



progress engine
we will develop it

#tceh

2016

Врубиться в Ruby

Лекция 2

Лекция 2 - базовые знания



- Базовые типы данных в ruby - числа, текст, булевы значения
- Коллекции - индексные и ассоциативные массивы (хэши), диапазоны
- Идентификаторы (символы)
- Комментарии в коде
- Регулярные выражения, сервис rubular
- Операторы
- Условные предложения if/else/elsif, а также “условие ? true : false”
- Циклы while/for
- Управление ходом выполнения return/break/next/redo

Что такое переменные

Вспомним прошлое занятие

puts “Hello World”

Немного доработаем код

```
puts "What is your name?"  
$var = gets  
puts "Hello World from " + $var
```

Базовые типы данных в Ruby

Boolean

True



False



Numeric

Integer

1, 2, 3 999999999

Float

1, 239999999999.....

String

```
puts “Некоторое строковое выражение”
```

```
puts “Некоторое строковое выражение”.size
```

```
age = 37
```

```
puts “Мой возраст ” + age + “ лет”
```

```
puts “Мой возраст #{age} лет”
```

puts “Некоторое строковое выражение”.**empty?**

puts “”.**empty?**

puts “Некоторое строковое выражение”.**[1]**

puts “Некоторое строковое выражение”.**[-1]**

puts “Алукард”.**reverse**

puts "шило в мешке не утаишь".**sub("шило", "мыло")**

Symbol

:symbol

: 'yet another symbol'

Разница между String и Symbol

```
puts :abc + :cde
```

```
puts 'abc' + 'cde'
```


Array

[a, b, c]

["Быть", "Или", "Не быть", "Вот в чем вопрос"]

[1, 2, 3]

[1, "a", :abc]

“Что грядущее нам готовит?”.**split**(“ ”)

[“Что”, “грядущее”, “нам”, “готовит?”].**join**(“ ”)

[“Что”, “грядущее”, “нам”, “готовит?”].**reverse**

```
string = "жыло-было шыбко шыпящее животное"
```

```
string.scan("шы") #=> ["шы", "шы"]
```

```
string.scan("шы").size #=> 2
```

```
string.scan("жы").size #=> 2
```

```
string.scan("жы").size + string.scan("шы").size #=> 4
```

Регулярные выражения

- Символьные классы
- Квартификаторы
- Альтернативы
- Группировки
- Модификаторы

Символьные классы - перечисление символов,
которые может содержать строка.

/abcXYZ/

/a-zA-Z/

/0-9a-zA-Z/

/^0-9/

Короткие записи популярных символьных классов

Короткая запись	Полная запись	Описание
<code>\s</code>	<code>[\f\t\n\r]</code>	Пробельный символ
<code>\S</code>	<code>[^\f\t\n\r]</code>	Любой символ, кроме пробельного
<code>\d</code>	<code>[0-9]</code>	Цифра
<code>\D</code>	<code>[^0-9]</code>	Любой символ, кроме цифры
<code>\w</code>	<code>[a-zA-Z0-9]</code>	Латиница или цифра
<code>\W</code>	<code>[^a-zA-Z0-9]</code>	Любой символ, кроме латиницы или цифры
<code>.</code>	<code>[^\n\r]</code>	Любой символ, кроме перевода строки
<code>\b</code>		Граница слова
<code>\B</code>		Не граница слова
<code>\A</code>		Начало строки
<code>\Z</code>		Конец строки

Квантификатор - Показывает, сколько раз может повторяться предыдущий символ, группа, альтернатива, etc. Квантификатор ограничивается фигурными скобками.

$\Lambda w\{3\}/$	$\# =>$ три латинских буквы или цифры
$\Lambda d\{1, 3\}/$	$\# =>$ одна, две или три цифры
$/[a-яA-Я]\{3,\}/$	$\# =>$ русское слово длиной три символа и больше

Короткие записи популярных квантификаторов

Короткая запись	Полная запись	Описание
*	{ 0 , }	Любое количество
+	{ 1 , }	Один и более
?	{ 0 , 1 }	Есть или нет

Альтернатива

/(жышы)/

#=> или "жы", или "шы"

Группировка

()

Фиксирующие директивы — это символы, которые привязывают правило к некоторому признаку. Например, к концу или началу строки.

<code>/^\d+/</code>	<code>#=></code> строка начинается с числа
<code>/\w+\$/</code>	<code>#=></code> последнее слово на латинице или число
<code>/^\$/</code>	<code>#=></code> пустая строка

Модификатор предназначен для изменения поведения правила. Он размещается сразу же после правила (после последней наклонной черты)

/(hello|world)/i #=> или "hello", или "world". Причём независимо от регистра

/s+/mix #=> несколько подряд идущих пробельных СИМВОЛОВ

- **m**ultiline — перенос строки считается простым символом,
- **i**gnorecase — поиск без учёта регистра,
- **e**xextended — игнорировать пробельные символы.

Игнорирование регистра работает только для латиницы.

"a b c 12".scan(^d+^)

"a b c 12".scan(/[a-z]+^)

"a b c 12".gsub(/^d+/, "xyz")

"a b c 12".gsub(/[a-z]+/, 9)

"a b c 12".gsub(/[a-z]+/, "9")

Практическое задание

Давайте напишем регулярное выражение,
проверяющее строковое значение - корректный ли
это формат email

Hash

(ассоциативный массив)

`{:first_name => "Alexey", :last_name => "Poimtsev"}`

`{first_name: "Alexey", last_name: "Poimtsev"}`

{first_name: “Alexey”, last_name: “Poimtsev”}.**keys**

{first_name: “Alexey”, last_name: “Poimtsev”}.**values**

{first_name: “Alexey”, last_name: “Poimtsev”}[:**first_name**]

{first_name: “Alexey”, last_name: “Poimtsev”}.**size**

```
hash = {first_name: "Alexey", last_name: "Pointsev"}
```

```
hash.merge({age: 37})
```

```
puts hash
```

```
hash = {first_name: "Alexey", last_name: "Poimtsev"}
```

```
hash.delete(:first_name)
```

```
puts hash
```


Домашнее задание

- 1 - Написать регулярное выражение для проверки url
- 2 - Дана строка слов, разделённых пробелами.
Вывести длиннейшее слово
- 3 - Найти в тексте время в формате
«часы:минуты:секунды»
- 4 - Найти в тексте слова, содержащие две
прописные буквы, и исправить.

Вопросы?

+7 (926) 889-16-32

alec@alec-c4.com

<http://alec-c4.com/>