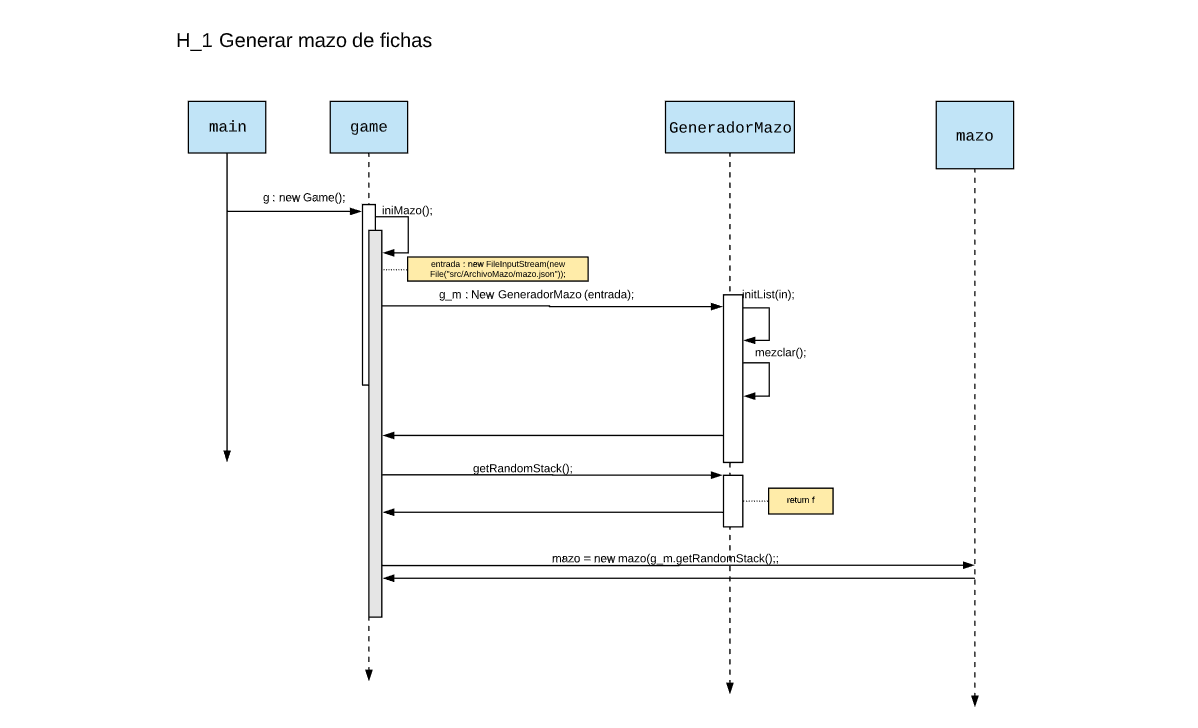
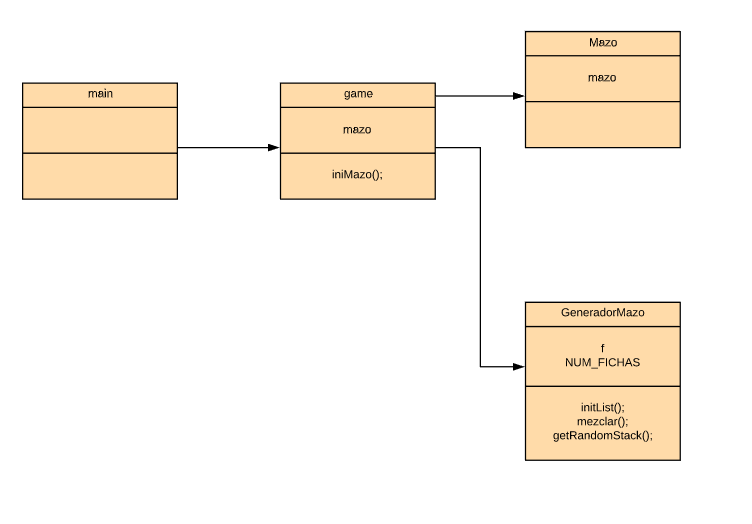
**SPRINT 1 DIAGRAMAS**

