元气骑士组-项目总报告

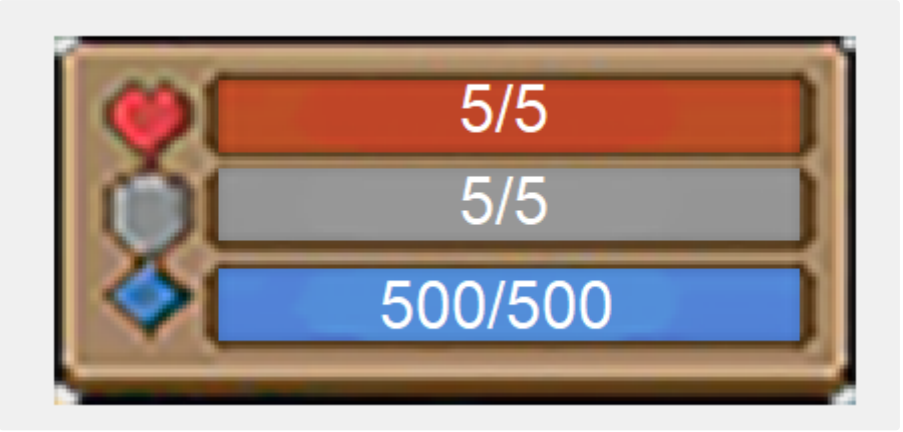
项目成果介绍

本次课程中我们通过对于MVVM框架的了解和深度学习，掌握了C++工程项目实践的相关方法和技巧，并通过开发一款游戏来验证我们的学习成果。我们选择参考市面上已存在的2D平面动作射击游戏《元气骑士》作为参照，开发了一款功能较为简单的类似游戏，并实现了部分游戏功能，具体如下

- 实现人物的自由移动，动作形象改变，射击子弹行为，碰撞检测
- 实现UI的展现，包括人物血量，蓝量，盾量的改变
- 实现地图的生成与展示，包括不可通过的“墙壁”方块，可以移动的“背景”方块，以及可以通过子弹击碎的“箱子”方块
- 实现怪物模型的展现，包括怪物的随机移动逻辑，主动发射子弹，碰撞人物等功能

