САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

244 группа

Малютин Данила Павлович

TRIKMobile: Мобильная среда программирования робота на платформе TRIK

Курсовая работа

Научный руководитель:
ст. преподаватель Литвинов Ю. В.

Санкт-Петербург
2015

Оглавление

Введение

TRIK Studio — среда обучения основам программирования и кибернетики. Среда позволяет создавать графические программы для роботов Lego® Mindstorms® NXT 2.0, Lego® EV3, TRIK и исполнять эти программы прямо на компьютере, посылая команды роботу через Bluetooth или USB-интерфейс, а также генерировать по диаграммам код на различных языках программирования и закачивать его для исполнения в робота.

TRIK Studio доступна для установки на настольные ОС Windows и Linux и рассчитана на управление с помощью клавиатуры и мыши. Однако, количество пользователей мобильных устройств (смартфонов, планшетов) уже давно превысило количество пользователей ПК и продолжает расти [1]. Эти люди тоже хотели бы пользоваться конструктором ТРИК, но у них попросту может не быть полноценного доступа к компьютеру. Например, у школьников младших классов редко когда есть свой ноутбук, но очень часто уже есть смартфон. Мобильное приложение позволило бы расширить аудиторию ТРИК, снизить порог вхождения, а также его можно было бы всегда иметь при себе.

TRIKMobile является попыткой перенести часть функциональности TRIK Studio на мобильную платформу, а именно возможность создавать и редактировать модели программ с последующей их отсылкой на робота для исполнения.

1. Постановка задачи

- Реализовать приложение, позволяющее создавать программы для роботов, сопоставимые с программами, созданными в TRIK Studio.
- Приложение должно иметь понятный пользователю интерфейс.
- Представление программ должно быть адаптировано под небольшие экраны.
- Приложение должно быть кроссплатформенным.

2. Обзор существующих решений

2.1. TRIK Gamepad

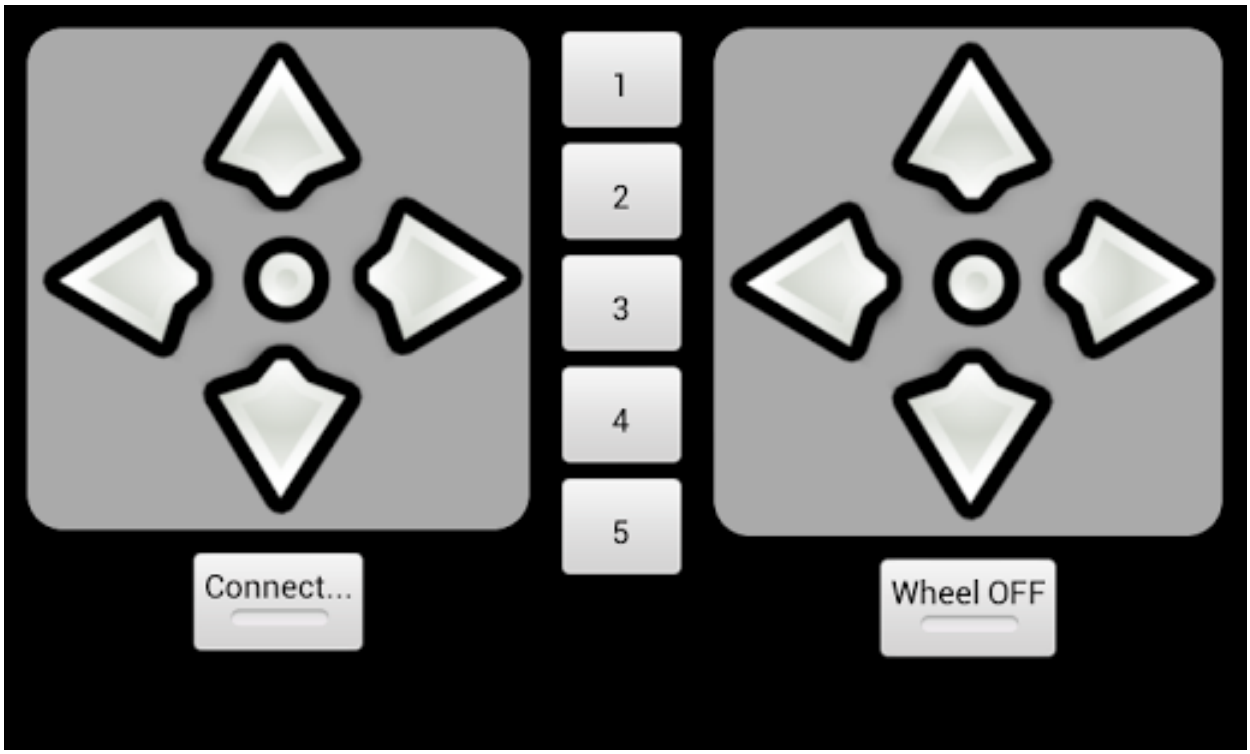

Рис. 1: Интерфейс TRIK Gamepad

На данный момент существует единственное приложение TRIK под ОС Android — это TRIK Gamepad[1].

