CARERRA:

INGENIERÍA EN SISTEMAS COMPUTACIONALES

ALUMNOS:

JETZE OFNIEL BUSTAMANTE TRUJILLO

MANUEL ALEJANDRO MARTÍNEZ RAMÍREZ

DOCENTE: Ing. Arturo Carlos Rodríguez Román

Actividad. – Documentación del Juego de Asteroides.

Iguala, Gro., a 28 de mayo de 2018

INTRODUCCIÓN

En esta documentación, hablaremos sobre el juego de Asteroides, sus funciones y cómo fue que los hicimos. También, explicaremos el código, sus métodos, como le dimos color del fondo y como importamos las imágenes del juego, cabe destacar que se desarrolló con los amplios conocimientos que nuestro profesor nos enseñó durante la clase de graficación.

En clase de graficación utilizamos varios métodos y varias formar de poder hacer nuestro proyecto.

FUNCIONES

* Size()
* Background()
* Fill
* Height y Width
* Rec()
* PImage()
* TextFont ()
* Text ()
* mousePressed ()
* keyPressed ()
* keyReleased ()
* rotate2D ()
* heading2D ()
* random ()
* sqrt ()
* pushMatrix ()
* translate ()
* popMatrix ()
* color ()
* ellipse ()

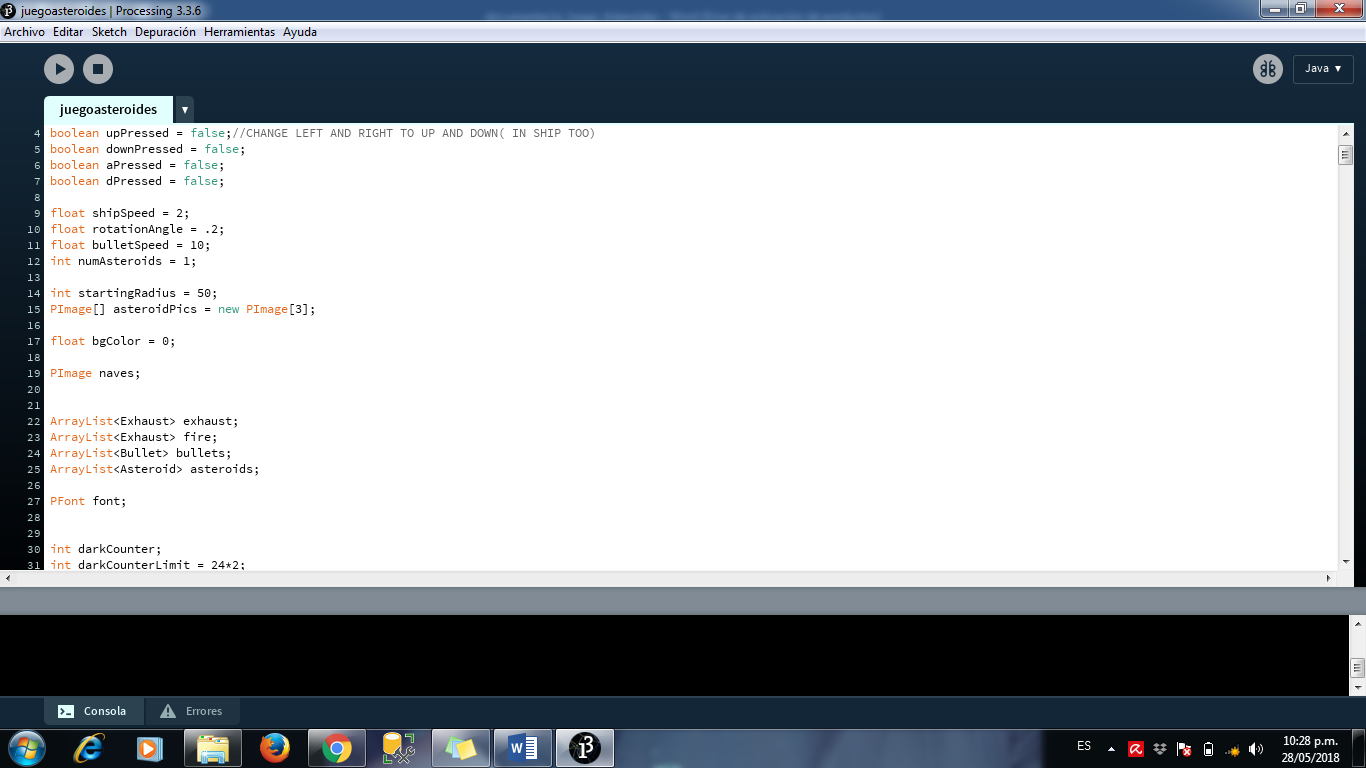
Empezamos a declarar las variables como boolean, float, int. Esto nos permitirá que estamos avisando al ordenador de que tenemos intención de utilizar una zona de la memoria para ir guardando un valor de referencia que puede ir cambiando. También cabe mencionar que se deben de declarar antes del “void setup”.