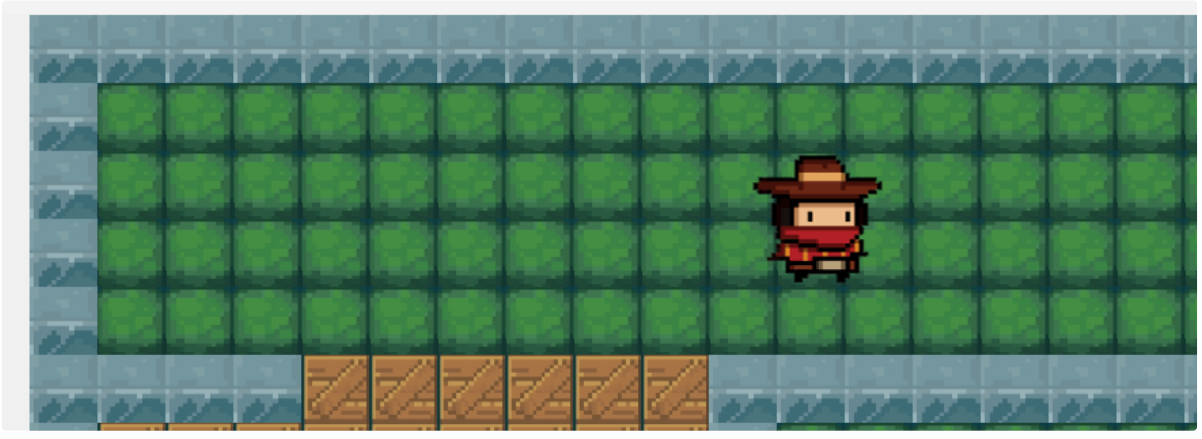
项目图片效果展示

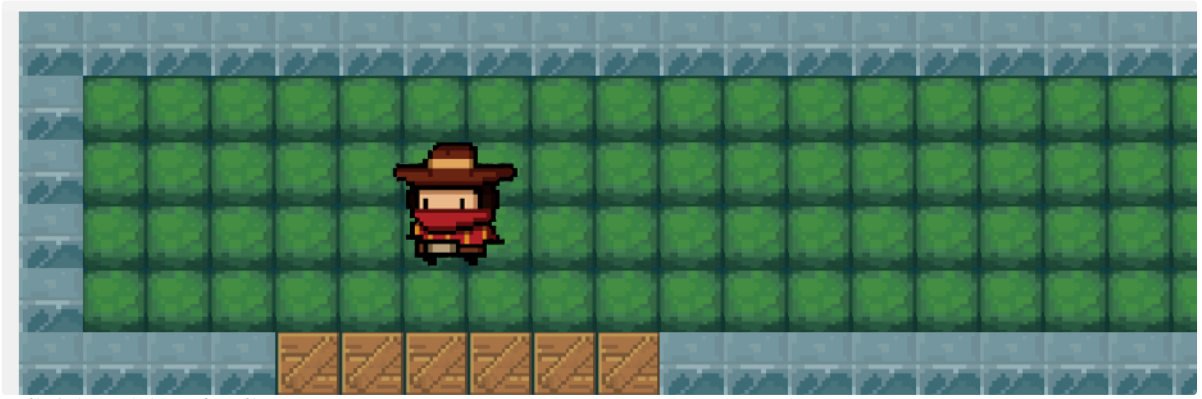
人物UI展示



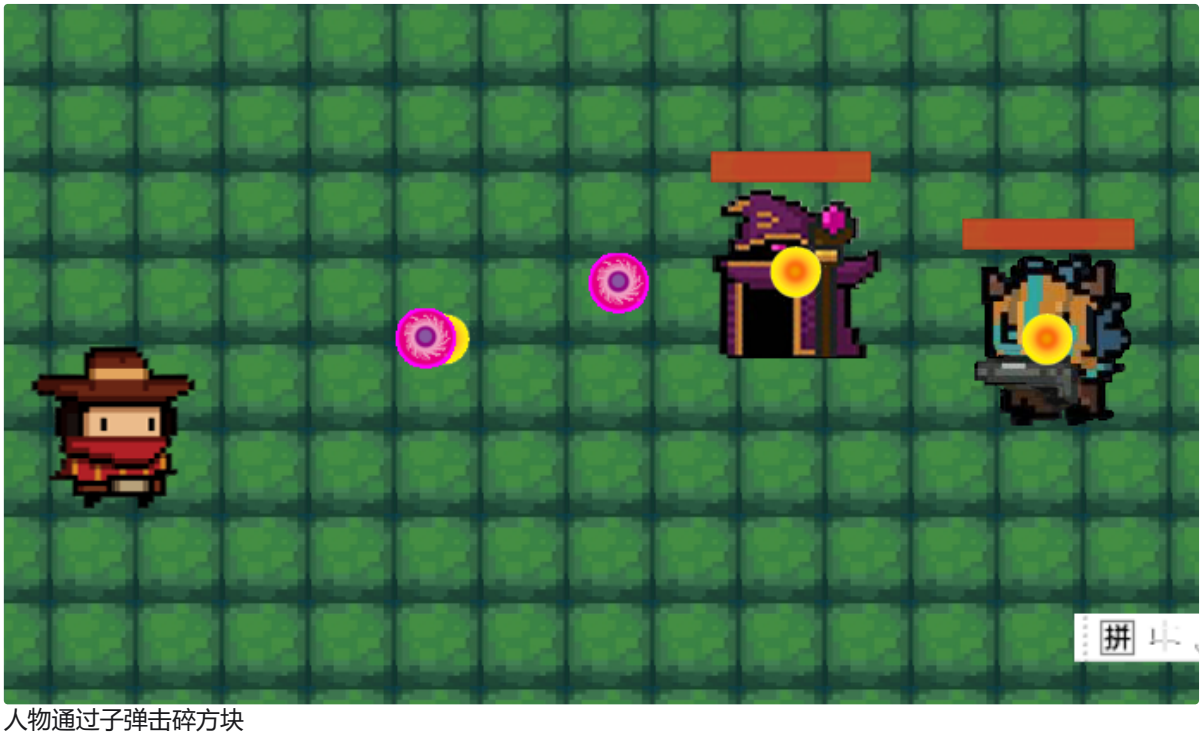
人物相关动作

人物左右移动

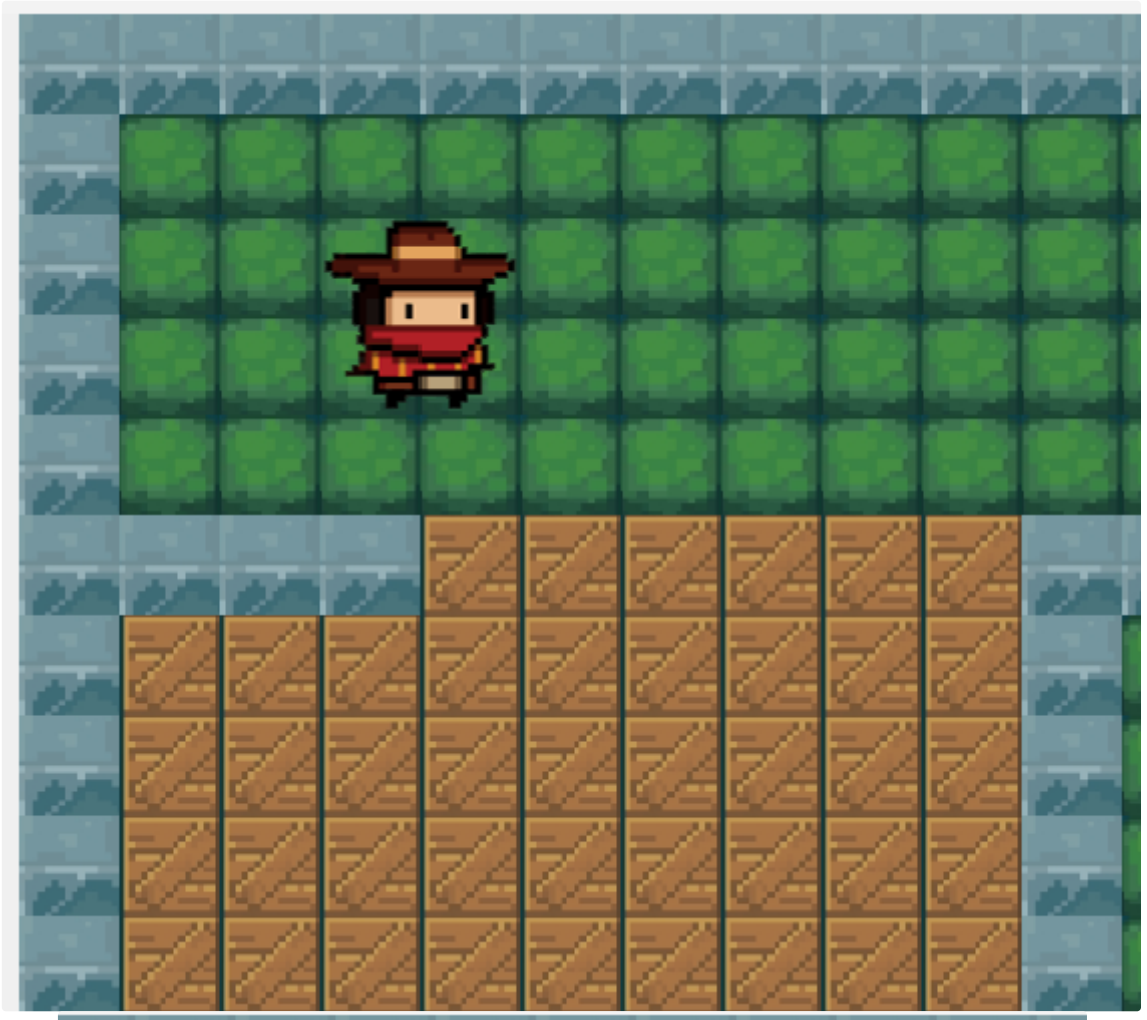


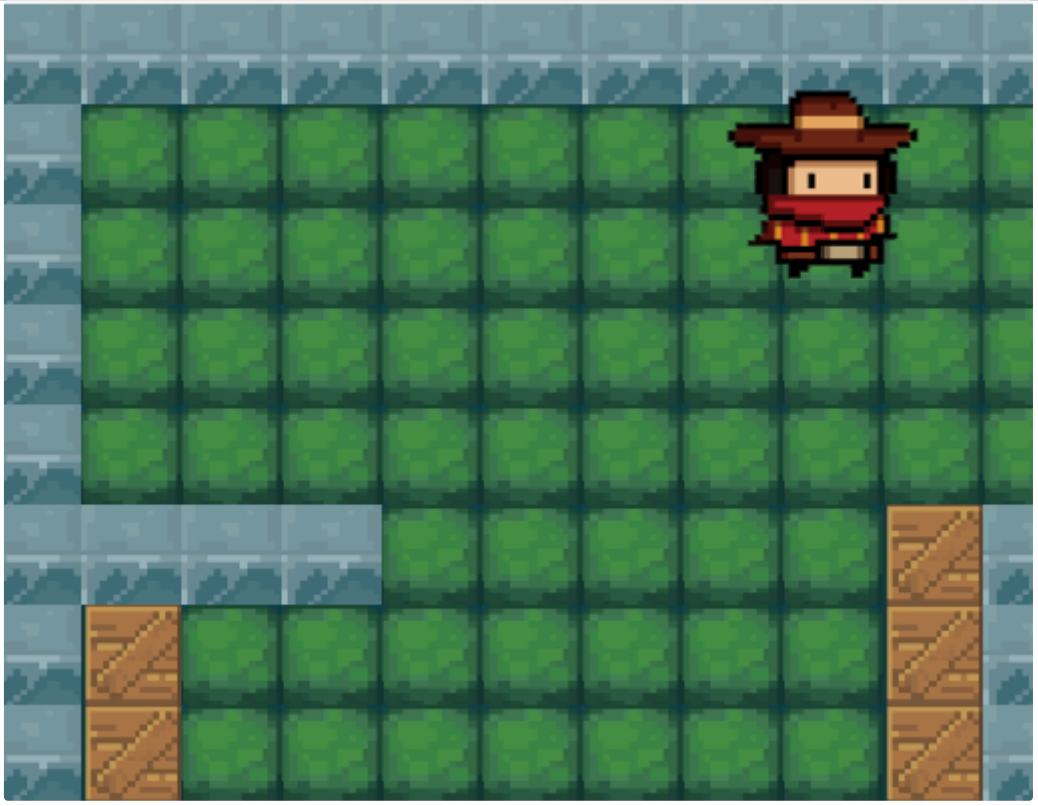


人物发射子弹并击中怪物



人物通过子弹击碎方块





怪物相关显示

