YO INTERNETO

Integrantes:

Iván Zabaleta Almeida

Mario Gallego Cano

Descripción del juego

Yo Interneto es un videojuego basado en un sistema de combate por turnos.

El tema de este videojuego toma como base el podcast del mismo nombre.

El podcast está compuesto por los streamers Orslok, Darío y Cheeto, los cuales tomamos como personajes principales para el juego.

Para los enemigos y mundos nos hemos decantado por un tema satírico relacionándolo con la sociedad y memes actuales. Teniendo disponibles los mundos: la calle, la playa y la jungla. Y los enemigos: los MDLR, Frank de la jungla y los Guiris.

En relación con la jugabilidad, se nos presentan los 3 personajes principales a elegir con sus respectivas estadísticas. Una vez elegido iremos de un mundo a otro encontrándonos distintos enemigos. Dependiendo del mundo en el que estemos tanto nosotros como los enemigos tendremos ventajas y desventajas.

Una vez seleccionado un mundo y un enemigo se iniciará el sistema de combate. Para empezar, se elige cara o cruz y se tira una moneda, el que acierte tendrá la ventaja de atacar primero.

Todos los personajes tienen unas acciones básicas: Atacar, Defenderse, Beber Mahou, coger un objeto y ver las Stats.

Además, cada uno tendrá una acción extra, la cual les ayudará sanando 10 de vida o restaurándoles a su estado activo en caso de estar paralizados.

Por otro lado, el enemigo decidirá en cada turno si atacarte o preparar su defensa. En caso de atacarte, decidirá entre 3 tipos de ataque.

Una vez elimines a un enemigo se cambiará automáticamente a otro mundo y otro enemigo con sus correspondientes ventajas. Esto se repetirá hasta que el personaje principal se quede sin vida, dando fin al videojuego.

Manual de uso

La selección de acciones se relizará enteramente por terminal mediante el uso de números.

En caso de querer terminar el juego en cualquier momento tendremos la opción de pulsar <7>, lo cual finalizará la ejecución de este.

El personaje puede tener 3 estados:

* Activo:

El personaje puede realizar acciones con normalidad.

* Herido:

El personaje se encuentra afectado por un ataque de veneno, durante 2 turnos recibirá 5 de daño.

* Paralizado:

El personaje ha sido paralizado por un ataque helado o por beber Mahou. No puede realizar más acciones que <4> para restablecerle a su estado Activo.

Acciones personaje:

* Atacar:

Intentas atacar a un enemigo con tu máximo de fuerza, en caso de llevar un objeto extra, como el martillo, que añada fuerza, esto se sumará a tu daño.

* Defenderse:

Preparas tu defensa para el siguiente turno. Si en el siguiente turno el enemigo decide atacarte dependerá de tu destreza como porcentaje para esquivar dicho ataque.

* Beber Mahou:

Te sanas 20 de vida, pero al beber alcohol, te quedas paralizado.

* Coger un Objeto:

Se te muestran los objetos disponibles que te amplificarán estadísticas de armadura, destreza o ataque. Estos objetos permanecerán contigo hasta el final de la partida.

* Ver Stats:

Se muestra en pantalla la información sobre las estadísticas tanto tuyas como del enemigo, indicando vida, ataque, destreza y estado.

Si el personaje está paralizado tras un ataque o beber Mahou las acciones no disponibles se encontrarán entre “~” (ejemplo <1> ~ Atacar ~). Por otro lado, la acción extra <4> siempre estará disponible, puesto que es necesaria para volver a la normalidad. Al igual que la acción Stats, ya que esta no gasta el turno.

Acciones enemigo:

* Normal:

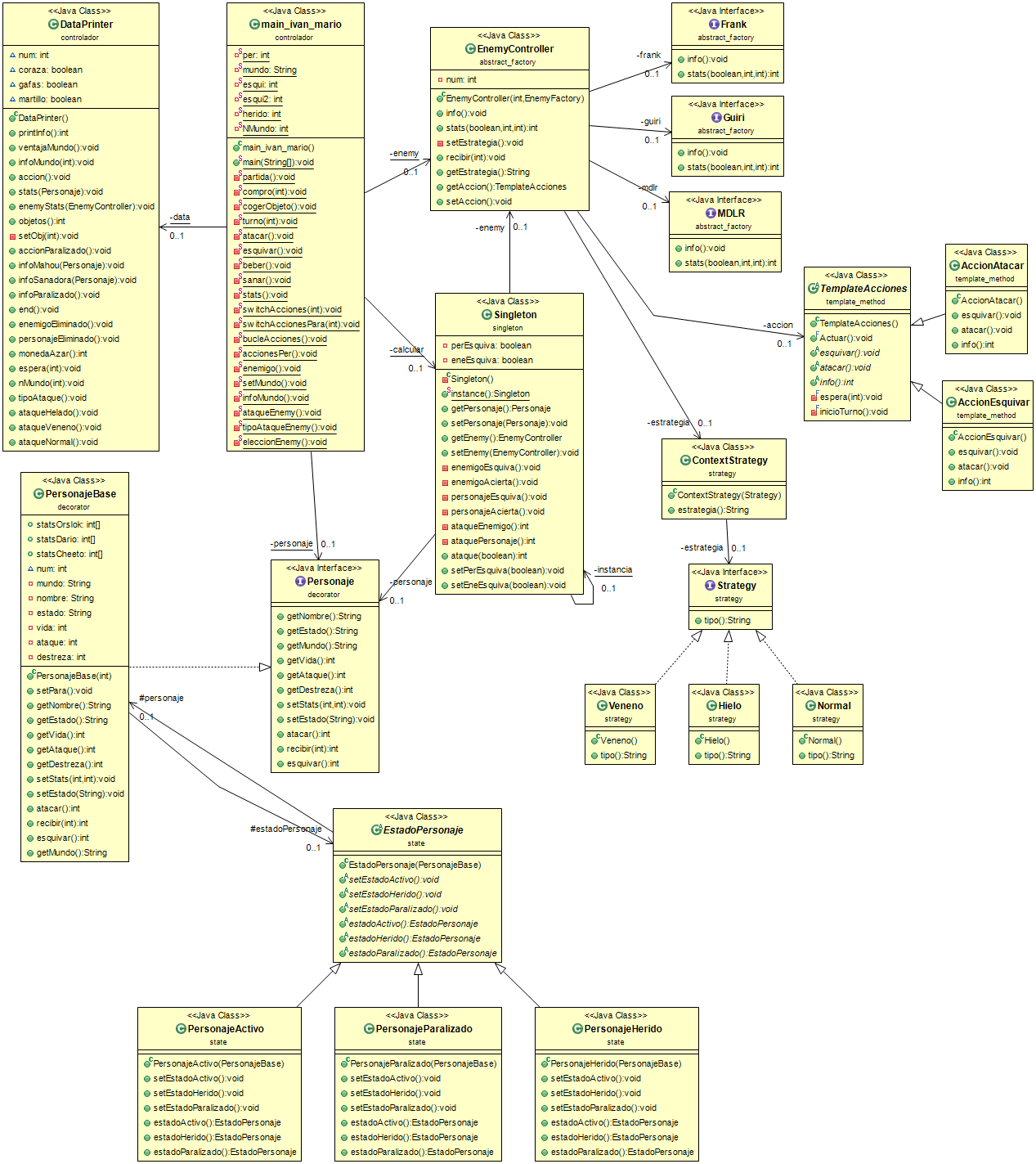
Sin ningún tipo de efecto extra.

* Helado:

Te dejará paralizado.

* Veneno:

Te hace 5 de daño durante 2 turnos.

