目录

[1． 屏幕适配方式 2](#_Toc519594234)

[2． UI刘海适配方式 2](#_Toc519594235)

[3． 动效 2](#_Toc519594236)

[4． 背景全屏背景图片 2](#_Toc519594237)

[5． 文字图片适配 2](#_Toc519594238)

[6． GUIPanel（UI特效分层和刘海适配） 3](#_Toc519594239)

[7． 排列控件 GNewList。。 4](#_Toc519594240)

[8． 按钮 Btn 5](#_Toc519594241)

[9． IOSToggle Button 5](#_Toc519594242)

[10． 滑动窗口ScrollRect 6](#_Toc519594243)

[11． 通用树形下拉菜单 GNewTabList 7](#_Toc519594244)

[12． 颜色渐变 GGradient 7](#_Toc519594245)

[13． ImageModel使用 7](#_Toc519594246)

具体用法请看自定义组件中的文档

### 屏幕适配方式

制作时以16：9无刘海环境为准，以后，UI预设必须在Game 16：9模式无刘海环境下，要在4：3，16：9 ，2436 \* 1125下分别做适配检查（IPX刘海也要检查）；

以UE文档为准，按照其中要求划分根节点Frame，设定锚点，增加刘海适配方式；

### UI刘海适配方式

昵称 躲刘海；以UE文档为准，一般：

1. 对话框适的UI只居中
2. 左右贴边的界面如果受刘海威胁，需要添加GUIPanel（参考6。GUIPanel）
3. 贴边躲刘海的界面，如果有背景，需要将背景向反方向延伸200px，以免适配时穿崩；
4. 一般，一个全屏界面会同时存在左右都需要适配的情况，需要建一个全屏Frame\_Center，把相应节点包进去，再加GUIPanel；

验证方法：Game选择2436 \* 1125（或实机分辨率） 在PanelRoot上的UISizeAdapter，选择Fake Device -> IPhongX\_L (IphongX左边刘海), IPhongX\_R ( IphongX右边刘海) 启动游戏；

### 动效

动效不可以放在根节点，不可以放在IPhoneX式配节点之外；

### Android返回键

[程序]UI内需要处理Android返回键的，要重载PanelBase的 HandleEscapeKey函数，

默认操作是关闭界面;

最后返回false 代表此UI不截停返回流程, true代表返回流程中止;

[策划]出UI文档时要规划UI中的Android返回键的使用；

[测试]PC上按ESC

### 背景全屏背景图片

基本分三种：1。只要全屏2。局部拉伸3。全屏覆盖并保持宽高比

分别对应 TERAMobile\Assets\Prefabs\UIExamples\ UI\_Frame. prefab \Img\_Cheap\_BG(example)

TERAMobile\Assets\Prefabs\UIExamples\ UI\_Frame. prefab \ Img\_Common\_BG(example)

TERAMobile\Assets\Prefabs\UIExamples\ UI\_Frame. prefab \ Img\_FullScreen(example)

### 文字图片适配

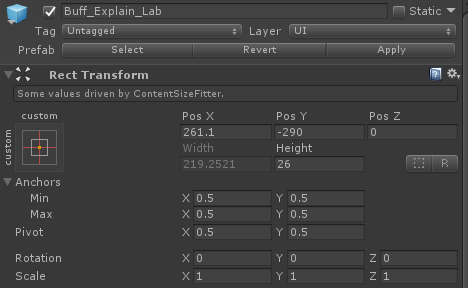
只要将Text节点加上ContentSizeFilter，再将图片的锚点设置到文字边缘即可。

注意Text，Image的Pivot设置，Text的溢出方式，ContentSizeFilter的限制方式都要根据需要设置；

例













也可参考TERAMobile\Assets\Prefabs\UIExamples\ UI\_Frame.prefab\ Frame\_Title(example)

