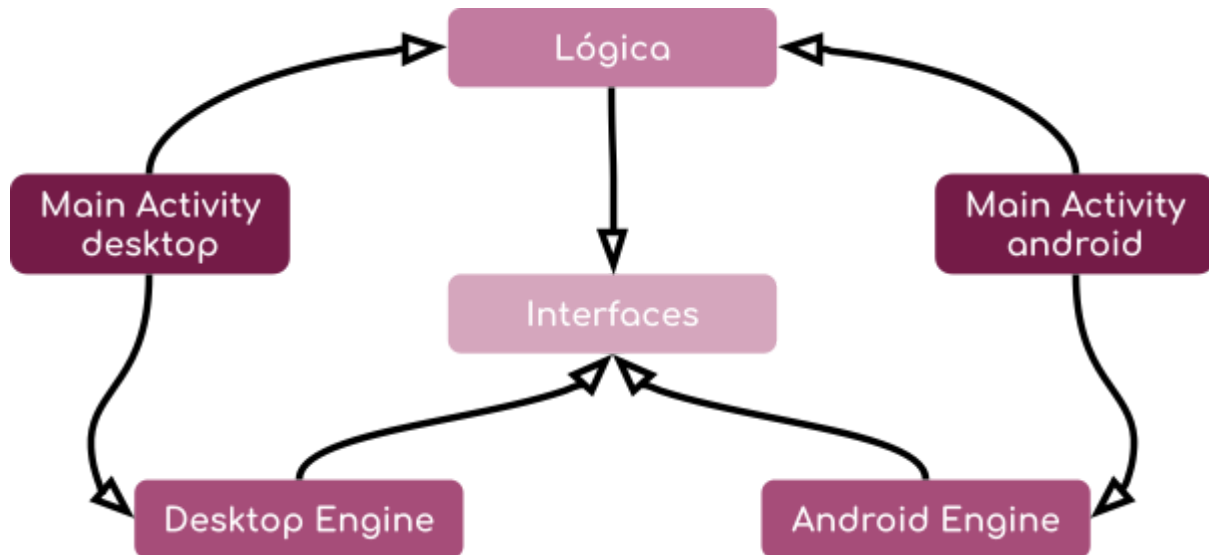


ESTRUCTURA DE LOS MÓDULOS



Interfaces que implementan los motores para que las use la lógica:

- **IAudio**-> AudioDesktop, AudioAndroid
- **IEngine**-> EngineDesktop, EngineAndroid
- **IFont**-> FontDesktop, FontAndroid
- **IGraphics**-> GraphicsDesktop, GraphicsAndroid
- **Image**-> ImageDesktop, ImageAndroid
- **ISound**-> SoundDesktop, SoundAndroid
- **IInput**-> InputDesktop, InputHandler(Desktop/Android), InputAndroid

Interfaces que implementa la lógica para que las usen los motores:

- **IGameObject**-> Circles y Board
- **IScene**-> EndScene, GameScene, Menu Scene

MÓDULOS DE LA PRÁCTICA:

- **AndroidEngine**: Módulo que implementa el motor para la ejecución en Android. Tiene las siguientes clases:
 - **AudioAndroid**: Se utiliza para facilitar la creación, gestión y reproducción de objetos SoundAndroid.
 - **EngineAndroid**: Es el motor de juego de Android. Se encarga de inicializar los recursos necesarios para el resto de motores así como gestionar escenas, la interfaz gráfica, la entrada del usuario, el audio, y, en general, la ejecución del juego mediante los métodos: "resume", "pause" y "run".
 - **FontAndroid**: Se utiliza para representar y gestionar fuentes de texto en Android a través de un objeto Typeface. Permite cargar

una fuente de texto desde el sistema de archivos de la aplicación y ajustar sus atributos, como el tamaño, la negrita y la cursiva.

- **GraphicsAndroid:** Se utiliza para proporcionar métodos encargados de dibujar formas, crear y mostrar imágenes, cargar fuentes de texto y proyectar los cambios en el objeto Canvas. También se encarga de gestionar la escala en caso de que cambie el tamaño de la ventana.
 - **ImageAndroid:** Se utiliza para representar imágenes en formato de mapa de bits a través de un objeto Bitmap en Android. También permite obtener propiedades de la imagen, como su alto y su ancho.
 - **SoundAndroid:** Se utiliza para cargar sonidos en la aplicación mediante el uso de un SoundPool. La clase ofrece un método para obtener el identificador del sonido cargado, de forma que pueda ser gestionado por AudioAndroid.
 - **InputHandler:** Se utiliza para procesar los eventos táctiles de la aplicación guardando información como el tipo de evento que es, o las coordenadas en las que se ha detectado. Utiliza una lista de eventos pendientes y un pool de eventos para administrar la entrada de manera que, los objetos de nuestra clase TouchEvent, puedan ser reutilizables.
 - **InputAndroid:** Se encarga de recopilar los eventos pendientes del InputHandler, los agrega a su propia lista y luego los borra del InputHandler. Además permite a otras partes de la aplicación acceder a los eventos registrados.
- **AndroidGame:** Módulo puente entre la lógica y el motor de Android que se encarga de ejecutar la lógica del juego en el motor AndroidEngine.
 - **DesktopEngine:** Módulo que implementa el motor para la ejecución en PC. Tiene las siguientes clases (no se han descrito porque hacen lo mismo que en Android):
 - AudioDesktop
 - EngineDesktop
 - FontDesktop
 - GraphicsDesktop
 - ImageDesktop
 - SoundDesktop
 - InputDesktop
 - InputHandler
 - **DesktopGame:** Módulo puente entre la lógica y el motor de PC, que se encarga de ejecutar la lógica del juego en el motor DesktopEngine.
 - **Engine:** Módulo abstracto que define las interfaces para comunicar la lógica con los motores
 - IAudio
 - IEngine
 - IFont
 - IGraphics
 - IImage

- ISound
- IInput
- IGameObject
- IScene
- **GameLogic:** Módulo que implementa la lógica del juego. Contiene las siguientes clases:
 - **Board:** Clase que gestiona la lógica del tablero, los intentos, y contiene los círculos de la solución y los colores disponibles
 - **Button:** Implementación general de los botones
 - **ButtonDaltonics:** botón que activa/desactiva los números encima de los círculos
 - **ButtonLevel:** botón que cambia de niveles
 - **ButtonImage:** Implementación general de los botones con imagen
 - **Circle:** Game object que implementa de forma general los círculos para el tablero del juego
 - **HintsCircle:** círculos blancos/negros de las pistas
 - **TryCircle:** círculos de los intentos del jugador
 - **SolutionCircle:** círculos de colores para los intentos
 - **EndScene:** Escena del final (ganar/perder) del juego
 - **GameScene:** Escena de partida
 - **MenuScene:** Escena inicial del juego
 - **LevelScene:** Escena de selección de nivel
 - **GameManager**
 - **Solution:** Implementa la creación de la solución y comprueba los intentos del jugador
 - **GameInit:** Almacena las variables de cada dificultad de partida y las distintas dificultades que puede tener el juego
 - **SceneNames:** Enum con los nombres de las distintas escenas