**E620A. Refactoring - Miscellaneous helper classes**

Class:

helpers/charts\_helper.rb (95 lines)

helpers/file\_helper.rb (91 lines)

helpers/leaderboard\_helper.rb (144 lines)

helpers/submitted\_content\_helper.rb (98 lines)

What needs to be done:

Methods 'get\_pie\_chart\_url' and 'get\_bar\_chart\_url' in charts\_helper.rb contain duplicate lines of code. Refactor both methods to avoid duplication.

In methods 'create\_directory' and 'create\_directory\_from\_path' there is code duplication. Refactor the methods and get rid of duplicate lines of code.

The naming convention for methods and variable names in ruby should follow snake case e.g. 'getCourseName' should have been named as 'get\_course\_name'. Refactor the method names and variable names in the leaderboard\_helper.rb and submitted\_content\_helper.rb files so that it follows the correct convention.

Method 'index' in leaderboard\_helper.rb is too long and must be refactored using 'extract method' functionality into smaller methods focussing on specific functionality.

1.charts\_helper.rb

(1).put the removing last character in a method

before refactor:

#value\_string = value\_string[0..-2] # remove last "|"

#label\_string = label\_string[0..-2] # remove last "|"

#color\_string = color\_string[0..-2] # remove last comma

after refactor:

value\_string=cut\_length\_one(value\_string)

label\_string=cut\_length\_one(label\_string)

color\_string=cut\_length\_one(color\_string)

(2).put the functionality of getting address in a method

before refactor:

#i = 0

#for value in values

# unless value == 0

#

# value\_string += (value.to\_f/max).to\_i.to\_s + ","

# label\_string += labels[i].to\_s + "|"

# color\_string += self.get\_rg\_color(i+0.5,values.length) + ","

# end

# i += 1

#end

#cut\_length\_one(value\_string)

#cut\_length\_one(label\_string)

#cut\_length\_one(color\_string)

#value\_string = value\_string[0..-2] # remove last comma

#label\_string = label\_string[0..-2] # remove last "|"

#color\_string = color\_string[0..-2] # remove last comma

#address += value\_string + label\_string + color\_string

after refactor:

address=get\_address(address,value\_string,label\_string,color\_string,ifpie)

2.file\_helper.rb

1. Create a method to catch the exception in creating dirctionary.

before refactor:

#begin

# if !File.exists? path

# FileUtils.mkdir\_p(path)

# end

#rescue PathError

# puts $!

#rescue

# puts $!

# raise "An error was encountered while creating this directory: "+$!

#end

After refactor

Check\_file\_exists(path, false)

put the code above in a method:

def check\_file\_exists(object\_path,ifobject)

begin

if ifobject

if !File.exists? object.get\_path

FileUtils.mkdir\_p(object.get\_path)

end

else

FileUtils.mkdir\_p(object)

end

rescue PathError

puts $!

rescue

puts $!

raise "An error was encountered while creating this directory: "+$!

end

end

3.leaderboard\_helper.rb

(1) refactor the name of methods to follow snake case.

Variables:

courseID -> course\_id

courseName -> course\_name

achieveName -> achieve\_name

assignmentID -> assignment\_id

assignmentName -> assignment\_name

requesterID -> requester\_id

userID -> user\_id

userName -> user\_name

instructorRoles -> instructor\_roles

@courseList -> @course\_list

assignPartList -> assign\_part\_list

participantEntry -> participant\_entry

courseTuples -> course\_tuples

courseHash -> course\_hash

userid -> user\_id

assignmentid -> assignment\_id

userGradeArray -> user\_grade\_array

csEntry -> cs\_entry

top3LeaderBoard -> top3\_leaderboard

Methods:

self.getcoursename -> self.get\_course\_name

self.getachievename -> self.get\_achieve\_name

self.getassignmentname -> self.get\_assignment\_name

self.getusername -> self.get\_user\_name

self.userisinstructor? -> self.user\_is\_instructor?

self.studentinwhichcourses -> self.student\_in\_which\_courses

self.instructorcourses -> self.instructor\_courses

self.gettop3leaderboards -> self.get\_top3\_leaderboards

self.dumpcstable -> self.dump\_cstable

4.leaderboard\_helper.rb

1. refactor the if statement of inferring whether the path is a directory

#if File.directory?(file)

#ret += "\n <a title='Expand/Collapse' href='#' onclick='javascript:collapseSubDirectory(#{index}); return false;'><img id='expand.#{index}' alt='Expand/Collapse' title='Expand/Collapse' src='/images/up.png'></a>&nbsp;"

# ret += link\_to File.basename(file), :controller => 'submitted\_content', :action => 'edit', :id => participant.id, "current\_folder[name]" => file

#ret += list\_sub\_directories(file, participant)

#else

# ret += "\n "

# parentFolder = File.dirname(file)

# if parentFolder != participant.get\_path

# parentFolder.sub!(participant.get\_path+"/","")

# parentFolder += "/"

# else

# parentFolder = ""

# end

# location = parentFolder + File.basename(file)

# ret += link\_to location, :controller => 'submitted\_content', :action => 'download', :id => participant.id, :download => File.basename(file), "current\_folder[name]" => File.dirname(file)

#end

after refactor:

ret=check\_directory(file,ret,participant,index)

(2) add file attributes to ret

#ret += "\n </td>\n <td valign = top>\n"

#ret += File.size(file).to\_s

#ret += "\n </td>\n <td valign = top>\n"

#ret += File.ftype(file)

#ret += "\n </td>\n <td valign = top>\n"

#ret += File.mtime(file).to\_s

#ret += "\n </td>\n </tr>"

after refactor:

ret=add\_file\_attribute(ret,file)