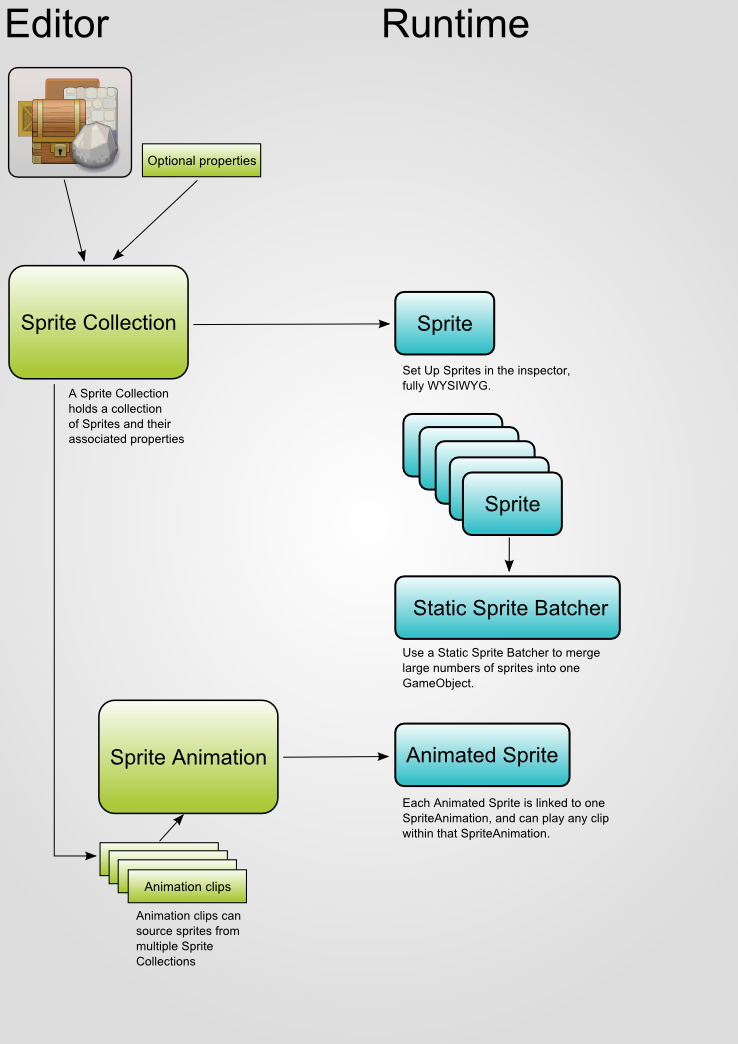
**系统综述**

2D Toolkit分为两个系统：运行时组件（runtime components）和脚本编辑器。

脚本编辑器在Assets目录下产生资源，运行时脚本在场景中产生objects。

两者关系如下图：



**术语和介绍**

**精灵集（sprite collections）**

精灵集是2D Toolkit的核心，可以把它想象成存放sprite的文件夹，sprite collection界面设置精灵及其关联属性，包括碰撞区。精灵集产生一个或多个贴图图集，替代你的原图片从而提高性能。

**精灵（sprite）**

精灵集中的独立组件视为精灵。这些应该和你的原图片统一，尽管内部可能有各种变化。你可以把精灵绑定在其他精灵上，或通过其他手段来控制它们。

**静态精灵批处理器（Static Sprite Batcher）**

批处理器把精灵集中的多个精灵联合在一起，合并它们成为一个物体。这个合并操作是无损的，日后可以逆操作（解散操作）再次使用它们。设置在精灵集中的碰撞区信息被直接转换到批处理器中。

**精灵动画（Sprite Animation）**

精灵动画包含动画切片。动画切片是一段时间内的sprite序列。动画切片在播放时可触发事件，这个事件你可在代码中响应。比如人物脚踩地，播放一个音效。动画切片可使用精灵集中的任意sprite。

**带动画的精灵（Animated sprites）**

带动画的精灵在场景中被创建并绑定了精灵动画。带动画的精灵可以播放绑定的所有动画切片。

**字体（Fonts）**

Font objects transform text or xml BMFonts to be consumed by TextMeshes

**文字网格（Text Meshes）**

文字网格使用字体对象显示字符串。文字网格定义了可显示的最大长度，所以它不用不断的新建内存。

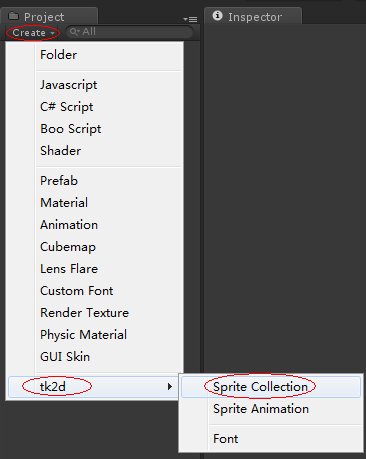
**快速开始**

1. **精灵**
   1. **创建精灵集**

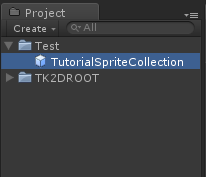
精灵集是2D Toolkit的一个基础形式。它以列表形式展示贴图和关联属性。精灵集可处理图集的创建和各种用户参数。一旦创建，精灵的任何改变都会自动重构图集。V1.6起可以加入字体到精灵集合中。添加到精灵集中的字体不能使用渐变，因为需要单独材质。

接下来通过2D Toolkit提供的资源创建一个简单的精灵集合。

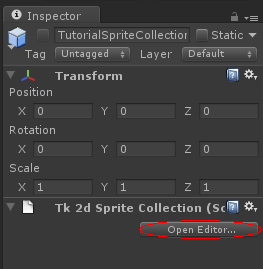
* Project视图中点击“Create > tk2d > Sprite Collection”来创建精灵集。也可在菜单栏中通过“Assets > Create > tk2d > Sprite Collection”来创建。（在文件夹中创建精灵集会是你的工程更整洁）



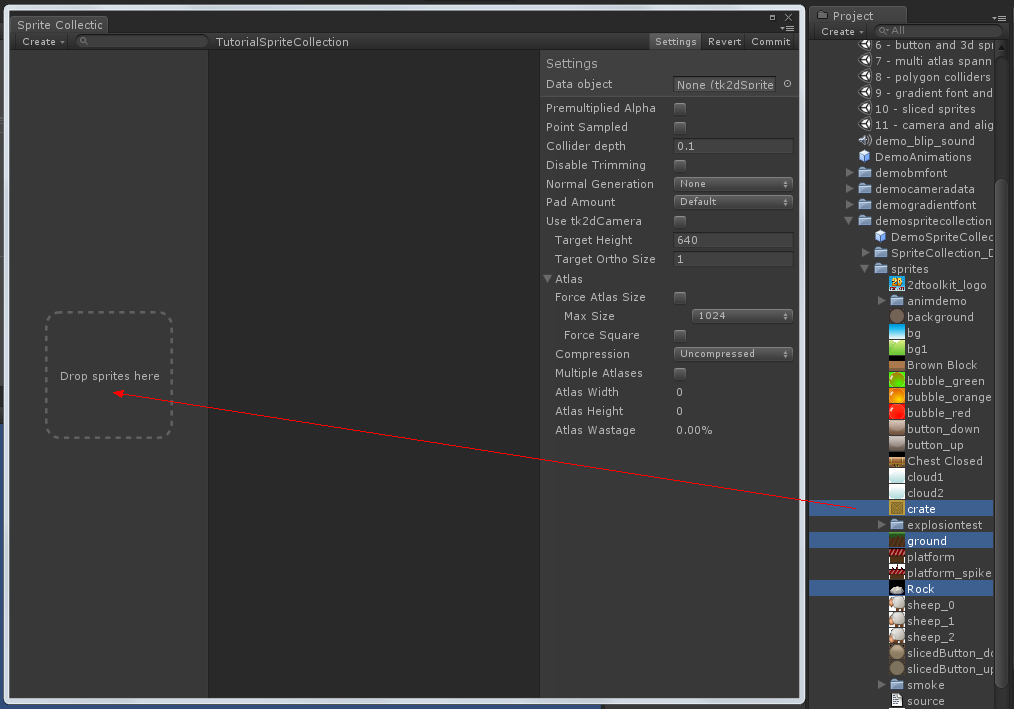
* 创建好精灵集后起一个合适的名字，本例中叫它“TutorialSpriteCollection”好了。



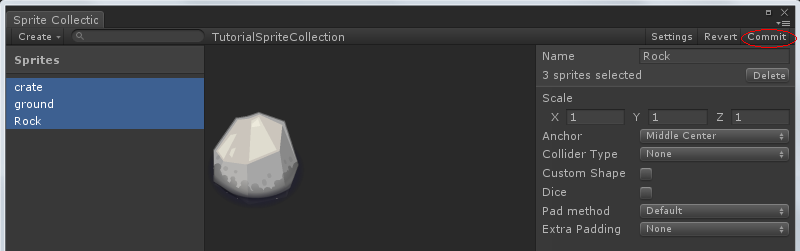
* 点击“Open Editor…”打开编辑器



* 把精灵加入到精灵集里，拖拽资源图片到“Drop sprites here”虚线框中。本例中拖入了demo里的资源图片，你也可以放入任何目录下的任何图片。

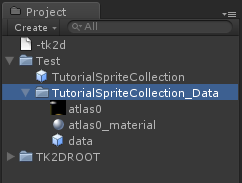


* 拖拽后这三张图片被选中，你可以改变其参数。然后点击“Commit”。



当你在此界面（精灵集编辑界面）做了任何修改都请点击“Commit”。

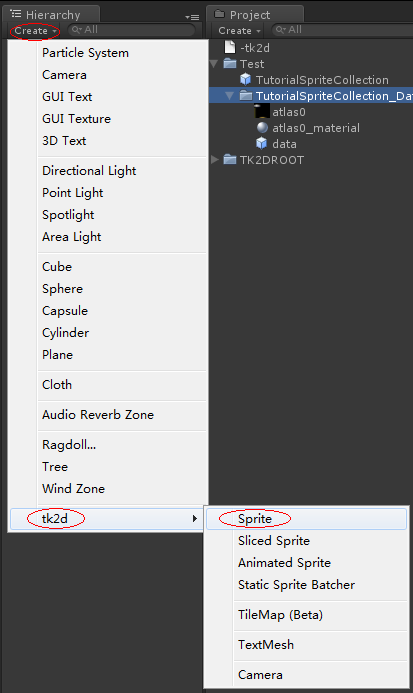
顺便说一下，点击Commit后Project界面中生成了“TutorialSpriteCollection\_Data”文件夹。



