高级程序设计

第三阶段课程项目报告：文档预览器

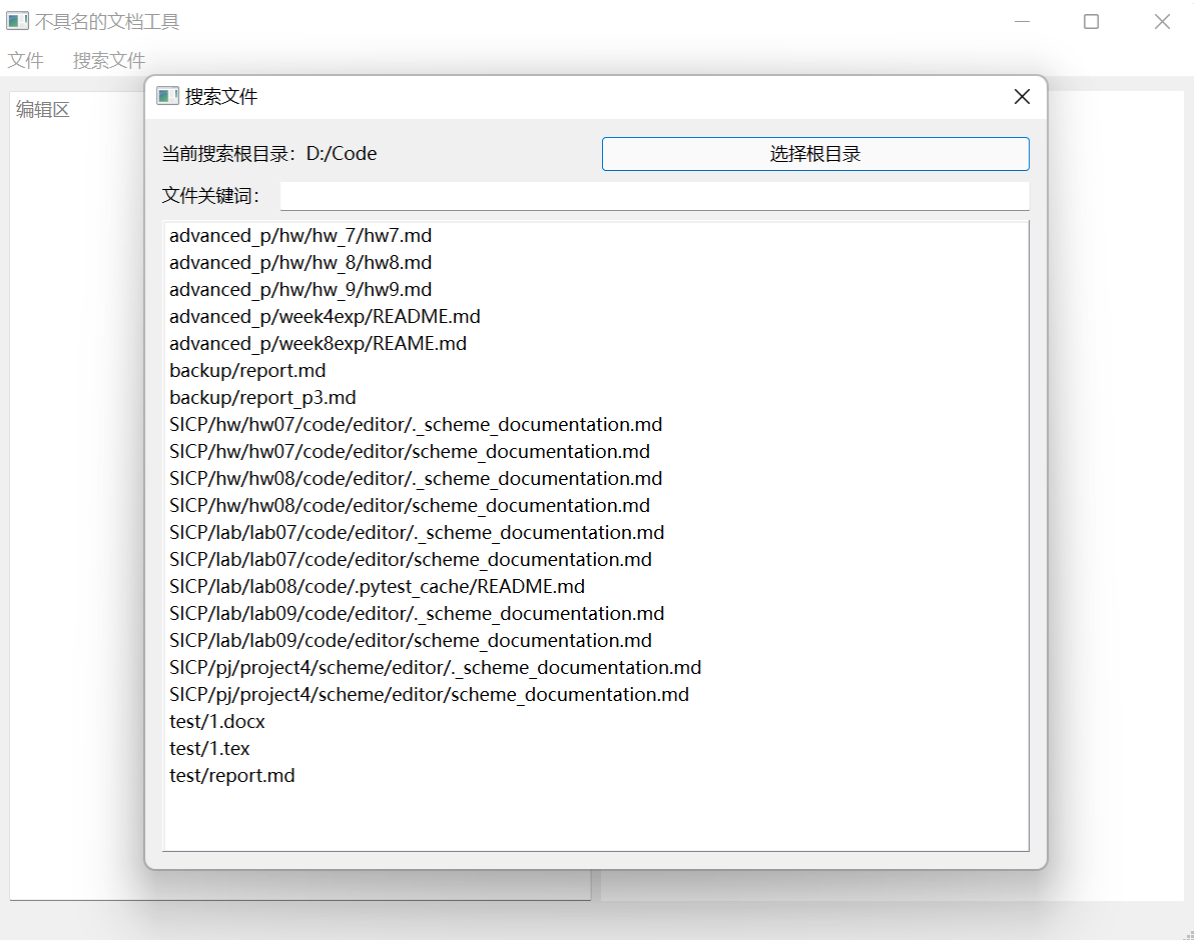
201220115 杨青云

功能

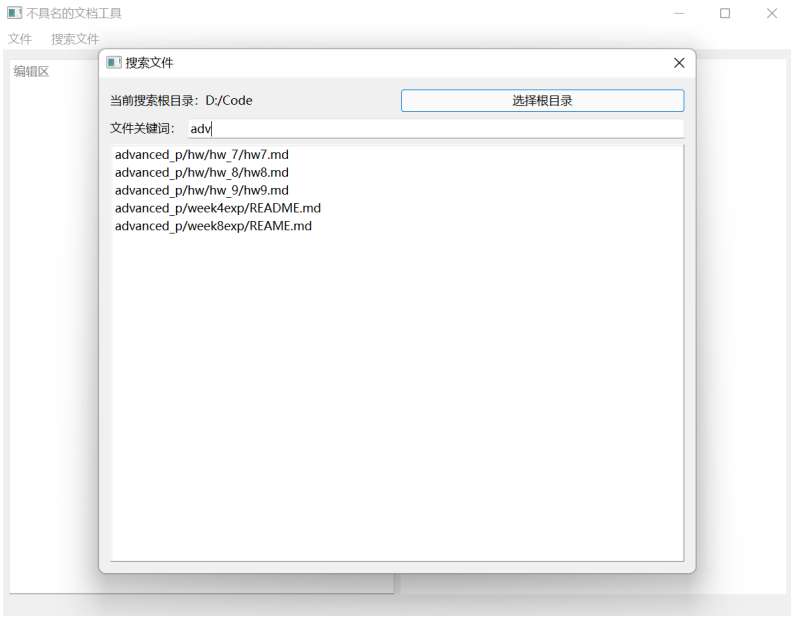
在前两个课程项目的基础上，新增了文件查找功能。只需指定根目录，即会展示出该目录及其子目录下所有`.md`, `.tex`以及`.docx`文件的名称及相对根目录的路径。提供一个搜索框，用户可以在搜索框中输入正则表达式，程序会实时筛选出**文件名**匹配该表达式的文件；当然，也可以输入其他任何字符，程序会实时筛选出**文件名或相对路径**中含有输入字符的文件。

