|  |  |  |
| --- | --- | --- |
| Texto  Descripción generada automáticamente con confianza media | TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS  FACULTAD DE INGENIERÍA  Universidad Nacional de Jujuy |  |

Trabajo Práctico

N°2

Apellido y Nombre

Ibañez Esteban Agustin

TUV000722

*Profesor:*

*Mg. Ing. Ariel Alejandro Vega*

*Año 2024*

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

**Punto 1:** Desarrolle una historia de usuario, en la cual defina la visualización y movimiento

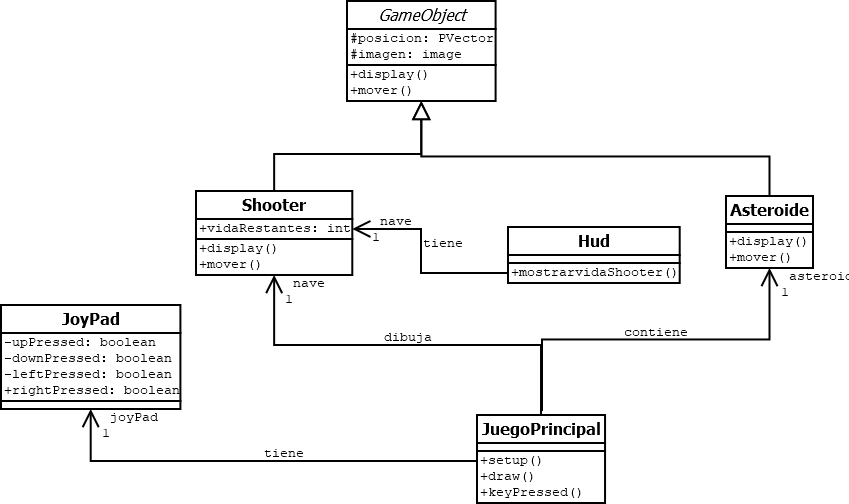
De una clase GameObject, de la que heredan Shooter y Asteroide. GameObjects es abstracta, y posee atributos protegidos: posición, imagen; además del método abstracto display() y mover(). Además debe poseer un HUD que visualice la cantidad de vidas del

Shooter. Utilice un JoyPad para generar los movimientos.

**Desarrollo:**

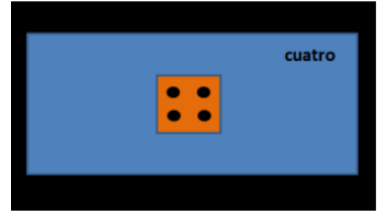
|  |
| --- |
| **Historia De Usuario** |
| **Nombre de la Historia Usuario** : Movimiento y visualización de GameObject |
| **Prioridad** : Elevada |
| **Estimación Esperada**: 1Dia |
| **Responsable** : Desarrollador |
| **Descripción**: Como Desarrollador de un juego de disparos, quiero implementar una funcionalidad para visualizar y mover (GameObject) en la pantalla, esto incluye a los objetos Shooter y Asteroide para brindar una experiencia interactiva y dinámica |
| **Criterios de Aceptacion:**  -Se implementa la clase abstracta GameObject con atributos protegidos de posicion e imagen. Donde su metodos abstractos de mover y dibujar se reflejen en la pantalla  -Se desarrollan clases Shooter y Asteroide que heredan del GameObject y implementan especificaciones de metodos dibujar y mover, brindando una mejora grafica y fluidez.  -Se integra un HUD que muestra la cantidad de vidas de la Class Shooter en la pantalla del juego , da a conocer al jugador cuantas vidas le quedan  -Se le permite al jugador el poder controlar el movimiento del shooter utilizando un JoyPad conectado al juego. Brinda al jugador una movilidad libre por todo el ancho y alto de la ventana.  -Los GameObjects , Shooter y Asteroide responden de manera suave a las entradas de Joypad , lo que permite al jugador moverse y esquivar de manera efectiva. |

**DIAGRAMA**



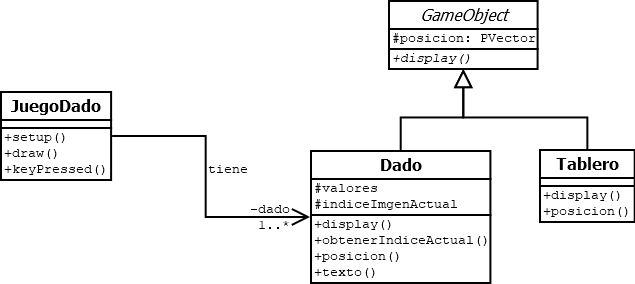
**Punto 2**: Desarrolle un videojuego que cumpla con las siguientes especificaciones:

Realice un diagrama de clases Como se observa se trata de un dado. El cual al presionar un botón debe generar un número aleatorio entre 1 y 6 y dibujarlo. Además, debe mostrar el número en la parte superior derecha. Repetir esto cuantas veces lo desee y al finalizar (con otro botón) debe dibujar por consola y agrupado en filas de 4 columnas los dados obtenidos. Al momento de programar utilice constructores sobrecargados. Considere que el dado se muestra en un tablero, este tablero contiene al dado, y al texto. Además, almacene cada dado obtenido en un arreglo. Considere aplicar la herencia respecto de que existe una clase abstracta padre GameObject, de la que hereda la posición y el método abstracto display(). Luego recrear otra versión donde use imágenes en lugar de dibujar con las primitivas.



