Qt大作业作业报告

徐海翁

2024.6.28

目录

1 综述

- 我们的项目的目标是实现一个能够满足北大学生尤其是新生个性化需求的日程管理应用，希望借此帮助用户提高效率，关键是尝试走出舒适区，尝试未尝试过的事情，活出多彩的人生。
- 与通常的日程管理软件不同，我们的项目的突出特点在其个性化的设计，希望项目能够帮助有缺乏规划意识的同学提高效率，同时能够降低走出舒适区所需要的各类成本

2 程序功能介绍

2.1 菜单

- 在菜单界面点击启动按钮可以进入主界面

2.2 主界面

2.2.1 导入课表功能

- 点击 `导入课表` 按钮，可以从文件资源管理器中打开从选课网或信息门户上下载的 `xls/xlsx` 格式课表，作为之后日程自动推荐的基础。
- 导入的课表会被自动保存到本地，用户重新打开应用时无需重新导入课表。

2.2.2 日程自动推荐功能

- 双击日历中未被点击且本地没有相应数据的一天时，程序将会根据用户的 `个性化设置` 生成一个当天的基本日程规划，并且展示在日程表中，基本日程包含就餐和自习场所的信息。
- 当双击的一天已被规划或本地已经有数据时，右侧的日程表将会自动回到那天已有的日程规划，用户可以点击 `重新安排` 按钮对当天进行的日程进行重新安排。

2.2.3 个性化事件推荐功能

- 点击 个性化推荐 按钮，程序将会自动生成3个推荐事件，并展示在弹出窗口中，用户可以点击其中的任意事件进行添加或 重开 按钮进行重新生成。

2.2.4 日程本地保存功能

- 选定日历中的某一天时，我们可以点击 保存到本地 按钮，将当天的日程记录到本地，关闭程序重新打开后再次双击该天可以将日程切换为当天的日程。

2.2.5 日程表交互

- 右键日程表任意一个位置会弹出包含 删除 ， 添加 ， 取消 ， 修改 等四个按钮：点击删除会删除当前行;点击添加会弹出一个设置事件信息的窗口，设置完成后可以添加事件，如果添加的事件存在时间冲突会进行 弹窗警告 ;点击修改则可以按照当前单元格的类别对当前单元格内容进行修改。

- 日程表下方有一个 日程倒计时器 ，将会显示当前到日程表中的下一个日程的剩余时间，为了提醒用户及时通勤，将会在倒计时结束10分钟前响铃并弹窗提醒用户。

2.2.6 讲座信息获取功能

- 选定日历中的一天时，点击讲座信息将会利用 爬虫 从北京大学讲座网中获取当天的讲座信息，并显示出来，帮助用户获取讲座信息，拓宽自己的眼界。

2.2.7 手动倒计时功能

- 为了在内卷模式下提高用户的内卷效率，用户可以操作主界面左下方的倒计时器，手动设计一个倒计时，到时提醒用户。

2.3 个性化设置功能

- 点击 个性化设置 按钮将会弹出一个个性化设置界面，主要的各项功能如下：

2.3.1 模式设置

- 用户可以选择合适的推荐模式：

1. 探索模式 (默认模式)：在这个模式下，在日程自动推荐功能中，我们会采用智能的方式为用户规划自习和就餐的时间和地点，当我们勾选记录并使用日程记录改进推荐选项时，程序将会优先推荐用户前往那些前往次数较少的就餐和自习场所，当不勾选该选项时，程序将会随机进行推荐。

2. 内卷模式 ：在这个模式下，在日程自动推荐功能中，程序会采用最短路径的原则为用户推荐自习和就餐地点，并且会将上午，下午，晚上没有课程的时间全部排满，提高用户的内卷效率。

3. 摆烂模式 ：在这个模式下，用户可以获得酣畅淋漓的摆烂体验。

2.3.2 偏好设置

- 用户可以勾选或者取消日程自动推荐功能中是否自动推荐 早餐 ， 中餐 ， 晚餐 的选项。

2.3.3 杂项

1. 用户可以勾选 明确的答复 选项，关闭用户添加事件发生时间冲突时的弹窗提醒。
2. 用户可以修改 宿舍位置 ，作为我们地图中路径的起始点和终止点。
3. 用户可以点击倒计时设置中的 修改图片 按钮，修改倒计时弹窗的图片。
4. 用户可以点击 清空历史记录 按钮，清空本地的课表，日程和访问记录等数据。

