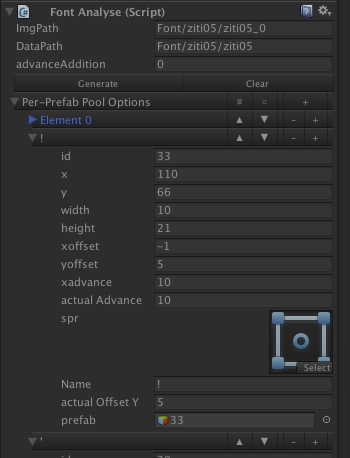
**Blablabla** 功能书

工具类：

1.字体生成：使用软件BmpFont



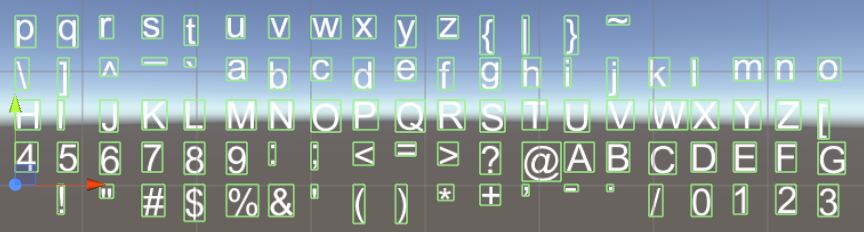
-在场景内新建一个空GameObject为其绑上Font Analyse组件（最好统一命名为FontSettingXXX.）

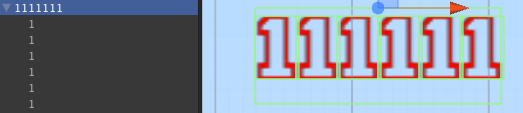
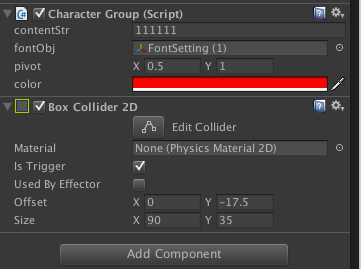
-把对应的字体图片拖和imgPath，TXT文件拖入DataPath

-指定额外字间距advanceAddition。默认为0，加宽1个单位就填1.

-点击Generate生成。

-双击此Object，定位到场景中的字体对象如下图所示，

-已经默认为每个字符绑上BoxCollider2D。对于某些字符例如小h。可以绑多个BoxCollider2D以更符合其形状。如果不能满足，就绑PolygonCollier2d(最好不要）

2.字符组生成：

-在场景中新建一个GameObject，为其绑上CharacterGroup.

-将『字体生成』步骤中新建的字体对象拖入fontObj。

-指定颜色color

-在contentStr中输入你想要的字符串。会自动生成字符对象和字符组的boxCollider。（建议将Gameobject的名字和你输入的内容改成一样的，便于寻找）

-根据需要修改pivot，即字符组的中心点（关系到旋转缩放的效果），（0，0）为左下角。默认为（0.5，0.5）即上下左右中心点。

注意：

若你拖拽新的字体文件对象到fontObj，或者改变pivot，以上步骤都会自动处理无需重新输入字符。

但如果你修改了此处引用的fontObj本身（就是修改了字体对象本身），就需要重新输入字符来刷新（否则会变不见）。所以最好一开始弄好字体就不要再改了。不然之后还要一个个改。

修改颜色也需要重新输入字符来刷新。

功能类

分为两种，

* 一种是针对具体Gameobject的功能，也就是说要在场景中生成这个物体后再为它绑定对应的功能。与这个物体是什么内容什么字体一点关系也没有。
* 一种是针对不同字符的功能，属于字符特性。也就是a有a的功能。无论他是什么颜色什么字体什么效果，只要是字符a,就会有这个功能。

1.具体对象功能：

目前具体对象功能可以绑定上字符组上，有的也可以绑定在字符组下的某个字符。具体功能具体注明。

* 移动、旋转、缩放（建议只绑给字符组对象，不要绑给单个字符，否则有碰撞问题）
* alpha(建议绑给单个字符而不是一组字符，因为alpha只能针对单个SpriteRenderer来做，一次性绑给一组等于对每个spriteRender去做处理，一定程度上影响效率。）
* 为对象加上Characters action组件。
* actionType选择所需要的类型
* TriggerCondition触发条件：Awake一进入场景就触发，OnCollider与主角碰撞后才触发
* Loop -1/0代表无限循环，1代表从开始值到结束值。2代表从开始值到结束值再返回开始值。以此类推
* startVal开始值 reset记录对象当前对应的属性，goto将对象设置为当前填的值
* endVal结束值
* duration loop=1所花费的时间

注意：Characters action组件的移动功能是附带碰撞判断的，也就是说，正常情况下如果你把2个字符组放得碰到一起，他们不会相互影响。但是如果这个物体绑了Characters action组件中的移动功能，碰撞触发的时候，会让物体往反方向运动。这也是不建议给单个字符绑定移动功能的原因。

* 冰（建议绑给一组字符，因为每次离开一块冰都要通知，进入一块冰也要通知，零碎连续绑就反复的通知，影响效率）
* 角色来到冰上面，速度暂定翻倍。在冰上运动时方向不可改变，静止时可以改变，可以跳起。跳起后就不受到方向限制。
* 掉落

2.字符本身特性：

功能直接写死在代码里。XX字符对应XX功能。要改的话需要程序改。

* < >   
  一个往左上弹，一个往右上弹。实现机制为碰撞后为主角加一个水平方向的力和一个向上的力。具体数值填在GlobalController的Setting组件上。往左弹会取X的负值。