Utilizamos el background para el color del fondo de nuestro codigo, size para poner el tamaño de nuestro lienzo.

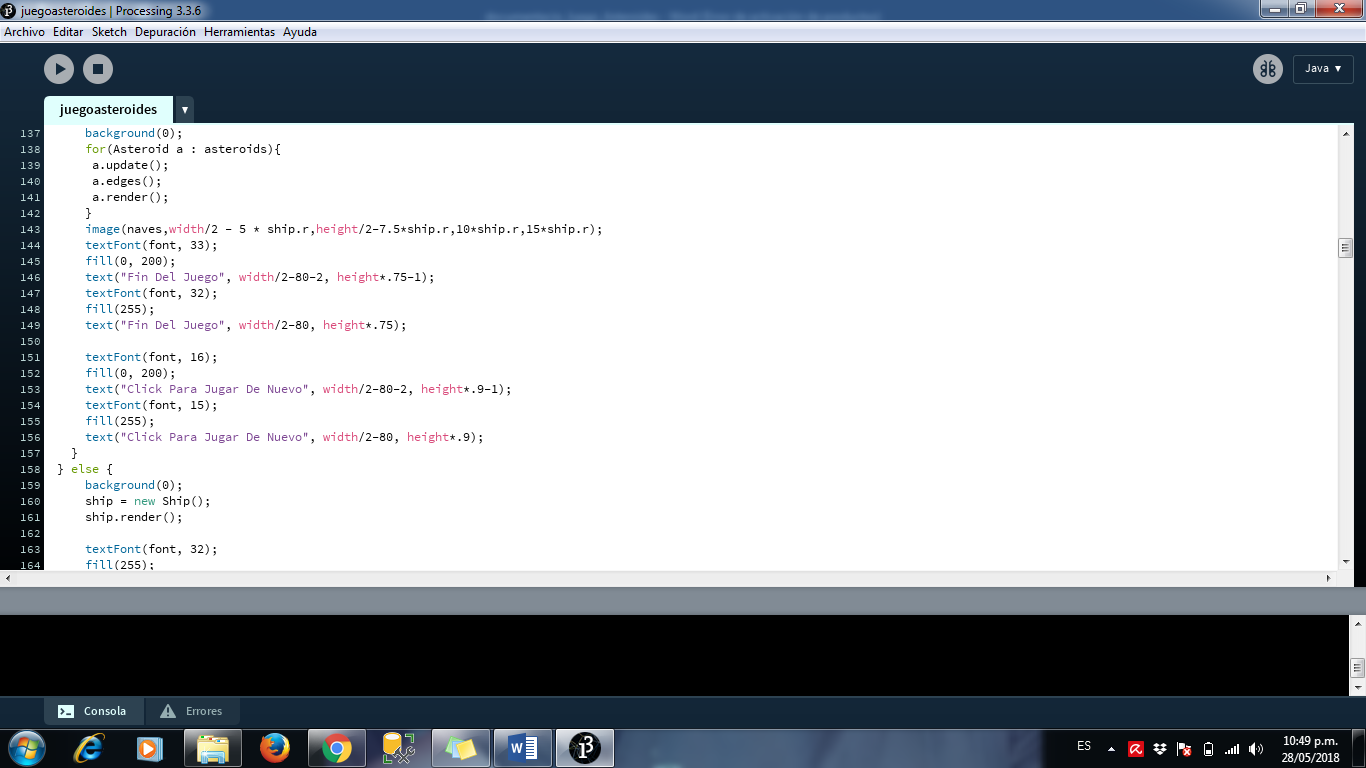
Fill para especificar donde queremos que se rellene. En este caso lo usaremos en los asteroides.

rect (0, 0, width, height) especificar el rectángulo con sus coordenadas “x, y” lo ancho y largo que queremos que sea el rectangulo usando Width y Height.

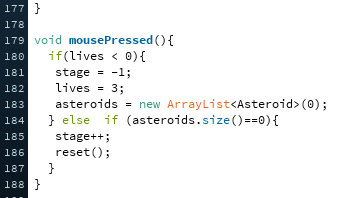
PImage utiliza imágenes e importa en el código, esto nos facilita el uso de las importaciones a processing. También debemos de declarar las imágenes. Y la declaramos con la siguiente instrucción “PImage naves”. La función PImage dibuja una imagen en la ventana de visualización. Las imágenes deben estar en el directorio de bocetos "datos" para cargar correctamente.



