Практическое задание №7.

Добавление генератора диаграмм в приложение QTabelo.

Цель работы: научиться использовать QGraphicsScene, QMap, а также классы для работы с 2D графикой в Qt и добавить в приложение QTabelo преобразование выделенных данных таблицы в диаграмму.

Задание:

- Реализовать визуализацию данных без копирования в Excel.
- Построить интерактивные диаграммы внутри QTabelo.
- Сохранять диаграммы как изображения.

Порядок выполнения:

1) Создайте новый класс, например ChartBuilder, для создания столбчатых диаграмм на основе выделенных данных из таблицы.
   - Визуализация данных должна происходить с использованием графической сцены, поэтому класс ChartBuilder должен быть унаследован от QGraphicsScene. В конструктор необходимо передавать таблицу, из которой берутся данные:
   explicit ChartBuilder(QTableWidget *sourceTable, QObject *parent = nullptr);
   - Добавьте в класс метод для построения диаграммы, в который будут передаваться выделенные ячейки:
   void buildBarChart(const QModelIndexList &selectedCells);
   - Добавьте в класс ChartBuilder метод для сохранения диаграммы в PNG файл, например: void exportToPng(const QString &path);
   - Приватные поля в классе ChartBuilder – это указатель на исходную таблицу с данными (например, QTableWidget *m_sourceTable) и контейнер QMap<QString, double> , например, m_data для хранения данных диаграммы.
2) Реализуйте логику диаграмм в методе buildBarChart(), следуя рекомендациям:

- Сначала метод удаляет все предыдущие элементы с графической сцены с помощью clear().

- Для сбора данных метод проходит по переданному ему списку выделенных ячеек, обрабатывая их парами (четные индексы - подписи, нечетные - значения). Данные сохраняются в контейнер QMap<QString, double>, определенным в классе ChartBuilder.
- Для подготовки к визуализации устанавливаются постоянные параметры: ширина столбцов и максимальная высота диаграммы.
- Находится максимальное значение в данных для пропорционального отображения всех столбцов.
- Отрисовка диаграммы происходит следующим образом. Для каждой пары «подпись-значение» создается столбец (QGraphicsRectItem) и заливается цветом. Затем добавляется текстовая подпись (QGraphicsTextItem) с названием и значением. И далее элементы добавляются на сцену методом addItem().
- Предусмотреть, что каждый новый столбец рисуется с отступом от предыдущего.
- Итоговый метод должен автоматически масштабировать диаграмму, обеспечивая наглядное представление данных независимо от их абсолютных значений.

3) Добавьте в контекстное меню таблицы вызов мастера диаграмм. Для этого в методе TableDocument::showTableContextMenu() необходимо добавить новое действие QAction и подключить его к имеющемуся меню. Обработка вызова данного пункта может быть реализована так:

```
    // Создаем и отображаем диаграмму
    auto *chart = new ChartBuilder(table);
    chart->buildBarChart(table->selectionModel()->selectedIndexes());

    QGraphicsView *chartView = new QGraphicsView(chart);
    int tabIndex = m_tabs->addTab(chartView, tr("Диаграмма"));
    m_tabs->setCurrentIndex(tabIndex);
}

emit contentChanged(true);
```

Рисунок 36. Действия для создания и построения диаграммы

4) Добавьте контекстное меню к диаграммам. Для этого можно переопределить метод contextMenuEvent() в классе ChartBuilder (так как он наследуется от GraphicsScene) и реализовать в нем действие для вызова слота сохранения диаграммы. Метод contextMenuEvent() будет обрабатывать клик правой кнопкой мыши по сцене и показывать контекстное меню с опцией экспорта в PNG.
5) Протестируйте работу приложения, следуя плану:

   - Проверить поведение при передаче пустого QModelIndexList (не должно быть ошибок, сцена должна остаться пустой).
   - Проверить обработку, когда количество выделенных ячеек нечетное (последняя ячейка должна игнорироваться).
   - Проверить реакцию на текст вместо чисел в ячейках значений, пустые ячейки, специальные символы в подписях.
   - Проверить корректное отображение столбцов: соответствие высоты столбцов переданным значениям и правильное позиционирование столбцов.
   - Подписи данных должны корректно отображать информацию и располагаться под столбцами.
   - Проверить, что повторный вызов с новыми данными полностью перерисовывает диаграмму.

Дополнительные задания:

1. Реализовать функцию обновления диаграммы при изменении данных в таблице.
2. Реализовать элементы управления для настройки диаграммы.

Контрольные вопросы:

1. Как связаны ChartBuilder и QGraphicsView?
2. Как сохранить диаграмму в изображение? Какие методы для этого используются?
3. Как обрабатывается контекстное меню (contextMenuEvent) на сцене?
4. Почему для хранения данных используется QMap<QString, double>?

5. Что происходит при многократном вызове buildBarChart с разными данными?
6. Как очищается сцена перед построением новой диаграммы?
7. Какие классы Qt используются для создания столбцов и подписей?
8. Как рассчитывается высота столбцов диаграммы?
9. Как обрабатываются очень длинные подписи (перенос, обрезание)?
10. Как позиционируются столбцы и подписи на сцене?