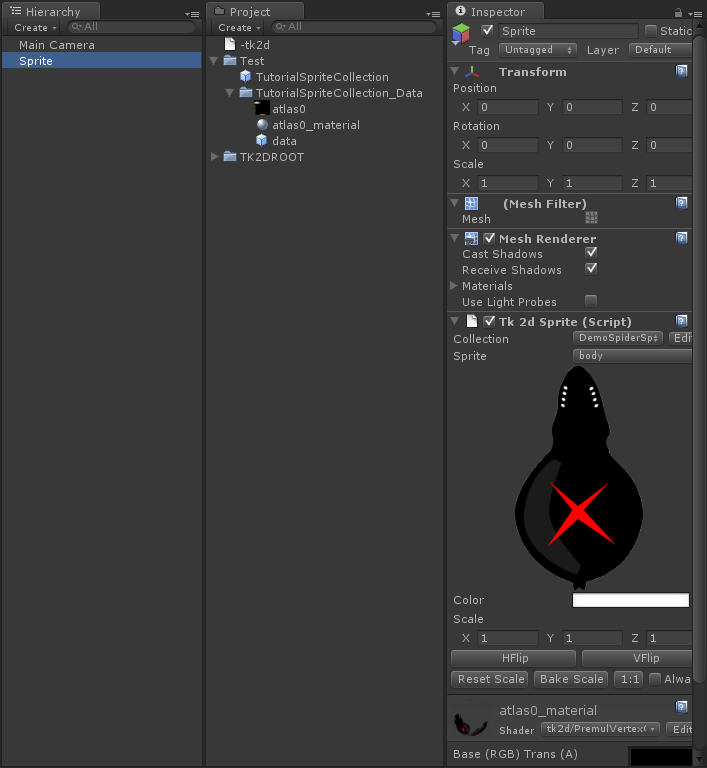
到此，通过资源图片创建的精灵集已完成。若资源图片改变了，这些精灵集中的图集及关联参数将自动重构。

注意：精灵集合默认是为960\*640分辨率创建的，如果设备分辨率不同，请改变后重新Commit。

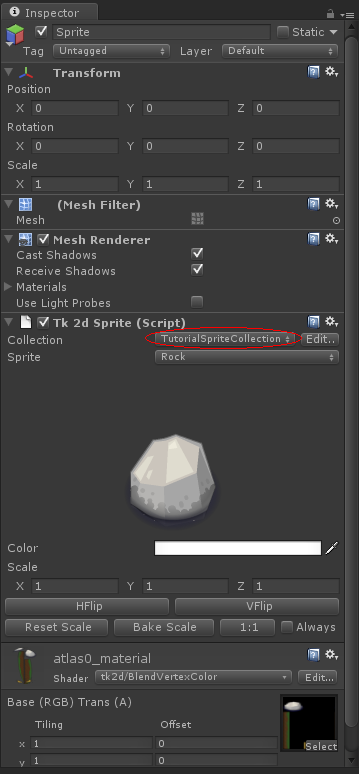
* 1. **创建精灵**
* Hierarchy界面中点击“Create > tk2d > Sprite”创建精灵。



* 创建好的精灵出现在Hierarchy界面中，并根据可用信息自动为其分配了名字。

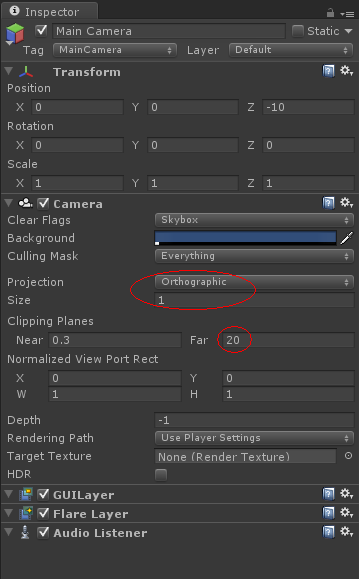


* 选中场景中刚建好的精灵，或在Hierarchy界面选中它，将在右侧出现Inspector界面。

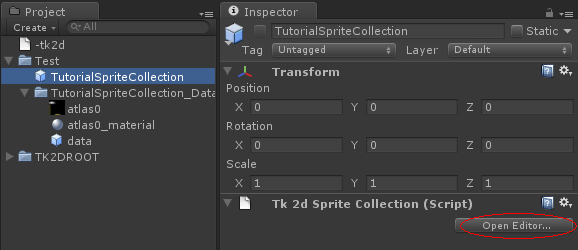


如果2D Toolkit没有为你自动选择正确的精灵集，你可以自行设置（如上图）。选择精灵集后选择一个精灵。场景中的实例将立刻改变。如果你的精灵集设置了碰撞区，则相应的碰撞区也会在场景中自动创建。

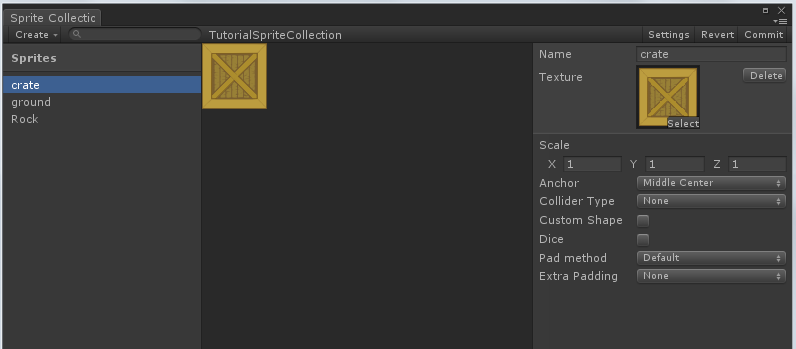
注意：将摄像机参数设置为和精灵集一样。默认参数为size为1的正交摄像机，目标分辨率为960\*640。如果你的摄像机以这个参数创建，则你的精灵将完美呈现。



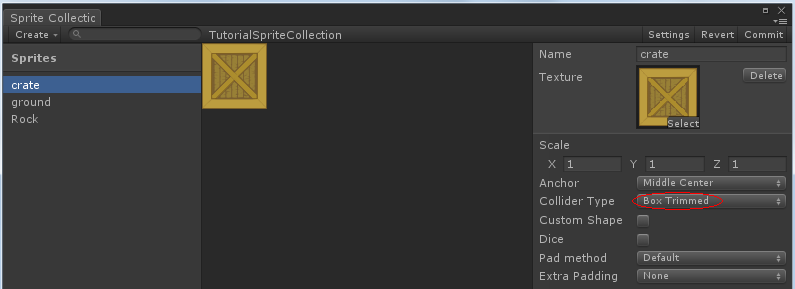
* 1. **为精灵绑定碰撞区**
* 选择精灵集，然后点击“Open Editor…”



* 你可以设置各种参数，诸如：锚点、切割和填充算法。你可以同时选中多个精灵来编辑。



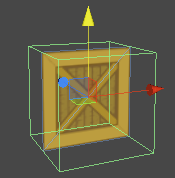
* 改变碰撞区类型为“Box Trimmed”（盒子边框型），这将在贴图四周创建边框。如想更多的控制其位置，请选择“Box Custom”，便可拖拽边框。



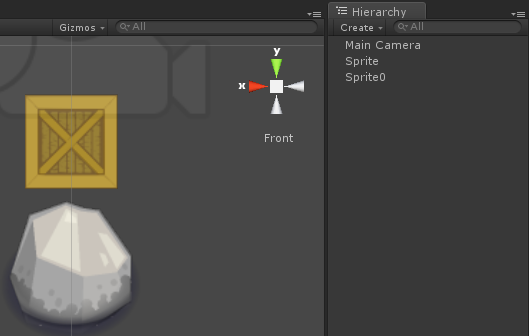
* 点击Commit。

每次做了改变都请点击Commit，以便重构图集和顶点数据。

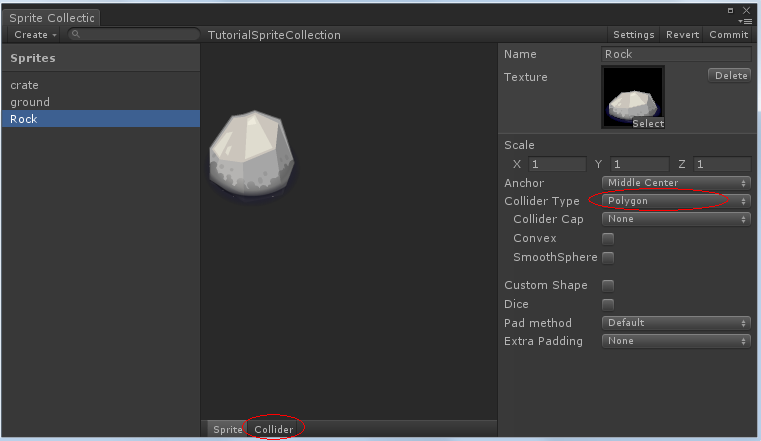
* 关闭精灵集编辑窗口，此时场景中的精灵已经绑定了碰撞区。用该精灵创建和即将用该精灵创建的实例都会绑定碰撞区。



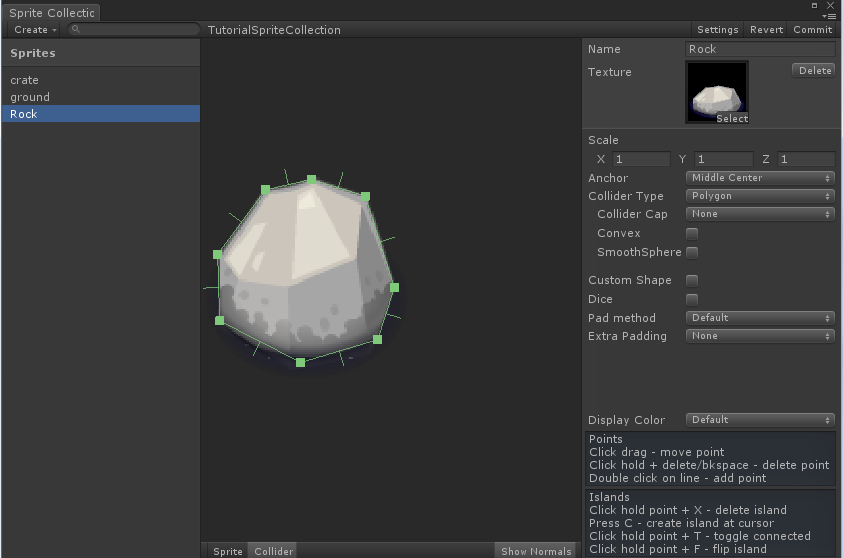
* 1. **创建多边形碰撞区**
* 在场景中添加石头精灵，至于如何创建，可参考前面的教程。

