

INSTITUTO POLITECNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO (ESCOM)

PROGRAMACIÓN ORIENTADA A OBJETOS

NOMBRE DEL ALUMNO:

* SANTOS MÉNDEZ ULISES JESÚS

PRÁCTICA:

* JAVA 3D

NÚMERO DE PRÁCTICA: 3

OPCIÓN 1:

* SISTEMA SOLAR(PLANETARIO)

FECHA DE ENTREGA:

* 27/05/2021

GRUPO:

* 2CM11

**Java 3D**

**Introducción**

El API Java 3D es una interface para escribir programas que muestran e interactúan con gráficos tridimensionales. El API Java 3D proporciona una colección de constructores de alto nivel para crear y manipular geometrías 3D y estructuras para dibujar esa geometría, de la misma manera proporciona las funciones para creación de imágenes, visualizaciones, animaciones y programas de aplicaciones gráficas 3D interactivas.

**API 3D de JAVA**

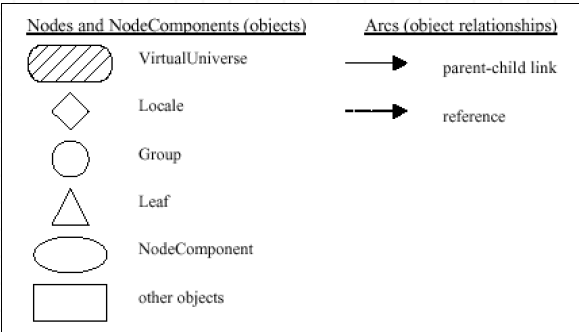
Es un árbol de clases Java que sirven como interface para sistemas de renderizado de gráficos tridimensionales y un sistema de sonido. El programador trabaja con constructores de alto nivel para crear y manipular objetos geométricos en 3D. Estos objetos geométricos residen en un universo virtual, que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

**Universo Virtual**

Un universo virtual Java 3D se crea desde un escenario gráfico. Un escenario gráfico se crea usando ejemplares de clase Java 3D, el escenario gráfico está ensamblado desde objetos que definen la geometría, sonidos, orientación, etc.

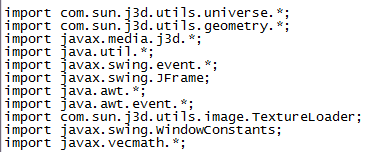
Las representaciones gráficas de un escenario gráfico pueden servir como herramienta de diseño y/o documentación para los programas Java 3D.

Para diseñar un universo virtual Java 3D se dibuja un escenario gráfico usando un conjunto de símbolos estándar. Después de completar el diseño, este escenario gráfico es la especificación del programa.



**Desarrollo**

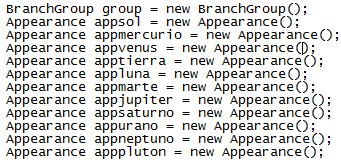
1. Se importaron las siguientes librerías de java para el diseño del universo en Java 3D, en ella están contenidas las clases y métodos que se utilizarán.



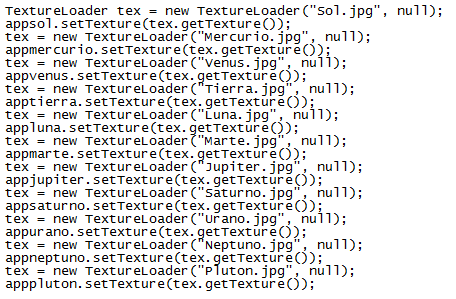
1. Se declaró la clase publica SolarSis



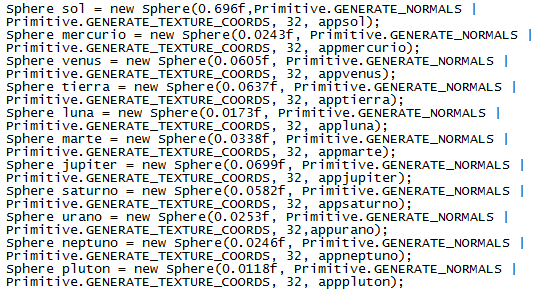
1. Se crean grupos con **BranchGroup** que es la raíz de un subgráfico o rama gráfica, la rama de contenido gráfico especifica el contenido del universo virtual, geometría, apariencia, etc.



