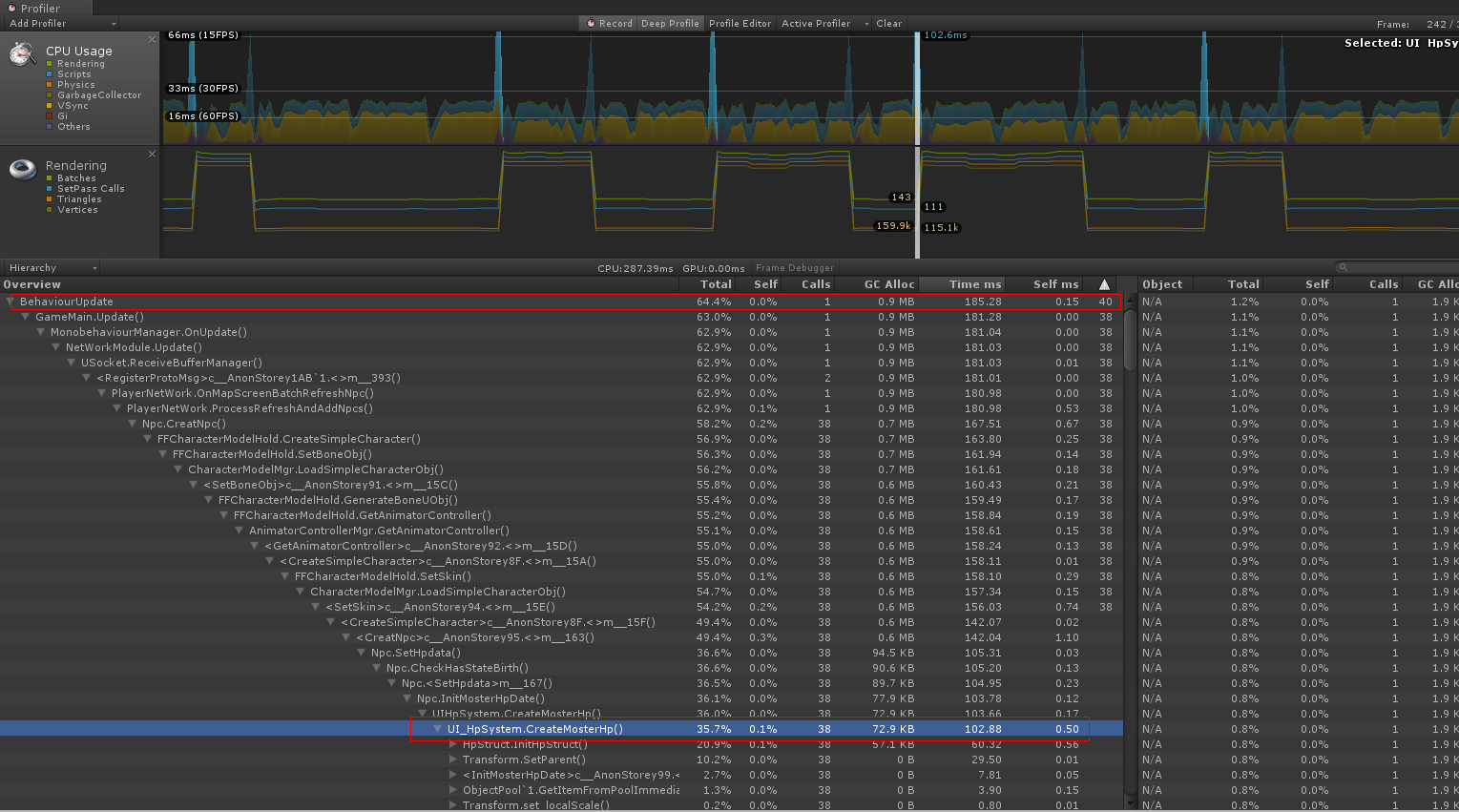
一、血条和名字的对象放入对象池管理

此次优化血条主要是把血条对象同一放入对象池管理，血条中用到的控件getcompment方法放在Init中，下次从对象池中获取就不必再次调用getcompment方法了。把血条和伤害数字逻辑分开。所以性能的提高在于加载CharactorBase的时候。

