UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 1

CATEDRÁTICO: ING. WILLIAM ESTUARDO ESCOBAR ARGUETA

TUTOR ACADÉMICO: JOSUÉ RODOLFO MORALES CASTILLO



**MANUAL TÉCNICO**

**Space Invaders**

JOSÉ ALEXANDER LÓPEZ LÓPEZ

CARNÉ: 202100305

SECCIÓN: A

GUATEMALA, 20 DE JUNIO DE 2,024

# ÍNDICE

[**ÍNDICE 1**](#_k1cgegpfdrrl)

[**INTRODUCCIÓN 1**](#_574nc6ehlpl)

[**OBJETIVOS 1**](#_3bsxbi49eln7)

[1. GENERAL 1](#_38w1k6cukb5i)

[2. ESPECÍFICOS 1](#_poh928tpbmga)

[**ALCANCES DEL SISTEMA 1**](#_twvsj8hvwtb9)

[**ESPECIFICACIÓN TÉCNICA 1**](#_hxtoxxv43krt)

[● REQUISITOS DE HARDWARE 1](#_mfriwpcdfh1l)

[● REQUISITOS DE SOFTWARE 1](#_8h6kxoigku56)

[**DESCRIPCIÓN DE LA SOLUCIÓN 2**](#_14axd4p9n3ry)

[**LÓGICA DEL PROGRAMA 2**](#_ng9quldintk8)

[3](#_xbejjcbelcz7)

# 

# INTRODUCCIÓN

El propósito de este manual técnico es proporcionar una guía exhaustiva y detallada sobre el desarrollo, la arquitectura y el mantenimiento del proyecto **Space Invaders**. Este documento está dirigido a desarrolladores y profesionales de TI que buscan comprender en profundidad los componentes internos del juego, sus dependencias de software, y el flujo de datos. Aquí se desglosan los detalles técnicos esenciales, incluyendo la configuración del entorno de desarrollo, la estructura del código, los patrones de diseño implementados y las mejores prácticas para la extensión y mejora del sistema.

El manual aborda cada aspecto del proyecto, comenzando con los requisitos del sistema necesarios para la correcta ejecución de la aplicación, pasando por una descripción detallada de los diferentes módulos y sus interacciones, hasta las instrucciones para la compilación, ejecución y depuración del juego. Además, se incluyen secciones dedicadas a la gestión de versiones, pruebas unitarias e integración continua, asegurando que el mantenimiento y la evolución del proyecto se realicen de manera eficiente y sin contratiempos.

El objetivo final de este manual es facilitar la comprensión y manipulación del código fuente, asegurando que los desarrolladores puedan integrarse rápidamente y trabajar de manera efectiva en el proyecto. Proporciona una base sólida para futuras actualizaciones y mejoras, permitiendo a los desarrolladores realizar modificaciones y añadir nuevas funcionalidades con confianza. A través de este documento, se busca mantener la coherencia y la calidad del código, así como garantizar que el videojuego Space Invaders siga siendo un proyecto robusto y sostenible a largo plazo.

# OBJETIVOS

## GENERAL

* 1. Proveer una guía técnica completa y detallada que permita a los desarrolladores y profesionales de TI comprender, mantener, y expandir el proyecto Space Invaders, asegurando la correcta configuración del entorno, el entendimiento profundo del código fuente y la implementación de mejores prácticas en el desarrollo de software.

## ESPECÍFICOS

* 1. Proporcionar instrucciones detalladas para la instalación de todas las herramientas necesarias, como el sistema operativo, JDK, y el IDE, asegurando que los desarrolladores puedan configurar su entorno de desarrollo de manera eficiente y sin problemas.
  2. Explicar la arquitectura del juego y la organización del código fuente, incluyendo los módulos principales, las clases, y sus interacciones.
  3. Proveer directrices y mejores prácticas para el mantenimiento y la extensión del proyecto, incluyendo la implementación de nuevas funcionalidades, la realización de pruebas unitarias, y la gestión de versiones.

# ALCANCES DEL SISTEMA

**Configuración del Entorno de Desarrollo:**

Instrucciones detalladas para la instalación y configuración de los requisitos del sistema, incluyendo el sistema operativo compatible, JDK, y el IDE recomendado.

