Práctica 1: Mastermind

1. Introducción

Esta práctica se compone en un total de 6 módulos distintos, dos de ellos siendo launchers para diferentes plataformas (PC y Android respectivamente), tres ellos siendo el motor básico y sus adaptaciones para cada una de las versiones, y por último tenemos la lógica del juego en sí.

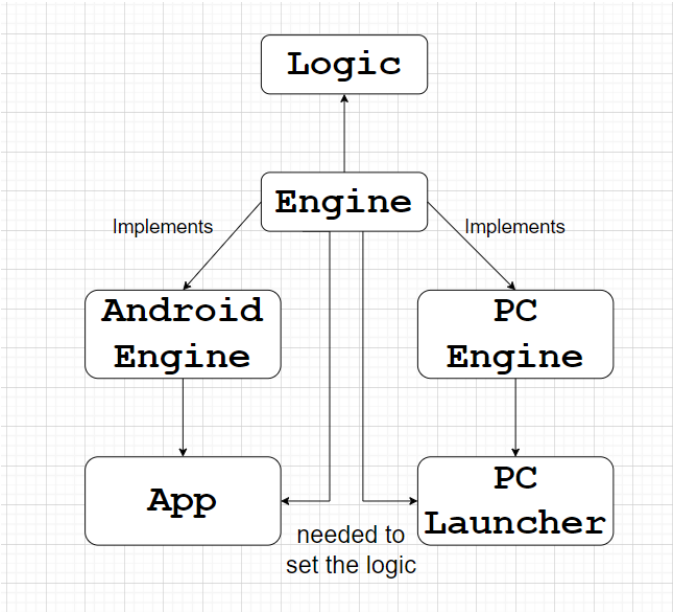

Diagrama de dependencias de los módulos del proyecto

En esta imagen se puede observar los diferentes módulos y la comunicación entre ellos, como se puede ver la lógica se comunica con los launchers pero es para después definirse en los diferentes motores como el "juego" que se lanza.

A continuación se explicará en detalle el contenido de cada módulo así como las clases que los forman.

2. Módulos e implementación

2.1. Lógica

Este módulo representa la funcionalidad del juego en sí. La representación de escenas, el gestor de escenas (Logic)y los diferentes elementos de las escenas (Game objects).

