[Nombre de la compañía] | [Dirección de la compañía]

[Subtítulo del documento]

yerko pinto

2022

Tabla de contenido

[1. Resumen 3](#_Toc104583137)

[1.1. Sumary 3](#_Toc104583138)

[2. Introducción 5](#_Toc104583139)

[2.1. Videojuego 5](#_Toc104583140)

[2.2. LibGDX 7](#_Toc104583141)

[2.3. ECS 7](#_Toc104583142)

[3. Objetivos y características del proyecto 9](#_Toc104583143)

[4. Finalidad 11](#_Toc104583144)

[5. Medios materiales usados 11](#_Toc104583145)

[5.1. Software 11](#_Toc104583146)

[5.1.1. IDE: Visual Studio Code 11](#_Toc104583147)

[5.1.2. Gradle 13](#_Toc104583148)

[5.1.3. Tiled 14](#_Toc104583149)

[5.1.4. GDX texture packer 16](#_Toc104583150)

[5.1.5. Photoshop 17](#_Toc104583151)

[5.1.6. Github 19](#_Toc104583152)

[5.1.7. Mysql / phpMyAdmin 20](#_Toc104583153)

[5.1.8. LibGDX 21](#_Toc104583154)

[5.1.9. Ashley 23](#_Toc104583155)

# Resumen

Se pretende realizar una Alpha del juego *Super Bombermam,* juego desarrollado por *Hudson Soft* , para la consola de videojuegos *SNES (Super Nintendo),* en el año de 1993.

Hecho completamente en *Java*, gracias a la librería de código abierto *LibGDX*, así como otros frameworks como lo son: *BOX2D, BOX2DLigth, Ashley* y *FreeType.* Que vienen incluidos en la librería anteriormente mencionada, adaptados para su correcto funcionamiento. Para esta versión de este juego mundialmente reconocido, se pretende hacer uso de la arquitectura *ECS (Entity Component System),* gracias al framework de *Ashley,* el cual pretende cambiar la forma de trabajar en los videojuegos, pasando de una arquitectura de Objeto (POO) a una orientada a entidades, componentes y sistemas (ECS), para dar a cualquier juego mayor adaptabilidad, y hacerlo más mantenible.

Por lo cual el proyecto se basa en crear una copia del juego de Super *Bombermam* usando, la arquitectura de *ECS* en *Java*.

## Sumary

We pretend to make an Alpha of the game Super Bombermam, a game developed by Hudson Soft, for the SNES (Super Nintendo) video game console, in 1993.

Made in Java, with the LibGDX open source library, as well as other frameworks such as: BOX2D, BOX2DLigth, Ashley and FreeType. That are included in the that library, adapted for its correct operation. For this version of this world-renowned game, it is intended to make use of the ECS (Entity Component System) architecture, thanks to the Ashley framework, which aims to change the way of working in video games, moving from an Object architecture (POO) to one oriented to entities, components and systems (ECS), to give any game greater adaptability, and make it more maintainable.

Therefore, the project is based on creating a copy of the Super Bombermam game using the ECS architecture in Java.

# Introducción

Para la realización de este proyecto se ha planificado primeramente una investigación del de las distintas librerías y frameworks a utilizar, para poder conseguir un mayor grado de acercamiento al titulo original del que se quiere inspirar este proyecto.

Por ende en los siguientes apartados se realizara unas breves explicaciones de algunos conceptos necesarios para entender los principios de la base de nuestro videojuego.

## Videojuego

Con relación al videojuego: *Super Bomberman* es el primer videojuego en la serie de *Bomberman* en aparecer en la consola *Super Nintendo*. También es el primer juego con opción de jugar 4 jugadores simultáneamente.

Este videojuego cuenta con 6 Stages con 7 niveles cada uno en los cuales se posee un TileMap para cada uno de los Stages



Y un Boss final por cada uno de los Stages, así como enemigos individuales

Como se puede ver en las imágenes, nos encontramos con un juego muy complejo.

Luego la dinámica del mismo se basa en la colocación de bombas para abrirte paso y eliminar a los enemigos, consiguiendo puntos y avanzando entre los distintos niveles hasta acabar el juego.

Se eligió este juego debido a su poca complejidad a nivel lógico, y al ser antiguo, se pudo conseguir varios de los elementos que forman al juego, los Sprites, música, etc.

