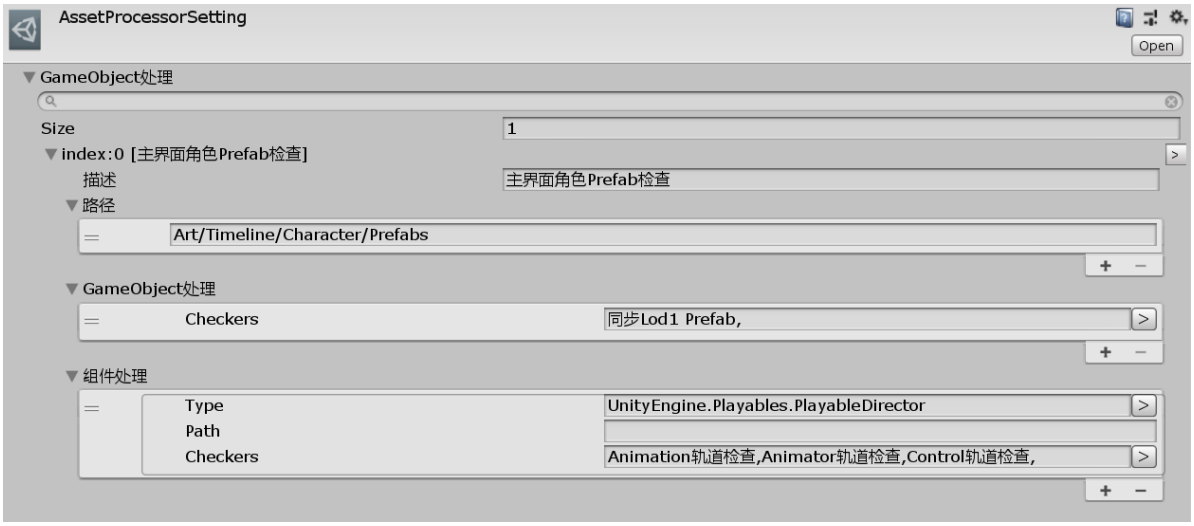
AssetProcessor

简介



目的

通过路径为切入，对游戏中的资源进行检查以及修正。通过标签和配置文件的方式，将具体的资源检查方法和检查模块进行解耦

实现方式

1. 注册GameObject或者Component处理等用于检查资源的方法
2. 在配置文件中定义指定路径下需要使用的资源检查方法
3. 菜单Asset/AssetProcessor/Check(Repair)，Check方法一般仅输出检查日志，Repair会进行相关修正

GameObject处理

用于Prefab的检查，通过GameObjectChecker标签进行注册，后续可以支持更丰富的类型，例如贴图，动画等等

```
public struct GameObjectCheckerContext
{
    public Object sourceObject;//源Object
    public bool doFix;//是否进行修复
}

[GameObjectChecker("Sample1")]//'Sample1'为处理函数的id,以保证多个处理方法可以并用
public static void Checker1(GameObjectCheckerContext ctx)
{
    Debug.Log($"check1 game object:
{AssetDatabase.GetAssetPath(ctx.sourceObject)}");
}
```

Component处理

用于Component组件进行检查，其依赖于具体的Prefab，往往用于检查Component上一些属性的设置是否合理

```
public struct ComponentCheckerContext
{
    public Component component;//待处理组件
    public Object sourceObject;//源Prefab
    public bool doFix;//是否进行修复
}

[ComponentChecker(typeof(PlayableDirector), "Sample1")]//PlayableDirector为需要处理的组件类型
public static void Check1(ComponentCheckerContext ctx)
{
    var director = (PlayableDirector)ctx.component;//将object类型转型为需要处理的类型
    Debug.Log($"check1 director:{director.gameObject.name}");
}
```