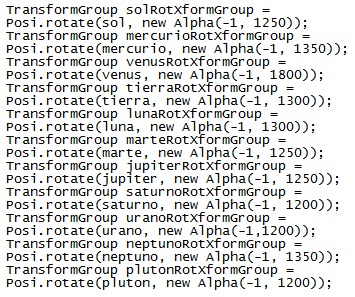
1. Se utiliza el objeto **TextureLoader** para el uso de ficheros como formatos, se pueden cargar JPEG, GIF, PNG entre otros.

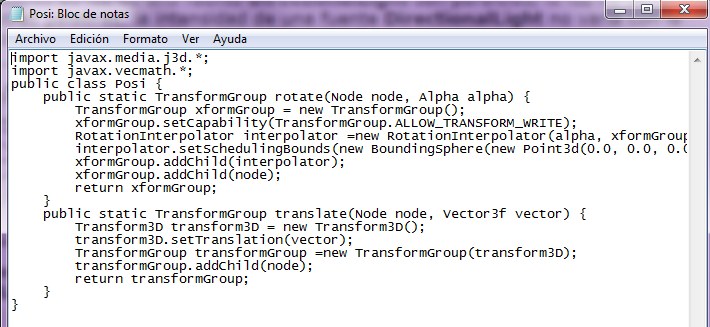


1. Se utilizan clases de utilidad geométrica para crear gráficos primitivos como cajas, conos, cilindros y esferas. La clase Sphere crea objetos visuales esféricos con el centro en el origen.

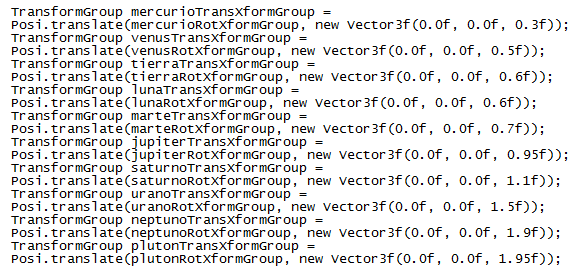


1. Se hacen girar los planetas sobre su propio eje, usando **Alpha** puede posicionar, orientar, y escalar cada uno de los nodos hijos, y se agrega duración y tiempo, se hace uso del método rotate en Posi.java, se hace uso de **RotationInterpolator** para que se pueda variar la orientación rotacional de un objeto visual sobre un eje.

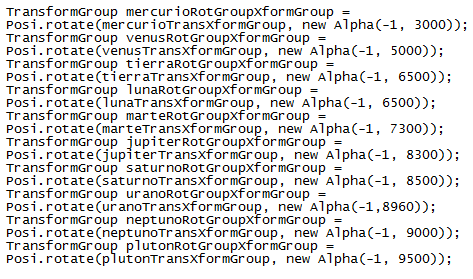




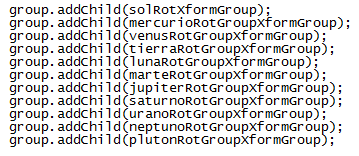
1. Se hace uso del método translate en Posi.java, se hace uso de la clase **Transform3D** y **setTranslation** y reemplaza los componentes de translación con los valores en el vector3D



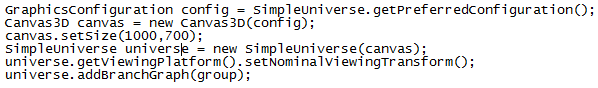
1. Se utiliza la clase TransformGroup para hacer rotar los planetas alrededor del Sol



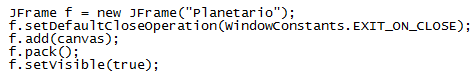
1. Se usa la clase Group para agrupar nodos teniendo un padre y los correspondientes hijos y se utiliza addChild para agregar los nodos a la lista de nodos.



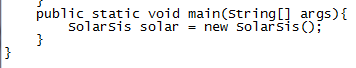
1. Se hace uso del objeto SimpleUniverse para crear una rama de vista gráfica completa para un universo virtual. En la rama se incluye un plato de imagen en donde se proyectará la imagen renderizada, y el objeto **Canvas3D** proporciona una imagen en la ventana de nuestra pantalla.



