**项目总结报告**

日期：2021.9.13

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 16 | 项目名称 | 火树银花星满昼 |
| 编程语言 | C#, Java | 开发平台和框架 | Unity, Vue, SpringBoot |

|  |  |
| --- | --- |
| **软件需求特性** | |
| 1.是否实现了项目立项时的所有需求？列出实现的新增需求和未实现的需求。  项目已实现立项时的所有需求。  实现的新增需求有：用户将布置好的烟花场景通过礼物盒分享给他人；增加剧情动画串联不同模块，吸引用户兴趣。  部分新增的低优先级需求，如烟花场景的排行榜、烟花场景中加入手势识别，在考虑时间安排和协商必要性之后决定暂不实现。  2.采用哪种架构风格？哪些设计模式？  项目采用Client-Server架构，前端采用面向对象模式与基于组件的软件开发架构模式。  项目采用单例模式，在一些复杂场景中创建唯一的Manager对象，对场景中的其他对象进行管理。  3.技术方案有哪些亮点？  1. 对AR技术的应用进行拓展，用户不仅可以通过AR在现实场景中放置模型、观察模型，还能够和模型进行交互，在模型上布置各种烟花。  2. 使用Unity自带的粒子系统制作烟花，而非使用VFX插件（一种容易制作出好的烟花效果的插件），从而避免了VFX与AR技术在Android平台不兼容的问题。  3. 用户可以通过设置参数，调整自定义烟花的颜色、高度等，程序能够在运行时动态生成对应参数的烟花粒子模型。  4. 使用URP（通用渲染管道），一方面使得烟花能够展现出更多、更逼真的效果（如色彩溢出、爆炸效果），让Unity自带的粒子系统也能做出接近VFX插件做的烟花的效果；另一方面能够与AR场景协调，降低背景亮度以凸显烟花效果。  5. 使用一系列特定类储存自定义烟花的参数，而非直接储存自定义后的烟花模型，一方面节省了app内部的内存占用，另一方面减少了网络数据的传输量、服务器端储存的使用量。  6. 对于烟花继承类，使用自定义反序列化函数，而非使用序列化工具提供的方法，避免传输的数据依赖于序列化工具，同时避免安全问题（序列化工具在对继承类进行序列化、反序列化的过程中会加入一些特定标识，攻击者可以通过更改特定标识以攻击服务器）  7. 自定义图片压缩过程、压缩后储存图片的储存格式，而不是储存用户上传的原图片，尽可能地减少图片数据的储存占用（每个图片的储存数据均不超过70KB），同时保证图片烟花的显示不受明显影响  8. 在比较EasyAR和ARFoundation两个AR插件之后，由于ARFoundation可扩展性更好,性能更优，决定使用ARFoundation，  4.是否做了单元测试？是否做了系统功能测试？是否做了性能测试？是否做了兼容性等其他非功能测试？  完成了前端烟花以及相关类的单元测试、后端Service类的单元测试  对前、后端均进行了系统功能测试  性能测试方面，对比较重要的app端-后端通信的接口进行了压力测试，均能支持正常数据量下100并发的正常响应（平均响应时间不超过100ms）  平均FPS帧率: 30 | |
| **项目组成员对项目的贡献度（%）** | |
| 康艺潇：25%  林舒怀：25%  林祺龙：25%  丁逸舟：25% | |
| **软件度量** | |
| 软件代码行数（不包括注解行、空行和复用代码）： | 15653 |
| 复用他人代码行数： | 285 |
| 类的个数： | 94 |

|  |
| --- |
| **经验、教训和建议** |
| 1. 在进行需求风险较大的项目时，需要进行多次迭代，尽可能减少需求变更带来的风险。 2. 多人合作完成项目时，需要确定统一的代码和命名规范，确定每个人负责的模块和代码，不能随意修改其他成员的代码文件。 3. 使用项目管理工具共同开发一个项目时，要协商好每个人当前修改的内容，并及时上传，防止合并时产生冲突。 4. 项目数据迁移时需要充分测试，提前备份，保留完整的日志。 5. 提前定义好接口，分实现者和使用者，以便合作开发。 6. 在开始插件的学习之前，先仔细了解插件是否会与项目中需要的其它插件冲突，并尽早进行简答的兼容性测试。 7. 安卓端和PC 端持久化存储方式不同，读写文件请求设置不同，需要分开做。 8. 项目迁移unity自动导出所有脚本，但是为了简洁，需要手动勾选需要迁移的脚本，因此最好每个人分开迁移自己的部分。 |