Язык Ruby

История

Создатель Ruby — Юкихиро Мацумото (Matz) — интересовался языками программирования, ещё будучи студентом, но идея о разработке нового языка появилась позже. Ruby начал разрабатываться 23 февраля 1993 года и вышел в свет в 1995 году.

Название навеяно языком Perl, многие особенности синтаксиса и семантики из которого заимствованы в Ruby: англ. pearl — «жемчужина», ruby — «рубин».

Одним из источников вдохновения для Мацумото для разработки Ruby был научно-фантастический роман «Вавилон-17», основанный на гипотезе Сепира — Уорфа.

Целью разработки было создание «настоящего объектно-ориентированного», лёгкого в разработке, интерпретируемого языка программирования.

Сейчас Ruby входит в большинство дистрибутивов ОС Linux, поставляется вместе с Mac OS X, доступен пользователям других операционных систем. Одним из основных приложений, связанных с Ruby, продолжает оставаться Ruby on Rails, который продолжает активно развиваться, но использование Ruby значительно шире — на нём разрабатывается большое количество приложений различного назначения, кроме того, он используется в качестве скриптового языка для автоматизации и настройки приложений и написания административных утилит, в частности, в ОС Linux.

Области применения.

Ruby — современный, постоянно развивающийся язык программирования, областей применения ему — масса. Наверняка вы слышали про Chef, Vagrant, Homebrew, но чаще всего все мы слышим о Rails.

это невероятно выразительный и гибкий язык, который позволяет одну и ту же задачу решить многими способами.

Ruby — интерпретируемый, полностью объектно-ориентированный язык программирования с четкой динамической типизацией. Он сочетает в себе Perl-подобный синтаксис с объектно-ориентированным подходом. Также некоторые черты заимствованы из языков программирования Python, Lisp, Dylan и CLU. Кроссплатформенная реализация интерпретатора языка Ruby распространяется на условиях открытого программного обеспечения. Код, написанный на Ruby, может быть понятен даже человеку, который не разбирается в программировании. На RoR были созданы такие проекты, как Redmine, Twitter, Shopify, Basecamp, GitHub, Kickstarter, Airbnb и другие.

Преимущества Ruby

Многочисленное и доброжелательное комьюнити.

Довольно высокий порог входа, что означает, что Ruby-разработчик с большой вероятностью имеет опыт работы как минимум с еще одним языком программирования.

Вы используете только те библиотеки и модули, которые необходимы.

Существует большое количество полезных библиотек, которые уже готовы к использованию (Ruby Gems).

В интернете есть много информации по Ruby, в структурированном и отсеянном виде.

В контексте обсуждения Ruby нельзя не упомянуть популярнейший фреймворк Ruby on Rails.

Основы синтаксиса

Абсолютно всё (кроме управляющих конструкций) в Руби является объектом, строки, числа, массивы, классы - всё это экземпляры классов Ruby. Например строка экземпляр класса String, а массив - экземпляр класса Array.

Интерпретатор Ruby читает вашу писанину как последовательности лексем. Лексемы состоят из ключевых слов, знаком пунктуации, комментариев, литералов и идентификаторов. Идентификатором в руби называют обычное имя, которое присваивается переменным, методам, классам, атрибутам классов и тд. Идентификатор не может начинаться с цифры, и состоит из букв, цифр и символа подчеркивания.

Комментарии

В процессе написания программы, по разным причинам, часто приходится оставлять комментарии к коду. Комментарии могут быть необходимы как для человека так и для среды разработки и даже для самого руби. Комментарии могут быть однострочные и многострочные.

Однострочные комментарии начинаются со знака # и продолжаются до конца строки. Когда интерпретатор встречает символ #, он смотрит, не является ли решетка частью строки или регулярного выражения, если не являеться - то считает все символы после решетки комментарием (кроме символа перехода на новую строку - \n).

Литералы

Литерал - это запись в исходном коде Ruby представляющая собой фиксированное значение (константа). Это могут быть числа , строки и регулярные выражения

Константы, переменные и типы

Переменные имеют несколько разновидностей: локальные переменные, переменные класса, переменные экземпляра класса, глобальные переменные, псевдопеременные и константы.

Константами называют идентификаторы, которые имеют фиксированное значение, другими словами это та же переменная, значение которой не изменяется. При попытке изменить значение константы, интерпретатор выдаст предупреждение (но не ошибку, что на мой взгляд "странно")**.** По соглашению, константы следует писать с заглавной буквы, или же полностью верхним регистром (второй вариант преподчтительнее). Т.к. имена классов и модулей мы пишем с большой буквы - значит эти имена ничто иное как константы.