## LibGDX

Como se ha mencionado anteriormente LibGDX es una librería de Java de código abierto basada en el desarrollo de los videojuegos. La cual posee distintos frameworks que se explicaran mas adelante.

Se opto por esta opción debido al lenguaje con el cual esta creado, ya que *Java* es el lenguaje de programación que se ha visto estos dos años con mas detenimiento y es por eso que se tiene una mayor afinidad.

Además se quería usar la mayor parte de los conceptos vistos en clase los cuales son : El modelo MVC, los hilos, POO, desarrollo con base de datos, etc.

## ECS

Como se ha mencionado anteriormente, en este proyecto se hace uso de una arquitectura pensada para el desarrollo de videojuegos, esta es ECS, la cual se explicará posteriormente.

Se quiso aprender mas de esta arquitectura en particular porque, aunque se podía hacer con POO perfectamente, al investigar sobre la resolución del proyecto, se vio la posibilidad de añadir ECS gracias a su facilidad de manejo con el framework de *Ashley* y la buena implementación de esta con LibGDX. Aunque en futuras actualizaciones, este framework paso a ser descartado por otro framework, con más potencia pero más complejo en su ejecución.

Es por ello mismo que se elegio una versión de LibGDX que poseyera este framework para este proyecto. Pero no se descarga la idea de hacer la implementación del nuevo framework de LibGDX dado su gran potencial demostrado en distintos proyectos estudiados para la resolución de este.

# Objetivos y características del proyecto

Como se ha mencionado en un inicio, este proyecto no deja de ser nada mas que una Alpha que intenta emular al juego original. Y dado el tiempo planteado para la resolución del mismo, y aun con todo el esfuerzo implantado, se hace muy difícil conseguir que se asemeje al juego.

Aun así se aspira a tener un proyecto bien encaminado, el cual posea los aspectos básicos del juego original asi como su esencia.

Por otra parte, se inicio este proyecto para poder ver a ciencia cierta que era enfrentarse a la resolución de un juego desde cero y es que aun quitando una de las partes mas desafiantes como lo es la planificación de la idea y el diseño de los niveles y los distintos elementos decorativos e interactivos de un juego. No deja atrás la dificultad luego de al darle vida al mismo gracias a la programación. Es con esa idea que se inicio este proyecto buscando aprender las distintas implementaciones, así como una forma de trabajar, dado que cada motor de juego así como cada lenguaje posee sus distintos distintivos que lo hacen diferentes a los demás. En esencia, podemos decir que al aprender como funciona uno de ello podemos transpolar lo aprendido a otros. Es por ello que se intento hacer el trabajo en el marco de *Java*.

Por otra parte, se intento aprender mas sobre el lenguaje de programación que se uso en el proyecto debido a su robustes y a sus millones de implementaciones. *Java* se ha convertido en un estándar en mas de una empresa a nivel mundial, y es por ello que se quería ver algunos de los aspectos de ella orientado a los videojuegos, aun sabiendo que no es la mejor opción en el mercado a día de hoy. Pero si teniendo en cuenta que se ha usado para crear uno de los juegos mundialmente reconocidos como lo es *Minecraft.*  Un juego de mundo abierto creado inicialmente en Java y que a día de hoy se ha convertido en un juego de culto. Y aun es más, sabiendo que las primeras versiones de ese gran juego se han creado en el seno de las primeras versiones de LibGDX. Aunque posteriormente se cambiaría de motor por uno desarrollado por los creadores del juego.

También cabe destacar que se ha buscado saber mas acerca de la arquitectura de ECS, el cual tiene una cierta peculiaridad que dota a los juegos con gran disponibilidad al cambio y que en sus inicios se lo llamo un estándar para los juegos de RPG o MMORPG, así como los JRPG. Debido a que estos juegos poseen una gran extensión a nivel de personajes, niveles, jerarquías, mapas, armas, etc que en un modelo de POO se hace muy difícil de mantener llegados a cierto punto. Es por ello que se crea ECS el cual intenta generalizar los distintos elementos de un videojuego para poder lograr una mayor adaptabilidad y optimización a la hora de ejecutar el código.

