INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE COMPÚTO

PROGRAMACIÓN ORIENTADA A OBJETOS

PRÁCTICA 3: PLANETARIO

PROF. ROBERTO TECLA PARRA

GRUPO 2CM4

ZEPEDA FLORES ALEJANDRO DE JESÚS

23/03/3018

**OBJETIVO**

Desarrollar en Java 3D una aplicación que simule el sistema solar, dicha aplicación deberá contar con al menos 3 planetas y para cada planeta crear una apariencia, cargar una textura, rotar y trasladar cada planeta alrededor del sol.

**INTRODUCCIÓN**

El API Java 3D es una interface para escribir programas que muestran e interactúan con gráficos tridimensionales. Java 3D es una extensión estándar del JDK 2 de Java. El API Java 3D proporciona una colección de constructores de alto-nivel para crear y manipular geometrías 3D y estructuras para dibujar esta geometría. Java 3D proporciona las funciones para creación de imágenes, visualizaciones, animaciones y programas de aplicaciones gráficas 3D interactivas.

El API 3D de Java es un árbol de clases Java que sirven como interface para sistemas de renderizado de gráficos tridimensionales y un sistema de sonido. El programador trabaja con constructores de alto nivel para crear y manipular objetos geométricos en 3D. Estos objetos geométricos residen en un universo virtual, que luego es renderizado. El API está diseñado con flexibilidad para crear universos virtuales precisos de una amplia variedad de tamaños, desde astronómicos a subatómicos.

**DESARROLLO**

Para poder desarrollar la práctica, primero necesitamos establecer las clases que vamos a utilizar para el funcionamiento óptimo de nuestro programa, para este caso son las siguientes:

import com.sun.j3d.utils.geometry.\*;

import com.sun.j3d.utils.image.TextureLoader;

import com.sun.j3d.utils.universe.\*;

import javax.media.j3d.\*;

import javax.vecmath.\*;

import java.awt.\*;

import javax.swing.\*;

import java.util.\*;

Algo nuevo que podemos notar son las librerías importadas de Sun, estás son las que contienen las herramientas para poder programar en las figuras en 3D.

BranchGroup sirve como un puntero a la raíz de una rama de gráfico de escena. Los objetos BranchGroup son los únicos objetos que se pueden insertar en un conjunto de objetos de Locale. Un subgrafo, enraizado por un nodo BranchGroup se puede considerar como una unidad de compilación.

public SolarSis(){

BranchGroup group = new BranchGroup();

Appearance appsol = new Appearance();

Appearance appearth = new Appearance();

Appearance appvenus = new Appearance();

Appearance appneptuno = new Appearance();

El siguiente paso es insertar la textura.

TextureLoader tex = new TextureLoader("TIERRA.JPG", null);

appearth.setTexture(tex.getTexture());

Después tenemos que crear la esfera con sus características.

Sphere earth = new Sphere(0.045f, Primitive.GENERATE\_NORMALS |

Primitive.GENERATE\_TEXTURE\_COORDS, 32, appearth);

Mediante las siguientes líneas de código, es como obtendremos el movimiento de la figura; en este caso requiere un movimiento de translación y otro de rotación.

TransformGroup earthRotXformGroup = Posi.rotate(earth, new Alpha(-1, 1250));

TransformGroup earthTransXformGroup = Posi.translat e(earthRotXformGroup, new Vector3f(0.0f, 0.0f, 0.7f));

TransformGroup earthRotGroupXformGroup = Posi.rotate (earthTransXformGroup, new Alpha(-1, 3000));

**CONCLUSIÓN**

La importancia de esta práctica radica en la utilización del entorno 3D que Java ofrece. Actualmente y con lo avanzado que esta la tecnología, visualizar animaciones en 3D se ha vuelto cotidiano, de ahí la importancia de aprender a utilizar este tipo de herramienta, ya que, además de dar presentación, representa para el cliente sentirse atraído en planos de 3 dimensiones.