Plan de pruebas de la historia de usuario 464705:

Ver Información Detallada gasolinera

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

- Pruebas de aceptación: Las pruebas de aceptación se definirán conforme a los criterios de aceptación definidos, habiendo una prueba por cada criterio.
- Pruebas de integración. La estrategia para la definición del orden de las pruebas de integración será incremental, probando primero la DAO con la base de datos, después el presentador con las anteriores y finalmente la vista con todas ellas. En esta última, el uso de Espresso será necesario, y en la del presentador con la DAO y BD se requerirá usar Mockito y Roboelectric, usando JUnit en todas ellas.
- Pruebas unitarias. Se utilizará la técnica de prueba de métodos para las clases baja prueba, usando técnicas de caja negra (partición equivalente y AVL) para la definición de los casos de prueba de cada método de cada clase. También se alcanzará la cobertura de decisiones para la técnica de caja blanca. Se utilizará JUnit, así como Mockito cuando sea necesario.

A continuación, se muestra una especificación detallada de los casos de prueba a aplicar en cada nivel mencionado anteriormente.

PRUEBAS DE ACEPTACIÓN VER INFORMACION DETALLADA GASOLINERA

a. Caso de Éxito al pulsar gasolinera
   1. El usuario realiza una pulsación de selección (un click) sobre una gasolinera de la lista principal de gasolineras.
   2. La aplicación abre una nueva vista donde mostrar la información de la gasolinera.
   3. La aplicación calcula el *precio sumario* para esa gasolinera.
   4. La aplicación visualiza los datos proporcionados por el servicio de datos para la gasolinera.
   5. Se verifica que los datos mostrados son correctos, incluyendo el *precio sumario*.

b. Datos ausentes

   Contexto: Uno o más de los datos a mostrar no están disponibles para esa gasolinera concreta. Por ejemplo, la gasolinera carece de diésel.

   1. El usuario realiza una pulsación de selección (un click) sobre una gasolinera de la lista principal de gasolineras.
   2. La aplicación abre una nueva vista donde mostrar la información de la gasolinera.
   3. La aplicación calcula el *precio sumario* para esa gasolinera.
   4. La aplicación visualiza los datos proporcionados por el servicio de datos para la gasolinera.

5. Se verifica que, para los los datos existentes, los valores mostrados son correctos.
6. Se verifica que para los datos no existentes se muestra un guion, indicando que el dato no está disponible.
7. Se verifica que el *precio sumario* se ha calculado utilizando sólo los datos existentes, ignorando los no existentes, y que su valor es correcto.

NOTA: En el caso extremo de no haber ningún dato disponible, el *precio sumario* se considerará como no existente y se mostrará con su correspondiente guion.

c. Precios anómalos.
Contexto: Los precios de algún carburante tienen valores negativos.

1. El usuario realiza una pulsación de selección (un click) sobre una gasolinera de la lista principal de gasolineras.
2. La aplicación abre una nueva vista donde mostrar la información de la gasolinera.
3. La aplicación calcula el *precio sumario* para esa gasolinera.
4. La aplicación visualiza los datos proporcionados por el servicio de datos para la gasolinera.
5. Se verifica que los datos sin valores anómalos se muestran correctamente.
6. Se verifica que para los datos con valores anómalos se muestra un guion,
7. indicando que el dato no está disponible, ya que hay un problema con ellos.
8. Se verifica que el *precio sumario* se ha calculado utilizando sólo los datos existentes y válidos, ignorando los no existentes, y que su valor es correcto.

   NOTA: En el caso extremo de que todos los valores sean anómalos o no existentes, el *precio sumario* se considerará como no existente y se mostrará con su correspondiente guion.

d. Gasolinera sin información.

Contexto: La gasolinera seleccionada no dispone de datos más allá de su nombre.

1. El usuario realiza una pulsación de selección (un click) sobre una gasolinera de la lista principal de gasolineras.
2. La aplicación abre una nueva vista donde mostrar la información de la gasolinera.
3. Se verifica que el sistema muestra en la vista simplemente un mensaje informando que hay un problema con los datos de esa gasolinera.