Por tanto, podemos decir que el proyecto nos ha ayudado a ver el desarrollo de videojuegos desde un punto de vista diferente y se ha intentado aprender tanto del lenguaje de programación utilizado como de las distintas funcionalidades que puede dar las librerías open source de java.

# Finalidad

Como se ha mencionado anteriormente, se quiere realizar un Alpha del clásico juego de Super Bomberman creado inicialmente para la SNES.

Como tal no se espera hacer el juego completo ni llegar a implementar todas las funcionalidades que posee el juego original, dado el tiempo así como los recursos utilizados que no son los mas apropiados para el desarrollo de videojuego.

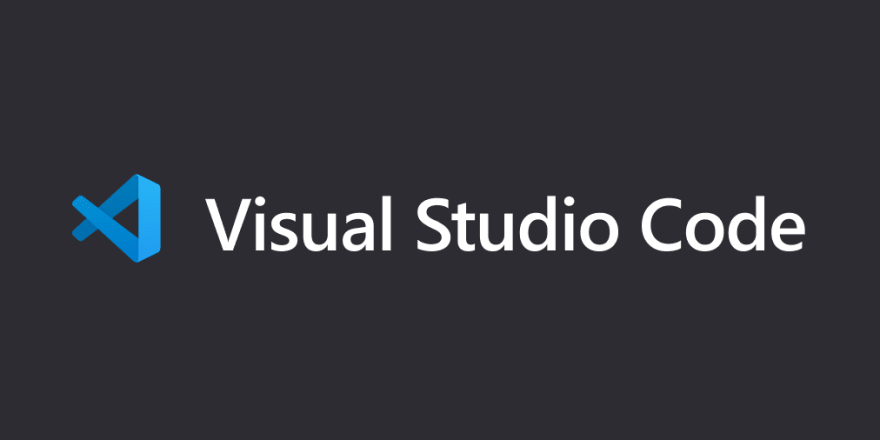
Por ende, se pretende crear un juego funcional en el cual se pueda mover al personaje y destruir algunos elementos del mapa, además de poder eliminar a los enemigos del mapa para encontrar un portal que lleve al nivel siguiente y finalizar con una pantalla de puntuación que se guarde en una base de datos y que se pueda visualizar directamente desde el juego teniendo así un recuento de las puntuaciones de las distintas personas que jueguen a nuestro videojuego, haciendo todas las validaciones pertinentes y buscando principalmente el entretenimiento y la diversión de del jugador.

# Medios materiales usados

En este apartado se hará un recuento junto con una breve explicación de los distintos elementos para la creación del nuestro videojuego.

## Software

## IDE: Visual Studio Code



Uno de los principales elementos a la hora de desarrollar software es un IDE que se ajuste a las necesidades del proyecto, así como uno que sea cómodo para los trabajadores, ya sea por su menor curva de aprendizaje o por su mayor comodidad visual que el usuario en cuestión pueda encontrar en este. Podemos afirmar que un buen IDE ayuda mucho a la ejecución del trabajo y es por eso que nos hemos decantado por *VS Code.*

*Visual Studio Code* es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto,​ aunque la descarga oficial está bajo software privativo e incluye características personalizadas por Microsoft.

Visual Studio Code se basa en Electron, un framework que se utiliza para implementar Chromium y Node.js como aplicaciones para escritorio, que se ejecuta en el motor de diseño Blink. Aunque utiliza el framework Electron, el software no usa Atom y en su lugar emplea el mismo componente editor (Monaco) utilizado en Visual Studio Team Services (anteriormente llamado Visual Studio Online).

Por estas razones y gracias a su plugin de Java, es que se ha podido realizar la totalidad del trabajo en este IDE, configurando el proyecto desde sus inicio con Gradle y haciendo un seguimiento con Git, tecnologías que veremos a continuación

## Gradle



Gradle es un sistema de automatización de construcción de código de software que construye sobre los conceptos de Apache Ant y Apache Maven e introduce un lenguaje específico del dominio (DSL) basado en Groovy en vez de la forma XML utilizada por Apache Maven para declarar la configuración de proyecto.​ Gradle utiliza un grafo acíclico dirigido ("DAG") para determinar el orden en el que las tareas pueden ser ejecutadas.