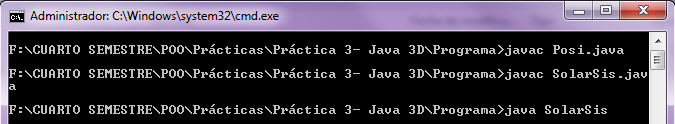
1. Se crea la ventana donde se agregará a canvas y se hará visible

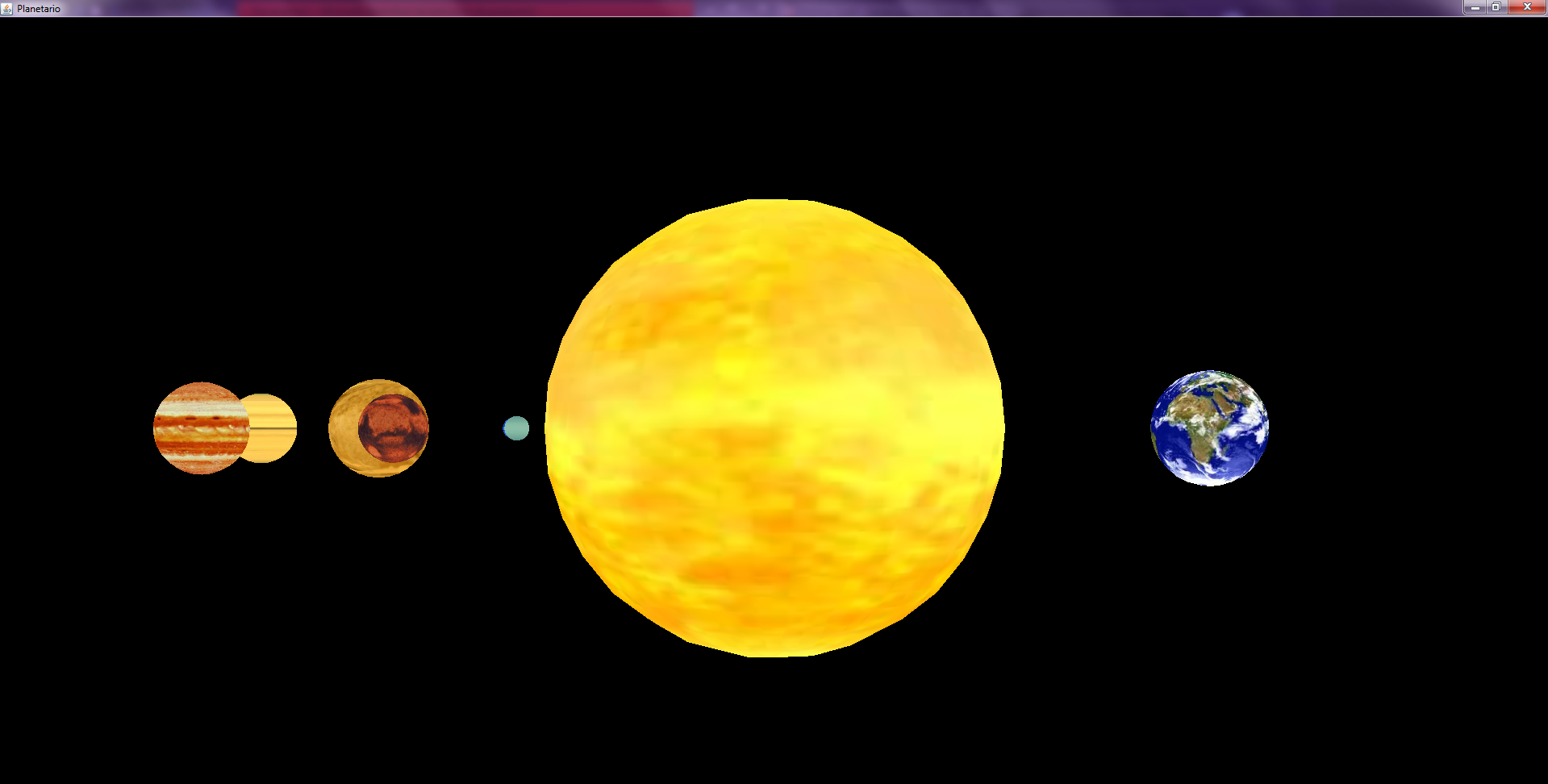


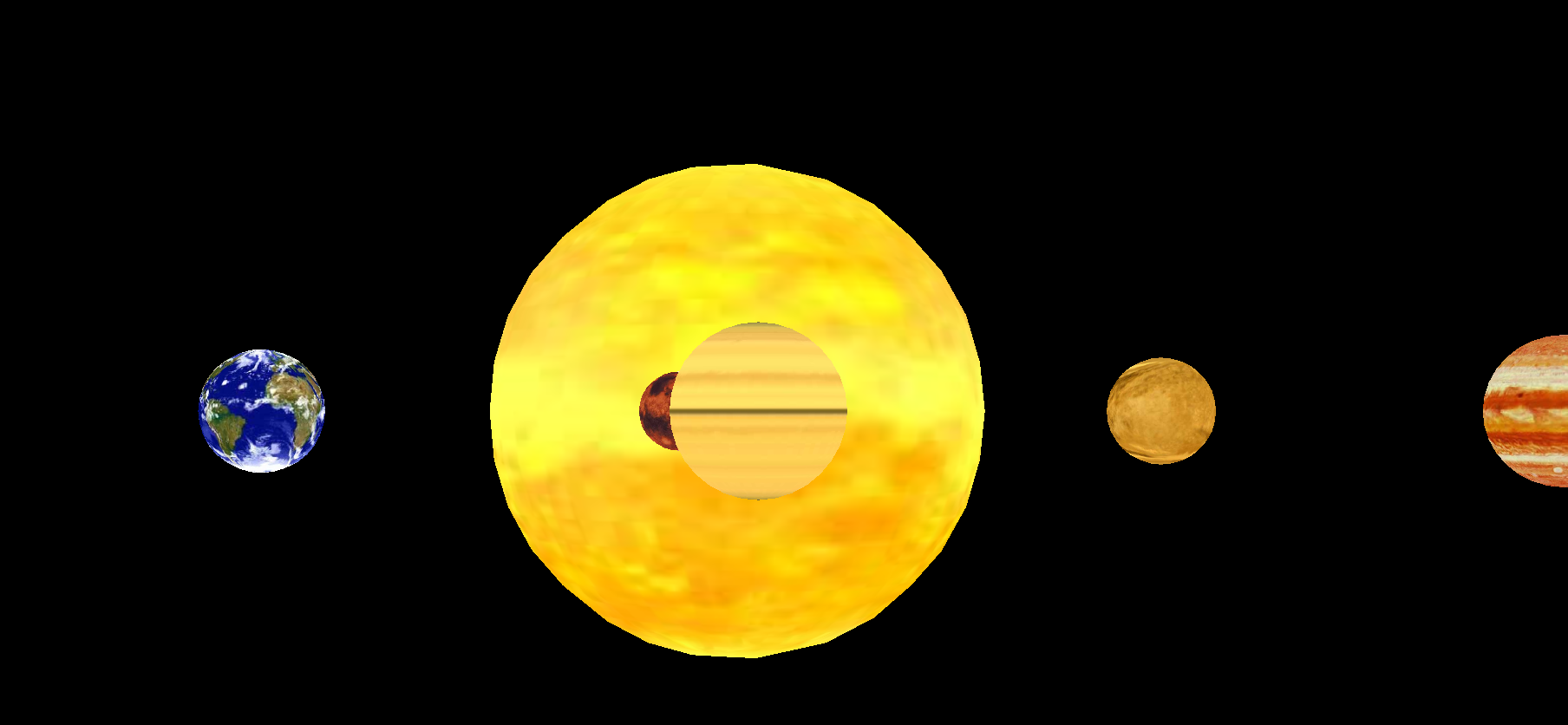
1. Por último se agrega el método main



1. Se procede a compilar y a ejecutar el programa en la consola CMD, aclarando que al inicio se tuvo un error pero se logró corregir copiando la carpeta lib de Java3D y pegándola en el jre de la versión de Java, de esta manera se eliminaron los mensajes de error del programa.







**Conclusión**

En conclusión esta práctica fue útil para conocer el proceso de la creación de objetos 3D a partir de una parte teórica y de ejemplos donde pudimos entender el funcionamiento de los objetos, clases y métodos empleados para renderizar, modelar, y darle consistencia a nuestra figura utilizando librerías de java3D.