PRUEBAS DE INTEGRACIÓN

El orden de las pruebas y los escenarios de prueba a realizar serían los siguientes:

1. ConveniosDAO con la base de datos de la aplicación, que usa Room. Se usarían los mismos casos de prueba que los definidos para las pruebas unitarias de ConveniosDAO, redefinidos como UPR464971.X

2. ConveniosPresenter con ConveniosDAO y base de datos. Estas pruebas coincidirían con las pruebas de aceptación, renombradas como IPR464971.X Se utilizará Mockito para crear un mock de la vista, y será necesario el uso de Roboelectric para utilizar la DAO.

3. ConveniosView con ConveniosPresenter con la base de datos y con ConveniosDAO. Estas pruebas coincidirían con las pruebas de aceptación, renombradas como IPR464971.X Se utilizará Mockito para crear un mock de la vista, y será necesario el uso de Roboelectric para utilizar la DAO.

4.BarraDeHerramientasView con BarraDeHerramientasPresenter con conveniosView y convenios presenter. Estas pruebas coincidirían con las pruebas unitarias, renombradas como IPR464971.X

5.MainPresenter con detallesPresenter con GasolienrasDAO. Estas pruebas coincidirían con las pruebas unitarias, renombradas como IPR464705.X

6. MainPresenter con MainView y con detallesView y detallesPresenter. Estas pruebas coincidirían con las pruebas de aceptación, aunque en este caso se automatizarían utilizando la librería Espresso. Estas pruebas de la interfaz gráfica se codificarán como IGUI464705.x.

Pruebas de integración del Presentador de Convenios

Se van a codificar las pruebas de integración IPR464971 x (del presentador de Convenios con la DAO y la base de datos), cuyos casos de prueba se corresponden con los de sus pruebas unitarias, UPR64971.x, pero renombrados.

Los casos definidos se muestran a continuación.

A. Método onSiSobrescribir(): En estos casos, v es un objeto de IConveniosContract.View y dao uno de ConvenioDao.

Iderntificador	Contexto	Resultado esperado
IPR464971.1A	Se ha pulsado onAnhadirClicked() y se sobrescribe un convenio que ya existía con los datos: Marca: Campsa Descuento: 80	Se llama a dao.insertConvenio(c) v.refresh() v.showConvenioAnhadido() Se comprueba que la lista de convenios se actualiza

B. Método onNoSobrescribir(): En estos casos, v es un objeto de IConveniosContract.View y dao uno de ConvenioDao.

Iderntificador	Contexto	Resultado esperado
IPR464971.2A	Se ha pulsado onAnhadirClicked() y ya existía un convenio con esa marca, pero no lo sobrescribimos	Se cierra la ventana emergente Se comprueba que la lista se mantiene igual

Pruebas integración del presentador de BarraHerramientas

Se van a codificar las pruebas de integración IPR464971 x (de la barra de herramientas con los convenios), cuyos casos de prueba se corresponden con los de sus pruebas unitarias, UPR64971.x, excepto el B que no se podía comprobar en la unitaria porque usamos la clase ConveniosView y ConveniosPresenter.

A. Método onAnhadeConvenioClicked(): En estos casos, v es un objeto de IBarraDeHerramientasContract.View y Pref ANHADIR es un campo de las preferencias locales. A su vez VConvenio es un objeto de IConvenioContract.View

Identificador	Contexto	Resultado esperado
IPR464971.3A	Se ha iniciado la app y queremos ver los convenios	Se Llama a: v.openConveniosView() Las preferencias cambian Pref ANHADIR = 1 Se llama a: VConvenio.showAnhadirConvenio() Pref ANHADIR = 0
IPR464971.3B	Se ha iniciado la app y queremos ver los convenios, pero no cargan las gasolineras	Se Llama a: v.openConveniosView() Las preferencias cambian Pref ANHADIR = 1 No se llama a: VConvenio.showAnhadirConvenio() Pref ANHADIR = 0

Pruebas integración del presentador Main

Se van a codificar las pruebas de integración IPR464705 x (del MainView y MainPresenter con GasolineraDetailPresenter y GasolienraDetailView), cuyos casos de prueba se corresponden con los de sus pruebas unitarias, UPR464705.x, pero ahora comprobamos los resultados en detalle.

A. Método OnGasolineraClicked(int index): En estos casos v es un objeto de IMainContract.View y gasolinera es el objeto sobre el que el usuario clica.

