Ruby Notes

# General

* Everything in Ruby is an object
* Constants must start with a capital letter

# Variables

* Snake case is the favored convention in Ruby, i.e. This\_Is\_A\_Method
* Local variables, i.e. variables only available inside a method or the scope in which they were declared, start with a lowercase letter
* Instance variables start with the @ symbol and equate to class-level variables in C#
* Instance variables are declared inside methods, not actually at the class level
* Class-level variables start with @@ and are static variables
* Instance and class-level variables are private by default
* Global variables start with $ and are available from anywhere
* Global variables don’t require a receiver, i.e. you can call $path instead of File.$path

# Strings

* “Text” and ‘Text’ are both valid
* “ recognizes escape sequences like \n
* ‘ just treats the above \n as two separate characters
* “Text #{expression} more text”

# Symbols

* Globally unique named representation of a single location in memory
* They’re much like enums
* Use them when you want a nice name for something but don’t care about the value
* Symbols are denoted with a colon
  + :symbol\_name
* Symbols are unique instances
* Use symbols in place of string identifiers
* It’s recommended to use symbols for unique identifiers instead of strings
  + Collection[:key] instead of Collection[key]

# Arrays

* Dynamic, grow automatically
* Can contain any combination of types, i.e. object[] in C#

# Hashes

* Similar to .NET dictionaries

# Methods

* Can be declared anywhere (inside or outside of a class)
* No void methods, must return a value
  + Result of the last line of the method is the return value if none is specified
* Methods don’t require parameters, and a method without parameters doesn’t need parentheses
* Conventions
  + ? - Method answers a question
  + ! - Method should be used with caution
  + = - Method is a setter

# Classes

* Must begin with an uppercase letter
* Class methods are public and instant by default
* Variables are private by default

# Modules

* Similar to classes, except…
  + Cannot be instantiated
  + Cannot inherit or be derived from
  + Can contain classes, methods, attributes, and other modules