****

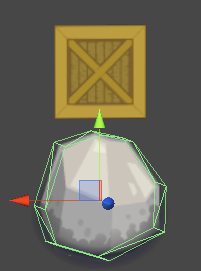
* 这个石头无法用box来精确表现，所以我们将创建多边形碰撞区。选择精灵集合然后点击“Open Editor…”。
* 在编辑器中选择石头。
* 选择Collider Type为“Polygon”。将产生一个box框住贴图，我们来塑造它。
* 切换到“Sprite View”到碰撞区编辑模式。



* 通过点击、拖拽多边形控制点粗略的适配到石头的形状。别忘了滚轮缩放，在mac系统的2手指滚动手势。
* 双击多边形的边可以添加控制点。更多的快捷键列表在右侧面板。
* 重复刚才步骤达到精确匹配这块石头。



* 点击Commit保存。场景中的石头将出现你定义的网格。



* 1. **精灵脚本**

你可以通过代码来控制各种参数，从而控制精灵。本例中将检测按键并改变box的颜色。

* 创建C#文件，命名为TutorialSpriteScript。复制下面代码。

[**TutorialSpriteScript.cs**](http://2dtoolkit.com/doc/tutorial/code/TutorialSpriteScript.cs)

|  |
| --- |
| using UnityEngine;  using System.Collections;  public class TutorialSpriteScript : MonoBehaviour {  tk2dSprite sprite;  // Use this for initialization  void Start () {  sprite = GetComponent<tk2dSprite>();  }  // Update is called once per frame  void Update () {  if (Input.GetKeyDown(KeyCode.A))  {  sprite.color = Color.red;  }  if (Input.GetKeyDown(KeyCode.S))  {  sprite.color = Color.white;  }  }  } |

* 将脚本绑定到精灵上。运行游戏，测试效果。
* 以下代码可以动态的缩放精灵

sprite.scale = new Vector3(xScale, yScale, zScale);

* 以下代码可改变要显示的精灵

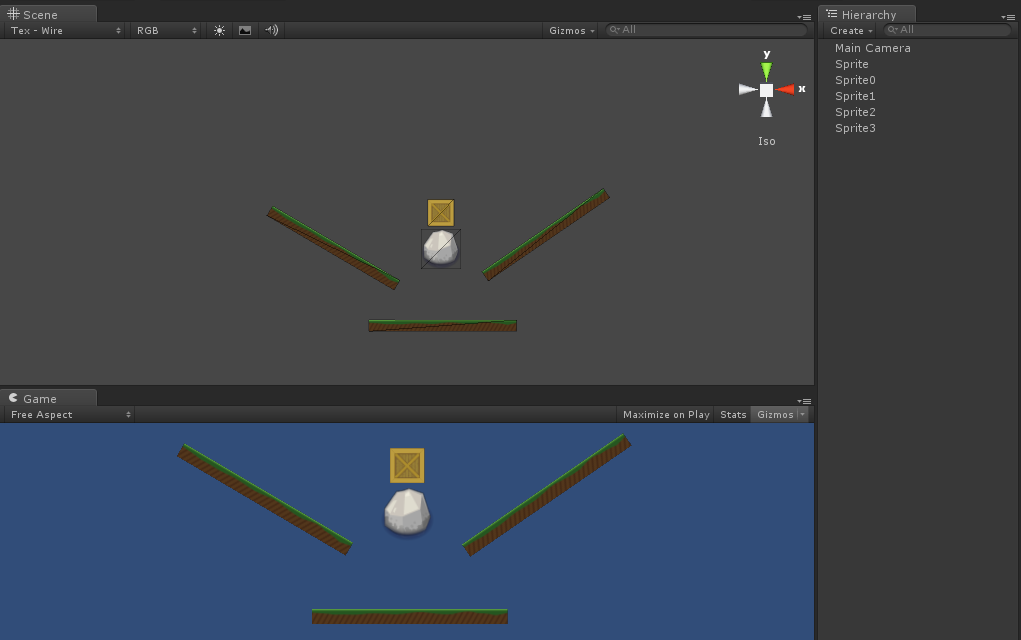
sprite.spriteId = newSpriteId;

* 以下代码可以得到id

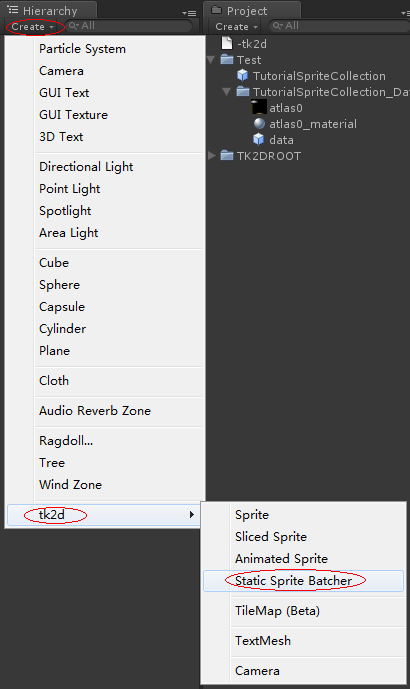
sprite.spriteId = sprite.GetSpriteIdByName("Rock");

1. **静态精灵批处理器（Static Sprite Batcher）以下简称“批处理器”**
   1. **使用批处理器**

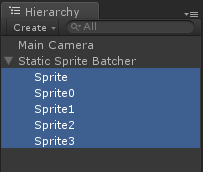
* 首先在场景中建立多个精灵。



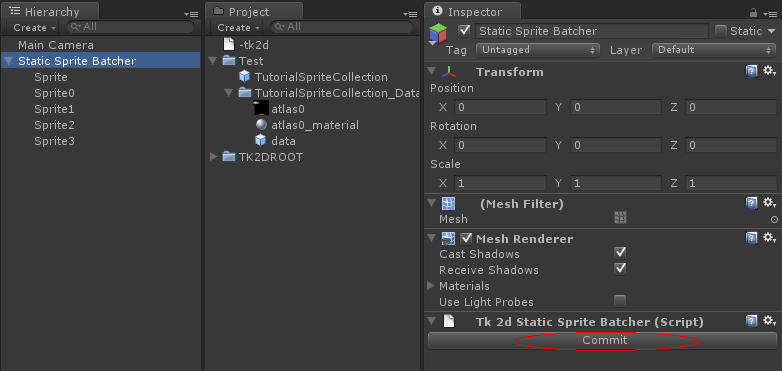
* 随着我们添加很多复杂的精灵，性能将急剧下降。为了避免这种现象我们可以用批处理器。这让你批处理很多数量的精灵（目前仅限于一个精灵集合中的精灵）成为一个大的网格（mesh）。物理碰撞区将被自动拷贝下来。
* 在Hierarchy窗口中点击“Create > tk2d >Static Sprite Batcher” 来创建批处理器。



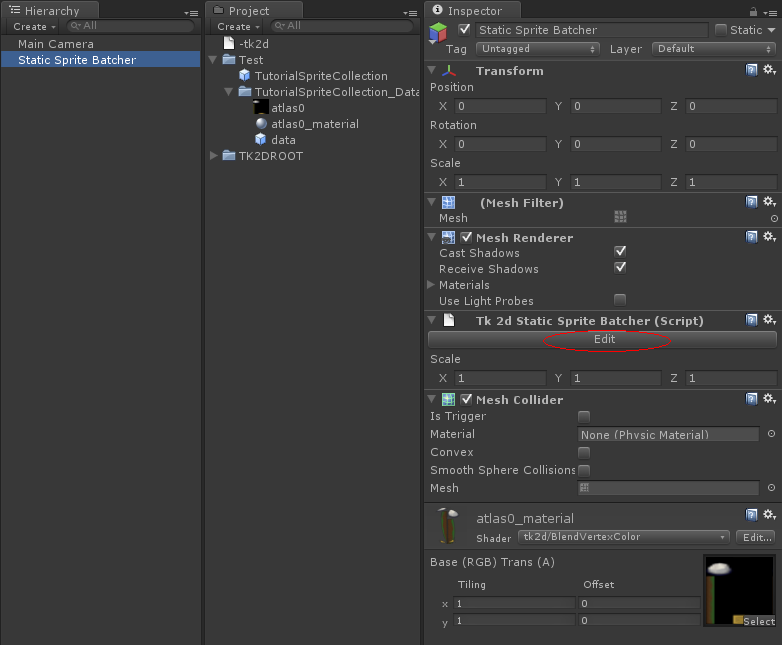
* 将之前创建的精灵拖到Static Sprite Batcher中



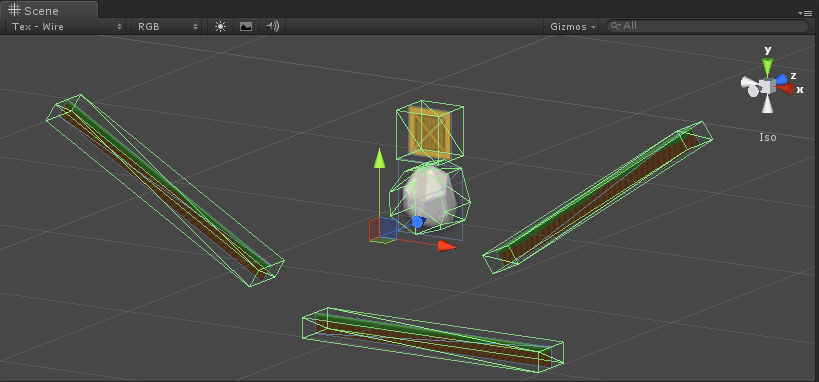
* 然后选中Static Sprite Batcher，点击Inspector界面中的Commit，将看见精灵都被移除了，只剩下Static Sprite Batcher。



* 任何时候想复原，再编辑精灵，点击“Edit”按钮即可。编辑完成后别忘了Commit。



* 1. **关卡创作技巧（Level Creation Tips）**
* 如果关卡贯穿好几屏内容，仅仅用一个批处理器是不够的。unity将描画整个批处理器（即使只有1%可见），换而言之不可见部分也描画了。按逻辑划分它们（建立多个批处理器），保持unity高效的绘制。
* 碰撞区表示了不可通行区域。如下图，别忘了Commit。

