

# Documentacion

1

Generado por Doxygen 1.12.0

---

<b>1 Índice de espacios de nombres</b>	<b>1</b>
1.1 Lista de paquetes . . . . .	1
<b>2 Índice jerárquico</b>	<b>2</b>
2.1 Jerarquía de clases . . . . .	2
<b>3 Índice de clases</b>	<b>3</b>
3.1 Lista de clases . . . . .	3
<b>4 Índice de archivos</b>	<b>5</b>
4.1 Lista de archivos . . . . .	5
<b>5 Documentación de espacios de nombres</b>	<b>6</b>
5.1 Paquete com.example.smariba_upv.airflow . . . . .	6
5.2 Paquete com.example.smariba_upv.airflow.API . . . . .	7
5.3 Paquete com.example.smariba_upv.airflow.API.MODELS . . . . .	7
5.4 Paquete com.example.smariba_upv.airflow.LOGIC . . . . .	7
5.5 Paquete com.example.smariba_upv.airflow.POJO . . . . .	7
5.6 Paquete com.example.smariba_upv.airflow.PRESENTACION . . . . .	8
5.7 Paquete com.example.smariba_upv.airflow.Services . . . . .	8
5.8 Paquete com.example.smariba_upv.btle_sento . . . . .	8
5.9 Paquete com.example.smariba_upv.btle_sento.API . . . . .	8
5.10 Paquete com.example.smariba_upv.btle_sento.LOGIC . . . . .	9
5.11 Paquete com.example.smariba_upv.btle_sento.POJO . . . . .	9
<b>6 Documentación de clases</b>	<b>9</b>
6.1 Referencia de la interface com.example.smariba_upv.airflow.API.ApiService . . . . .	9
6.1.1 Descripción detallada . . . . .	10
6.1.2 Documentación de funciones miembro . . . . .	10
6.2 Referencia de la interface com.example.smariba_upv.btle_sento.API.ApiService . . . . .	12
6.2.1 Descripción detallada . . . . .	12
6.2.2 Documentación de funciones miembro . . . . .	13
6.3 Referencia de la clase com.example.smariba_upv.airflow.Services.ArduinoGetterService . . . . .	14
6.3.1 Descripción detallada . . . . .	15
6.3.2 Documentación de funciones miembro . . . . .	15
6.4 Referencia de la interface com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener . . . . .	18
6.4.1 Descripción detallada . . . . .	20
6.4.2 Documentación de funciones miembro . . . . .	20
6.5 Referencia de la clase com.example.smariba_upv.airflow.LOGIC.BiometricUtil . . . . .	21
6.5.1 Descripción detallada . . . . .	22
6.5.2 Documentación de funciones miembro . . . . .	22
6.6 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity . . . . .	23
6.6.1 Descripción detallada . . . . .	24
6.6.2 Documentación de funciones miembro . . . . .	25

---

6.7 Referencia de la clase com.example.smariba_upv.airflow.API.EnviarPeticionesUser . . . . .	26
6.7.1 Descripción detallada . . . . .	27
6.7.2 Documentación de constructores y destructores . . . . .	27
6.7.3 Documentación de funciones miembro . . . . .	27
6.8 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.HomeFragment . . . . .	32
6.8.1 Descripción detallada . . . . .	33
6.8.2 Documentación de constructores y destructores . . . . .	34
6.8.3 Documentación de funciones miembro . . . . .	34
6.9 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.LandActivity . . . . .	35
6.9.1 Descripción detallada . . . . .	36
6.9.2 Documentación de funciones miembro . . . . .	36
6.10 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.LogInActivity . . . . .	37
6.10.1 Descripción detallada . . . . .	39
6.10.2 Documentación de funciones miembro . . . . .	40
6.11 Referencia de la clase com.example.smariba_upv.airflow.MainActivity . . . . .	42
6.11.1 Descripción detallada . . . . .	45
6.11.2 Documentación de funciones miembro . . . . .	45
6.12 Referencia de la clase com.example.smariba_upv.btle_sento.MainActivity . . . . .	47
6.12.1 Descripción detallada . . . . .	49
6.12.2 Documentación de funciones miembro . . . . .	49
6.13 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.MapFragment . . . . .	51
6.13.1 Descripción detallada . . . . .	52
6.13.2 Documentación de constructores y destructores . . . . .	53
6.13.3 Documentación de funciones miembro . . . . .	53
6.14 Referencia de la clase com.example.smariba_upv.airflow.POJO.Medicion . . . . .	55
6.14.1 Descripción detallada . . . . .	56
6.14.2 Documentación de constructores y destructores . . . . .	56
6.14.3 Documentación de funciones miembro . . . . .	57
6.15 Referencia de la clase com.example.smariba_upv.btle_sento.POJO.Medicion . . . . .	60
6.15.1 Descripción detallada . . . . .	60
6.15.2 Documentación de constructores y destructores . . . . .	61
6.15.3 Documentación de funciones miembro . . . . .	61
6.16 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.PaginaDeCarga . . . . .	64
6.16.1 Descripción detallada . . . . .	65
6.16.2 Documentación de funciones miembro . . . . .	65
6.17 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity . . . . .	65
6.17.1 Descripción detallada . . . . .	66
6.17.2 Documentación de funciones miembro . . . . .	66
6.17.3 Documentación de datos miembro . . . . .	67
6.18 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment . . . . .	68
6.18.1 Descripción detallada . . . . .	70
6.18.2 Documentación de constructores y destructores . . . . .	70

6.18.3 Documentación de funciones miembro . . . . .	70
6.19 Referencia de la clase com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil . . . . .	72
6.19.1 Descripción detallada . . . . .	73
6.19.2 Documentación de funciones miembro . . . . .	73
6.20 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.QRreader . . . . .	75
6.20.1 Descripción detallada . . . . .	77
6.20.2 Documentación de funciones miembro . . . . .	77
6.21 Referencia de la clase com.example.smariba_upv.airflow.API.RetrofitClient . . . . .	79
6.21.1 Descripción detallada . . . . .	79
6.21.2 Documentación de funciones miembro . . . . .	79
6.22 Referencia de la clase com.example.smariba_upv.btle_sento.API.RetrofitClient . . . . .	80
6.22.1 Descripción detallada . . . . .	80
6.22.2 Documentación de funciones miembro . . . . .	81
6.22.3 Documentación de datos miembro . . . . .	81
6.23 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.SaludFragment . . . . .	82
6.23.1 Descripción detallada . . . . .	83
6.23.2 Documentación de constructores y destructores . . . . .	84
6.23.3 Documentación de funciones miembro . . . . .	84
6.24 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter . . . . .	85
6.24.1 Descripción detallada . . . . .	86
6.24.2 Documentación de constructores y destructores . . . . .	87
6.24.3 Documentación de funciones miembro . . . . .	87
6.25 Referencia de la clase com.example.smariba_upv.airflow.PRESENTACION.SensorFragment . . . . .	89
6.25.1 Descripción detallada . . . . .	90
6.25.2 Documentación de constructores y destructores . . . . .	90
6.25.3 Documentación de funciones miembro . . . . .	91
6.26 Referencia de la clase com.example.smariba_upv.airflow.POJO.SensorObject . . . . .	92
6.26.1 Descripción detallada . . . . .	93
6.26.2 Documentación de constructores y destructores . . . . .	93
6.26.3 Documentación de funciones miembro . . . . .	93
6.27 Referencia de la clase com.example.smariba_upv.airflow.API.MODELS.SensorRequest . . . . .	97
6.27.1 Descripción detallada . . . . .	98
6.27.2 Documentación de constructores y destructores . . . . .	98
6.27.3 Documentación de funciones miembro . . . . .	99
6.28 Referencia de la clase com.example.smariba_upv.airflow.API.MODELS.SensorResponse . . . . .	103
6.28.1 Descripción detallada . . . . .	103
6.28.2 Documentación de funciones miembro . . . . .	104
6.29 Referencia de la clase com.example.smariba_upv.airflow.POJO.TramalBeacon . . . . .	105
6.29.1 Descripción detallada . . . . .	106
6.29.2 Documentación de constructores y destructores . . . . .	106
6.29.3 Documentación de funciones miembro . . . . .	106
6.30 Referencia de la clase com.example.smariba_upv.btle_sento.POJO.TramalBeacon . . . . .	109

6.30.1 Descripción detallada . . . . .	110
6.30.2 Documentación de constructores y destructores . . . . .	110
6.30.3 Documentación de funciones miembro . . . . .	110
6.31 Referencia de la clase com.example.smariba_upv.airflow.POJO.User . . . . .	113
6.31.1 Descripción detallada . . . . .	114
6.31.2 Documentación de constructores y destructores . . . . .	114
6.31.3 Documentación de funciones miembro . . . . .	115
6.32 Referencia de la clase com.example.smariba_upv.airflow.LOGIC.Utilidades . . . . .	119
6.32.1 Descripción detallada . . . . .	121
6.32.2 Documentación de funciones miembro . . . . .	121
6.33 Referencia de la clase com.example.smariba_upv.btle_sento.LOGIC.Utilidades . . . . .	126
6.33.1 Descripción detallada . . . . .	128
6.33.2 Documentación de funciones miembro . . . . .	128
<b>7 Documentación de archivos</b> . . . . .	<b>133</b>
7.1 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/ApiService.java . . . . .	133
7.1.1 Descripción detallada . . . . .	134
7.2 AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/ApiService.java . . . . .	134
7.3 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/ApiService.java . . . . .	135
7.3.1 Descripción detallada . . . . .	135
7.4 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/ApiService.java . . . . .	136
7.5 Referencia del archivo EnviarPeticionesUser.java . . . . .	136
7.6 EnviarPeticionesUser.java . . . . .	137
7.7 Referencia del archivo SensorRequest.java . . . . .	139
7.7.1 Descripción detallada . . . . .	139
7.8 SensorRequest.java . . . . .	140
7.9 Referencia del archivo SensorResponse.java . . . . .	141
7.9.1 Descripción detallada . . . . .	142
7.10 SensorResponse.java . . . . .	142
7.11 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/RetrofitClient.java . . . . .	143
7.11.1 Descripción detallada . . . . .	144
7.12 AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/RetrofitClient.java . . . . .	144
7.13 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/RetrofitClient.java . . . . .	145
7.13.1 Descripción detallada . . . . .	145
7.14 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/RetrofitClient.java . . . . .	146
7.15 Referencia del archivo BiometricUtil.java . . . . .	146
7.15.1 Descripción detallada . . . . .	147

7.16 BiometricUtil.java . . . . .	147
7.17 Referencia del archivo PeticionesUserUtil.java . . . . .	148
7.18 PeticionesUserUtil.java . . . . .	148
7.19 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/LOGIC/Utilidades.java . . . . .	149
7.19.1 Descripción detallada . . . . .	150
7.20     AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/LOGIC/Utilidades.java . . . . .	150
7.21 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/LOGIC/Utilidades.java . . . . .	151
7.21.1 Descripción detallada . . . . .	152
7.22 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/LOGIC/Utilidades.java . . . . .	153
7.23 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/MainActivity.java . . . . .	154
7.23.1 Descripción detallada . . . . .	155
7.24     AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/MainActivity.java . . . . .	155
7.25 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/MainActivity.java . . . . .	160
7.25.1 Descripción detallada . . . . .	161
7.26 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/MainActivity.java . . . . .	161
7.27 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/Medicion.java . . . . .	166
7.27.1 Descripción detallada . . . . .	166
7.28     AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/Medicion.java . . . . .	167
7.29 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/Medicion.java . . . . .	167
7.29.1 Descripción detallada . . . . .	168
7.30 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/Medicion.java . . . . .	168
7.31 Referencia del archivo SensorObject.java . . . . .	170
7.32 SensorObject.java . . . . .	170
7.33 Referencia del archivo AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/TramalBeacon.java . . . . .	172
7.33.1 Descripción detallada . . . . .	172
7.34     AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/TramalBeacon.java . . . . .	173
7.35 Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/TramalBeacon.java . . . . .	174
7.35.1 Descripción detallada . . . . .	174
7.36 Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/TramalBeacon.java . . . . .	175
7.37 Referencia del archivo User.java . . . . .	176

7.38 User.java . . . . .	176
7.39 Referencia del archivo EditarPerfilActivity.java . . . . .	177
7.39.1 Descripción detallada . . . . .	177
7.40 EditarPerfilActivity.java . . . . .	178
7.41 Referencia del archivo HomeFragment.java . . . . .	179
7.42 HomeFragment.java . . . . .	179
7.43 Referencia del archivo LandActivity.java . . . . .	180
7.43.1 Descripción detallada . . . . .	181
7.44 LandActivity.java . . . . .	181
7.45 Referencia del archivo LogInActivity.java . . . . .	182
7.45.1 Descripción detallada . . . . .	183
7.46 LogInActivity.java . . . . .	183
7.47 Referencia del archivo MapFragment.java . . . . .	186
7.48 MapFragment.java . . . . .	186
7.49 Referencia del archivo PaginaDeCarga.java . . . . .	187
7.50 PaginaDeCarga.java . . . . .	188
7.51 Referencia del archivo PerfilActivity.java . . . . .	188
7.52 PerfilActivity.java . . . . .	189
7.53 Referencia del archivo PerfilFragment.java . . . . .	190
7.53.1 Descripción detallada . . . . .	190
7.54 PerfilFragment.java . . . . .	190
7.55 Referencia del archivo QRreader.java . . . . .	191
7.55.1 Descripción detallada . . . . .	192
7.56 QRreader.java . . . . .	192
7.57 Referencia del archivo SaludFragment.java . . . . .	194
7.58 SaludFragment.java . . . . .	195
7.59 Referencia del archivo SensorAdapter.java . . . . .	196
7.60 SensorAdapter.java . . . . .	196
7.61 Referencia del archivo SensorFragment.java . . . . .	198
7.62 SensorFragment.java . . . . .	198
7.63 Referencia del archivo ArduinoGetterService.java . . . . .	200
7.64 ArduinoGetterService.java . . . . .	200
<b>Índice alfabético</b>	<b>207</b>

## 1. Índice de espacios de nombres

### 1.1. Lista de paquetes

Estos son los paquetes con breves descripciones (si están disponibles):

**com.example.smariba\_upv.airflow**

**6**

com.example.smariba_upv.airflow.API	7
com.example.smariba_upv.airflow.API.MODELS	7
com.example.smariba_upv.airflow.LOGIC	7
com.example.smariba_upv.airflow.POJO	7
com.example.smariba_upv.airflow.PRESENTACION	8
com.example.smariba_upv.airflow.Services	8
com.example.smariba_upv.btle_sento	8
com.example.smariba_upv.btle_sento.API	8
com.example.smariba_upv.btle_sento.LOGIC	9
com.example.smariba_upv.btle_sento.POJO	9

## 2. Índice jerárquico

### 2.1. Jerarquía de clases

Este listado de herencia está ordenado de forma general pero no está en orden alfabético estricto:

RecyclerView.Adapter	
com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter	85
com.example.smariba_upv.airflow.API.ApiService	9
com.example.smariba_upv.btle_sento.API.ApiService	12
com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener	18
com.example.smariba_upv.airflow.PRESENTACION.LogInActivity	37
com.example.smariba_upv.airflow.LOGIC.BiometricUtil	21
com.example.smariba_upv.airflow.APIEnviarPeticionesUser	26
com.example.smariba_upv.airflow.POJO.Medicion	55
com.example.smariba_upv.btle_sento.POJO.Medicion	60
com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil	72
com.example.smariba_upv.airflow.API.RetrofitClient	79
com.example.smariba_upv.btle_sento.API.RetrofitClient	80
com.example.smariba_upv.airflow.POJO.SensorObject	92
com.example.smariba_upv.airflow.API.MODELS.SensorRequest	97
com.example.smariba_upv.airflow.API.MODELS.SensorResponse	103
com.example.smariba_upv.airflow.POJO.TramalBeacon	105

com.example.smariba_upv.btle_sento.POJO.TramalBeacon	109
com.example.smariba_upv.airflow.POJO.User	113
com.example.smariba_upv.airflow.LOGIC.Utilidades	119
com.example.smariba_upv.btle_sento.LOGIC.Utilidades	126
AppCompatActivity	
com.example.smariba_upv.airflow.MainActivity	42
com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity	23
com.example.smariba_upv.airflow.PRESENTACION.LandActivity	35
com.example.smariba_upv.airflow.PRESENTACION.LogInActivity	37
com.example.smariba_upv.airflow.PRESENTACION.PaginaDeCarga	64
com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity	65
com.example.smariba_upv.airflow.PRESENTACION.QRreader	75
com.example.smariba_upv.btle_sento.MainActivity	47
Fragment	
com.example.smariba_upv.airflow.PRESENTACION.HomeFragment	32
com.example.smariba_upv.airflow.PRESENTACION.MapFragment	51
com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment	68
com.example.smariba_upv.airflow.PRESENTACION.SaludFragment	82
com.example.smariba_upv.airflow.PRESENTACION.SensorFragment	89
Service	
com.example.smariba_upv.airflow.Services.ArduinoGetterService	14

## 3. Índice de clases

### 3.1. Lista de clases

Lista de clases, estructuras, uniones e interfaces con breves descripciones:

com.example.smariba_upv.airflow.API.ApiService	
Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit	9
com.example.smariba_upv.btle_sento.API.ApiService	
Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit	12
com.example.smariba_upv.airflow.Services.ArduinoGetterService	
com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener	
Interfaz para gestionar los eventos de la autenticación biométrica	18
com.example.smariba_upv.airflow.LOGIC.BiometricUtil	
Clase que gestiona la autenticación biométrica	21

<a href="#">com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity</a>	
Clase que permite editar el perfil del usuario	23
<a href="#">com.example.smariba_upv.airflow.API.EnviarPeticionesUser</a>	
	26
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.HomeFragment</a>	
A simple Fragment subclass	32
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.LandActivity</a>	
	35
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.LoginActivity</a>	
Clase que permite iniciar sesión en la aplicación	37
<a href="#">com.example.smariba_upv.airflow.MainActivity</a>	
La clase principal que gestiona la interfaz y las funciones BTLE	42
<a href="#">com.example.smariba_upv.btle_sento.MainActivity</a>	
La clase principal que gestiona la interfaz y las funciones BTLE	47
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.MapFragment</a>	
A simple Fragment subclass	51
<a href="#">com.example.smariba_upv.airflow.POJO.Medicion</a>	
	55
<a href="#">com.example.smariba_upv.btle_sento.POJO.Medicion</a>	
Clase que encapsula los datos de una medición de gas, incluyendo el lugar, el tipo de gas y su valor	60
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.PaginaDeCarga</a>	
	64
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity</a>	
	65
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment</a>	
Clase que permite visualizar el perfil del usuario	68
<a href="#">com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil</a>	
	72
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.QRreader</a>	
Clase que permite leer un código QR	75
<a href="#">com.example.smariba_upv.airflow.API.RetrofitClient</a>	
Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST	79
<a href="#">com.example.smariba_upv.btle_sento.API.RetrofitClient</a>	
Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST	80
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.SaludFragment</a>	
	82
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter</a>	
	85
<a href="#">com.example.smariba_upv.airflow.PRESENTACION.SensorFragment</a>	
	89
<a href="#">com.example.smariba_upv.airflow.POJO.SensorObject</a>	
	92
<a href="#">com.example.smariba_upv.airflow.API.MODELS.SensorRequest</a>	
	97
<a href="#">com.example.smariba_upv.airflow.API.MODELS.SensorResponse</a>	
Clase que representa un objeto SensorResponse	103
<a href="#">com.example.smariba_upv.airflow.POJO.TramalBeacon</a>	
Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más	105