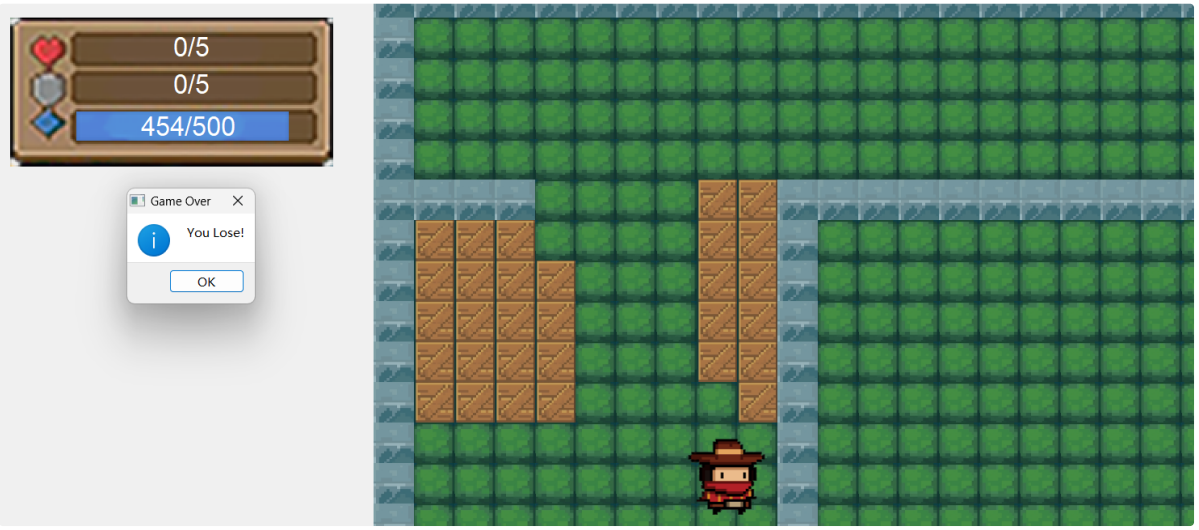
怪物种类1

怪物种类2

怪物种类3

怪物发射子弹并击中人物

人物血量归零时失败界面



游戏获胜界面

项目开发工具展示

git版本控制

improve monster attack and move
ken0yuan committed 3 hours ago — Verified — 83aa16a

improve monster attack and move
ken0yuan committed 4 hours ago — Verified — 6ff55f1

三轮迭代初步完成
