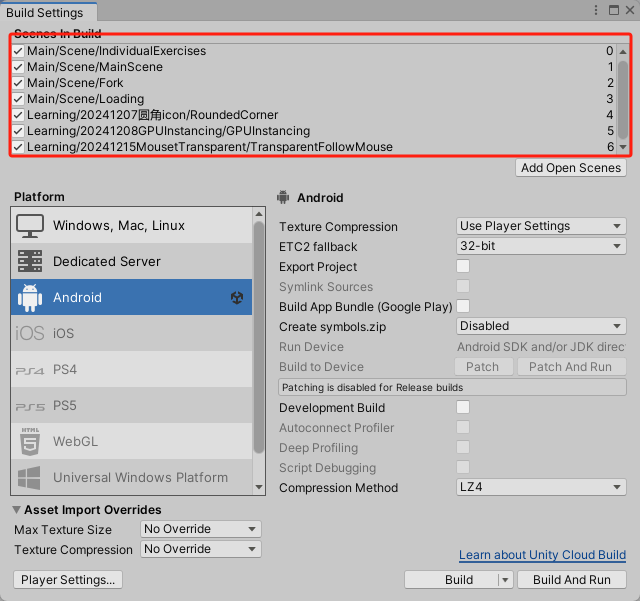
假设：

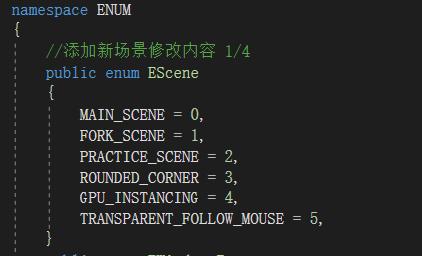
场景名为sceneName;

第一步：ctrl + shift + b，将场景sceneName拖拽至此



第二步：

找到这个枚举，在其中添加一个值，用于代码内部建立映射关系



第三步：（可选项，配置文件用于配置场景中的UI）

在StreamingAssets文件夹任意路径下创建xml文件。可以直接复制Config\SceneConfig\MainScene\interface\_xxx\_scene.xml，这个xml是一个模板，会在场景中创建一个层级为1的canvas，并在左上角创建一个返回按钮，用于返回上一个界面。

并将复制出来的xml名字中的xxx修改为场景名，或者其他不重复的名字。

之后新建一个变量，并将其默认值修改为xml对应的路径。

namespace TEN.GLOBAL

{

public static class Path

{

//添加新场景修改内容 2/4

public static class XMLCongigPath

{

public const string MainSceneConfig = @"\Config\SceneConfig\MainScene\interface\_main\_scene.xml";

public const string ForkSceneConfig = @"\Config\SceneConfig\MainScene\interface\_fork\_scene.xml";

public const string SelfPracticeSceneConfig = @"\Config\SceneConfig\MainScene\interface\_self\_practice\_scene.xml";

public const string RoundedCornerSceneConfig = @"\Config\SceneConfig\MainScene\interface\_rounded\_corner\_scene.xml";

public const string GPUInstancingSceneConfig = @"\Config\SceneConfig\MainScene\interface\_GPU\_instancing\_scene.xml";

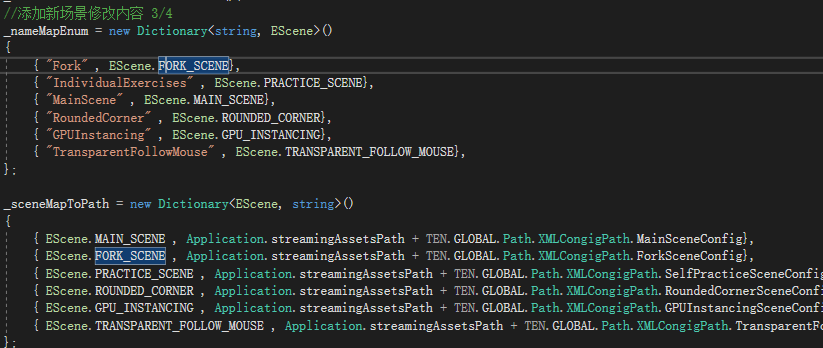
public const string TransparentFollowMouseSceneConfig = @"\Config\SceneConfig\MainScene\interface\_Transparent\_Follow\_Mouse\_scene.xml";

}

}

TODO : 后续可添加判断逻辑，当无法找到场景对应的xml时，可以直接读取interface\_xxx\_scene.xml中的内容以创建返回按钮。

第四步：



第一个字典\_nameMapEnum，键为场景名(sceneName)，值为场景枚举即第二步创建的内容；

第二个字典\_sceneMapToPath，键为场景枚举，值为配置xml路径。

根据以上描述以及之前的配置，新增键值对。

TODO : 后续可将映射关系写入xml，确保符合开闭原则。

第五步：

在StreamingAssets\Config\SceneConfig\MainScene\interface\_self\_practice\_scene.xml文件中（或者其他场景的配置文件中）添加一个按钮，同时将Event节点中的parameter参数修改为场景名字，如sceneName。

可选项：

Name节点用于修改button中的文字描述

<Content>

<Name>TransparentFollowMouse</Name>

<Type>1</Type>

<Image path="images/button1.png" />

<Anchor x = "0.5" y ="1"/>

<Layout>0</Layout>

<Position x="0" y="0" />

<Size width="300" height="100" />

<Event name="LoadScene" parameter="sceneName" />

</Content>

以上操作即可完成新增一个场景，同时新增一个按钮，并将点击事件设置为加载对应场景。

2024年12月15日17点35分

赵岩昌