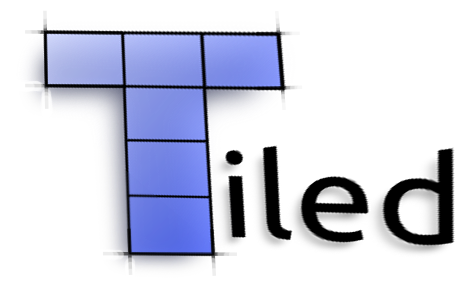
Gradle fue diseñado para construcciones multi-proyecto las cuales pueden crecer para ser bastante grandes, y da apoyo a construcciones incrementales determinando inteligentemente qué partes del árbol de construcción están actualizadas, de modo que cualquier tarea dependiente a aquellas partes no necesitarán ser reejecutada.

Los plugins iniciales están principalmente centrados en el desarrollo y despliegue en Java, Groovy y Scala, pero existen más lenguajes y workflows de proyecto en el roadmap.

Esta tecnología ha sido fundamental para el proyecto, ya no tanto por su sistema de automatización que, por causas de tiempo, no se ha podido exprimir lo que se hubiera querido. Sino por su gran capacidad de gestión de multiproyectos y es que LibGdx al ser una librería que otorga la posibilidad de, ya no solo hacer nuestro juego multiplataforma a nivel de sistemas operativos como Linux, MacOs o Windows. Sino que también permite portearlos para dispositivos móviles, web, e incluso IOS. Es por esta gran gama de dispositivos que se requiere separarlos para poder gestionar de manera mas eficiente el comportamiento que tendrá. Y es ahí en donde Gradle resulta muy útil para gestionar estos proyectos por separado.

Si es verdad que cabe recalcar que, en nuestro juego, solo hemos acogido la posibilidad de ejecutarlos en varios sistemas operativos y no tanto en dispositivos móviles o de web debido al tiempo y a la complejidad a la hora de configurar cada uno de estos dispositivos para que puedan funcionar de manera correcta.

