Análisis y Diseño

Para la construcción de este simulador es necesario una clase IPod que contenga todas las características de bloqueo/desbloqueo, encendido/apagado, control de volumen y canciones (esta última podría ser otra clase para que contenga los atributos: artista, nombre, duración, etc.), la lista de canciones que será en una lista estática de java dado que tienen un número establecido de espacios. Sin embargo, para que se tenga contabilidad con las interfaces de otros, es necesario que esta clase implemente una interfaz que considere todos los métodos necesarios del problema.

Los métodos que debe tener la clase IPod y que tendrán la interfaz son (idea inicial, posteriormente se cambió por una interfaz para toda la clase):

- Public void volumeUp() //Altera el atributo del IPod para aumentar el volumen
- Public void volumeDown()//Altera el atributo del IPod para aumentar el volumen
- Public Song NextSong()//De la lista de canciones, altera el índice actual que apunta a la canción actual y obtiene la siguiente canción, el mismo método devuelve esta canción.
- Public Song PreviousSong() // De la lista de canciones, altera el índice actual que apunta a la canción actual y obtiene la canción anterior, el mismo método devuelve esta canción.
- Public void saveAsFavourite(favourite_position int)//Guarda la cancion escuchada actualmente (este indice es un atributo de la clase), y la pone en una segunda lista de favorita según la posición establecida (sobreescribe datos de ser necesario).
- Public Song goToFavourite(fav_pos int) //Identifica la cancion en la lista de favoritas y establece el índice de la misma (pero en la lista global) como la canción que se escucha actualmente
- Public String watchState() //Indica el estado del IPod
- Public void OnOff()//Cambia al estado opuesto encendido al actual
- Public void LockUnlock()//Cambia al estado opuesto de bloqueo al actual
- Public boolean isLocked()//Devuelve bool indicando si está bloqueado
- Public boolean isOn()//Devuelve bool indicando si esta encendido
- Public Song getCurrentSong()//Devuelve la cancion actual

El sistema para la interacción se hará a través de la consola (sin GUI) y el estado del IPod se mostrará a través de System.out.println().

En todo caso, la línea que cambia al momento de intercambiar interfaces es la instanciación del objeto IPod.

Ejemplo:

Dado que ambos deben surgir de una misma interfaz la declaración del objeto se puede hacer sobre esta interfaz como su tipo

Instanciación de clase propia->

InterfazIPod ipod1 = new IPod(); //Donde IPod es la propia clase creada implementando la interfaz.

Instanciación de otra clase->

InterfazIPod ipod1 = new IPod2(); //Donde IPod2 es otra clase que implementa la interfaz.

La clase de interacción deberá seguir funcionando porque ambas clases debieron haber implementado los métodos descritos en la interfaz (después de todo ese es el punto de las interfaces de Java).