Интерфейс этого приложения представлен на рис. 1. У него есть две активные зоны и пять функциональных кнопок, кроме того, оно позволяет использовать акселерометр телефона. Однако, данное приложение работает только в качестве пульта для управления движением робота, то есть не решает поставленной задачи, а именно не позволяет составлять свои собственные программы для отправки и исполнения на роботе.

[1] `https://play.google.com/store/apps/details?id=com.trik.gamepad`

3. Реализация

3.1. Общее описание и инструменты

Для реализации в целях достижения кроссплатформенности была выбрана библиотека Qt5[2]. Основная логика приложения была написана на C++, а для описания интерфейса использовался Qml/QtQuick 2.

Для удобства просмотра и редактирования программы для робота на мобильных устройствах она представляется в виде списка идущих друг за другом блоков, некоторые из которых могут раскрываться в другие списки по принципу папок в проводнике.

3.2. Особенности реализации

3.2.1. Общая структура

Основными компонентами являются QmlSignalHandler, Connector, ConnectorHandler, ScriptGenerator, BlockModel и BlockFactory (их можно видеть на рис. 2).

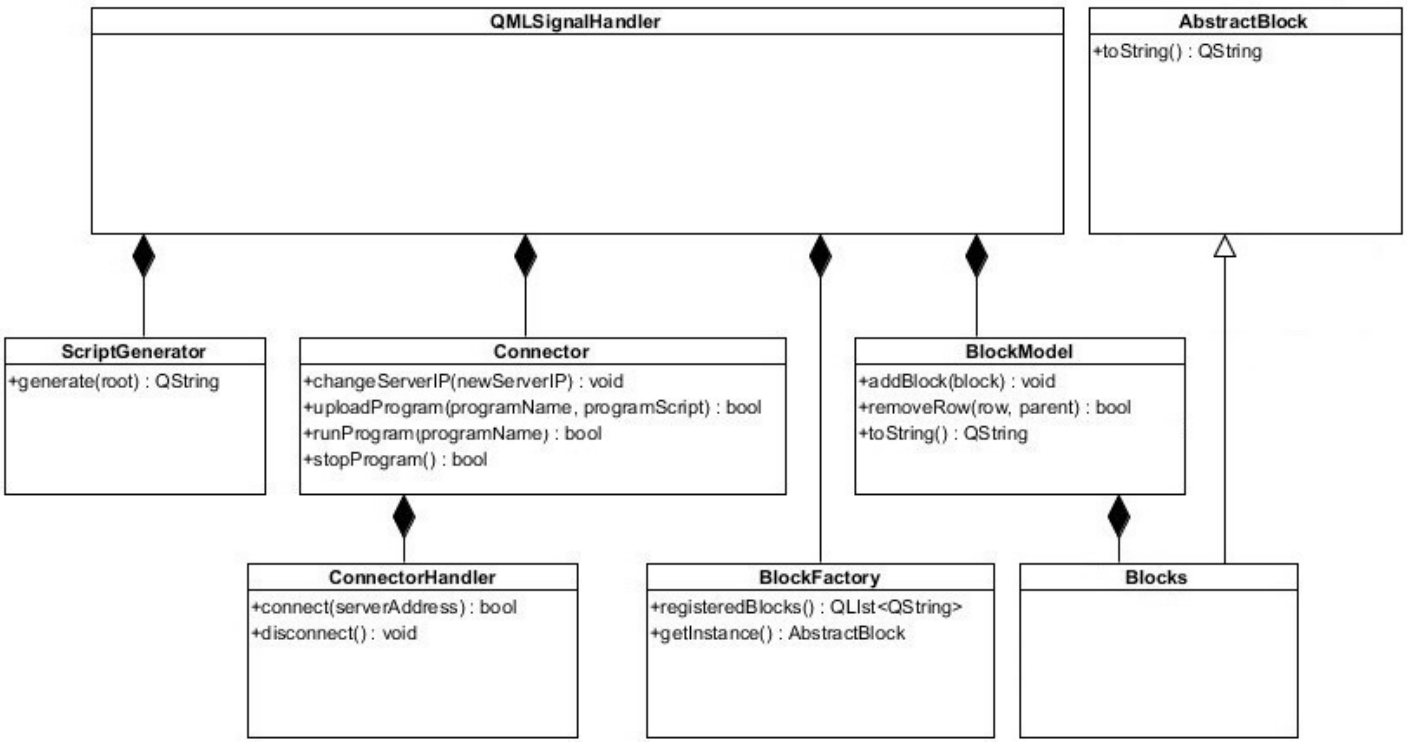

Рис. 2: UML-диаграмма общей структуры приложения

- QmlSignalHandler отвечает за общее состояние программы и за взаимодействие с интерфейсом на qml, а также за сохранение и считывание модели в/из файла.

