OBJECTS AND CLASSES

TODAY'S AGENDA

- Review iteration and APIs
- Learn how to make our own data types in Ruby and use them
- What classes and objects are and why they are necessary.
- Learn how to make more than one code file work together

LET'S GO OVER LAST WEEK'S HOMEWORK TOGETHER

OBJECTS

PROCEDURAL VS OBJECT-ORIENTED PROGRAMMING

- So far, we have done a lot of *procedural* programming.
- Our programs just read from top to bottom, do this, do that, if this is the case, then do something else.
- This gets repetitive, cumbersome and actually dangerous
- For example: Three hashes representing students.

CLASSES

- They represent a type of thing
- They can perform actions
  - Actions are performed with methods you create on the Class
- The can contain their own values (think variables, but just for the class)

CLASSES

- Everything in Ruby has a class. Every data type we have touched is an object. Even `nil` is an object with a class!
- Arrays
- Hashes
- Strings
- Even true or false has a class

CHECKING WHAT A THING IS

- You can always call .class on an object to get its class.
- For example
- `"Im a string".class`
- This would give you `String`

- A class can be thought of like a template, or blue print.
- They contain the basic details of how something should look, but not the final detail
- An object, or instance of a class is when you build that blueprint.
- Think: Cookie Cutter house blueprints are the class, the actual houses when built, are instances of that blueprint

Class =

the abstract idea of 'house'
There are many like it.

Instances of that class =

This house in particular.
There are many like it but
this one is mine.

- Classes start with:
  - `class MyClassName`
- And end with:
  - end

```
class MyNewClass
end
```

- Classes MUST start with a capital letter

You create Objects from classes by...

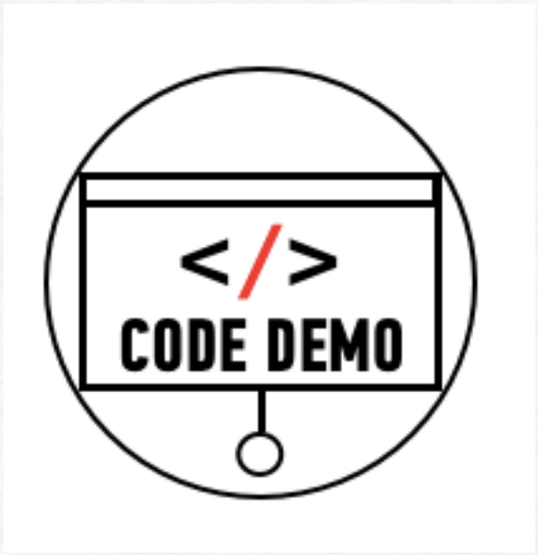
```
class House
end


# instantiate a new object from the class
my_house = House.new
```

DONEC QUIS NUNC

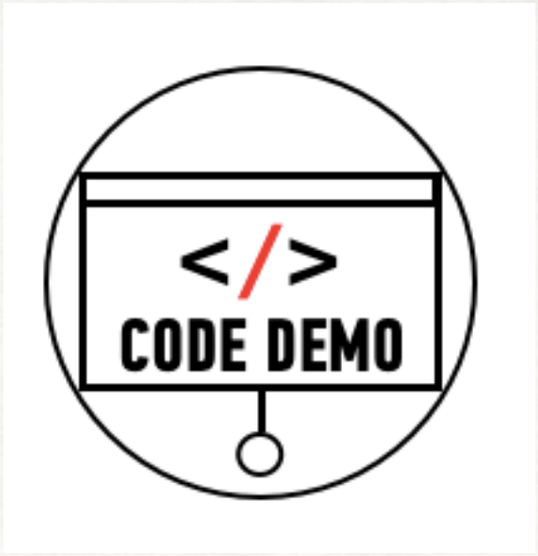
- You can define one or more attributes on a class by typing:
  - attr_accessor :name, :age, :favorite_food
- An attribute is to contain data within an instance of a class
- Think: When we build a house from a blueprint, the color of paint is different per house. It is an attribute of the house.
- So paint color would be an attribute of the house

GETTING AND SETTING ATTRIBUTES

```ruby
class House
  attr_accessor :color
end

# instantiate a new object from the class
my_house = House.new

# set the color of the house with dot notation

my_house.color = 'Red'
my_house.color
=> 'Red'
```



REVISITING OUR STUDENT OBJECTS



Creating a class called “Person” and adding a few attributes of your choosing.

