
UGUI 全面实践教程

(内部资料 1.0)



大学霸

www.daxueba.net



前言

UGUI 是 Unity 官方推出的最新 UI 系统。它从 Unity 4.6 开始，被集成到 Unity 的编辑器中。相较于旧的 UI 系统，它绝对属于一个巨大的飞跃！因为只要有旧 UI 系统使用体验的开发者，大部分都对它没有任何好感，以至于在过去的很长一段时间里，大家都在使用资源商店（Asset Store）里，由第三方开发者开发的付费插件 NGUI，实现游戏中与 UI 相关的功能部分。现在，Unity 官方隆重地推出了新的 UI 系统，大有与 NGUI 争锋的势头。它作为新的系统，被长期放置在 Unity 官网的推广页面上宣传。

新的 UI 系统，有以下三个优点：

- ☐ 灵活
- ☐ 快速
- ☐ 可视化

这对于开发者而言，带来的好处是：

- ☐ 运行效率高，执行效果好
- ☐ 易于使用，方便扩展（新 UI 系统代码开源）
- ☐ 与 Unity 的兼容性高

本书首先带领读者从基础和概念上认识新的 UI 系统，然后全面讲解 UGUI 的交互控件、自动布局、特效、Canvas、事件触发等内容。掌握这些内容后，读者不仅可以轻松分析官方实例，还可以使用 UGUI 做出精密的界面。

1. 学习所需的系统和软件

- ☐ 安装 Windows 7 操作系统
- ☐ 安装 Unity 4.6

2. 学习建议

大家学习之前，可以致信到 xxxxxxxxxx，获取相关的资料 and 软件。如果大家在学习过程遇到问题，也可以将问题发送到该邮箱。我们尽可能给大家解决。

目 录

第 1 章 新 UI 系统概述.....	1
1.1 优点.....	1
1.1.1 创建速度快.....	1
1.1.2 直观、易于使用.....	2
1.1.3 效率高效果好.....	3
1.2 开源.....	3
1.2.1 开源授权协议——MIT/X11.....	3
1.2.2 源代码托管网站——BitBucket.....	4
1.2.3 查看源代码项目.....	5
第 2 章 UI 系统基础.....	8
2.1 画布——Canvas.....	8
2.1.1 创建 Canvas 对象.....	8
2.1.2 UI 控件的绘制顺序.....	9
2.1.3 绘制模式.....	10
2.2 UI 控件的布局.....	13
2.2.1 Rect Tool 小工具.....	13
2.2.2 Rect Transform 组件.....	15
2.3 提供可视功能的 UI 组件.....	21
2.3.1 显示文字——Text(Script).....	21
2.3.2 显示图片 1——Image(Script).....	25
2.3.3 显示图片 2——Raw Image(Script).....	30
2.3.4 遮罩效果——Mask(Script).....	31
第 3 章 可交互的 UI 控件.....	34
3.1 Selectable 类.....	34
3.1.1 是否可交互——Interactable.....	35
3.1.2 状态转换——Transition.....	35
3.1.3 状态导向——Navigation.....	38
3.2 Button——按钮控件.....	42
3.2.1 Button 及其子对象.....	43
3.2.2 Button(Script).....	43
3.3 Toggle——开关控件.....	44
3.3.1 Toggle 及其子对象.....	44
3.3.2 Toggle(Script).....	46
3.3.3 Toggle Group.....	46
3.4 Slider——滑块控件.....	49
3.4.1 Slider 及其子对象.....	49
3.4.2 Slider(Script).....	51
3.5 Scrollbar——滚动条控件.....	52

3.5.1	Scrollbar 及其子对象	52
3.5.2	Scrollbar(Script)	53
3.6	InputField——文本框控件	55
3.6.1	InputField 及其子对象	55
3.6.2	InputField(Script)	56
3.7	Scroll Rect——滚动矩形控件	57
3.7.1	滚动矩形控件的构建和使用示例	57
3.7.2	Scroll Rect(Script)	62
第 4 章	自动布局与特效	64
4.1	自动布局——Auto Layout	64
4.2	布局元素——Layout Elements	64
4.2.1	Layout Properties 视图	65
4.2.2	Layout Element 组件	66
4.2.3	布局规则	67
4.3	布局控制——Layout Controllers	67
4.3.1	Content Size Fitter 组件	68
4.3.2	Aspect Ratio Fitter 组件	69
4.3.3	Horizontal Layout Group 组件	71
4.3.4	Vertical Layout Group 组件	73
4.3.5	Grid Layout Group 组件	75
4.4	使用示例——依据内容自动缩放的按钮	79
4.5	UI 特效组件	81
4.5.1	Shadow 组件	81
4.5.2	Outline 组件	83
第 5 章	Canvas 相关组件	86
5.1	Canvas Scaler 组件	86
5.1.1	像素大小固定——Constant Pixel Size	86
5.1.2	与游戏屏幕一同变化——Scale With Screen Size	87
5.1.3	物理尺寸固定——Constant Physical	88
5.1.4	作为场景中的立体对象——World	89
5.2	Canvas Group 组件	89
5.3	Canvas Renderer 组件	91
5.4	游戏示例——多分辨率下的 UI 布局的一致性	91
5.4.1	环境搭建	91
5.4.2	使用 anchors	92
5.4.3	使用 Canvas Scaler 组件	94
5.4.4	示例效果展示	97
第 6 章	事件触发	98
6.1	Event System 组件	98
6.2	Standalone Input Module 组件	99
6.3	Touch Input Module 组件	99

6.4	游戏示例——UI 屏幕切换	100
6.4.1	实现思路	100
6.4.2	制作 UI 屏幕	101
6.4.3	制作、添加动画效果	102
6.4.4	编写脚本——ScreenManager	113
6.4.5	脚本代码说明	116
6.4.6	游戏效果展示	117
第 7 章	官方示例及其简要说明	122
7.1	常见控件示例——Controls.....	122
7.2	拖拽效果示例——Drag And Drop	124
7.3	可拖动的面板示例——Draggable Panel.....	124
7.4	界面布局示例——Layout Groups	126
7.5	UI 与光照示例——Lighting	127
7.6	游戏菜单示例——Menu 3D	128
7.7	实时纹理绘制示例——RenderTexture	129
7.8	综合性的游戏示例——Survival Shooter	131
7.9	事件系统	133
7.10	学以致用	134

第 1 章 新 UI 系统概述

新的 UI 系统相较于旧的 UI 系统而言，是一个巨大的飞跃！有过旧 UI 系统使用体验的开发者，大部分都对它没有任何好感，以至于在过去的很长一段时间里，大家都在使用资源商店（Asset Store）里，由第三方开发的付费插件 NGUI，实现游戏中与 UI 有关的部分。现在，Unity 官方隆重的推出了新的 UI 系统，大有与 NGUI 争锋的势头，如图 1-1 所示。为此很多开发者都对这个新的 UI 系统产生了兴趣，打算一探究竟！本章就是在此大环境之下应运而生的。

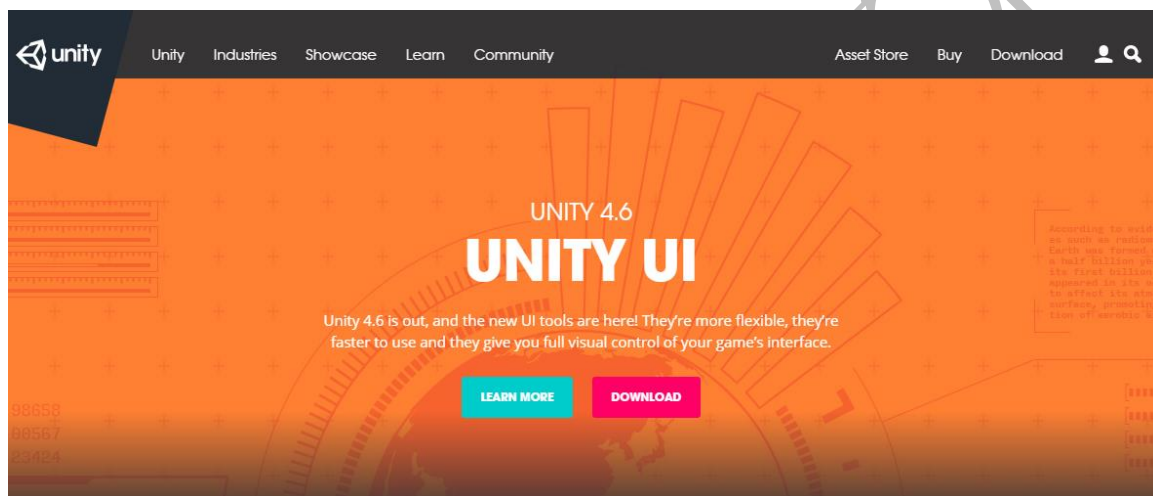


图 1-1 新 UI 系统

1.1 优点

新的 UI 系统是从 Unity 4.6 开始，被集成到 Unity 的编辑器中的。Unity 官方给这个新的 UI 系统赋予的标签是：灵活、快速和可视化！简单来说，对于开发者而言，就是有三个优点：效率高效果好，易于使用、扩展，以及与 Unity 的兼容性高（毕竟是 Unity 官方集成到 Unity 编辑器中的新系统）。本节就来对这些“标签”和“优点”展开进一步的说明和展示！

1.1.1 创建速度快

在不使用任何代码的前提下，就可以简单而快速的在游戏中建立起一套 UI 界面。这在过去绝对是不可想象的，但是新的 UI 系统确实做到了这一点。因为 Unity 预定义了很多常见的 UI 控件，它们是以“游戏对象”的形式存在于游戏的场景中的。在 Unity 中，单击 GameObject|UI 命令，即可看到这些预定义的 UI 控件，如图 1-2 所示。

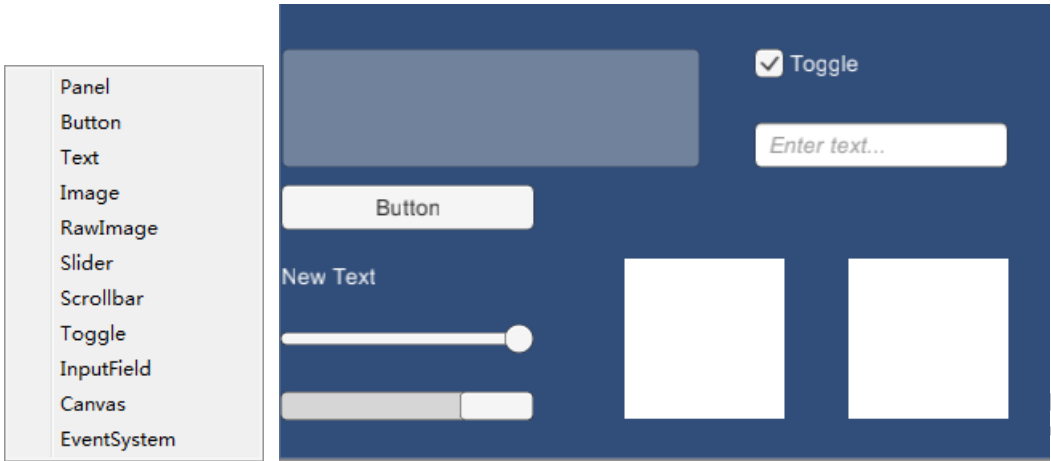


图 1-2 Unity 预定义的 UI 控件

可知，预定义的 UI 控件涵盖了大部分读者所熟知的 UI 控件，如按钮控件、开关控件、滑块控件、滚动条控件和文本框控件等等。

1.1.2 直观、易于使用

对于 UI 控件，开发者可以直接使用鼠标在 Scene 视图里编辑它们的大小、位置和旋转角度，而无需编写任何代码，以 Button 为例，如图 1-3、图 1-4 和图 1-5 所示。

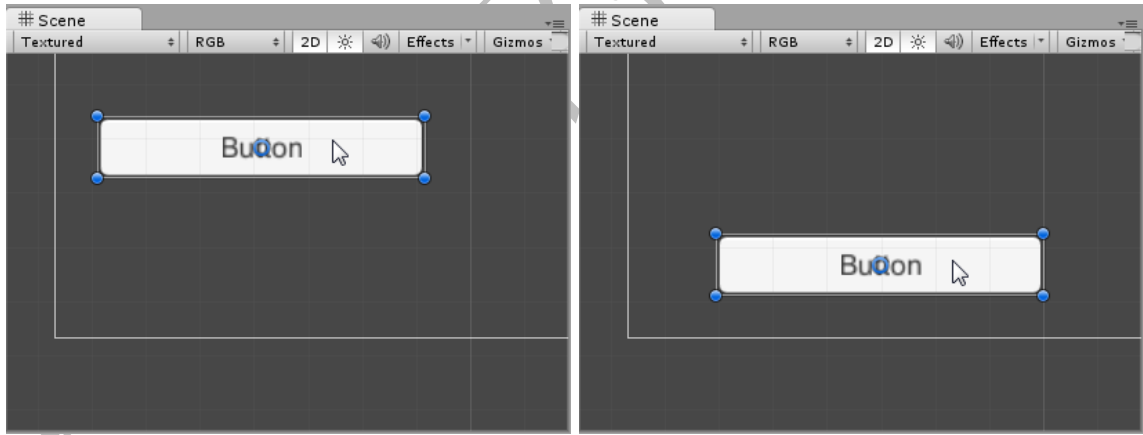


图 1-3 使用鼠标编辑 UI 控件的位置（以 Button 为例）

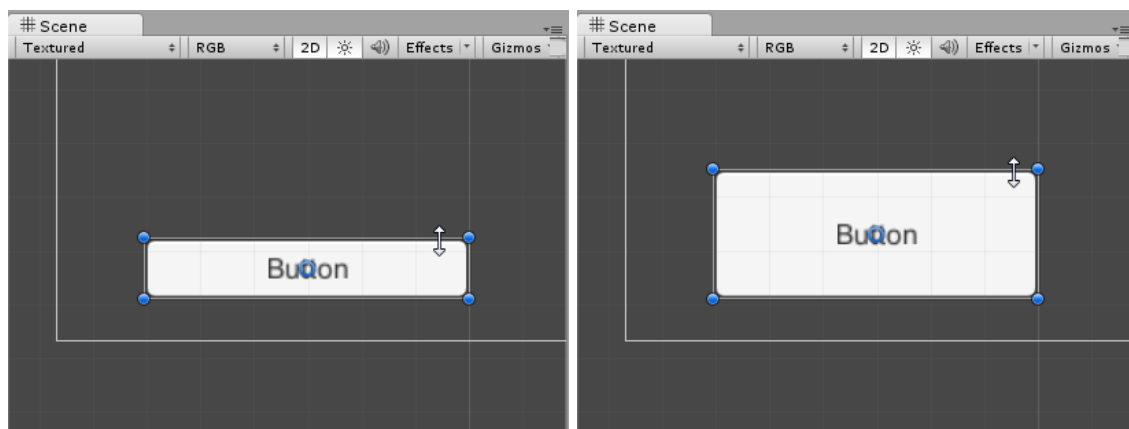


图 1-4 使用鼠标编辑 UI 控件的大小（以 Button 为例）

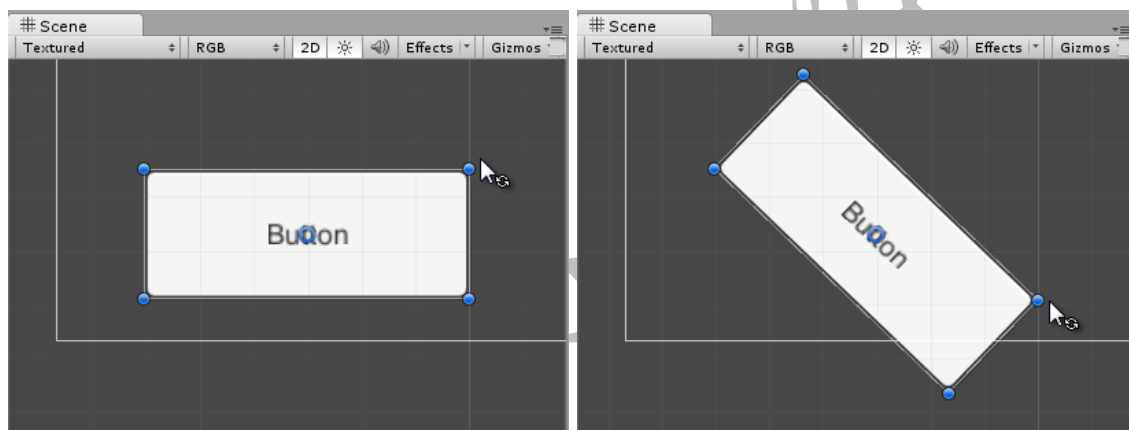


图 1-5 使用鼠标编辑 UI 控件的旋转角度（以 Button 为例）

1.1.3 效率高效果好

通过对批处理（batching）、纹理图集（texture atlasing）和新的 canvas 组件的支持，新 UI 系统提供了一个经过优化的解决方案，使得开发者添加到游戏中的 UI 能够快速的被 GPU 执行（绘制）。而且在 Unity 支持的所有硬件平台上，Draw Call 可以降的很低，与此同时效率与效果却依然能够维持在很高的水平。

1.2 开源

新 UI 系统是开源的，所以开发者可以看到新 UI 系统实现的源码，并加以修改和使用。

1.2.1 开源授权协议——MIT/X11

Unity 所搭载的新 UI 系统,是在开源授权协议 MIT/X11(如图 1-6 所示)之下被公开的!所以开发者可以查看、修改其中的源码,来为己所用!

