
RUBOCOP

RUBOCOP

- Is a GEM
  - gem install rubocop
- Examples
  - rubocop main.rb
  - rubocop .
  - rubocop . -a

OBJECT ORIENTED PROGRAMMING

OBJECT ORIENTED PROGRAMMING

- Distinction Class vs Object
  - How to define a class
  - What are the conventions?
    - Class Name vs File Name
- How to define an attribute?
  - How to set an attribute
  - How to get an attribute
- Relationship between the attribute and the instance variable?

OBJECT ORIENTED PROGRAMMING

- How to define a class constructor?
- How to set a variable in a constructor?
- How to make an attribute read only?
- How to make an attribute write only?
- How to hide an attribute?
- How to define a method?
  - How does our method act on instance variables?
  - How to hide a method?

SOLID PRINCIPLES

SOLID

- The goal of good code is to aim for simplicity
- Single Responsibility Principle
  - Design methods with a single logical purpose
    - Avoid writing methods which perform multiple tasks
  - A class should also serve one logical purpose
- This help keep our classes and methods readable and bug free

COMPLEX OO CHALLENGE

OO CHALLENGE

- A shipping company would like to build a simple system to add and remove parcels to their containers for the purpose of delivery. A container has a volume (only settable at construction) and can contain a collection of parcels.
- A parcel will have a unique identifier and dimensions. There are two types of parcels, a cuboidal and cylindrical. A cuboidal parcel has 3 dimensions, height, width and breath. A cylindrical parcel has 2 dimensions, height and radius. Each parcel type can report it's volume.
- A parcel can be added to a container as long as the container has space for the parcel based on the available volume of the container. A parcel can also be removed from the container as long as the parcel was in the container to begin with. The container should be able to report on the available volume based on the number of parcels in the container.
- In your solution think carefully about your object model, design classes that are fit for purpose. Also think about exposition, do not expose methods on attributes that do not require exposition. Also keep your methods short and simple.