<b>com.example.smariba_upv.btle_sento.POJO.TramalBeacon</b>	
Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más	109
<b>com.example.smariba_upv.airflow.POJO.User</b>	113
<b>com.example.smariba_upv.airflow.LOGIC.Utilidades</b>	
Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs	119
<b>com.example.smariba_upv.btle_sento.LOGIC.Utilidades</b>	
Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs	126

## 4. Índice de archivos

### 4.1. Lista de archivos

Lista de todos los archivos con breves descripciones:

<b>AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/ApiService.java</b>	
Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST	133
<b>Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/ApiService.java</b>	
Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST	135
<b>EnviarPeticionesUser.java</b>	136
<b>SensorRequest.java</b>	
Clase que representa un objeto SensorRequest	139
<b>SensorResponse.java</b>	
Clase que representa un objeto SensorResponse	141
<b>AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API/RetrofitClient.java</b>	
Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor	143
<b>Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/API/RetrofitClient.java</b>	
Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor	145
<b>BiometricUtil.java</b>	
@autor: SentoMarcos	146
<b>PeticionesUserUtil.java</b>	148
<b>AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/LOGIC/Utilidades.java</b>	
Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos	149
<b>Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/LOGIC/Utilidades.java</b>	
Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos	151
<b>AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/MainActivity.java</b>	
Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE	154
<b>Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/MainActivity.java</b>	
Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE	160

<a href="#">AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/Medicion.java</a>	
Clase que representa una medición de gas en un lugar determinado	166
<a href="#">Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/Medicion.java</a>	
Clase que representa una medición de gas en un lugar determinado	167
<a href="#">SensorObject.java</a>	170
<a href="#">AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/POJO/TramaIBeacon.java</a>	
Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes	172
<a href="#">Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba_upv/airflow/POJO/TramaIBeacon.java</a>	
Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes	174
<a href="#">User.java</a>	176
<a href="#">EditarPerfilActivity.java</a>	
Clase que permite editar el perfil del usuario	177
<a href="#">HomeFragment.java</a>	179
<a href="#">LandActivity.java</a>	
Clase que permite visualizar la pantalla principal de la aplicación	180
<a href="#">LoginActivity.java</a>	
Clase que permite iniciar sesión en la aplicación	182
<a href="#">MapFragment.java</a>	186
<a href="#">PaginaDeCarga.java</a>	187
<a href="#">PerfilActivity.java</a>	188
<a href="#">PerfilFragment.java</a>	
Clase que permite visualizar el perfil del usuario	190
<a href="#">QRreader.java</a>	
Clase que permite leer un código QR	191
<a href="#">SaludFragment.java</a>	194
<a href="#">SensorAdapter.java</a>	196
<a href="#">SensorFragment.java</a>	198
<a href="#">ArduinoGetterService.java</a>	200

## 5. Documentación de espacios de nombres

### 5.1. Paquete com.example.smariba\_upv.airflow

#### Paquetes

- package API
- package LOGIC
- package POJO
- package PRESENTACION
- package Services

## Clases

- class [MainActivity](#)  
*La clase principal que gestiona la interfaz y las funciones BTLE.*

## 5.2. Paquete com.example.smariba\_upv.airflow.API

### Paquetes

- package [MODELS](#)

## Clases

- interface [ApiService](#)  
*Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.*
- class [EnviarPeticionesUser](#)
- class [RetrofitClient](#)  
*Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.*

## 5.3. Paquete com.example.smariba\_upv.airflow.API.MODELS

### Clases

- class [SensorRequest](#)
- class [SensorResponse](#)  
*Clase que representa un objeto SensorResponse.*

## 5.4. Paquete com.example.smariba\_upv.airflow.LOGIC

### Clases

- class [BiometricUtil](#)  
*Clase que gestiona la autenticación biométrica.*
- class [PeticionesUserUtil](#)
- class [Utilidades](#)  
*Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.*

## 5.5. Paquete com.example.smariba\_upv.airflow.POJO

### Clases

- class [Medicion](#)
- class [SensorObject](#)
- class [TramalBeacon](#)  
*Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.*
- class [User](#)

## 5.6. Paquete com.example.smariba\_upv.airflow.PRESENTACION

### Clases

- class [EditarPerfilActivity](#)  
*Clase que permite editar el perfil del usuario.*
- class [HomeFragment](#)  
*A simple Fragment subclass.*
- class [LandActivity](#)
- class [LogInActivity](#)  
*Clase que permite iniciar sesión en la aplicación.*
- class [MapFragment](#)  
*A simple Fragment subclass.*
- class [PaginaDeCarga](#)
- class [PerfilActivity](#)
- class [PerfilFragment](#)  
*Clase que permite visualizar el perfil del usuario.*
- class [QRReader](#)  
*Clase que permite leer un código QR.*
- class [SaludFragment](#)
- class [SensorAdapter](#)
- class [SensorFragment](#)

## 5.7. Paquete com.example.smariba\_upv.airflow.Services

### Clases

- class [ArduinoGetterService](#)

## 5.8. Paquete com.example.smariba\_upv.btle\_sento

### Paquetes

- package [API](#)
- package [LOGIC](#)
- package [POJO](#)

### Clases

- class [MainActivity](#)  
*La clase principal que gestiona la interfaz y las funciones BTLE.*

## 5.9. Paquete com.example.smariba\_upv.btle\_sento.API

### Clases

- interface [ApiService](#)  
*Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.*
- class [RetrofitClient](#)  
*Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.*

## 5.10. Paquete com.example.smariba\_upv.btle\_sento.LOGIC

### Clases

- class [Utilidades](#)

*Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.*

## 5.11. Paquete com.example.smariba\_upv.btle\_sento.POJO

### Clases

- class [Medicion](#)

*Clase que encapsula los datos de una medición de gas, incluyendo el lugar, el tipo de gas y su valor.*

- class [TramalBeacon](#)

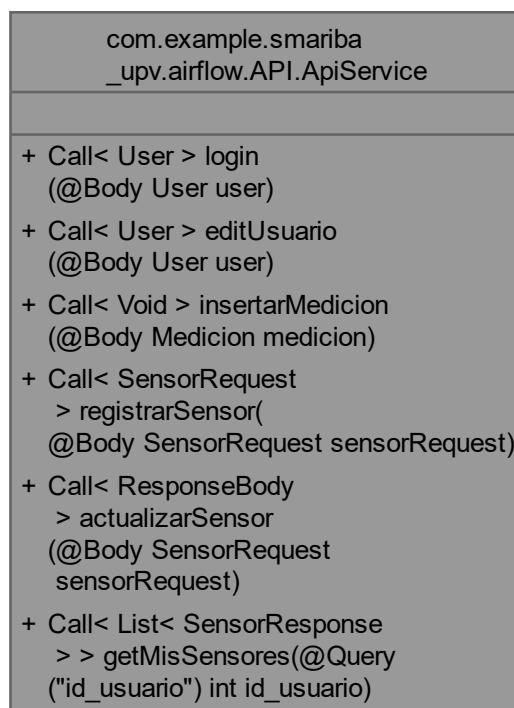
*Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.*

## 6. Documentación de clases

### 6.1. Referencia de la interface com.example.smariba\_upv.airflow.API.ApiService

Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.

Diagrama de colaboración de com.example.smariba\_upv.airflow.API.ApiService:



## Métodos públicos

- Call< User > login (@Body User user)
 

*Método que realiza una petición POST al servidor para iniciar sesión.*
- Call< User > editUsuario (@Body User user)
- Call< Void > insertarMedicion (@Body Medicion medicion)
- Call< SensorRequest > registrarSensor (@Body SensorRequest sensorRequest)
- Call< ResponseBody > actualizarSensor (@Body SensorRequest sensorRequest)
- Call< List< SensorResponse > > getMisSensores (@Query("id\_usuario") int id\_usuario)

### 6.1.1. Descripción detallada

Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.

Definición en la línea 32 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow...

### 6.1.2. Documentación de funciones miembro

#### actualizarSensor()

```
Call< ResponseBody > com.example.smariba_upv.airflow.API.ApiService.actualizarSensor (
    @Body SensorRequest sensorRequest)
```

Gráfico de llamadas a esta función:



#### editUsuario()

```
Call< User > com.example.smariba_upv.airflow.API.ApiService.editUsuario (
    @Body User user)
```

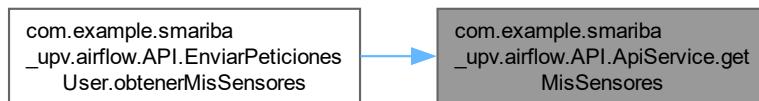
Gráfico de llamadas a esta función:



**getMisSensores()**

```
Call< List< SensorResponse > > com.example.smariba_upv.airflow.API.ApiService.getMisSensores
(
    @Query("id_usuario") int id_usuario)
```

Gráfico de llamadas a esta función:

**insertarMedicion()**

```
Call< Void > com.example.smariba_upv.airflow.API.ApiService.insertarMedicion (
    @Body Medicion medicion)
```

**login()**

```
Call< User > com.example.smariba_upv.airflow.API.ApiService.login (
    @Body User user)
```

Método que realiza una petición POST al servidor para iniciar sesión.

**Parámetros**

in	<i>user</i>	Objeto de tipo User que contiene los datos de inicio de sesión.
----	-------------	---

Gráfico de llamadas a esta función:



## registrarSensor()

```
Call< SensorRequest > com.example.smariba_upv.airflow.API.ApiService.registrarSensor (
    @Body SensorRequest sensorRequest)
```

Gráfico de llamadas a esta función:



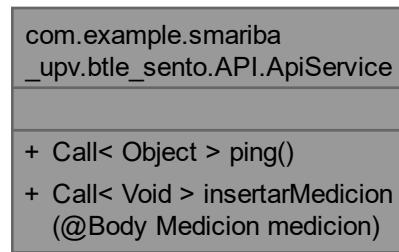
La documentación de esta interfaz está generada del siguiente archivo:

- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/API/ ApiService.java](#)

## 6.2. Referencia de la interface com.example.smariba\_upv.btle\_sento.API.ApiService

Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.

Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.API.ApiService:



### Métodos públicos

- Call< Object > **ping ()**  
*Método que realiza una petición GET al servidor para verificar su disponibilidad.*
- Call< Void > **insertarMedicion (@Body Medicion medicion)**  
*Método que realiza una petición POST para insertar una medición en el servidor.*

### 6.2.1. Descripción detallada

Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.

Definición en la línea 20 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API](#)

### 6.2.2. Documentación de funciones miembro

#### **insertarMedicion()**

```
Call< Void > com.example.smariba_upv.btle_sento.API.ApiService.insertarMedicion ( @Body Medicion medicion)
```

Método que realiza una petición POST para insertar una medición en el servidor.

##### Parámetros

in	<i>medicion</i>	Objeto de tipo Medicion que contiene los datos de la medición a insertar.
----	-----------------	---

##### Devuelve

Retorna un objeto de tipo Call<Void> que representa el resultado de la operación.

#### **ping()**

```
Call< Object > com.example.smariba_upv.btle_sento.API.ApiService.ping ()
```

Método que realiza una petición GET al servidor para verificar su disponibilidad.

##### Devuelve

Retorna un objeto de tipo Call que contiene la respuesta del servidor.

La documentación de esta interface está generada del siguiente archivo:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/ ApiService.java](#)

### 6.3. Referencia de la clase

**com.example.smariba\_upv.airflow.Services.ArduinoGetterService**

Diagrama de herencia de com.example.smariba\_upv.airflow.Services.ArduinoGetterService

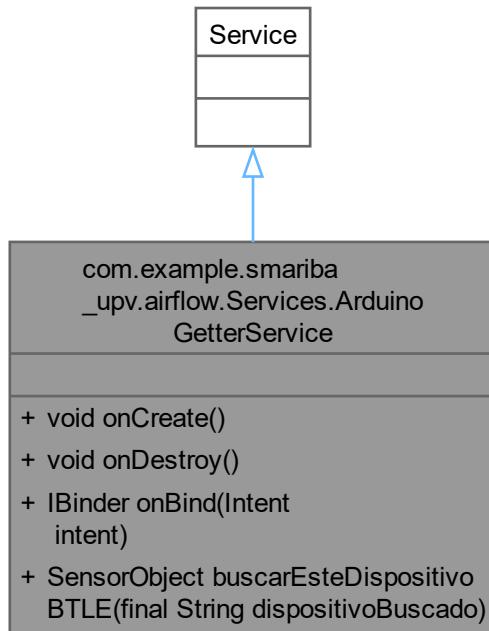
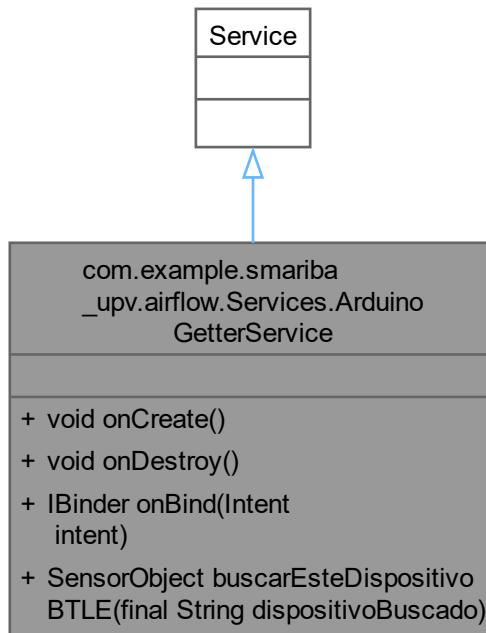


Diagrama de colaboración de com.example.smariba\_upv.airflow.Services.ArduinoGetterService:



## Métodos públicos

- void `onCreate ()`  
    *@function onCreate*
- void `onDestroy ()`  
    *@function onDestroy*
- IBinder `onBind (Intent intent)`  
    *@function onBind*
- SensorObject `buscarEsteDispositivoBTLE (final String dispositivoBuscado)`  
    *@function buscarEsteDispositivoBTLE*

### 6.3.1. Descripción detallada

Definición en la línea 43 del archivo [ArduinoGetterService.java](#).

### 6.3.2. Documentación de funciones miembro

#### `buscarEsteDispositivoBTLE()`

```
SensorObject com.example.smariba_upv.airflow.Services.ArduinoGetterService.buscarEsteDispositivoBTLE (
    final String dispositivoBuscado)
```

@function buscarEsteDispositivoBTLE

Método buscarEsteDispositivoBTLE

Busca un dispositivo BTLE con el nombre proporcionado

Crea un ScanFilter con el nombre del dispositivo

Inicia el escaneo

Si el dispositivo es detectado, se crea un objeto SensorObject y se maneja

Si el dispositivo no es detectado, se envía una notificación de desconexión

#### Parámetros

<i>dispositivoBuscado</i>	Nombre del dispositivo a buscar
---------------------------	---------------------------------

#### Devuelve

SensorObject Texto:dispositivoBuscado => [buscarEsteDispositivoBTLE\(\)](#) => SensorObject

Definición en la línea [257](#) del archivo [ArduinoGetterService.java](#).

Gráfico de llamadas de esta función:

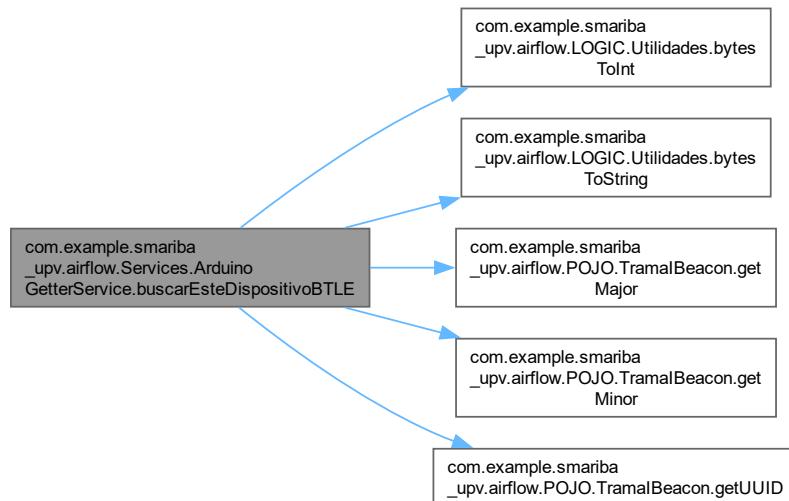
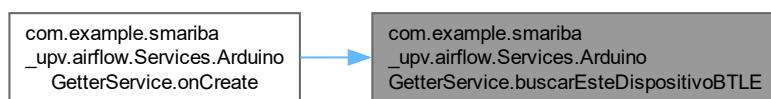


Gráfico de llamadas a esta función:



**onBind()**

```
IBinder com.example.smariba_upv.airflow.Services.ArduinoGetterService.onBind (
    Intent intent)
```

@function onBind

Método onBind

Método onBind

No se utiliza en este servicio Intent intent => onBind() => IBinder:null

Definición en la línea 173 del archivo [ArduinoGetterService.java](#).

**onCreate()**

```
void com.example.smariba_upv.airflow.Services.ArduinoGetterService.onCreate ()
```

@function onCreate

Método onCreate

Llamado cuando el servicio se crea

crea el canal de notificación y llama a startForegroundService

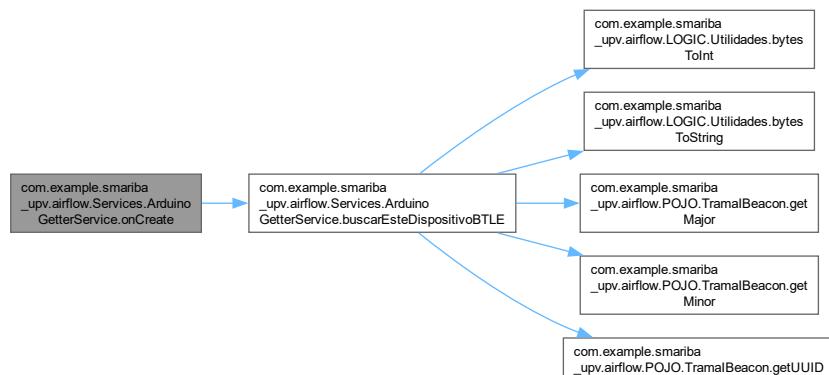
inicializa el bluetooth y busca el dispositivo

Configura el temporizador para las notificaciones

handler.post(temporizador) inicia el temporizado

Definición en la línea 95 del archivo [ArduinoGetterService.java](#).