测试用例：在界王神界召38个NPC，玩家进出九屏，检查消耗。

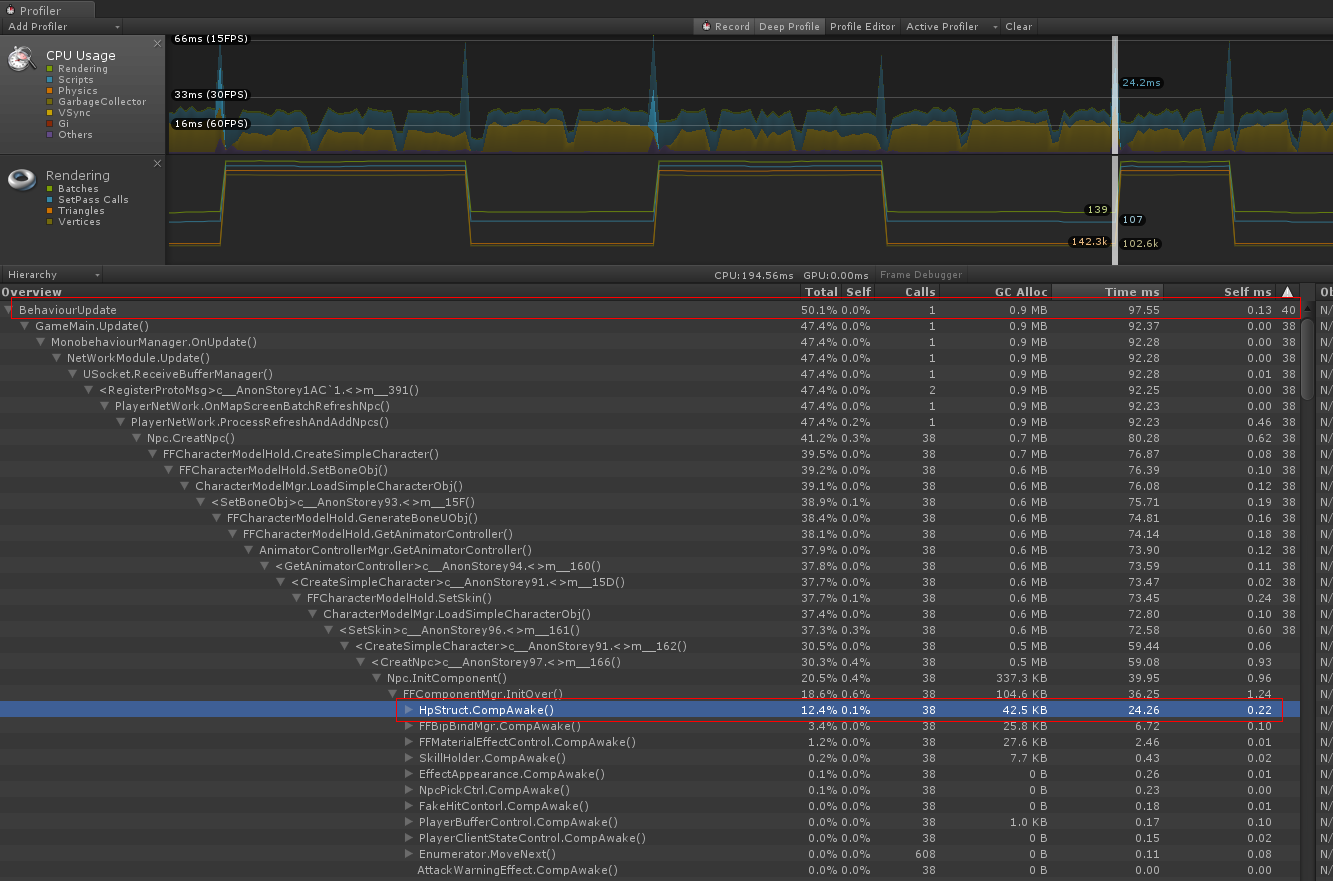
优化前：CreateMosterHp()为创建血条方法

GC： 72.9KB Time:102.88ms 总耗时185.28ms



优化后：HPStruct.CompAwake()为创建血条方法

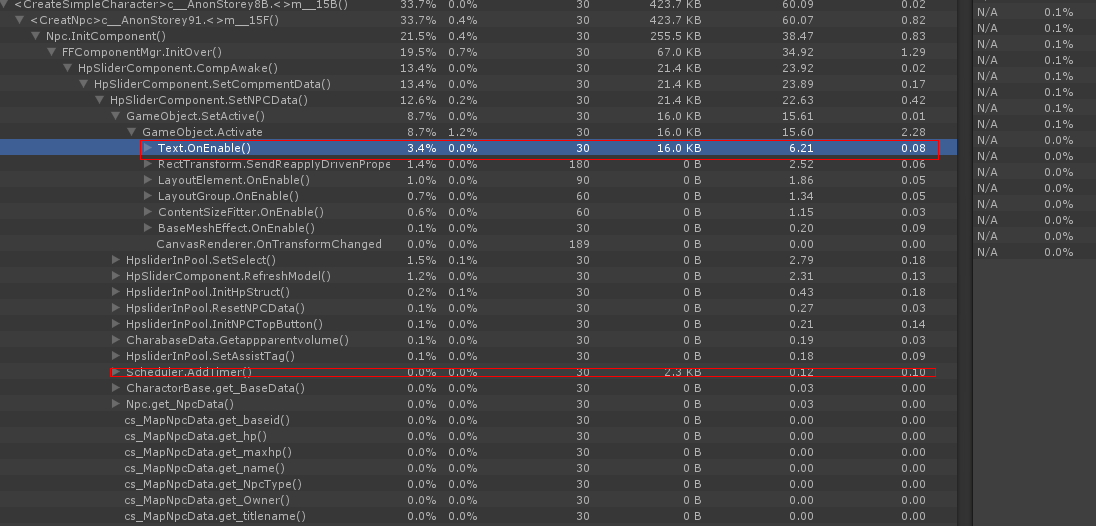
GC: 42.5KB Time:25.03ms 总耗时：97.55ms



|  |  |  |  |
| --- | --- | --- | --- |
|  | GC | Time | 总时间 |
| 优化前 | 72.9K | 102.88ms | 185.28ms |
| 优化后 | 42.5K | 25.03ms | 97.55ms |
| 优化数据 | 降低30.4K (41%) | 降低77.85ms(75%) | 降低87.73ms(47%) |