Diagrama de clases de los patrones

*(Imagen del diagrama UML final con todos los patrones comunicados).*

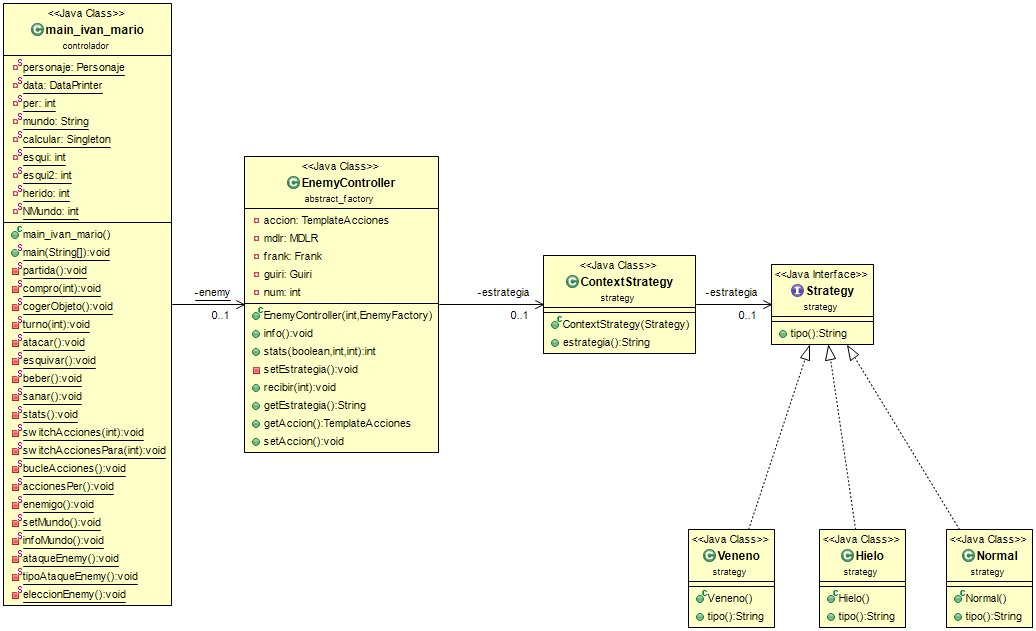
A la hora de crear Yo Interneto se han aplicado varios patrones aprendidos en la asignatura **Diseño de Software** convirtiendo así el código del proyecto en un código más flexible, robusto y reutilizable.

En las paginas posteriores se irán mostrando cada uno de los patrones explicando su motivo de uso, sus respectivos UMLs y el funcionamiento interno del patrón.

Yo Interneto se compone de los siguientes patrones:

* Strategy.
* Decorator.
* State.
* Abstract factory.
* Singleton.
* Template Method.
* **Strategy**

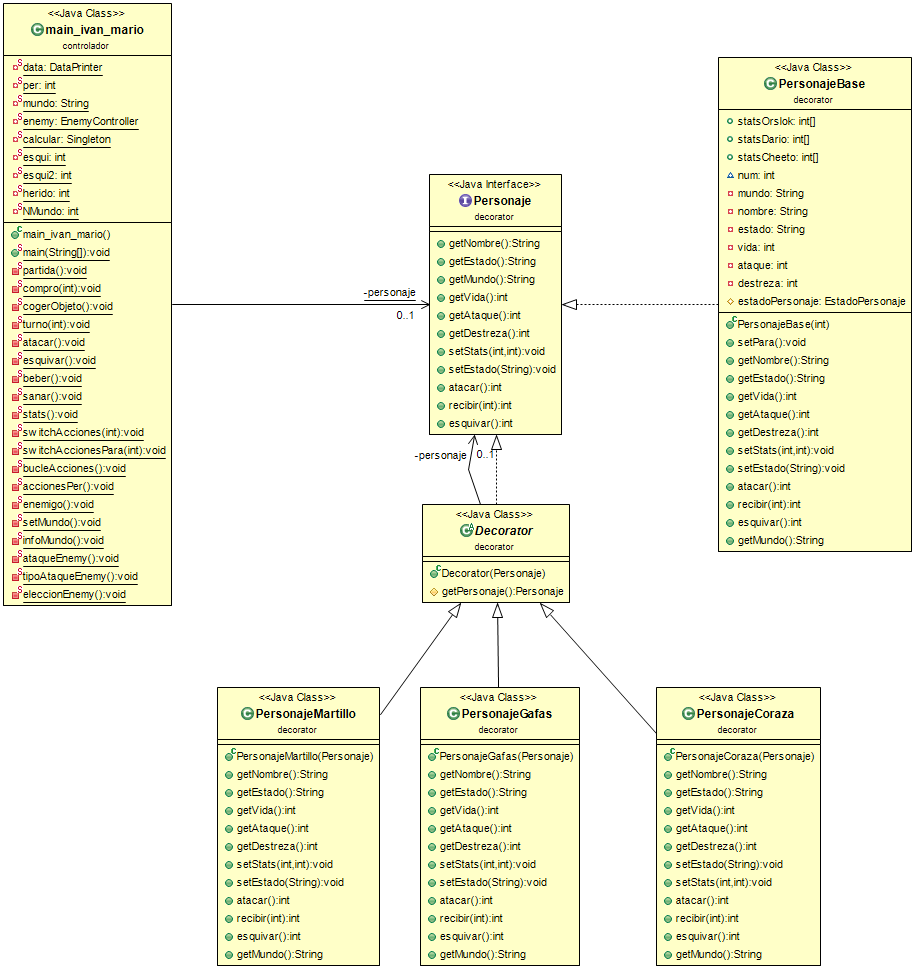
El motivo de uso del patrón **Strategy** es el de abstraer la gestión de las diferentes estrategias posibles que puede adoptar un enemigo en cada uno de los turnos, estas estrategias se determinarán con aleatoriedad en cada uno de los turnos durante toda la partida.