对于在列表中的文件，用户双击即可打开并进行文档的实时预览和编辑（其中编辑功能仅支持`.md`和`.tex`文档）。

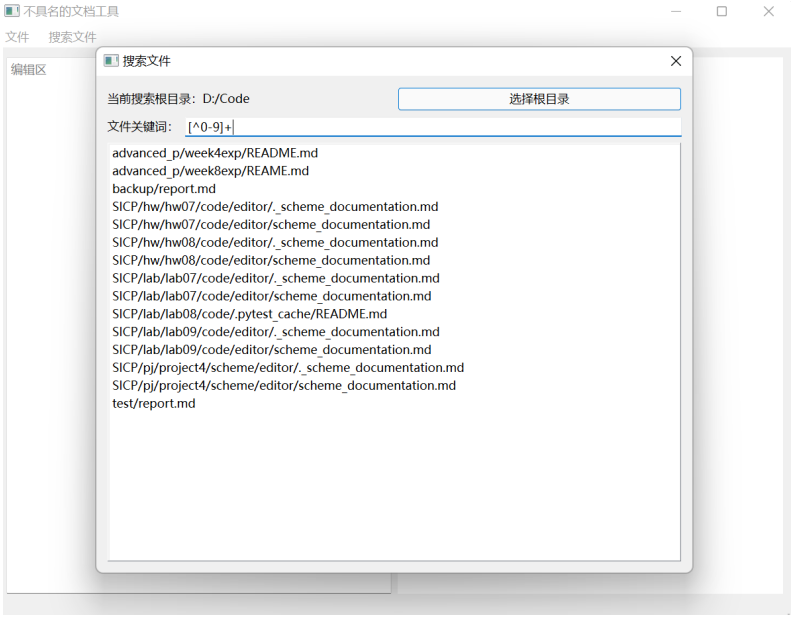
效果展示



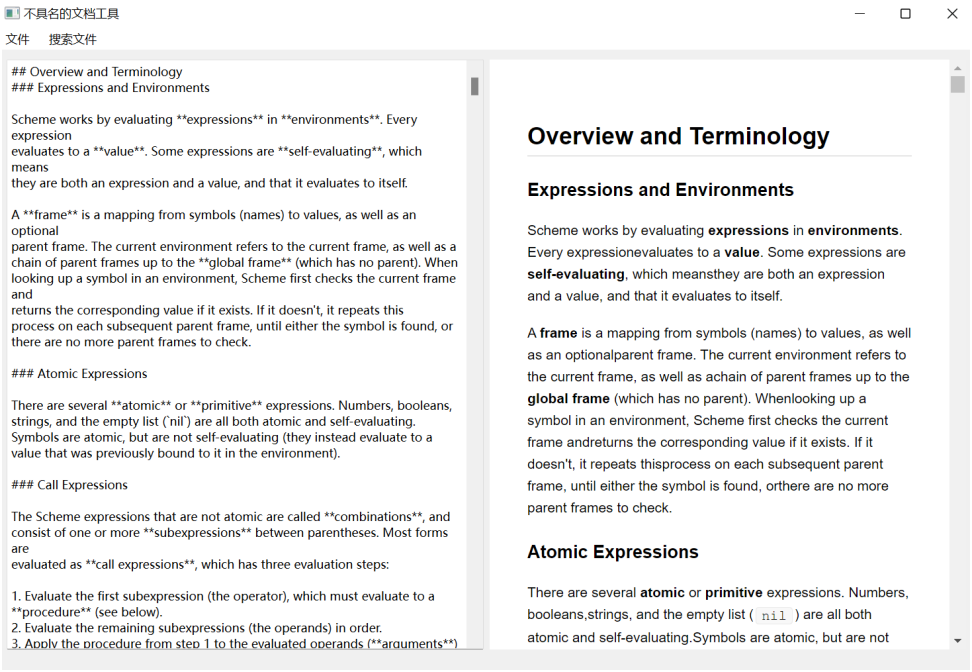
图：搜索框无输入时，显示根目录下所有可预览文档及其相对路径



图：搜索框输入不为正则表达式时，筛选出含有输入字符串的文件（图中的所有文件相对路径或文件名中都有"adv"）



图：搜索框输入为正则表达式时，筛选出文件名匹配的文件（图中所有文件名都不含数字）



图：双击打开文件

具体实现

利用Qt的自定义对话框实现搜索文件功能的交互界面，按下“选择根目录”后会调用系统的打开目录对话框，如果成功获得目录信息，则将目录信息更新到按钮左侧的QLabel组件，并立即调用函数getFiles递归获取目录下的所有文件名及相对路径

```
void getFiles(QString path, vector<QString> &files, QString subdir) //path: 根目录, subdir:相对根目录的路径, files: 存储文件名及路径
{
    long long file_handle = 0; //文件句柄
    _finddata_t file_info; //文件信息
    QString p = path+"/*.*"; //准备获取当前文件夹下的所有文件
    QByteArray tmp = p.toLocal8Bit(); //将QString转换为QByteArray，并在下面转为char*，这是为了解决中文路径乱码问题
    if ((file_handle = _findfirst(tmp.data(), &file_info)) != -1)
    {
        do
        {
            if (strcmp(file_info.name, ".") != 0 && strcmp(file_info.name, "..") != 0)
            {
                if (file_info.attrib == _A_SUBDIR)
                {
                        getFiles(path+"/"+QString::fromLocal8Bit(file_info.name), files, subdir+QString::fromLocal8Bit(file_info.name)+"/"); //递归搜索子目录，同样为了中文兼容性转换编码
                }
                else
                {
                    files.push_back(subdir+QString::fromLocal8Bit(file_info.name)); //存储文件名及路径
