**РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ГАЗА (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРИСИТЕТ) ИМЕНИ И.М. ГУБКИНА**

**Кафедра автоматизированных систем управления**

*Курсовая работа по дисциплине:*

***«Основы алгоритмизации и программирования»***

на тему: «Создание программы расписания занятий»

*Выполнил:* Ханевский Ярослав Александрович

студент группы АС-23-04

*Проверил:*

Москва, 2024

**Оглавление**

[**Введение** 3](#_Toc168151458)

[**Постановка цели и задач и выбор инструментов** 4](#_Toc168151459)

[**Описание работы программы** 6](#_Toc168151460)

[**Заключение** 21](#_Toc168151461)

[**Список использованных литературных источников** 22](#_Toc168151462)

# **Введение**

Актуальность выбранной темы состоит в том, что существующий веб-формат расписания занятий имеет несколько существенных недостатков: большие затраты времени для доступа к расписанию (в среднем 20 секунд), доступ к расписанию занимает несколько действий (запуск браузера, вход в личный кабинет, переход на вкладку расписание), необходимость запускать тяжеловесный браузер и др.

Практическая значимость разрабатываемого ПО велика, так как студенты в период обучения каждый день сталкиваются с необходимостью ознакомления с актуальным расписанием.

При рассмотрении выявленных проблем существующего формата расписания были обозначены некоторые постулаты, которых нужно придерживаться при разработке программы. Во-первых, вывод расписания должен производиться сразу после запуска программы. Во-вторых, никаких лишних элементов интерфейса быть не должно. Созданная в результате разработки программа удовлетворяет поставленным постулатам и не имеет вышеперечисленных недостатков.

# **Постановка цели и задач и выбор инструментов**

Любую программу можно разделить на две составляющие: frontend и backend. Frontend отвечает за взаимодействие с пользователем, включая интерфейс и визуальные элементы, в то время как backend управляет серверной логикой, базами данных и обработкой запросов. Совместная работа этих двух частей позволяет создавать полноценные и функциональные приложения, обеспечивая пользователям удобное взаимодействие. Специализированные языки программирования позволяют разработать удобный и однозначный интерфейс. Так, в данной курсовой работе в качестве языка программирования используется Python, так как он обладает рядом преимуществ: простым и понятным синтаксисом, что делает его легче изучаемым для начинающих, множеством устанавливаемых библиотек (которые в дальнейшем будут использованы), позволяющим расширить возможности языка, универсальностью для решения задач. Python не компилируется и не нужно создавать исполняемый файл, что упрощает процесс отладки программы и ускоряет разработку. Именно благодаря устанавливаемым библиотекам можно с легкостью организовать интерфейс и разработать программу.

Для разработки frontend’а были выбраны инструменты PyQt и Qt Designer. PyQt - это набор Python-привязок для фреймворка Qt, который используется для создания кроссплатформенных графических пользовательских интерфейсов (GUI). Эта библиотека позволяет разработчикам писать программы на Python с мощными и гибкими интерфейсами, используя компоненты и функциональные возможности, предоставляемые Qt. Qt Designer — это визуальный инструмент с технологией “Drag and drop” для разработки графических пользовательских интерфейсов (GUI) с использованием фреймворка Qt. Он позволяет разработчикам создавать и настраивать интерфейсы путем перетаскивания виджетов и настройки их свойств без необходимости написания кода.

В качестве текстового редактора был выбран Sublime Text. Sublime Text — это мощный и высокопроизводительный текстовый редактор, который поддерживает широкий спектр языков программирования и разметки, предлагая возможности для настройки и расширения через плагины и пользовательские настройки.

Для контроля версий программы был использован Git и Github. Git — это система контроля версий, которая используется для отслеживания изменений в исходном коде и упрощения совместной работы разработчиков. GitHub — это платформа для хостинга и совместной разработки программных проектов с использованием системы контроля версий Git.

Для создания программы были выбраны следующие инструменты:

1. Sublime Text (версии 4169)
2. Python (версии 3.9.13)
3. PyQt и Qt Designer (версии 5.15.9 и 5.11.1 соответственно)
4. Git и Github (версии 2.44.0)

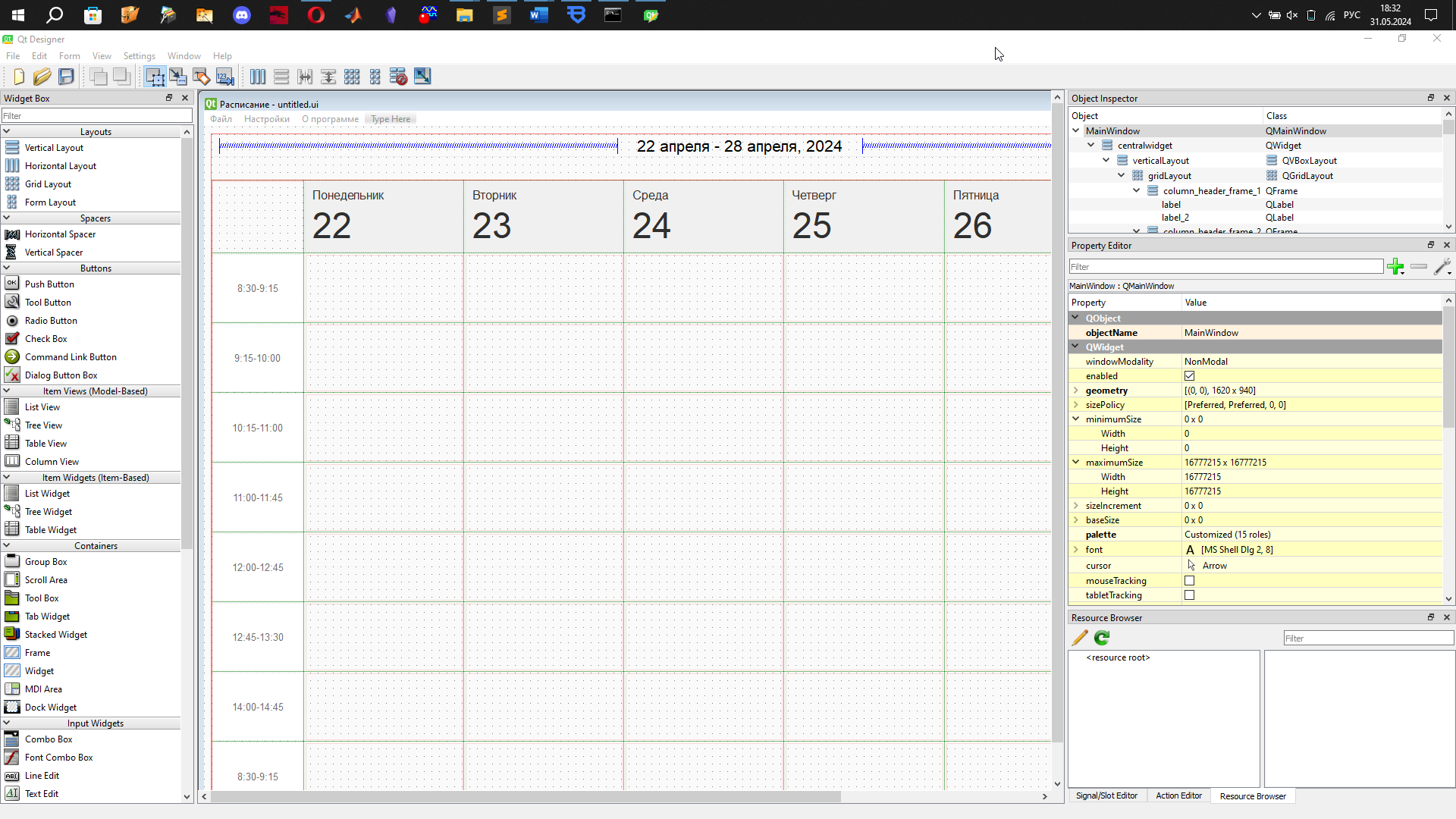
Цель: разработать ПО, отображающее расписание занятий для учебной группы АС-23-04.

