Diseño de los casos de prueba

Daniel Ferreiro Presedo
Alejandro Sánchez Sánchez

20 de octubre de 2016

Índice

1. Observaciones

Debido a la falta de tiempo, hemos realizado evitar realizar las siguientes comprobaciones, teniendo en cuenta las otras, puesto que las consideramos más importantes al ser más críticas o al ser más usadas:

- Las relacionadas con la clase heredada *GenericDao*, suponiendo que el resultado no se verá modificado respecto al obtenido en la sección ApuestaDao.
- En la sección OpcionDao, el método *showWinners* cuando todas las Opciones son ganadoras porque suponemos que ya las filtra bien en las otras pruebas del método.
- En la sección TipoApuestaDao , el método *findDuplicateBetTypes* para cuando no hay ningún TipoApuesta duplicado, puesto que es una simple comprobación.
- En la sección CategoriaDao, el método *findCategories* cuando no hay ninguna, puesto que se supone que devolverá una lista vacía.
- En la sección EventoDao :

  El método *findEventNoDate*, puesto que realiza lo mismo que el método *findEvent*, por lo que se podría refactorizar y solo probar un método.

  El método *findDuplicateEvents* cuando no hay ningún Evento con el mismo nombre programado para la misma fecha de la misma Categoría, puesto que es una simple comprobación.

  El método *getNumberOfEvents* y *getNumberOfEvents2*, puesto que se podrían refactorizar en el método *findEvent*.

  El método *findAllEvents* y *findAllEventsNoDate*, puesto que se podrían refactorizar del método *findEvent*, ya que realizan lo mismo pero sin ningún filtro.

- En la sección AdminService :

  El método *createEvent* cuando se inserta con una fecha anterior a la actual, puesto que es una simple comprobación.

El método *createEvent* cuando se intenta insertar un evento con una categoría inexistente, puesto que se supone que se soltará correctamente la excepción al ser una simple comprobación.

El método *findEvent* cuando hay menos eventos de los que se piden y cuando no hay palabras clave, puesto que ya se han realizado estas pruebas con el *findEventNoDate* del EventoDao, que es el método al que llama.

El método *findCategoriaById*, puesto que es una simple llamada al método *find* del CategoriaDao, el cual ya se ha probado anteriormente.

El método *markAsWinner* cuando se quiere establecer varias Opciones ganadoras cuando el TipoApuesta solo permite una, cuando el Evento ya ha expirado y cuando alguna de las Opciones no pertenece a ese TipoApuesta, puesto que son simples comprobaciones.

El método *addBettingType* cuando ya existe en base de datos el TipoApuesta puesto que es una llamada al método *findDuplicateBetType* del TipoApuestaDao, el cual ya está probado; así como cuando no existe el Evento al que asociar o cuando el Evento ya ha expirado, ya que son comprobaciones simples.

El método *getNumberOfEvents*, puesto que es una llamada al método *getNumberOfEvents2* del EventoDao, el cual ya está probado con anterioridad.

El método *findTipoApuesta*, puesto que es una simple llamada al método *find* del TipoApuestaDao, el cual ya está probado con anterioridad.

El método *findOptionById*, puesto que es una simple llamada al método *find* del OpcionDao, el cual ya está probado con anterioridad.

El método *findAllEvents*, puesto que es una simple llamada al método *findAllEventsNoDate* del EventoDao, el cual ya está probado anteriormente.

- En la sección BetService :

  El método *findOptionById*, puesto que es una llamada al método *find* del OpcionDao, el cual ya está probado con anterioridad. Tampoco se probará el caso de que el evento ya haya expirado, puesto que es una simple comprobación.

El método *findEvent* cuando hay menos eventos de los que se piden y cuando no hay palabras clave, puesto que ya se han realizado estas pruebas con el *findEvent* del EventoDao, que es el método al que llama.

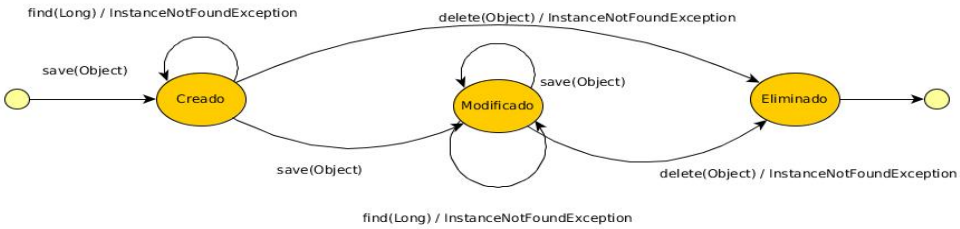
El método *bet* cuando se intenta apostar sobre un Evento ya pasado , puesto que es una simple comprobación. Tampoco se probará cuando se intenta apostar sobre un Evento inexistente, puesto que es una llamada al *find* del EventoDao, el cual ya está probado con anterioridad; así como cuando es un usuario inexistente, puesto que es una llamada al método *find* de la clase otorgada *UserService*.

