#### Unidad 2

#### 2.1)Estructuras de control en Ruby

**Temas:** Temas: Estructura de control en Ruby: while, each, switch, if, for, ciclos incrementables. Rangos. Tiempo. Dibujar patrones con ciclos. Ciclos anidados.







### Estructuras de control

Hasta acá habíamos ejecutado código que imprime texto, operaciones matemáticas y hemos evaluado algunas condiciones o valores de los datos.

En esta unidad revisaremos las estructuras de control, ciclos, iteradores, que permiten ya no solamente evaluar el valor de un objeto, también permitirán generar o desencadenar rutinas, asignaciones o respuestas mientras se cumpla una condición o varias condiciones.

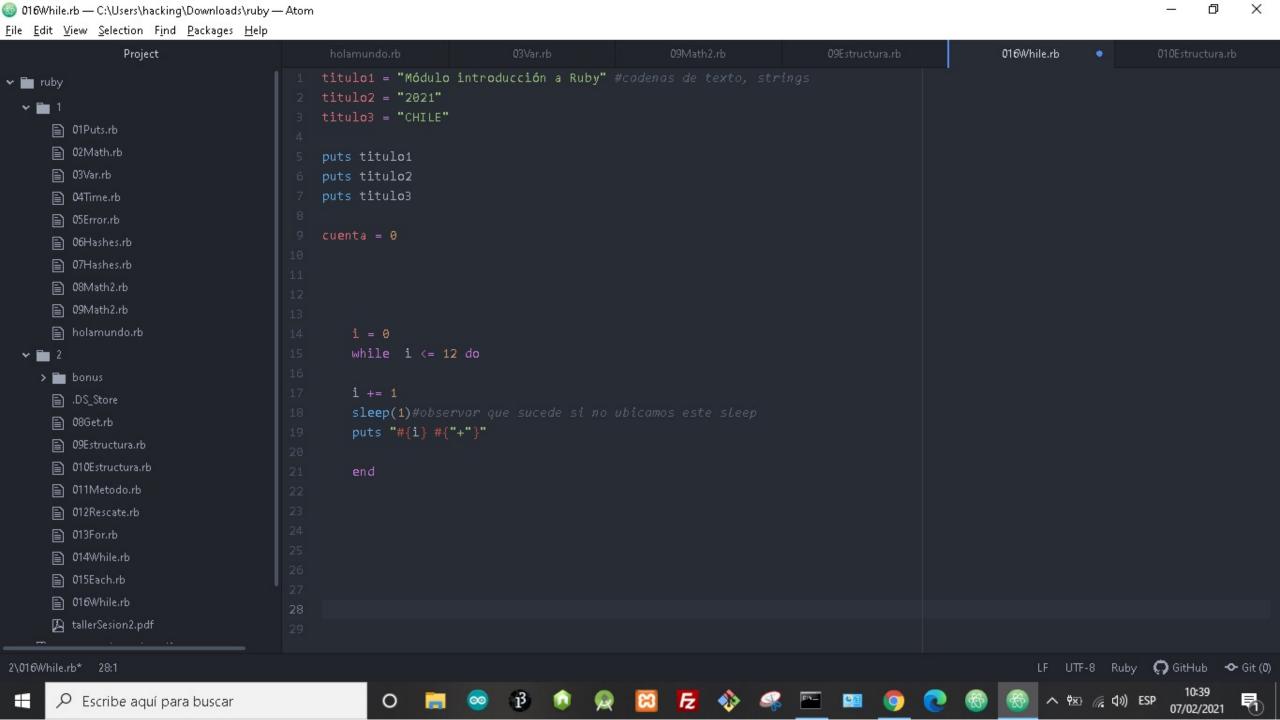
## while

### While

La estructura es la siguiente:

while condición hace esto

end



### each

El ciclo each permitirá iterar sobre un objeto, un array por ejemplo, permitiendo traer todos los datos de ese array de manera simultánea, especialmente útil por ejemplo a la hora de hacer consultas en BBDD. Permite también hacer una lectura retrógrada con el comando .reverse\_each

```
arr = ["Mane, Carolina, Claudia, Marco, Sebastián, Eduardo, Freddy"]
arr.each do |i|
  puts i
end

nums = %w[uno dos tres cuatro cinco seis siete ocho]
str = " "
nums.reverse_each { |nums| str += "#{nums} " }
nums.each { |nums| str += "#{nums} " }
```

## switch case

El switch permite trabajar con casos (case), se pueden ejecutar diferentes respuestas de acuerdo al caso.

