Лабы

Главная » СПО

СПО. Лабораторная работа №4

Разработка кроссплатформенных приложений

▼ Оглавление

Задание

Написать программу-обёртку для базы данных SQLite. Программа должна обладать следующими функциями:

1. Отображение данных таблиц и запросов
2. Редактирование данных

3. Добавление данных
4. Удаление данных

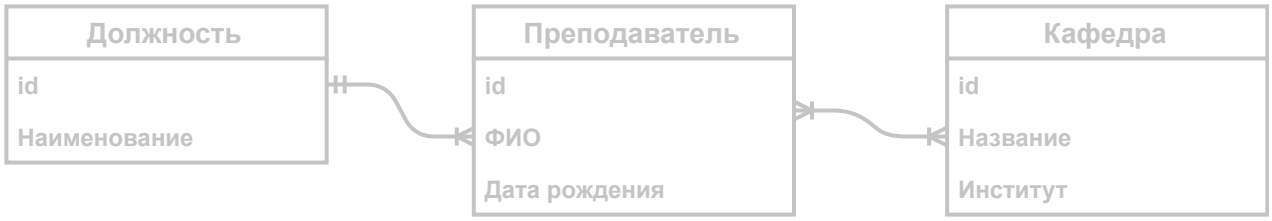
Необходимо создать два варианта программы: с использованием GTK+ и Qt.

Программы должны компилироваться на Linux и Windows **без изменения исходного кода**. В данной работе вам придётся устанавливать и настраивать большое количество инструментов разработки, поэтому рекомендую использовать виртуальные машины, чтобы не забивать свою основную систему.