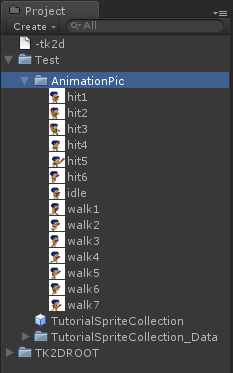


1. **含有动画的精灵（Animated Sprites）**
   1. **创建精灵动画**

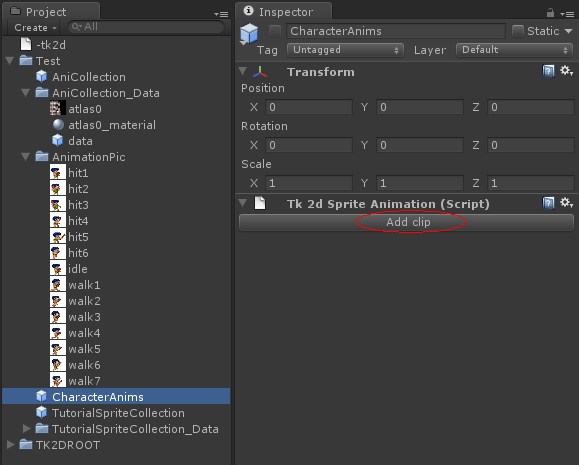
* 我们用额外的精灵举例，资源可以从这里下载

<http://2dtoolkit.com/doc/tutorial/img/charactersprites.zip>

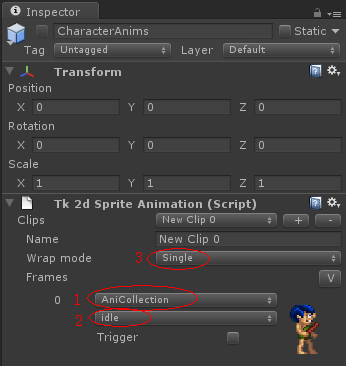
* 将图片放到你的工程中，本例放在\Assets\Test\AnimationPic



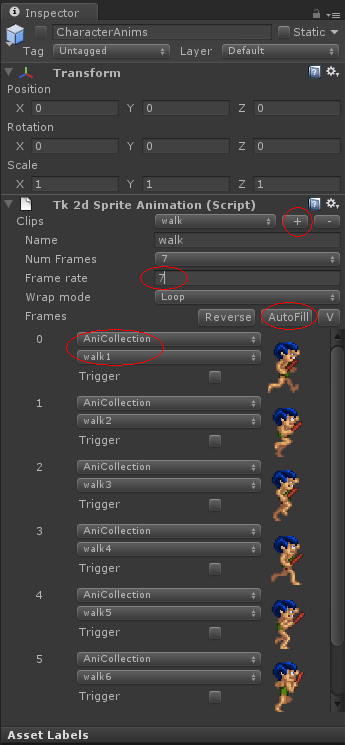
* 创建精灵集，命名为“AniCollection”，然后“Open Editor…”，把图片拖进去。（此处和官方教程不太一样，官方教程的做法实在找不到怎么做的，可能是版本不同）
* 接下来创建动画，在Project窗口选择“Create > tk2d > Sprite Animation”，起名为“CharacterAnims”。
* 选中“CharacterAnims”，然后在Inspector窗口点“Add clip”。



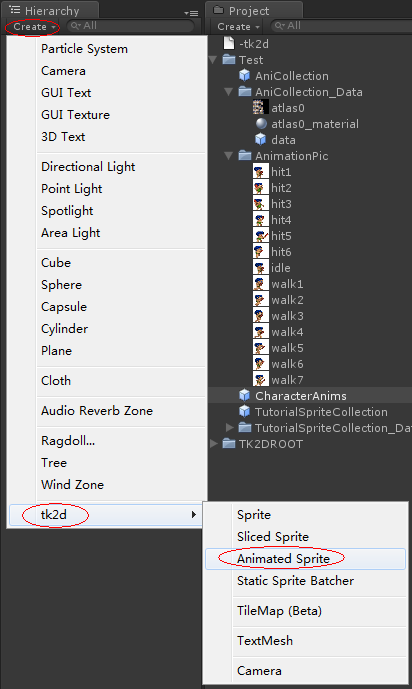
* 创建空闲切片。如下图：



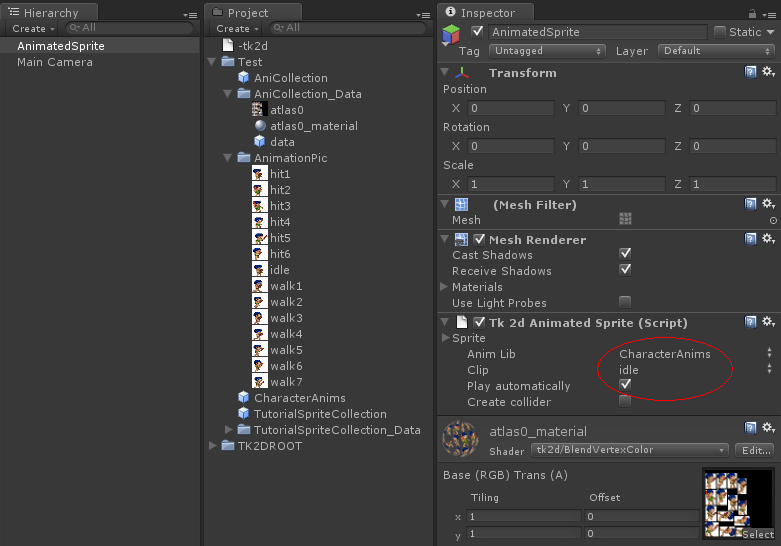
* 为行走循环添加新切片。点击“+”按钮，命名为“walk”。将Frame rate改为7，然后下拉菜单中选择“anicollication”和“walk1”，然后点击“AutoFill”，剩下的所有帧将被自动选择。



* 同样方式添加“hit”动画。
  1. **创建带动画的精灵**
* Hierarchy窗口中点击“Create > tk2d > Animated Sprite”



* 选择刚建立的动画精灵，右侧参数“Play Automatically”为游戏运行时就播放。适合背景动画。本例以idle为例。



* 1. **动画绑定脚本**
* 通过脚本控制各个参数和让它播放各个切片。新建脚本，代码如下：

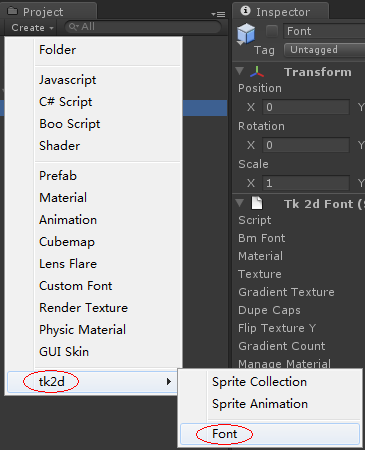
**TutorialAnimController.cs**

|  |
| --- |
| using UnityEngine;  using System.Collections;  public class TutorialAnimController : MonoBehaviour  {  tk2dAnimatedSprite anim;  // Use this for initialization  void Start () {  anim = GetComponent<tk2dAnimatedSprite>();  }  bool walking = false;  void HitCompleteDelegate(tk2dAnimatedSprite sprite, int clipId)  {  if (walking)  {  anim.Play("walk");  }  else  {  anim.Play("idle");  }  }  // Update is called once per frame  void Update ()  {  if (Input.GetKeyDown(KeyCode.A))  {  anim.Play("hit");  anim.animationCompleteDelegate = HitCompleteDelegate;  }    if (Input.GetKeyDown(KeyCode.D))  {  anim.Play("walk");  anim.animationCompleteDelegate = null;  walking = true;  }    if (Input.GetKeyDown(KeyCode.W))  {  anim.Play("idle");  anim.animationCompleteDelegate = null;  walking = false;  }  }  } |

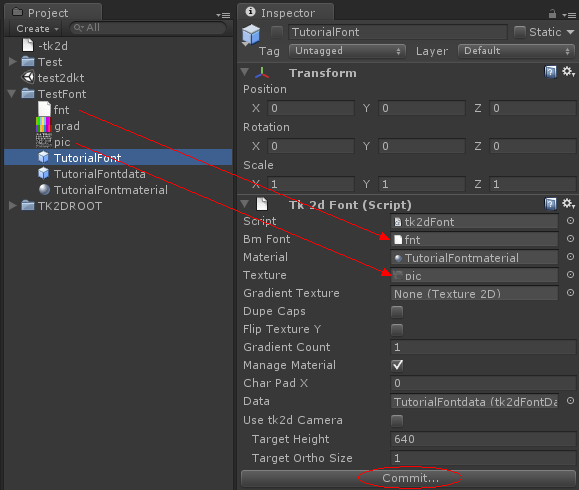
* 运行，AWD键测试效果。

1. **字体（Font）**
   1. **准备字体**

* 2D Toolkit支持xml和text字体格式。唯一限制是它仅支持单页贴图。最大ASCII字符默认为128，如果需要额外字体你可以在Font object增加这个值。
* 用以下配置达到最好效果：
* **BMFont** – 导出选项：32 bit, Channels A - outline, RGB - one (no outline) or glyph (when an outline has been set up).
* **Hiero** - In Gylph Cache, Set page width and height so you end up with 1 page exactly. Increase padding around characters.
* 在Project窗口点击“Create > tk2d > Font” ，最好导出在同一目录（这不是必须的），然后命名为TutorialFont。

****

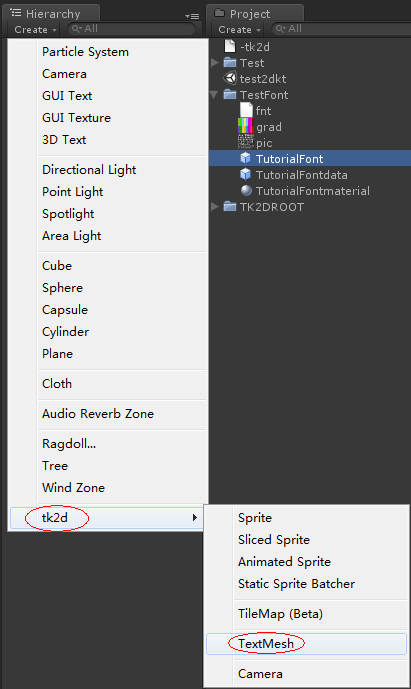
* 给字体对象指定xml/text字体和贴图。自从V1.51后材质不是必须的。如果没有一个可用的，2D Toolkit将自动为你创建。



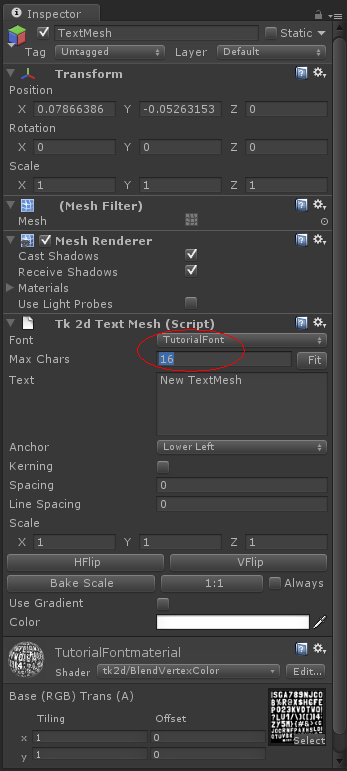
* 点击commit。现在你可以在场景中TextMeshes中使用字体了。
  1. **创建和使用TestMesh**

