# Especificación de la Máquina de estados.

La clase Juego tiene como miembro un objeto Escena que es una clase base interfaz que implementan las clases que representan los diferentes estados y que heredan de esta. Estas clases además son Singleton, lo que implica que solo podrán ser instanciadas una vez.

Las clases que heredan de Escena son:

* ECinematica
* Emenu
* EIntro
* EInGame
* EMenuInGame

Todas estas clases tienen las variables de instancia:

* Juego\* \_context: Representa el puntero al objeto Juego que instanció el estado.
* Bool initState: Indica si el estado ya ha sido inicializado.

Todas estas clases Escena comparten los siguientes métodos:

* ECinematica \* Instance(Juego\* context): Recibe el puntero al Juego, Construye y devuelve la instancia Singleton del estado. Para recuperar el puntero a este estado, hay que utilizar este método.
* Void Handle(): Accede al Juego que contiene el estado a través de \_context y a través de su método **setState()** actualiza el estado activo en el Juego.
* void Init(): En este método se declaran e inicializan todos los elementos necesarios en el estado.
* void Update(): Contendrá el trozo de lógica que deberá ejecutar ese estado. Si el objeto todavía no ha sido inicializado (definido por su variable initState, se llamará a **Init()** y continuará la ejecución.
* void Render(): Se mandan al window de Juego las órdenes de dibujado.
* Juego\* getContext(): Devuelve el objeto Juego que instanció el estado.

Preguntas para Fidel:

1. ¿Se podría hacer sólo un método que estuviera por ejemplo en Escena? ¿Y/o en Juego?
2. ¿Está bien la estructura de métodos planteada hasta ahora para la máquina de estados?
3. ¿Dónde debería ir el método input?