El método *checkBet* cuando hay menos Apuestas de las pedidas, puesto que es una llamada al método *findApuestasByIdUsuario* del ApuestaDao.

Los métodos *findEventById, getNumberOfEvents,getNumberOfBets, showWinners y findAllEvents*, puesto que son simples llamadas a los Daos correspondientes, los cuales han sido probados con anterioridad.

2. Diagramas

2.1. GenericDao



3. ApuestaDao

3.1. PR-UN-AD-01

Unidad : ApuestaDao

Método : save

Motivación : Creación de una apuesta en la base de datos no previamente creado

Entrada : Apuesta

Salida : Poder continuar con el programa

Inicialización : El usuario que realizará la apuesta, un Evento, un TipoApuesta asociado a ese Evento y una Opcion asociada a ese TipoApuesta sobre el que realizar la apuesta.

3.2. PR-UN-AD-02

Unidad : ApuestaDao

Método : save

Motivación : Modificación de una apuesta en la base de datos previamente creado

Entrada : Apuesta

Salida : Poder continuar con el programa

Inicialización : El usuario que realizará la apuesta, un Evento, un TipoApuesta asociado a ese Evento y una Opcion asociada a ese TipoApuesta sobre el que realizar la apuesta, y la apuesta

3.3. PR-UN-AD-03

Unidad : ApuestaDao

Método : find

Motivación : Recuperar de la base de datos una apuesta

Entrada : Long con el identificador de la apuesta

Salida : La apuesta previamente creada

Inicialización : El usuario que realizará la apuesta, un Evento, un TipoApuesta asociado a ese Evento y una Opcion asociada a ese TipoApuesta sobre el que realizar la apuesta, y la apuesta

3.4. PR-UN-AD-04

Unidad : ApuestaDao

Método : find

Motivación : Recuperar de la base de datos una apuesta no existente

Entrada : Long con el identificador de la prueba

Salida : La excepción *InstanceNotFoundException*

Inicialización : -

3.5. PR-UN-AD-05

Unidad : ApuestaDao

Método : save / find

Motivación : Actualizar los datos de una apuesta existente y después comprobar que al recuperarlos sean iguales

Entrada : Apuesta a actualizar / Long con el identificador de la apuesta actualizada

Salida : La apuesta con los datos actualizados

Inicialización : Una apuesta

3.6. PR-UN-AD-06

Unidad : ApuestaDao

Método : remove

Motivación : Eliminar una apuesta de la base de datos

Entrada : Long con el identificador de la apuesta

Salida : - Poder continuar con el programa

Inicialización : El usuario que realizará la apuesta, un Evento, un TipoApuesta asociado a ese Evento y una Opcion asociada a ese TipoApuesta sobre el que realizar la apuesta, y la apuesta

3.7. PR-UN-AD-07

Unidad : ApuestaDao

Método : remove

Motivación : Eliminar una apuesta inexistente de la base de datos

Entrada : Long con el identificador de la apuesta

Salida : La excepción *InstanceNotFoundException*

Inicialización : -

3.8. PR-UN-AD-08

Unidad : ApuestaDao

Método : save / find / remove / find

Motivación : Comprobar que se crea y se elimina una apuesta

Entrada : Apuesta / Long con el identificador / Long con el identificador / Long con el identificador

Salida : Poder continuar con el programa

Inicialización : -

3.9. PR-UN-AD-09

Unidad : ApuestaDao

Método : findApuestasByIdUsuario

Motivación : Comprobar que se recupera correctamente las apuestas de un usuario con la cantidad esperada y empezando en el índice esperado

Entrada : Long con el identificador del usuario / Índice inicial / Cantidad de apuestas

Salida : Las X primeras apuestas del usuario ordenadas por fecha

Inicialización : Varios usuarios que realizarán apuestas, varios Eventos, varios TipoApuestas asociado a eses Eventos ,varias Opciones asociada a eses TipoApuesta sobre el que realizar la apuesta, y varias apuestas.

3.10. PR-UN-AD-10

Unidad : ApuestaDao

Método : findApuestasByIdUsuario

Motivación : Comprobar que se recupera correctamente las apuestas de un usuario con la cantidad esperada y empezando en el índice esperado, cuando hay menos apuestas de las que se piden

Entrada : Long con el identificador del usuario / Índice inicial / Cantidad de apuestas

Salida : Las primeras apuestas del usuario ordenadas por fecha

Inicialización : Varios usuarios que realizarán apuestas, varios Eventos, varios TipoApuestas asociado a eses Eventos ,varias Opciones asociada a eses TipoApuesta sobre el que realizar la apuesta, y varias apuestas.