1. **Generar mazo de fichas**
   1. **Diagrama de secuencia**



* 1. **Diagrama de clases**



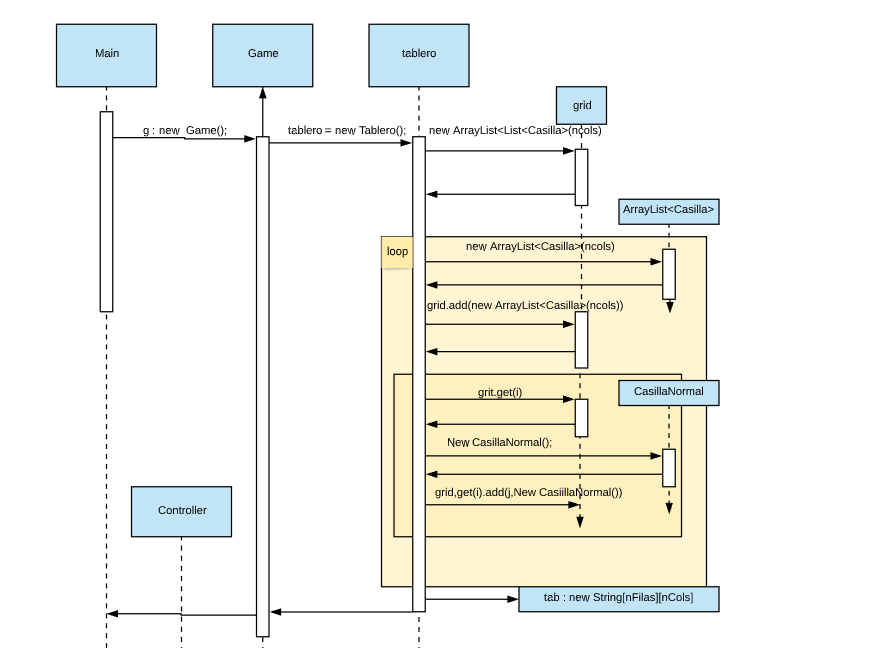
* 1. **Explicación**

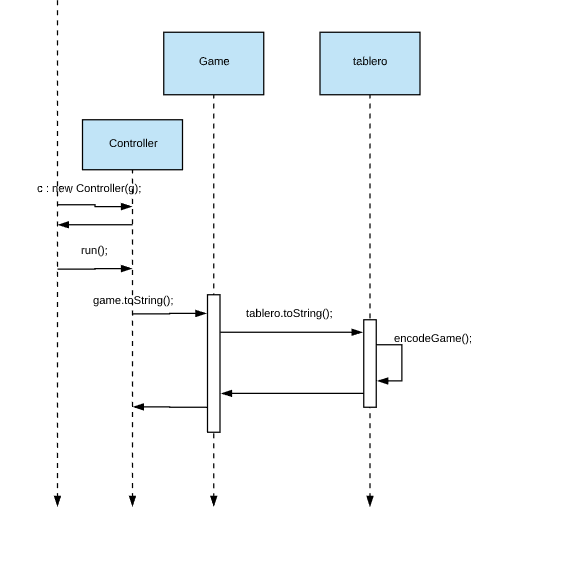
Para generar un mazo aleatorio de fichas, la clase Game llama a un método (iniMazo()) que primero guarda en un FileInputStream el archivo JSON donde tenemos la información sobre las fichas. A continuación, crea una instancia de GeneradorMazo.

Esta clase, es su método initList(in), a partir de un partir de un parámetro de tipo FileInputStream, construye un mazo de Fichas según la información de un archivo JSON, en el que para cada letra del abecedario teniendo en cuenta su frecuencia en el juego, construye una Ficha y la añade a la lista de fichas f.

Para terminar, como queremos que sea un mazo aleatorio, llamamos al método mezclar() que realiza una llamada al método shuffle().