Guía paso a paso para importar el proyecto en el IDE y asegurarse de que el entorno esté listo para el desarrollo y la ejecución del juego.

**Descripción de la Arquitectura del Sistema:**

Desglose completo de la arquitectura del juego, incluyendo diagramas y explicaciones de los componentes principales, como la nave, los enemigos, los ítems y las pantallas del juego.

Descripción de la estructura del código fuente, detallando las clases y métodos principales y cómo interactúan entre sí.

Flujo de Funcionalidades:

Explicación detallada del flujo de la aplicación desde la pantalla principal hasta el fin del juego, incluyendo el inicio de un nuevo juego, la carga de partidas guardadas, la visualización de puntuaciones máximas y la salida del juego.

Instrucciones sobre cómo utilizar cada funcionalidad del juego, con ejemplos y capturas de pantalla para ilustrar cada paso.

**Mantenimiento y Extensión del Proyecto:**

Directrices para el mantenimiento del código, incluyendo mejores prácticas para la escritura y organización del código, así como la implementación de pruebas unitarias. Procedimientos para la gestión de versiones y la integración continua, asegurando que las modificaciones se realicen de manera controlada y eficiente.

Estrategias para la extensión del juego, proporcionando pautas para añadir nuevas funcionalidades, mejorar las existentes y asegurar la compatibilidad y estabilidad del sistema.

**Solución de Problemas y Depuración:**

Consejos y técnicas para la depuración del código y la resolución de problemas comunes que puedan surgir durante el desarrollo y la ejecución del juego.

Lista de errores conocidos y sus soluciones, ayudando a los desarrolladores a identificar y corregir problemas de manera rápida y efectiva.

**Documentación y Comentarios en el Código:**

Importancia de mantener el código bien documentado y comentado para facilitar la comprensión y el mantenimiento del proyecto por parte de otros desarrolladores.

Ejemplos de buenas prácticas en la documentación y comentarios del código.

Al proporcionar una guía detallada y práctica sobre estos aspectos, el manual técnico de Space Invaders tiene como objetivo asegurar que cualquier desarrollador pueda integrarse rápidamente al proyecto, mantenerlo de manera eficiente y contribuir a su mejora y evolución continua.

# ESPECIFICACIÓN TÉCNICA

Para asegurar un entorno de desarrollo adecuado y eficiente para el proyecto Space Invaders, es necesario que los programadores cuenten con ciertos requisitos de hardware y software. A continuación, se detallan los requisitos necesarios para trabajar con la aplicación y continuar con su desarrollo futuro:

## REQUISITOS DE HARDWARE

## Procesador:

## Procesador de 1 GHz o superior (se recomienda un procesador multinúcleo para un rendimiento óptimo).

## Memoria RAM:

## Al menos 4 GB de RAM (se recomiendan 8 GB o más para un desarrollo fluido y ejecución de múltiples herramientas de desarrollo simultáneamente).

## Espacio en Disco:

## Al menos 500 MB de espacio libre en disco para la instalación de JDK, IDE, y otros recursos necesarios.

## Espacio adicional para la creación y almacenamiento de archivos del proyecto, incluyendo archivos de juego guardados y binarios generados.

## Tarjeta Gráfica:

## No se requiere una tarjeta gráfica avanzada, pero se recomienda una tarjeta con soporte para gráficos 2D básicos.

## Pantalla:

## Resolución mínima de 1024x768 píxeles (se recomienda una resolución más alta para una mejor experiencia de desarrollo).

## REQUISITOS DE SOFTWARE

* + Q Sistema Operativo:
  + Windows 7 o superior
  + macOS 10.12 (Sierra) o superior
  + Linux (cualquier distribución moderna)
  + **Java Development Kit (JDK):**

JDK 8 o superior. Es esencial tener el JDK instalado para compilar y ejecutar el proyecto. Se puede descargar desde el sitio oficial de Oracle o utilizar distribuciones de código abierto como OpenJDK.

