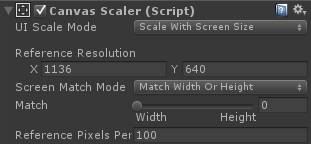
# 新UI迭代制作规范

## 新UI的制作流程

1. 制作新UI prefab
2. 打开MainUI场景，检查Canvas节点的CanvasScaler 是  
   ScaleWithScreenSize,1136x640，screen Match Mode = Expand，来做；  
     
   PanelRoot节点四角拉伸，边距为0；如有其他查看UI组件使用指南；  
     
   新UI预设的命名规则是UI\_XX，放在Prefabs/UI，（如果是替换老UI, 则同时将替换的Panel\_XX预设移动到Prefabs/UI\_Old（打资源时忽略））
3. 全屏UI需要按照UE的设计版式查找版式预设; 版式预设以通用UI的形式存在于Assets〉Prefabs〉UITemplates目录下UI\_TPL开头的预设, 其中包含了统一的屏幕试配和界面动画, 以其为基础构建UI预设; 详细见**[必须看]UI通用组件.docx**第1部分；非全屏UI要注意根节点的试配;
4. 要注意检查UI的各项试配,包括屏幕试配, 图片试配, 文字图片试配, 列表排列试配; 主要是确保UI跟据显示长宽比不同的正确性, 请参考**[必须看]UI组件使用指南.docx** 第1,4,5部分; 刘海试配指界面随手机屏幕样式(如刘海)变化的工能,要使用GUIPanel, 使用时请参考**[必须看]UI组件使用指南.docx**第2部分;做完预设要在16：9和2236\*1080 下验证效果；
5. 在做特效时, 会涉及到UI层级, 不能使用Canvas组件,要使用GUIPanel, 使用时请参考**[必须看]UI组件使用指南.docx** 第6部分;
6. 通用组件是已经制作好的统一样式的功能单元, 遇到统一样式的按钮等组建时,要询问UE使用哪种通用组件, 所有通用组件见**[UI]通用控件目录表.xlsx**, 使用时参考**[必须看]UI通用组件.docx**;
7. 对于C#编写的组件功能, 如List Button 等, 使用时请参考**[必须看]UI组件使用指南.docx**;
8. 动效, 有时美术会要求制作动画效果; 参考**UI动效（功能解锁制作过程）.docx**和**UI动效组件说明.docx;**
9. 编写Lua脚本
10. 打包生成的AssetBundle Path定义在客户端 \GameRes\Lua\Data\ResPath中以便lua中引用。客户端引用时，用全局的Path引用定义的路径即可。
11. 生成AssetBundle时会在.\GameRes\Lua\GUI\ObjectCfg对应生成一份同名lua变量哈希值文件，用以从预设获取对象，需要添加上传。
12. 在GUI目录增加CPanelBase的派生类以控制界面逻辑，按规则命名，例：预设本体名称为：UI\_Equip，则定义的Path.UI\_Equip 为其路径的变量名称，Lua文件名CPanelUIEquip.Lua。
13. 在UISettingCfg中增加相应配置，参考新UI层级管理的说明；
14. 打测试资源, 进前端测试;

## 2.美术资源管理

参考[必须看]UI图片资源管理.docx

## 3.规则和注意事项

1. 界面元素命名规范

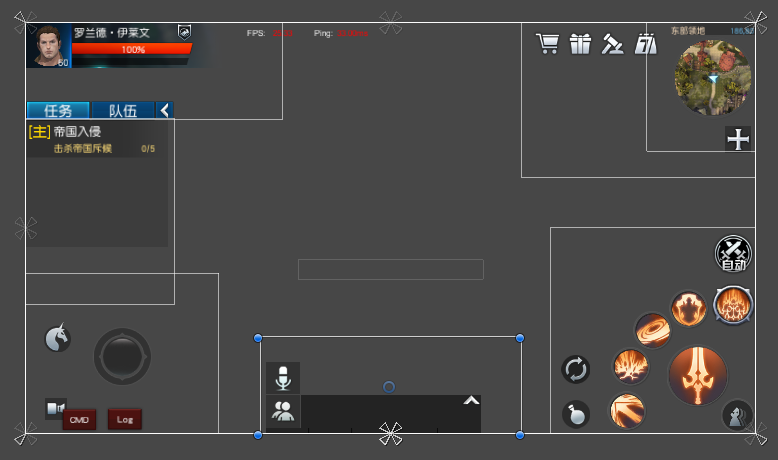
前缀\_名称，单词首字母大写；

前缀

* Panel\_Xxx,界面的根节点 \_UI\_
* Frame\_Xxx，界面内分组父节点
* Tab\_Xxx，界面内子页面父节点
* Img\_Xxx,图片
* Btn\_Xxx,按钮
* Lab\_Xxx,文字
* List\_Xxx,列表 \_Property\_，父节点用ScrollView\_Xxx
* Menu\_Xxx,树状下拉菜单 \_Property\_,父节点用 ScrollView\_Xxx
* Model\_Xxx,UI模型 \_Property\_
* Sld\_Xxx,滑动进度条
* Scl\_Xxx,滑动条
* Rdo\_Xxx,toggle有Group的
* Chk\_Xxx,toggle没Group的
* Grp\_Xxx,选项组
* Drop\_Xxx,下拉菜单
* Input\_Xxx，输入框 \_Property\_
* Bld\_Xxx，血条
* (后面的tag还是建议打上去)
* 名称中不能有括号；