Your Name committed 5 hours ago — 3c55787

Merge branch 'master' of https://github.com/ken0yuan/yuanqi_qishi
Your Name committed 8 hours ago — 0ab2e0e

enemy_move.1
Your Name committed 8 hours ago — 5dbfd78

add monster move and shot in model and viewmodel
ken0yuan committed 8 hours ago — Verified — d74a317

add MP consume
ken0yuan committed 12 hours ago — Verified — d36598d

Role UI complete
ILHZZZ committed yesterday — 6a382fe

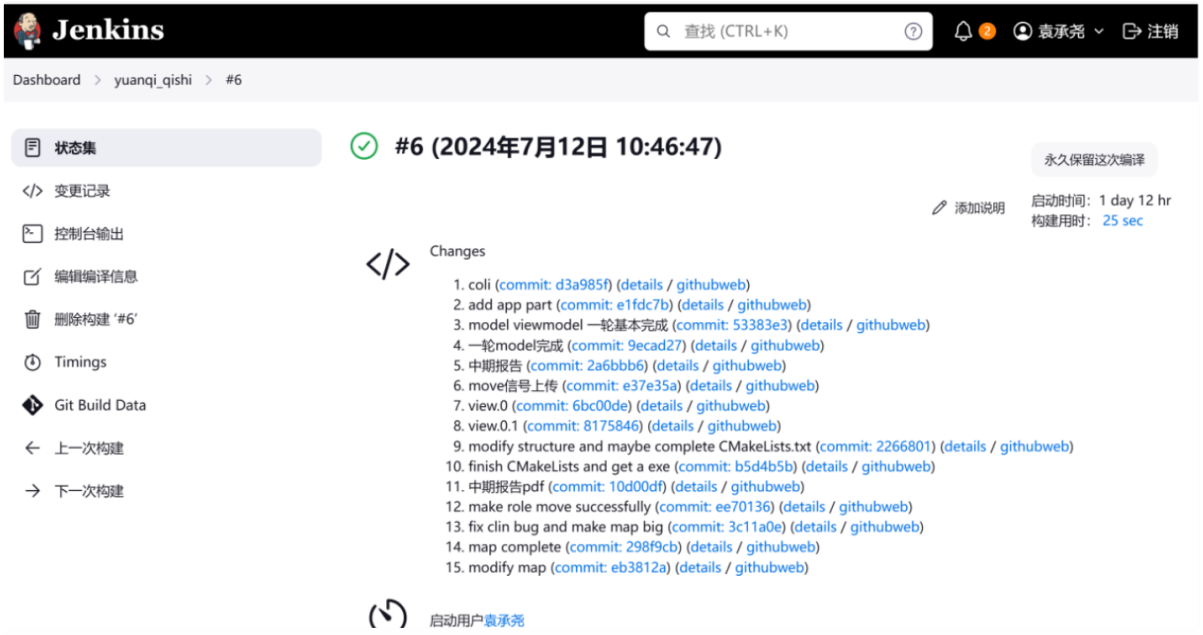
Commits on Jul 12, 2024

modify map and clion to make it great
ken0yuan committed yesterday — Verified — 1105e1e

