在Unity中需要截取屏幕画面用于分享，或者截取一些帧来做序列帧动画，今天分享一下如何来实现。

第一种方法：

1. **private** **static** **void** CaptureScreen()
2. {
3. Application.CaptureScreenshot("Screenshot.png", 0);
4. }

注：这个方法截取的是某一帧时整个游戏的画面，或者说是全屏截图吧，简单粗暴，分享到朋友圈之类可以用到。

第二种方法：使用Texture2d类下的相关方法，也可实现截图功能。

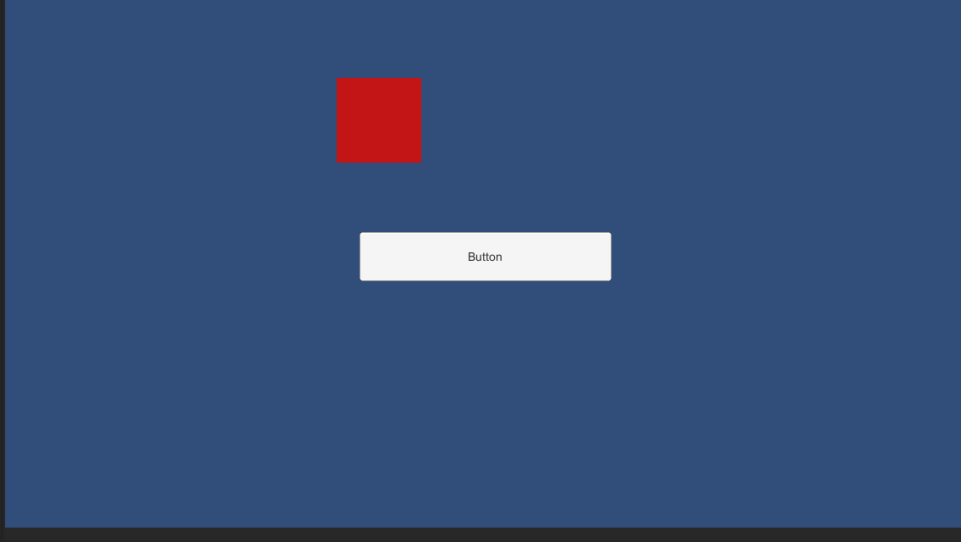
截全屏：

CaptureScreenshotRect( new Rect(Screen.width\*0f, Screen.height\*0f, Screen.width\*1f, Screen.height\*1f));

截中间4分之1:

CaptureScreenshotRect( new Rect(Screen.width\*0.25f, Screen.height\*0.25f, Screen.width\*0.5f, Screen.height\*0.5f));

注：读取的是整个Unity Game视图的画面，没什么实际用处。



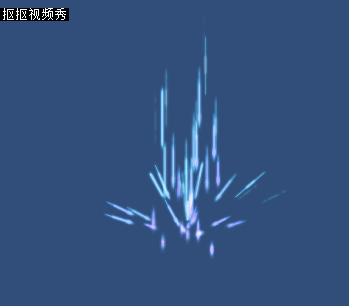
参考：<https://docs.unity3d.com/ScriptReference/Texture2D.ReadPixels.html>

1. **private** **static** Texture2D CaptureScreenshotRect(Rect rect)
2. {
3. // 先创建一个的空纹理，大小可根据实现需要来设置
4. Texture2D screenShot = **new** Texture2D((**int**)rect.width, (**int**)rect.height, TextureFormat.RGB24, **false**);
6. // 读取屏幕像素信息并存储为纹理数据，
7. screenShot.ReadPixels(rect, 0, 0, **false**);
8. screenShot.Apply();
10. // 然后将这些纹理数据，成一个png图片文件
11. **byte**[] bytes = screenShot.EncodeToPNG();
12. **string** filename = Application.dataPath + "/Screenshot.png";
13. System.IO.File.WriteAllBytes(filename, bytes);
14. Debug.Log(**string**.Format("截屏了一张图片: {0}", filename));
16. AssetDatabase.Refresh();
17. **return** screenShot;
18. }