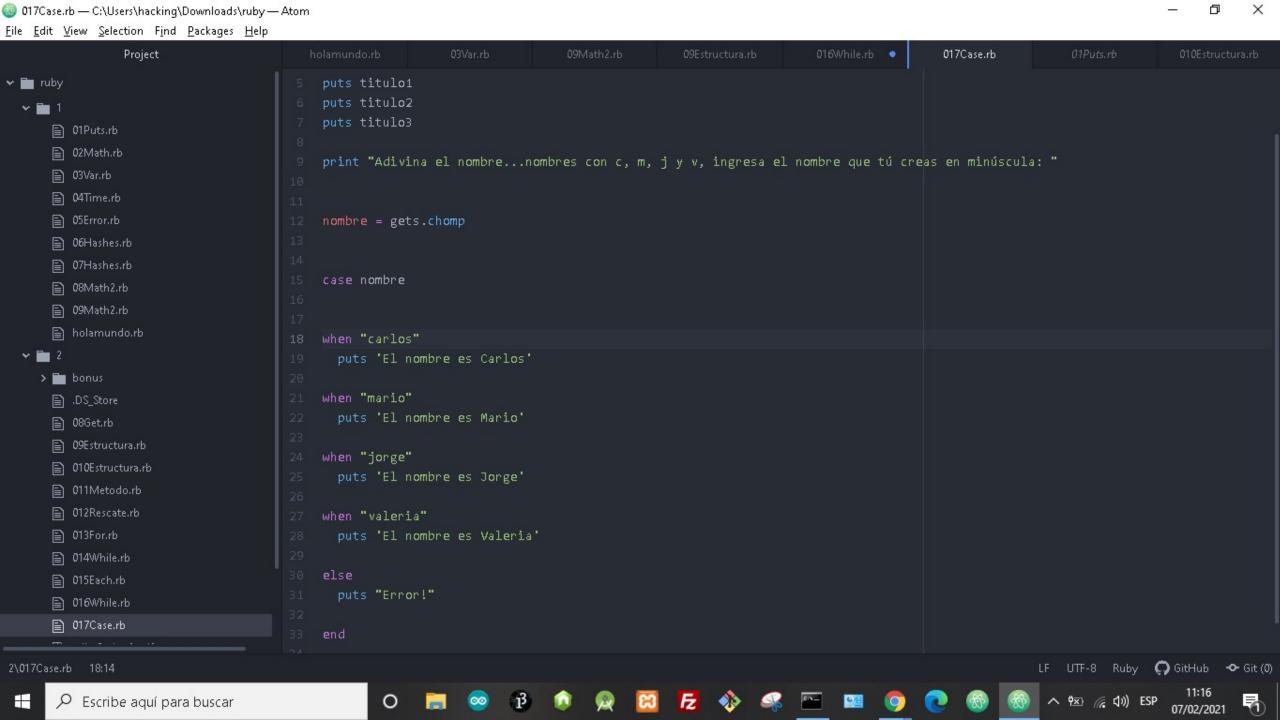
Switch

case

when

else #en caso que no coincida ningún caso

end



## for

El ciclo for ejecutará una instrucción mediante la evaluación de una condición

En un lenguaje tradicional el for sería\_

for (int 
$$x = 0$$
,  $x < 10$ ,  $x+1$ )