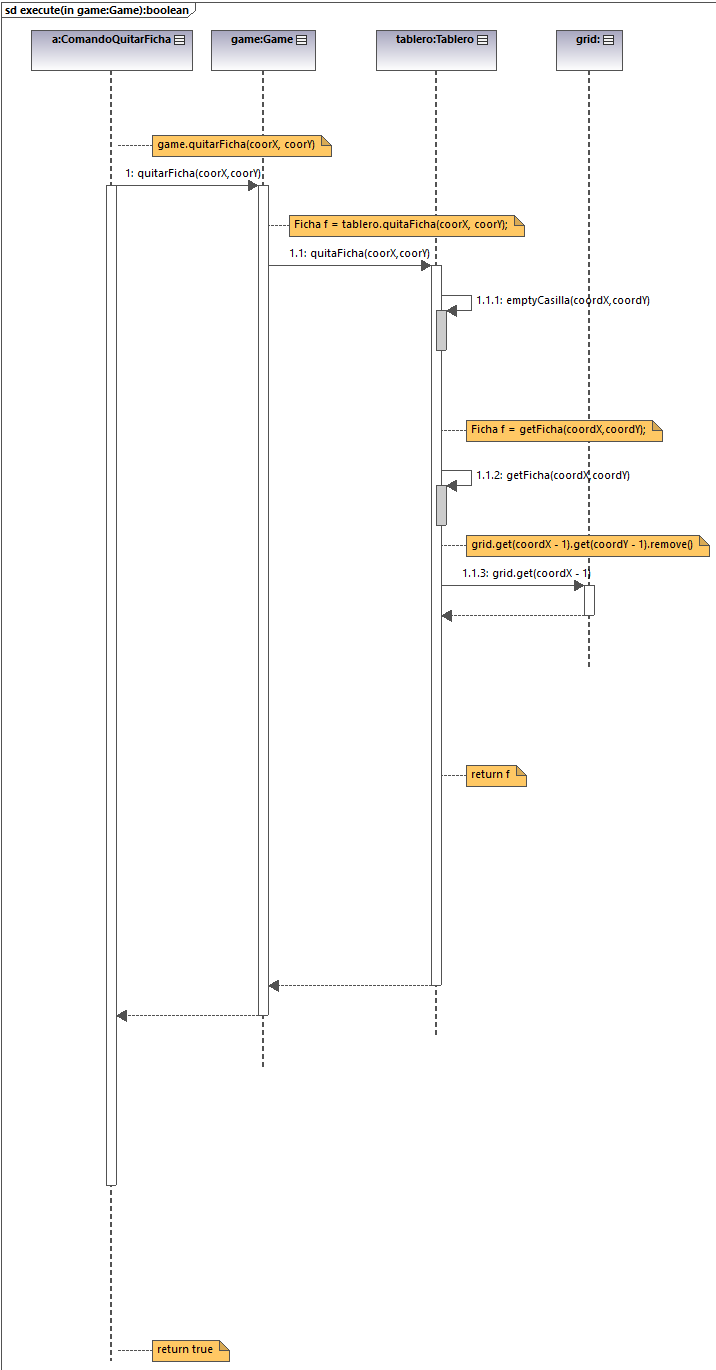
De vuelta al Game, creamos un Mazo al que le pasamos una Colección aleatoria de fichas. De esta forma obtenemos un mazo aleatorio de Fichas.

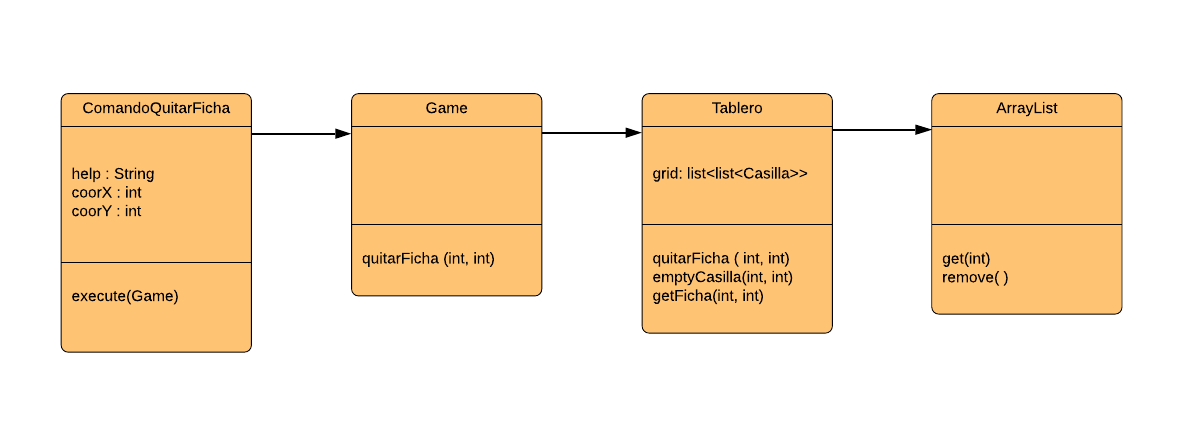
1. **Ver el tablero**
   1. **Diagrama de secuencia**