## Tiled

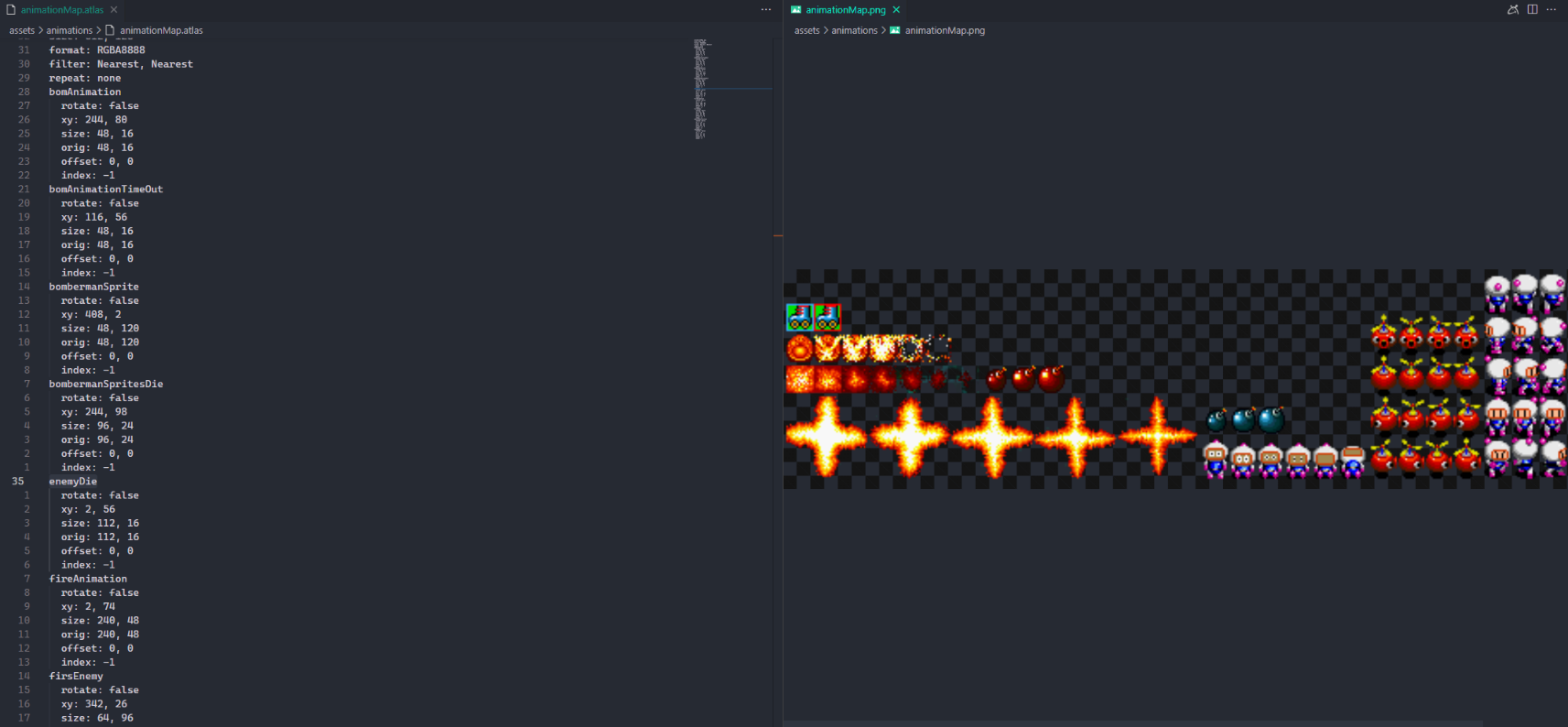


Tiled es un editor de niveles 2D que ayuda a desarrollar el contenido del juego. Su característica principal es editar mapas de mosaicos de varias formas, pero también admite la colocación de imágenes gratuitas, así como formas poderosas de anotar su nivel con información adicional utilizada por el juego. Tiled se enfoca en la flexibilidad general mientras intenta mantenerse intuitivo.

En términos de mapas de mosaicos, admite capas de mosaicos rectangulares rectos, pero también capas isométricas proyectadas, isométricas escalonadas y hexagonales escalonadas. Un mosaico puede ser una sola imagen que contenga muchos mosaicos o puede ser una colección de imágenes individuales. Para admitir ciertas técnicas de falsificación de profundidad, los mosaicos y las capas se pueden compensar con una distancia personalizada y se puede configurar su orden de representación.

Es por esto que esta herramienta ha sido fundamental para el desarrollo de nuestro videojuego. Aunque si es verdad que no es un editor tan complejo como lo es el de Unity u otros, este editor de niveles no tiene nada que envidiarles y su forma de trabajar es muy intuitiva y fácil de aprender.

Su implementación, también cabe destacar, ha sido sencilla gracias a que posee gran compatibilidad con distintos lenguajes entre los cuales no solo se encuentra Java, sino que la librería de LibGDX posee varios métodos para leer los mapas generados por esta herramienta y es que la implementación llega a tal nivel que poseen la misma manera de incorporar SpriteMaps. El cual se basa en el uso de atlas.

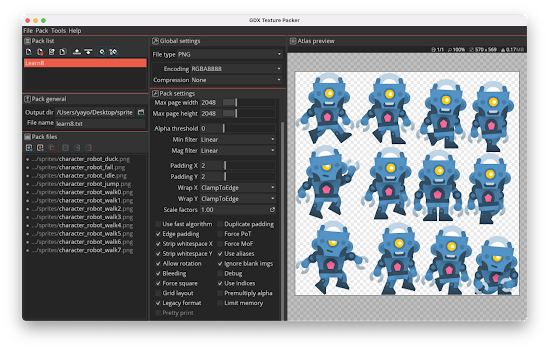
Un atlas no es mas que un archivo de información que contiene todos los estados y posiciones de una imagen de un grupo de imágenes. Esto se puede ver en la siguiente imagen:

Por ejemplo en este atlas que hace referencia a las distintas animaciones usadas en el juego, se pude ver como resguarda la información que tiene el conjunto de imágenes unificado en un *.png* , y de esta manera se puede crear tanto mapas enteros de *png* que luego tengan información para el juego.