* + **Integrated Development Environment (IDE):**

Un IDE es altamente recomendable para facilitar el desarrollo, la depuración y la gestión del código. Los IDEs recomendados son:

* + IntelliJ IDEA
  + Eclipse
  + NetBeans
  + **Control de Versiones:**

Git debe estar instalado para la gestión de versiones del código fuente. Es útil para la colaboración y el seguimiento de cambios en el proyecto. Se puede descargar desde el sitio oficial de Git.

* + **Dependencias de Java:**

Asegurarse de tener instaladas todas las bibliotecas estándar de Java. No se requieren bibliotecas externas específicas para este proyecto, pero es crucial verificar que las bibliotecas estándar de Java (javax.swing, java.io, etc.) estén disponibles.

* + **Herramientas de Compilación y Construcción:**

Aunque el uso de herramientas como Maven o Gradle no es obligatorio para este proyecto, su utilización puede facilitar la gestión de dependencias y la automatización de tareas de construcción.

* + **Herramientas de Pruebas Unitarias:**

JUnit o TestNG para escribir y ejecutar pruebas unitarias, asegurando que el código nuevo y existente funcione correctamente.

**Herramientas de Integración Continua:**

* + Jenkins, Travis CI, o cualquier otra herramienta de integración continua para automatizar la construcción, prueba y despliegue del proyecto.

Siguiendo estos requisitos de hardware y software, los programadores estarán bien equipados para trabajar de manera efectiva con la aplicación Space Invaders, asegurando un desarrollo continuo y sostenible del proyecto.

# DESCRIPCIÓN DE LA SOLUCIÓN

* Para desarrollar Space Invaders, primero se revisó minuciosamente el enunciado del proyecto para comprender todos los requisitos. Se identificaron las funcionalidades clave como el control de la nave, la aparición de enemigos, la interacción con ítems, y la gestión de diferentes pantallas del juego.
* Decidí modularizar el código, dividiendo el juego en componentes independientes como la nave, los enemigos, y la interfaz de usuario. Utilizando patrones de diseño orientados a objetos para garantizar un código mantenible y escalable.
* Se comenzó con prototipos iniciales para validar nuestras ideas y ajustamos el diseño según fuera necesario. Se implementó el control de la nave, asegurando que se moviera y disparara de manera fluida. Desarre tres tipos de enemigos con diferentes atributos y lógica de movimiento. Integré ítems potenciadores y debilitadores que afectan el temporizador y los puntos del jugador, añadiendo dinamismo al juego.
* Implementé las pantallas del juego, asegurando transiciones suaves y acciones intuitivas para el usuario. Realicé pruebas unitarias y funcionales para validar cada componente y el juego en su conjunto. Durante las pruebas, corregí errores y optimizé el rendimiento del juego.
* Este enfoque nos permitió cumplir con todos los requisitos del enunciado y desarrollar un Space Invaders funcional, robusto y mantenible.

# LÓGICA DEL PROGRAMA

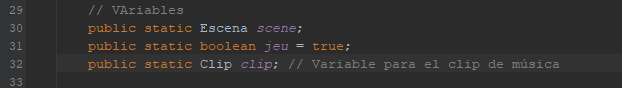
## Clase: Main*:*

## *Texto Descripción generada automáticamente*

### Librerías

* + - java.awt: Utilizada para manejar componentes gráficos como JFrame, JPanel, JLabel, y JButton, permitiendo la creación de la interfaz de usuario.
    - javax.sound.sampled: Usada para cargar y reproducir archivos de audio (AudioInputStream, Clip), necesario para la reproducción de música de fondo en el juego.

### Variables Globales de la clase Main



**scene:** Objeto de tipo Escena que representa el contenido principal del juego.

**jeu:** Variable booleana que controla el estado del juego.

**clip:** Variable estática de tipo Clip para la reproducción de música de fondo.

### Función Main

La función main se encarga de iniciar la aplicación creando y configurando la ventana principal (JFrame) del juego. Carga la música de fondo desde un archivo de audio, configura la imagen de fondo y los botones principales (Nuevo Juego, Cargar Juego, Puntuaciones, Salir) usando JButton y JPanel.

*Texto

Descripción generada automáticamente*

### Procedimientos, métodos y Funciones utilizadas

A continuación se dará una explicación general de lo que hace cada función:

Texto

Descripción generada automáticamente

**demarrerJeu():** Inicia la ventana principal del juego (JFrame jeuFrame) donde se mostrará la escena del juego (Escena).