TextFont y Text establece la fuente actual que se dibujará con el texto. La fuente debe de crearse con el “createFont” o cargado con “loadFont “antes que se puedan usar. Usamos estas funciones para ponerle nombre en nuestro juego como el de “Bienvenido al juego”, “Click para jugar de nuevo”, “Fin del juego”. Como se muestra en el siguiente ejemplo.

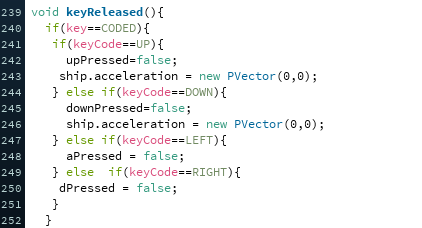


Para que juego agarre y se pueda mover y aventar los lazer cada vez que presionamos la tecla del espacio, es necesario implementar la función del mousePressed. MousePressed se llama una vez cada vez que se presiona un botón del mouse. Aquí empezará a correr el juego cuando nosotros hayamos presionado el raton.

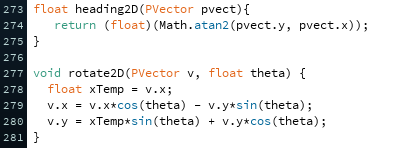


keyPressed. Usaremos la función keyPressed en nuestro para que cada que movamos nuestro coete con las flechas el coete se moverá a donde le hayamos indicado ya se arriba, abajo, izquierda, derecha.



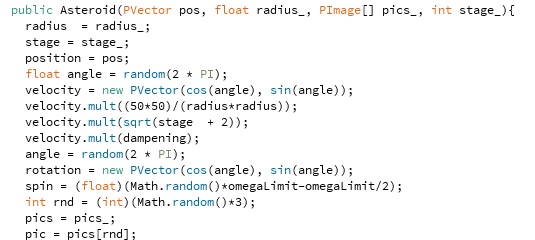
keyReleased funciona cada vez que un botón o tecla es liberada entonces entrará en función.

Roate 2D.- Es la cantidad especificada por el parámetro de ángulo. El ángulo debe especificarse en radianes. Y lo implementamos con un vector para que pueda girar a donde nosotros le indicamos.

Heading 2D.- Calcula el Angulo del vector, esto debe de ser en 2D.

Random. - Genera números aleatorios. Cada vez que se llama a la función random (), devuelve un valor inesperado dentro del rango especificado. Si solo se pasa un parámetro a la función, devolverá un flotante entre cero y el valor del parámetro alto.

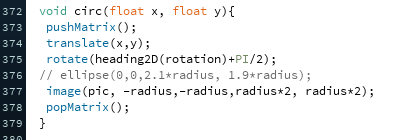
sqrt().- Calcula la raíz cuadrada de un número. La raíz cuadrada de un número siempre es positiva, aunque puede haber una raíz negativa válida.



pushMatrix (). - Impulsa la matriz de transformación actual a la pila de la matriz. La función pushMatrix () guarda el sistema de coordenadas actual en la pila y popMatrix () restaura el sistema de coordenadas anterior. pushMatrix () y popMatrix () se usan en conjunción con las otras funciones de transformación y se pueden incorporar para controlar el alcance de las transformaciones.

popMatrix (). - Coloca la matriz de transformación actual fuera de la pila de la matriz.

Translate().- Especifica una cantidad para desplazar objetos dentro de la ventana de visualización. El parámetro x especifica la traducción izquierda / derecha, el parámetro y especifica la conversión ascendente / descendente, y el parámetro z especifica las traducciones hacia / desde la pantalla.

En estas 3 funciones estamos especificando los asteroides, como van a girar y hacia dónde van a estar girando.

Ellipse().- El ellipse es usando como un ovalo pero en nuestro código del juego lo usamos para darle diseño a los asteroides.

Color().- En color le asignamos el color que nosotros queramos usar, en esta caso nosotros usamos el color gris a los asteroides.



**CONCLUSIÓN**

Al finalizar con este proyecto nos dimos cuenta lo fundamental que es aprenderse las instrucciones básicas de java, ya que, en la materia de graficación pusimos en práctica nuestros conocimientos adquiridos durante la carrera. También el programa processing es un programa muy fácil tanto como en el diseño y en la programación. Incluso la sencillez que tiene processing para crear y diseñar juegos, ya sea medianos o pequeños como en este caso el de “Asteroides”.

El conocimiento adquirido que obtuvimos con el profesor fue de mucha ayuda, ya que, hay unas funciones que con solo declararlas te hacen lo que uno quiere hacer.