3.11. PR-UN-AD-11

Unidad : ApuestaDao

Método : findNumberOfBets

Motivación : Comprobar que se recuperan correctamente la cantidad de apuestas que un usuario tiene realizada

Entrada : Long con el identificador del Usuario

Salida : El número de apuestas que tiene el usuario

Inicialización : Usuario / Apuestas asociadas al usuario

3.12. PR-UN-AD-12

Unidad : ApuestaDao

Método : findNumberOfBets

Motivación : Comprobar que se recuperan correctamente la cantidad de apuestas que un usuario tiene realizada cuando no tiene ninguna hecha

Entrada : Long con el identificador del Usuario

Salida : 0

Inicialización : Usuario

4. OpcionDao

4.1. PR-UN-OD-01

Unidad : OpcionDao

Método : showWinners

Motivación : Comprobación de que no devuelva ninguna opción ganadora cuando no exista ninguna

Entrada : Long con el identificador del TipoApuesta

Salida : Una lista vacía

Inicialización : TipoApuesta / Opciones asociadas a ese TipoApuesta, pero ninguna marcada como ganadora

4.2. PR-UN-OD-02

Unidad : OpcionDao

Método : showWinners

Motivación : Comprobación de que devuelva correctamente las opciones ganadoras de un tipo de apuesta cuando existen algunas marcadas como ganadoras

Entrada : Long con el identificador del TipoApuesta

Salida : La lista con opciones ganadoras

Inicialización : TipoApuesta / Opciones asociadas a ese TipoApuesta, unas marcadas como ganadoras.

5. TipoApuestaDao

5.1. PR-UN-TAD-01

Unidad : TipoApuestaDao

Método : findDuplicateBetTypes

Motivación : Comprobar que devuelve true cuando hay un TIpoApuesta con la misma pregunta para el mismo Evento

Entrada : Pregunta del TipoApuesta / Long con el identificador del Evento

Salida : true

Inicialización : Evento / TipoApuesta con la pregunta

6. CategoriaDao

6.1. PR-UN-CD-01

Unidad : CategoriaDao

Método : findCategories

Motivación : Obtener de la base de datos todas las categorías existentes para los eventos deportivos

Entrada : -

Salida : Todas las categorías encontradas

Inicialización : Categorias

7. EventoDao

7.1. PR-UN-ED-01

Unidad : EventoDao

Método : findEvent

Motivación : Comprobar que encuentra la cantidad de eventos de una Categoría filtrados por palabras clave y posteriores a la fecha actual, empezando por un índice.

Entrada : Long con el identificador de la Categoria / Palabras clave / Índice inicial / Cantidad de eventos a recuperar

Salida : Los X primeros eventos ordenados por fecha

Inicialización : Eventos con fecha anterior y fecha posterior a la actual

7.2. PR-UN-ED-02

Unidad : EventoDao

Método : findEvent

Motivación : Comprobar que encuentra la cantidad de eventos de una Categoria filtradas por palabras clave y posteriores a la fecha actual, empezando por un índice cuando hay menos soluciones que la cantidad pedida

Entrada : Long con el identificador de la Categoria / Palabras clave / Índice inicial / Cantidad de eventos a recuperar

Salida : Los eventos que coincidan

Inicialización : Eventos con fecha anterior y fecha posterior a la actual

7.3. PR-UN-ED-03

Unidad : EventoDao

Método : findEvent

Motivación : Comprobar que encuentra la cantidad de eventos de una Categoría filtradas por palabras clave y posteriores a la fecha actual, empezando por un índice cuando no hay ningún Evento que coincida con las palabras clave.

Entrada : Long con el identificador de la Categoria / Palabras clave que no coincida que ningún evento / Índice inicial / Cantidad de eventos a recuperar

Salida : Una lista vacía

Inicialización : Eventos con fecha anterior y posterior a la actual

7.4. PR-UN-ED-04

Unidad : EventoDao

Método : findEvent

Motivación : Comprobar que busca la cantidad de eventos de una Categoría filtradas por palabras clave y posteriores a la fecha actual, empezando por un índice cuando no hay ninguna palabra clave

Entrada : Long con el identificador de la Categoria / Palabras clave vacías / Índice inicial / Cantidad de eventos a recuperar

Salida : Una lista con los X eventos de la misma categoría

Inicialización : Eventos con fecha anterior y posterior a la actual

7.5. PR-UN-ED-05

Unidad : EventoDao

Método : findEventNoDate

Motivación : Comprobar que devuelve la cantidad de eventos de una Categoría filtradas solamente palabras clave, empezando por un índice.

