

Fundación Universitaria Internacional De La Rioja

PREGRADO DE INGENIERÍA INFORMÁTICA

# INTRODUCCIÓN A OPENGL

*Informática Gráfica y Visualización (COLGII) - Enero2022*

Docente:

Rogelio Orlando Beltrán Castro

Autor:

Nicolás Zapata Álzate

Abril de 2022

# Índice general

<b>1. Herramientas utilizadas para el laboratorio</b>	<b>3</b>
1.1. Entorno de desarrollo . . . . .	3
1.2. Organización del proyecto. . . . .	3
1.2.1. Librerías utilizadas. . . . .	3
1.2.2. Árbol del proyecto. . . . .	4
<b>2. Descripción del código</b>	<b>6</b>
2.1. Clase Main. . . . .	6
2.2. Variables de iniciación . . . . .	6
2.2.1. Clase Clock. . . . .	6
2.2.2. Clase Display. . . . .	7
2.2.3. Clase Draw. . . . .	7
<b>3. conclusiones</b>	<b>8</b>
3.1. Resultados y pautas de la actividad. . . . .	8
3.1.1. Resultado. . . . .	8
3.1.2. Pautas del ejercicio. . . . .	8
<b>4. Recursos</b>	<b>10</b>
<b>5. Agradecimientos</b>	<b>11</b>

Cordial Saludo para el docente o quién tenga la autorización de revizar/calificar éste documento, como siempre, agradecido con Dios de poder estar vivo presentando éstos trabajos y poder seguir creciendo en la carrera de Ingeniería Informática.

Espero que el trabajo que estoy presentando cumpla con las pautas y pueda meritar una buena calificación.

El archivo está todo organizado en el apartado de la asignatura, en una carpeta llamada «**app**». De igual manera, se encuentra anexado el link del repositorio de Github de éste mismo proyecto, tanto del proyecto (alojado en la carpeta app) como del proyecto del documento (alojado en la carpeta latex).

# Capítulo 1

## Herramientas utilizadas para el laboratorio

Para el desarrollo de éste laboratorio, hicimos uso de herramientas de desarrollo, en su mayoría, basadas para entornos Linux, ya que, he percatado de que tiene mejor rendimiento y mejores resultados que trabajar dentro del sistema operativo Windows.

### 1.1. Entorno de desarrollo

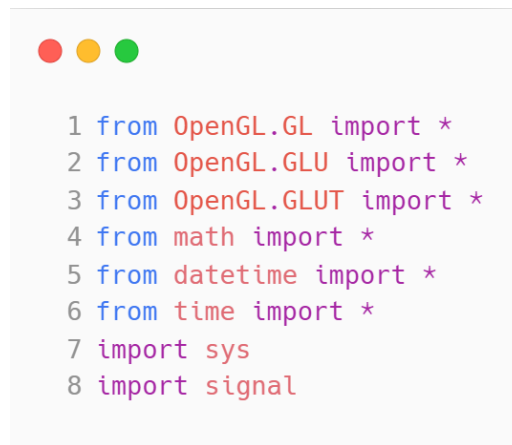
Se usaron las siguientes herramientas.

- WSL para Windows.
- Python 3.
- freegut-dev.
- Neovm
- Anaconda

### 1.2. Organización del proyecto.

#### 1.2.1. Librerías utilizadas.

Como se mencionó en la anterior lista, he decidido prescindir del lenguaje de programación Python para el desarrollo del laboratorio. Puesto que es un lenguaje en la que me he desempeñado en trabajar desde hace un buen tiempo desde que estudio en la universidad.



```
1 from OpenGL.GL import *
2 from OpenGL.GLU import *
3 from OpenGL.GLUT import *
4 from math import *
5 from datetime import *
6 from time import *
7 import sys
8 import signal
```

Figura 1.1: Librerías usadas para el laboratorio

Las librerías más importantes para el laboratorio son las que vienen con PyOpenGL, que vienen con sus clases **GL**, **GLU** y **GLUT**, las cuales ejecutan las funciones y comandos de OpenGL en Python.

En segundo lugar, tenemos la librería de **Time** y **Datetime** para el manejo del tiempo y de las fechas, nos ayuda a saber como está la hora y como ésta va repercutiéndose en el reloj.

Otra librería que no tiene mucho que ver con el uso de OpenGL, pero que, hace que sea más estilizado el programa, es la librería **sys** que, llama comandos de la terminal, como por ejemplo, limpiar la consola, realizar o llamar comandos, entre otras acciones que hacemos dentro de la terminal, y que en Python nos ayuda a ejecutar dentro del programa.

### 1.2.2. Árbol del proyecto.

El archivo en general cuenta con una clase **main** que es la principal, contamos con una carpeta llamada **components**, en donde están los archivos **clock**, **display** y **draw**, la primera, es la clase en donde está trabajando las funciones que tienen que ver con la organización del reloj, la clase **display**, es la clase en donde se encuentra el método para la iniciación de la ventana, que es la gui en donde se visualiza el reloj, por último, está la clase **draw**, en donde están los métodos en donde se hace la manipulación de los elementos del reloj.