* 1. **Diagrama de clase**
  2. **Explicación**

1. **Quitar ficha**
   1. **Diagrama de secuencia**



* 1. **Diagrama de clase**
  2. **Explicación**

Para quitar una ficha del tablero, el método execute de la clase ComandoQuitarFicha (que recibe un parámetro de tipo Game) llama al método quitarFicha de la clase Game usando como parámetros los atributos coorX y coorY de la clase ComandoQuitarFicha, posteriormente, el método quitarFicha de la clase Game llama al método quitarFicha de la clase Tablero pasándole dichos parámetros.

El método quitarFicha de la clase Tablero comprueba a través del método emptyCasilla (que recibe como parámetros los parámetros de quitarFicha) si la Casilla seleccionada está vacía, en caso de que no esté vacía lo que hace el método guardar la ficha a extraer en una variable auxiliar de tipo Ficha (para luego poder incorporarla al mazo del jugador) y posteriormente eliminarla de grid usando el método remove (método heredado de la clase ArrayList), en caso de que la Casilla esté vacía el método lanza una excepción.

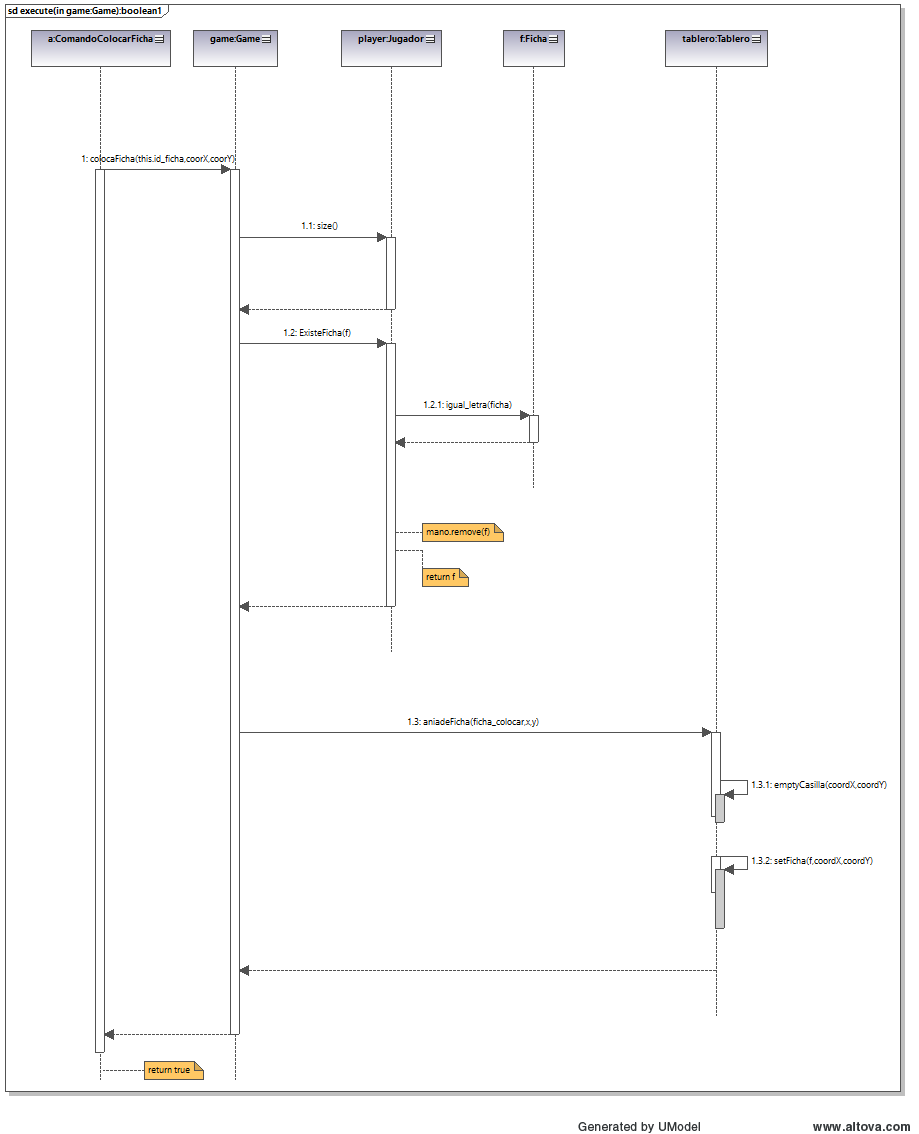
1. **Rellenar mano**
   1. Imagen que contiene captura de pantalla