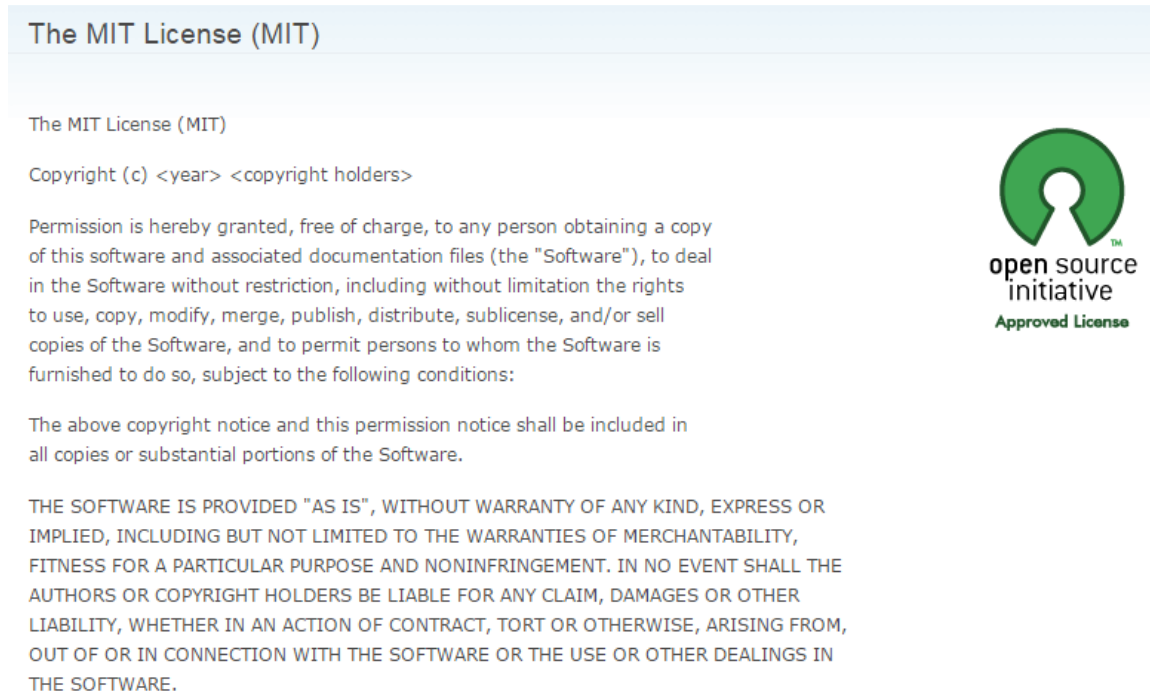


图 1-6 MIT/X11 开源授权协议

提示: MIT/X11 开源授权协议的查看网址是: <http://opensource.org/licenses/MIT>。

1.2.2 源代码托管网站——BitBucket

新 UI 系统的源代码被托管在 BitBucket 上,读者可以从此网站上下载源码,网址如下,打开的页面如图 1-7 所示。

<https://bitbucket.org/Unity-Technologies/ui>

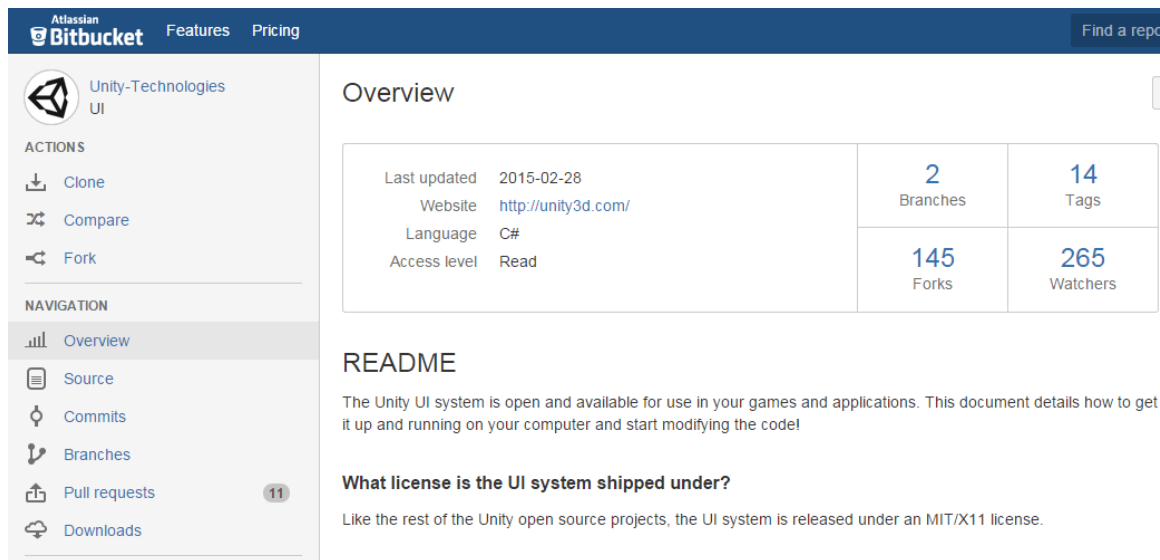


图 1-7 新 UI 系统源代码托管网站 BitBucket 的页面

单击页面左侧列表中的 Downloads 选项以后，进入的页面如图 1-8 所示，单击 Download repository 链接即可开始源代码的下载，下载到的源码压缩包，及其中的文件和文件夹，如图 1-9 所示。

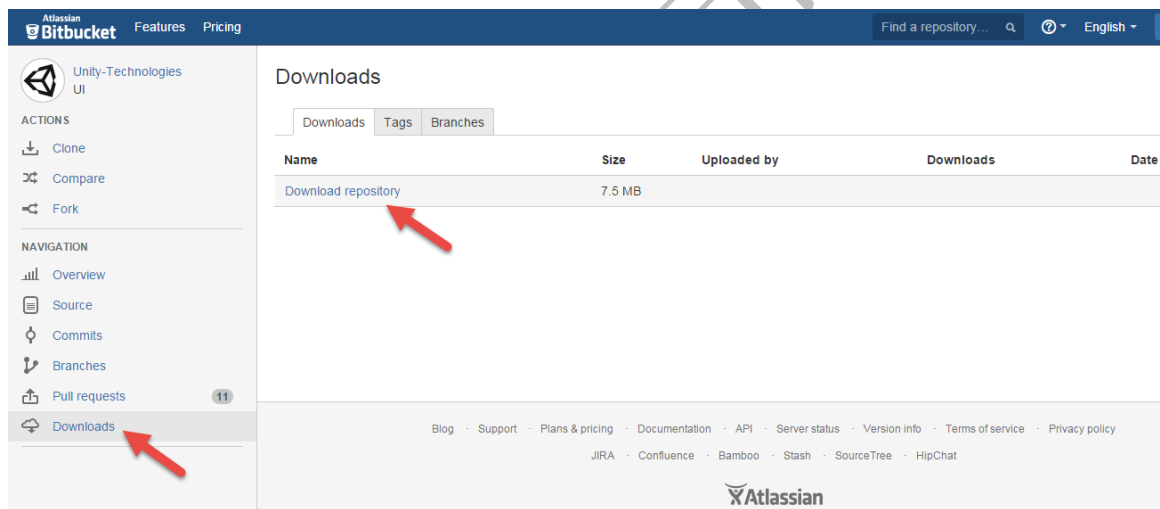


图 1-8 源代码的下载页面

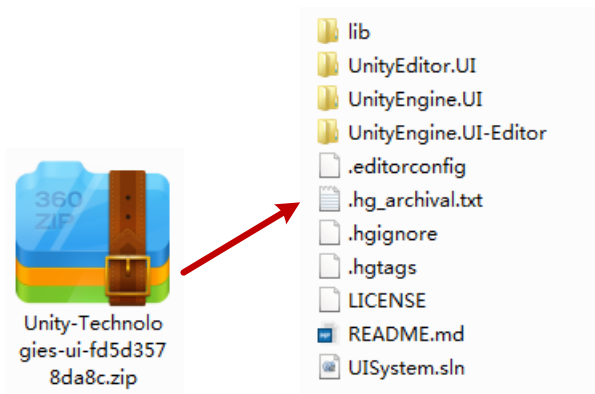


图 1-9 源码压缩包，及其中的文件和文件夹

1.2.3 查看源代码项目

读者可以使用 Visual Studio（微软提供的代码编辑器）或者 MonoDevelop（Unity 自带的代码编辑器），打开新 UI 系统源代码项目。以 MonoDevelop 为例，选中从源码压缩包中解压出来的文件 UISystem.sln，右击鼠标从打开方式中选择 MonoDevelop，如图 1-10 所示。

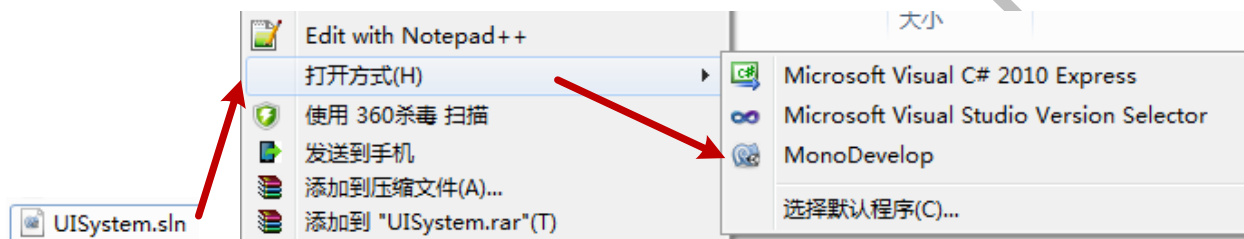


图 1-10 使用 MonoDevelop 打开新 UI 系统源代码项目

在 MonoDevelop 中，被打开的项目结构如图 1-11 所示。此项目中就包含了新 UI 系统的所有源代码，开发者可以任意查看和修改。

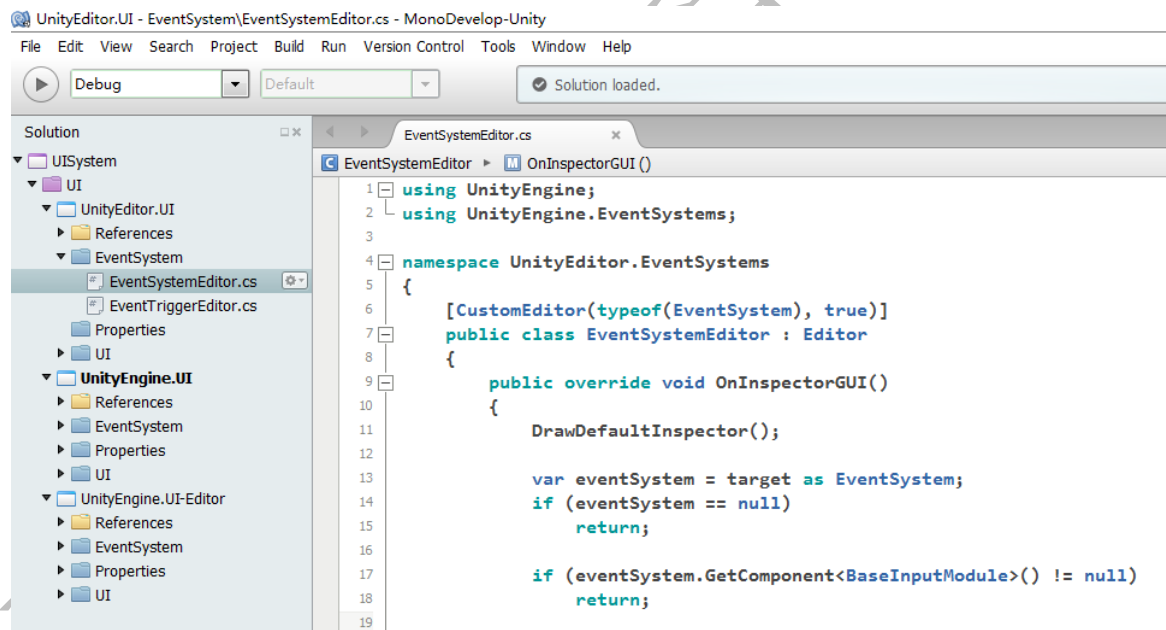


图 1-11 在 MonoDevelop 中，新 UI 系统的项目结构

提示：有关此源代码项目的更多介绍，读者可以查看源代码压缩包解压出来的文件 README.md。此文件是以 md 为后缀的文本文件，在 Windows 平台上，推荐使用 MarkdownPad 对其进行查看和编辑（使用免费的版本），其官方软件的下载网址如下，页面如图 1-12 所示。另外，主页上也显示了此软件打开 md 文件的效果图。

<http://www.markdownpad.com/>

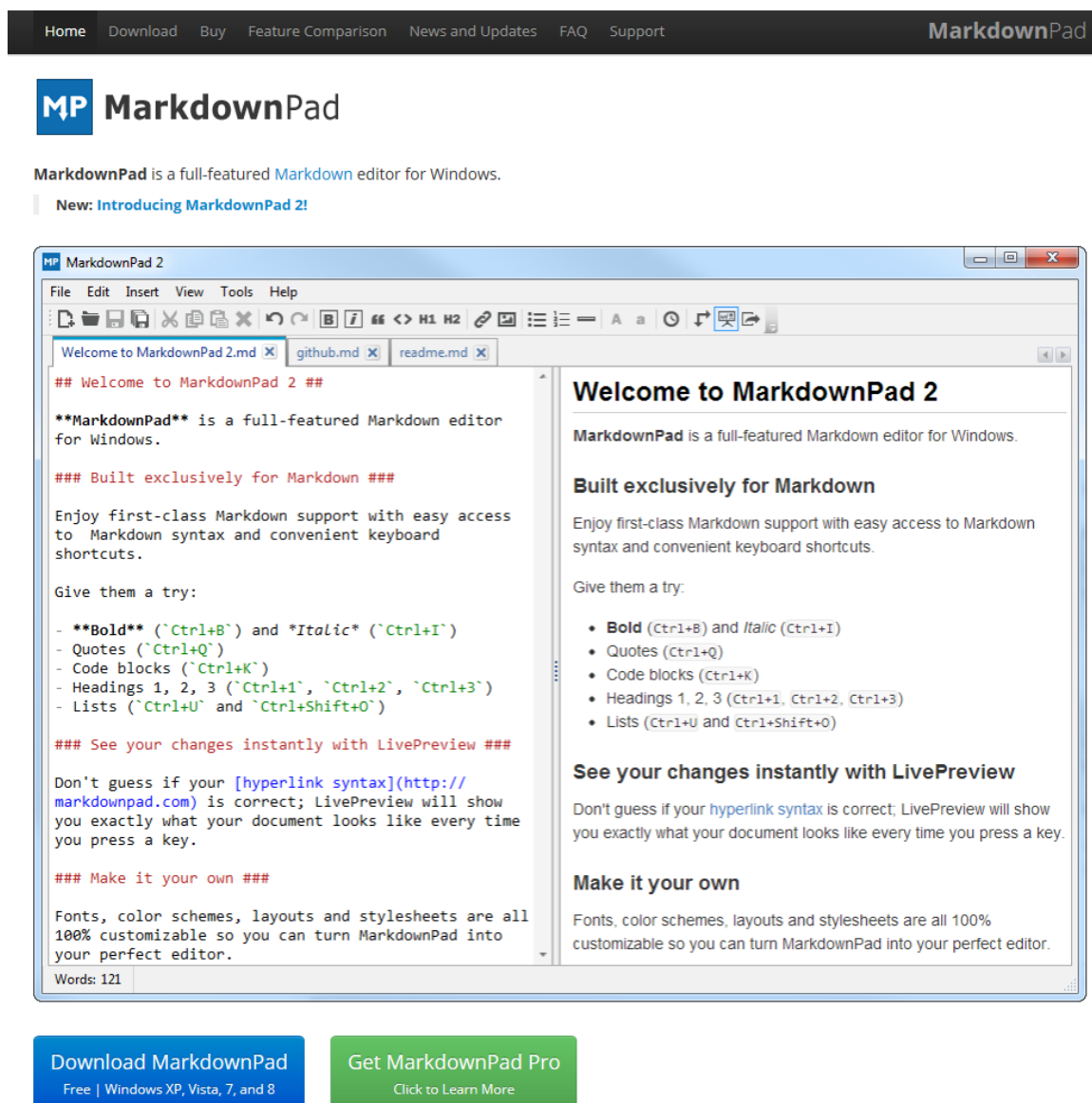


图 1-12 MarkdownPad 编辑器官网主页

第 2 章 UI 系统基础

通过第 1 章的介绍，想必读者已经对 Unity 新推出的 UI 系统，有了一个简单的了解了。而本章要学习的内容是读者对 UI 系统的第一次深入。内容涉及 UI 系统的基础。内容较为零散，主要分为以下 3 个部分：

