南开大学 软件工程

姓名 袁正泽

学号 2312277

班级 2023级软件工程2班

2025年5月11日

高级语言程序设计

实验报告

目录

[高级语言程序设计大作业实验报告 2](#_Toc197893539)

[**一.** **作业题目** 2](#_Toc197893540)

[**二.** **开发软件** 2](#_Toc197893541)

[**三.** **课题要求** 2](#_Toc197893542)

[**四.** **主要流程** 2](#_Toc197893543)

[**1． 整体设计流程** 2](#_Toc197893544)

[**2． 核心模块结构图** 3](#_Toc197893545)

[**3． 算法逻辑与动画控制** 3](#_Toc197893546)

[**（1） 跳跃模拟（QPropertyAnimation）** 3](#_Toc197893547)

[**（2） 跑酷游戏逻辑** 3](#_Toc197893548)

[**五.** **单元测试** 4](#_Toc197893549)

[**六.** **收获** 4](#_Toc197893552)

[**七.** **对AI辅助编程工具Cursor、Trae的探索和反思** 4](#_Toc197893554)

[**八.** **致谢** 6](#_Toc197893559)

高级语言程序设计大作业实验报告

1. **作业题目**

-meme-：可爱的猫Meme桌面宠物

1. **开发软件**

Vs Code、QT Creater、Trae、Cursor

1. **课题要求**
2. 面向对象
3. 单元测试
4. 模型部分
5. 验证
6. **主要流程**
   1. **整体设计流程**

本项目设计了一个具备基础动画行为、交互事件响应与小游戏模块的桌面宠物系统。

**主要实现内容：**

1.使用 Qt 编写 GUI 程序，主窗口即为“宠物实体”

2.宠物表现形式为动态 gif 图像，通过 QMovie 播放

3.鼠标点击响应动作（如弹跳、喵叫、吐槽弹幕）

4.右键菜单支持切换表情、召唤新宠物、退出程序、召唤猫猫军团等

5.多宠物共存：每个宠物是独立窗口对象，可独立控制

6.集成小游戏（如“跑酷躲狗头”），提供用户互动娱乐

**采用面向对象设计思路：**

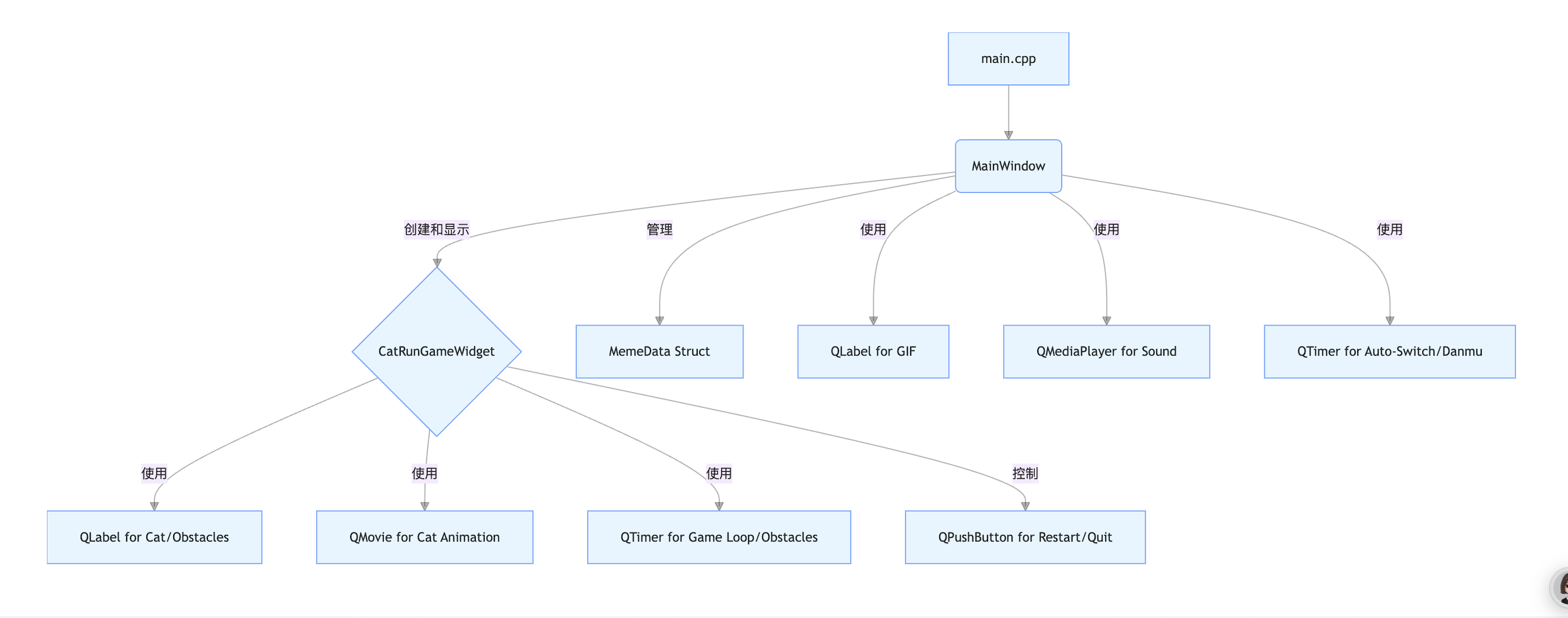
1.主窗口类 MainWindow 封装宠物行为

2.游戏窗口类 GameWindow 负责小游戏逻辑与碰撞判定

3.使用信号槽连接用户输入与宠物响应

4.所有资源（图像、音效）统一注册至 resources.qrc 文件

* 1. **核心模块结构图**



* 1. **算法逻辑与动画控制**
     1. **跳跃模拟（QPropertyAnimation）**

1.使用Y轴属性构造一个上下运动轨迹，模拟猫猫跳跃

2.曲线使用QEasingCurve::OutQuad实现自然跳跃

* + 1. **跑酷游戏逻辑**

1.角色：固定左下角 gif 图

2.障碍：QLabel 动态创建，定时从右向左移动

3.判定：使用geometry().intersects(...)判断是否碰撞

4.得分：成功跳过障碍则加分，否则弹窗“Game Over”

