

# BEWD - COLLECTIONS & LOOPS

**COLT STEELE** 

#### **AGENDA**

- Iteration Loops
- Collections
  - Arrays
  - Hashes

# ITERATION REPETITION REPETITION REPETITION

### ITERATION (RUBY-ESQUE LOOPS)

#### TIMES ITERATOR

```
3.times do
    puts "going. . ."
end
puts "gone"

# going...
# going...
# going...
# going...
# gone
```

### ITERATION (RUBY-ESQUE LOOPS)

#### .UPTO

```
1.upto(3) do |num|
    puts "#{num}.going"
end
# 1. going
# 2. going
# 3. going
```

### ITERATION (RUBY-ESQUE LOOPS)

#### .DOWNTO

```
3.downto(1) do |guess|
    puts "You have #{guess} guesses left"
end

# You have 3 guesses left
# You have 2 guesses left
# You have 1 guesses left
```

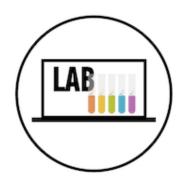
### ITERATION (RUBY-ESQUE LOOPS) LESS COMMON IN RUBY

- These loops are less common in Ruby, but good to know as a programmer.
  - X.times
  - upto
  - downto
- For additional help with syntax, see the Resources at the end of the slides.

#### **CONDITIONAL LOOPS**

```
count = 10
while count > 0
    puts "Looping"
    count -=1
end
```

```
count = 10
until count < 1
    puts "Looping"
    count -= 1
end</pre>
```



#### 99 BOTTLES OF BEER ON THE WALL

### ITERATION RECAP RECAP

- Iteration in programming allows us to keep our code DRY
- Loops are used to repeat lines of code
- Common or Ruby-esque loops are
  - .times
  - .upto
  - .downto
  - .each (we will see in a moment)

### **BREAK!**

# COLLECTIONS WORKING WITH COLLECTIONS IN RUBY

### COLLECTIONS ARRAYS - LISTS

# ARRAYS FIND BY INDEX

```
my_array = ["NYC", "LA", "SYD", "LDN"]
my_array[0] #"NYC"
my_array[1] #"LA"
my_array[-1] #"LDN"
```

### ARRAYS FIND BY POSITION

```
my_array = ["NYC", "LA", "SYD", "LDN"]
my_array.first #"NYC"
my_array.last #"LDN"

# In rails...
# Will not work in IRB
my_array = ["NYC", "LA", "SYD", "LDN"]
my_array.second
my_array.third
my_array.forth
my_array.forth
my_array.forty_two # known as the reddit
```

# ARRAYS ARRAY METHODS

```
name = "charlie"
name.upcase

my_array = ["NYC", "LA", "SYD", "LDN"]
my_array.reverse
```

#### ARRAYS RECAP

- A collection of data
- Can search an array by index or position
- Arrays are objects and therefore have methods.

### COLLECTIONS HASHES - DICTIONARIES

- Each entry in a hash needs a key and a value
- If you access a hash at a specific key, it will return the value at that key

#### HASHES FIND BY KEY

```
ga_markets = {"NYC" => "New York City", "LA" => "Los Angeles
", "SYD" => "Sydney", "LDN" => "London"}

ga_markets["NYC"]
ga_markets["LA"]
ga_markets["SYD"]
```

"New York City"
"Los Angeles"
"Sydney"

### HASHES SETTING VALUES

```
user_hash = {}
user_hash["name"] = "Salman"
user_hash["favorite_color"] = "Green"
user_hash
>> {"name"=>"Salman", "favorite_color"=>"Green"}
```

### SYMBOLS NEW RUBY TYPE

- A symbol is a special type of object in ruby, used extensively
- It is always preceded by a colon
- Cannot contain spaces or numbers
- Symbols are used because:
  - they are immutable and take less memory
  - they are easier to compare to other objects
  - they are cleaner in syntax
- Examples:
  - :hello
  - :this is a symbol

### SYMBOLS PRIMARILY USED AS KEYS FOR HASHES

```
ga_markets = {}
ga_markets = {:NYC => "New York City"}
ga_markets[:LA] = "Los Angeles"
ga_markets
>> {:NYC => "New York City", :LA => "Los Angeles"}
```

#### HASH METHODS

```
user = {:user_name => "SalmanAnsari", :email => "salman.ansa
ri@gmail.com"}

user.has_key? :email #true
user.key? :email #true
user.include? :email #true
user.has_value? "SalmanAnsari" #true (note: extremely ineffi
cient!)
```

#### HASH RUBY 1.9+ ALTERNATE SYNTAX

```
user = {:user_name => "SalmanAnsari", :email => "salman.ansa
ri@gmail.com"}

# becomes

user = {user: "SalmanAnsari", email: "salman.ansari@gmail.co
m"}

# a little bit more concise
# more closely matches JSON format
# considered an 'alternate' syntax, not a replacement
```

### COLLECTIONS ARRAY OF HASHES

### ITERATING OVER COLLECTIONS

#### .EACH

```
ga_markets = ["NYC", "LA", "SYD", "LDN"]
ga_markets.each {|market| puts market}
```

# LAB TIME REDDIT DATA EXERCISE

#### **HOMEWORK**

Finish at least 3 exercises in exercises.md.

### RESOURCES: COLLECTIONS, LOOPS AND APIS CHEAT SHEET

**ARRAYS** 

**Creating Arrays** 

```
my_array = ["Apples", "Oranges", "Pears"]
                ["Apples", "Oranges", "Pears"]
my array = Array.new
Array.new(3)
                         [nil, nil, nil]
Array.new(3, "BEWD")
                ["BEWD", "BEWD", "BEWD"]
```