Gráfico de llamadas de esta función:



**onDestroy()**

```
void com.example.smariba_upv.airflow.Services.ArduinoGetterService.onDestroy ()
```

@function onDestroy

Método onDestroy

Llamado cuando el servicio se destruye

Detiene el temporizador al detener el servicio

Definición en la línea 157 del archivo [ArduinoGetterService.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [ArduinoGetterService.java](#)

#### 6.4. Referencia de la interface

**com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener**

Interfaz para gestionar los eventos de la autenticación biométrica.

Diagrama de herencia de com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener

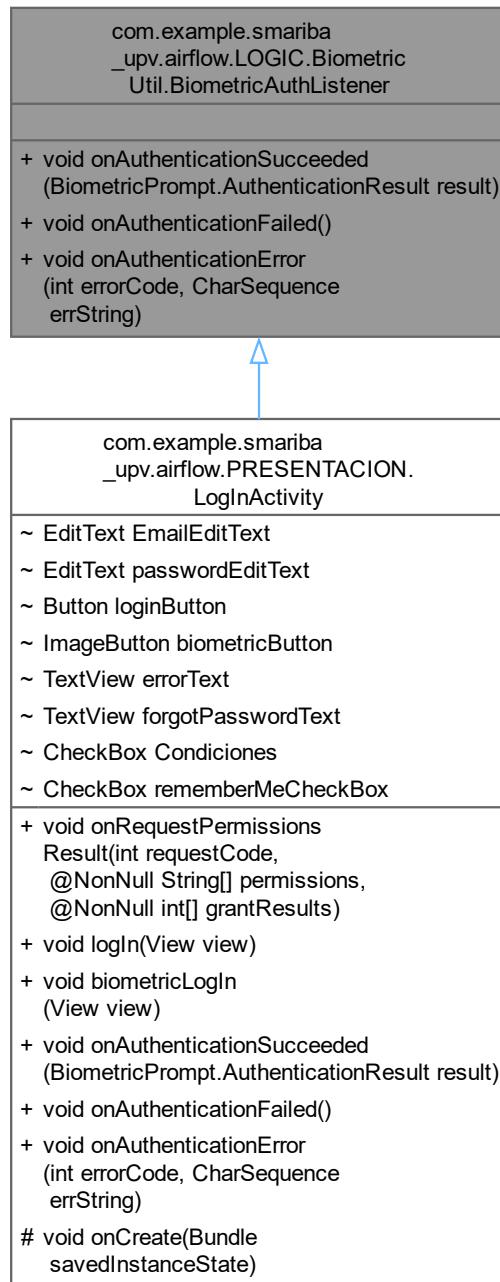
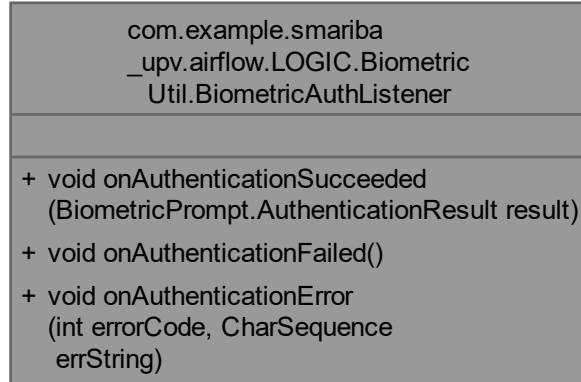


Diagrama de colaboración de com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener:



## Métodos públicos

- `void onAuthenticationSucceeded (BiometricPrompt.AuthenticationResult result)`
- `void onAuthenticationFailed ()`
- `void onAuthenticationError (int errorCode, CharSequence errString)`

### 6.4.1. Descripción detallada

Interfaz para gestionar los eventos de la autenticación biométrica.

Definición en la línea 28 del archivo [BiometricUtil.java](#).

### 6.4.2. Documentación de funciones miembro

#### **onAuthenticationError()**

```
void com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener.onAuthenticationError (
    int errorCode,
    CharSequence errString)
```

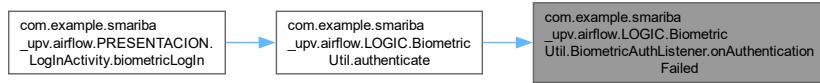
Implementado en [com.example.smariba\\_upv.airflow.PRESENTACION.LogInActivity](#).

**onAuthenticationFailed()**

```
void com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener.onAuthenticationFailed ()
```

Implementado en [com.example.smariba\\_upv.airflow.PRESENTACION.LogInActivity](#).

Gráfico de llamadas a esta función:

**onAuthenticationSucceeded()**

```
void com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener.onAuthenticationSucceeded (
    BiometricPrompt.AuthenticationResult result)
```

Implementado en [com.example.smariba\\_upv.airflow.PRESENTACION.LogInActivity](#).

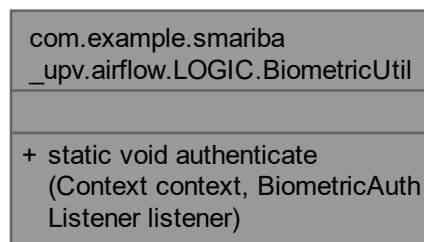
La documentación de esta interfaz está generada del siguiente archivo:

- [BiometricUtil.java](#)

## 6.5. Referencia de la clase com.example.smariba\_upv.airflow.LOGIC.BiometricUtil

Clase que gestiona la autenticación biométrica.

Diagrama de colaboración de com.example.smariba\_upv.airflow.LOGIC.BiometricUtil:



## Clases

- interface [BiometricAuthListener](#)

*Interfaz para gestionar los eventos de la autenticación biométrica.*

## Métodos públicos estáticos

- static void [authenticate](#) (Context context, [BiometricAuthListener](#) listener)  
`@func authenticate`

### 6.5.1. Descripción detallada

Clase que gestiona la autenticación biométrica.

Definición en la línea 22 del archivo [BiometricUtil.java](#).

### 6.5.2. Documentación de funciones miembro

#### **authenticate()**

```
static void com.example.smariba_upv.airflow.LOGIC.BiometricUtil.authenticate (
    Context context,
    BiometricAuthListener listener) [static]
```

`@func authenticate`

Método para autenticar al usuario mediante biometría

#### Parámetros

<code>context</code>	Contexto de la actividad
<code>listener</code>	Listener para gestionar los eventos de la autenticación

`@func authenticate` Configurar la información de la autenticación

Definición en la línea 40 del archivo [BiometricUtil.java](#).

Gráfico de llamadas de esta función:

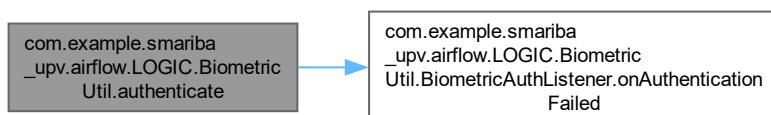
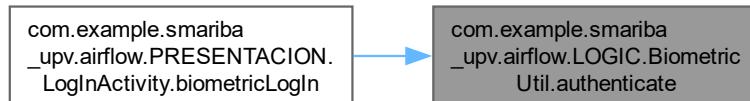


Gráfico de llamadas a esta función:



La documentación de esta clase está generada del siguiente archivo:

- [BiometricUtil.java](#)

## 6.6. Referencia de la clase com.example.smariba\_upv.airflow.PRESENTACION.EditarPerfilActivity

Clase que permite editar el perfil del usuario.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.EditarPerfilActivity

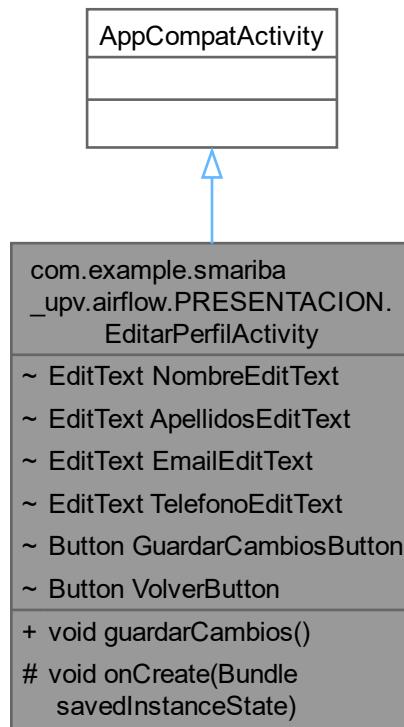
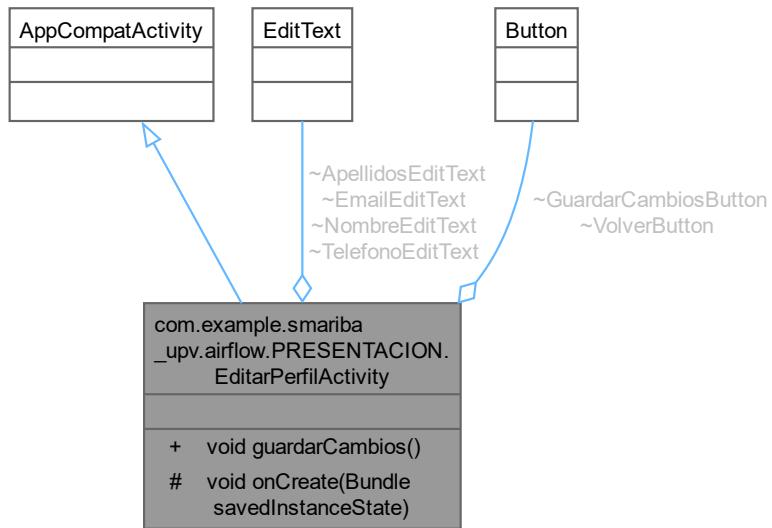


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.EditarPerfilActivity:



## Métodos públicos

- `void guardarCambios ()`  
`@function guardarCambios`

## Métodos protegidos

- `void onCreate (Bundle savedInstanceState)`  
`@function onCreate`

### 6.6.1. Descripción detallada

Clase que permite editar el perfil del usuario.

Clase que permite editar el perfil del usuario y guardar los cambios realizados

Definición en la línea 26 del archivo [EditarPerfilActivity.java](#).

### 6.6.2. Documentación de funciones miembro

#### guardarCambios()

```
void com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity.guardarCambios ()
```

@function guardarCambios

Método que permite guardar los cambios realizados en el perfil del usuario

Método que permite guardar los cambios realizados en el perfil del usuario y enviar los datos a la base de datos

Definición en la línea 77 del archivo [EditarPerfilActivity.java](#).

Gráfico de llamadas de esta función:

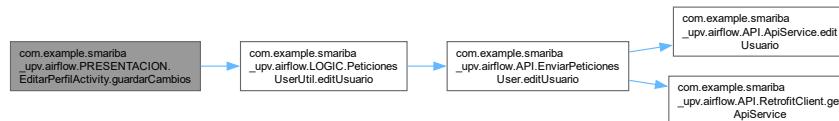


Gráfico de llamadas a esta función:



#### onCreate()

```
void com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

@function onCreate

Parámetros

<code>savedInstanceState</code>	<input type="text"/>
---------------------------------	----------------------

Definición en la línea 48 del archivo [EditarPerfilActivity.java](#).

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- [EditarPerfilActivity.java](#)

## 6.7. Referencia de la clase com.example.smariba\_upv.airflow.APIEnviarPeticionesUser

Diagrama de colaboración de com.example.smariba\_upv.airflow.APIEnviarPeticionesUser:

com.example.smariba_upv.airflow.APIEnviarPeticionesUser
<ul style="list-style-type: none"> <li>+ EnviarPeticionesUser (Context context)</li> <li>+ EnviarPeticionesUser()</li> <li>+ boolean login(String email, String password)</li> <li>+ void editUsuario(int id, String nombre, String apellidos, String email, String telefono)</li> <li>+ void registrarSensor (int id_usuario, SensorObject sensorObject)</li> <li>+ void actualizarSensor (int id_sensor, String estado, boolean conexion, int bateria)</li> <li>+ void obtenerMisSensores (Context context)</li> </ul>

### Métodos públicos

- [EnviarPeticionesUser](#) (Context context)
 

*Constructor de la clase.*
- [EnviarPeticionesUser](#) ()
 

*Constructor de la clase.*
- boolean [login](#) (String email, String password)
 

*Método para registrar un usuario.*
- void [editUsuario](#) (int id, String nombre, String apellidos, String email, String telefono)
 

*Método para editar un usuario.*
- void [registrarSensor](#) (int id\_usuario, SensorObject sensorObject)
 

*Método para registrar un sensor.*
- void [actualizarSensor](#) (int id\_sensor, String estado, boolean conexion, int bateria)
 

*Método para actualizar un sensor.*
- void [obtenerMisSensores](#) (Context context)
 

*Método para obtener los sensores de un usuario.*

### 6.7.1. Descripción detallada

Definición en la línea 25 del archivo [EnviarPeticionesUser.java](#).

### 6.7.2. Documentación de constructores y destructores

#### EnviarPeticionesUser() [1/2]

```
com.example.smariba_upv.airflow.APIEnviarPeticionesUser.EnviarPeticionesUser (
    Context context)
```

Constructor de la clase.

##### Parámetros

<i>context</i>	Contexto de la aplicación Context:context => <a href="#">EnviarPeticionesUser()</a>
----------------	---

Definición en la línea 39 del archivo [EnviarPeticionesUser.java](#).

#### EnviarPeticionesUser() [2/2]

```
com.example.smariba_upv.airflow.APIEnviarPeticionesUser.EnviarPeticionesUser ()
```

Constructor de la clase.

Definición en la línea 46 del archivo [EnviarPeticionesUser.java](#).

### 6.7.3. Documentación de funciones miembro

#### actualizarSensor()

```
void com.example.smariba_upv.airflow.APIEnviarPeticionesUser.actualizarSensor (
    int id_sensor,
    String estado,
    boolean conexion,
    int bateria)
```

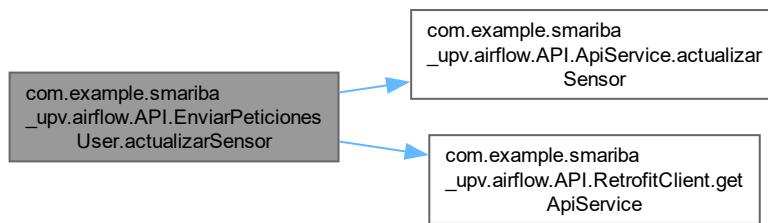
Método para actualizar un sensor.

##### Parámetros

<i>id_sensor</i>	ID del sensor
<i>estado</i>	Estado del sensor
<i>conexion</i>	Conexión del sensor
<i>bateria</i>	Batería del sensor N:id_sensor, Texto:estado, VoF:conexion, N:bateria => <a href="#">actualizarSensor()</a>

Definición en la línea 168 del archivo [EnviarPeticionesUser.java](#).

Gráfico de llamadas de esta función:



### `editUsuario()`

```

void com.example.smariba_upv.airflow.API.EnviarPeticionesUser.editUsuario (
    int id,
    String nombre,
    String apellidos,
    String email,
    String telefono)
  
```

Método para editar un usuario.

#### Parámetros

<i>id</i>	ID del usuario
<i>nombre</i>	Nombre del usuario
<i>apellidos</i>	Apellidos del usuario
<i>email</i>	Correo electrónico del usuario
<i>telefono</i>	Teléfono del usuario N:id, Texto:nombre, Texto:apellidos, Texto:email, Texto:telefono => registrarUsuario()

Definición en la línea 110 del archivo [EnviarPeticionesUser.java](#).

Gráfico de llamadas de esta función:

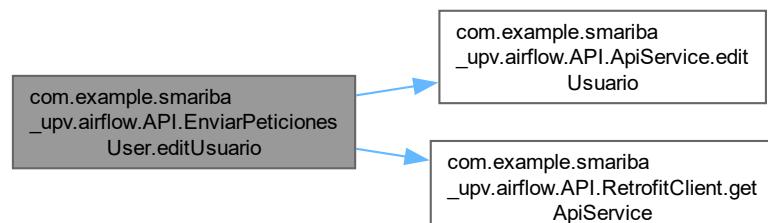


Gráfico de llamadas a esta función:



### login()

```
boolean com.example.smariba_upv.airflow.API.EnviarPeticionesUser.login (
    String email,
    String password)
```

Método para registrar un usuario.

#### Parámetros

<i>email</i>	Correo electrónico del usuario
<i>password</i>	Contraseña del usuario Texto:email, Texto:password => <a href="#">login()</a> => VoF

Definición en la línea 55 del archivo [EnviarPeticionesUser.java](#).

Gráfico de llamadas de esta función:

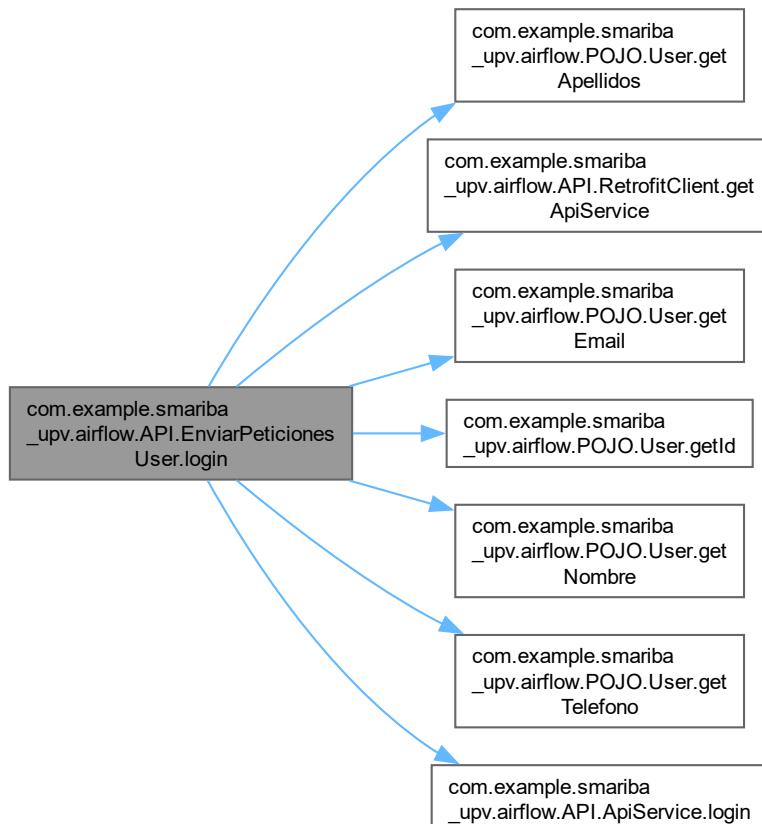
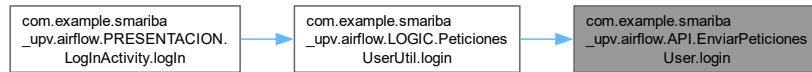


Gráfico de llamadas a esta función:



### obtenerMisSensores()

