Wikidot.com

site-name.wikidot.com
Share on
Edit History Tags Source
Explore »

Ruby Tutorial

...o como pasar un buen rato programando

Search this site Search

- admin
  - site manager

Create account or Sign in

Lección 1

- Introducción
- Instalación
- El Primer Programa
- Números en Ruby
- Strings y diversión
- Variables
- Alcance de las variables

Lección 2

- Introduciendo Datos
- Normas en los nombres

- Los métodos
- Los métodos: argumentos
- Rangos
- Arrays

Lección 3

- Bloques
- Más malabares con strings
- Expresiones Regulares
- Condicionales
- Bucles
- Números Aleatorios

Lección 4

- Clases y Objetos
- Accesores
- Ficheros: lectura/escritura
- Cargando librerías
- Herencia de clases
- Modificando clases
- Congelando objetos
- Serializando objetos

Lección 5

- Control de acceso
- Excepciones
- Módulos
- Constantes
- Hashes y Símbolos
- La clase Time

Lección 6

- self
- Duck Typing
- Azúcar Sintáctico
- Test de unidades

contacto

e-mail

Rangos

El principal uso y quizás el más apropiado para los rangos, es expresar una secuencia: las secuencias tienen un punto inicial y un punto final, y una forma de producir los sucesivos valores entre ambos. En Ruby, esas secuencias son creadas usando los operandos **..** y **...**

- **..** genera una secuencia donde los puntos límites están incluidos.

```
(1..3).to_a

#es la secuencia 1, 2, 3
```

- **...*** genera una secuencia en la que no está incluida el límite superior.

```
(1...5).to_a
```

```
#equivale a 1, 2, 3, 4
```

En Ruby los rangos no son almacenados como una lista: los rangos se almacenan como un objeto `Range`, y contiene referencias a dos objetos `Fixnum` (su límite superior e inferior). Se puede convertir un rango en un array (array = lista, conjunto ordenado de elementos), mediante el método **`to_a`**.

```
(1..10).to_a

#obtenemos [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

métodos

Los rangos en ruby tienen diversos métodos:

```
nums = -1...9
puts nums.include?(5)   # true
puts nums.min           # -1
puts nums.max           # 8
puts nums.reject {|i| i < 5} # [5, 6, 7, 8, 9]
```

Uno de los usos útiles de los rangos, es comprobar si un determinado valor está en el intervalo representado por el rango. Para eso usamos el operador ===

```
(1..10) === 5       # true
(1..10) === 15      # false
(1..10) === 3.14159 # true
('a'..'j') === 'c'  # true
```

page_revision: 2, last_edited: 15 Jun 2010, 12:33 GMT-05 (167 days ago)
EditTags History Files Print Site tools+ Options
Help | Terms of Service | Privacy | Report a bug | Flag as objectionable
Powered by Wikidot.com
Unless otherwise stated, the content of this page is licensed under Creative Commons Attribution-ShareAlike 3.0 License

Other interesting sites

The Future of Energy

Analyzing The Options

ABCdba.com

The DBA's quick reference website

Chemistry

Gmamber

The Center of All Things