- ❑ Canvas 是 UI 系统提供的一个“画布”，用于承载 UI 系统提供的所有 UI 控件，而后者必须绘制在这个画布上。
- ❑ 为 UI 控件提供在“画布”上的布局支持，一个工具（Rect Tool）和一个组件（Rect Transform）。
- ❑ 让 UI 控件在游戏视图中可见的 UI 组件。

2.1 画布——Canvas

Canvas 是 Unity 新 UI 系统里的游戏对象，可以译为“画布”。UI 系统中所有的 UI 控件都应该被“绘制”在它的上面，也就是说所有的 UI 控件都应该是它的子对象。

注意：若是开发者在未创建 Canvas 的时候，就先创建了其它的 UI 控件（例如 Image），那么 Unity 会自动为其创建 Canvas，并将新创建的 UI 控件置于 Canvas 下，成为后者的子对象，如图 2-1 所示。

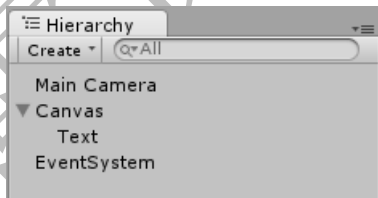


图 2-1 创建 Text 控件的时候，Unity 一并为其创建了 Canvas 对象，并使后者成为前者的父对象

本节会向读者介绍 Canvas 对象，包括它的创建，及其属性的作用。当然还有那些成为 Canvas 子对象的 UI 控件，以及它们在游戏视图中绘制的先后顺序。

2.1.1 创建 Canvas 对象

在 Unity 中，单击 GameObject|UI|Canvas 命令，即可在 Hierarchy 视图里创建名为 Canvas 的游戏对象。游戏对象 Canvas 上，还有一个同名的组件，如图 2-2 所示。

提示：此组件中的 Render Mode 属性是后面介绍的重点，也就是后面要详细展开介绍的“绘制模式”。

新创建的 Canvas 对象在 Scene 视图中呈现为一个矩形，如图 2-3 所示。这便于开发者在其中添加 UI 控件，而不再需要同时对照 Game 视图。

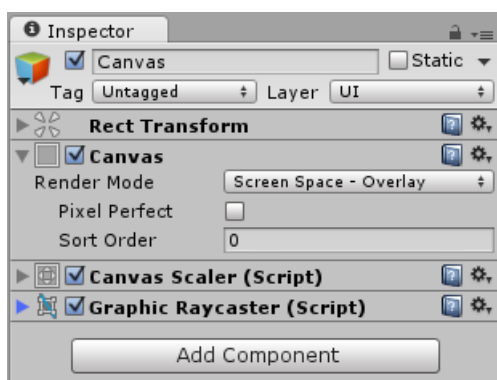


图 2-2 Canvas 游戏对象，及其同名组件

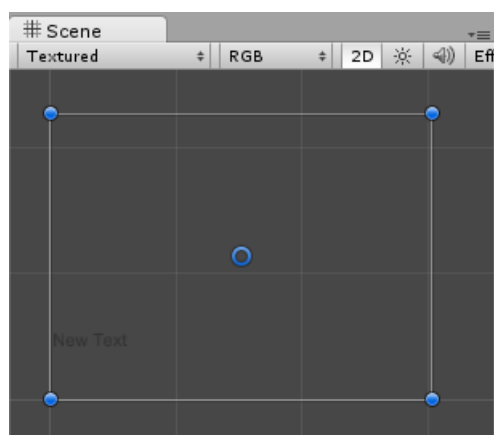


图 2-3 Scene 视图中的 Canvas 对象

2.1.2 UI 控件的绘制顺序

Canvas 下的 UI 控件在游戏视图中的绘制顺序，是与它们在 Hierarchy 视图里的排列顺序一致的。也就是说 Canvas 的第一个子对象最先被绘制，接着绘制下一个子对象，依此类推，如图 2-4 所示。

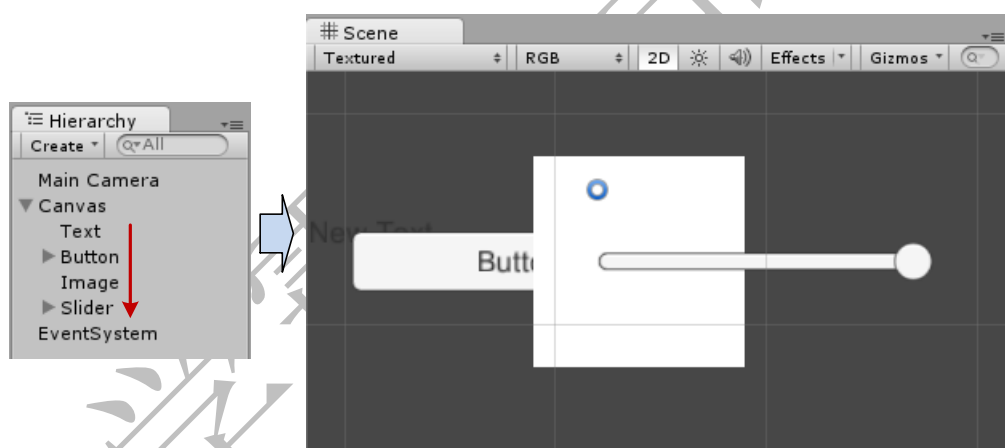


图 2-4 Canvas 子对象 UI 控件的绘制顺序，与 Hierarchy 视图里的排列顺序一致

注意：如果两个 UI 控件发生了重叠，那么先被绘制的 UI 控件，会被后绘制的 UI 控件覆盖。

为了修改各 UI 控件的绘制顺序，开发者可以采用以下两种方法：

- ☐ 拖动 Hierarchy 视图里的各 UI 控件对象，改变它们在 Canvas 下的排列顺序；
- ☐ 在脚本代码中使用类 Transform 的 3 个方法：SetAsFirstSibling()、SetAsLastSibling() 和 SetSiblingIndex()，各方法的使用说明，如表 2.1 所示；

表 2.1 修改对象在 Hierarchy 视图里排列顺序的 3 个方法

方法原型	参数说明	作用说明
Transform.SetSiblingIndex (int index)	index，用于指定游戏对象的排列位置	将游戏对象置于指定的位置处
Transform.SetAsFirstSibling()	无参数	将游戏对象置于最前面的位置处
Transform.SetAsLastSibling()	无参数	将游戏对象置于最后面的位置处

		处
--	--	---

2.1.3 绘制模式

Canvas 组件上有一个名为 **Render Mode**（即“绘制模式”）的属性，如图 2-5 所示。它可以决定 UI 控件是被绘制在屏幕空间还是世界空间里。此属性有 3 个可选项：**Screen Space - Overlay**、**Screen Space - Camera** 和 **World Space**，本小节会分别介绍它们。

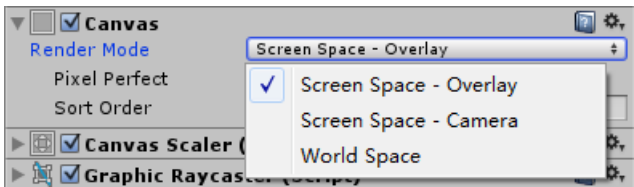


图 2-5 Canvas 组件上的 Render Mode 属性

1.Screen Space - Overlay

在此种绘制模式下，Canvas 下的 UI 控件会被绘制在游戏场景的最上层，而且在游戏视图的分辨率发生变化时，Canvas 的大小会随之一同发生改变，如图 2-6 所示。

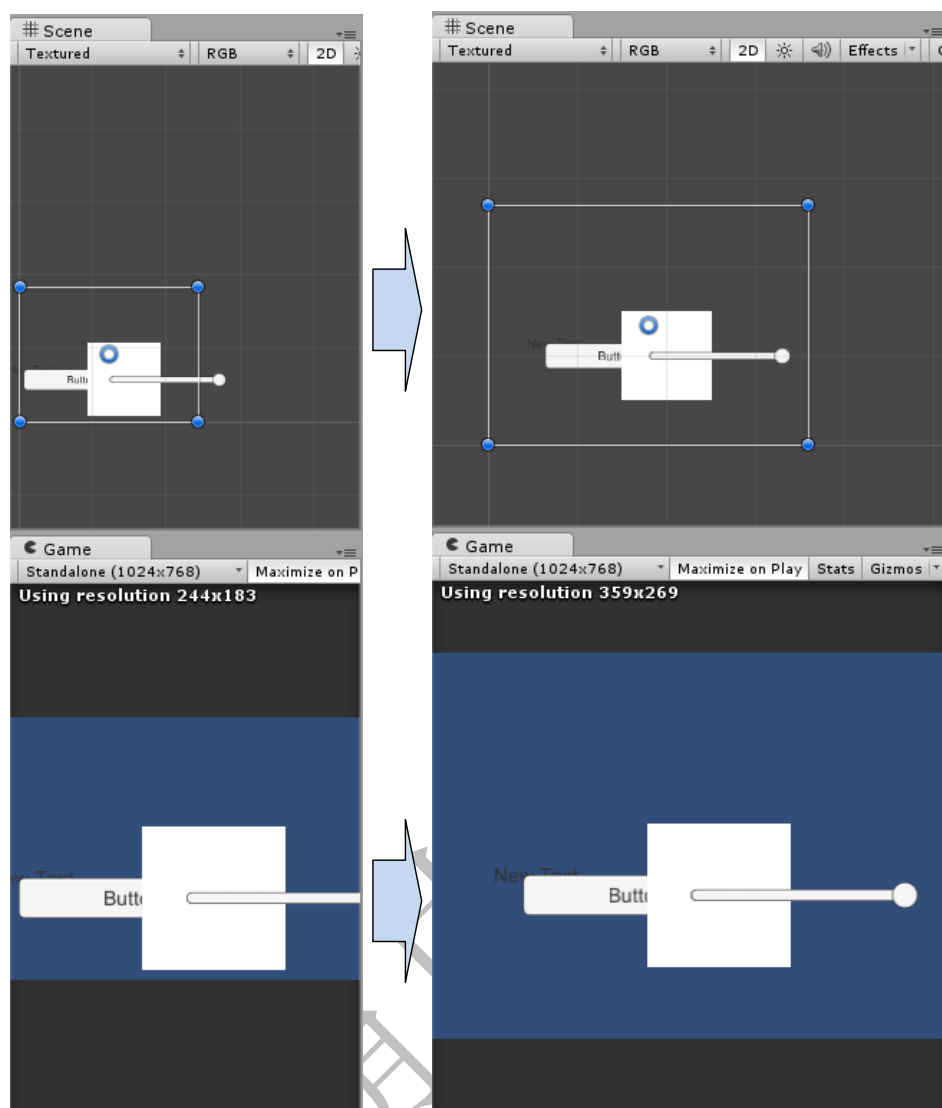


图 2-6 Canvas 的大小，会随着游戏视图的分辨率一同改变

2.Screen Space – Camera

在此种绘制模式下，Canvas 下的 UI 控件同样会被绘制在游戏场景的最上层，而且在游戏视图的分辨率发生变化时，Canvas 的大小会随之一同发生改变。但是此种模式还有更多的规则：Canvas 所在的平面，必须与一个指定的 Camera 保持固定的距离；而 Canvas 下的所有 UI 控件，也将由此 Camera 负责绘制（同时也意味着对此 Camera 的设置，会影响到 UI 控件的绘制效果，如图 2-7 所示）

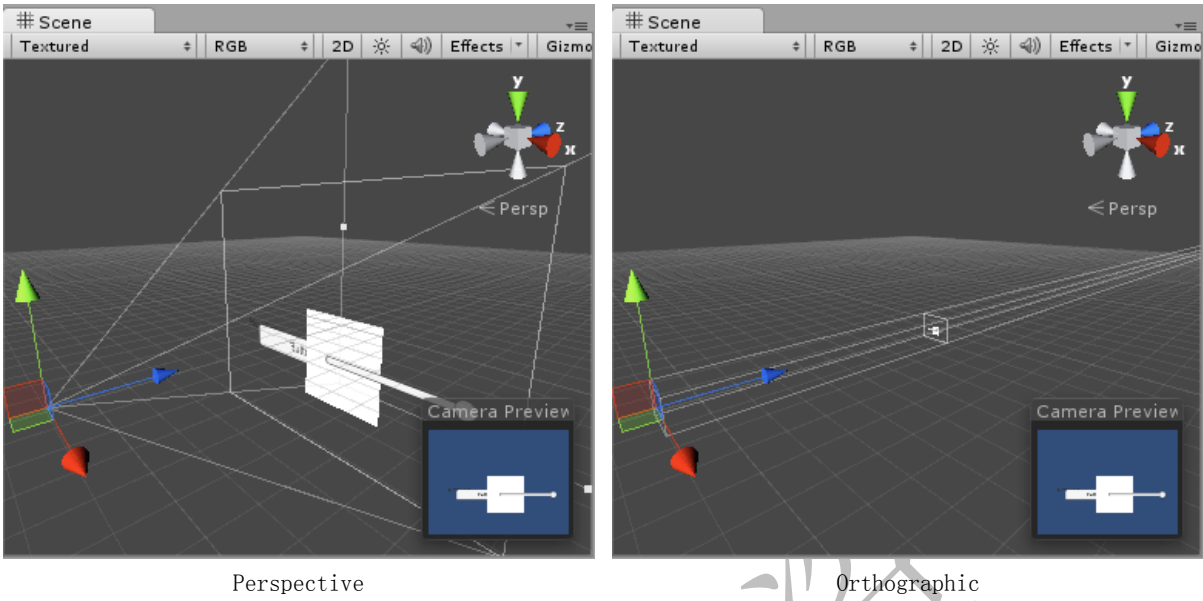


图 2-7 Camera 的投影方式不同（Projection 属性设置值不同）

注意：当 Camera 处于 Perspective 的投影下，截头锥体（frustum）大小的改变，会影响到 Canvas 及其子对象 UI 控件大小的改变，如图 2-8 所示。

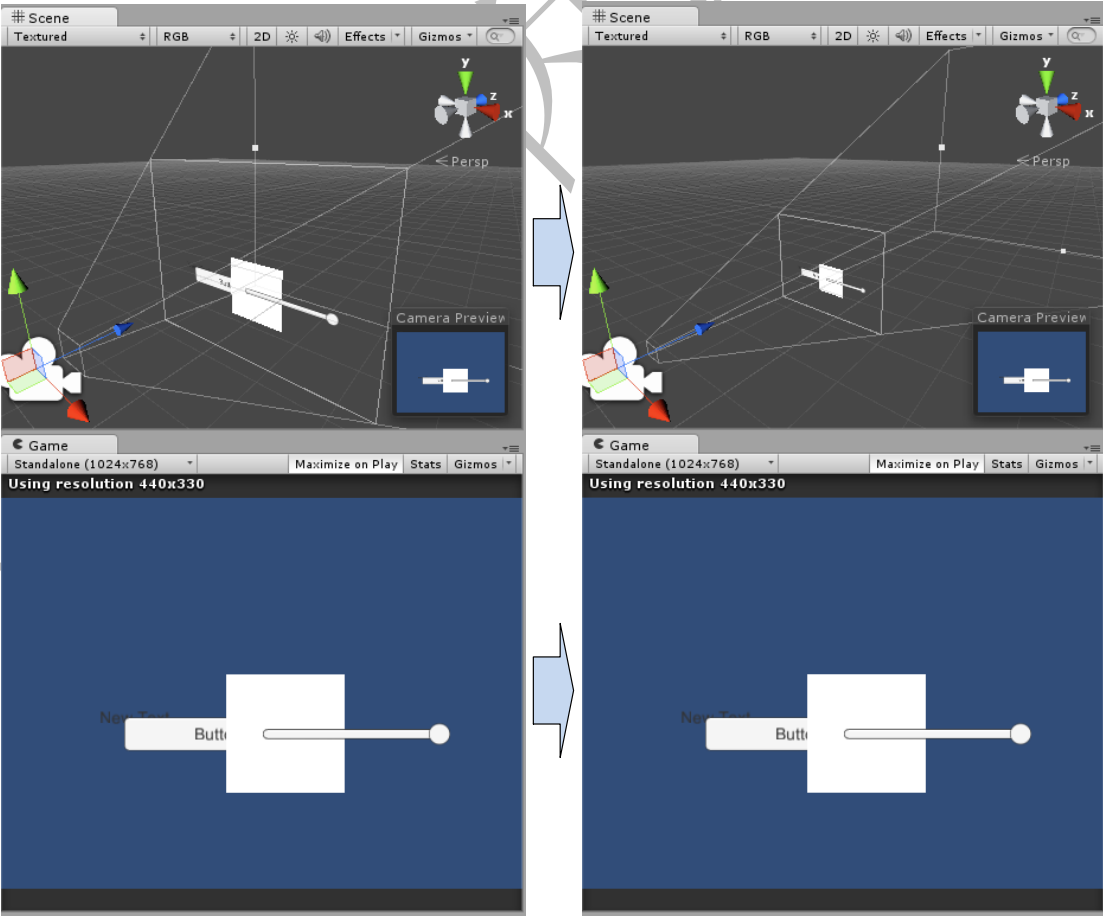


图 2-8 Canvas 的大小，会随着 Camera 截头椎体的大小一同改变

3.World Space

在此种绘制模式下，Canvas 会位于游戏场景中所有游戏对象的最后面。而 Canvas 下的 UI 控件，在游戏场景中的位置可以任意设置，如图 2-9 所示。读者可以理解为 UI 控件被绘制在了游戏场景中。

