SCALE FOR PROJECT PISCINE PHP (/PROJECTS/PISCINE-PHP) / DAY 06 (/PROJECTS/42-PISCINE-C-FORMATION-PISCINE-PHP-DAY-06)

Guidelines

Preliminaries are important!

- Is everybody present?
- Only grade the work that is in the student or group's GiT repository.
- Chrome as well as curl must be used for this defence (the version available on the dumps).

If there is nothing in the repository put 0 and stop evaluation.
Discuss what went wrong (not necesseraly on the technical side) and how to avoid same situalion tomorrow.

It is mandatory to have all point for an exercise to move to the next. The first exercise that doesn't get all the points stops the evaluation.

The following points stop the evaluation and give a 0:
- To use forbidden PHP parts, expecially notions from d07 and d08.
- An output that differ consequently from the required output (incomplete information, wrong computation, missing feature, etc).
- A public method or attribut without explanation. If there is an explanation but you'recompenser not convinced it's up to you to decide.
- Failing one of the following instructions:
  - Only one unique Class per file.
  - One file that includes the definition of a class cannot include any other except for require or require_once if necessary.
  - A file containing a class must ALWAYS be named ClassName.class.php.
  - A class must ALWAYS be accompanied by a documentation file which MUST be named ClassName.doc.txt.
  - The documentation of a class must ALWAYS be useful and correspond to the implementation.
  - A class must ALWAYS have a static Boolean attribute called verbose.
  - A class must ALWAYS have a static method called doc that returns the documentation in a string.
  - An exercise folder but countain the files from previous exercises, wether they are identical, altered or new.

The following aren't eliminatory:

- To have an output that differs in its formatting (one more or less spase, etc.).
- To have a method or an attribute that doesn't have exactly the same name but keep the same semantic though.
- To solve te problem with a different algorithm than the one explained in the subject, as long as the result is identical.

The Vector Class

The Vector Class

- The Vertor class must have 4 private attributes to represent x, y, z et w.

- The Vector Class must have read only assessors for its four attributes.

- The Class' constructor is waiting for an array. The following keys are required:
- 'dest': the vector's destination vertex, mandatory.
- 'orig': the vector's origin vertex, optional by default is worth a new instance of the x=0, y=0, z=0, w=1 vertex.

- The Vertor Class must have a __toString method which output matched the examples.

- The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

- If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

- The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Vector.doc.txt file.

- Methods from the Vector Class should never modify the current instance.

- The Vector Class must have at least the following public methods and they're fonctional:
- magnitude
- normalize
- add
- sub
- opposite
- scalarProduct
- dotProduct
- cos
- crossProduct

- Running the main_02.php script generates an output like the one in the main_02.out file.

☑ Yes ☒ No

The Render Class

The Render Class

- The Class' constructor is waiting for an array. The following keys are required:
- 'widht' : The generated image's width, mandatory.r
- 'height' : The generated image's height, mandatory.
- 'filename' : Filename in which the png image created will be saved, mandatory.

- La classe Render Class must have three Class constants: VERTEX, EDGE et RASTERIZE that will be used to select the rendering mode.

- The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

- If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

- The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Render.doc.txt file.

- The Render Class must have at least the following public methods and they're fonctional:
- renderVertex
- renderTriangle
- develop

- The VERTEX mode works ==> +1 point
- The EDGE mode works ==> +1 point
- The RASTER mode works ==> +3 point

Rate it from 0 (failed) through 5 (excellent)

Bonus - The Texture Class

Bonus - The Texture Class

Don't try to nitpick on this one. If the project has proper and fully functional (not half with excuses or explanations) textures you can give 10 points.
We're no beasts.

You can discuss about it and go get a coffee.
Congratulations.

☑ Yes ☒ No

The Vertex Class

The Vertex Class

- The Vertex class must have 5 private attributes to represent x, y, z, w and its color.
- The Vertex's color is always an instance of the Color Class from the previous exercise.
- The Vertex Class must have reading and writing assessors for its five attributes.
- The Class' constructor is waiting for an array. The following keys are required:
- 'x': x axis coordinate, mandatory.
- 'y': y axis coordinate, mandatory.
- 'z': axis coordinate, mandatory.
- 'w': optional, by default is worth 1.0.
- 'color': optional, by default is worth a new instance of the color white.
- The Vertex Class must have a __toString method which output matched the examples.
- The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.
- If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.
- The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Vertex.doc.txt file.
- Running the main_01.php script generates an output like the one in the main_01.out file.

