Ruby Scripting language

CONCEPTS

> Everything is an object and almost every call takes an optional block: 3.times { |i| ... }.

RUNNING RUBY

`ruby -e '<code>'`
Execute code from the command line

`ruby <file>.rb [args]`
Run a script (args in ARGV)

`ruby -c <file>.rb`
Syntax check only

`ruby -w <file>.rb`
Enable warnings

`ruby -r <lib> -e '<code>'`
Require a library first

`ruby -I <dir> <file>.rb`
Add a dir to the load path

`ruby -n -e '<code>'`
Run code for each input line

`ruby -p -e '<code>'`
Like -n, then print each line

`ruby -lane 'puts $F[0]'`
Split lines into fields in $F

IRB

`irb`
Start the console

`ls <obj>`
List methods and variables of an object

GEMS

`gem install <name>`
Install a gem

`gem list`
List installed gems

SYNTAX

`puts <x>`
Print with a newline

SYNTAX (cont.)

`p <x>`
Print inspect form, returns the value

`name = "#{first} #{last}"`
Interpolate into a double-quoted string

`def <name>(a, b = 1, *rest, k:, **opts)`
Define a method

`if <a> ... elsif ... else ... end`
Conditional

`{ a: 1, "b" => 2 }`
Hash literals

COLLECTIONS

`arr.each { |x| ... }`
Iterate

`arr.map { |x| x * 2 }`
Transform each element

`arr.select { |x| x.even? }`
Keep matches (reject inverts)

`h.each { |k, v| ... }`
Iterate pairs

PATTERN MATCHING

`case x in { name: String => n, age: } ... end`
Match and bind a hash

CLASSES, STRUCT & DATA

`attr_reader :a / attr_accessor :a`
Generate getters / getters+setters

ERRORS & REQUIRES

`begin ... rescue Err => e ... ensure ... end`
Handle errors

`raise ArgumentError, "<msg>"`
Raise an exception

`require "<lib>"`
Load a library or gem