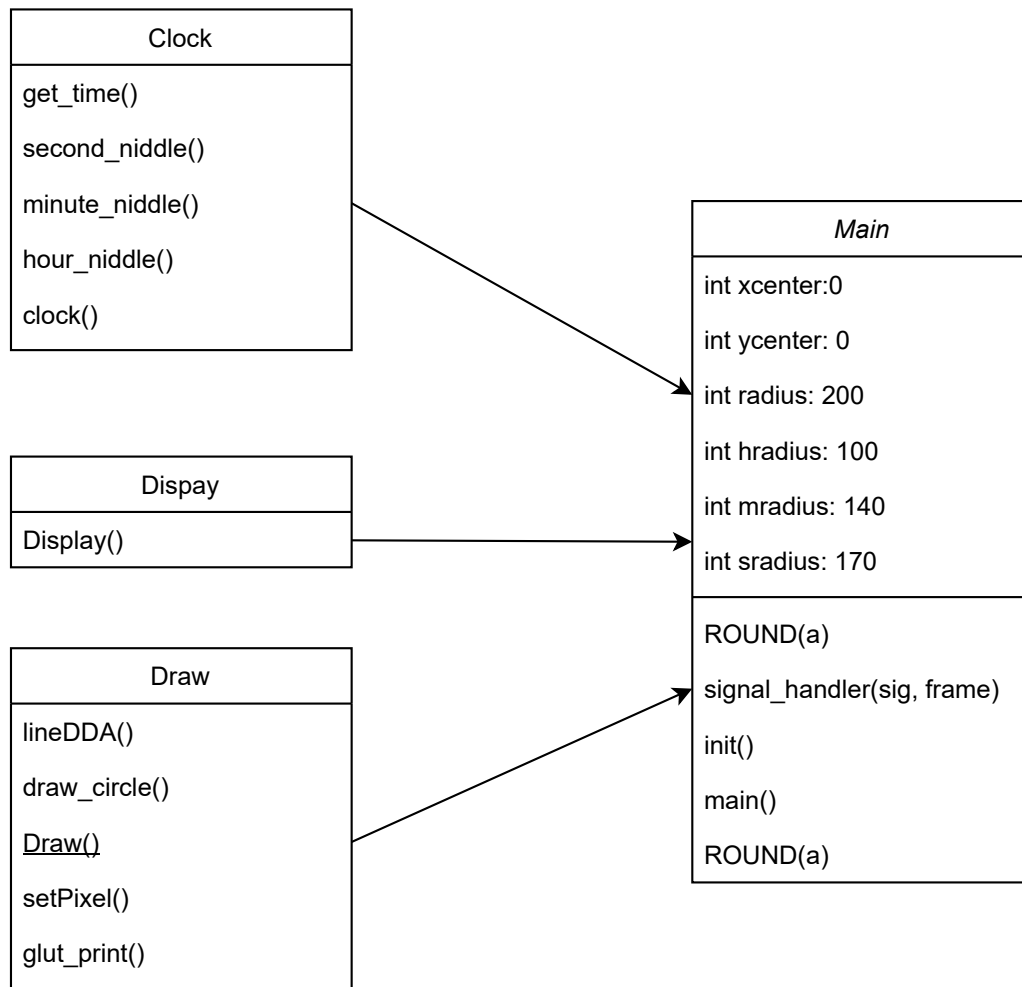


Figura 1.2: Diagrama de clases del proyecto.

## Capítulo 2

# Descripción del código

### 2.1. Clase Main.

### 2.2. Variables de iniciación

las variables que se usaron, son las variables que definen la lógica del programa.



```
1 // Centro y ejes del círculo
2 xcenter=0
3 ycenter=0
4 //Radio del círculo
5 radius=200
6 //length por hora
7 hradius=100
8 //length por minuto
9 mradius=140
10 //length por segundo
11 sradius=170
```

Figura 2.1: Variables inicializadas

Como se observa en la figura 2.1, están los ejes **x** y **y**, Tenemos los radios del círculo para éste caso, los cuales están los respectivos radios para las manecillas del reloj (segundo definido con valor 200, minuto 100 y hora 170).

#### 2.2.1. Clase Clock.

Como se explicón en el apartado 1.2.2, conversamos de una clase que tiene que los métodos que trabajan con las animaciones que simula un reloj andando (que no es algo obligatorio para el laboratorio, es un extra que trae el proyecto que edité).