2.4 地图

- 点击主界面上方菜单栏中的 地图 项可以切换主界面至地图界面。
- 在地图界面中可以使用界面中的按钮 上一步 ， 下一步 ， 显示全部 ， 隐藏全部 在地图上分步或一次性显示路径

2.5 说明

- 点击主界面上方菜单栏中的说明项会自动在浏览器中打开本项目的 Github 仓库，在这里用户可以查看 README 文档获取应用使用说明。

3 项目各模块与类设计细节

3.1 主界面模块

- 主界面模块主要包含 日历 ， 日程表 (包含倒计时)组成，由于彼此相互关联程度高，因此没有对它们做进一步的细分，我们可以分别介绍其相关设计细节。

3.1.1 日历

当我们点击日历上任意一天时，会触发 oneday 函数，函数将会考虑当天是否存在本地数据以及当天是否被重新生成两个因素，依次做如下判断：

1. 程序首先会试图在本次运行中的缓存哈希表 Memo 中寻找当天的有关日程,如果有,就将当天日程复制到当天的 activities 中
2. 如果本次运行中的缓存哈希表 Memo 中找不到当天有关日程,程序会尝试从本地的 json 文件中寻找是否有当天的日程,如果有日程,就通过 Info 格式将当天的日程读入到 activities 中（ Info 格式见3.7.2）
3. 如果本地也不存在相关日程,则会调用下面的日程自动推荐功能3.1.4

3.1.2 日程表

1. 为了日程表高阶交互操作的构建方便，我们构建了一些基础接口例如 AddRow ， AddCol ， AddEvent ， DeleteEvent ，在之后更为复杂的交互部分实现更加复杂的抽象。

2. 活动管理功能主要包括以下几个方面：

   当我们右键日程表时，槽函数 onItemContextMenuRequested 会被触发，并提供删除、添加、取消、修改四种选项

   - 插入活动：程序设计了槽函数 onActionaddTriggered ，槽函数使用 AddHelper 函数作为通用的插入事件模板，在后续的个性化模块中也复用了 AddHelper 函数。用户可以在日程中插入新的活动，包括活动名称、时间、地点等信息。
   - 修改活动：程序设计了槽函数 onActionreviseTriggered ，用户可以修改已存在的活动信息。
   - 删除活动：程序设计了槽函数 onActiondeleteTriggered ，用户可以删除已有的活动。

3. 冲突检测与重新排序：每次试图插入活动后，系统会利用 Whetherclash 函数判定是否存在时间冲突，并在发现冲突时进行 提示 ，让用户自己选择是否接受该冲突。此外，程序在每次对活动进行调整后会对日程进行重新排序，确保所有活动按照时间顺序排列。

4. 到时提醒： updateTimeDisplay 函数会实时监控并更新显示的距离下一个日程的时间，并且提前10分钟会响铃和弹窗提醒。

3.1.3 课表导入

课表导入功能中设计了 ClassImport 槽函数，并调用文件读写模块中的 getClass 函数完成课表的读写

3.1.4 日程自动推荐功能

- 构建了一个封装的 GetSingle 接口，作为任何日程自动推荐函数的一个子接口，并利用 Available 函数判断事件合适的添加时间（例如如果上午最后一节有课，自动安排的午餐时间就将是12:00而不是11:30）。
- 构建了 GetFood 函数，用于根据用户设置提供早餐/中餐/晚餐的推荐。
- 构建了 GetStudy 函数，用于根据用户设置智能推荐自习场所：在探索模式中，利用$0-7$的随机二进制数实现上午/下午/晚上各$\frac{1}{2}$的推荐自习概率。

3.2 地图模块PKUMap

- 为了地图模块更高阶的交互的构建方便，我们首先定义了 ShowPath ， HidePath 作为底层接口，它们接收一个整型参数 idx 表示我们的边的序号，分别用来显示/隐藏第 idx 条边。