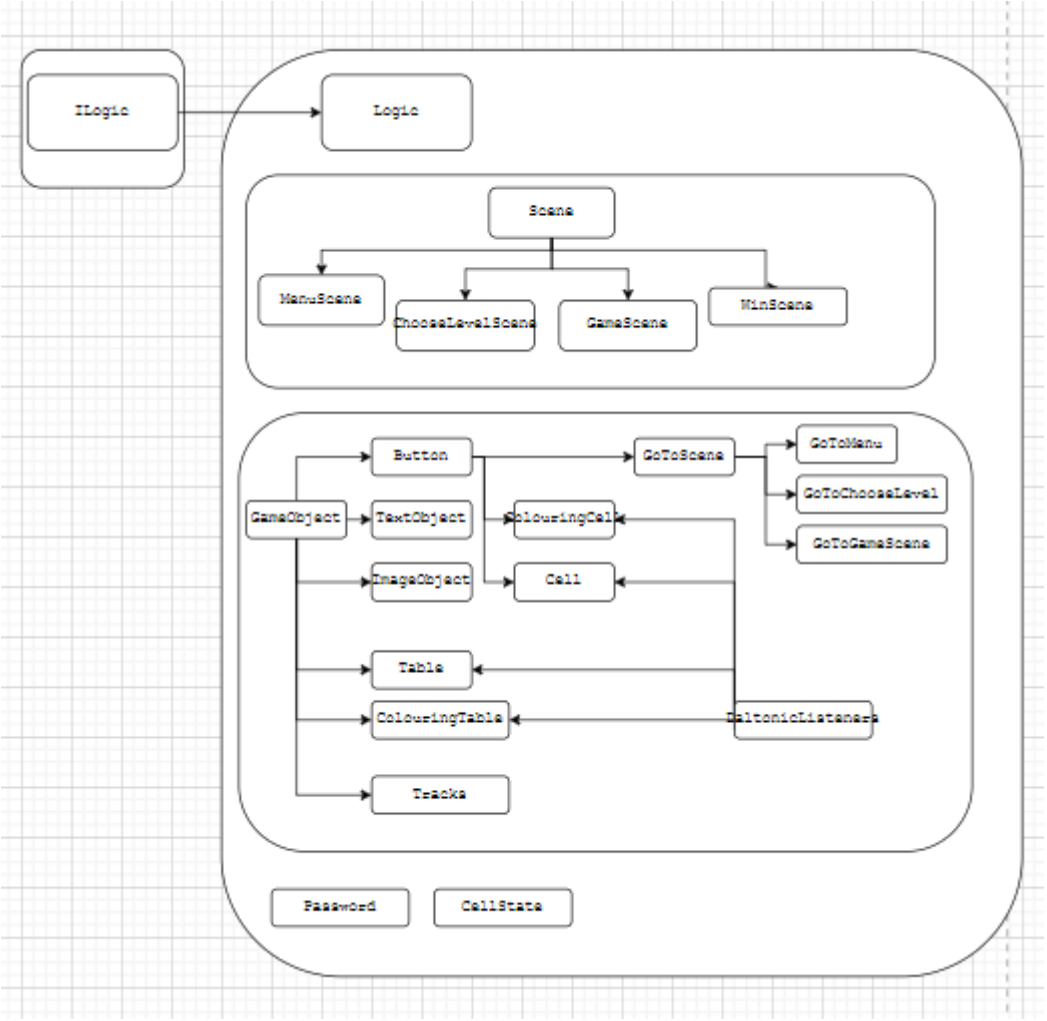

Diagrama de clases que están en el módulo de Lógica

- **Clases de Logica**
- ***Logic:*** Implementa la interfaz de ILogic del módulo de engine para funcionar como gestor de escenas y base del juego al motor.

Escenas

- ***Scene:*** Clase abstracta que funciona como contenedor de Game Objects
- ***Menu Scene:*** Escena del menú principal
- ***Choose Level Scene:***Escena de elegir dificultad
- ***GameScene:***Escena del juego en sí
- ***Win Scene:***Escena de ganar o perder

GameObjects

- ***GameObject:*** Clase abstracta básica que representa un objeto de juego
- ***Image Object:*** Objeto que renderiza una imagen
- ***Text Object:*** Objeto que renderiza un texto con una determinada fuente
- ***Button:*** Objeto abstracto que al tocarlo hace una accion (de la cual heredan muchos objetos)
  - ***Go To Scene:*** Clase que al pulsar va de una Escena a otra(hay varios de estos)
  - ***Cell:*** Clase que representa cada una de las casillas del juego (circulo gris)
  - ***Tracks:*** Clase que representa las pistas (círculos rojos)

  - ***Colouring Cell:*** Clase que al pulsar, colorea la primera celda disponible.
  - ***Daltonic Button:*** Botón con dos imagenes que al pulsarlo activa el modo daltonismo. El cual hace que sobre las casillas rellenas y los colouring buttons se ponga en lugar de un color, un número. Para la comunicación de este modo se ha implementado una interfaz *Daltonic Listener* la cual al informar a la escena de haberse activado, informa a todos los elementos "escuchantes".

- ***Table:*** Clase que representa cada uno de los intentos en sí
- ***Colouring Table:*** Clase que contiene los *Colouring Cells*

2.2. Motor Común (Engine)

El motor común consta de la definición básica de las funcionalidades que tiene que ir para cada uno de los motores concretos, así como de la comunicación con lógica (por ejemplo: dibujar una imagen de un botón)

