YY33技能系统设计

技能系统简介

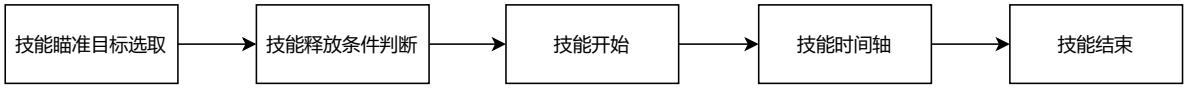
从2019年底开始，我的工作重心转移到YY33，主要负责设计了YY33的技能系统以及后续跟进技能相关的需求。在设计YY33的技能系统之前，我的工作内容主要也是围绕着H52-非人学园的技能系统；此外，YY33开始时的玩法设计类似MOBA游戏中的自走棋玩法，由于以上两个原因，YY33的技能系统在某些方面和H52有一定的相似和传承的地方。

技能系统的各组成部分及设计

技能系统主要管理着战场中单位的行为和能够产生的效果，例如主动技能/被动技能/BUFF/攻击盒/法术场/位移/伤害结算等等。一个常规的技能通常都伴随着单位进入某一个动作，在动作播放的过程中能够释放出攻击盒，或是能够给自己加上一个增益状态(BUFF)。

主动技能

主动技能通常是玩家/AI主动使用，首先技能会选择一个瞄准目标，在满足一系列释放条件后技能开始，并进入了技能的时间轴，在技能时长结束或者被外部因素打断时中止。



瞄准目标选取

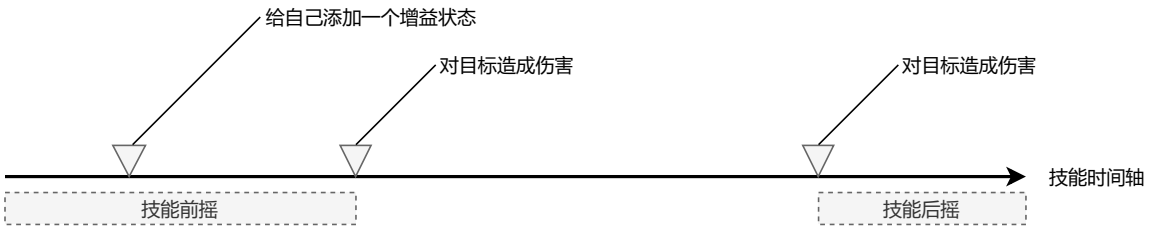
瞄准目标的选取在游戏中是AI来完成的，目前有三种瞄准目标类型：

- 选择一个战场单位
- 选择一个位置
- 选择一个方向

选择后的瞄准目标会缓存在技能对象中，供技能时间轴上的触发事件使用。

技能时间轴

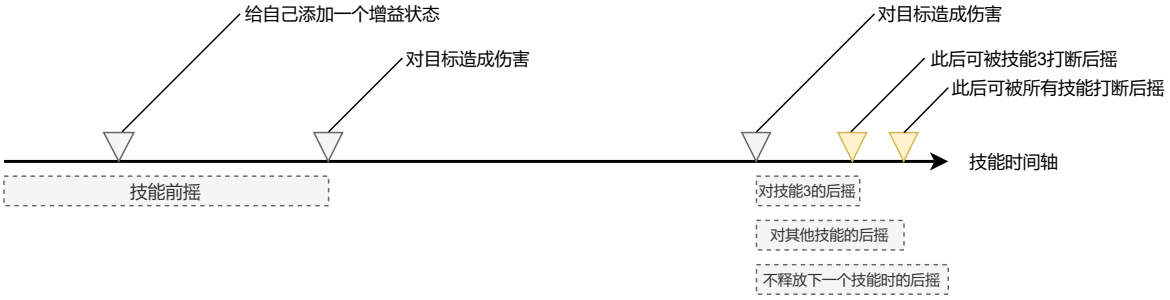
时间轴是主动技能所产生的技能效果的抽象。通常时间轴的长度与该技能的动作相同，通常策划也是对着技能动作逐帧配置伤害帧等技能效果的时间。时间轴上可配置的触发效果是一系列封装好的，供策划调用的条件和效果函数。



