Market Scraper:

require "MarketScraper/version"

module MarketScraper

  class Error < StandardError; end

*# Your code goes here...*

  class InvalidType < StandardError; end

end

Portfolio

require pry

class Portfolio

    extend MarketScraper

    attr accessor :name, :value

    attr\_reader :stocks

    @@all = []

    def initialize(name)

        binding.pry

        @name = name

        @stocks =[]

        save

    end

    def save

        @@all << self

    end

    def self.clear\_all

        @@all.clear

    end

    def add\_stock(stock)

        if !stock.is\_a(Stock)

            raise InvalidType, "Invalid stock type, stock must be a Stock!"

        else

            @stocks << stock

            @value += stock.total\_value

        end

    end

    def stock

        @stock.dup.freeze

    end

    def value

        total = []

        self.stocks.each do |stock|

            total += stock.total\_value

        end

        total

    end

*#stores an array of included stocks in a given portfolio*

*#aggregates the value of the portfolio based on the number of each stock and their values.*

end

Scraper

class Scraper

    extend MarketScraper

    def initialize (symbol)

        @doc = Nokogiri::Html(open("https://finance.yahoo.com/quote/#{symbol)}"))

        stock.price = doc.search(*#quote-header-info > div.My\(6px\).Pos\(r\).smartphone\_Mt\(6px\) > div.D\(ib\).Va\(m\).W\(65\%\).W\(60\%\)--tab768.Ov\(h\) > div > span.Trsdu\(0\.3s\).Fw\(b\).Fz\(36px\).Mb\(-4px\).D\(ib\))*

        Stock.new(name)

    end

*# will take in a symbol provided by user and get the data*

*# will parse the info and call the Stock class*

*# Can be called to check the valye of the stock*

end

Stock

class Stock

    extend MarketScraper

    attr\_acessor :name, :symbol, :value, :shares, :portfolio

    @@all =[]

*# will instantiate the stock*

    def initialize (symbol, shares = 1)

        @name = name

        @symbol = symbol

        @value = value

        save

    end

    def save

        @@all << self

    end

    def self.all

        @@all

    end

    def get\_info

        Scraper.new(symbol)

    end

*#will send the created stock to the portfolio class*

    def add\_to\_portfolio(portfolio\_name)

        portfolio\_name.add\_stock(self)

    end

    def self.clear\_all

        @@all.clear

    end

    def total\_value

        total = self.value \* self.shares

    end

    def self.find\_or\_create\_stock\_from\_symbol(symbol)

        if self.find\_by\_symbol(symbol)

        else

            self.create

        end

    end

    def self.find\_by\_symbol(symbol)

        self.all.find( |i| i.symbol = symbol)

    end

*# will store the stock and relevant information*

end

Market Scraper exe

    def start

      puts "Hello World!"

      puts "-----------------"

      puts "How may we be of assistance today?"

      puts "Would you like to:

          1) Create a new portfolio.

          2) Review an existing Stock or Portfolio?

          3) Add a new stock or update a Portfolio?"

      input = gets.strip.to\_i

      case input

      when 1

          puts "What would you like the name of the new portfolio to be?"

          input = gets.chomp

          Portfolio.new(input)

          puts "Would you like to add a stock to the new portfolio? Y/N"

          value = gets.chomp

          until value == "N"

              "What stock would you like to add to the new portfolio?"

              stock = gets.chomp

              if name.add\_stock(stock)

              end

          end

      end

  end

start