☑ Yes ☒ No

The Color Class

The Color Class

- The Color Class must have 3 public integer attributes red, green and blue that will be used to represent to components of a color.
- The Class's constructor requires an array. An instance must be able to be built, either by passing a value for the rgb key which will be split into three red, green and blue components, either by passing a value for the red, green and blue keys which will directly

represent the three components.

- Each of the values for the four possible keys will be converted into an integer before use.

- The Color Class must have a __toString method which output matched the examples.

- The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

- If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

- The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Color.doc.txt file.

- The Class must have a method called add that allows you to add the the components of the current instance to the components of another instance argument. The resulting color is a new instance.

- The Class must have a method called sub that allows you to substract the the components of the current instance to the components of another instance argument. The resulting color is a new instance.

- The Class must have a method called mult that allows you to multiply the components of the current instance to an argument value. The resulting color is a new instance.

- Running the main_00.php script generates an output like the one in the main_00.out file.

☑ Yes ☒ No

The Matrix Class

The Matrix Class

- The Matrix Class must have seven Class constants: IDENTITY, SCALE, RX, RY, RZ, TRANSLATION et PROJECTION.

- The Class' constructor is waiting for an array. The following keys are required:
- 'preset' : the matrix type to create, mandatory. The value must be one of the Class constants previoulsy defined.
- 'scale': the scale factor, mandatory when 'preset' is worth SCALE.
- 'angle': the rotation angle in radians, mandatory when 'preset' is worth RX, RY ou RZ.
- 'vtc': translation vector, mandatory when 'preset' is worth TRANSLATION.
- 'fov': projection field of view in degrees mandatory when 'preset' is worth PROJECTION.
- 'ratio': projected image ratio, mandatory when 'preset' is worth PROJECTION.
- 'near: projection's near clipping plane mandatory when 'preset' is worth PROJECTION.
- 'far': projection's far clipping plane mandatory when 'preset' is worth PROJECTION.

- The Matrix Class must have a __toString method which output matched the examples.

• The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

• If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

• The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Matrix.doc.txt file.

• Methods from the Matrix Class should never modify the current instance.

• The Matrix Class must have at least the following public methods and they're fonctional:
- mult
- transformVertex

• Running the main_03.php script generates an output like the one in the main_03.out file.

☑ Yes ☒ No

The Camera Class

The Camera Class

• The Class' constructor is waiting for an array. The following keys are required:
- 'origin': The vertex positioning the camera in the world mark.
- 'orientation': Rotation matrix orienting the camera in the world mark.
- 'width': Width in pixel of the desired image. Is used to compute the ratio. Not compatible with the 'ratio' key.
- 'height': Height in pixel of the desired image. Is used to compute the ratio. Not compatible with the 'ratio' key.
- 'ratio': Image's ratio. Not compatible with the 'width' and 'height' keys.
- 'fov': The projected image's field of view in degree.
- 'near': The near clipping plane.
- 'far': The far clipping plane.

• The Camera Class must have a __toString method which output matched the examples.

• The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

• If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

• The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Camera.doc.txt file.

• The Camera Class must have at least the following public methods and they're fonctional:

- watchVertex

- Running the main_04.php script generates an output like the one in the main_04.out file.

Yes No

The Triangle Class

The Triangle Class

- The Class' constructor is waiting for an array. The following keys are required:
- 'A' : Vertex of the first point of the triangle, mandatory.
- 'B': Vertex of the second point of the triangle, mandatory.
- 'C': Vertex of the third point of the triangle, mandatory.

- The Triangle Class must have a __toString method which output matched the examples.

- The Class must include a Boolean static attribute called verbose to control the displays related to the use of the Class. This attribute is initially False.

- If and only if the static attribute verbose is true, then the Class constructor and destructor will produce an output.

- The Class must have a static method called doc that returns the documentation of the class in a string. The content of the documentation must be read from a Triangle.doc.txt file.

- There are no mandatory methods for your Triangle Class. However writing a few of them can be useful. If the following methods aren't implemented it's not eliminatory and still still allow access to the bonus part. However it's possible to give more points to reward reflexion. Without methods the exercises is worth 2 points
- To be able to iterate or map the vertices of the triangle. ==> +1 point
- To be able to iterate or map the edges of the triangle. ==> +1 point
- To be able to sort the vertices or the edges under certain conditions. ==> +1 point
- Any additional and Fonctionnalites relevant feature ==> +1 point up to a maximum of 2 points

Rate it from 0 (failed) through 5 (excellent)

Ratings

Don't forget to check the flag corresponding to the defense

✔ Ok

Empty work | Incomplete work | No author file | Invalid compilation

Norme | Cheat | Crash | Forbidden function

Conclusion

Leave a comment on this evaluation

Preview!!!