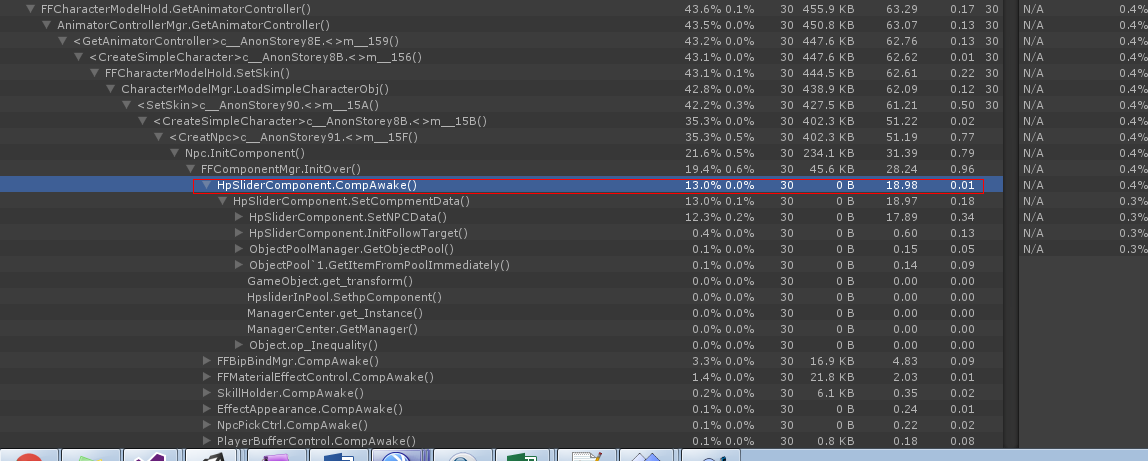
二、GC优化：

1、在界王神界自summon30个高级TF(NPC)，反复进出九屏，发现在创建NPC时HPSliderComponent. CompAwake中，有21.4k的GC，如图，其中16k来自于Text的OnEnable方法。2.3k来自于AddTimer。



解决办法，在HPSliderComponent. CompAwake去掉AddTimer方法的使用，在血条组件中，使用NoMaskText替换所有的Text，注意：NoMaskText组件不能被Mask裁切，所以在需要被Mask裁切的组件中不能使用NoMaskText。

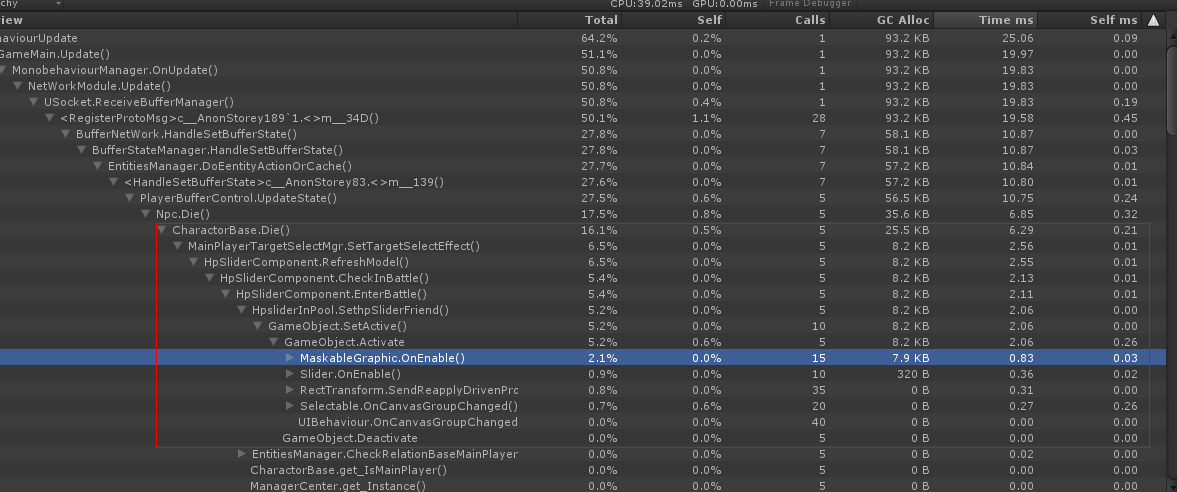
优化之后：HPSliderComponent.CompAwake已经不再产生GC了。



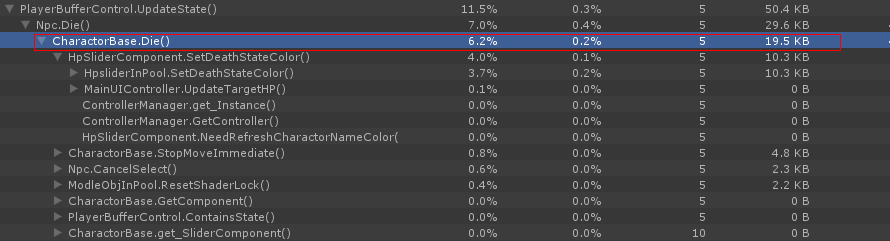
|  |  |  |
| --- | --- | --- |
| HPSliderComponent. CompAwake | GC | Time |
| 优化前 | 24.1K | 23.92ms |
| 优化后 | 0K | 18.98ms |
| 优化数据 | 降低24.1K | 降低4.94ms |