Варианты БД

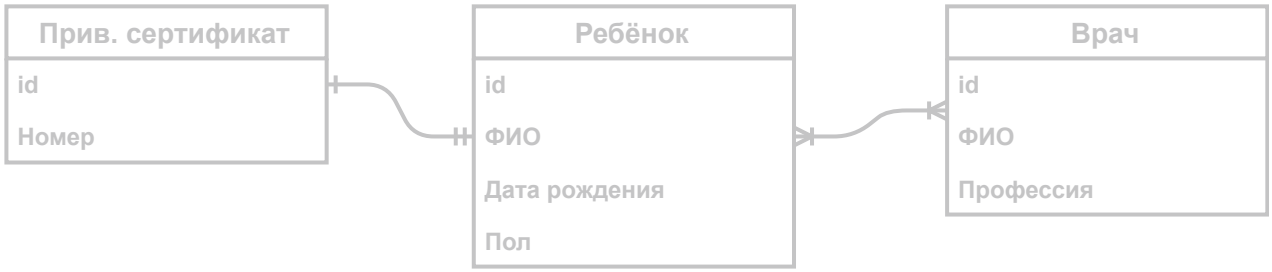
1. **Книги**

2. **Преподаватели кафедр**

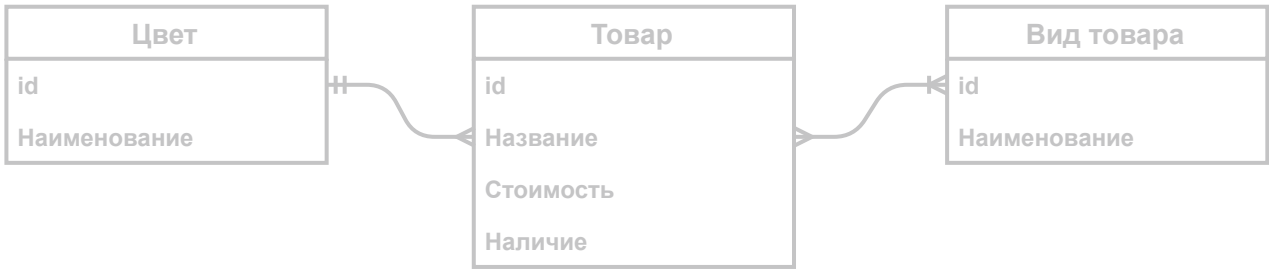


3. **Животные**

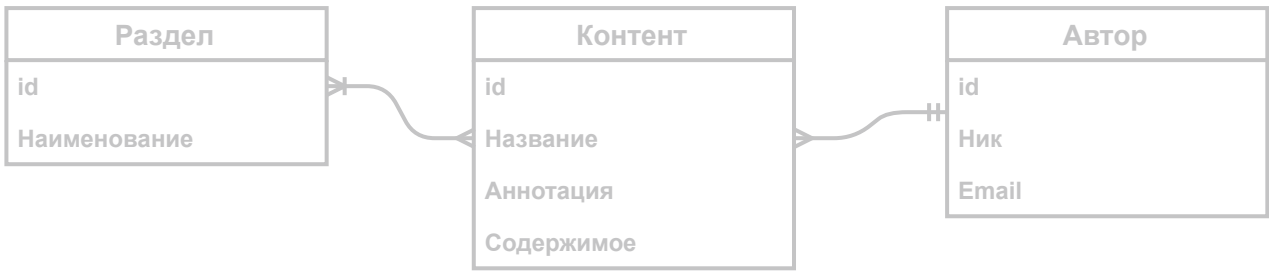
4. **Дети и врачи**



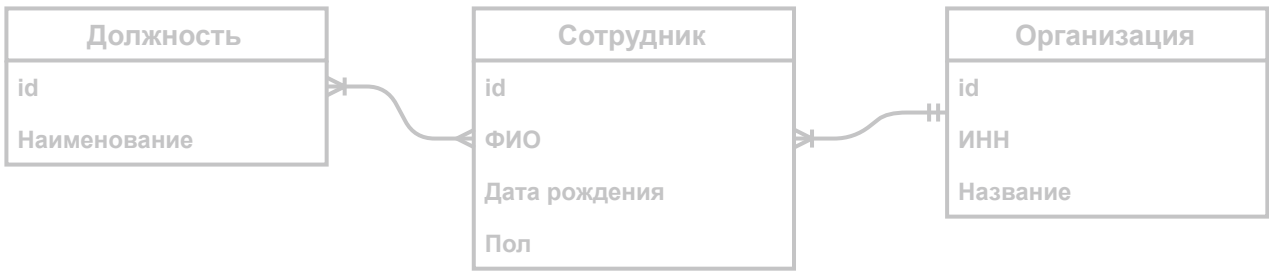
5. **Товары**



6. **Контент авторов**

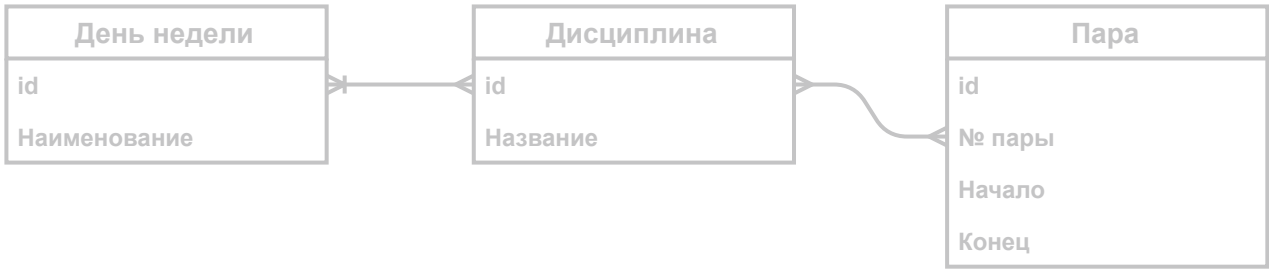


7. **Сотрудники фирм**



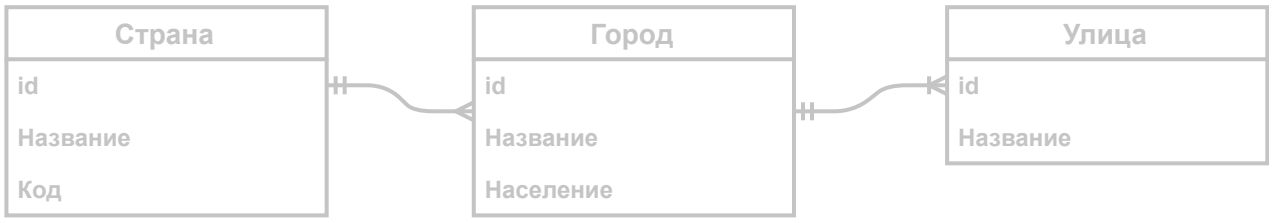
8. **Преподаватели и дисциплины**

9. **Расписание**



10. **Кафедры и дисциплины**

11. **Адреса**

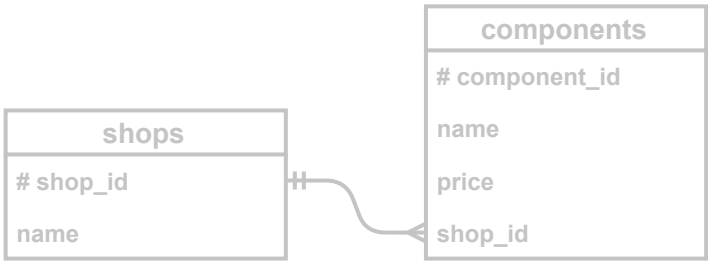


12. **Автомобили и владельцы**

Пример

База данных

Создадим и заполним простую базу данных:



Для этого установите sqlite3, в терминале наберите `sqlite3 mydb.db` и выполните следующие запросы:

```
CREATE TABLE shops (shop_id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT N
```

```
INSERT INTO shops (name) VALUES ("PNS"), ("Второй");
```

```
SELECT * FROM shops;
```

```
CREATE TABLE components (
    component_id   INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    price REAL NOT NULL,
    shop_id INTEGER NOT NULL,
    FOREIGN KEY (shop_id)
       REFERENCES shops (shop_id)
       ON UPDATE CASCADE
       ON DELETE CASCADE
);
```

```
INSERT INTO components (name, price, shop_id)
VALUES ("AMD Ryzen 7 3700X", 25000, 1),
       ("Nvidia RTX 2060 Super", 30000, 2),
       ("Intel Core i7 9700KF", 30000, 2),
       ("AMD RX 5700 XT", 38000, 1);
```

```
SELECT
    components.component_id,
    components.name,
    price,
    shops.shop_id,
    shops.name
FROM
    components
INNER JOIN shops ON components.shop_id = shops.shop_id;
```

Теперь можно выйти из консольного интерфейса sqlite3 командой `.exit` .

GTK+3. Сборка в Linux

Файлы исходного кода находятся тут. Установите необходимые пакеты.