```
void com.example.smariba_upv.airflow.API.EnviarPeticionesUser.obtenerMisSensores (
    Context context)
```

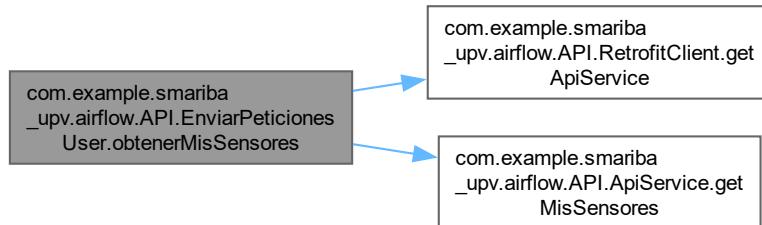
Método para obtener los sensores de un usuario.

#### Parámetros

<code>context</code>	Contexto de la aplicación Context:context => <a href="#">obtenerMisSensores()</a>
----------------------	---

Definición en la línea 192 del archivo [EnviarPeticionesUser.java](#).

Gráfico de llamadas de esta función:



### registrarSensor()

```
void com.example.smariba_upv.airflow.API.EnviarPeticionesUser.registrarSensor (
    int id_usuario,
    SensorObject sensorObject)
```

Método para registrar un sensor.

#### Parámetros

<code>id_usuario</code>	ID del usuario
<code>sensorObject</code>	Objeto del sensor N:id_usuario, Objeto:sensorObject => <a href="#">registrarSensor()</a>

Definición en la línea 138 del archivo [EnviarPeticionesUser.java](#).

Gráfico de llamadas de esta función:

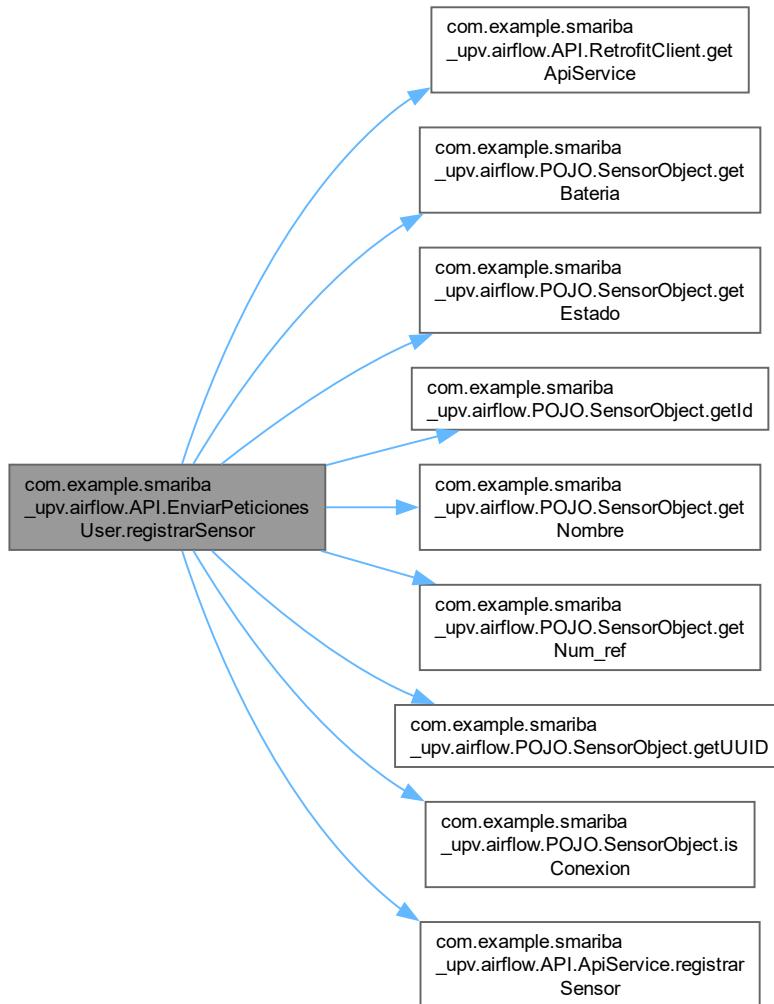


Gráfico de llamadas a esta función:



La documentación de esta clase está generada del siguiente archivo:

- [EnviarPeticionesUser.java](#)

## 6.8. Referencia de la clase

**com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment**

A simple Fragment subclass.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment

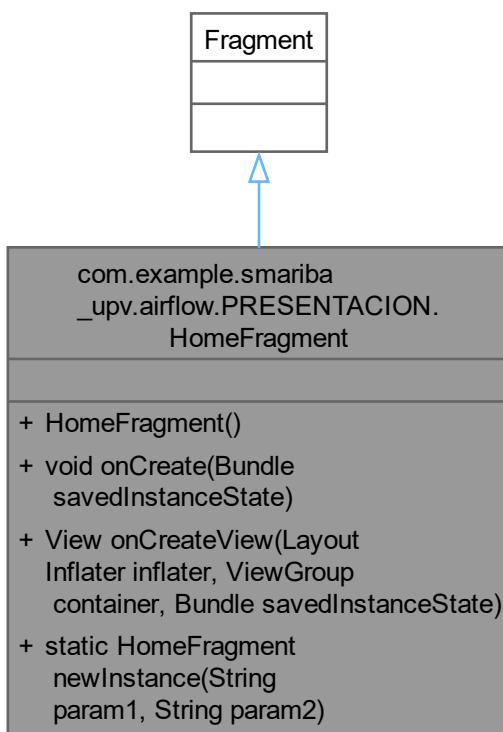
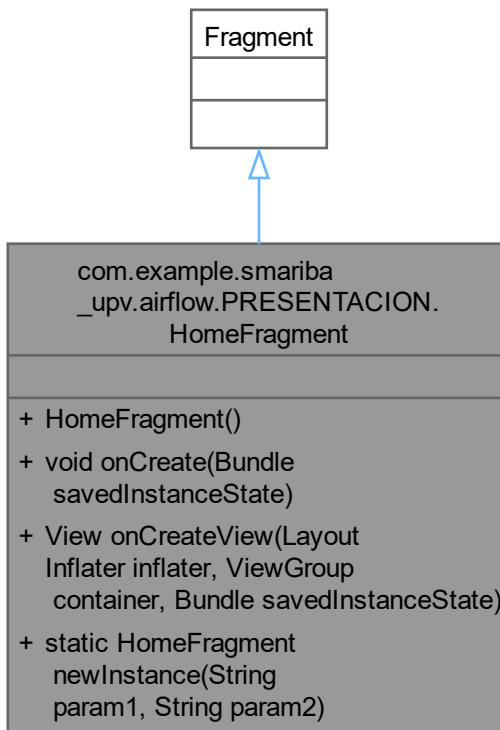


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment:



## Métodos públicos

- [HomeFragment \(\)](#)
- [void onCreate \(Bundle savedInstanceState\)](#)
- [View onCreateView \(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState\)](#)

## Métodos públicos estáticos

- [static HomeFragment newInstance \(String param1, String param2\)](#)  
*Use this factory method to create a new instance of this fragment using the provided parameters.*

### 6.8.1. Descripción detallada

A simple `Fragment` subclass.

Use the [HomeFragment#newInstance](#) factory method to create an instance of this fragment.

Definición en la línea 19 del archivo [HomeFragment.java](#).

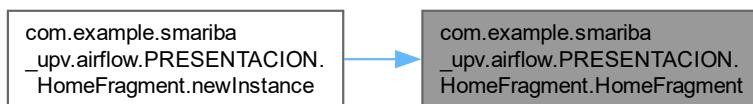
## 6.8.2. Documentación de constructores y destructores

### **HomeFragment()**

```
com.example.smariba_upv.airflow.PRESENTACION.HomeFragment.HomeFragment ()
```

Definición en la línea 30 del archivo [HomeFragment.java](#).

Gráfico de llamadas a esta función:



## 6.8.3. Documentación de funciones miembro

### **newInstance()**

```
static HomeFragment com.example.smariba_upv.airflow.PRESENTACION.HomeFragment.newInstance (
    String param1,
    String param2) [static]
```

Use this factory method to create a new instance of this fragment using the provided parameters.

#### Parámetros

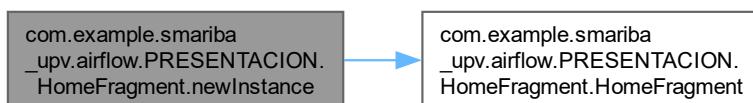
<i>param1</i>	Parameter 1.
<i>param2</i>	Parameter 2.

#### Devuelve

A new instance of fragment [HomeFragment](#).

Definición en la línea 43 del archivo [HomeFragment.java](#).

Gráfico de llamadas de esta función:



**onCreate()**

```
void com.example.smariba_upv.airflow.PRESENTACION.HomeFragment.onCreate (
    Bundle savedInstanceState)
```

Definición en la línea 53 del archivo [HomeFragment.java](#).

**onCreateView()**

```
View com.example.smariba_upv.airflow.PRESENTACION.HomeFragment.onCreateView (
    LayoutInflater inflater,
    ViewGroup container,
    Bundle savedInstanceState)
```

Definición en la línea 62 del archivo [HomeFragment.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [HomeFragment.java](#)

**6.9. Referencia de la clase****com.example.smariba\_upv.airflow.PRESENTACION.LandActivity**

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.LandActivity

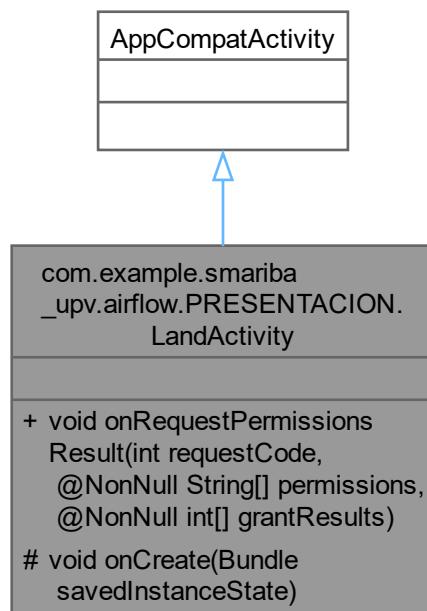
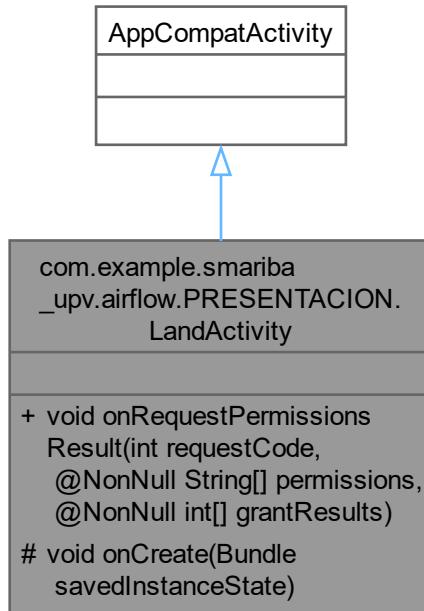


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.LandActivity:



## Métodos públicos

- void `onRequestPermissionsResult` (int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults)  
`@function onRequestPermissionsResult`

## Métodos protegidos

- void `onCreate` (Bundle savedInstanceState)  
`@function onCreate`

### 6.9.1. Descripción detallada

Definición en la línea 36 del archivo [LandActivity.java](#).

### 6.9.2. Documentación de funciones miembro

#### `onCreate()`

```
void com.example.smariba_upv.airflow.PRESENTACION.LandActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

`@function onCreate`

Método que se ejecuta al crear la actividad

**Parámetros**

<i>savedInstanceState</i>	<input type="text"/>
---------------------------	----------------------

Definición en la línea 50 del archivo [LandActivity.java](#).

**onRequestPermissionsResult()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LandActivity.onRequestPermissionsResult (int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults)
```

@function onRequestPermissionsResult

**Parámetros**

<i>requestCode</i>	<input type="text"/>
<i>permissions</i>	<input type="text"/>
<i>grantResults</i>	<input type="text"/>

Método que se ejecuta al recibir

Definición en la línea 82 del archivo [LandActivity.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [LandActivity.java](#)

## 6.10. Referencia de la clase com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity

Clase que permite iniciar sesión en la aplicación.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity

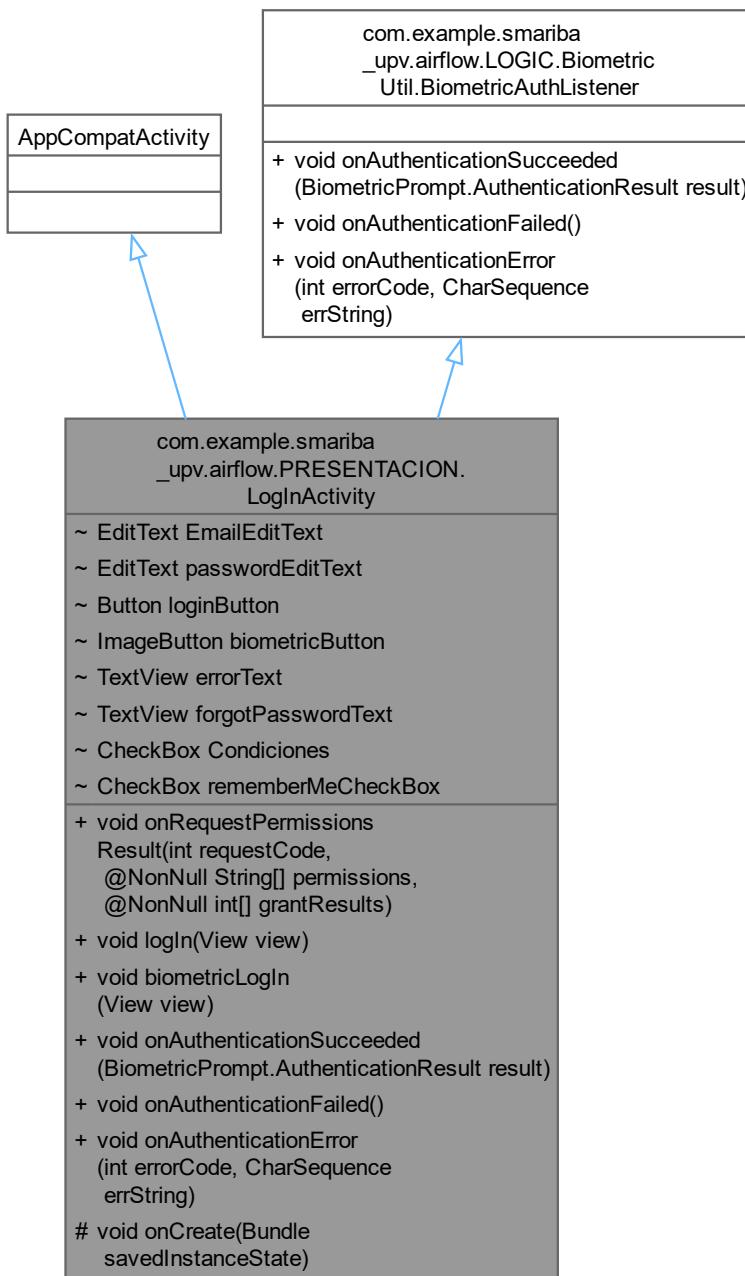
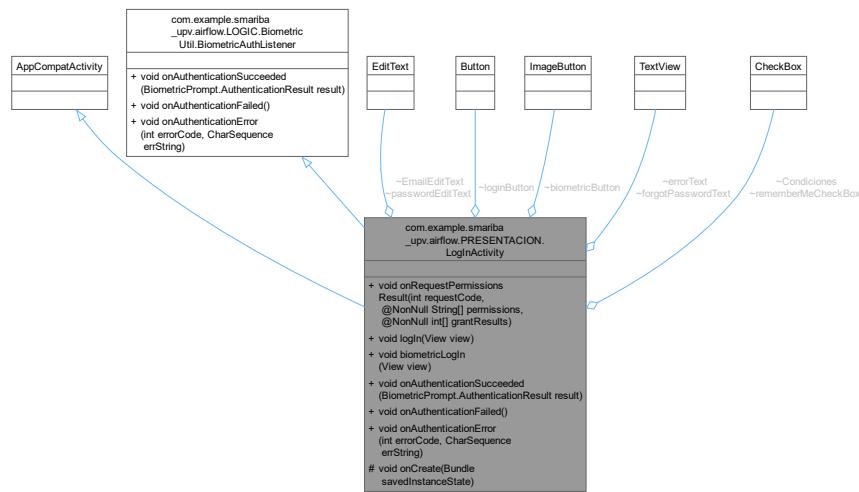


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.LoginActivity:



## Métodos públicos

- `void onRequestPermissionsResult (int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults)`  
`@function onRequestPermissionsResult`
- `void logIn (View view)`  
`@function logIn`
- `void biometricLogin (View view)`  
`@function biometricLogin`
- `void onAuthenticationSucceeded (BiometricPrompt.AuthenticationResult result)`  
`@function onAuthenticationSucceeded`
- `void onAuthenticationFailed ()`  
`@function onAuthenticationFailed`
- `void onAuthenticationError (int errorCode, CharSequence errString)`  
`@function onAuthenticationError`

## Métodos públicos heredados de

`com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener`

## Métodos protegidos

- `void onCreate (Bundle savedInstanceState)`  
`@function onCreate`

### 6.10.1. Descripción detallada

Clase que permite iniciar sesión en la aplicación.

Clase que permite iniciar sesión en la aplicación mediante correo electrónico y contraseña o mediante autenticación biométrica

Definición en la línea 41 del archivo [LoginActivity.java](#).

### 6.10.2. Documentación de funciones miembro

#### **biometricLogin()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.biometricLogin (
    View view)
```

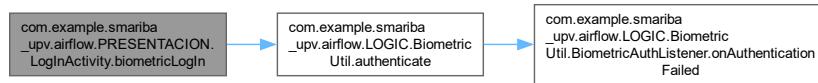
@function biometricLogin

##### Parámetros

view	<input type="text"/>
------	----------------------

Definición en la línea 248 del archivo [LogInActivity.java](#).