## GDX texture packer



Y así como hemos hablado en el apartado anterior en este apartado hablaremos de como se pueden conseguir los atlas. Estos son autogenerados por una herramienta creada por los desarrolladores de la librería de LibGDX la cual es *GDX texture packer*  esta herramienta unifica un conjunto de imágenes que pueden o no ser pertenecer a una secuencia. Ya que está diseñada para realizar el empaquetado de la manera mas optima posible, dando en estas muchas opciones como se puede ver a continuación:



Como se puede ver en esta imagen, ya no solo se puede crear el package propiamente dicho, sino que se puede dar distintas opciones para que se guarden de maneras diferentes. Para el desarrollo de nuestro proyecto se ha optado por una configuración de 1024x1024 en casi todos los mapas de imágenes, esto debido a que algunos poseen mas información en imágenes y requieren una resolución diferente. Por parte del espaciado, siempre se ha dejado en cero y buscando la división en grid para dar mas información a la hora de crear los mapas en Tiled.

## Photoshop



Adobe Photoshop es un editor de fotografías desarrollado por Adobe Systems Incorporated. Usado principalmente para el retoque de fotografías y gráficos, su nombre en español significa "taller de fotos". Es conocido mundialmente. Fue creado en 1986 por los hermanos Thomas Knoll y John Knoll, desde entonces se ha convertido en una marca de uso común, lo que lleva a su uso como un verbo, aunque Adobe desaconseja su uso.

Photoshop puede editar y componer imágenes rasterizadas y soporta varios modelos de colores: RGB, CMYK, CIELAB, colores sólidos y semitonos. Photoshop usa sus propios formatos de archivo PSD y PSB para soportar estas características. Desde junio de 2013, con la presentación de Creative Cloud, el esquema de licencia de Photoshop se cambió al modelo de software como servicio.

Y es que para poder apaliar algunas de las herramientas faltantes que ya te vienen por defecto en Unity, hemos utilizado a esta gran herramienta la cual da nos ha ayudado a la modificación de las imágenes del juego original. Esto es debido a que, en un primer momento las imágenes estaban planificadas para usarse en un sistema de Grid con relación al mapa que esta dividido en una cuadricula en la que cada espacio tiene un tamaño de 16x16. Pero en esta versión . Para dar mas dinamismo al juego y aun usando las texturas del juego original. Este sistema de Grid no ha sido tan bien implementado. Resultando en un cambio de la mayoría de imágenes para quitarles el fondo y en algunos casos cambiando el orden de las imágenes para poder usarlas de manera mas eficiente en nuestro juego, he aquí un ejemplo:

En esta imagen en el lado izquierdo se puede ver como la imagen original le da un fondo a los enemigos para identificarlos, además de pones solo algunos de sus movimientos y en un orden lineal. Es por eso que para este proyecto se le ha dado un orden diferente quitándole el fondo como se pude apreciar en el lado derecho de la imagen.

Esto se ha hecho principalmente por comodidad y optimización, ya que evitamos que el código haga los cambios en tiempo de ejecución y solo hacemos que siga la secuencia que queramos en el momento en el que lo deseemos.

## Github



GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena generalmente de forma pública.

Y es que el control de versiones más usado en el mundo de la informática y es por ello que se ha implementado para poder tener un mejor resguardo de todos los cambios que se han tenido a lo largo de la creación de este proyecto y es que al ser un videojuego. El código resultante del mismo es enorme y difícil de mantener si no se usan herramientas como esta.

## Mysql / phpMyAdmin



MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base de datos de código abierto más popular del mundo,​ y una de las más populares en general junto a Oracle y Microsoft SQL Server, todo para entornos de desarrollo web.

MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius). MySQL AB fue adquirida por Sun Microsystems en 2008, y ésta a su vez fue comprada por Oracle Corporation en 2010, la cual ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de doble licenciamiento anteriormente mencionado. La base de datos se distribuye en varias versiones, una Community, distribuida bajo la Licencia pública general de GNU, versión 2, y varias versiones Enterprise, para aquellas empresas que quieran incorporarlo en productos privativos. Las versiones Enterprise incluyen productos o servicios adicionales tales como herramientas de monitorización y asistencia técnica oficial. En 2009 se creó un fork denominado MariaDB por algunos desarrolladores (incluido algunos desarrolladores originales de MySQL) descontentos con el modelo de desarrollo y el hecho de que una misma empresa controle a la vez los productos MySQL y Oracle Database.

Está desarrollado en su mayor parte en ANSI C y C++.​ Tradicionalmente se considera uno de los cuatro componentes de la pila de desarrollo LAMP y WAMP.