### GUIPanel（UI特效分层和刘海适配）

有时面板间要插特效，需要UI分层；有时面板间要躲刘海，需要刘海适配；

整理好UI结构，在要控制的节点使用GUIPanel来控制层级和适配，同时设置其选项；

DontOverrideSorting ：勾选此项则不更改层级（只需刘海适配时一般勾选此项）；

OrderOffset：未勾DontOverrideSorting时，可调整层级；1-5可用；

IsIgnoreResizing ：勾选此项时，适配时会忽略此物体（只需UI分层时一般勾选此项）；

IsKeptFullScreen ：强迫使用屏幕的尺寸计算锚点，否则使用安全区（父节点躲了刘海子节点却要全屏的要勾选此项）；

IsNeedBlockRay ：是否阻挡射线（需要操作的面板要勾选）；

注：UISfx接口播放特效时，会按接口的target和offset设置层级；只需调整GUIPanel的OrderOffset即可实现不同的遮挡关系；（不要让特效或面板的总偏移超出0-5）

注：用作刘海适配时，同级、上级不可以有Transform动效和导致Transform变化的脚本；

例：



图中做了特效分层和刘海适配；背景保留全屏尺寸，UI内容左侧缩进；背景和人物之间空出了1个order，用于插入特效；

### 排列控件 GNewList/GNewListLoop

只需同级自动排列，而且元素没事么交互事件，可以使用（UGUI自带）GridLayoutGroup组件来实现，只要设置好cell宽高即可；设置好子节点的对齐位置；

也可以使用VerticalLayOutGroup和HorizontalLayOutGroup，注意在子节点上挂LayOutElement组件来标识大小。

GNewList，GNewListLoop:

所有Item等宽等高，可设置行列数；

超过10个数据的列表就可以考虑使用循环列表GNewListLoop，相对应的是有限列表GNewList；

SingleSelect模式下，如果要使用GNewList作选中效果，需要在OnSelectItem 调用Selection来设置；

~~级联排序再用GLayOut；级联动态排序（子节点随时影响排序）用GLayOutAuto，非动态排序可以考虑用GNewGrid，GNewList；但此类脚本会生成大量附加信息，所以如有可能尽量使用前面简单的实现；~~

~~GLimitedList和GUnLimitedList已经废弃；~~

~~GLayout，GLayOutAuto ☹不推荐使用~~

~~元素不等宽高，或需要深层递归；~~

~~尽量避免使用GLayOutAuto，考虑使用Anchor和Content Size Fitter（改大小）组件（UGUI内置）来实现；GLayOut可以使用，但最好在Lua中手动调用ChangeLayOut，不要依赖于它的递归（父子）连带功能，以后可能会去掉此功能；~~

~~只有自身改变自动影响他人的物体才挂GLayOutAuto；~~

~~☺完全由自己逻辑触发的排列，回调简单的，建议自己在逻辑中完成；~~

~~无论是LayoutGroup 还是 GLayout都是要监听非常复杂的Unity活动的，有Rebuild的开销；~~

~~而GNewList这样的虽没有这样的开销，但要注册很多Lua回调；~~

### GNewTable/GNewLayoutTable/GNewTabList

**GNewTable：**

单列列表，Item不等高，据有两个children类GNewLayoutTable/GNewTabList

**GNewLayoutTable：**

Content 和 Item上带有UGUI的 VerticalLayoutGroup 来排列列表；

如果要使用GNewList作选中效果， 只有IsMultiSelect模式下需要在OnSelectItem 调用Selection来设置；

参考预设TERAMobile\Assets\Prefabs\UI\ QuestN.prefab

**GNewTabList:**

通用树形下拉菜单，单列Tab表（可滑动），~~每一项可展开为最大5.5行的小滑窗~~；

如果要使用GNewList作选中效果，不需要在OnSelectItem 调用Selection来设置；

参考预设TERAMobile\Assets\Prefabs\UIExamples\ TabList.prefab

详细接口见 Tera项目中UGUI自定义组件的使用说明 下的文档；

### 按钮 Btn

制作Button时组件一律使用GButtonColorElastic，根节点不挂图，只能加响应区；

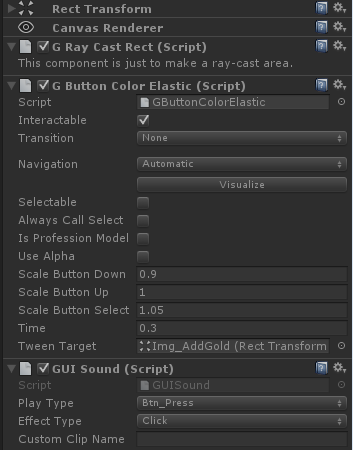
TweenTarget设置为要缩放的物体，不能设自己（不能响应区）；

响应区使用GRayCastArea组件（☹不要挂透明图），此组件用于提供Button等物体的相应区域，不会提交顶点和绘制；（此组件不能与Graphic系的Text、Image共存）

