Конкатенативный язык программирования Factor

Алесь Гузик*

Аннотация

Concatenative programming languages is growing in popularity family of programming languages. Factor is one of the most mature of them and one of the most pragmatic. It has lots of features and is greatly extensible. Article gives brief introduction to Factor programming language, to its history, features and differences from other programming languages.

Конкатенативными называются такие языки программирования в которых композиция фрагментов кода выражается их конкатенацией. Обычно в таких языках используется стек для передачи данных между функциями без явного указания аргументов. В функциональных языках программирования аналогичный подход называется бесточечной нотацией (pointfree style).

Конкатенативные языки используют обратную польскую запись (RPN) для описания последовательности вычислений. Часто синтаксис таких языков представляется в виде линейной последовательности токенов, а не в виде дерева.

Одним из наиболее актуальных конкатенативных языков на данный момент является язык Factor. Первая версия языка была написана в 2003 году Святославом Пестовым в качестве скриптового языка для игры и была реализована на Java. В ходе развития реализация была переписана на смеси C и Factor и позднее C++ и Factor. Язык доступен для 32 и 64-битных версий Linux, Windows и Mac OS X и распространяется на условиях лицензии BSD.

*Минск, Беларусь, me@aguzik.net

Factor является динамически-типизированным конкатенативным языком программирования с автоматическим управлением памятью, придерживается концепций функционального и объектно-ориентированного программирования. Язык включает в себя множество свойств таких языков как Common Lisp, Smalltalk и Forth. Наследием Forth является конкатенативная натура языка, постфиксная запись, а так же многие операторы и встроенные функции. От Common Lisp взята концепция гомоиконности (код как данные), макросов и reader-макросов как средств расширения возможностей языка, а так же подход к реализации объектной системы. Из мира Smalltalk взята концепция хранения состояния программы в виде образа памяти, который можно сохранить в любой момент и позднее с того же места продолжить выполнение программы. Так же из Smalltalk взят подход к организации интерактивной среды разработки, когда любой объект можно исследовать и изменить, после чего изменения сразу же вступают в силу.

Factor имеет два компилятора — оптимизирующий и неоптимизирующий. Оптимизирующий компилятор написан на факторе и используется практически во всех случаях взаимодействия программиста с языком. Неоптимизирующий компилятор написан на C++ и используется на стадии бутстрапа среды, поскольку оптимизирующий компилятор не может быть доступен в этот момент.

Язык содержит развитые средства для двустороннего взаимодействия с внешним кодом, написаным на C — привязки к библиотеке FFI позволяют вызывать код библиотек, написанных на C и передавать код написанный на факторе для вызова из библиотек. Так же имеется возможность автоматической генерации привязок к библиотекам, использующим технологию GObject Introspection, а синтаксические расширения языка позволяют сделать описание интерфейса внешних библиотек тривиальным для тех случаев, когда автоматическая генерация привязок невозможна.

Стандартная библиотека крайне обширна и содержит в себе многие составляющие языка, которые в других языках не могут быть вынесены в библиотеки (поддержка локальных переменных и замыканий, синтаксис регулярных выражений и др.). На данном этапе развития язык не имеет пакетной системы для поиска и установки сторонних библиотек — многие библиотеки просто включаются в директорию `/extra` в основном дереве проекта. В стандартной библиотеке доступны ленивые коллекции, монады, опциональная статическая типизация, алгебраические типы данных,

паттерн-матчинг, поддержка literate programming, обратимые блоки кода (invertible quotations), модели (data-flow объекты, аналог Cells из Common Lisp), ленивые парсер-комбинаторы, генератор парсеров для заданых в виде EBNF грамматик, реализации различных подходов к многопоточности (обмен сообщениями, futures, promises, параллельные комбинаторы, классические мьютексы/семафоры и др.), своя GUI-библиотека поверх OpenGL, привязки к GTK, Cocoa, WinAPI, привязки к базам данных (SQLite, Postgres, MongoDB, CouchDB, Redis, Tokyo), веб-фреймворк Furnace, обёртки над API сервисов Twitter, Reddit, xkcd и многое другое.

Несмотря на то что версия 1.0 языка ещё не выпущена (текущая стабильная версия 0.96) и в данный момент ведётся активная работа по добавлению новых возможностей, существующие возможности языка достаточно стабильны и его можно рекомендовать как минимум для быстрой разработки прототипов приложений различных направлений — от игр и прикладных программ с графическим интерфейсом до серверных приложений и ведения научных расчётов.