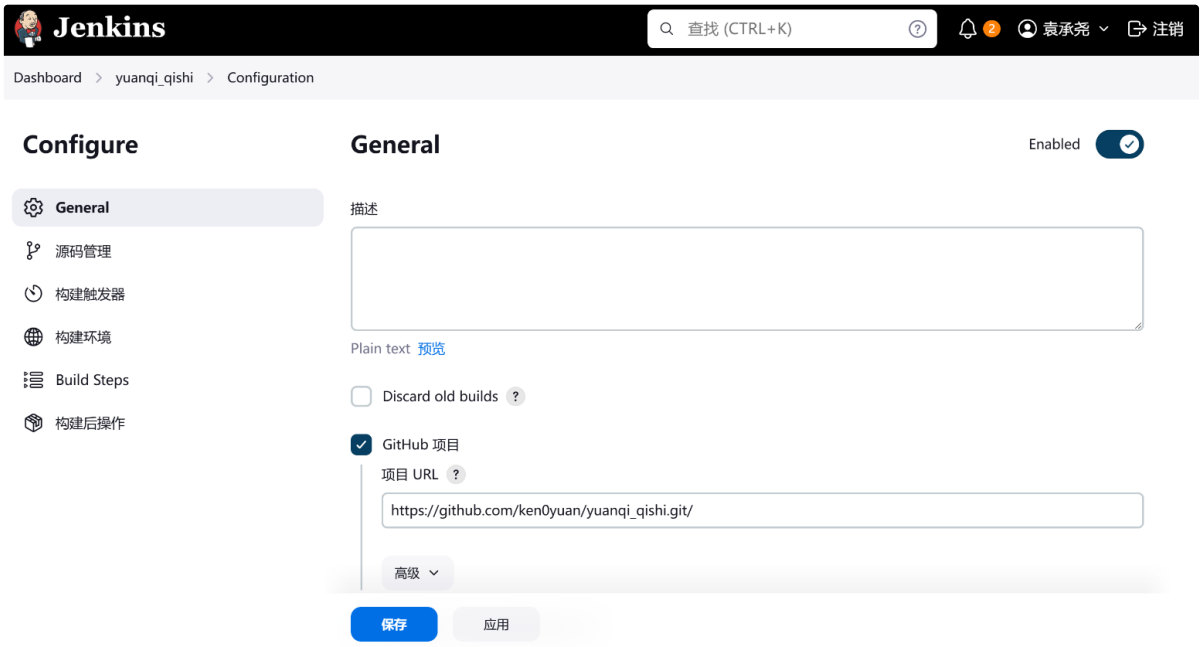
finish 2nd iteration, finish bullet shot
ken0yuan committed yesterday — Verified — 8481a90