Gráfico de llamadas de esta función:



#### **logIn()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.logIn (
    View view)
```

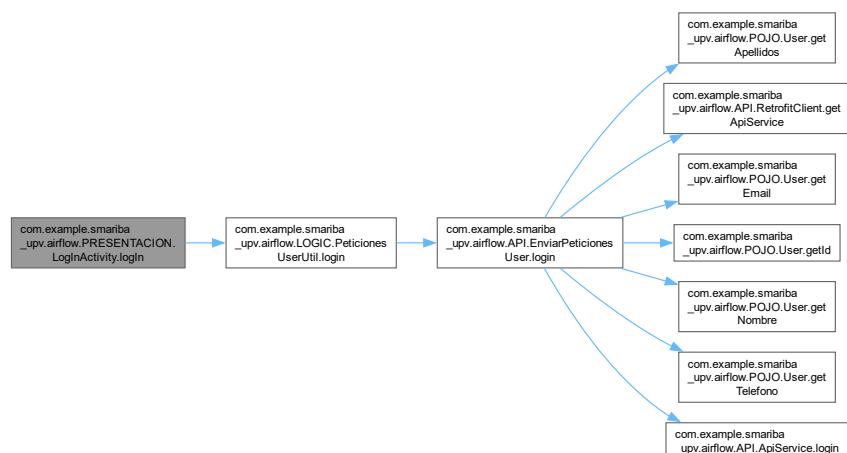
@function logIn

##### Parámetros

view	<input type="text"/>
------	----------------------

Definición en la línea 216 del archivo [LogInActivity.java](#).

Gráfico de llamadas de esta función:



**onAuthenticationError()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.onAuthenticationError (
    int errorCode,
    CharSequence errString)
```

@function onAuthenticationError

**Parámetros**

<i>errorCode</i>	
<i>errString</i>	

Implementa [com.example.smariba\\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener](#).

Definición en la línea 277 del archivo [LogInActivity.java](#).

**onAuthenticationFailed()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.onAuthenticationFailed ()
```

@function onAuthenticationFailed

Implementa [com.example.smariba\\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener](#).

Definición en la línea 266 del archivo [LogInActivity.java](#).

**onAuthenticationSucceeded()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.onAuthenticationSucceeded (
    BiometricPrompt.AuthenticationResult result)
```

@function onAuthenticationSucceeded

**Parámetros**

<i>result</i>	
---------------	--

Implementa [com.example.smariba\\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener](#).

Definición en la línea 257 del archivo [LogInActivity.java](#).

**onCreate()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

@function onCreate

Método que se ejecuta al crear la actividad

**Parámetros**

<i>savedInstanceState</i>	<input type="text"/>
---------------------------	----------------------

Definición en la línea 70 del archivo [LogInActivity.java](#).

**onRequestPermissionsResult()**

```
void com.example.smariba_upv.airflow.PRESENTACION.LogInActivity.onRequestPermissionsResult (   
    int requestCode,  
    @NonNull String[] permissions,  
    @NonNull int[] grantResults)
```

@function onRequestPermissionsResult

Método que se ejecuta al recibir

Método que se ejecuta al recibir

**Parámetros**

<i>requestCode</i>	<input type="text"/>
<i>permissions</i>	<input type="text"/>
<i>grantResults</i>	<input type="text"/>

Definición en la línea 134 del archivo [LogInActivity.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [LogInActivity.java](#)

## 6.11. Referencia de la clase com.example.smariba\_upv.airflow.MainActivity

La clase principal que gestiona la interfaz y las funciones BTLE.

Diagrama de herencia de com.example.smariba\_upv.airflow.MainActivity

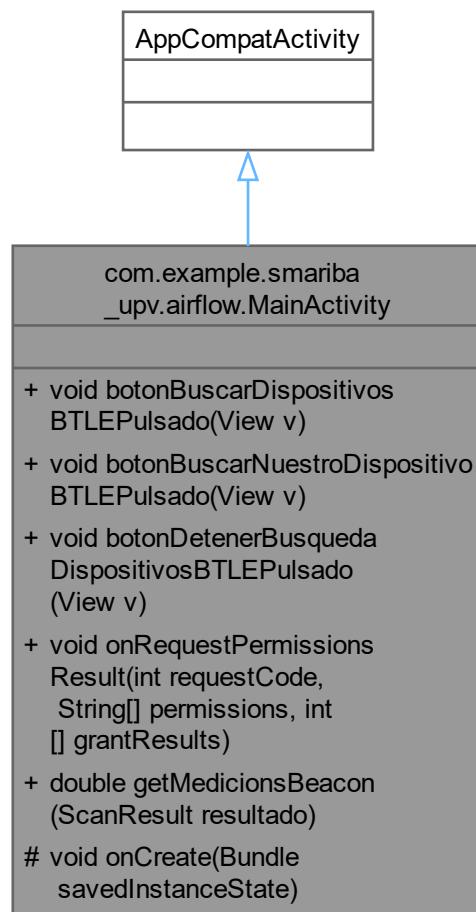
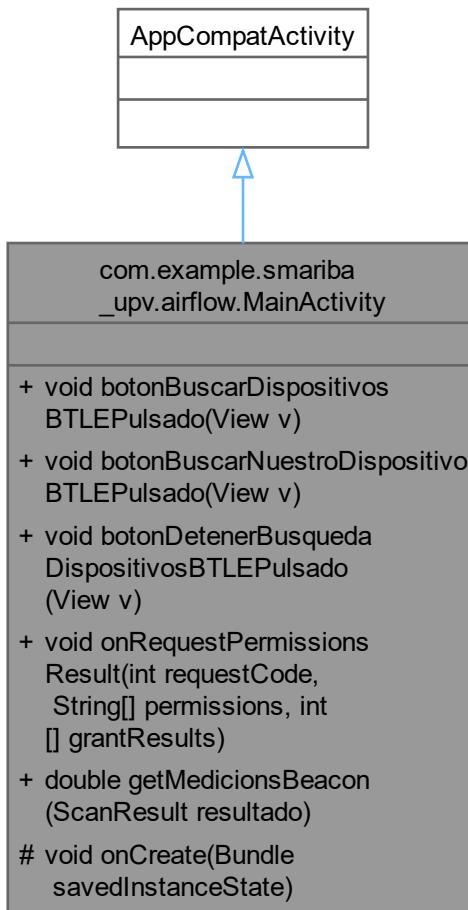


Diagrama de colaboración de com.example.smariba\_upv.airflow.MainActivity:



## Métodos públicos

- void **botonBuscarDispositivosBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para buscar dispositivos BTLE.*
- void **botonBuscarNuestroDispositivoBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para buscar un dispositivo BTLE específico.*
- void **botonDetenerBusquedaDispositivosBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para detener la búsqueda de dispositivos BTLE.*
- void **onRequestPermissionsResult** (int requestCode, String[] permissions, int[] grantResults)  
*Método que se ejecuta al solicitar permisos.*
- double **getMedicionesBeacon** (ScanResult resultado)  
*Método que obtiene el valor de la medición de un beacon.*

## Métodos protegidos

- void **onCreate** (Bundle savedInstanceState)  
*Método que se ejecuta al crear la actividad.*

### 6.11.1. Descripción detallada

La clase principal que gestiona la interfaz y las funciones BTLE.

Definición en la línea 43 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

### 6.11.2. Documentación de funciones miembro

#### **botonBuscarDispositivosBTLEPulsado()**

```
void com.example.smariba_upv.airflow.MainActivity.botonBuscarDispositivosBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para buscar dispositivos BTLE.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 273 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

#### **botonBuscarNuestroDispositivoBTLEPulsado()**

```
void com.example.smariba_upv.airflow.MainActivity.botonBuscarNuestroDispositivoBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para buscar un dispositivo BTLE específico.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 284 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

#### **botonDetenerBusquedaDispositivosBTLEPulsado()**

```
void com.example.smariba_upv.airflow.MainActivity.botonDetenerBusquedaDispositivosBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para detener la búsqueda de dispositivos BTLE.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 299 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

#### **getMedicionesBeacon()**

```
double com.example.smariba_upv.airflow.MainActivity.getMedicionesBeacon (
    ScanResult resultado)
```

Método que obtiene el valor de la medición de un beacon.

### Parámetros

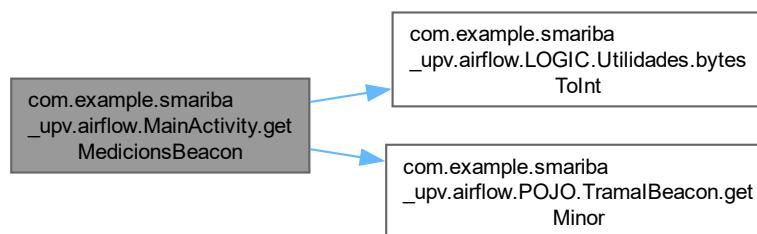
in	<i>resultado</i>	Objeto ScanResult con la información del dispositivo detectado.
----	------------------	---

### Devuelve

Valor de la medición.

Definición en la línea 462 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

Gráfico de llamadas de esta función:



### onCreate()

```
void com.example.smariba_upv.airflow.MainActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

Método que se ejecuta al crear la actividad.

### Parámetros

in	<i>savedInstanceState</i>	Estado de la instancia guardada.
----	---------------------------	----------------------------------

Definición en la línea 397 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

### onRequestPermissionsResult()

```
void com.example.smariba_upv.airflow.MainActivity.onRequestPermissionsResult (
    int requestCode,
    String[] permissions,
    int[] grantResults)
```

Método que se ejecuta al solicitar permisos.

#### Parámetros

in	<i>requestCode</i>	Código de solicitud.
in	<i>permissions</i>	Permisos solicitados.
in	<i>grantResults</i>	Resultados de los permisos.

Definición en la línea 418 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

La documentación de esta clase está generada del siguiente archivo:

- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

## 6.12. Referencia de la clase com.example.smariba\_upv.btle\_sento.MainActivity

La clase principal que gestiona la interfaz y las funciones BTLE.

Diagrama de herencia de com.example.smariba\_upv.btle\_sento.MainActivity

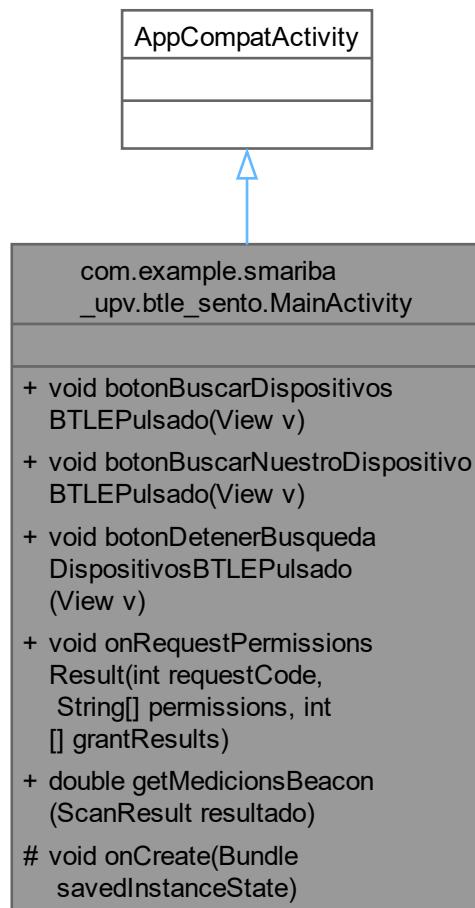
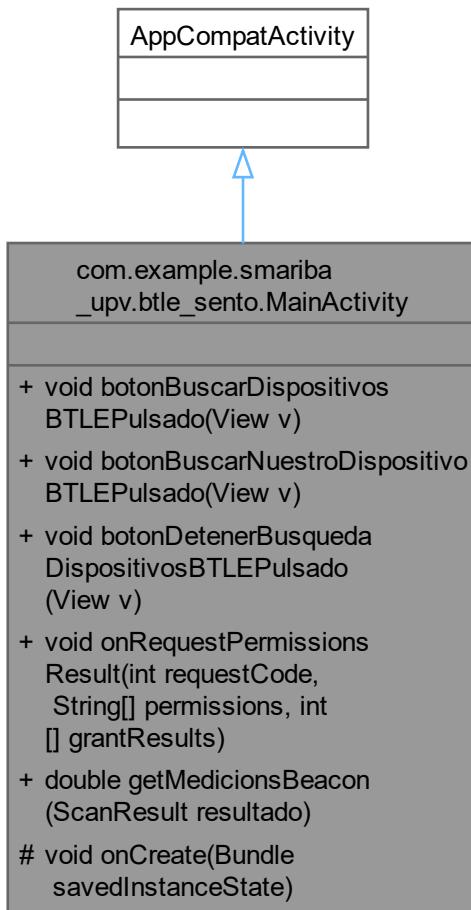


Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.MainActivity:



## Métodos públicos

- void **botonBuscarDispositivosBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para buscar dispositivos BTLE.*
- void **botonBuscarNuestroDispositivoBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para buscar un dispositivo BTLE específico.*
- void **botonDetenerBusquedaDispositivosBTLEPulsado** (View v)  
*Método que se ejecuta al pulsar el botón para detener la búsqueda de dispositivos BTLE.*
- void **onRequestPermissionsResult** (int requestCode, String[] permissions, int[] grantResults)  
*Método que se ejecuta al solicitar permisos.*
- double **getMedicionesBeacon** (ScanResult resultado)  
*Método que obtiene el valor de la medición de un beacon.*

## Métodos protegidos

- void **onCreate** (Bundle savedInstanceState)  
*Método que se ejecuta al crear la actividad.*

### 6.12.1. Descripción detallada

La clase principal que gestiona la interfaz y las funciones BTLE.

Definición en la línea 43 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

### 6.12.2. Documentación de funciones miembro

#### botonBuscarDispositivosBTLEPulsado()

```
void com.example.smariba_upv.btle_sento.MainActivity.botonBuscarDispositivosBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para buscar dispositivos BTLE.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 271 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

#### botonBuscarNuestroDispositivoBTLEPulsado()

```
void com.example.smariba_upv.btle_sento.MainActivity.botonBuscarNuestroDispositivoBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para buscar un dispositivo BTLE específico.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 282 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

#### botonDetenerBusquedaDispositivosBTLEPulsado()

```
void com.example.smariba_upv.btle_sento.MainActivity.botonDetenerBusquedaDispositivosBTLEPulsado (
    View v)
```

Método que se ejecuta al pulsar el botón para detener la búsqueda de dispositivos BTLE.

##### Parámetros

	v	the v
in	v	View asociada al botón.

Definición en la línea 297 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

#### getMedicionesBeacon()

```
double com.example.smariba_upv.btle_sento.MainActivity.getMedicionesBeacon (
    ScanResult resultado)
```

Método que obtiene el valor de la medición de un beacon.

## Parámetros

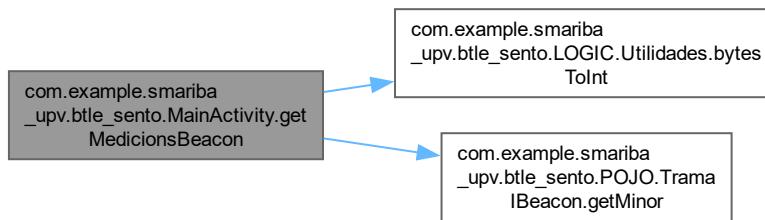
in	<i>resultado</i>	Objeto ScanResult con la información del dispositivo detectado.
----	------------------	---

## Devuelve

Valor de la medición.

Definición en la línea 460 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

Gráfico de llamadas de esta función:



## onCreate()

```
void com.example.smariba_upv.btle_sento.MainActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

Método que se ejecuta al crear la actividad.

## Parámetros

in	<i>savedInstanceState</i>	Estado de la instancia guardada.
----	---------------------------	----------------------------------

Definición en la línea 395 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

## onRequestPermissionsResult()

```
void com.example.smariba_upv.btle_sento.MainActivity.onRequestPermissionsResult (
    int requestCode,
    String[] permissions,
    int[] grantResults)
```

Método que se ejecuta al solicitar permisos.

**Parámetros**

in	<i>requestCode</i>	Código de solicitud.
in	<i>permissions</i>	Permisos solicitados.
in	<i>grantResults</i>	Resultados de los permisos.

Definición en la línea 416 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

La documentación de esta clase está generada de los siguientes archivos:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)
- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

**6.13. Referencia de la clase****com.example.smariba\_upv.airflow.PRESENTACION.MapFragment**

A simple Fragment subclass.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.MapFragment

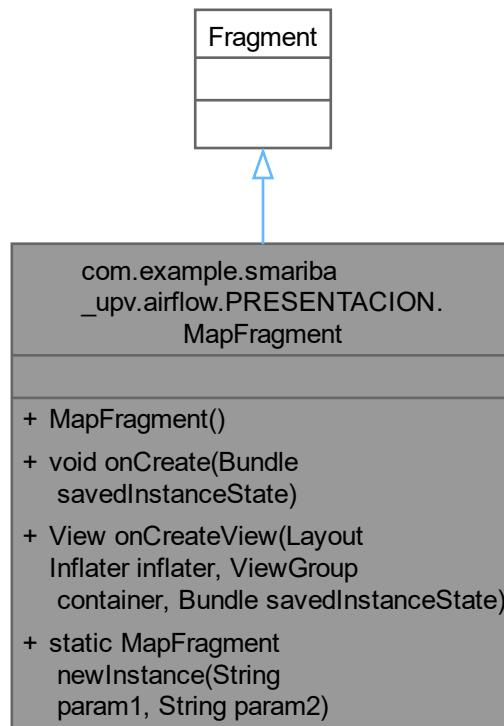
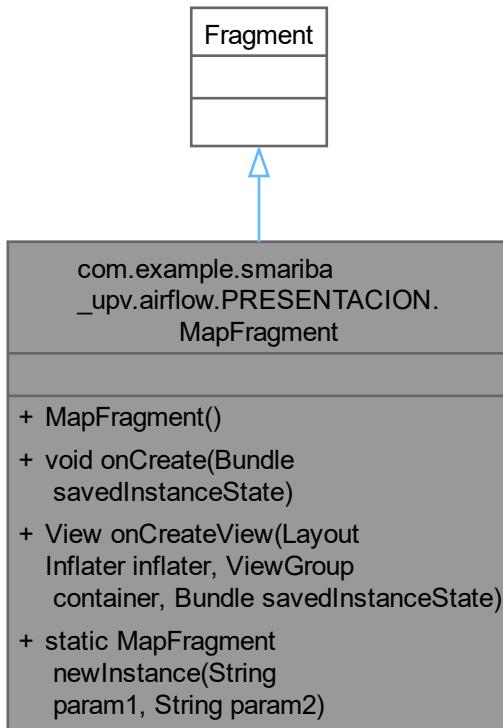


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.MapFragment:



### Métodos públicos

- [MapFragment \(\)](#)
- [void onCreate \(Bundle savedInstanceState\)](#)
- [View onCreateView \(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState\)](#)

### Métodos públicos estáticos

- [static MapFragment newInstance \(String param1, String param2\)](#)  
*Use this factory method to create a new instance of this fragment using the provided parameters.*

#### 6.13.1. Descripción detallada

A simple `Fragment` subclass.

Use the [MapFragment#newInstance](#) factory method to create an instance of this fragment.

Definición en la línea 18 del archivo [MapFragment.java](#).

### 6.13.2. Documentación de constructores y destructores

#### MapFragment()