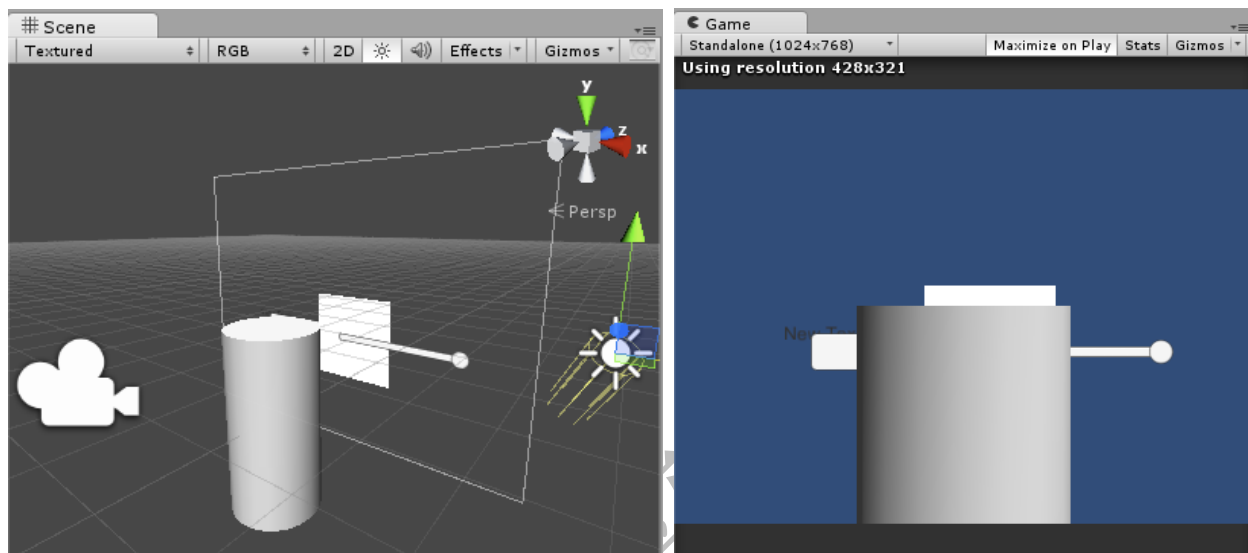


图 2-9 Canvas 及其子对象，与游戏场景中其它 3D 对象之间的位置关系

注意：此时 Canvas 及其子对象 UI 控件的大小，将不会再随着游戏视图分辨率的大小而改变了。Canvas 的大小只能有开发者自己手动去设置。设置方法是在 Inspector 视图里，设置 Canvas 对象 Rect Transform 组件的值，如图 2-10 所示。

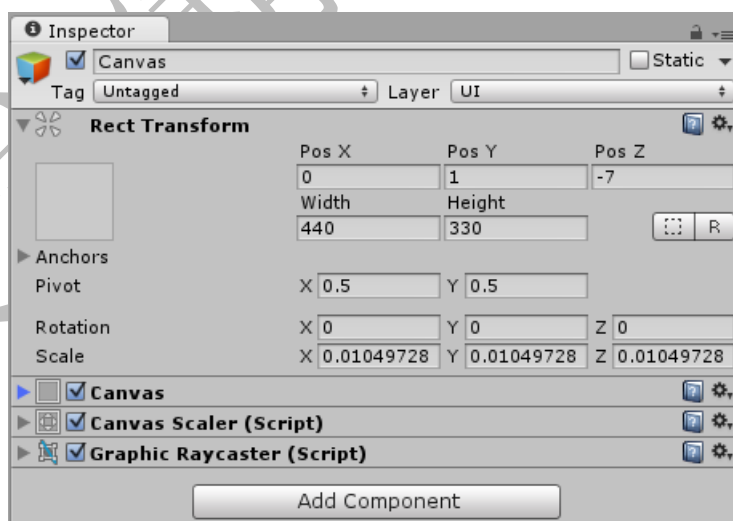


图 2-10 Rect Transform 组件

提示：有关 Rect Transform 组件的内容，会在本章后面的部分被介绍。

2.2 UI 控件的布局

在“画布”上“绘制”的 UI 控件，除了要考虑上一节介绍到的绘制顺序以外，还需要考虑它们的布局问题，以及当布局不符合开发者预期时的处理方法。为此，本节介绍了一个工具——Rect Tool，让开发者有了直接操作 UI 控件的能力。另外还介绍了一个组件——Rect Transform，让 UI 控件有了“自适应”的本事！

2.2.1 Rect Tool 小工具

Rect Tool，是新版本 Unity 编辑器（Unity 4.6）在工具栏上添加的一个新按钮，位于界面左上角，如图 2-11 所示。单击此按钮即可调用 Rect Tool，而后者使得开发者能够在 Scene 视图里直接操作 UI 控件。

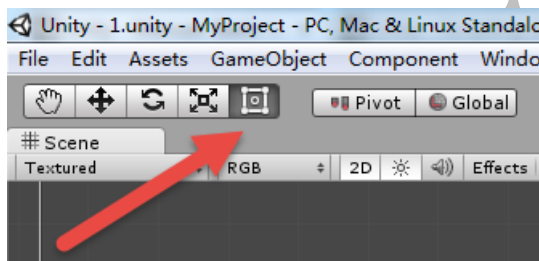


图 2-11 工具栏上的 Rect Tool 按钮

提示：Rect Tool 不单单专用于操作 UI 控件，实际上它可以被用于操作任何游戏对象。只不过这个工具是在 UI 系统加入的时候一并被添加进来的。

Rect Tool 可以对 UI 控件展开下列操作：

- ❑ 改变位置：将鼠标置于 UI 控件矩形框的内部，按下鼠标左键任意拖动，即可改变 UI 控件的位置，如图 2-12 所示。

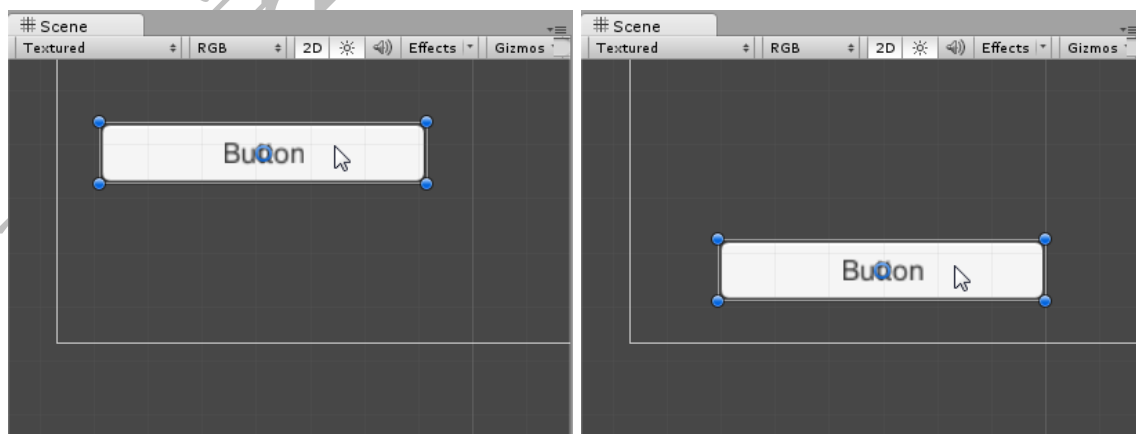


图 2-12 使用 Rect Tool，改变 UI 控件的位置

- ❑ 改变大小：将鼠标置于 UI 控件矩形框上，待鼠标变成双向的箭头，按下鼠标左键拖动，即可改变 UI 控件的大小，如图 2-13 所示。

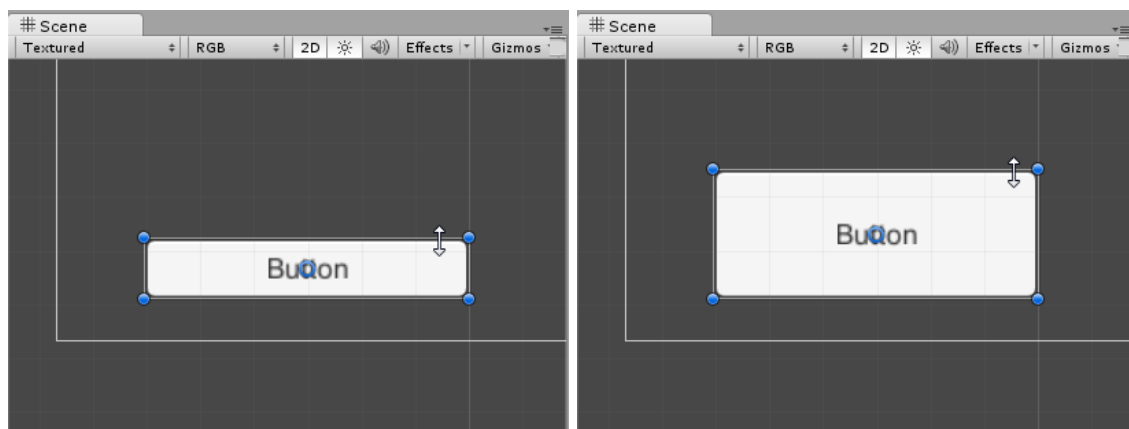


图 2-13 使用 Rect Tool，改变 UI 控件的大小

- 旋转：将鼠标移动到 UI 控件矩形框靠近 4 个角的外部，待鼠标出现一个旋转的标志，按下鼠标左键拖动，即可令 UI 控件旋转，如图 2-14 所示。

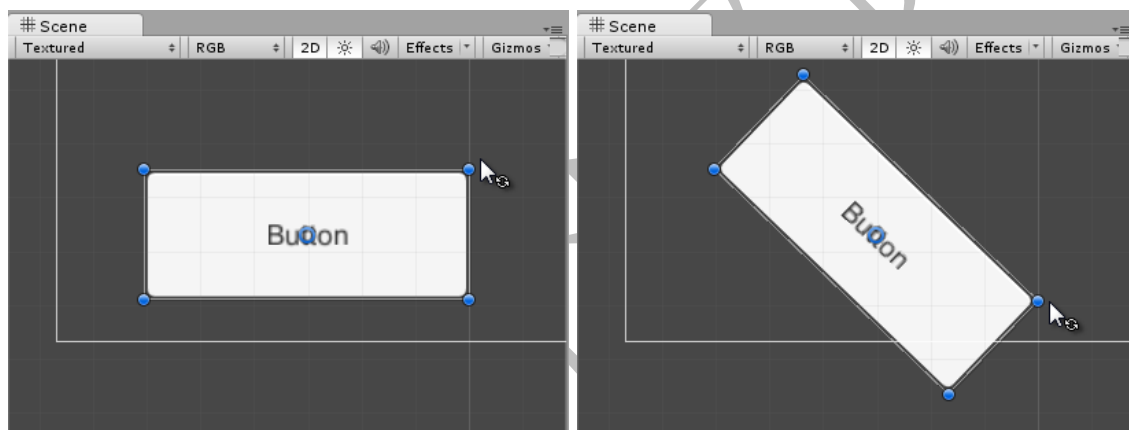


图 2-14 使用 Rect Tool，旋转 UI 控件

2.2.2 Rect Transform 组件

Rect Transform 组件是新版 Unity 为 UI 控件提供的新组件。它只在 UI 控件上取代 Transform 组件，而后者是其它任何游戏对象所必须的组件。如图 2-15 所示，对比了 Transform 和 Rect Transform 组件。

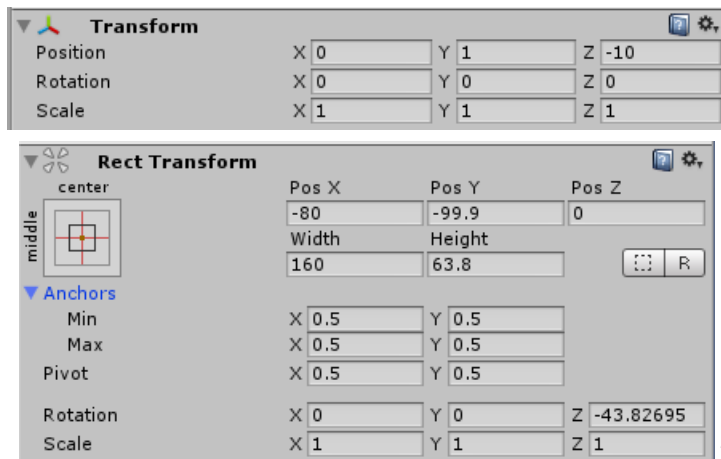


图 2-15 Transform 和 Rect Transform 组件

Rect Transform 组件除了拥有 Transform 组件的位置（Position）、朝向（Rotation）和缩放（Scale）属性外，还纳入了更多的属性：Width、Height、Anchors、Pivot 和 Anchor Presets。本小节会依次介绍它们。

1. 基本属性

先来看 Rect Transform 组件的基本属性：

- Width 和 Height 用于表示 UI 控件的长和宽，如图 2-16 所示；

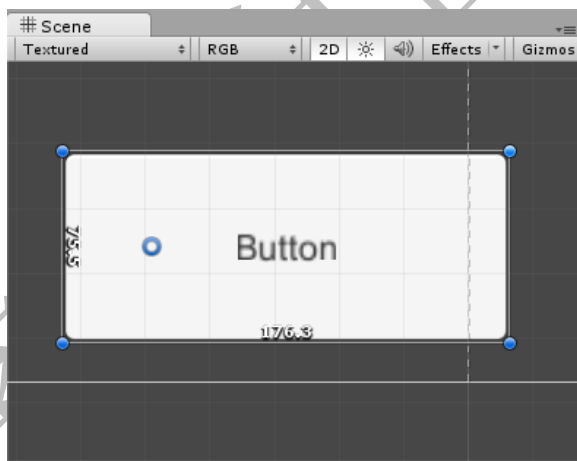


图 2-16 UI 控件的长和宽

- Pivot 用于表示 UI 控件的轴心点，在 Scene 视图中用蓝色的空心小圆圈表示；当开发者对 UI 控件做旋转处理的时候，UI 控件旋转时所围绕的点就是轴心点，如图 2-17 所示。

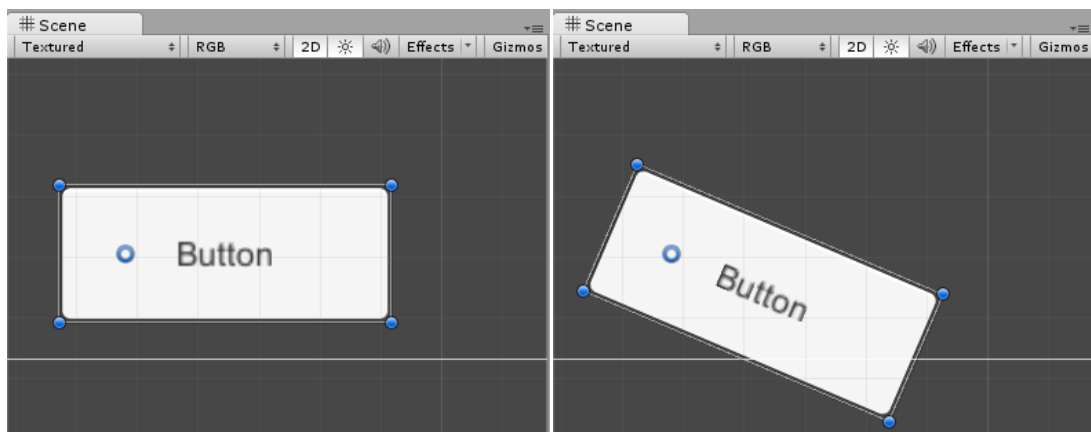


图 2-17 UI 控件中轴心点在旋转时所起的作用

2. Anchors

Anchors 属性是重点，因此拿出来单独介绍。它本身在 Scene 视图中有专门的图标来表示，同时它还有自己的子属性，最后它还会影响除 Anchor 属性以外的其它属性。下面依次介绍：

- Anchors 用于表示 UI 控件的锚点，在 Scene 视图中使用 4 个小三角表示。当 UI 控件的父对象也拥有 Rect Transform 组件的时候，锚点才会出现。锚点的 4 个小三角依次与 UI 控件四边形边框的四个角对应，且在父对象的大小发生改变时，锚点会令子对象自动完成自适应操作。例如，Button 的锚点位于 Canvas 的中央、两角和一边上时，那么当 Canvas 的大小发生改变的时候，Button 的自适应效果如图 2-18、图 2-19 和图 2-20 所示。