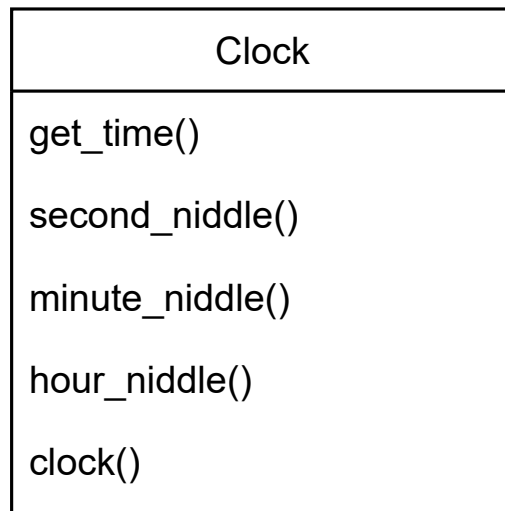


Figura 2.2: Clase Clock

Esta clase, como se muestra en la figura 2.4, tiene los métodos `get_time`, `second_niddle`, `minute_niddle`, `hour_niddle` y `clock`.

### 2.2.2. Clase Display.

Esta clase sólo tiene un método que es, irónicamente, un método que tiene el mismo nombre que la clase

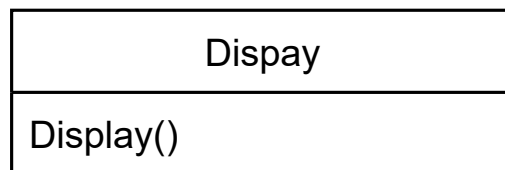


Figura 2.3: Clase Clock

La función de la ésta clase, maneja la posición, el tamaño, y demás items de las ventanas visuales del programa.

### 2.2.3. Clase Draw.

Esta clase contiene los métodos `lineDDA`, `draw_circle`, `draw`, `setPixel` y `glutPrint`.

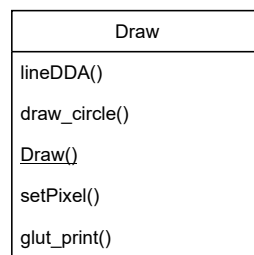


Figura 2.4: Clase Clock



# Capítulo 3

## conclusiones

### 3.1. Resultados y pautas de la actividad.

#### 3.1.1. Resultado.

Éste es el resultado del proyecto editado.

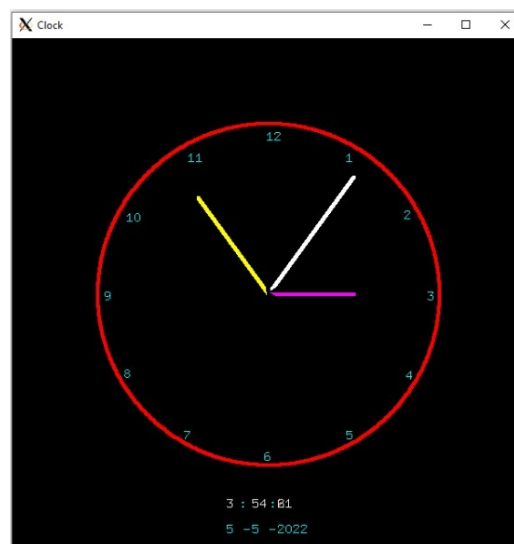


Figura 3.1: Resultado del proyecto

La diferencia que se puede ver aquí, es que, ya que WSL (por lo menos en su versión 1) no tiene capacidad de ejecutar gui nativamente, por lo que tuve que usar un servidor para que pudiera correr la gui de la aplicación desde WSL.

#### 3.1.2. Pautas del ejercicio.

Luego de lo mostrado alrededor de los capítulos, y de ilustrar el resultado obtenido en el apartado 3.1.1, vamos a disponer de responder unas pautas que nos deja el ejercicio.

¿Funciona bien tu programa?

como vimos en la figura 3.1, el reloj funciona correctamente.

¿Qué fallos existen?

Como tal, no se encontraro errores en la primera versión del archivo donde no hace uso de la programación orientada a objetos, sin embargo, por la dificultad de poder visualizar el resultado por mí mismo (porque el proceso es muy lento por el uso de XLaunch para windows, la primera versión me la revizó un compañero de la universidad, a quién nombraré en el apartado de agradecimientos).

## Capítulo 4

# Recursos

<https://pharos.sh/breve-introduccion-a-opengl-en-python-con-pyopengl/>

<https://codingshiksha.com/python/python-3-opengl-script-to-build-3d-digital-analog-clock-using-pyopengl-library-gui-desktop-app-full-project-for-beginners/>

<https://yewtu.be/watch?v=9rmxMUyFcj4>

## Capítulo 5

# Agradecimientos

Mi especial agradecimiento a mi compañero **Sebastian Steffen Castañeda Castillo** por la ayuda y colaboración para ayudarme a comprobar el funcionamiento del proyecto debido que a causas técnicas, no pude comprobarlos por mí mismo.

De igual manera, agradecer a todos aquellos compañeros que, de forma desinteresada, me han aportado con sus conocimientos el poder llevar adelante éste laboratorio.