                }
            }
        } while (_findnext(file_handle, &file_info) == 0);
        _findclose(file_handle);
    }
}
```

值得一提的是，原先的文件句柄定义为long file_handle，却无法正确获取文件，反而会使程序崩溃，经查询得知Windows 10及之后的Windows文件句柄类型需要为long long，修改后正常工作。

由于获取文件的_findfirst等接口只接受char *类型的字符串，所以需要将QString转换为char *类型，由于编码问题，直接转换会产生中文路径乱码问题，故采用上述方案。

获取文件列表之后，就要将它们绘制出来，这里采用QListWidget类存储和绘制文件列表，自行定义draw_search_result()函数，在每次更改根目录以及搜索框内容发生变化时调用该函数更新文件列表，

```
void MainWindow::draw_search_result()
{
    search_result->clear();
    try
    {
        std::regex name_reg(search_content->text().toStdString()); //从C++11开始支持regex
        //输入是合法的正则表达式
        //进行正则表达式匹配和子串匹配
    }
    catch (std::regex_error)//输入不是合法的正则表达式
    {
        //只进行子串匹配
    }
}
```

更新文件列表时，先清空显示列表，然后尝试利用输入生成正则表达式并进行异常捕获，如果出现`regex_error`，则说明是常规字符串，否则是正则表达式。对`getFiles`返回的文件列表`vector`中的元素进行遍历，找出其中符合要求的文件名并打印出来。

将列表项被双击的信号与打开列表项文件的函数`open_file_list(QListWidgetItem *item)`绑定，双击列表项就会调用这个函数，因为存储了相对路径，因此将该路径与根目录拼起来即可获得路径，这时就可以关闭搜索对话框后打开文件并调用相应处理函数进行文件预览了。

QString与char*的相互转换: https://blog.csdn.net/qq_33485434/article/details/78790285

解决Win10下_findnext()异常: https://blog.csdn.net/hemmingway/article/details/73716980

C/C++遍历目录下的所有文件: https://www.cnblogs.com/collectionne/p/6815924.html