TextMesh用指定字体描画字符串。

* 在Hierarchy窗口中点击“Create > tk2d > TextMesh” 创建TextMesh。

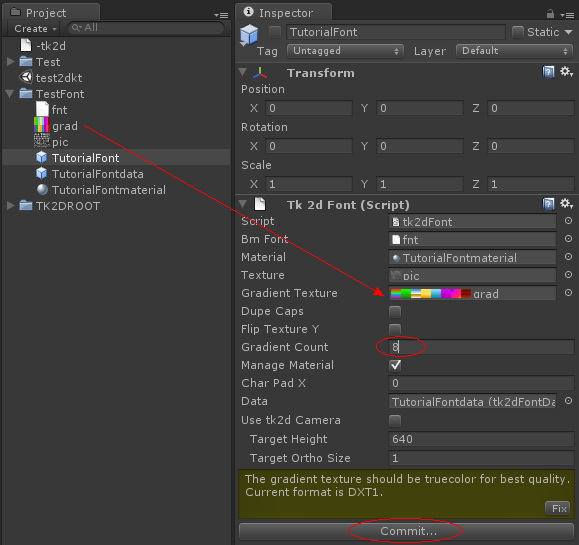
****

* Font下拉菜单中选择我们创建的字体，然后填写想要在TextMesh上显示多少个字符。如果超过这个限制你的字符串将被删减。这里还可调节其他参数。

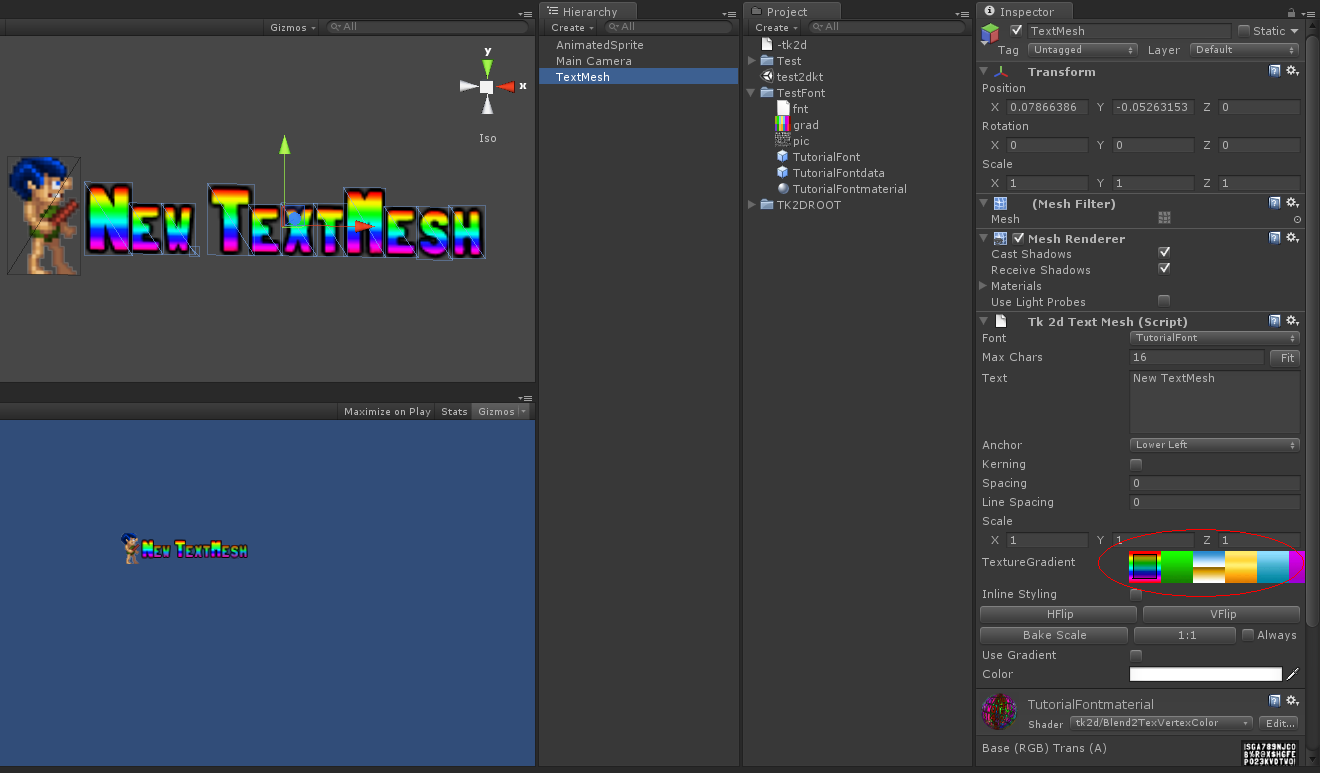
****

* 1. **使用渐变和内部风格的变化**
* 你可以使用渐变贴图产生高质量的渐变。除此之外，2Dtoolkit还支持内部风格的变化：一个TextMesh中渐变。
* 为TutorialFont添加一个渐变。设置渐变块数，本例有8块。这是本例中用的渐变图：

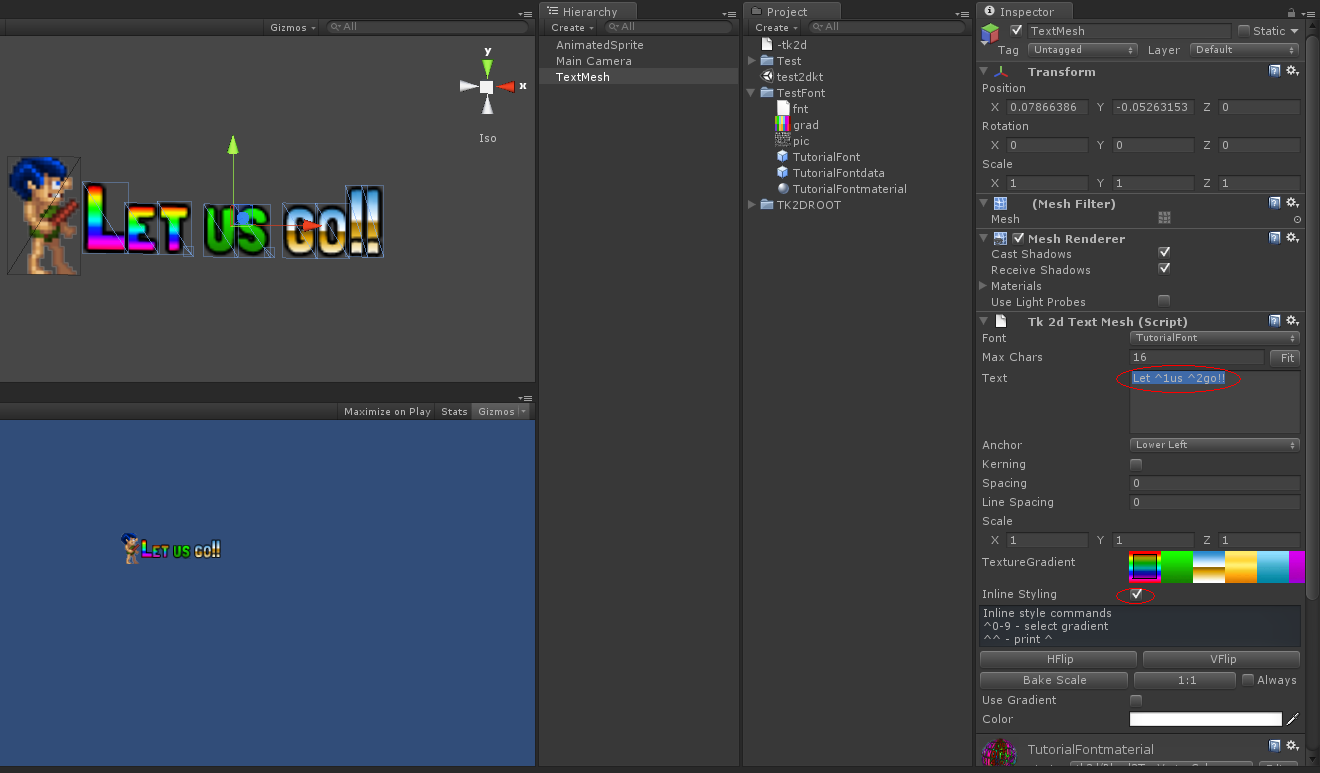
C:\Users\xiangyan.CHAINGUNSOFT\Desktop\TextGradient.png

****

* 点击Commit。任何TextMeshes用这个字体都将有渐变。
* 在场景中创建TextMesh并选择TutorialFont。你将可以选择任意渐变块来给TextMesh着色

****

* 在TextMesh的Inspector界面勾选“Inline Styling”。可以使用内部风格，可挑选前10个渐变。例如输入Let us go!!效果如下：

****

（注意Text的内容中^1 ^2是指定用第几个渐变块的命令，渐变块索引从0开始。）

* 这也可以使用在sprite中，功能很强大。
  1. **TextMesh绑定脚本**
* 本例用Q键来增加分数，代码如下：

[TextMeshExample.cs](http://2dtoolkit.com/doc/tutorial/code/TextMeshExample.cs)

|  |
| --- |
| using UnityEngine;  using System.Collections;  public class TextMeshExample : MonoBehaviour {  tk2dTextMesh textMesh;  int score = 0;  // Use this for initialization  void Start () {  textMesh = GetComponent<tk2dTextMesh>();  }  // Update is called once per frame  void Update () {  if (Input.GetKey(KeyCode.Q))  {  score++;  textMesh.text = "SCORE: " + score.ToString();  // This is important, your changes will not be updated until you call Commit()  // This is so you can change multiple parameters without reconstructing  // the mesh repeatedly  textMesh.Commit();  }  }  } |

* 将脚本绑定到场景中的TextMesh上，执行，测试效果。
* 另外，缩放代码：

textMesh.scale = Vector3(xScale, yScale, zScale);

* 颜色代码：

textMesh.color = Color.red;

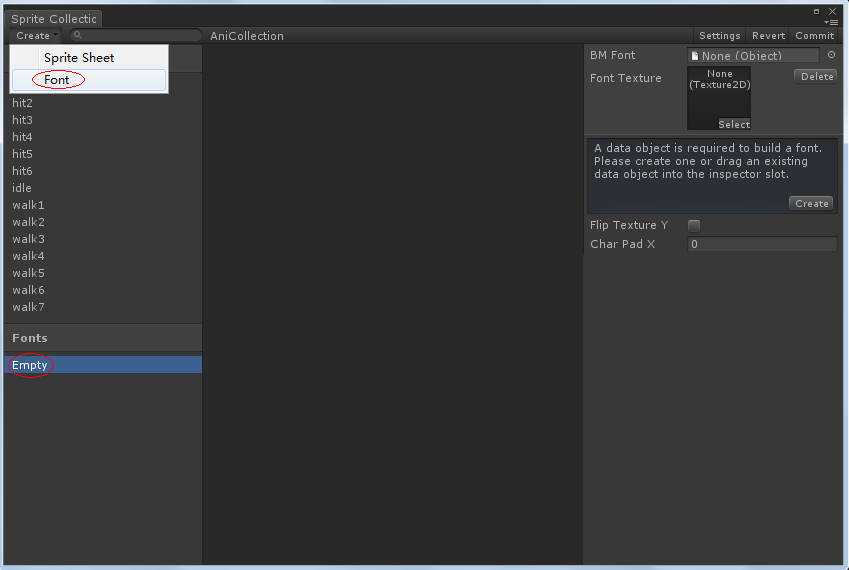
* 如果允许渐变，设置第二颜色如下：

textMesh.color2 = Color.green;