2、NPC.Die()优化5个NPC同时死亡。

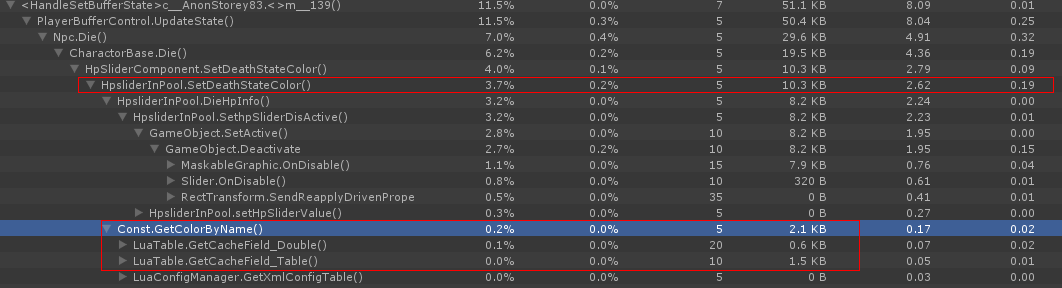
（1）NPC死亡时会调用MainPlayerTargetSelectMgr.SetTargetSelectEffect(),这个方法是为了在死亡时取消选中，而MainPlayerTargetSelectMgr.SetTargetSelectEffect()方法会额外做RefreshModle操作，所以此处用SliderComponent.SetSelect(false)方法替换



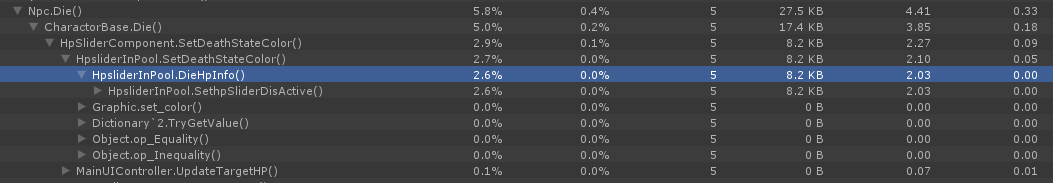
优化后：NPC.Die（）调用5次，GC由原来的25.5k降为19.5k。



（2）SetDeathStateColor方法中给Img赋值颜色时使用Const.GetColorByName("death") 方法，此处改为初始化时Init到颜色字典中，调用Const.GetColorByName("death") 方法时直接获取字典中的颜色值，避免调用lua产生GC



