First Assignment

“Plantaro”

Coder Academy

John Rubio

Juan Manuel Sánchez-Ortiz

[Figure Index 4](#_Toc2764324)

[Content 5](#_Toc2764325)

[- Code (and #pseudocode) 5](#_Toc2764326)

[- Gems reference 10](#_Toc2764327)

[Rainbow 10](#_Toc2764328)

[Terminal-Table 10](#_Toc2764329)

[Tty-font 11](#_Toc2764330)

[Catpix 11](#_Toc2764331)

[- Workflow 11](#_Toc2764332)

[- User Stories: 12](#_Toc2764333)

[As George 12](#_Toc2764334)

[As Brain 12](#_Toc2764335)

[As Georgina 12](#_Toc2764336)

[As Betty 13](#_Toc2764337)

[- Project Management Tools 14](#_Toc2764338)

[o Slack 14](#_Toc2764339)

[o Trello 14](#_Toc2764340)

# Figure Index

[Figure 1. Future App Workflow 11](#_Toc2764341)

[Figure 2. Current App Workflow 12](#_Toc2764342)

[Figure 3.User Stories(a) 13](#_Toc2764343)

[Figure 4. Slack 14](#_Toc2764344)

[Figure 5. Trello 14](#_Toc2764345)

# Content

* Code (and #pseudocode)**:**

**require** 'date' #we require `date´ so our program can calculate the age of the plant.

$:.unshift File.dirname(**\_\_FILE\_\_**) + '/../lib'

**require** 'terminal-table/import'

**require** 'rainbow'

**require** 'rainbow/refinement'

using Rainbow

**require** "tty-font"

font = TTY::Font.new(:doom)

*# require 'catpix'*

**class** Plant

attr\_accessor :birthday, :age *#just for testing*

**def** initialize(username, name, birthday = Date.new(1993, 04, 16))

@owner = username

@name = name

@happiness = 20

@birthday = birthday

@age = (Date.today - @birthday).to\_i

@pest = **false**

@messages = {

:happy100 => "#{@name} is vigorous! Like a beanstalk",

:happy75 => "#{@name}'s stem is looking thick and muscular",

:happy50 => "#{@name} is chillin like a plant",

:happy30 => "#{@name} is feeling standard...",

:happy20 => "Mate! #{@name} is sad and weak!",

:dead => "#{@name} died! Rest In Plant... Good plant, better person",

:water => "#{@name} is glistening! sparkle sparkle.",

:sun => "#{@name} is getting a nice healthy tan.",

:song => "#{@name} is reacting to your angel-like voice.",

:healed => "You saved #{@name} from ferocious pests, Hero!!",

:overwater => "Too much water! #{@name} is not the Titanic!",

:oversun => "Too much sun! #{@name} became a tomato",

:poison => "Too much pesticide! #{@name} looks diseased!",

:noisy => "Don´t shout. #{@name} is scared!"

}

@last\_action = []

**end**

#Then we defined the initialize method according to which each plant created would have an owner, a name, a happiness rate starting from 20, a birthday and an age, a pest status (if there are insects or not), and a list of messages that will be display according to user´s decisions.

**def** check\_happiness

info = "Your plant's name is #{@name}, " +

"this fella was born on #{@birthday}, " +

"thanks to you #{@owner}:)"

puts "#{info} \nPlant status:"

**if** @happiness >= 100

puts @messages[:happy100]

**elsif** @happiness >= 75

puts @messages[:happy75]

**elsif** @happiness >= 50

puts @messages[:happy50]

**elsif** @happiness >= 30

puts @messages[:happy30]

**elsif** @happiness < 30

puts @messages[:happy20]

**end**

**end**

**def** water

**if** @last\_action[-1] == @last\_action[-2] && @last\_action[-1] == "water"

# ((@last\_action[-2] == @last\_action[-3] || @last\_action[-1] == @last\_action[-2]) && @last\_action.size > 1) #when there have been carried out more than two actions, what takes into account the program is the previous two actions

@happiness -= 10

puts @messages[:overwater]

