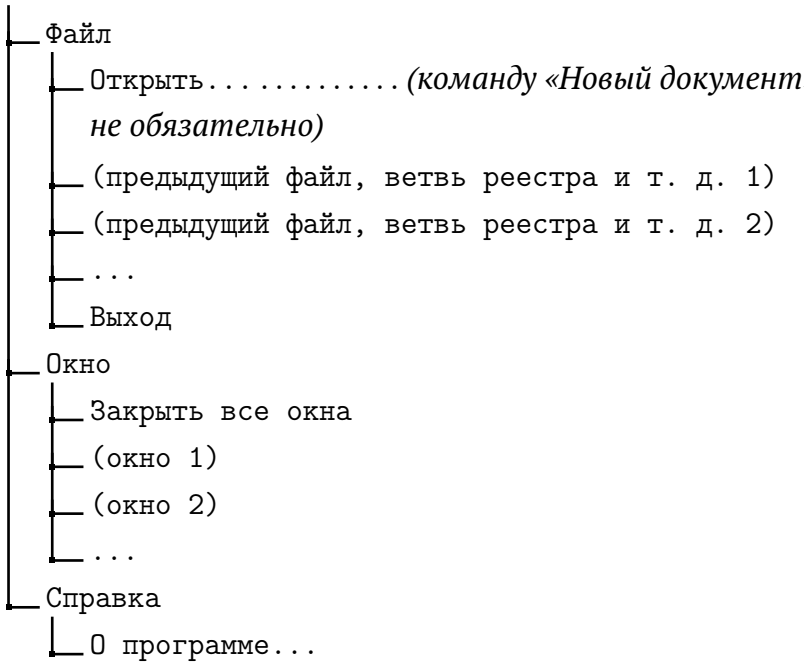
Индивидуальные задания по модулю «Парадигма „модель — представление — контроллер“ в Qt»

Разработка многоплатформенного ПО

- В этом задании решение должно использовать классы Qt для поддержки идиомы «модель — представление — контроллер» для доступа к указанным данным.
- Необходимо реализовать многооконный пользовательский интерфейс программы (MDI).
- Для реализации пользовательского интерфейса (главное окно, меню и т. д.) необходимо использовать визуальный редактор (Qt Designer).
- Необходимо реализовать систему локализации элементов пользовательского интерфейса приложения при помощи инструмента Qt Linguist и соответствующих средств библиотек Qt.
- Для редактирования полей перечислимого типа (в некоторых заданиях) необходимо реализовать делегат, создающий выпадающие списки (`QComboBox`).
- Для описания проекта необходимо использовать систему CMake.
- В проекте должна быть реализована цель для генерирования файлов перевода (исходных + двоичных или только двоичных — в зависимости от настроек).
- В проекте необходимо реализовать цель установки программы.
- В программе должно быть реализовано меню:



1. Редактор реестра Windows. В меню программы необходимо предусмотреть возможность открытия заданного раздела реестра. Необходимо реализовать возможность редактирования как минимум названий разделов и значений, а также значений целого (`REG_DWORD`) и строкового (`REG_SZ`) типов.

2. Редактор таблицы в формате CSV. Первая строка таблицы задаёт типы колонок данных: строка, целый тип, логическое значение, перечислимый тип (значения перечислимого типа могут быть предопределены в программе).

3. Редактор таблицы записей, хранящейся в файле. Одновременно в памяти необходимо хранить не более фиксированного количества записей из файла. Структура записей может иметь фиксированный формат и содержать, как минимум, записи строкового, целого, логического и перечислимого типов.

4. Редактор дерева записей, хранящегося в памяти. Структура записей может иметь фиксированный формат и содержать, как минимум, записи строкового, целого, логического и перечислимого типов (значения перечислимого типа могут быть предопределены в программе).

5. Редактор дерева записей, хранящегося в реляционной таблице SQLite, работа с которой ведётся средствами библиотеки Qt. Структура записей может иметь фиксированный формат и содержать, как минимум, записи строкового, целого, логического и перечислимого типов (значения перечислимого типа могут быть предопределены в программе).

6. Редактор XML-файла, представленного в виде дерева узлов и атрибутов. Необходимо использовать классы Qt `QDomDocument`, `QDomNode` и т. д. Необходима поддержка, как минимум, узлов типов `DocumentTypeNode`, `ElementNode` и `AttributeNode`, целых и строковых атрибутов. В модели атрибуты должны быть дочерними по отношению к их элементам.

7. Редактор XML-файла, представленного в виде дерева узлов и атрибутов. Необходимо использовать классы Qt `QXmlStreamReader` и `QXmlStreamWriter`. Необходима поддержка, как минимум, лексем типов `StartDocument`/`EndDocument` и `StartElement`/`EndElement`. В модели атрибуты должны быть дочерними по отношению к их элементам.

8. Редактор таблицы СУБД CUBRID: `http://www.cubrid.org/manual/92/en/api/cci.html`.

9. Редактор таблицы СУБД MonetDB: `https://www.monetdb.org/Documentation/Manuals/SQLreference/Programming/MAPI`.