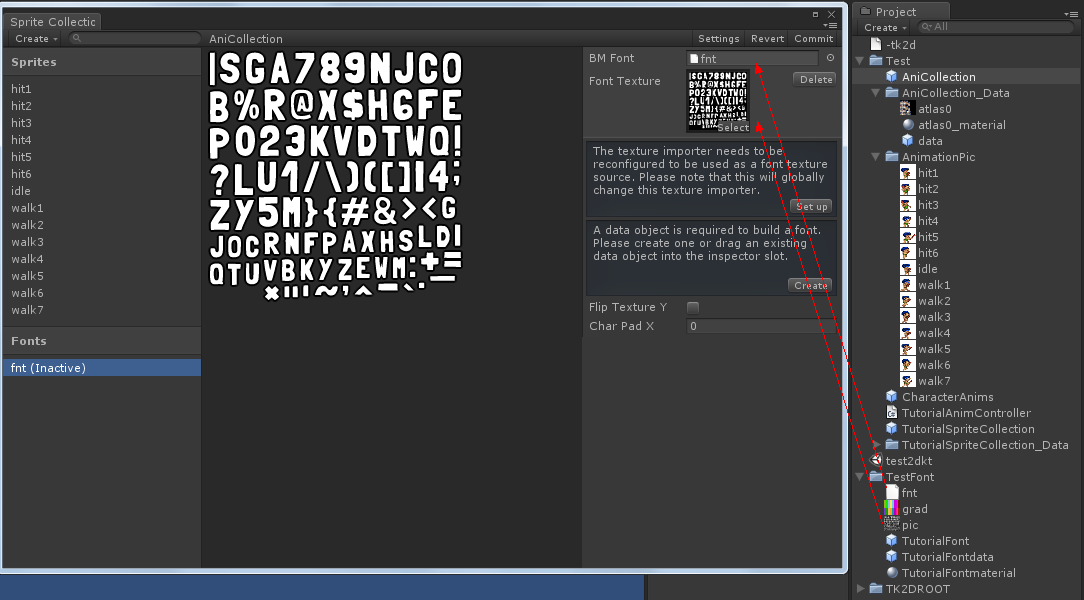
* 注意：虽然你可以在代码中改变Max Chars的值，但是你应该避免这样做，因为运行时会重新分配内存。

1. **高级技巧（Advanced）**
   1. **导入字体到精灵集**

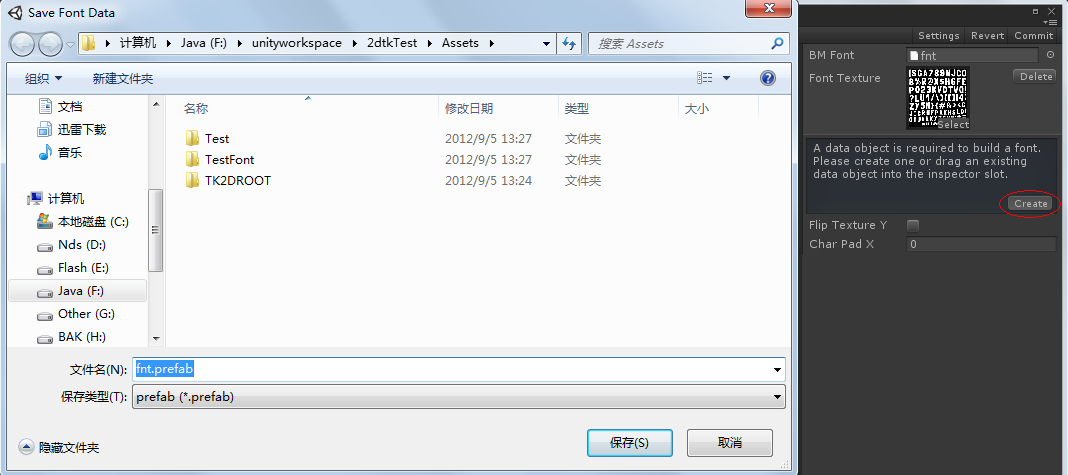
* 你可导入字体到精灵集中以便将来描画。但这种方式不能使用渐变。
* 新建精灵集，并“Open Editor…”，或使用一个现成的精灵集。
* 点击“Create > Font”。
* 我们用之前的小人动画这个精灵集来演示。



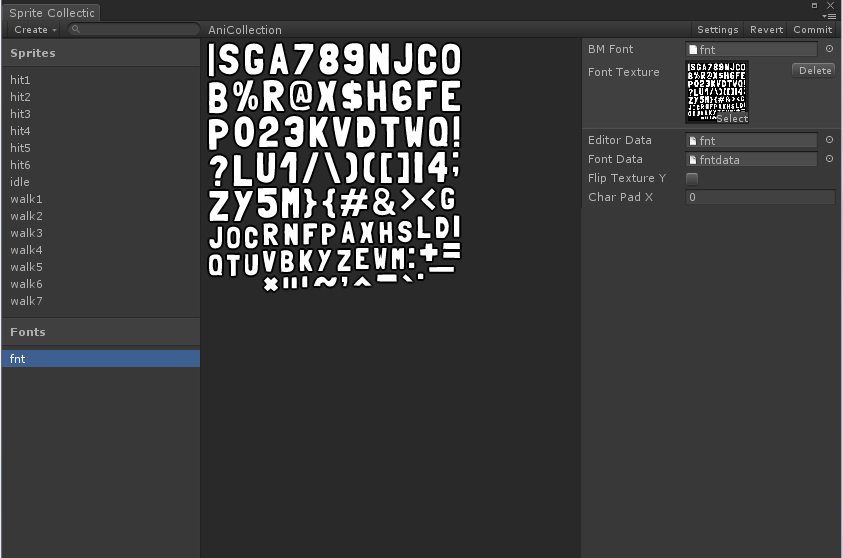
* 拖入bmfont的xml/text文件到“BM Font”。（这里我们拖入之前创建的fnt文件）
* 拖入字体贴图到“Font Textture”。如果格式不对，将有错误提示。



* 记住：这是一个全局操作，这将影响到所有font实例。如果你打算在其他方法中使用相同字体，复制这个来使用将很方便。
* 如果你没有data object将提示新建一个。点击create并选择一个目标文件写入，如下图：

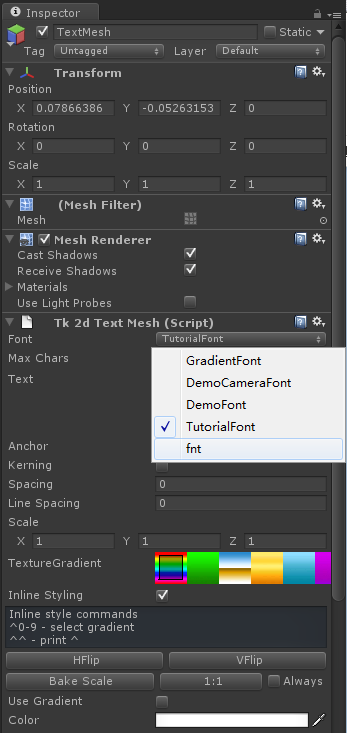


* “保存”后，如下图：

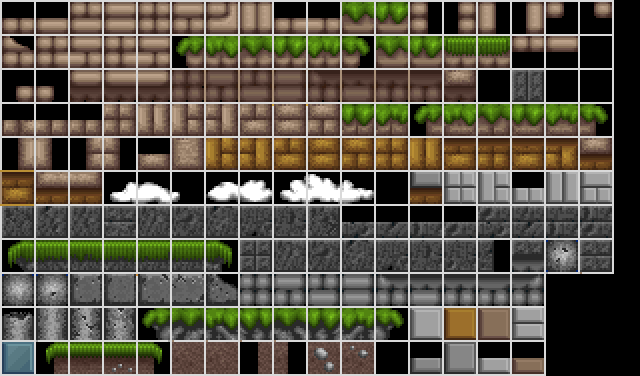


* 点击Commit，就可以使用这个font了。

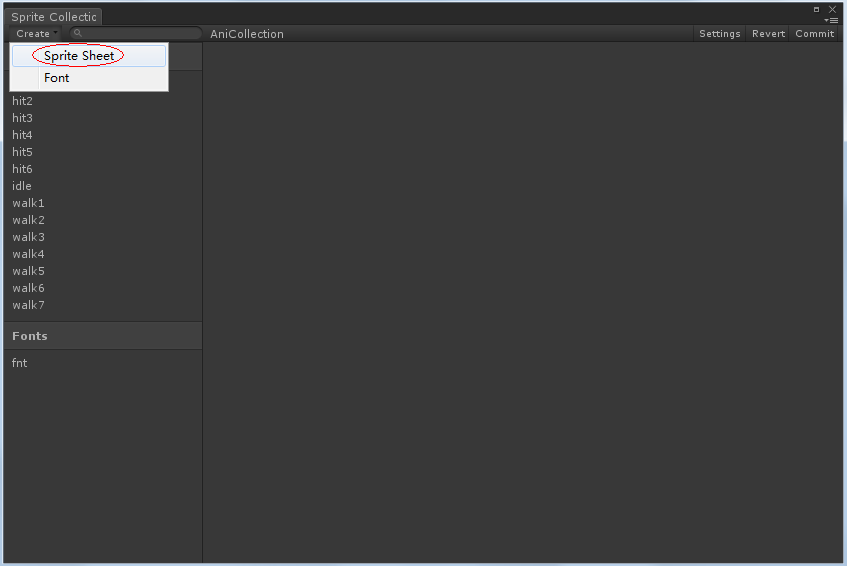
比如选中之前Hierarchy中创建的TextMesh，可在Inspector界面看到;



