Модули в языке программирования КуМир 2.0

Пронин А.А.[1], Синельников Е.А.[2]
[1]*gorka19800@gmail.com*, [2]*sin@altlinux.org*
[1,2]*Саратовский государственный университет имени Н.Г. Чернышевского*

Аннотация. Статья посвящена описанию инструмента расширения набора библиотек школьного алгоритмического языка программирования КуМир.

Ключевые слова: Язык программирования КуМир, JSON, C++, разработка с открытым исходным кодом.

КуМир (Комплект Учебных МИРов) – система программирования с открытым исходным кодом, предназначенная для поддержки начальных курсов информатики и программирования в средней и высшей школе с открытым исходным кодом [1]. Спектр возможностей применения данной системы программирования ограничен, ряд стандартных команд позволяет разрабатывать небольшие приложения с целью изучения основ программирования на процедурных языках. На данный момент язык КуМир - это язык, с которого хорошо начать, чтобы освоить основы алгоритмического подхода и процедурный стиль программирования [2].

Система КуМир [3] в современном ее состоянии (с подсистемой ПиктоМир) состоит из расширяемого набора исполнителей (или роботов), набора систем программирования и вспомогательных утилит и программ. Расширяемый набор исполнителей (роботов) представляет собой отдельные самостоятельные программы и (или) электронно-механические устройства, имеющие собственное пультовое управление (интерфейс), а также возможность локального или сетевого управления из выполняющей системы.

Данная платформа поддерживает возможность добавления модулей и систем программирования, расширяющих спектр возможностей использования языка.

Ввиду активного использования языка КуМир обучающимися средней школы, расширение языка путем добавления новых модулей может предоставлять новые перспективы для школьников – изучение новых областей знаний и использование полученных навыков программирования в них. Например, язык КуМир может быть использован для обучения робототехнике – его исходные программные коды позволяют создать модуль, осуществляющий трансляцию программного кода, написанного на языке КуМир в язык С для программирования микроконтроллеров и далее осуществить прошивку платы. Так обучающийся сможет разрабатывать роботов, используя навыки программирования в системе КуМир.

Расширение набора исполнителей

Среда программирования КуМир – разработка с исходным программным кодом [4]. Данный факт позволяет свободно модифицировать, изменять и расширять её. Помимо раширения среды программирования, также можно расширять и сам язык КуМир, добавляя библиотеки и модули, предоставляющие готовые функции для решения определенных задач. В качестве модуля, в языке программирования КуМир выступают исполнители. С точки зрения разработчика языка КуМир, исполнитель – программный код, написанный на языке программирования C++, генерируемый по заданному декларативному описанию.

Для создания новго исполнителя системы КуМир 2.x следует:

– создать декларативное описание исполнителя: его систему команд, при необходимости пользовательский интерфейс;

– сгенерировать C++-проект по декларативному описанию, содержащему заготовки файлов, необходимых для реализации;

– реализовать функциональность исполнителя.

Под декларативным описанием понимается файл в формате JSON [5]. Оно предназначено для создания «скелета» исполнителя, использующегося для автоматической генерации специфичного для системы КуМир кода, взаимодействия с исполнителем и авто-генерации оберток для использования исполнителя совместно с языками программирования Python и Pascal.

Для создания C++-проекта по декларативному описанию используется скрипт gen_actor_source.py, расположенный в папке src/scripts репозитория[4]. Для запуска, необходимо передать флаг «-project», а также путь до файла с декларативным описанием в качестве аргументов.

По завершению работы скрипта, будут созданы 3 файла – CMakeLists.txt (файл проекта CMake), имя_исполнителя_module.cpp и имя_исполнителя_ module.h (заготовки для реализации функциональности исполнителя). Данные файлы создаются 1 раз при создании проекта и впоследствии не перезаписываются.

Описание генерируемого программного кода

При сборке (компиляции) проекта также генерируются вспомогательные файлы исходных текстов: имя_исполнителя_plugin.cpp/h и имя_исполнителя_modulebase.cpp/h. Модификация этих файлов запрещена, поскольку они будут перезаписаны при каждой повторной сборке проекта, и соответственно, все изменения в них – потеряны, о чем предупреждает комментарий в начале этих файлов.

Во время сборки генерируются исходные файлы классов ИмяИсполнителяModuleBase и ИмяИсполнителяPlugin. Класс ИмяИсполнителяModuleBase является абстрактным базовым классом для класса ИмяИсполнителяModule, в котором реализуется функциональность модуля. В случае модификации декларативного описания исполнителя, при изменении сигнатур команд исполнителя, изменится описание базового класса, что может привести к ошибке компиляции проекта.

Класс ИмяИсполнителяPlugin реализует механизм взаимодействия исполнителя с системой Кумир.