**else**

@happiness += 10

puts @messages[:water]

**end**

@last\_action.push "water"

**end**

**def** give\_sun

**if** @last\_action[-1] == @last\_action[-2] && @last\_action[-1] == "give sun"

# ((@last\_action[-2] == @last\_action[-3] || @last\_action[-1] == @last\_action[-2]) && @last\_action.size > 1)

@happiness -= 10

puts @messages[:oversun]

**else**

@happiness += 10

puts @messages[:sun]

**end**

@last\_action.push "give sun"

**end**

**def** sing

puts "What do you want to sing to your plant?"

song = gets.chomp

#shouting (writing in upcase) will reduce happiness

**if** song == song.upcase

@happiness -= 10

puts @messages[:noisy]

**else**

#length of song will add bonus points to happiness

@happiness += 10 + song.length

puts @messages[:song]

**end**

@last\_action.push "sing"

**end**

**def** pest?

#33% chance to get infested with bugs, previous action must not be spray

#will not run if no other action has been taken first (newly created)

**if** rand(3) == 1 && @last\_action[-1] != "spray" && @last\_action != []

@pest = **true**

@happiness -= 20

**end**

@pest

**end**

**def** spray

**if** @pest

#spraying plant while infested will return lost points

@happiness += 20

@pest = **false**

puts @messages[:healed]

**else**

#if plant is not infested with bugs, spray will reduce happiness

@happiness -= 10

puts @messages[:poison]

**end**

@last\_action.push "spray"

**end**

**def** death?

@happiness <= 0

**end**

**end**

puts font.write("plantaro", letter\_spacing: 4)

puts "Give your little plantling a name?"

plant\_name = gets.chomp

plantaro = Plant.new "John", plant\_name

*#puts plantaro.birthday*

*#puts plantaro.age*

*#program loop*

loop **do**

**if** plantaro.death?

puts "Your plant has withered away dues to poor plant-caring skills :("

**break**

**end**

**if** plantaro.pest?

puts "Oh no! Your plant is infested with bugs. Kill them quick!!"

**end**

#options list

puts "", "What would you like to do?"

*# puts "-Water\n-Give Sun\n-Sing\n-Spray Pests\n-Status\n-Quit", ""z*

puts

option\_table = table **do**

**self**.headings = ['Option','Description']

add\_row ["Give sun", 'You will give sun to your plant so it can do photosynthesis']

add\_row :separator

add\_row ['Sing', "You know singing to plants helps give them Carbon Dioxide?"]

add\_row :separator

add\_row ['Spray Pests', "You will kill the insects annoying your plant"]

add\_row :separator

add\_row ["Water", "You will water your plant so it won´t dehidrate"]

**end**

puts option\_table

input = gets.chomp

**if** input == "quit"

puts "Your plant has lived great life at #{plantaro.age} days old.",

"Time to say goodbye :("

**break**

**end**

**case** input.downcase

**when** "water"

plantaro.water

**when** "give sun"

plantaro.give\_sun

**when** "sing"

plantaro.sing

**when** "spray pests"

plantaro.spray

**when** "status"

plantaro.check\_happiness

**else**

puts "Input invalid"

**end**

**end**

* Gems reference**:**

### Rainbow

* + Documentation: https://github.com/sickill/rainbow
  + Tutorial: <http://ku1ik.com/2009/03/24/colorizing-console-output-with-rainbow-ruby-gem.html>

### Terminal-Table

* + Documentation/examples.rb: <https://github.com/tj/terminal-table/blob/master/examples/examples.rb>

### Tty-font

* + Documentation/examples.rb: <https://github.com/piotrmurach/tty-font/blob/master/examples/star_wars_logo.rb>

### Catpix

* + Documentation: <https://github.com/pazdera/catpix>
  + Help: <https://www.rubydoc.info/github/pazdera/catpix/master/Catpix.print_image>
  + Imagemagick (library): <https://stackoverflow.com/questions/7053996/how-do-i-install-imagemagick-with-homebrew>
* Workflow:



Figure 1. Future App Workflow



Figure 2. Current App Workflow

## User Stories:

As George:

• I want:

o a pet but cant have the responsibility of a real one.

o a several days-term challenge so that I can train my patience.

o to take care of a bein so that I can feel useful.

o to follow a trend, so that I can talk with my classmates about something.

o to have a side distraction so that I can take a break from study.

