Ruby Tutorial

...o como pasar un buen rato programando

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

Ficheros: lectura/escritura

Veamos como se puede leer/escribir un fichero con un ejemplo:

```
# Abre y lee un fichero
# Se usa un bloque: el fichero se cierra
# automáticamente al acabar el bloque.

File.open('fichero.txt', 'r') do |f1|
  while linea = f1.gets
    puts linea
  end
end

# Crea un nuevo fichero, y escribe
File.open('text.txt', 'w') do |f2|
```

```
  # '\n' es el retorno de carro
  f2.puts "Por que la vida \n puede ser maravillosa"
end
```

El método **`File.open`** puede abrir el fichero de diferentes formas:

- '**r**' sólo-lectura, comienza al principio del fichero.
- '**r+**' lectura/escritura, comienza al principio del fichero.
- '**w**' sólo-escritura, crea un nuevo fichero o elimina el contenido del fichero, para empezar de cero.

Hay que consultar la documentación para ver una lista completa de los posibles modos. `File.open` abre un nuevo fichero si no hay un blque; pero si usa un bloque, entonces el fichero será el argumento del bloque, y se cerrará automáticamente cuando termine el bloque. Si no hay bloque, el fichero no se cierra de forma automática, y siempre, siempre hay que cerrar un fichero: en el caso de un fichero abierto para escritura, si no se cierra, podrían perderse datos.

El método **`readline`** de `File` copia el fichero línea por línea dentro de un array.

Ambos métodos, `open` y `readlines` pertenecen a la clase **`IO`**, de la cual desciende la clase `File`; y por tanto, son heredados.

Manejando directorios

El módulo **`Find`** hace un búsqueda descendente en los directorios de un conjunto de rutas/directorios. Si el argumento es un directorio, entonces el resultado será el directorio, y todos los ficheros y subdirectorios que le pertenecen.

```
require 'find'

# muestra la ruta ./
# que es el directorio de Ruby
Find.find('./') do |f|
    type = case
    # si la ruta es un fichero -> F
        when File.file?(f) then "F"
        # si la ruta es un directorio -> D
        when File.directory?(f) then "D"
        # si no sabemos lo que es -> ?
        else "?"
    end
  # formatea el resultado
```

```
  puts "#{type}: #{f}"
end
```

Acceso aleatorio

Por acceso aleatorio, se entiende, empezar a leer el fichero en cualquier parte.
Supongamos 'aleatorio.rb':

aleatorio.rb

```
puts 'Surfing USA'
```

Ahora queremos leer el fichero a partir de la palabra "USA":

```
f = File.new("aleatorio.rb")

f.seek(12, IO::SEEK_SET)
print f.readline
f.close
```

El método **`seek`** de la clase `IO`, busca una posición dada por el primer parámetro, de la forma indicada por el segundo parámetro. Las posibles formas son:

- **SEEK_CUR** - busca (seek) desde el primer parámetro, un número entero, hasta la posición actual.
- **SEEK_END** - busca desde el parámetro dado, hasta el final del fichero.
- **SEEK_SET** - busca la posición absoluta dada por el parámetro
- **::** es el operador de alcance

¿Permite Ruby la serialización de objetos?

Java tiene la habilidad de serializar objetos: te permite almacenarlos en un fichero y luego reconstruirlos cuando es necesario. Ruby llama a este tipo de serialización **marshalling**.

Salvar un objeto y algunos o todos de sus componentes, se hace mediante el método **`Marshall.dum`**. Después, se puede recuperar el objeto usando **`Marshall.load`**. Se profundizará más adelante en el tema.

Other interesting sites

Marvel Untold MUX

A Barrel Full

A Barrel Full of information for Oil Industry Professionals

BlueMist

Philanthropic Art Society

Ou la promotion de l'Art par la Philanthropie