Сигнатуры команд исполнителя

При описании системы команд исполнителя обязательным является указании ASCII-имени каждой команды (как правило, на английском языке). При генерации C++классов исполнителя ИмяИсполнителяModule и ИмяИсполнителяModuleBase, имена соответствующих команд образуются из ASCII-имен следующим обрзом: удаляются пробелы из имени, при этом первые буквы «слов» имени становятся заглавными; к имени добавляется префикс «run».

Структура файла с декларативным описанием

Ниже приведены возможные блоки, использующиеся в декларативном описании.

```
ИСПОЛНИТЕЛЬ: {
        name: ИМЯ,
        description: ДОКУМЕНТАЦИЯ, /* опционально */
        methods: [ список описаний АЛГОРИТМ ],
        gui: описание пользовательского интерфейса GUI /* опционально */
}

ИМЯ: {
        ascii: имя в латинице,
        ru_RU: имя на русском языке,
        /* имена на других языках по мере необходимости
          ключ -- имя POSIX-локали
*/
}

ДОКУМЕНТАЦИЯ: {
        ru_RU: "текст на русском языке",
        /* ….. тексты на любых языках, ключ -- имя POSIX-локали */
}

АЛГОРИТМ: {
        name: ИМЯ,
        description: ДОКУМЕНТАЦИЯ, /* опционально */
        returnType: БАЗОВЫЙ ТИП ДАННЫХ,
        /* если returnType не указан, подразумевается "void" */
        arguments: [ список описаний АРГУМЕНТ ],
        async: True или False /* опционально */
        /* async=True -- алгоритму требуется некоторое время на работу
          (например, анимировать перемещение),
          async=False -- алгоритм выполяется немедленно
          (например, проставляет некий флаг или возвращает резуль-тат).
          По умолчанию async=True для алгоритмов, которые ничего не возвращают,
          и async=False для алгоритмов, которые что-то возвращают.
       */
}

БАЗОВЫЙ ТИП ДАННЫХ: {
        "void" /* нет типа данных */
```

```
        "int" /* цел */
        "double" /* вещ */
        "bool" /* лог */
        "char" /* сим */
        "string" /* лит */ }

ТИП ДОСТУПА: {
        "in" /* арг */
        "in/out" /* аргрез */
        "out" /* рез */
}

АРГУМЕНТ: {
        baseType: БАЗОВЫЙ ТИП ДАННЫХ,
        dim: размерность (число от 0 до 3),
        /* если dim не указан, то подразумевается размерность 0
        , то есть не таблица */
        access: ТИП ДОСТУПА,
        /* если access не указан, то подразумевается "in" */
        name: ИМЯ,
        description: ДОКУМЕНТАЦИЯ /* опционально */
}

КЛАСС ОКНА: {
        "main"
        "pult"
        /* …. любой произвольный тип */
}

ОКНО: {
        role: КЛАСС ОКНА,
        icon: имя файла с иконкой /* опционально */
}

ЭЛЕМЕНТ МЕНЮ: {
        title: ИМЯ,
        icon: имя файла с иконкой /* опционально */,
        items: [ список описаний ЭЛЕМЕНТ МЕНЮ ] /* опционально */
        /* если список items не объявлен, то подразумевается, что
           элемент меню является действием, а не вложенным меню */
}



GUI: {
        windows: [ список описаний ОКНО ],
        menus: [ список описаний ЭЛЕМЕНТ МЕНЮ ] /* top-level меню исполни-теля
*/
}
```

В ходе работы были изучены средства и методы разработки языка программирования КуМир, а именно механизм добавления исполнителей – библиотек программного кода, расширяющих набор доступных функций. Были

рассмотрены формат файла с декларативным описанием исполнителей, а также скрипт для авто-генерация заготовок программного кода по нему.

В дальнейшем планируется использовать результаты данной работы при разработке модуля-транслятора из языка КуМир в язык C++ и исполнителя «Ардуино».

Список литературы

[1] Оффициальный сайт КуМир [Электронный ресурс] URL: https://www.niisi.ru/kumir/ (дата обращения: 26.09.2022).

[2] Статья о КуМире на электронном образовательном портале Фоксфорд [Электронный ресурс] URL: https://foxford.ru/wiki/informatika/sreda-programmirovaniya-kumir (дата обращения: 10.10.2022).

[3] *Леонов А.Г., Кушниренко А.Г.* Методика преподавания основ алгоритмизации на базе системы «КуМир». М.: «Первое сентября», 2009.

[4] Репозиторий с исходным кодом языка программирования КуМир [Электронный ресурс] URL: https://github.com/a-a-maly/kumir2 (дата обращения: 22.09.2022).

[5] Спецификация стандарта JSON [Электронный ресурс] URL: https://www.json.org/json-en.html (дата обращения: 20.09.2022).