```
com.example.smariba_upv.airflow.PRESENTACION.MapFragment ()
```

Definición en la línea 29 del archivo [MapFragment.java](#).

Gráfico de llamadas a esta función:



### 6.13.3. Documentación de funciones miembro

#### newInstance()

```
static MapFragment com.example.smariba_upv.airflow.PRESENTACION.MapFragment.newInstance (
    String param1,
    String param2) [static]
```

Use this factory method to create a new instance of this fragment using the provided parameters.

##### Parámetros

<i>param1</i>	Parameter 1.
<i>param2</i>	Parameter 2.

##### Devuelve

A new instance of fragment [MapFragment](#).

Definición en la línea 42 del archivo [MapFragment.java](#).

Gráfico de llamadas de esta función:



**onCreate()**

```
void com.example.smariba_upv.airflow.PRESENTACION.MapFragment.onCreate ( 
    Bundle savedInstanceState)
```

Definición en la línea [52](#) del archivo [MapFragment.java](#).

**onCreateView()**

```
View com.example.smariba_upv.airflow.PRESENTACION.MapFragment.onCreateView ( 
    LayoutInflater inflater,
    ViewGroup container,
    Bundle savedInstanceState)
```

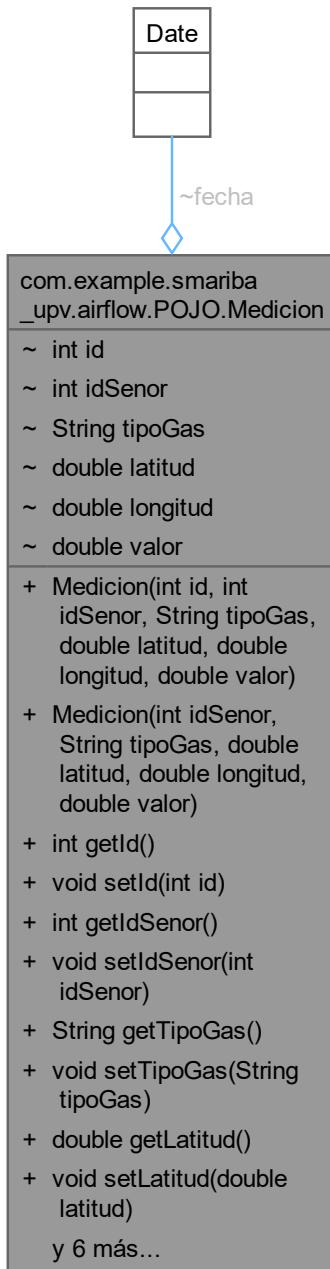
Definición en la línea [61](#) del archivo [MapFragment.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [MapFragment.java](#)

## 6.14. Referencia de la clase com.example.smariba\_upv.airflow.POJO.Medicion

Diagrama de colaboración de com.example.smariba\_upv.airflow.POJO.Medicion:



### Métodos públicos

- [Medicion](#) (int id, int idSenor, String tipoGas, double latitud, double longitud, double valor)
- [Medicion](#) (int idSenor, String tipoGas, double latitud, double longitud, double valor)
- int [getId \(\)](#)

- void **setId** (int id)
- int **getIdSenor** ()
- void **setIdSenor** (int idSenor)
- String **getTipoGas** ()
- void **setTipoGas** (String tipoGas)
- double **getLatitud** ()
- void **setLatitud** (double latitud)
- double **getLongitud** ()
- void **setLongitud** (double longitud)
- double **getValor** ()
- void **setValor** (double valor)
- Date **getFecha** ()
- void **setFecha** (Date fecha)

#### 6.14.1. Descripción detallada

Definición en la línea 13 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflo

#### 6.14.2. Documentación de constructores y destructores

##### Medicion() [1/2]

```
com.example.smariba_upv.airflow.POJO.Medicion.Medicion (
    int id,
    int idSenor,
    String tipoGas,
    double latitud,
    double longitud,
    double valor)
```

Definición en la línea 19 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflo

##### Medicion() [2/2]

```
com.example.smariba_upv.airflow.POJO.Medicion.Medicion (
    int idSenor,
    String tipoGas,
    double latitud,
    double longitud,
    double valor)
```

Definición en la línea 28 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflo

### 6.14.3. Documentación de funciones miembro

#### getFecha()

```
Date com.example.smariba_upv.airflow.POJO.Medicion.getFecha ()
```

Definición en la línea 85 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

Gráfico de llamadas a esta función:



#### getId()

```
int com.example.smariba_upv.airflow.POJO.Medicion.getId ()
```

Definición en la línea 37 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

#### getIdSenor()

```
int com.example.smariba_upv.airflow.POJO.Medicion.getIdSenor ()
```

Definición en la línea 45 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

#### getLatitud()

```
double com.example.smariba_upv.airflow.POJO.Medicion.getLatitud ()
```

Definición en la línea 61 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

#### getLongitud()

```
double com.example.smariba_upv.airflow.POJO.Medicion.getLongitud ()
```

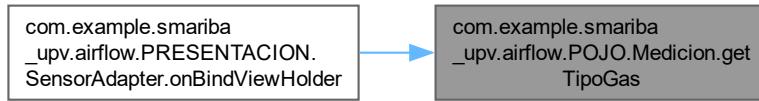
Definición en la línea 69 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**getTipoGas()**

```
String com.example.smariba_upv.airflow.POJO.Medicion.getTipoGas ()
```

Definición en la línea 53 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

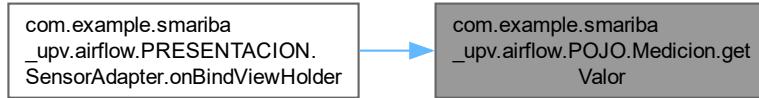
Gráfico de llamadas a esta función:

**getValor()**

```
double com.example.smariba_upv.airflow.POJO.Medicion.getValor ()
```

Definición en la línea 77 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

Gráfico de llamadas a esta función:

**setFecha()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setFecha (
    Date fecha)
```

Definición en la línea 89 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setId()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setId (
    int id)
```

Definición en la línea 41 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setIdSenor()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setIdSenor (
    int idSenor)
```

Definición en la línea 49 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setLatitud()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setLatitud (
    double latitud)
```

Definición en la línea 65 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setLongitud()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setLongitud (
    double longitud)
```

Definición en la línea 73 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setTipoGas()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setTipoGas (
    String tipoGas)
```

Definición en la línea 57 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

**setValor()**

```
void com.example.smariba_upv.airflow.POJO.Medicion.setValor (
    double valor)
```

Definición en la línea 81 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

La documentación de esta clase está generada del siguiente archivo:

- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

## 6.15. Referencia de la clase com.example.smariba\_upv.btle\_sento.POJO.Medicion

Clase que encapsula los datos de una medición de gas, incluyendo el lugar, el tipo de gas y su valor.

Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.POJO.Medicion:

com.example.smariba_upv.btle_sento.POJO.Medicion
<ul style="list-style-type: none"> <li>+ Medicion(String Lugar, String Gas, int Valor)</li> <li>+ String getLugar()</li> <li>+ void setLugar(String lugar)</li> <li>+ String getGas()</li> <li>+ void setGas(String gas)</li> <li>+ int getValor()</li> <li>+ void setValor(int valor)</li> </ul>

### Métodos públicos

- **Medicion** (String Lugar, String Gas, int Valor)

*Constructor de la clase Medicion.*

- String **getLugar** ()

*Obtiene el lugar de la medición.*

- void **setLugar** (String lugar)

*Establece el lugar de la medición.*

- String **getGas** ()

*Obtiene el tipo de gas medido.*

- void **setGas** (String gas)

*Establece el tipo de gas medido.*

- int **getValor** ()

*Obtiene el valor de la medición del gas.*

- void **setValor** (int valor)

*Establece el valor de la medición del gas.*

#### 6.15.1. Descripción detallada

Clase que encapsula los datos de una medición de gas, incluyendo el lugar, el tipo de gas y su valor.

Definición en la línea 14 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

### 6.15.2. Documentación de constructores y destructores

#### Medicion()

```
com.example.smariba_upv.btle_sento.POJO.Medicion.Medicion (
    String Lugar,
    String Gas,
    int Valor)
```

Constructor de la clase [Medicion](#).

##### Parámetros

<i>Lugar</i>	Nombre del lugar donde se realiza la medición.
<i>Gas</i>	Tipo de gas medido.
<i>Valor</i>	Valor medido del gas.

Definición en la línea [25](#) del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

### 6.15.3. Documentación de funciones miembro

#### getGas()

```
String com.example.smariba_upv.btle_sento.POJO.Medicion.getGas ()
```

Obtiene el tipo de gas medido.

##### Devuelve

Un String que representa el tipo de gas.

Definición en la línea [51](#) del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

#### getLugar()

```
String com.example.smariba_upv.btle_sento.POJO.Medicion.getLugar ()
```

Obtiene el lugar de la medición.

##### Devuelve

Un String que representa el lugar.

Definición en la línea [35](#) del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

**getValor()**

```
int com.example.smariba_upv.btle_sento.POJO.Medicion.getValor ()
```

Obtiene el valor de la medición del gas.

**Devuelve**

Un entero que representa el valor medido.

Definición en la línea [67](#) del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/PO...](#)

**setGas()**

```
void com.example.smariba_upv.btle_sento.POJO.Medicion.setGas (
    String gas)
```

Establece el tipo de gas medido.

**Parámetros**

<i>gas</i>	String que representa el tipo de gas.
------------	---------------------------------------

Definición en la línea 59 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

**setLugar()**

```
void com.example.smariba_upv.btle_sento.POJO.Medicion.setLugar (
    String lugar)
```

Establece el lugar de la medición.

**Parámetros**

<i>lugar</i>	String que representa el lugar.
--------------	---------------------------------

Definición en la línea 43 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

**setValor()**

```
void com.example.smariba_upv.btle_sento.POJO.Medicion.setValor (
    int valor)
```

Establece el valor de la medición del gas.

**Parámetros**

<i>valor</i>	Entero que representa el valor medido.
--------------	--

Definición en la línea 75 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

La documentación de esta clase está generada del siguiente archivo:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

## 6.16. Referencia de la clase

**com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCarga**

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCarga

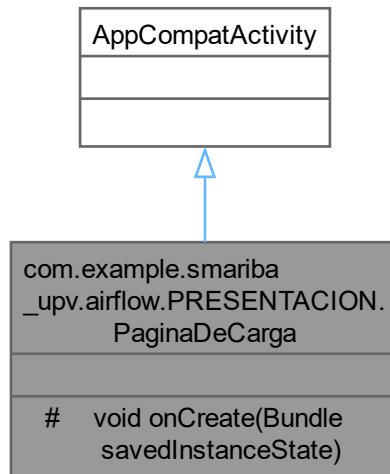
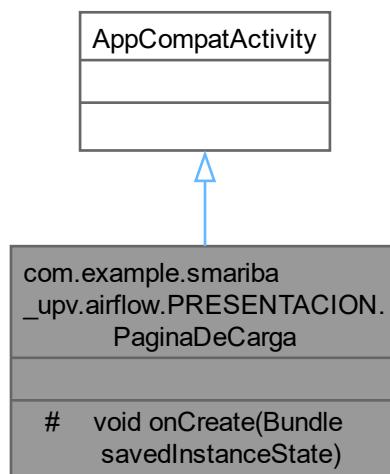


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCarga:



### Métodos protegidos

- void **onCreate** (Bundle savedInstanceState)

### 6.16.1. Descripción detallada

Definición en la línea 15 del archivo [PaginaDeCarga.java](#).

### 6.16.2. Documentación de funciones miembro

#### onCreate()

```
void com.example.smariba_upv.airflow.PRESENTACION.PaginaDeCarga.onCreate ( Bundle savedInstanceState) [protected]
```

Definición en la línea 18 del archivo [PaginaDeCarga.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [PaginaDeCarga.java](#)

## 6.17. Referencia de la clase

### com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity

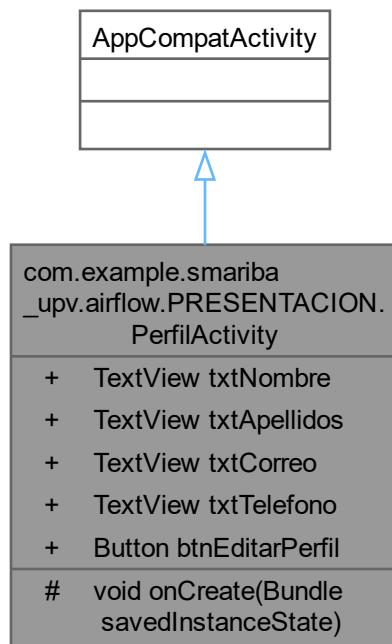
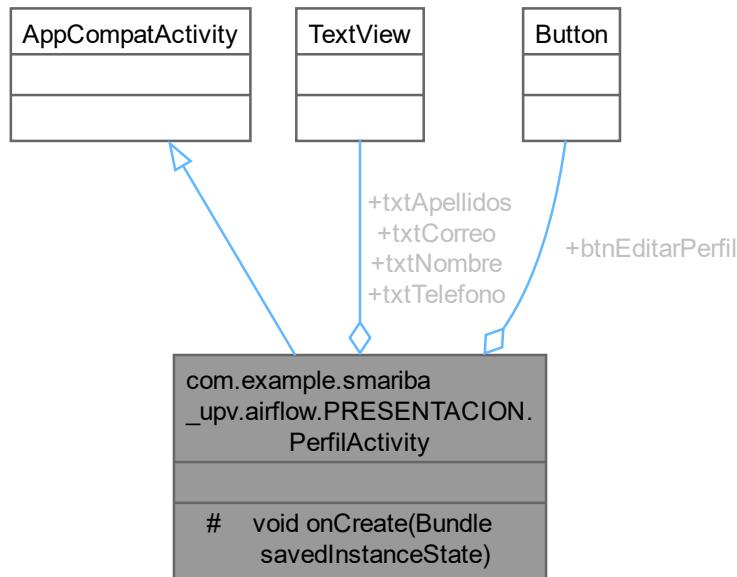


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity:



### Atributos públicos

- TextView [txtNombre](#)
- TextView [txtApellidos](#)
- TextView [txtCorreo](#)
- TextView [txtTelefono](#)
- Button [btnEditarPerfil](#)

### Métodos protegidos

- void [onCreate](#) (Bundle savedInstanceState)

#### 6.17.1. Descripción detallada

Definición en la línea 15 del archivo [PerfilActivity.java](#).

#### 6.17.2. Documentación de funciones miembro

##### [onCreate\(\)](#)

```
void com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.onCreate (
    Bundle savedInstanceState) [protected]
```

Definición en la línea 25 del archivo [PerfilActivity.java](#).

### 6.17.3. Documentación de datos miembro

#### **btnEditarPerfil**

```
Button com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.btnEditarPerfil
```

Definición en la línea 21 del archivo [PerfilActivity.java](#).

#### **txtApellidos**

```
TextView com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.txtApellidos
```

Definición en la línea 18 del archivo [PerfilActivity.java](#).

#### **txtCorreo**

```
TextView com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.txtCorreo
```

Definición en la línea 19 del archivo [PerfilActivity.java](#).

#### **txtNombre**

```
TextView com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.txtNombre
```

Definición en la línea 17 del archivo [PerfilActivity.java](#).

#### **txtTelefono**

```
TextView com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity.txtTelefono
```

Definición en la línea 20 del archivo [PerfilActivity.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [PerfilActivity.java](#)

### 6.18. Referencia de la clase

**com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment**

Clase que permite visualizar el perfil del usuario.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment

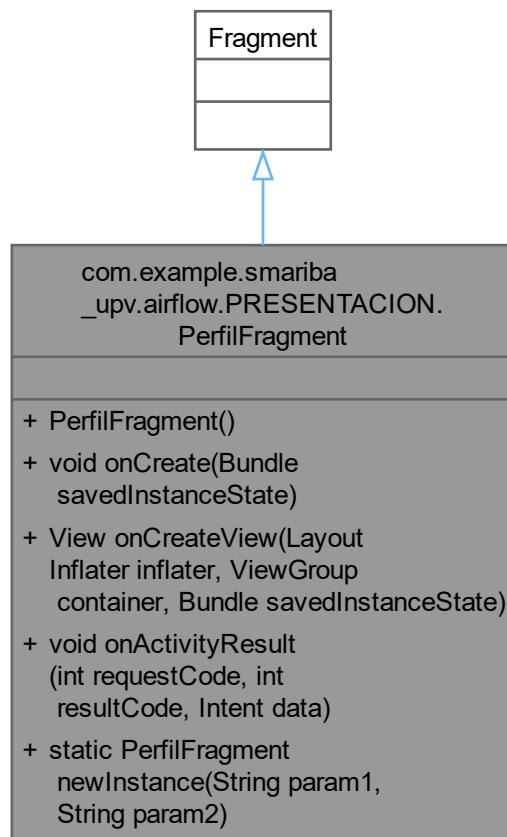
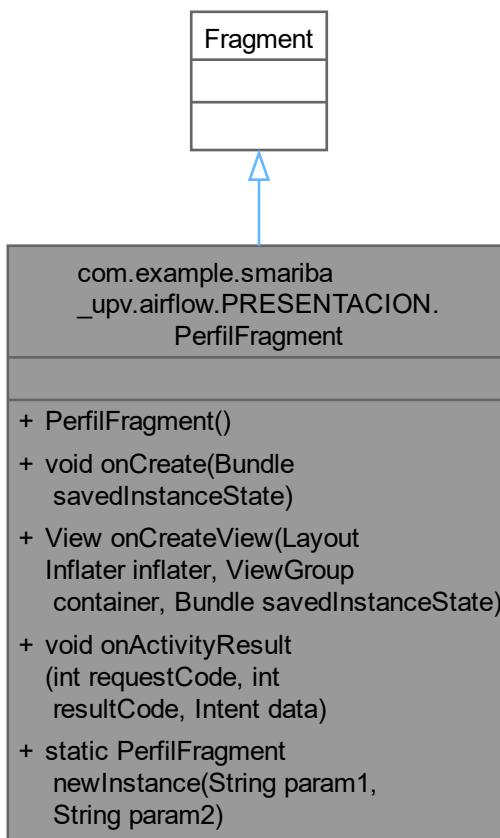


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment:



## Métodos públicos

- `PerfilFragment ()`  
    *@function PerfilFragment*
- `void onCreate (Bundle savedInstanceState)`  
    *@function onCreate*
- `View onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)`  
    *@function onCreateView*
- `void onActivityResult (int requestCode, int resultCode, Intent data)`  
    *@function onActivityResult*

## Métodos públicos estáticos

- `static PerfilFragment newInstance (String param1, String param2)`  
    *@function newInstance*

### 6.18.1. Descripción detallada

Clase que permite visualizar el perfil del usuario.

Clase que permite visualizar el perfil del usuario y editar la información del usuario

Definición en la línea 25 del archivo [PerfilFragment.java](#).

### 6.18.2. Documentación de constructores y destructores

#### **PerfilFragment()**

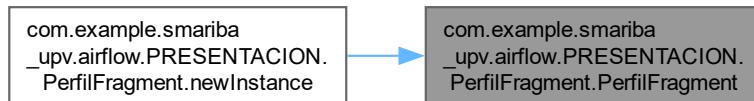
```
com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment.PerfilFragment ()
```

@function [PerfilFragment](#)

Constructor de la clase [PerfilFragment](#)

Definición en la línea 43 del archivo [PerfilFragment.java](#).

Gráfico de llamadas a esta función:



### 6.18.3. Documentación de funciones miembro

#### **newInstance()**