3、第三个方法，可以针对某个相机进行截图。

1. /// <summary>
2. /// 对相机截图。
3. /// </summary>
4. /// <returns>The screenshot2.</returns>
5. /// <param name="camera">Camera.要被截屏的相机</param>
6. /// <param name="rect">Rect.截屏的区域</param>
7. **private** **static** Texture2D CaptureCamera(Camera camera, Rect rect)
8. {
9. // 创建一个RenderTexture对象
10. RenderTexture rt = **new** RenderTexture((**int**)rect.width, (**int**)rect.height, 0);
12. // 临时设置相关相机的targetTexture为rt, 并手动渲染相关相机
13. camera.targetTexture = rt;
14. camera.Render();
16. //ps: --- 如果这样加上第二个相机，可以实现只截图某几个指定的相机一起看到的图像。
17. //ps: camera2.targetTexture = rt;
18. //ps: camera2.Render();
19. //ps: -------------------------------------------------------------------
21. // 激活这个rt, 并从中中读取像素。
22. RenderTexture.active = rt;
23. Texture2D screenShot = **new** Texture2D((**int**)rect.width, (**int**)rect.height, TextureFormat.RGB24, **false**);
24. // 注：这个时候，它是从RenderTexture.active中读取像素
25. screenShot.ReadPixels(rect, 0, 0);
26. screenShot.Apply();
28. // 重置相关参数，以使用camera继续在屏幕上显示
29. camera.targetTexture = **null**;
31. //ps: camera2.targetTexture = null;
33. RenderTexture.active = **null**; // JC: added to avoid errors
34. GameObject.Destroy(rt);
36. // 最后将这些纹理数据，成一个png图片文件
37. **byte**[] bytes = screenShot.EncodeToPNG();
38. **string** filename = Application.dataPath + "/Screenshot1.png";
39. System.IO.File.WriteAllBytes(filename, bytes);
40. Debug.Log(**string**.Format("截屏了一张照片: {0}", filename));
41. AssetDatabase.Refresh();
42. **return** screenShot;
43. }

上述方法会留下背景，如果需要截取半透明特效，参考以下：

**

1. IEnumerator CaptureFrame()
2. {
3. Time.timeScale = 0;
5. yield **return** **new** WaitForEndOfFrame();
7. **string** filename = String.Format("{0}/{1:D04}.png", realFolder, ++currentIndex);
8. Debug.Log(filename);
10. **int** width = Screen.width;
11. **int** height = Screen.height;
13. RenderTexture blackCamRenderTexture = **new** RenderTexture(width, height, 24, RenderTextureFormat.ARGB32);
14. RenderTexture whiteCamRenderTexture = **new** RenderTexture(width, height, 24, RenderTextureFormat.ARGB32);
16. exportCamera.targetTexture = blackCamRenderTexture;
17. exportCamera.backgroundColor = Color.black;
18. exportCamera.Render();
19. RenderTexture.active = blackCamRenderTexture;
20. Texture2D texb = GetTex2D();
22. //Now do it for Alpha Camera
23. exportCamera.targetTexture = whiteCamRenderTexture;
24. exportCamera.backgroundColor = Color.white;
25. exportCamera.Render();
26. RenderTexture.active = whiteCamRenderTexture;
27. Texture2D texw = GetTex2D();
29. **if**( texw && texb )
30. {
31. Texture2D outputtex = **new** Texture2D(width, height, TextureFormat.ARGB32,**false**);
33. **for**( **int** y = 0; y < outputtex.height; ++y )
34. {
35. **for**( **int** x = 0; x < outputtex.width; ++x )
36. {
37. **float** alpha;
38. alpha = texw.GetPixel(x, y).r - texb.GetPixel(x, y).r;
39. alpha = 1.0f - alpha;
40. Color color;
41. **if**( alpha == 0 )
42. {
43. color = Color.clear;
44. }
45. **else**
46. {
47. color = texb.GetPixel(x, y);
48. }
49. color.a = alpha;
50. outputtex.SetPixel(x, y, color);
51. }
52. }
54. **byte**[] pngShot = outputtex.EncodeToPNG();
55. File.WriteAllBytes(filename, pngShot);
57. pngShot = **null**;
58. RenderTexture.active = **null**;
59. DestroyImmediate(outputtex);
60. outputtex = **null**;
61. DestroyImmediate(blackCamRenderTexture);
62. blackCamRenderTexture = **null**;
63. DestroyImmediate(whiteCamRenderTexture);
64. whiteCamRenderTexture = **null**;
65. DestroyImmediate(texb);
66. texb = **null**;
67. DestroyImmediate(texw);
68. texb = **null**;
70. System.GC.Collect();
72. Time.timeScale = originaltimescaleTime;
73. }
74. }
76. **private** Texture2D GetTex2D()
77. {
78. **int** width = Screen.width;
79. **int** height = Screen.height;
80. Texture2D tex = **new** Texture2D(width, height, TextureFormat.ARGB32, **false**);
81. tex.ReadPixels(**new** Rect(0, 0, width, height), 0, 0);
82. tex.Apply();
83. **return** tex;
84. }

源码：https://github.com/l2xin/ExportEffectPng