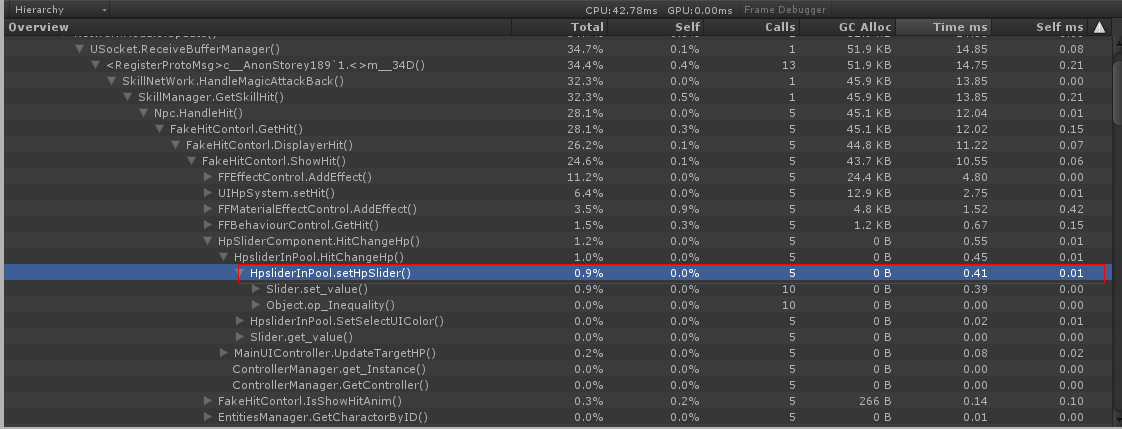
优化后：已经没有了Const.GetColorByName()的调用。



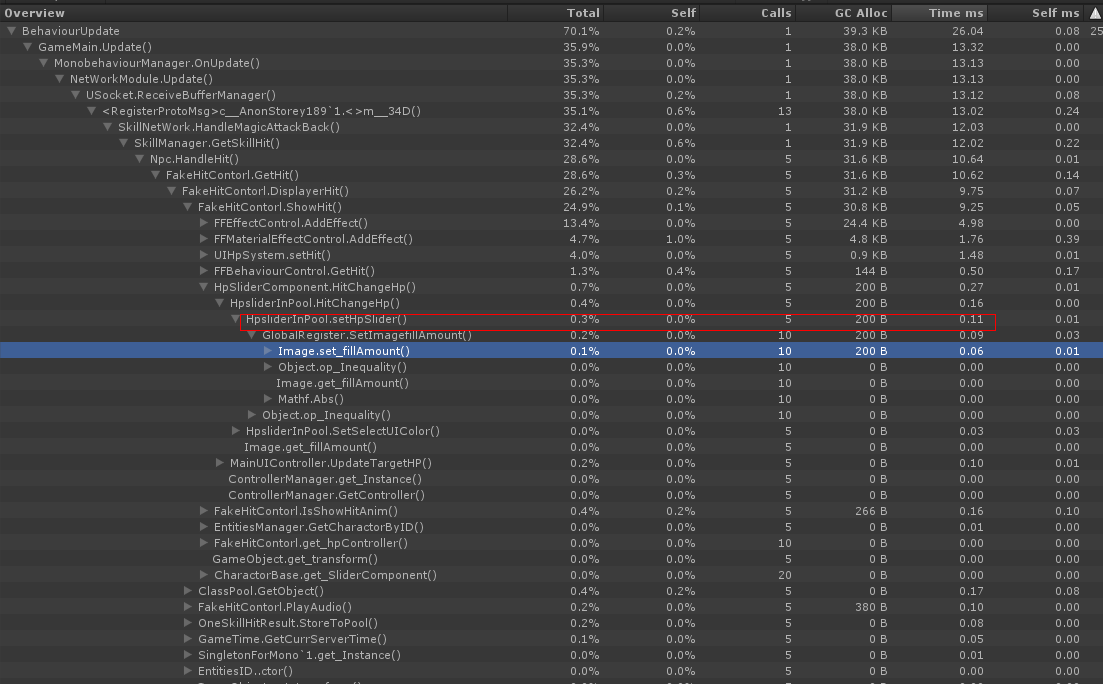
|  |  |  |
| --- | --- | --- |
| NPC.Die | GC | Time |
| 优化前 | 35.6K | 6.85ms |
| 优化后 | 27.5K | 4.41ms |
| 优化数据 | 降低8.1K(22%) | 降低2.24ms(32%) |

3、把Slider换成Image。

优化前：



优化后：



|  |  |  |
| --- | --- | --- |
| HpSliderInPool.SetHpSlider() | GC | Time |
| 优化前 | 0 | 0.41ms |
| 优化后 | 200B | 0.11ms |
| 优化数据 | 增加了200B | 降低0.3ms(73%) |

4、其他优化：

（1）血条层级简化，原来的父节点PlayerHpInfo->PanelPlayers->content三层简化为PlayerHpInfo一层。

（2）FollowTargetUpdate优化，原来的FollowTargetUpdate每一帧都会设置血条到Target的postion，改为距离大于0.001时在设置。Rotation改为变化时在设置，Scale改为当血条与相机的距离变化超过0.001时在改变。

（3）玩家血条和NPC血条在对象池中各缓存20个。超过这个数量才会进行New操作。

（4）去掉原来血条上SelectUI的AddTimmer操作，使用一个bool值作为标记，其逻辑在本身的UpDate中执行（标记为true时执行该逻辑，否则return）