Необходимые пакеты

Debian-based

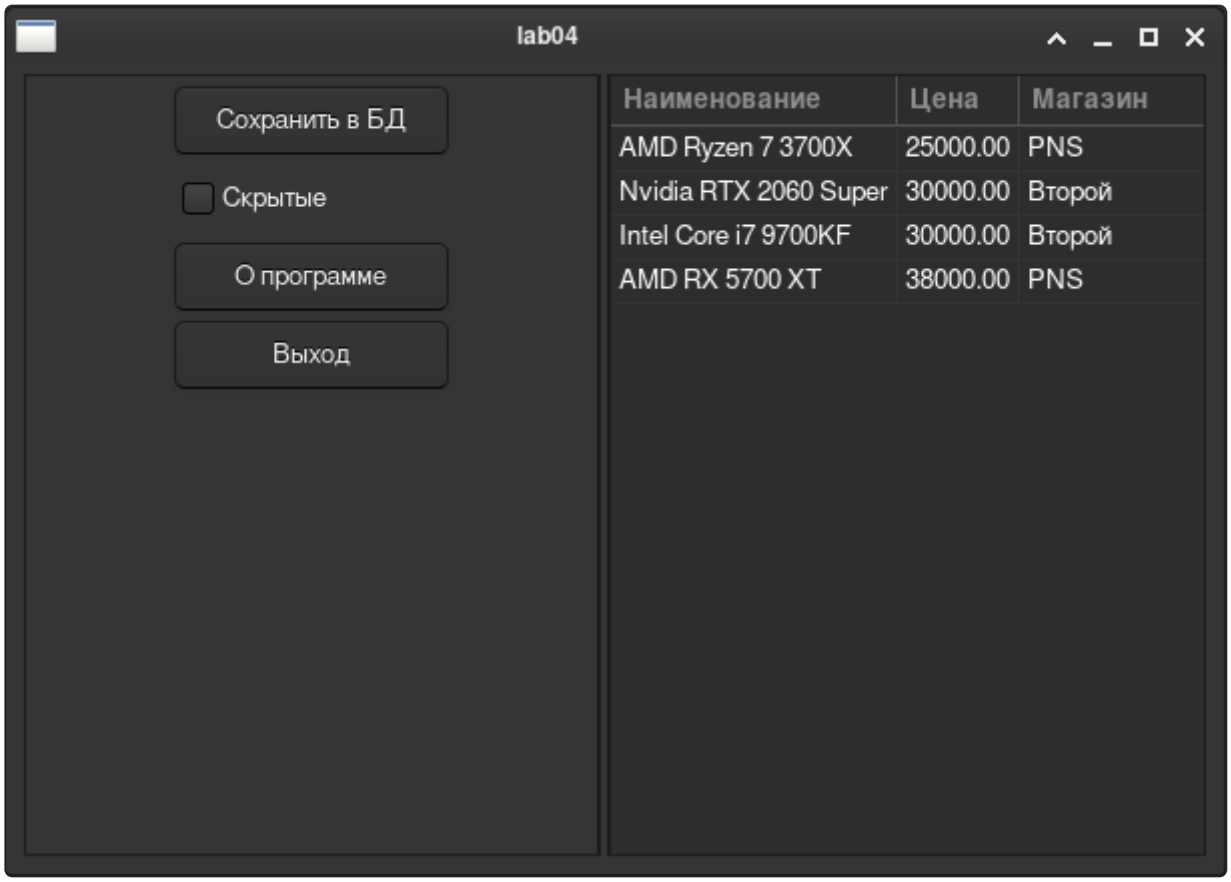
- sqlite3
- libsqlite3-dev
- gcc
- make
- libgtk-3-0
- libgtk-3-dev
- glade

Arch-based

- sqlite
- pkg-config
- gcc
- make
- gtk3

- glade

Проанализируйте исходный код и прилагающийся `makefile`, это поможет в написании собственного варианта. Скомпилируйте программу командой `make`. Примерно так будет выглядеть получившееся приложение:



> GUI для этой программы разрабатывался с помощью Glade. Откройте файл `lab04.glade` и проанализируйте структуру интерфейса. Чтобы освоить разработку в Glade, вам пригодится:
>
> - статья на хабре

> Для разработки с GTK+ могут быть полезными:
>
> - GTK Tutorials
> - Tim-Philipp Müller. GTK+ 2.0 Tree View Tutorial
> - Andrew Krause. Foundations of GTK+ Development

GTK+3. Сборка в Windows

1. Скачайте, установите и обновите MSYS2, следуя инструкциям на сайте.

2. В консоли MSYS2 установите необходимые для сборки пакеты:

```
pacman -Sy mingw-w64-x86_64-gtk3 mingw-w64-x86_64-toolchain base-devel sql
```

3. После установки закройте консоль MSYS2 MSYS и откройте MSYS2 MINGW64. Нужно удостовериться, что в переменной окружения PKG_CONFIG_PATH есть пути `/mingw64/lib/pkgconfig` и `/mingw64/share/pkgconfig` :

```
echo $PKG_CONFIG_PATH
```

4. Создайте каталог для своего проекта и перейдите в него:

```
mkdir ~/lab04
cd ~/lab04
```