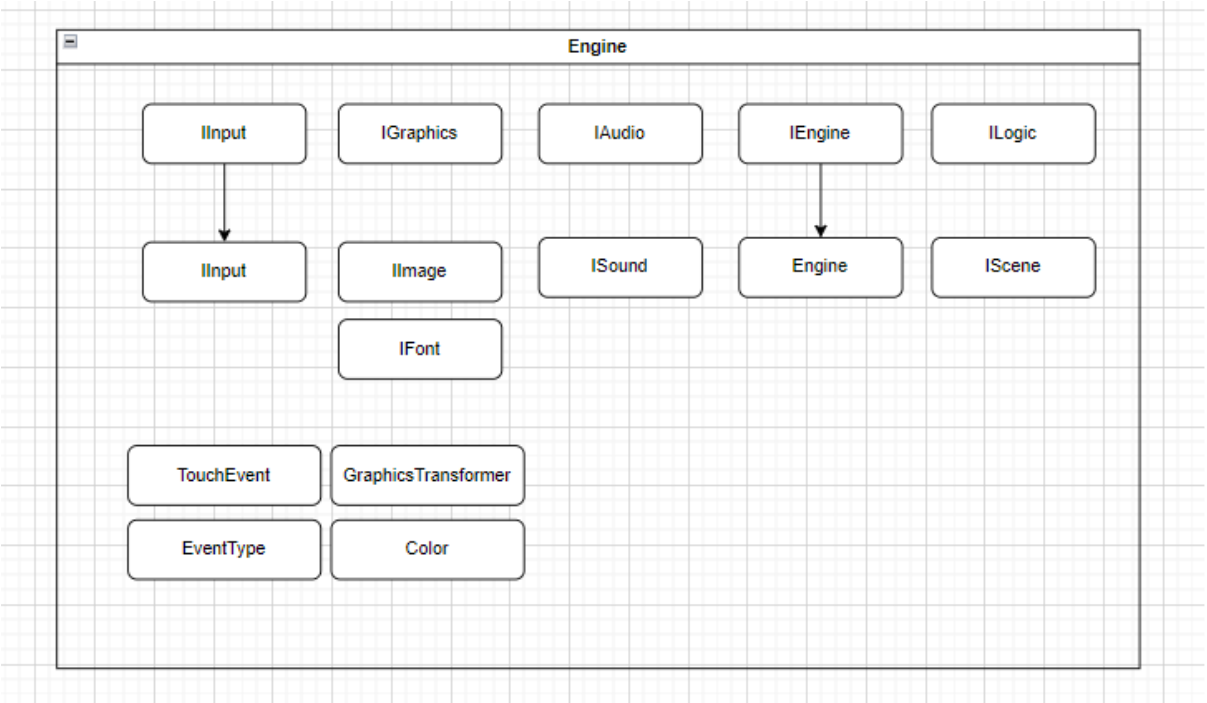

Diagrama de clases que están en el módulo de Engine

- **Clases del motor común**
- ***IEngine:*** Interfaz que contiene cada uno de los managers de las funcionalidades(graphics, audio, logic e input)
- ***Engine:*** Implementa la interfaz anterior para poder asignar y obtener estos managers.

- ***IGraphics:*** Interfaz que define los métodos de dibujado
- ***IImage:*** Interfaz que define una imagen
- ***IFont:*** Interfaz que define una fuente de escritura.
- ***Color:*** Clase que nos presenta un modelo de color rgba y nos lo abstrae al renderizado (que utiliza argb con hexadecimales y para colores concretos nos manejamos mejor con rgb y rango de 0 a 255)

- ***GraphicsTransform:*** clase que se encarga del redimensionado de la "pantalla lógica" respecto a la pantalla del dispositivo (actualizado cuando se gira el móvil o cuando se redimensiona la pantalla en PC)

- ***IAudio:*** Interfaz que define las funcionalidades de sonido
- ***ISound:*** Interfaz que representa un efecto de sonido

- ***IInput:*** Interfaz que define la cola de eventos de input (toques en la pantalla o clicks de ratón)
- ***Input:*** Clase que implementa la interfaz IInput
- ***TouchEvent:*** Clase que representa un evento de input contiene las coordenadas de la pantalla, las coordenadas de la pantalla lógica y el tipo de evento

- ***ILogic:*** Interfaz que define el core del juego.