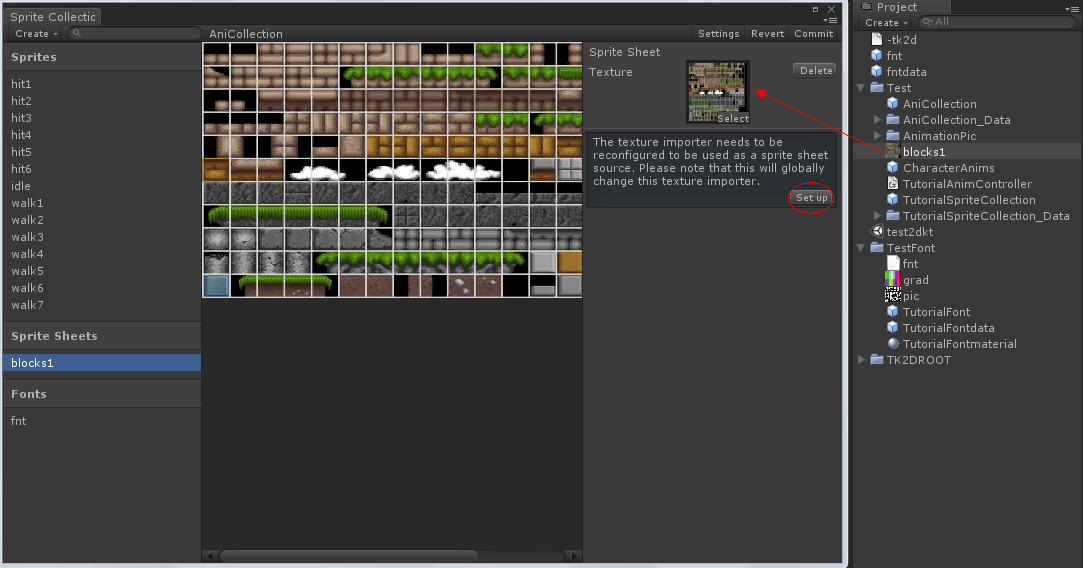
* 1. **导入精灵表（Spritesheet）**
* 你可以导入精灵表到2D Toolkit直接使用。下面来演示。使用这张图：



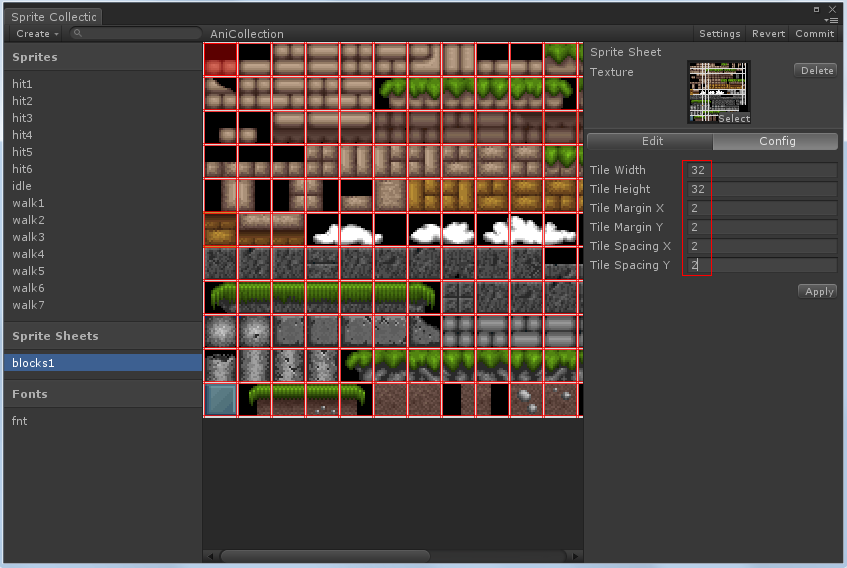
* 新建精灵集，然后“Open Editor…”，或使用现成的精灵集。
* 我们还是以小人动画为例，点击“Create > Sprite Sheet”



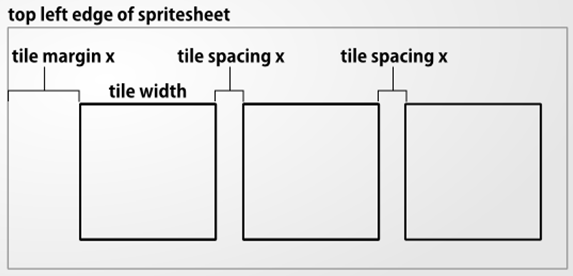
* 拖入贴图，如果没有正确建立，将提示。点击“Set up”让2D Toolkit自动做必要的改变。



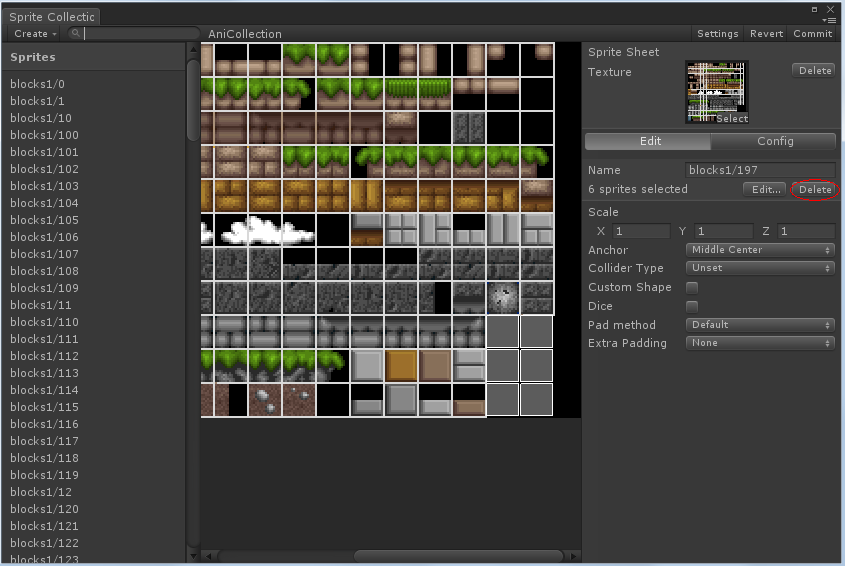
* 设定tile的参数，改变参数后将以红色覆盖切片。如下图：



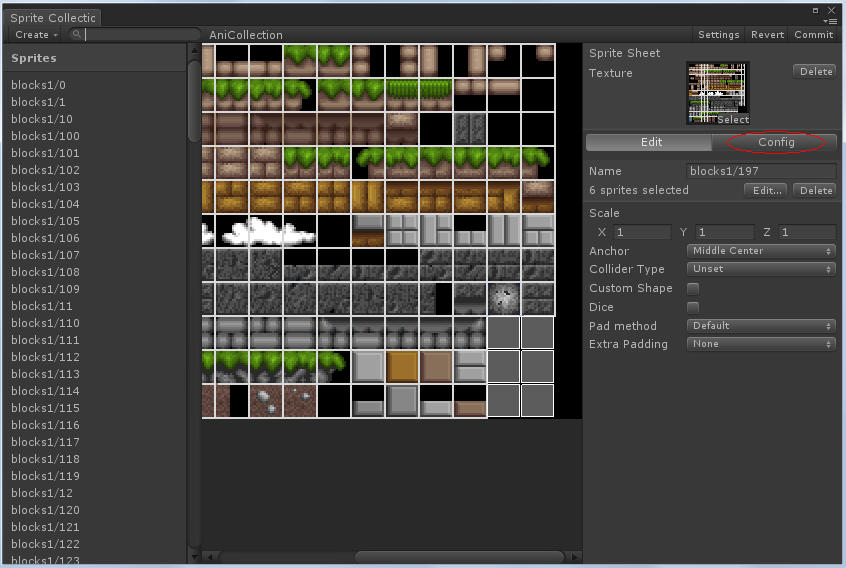
* 图示说明参数意义：



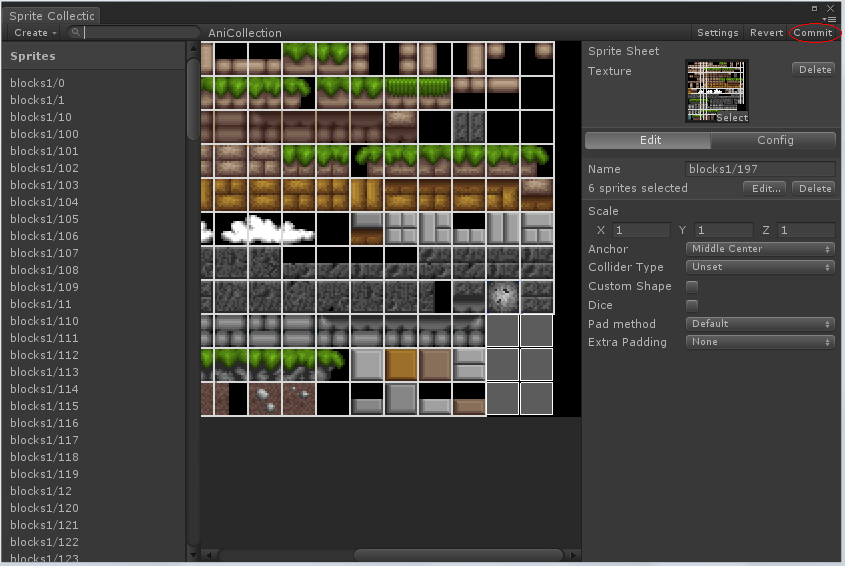
* 参数设置完成后点击“Apply”，编辑器将切割图片。并切换为“编辑模式（Edit Mode）”
* 该模式中，可在精灵表中选择块并单独设置参数。可单选或用Ctrl来复选，或框选。我们可以选中黑色无用的块，并点击“Delete”删掉它们。



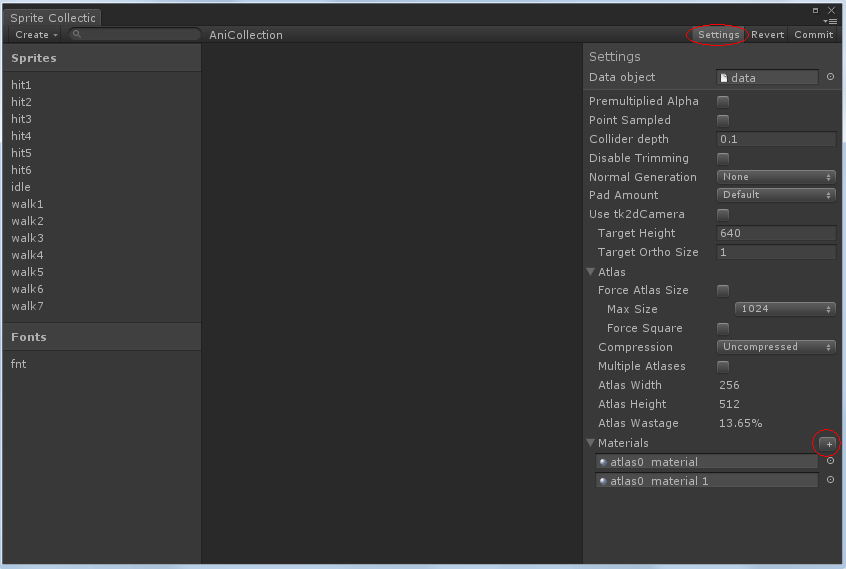
* 这些贴图不会在最终图集中使用。想重新编辑图块的话，切换到“Config”面板就好了。