Transition不需要时选None；

不需缩放时EnableScale 不勾选；

结构范例：



不规则钮需要将GRayCastArea替换成GRaycastPoly

使用时，在GRaycastPoly所添加的2Dploygon collider中完成形状的制作；此组件在Transform改变时会产生额外开销；

常见Button参考TERAMobile\Assets\Prefabs\UIExamples\ Btn\_\*.prefab



按钮三态的实现：

1．通用按钮，图片使用Common\_Btn\_xx，这种按钮置灰使用替换图片的规则，如果不合适请联系相关策划询问，动效待定



2．特殊图片按钮，图片使用该功能提供的图片，这种图片置灰使用gary材质置灰（Lua下有通用的方法），具体灰度，功能负责人跟踪调整



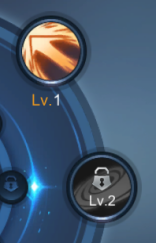
PS：所有置灰都不可以点击，如果需要反馈内容，请不要置灰或换成灰图

## 单选内容Toggle

各种理论上应当使用Toggle实现；

Transition不需要时选None，灵活运用Fade功能，不需要一样None；

Toggle不应有置灰情况，如果需要表现置灰，同样遵循通用更换灰图，特殊图标材质置灰的规则。

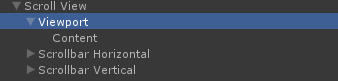


### IOSToggle Button

 见通用组件

### 滑动窗口ScrollRect

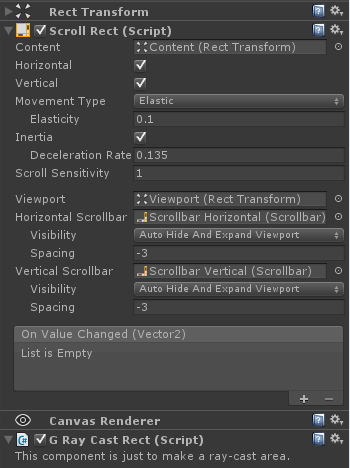
结构范例：使用三层结构



第一层：放置Scroll和响应区（如果有图也可以

把图片设置成响应区）

Scroll应当设置Viewport和Content；



第二层：放置Viewport和Scrollbar

Viewport挂2DMask；

第三层: Content下可挂GNewList等排列组件；（GNewList的item template不需要做处理会自动隐藏；其他组件的item基本上用一个节点包住一起隐藏；）

### 颜色渐变 GGradient



加上试试，没啥好说的

### ImageModel使用

除对话界面外，ImageModel的节点命名必须是“Img\_Role\_数字ID”，ID从最好1开始，除非复用参数否则不能重名；

在Lua中使用时，如果是主角，尽量使用GUITools.CreateHostUIModel接口进行创建；

创建完用self.\_UIModel4ImgRender:AddLoadedCallback(function()

self.\_UIModel4ImgRender:SetModelParam(self.\_PrefabPath, profession)

end)

设置镜头参数，

CuiModel.SetModelParam(self,@1,@2)

参数:@1 是UI的prefab路径;

参数@2所加载是角色时为ProfessionID, 所加载是其他时为AssetPath;

示例：

if uiItem.\_UIModel4ImgRender == nil then

local model = nil

if hostId == PlayerInfo.RoleID then

model = GUITools.CreateHostUIModel(uiItem.\_RoleImg, EnumDef.RenderLayer.UI, nil)

profession = game.\_HostPlayer.\_InfoData.\_Prof

else

local ModelParams = require "Object.ModelParams"

local params = ModelParams.new()

params:MakeParam(PlayerInfo.Exterior, PlayerInfo.Profession)

model = CUIModel.new(params, uiItem.\_RoleImg, EnumDef.UIModelShowType.All, EnumDef.RenderLayer.UI, nil)

profession = PlayerInfo.Profession

end

uiItem.\_UIModel4ImgRender = model

self.\_UIModel4ImgRender:AddLoadedCallback(function()

self.\_UIModel4ImgRender:SetModelParam(self.\_PrefabPath, profession)

end)