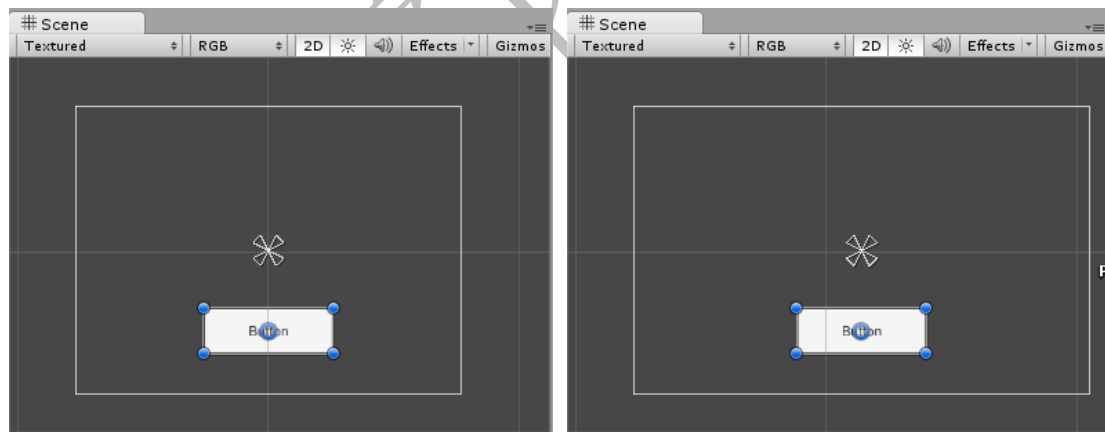


图 2-18 Button 的锚点位于 Canvas 的中央（锚点的 4 个三角形，与 UI 控件边框的 4 个角的距离固定）

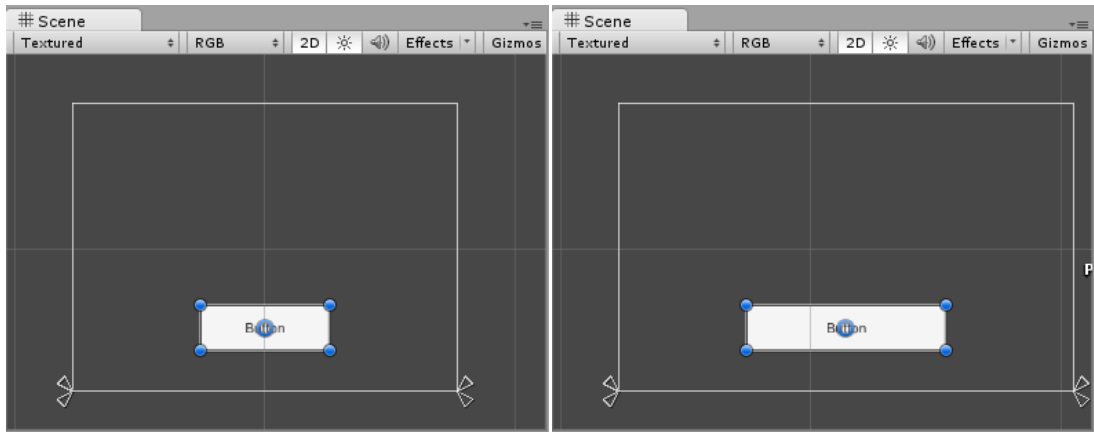


图 2-19 Button 的锚点位于 Canvas 的两角（锚点的 4 个三角形，与 UI 控件边框的 4 个角的距离固定）

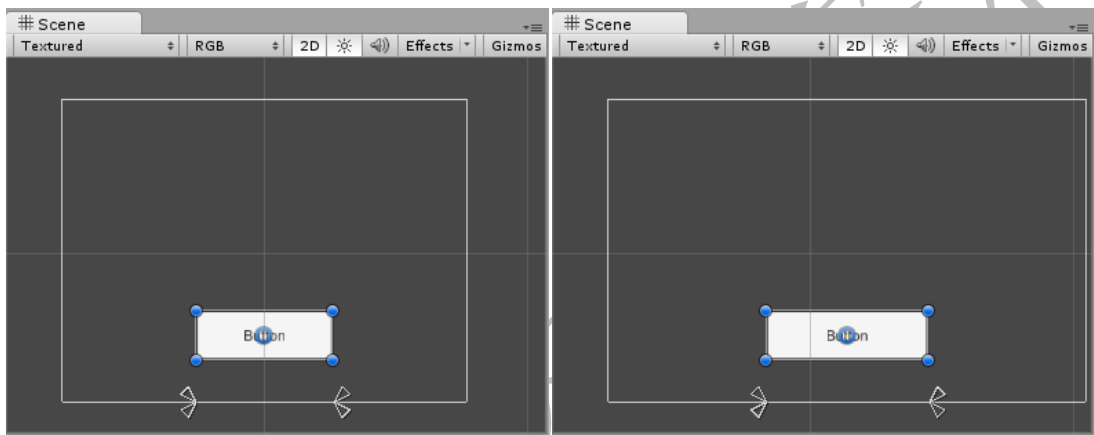


图 2-20 Button 的锚点位于 Canvas 的一边（锚点的 4 个三角形，与 UI 控件边框的 4 个角的距离固定）

注意：从锚点的位置所导致 UI 控件自适应的效果上来看，读者应该发现一个不变的真理，就是表示锚点的 4 个三角形，始终与 UI 控件边框 4 个角的距离是固定的。

- 单击 Rect Transform 组件里的 Anchors 属性左侧的小三角，可以展开此属性的子属性 Min 和 Max。Min 记录的是 Anchors 左下角小三角的位置，Max 记录的是 Anchors 右上角小三角的位置，如图 2-21 所示。

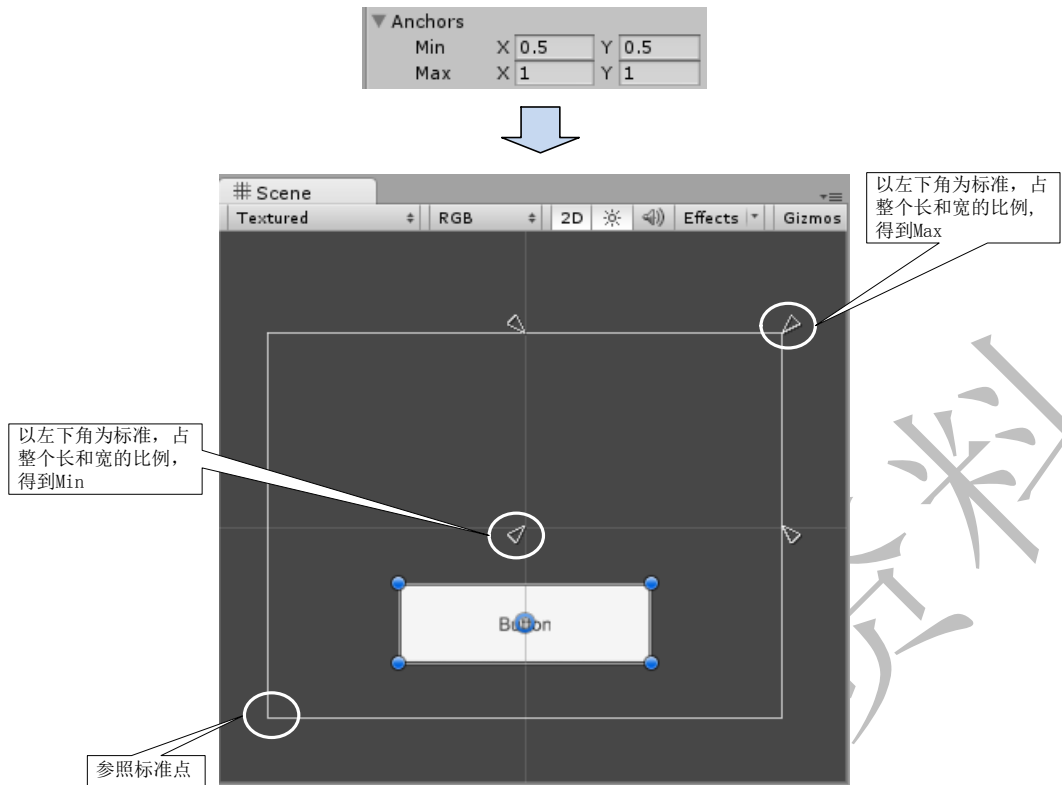


图 2-21 Anchors 属性的子属性 Min 和 Max，及其含义

- 当表示 Anchors 的 4 个小三角处于同一位置的时候，Rect Transform 组件里就会出现我们前面一直在介绍的 Pos X、Pos Y、Width 和 Height 属性。Width 和 Height 我们前面介绍过了，而 Pos X 和 Pos Y 的值表示以 Anchors 为原点的 Pivot 的坐标，如图 2-22 所示。

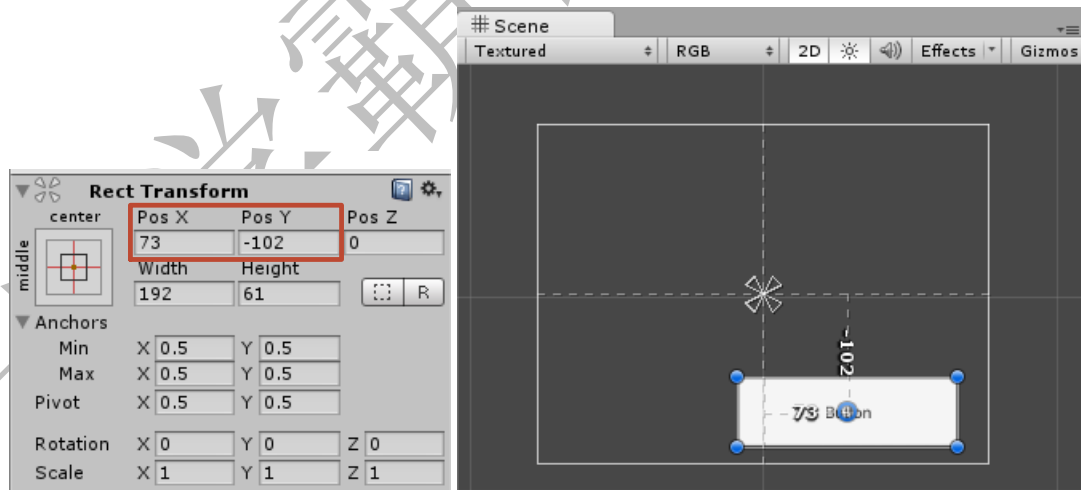


图 2-22 Rect Transform 组件里 Pos X 和 Pos Y 属性的含义

- 当表示 Anchors 的 4 个小三角不在一起的时候，这 4 个属性名就变成了 Left、Top、Right 和 Bottom。它们分别表示 UI 控件的 4 条边，到 4 个三角形所构成矩形对应边的距离，如图 2-23 所示。

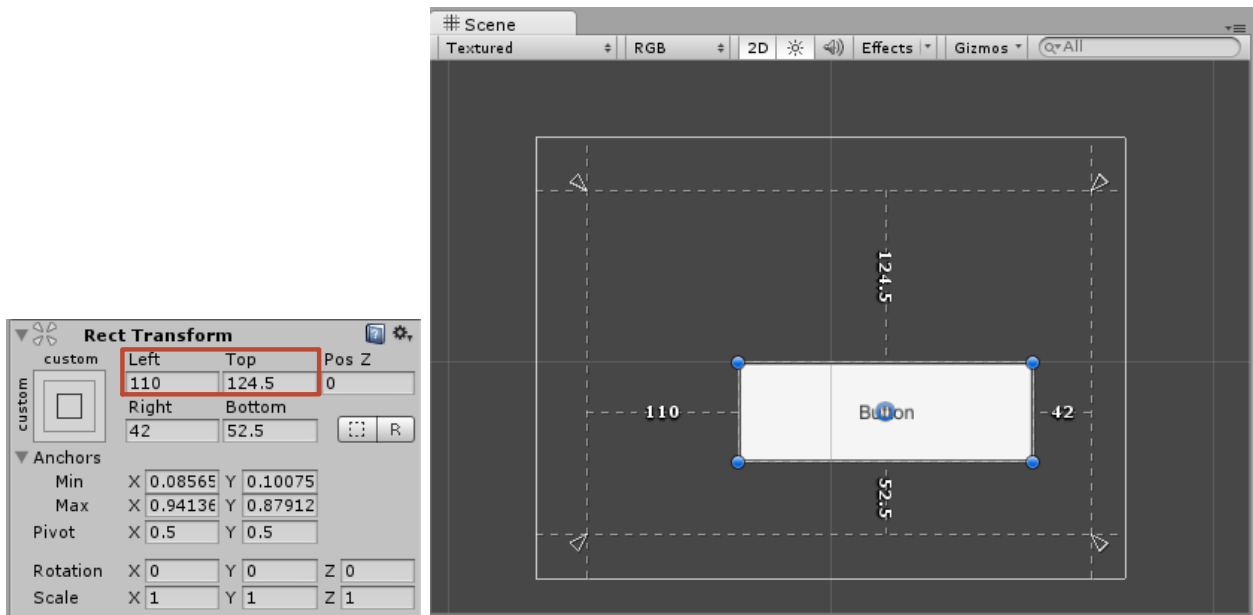


图 2-23 Rect Transform 组件里 Left、Top、Right 和 Bottom 属性的含义

3.Anchor Presets

Anchor Presets 属性是一个“样子特殊的下拉列表”，一般容易被忽略。但是如果灵活的使用它，可以为 UI 控件的布局提供大大的便利，因为它提前预置了常见的布局方式，接下来就详细介绍它：

- ❑ Anchor Presets，此属性位于 Rect Transform 组件的左上角，使用鼠标单击那个正方形即可打开 Anchor Presets 属性，如图 2-24 所示。此属性中预先定义了锚点的位置，开发者可以直接从预定义的锚点中选择。例如，将 Button 的锚点设置到左上角，效果如图 2-25 所示。

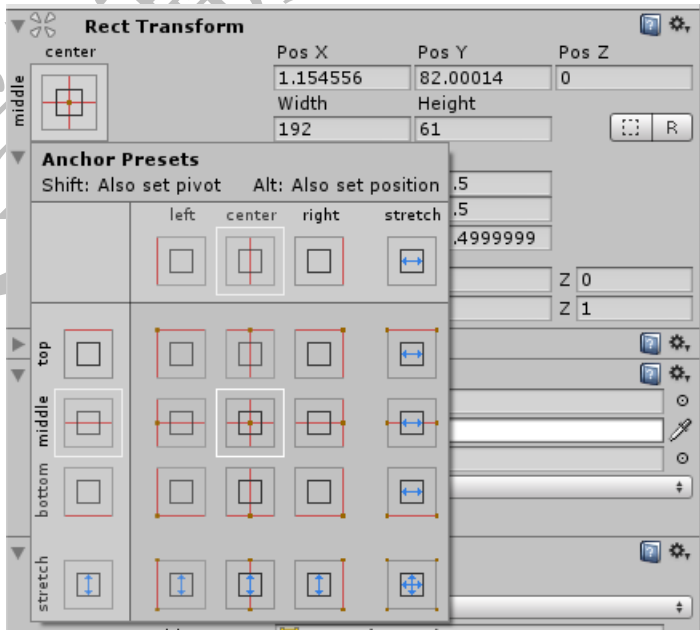


图 2-24 Rect Transform 组件中的 Anchor Presets 属性

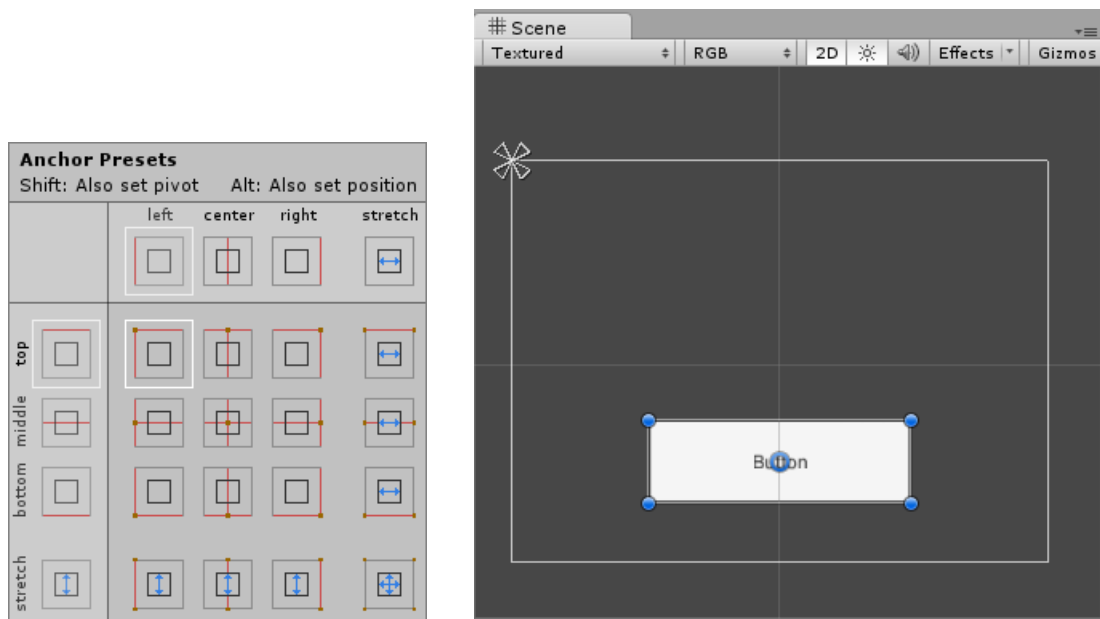


图 2-25 使用 Anchor Presets 属性为 Button 设置锚点

注意：使用 Anchor Presets 属性也可以一并设置 UI 控件的 Pivot 和 Position 属性，只需要按下 Shift 和 Alt 键盘即可，如图 2-26 和图 2-27 所示。