## Clase: Escena:

Imagen de la pantalla de un computador

Descripción generada automáticamente con confianza baja

### Librerías

* + - Java.awt: Proporciona las clases necesarias para manejar gráficos y eventos en la interfaz de usuario.
    - javax.swing: Utilizada para componentes gráficos como JPanel, JFrame y JOptionPane, facilitando la creación de la interfaz de usuario.

### Variables Globales de la clase Escena:

**fondo:** Imagen de fondo del juego.

**vaisseau:** Objeto de tipo Nave que representa la nave del jugador.

**groupeAliens:** Objeto de tipo GrupoAliens que maneja el grupo de aliens en el juego.

**tirVaisseau, tirAlien1, tirAlien2, tirAlien3:** Objetos de proyectiles para la nave y aliens respectivamente.

**item1, item2, item3, item4:** Objetos de tipo Item que representan los items del juego.

**scoreImage, timeImage:** Imágenes utilizadas para representar el score y el tiempo en la interfaz.

**afficheScore, afficheTexte, timerFont:** Fuentes utilizadas para mostrar texto en la interfaz.

**score:** Puntaje actual del jugador.

**timer:** Objeto Timer utilizado para contar el tiempo restante en el juego.

**tiempoRestante:** Tiempo restante en segundos para la partida.

**gameOverDisplayed:** Variable booleana que controla la visualización del mensaje de Game Over.

### Constructor:

El constructor Escena() inicializa los componentes principales del juego, carga las imágenes de fondo y de los items, configura el temporizador y maneja eventos de teclado para interactuar con la interfaz.

Texto

Descripción generada automáticamente

### Métodos:

* **paintComponent(Graphics g):** Método sobrecargado para dibujar todos los elementos visuales en la pantalla de juego. Dibuja el fondo, el score, el tiempo, la nave, los aliens, los proyectiles y los items. También gestiona la detección de colisiones y el estado de Game Over.

Texto

Descripción generada automáticamente

* **guardarPuntaje(String nombre, int puntaje):** Método privado que guarda el puntaje del jugador en un archivo de texto. Utiliza BufferedWriter para escribir el puntaje junto con el nombre del jugador en el archivo.

Texto

Descripción generada automáticamente

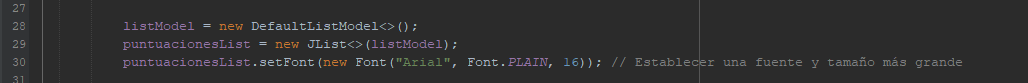
## Clase: PuntuacionesFrame*:*

Texto

Descripción generada automáticamente

### Librerías

* javax.swing: Utilizada para la interfaz gráfica de usuario, incluyendo componentes como JFrame, JList, DefaultListModel, JScrollPane y JOptionPane.
* java.awt: Proporciona clases para manejar gráficos y eventos en la interfaz de usuario.
* java.io: Utilizada para la lectura de archivos de texto con BufferedReader y FileReader.
* java.util: Utilizada para trabajar con colecciones como ArrayList y Collections, así como Comparator para ordenar las puntuaciones.



### Variables Globales de la clase Escena:

* **listModel:** DefaultListModel utilizado para almacenar y mostrar las puntuaciones en la interfaz.
* **puntuacionesList:** JList que muestra las puntuaciones almacenadas.