技能结束/打断

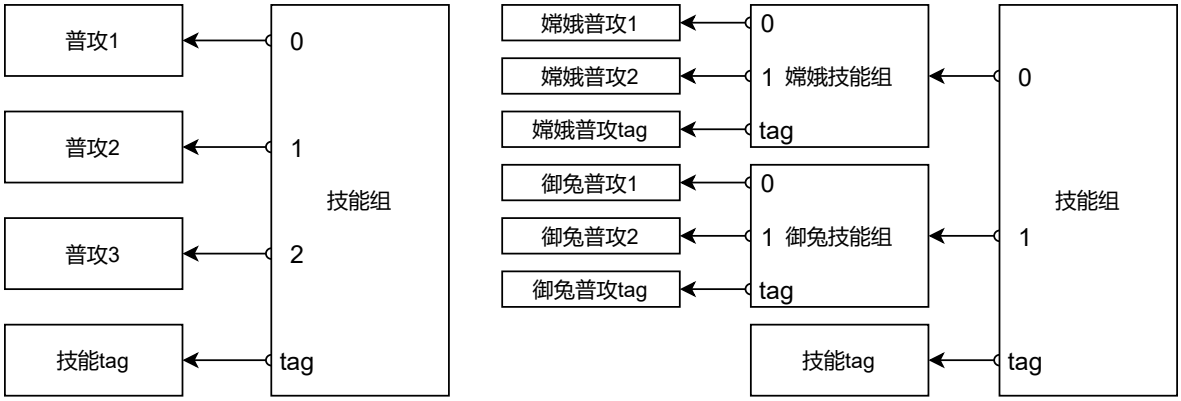
技能的打断和结束都是动作和时间轴的停止和复位，因此实现上基本一致，时间轴在停止后后续的事件不再触发，从而达到逻辑和表现的一致性。通常来说，技能被打断都是源于外部因素，例如其他单位的控制性技能；技能结束主要都是来自自身因素，包括达到技能时长的正常结束，以及施放超必杀等高优先级技能时也会结束上一个技能。

此外，策划考虑到技能衔接的流畅度，会希望在不同技能前后释放时，前一个技能的动作提早结束，立即放出下一个技能，也就是打断技能后摇。由此引出了一个额外的技能结束机制，程序中命名为`supercancel`点。`supercancel`点可以针对特定技能配置，通过安插多个`supercancel`点，能够配置对不同技能不同的后摇



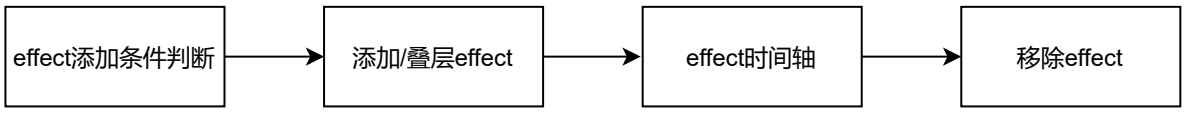
技能组的概念

常见的moba类游戏中，通常都会有多端技能的概念，即同一个按键点击多次会释放不同的技能，最典型的例子就行英雄的普攻，其会在多个动作间顺序切换。为了实现这一类需求，我引入了技能组的概念，技能组是一类虚拟的技能，相当于技能的选择器，其根据当前的技能tag选择其中的一个子技能来释放。同时，技能组也支持嵌套，从而能够使得英雄在两套技能组之间切换，例如YY33中能够变身的英雄嫦娥就使用了嵌套的普攻技能组。



被动技能与BUFF

在H52的技能系统中，被动技能与BUFF是两种实现方式，但是在开发的过程中会发现，有大量代码是可以复用的，甚至被动技能本身也可以看作是一个常驻的BUFF。因此，在YY33的技能系统中，被动技能与BUFF的实现方式是相同的，程序中将其统称为`effect`。



`effect`的第一个主要作用是能够方便策划进行时间相关的操作，例如想要将某个单位眩晕5秒，或者是增强某个单位的攻击力持续3秒，策划通常都是通过配置一个对应时间的`effect`来实现；effect的第二个主要作用是完成一些特定时机触发的效果，例如想要在受到伤害时对敌方反伤，或者是使用技能后强化下一次普攻，这些都能够通过`effect`来配置。

effect的添加与叠加

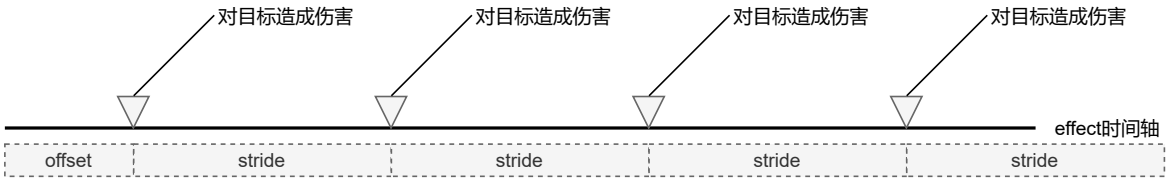
当单位A试图向单位B添加一个编号为C的effect时，如果B身上此时还没有编号为C的effect，那么就触发了effect的添加（add）；如果此时B身上已经有了编号为C的effect，那么就触发了effect的叠加（stack）。YY33的技能系统中定义了以下几种叠加规则：