- github上的commit相关commit记录
- 展示每一轮迭代和相关完成的工作

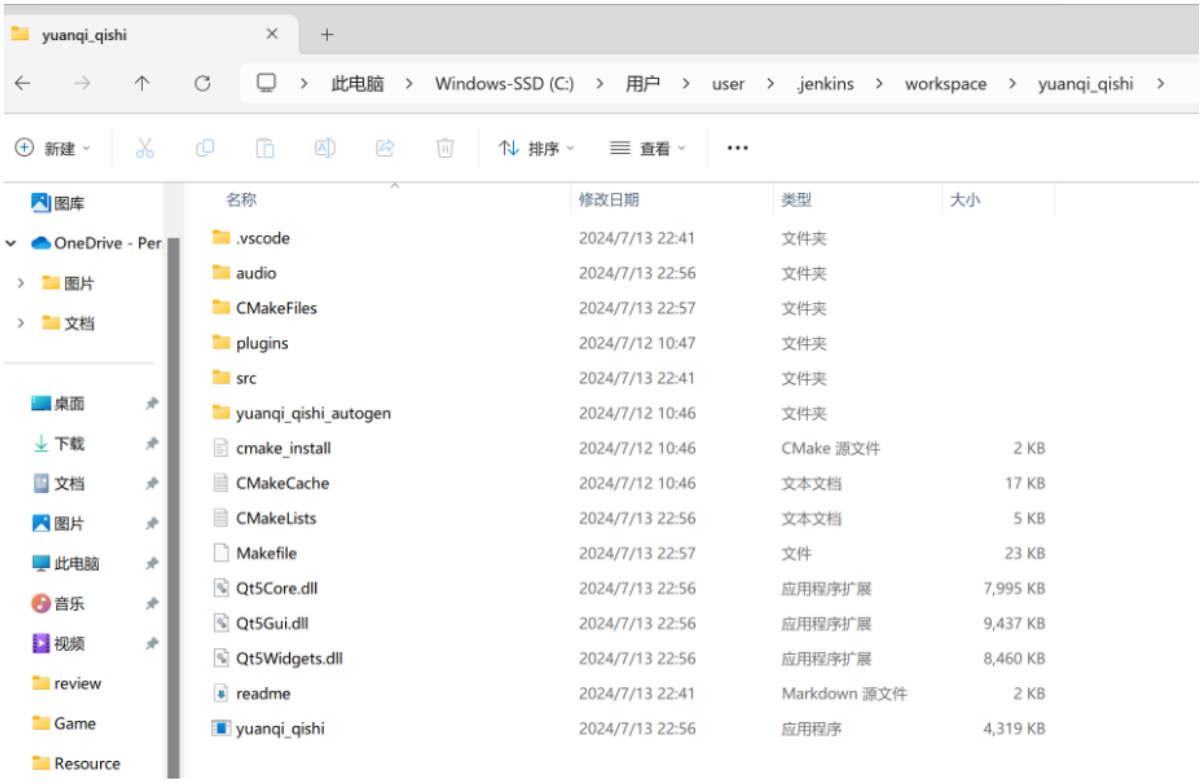
Jenkins



- Jenkins检测文件变化和自动构建



- Jenkins配置界面



- Jenkins自动编译后构建的文件展示（非本地编译）

QT+Vscode开发

- 由于单独使用QT creator进行开发进行文件管理较难，所以尝试在Vscode上部署QT项目

CMakeLists.txt images.qrc

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.5)
# 项目名称

# 使用C++ 11标准进行编译
set(CMAKE_PREFIX_PATH "D:/QT/5.12.12/mingw73_64")

set(CMAKE_CXX_STANDARD 17)
# 要求代码必须遵守C++11标准
set(CMAKE_CXX_STANDARD_REQUIRED ON)

#支持自定义信号，为Qt中的QObject子类自动生成moc文件*
set(CMAKE_AUTOMOC ON)
# 为Qt资源文件(.qrc)自动生成源文件
set(CMAKE_AUTORCC ON)
#为Qt设计器中生成的UI文件(.ui)自动生成头文件和源文件
set(CMAKE_AUTOUIC ON)
set(CMAKE_C_COMPILER gcc)
set(CMAKE_CXX_COMPILER g++)