Make two instances of Person and set their attributes.

Work with one or two partners.

INITIALIZE

- This is a special method that determines the parameters for creating new instances with `.new`.
- The parameters passed into this method can set *instance variables*, which correspond to attributes.
- Initialize can also set defaults for new objects.

INITIALIZE

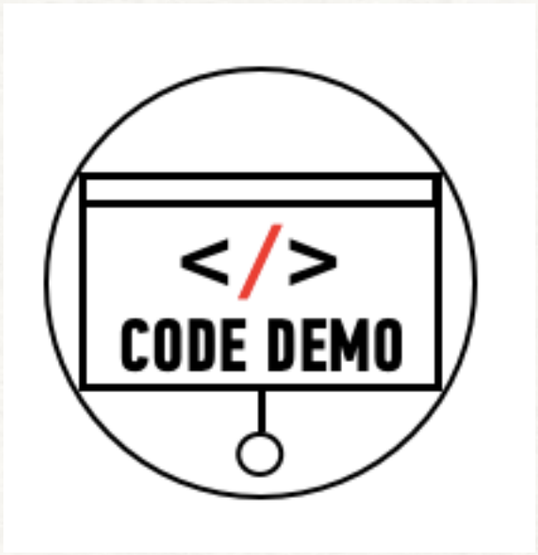
```
class House
  attr_accessor :color, :stories, :rooms

  def initialize(stories, rooms, color)
    @stories = stories
    @rooms = rooms
    @color = color
  end
end

# instantiate a new object from the class
my_house = House.new(2, 3, 'blue'

my_house.color
=> 'blue'
```

INITIALIZE

```
class House
  ..........
  def to_s
     "This #{@color} house has #{@stories.to_s}
      stories and #{ @rooms.to_s } rooms."
  end
end

# instantiate a new object from the class
my_house = House.new(2, 3, 'blue')

puts my_house
"This blue house has 2 stories and 3 rooms."
```

INITIALIZE

```
class House
  ..........

  def paint(new_color)
    @color = new_color
  end
end

# instantiate a new object from the class
my_house = House.new(2, 3, 'blue')

puts my_house
"This blue house has 2 stories and 3 rooms."
```



FURTHER REFACTORING OUR STUDENT OBJECTS

YOUR TURN

- With your partner, refactor your Person class to use the initialize and to_s methods to clean up your code.
- If you don't already, add an attribute for age
- Create a method called birthday which increment's the age of the person that it is done on.

RECAP

- Classes are blueprints, general designs for objects
- You instantiate them with `.new`
- An instance can have it's attributes set and accessed with `attr_accessor`
- They can have their own custom methods defined inside the class.
- Those methods *only work on instances of the right class.*
- The initialize method determines the parameters for creating new instance.
- You can override `to_s`

INTERMISSION

MAKING APARTMENTS

CODE ALONG

SEPARATE CLASSES, SEPARATE FILES

- Ruby is big on organization and encapsulation
- Your ruby files should only declare 1 ruby class

SOME OF SANDI METZ'S RULES

- Classes can be no longer than one hundred lines of code.
- Methods can be no longer than five lines of code.

SEPARATE CLASSES, SEPARATE FILES

- You should require your files only as you need them from other files
- You can require these files with an internal ruby method require_relative “<relative path>”
- Unlike requiring gems (remember `require 'httparty'` ?), you need to specify the relative path to the file.
- You don’t need ".rb" at the end of the file path if you don’t want to.

APARTMENT MANAGEMENT LAB

APTMGMT

- In main.rb, create an empty array called `apartments`
- Create a method to create apartments in main.rb. In the same method, push the newly created array into `apartments`
- Use user input with gets to solicit attributes for new apartments.
- Try a while loop to continue soliciting user input for new apartments (look at last week's homework for inspiration).
- Make another method for printing all of the available apartments. Give the user an option to ask for this.

APTMGMT

- If you finish that, move person.rb into the lib folder.
- make a method in main.rb to create people.
- Make another method that places a Person object as the renter of the apartment, rather than a string.
- Wows! You have two classes of objects, sending each other methods. Object oriented fun!

READING FOR TONIGHT/TOMORROW

“
HTTP://
RUBY.BASTARDSBOOK.COM
/CHAPTERS/OOPS/
”