- 地图模块中包含一个 QGraphicsScene* 指针，指向我们以北京大学燕园地图为像素图的一个 QGraphicsScene 对象，我们之后显示路径，添加滚轮功能等操作都是添加在这个 QGraphicsScene 上的。
- 设计了一个 WheelEventFilter 封闭类，过滤掉除鼠标滚轮外的信号，用于实现鼠标滚轮对图像的缩放，提升用户体验。
- 设计了一个 ArrowLine 封闭类，包含一个 paint 接口，用于绘制其在 UI 中可以表现为一个红色的箭头。
- 地图模块设置了四个按钮，每个按钮对应着一个槽函数，分别为 Prev ， Next ， ShowAll ， HideAll ，用于逐步或者一次性显示/隐藏路径。其中路径会被保存在一个 ArrowLine* 类型的数组 edges 内。
- 在地图模块中我们设计了 Update 接口用来维护地图界面右下角显示的信息，告诉用户当天的行程的总边数，以及目前展示的线段数。

3.3 事件类Event

事件类是我们所有日程的最基本组成单位，我们设置了如下的几个变量

1. 事件名称： Sname ，用于保存事件名称
2. 事件地点： Sposition ，用于保存地点； iposition ，用于保存地点的序号
3. 开始/结束时间： QTime 对象 begin ， end

3.4 设置模块Config

为了满足用户的个性化需求，我们设计了功能完善的设计模块：

- 使用变量 mode 记录用户所选的模式, breakfast ， lunch ， dinner 记录用户是否需要推荐就餐， answer 记录用户是否有 明确的答复 ， dest 记录用户的选择的倒计时弹窗图片路径， origin 记录用户选择的宿舍位置，主窗口的行为由这里的变量决定。
- 在进行设置时，例如勾选某个项目时，变化将会反映在 UI 中空间的属性上，当我们点击确定或者取消按钮时，分别触发 Accept 和 Refuse 槽函数。 Accept 槽函数将会将 UI 控件的属性给予我们前面的变量，而 Refuse 槽函数则不会进行这样的操作。
- 无论是点击确认/取消或是直接关闭该设置窗口，都会触发 Synchronize 函数，使得 UI 中的控件属性与 mode 等内部变量一致，使得我们每次重新打开设置界面时， UI 控件的状态都和我们实际设置的状态保持一致。

3.5 选择模块Selection

为了给用户提供智能的日程自动推荐功能，我们设计了选择模块，其中集成了以下三种选择方式：

1. 随机选择：设计了 RandomSelection 函数，会从所有合法的同类型项目中随机选择一项
2. 最近距离选择：设计了 NearestSelection 函数，会考虑当前推荐事件的上一个事件和下一个事件，选择最小化与二者距离之和的事件
3. 偏好选择：设计了 PreferenceSelection 函数，用户访问次数较少的地点将会被优先自动推荐，具体流程如下：

   我们指定一个阈值10,初始化$acc = 0$随机选择一个序号起始,对所有序号循环做如下操作:

   $$acc += median \times \log\left(\frac{1}{1+num}+1\right)$$

   其中$median$是所有地点访问次数中位数和1中的最大值,num是对应节点的访问次数.直到$acc > 10$时的序号即为我们选择的序号

- 与此同时,为了保证随机数的随机性,我们使用系统时钟的时间作为我们的随机数种子,并且将随机数生成器作为静态变量保存.

3.6 倒计时模块CountDownTimer

考虑到北大同学对定时完成任务或者一段专注时间的需求，我们在日程管理中加入倒计时模块。倒计时部分采取常规的倒计时的结构，包括开始、重置和结束。我们的 CountDownTimer 模块通过精心设计的类结构和细致的功能实现，提供了一个易于使用且功能丰富的倒计时器工具。