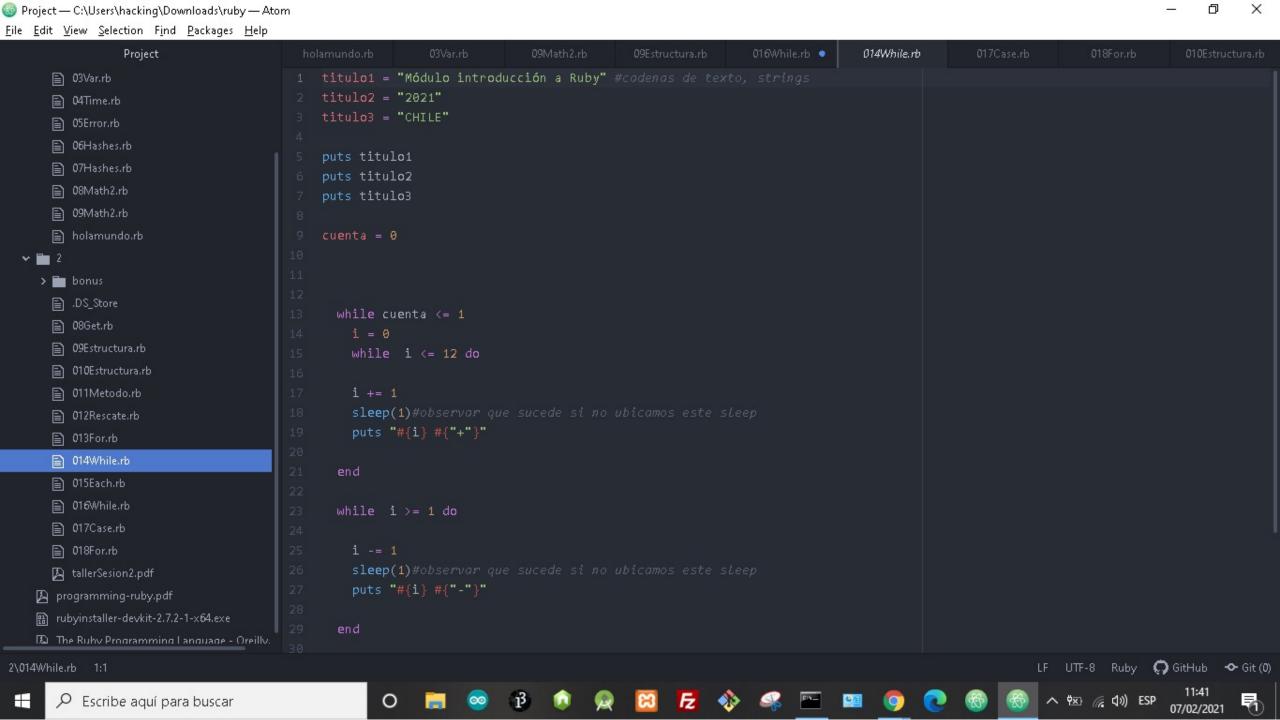
Donde x está inicializada con valor 0, luego se evalúa, x es menor que 0, si es verdad le suma 1 a x, por tanto x ahora vale 2, hará esta operación hasta que x sea menor a 10, luego terminará el proceso. El for en ruby establece rangos, es un poco diferente a lo que estábamos habituados.

# **Tiempo**

Ruby también permite trabajar con tiempo, especialmente útil para automatizar rutinas, ciclos o iteraciones.

## Ciclos anidados

Como cualquier otro lenguaje de programación es posible anidar ciclos, en el siguiente ejemplo revisaremos cómo anidar un while



### Material complementario de la unidad

Link a video relacionado

1. <a href="https://www.youtube.com/watch?v=qDdbArXJklg">https://www.youtube.com/watch?v=qDdbArXJklg</a>

Link a lectura complementaria

1. <a href="https://docs.microsoft.com/es-es/dotnet/visual-basic/programming-guide/language-features/control-flow/loop-structures">https://docs.microsoft.com/es-es/dotnet/visual-basic/programming-guide/language-features/control-flow/loop-structures</a>

Link a investigación relacionada

1.https://www.rubyguides.com/ruby-tutorial/loops/