[2] http://www.qt.io/

- Connector и ConnectorHandler служат для установки и поддержания связи с роботом.
- ScriptGenerator контролирует генерацию qts скриптов по модели, для отправки их на робота.

3.2.2. Модель

В силу отсутствия подходящих моделей/представлений в QtQuick была написана своя модель, которая отображала древовидную структуру программы в плоский список, который может быть отображён с помощью ListView. Модель расширяет QAbstractListModel и хранит в себе список "абстрактных" блоков (базового класса для всех функциональных блоков программы). Также модель реализует возможности по удалению, добавлению и изменению своих элементов. Все блоки являются потомками базового класса AbstractBlock. Он определяет их интерфейс: каждый блок имеет таблицу propertyMap, список моделей (в случае, если он может иметь потомков как, например, for) и строковое представление типа блока (его уникальное название).

3.2.3. Фабрика

Фабрика — один из важнейших классов, позволяющий создавать новые блоки, зная лишь их имя. Благодаря этому классу нам не только становится проще управлять созданием новых блоков из qml интерфейса, но и становится относительно легко реализовать загрузку сохранённой модели. Из-за ограничений C++, таких как отсутствие рефлексии и невозможность передавать указатель на конструктор, было решено хранить в таблице соответствие "идентификатор блока — указатель на функцию, возвращающую экземпляр этого блока". Эти функции, а также метод, позволяющий регистрировать в фабрике блоки под определёнными именами, были реализованны с помощью шаблонов C++.

3.2.4. Сохранение и загрузка модели

Для хранения модели был выбран формат JSON, так как он не только хорошо поддерживается Qt 5, но и является легко читаемым, что полезно для отладки. Сохранение модели происходит следующим образом:

1. Сначала создаётся корневой JSON объект, который содержит основную информацию о модели. На данный момент в нём хранится сама модель и версия "формата", для сохранения обратной совместимости на случай изменения способа хранения данных.
2. Затем модель преобразуется в JSON Array — массив своих элементов (блоков) тоже преобразованных в JSON.

3. Блоки преобразуются в JSON Object с тремя полями: type (имя блока), propertyMap (таблица свойств) и children (потомки-модели, вложенные в блок, преобразованные в JSON). Притом конвертация очень простая, так как все поля блока уже имеют тип, поддерживаемый JSON-ом, а именно строчка либо QVariantMap.

4. Наконец, после того как получено финальное представление модели в виде JSON, происходит запись на диск с помощью QJsonDocument и QFile.

Результат можно видеть на рис. 3.

```
{
    "model": [
        {
            "children": [
            ],
            "propertyMap": {
                "text": "Hello!"
            },
            "type": "sayBlock"
        }
    ],
    "version": "0.9"
}
```

Рис. 3: Простейший пример файла сохранения модели

Загрузка происходит обратным способом с той лишь особенностью, что решение о том, какой блок создать, принимается уже описанной выше фабрикой на основании поля type.

3.2.5. Интерфейс

Так как на мобильных устройствах главным методом ввода является сенсорный экран, то от варианта с диаграммами было решено отказаться. При таком подходе получается слишком сложная навигация и слишком много места уходит впустую.

Было принято решение остановиться на последовательном "папочном" представлении блоков, где блоки на одном уровне идут друг за другом, а "сложные" (for, ifelse и т.п.) — раскрываются по нажатию (рис. 4).

Такая модель обеспечила удобное управление блоками с возможностью прокрутки и анимации. Основной проблемой при реализации данной модели стало то, что в

