Web Application Architectures

Module 4: The Ruby Programming Language
Lecture 5: Expressions and Control Structures

Expressions

- The Ruby syntax is expression-oriented.
- Everything in Ruby is treated as an expression and therefore evaluates to something.

  Ex. Control structures for conditional execution or looping, which would be treated as statements in other languages, are treated as expressions in Ruby.

  In Ruby, `if`, `case` and `for` structures return the value of the last expression evaluated within the structure.

Control Structures – Conditional Execution

- Ruby has a rich syntax for expressing conditionals – the most basic is:

```
if expression
  code
end
```

  where *code* is executed if and only if the conditional *expression* evaluates to something other than `false` or `nil`.

- Else clauses can be added to specify code that should be executed if the conditional expression is not true:

```
if expression1
  code
elsif expression2
  code
else
  code
end
```

Control Structures – Conditional Execution

- There's a shorthand way of expressing the `if` conditional that treats it as an expression modifier:

  code `if` *expression*

- Ruby also has a `?:` operator, as in C/C++.

- Comparison operators:

  `==,  !=,  =~,  !~,  ===`

- There is a `case` structure in Ruby, `===` is the case-equality operator.

Control Structures – Conditional Execution

- In addition to the "standard" set of conditionals, Ruby has added some that are intended to increase the readability/understandability of code. E.g., the following is the opposite of an if statement:

```
until expression
  code
end
```

  where *code* is executed until the conditional *expression* evaluates to something other than `false` or `nil`.

- You cannot attach else clauses to the until conditional.

Control Structures – Iteration

- The `for/in` loop iterates over an enumerable collection:

```
for var in collection do
  body
end
```

- Exit condition loop:

```
while condition do
  body
end
```

- Exit condition loop, opposite of while:

```
until condition do
  body
end
```

- In Ruby, it's more common to use iterators (next lecture).