Plan de pruebas US 505814-Mostrar puntos de interés

Pruebas de UI

Son las mismas pruebas que las pruebas de aceptación, renombradas como “UI.x”. Los datos de los puntos de interés se filtran en base a las tablas de datos situadas al final del documento.

Identificador	Entrada	Resultado
UI.1	puntos_interes_ejemplos_1	muestra: [punto 2, punto 1, punto 3]
UI.2	puntos_interes_vacio	muestra que no hay puntos
UI.3	puntos_interes_ejemplos_1	muestra: [punto 2, punto 1, punto 3] Se vuelve a la pantalla de inicio
UI.4	Error en la DAO	muestra Toast con mensaje de error

Pruebas de unitarias

En esta historia de usuario se accede a persistencia de datos, pero como está hecha con room no es necesario probarla. También tiene una capa de negocio de la clase `InterestPoint` pero como es una clase unicamente de datos con getters y setters no es necesario probarla. Por lo tanto, las pruebas unitarias solo consistirán en pruebas de las clases de negocio y presentación.

Pruebas unitarias de clases de negocio

Deben de probarse los métodos de la clase `PointsPresenter` mediante el uso de mocks de `IPointsContract#View` y `InterestPointsDAO`

Deberían de probarse los siguientes metodos de la clase `PointsPresenter`:

- init(view : IPointsContract.View) : Void
- onHomeClicked( ): Void

Se va a implementar la prueba unitaria del método **init(view : IPointsContract.View)**:

Identificador	Entrada	Valor esperado
UD1.a	DAO con: puntos_interes_ejemplos_1	1- View: llamado a init() 2- View: llamado a showpoints([punto 2, punto 1, punto 3])
UD1.b	DAO con: puntos_interes_vacio	1- View: llamado a init() 2- View: llamado a showpoints([])

Identificador	Entrada	Valor esperado
UD1.c	Error en la DAO	1- View: llamado a init() 2- View: llamado a showLoadError()

Pruebas de integración

Para las pruebas de integración se ha añadido las interfaces de `IPointsContract#View`, `IPointsContract#Presenter` y `InterestPointsDAO`. Se va a probar la integración entre el `PointsPresenter` y `InterestPointsDAO` mockeando el `PointsView`

Se probará el funcionamiento de:

- init(view : IPointsContract.View) : Void
- onHomeClicked( ): Void

Como todos los métodos son muy sencillos se va a implementar el método **onFiltersPopUpFuelTypesOneSelected(int index, boolean value): void** de la historia de usuario *feature/500955-Filtrar_por_tipo_de_combustible*. Esto debería de estar en el plan de pruebas de esa historia de usuario, pero como es de un sprint viejo y no disponemos del archivo viejo esta escrito aquí:

Identificador	Entrada	Valor esperado

UD3.a	tempListSelection: {
	Selection(Todos, False),
	Selection(Gasolina 95, False),
	Selection(Diesel, True)
	}
	index: 0
	value: True
		- View: llamado a updateFiltersPopUpSelection(1, false)
		- View: llamado a updateFiltersPopUpSelection(2, false)
		- Presenter.tempListSelection: {
		Selection(Todos, True),
		Selection(Gasolina 95, False),
		Selection(Diesel, False)
		}

UD3.b	tempListSelection: {
	Selection(Todos, True),
	Selection(Gasolina 95, False),
	Selection(Diesel, False)
	}
	index: 0
	value: False
		- View: llamado a updateFiltersPopUpSelection(0, true)
		- Presenter.tempListSelection: {
		Selection(Todos, True),
		Selection(Gasolina 95, False),
		Selection(Diesel, False)
		}

UD3.c	tempListSelection: {
	Selection(Todos, True),
	Selection(Gasolina 95, False),
	Selection(Diesel, False)
	}
	index: 2
	value: True
		- View: llamado a updateFiltersPopUpSelection(0, false)
		- Presenter.tempListSelection: {
		Selection(Todos, False),
		Selection(Gasolina 95, False),
		Selection(Diesel, True)
		}

Identificador	Entrada	Valor esperado

UD3.d	tempListSelection: {
	Selection(Todos, False),
	Selection(Gasolina 95, False),
	Selection(Diesel, True)
	}
	index: 1
	value: True
		- View: llamado a updateFiltersPopUpSelection(0, true)
		- View: llamado a updateFiltersPopUpSelection(1, false)
		- View: llamado a updateFiltersPopUpSelection(2, false)
		- Presenter.tempListSelection: {
		Selection(Todos, True),
		Selection(Gasolina 95, False),
		Selection(Diesel, False)
		}

UD3.e	tempListSelection: {
	Selection(Todos, False),
	Selection(Gasolina 95, False),
	Selection(Diesel, True)
	}
	index: 2
	value: False
		- View: llamado a updateFiltersPopUpSelection(0, true)
		- View: llamado a updateFiltersPopUpSelection(1, false)
		- Presenter.tempListSelection: {
		Selection(Todos, True),
		Selection(Gasolina 95, False),
		Selection(Diesel, False)
		}

Reporte final

---------- TODO : Indicar que pruebas se han hecho ----------

Autoría

- Plan de pruebas: Pablo Landeras
- La codificación y ejecución de pruebas unitarias y de integración: Lucía Sañudo
- La codificación y ejecución de pruebas de UI: Adrián del Río

Tablas de datos

Estos son listas de puntos de interés creadas para poder tener datos de prueba y comprobar el comportamiento en todos los casos

puntos_interes_vacio

id	name	color	latitude	longitude	radius	creationDate
----	------	-------	----------	-----------	--------	--------------

puntos_interes_ejemplos_1

id	name	color	latitude	longitude	radius	creationDate

id	name	color	latitude	longitude	radius	creationDate
1	punto 1	#ff0000	45.0000	-123.3450	12.4	12/08/2024
2	punto 2	#00ff00	65.0400	23.3770	6.0	10/07/2024
3	punto 3	#0000ff	-25.6783	3.3422	53.2	01/10/2024