1. CountDownTimer 类继承自 QWidget ，在构造函数中，初始化了用户界面（ UI ）组件，并设置了初始状态。特别地，使用 QLCDNumber 显示时间，设定为 5 位数，十进制模式，并以填充样式显示。初始显示为 "00:00" ，呈现简洁美观。
2. 倒计时功能通过 QTimer 实现。每秒钟定时器触发一次 updateTime 槽函数。该函数负责更新显示的时间：秒数减一，并重新计算分钟和秒数。如果倒计时结束，定时器停止，并弹出提示框通知用户时间到。这种用户友好的设计，确保了用户体验的流畅性。
3. 模块提供了三个主要操作按钮：开始（ Start ）、结束（ End ）和重置（ Reset ）。在用户点击开始按钮时，计时器启动，并禁用开始按钮，启用结束按钮，防止重复启动。结束按钮则用于停止计时器，并重置时间。重置按钮允许用户输入新的倒计时分钟数，使用 QInputDialog 获取用户输入，如果输入有效，调用 resetTimer 函数重新设置倒计时。
4. resetTimer 函数不仅重置了秒数和分钟数，还更新了 UI 显示，并重置按钮的状态。这种设计保证了在任意时刻重置倒计时的灵活性和准确性。

3.7 文件读写模块FileIO

由于本项目中各类信息较多，为了方便维护，例如地图节点信息和默认的事件池按照格式记录在csv 文件中，而导入课表则要处理Excel 文件，更好记录用户信息也需要大量的json 文件交互操作，因此单独设计一个文件读写模块十分有必要。

3.7.1 使用的数据文件

- nodes.csv 用于储存地图上的节点信息,程序启动时从中获取节点坐标及其序号
- events.csv 用于储存默认事件池的信息,程序启动时从中获取地点,类别等信息
- activities.json 用于记录用户选择保存到本地的信息
- course.json 用于记录用户导入到本地的课表
- UserInfo.json 用于记录用户对各个节点的访问次数

3.7.2 部分数据文件的格式

我们的activities.json 和course.json 中每一项都是一天的日程安排，其中每一个事件按照我们规定的Info 格式进行记录：（变量名称参见3.3）

$$\text{Sname,Sposition,iposition,begin,end}$$

多条Info 利用英文分号';'分割：

$$\text{Info}_1;\text{Info}_2;\cdots;\text{Info}_n$$

形成一整个字符串代表整天的日程安排。有了这样明确的格式之后，我们设计了利用Info 的Event 构造函数，和将 Event 转换为Info 格式的转换函数，从而辅助我们的文件实际读写操作。

3.7.3 设计的文件读写函数

- 课表导入：使用QAxObject 类完成Excel 的读取，实现允许用户将门户中的课表信息以xlsx 格式导入到日程系统中，导入的课表包括课程名称、时间、地点等详细信息。
- 初始化Json 文件：使用initUserInfo 接口，初始化用户的访问记录，课表和保存到本地的事件
- 读取Json 文件：使用readUserInfo 接口，从某个文件中读取指定的某条数据
- 写入Json 文件：使用saveUserInfo 接口，向某个指定文件中写入指定数据
- 读取节点和事件：利用getNodes 函数和getEvents 函数读入保存在csv 文件中的节点和事件。

3.8 爬虫模块

考虑到北京大学丰富的讲座资源和北大同学对于学术信息的需求，我们为北京大学的同学提供每日的讲座信息以丰富自己的每日活动。我们从北京大学讲座网中提取讲座信息：

$$https://resource.pku.edu.cn$$

1. 这个网站的讲座信息由 javascript 即时渲染，一般的对静态网页 html 源代码的操作不能获得相关信息，而由于 C++ 的爬虫功能相对 python 较弱而且 Qt 的 network 在 qt5.6 的版本下不支持，所以我们选择了 selenium 来模拟对浏览器的操作。

2. 在 C++ 的项目里运行 py 脚本主要有 2 种方法，一种是通过官方 python.h 调用，另一种是采用 QProcess 进行多进程调用。实际选择中我们发现第一种方法可能需要系统级别的 LIBS 对于项目文件的要求更多，第二种方法的实现更为简单易行。在 QProcess 中利用命令行参数和 standardoutput 实现了主项目与 lecture_info.py 的交互

3. 讲座网上的日期不好选择，我们没有找到直接定位某个日期的做法，于是年月我们利用 python 的 time 模块得到当天的时间然后采用了对 prev ， next 4个按钮的操作定位到target date。