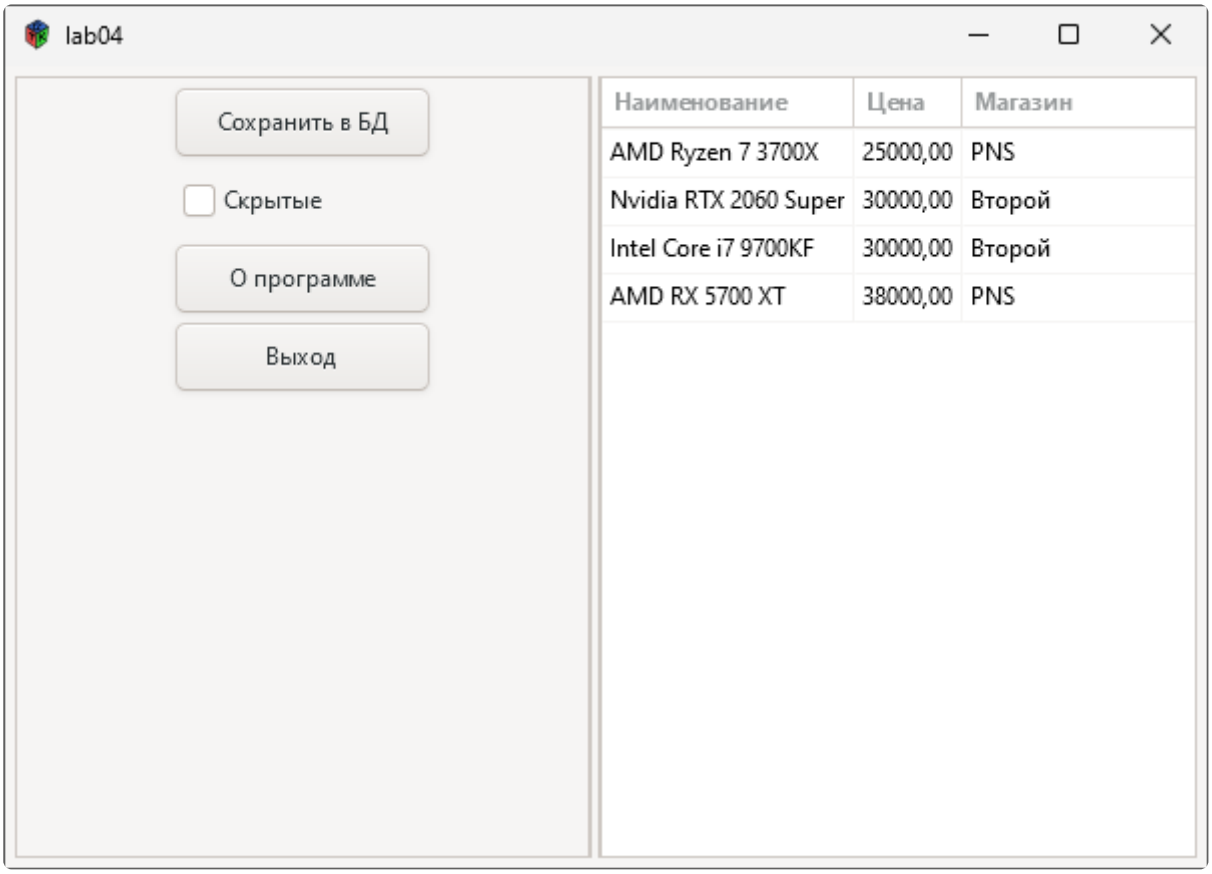
Для Windows этот каталог имеет путь `C:\msys64\home\%USERNAME%\lab04` , если вы следовали иснтрукции в п.1.