### Constructor:

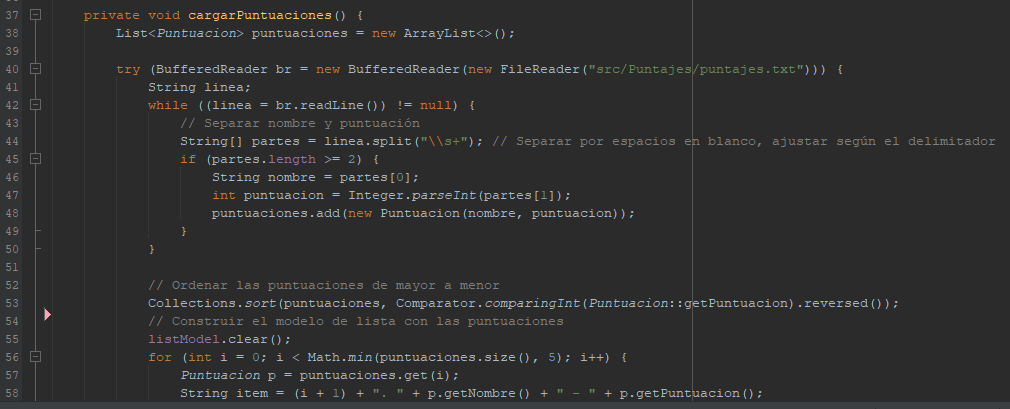
Texto

Descripción generada automáticamente

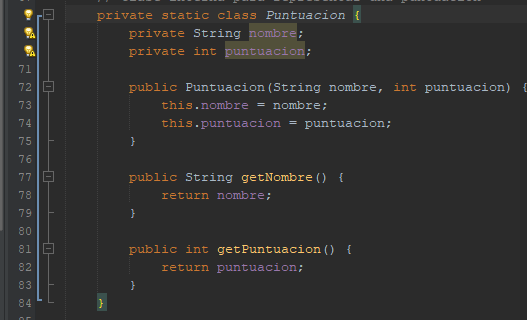
* El constructor PuntuacionesFrame() inicializa la ventana de la aplicación de mejores puntuaciones, estableciendo el título, tamaño, cierre de operación y ubicación. Carga las puntuaciones desde el archivo puntajes.txt utilizando el método cargarPuntuaciones(), y las muestra en una lista JList.

### Métodos:

* **cargarPuntuaciones():** Lee las puntuaciones desde el archivo puntajes.txt, separa el nombre y la puntuación de cada línea, crea objetos Puntuacion y los ordena de mayor a menor puntuación. Construye el modelo de lista listModel con las cinco mejores puntuaciones y lo muestra en puntuacionesList.



* **Puntuacion (Clase interna):** Representa una puntuación con un nombre y un valor de puntuación. Proporciona métodos para obtener el nombre y la puntuación.



## Clase: Alien:

Texto

Descripción generada automáticamente

### Constructor:

* Constructor Alien(int xPos, int yPos, String strImg1, String strImg2): Este constructor inicializa las variables de instancia heredadas de la clase Entidad y algunas específicas para el alienígena.

Texto

Descripción generada automáticamente

### Métodos:

Texto

Descripción generada automáticamente

* EligeImagen(boolean pos1): Este método selecciona la imagen a mostrar dependiendo del estado del alienígena (vivo o muerto).

## Clase: Entidades*:*

Texto

Descripción generada automáticamente

### Librerías

* java.awt.Image: Se utiliza para representar imágenes en el juego.
* javax.swing.ImageIcon: Permite cargar imágenes desde el sistema de archivos o desde recursos empaquetados en el archivo JAR.

### Variables globales de la clase Entidad

Texto

Descripción generada automáticamente

* largeur, altura: Representan las dimensiones (ancho y alto) de la entidad.
* xPos, yPos: Coordenadas x e y de la posición de la entidad en la pantalla.
* dx, dy: Velocidades de desplazamiento en las direcciones x e y.
* vivo: Indica si la entidad está viva o no.
* strImg1, strImg2, strImg3: Nombres de los archivos de imagen asociados a la entidad en diferentes estados.
* ico: Objeto ImageIcon que carga y gestiona las imágenes asociadas a la entidad.
* img: Objeto Image que representa la imagen cargada desde el ImageIcon.

