**2024.12.21 周报**

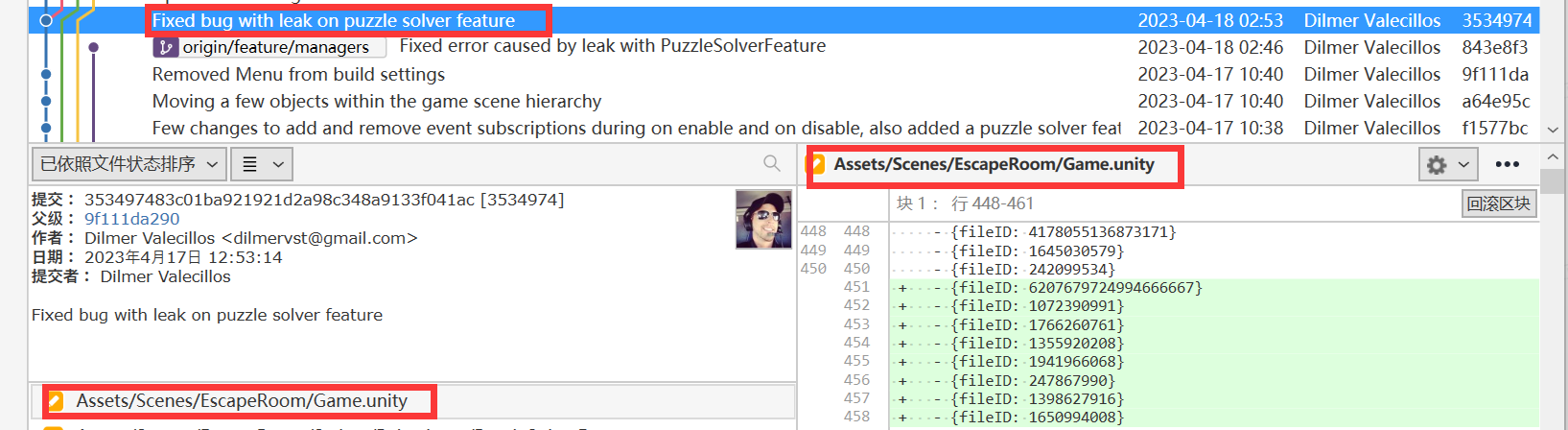
1. **通过VRAgent，测试数据集中的开源VR APP项目的潜在bug。**目前来看，VRAgent可以检测到一些bug，不过bug的种类尚且比较单一，基本上以物体和脚本的空引用为主。

  下面列举一个比较有代表性的**case study：**

  仓库 dilmerv/XRToolkitMVP

这是一个VR游戏恐怖游戏项目，场景中有诸如椅子、凳子等可交互物体，数量在20个左右，作者提供了主场景Game.scene。

  在最新commit版本上，VRAgent测试并未发现任何error。于是回退到最近的一次fix commit之前，VRAgent触发了两种报错。



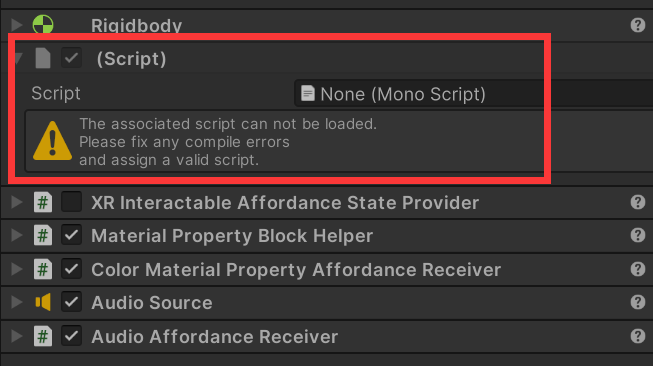
  报错1:

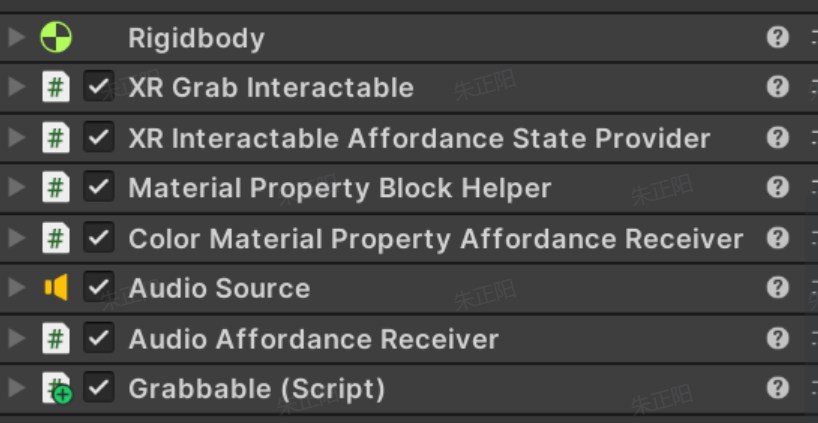
SerializedObjectNotCreatableException: Object at index 0 is null.