```
static PerfilFragment com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment.newInstance()
(
    String param1,
    String param2) [static]
```

@function newInstance

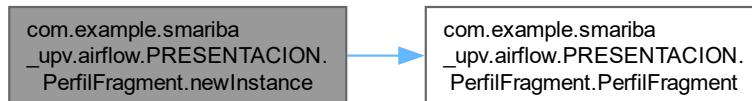
Parámetros

param1	
param2	

Devuelve

Definición en la línea 53 del archivo [PerfilFragment.java](#).

Gráfico de llamadas de esta función:



### **onActivityResult()**

```
void com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment.onActivityResult ( int requestCode, int resultCode, Intent data)
```

@function onActivityResult

Parámetros

requestCode	<input type="text"/>
resultCode	<input type="text"/>
data	<input type="text"/>

Definición en la línea 110 del archivo [PerfilFragment.java](#).

### **onCreate()**

```
void com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment.onCreate ( Bundle savedInstanceState)
```

@function onCreate

Parámetros

savedInstanceState	<input type="text"/>
--------------------	----------------------

Definición en la línea 66 del archivo [PerfilFragment.java](#).

### **onCreateView()**

```
View com.example.smariba_upv.airflow.PRESENTACION.PerfilFragment.onCreateView ( LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
```

@function onCreateView

#### Parámetros

<i>inflater</i>	
<i>container</i>	
<i>savedInstanceState</i>	

#### Devuelve

Definición en la línea 81 del archivo [PerfilFragment.java](#).

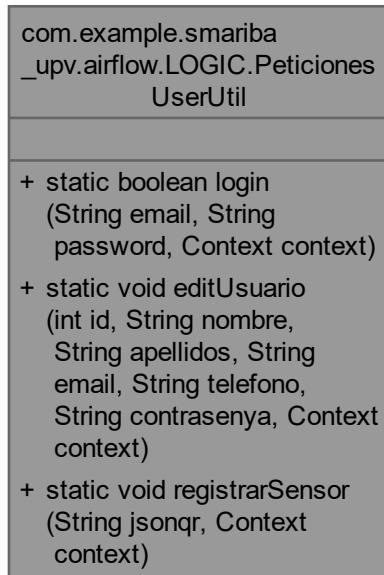
La documentación de esta clase está generada del siguiente archivo:

- [PerfilFragment.java](#)

### 6.19. Referencia de la clase

#### **com.example.smariba\_upv.airflow.LOGIC.PeticionesUserUtil**

Diagrama de colaboración de com.example.smariba\_upv.airflow.LOGIC.PeticionesUserUtil:



#### Métodos públicos estáticos

- static boolean [login](#) (String email, String password, Context context)  
    *@function login*
- static void [editUsuario](#) (int id, String nombre, String apellidos, String email, String telefono, String contrasenya, Context context)
- static void [registrarSensor](#) (String jsonqr, Context context)

### 6.19.1. Descripción detallada

Definición en la línea 15 del archivo [PeticionesUserUtil.java](#).

### 6.19.2. Documentación de funciones miembro

#### **editUsuario()**

```
static void com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil.editUsuario (
    int id,
    String nombre,
    String apellidos,
    String email,
    String telefono,
    String contrasenya,
    Context context) [static]
```

Definición en la línea 33 del archivo [PeticionesUserUtil.java](#).

Gráfico de llamadas de esta función:

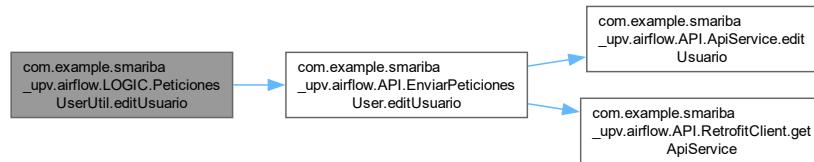
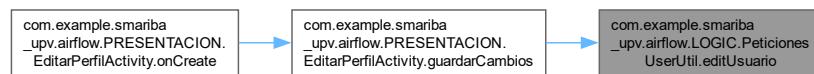


Gráfico de llamadas a esta función:



#### **login()**

```
static boolean com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil.login (
    String email,
    String password,
    Context context) [static]
```

@function login

### Parámetros

<i>email</i>	
<i>password</i>	

Función que comprueba si el email y la contraseña cumplen el formato correcto. antes de enviarlos al servidor.

Devuelve

void

Definición en la línea 26 del archivo [PeticionesUserUtil.java](#).

Gráfico de llamadas de esta función:

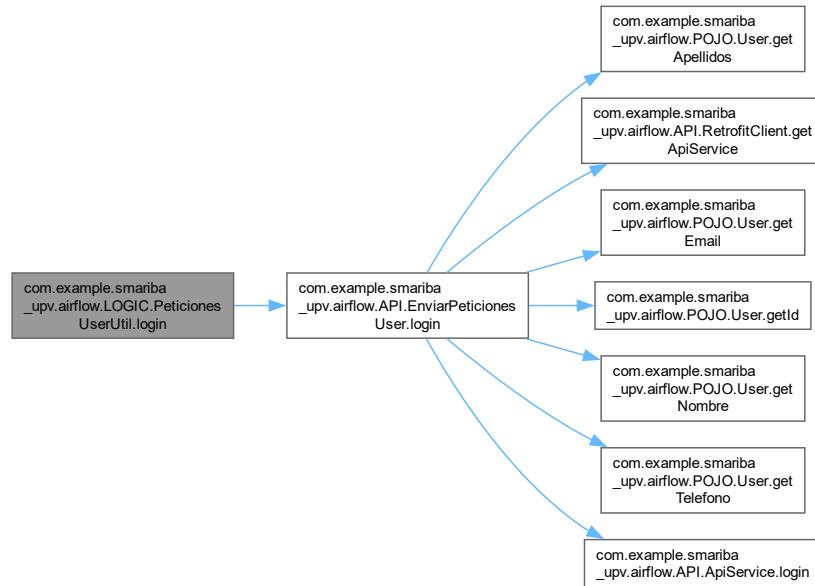
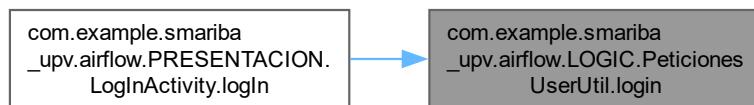


Gráfico de llamadas a esta función:

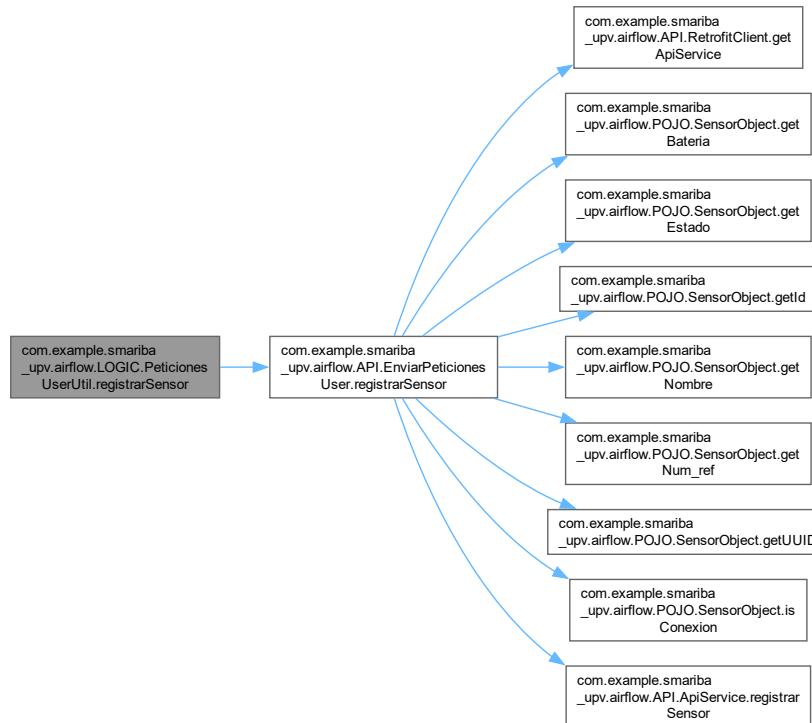


**registrarSensor()**

```
static void com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil.registrarSensor (
    String jsonqr,
    Context context) [static]
```

Definición en la línea 39 del archivo [PeticionesUserUtil.java](#).

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- [PeticionesUserUtil.java](#)

## 6.20. Referencia de la clase com.example.smariba\_upv.airflow.PRESENTACION.QRreader

Clase que permite leer un código QR.

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.QRreader

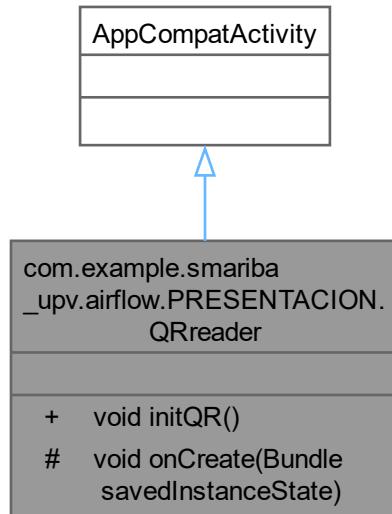
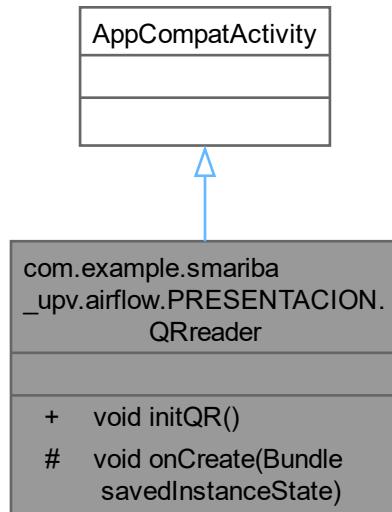


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.QRreader:



### Métodos públicos

- void **initQR** ()  
    *@function initQR*

## Métodos protegidos

- void [onCreate](#) (Bundle savedInstanceState)

*Método que se ejecuta al crear la actividad.*

### 6.20.1. Descripción detallada

Clase que permite leer un código QR.

Clase que permite leer un código QR y registrar un sensor en la base de datos

Definición en la línea [37](#) del archivo [QRreader.java](#).

### 6.20.2. Documentación de funciones miembro

#### [initQR\(\)](#)

```
void com.example.smariba_upv.airflow.PRESENTACION.QRreader.initQR ()
```

@function initQR

Método que inicializa el lector de QR

Método que inicializa el lector de QR y registra un sensor en la base de datos

Parámetros

<i>barcodeDetector</i>	Detector de códigos de barras
<i>cameraSource</i>	Fuente de la cámara

@function surfaceChanged

Parámetros

<i>holder</i>	Holder
<i>format</i>	Formato
<i>width</i>	Ancho
<i>height</i>	Alto

@function surfaceDestroyed

Parámetros

<i>holder</i>	Holder
---------------	--------

@function setProcessor Método que procesa el código QR

Método que procesa el código QR y registra un sensor en la base de datos

Definición en la línea 72 del archivo [QRreader.java](#).

Gráfico de llamadas a esta función:



### onCreate()

```
void com.example.smariba_upv.airflow.PRESENTACION.QRreader.onCreate ( Bundle savedInstanceState) [protected]
```

Método que se ejecuta al crear la actividad.

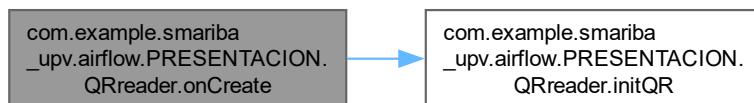
Método que se ejecuta al crear la actividad y permite leer un código QR

#### Parámetros

<i>savedInstanceState</i>	Instancia guardada
---------------------------	--------------------

Definición en la línea 57 del archivo [QRreader.java](#).

Gráfico de llamadas de esta función:



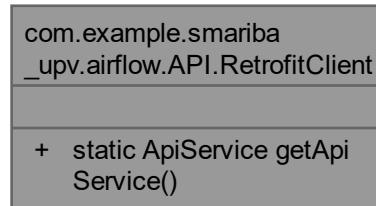
La documentación de esta clase está generada del siguiente archivo:

- [QRreader.java](#)

## 6.21. Referencia de la clase com.example.smariba\_upv.airflow.API.RetrofitClient

Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.

Diagrama de colaboración de com.example.smariba\_upv.airflow.API.RetrofitClient:



### Métodos públicos estáticos

- static ApiService getApiService ()

#### 6.21.1. Descripción detallada

Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.

Definición en la línea 17 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/RetrofitClient.java

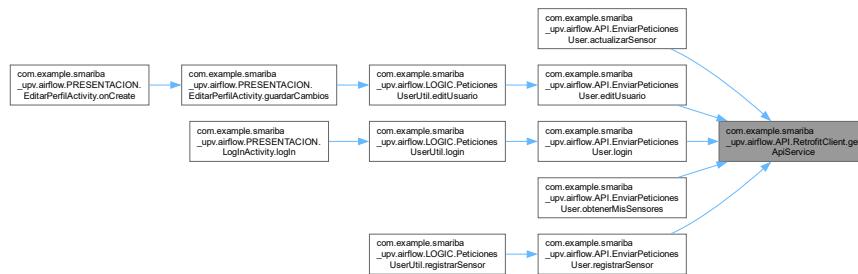
#### 6.21.2. Documentación de funciones miembro

##### getApiService()

```
static ApiService com.example.smariba_upv.airflow.API.RetrofitClient.getApiService () [static]
```

Definición en la línea 22 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/RetrofitClient.java

Gráfico de llamadas a esta función:



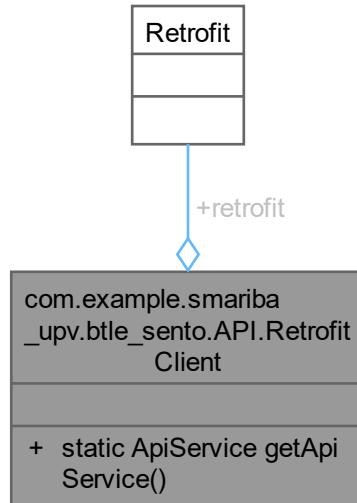
La documentación de esta clase está generada del siguiente archivo:

- Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/RetrofitClient.java

## 6.22. Referencia de la clase com.example.smariba\_upv.btle\_sento.API.RetrofitClient

Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.

Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.API.RetrofitClient:



### Métodos públicos estáticos

- `static ApiService getApiService ()`

*Método estático que devuelve una instancia de `ApiService` para hacer llamadas a la API.*

### Atributos públicos estáticos

- `static Retrofit retrofit = null`

*Instancia única de Retrofit, utilizada para realizar las solicitudes a la API.*

#### 6.22.1. Descripción detallada

Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.

Definición en la línea 17 del archivo `AndroidStudioProjects/BTLE_Sento/app/src/main/java/com/example/smariba_upv/btle_sento/API`

### 6.22.2. Documentación de funciones miembro

#### getApiService()

```
static ApiService com.example.smariba_upv.btle_sento.API.RetrofitClient.getApiService () [static]
```

Método estático que devuelve una instancia de [ApiService](#) para hacer llamadas a la [API](#).

Este método comprueba si la instancia de Retrofit ya ha sido creada. Si no lo ha sido, la crea y configura. Utiliza el patrón Singleton para asegurarse de que solo se cree una instancia de Retrofit.

Devuelve

Retorna una instancia de [ApiService](#) que contiene las definiciones de las operaciones [API](#).

Definición en la línea 34 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/RetrofitClient.java](#)

### 6.22.3. Documentación de datos miembro

#### retrofit

