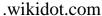
Wikidot.com





Ruby Tutorial

...o como pasar un buen rato programando

- admin
 - o 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
- <u>La clase Time</u>

Lección 6

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

contacto

e-mail

Congelando objetos

Los **objetos inmutables** son aquellos que no pueden cambiar de estado después de ser creados. Las propiedades por las que destacan son:

•

- son thread-safe. Los threads no pueden corromper lo que no pueden cambiar.
- hacen más fácil el implementar la **encapsulación**: si parte del estado de un objeto es almacenado dentro un objeto inmutable, entonces los métodos modificadores pueden leer el estado de dicho objeto, sin miedo a que modifiquen su estado.
- son buenos índices en los hashes, ya que no pueden cambiar.

En Ruby, la **mutabilidad** es una propiedad de los objetos, no de una clase entera. Cualquier objeto (o instancia) se puede volver inmutable usando **freeze**.

freeze

El método freeze (congelar) evita que un objeto pueda modificarse, convirtiéndolo en una constante. Después de "congelar" el objeto, cualquier intento de modificarlo da como resultado un TypeError.

```
str = 'Un simple string'
str.freeze # congelamos el string

# se intenta modificar (begin)
# y en caso de error (rescue)
# se lanza un mensaje. Ver Excepciones.

begin
    str << 'Intento de modificarlo'
rescue => err
    puts "#{err.class} #{err}"
end

# La salida es - TypeError can't modify frozen string.
```

Sin embargo, freeze funciona con las referencias, no con las variables: esto significa que si creamos un objeto nuevo, y sobreescribimos la variable, este se podrá modificar:

```
str = 'string original - '
str.freeze
str += 'añadido a posteriori'
puts str

# La salida es - 'Original string - añadido a posteriori'
```

El objeto original no cambió. Sin embargo, la variable str se refiere a un nuevo objeto.

El método frozen? nos dice si un objeto está congelado o no.

page_revision: 0, last_edited: 5 Dec 2007, 08:03 GMT-06 (1091 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



Heroesdarkfuture



2012hoax

Debunking the "2012 Doomsday"



Lord-Lieutenant of Ross and Cromarty



Dragons of Darkmoon

...do not meddle in the affairs of dragons...