2.3. Motor de Android (Android Engine)

Este módulo consiste en la implementación del motor adaptado para la plataforma de android

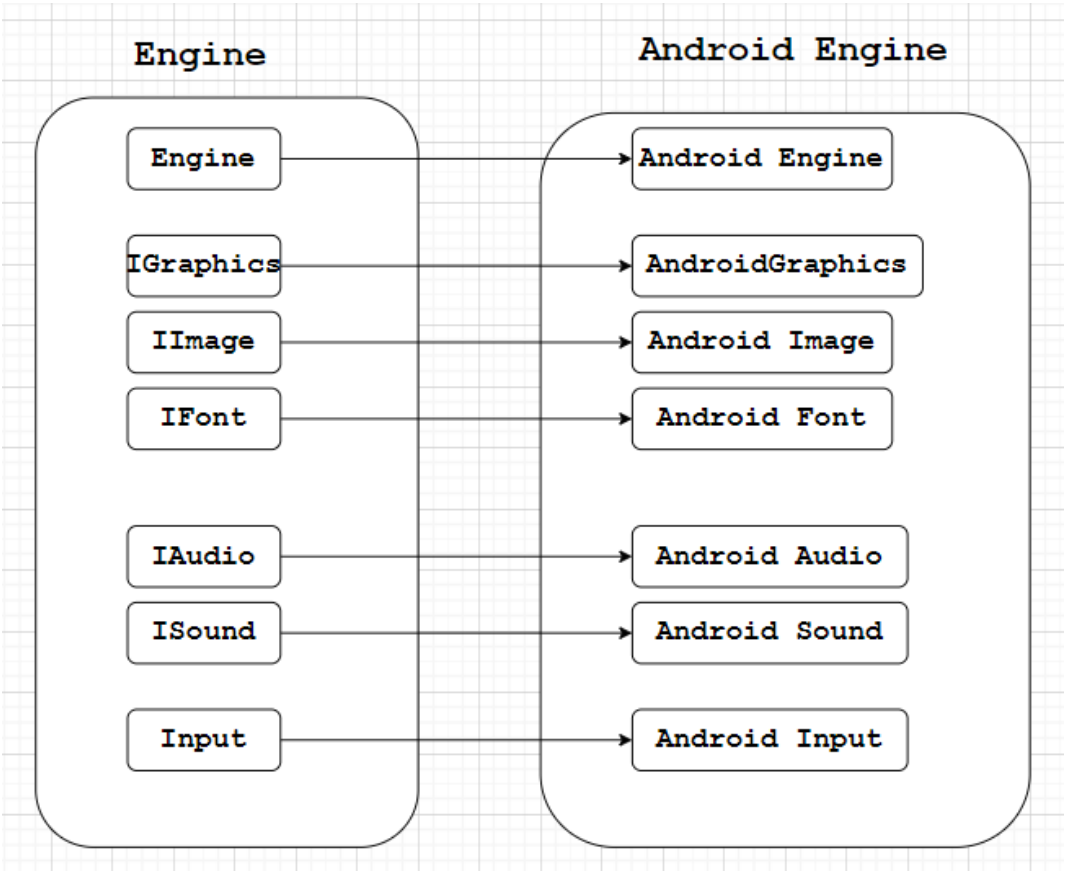

Diagrama con la herencia de las clases que están en el módulo de Android Engine

Clases de Android Engine:

- **Android Engine:** Hereda de Engine e implementa Runnable, crea los otros managers de android y representa la clase con el bucle principal del juego

- **Android Graphics:** Implementa los métodos de *IGraphics* con las librerías de Android
- **Android Image:** Implementa la interfaz IImage usando los Bitmaps de Android.
- **Android Font:** Implementa la interfaz IFont usando las Typefaces de Android.

- **Android Audio:** Implementa la interfaz IAudio usando SoundPool para efectos de sonido cortitos
- **Android Sound:** Implementa la interfaz ISound para efectos de sonido.