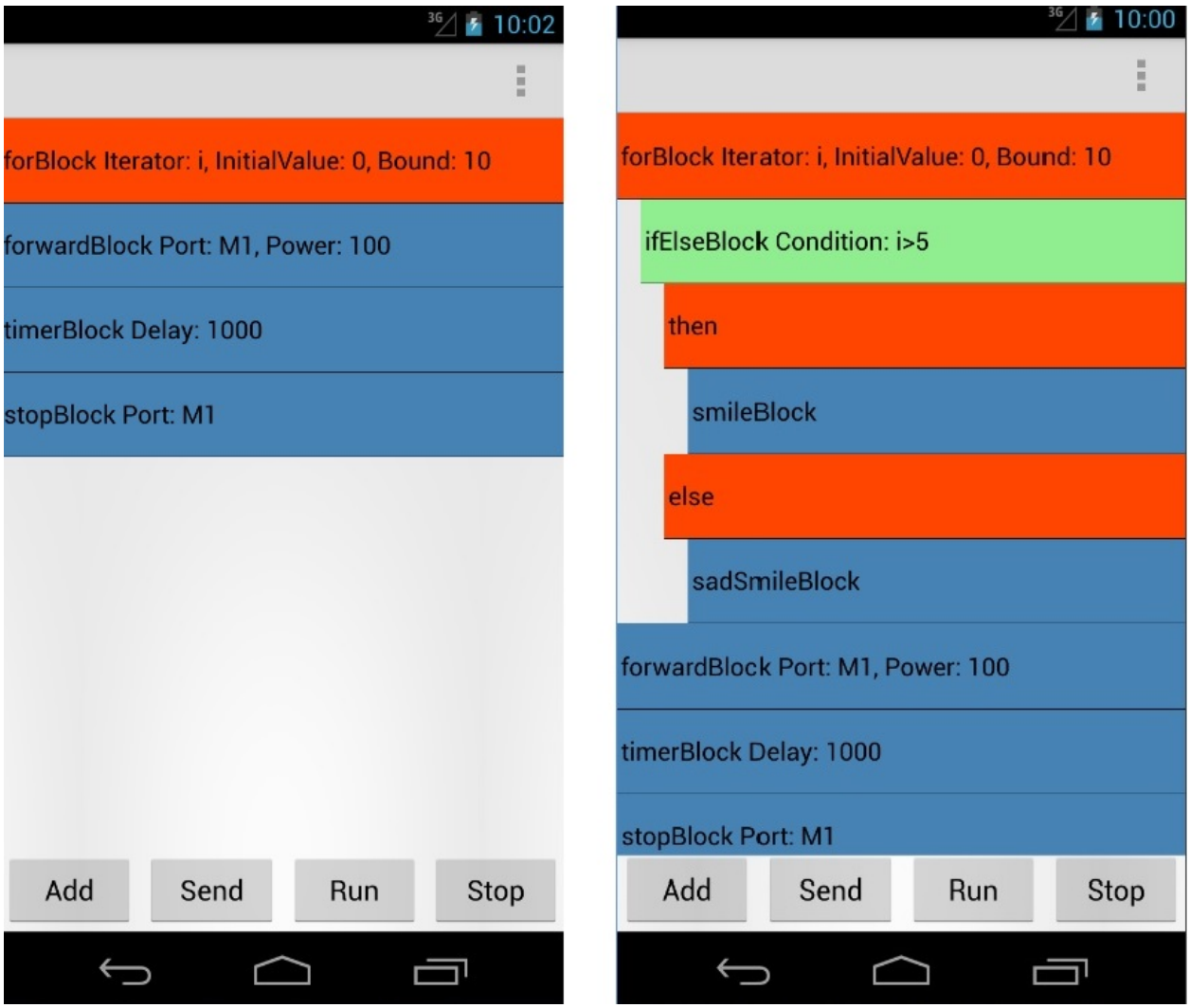

Рис. 4: Модель в свёрнутом и развёрнутом виде

QtQuick 2 на данный момент ещё нет TreeView, т.е. встроенного представления для отображения древовидных структур. Поэтому пришлось отображать модель дерева в плоскую модель списка, что заметно усложнило работу с добавлением, редактированием, анимацией и свёртыванием/развёртыванием блоков.

Заключение

Что было сделано

- Было написано кроссплатформенное приложение, позволяющее создавать программы почти эквивалентные по сложности программам, созданным в полноценной TRIK Studio.
- Разработан интерфейс удобный для использования на мобильных устройствах.
- Приложение было протестировано на работоспособность на платформах Android и Windows.
- Также были воссозданы примеры из поставки TRIK Studio.

Что предстоит сделать

- Протестировать работоспособность приложения на других платформах.
- Упростить и отшлифовать пользовательский интерфейс.
- Добавить поддержку других роботов, поддерживаемых TRIK Studio.

Список литературы

[1] Wikipedia. Usage share of operating systems // Википедия, свободная энциклопедия. — URL: http://en.wikipedia.org/wiki/Usage_share_of_operating_systems (дата обращения: 11/05/2015).

[2] Кибернетический конструктор ТРИК. — URL: http://www.trikset.com (дата обращения: 10/03/2015).

[3] Эрих Гамма Ричард Хелм Ральф Джонсон Джон Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования. — СПб. : Питер, 2007. — 366 с.