# **Sprint 2 Review**

1. **Saber mi conjunto de fichas actual**

Para saber la mano que tiene el jugador que tiene el turno, se ha introducido en el toString de la clase jugador, que muestra por pantalla la mano que en ese momento tiene dicho jugador.

Ese toString es llamado por la clase Game cada vez que se imprime el tablero. (El atributo Player de la clase Game cambia cada turno).

1. **Saber cuántos puntos vale cada ficha**

Para saber cuántos puntos vale cada ficha, primero se asignan los puntos al mazo.json. Cuando se crean las fichas se guarda el carácter y los puntos correspondientes a la respectiva letra. Solo se altera el método toString de Ficha para poder conocer los puntos, que ahora además de mostrarme la letra, también muestra su puntuación.

1. **Introducir nicks**

Se ha implementado la función de introducir jugadores dentro de la clase Controller, creando una lista de jugadores a la cual vamos añadiendo participantes introduciendo sus nicks.

1. **Conocer turnos de la partida**

Para conocer los turnos de la partida se ha implementado el método

toString en la clase AdminTurnos, este método es llamado en la clase Turno, concretamente en el método Run. El método toString se ha implementado usando un bucle y una lista circular que es recorrida dentro de este, al finalizar el bucle se muestra el orden en el que los jugadores realizarán sus turnos por pantalla.

1. **Pasar de turno**

Se ha creado un comando nuevo para pasar de turno en la clase ComandoPasarTurno que implementa el método parse para verificar el comando y el método execute, que devuelve un booleano para indicar a la clase Turno que el jugador pasará su turno al siguiente.

1. **Colocar más de una ficha por turno**

Aprovechando que la la introducción de comandos se encuentra en la clase Turno, lo que hacemos es darle libertad al usuario para poner el número de fichas que quiera hasta que seleccione el comando pasar turno.

1. **Jugar con otro jugador**

Se ha creado una nueva clase llamada AdminTurnos que se encarga de ir asignando turnos a cada jugador en el orden establecido. Además, se ha creado una clase turno que encapsula la información de lo que sucede en el turno de un jugador y le dice al Game sobre qué jugador tiene que efectuar los cambios En esta clase turno es donde se encuentra el bucle de introducir comandos que antes se encontraba en la clase Controller.

1. **Saber quien empieza la partida**

Se ha creado un método en la clase AdminTurnos que utiliza la función de math que te da un valor aleatorio entre dos números. En este caso te da un valor entre 0 y el número de jugadores menos uno, este valor se corresponde con la posición en la lista de jugadores que ocupa el primer jugador.

1. **Verificar si una palabra es válida**

Se ha creado un comando verificar al que se le pasa la posición de la primera ficha de la palabra y la posición de la última. El execute de este comando comprueba que las posiciones son válidas y obtiene la palabra contenida a partir de las letras de cada una de las fichas entre ambas posiciones. Además, se ha creado un diccionario con una lista de string y un método iniDiccionario que lee las palabras de un documento de texto y las almacena en la lista. De esta manera, en el execute también se comprueba que el diccionario contiene la palabra obtenida. Si la palabra está en el diccionario se confirma la palabra y pasa el turno. Si no, continúa el turno del jugador.

1. **Obtener puntos por mis jugadas**

Se ha modificado el execute del comando verificar de forma que al mismo tiempo que se obtienen las letras de la palabra se van sumando los puntos. Al final, si la palabra está en el diccionario se suman los puntos al jugador.

1. **Obtener bonificaciones en función de las casillas que ocupa la palabra que pongo**

Se han creado las clases correspondientes a las casillas especiales (casillas roja, naranja, morada y azul) que extienden a casilla y la clase casilla normal que también extiende a casilla. Estas casillas tienen un atributo multiplicador que determina qué tipo de bonificación se obtiene con las puntuaciones de cada una (1 = casilla normal | 2 = doble por letra | 3 = triple por letra | 4 = doble por palabra | 5 = triple por palabra).

Se ha modificado la clase Tablero de manera que cuando se inicializa el tablero se crea con la disposición de casillas correspondiente al juego del Scrabble.

Se ha modificado el método execute del comando verificar de manera que al sumar los puntos para las palabras se tenga en cuenta la casilla en la que está cada ficha. Si las casillas son de doble por letra o triple por letra se multiplican los puntos obtenidos por la ficha. Si son de doble por palabra o triple por palabra se acumula el multiplicador en una variable y luego se multiplican los puntos totales al final.

1. **Tener en la pila de fichas dos comodines**

Se ha añadido en el archivo mazo.json, un nuevo elemento, el comodín. Este tiene como letra el carácter ‘\*’ y como puntos 0, según las normas oficiales del Scrabble.

1. **Canjear comodines de la mano por letras**

Para canjear los comodines se jugador solo tiene que colocar la ficha como cualquier otra. Se han realizado unos cambios en el método colocarFicha. Se han añadido, que cuando un jugador quiere colocar un comodín (carácter ‘\*’), se le solicita la letra y se crea una nueva ficha que es la que se añade al tablero.

1. **Guardar partida**

Para guardar el estado de la partida se ha creado un método save en la clase AdminTurnos el cual es llamado desde la clase Controller una vez finaliza el bucle del juego. El método save mediante un bucle recorre la lista de jugadores guardando sus nicks, puntuaciones y monedas en el fichero que previamente el usuario elija.

1. **Quitar una ficha del tablero**

Para quitar una ficha del tablero, se ha incluido en el game una lista de pares que indican las coordenadas de las fichas puestas por un jugador durante su turno. Cuando se ejecuta el comando quitar ficha sobre una de las casillas del tablero, se comprueba, mirando en esa lista que la ficha ha sido colocada durante ese turno y si es así, se devuelve a la mano del jugador.

Esta lista de coordenadas se vacía al acabar cada turno.

1. **Eliminar Jugadores:**

Se ha implementado la función de eliminar jugadores dentro de la clase Controller. Esta función muestra una lista numerada de los jugadores, el usuario que quiera eliminar a un jugador solo debe introducir la posición en la que se encuentra dicho jugador pudiendo repetir esta operación las veces que vea necesario.

1. **Mostrar información de los jugadores en cada turno:**

Este método está implementado dentro de la clase AdminTurnos, concretamente dentro de siguiente turno, esta función muestra los nicks de los jugadores de la partida y su puntuación en cada momento.