4. 写爬虫时出现了一些小问题，有些细节需要注意：

   （1）在日期点击之后必须等讲座日期加载完毕，所以要加 time.sleep(5) 防止爬取了未加载时的信息

   （2）如何精准定位讲座信息是有一定困难的，我们也学习了网页的相关内容

4 小组成员分工情况

- 徐海翁：事件类 Event ，选择模块 Selection ，地图模块 PKUMap ，设置模块 Config ，文件读写模块 FileIO 等模块的设计和实现，主界面中日程自动推荐机制，手动个性化推荐功能的设计和实现，利用 QProcess 整合 Python 爬虫模块到 Qt 项目中， README 文件和作业报告的书写， python 代码环境配置的导引书写。

- 黄源森：主界面中日历和日程表高级交互的设计和实现，事件添加的通用模板设计，文件读写模块中课表导入功能的设计和实现，倒计时 CountDownTimer 模块的设计和实现，利用 Python 和 Selenium 库模拟用户实际操作实现网络爬虫功能

- 谢宇翔：日程倒计时机制的实现，菜单和主界面的 UI 美化。

5 项目总结与反思

- 代码组织上：在实际进行大规模的代码实现前，我们首先给例如日程表的交互封装了很多底层接口，例如插入一项 AddItem 等，这样的接口极大方便了日后代码的维护和优化，例如我们想要修改表格中所有内容的对齐方式时，由于所有项都是采用 AddItem 插入的，只需要对 AddItem 添加一行代码即可。

- 模块设计上：我们对部分模块例如选择模块 Selection ，地图模块 PKUMap ，设置模块 Config ，文件读写模块 FileIO 进行了较好的封装以及模块化处理，有效地减少了代码地相互依赖性，并按照模块进行了较好的分工合作，使得代码结构相对清晰，且维护起来相对容易。当然由于时间限制以及部分代码的依赖性，在主界面中我们没有将日历和日程表这两个功能复杂且几种的组件进行模块化，导致 mainwindow.cpp 文件相对冗长，维护性受到些许影响，而自动推荐事件的部分功能由于过程性较强，出现了部分重复性较强的代码。

- 项目创意上：我们的项目聚焦日程安排的个性化，力图让各类用户在不同场景下(例如走出舒适区，内卷，摆烂)都能使用这个程序，因此花了较多精力在个性化的设计上，从用户的角度思考合适的推荐形式，给用户提供了多种选择。但与此同时由于个性化以及用户群体的广泛导致了我们项目模块类别较多且独立，在部分细节的设计有些粗糙, UI 的美化上都有很大改进空间。

- 技术使用上：由于我们项目模块类别较多且独立，在项目的完成中我们接触了非常多的模块与库，处理了各式各样的问题，锻炼了组员的学习能力和代码能力，我们主要使用的相对复杂的技术包括以下：

  1. 利用 QAxObject 操作 excel 等表格数据，并处理编码问题转化为 Event 类对象
  2. 使用 QGraphicScene 来展示燕园地图，从而支持路径在像素点上的精确显示
  3. 使用 Python 中的 Selenium 外部库实现爬虫功能，模拟人类使用浏览器时进行的行为在北大讲座网中选择日期获取讲座信息。
  4. 在 Qt 中利用 QProcess 类实现运行 Python 代码，并且通过其标准输出获取运行结果
  5. 我们同样尝试了使用 Qt 内置的 Network 模块完成爬虫任务，但是由于我们使用的 Qt 版本相对较低，支持的 OpenSSL 版本较低，在配置 OpenSSL 以实现访问 https 类型网页时遇到困难，因此我们采取了前面基于 Python 库 Selenium 的方法。
  6. 文件读写模块 FileIO 中读写了 csv 和 json 两种格式的文件，分别为程序预设数据如地点与用户数据两类。
  7. 虚拟环境配置：由于我们的爬虫代码基于 Selenium 库，这需要虚拟环境的参与，我们在 README.md 文件中详细给出了使用命令行完成虚拟环境配置的过程

- 沟通协作上：我们的项目使用 Github 进行代码托管，在项目早期小组成员共同讨论项目设计思路，探索各类设计的可行性，在代码的具体实现阶段，组长安排每一阶段工作任务，组员在小组群内及时沟通交流，分工到个人，沟通高效。但在项目早期由于相关经验缺乏，没有统一 Qt 的版本，之后花费了部分时间进行调整。