Conditionals

Outline

1. Conditional introduction
2. If statement
3. Unless
4. Case when
5. Case in

1. Conditional introduction

A conditional Branch takes the result of a test expression and executes a block of code depending whether the test expression is true or false:

- false and nil is evaluated FALSE
- Otherwise, it is TRUE

Note: the number zero (0) is considered true, whereas many other programming languages consider it false

In many popular programming languages, conditional branches are statements. In Ruby, however, conditional branches are expressions

2. If statement

```
#Syntax
if conditional [then]
  # write something here...
[elsif conditional [then]
  # write something here...]
[else
  # write something here...]
end

If modifier syntax (short if)
[code goes here] if condition
```

```
def demo_if_statement x = 1
  if x >= 2
    p "x is greater than 2"
  elsif x <= 2 and x != 0
    p "x is 1"
  else
    p "I can't guess the number"
  end
end

demo_if_statement 4

demo_if_statement 1

demo_if_statement 0
```

3. Unless

```
#Syntax
unless conditional [then]
  code
[else
  code]
end

Unless modifier syntax (short unless)
[code goes here] unless conditional
```

```
#Example 1
x = 1
unless x >= 2
  puts "x is less than 2"
else
  puts "x is greater than 2"
end

#Example 2
$var =  1
puts "1 -- Value is set\n" if $var
puts "2 -- Value is set\n" unless $var

#Example 3
$var = false
puts "3 -- Value is set\n" unless $var
```

4. Case when

```
#Syntax
case expression
[when expression [, expression ...] [then]
  # write something here...]
[else
  # write something here...]
end
```

```
#Example
$age =  5
case $age
when 0 .. 2
  puts "baby"
when 3 .. 6
  puts "little child"
when 7 .. 12
  puts "child"
when 13 .. 18
  puts "youth"
else
  puts "adult"
end
```

5. Case in

```
#Syntax
case <expression>
in <pattern1>
  ...
in <pattern2>
  ...
in <pattern3>
  ...
else
  …
end
```

```ruby
#Example
case ["a", 1, "b", "c", 2, "d", "e", "f", 3]
in [*pre, String => x, String => y, *post]
  p pre  #=> ["a", 1]
  p x    #=> "b"
  p y    #=> "c"
  p post #=> [2, "d", "e", "f", 3]
end
```

References

- http://ruby-doc.org/core-3.1.0/doc/syntax/control_expressions_rdoc.html
- https://github.com/awesome-academy/RubyExample_TFW

Question & Answer?