enable_language(CXX)
#乱码问题处理
project(yuanqi_qishi)
add_compile_options("$<$<CXX_COMPILER_ID:MSVC>:/utf-8>")
# 获取.h文件
file(GLOB headers
${CMAKE_CURRENT_LIST_DIR}/../src/app/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/common/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/model/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/view/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/viewmodel/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/view/sink/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/viewmodel/sink/*.h
${CMAKE_CURRENT_LIST_DIR}/../src/viewmodel/commands/*.h
)
```

- 由于Qt creator在创建项目和增加文件时会自动生成CmakeLists，但是更改项目架构后就无法继续使用，考虑自行编写和修改CmakeLists
- 随后正常使用QT库的相关函数和组件进行编写程序即可

个人心得体会

- 袁承尧：这次实验一轮迭代我负责了common和app层，同时开始做jenkins和CMake相关的任务，这部分相当复杂，我们采用Qt库，其中的库文件d3d12在编译的时候出现了找不到的问题，非常的麻烦，在老师的帮助下，我们采用Qt5进行尝试，重新撰写CMakeLists的过程让我更加了解了这个工具的用处。二轮以及后续的三轮四轮迭代中我负责model和viewmodel的部分，这部分非常的内容多样，我们每新增一个功能就需要写大量的配套函数，我经常漏一两个。在一次次的debug过程中，我更加了解了框架的运转原理，并且愈发觉得这个只要写好了common以后的低耦合框架的优越性。总而言之，这次短学期课程是一个不错的大程体验，我们从中学到了很多。
- 周俊：在开始这门课之前，我对c++还不算掌握很好，尤其是听说这门课要使用到从没接触过的Qt库和MVVM框架，因此我还不太有信心能够很好地完成这门课的任务。在课程开始的前几天，与预料一致，我在理解MVVM框架上遇到了许多问题，对这个框架没有什么理解，因此在那几天里，我基本上都在搜索资料以及学习前人使用该框架的经验，来增加自己对这个框架的理解。因此在前面一周多，我们都进展很慢。但在逐渐尝试的过程中，查询资料与实践相结合，我对该框架有了初步的理解。MVVM框架由三个部分组成，即Model，ViewModel和View。Model负责存储应用程序的数据，ViewModel负责将Model中的数据与View进行绑定，以便在数据变化时更新视图。当数据源发生变化时，Model需要能够通知ViewModel和View，以便及时更新用户界面。ViewModel处理用户界面的命令，例如按键点击等操作，将其转换为对Model的相应操作。因此，这两个模块基本完成了程序的底层逻辑，维护了应用程序的数据模型。而View模块则主要是处理图像的显示以及与外界的交互，将model中的数据用图像显示出来，再接受指令，传递给ViewModel以执行对应的操作。在我们完成了第一个功能即人物的移动之后，我们对这个框架的理解就已经加深了许多。由于该框架极低的耦合性，在完成了common之后，需要添加新功能时每个人只需要在自己的部分添加相应的内容即可，操作相当简单，非常方便，也非常便利团队协作。因此，在实现后续的功能时，我们的效率高了好几倍，迅速完成了预想的内容。通过这门课的实践，我对c++的理解更加深刻，也熟悉了Qt库结合MVVM框架的开发模式，为以后积攒了许多有用的经验。在最后看着自己完成的项目，我们也是非常有成就感，竟然在短短两周时间内就完成了这样一个规模不小的项目。对于这门课程，我提出一个建议：希望在以后的课程中，在前几天理论讲解时，多结合项目实例进行讲解，以帮助同学们更好地理解这个框架的使用。
- 郑伟廷：本次课程最大的心得和难点就是在于对于MVVM框架的理解以及将该框架具体应用到对应的C++工程项目当中。在前期的理论课讲解过程当中，我只对于MVVM框架有了一个大致的印象，也就是大致了解common,model和viewmodel,view三层之间的简单关系，也就是common提供数据类型，model提供数据修改，viewmodel管理命令，将model的数据和view进行绑定，view只要负责渲染画面即可。但是在实际的操作过程中，初期开发的时候完全无法体会到MVVM框架的“低耦合性”的体现，常常是model修改后viewmodel发现没有传一些参数，view里面又发现没有绑定数据等。对于细节方面在理论课上没有得到深刻的理解，例如在第一轮迭代的时候model里面没有传递消息给view，view也就无法渲染，种种细节的理解和实现的差异等给我们带来了很大的困难。最后在开发过程中逐渐理解了“低耦合性”的关键所在，只需要和队友说好command

的相关实现，以及如何让接收器接受的信号，三个层次就可以互不干扰的进行独立开发，大大增加了开发的效率。其次是C++，Qt和Cmake相关环境的搭建，这一部分也折磨了相当长的一段时间，C++环境的配置过程也一直是很大的问题。最后看到两周之内实现了这样一个项目，还是感到非常的开心的。

课程建议

- 对于前期理论课的讲解过程中，希望根据框架配套更多的例程以及讲解，以及在代码中配套一些对应的注释帮助理解。