器List<SkillData> 所有技能**

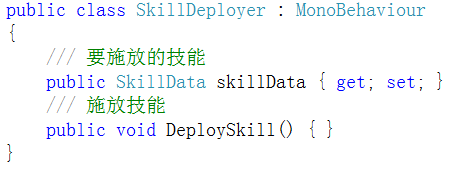
**方法：**

* + 1. **初始化**
    2. **准备技能**
    3. **施放技能**
    4. **技能冷却处理**
    5. **获取技能冷却剩余时间**

**类图:**

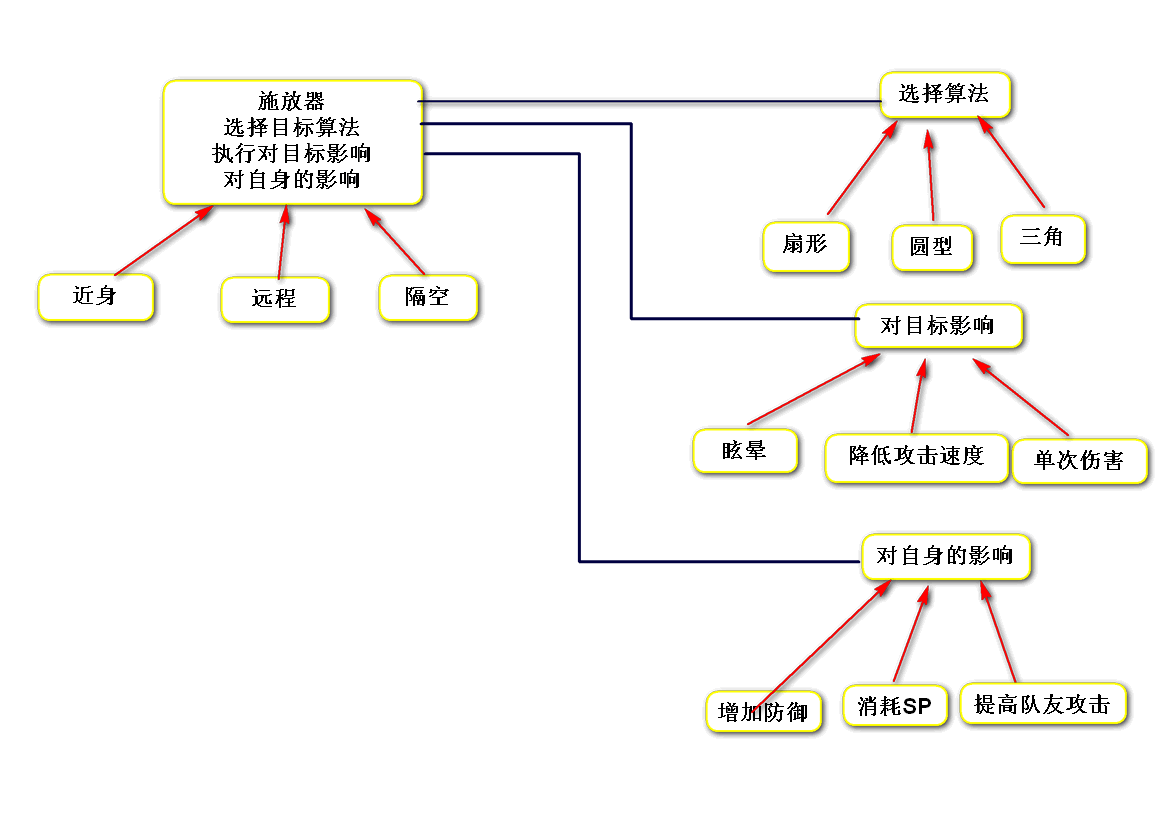


## 为了方便测试，暂时先实现施放器类部分功能。

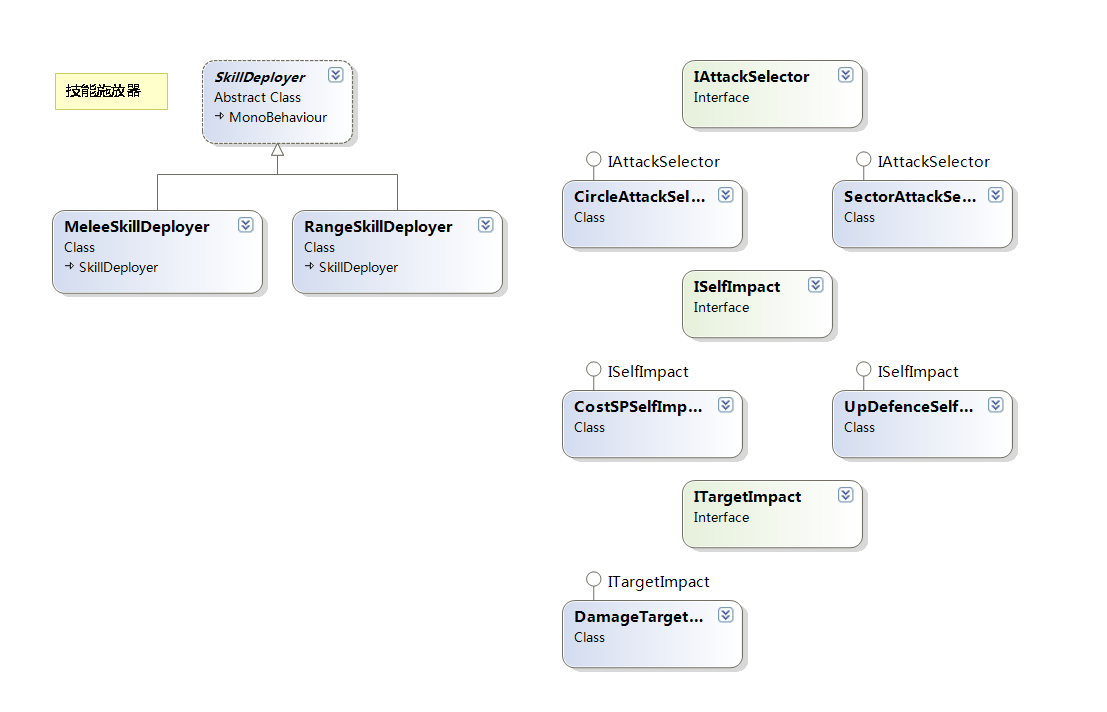
****

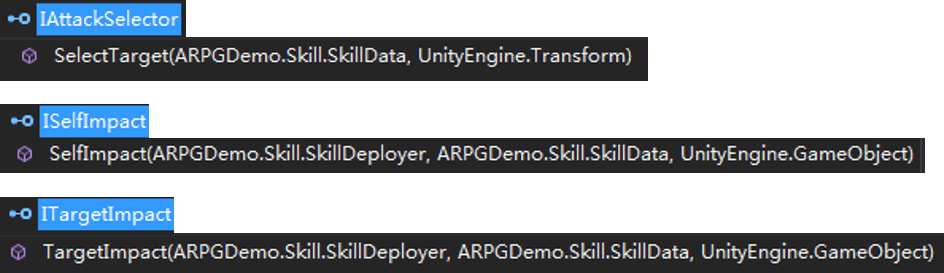