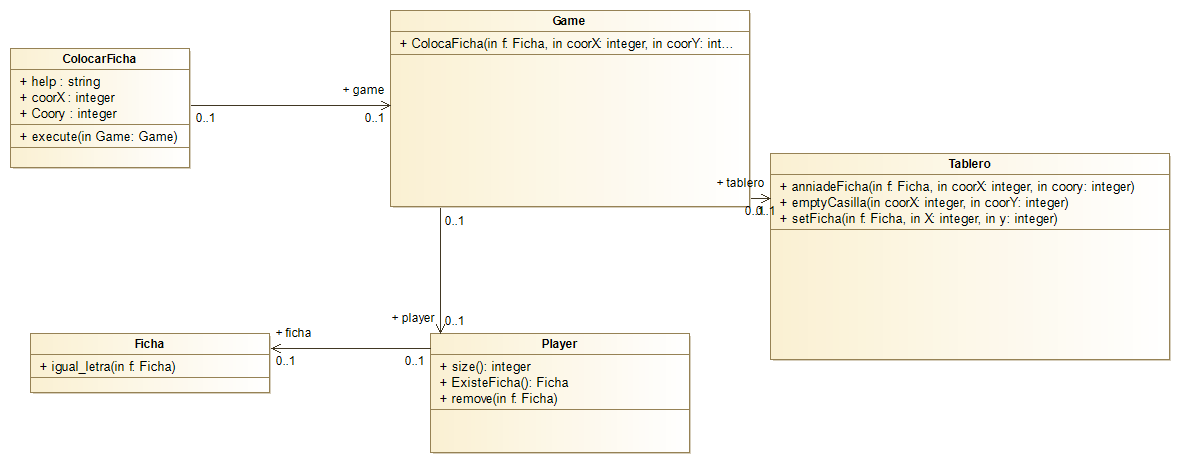
      Descripción generada automáticamente**Diagrama de secuencia**
   2. Imagen que contiene captura de pantalla

      Descripción generada automáticamente**Diagrama de clase**
   3. **Explicación**

Para rellenar la mano de un jugador, la clase Game llama a un método rellenarManos() que consiste en un bucle while-do que se repite mientras no se llene la mano de un jugador, es decir, mientras no se alcance un máximo que está definido por una constante (número máximo de fichas que puede tener cada jugador).

En cada vuelta del bucle se pedirá a la clase Mazo una ficha a través del método de la clase Game robar(). Una vez obtenida la ficha se llamará a otro método robar(Ficha f) de la clase Jugador en el que se añade la ficha f a la mano del jugador, implementada con una lista de fichas.

1. **Colocar ficha**
   1. **Diagrama de secuencia**
   2. **Diagrama de clase**



* 1. **Explicación**

Para colocar una ficha en el tablero se llama a un metodo de la clase que comprueba si la casilla está vacía. Si lo está, quita de la mano del jugador la ficha que se pasa por consola (si se encuentra en la mano), y la coloca en la casilla indicada por coorX coorY.

1. **Saber el número de fichas que quedan por robar**
   1. **Diagrama de secuencia**
   2. **Diagrama de clase**
   3. **Explicación**
2. **Salir de la partida**
   1. **Diagrama de secuencia**
   2. **Diagrama de clase**
   3. **Explicación**