Задачи:

1. Придумать и создать интерфейс программы в Qt Designer
2. Разработать функции прорисовки элементов окна
3. Написать функции обработки кнопок

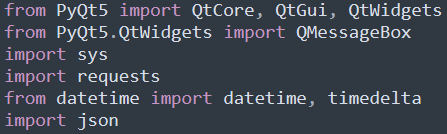
# **Описание работы программы**

Первым делом был создан дизайн программы в Qt Designer:

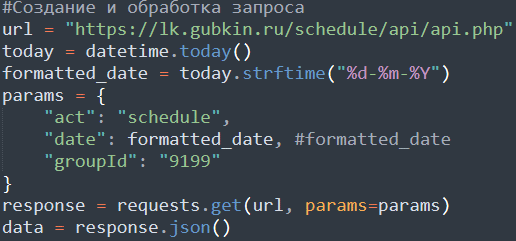


В окне располагается главный виджет (объект QWidget) centralwidget, включающий в себя все элементы интерфейса. Внутри centralwidget находится виджет (объект QVBoxLayout) layout, который обеспечивает вертикальное расположение дочерних виджетов. Он включает в себя элемент scrollArea (объект QScrollArea), в который помещён виджет (объект QWidget) scrollAreaWidgetContents. Элемент scrollArea служит для создания области прокрутки. Это необходимо, так как полное расписание не помещается в окно. Внутри scrollArea находится виджет (объект QGridLayout) gridLayout, который обеспечивает табличное расположение дочерних виджетов. gridLayout является каркасом расписания, в ячейках которого располагаются объекты QFrame. Они могут быть стилизованы нужным образом и включать в себя виджеты. Все текстовые элементы являются объектами QLabel. Данное описание содержит лишь основные элементы окна ввиду того, что код, сгенерированный после преобразования проекта Qt Designer в Python файл, был отредактирован, и многие элементы, созданные в Qt Designer, были изменены или удалены.

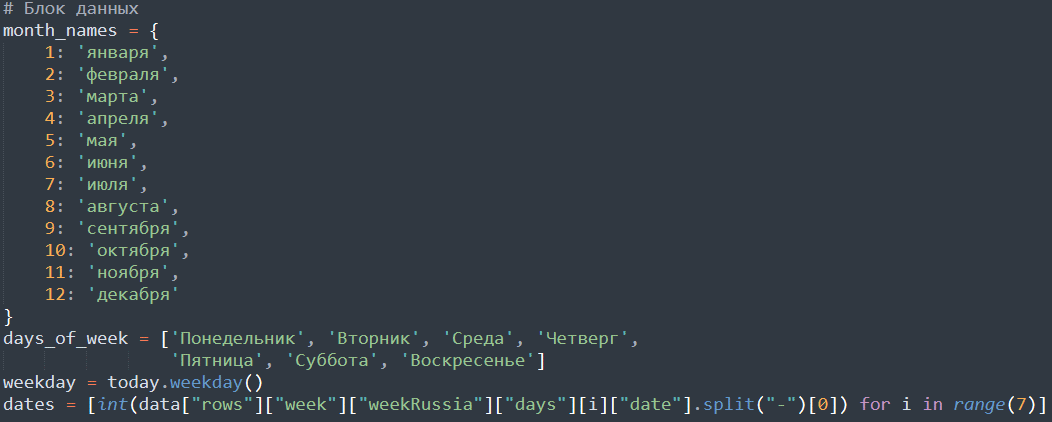
Перейдём к коду программы. Были импортированы следующие библиотеки:



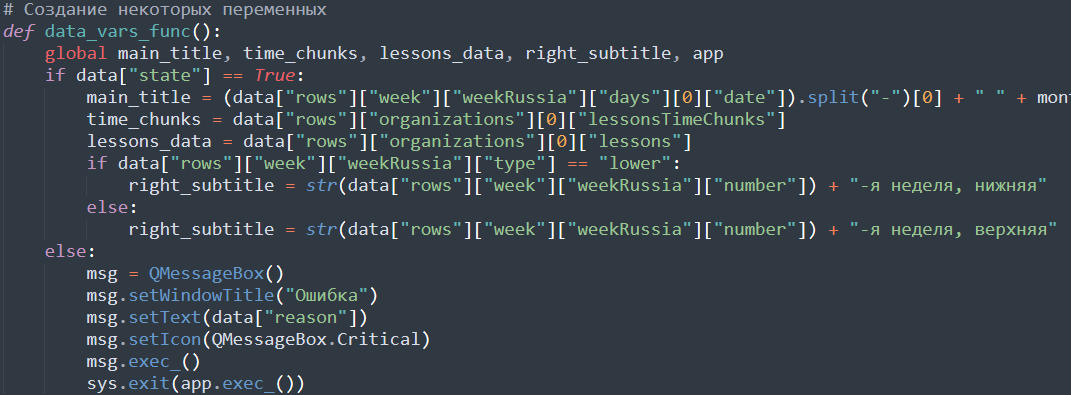
Библиотека sys будет использована для получения информации о системе, requests – для создания запроса к серверу, datetime – для управления системной датой и временем, json – для обработки ответа сервера. Код создания и обработки запроса:



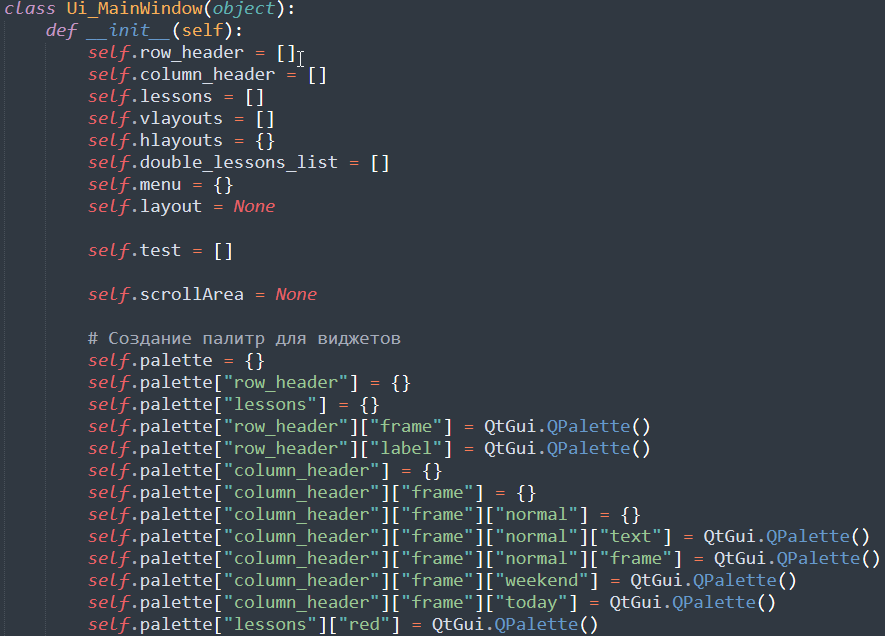
Переменная url содержит ссылку на сервер с расписанием занятий. В переменную today помещается системная дата, после чего преобразуется в нужный формат и присваивается переменной formatted\_date. Словарь params содержит некоторые параметры, необходимые для создания запроса к серверу. Создаётся запрос, результат находится в переменной response. Последним шагом ответ сервера в формате json преобразуется в словарь data. В следующем фрагменте кода создаются некоторые переменные, связанные с системной датой:



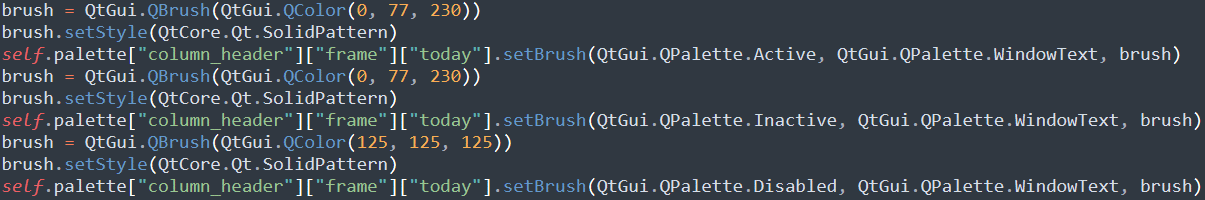
Массив dates заполняется днями месяца текущей недели. Следующая функция создаёт переменные для вывода в окно:



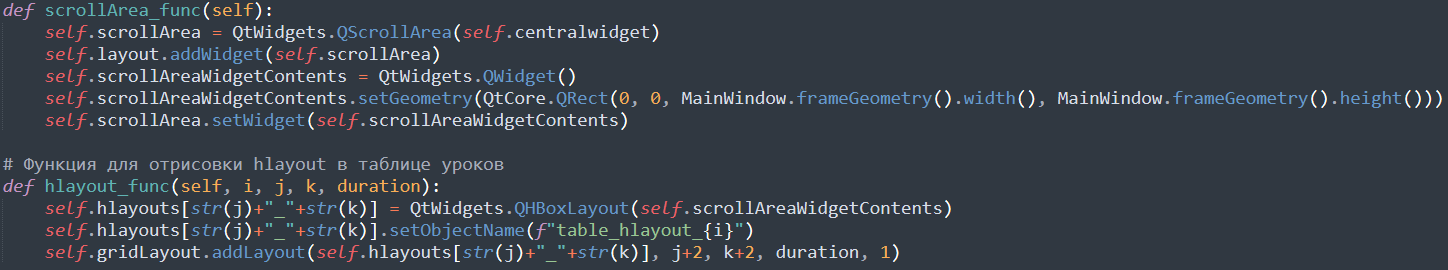
Переменная main\_title содержит текст главного заголовка, time\_chunks – массив временных отрезков занятий, lessons\_data – словарь, содержащий информацию о занятиях, right\_subtitle – текст правого заголовка. Если флаг state = False (расписания на неделю нет), то выводится окно с ошибкой, причина ошибки берётся с сервера, программа закрывается. Далее создаётся класс главного окна:



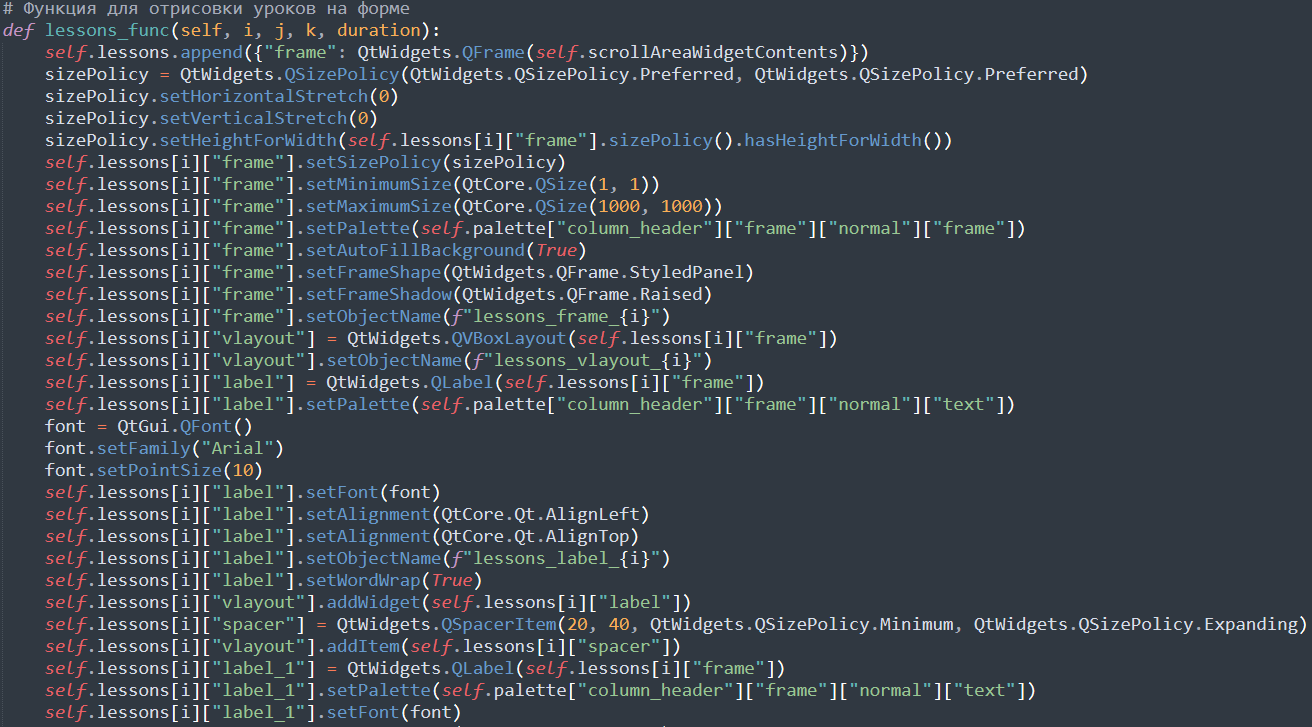
Первый блок кода инициализирует списки и словари, в которые будут помещаться виджеты окна, второй – палитры (наборы цветов), используемые в виджетах. Затем идёт создание палитр, приведённый ниже фрагмент кода – пример создания палитры:



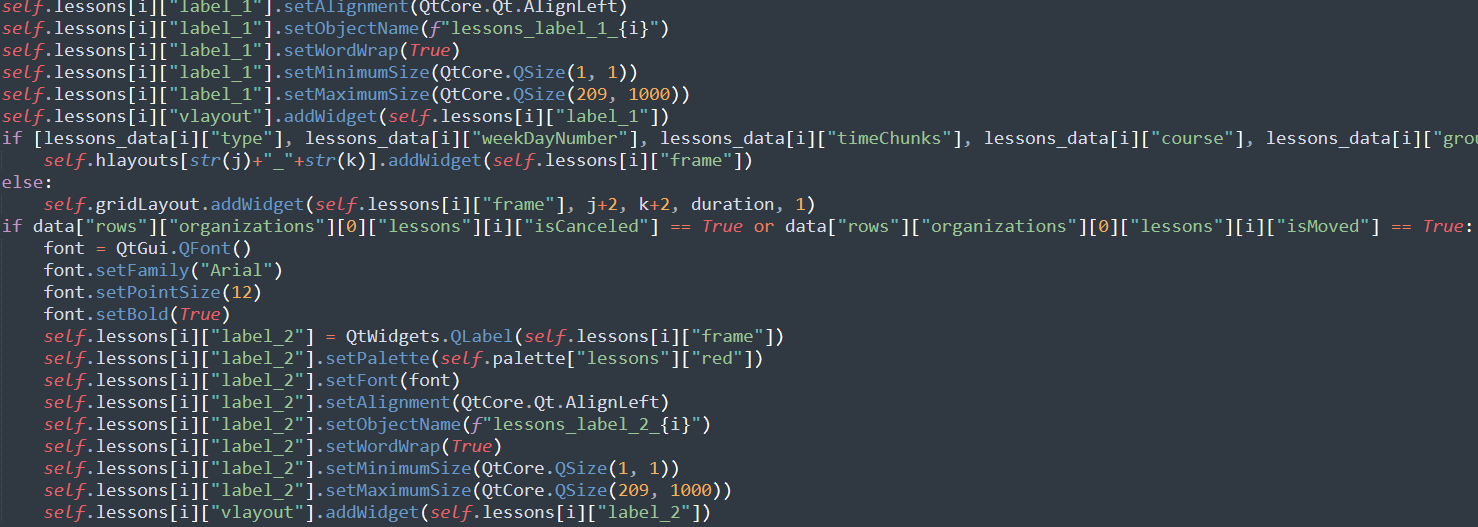
В данном случае, создаётся палитра для заголовков столбцов, дата которых совпадает с системной датой. Далее приведен код функции создания поля прокрутки и отрисовки hlayout в таблице уроков:



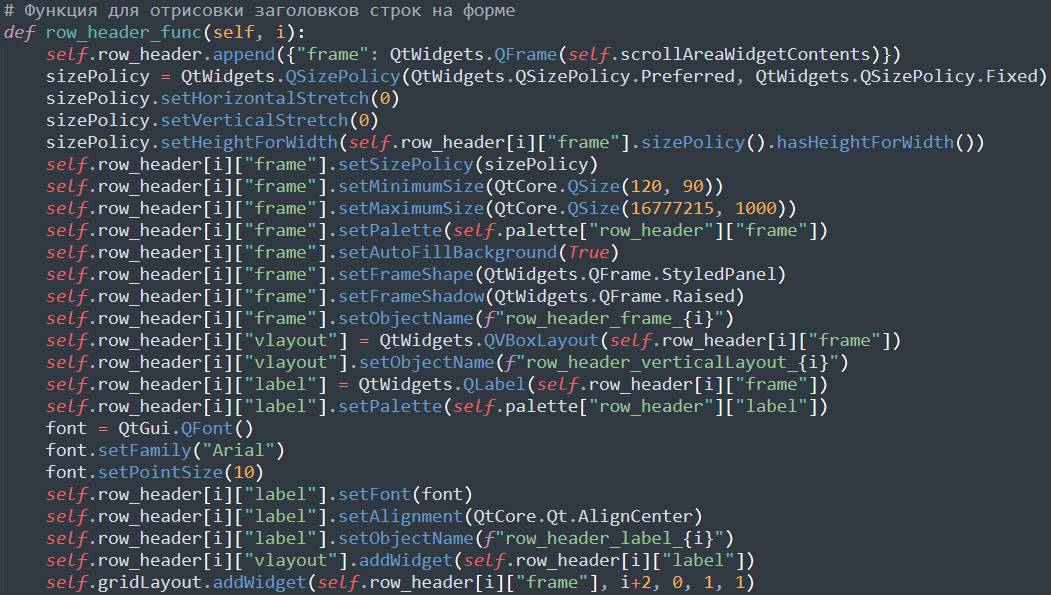
Вторая функция принимает на вход переменные i, j, k, которые отвечают за порядковый номер поля hlayout и за ячейку в таблице расписания, duration – длительность пары или мероприятия. Hlayout используется в случае, когда мероприятий в данный промежуток времени несколько для того, чтобы виджеты QFrame, создаваемые в каждой ячейке таблицы, располагались горизонтально в ячейке. После этого создаётся функция для отрисовки уроков на форме:



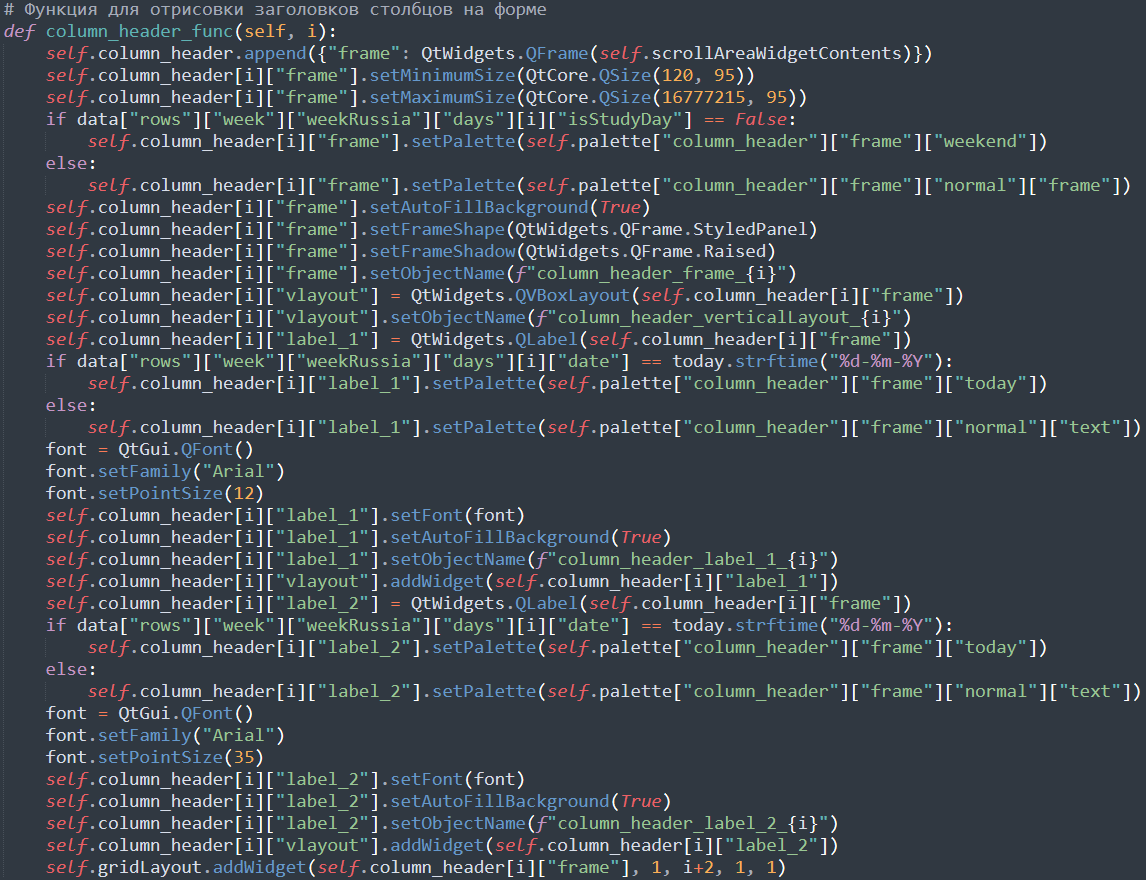
Сначала для ячейки создаётся элемент объекта QFrame, в который будут помещаться все дальнейшие виджеты ячейки и помещается в словарь lessons, содержащий все виджеты, связанные с занятиями. Затем создаётся элемент объекта QVBoxLayout, служащий для горизонтального расположения текстовых виджетов внутри ячеек. После этого создаются сами текстовые виджеты объекта QLabel и элемент объекта QSpacerItem, служащий в качестве распорки между текстовыми виджетами, которая «расталкивает» виджеты вверх и вниз. Следующий фрагмент кода данной функции:



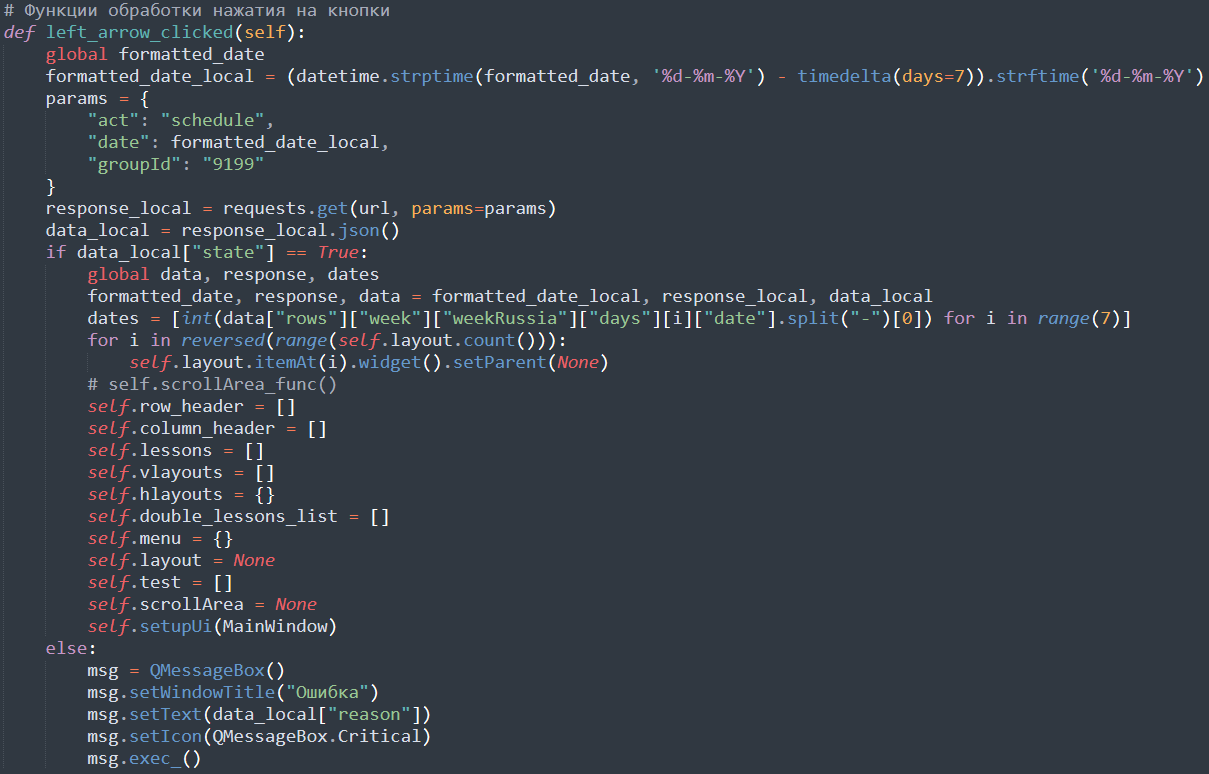
После создания текстовых виджетов следует условие: если мероприятий в промежутке времени несколько, то в hlayout добавляется frame, иначе frame добавляется сразу в ячейку. Далее если мероприятие отменено или перенесено, то добавляется ещё один текстовый виджет с информацией об отмене или переносе. Затем следует функция для отрисовки заголовков строк на форме:

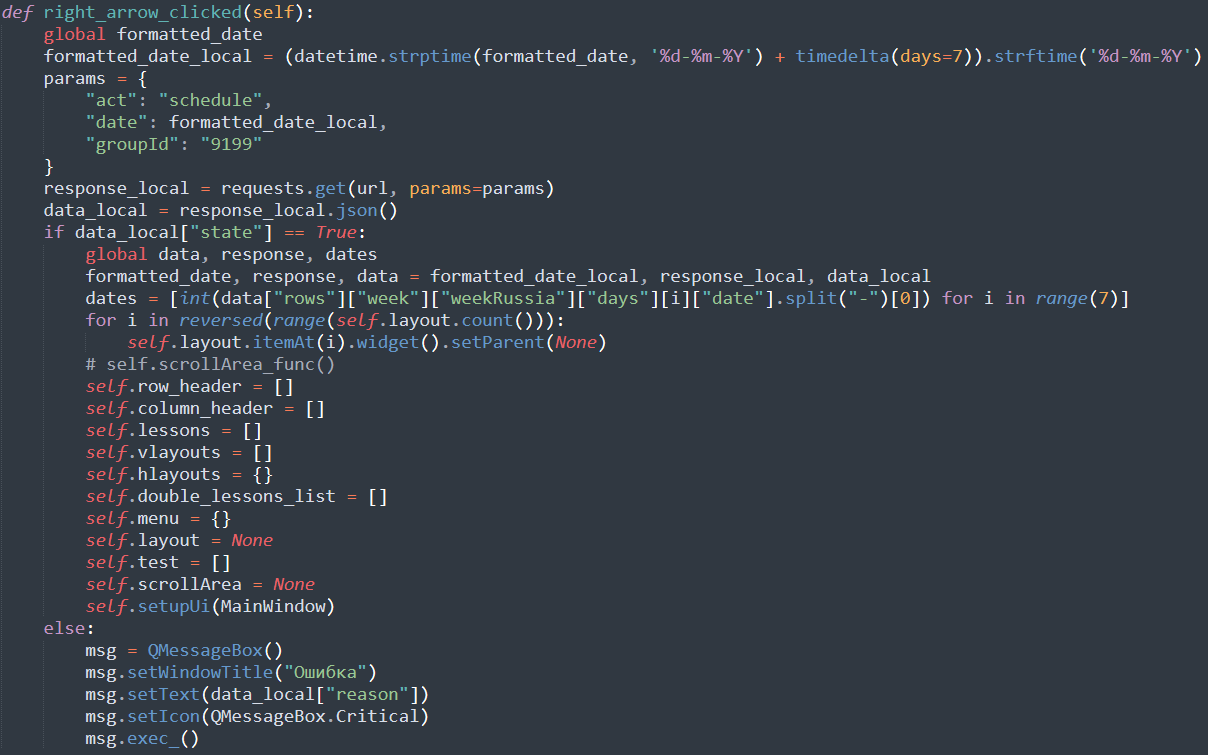


Создаётся frame, в него помещается vlayout, а в него – label. Функция для отрисовки заголовков столбцов на форме:

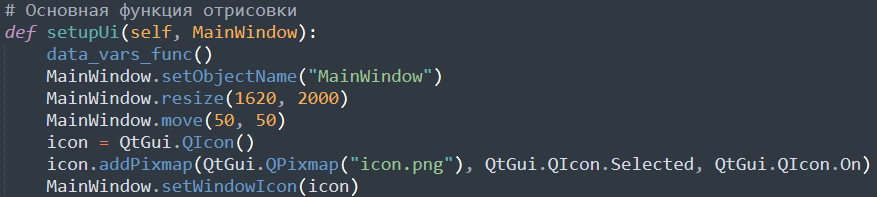


Так же, как и в предыдущем случае создаётся frame, далее происходит его стилизация в соответствии с условием: если день нерабочий, то производится одна стилизация, если рабочий – другая. Затем создаётся vlayout и label\_1, который стилизуется таким образом: если дата заголовка совпадает с текущей датой, то производится одна стилизация, иначе – другая. Аналогичная стилизация применяется и к label\_2. Функции обработки нажатия на кнопки:

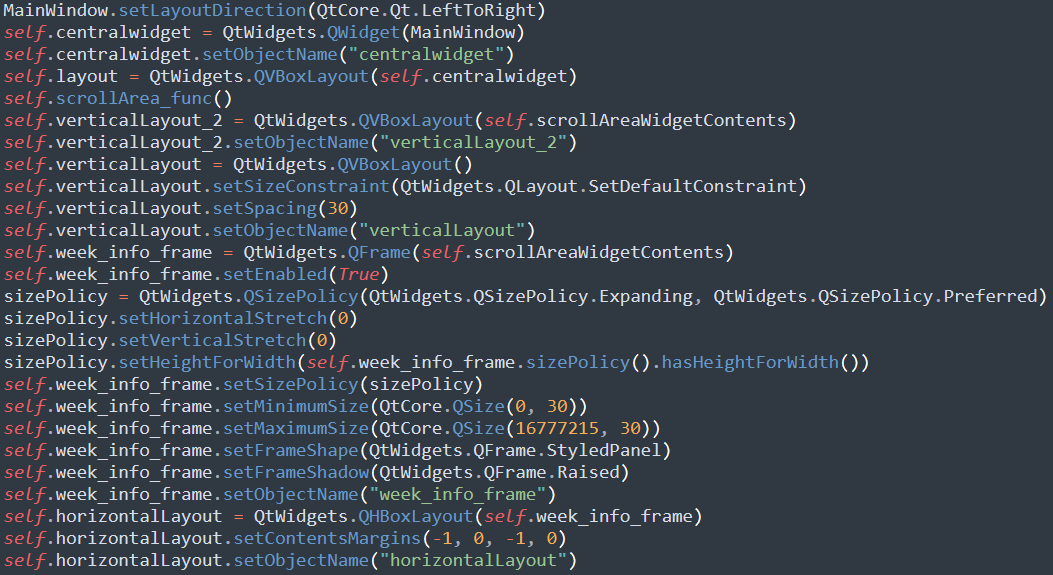




Данные функции работают похожим образом. Сначала создаётся список formatted\_date, которому присваиваются даты следующей недели. После этого создаётся и обрабатывается запрос с обновлёнными параметрами. Затем если есть расписание на выбранную неделю, то локальные параметры становятся глобальными, все виджеты в окне удаляются, списки и словари с виджетами очищаются и снова запускается функция отрисовки окна. Иначе выводится окно с ошибкой, при этом выхода из программы не осуществляется. Далее следует код основной функции отрисовки:



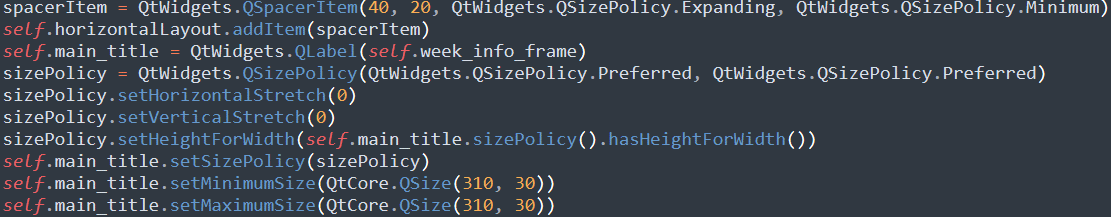
Функция принимает на вход элемент объекта QMainWindow, который отвечает за создание самого окна. Сначала вызывается функция data\_vars\_func, которая создаёт заголовки для окна, затем происходит конфигурация окна и установка иконки программы. После этого следует код создания палитры цветов для MainWindow (код большой, поэтому он не представлен здесь). Другой фрагмент кода функции:



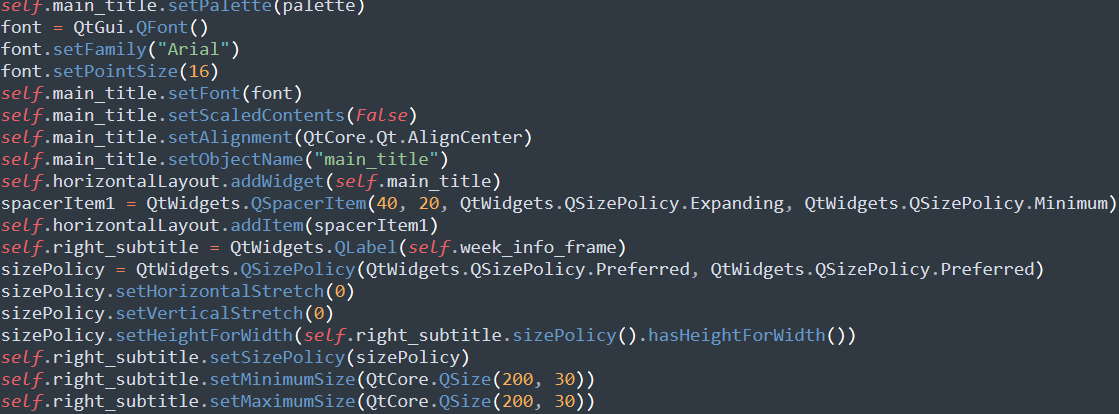
Вначале создаётся centralwidget (объект QWidget), в него помещается layout (объект QVBoxLayout), вызывается функция scrollArea\_func, создающая поле прокрутки. После этого создаются verticalLayout и verticalLayout\_2, служащие для правильного расположения дочерних виджетов. Создаётся week\_info\_frame (объект QFrame), служащий для вывода основного заголовка, horizontalLayout (объект QHBoxLayout), необходимый для правильного отображения кнопок (стрелок) и заголовков, которые будут добавлены ниже:



Важно заметить, что в PyQt нет встроенного виджета для отображения картинки, однако объект QLabel имеет метод setPixmap, позволяющий вставить элемент объекта QPixmap в label. Все 3 картинки выводятся в окно путём добавления их в horizontalLayout. Для стрелок определяются функции, которые будут исполняться при нажатии на кнопки. Lambda функция используется для корректного запуска функций. Следующий фрагмент кода функции:



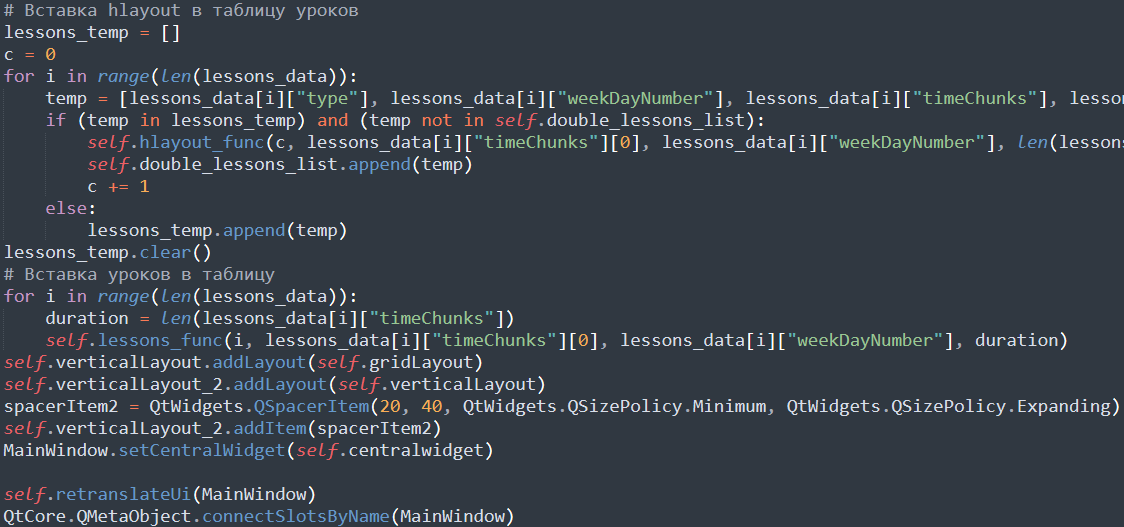
Здесь создаётся spacerItem (объект QSpacerItem), который «расталкивает» блок стрелок и главный заголовок, создаваемый далее (main\_title). После этого опять создаётся палитра цветов (для main\_title). Другой фрагмент кода функции:



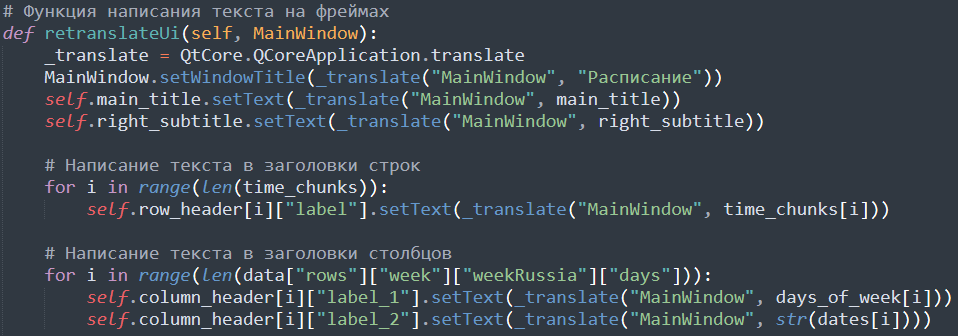
Конфигурируется main\_title, создаётся spacerItem1, «расталкивающий» main\_title и right\_subtitle (правый заголовок), создаваемый далее. После этого создаётся палитра цветов для right\_title и происходит окончательная конфигурация виджета. Следующий фрагмент кода функции:



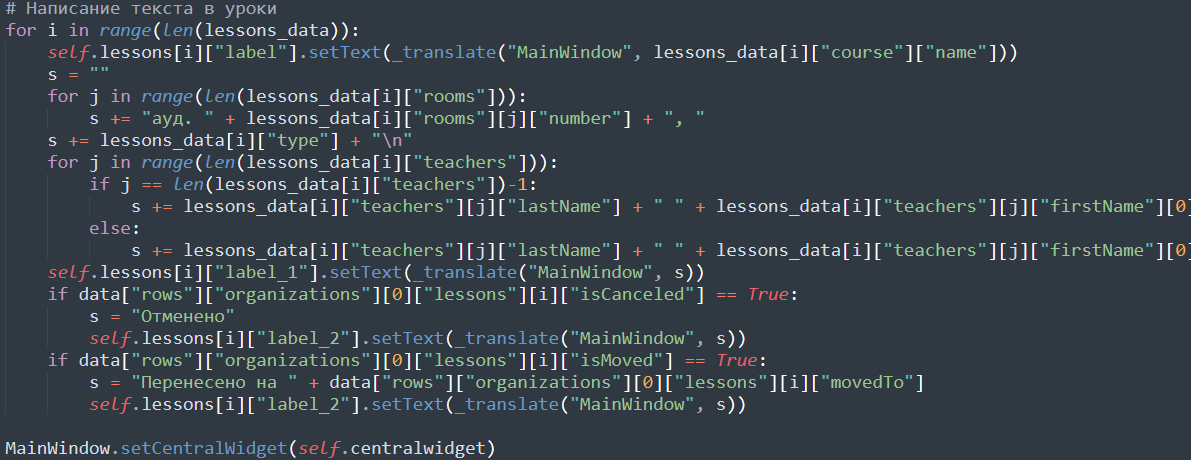
Создаётся gridLayout (объект QGridLayout), каркас будущего расписания. В первую ячейку добавляется empty\_header\_frame (объект QFrame), не несущий особой функциональной составляющей. Затем в цикле для всех временных промежутков создаются заголовки строк через функцию row\_header\_func. Происходит изменение размеров окна и scrollAreaWidgetContents для того, чтобы строки в таблице расписания отображались корректно (не обрезались и не выходили за границы окна). После этого также через цикл по дням недели создаются заголовки столбцов функцией column\_header\_func. Другой фрагмент кода функции:



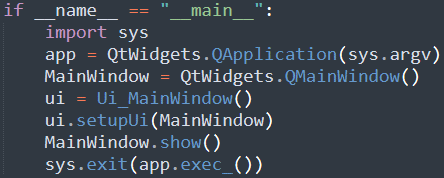
Сначала создаётся список lessons\_temp, который будет содержать временную информацию о мероприятии, такую как тип мероприятия, день недели, временной промежуток, название дисциплины, id учебной группы. Переменная c отвечает за порядковый номер hlayout. В цикле по всем мероприятиям недели проверяется условие: если мероприятие уже есть в временном списке и его нет в списке повторяющихся мероприятий double\_lessons\_list, то в ячейку функцией hlayout\_func помещается hlayout, и в список повторяющихся мероприятий добавляется это мероприятие. Иначе –мероприятие добавляется во временный список. В конце работы цикла временный список очищается. В следующем цикле по всем мероприятиям недели в ячейки добавляются мероприятия функцией lessons\_func. Затем происходит распределение некоторых виджетов в родительские виджеты для корректного отображения. Добавляется ещё один элемент – spacerItem2, «раздвигающий» gridLayout от нижнего края окна. Далее вызывается функция retranslateUi:



Для корректного отображения русского текста используется метод translate. В этой функции в текстовые виджеты добавляется текст. Другой фрагмент кода функции:

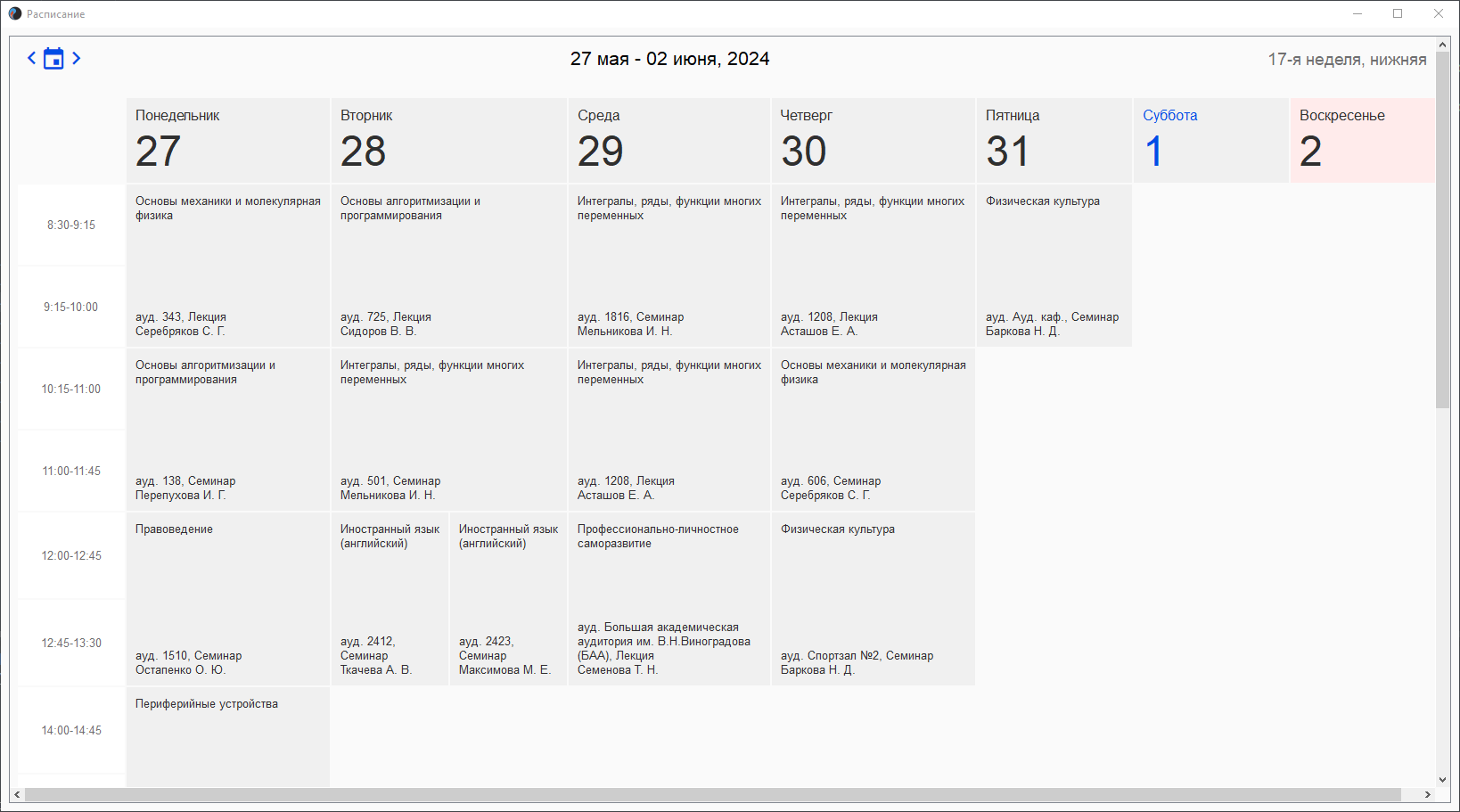


Здесь для добавления текста в виджеты используется временная переменная s, наполняемая нужным текстом. После добавления текста обновляется centralwidget. Вне класса располагается такая конструкция:



Создаётся элемент объекта QApplication (на вход подаются системные параметры, получаемые модулем sys), отвечающий за запуск приложения, содержащего элемент объекта QMainWindow, который создаётся далее. В переменную ui добавляется экземпляр класса Ui\_MainWindow, запускается функция setupUi. Показывается окно и для корректного закрытия программы добавляется последняя строчка.

Скриншот готового интерфейса программы:



Отдельно стоит отметить, что отображается расписание только для учебной группы АС-23-04. Это обусловлено тем, что для доступа к расписанию конкретной группы, необходимо знать такой параметр, как groupId, однако алгоритма, по которому groupId присваивается конкретной группе, мной найдено не было. Существует возможность парсинга groupId брутфорсом (перебором), в дальнейших версиях программы будет использован текстовый файл, содержащий пары «название группы, groupId», полученный перебором номеров групп от 1 до 15000.

# **Заключение**

Созданная программа предоставляет пользователям актуальное расписание занятий, используя запросы к серверу и обработку полученных данных. Интерфейс программы включает элементы для удобного просмотра и навигации по расписанию, такие как область прокрутки и кнопки для переключения между неделями.

Основной вклад данной курсовой работы заключается в создании эффективного и удобного инструмента для студентов, который значительно упрощает процесс получения информации о расписании занятий. Программа успешно решает поставленные задачи и может быть использована в реальных условиях учебного процесса. В дальнейшем проект может быть расширен для поддержки расписаний других учебных групп, что сделает его еще более полезным и универсальным.

# **Список использованных литературных источников**

1. <https://www.cyberforum.ru/>
2. <https://doc.qt.io/qtforpython-6/PySide6/>
3. <https://python-scripts.com/>
4. <https://pythonru.com/primery/kak-ispolzovat-modul-datetime-v-python>
5. <https://ru.stackoverflow.com/>
6. <https://www.geeksforgeeks.org/>
7. <https://codebeautify.org/json-decode-online>
8. <https://tproger.ru/>
9. Прохоренок Н. А. Python 3 и PyQt 6. Разработка приложений. - СПб.: Б:ХВ-Петербург, 2023. — 832 с.