УДК 004

ВОЗМОЖНОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ CRYSTAL

О.С. МИХНОВИЧ
(Представлено: Е.В. ДАНЧЕНКО)

В статье обсуждаются возможности языка программирования Crystal и его преимущества перед языками программирования C и Ruby на примере реализации задачи Ханойской башни.

На сегодняшний день существует множество различных языков программирования. Одни из них позволяют разработчику писать хорошо читабельный и эффективный код, другие же позволяют сосредоточить внимание на производительности будущего продукта. Языки программирования по виду типизации принято делить на две категории – статические и динамические. К первому, например, относятся: C/C++, Java, C#, а ко второму – Ruby, Python, JavaScript. Одним из популярных динамических интерпретируемых языков программирования является Ruby.

Ruby обладает независимой от операционной системы реализацией многопоточности, сильной динамической типизацией, сборщиком мусора и многими другими возможностями. По особенностям синтаксиса он близок к языкам Perl и Eiffel, по объектно-ориентированному подходу – к Smalltalk. Также некоторые черты языка взяты из Python, Lisp, Dylan и Клу. На сегодняшний день Ruby используется в связке с фреймворком Rails. Однако у Ruby есть и недостатки. Наиболее существенным недостатком Ruby является его производительность. Рассмотрим язык программирования Crystal, как более производительную альтернативу Ruby [1].

Crystal – это объектно-ориентированный язык общего назначения, спроектированный и созданный Ary Borenszweig и Juan Wajnerman. Crystal статически типизирован и имеет Ruby-подобный синтаксис. Первый официальный релиз языка произошел в июне 2014 года. Изначально компилятор языка был написан на Ruby, пока в 2013 году не был переписан на Crystal. Язык находится в активной разработке. Несмотря на схожесть синтаксиса, Crystal намного эффективнее, чем Ruby, компилируется в машинный код, используя LLVM, жертвуя при этом динамическими аспектами языка. По результатам тестов Crystal показывает схожую с языком C производительность. Язык использует Boehm garbage collector, обладает системой макросов, поддерживает дженерики, а также перегрузку методов и операторов[2].

При проектировании и разработке Crystal создатели ставили перед собой следующие цели:

1. Ruby-подобный синтаксис.
2. Статическая типизация, но без указания типа переменных или аргументов метода.
3. Возможность вызывать код на C путем написания биндингов внутри Crystal.
4. Есть оценка времени компиляции и генерации кода, дабы избежать шаблонности кода.
5. Эффективная компиляция в машинный код.

Для сравнения производительности Crystal возьмем два языка программирования: Ruby и C. Выбор пал на данные языки в связи с тем, что Crystal позиционируется как сбалансированная между данными языками технология. Для сравнения производительности реализуем решение задачи о Ханойской башне.

Ханойская башня[3] является одной из популярных головоломок XIX века. Даны три стержня, на один из которых нанизаны восемь колец, причём кольца отличаются размером и лежат меньшее на большем. Задача состоит в том, чтобы перенести пирамиду из восьми колец за наименьшее число ходов на другой стержень. За один раз разрешается переносить только одно кольцо, причём нельзя класть большее кольцо на меньшее.

Существует несколько подходов к решению (рекурсивно, «треугольное» решение, циклическое решение), все они дают идентичные результаты. Реализуем рекурсивное решение.

Реализация решения на Ruby представлена в листинге ниже.

```
require 'benchmark'
N=20
A=Array.new
B=Array.new
C=Array.new

LOG = false

def move(n, source, target, auxiliary)
  if n > 0
    move(n - 1, source, auxiliary, target)
    target.append(source.pop)
    print "*********\n#{A}\n#{B}\n#{C}\n" if LOG
    move(n - 1, auxiliary, target, source)
  end
```

```
end

def run()
  (1..N).each { |v| A << v }
  A.reverse
  move(N, A, C, B)
end
puts "\nRuby work time: #{Benchmark.measure { run() } }"
```

Реализация решения на Crystal представлена в листинге ниже.

```
require "benchmark"
N = 20
A = Array(Int32).new
B = Array(Int32).new
C = Array(Int32).new

LOG = false

def move(n, source, target, auxiliary)
  if n > 0
    move(n - 1, source, auxiliary, target)
    target << source.pop
    print "********\n#{A}\n#{B}\n#{C}\n" if LOG
    move(n - 1, auxiliary, target, source)
  end
end

def run()
  (1..N).each { |v| A << v }
  A.reverse
  move(N, A, C, B)
end

puts "\nCrystal work time: #{Benchmark.measure { run() } }"
```

Листинги с решениями задачи «Ханойская башня» на языках Crystal, C, Ruby находятся на GitHub [4]. После запуска решений на выбранных языках получаем следующие результаты (в секундах):

Таблица 1 – Результаты тестов

Язык	Тест 1	Тест 2	Тест 3	Среднее значение
C	0.000895	0.000957	0.000955	0.000936
Crystal	0.021738	0.021611	0.013851	0.019067
Ruby	0.364303	0.359521	0.359140	0.360988

Проанализировав полученные результаты можно сделать вывод: язык программирования Crystal сочетает в себе достаточно высокую, как в C, производительность, и читабельный, и лаконичный, как в Ruby, синтаксис и может применяться для разработки программных продуктов под операционные системы семейства Unix. Из минусов данного языка можно отметить пока слабое комьюнити и малое количество библиотек.

ЛИТЕРАТУРА

1. Ruby [Электронный ресурс] – Режим доступа: https://www.ruby-lang.org/. – Дата доступа: 20.01.2020.
2. Crystal [Электронный ресурс] – Режим доступа: https://crystal-lang.org/. – Дата доступа: 20.01.2020.
3. The Tower of Hanoi [Электронный ресурс] – Режим доступа: https://en.wikipedia.org/wiki/Tower_of_Hanoi/. – Дата доступа: 20.01.2020.
4. Crystal benchmark [Электронный ресурс] – Режим доступа: https://github.com/olegmikhnovich/cr-rb-c-benchmark/. – Дата доступа: 20.01.2020.