**Desarrollo:**

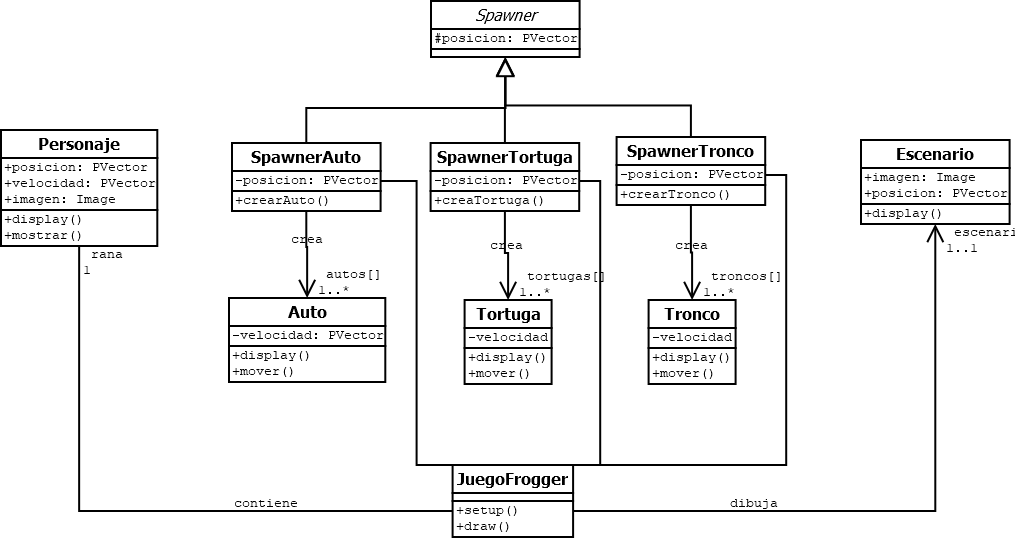
**DIAGRAMA**



**Punto 3:** Realice el modelado de las clases que intervienen en el juego frogger a partir de la Fig. 1. Realice la construcción de las clases en processing. El juego debe llegar a poder mostrar en pantalla la visualización de los diferentes objetos modelados. Utilice herencia y encapsulamiento para los vehículos. Además, los vehículos deben guardarse en una lista de objetos que es atributo de la clase SpawnerVehiculos.

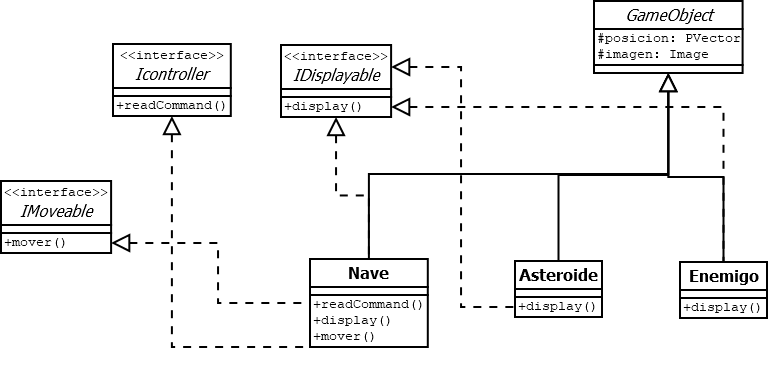
**DIAGRAMA**

**Punto 4**



Considere programar un juego de naves. Debe usar imágenes para las naves, los asteroids y los enemigos. Aplique herencia. Use una interface denominada IDisplayable que tenga el método display(). Defina dos interfaces más: IMoveable que tenga el método mover() y Otra IControler que tenga el método readCommand(); Usando el sentido común haga que las clases Nave, Asteroid y Enemy implementen las interfaces correspondientes. Finalmente use la dependencia para que la nave dispare balas que serán almacenadas en una lista de balas. Las balas se deben destruir cuando salen de pantalla.

**DIAGRAMA**



Fuentes bibliográficas:

Historia de Usuarios:

<https://www.atlassian.com/es/agile/project-management/user-stories>

Diseño de dados:

<https://www.canva.com/es_419/>