5. В созданный каталог положите файлы исходного кода, файл .glade, makefile и файл базы данных.
6. Скопируйте в этот каталог ресурсы, нужные для работы собираемой программы:

```
cp /mingw64/bin/*dll .
mkdir -p ./lib/gdk-pixbuf-2.0
cp -r /mingw64/lib/gdk-pixbuf-2.0/* ./lib/gdk-pixbuf-2.0/
mkdir -p ./share/glib-2.0/schemas
cp -r /mingw64/share/glib-2.0/schemas/* ./share/glib-2.0/schemas
mkdir -p ./share/icons
cp -r /mingw64/share/icons/* ./share/icons
```

7. Наконец-то можно собрать программу командой `make` и запустить её. Чтобы удостовериться, что программа запускается и работает в Windows правильно, её следует запустить просто даблкликом `lab04.exe` , т.е. не из консоли MSYS2 -

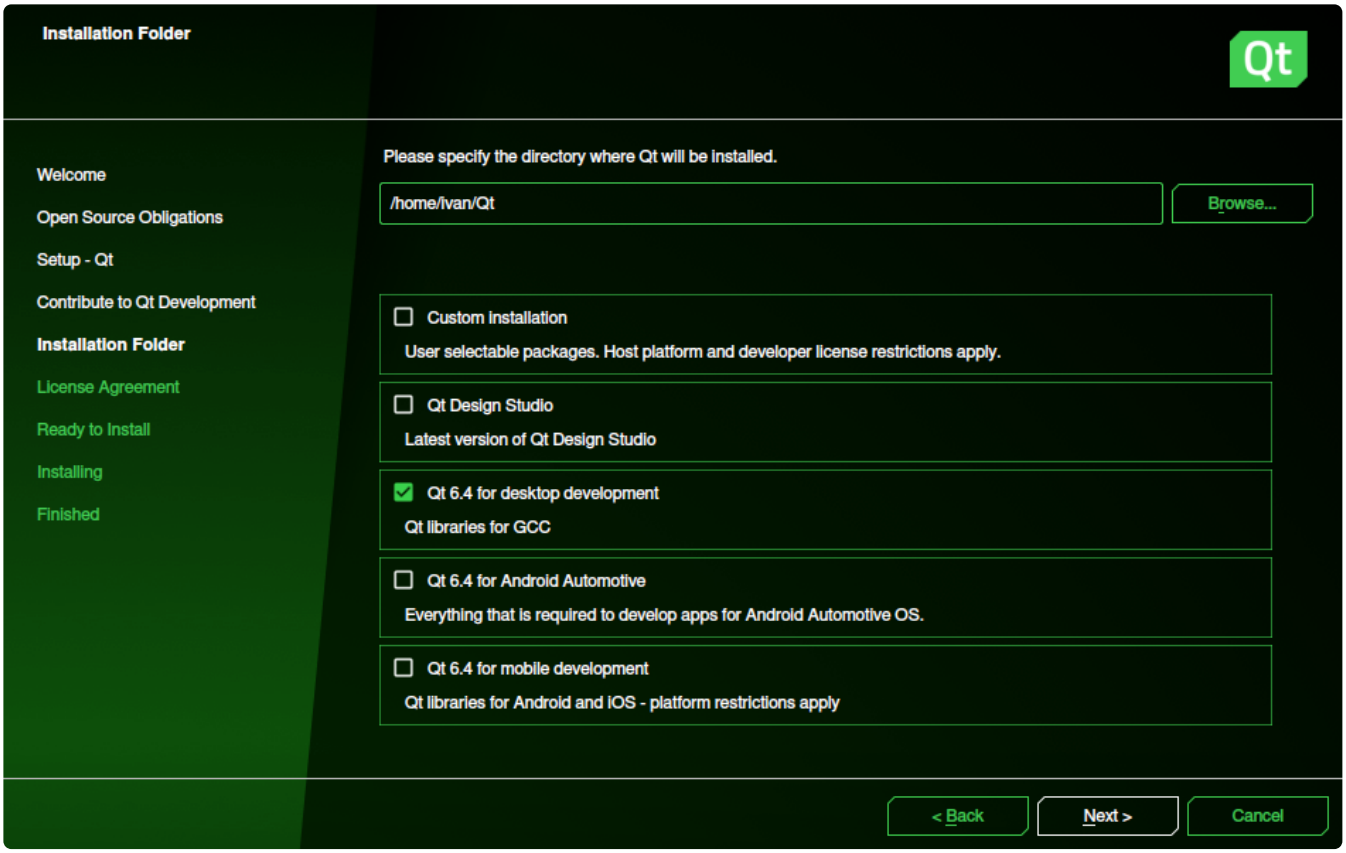
это нужно для того, чтобы программа не видела переменные окружения и настройки MSYS, как будто мы запускаем приложение в роли обычного пользователя. Если всё прошло успешно, вы увидите нечто подобное:



> **Примечание:** Конечно, на данном этапе папка с программой занимает ~130 Мб, но для простоты мы скопировали в неё все иконки и dll файлы, хотя конечно же, она использует только некоторые из них. Вы можете самостоятельно подобрать минимально необходимый набор файлов для поставки вашей программы.

Qt. Сборка в Linux

Скачайте и установите Qt с помощью онлайн-установщика `qt-unified-linux-x64-online.run` отсюда. Для загрузки и установки понадобится VPN. Если у вас нет аккаунта Qt, его можно будет создать прямо в установщике, но скорее всего потребуется подтвердить email. Если установщик будет жаловаться на отсутствие свободного места для временных файлов, нужно увеличить размер tmpfs, выделенный для директории `/tmp` .



Установите необходимые пакеты.

Необходимые пакеты

Debian-based

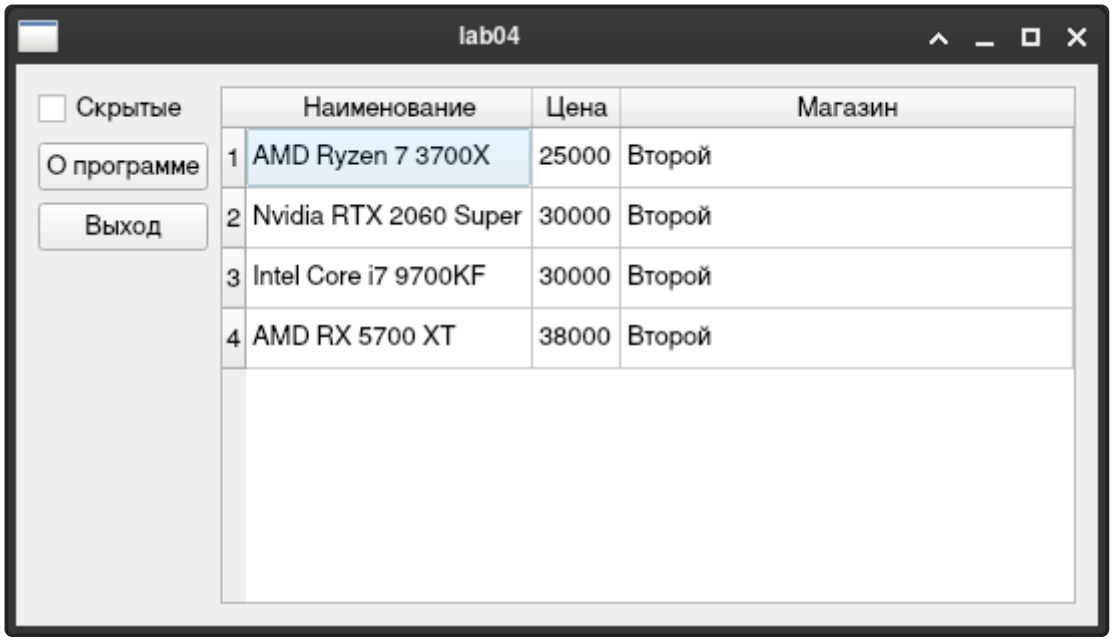
- sqlite3
- g++
- make
- libgl1-mesa-dev

Arch-based

- sqlite
- gcc
- make
- mesa

Исходники находятся тут. Откройте Qt Creator и добавьте проект. Нужно будет добавить комплект сборки для вашей системы, скорее всего это будет Desktop. В этом варианте программа написана уже на C++, что позволяет использовать объектно-ориентированный подход и готовые высокоуровневые компоненты.

