Plan de pruebas – US434996-OrdenarPorHoraComienzo (Juan Vélez Velasco)

Los niveles de prueba que se van a aplicar son los siguientes:

- Pruebas de aceptación: las pruebas de aceptación se definirán siguiendo una estrategia basada en historias de usuario y serán ejecutadas de forma manual.
- Pruebas de integración: Estas pruebas verifican la interacción entre clases. Se llevarán a cabo utilizando los framework *Junit, Robolectric y Mockito.*
- Pruebas de interfaz: se pruebas todos los elementos desde la interfaz, comprobando que la funcionalidad es correcta y que las salidas son las esperadas respecto a las entradas que se han introducido. Para ello, se ha usado el framework Espresso, que nos permite automatizar las pruebas.

Nota: Todos los niveles de prueba de ejecutarán sobre la fuente de datos disponible en: https://personales.unican.es/rivasjm/resources/agenda_cultural.json

PRUEBAS DE ACEPTACIÓN

Se muestran las pruebas en base a los criterios de aceptación acordados con el *Product Owner*.

Prueba 00: Éxito

1. El usuario selecciona pulsa el botón de ordenar.
2. El sistema crea una ventana emergente con los tipos de ordenación.
3. El usuario selecciona ordenar por hora de comienzo más cerca.
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente.
6. El sistema muestra la lista de eventos ordenados de manera que los que estén más cerca a la fecha actual aparezcan antes.

NOTA: Los eventos sin fecha se mostrarán los últimos de la lista, los eventos con fecha pero sin hora se mostrarán los primeros en ese día.

Prueba 01: Cancelar

1. El usuario selecciona pulsa el botón de ordenar.
2. El sistema crea una ventana emergente con los tipos de ordenación.
3. El usuario selecciona ordenar por hora de comienzo más cerca o más lejos.
4. El usuario selecciona cancelar.
5. Se cierra la ventana emergente.
6. El sistema muestra la lista de eventos previa a la selección del botón ordenar.

PRUEBAS DE INTEGRACIÓN

Se aplica prueba de métodos, siendo los casos de prueba definidos para cada método los siguientes:

Método **onOrdenarClicked(int tipoOrdenación)** de la clase **EventsPresenter.java**

Casos de prueba válidos:

Identificador	**Entrada**	**Resultado**
IT. 1A	2 -> se corresponde a la ordenación "más próximo primero". Lista *eventosEnFiltrosCombinados* **con eventos**.	Lista *eventosEnFiltrosCombinados* ordenada de manera que los eventos más próximos a la fecha actual aparezcan primero.
IT. 1B	3 -> se corresponde a la ordenación "más lejano primero". Lista *eventosEnFiltrosCombinados* **con eventos**.	Lista *eventosEnFiltrosCombinados* ordenada de manera que los eventos más lejanos a la fecha actual aparezcan primero.
IT. 1C	2 -> se corresponde a la ordenación "más próximo primero". Lista *eventosEnFiltrosCombinados* **vacía**.	Lista *eventosEnFiltrosCombinados* ordenada de manera que los eventos más próximos a la fecha actual aparezcan primero.
IT. 1D	3 -> se corresponde a la ordenación "más lejano primero". Lista *eventosEnFiltrosCombinados* **vacía**.	Lista *eventosEnFiltrosCombinados* ordenada de manera que los eventos más lejanos a la fecha actual aparezcan primero.

PRUEBAS DE INTERFAZ DE USUARIO

A. Ordenación por fecha más cercana.
   1. Se abrirá la interfaz pulsando el botón de *ordenar*.
   2. El usuario selecciona el botón de ordenar por fecha más cercana.
   3. El usuario pulsará el botón "Aplicar" de la ventana flotante.
   4. Se comprobará que la lista de eventos que aparece está ordenada mostrándose los eventos mas cercanos a la fecha actual antes.

B. Ordenación por fecha más lejana.
   1. Se abrirá la interfaz pulsando el botón de *ordenar*.
   2. El usuario selecciona el botón de ordenar por fecha más lejana.
   3. El usuario pulsará el botón "Aplicar" de la ventana flotante.
   4. Se comprobará que la lista de eventos que aparece está ordenada mostrándose los eventos más lejanos a la fecha actual antes.

C. Pulsar el botón cancelar con un modo de ordenación por fecha seleccionado.
   1. Se abrirá la interfaz pulsando el botón de *ordenar*.
   2. El usuario selecciona el botón de ordenar por fecha más cercana.
   3. El usuario pulsará el botón "Cancelar" de la ventana flotante.
   4. Se comprobará que la lista de eventos que aparece es igual que antes de pulsar el botón de ordenar.

D. Pulsar el botón cancelar sin haber marcado ningún tipo de ordenación.
   1. Se abrirá la interfaz pulsando el botón de *ordenar*.
   2. El usuario pulsará el botón "Cancelar" de la ventana flotante.
   3. Se comprobará que la lista de eventos que aparece es igual que antes de pulsar el botón de ordenar.

En este informe se describe el resultado de ejecutar las pruebas para la historia de usuario FiltrarPorFecha, así como los problemas encontrados durante la realización de las pruebas y las soluciones aportadas.

Autor del plan de pruebas: Adrián García Cubas

Implementador de las pruebas: Juan Vélez

PRUEBA DE INTERFAZ DE USUARIO

La prueba de interfaz de usuario se ha realizado utilizando los frameworks JUnit y Espresso, se han implementado las pruebas IUT.A, IUT.B, IUT.C., IUT.D, IUT.E., IUT.F. Durante la implementación de dichas pruebas el ejecutor tuvo problemas a la hora de recoger los datos del pop-up implementado en la lógica y al intentar meter valores al calendario para insertar una fecha. El problema de recoger los datos del pop-up no ha sido solventado, el problema de meter valores al calendario para insertar una fecha, se ha solucionado tras una exhaustiva búsqueda por internet hasta dar con un método adecuado.

PRUEBA DE INTEGRACIÓN

Las pruebas unitarias han sido realizadas sobre los métodos onFiltrarDate() y onEventClicked() de la clase EventsPresenter.java. En la realización de estas pruebas se han usado los frameworks Junit, Mockito y Robolectric.

Durante la realización de las pruebas el ejecutor no encontró ningún problema.