Pragmatic Studio : Ruby Programming

Modules

Day 1: 1. Intro

Day 2 2. Running Rb

3. Numbers & Strings

4. Variables & Objects

5. Self

6. Methods

7. Classes

8. Attributes

9. Arrays

10. Objects interacting

11. Separate source files

12. Unit testing

13. Conditionals & TDD

14. Modules   Day 3

15. Blocks

16. Symbols & Structs

17. Hashes

18. Custom Iterators

19. Input / Output

20. Inheritance

21. Mixins

22. Distribution

23. Wrapping up.

1.1.

yellow highlight for sections

+ Set the expectations
  + build a Rb app, from start to finish
  + Rb : OO programming language
    + Classes, objects, methods, etc
  + Understand what makes Rb different
    + discuss techniques
    + design principles
+ How is this course taught?
  + module w/ specific objective : video + exercise
  + Listen, watch, practice writing Rb code

1.2 Intro to the Game

unique player types

+ Where is this Project heading?
  + text-based Game

A loaded die

Who wins?

  • Player : name, health
  • Roll a die : the player's health changes.
  • Find treasures to collect points
  • Test using RSpec

green highlight for concepts

  • Game statistics
  • Package as a gem for distribution
+ a Playlist of Movies

2. Running Ruby ⌐ in `irb`
                 ⌐ in script files (*.rb) ← program file

- **assign** the first **variable** : greeting
- **call** the first **method** : upcase
- **use** the first **class** : Time
- write the first **loop** : 3.times

(pink highlight for terminology jargon)

Daily Plan : Online Ruby Course in ~~2~~ 3 Days

- the key to learning anything new
  - consistent, deliberate practice
  - having an actual plan to follow can help.

Next: add the first player to the game.

VS Code extensions + deps

💡 debugger?

1. Ruby — (format), (lint) etc
   (RuboCop) — (Reek)

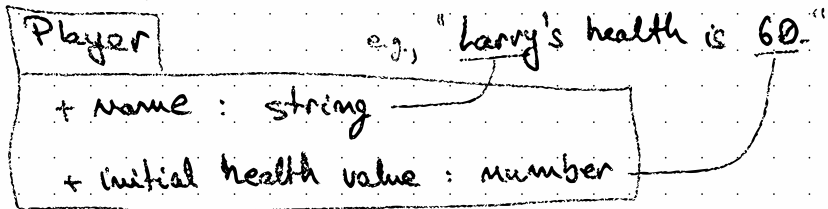2. Solargraph — code completion, inline docs, etc.

3. endwise 💡 extend support for Elixir?

4. Code Runner

Dive Deeper: Programming Ruby (aka Pickaxe)
Chapter 18

3. Numbers & Strings

Product requirement: Every **Game** needs a **Player**

```
┌─────────────────────────────────────────────┐
│ Player              e.g., "Larry's health is 60."│
├─────────────────────────────────────────────┤
│  + name : string                             │
├─────────────────────────────────────────────┤
│  + initial health value : number             │
└─────────────────────────────────────────────┘
```

Alternatively, a **Movie** has a **rank**. And a **title**.

3.1 Numbers
- int : **Fixnum**
- float : **Float**

> Rb casts to the more general type: float > int.

- (Domain) Movie rank = merits - demerits
- (Rb) All numbers are objects => they respond to **messages**

   (e.g., **.class**)

3.2. Strings
- single quoted
- double quoted — more processing

- escape sequences : ****
- string concatenation : **+**
- ! concat String w/ Fixnum ... **Object # to_s**
- string interpolation : **#{ < Rb expression >}**
   ↳ to_s automatically applied

Exercise
- single vs double-quoted strings
- string concatenation
- concat. strings & numbers
- escape sequences
- interpolate Rb expressions in strings.

Next
- objects
- methods

4 Variables & Objects