Соберите проект. Перед запуском не забудьте добавить файл базы данных в папку `build...` , где появится исполняемый файл. Получится что-то такое:

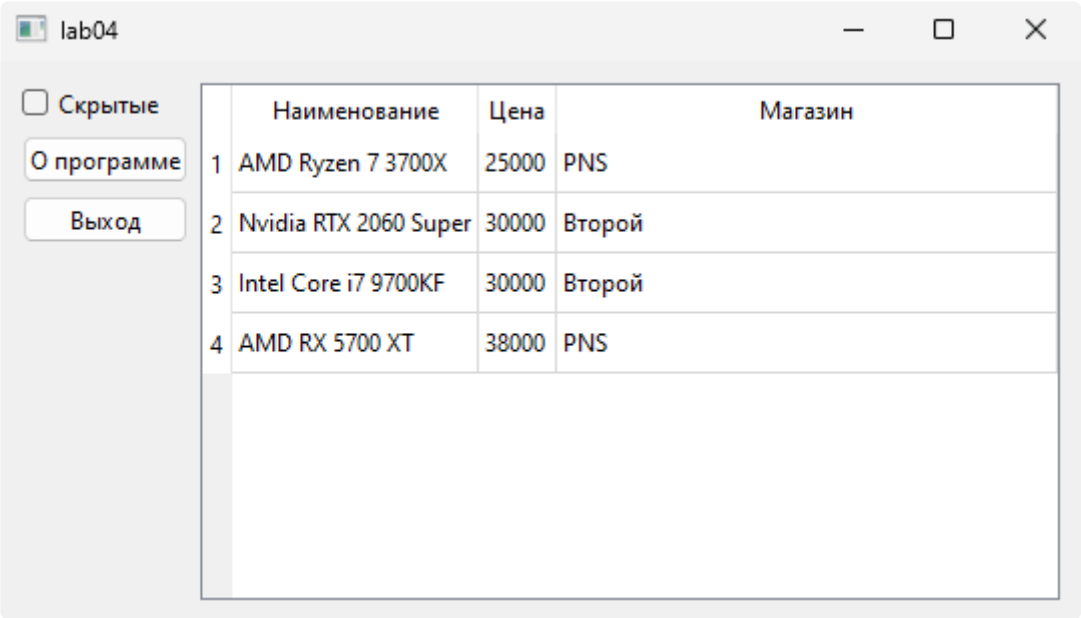


> Для разработки с Qt могут быть полезными:
>
> - Программа Qt Assistant для просмотра справочной информации
> - Qt Documentation
> - Уроки с примерами по Qt

Qt. Сборка в Windows

В Windows также нужно установить Qt. Скачайте онлайн-установщик `qt-unified-windows-x86-online.exe` отсюда. Удостоверьтесь, что вместе с устанавливаемой версией Qt будет установлен MinGW, иначе нельзя будет собрать проект.

После установки откройте Qt Creator, добавьте и настройте проект, и снова не забудьте про файл БД. Всё, можно запускать!



Слишком просто для Windows, правда? Действительно, если теперь попытаться запустить собранное приложение не из Qt Creator, а просто двойным щелчком по .exe файлу, то оно не заработает. Чтобы это исправить, откройте командную строку Qt для MinGW соответствующей разрядности из меню Пуск и выполните следующее:

```
cd bin
windeployqt <путь>
```

Где вместо `<путь>` укажите папку с исполняемым файлом. Например, у меня эта команда выглядела так: `windeployqt C:\Users\ivan\Desktop\build-lab04-Desktop_Qt_6_4_2_MinGW_64_bit-Release\release` . Конечно, для этого лучше использовать релизную конфигурацию - это можно сделать, указав “Release” или “Выпуск” в Qt Creator по клику на значок монитора рядом с кнопкой запуска. `Windeployqt` поместит все необходимые для работы приложения файлы в указанную директорию, и его можно будет запускать отдельно.

Поздравляю, теперь вы можете разрабатывать кроссплатформенные десктопные приложения!

← ПРЕДЫДУЩАЯ
СПО. Лабораторная работа №3

СЛЕДУЮЩАЯ →
СПО. Лабораторная работа №5