MySQL es usado por muchos sitios web grandes y populares, como Wikipedia, Google​(aunque no para búsquedas), Facebook, ​Twitter,​ Flickr,​ y YouTube.​

Por estas razones es por lo que se ha decidido implementar esta base de datos en el proyecto ya que para los datos a guardar en la misma solo es requerido un cliente y la propia base de datos.

## LibGDX



LibGDX es un librería para el desarrollo de videojuegos multiplataforma, soportando actualmente Windows, Linux, Mac OS, Android, IOS y HTML5

Uno de los objetivos principales de la biblioteca es mantener la simplicidad, sin renunciar al amplio abanico de plataformas finales. Para ello, permite escribir el código base en un único proyecto y exportarlo a las tecnologías mencionadas anteriormente sin modificación alguna. Pudiendo utilizar la versión de escritorio como entorno de pruebas para el resto, siguiendo así una iteración de desarrollo rápida e integrable con el resto de herramientas de Java.

Además, LibGDX permite bajar el nivel de abstracción tanto como se quiera, dando acceso directo al sistema de archivos, dispositivos de entrada y audio, e incluso a las interfaces de OpenGL ES 2.0 y 3.0.

Encima de esta capa de interacción, se establece un potente conjunto de APIs que permite mostrar imágenes y texto, construir interfaces de usuario, reproducir sonidos o música, realizar cálculos matemáticos, parsear ficheros XML o JSON, etc.

LibGDX pretende ser un librería más que un motor de desarrollo, admitiendo que no es una solución todo en uno, sino que provee al programador de un potente conjunto de abstracciones y le dan total libertad para desarrollar su aplicación.

Dados estos atributos y dada su gran mantenimiento con respecto a otras librerías al igual que su documentación, hicieron de esta la mejor de las opciones para este proyecto, y es que aun siendo una Alpha, es gracias a esta librería que se ha podido llegar a una aproximación muy elaborada al que es el juego original.

## Ashley

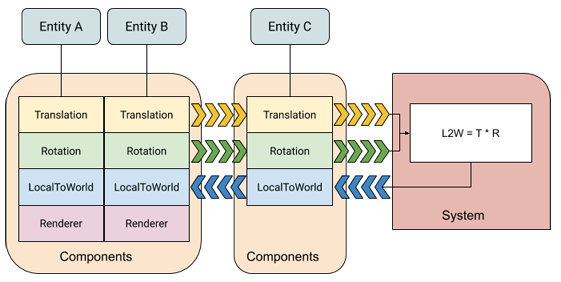
Ashley es uno de los frameworks que incluye LibGDX en su repertorio de herramientas. Este en especifico se utiliza para la implementación de la arquitectura de ECS esta, explicándola mas detalladamente:

Es una estructura que intenta dividir la carga de trabajo del equipo, formando grupos de Entidades, estas pueden ser cualquier elemento del juego ya que solo son marcos que ni siquiera poseen información por ellas mismas. De esto se encargan los componentes, que no son mas que contenedores de datos, generalmente generales, a esto nos referimos que pueden ser reutilizados, puede ser por ejemplo un *MoveComponent* el cual solo tenga la información que se requiera para moverse, en este caso las coordenadas de x e y.

Como se puede ver este componente luego puede ser usada por nuestro player o incluso nuesto enemigo que, aunque su sistema de movimiento sea diferente, si va a necesitar de los mismos atributos para moverse.

Posteriormente los sistemas serás los encargados de manipular esta información según nosotros lo programemos. De esta manera al finalizar, poseemos componentes reutilizables que al ponerlo en una entidad en concreto, reaccionará a un sistema del juego para tener un comportamiento distinto en el mismo.

Esto se puede ver en el siguiente diagrama:



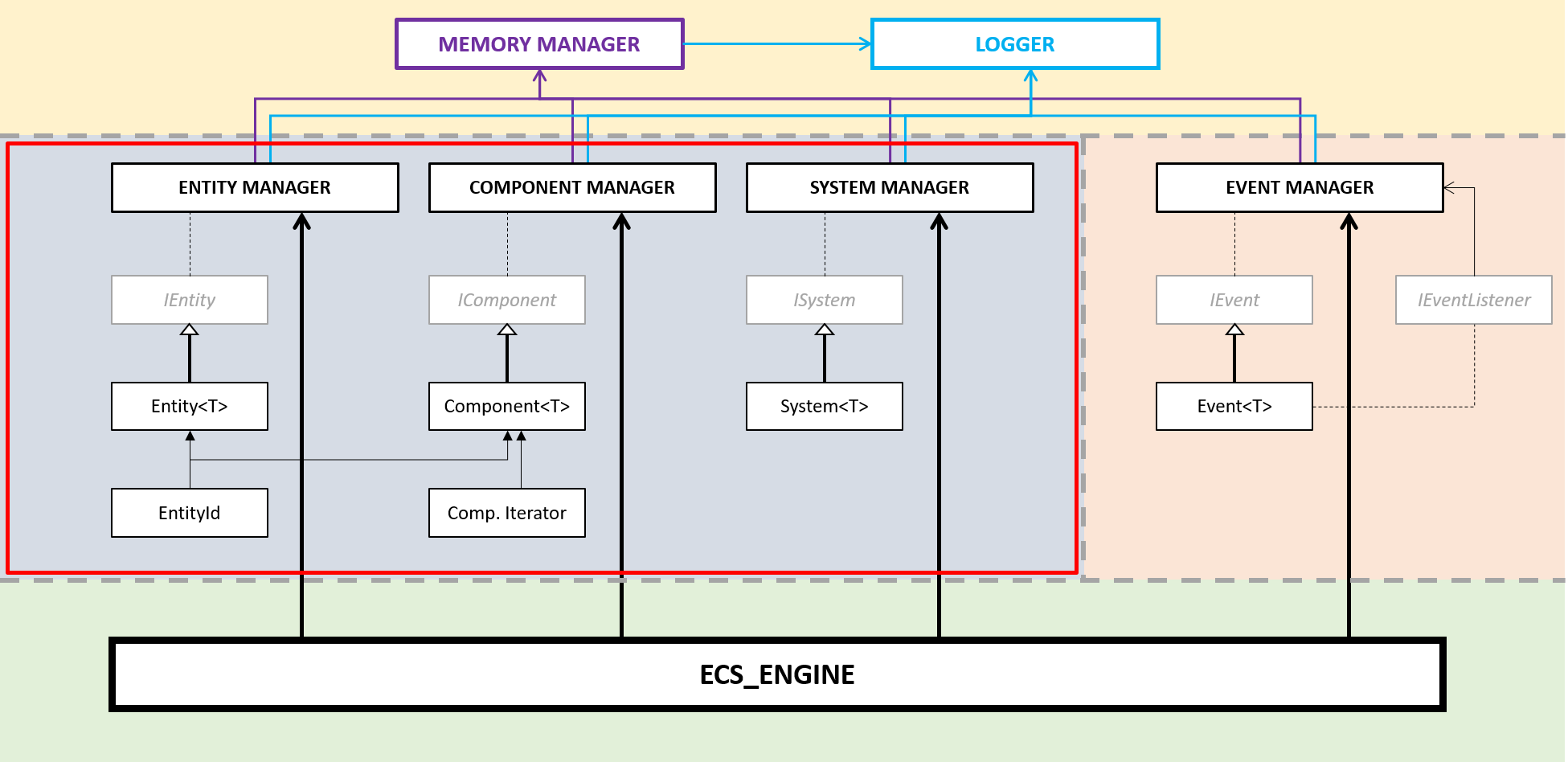
Esta arquitectura se puede desarrollar de muchas maneras diferentes y es que Ashley tiene una manera específica de hacerlo, pero hay muchas otras, por ejemplo, el que se usa en las últimas versiones de LibGDX *Artemis-ods*  es un framework de ECS pero con muchas mas opciones y mas completo que Ashley.

Pero en este proyecto se eligió esta ultima debido a su gran facilidad de manejo y gran adaptabilidad para distintos tipos de proyectos.

En este caso Ashley, apuesta por un Engine que maneje todo el marco de ECS y se encargue de gestionar tanto las entidades como los sistemas y los componentes usando el Deltatime del juego.

Así mismo deja la opción de crear un Engine o un PoolEngine, los cuales se diferencian en eficiencia a la hora de lidiar con varias entidades al mismo tiempo, cosa la cual puede servir mas en un juego shotter que se crean y destruyen entidades continuamente.

Esto se puede ver en el siguiente diagrama :



En el se puede ver mejor la gestión de Ashley con las entidades.