


补充内容与细节

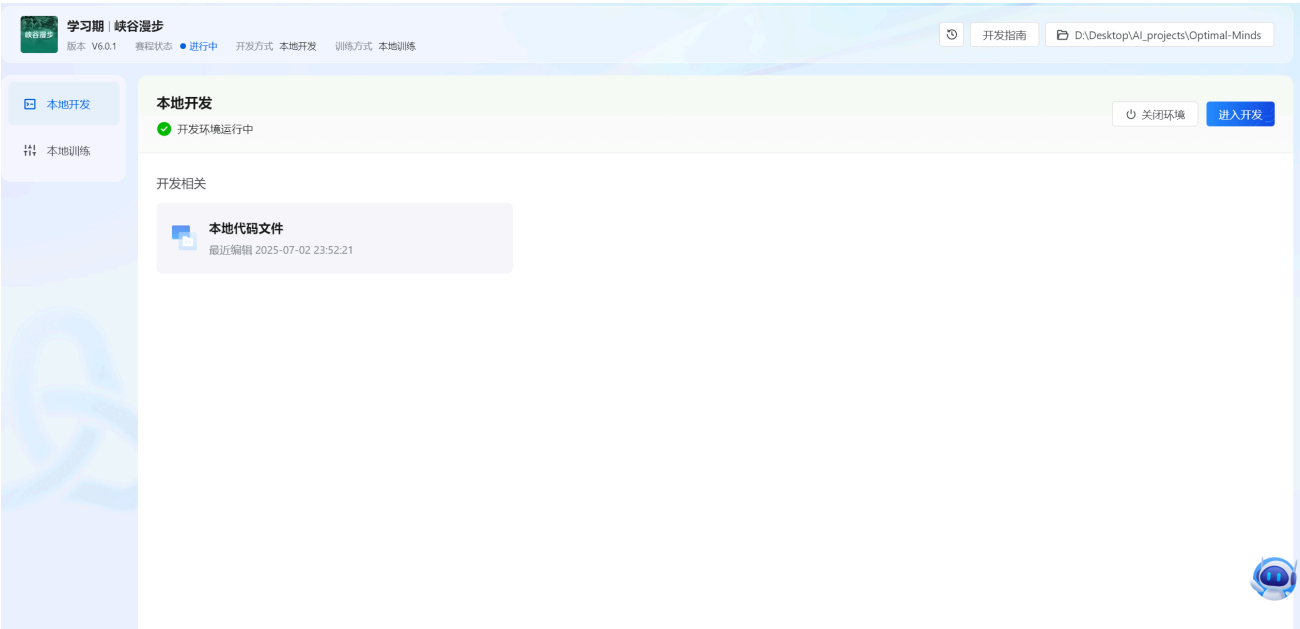
 创建时间：2025-07-03

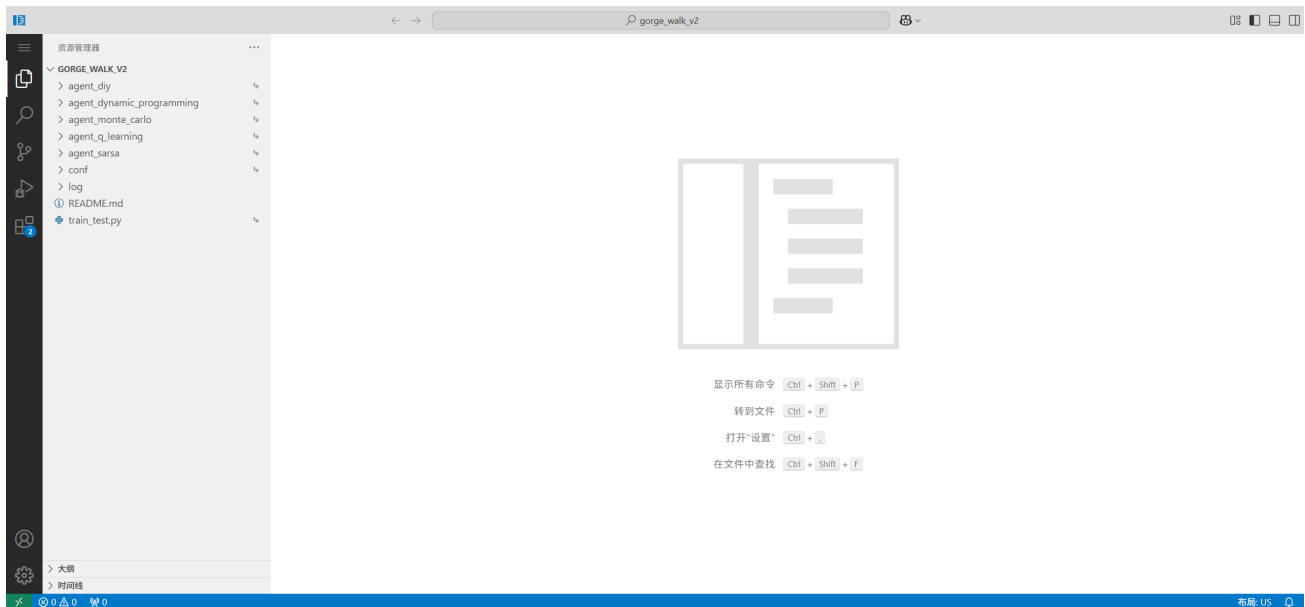
1. 具体开发

注意**使用手册**讲的内容是如何下载客户端并且配置相关环境的，**开发指南**是关于代码框架与接口的详细描述。在配置环境时看**使用手册**，研究代码时看**开发指南**，下面有一些常见的问题：

- 工作路径的设置：按照指引，需要是一个空目录，同时注意不要放在C盘的根目录即可。
- 完成环境配置后，点击进入开发即可进入开发环境，客户端会自动launch一个网页版的Vscode进行开发(暂时无法使用本地的Vscode)，在该网页版上进行开发即可
- 完成开发后，将保存好的内容上传到github上即可共享

参考开发页面如下：





2. github使用

多尝试使用github，包括克隆仓库，拉取最新的内容，以及推送内容，具体操作可见上一笔记

- 注意推送修改的内容时不要直接推送到main分支上，可以推送到自己创建的分支上，防止修改当前最佳的版本
- 推送的内容可以在后台看到，这样到时我们比对内容，可以决定是否更新当前最佳的版本
- 我们的仓库是private仓库，如果你们访问不到的可能要我添加成员

项目初步认知

代码目录：

- code/
 - agent_dynamic_programming/: 动态规划算法实现
 - agent_monte_carlo/: 蒙特卡洛控制算法实现
 - agent_q_learning/: Q-learning 算法实现
 - agent_sarsa/: SARSA 算法实现
 - agent_diy/: 留给参赛者自定义实现的模块，许多函数仍为空
 - conf/: 训练与运行相关的配置文件，如 `configure_app.toml` 指定算法、日志目录等；`algo_conf_gorge_walk_v2.toml` 列出了不同算法所对应的 agent 和 workflow 定义
 - train_test.py: 用于启动训练流程的脚本，可指定使用哪种算法
 - kaiwu.json: 框架相关的元数据文件

主要模块内部：

- agent.py: 定义了 Q-learning 智能体，包括 epsilon-greedy 动作选择及模型保存/加载等
- algorithm/algorithm.py: 核心学习逻辑，对 Q 表按公式更
- workflow/train_workflow.py: 读取配置、与环境交互并不断训练

- `feature/definition.py`: 定义样本结构和奖励计算方式