### Métodos:

* **getLargeur():** Devuelve el ancho (largeur) de la entidad.
* **setLargeur(int largeur):** Establece el ancho (largeur) de la entidad.
* **getAltura():** Devuelve la altura (altura) de la entidad.
* **setAltura(int altura):** Establece la altura (altura) de la entidad.
* **getxPos():** Devuelve la posición x (xPos) de la entidad.
* **setxPos(int xPos):** Establece la posición x (xPos) de la entidad.
* **getyPos():** Devuelve la posición y (yPos) de la entidad.
* **setyPos(int yPos):** Establece la posición y (yPos) de la entidad.
* **getDx():** Devuelve el desplazamiento en x (dx) de la entidad.
* **setDx(int dx):** Establece el desplazamiento en x (dx) de la entidad.
* **getDy():** Devuelve el desplazamiento en y (dy) de la entidad.
* **setDy(int dy):** Establece el desplazamiento en y (dy) de la entidad.
* **isVivo():** Devuelve el estado de vida (vivo) de la entidad.
* **setVivo(boolean vivo):** Establece el estado de vida (vivo) de la entidad.
* **getStrImg1():** Devuelve el nombre del archivo de imagen 1 (strImg1) asociado a la entidad.
* **setStrImg1(String strImg1):** Establece el nombre del archivo de imagen 1 (strImg1) asociado a la entidad.
* **getStrImg2():** Devuelve el nombre del archivo de imagen 2 (strImg2) asociado a la entidad.
* **setStrImg2(String strImg2):** Establece el nombre del archivo de imagen 2 (strImg2) asociado a la entidad.
* **getStrImg3():** Devuelve el nombre del archivo de imagen 3 (strImg3) asociado a la entidad.
* **setStrImg3(String strImg3):** Establece el nombre del archivo de imagen 3 (strImg3) asociado a la entidad.
* **getIco():** Devuelve el ImageIcon (ico) asociado a la entidad.
* **setIco(ImageIcon ico): Establece** el ImageIcon (ico) asociado a la entidad.
* **getImg():** Devuelve el objeto Image (img) asociado a la entidad.
* **setImg(Image img):** Establece el objeto Image (img) asociado a la entidad.

Imagen que contiene texto, tabla, computer, computadora

Descripción generada automáticamente

## Clase: GrupoAliens*:*

Texto

Descripción generada automáticamente

### Librerías:

* **java.awt.Graphics:** Se utiliza para realizar operaciones de dibujo en el contexto gráfico, como dibujar imágenes.
* **java.util.Random**: Permite generar números aleatorios, utilizado para seleccionar un alien aleatorio que realice un disparo.
* **juego.Main:** No proporciona detalles explícitos sobre esta clase, pero se utiliza para acceder a la variable score de Main.scene.
* **recursos.Audio:** Proporciona métodos para reproducir sonidos en el juego.
* **recursos.Tiempo:** No se muestra explícitamente en el código proporcionado, pero parece relacionarse con el manejo del tiempo en el juego.
* **recursos.Constantes:** Contiene valores constantes utilizados en diferentes partes del juego, como posiciones iniciales, espaciado, velocidades, etc.