Что такое вообще переменная? Переменная - это область памяти, адрес которой можно использовать для осуществления доступа к данным. Данные, находящиеся в переменной (то есть по данному адресу памяти), называются значением этой переменной. То есть выражение a = 123 говорит о том, что объявляется переменная a (если она небыла объявлена ранее) и ей присваивается значение 123 . Это значение записывается в оперативную память компьютера, и чтобы извлечь эту информацию в будущем, нам нужно вызвать переменную a которая и является адресом ячейки памяти, где хранится нужное нам значение.

Локальные переменные. Видимость локальных переменных не выходит за пределы области где они объявлены. Обозначаются они с маленькой буквы, первым символом не должна быть цифра, может состоять из букв, цифр и знака подчеркивания (_).

Переменные экземпляра класса. Используются внутри класса, и областью видимости этих переменных является объект (экземпляр класса). Синтаксис написания: первый символ - собака (@), вторым символом не должна быть цифра, может состоять из букв, цифр и знака подчеркивания (_)

Переменные класса. Синтаксис написания: первые два символа - собаки (@@), третьим символом не должна быть цифра, может состоять из букв, цифр и знака подчеркивания (_). Между переменными класса и переменными экземпляра класса разница очень существенная, и заключается она только в области видимости. Область видимости переменной класса - сам класса и ВСЕ, повторюсь ВСЕ экземпляры класса.

Глобальные переменные.
Имена глобальных переменны начинаются со знака $. Область видимости глобальных переменных - вся программа. Используя метод global_variables вы можете посмотреть системные глобальные переменные.

Псевдопеременные
Псевдопеременные выглядят также как и локальные переменные. Что это такое понятно из самого названия - это переменные которые не переменные, как "курица что не курица" в Пятом правиле волшебника Терри Гудкайнда. Создавать псевдопеременные самому нельзя. Список псевдопеременных Ruby: self , nil , true , false , __FILE__ и __LINE__.

Строки

Строка яляется объектом класса String, а раз так, то она имеет в своем арсенале все доступные экземпляру класса методы по работе со строками.
Строки это ни что иное как набор любых символов заключенный в одинарные или двойные кавычки. Так это у большинства ЯП (языков программирования), но руби такой руби, что и тут сумел выделить себя среди других ЯП. Помимо стандартного синтаксиса литерала строк, у руби есть специальные идентификаторы, среди которых есть и идентификатор строк %q. Он позволяет указывать любую пару символов, внутри которой будет находится строка.

Массивы и хеши

Массивы это своего рода упорядоченный набор данных, который может хранить разные типы данных (разные экземпляры классов чтоб быть точнее) внутри себя, и т.к. массив это

экземпляр класса Array, то мы имеем в распоряжении все публичные методы класса Arrayудля работы с ним. Всё это мы рассмотрим в отдельной главе, которая будет посвещена только массивам и хешам, а сейчас мы просто вскользь глянем что это такое и зачем оно нужно. У каждого значения массива есть свой ключ или как многие называют - индекс, по которому происходит доступ к значению индекса, или как правильнее будет сказать - соответствующему элементу массива., т.е. элемент массива связан со своим индексом, а индекс соответственно с элементом. Индексы имеют последовательность от нуля до предела ОЗУ компьютера

Хеши. Под хешами подразумевается ассоциативный массив. Крайне удобная вещь, и очень часто используется на практике. Хеши являются экземплярами класса Hash, и методы для работы с хешами в большинстве своем те же самые что и с массивом. Разница между массивом и хешем (ассоциативным массивом) заключается в том, что хеши требуют явного определения индекса значения. Индекс может быть строкой, числом или символом.

Булевые типы данных

К булевым типам данных относятся псевдопеременные true(истина), false(ложь) и nil(пустота, вакуум, ничто)
При программировании абсолютно на любом языке, мы часто проверяем что либо на соответствие определенному условию, например такаято переменная пуста? или соответствует ли введенный юзером логин необходимому формату? А передал ли пользователь нам такуюто переменную? В таких случаях используются булевые проверки.

Символы

Символ это экземпляр класса Symbol. Синтаксис - двоеточие перед самим идентификатором (:symbol, :user , :qpi). Символ (symbol) имеет кое что общее со строкой, и с числом. Он похож на строку, так как тоже имеет последовательность символов, и похож на число, т.к. хранится в памяти как число.

Базовые типы данных

Ruby реализует идеологию «всё — объект», то есть любая единица данных является объектом — экземпляром некоторого класса, к которому применимы все синтаксические средства, предназначенные для работы с объектами. В этом смысле язык не содержит встроенных примитивных типов данных. Условно таковыми можно считать типы, предоставляемые интерпретатором и системной библиотекой, используемые наиболее часто и не требующие для использования специального указания имени класса.