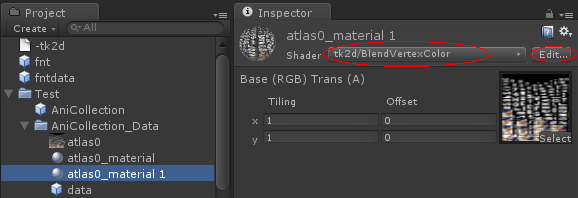
* 若编辑完成点击Commit。便可使用精灵集了。原图的任何改变都会引起重构精灵集。



* 1. **一个精灵集多种材质（Multiple Materials in a Sprite Collection）**
* 某些情况下，将多个精灵合并到一个图集是有益的，这些精灵保持不同的材质。比如，导入切片图块到一张切片地图。它们大多数是实体（可能指整个图块不透明），简单的分配实体着色器到这些精灵将获得显著的GPU收益。
* 打开精灵集编辑器，点击“Setting”

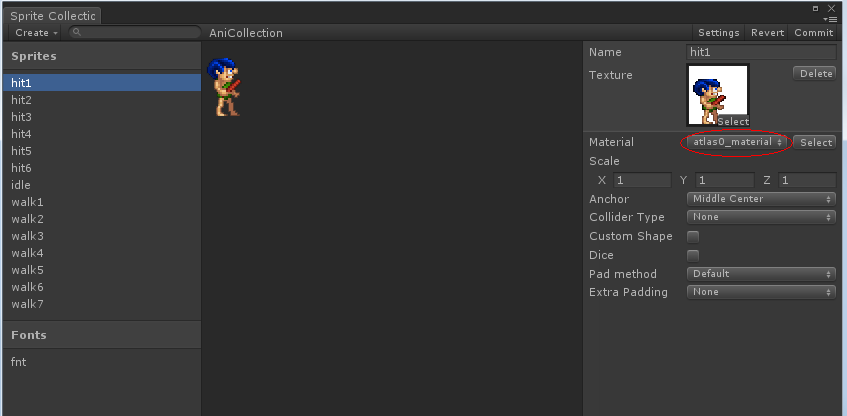


* 点击“+”材质列表。这将复制材质，并将拷贝保存在同一目录。你可以点击材质，改变其着色器。你可以建立任意多个拷贝。

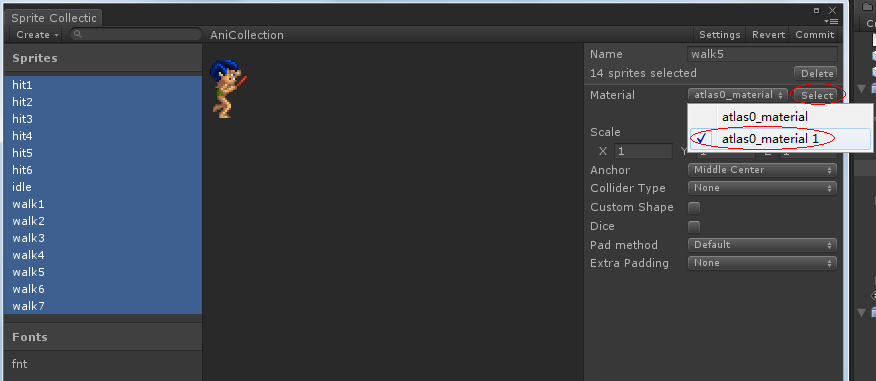


（可选择一些现有着色器，也可点击“Edit…”来编辑代码实现着色器效果）

* 在Project界面选中新建的材质，点击delete键，便可删除你复制的材质。
* 选中精灵，你可以看到Material下拉菜单。可以选择精灵材质。

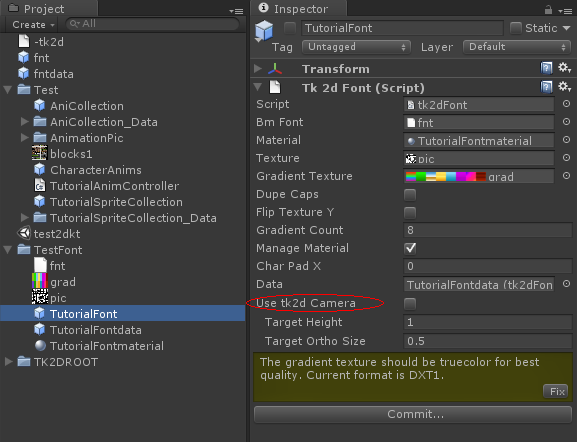


* 点击“Select”按钮，可以将有相同材质的精灵都选中。然后可以批量编辑材质，然后点击新出现的“Apply”按钮来确认改变。

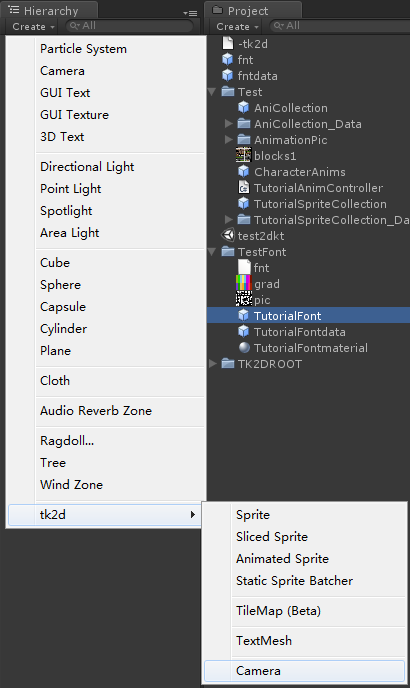


1. **摄像机（Camera）**
   1. **tk2dCamera**

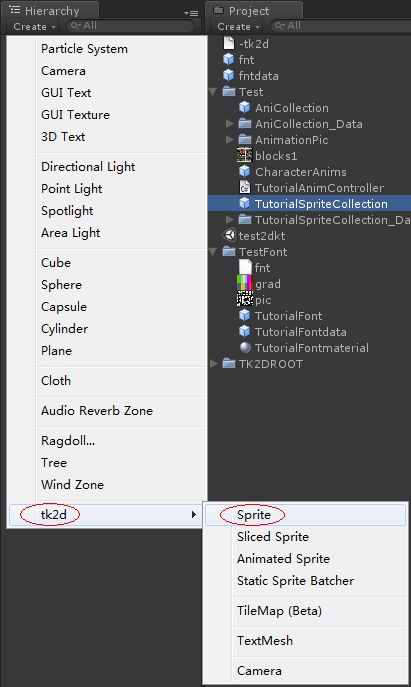
* tk2dCamera是使之以像素为单位工作的很好的解决方案，并在无用户干预的情况下自动适配各种分辨率。
* 为了使精灵无缝的与tk2dCamera工作，需要在Inspector界面勾选“Use tk2d Camera”项。（选中精灵时没找到，选中font时有这个选项）



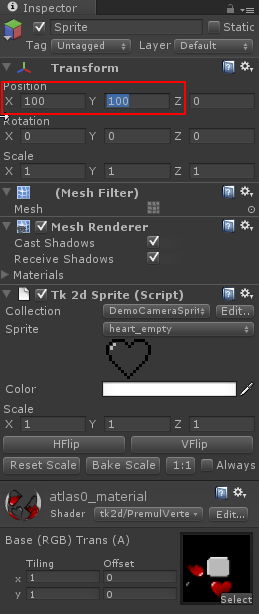
* 请记住，tk2dCamera与旧方法建立的精灵也兼容。勾选“Use tk2d Camera”选框来创建的精灵不会在标准正交摄像机中自动调整大小。
* 在Project窗口点击“Create > tk2d > Camera”来创建tk2dCamera，也可在菜单中“Assets > Create > tk2d > Camera”来创建。



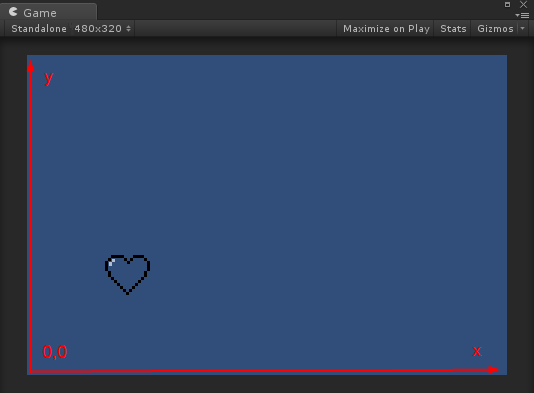
* 如果你打算把这个作为游戏的主摄像机，请确保场景中没有其他摄像机。
* 添加精灵



添加一个精灵到场景中。设置精灵集合为DemoCameraSpriteCollection。

把精灵放到100，100位置

无论屏幕分辨率，精灵将被放在100，100



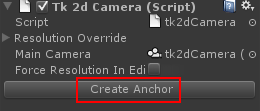
有事改变分辨率后，不会马上刷新，请按播放，停止就好了。

锚点

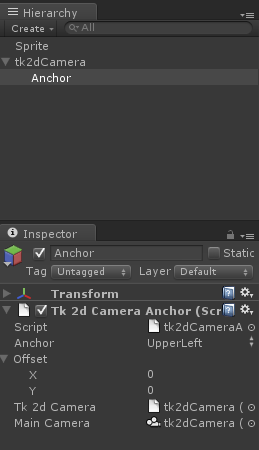
Tk2d摄像机支持锚点。支持9个点“田”。你也可以指定锚点的像素偏移。锚点坐标系统和摄像机坐标系统一致。

选择tk2dCamera

在inspector界面点击create anchor，来创建锚点。

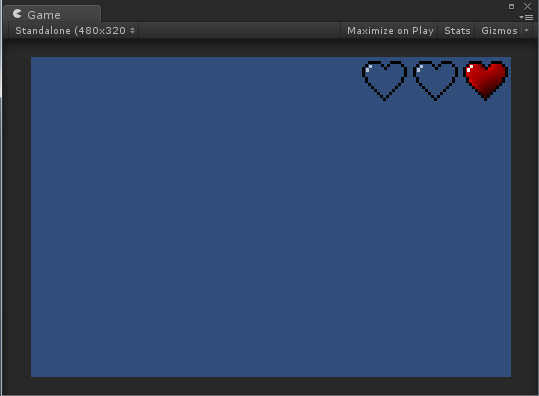


设置锚点，如下图



任何sprites父化到这个锚点，将被固定到锚点的相对位置。下面的心固定到top right 锚点，

480\*320



320\*240



7.Tile Map教程 beta版

教程是草稿版，未翻译。