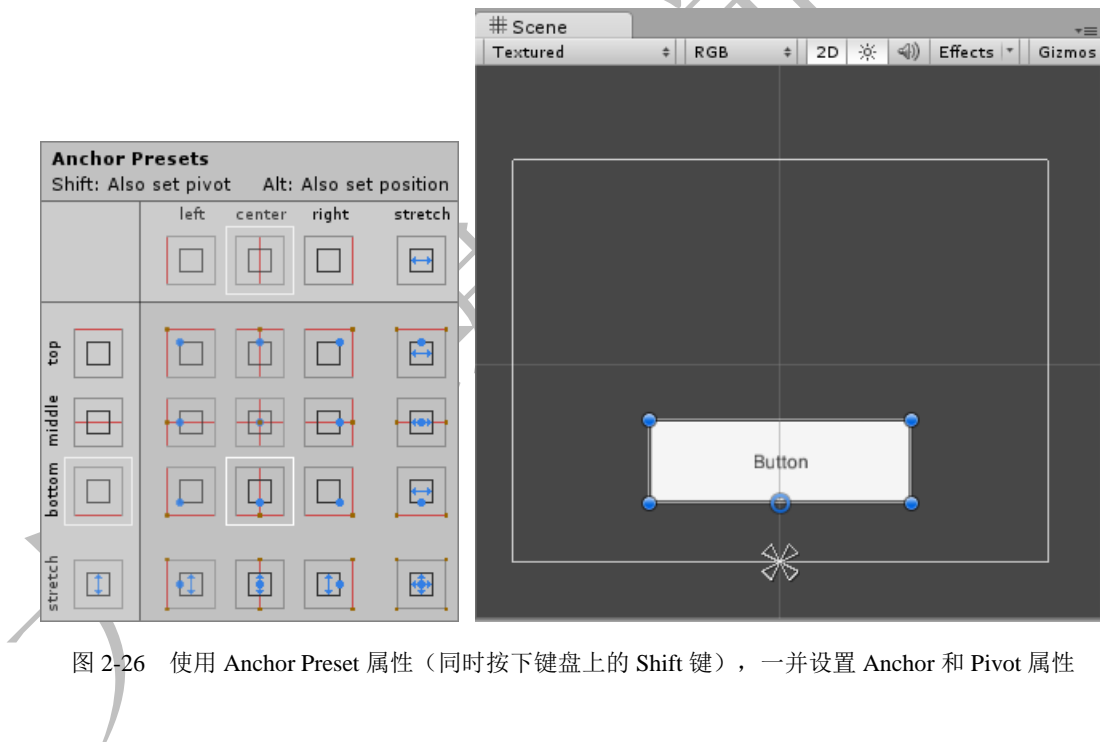


图 2-26 使用 Anchor Preset 属性（同时按下键盘上的 Shift 键），一并设置 Anchor 和 Pivot 属性

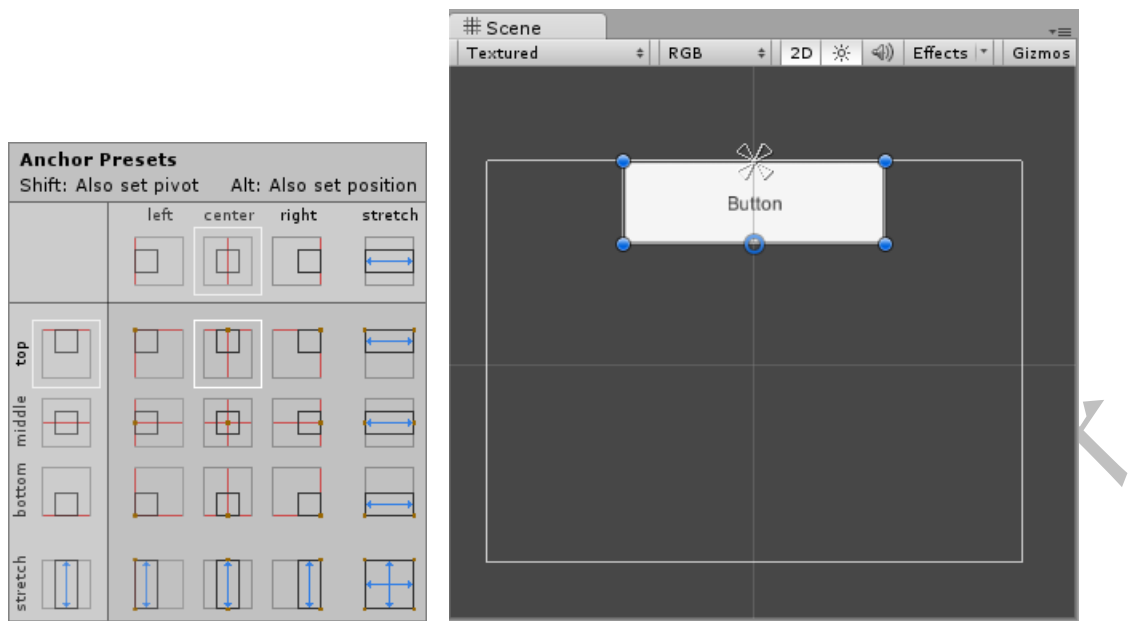


图 2-27 使用 Anchor Preset 属性（同时按下键盘上的 Alt 键），一并设置 Anchor 和 Position 属性

2.3 提供可视功能的 UI 组件

在介绍 UI 系统的 UI 控件之前，读者首先应该有一个这样的认识：UI 控件之所以在游戏场景中是可见的，是因为 UI 控件上有一些特定功能的 UI 组件！本节就来介绍这些为 UI 控件提供了可视功能的 UI 组件。

2.3.1 显示文字——Text (Script)

Text (Script) 组件的文字显示效果，如图 2-28 所示。该组件用于向玩家显示不可交互的文本信息。玩家无法修改、删除其中的文本，只能从中了解信息。

注意：Text (Script) 组件只包括可见的文字，它的背景是透明的。为了更形象的展示效果，本小节在文字信息的后面使用了图片控件作为那个“白色的背景”，如图 2-29 所示。



图 2-28 Text (Script) 组件的文字显示效果

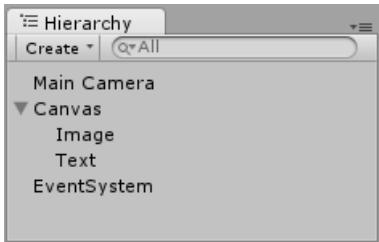


图 2-29 图片控件作为文字信息的“白色背景”

1.Text(Script)组件基本属性

Text(Script)组件如图 2-30 所示。它包含 3 个基本属性：

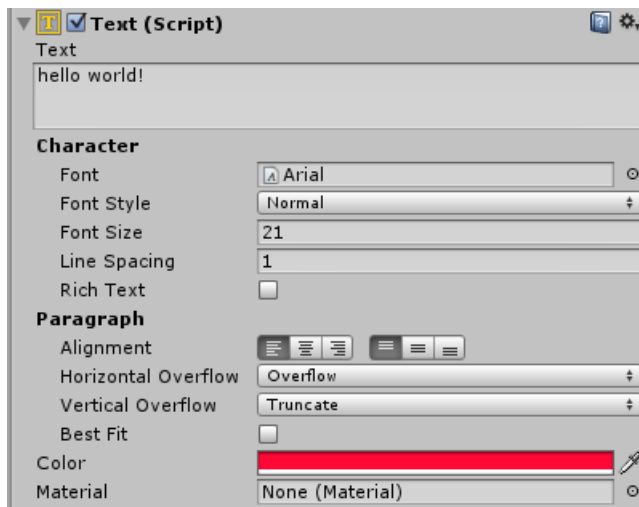


图 2-30 Text(Script)组件及其属性

- Text: 此属性中的内容，就是 Text(Script)组件要显示的文本信息；
- Color: 用于设置文字的颜色；
- Material: 用于设置文字的材质；

注意: 游戏场景中的文字控件本身就是一个游戏对象, 当前就可以设置它的材质属性了。

2.Text(Script)组件属性——Character

Text(Script)组件的属性中，Character 表示接下来的属性负责设置字符的格式：

- Font: 用于设置文本信息的字体格式；
- Font Style: 用于设置文本信息的样式，可选择的样式有：Normal（正常）、Bold（加粗）、Italic（倾斜）和 Bold And Italic（倾斜并加粗），如图 2-31 所示；

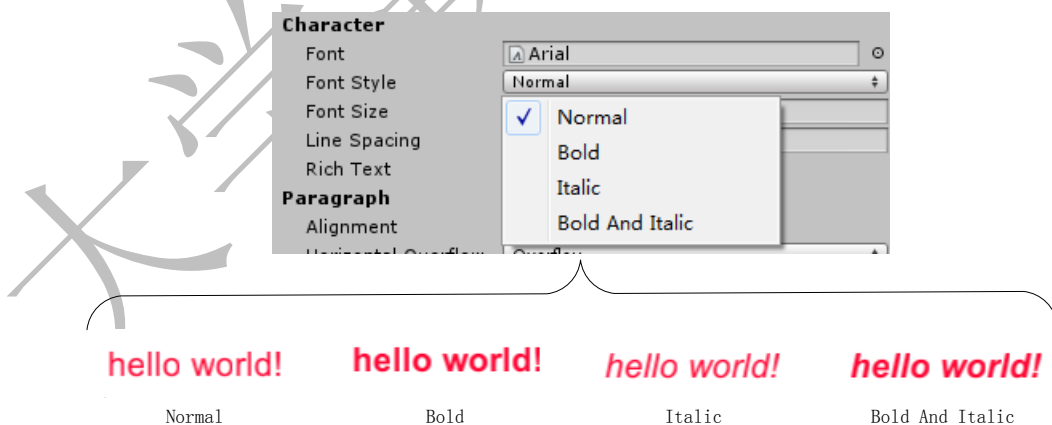


图 2-31 可选择的文本样式

- Line Spacing: 用于设置文字的行间距，效果如图 2-32 所示；



图 2-32 文字的行间距

- ❑ Rich Text: 决定文本中的标记元素是否可用，效果如图 2-33 所示；

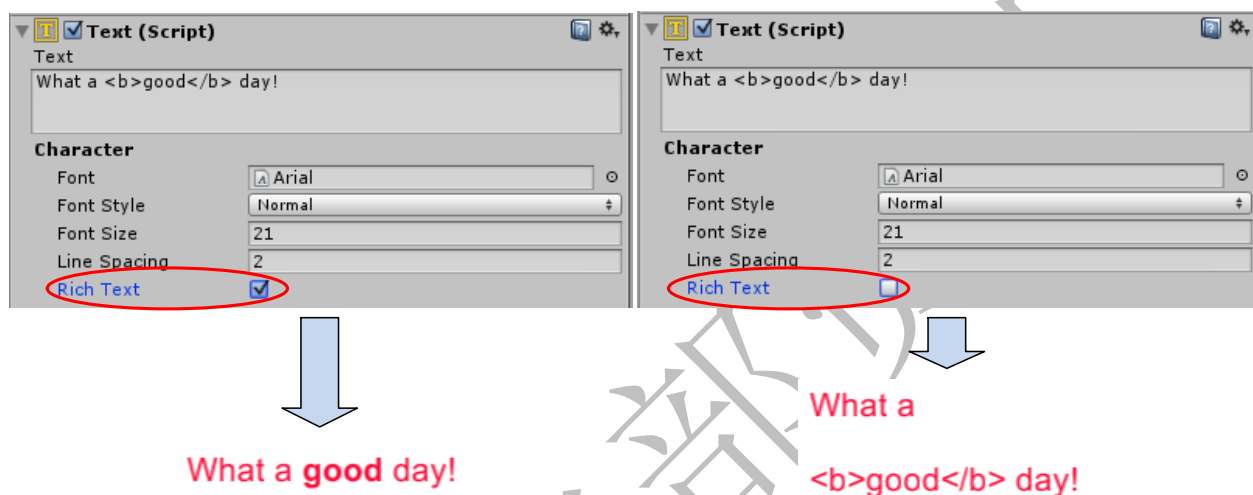


图 2-33 Rich Text 属性的作用

提示：Text 属性中的和就是“标记元素”，有关标记元素（markup elements）的内容，会在本章后面的部分详细介绍：

3.Text(Script)组件属性——Paragraph

Text(Script)组件的属性中，Paragraph 表示接下来的属性负责设置字符构成的段落的格式：

- ❑ Alignment: 用于设置段落的对齐方式，如图 2-34 所示；

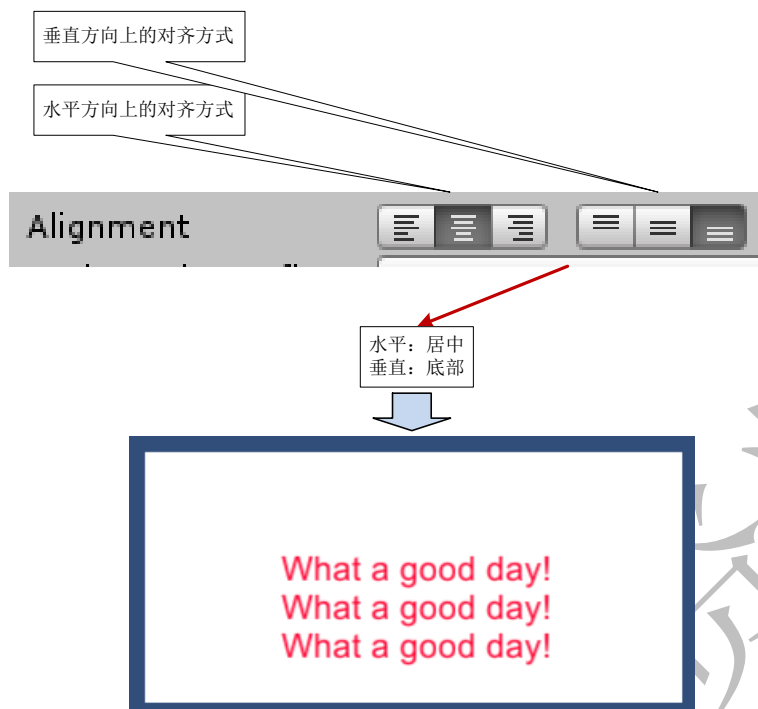


图 2-34 段落的对齐方式

- ❑ **Horizontal Overflow:** 当段落内容的宽度大于 Text(Text)组件本身的宽度时，处理段落内容有两种可选的方式，分别为 **Wrap**（换行）和 **Overflow**（溢出）。效果如图 2-35 所示。

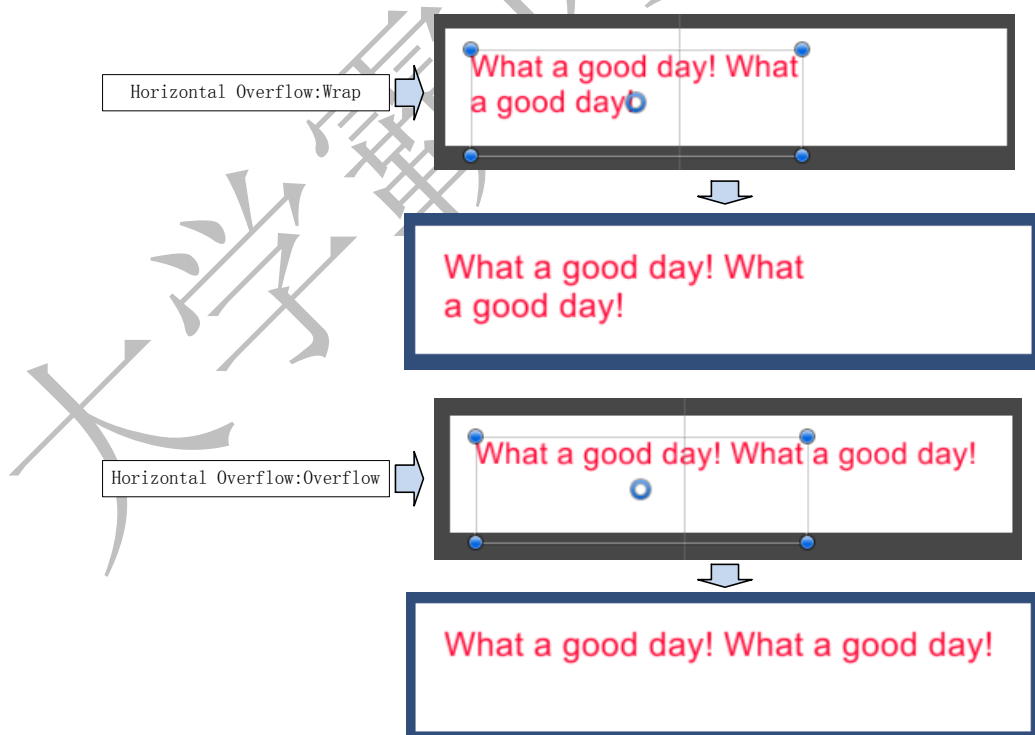


图 2-35 段落内容在水平方向上的溢出处理方式

- ❑ **Vertical Overflow:** 当段落内容的高度大于 Text(Text)组件本身的高度时，处理段落内容有两种可选的方式，分别为 **Truncate**（截断）和 **Overflow**（溢出）。效果如