Целые числа

Представлены типами Fixnum и Bignum. Первый тип используется для чисел, по модулю не превышающих 2^{30}, второй — для чисел более 2^{30}. В арифметических операциях эти числовые типы полностью совместимы и могут свободно использоваться вместе, между ними обеспечивается прозрачная конвертация. Тип Fixnum имеет ограниченную разрядность и использует стандартные арифметические команды процессора; разрядность Bignum ограничена только объёмом доступной оперативной памяти, а операции с ними базируются на алгоритмах вычислений с неограниченной точностью. Это позволяет производить точные вычисления с любым требуемым количеством знаков. Например, для большинства языков программирования написать программу точного вычисления факториала, которая работала бы для аргумента порядка сотни — достаточно сложная задача. В Ruby это делается элементарно, так как проблемы работы с длинными числами берёт на себя интерпретатор.

Другие числовые типы.

Тип Float — числа с плавающей запятой, представленные фиксированным числом разрядов. Плавающие числа записываются либо в естественной, либо в экспоненциальной форме. Системная библиотека mathn предоставляет также типы Rational (рациональное число) и Complex (комплексное число). Оба этих типа автоматически преобразуются к целым и плавающим при единичном знаменателе и нулевой мнимой части соответственно.

Строки

Строка — изменяемый массив байтов, представляющий символы в кодировке UTF-8. Реализуются классом String, имеющим большой набор методов, обеспечивающих анализ, манипуляции с содержимым, преобразования, поиск.

Строковые литералы ограничиваются апострофами, двойными кавычками, либо заключаются в конструкцию %q[…] или %Q[…]. Ограничители строки влияют на использование внутри неё специальных символов:

Если строка ограничена апострофами, внутри неё распознаются только специальные последовательности «\\» и «\'», обозначающие соответственно, обратный слэш и апостроф.

Если строка ограничена двойными кавычками, то в ней распознаются также управляющие символы «\t» (знак табуляции), «\n» (перенос строки), «\010» (любой символ в восьмеричной кодировке) и другие.

Ограничители %q[…] или %Q[…] (можно использовать круглые, квадратные, фигурные и угловые скобки) позволяют записывать строки с использованием апострофов и кавычек без экранирования. Форма %q[…] также обеспечивает непосредственный вывод управляющих последовательностей

Символы. Диапазоны

Символ — это неизменяемая строка. Символьные литералы записываются с префиксом «:» (двоеточие)

Диапазон — это экземпляр класса Range, он представляет собой непрерывную последовательность целых чисел от начального до конечного значения. Диапазон задаётся парой целых чисел, разделённых двумя или тремя точками.

Структура программы

Программа на Ruby представляет собой текстовый файл, содержащий последовательность инструкций — команд и описаний. При запуске программного файла на исполнение интерпретатор последовательно читает файл и выполняет инструкции. В Ruby не требуется организовывать тело главной программы в виде специального программного модуля (подобно функции main() в языке Си), составляющие его команды просто записываются непосредственно в тексте файла программы. Поскольку программный файл обрабатывается интерпретатором последовательно, любые функции, методы, описания должны предшествовать в тексте программы их первому использованию.

Управляющие конструкции

Ruby содержит богатый набор управляющих конструкций; многие их варианты являются достаточно редкими.

Условный оператор if выглядит традиционно:

```
if x > 0 then
   puts "x - положительное число"
x < 0 then
   puts "x - отрицательное число"
else
   puts "x - нуль"
end
```

Ветвей elsif может быть любое количество, использование ключевого слова then допустимо, но не обязательно, ветви elsif и else могут отсутствовать. Помимо этой «канонической» формы условного оператора, язык поддерживает и несколько других.

Можно использовать сокращённые формы условного оператора как *модификаторы* инструкций. Они пишутся после инструкции и интерпретируются как условие, при котором данную инструкцию следует выполнять. Ветви else в модификаторах быть не может.

Контейнеры

Ruby поддерживает динамические гетерогенные массивы, которые автоматически изменяют размер и могут содержать элементы любых типов. Массив является экземпляром класса Array, который предоставляет мощные средства для работы с хранимыми данными.

Обзор объектно-ориентированного программирования

- Определите вещи, с которыми ваша программа имеет дело
- Этими вещами (их «чертежами») являются классы
- Классы содержат методы (поведение)
- Объекты являются экземплярами этих вещей
- Объекты содержат переменные экземпляров (состояние)

Итого

Ruby — это язык, который позволяет работать без большого количества неудобств и церемоний, которые приходят со строго типизированными языками. С Ruby легко начать работать, особенно если у вас уже есть опыт разработки на других языках программирования, и вы сможете быстро создавать прототипы с Ruby on Rails. В Японии, откуда он появился, Ruby использовался для создания игр. Ruby лаконичен и читается как английский, что делает код понятным для новичков.