El flujo de ejecución del patrón **Strategy** comienza en **eleccionEnemy()** en el main. Desde este método llamamos a **enemy.setAccion()**, un método de **EnemyController**. Este método se encarga de escoger de forma aleatoria si el enemigo va a atacar o defender. En caso de atacar se llamará al método **setEstrategia()** que, de forma aleatoria, definirá una estrategia entre los 3 tipos que hay como **new ContextEstrategy()**.

* **Decorator**

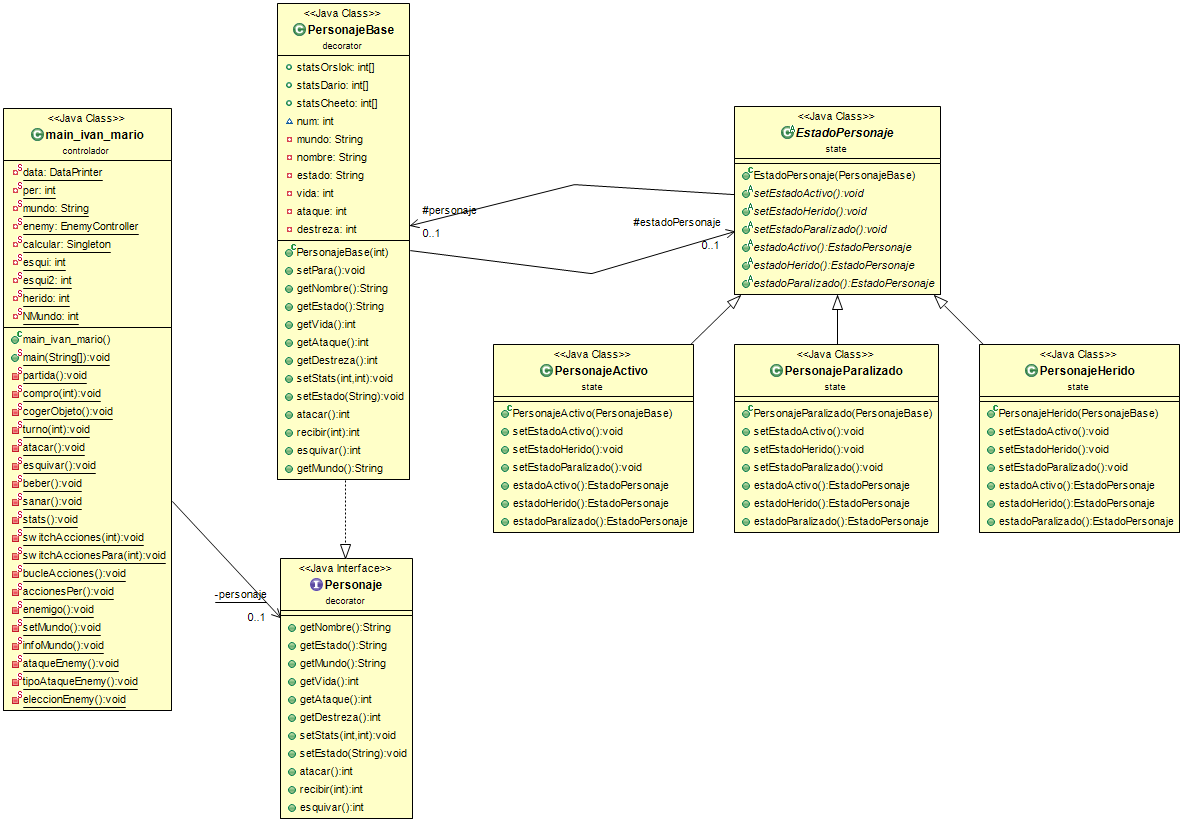
El motivo de uso del patrón **Decorator** es el de abstraer la gestión de uso de los posibles *ítems* que tiene cada personaje, dependiendo del tipo de clase decoradora que tenga, sus atributos (vida, ataque, destreza, estado) serán modificados según las características de cada *ítem*.