图 2-36 所示。



图 2-36 段落内容在垂直方向上的溢出处理方式

- ❑ **Best Fit:** 是否应该忽略对文字大小的设置，让字体的大小自动改变到能让段落的内容全部显示出来的状态，效果如图 2-37 所示。

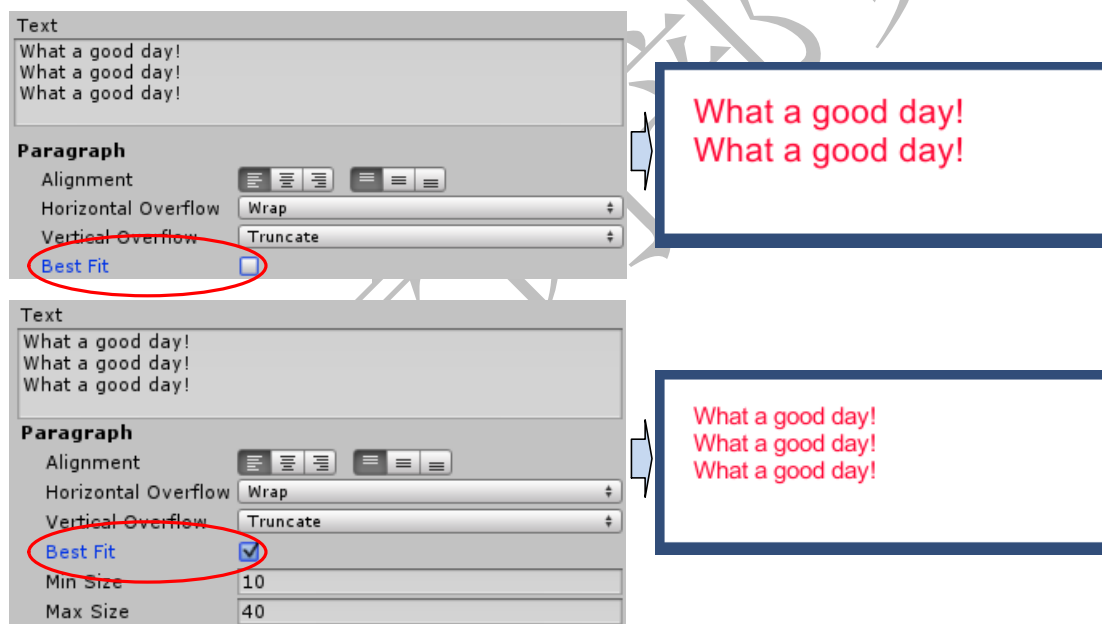


图 2-37 Best Fit 属性的设置效果对比

提示：当属性 Best Fit 被复选以后，开发者还可以设置字体大小自动改变的范围，即属性 Min Size 和 Max Size 所表示的含义。

2.3.2 显示图片 1——Image(Script)

Image(Script)组件的图片显示效果，如图 2-38 所示。它用于向玩家显示不可交互的图片信息，常作为游戏场景的装饰。



图 2-38 Image(Script)组件的图片显示效果

1.Image(Script)组件基本属性

Image(Script)组件如图 2-39 所示。它包含 3 个基本属性：

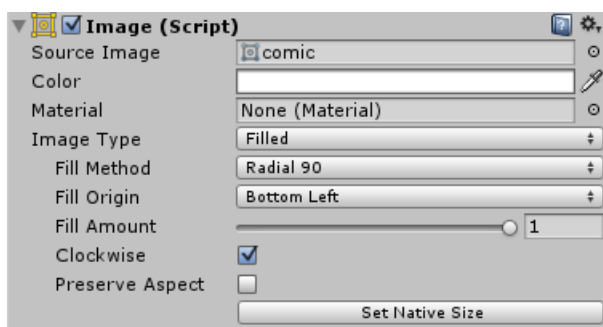


图 2-39 Image(Script)组件

□ Source Image: 指定 Image(Script)组件要显示的图片；

注意：Image(Script)组件显示的图片必须是 Sprite 类型的。这就要求开发者在将这个图片导入到 Unity 中的时候，设置此图片的 Texture Type 属性为 Sprite(2D and UI)，如图 2-40 所示。

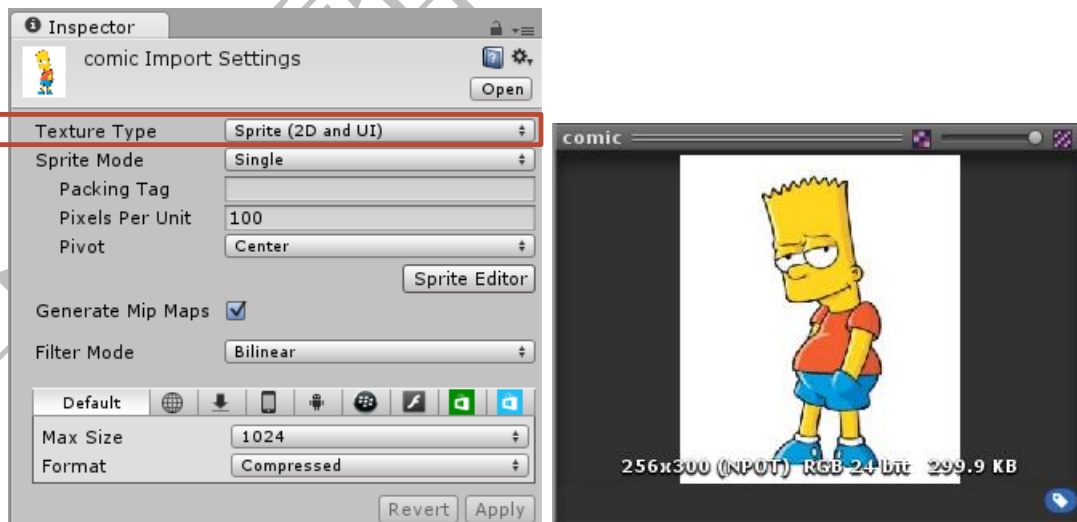


图 2-40 设置图片的 Texture Type 属性

□ Color: 设置图片的颜色；

提示：对此属性的设置会改变图片显示的主色调，就像是在用有颜色的光照射图片。

□ Material: 设置图片控件的材质；

2.Image(Script)组件属性——Image Type

Image Type 用于设置图片的显示类型，可选的属性值有 Simple、Sliced、Tiled 和 Filled。不同的图片显示类型，导致了 Sprite “填充” 图片控件的不同方式。

- ❑ Simple: Sprite 将直接显示在图片控件中。默认情况下，如果图片控件的大小与 Sprite 不一致时，后者将经过拉伸处理来符合前者的尺寸，如图 2-41 所示。但如果此时复选了 Preserve Aspect 属性，那么图片在经过缩放处理时，长宽的比例将保持恒定，如图 2-42 所示。

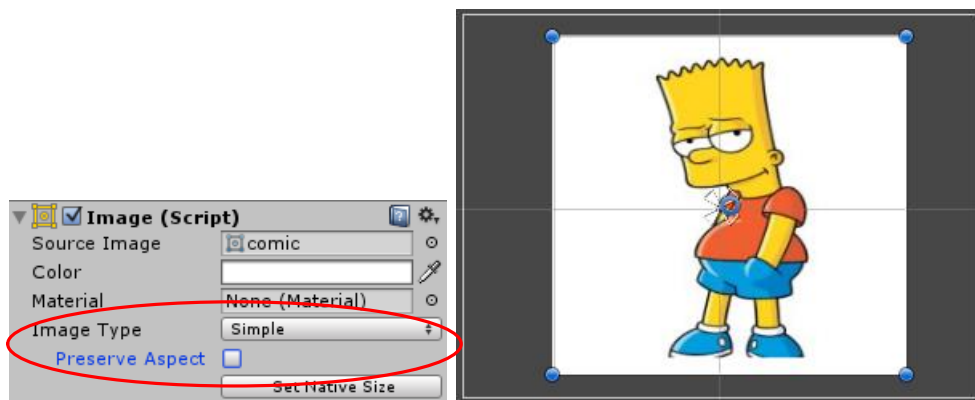


图 2-41 图片的显示类型：Simple（Sprite 长宽比例不固定）

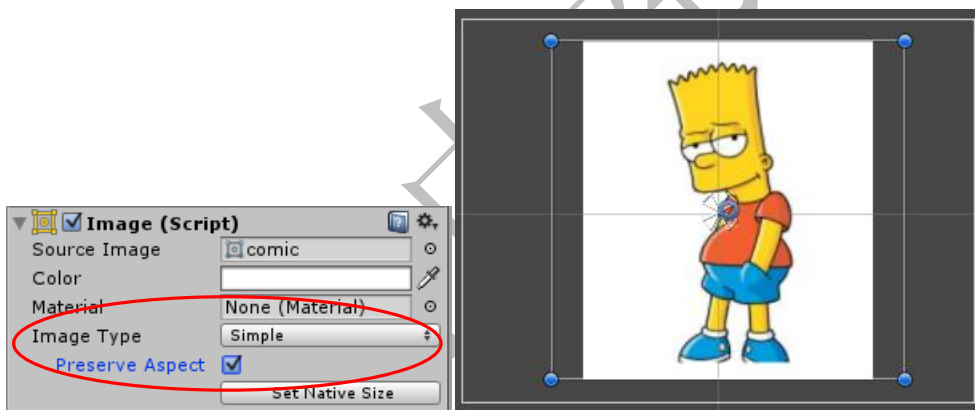


图 2-42 图片的显示类型：Simple（Sprite 长宽比例固定）

- ❑ Sliced: Sprite 将被看作是由 9 个切片组成的，如图 2-43 所示。而图片控件将只显示中间切片的边缘，如图 2-44 所示，若此时选中了 Fill Center 属性，那么将显示完整的中间切片，如图 2-45 所示。

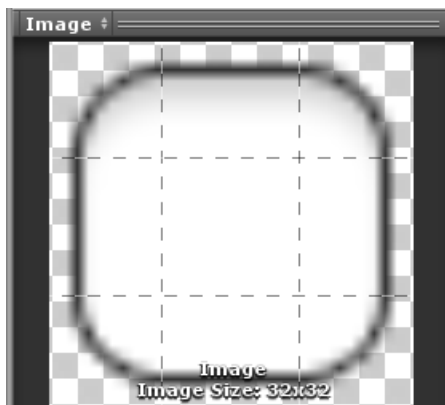


图 2-43 Sliced 类型的 Sprite，由 9 个切片组成

提示：Sliced 类型的 Sprite，一般情况下都有较为清晰的边缘。若对整个 Sprite 做缩放操作时，四个角的切片将不做缩放操作，四个边的切片将只完成拉伸操作，只有中间的切片做缩放操作。

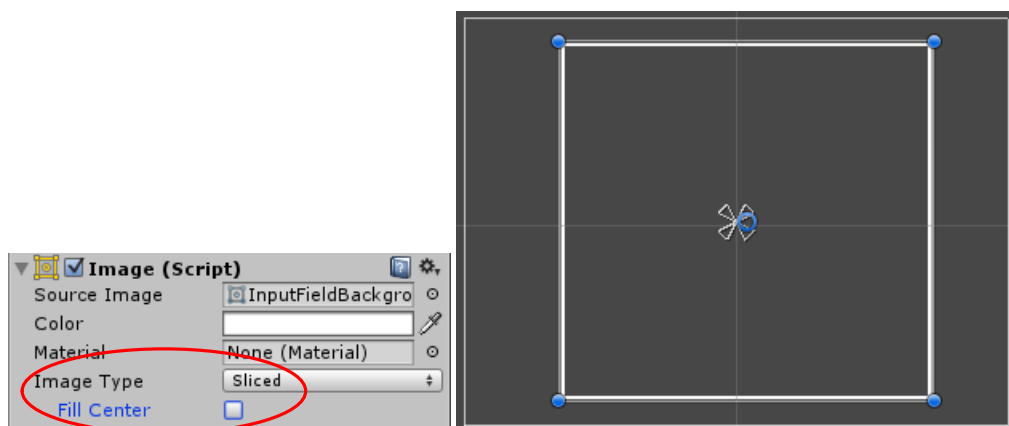


图 2-44 图片的显示类型：Sliced（只显示切片的边缘）

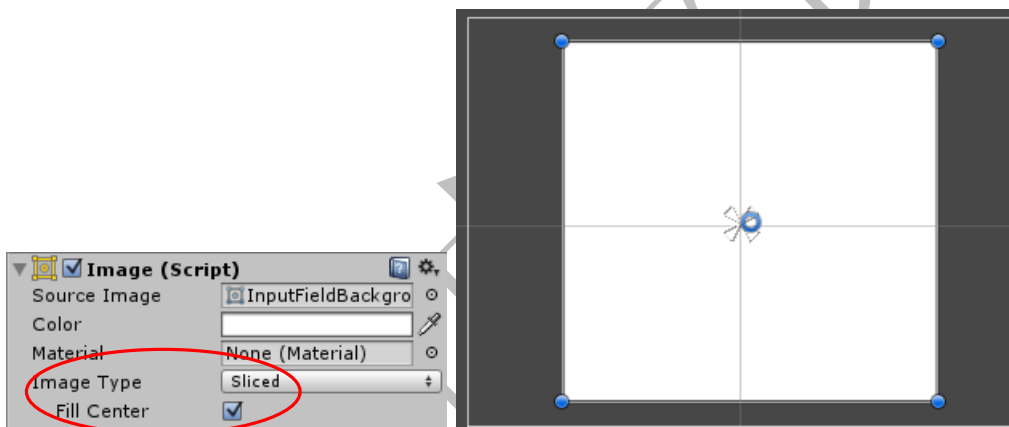


图 2-45 图片的显示类型：Sliced（完整显示中间的切片）

- ❑ **Tiled:** 此种类型的 Sprite 一般尺寸较小，如图 2-46 所示。为使此 Sprite 填满整个图片控件，就会在保持 Sprite 尺寸不变的前提下不断重复，就像是在铺地板砖一样，如图 2-47 所示。



图 2-46 Tiled 类型的 Sprite（大小为 64×75）

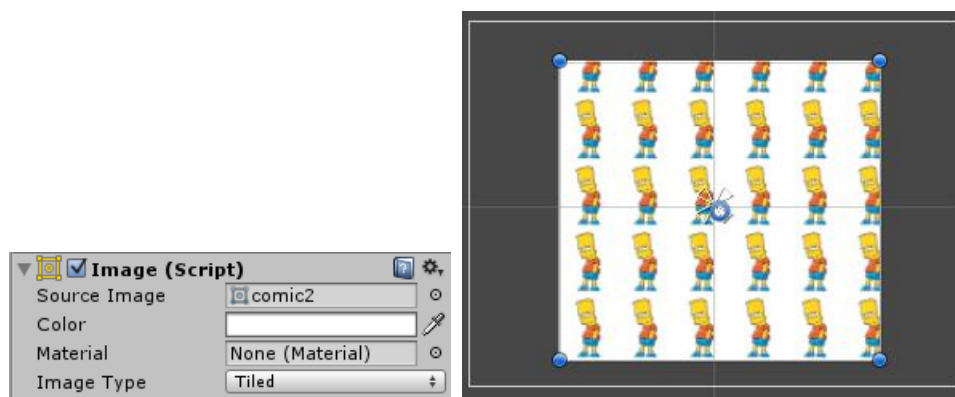


图 2-47 图片的显示类型：Tiled（不断重复，直至充满整个图片控件）

- ❑ Filled: 此种类型的 Sprite 与 Simple 相似，但是它可以表现出一种“从无到有”的呈现过程，如图 2-48 所示。

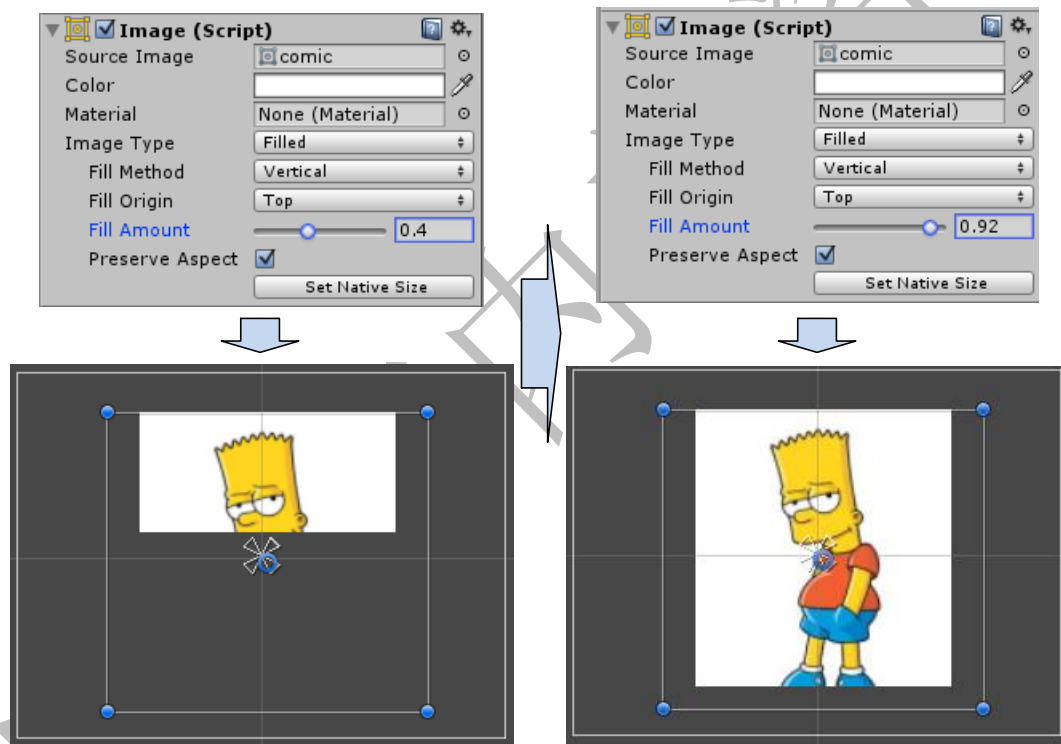


图 2-48 图片的显示类型：Filled（从上到下，逐渐呈现）

提示：“呈现方式”也叫“填充方式”，由 Fill Method 属性决定，可选项有 Horizontal、Vertical、Radial 90、Radial 180 和 Radial 360，效果如图 2-49 所示。Fill Origin 决定了填充操作的起点，Fill Amount 决定了填充的进度。