Iderntificador	Contexto	Resultado esperado
IPR464705.4A	La gasolinera de ese índice (primera gasolinera) existe y todos sus datos son correctos DieselA: “1,999" Normal95: "1,859" Municipio: "Alfoz de Lloredo" Rotulo: "CEPSA” Schedule: "L-D: 08:00-21:00"	Se Llama a: v.openGasolineraDetails(gasolinera) Se comprueba que los datos son los esperados DieselA: “1,99 €" Normal95:"1,85 €" Municipio: "Alfoz de Lloredo" PrecioSumario:”1,90 €” Rotulo: "CEPSA” Schedule: "L-D: 08:00-21:00
IPR464705.4B	La gasolinera de ese índice (última gasolinera) existe y tiene datos ausentes. Municipio: "Voto" Normal95:"1,799" Rotulo:"" DieselA:""	Se Llama a: v.openGasolineraDetails(gasolinera) Se comprueba que los datos ausentes son un - Municipio: "Voto" Normal95:"1,79 €" Rotulo:"-"

	Schedule: "L-D: 09:00-20:30");	PrecioSumario"1,79 €") DieselA:"-" Schedule: "L-D: 09:00-20:30");
IPR464705.4C	La gasolinera de ese índice (ante última gasolinera) existe y tiene datos de precios anómalos siendo estos: Municipio: "Villafufre" Normal95:"1,738" Rotulo: "AVIA" DieselA:”-1,808" Schedule: "L-S: 06:00-22:00; D: 08:00-22:00");	Se Llama a: v.openGasolineraDetails(gasolinera) Se comprueba que los datos anómalos son un - Municipio: "Villafufre"; Normal95:"1,73 €” Rotulo: "AVIA" DieselA:”-" PrecioSumario,"1,73 €" Schedule: "L-S: 06:00-22:00; D: 08:00-22:00"
IPR464705.4D	La gasolinera de ese índice existe y no tiene ningún dato	Se Llama a: v.openGasolineraDetails(gasolinera) Se comprueba que se lanza un mensaje
IPR464705.4E	Se introduce el índice -1	No se hace nada y la app sigue funcionando
IPR464705.4F	Se introduce el índice 100000000	No se hace nada y la app sigue funcionando

Pruebas de la interfaz gráfica

Se va a realizar una prueba de interfaz gráfica que compruebe que en la vista de detalle de las gasolineras se muestren todos los datos correctos de una gasolinera tras llamar al método OnGasolineraClicked(int index). En este caso se implementará IGUI464705.1A.

A continuación, se describe en detalle la implementación concreta de este caso de prueba.

Identificador	Contexto	Resultado esperado

IGUI464705.1A	El usuario pulsa sobre la primera gasolinera que aparece en la lista	Se muestran los campos: -Logo -Nombre: CEPSA -Calle: Alfoz de Lloredo -Precio medio: 1,90 € -Precio diésel: 1,99 € -Precio gasolina 1,85€ -Horario: L-D: 08:00-21:00 -TextBox del Horario -Distancia: 7072,50 km -TextBoxDistancia
IGUI464705.1B	El usuario pulsa sobre la segunda gasolinera. Esta contiene precios anómalos.	Se muestran los campos: -Logo: -Nombre: CEPSA -Calle: Ampuero -Precio medio: 1,85€ -Precio diésel: - -Precio gasolina 1,85€ -Horario: L-D: 08:00-21:00 -TextBox del Horario -Distancia: 7073,50 km -TextBoxDistancia
IGUI464705.1C	El usuario pulsa sobre la tercera gasolinera. Esta contiene datos ausentes.	Se muestran los campos: -Logo: -Nombre: - -Calle: Arenas de Iguña -Precio medio: 1,85€ -Precio diésel: - -Precio gasolina 1,85€ -Horario: L-D: - -TextBox del Horario -Distancia: 9073,50 km -TextBoxDistancia
IGUI464705.1C	El usuario pulsa sobre la cuarta gasolinera de la lista. Esta no tiene ningún dato	Se muestra una ventana en blanco con un mensaje que pone: “Error no se han cargado los datos”

PRUEBAS UNITARIAS

En este plan solo se detallan las pruebas unitarias del presentador, sin definir ninguna para las clases de la vista o DAO, y por lo tanto ignorándolas en esta sección.