El flujo de ejecución del patrón **Decorator** comienza en

* **State**

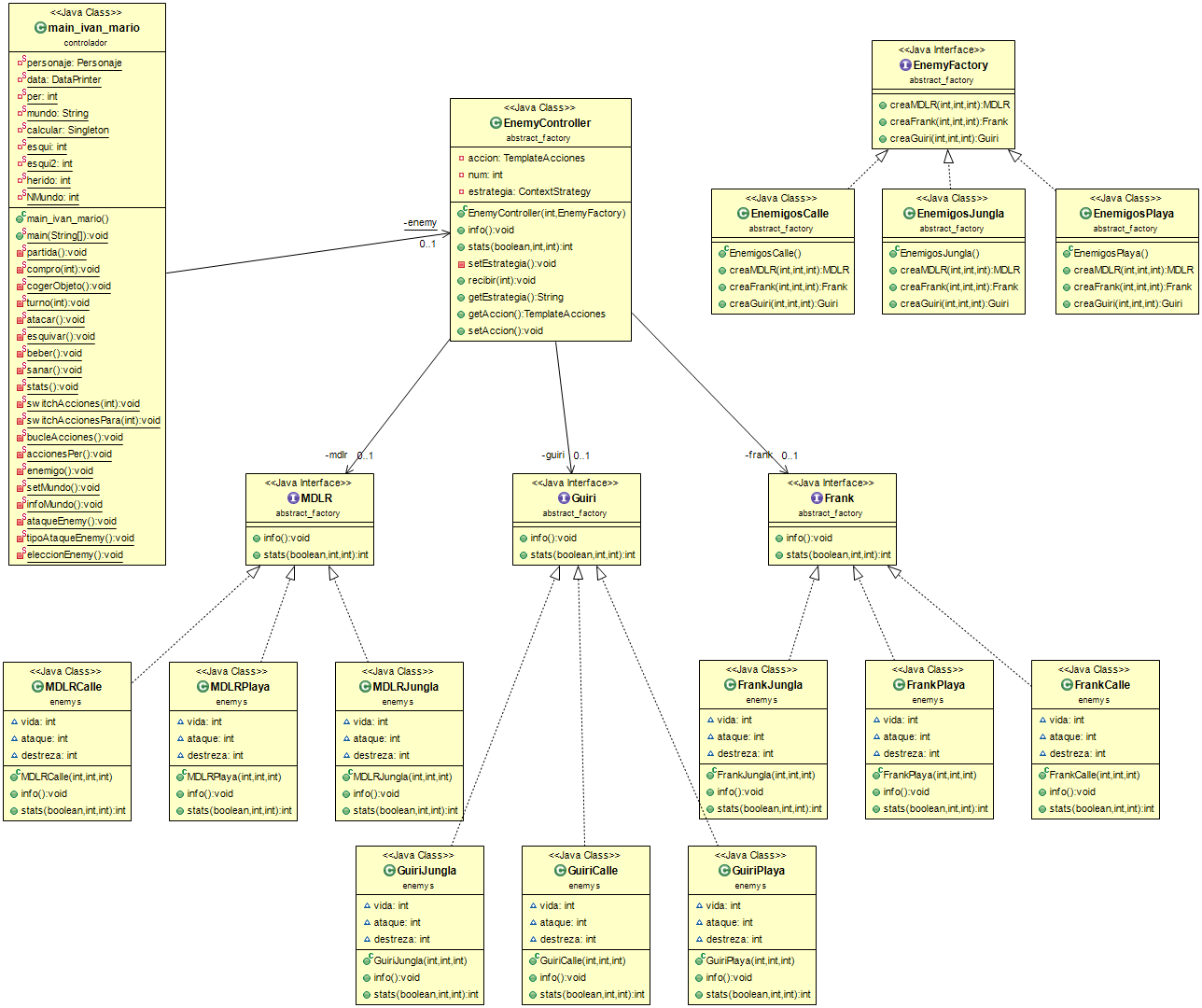
El motivo de uso del patrón **State** es el de abstraer los diferentes estados que puede adoptar nuestro personaje base en cada uno de los turnos según los acontecimientos durante la partida.