报错2：

NullReferenceException: Object reference not set to an instance of an object

  这两种报错，均是由于编辑器脚本在尝试访问某个对象时，发现该对象为 null。可能是场景中的某个组件被删除，但引用仍然存在。通过比较fix前后版本中Grabbable物体脚本组件，确认的bug是由于Grabbable物体的某个脚本丢失导致的：





1. 初步**总结了一下VRAgent 对新项目bug进行检测的工作流**。包括导入本地项目后配置插件，寻路烘焙，挂载脚本，运行Agent并触发bug，人工分析bug并修复等步骤。

<https://icnw6pwioyaz.feishu.cn/sync/OiebdTq6fsZaQTbsnhKcYa17nyf>

Ps: 工作流图已改用Inkscape绘制的.svg格式矢量图

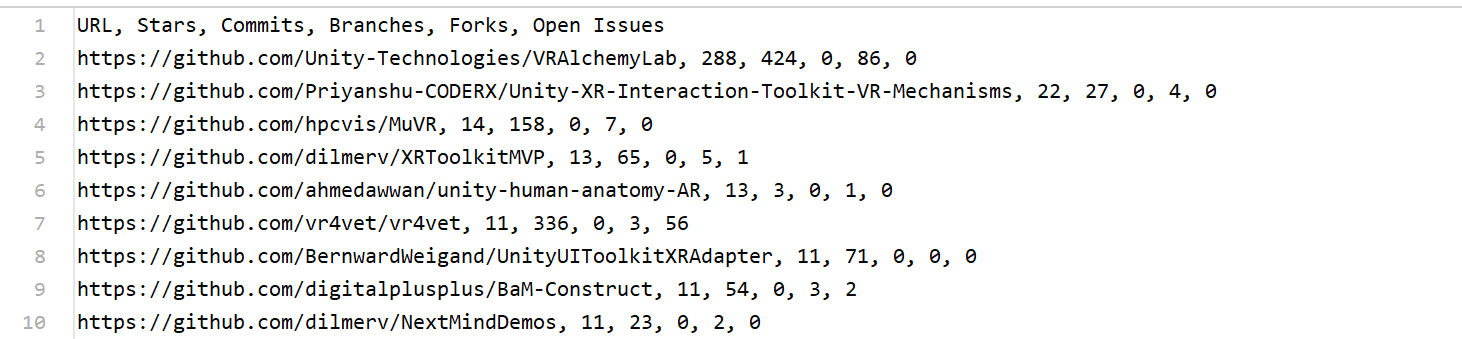
1. 在实际跑VRAgent的时候，发现有些物体会出现在导航网格以外的区域，导致VRAgent无法真正抓取到。因此进行修改，增加抓取保护时间机制，在过久等待以后则强制位移物体到Agent手中。
2. **开源VR APP项目数据集构建工作比较顺利**：利用Github提供的Api，爬取符合条件的开源仓库，并结合人工手动收集的方式。

原计划收集10个以上的仓库就很满意了，现在收集到有关'unity', 'unity-vr', 'vr'的开源仓库（条件相对较宽泛）**971**个；其中，进行二次筛选，排除不符合条件的仓库，剩下使用到 VR-IF 插件的仓库（条件更为严苛）共**165**个。这个数字比较客观，也为后续跑VRAgent收集错误取得进展。

关于不符合条件的仓库，定义为

1. 不包含 Packages文件夹，即不包含manifest.json 清单文件，这是一个用于说明项目使用的第三方库的清单。不包含它的仓库，是一个不规范的仓库，首先进行排除。
2. manifest.json的依赖中不包含'com.unity.xr.interaction.toolkit' 的，这意味着，该仓库并没有使用到 VR Interaction Framework框架，自然就无法进行VRAgent实验。

接着，对仓库清单按照'Stars, Commits, Branches, Forks, Open Issues'的关键字进行排序，Stars代表关注这个仓库的数量，Commits是提交数，可以认为数量较高的仓库质量更高。



本周优先选取质量较高的前20个仓库拉取并配置本地环境，过程中发现了**数据集仓库规范性和版本上新的问题**：

1. 若干仓库出现**库依赖和版本问题**，这是开发者和Unity版本更新速度较快导致的。（可以认为Unity的版本很多，前后在兼容上有时候会出现错误）。这些问题导致该项目暂时无法在本地跑通；
2. 少数仓库还是不够规范，打开会出现**低级的编译问题**。这从上图的清单中也可以看出来，很多仓库的star都是两位数出头。

总而言之，**这周在收集数据集的数量上，超出预期的多；但仓库本身大多数都是独立开发者，且由于Unity版本迭代快、插件速度迭代快、独立开发者缺少规范性等问题，一部分仓库难以直接在本地跑通，我暂时视作低质量仓库并进行剔除。**

下一周将重点回到VRAgent对高质量的仓库的bug检测上。