### Variables Globales:

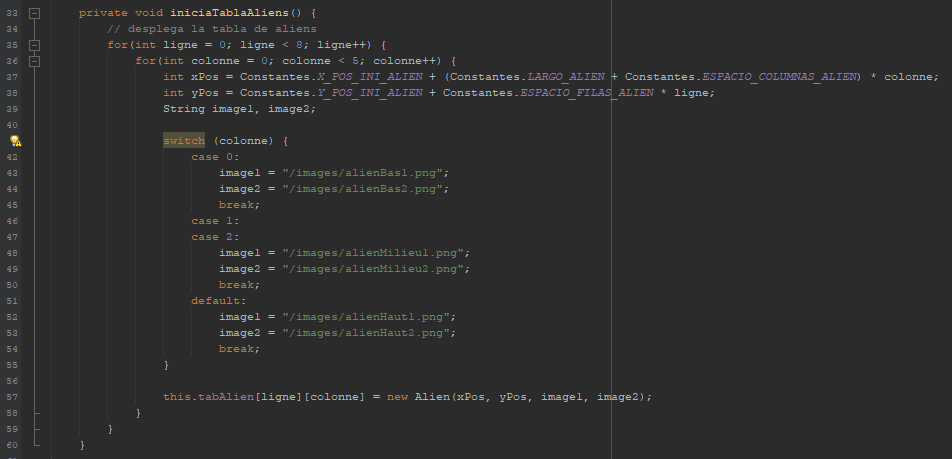
Texto

Descripción generada automáticamente

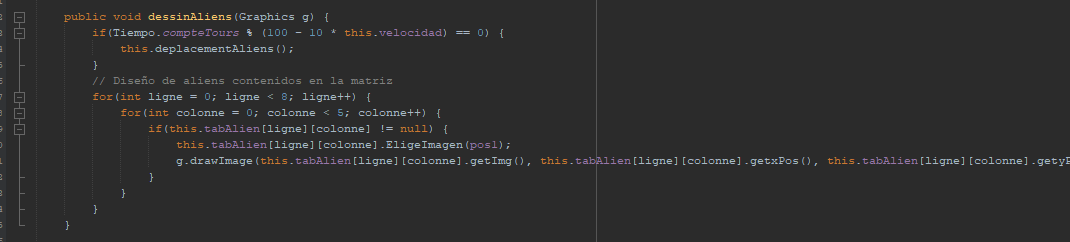
* **tabAlien[][]:** Matriz que contiene a los aliens organizados en filas y columnas.
* **vaAbajo:** Indica la dirección actual de movimiento de los aliens (hacia abajo o hacia arriba).
* **pos1:** Variable utilizada para determinar qué imagen de cada alien se debe mostrar.
* **velocidad:** Velocidad de movimiento de los aliens, definida en Constantes.VELOCIDAD\_ALIEN.
* **tabAlienMuerto[]:** Almacena la posición del alien que ha sido eliminado.
* **nombreAliens:** Número total de aliens en la matriz.
* **contadorSonAlien:** Contador utilizado para reproducir sonidos de alienígenas en un ciclo determinado.

### Métodos:

* **iniciaTablaAliens():** Inicializa la matriz de aliens con posiciones y tipos de imágenes según la columna.



* **dessinAliens(Graphics g**): Dibuja los aliens en la pantalla, seleccionando la imagen adecuada según el estado y posición.



* **toucheHaut():** Verifica si algún alien toca el borde superior de la pantalla.

Texto

Descripción generada automáticamente

* **toucheBas():** Verifica si algún alien toca el borde inferior de la pantalla.

Texto

Descripción generada automáticamente

* **alienTourneEtDescend():** Maneja el movimiento y rotación de los aliens cuando alcanzan los bordes superior e inferior.

Texto

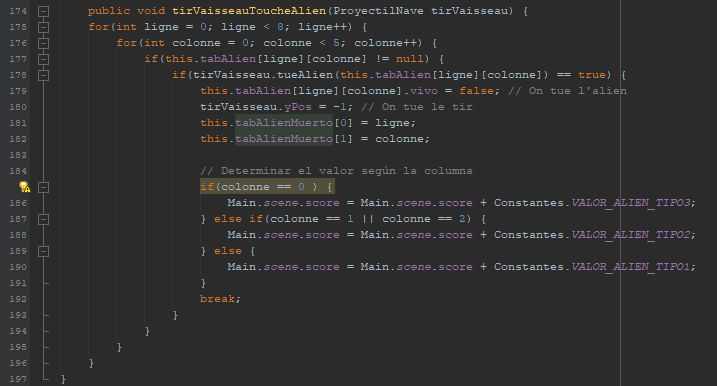
Descripción generada automáticamente

* **deplacementAliens():** Controla el desplazamiento horizontal y vertical de los aliens, gestionando también la velocidad y la reproducción de sonidos.

Texto

Descripción generada automáticamente

* **tirVaisseauToucheAlien(ProyectilNave tirVaisseau):** Detecta si un proyectil disparado por la nave impacta y elimina a un alien, actualizando el puntaje del juego.



* **elimineAlienMort(int[] tabAlienMort):** Elimina un alien de la matriz cuando ha sido derrotado.

Texto

Descripción generada automáticamente

* **choixAlienQuiTire():** Selecciona aleatoriamente un alien vivo que disparará.

Texto

Descripción generada automáticamente

* **joueSonAlien():** Reproduce sonidos de alienígenas en ciclos determinados.

Texto

Descripción generada automáticamente

* **getNombreAliens():** Obtiene el número actual de aliens vivos en la matriz.

Texto

Descripción generada automáticamente

* **positionAlienLePlusGauche():** Calcula la posición más alta en la pantalla ocupada por los aliens en la columna más a la izquierda.

Texto

Descripción generada automáticamente