El flujo de ejecución del patrón **State** comienza en

* **Abstract Factory**

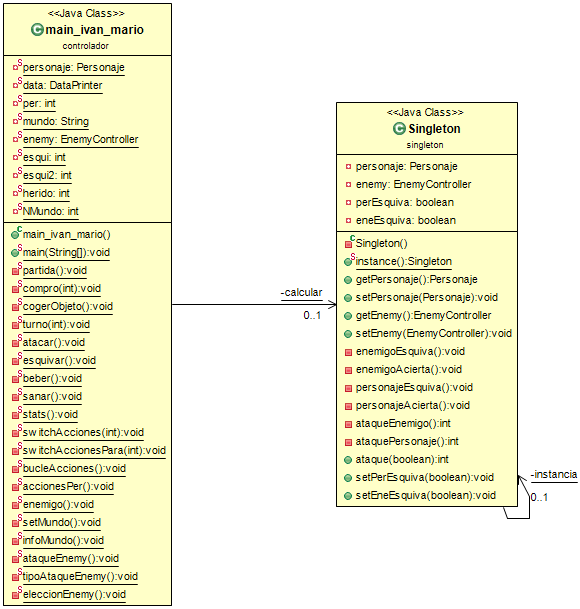
El motivo de uso del patrón **Abstract Factory** es el de abstraer la fabricación de nuevos enemigos durante la partida, mediante plantillas ya existentes los enemigos se instanciarán mediante dichas plantillas y tendrá diferentes ventajas o debilidades según el entorno (mundo) donde se instancien.



El flujo de ejecución del patrón **Abstract Factory** comienza en

* **Singleton**

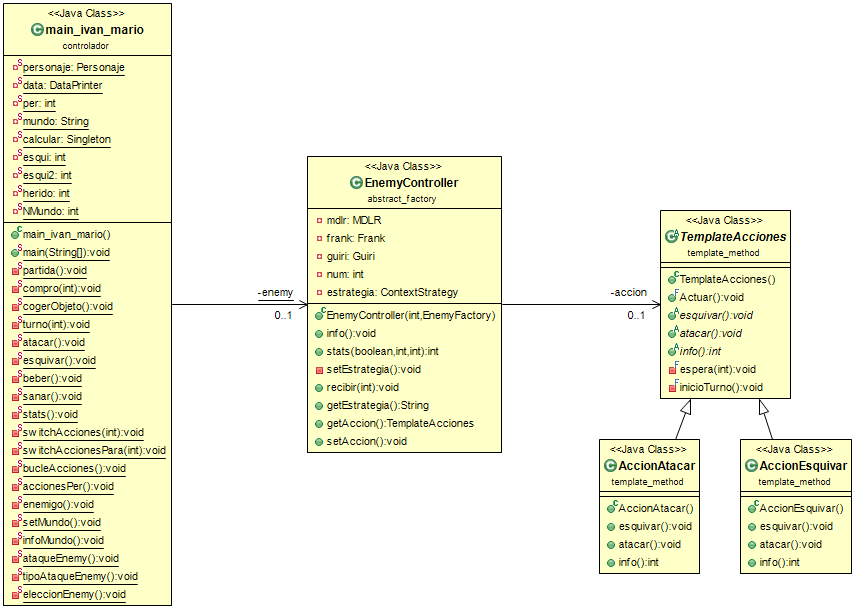
El motivo de uso del patrón **Singleton** es el de abstraer el cálculo de probabilidades de las diferentes acciones que realizan tanto el personaje base como el enemigo en sí.



El flujo de ejecución del patrón **Singleton** comienza en

* **Template Method**

El motivo de uso del patrón **Template Method** es el de abstraer las acciones a realizar del enemigo actual de la partida, en cada uno de los turnos esta decisión será tomada aleatoriamente.



El flujo de ejecución del patrón **Template Method** comienza en