As Brain:

• I want:

o a plant but my girlfriend doesn't want one, so I need a substitute.

o a game which follows a logic so that I can play without feeling frustrated.

As Georgina:

• I want:

o to take care of something so that I can feel motherly.

o to know how to use a tech device so I do not feel overwhelmed.

o to be able to see what I am using so that I can deal with it.

As Betty:

• I have a lot of love to give but no one to give it to.

Imagen que contiene texto, pizarra

Descripción generada automáticamente

Figure 3.User Stories(a)

* Project Management Tools:
  + Slack:

Imagen que contiene captura de pantalla

Descripción generada automáticamente

Figure 4. Slack

* + Trello:

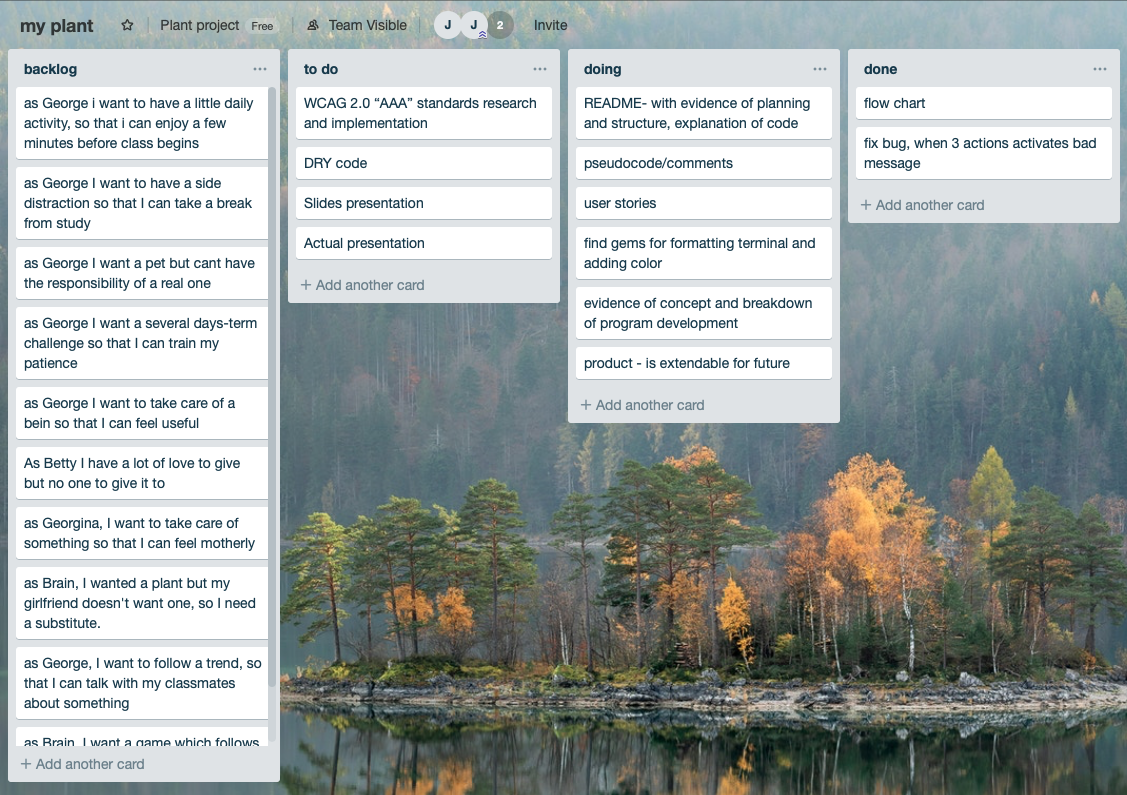


Figure 5. Trello