* **Employ** test-driven development best practices in their labs
  + - Command Line to use
      * rspec (Problem with scrolling up)
        + Command + up/down arrow
      * rspec --fail-fast handle one error at a time
    - How to read test results
      * Red
      * Green
    - Demystifying Rspec
      * This is just Ruby code that you already know.
    - Running a particular test
      * rspec spec/hashketball\_spec.rb
      * rspec spec/hashketball\_spec.rb:28
    - Explain how to solve test step by step.
      * Explain expected:
        + NoMethod error
      * Expected vs Given
* **Distinguish** between data types in Ruby and **Show** how to look up documentation for data types in Ruby
  + - Change from prep work : Array instead of hash
    - Using binding.pry
      * Explain what is binding.pry
    - What is the data Structure we have right now?

{ home: {player: [ ] }, away: {player: [ ] } }

* + - What we want?

[ { } , { }, { }, ... ]

* + - How to look up documentation for data types in Ruby?
* **Demonstrate** the use of common Array methods and **Differentiate** array methods by their respective return values
  + - How to get all players in array?
    - each method
    - map method
    - irb
    - map vs each
    - Build function that gives all players in an array
    - Select
    - Find
    - animals = [‘cat’, ‘dog’, ‘pig’]

numbers = [1, 4, 5, 7, 8]

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **What does it return?** | **What should the block return?** | **What does it look like?** |
| each | nothing | Nothing but prints animal | animals.each do |animal|   puts animal  end |
| map | New array | [ “cute cat”, “cute dog”, “cute pig”] | mapped\_animals = animals.map do |animal|   “cute #{animal}”  end |
| find | Returns first element that meets the condition | “dog” | found\_animal = animals.find do |animal|   animal == ‘dog’  end |
| select | Returns the array with all elements that meets the condition | [4,8] | found\_numbers = numbers.select do |number|   number.even?  end |

* **Define** the Single Responsibility Principle

Here we just want to emphasize that we were breaking down the responsibility of each method to its smallest parts, so that *every method has one job*.