1. **单元测试**

虽然 Qt 项目不适合传统 console 单元测试，但我们采用手动行为验证 + 自动逻辑检测结合方式：

|  |  |  |  |
| --- | --- | --- | --- |
| 测试功能 | 预期行为 | 实际表现 | 是否通过 |
| 表情切换 | 切换后动画正常播放 | 正常 | 是 |
| 播放音效 | 点击播放喵叫声或弹幕吐槽 | 正常 | 是 |
| 多宠物共存 | 可重复右键“召唤新宠物”或点击“召唤猫咪军团” | 正常 | 是 |
| 跑酷游戏角色响应 | 鼠标点击或点击空格键或“↑”键角色起跳，“A”“D”键或“←”“→”角色左右移动 | 正常 | 是 |
| 障碍碰撞判定 | 碰撞后游戏结束弹窗 | 正常 | 是 |
| 加分系统 | 成功跳过障碍加1分 | 正常 | 是 |

### 测试结果

测试全部通过。

1. **收获**

借助完成本次大作业的契机，我理解了 Qt 中事件处理、窗口管理、动画机制、资源打包等完整流程，熟悉了面向对象在 GUI 应用中的实际应用方式，体验了从零构建一个完整桌面交互系统的设计过程，掌握了 Git 的版本控制、推送流程及大文件管理方法，意识到良好的模块划分和可测试性对于项目维护的重要性。

1. **对AI辅助编程工具Cursor、Trae的探索和反思**

在本次 C++ Qt 桌面宠物项目的开发过程中，我使用了 AI 编程辅助代码编辑器 Cursor和Trae（分别由Anysphere和字节跳动开发），它们基于大语言模型，为我在编码、调试、重构等多个环节提供了实质性的帮助。我认为在当今AI agent快速发展的今天，使用AI辅助完成作业并不是一件可耻的事情，只有更好地拥抱时代浪潮，发挥AI的辅助作用，我们才能更好地跟上时代。于是，在此我希望基于我的使用体验与大家讨论一下我对当今时代最先进的AI IDE的探索和反思，虚妄对诸君有所帮助。

这两个AI 编程辅助代码编辑器能够自动生成类定义、构造函数、getter/setter 结构，以及标准事件函数等模板代码，极大减轻了我在编写样板代码（boilerplate）上的负担，让我可以更专注于业务逻辑本身。在语法拼写、变量命名、类型转换等容易出错的细节上，ai编程工具能够即时给出提示并修正建议，这在 Qt 项目中尤其有价值，比如信号与槽的连接、资源文件路径错误等，都能第一时间识别。通过它集成的“Builder 模式”交互，我可以用自然语言描述我想要的功能（例如“实现一个跳跃动画，持续 0.6 秒，模拟重力加速度”），Cursor 和Trae会生成代码框架，再由我根据项目上下文进行完善。它在“不会写”与“知道方向但写不出来”的中间地带，发挥了极大作用。当我面对 Qt 的一些晦涩编译器错误信息或链接错误时，ai工具帮我快速翻译错误提示，解释含义，并给出可能的解决方案，这比单纯在 Stack Overflow 上搜索更直接、更上下文相关，节省了大量调试时间。

同时，作为ai代码编辑工具的受益者和实际使用者，我想就我的体验而言指出现在ai辅助编程的短板和瓶颈所在。

* + 1. **纠错能力有限**

AI 在自动生成代码时容易“理直气壮地出错”。如果初始理解偏差，后续修改往往是在错误基础上反复自我修复，导致“以错纠错”的循环——这会加重调试负担，甚至把原本能手动快速修好的 bug 变得复杂化。

* + 1. **易引入冗余和垃圾代码**

AI 自动补全有时会过度“想象需求”，插入无用的变量声明、多余的判断结构或逻辑分支，导致代码变得冗长、难读、不易维护，有时还会拖慢程序运行速度。

* + 1. **对复杂逻辑掌控力差**

面对较复杂或高度抽象的逻辑结构（如异步状态管理、复杂对象图、跨线程信号调度等），AI 往往表现为“幻觉”严重，会编造不存在的类方法、误解控制流程，甚至给出完全逻辑错误的方案。

* + 1. **删除与重构操作风险大**

AI 在自动删除/重构代码时，可能误删关键组件或遗漏引用关系，导致程序报错却难以定位。有时 AI 的建议还会在多个文件中同时生成未用到的辅助函数，破坏代码整洁性。

AI 编程工具确实为开发带来了前所未有的便利和可能性，尤其对代码生成、思路辅助、调试定位等方面效果显著。但这并不意味着可以完全依赖 AI 来驱动开发。使用 AI 工具的前提，仍然是开发者自身对编程语言、框架结构与项目逻辑具备充分掌控力。AI 工具应该成为程序员的得力助手，而不是我们写代码的主导者。作为开发者，我们要成为 AI 的主人，而不是被动的执行器或“AI 的打字机”。

1. **致谢**

感谢《高级语言程序设计》课程的项目支持，感谢殷爱茹老师一学期的辛勤付出，也感谢本次实验过程中 VSCode、Cursor、Trae 等开发工具带来的辅助体验，有了这段宝贵的经历和老师的鼓励，我定会在程序员的路上走的更加舒畅从容。