- 覆盖。使用新effect的等级
- 叠层。使用两者等级之和
- 冲突。使用原来effect的等级

目前时间上的叠加统一采取了两者剩余时长中取长，根据需求可补充新的时间叠加规则。

effect时间轴

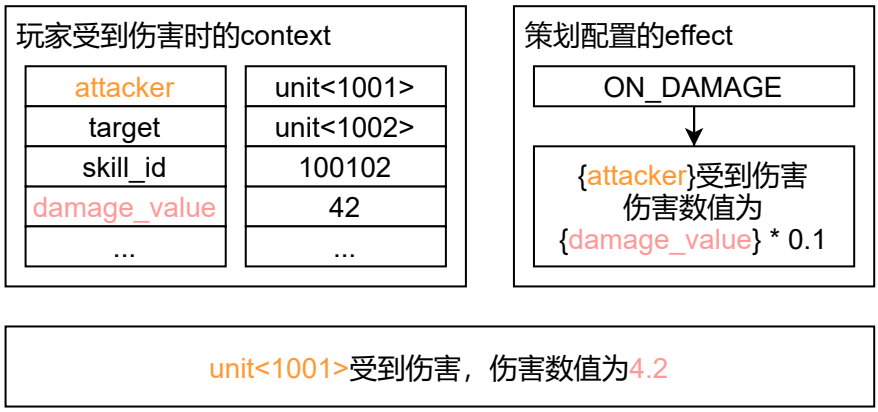
相比与技能的时间轴，effect的时间轴并不对应于某个动作，因此即使对于同一个effect来说，其时间轴的长度是任意可变的；除此之外，effect的时间轴是可循环的（例如每0.5秒造成灼伤效果的debuff）。根据经验，effect大多都使用的是循环的触发效果，但同时也有一定数量的一次性的触发效果，在程序中通过一个第一次触发的时间（offset）和循环间隔（stride）加上循环次数（count）即可描述一系列循环的时间轴上的点。



effect中的被动事件与context

effect中的被动事件主要通过简单的观察者模式，配合合适的外部埋点触发。例如，在玩家受到伤害时，即会发出一个玩家受到伤害的事件（`send_event(ON_DAMAGE)`），对应的effect能够接收到对应的事件并触发对应的技能效果。通常在某个事件产生的时候，我们也需要当时的上下文信息，例如攻击者，攻击者使用的技能，伤害值等等，这些信息会在事件发出的同时被打包成一个context传递到接收者（通常是effect）中。

context类实际上就是一个dict的派生类，其中简单地封装了一些公式解析相关的方法，因为context中的数值大多都是在策划填写的公式中所使用的。一个简化了的产生反伤效果的被动事件配置和流程大致如下。



触发效果

程序中的触发效果是真正落实技能效果的模块，前文中所使用的**触发效果**一次均指代该模块中的内容。也就是说，技能时间轴/effect时间轴/被动触发事件等等所有能够触发的技能效果，共用一套外部接口。一个触发效果一共包含以下部分信息，即：

- 谁来触发，即触发者`target`
- 在什么条件下触发，即触发条件`condition`
- 触发什么技能效果，即真正的触发函数`modifier`
- 触发时刻的上下文

触发者

常见的触发者包括被动技能所有者/攻击方/受击方/瞄准单位等等。但是触发者并不一定局限于战场中的单位，其理论上能够是任意的对象，例如技能对象/effect对象/甚至伤害对象等等。

触发条件与触发函数

这些实际上都是预定义的规范化的接口，供策划在外部配置。在YY33项目中，由于目前还没有太复杂的技能，触发条件基本没有使用，触发函数目前控制在21个左右（yy33目前有6个英雄和1个boss）。

单位状态

后续的工作方向

技能编辑器

pyflow编辑器

polaris编辑器

技能结构的改进

纯粹的ECS

小结