1. 注意选择anchor的对齐方式，例如角对齐，横向拉伸，全屏等等，不需随父节点变化的，选中心点；

例如，主界面每个区应该这样



按背景或屏幕拉伸的应该是这个或任一个方向拉伸的

不需要对齐的应该是

做完预设要在16：9和2236\*1080 下验证效果；

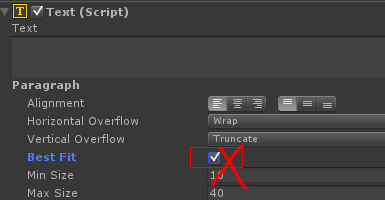
1. 元素太多或特殊需求（粒子特效）~~可以考虑手动添加Canvas（当然不要多）；勾选override sorting的话，同一个UI之下，Canvas的order数要在10以内；~~ 不要自己挂Canvas，用GUIPanel来解决，具体询问UE；

## 4.性能指南

1. 使用尽可能少的UI元素；在制作UI时，一定要仔细查检UI层级，删除不必要的UI元素，这样可以减少深度排序的时间以及Rebuild的时间；
2. 不要滥用GLayOutAuto，不要非常简单的排列也使用List；

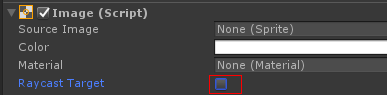
参考组件使用指南扩展docx；

1. 为优化重建，动静分离；经常需要更新的物体（频繁改变例如顶点、alpha、坐标和大小等的元素）尽量集中在一个GUIPanel内（有待扩展），避免因为几个物体而经常重画大量元素；
2. 为优化合并，如有可能同一类元素（如：同图集的底图，文字）放在同一节点下，不要相互插断；同层同类的元素尽量不重叠；参考[科普]ugui batch 规则和优化.txt
3. 谨慎使用（就不要用）Text的Best Fit选项，虽然这个选项可以动态的调整字体大小以适应UI布局而不会超框，但其代价是很高的，Unity会为用到的该元素所用到的所有字号生成图元保存在atlas里，不但增加额外的生成时间，还会使得字体对应的atlas变大。
4. 图片：图片4.png



谨慎使用（就不要用）Canvas的Pixel Perfect选项，该选项会使得ui元素在发生位置变化时，造成layout Rebuild。（比如ScrollRect滚动时，如果开启了Canvas的pixel Perfect，会使得Canvas.SendWillRenderCanvas消耗较高）

1. UGUI的touch处理消耗也可能会成为性能热点。因为UGUI在默认情况下会对所有可见的Graphic组件调用raycast。对于不需要接收touch事件的grahic，一定要禁用raycast。对于unity5以上的可以关闭graphic的Raycast Target而对于unity4.6，可以给不需要接收touch的UI元素加上canvasgroup组件



1. 做遮罩时，能够使用能用2D Rect Mask就不要使用Mask，具体好处去百度；
2. 做ScrollRect时，尽量不要省略ViewPort，否则LateUpdate可能会高；
3. 尽量使用（Lua） GUITools.setUIActive 来设置UGUI物体的可见性（单纯修改scale），可避免不必要的开销；（这点有更好的办法请告诉我）

但当物体具有逻辑和更新开销时才应该disable组件或动用SetActive.

11．使用GButtonColorElastic时会对tweenTarget产生缩放，注意不要使其缩小响应区域，否则对体验有非常不好的影响（子节点如带有RayCast属性也算响应区域）；

## 5.置灰，隐藏和锁的使用条件

1. 所有置灰使用在不可操作的情况下，如材料不足或已达上限。默认都不可以点击，如果需要反馈内容，请不要置灰或换成灰图，用文字提示同时操作无效
2. 隐藏或锁使用在功能未存在的状态，如功能未开启或不存在的情况下，如未到开启等级或选中无效的目标，使用隐藏还是锁来处理依据功能是否需要提前占位或玩家提前可知

## 6.UI分层

Layer：由低到高

GameWorld - 游戏世界中，与场景对象紧密相关的3D UI层，比如头顶血条，头顶名

RootPanel - （HUD）游戏主界面所在层，MainCity相关，比如技能操作界面、小地图、任务追踪等等

SubPanel - （普通界面）通过主界面打开的界面所在层，比如人物信息界面、背包界面

Dialog - 对话框

NormalTip - （对话框）普通提示界面层

Guide - （新手引导）新手引导专用

ImportantTip - （确认对话框）重要提示界面层，比如死亡复活界面、短线重连界面等

Debug - 调试辅助界面层，现在只有PanelDebug，不经常使用

每个层内OrderType：

Fixed 固定Order - 手动设置。1~40，81~100

Floating 自动Order - 在41-80，新开的浮在旧的上面。

## 7.其他（也很重要）

1. 参考UGUI扩展组件+编辑工具说明doc 和 UGUI培训ppt；
2. 注意阅读 Tera项目中UGUI自定义组件的使用说明 下的文档，适当选取组件；
3. 拖拉时尺寸临时参考M1Project\需求管理\A02\_美术需求\A03平面需求\A01\_UI需求\UE优化 下的通用尺寸(初版)
4. 还没想好。。。

有问题讨论。