Se aplica prueba de métodos, siendo los casos de prueba definidos para cada método los que se exponen a continuación. Los casos expuestos para cada método parten del contexto indicado.

Pruebas unitarias del presentador de Convenios

A. Método onSiSobrescribirClicked(): En estos casos, v es un objeto de IConveniosContract.View y dao uno de ConvenioDao.

Identificador	Contexto	Resultado esperado
UPR464971.1A	Se ha pulsado onAnhadirClicked() y se sobrescribe un convenio que ya existía con los datos: Marca: Campsa Descuento: 80	Se llama a dao.updateConvenio(c) v.refresh() v.showConvenioAnhadido()

B. Método onNoSobrescribirclicked():

Identificador	Contexto	Resultado esperado
UPR464971.2A	Se ha pulsado onAnhadirClicked() y ya existía un convenio con esa marca, pero no lo sobrescribimos	Se cierra la ventana emergente No se llama a: v.refresh() v.showConvenioAnhadido()

Pruebas unitarias del presentador de BarraHerramientas

A. Método onAnhadeConvenioClicked(): En estos casos, v es un objeto de IBarraDeHerramientasContract.View y Pref ANHADIR es un campo de las preferencias locales.

Identificador	Contexto	Resultado esperado
UPR464971.3A	Se ha iniciado la app y queremos ver los convenios	Se Llama a: v.openConveniosView() Pref.putInt(ANHADIR,1)

Pruebas unitarias del presentador Main

A. Método OnGasolineraClicked(int index): En estos casos v es un objeto de IMainContract.View y gasolinera es el objeto sobre el que el usuario clica.

Identificador	Contexto	Resultado esperado
UPR464705.4A	Se ha llamado a v.showGasolineras() y pulsamos sobre una con todos los datos correctos	Se Llama a: v.openGasolineraDetails(gasolinera)
UPR464705.4B	Se ha llamado a v.showGasolineras() y pulsamos	Se Llama a: v.openGasolineraDetails(gasolinera)

	sobre una con datos ausentes	
UPR464705.4C	Se ha llamado a v.showGasolineras() y pulsamos sobre una con datos anomalos	Se Llama a: v.openGasolineraDetails(gasolinera)
UPR464705.4D	Se ha llamado a v.showGasolineras() y pulsamos sobre una sin datos	Se Llama a: v.openGasolineraDetails(gasolinera) Se comprueba que se lanza un mensaje
UPR464705.4E	El índice de la gasolinera es -1	No se hace nada, pero la app sigue funcionando
UPR464705.4F	El índice de la gasolinera es 100000000	No se hace nada, pero la app sigue funcionando

Informe de pruebas

Prueba de integración:

-Gracias al test IPR464705.4C me di cuenta de que a la hora de llamar al método substring(0, 4) puede darte un IndexOutOfBounds si el string que queremos recortar esta vacío. Como esto se utiliza para cortar el precio del Diesel y de la gasolina que viene de la respuesta de la api (y pueden estar vacíos) lo que hacemos es simplemente recoger la excepción IndexOutOfBounds y devolver el string "-" para que la aplicación siga funcionando correctamente.

-Gracias al test IPR464971.1A me he dado cuenta que no estaba bien implementada la lógica del método onSiSobreescribirClicked(Convenio c), en el que previamente solo se añadía el convenio ahí y se eliminaba en onConvenioAnhadirClicked lo cual no tenía mucho sentido y hacía al programa menos legible. En la primera resolución se planteó hacer un update pero este no se implementaba bien puesto que actualizaba el convenio anterior pero con sus datos anteriores. Fue necesaria una segunda revisión para hacer que actualizase el convenio anterior con los nuevos datos.

Prueba unitarias:

-Gracias al test UPR464705.4E me he dado cuenta que no se contemplaba la posibilidad de introducir un índice negativo, lo cual a priori es imposible, pero para evitar posibles problemas se ha añadido la condición de que el índice sea mayor que cero en el método onGasolineraClicked(int index)

-Gracias al test UPR464971.1A me he dado cuenta de que es más inteligente hacer un update antes que borrar al convenio para después volver a añadirlo con los nuevos datos en el método onSiSobreescribirClicked(Convenio c)

Pruebas de interfaz:

No se ha encontrado ningún error.

Marcos Fernández Alonso