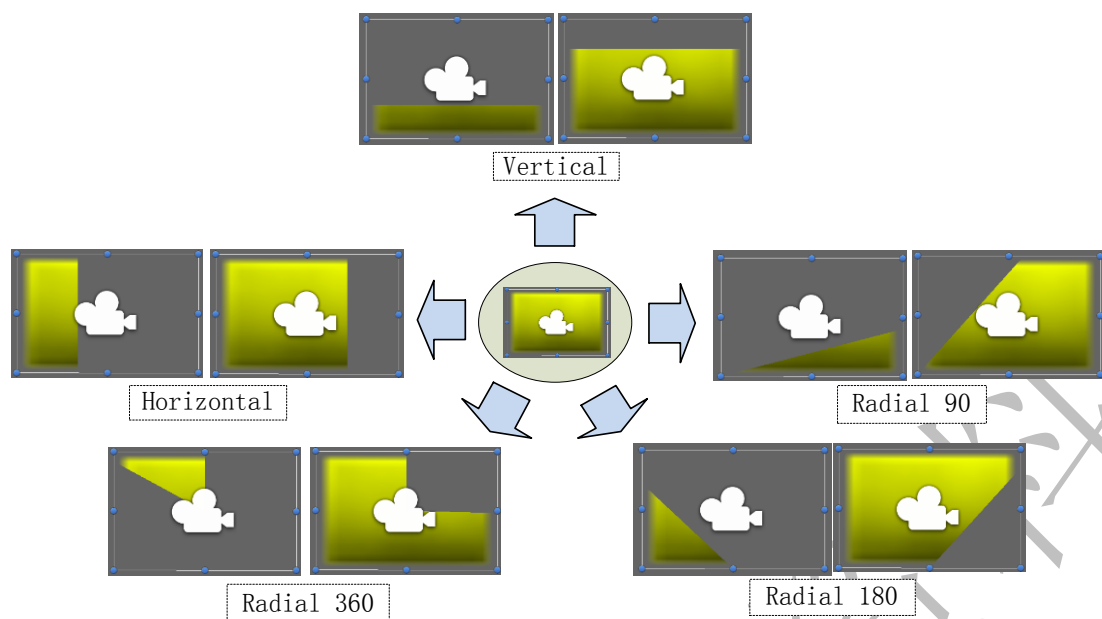


图 2-49 图片填充方向的 5 种效果

- ❑ **Set Native Size:** 单击此按钮，将使得图片控件的大小主动调节到与 Sprite 原始大小一致，如图 2-50 所示。

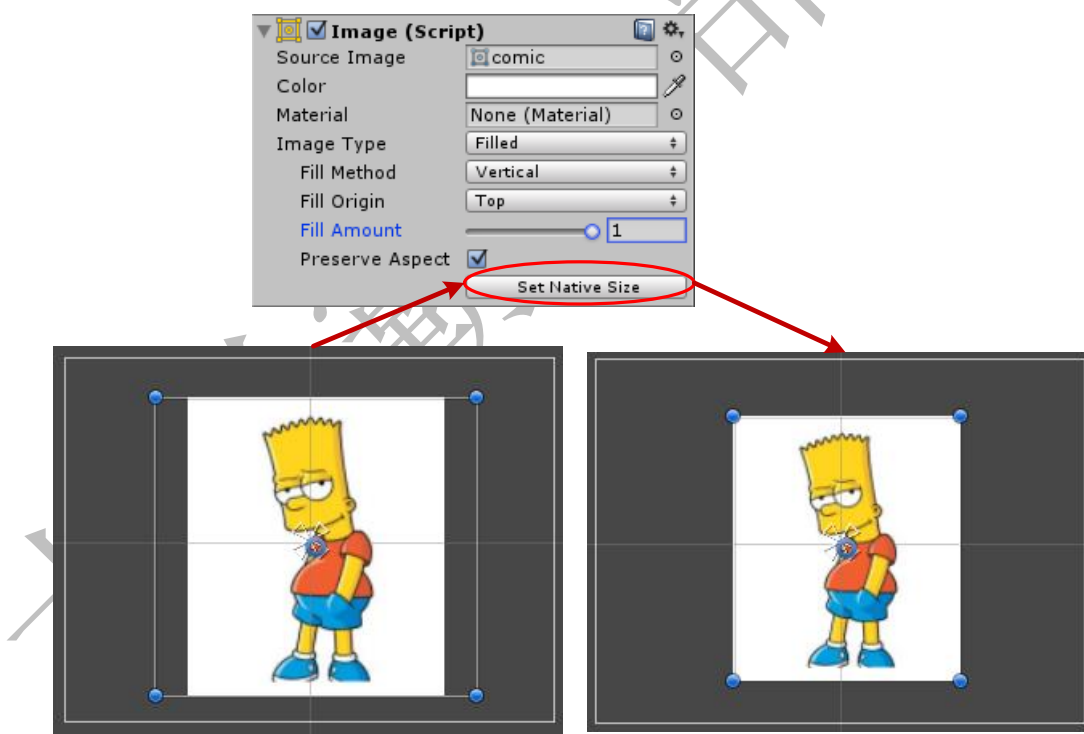


图 2-50 图片控件的大小主动调节到与 Sprite 原始大小一致

2.3.3 显示图片 2——Raw Image(Script)

Raw Image(Script)组件与 Image(Script)组件的功能类似，如图 2-51 所示，用于向玩家显示不可交互的图片信息，常作为游戏场景的装饰。但 Raw Image(Script)组件与 Image(Script)

组件上的属性并不完全一致，Raw Image(Script)组件如图 2-52 所示。

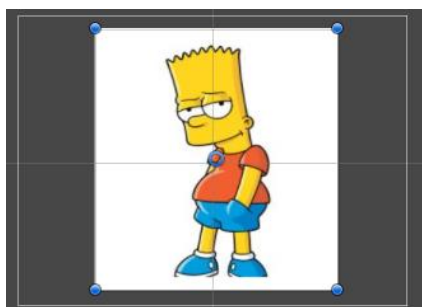


图 2-51 Raw Image(Script)组件的图片显示效果

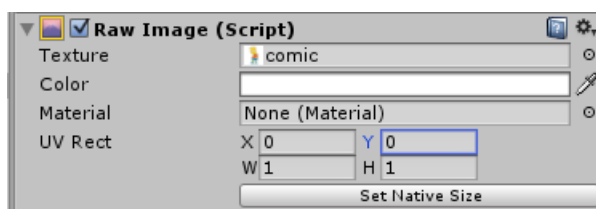


图 2-52 Raw Image(Script)组件

Raw Image(Script)组件各属性作用的说明如下：

❑ Texture: 指定 Raw Image(Script)组件要显示的图片；

注意：Raw Image(Script)组件所显示的图片，可以是任何类型，而不单单只是 Sprite 类型。

❑ Color: 设置图片的颜色；

提示：对此属性的设置，会改变图片显示的主色调，就像是在用有颜色的光照射图片。

❑ Material: 设置图片控件的材质；

❑ UV Rect: 令图片中的一部分显示在 Raw Image(Script)组件里。X 和 Y 属性指定图片左下角的位置，W 和 H 属性指定图片右上角的位置，如图 2-53 所示。

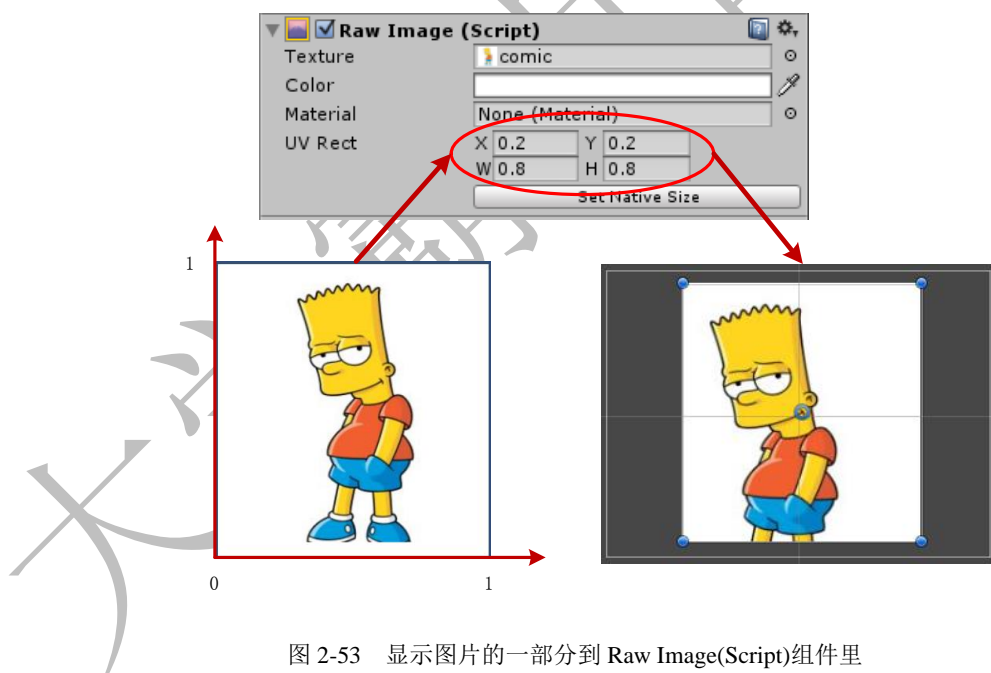


图 2-53 显示图片的一部分到 Raw Image(Script)组件里

2.3.4 遮罩效果——Mask(Script)

拥有 Mask(Script)组件的 UI 控件，可以限制其子对象的显示范围，即当子对象的显示范围明显大于父对象的显示范围时，游戏视图就只显示父对象范围内的子对象，其它部分自动隐藏。父对象是 Panel，子对象是 Image，为前者添加 Mask(Script)组件，作用效果如图 2-54 所示。

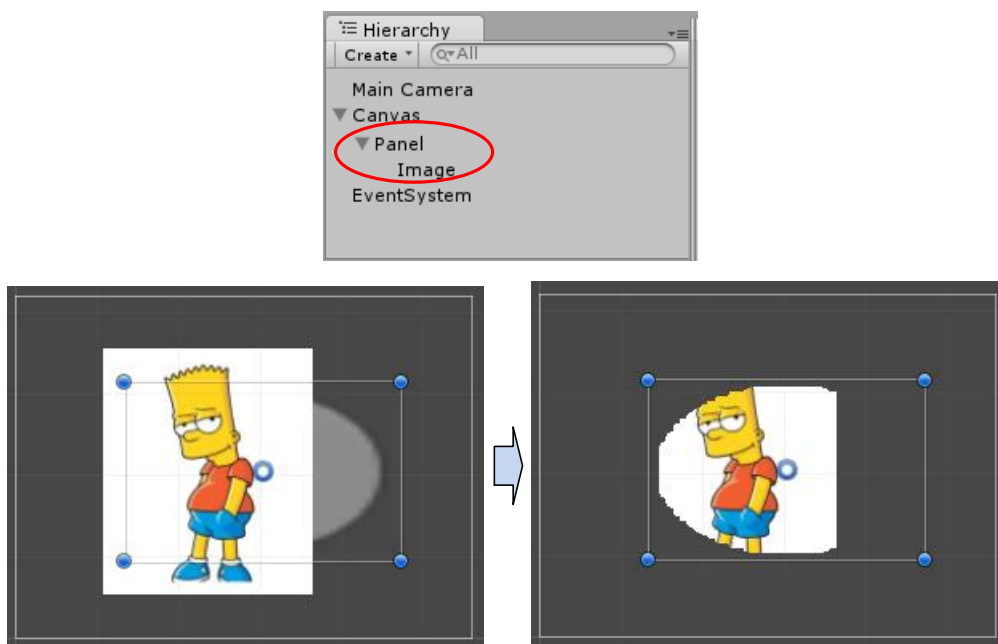


图 2-54 Mask (Script) 组件的作用效果

注意：添加了 Mask (Script) 组件的游戏对象，要想发挥出此组件的效果，要求此对象上必须包含 Image (Script) 组件。如图 2-55 所示，查看 Panel 上的所有组件，可以看到它默认就有 Image (Script) 组件，所以才能发挥出 Mask (Script) 组件的遮罩效果。

Mask (Script) 组件如图 2-56 所示。

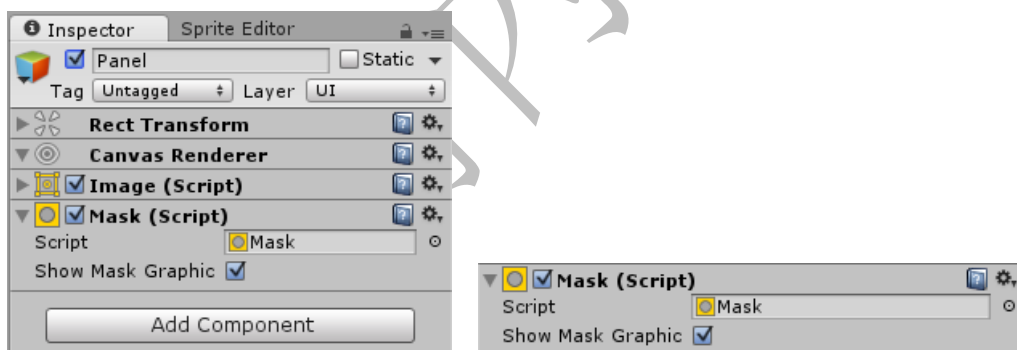


图 2-55 Panel 上的所有组件（包含了 Image (Script) 组件） 图 2-56 Mask (Script) 组件

☐ Show Mask Graphic: 此属性决定是否显示父对象上的图片，效果如图 2-57 所示；

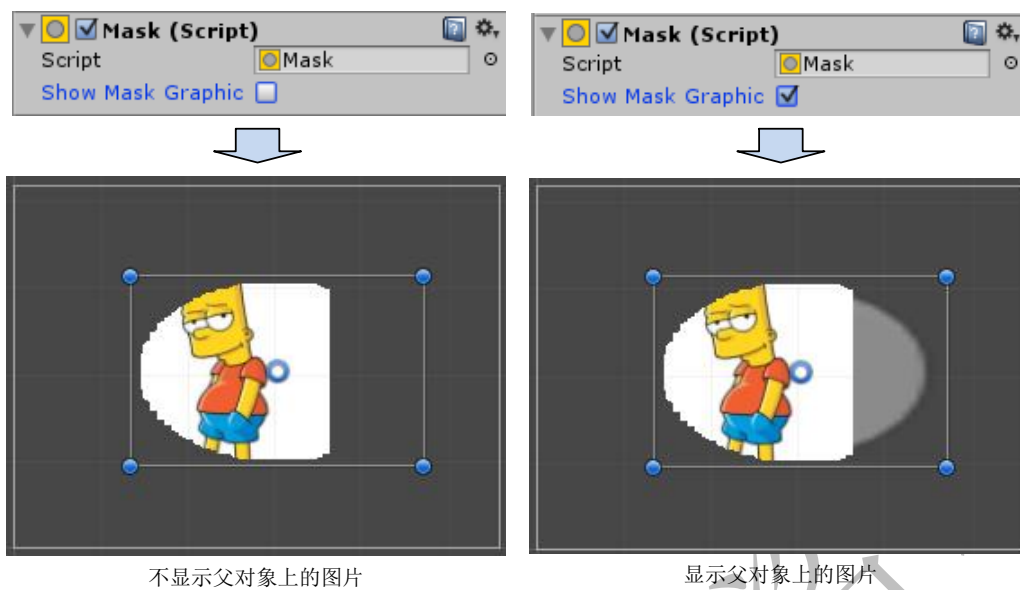


图 2-57 Show Mask Graphic 属性作用效果展示

大学霸内部资料