- **Android Input:**Hereda de Input e implementa View.OnTouchListener

2.4. Motor de PC (PCEngine)

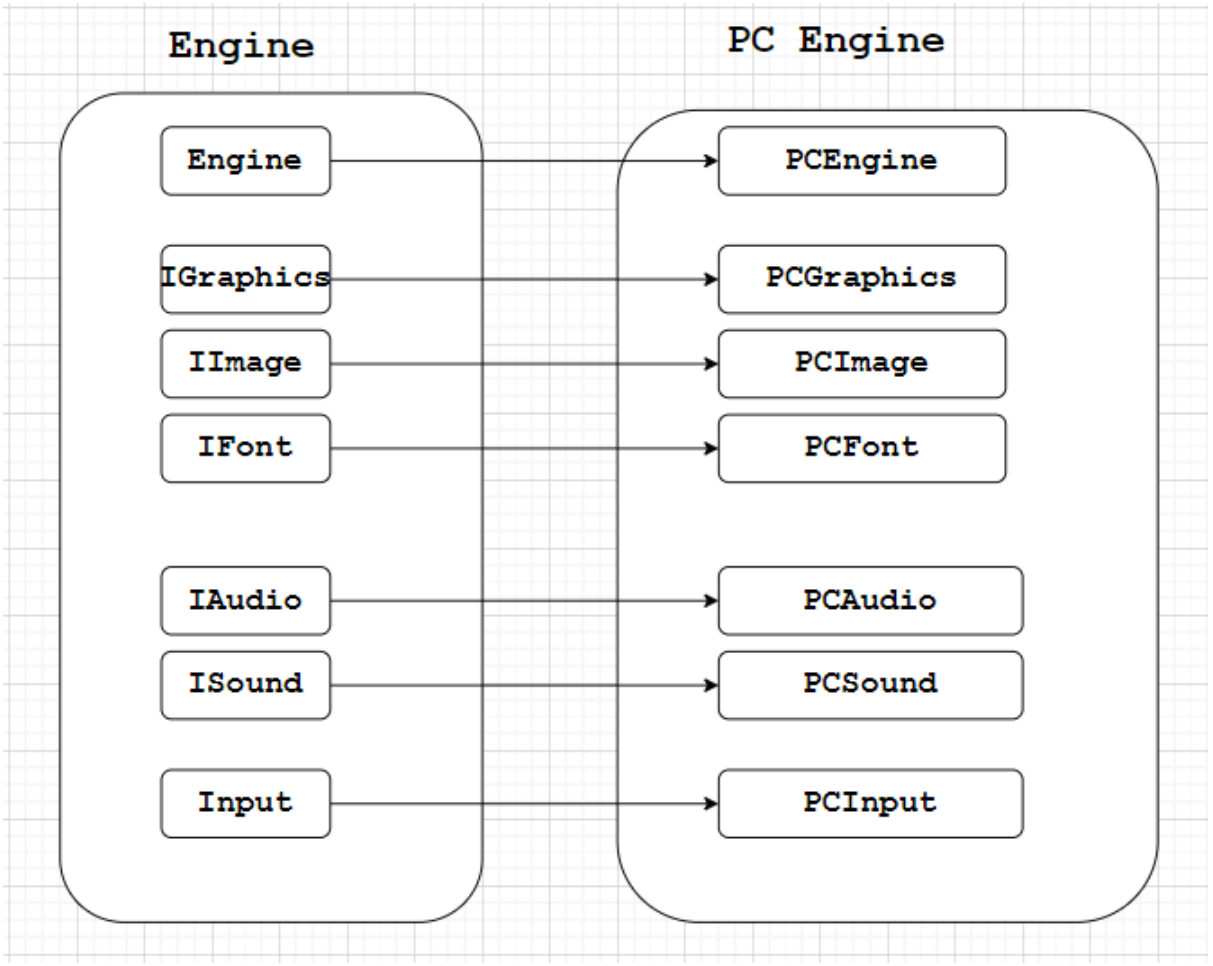

Diagrama con la herencia de las clases que están en el módulo de PC Engine

Clases de PC Engine:

- **PC Engine:** Hereda de Engine e implementa Runnable, crea los otros managers de android y representa la clase con el bucle principal del juego

- **Android Graphics:** Implementa los métodos de *IGraphics* con las librerías de Android
- **Android Image:** Implementa la interfaz IImage usando los Bitmaps de Android.
- **Android Font:** Implementa la interfaz IFont usando las Typefaces de Android.

- **Android Audio:** Implementa la interfaz IAudio usando SoundPool para efectos de sonido cortitos
- **Android Sound:** Implementa la interfaz ISound para efectos de sonido.

3. Referencias

Este apartado se usa como créditos a los recursos usados en la práctica:

- Sonidos : *Miximil en itchio*
- Fuentes: *Falling for autumn* y *Shade june*
- Imágenes:Flaticon