HyperCraft

系统测试报告

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2023/06/17 | 1.0 | 初版 | 任柏俊、袁翊天 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 概述 4

2. 测试概要 4

3. 测试环境 4

4. 测试结果及分析 4

4.1 需求覆盖率及缺陷分布 4

4.2 缺陷严重程度 5

5. 缺陷清单 6

5.1 功能性缺陷 6

5.2 非功能性缺陷 6

6. 测试结论与建议 7

系统测试报告

# 简介

## 目的

本文档是对HyperCraft项目验收系统测试所做的说明，为充分利用已有的软硬件资源，配合对系统应用模块的运行测试方案，查缺补漏，完善系统的各项具体功能，保证项目的顺利完成，本测试报告有助于明确本次验收测试的测试资源、测试内容、测试方法、测试结果。

## 范围

本次系统测试涵盖HyperCraft项目开发四次迭代完成的所有内容，除了对HyperCraft的四次迭代进行系统且完整的软件评估外，也应当为项目后续的升级和维护提供参考和指导意见。

## 概述

该文档罗列出为整体评估HyperCraft所设计的一些测试用例，根据在具体环境下的实际测试结果，对软件当前的功能性、鲁棒性进行了分析，并总结出了现阶段存在的软件缺陷，最后给出了一些建议。

# 测试概要

测试时间：2023年6月17日-18日

测试地点：上海交通大学

测试人员：HyperCraft项目组全体成员

测试方法：黑盒测试、白盒测试、压力测试

测试内容：

* 1. 跨平台测试：各成员分别在个人PC上通过源码构建项目。
  2. 基本的功能测试：Client 程序构建完成后，尝试进入游戏世界。成功进入后，测试与游戏世界的交互是否符合预期逻辑。
  3. 性能测试：在游戏世界内模拟正常的玩家行为，实时监控和记录帧数，进行性能评估。
  4. 其他测试：通过超量渲染、频繁加载进行压力测试，通过长时间运行进行稳定性测试……

# 测试环境

操作系统：涵盖主流的Windows, MacOS, Linux 操作系统。

硬件环境：处理器 AMD Ryzen 7 5800H，内存 16.0 GB，配有独立显卡 RTX 3060。

# 测试结果及分析

## 需求覆盖率及缺陷分布

**表1 需求覆盖率及缺陷分布**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **测试内容** | | **测试**  **用例数** | **需求**  **覆盖率** | **缺陷数** | **缺陷率** | **备注** |
| **功能项** | **创建世界** | 5 | 100% | 0 | 0% | 使用数字、字符串、默认随机种子生成世界 |
| **进入世界** | 1 | 100% | 0 | 0% |  |
| **编辑世界** | 2 | 100% | 0 | 0% |  |
| **删除世界** | 2 | 100% | 0 | 0% |  |
| **放置方块** | 2 | 100% | 0 | 0% |  |
| **破坏方块** | 1 | 100% | 0 | 0% |  |
| **碰撞检测** | 2 | 100% | 1 | 50% | 是否出现穿墙 |
| **地形生成** | 1 | 100% | 0 | 0% |  |
| **地图保存** | 1 | 100% | 0 | 0% |  |
| **功能项小计** | 17 | 100% | 1 | 5.9% |  |
| **非功**  **能项** | **性能** | 1 | 100% | 0 | 0% | 1080p下平均fps能否达到500+ |
| **可靠性** | 1 | 100% | 0 | 0% |  |
| **压力测试** | 2 | 100% | 1 | 50% |  |
| **兼容性测试** | 1 | 100% | 0 | 0% |  |
| **部署测试** | 1 | 100% | 0 | 0% |  |
| **非功能项小计** | 6 | 100% | 1 | 16.7% |  |
|  | **总计** | 23 | 100% | 21 | 8.7% |  |

## 缺陷严重程度

**表2 缺陷严重程度**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **严重程度** | **致命** | **严重** | **一般** | **细微** | **总数** |
| **缺陷个数** |  |  | 1 | 1 | 2 |
| **占缺陷百分比** |  |  | 50% | 50% | 100% |

# 缺陷清单

## 功能性缺陷

**表3 功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **功能模块** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
| 1 | 1 | 细微 | 碰撞检测 | 穿墙 | 当移动速度大于帧率时，有概率出现穿墙，但正常情况不可能出现该速度 | hc-testcase-015 |

表中相关项说明：

1. 缺陷编号：为每个缺陷赋予一个唯一的编号，可以通过此编号对缺陷进行跟踪。例如：Bug001。
2. 缺陷严重程度：缺陷可以根据严重程度分为以下几种情况。
3. 致命（fatal）：致命的错误，测试执行直接导致系统死机、蓝屏、挂起、或是程序非法退出；系统的主要功能或需求没有实现。
4. 严重（critical）：严重错误，系统的次要功能点或需求点没有实现；数据丢失或损坏。执行软件主要功能的测试用例导致系统出错，程序无法正常继续执行；程序执行过于缓慢或是占用过大的系统资源。
5. 一般（major）：不太严重的错误，这样的缺陷虽然不影响系统的基本使用，但没有很好地实现功能，没有达到预期的效果。如次要功能丧失，界面错误，打印内容、格式错误，提示信息不太正确，或用户界面太差，简单的输入限制未放在前台进行控制，删除操作未给出提示，操作时间长等。
6. 细微（minor）：一些小问题，对功能几乎没有影响，产品及属性仍可使用。如软件的实际执行过程与需求有较小的差异；程序的提示信息描述容易使用户产生混淆；有个别错别字、 文字排列不整齐；辅助说明描述不清楚，显示格式不规范，长时间操作未给用户进度提示，提示窗口文字未采用行业术语，可输入区域和只读区域没有明显的区分标志，系统处理未优化等。
7. 功能模块：所测试并出现该缺陷的功能模块名称。
8. 缺陷标题：描述缺陷的标题。
9. 缺陷描述：对缺陷的详细描述。描述缺陷的重现步骤和问题所在，描述需清晰明了，使研发人员可以按照描述将缺陷重现出来。必要时可以放上相应的截图，以便于研发人员理解问题所在。

## 非功能性缺陷

**表4 非功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **测试类型** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
| 1 | 2 | 一般 | 压力测试 | Mesh Pool溢出 | 需要渲染的Mesh数据量过大超出预设的数据量限制导致部分区块无法显示 | hc-testcase-020 |

# 测试结论与建议

HyperCraft软件的编码质量较高，测试出的缺陷较少。

缺陷1，即穿墙问题可以通过编码限制玩家的移动速度小于帧数来解决。

缺陷2，即Mesh Pool溢出问题可以通过维护动态数量的Mesh Pool来解决，但这样会增加渲染架构的复杂性和程序的总体性能，还可能导致显存溢出，产生更严重的问题。同时正常情况（即方块为玩家放置）下不可能出现Mesh Pool溢出。因此维持原有设计，限制Mesh Pool数量。