## 测试技能管理可以管理技能

## 施放器架构图

****

## 施放器类图

****

****

### 技能施放器类图

**技能释放器类SkillDeployer**

**类成员:**

**数据：**

**当前施放的技能**

**确定技能目标选择算法**

**确定技能自身影响算法**

**确定技能目标影响算法**

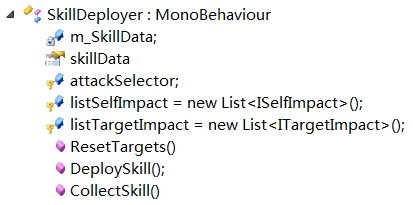
**方法：**

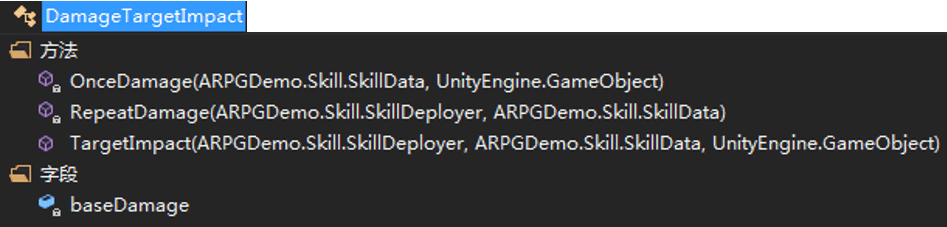
**重置目标**

**施放技能**

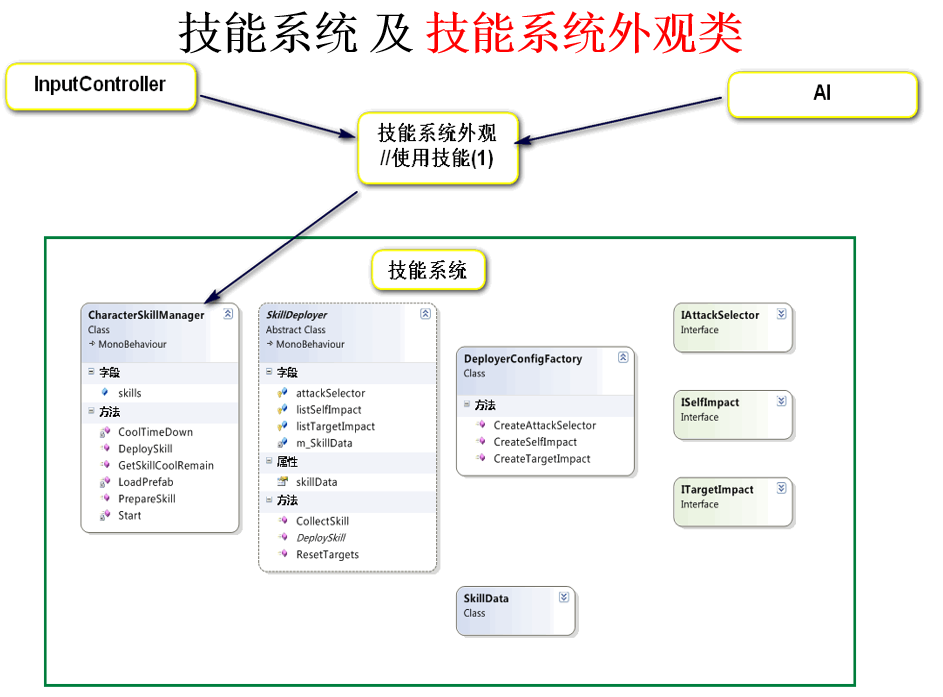
**回收技能**

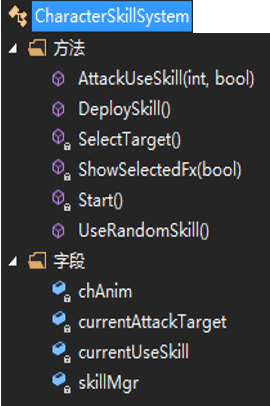
**类图：**





### 技能系统外观类

****

****