Entrada : Long con el identificador de la Categoria / Palabras clave que no coincida que ningún evento / Índice inicial / Cantidad de eventos a recuperar

Salida : Los eventos que contengan las palabras clave

Inicialización : Eventos con fecha anterior y posterior a la actual

7.6. PR-UN-ED-06

Unidad : EventoDao

Método : findDuplicateEvents

Motivación : Comprobar si devuelve true cuando hay un Evento de la misma Categoría programado para la misma fecha con el mismo nombre

Entrada : Nombre del evento / Fecha programada / Long con el identificador de la Categoría

Salida : true

Inicialización : Categoría / Evento con la misma fecha y mismo nombre

8. AdminService

8.1. PR-UN-AS-01

Unidad : AdminService

Método : createEvent

Motivación : Comprobar que crea correctamente el Evento en la base de datos

Entrada : Evento a crear / Long con el identificador de la Categoría

Salida : Evento creado

Inicialización : Categorías

8.2. PR-UN-AS-02

Unidad : AdminService

Método : createEvent

Motivación : Comprobación de que no inserta en la base de datos el Evento si ya existe previamente

Entrada : Evento creado previamente / Long con el identificador de la Categoría

Salida : La excepción *DuplicateInstanceException*

Inicialización : Evento

8.3. PR-UN-AS-03

Unidad : AdminService

Método : findEvent

Motivación : Comprobar que se encuentra los eventos en la base de datos que coincidan con los parámetros de búsqueda, indicando que hay más eventos a mostrar

Entrada : Long con el identificador de la Categoría / Palabras clave / Índice inicial / Cantidad de objetos a recuperar

Salida : EventoBlock con los Eventos coincidentes e indicando que hay más Eventos.

Inicialización : Varios eventos perteneciente a la misma Categoría

8.4. PR-UN-AS-04

Unidad : AdminService

Método : markAsWinner

Motivación : Comprobar que marca correctamente las Opciones indicadas como ganadoras.

Entrada : Una lista con los identificadores de las Opciones / El identificador del TipoApuesta,

Salida : Poder continuar con el programa

Inicialización : Evento / TipoApuestas del Evento / Opciones del TipoApuesta

8.5. PR-UN-AS-05

Unidad : AdminService

Método : markAsWinner

Motivación : Comprobar que no marca las Opciones como ganadoras cuando ya estaban marcadas con un estado.

Entrada : Una lista con los identificadores de las Opciones / El identificador del TipoApuesta

Salida : La excepción AlreadySetOptionException

Inicialización : Evento / TipoApuestas del Evento / Opciones del TipoApuesta

8.6. PR-UN-AS-06

Unidad : AdminService

Método : addBettingType

Motivación : Comprobar que inserta correctamente el TipoApuesta

Entrada : TipoApuesta a introducir en la base de datos / Identificador del evento al que asociar el TipoApuesta / Lista de Opciones a asociar al TipoApuesta

Salida : TipoApuesta creado

Inicialización : Evento a que asociar el TipoApuesta / Opciones a asociar al TipoApuesta

9. BetService

9.1. PR-UN-BS-01

Unidad : BetService

Método : findEvent

Motivación : Comprobar que se busca correctamente los Eventos según los parámetros indicados, indicando además que hay más Eventos coincidentes.

Entrada : Identificador de la Categoría / Palabras clave / Índice inicial / Cantidad de Eventos a recuperar

Salida : EventoBlock con los Eventos encontrados y un true en el indicador de que existen más Eventos coincidentes.

Inicialización : Categoría / Eventos asociados a esa Categoría

9.2. PR-UN-BS-02

Unidad : BetService

Método : bet

Motivación : Comprobar que se realice correctamente una Apuesta sobre una Opción para el Usuario correspondiente.

Entrada : Identificador de la Opción / La cantidad a apostar por esa Opción / Identificador del Usuario

Salida : La Apuesta creada

Inicialización : Usuario / Evento / TipoApuesta asociada a ese evento / Opción asociada a ese TipoApuesta

9.3. PR-UN-BS-03

Unidad : BetService

Método : checkBet

Motivación : Comprobación de que se recuperan la cantidad de Apuestas indicadas de un Usuario indicando por un índice inicial, devolviendo un true para indicar que hay más.

Entrada : Identificador del Usuario / Índice inicial / Cantidad de Apuestas a recuperar

Salida : Las Apuestas del Usuario ordenadas por fecha, y un true en el indicador de que hay más Apuestas.

Inicialización : Usuario / Apuestas realizadas por ese Usuario