```
Retrofit com.example.smariba_upv.btle_sento.API.RetrofitClient.retrofit = null [static]
```

Instancia única de Retrofit, utilizada para realizar las solicitudes a la [API](#).

Definición en la línea 23 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/RetrofitClient.java](#)

La documentación de esta clase está generada del siguiente archivo:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/RetrofitClient.java](#)

**6.23. Referencia de la clase****com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment**

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment

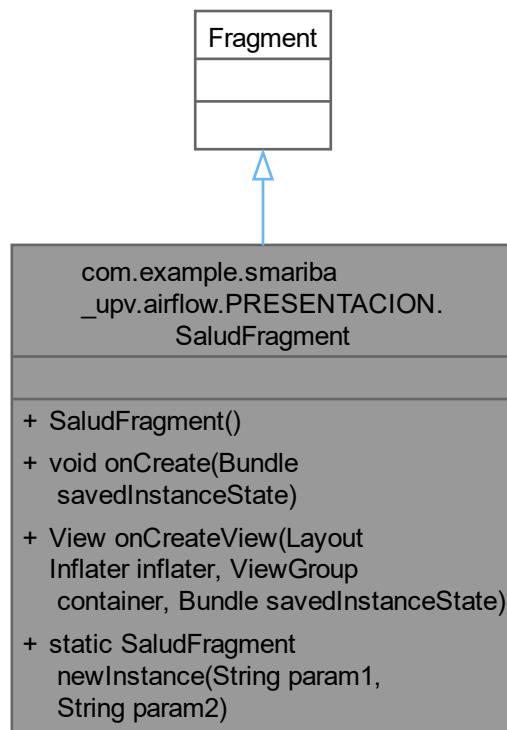
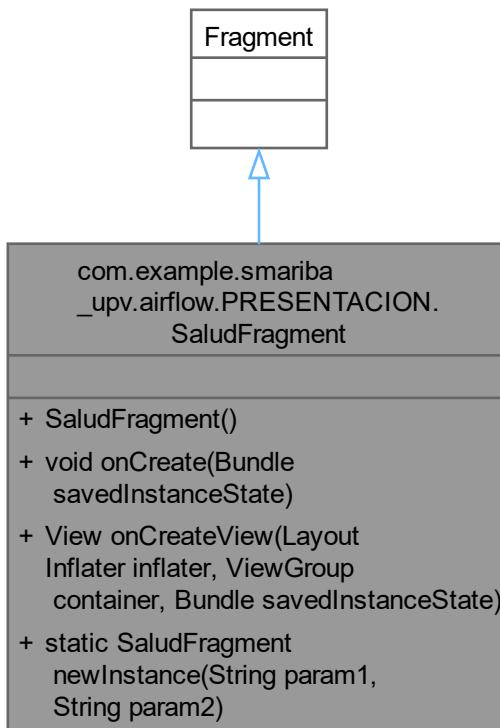


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment:



### Métodos públicos

- [SaludFragment \(\)](#)
- [void onCreate \(Bundle savedInstanceState\)](#)
- [View onCreateView \(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState\)](#)

### Métodos públicos estáticos

- [static SaludFragment newInstance \(String param1, String param2\)](#)  
*Use this factory method to create a new instance of this fragment using the provided parameters.*

#### 6.23.1. Descripción detallada

Definición en la línea 14 del archivo [SaludFragment.java](#).

### 6.23.2. Documentación de constructores y destructores

#### **SaludFragment()**

```
com.example.smariba_upv.airflow.PRESENTACION.SaludFragment.SaludFragment ()
```

Definición en la línea 25 del archivo [SaludFragment.java](#).

Gráfico de llamadas a esta función:



### 6.23.3. Documentación de funciones miembro

#### **newInstance()**

```
static SaludFragment com.example.smariba_upv.airflow.PRESENTACION.SaludFragment.newInstance (
    String param1,
    String param2) [static]
```

Use this factory method to create a new instance of this fragment using the provided parameters.

##### Parámetros

<i>param1</i>	Parameter 1.
<i>param2</i>	Parameter 2.

##### Devuelve

A new instance of fragment [SaludFragment](#).

Definición en la línea 38 del archivo [SaludFragment.java](#).

Gráfico de llamadas de esta función:



**onCreate()**

```
void com.example.smariba_upv.airflow.PRESENTACION.SaludFragment.onCreate (
    Bundle savedInstanceState)
```

Definición en la línea 48 del archivo [SaludFragment.java](#).

**onCreateView()**

```
View com.example.smariba_upv.airflow.PRESENTACION.SaludFragment.onCreateView (
    LayoutInflater inflater,
    ViewGroup container,
    Bundle savedInstanceState)
```

Definición en la línea 57 del archivo [SaludFragment.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SaludFragment.java](#)

**6.24. Referencia de la clase****com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter**

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter

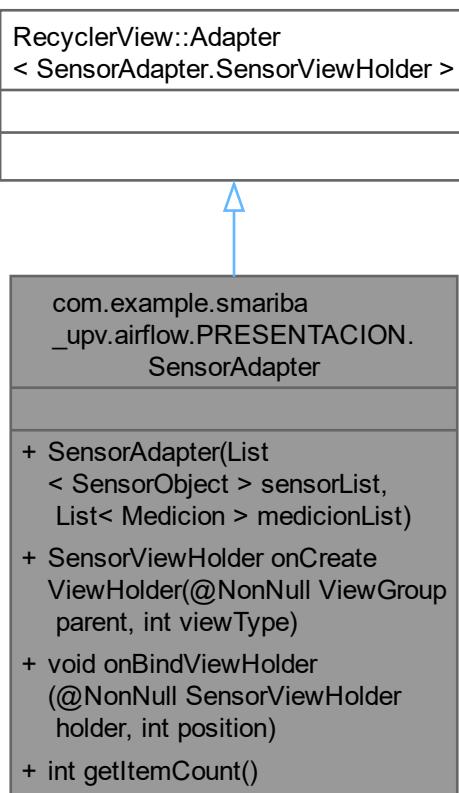
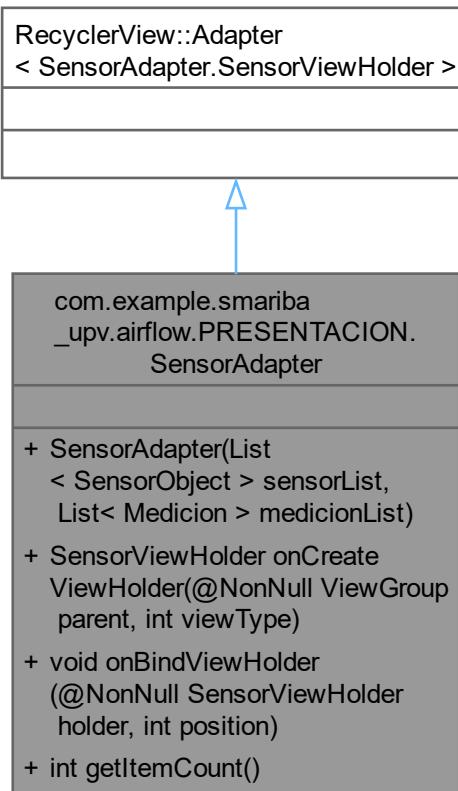


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter:



## Clases

- class **SensorViewHolder**

## Métodos públicos

- **SensorAdapter** (`List< SensorObject > sensorList, List< Medicion > medicionList)`  
*Constructor.*
- **SensorViewHolder** `onCreateViewHolder` (`@NonNull ViewGroup parent, int viewType)`
- **void** `onBindViewHolder` (`@NonNull SensorViewHolder holder, int position)`
- **int** `getItemCount ()`

### 6.24.1. Descripción detallada

Definición en la línea 21 del archivo [SensorAdapter.java](#).

### 6.24.2. Documentación de constructores y destructores

#### SensorAdapter()

```
com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter.SensorAdapter (
    List< SensorObject > sensorList,
    List< Medicion > medicionList)
```

Constructor.

##### Parámetros

<i>sensorList</i>	List of sensors
<i>medicionList</i>	List of measurements

Definición en la línea 35 del archivo [SensorAdapter.java](#).

### 6.24.3. Documentación de funciones miembro

#### getItemCount()

```
int com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter.getItemCount ()
```

##### Devuelve

The total number of items in the data set held by the adapter.

Definición en la línea 90 del archivo [SensorAdapter.java](#).

#### onBindViewHolder()

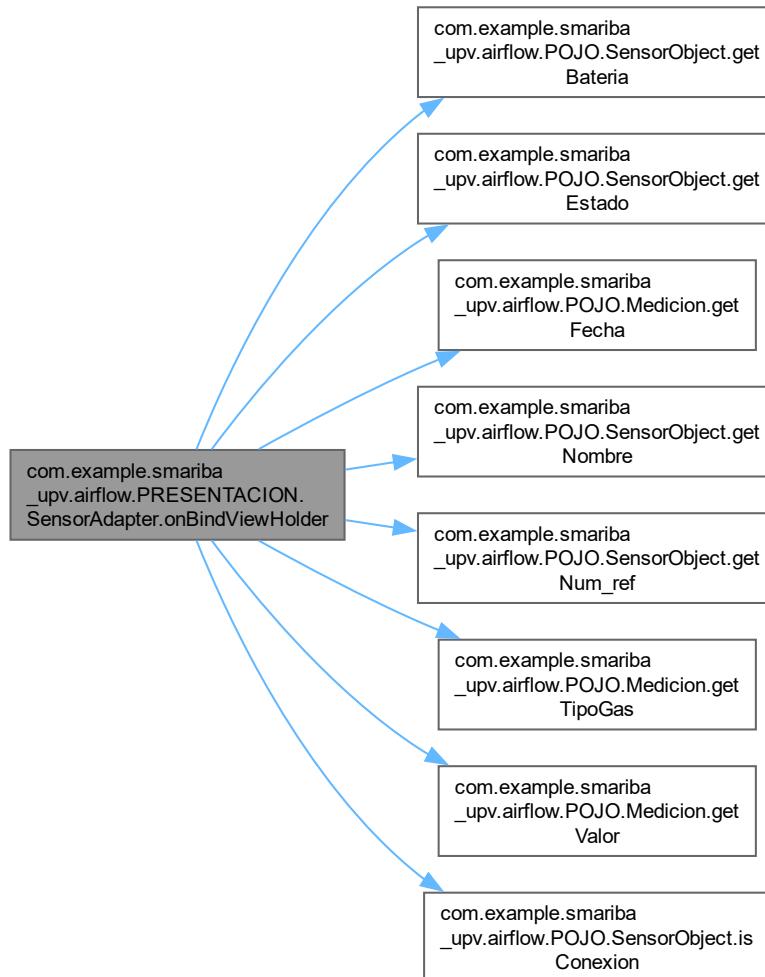
```
void com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter.onBindViewHolder (
    @NonNull SensorViewHolder holder,
    int position)
```

##### Parámetros

<i>holder</i>	The ViewHolder which should be updated to represent the contents of the item at the given position in the data set.
<i>position</i>	The position of the item within the adapter's data set.

Definición en la línea 59 del archivo [SensorAdapter.java](#).

Gráfico de llamadas de esta función:



### **onCreateViewHolder()**

```

SensorViewHolder com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter.onCreateViewHolder
(
    @NonNull ViewGroup parent,
    int viewType)
  
```

#### Parámetros

<i>parent</i>	The ViewGroup into which the new View will be added after it is bound to an adapter position.
<i>viewType</i>	The view type of the new View.

Devuelve

Definición en la línea 48 del archivo [SensorAdapter.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SensorAdapter.java](#)

## 6.25. Referencia de la clase

### com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment

Diagrama de herencia de com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment

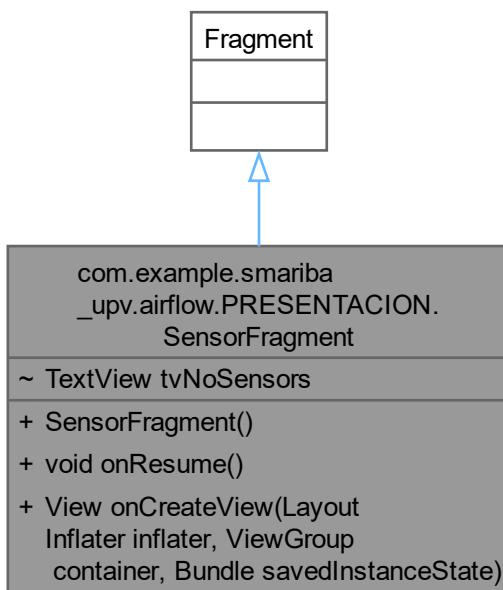
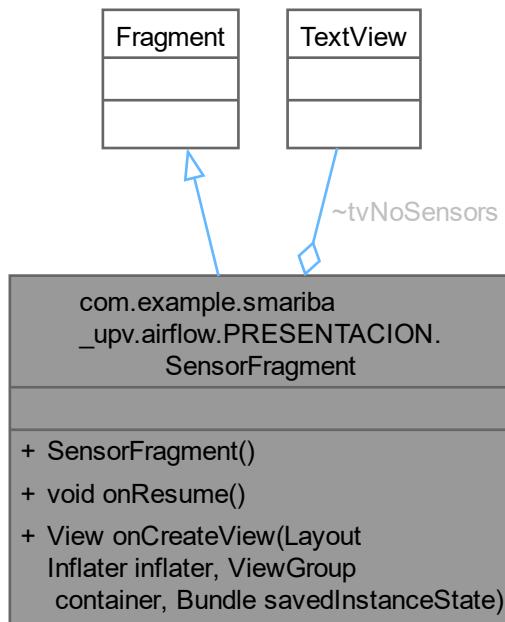


Diagrama de colaboración de com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment:



## Métodos públicos

- [SensorFragment \(\)](#)
- [void onResume \(\)](#)
- [View onCreateView \(LayoutInflator inflater, ViewGroup container, Bundle savedInstanceState\)](#)

### 6.25.1. Descripción detallada

Definición en la línea [29](#) del archivo [SensorFragment.java](#).

### 6.25.2. Documentación de constructores y destructores

#### **SensorFragment()**

```
com.example.smariba_upv.airflow.PRESENTACION.SensorFragment.SensorFragment ()
```

Definición en la línea [37](#) del archivo [SensorFragment.java](#).

### 6.25.3. Documentación de funciones miembro

#### **onCreateView()**

```
View com.example.smariba_upv.airflow.PRESENTACION.SensorFragment.onCreateView (
    LayoutInflater inflater,
    ViewGroup container,
    Bundle savedInstanceState)
```

Definición en la línea 74 del archivo [SensorFragment.java](#).

#### **onResume()**

```
void com.example.smariba_upv.airflow.PRESENTACION.SensorFragment.onResume ()
```

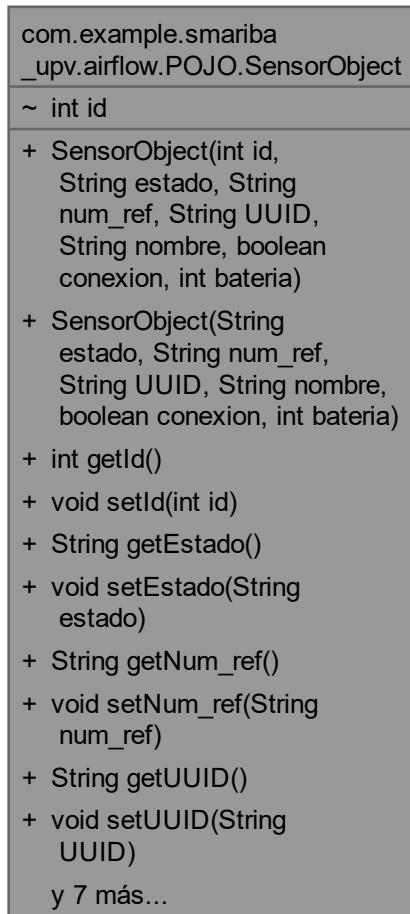
Definición en la línea 42 del archivo [SensorFragment.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SensorFragment.java](#)

## 6.26. Referencia de la clase com.example.smariba\_upv.airflow.POJO.SensorObject

Diagrama de colaboración de com.example.smariba\_upv.airflow.POJO.SensorObject:



### Métodos públicos

- [SensorObject](#) (int id, String estado, String num\_ref, String UUID, String nombre, boolean conexion, int bateria)
- [SensorObject](#) (String estado, String num\_ref, String UUID, String nombre, boolean conexion, int bateria)
- int [getId](#) ()
- void [setId](#) (int id)
- String [getEstado](#) ()
- void [setEstado](#) (String estado)
- String [getNum\\_ref](#) ()
- void [setNum\\_ref](#) (String num\_ref)
- String [getUUID](#) ()
- void [setUUID](#) (String UUID)
- String [getNombre](#) ()
- void [setNombre](#) (String nombre)

- boolean `isConexion ()`
- void `setConexion (boolean conexion)`
- int `getBateria ()`
- void `setBateria (int bateria)`
- String `toString ()`

### 6.26.1. Descripción detallada

Definición en la línea 9 del archivo [SensorObject.java](#).

### 6.26.2. Documentación de constructores y destructores

#### **SensorObject()** [1/2]

```
com.example.smariba_upv.airflow.POJO.SensorObject.SensorObject (
    int id,
    String estado,
    String num_ref,
    String UUID,
    String nombre,
    boolean conexion,
    int bateria)
```

Definición en la línea 28 del archivo [SensorObject.java](#).

#### **SensorObject()** [2/2]

```
com.example.smariba_upv.airflow.POJO.SensorObject.SensorObject (
    String estado,
    String num_ref,
    String UUID,
    String nombre,
    boolean conexion,
    int bateria)
```

Definición en la línea 37 del archivo [SensorObject.java](#).

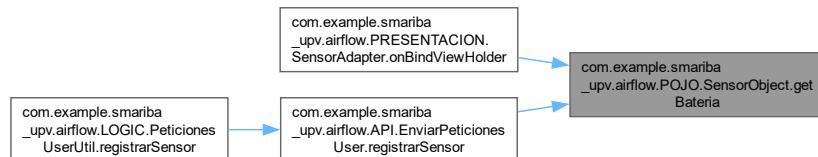
### 6.26.3. Documentación de funciones miembro

#### **getBateria()**

```
int com.example.smariba_upv.airflow.POJO.SensorObject.getBateria ()
```

Definición en la línea 94 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:

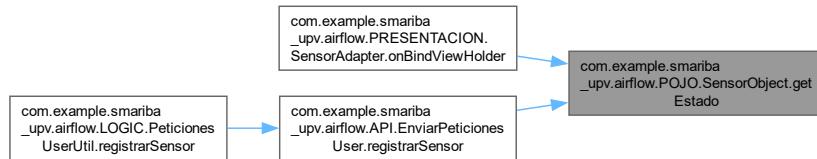


### getEstado()

```
String com.example.smariba_upv.airflow.POJO.SensorObject.getEstado ()
```

Definición en la línea 54 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:

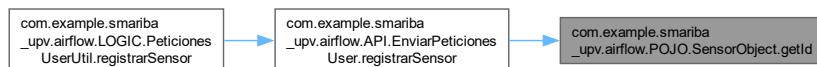


### getId()

```
int com.example.smariba_upv.airflow.POJO.SensorObject.getId ()
```

Definición en la línea 46 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:

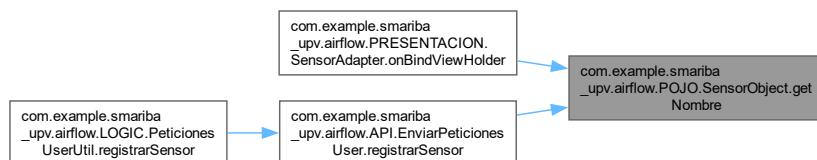


### getNombre()

```
String com.example.smariba_upv.airflow.POJO.SensorObject.getNombre ()
```

Definición en la línea 78 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:

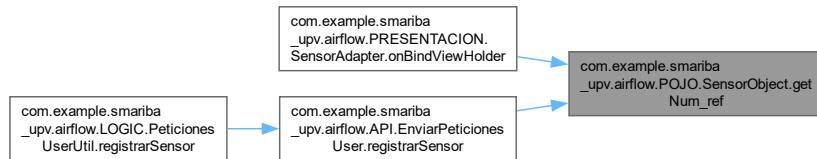


**getNum\_ref()**

```
String com.example.smariba_upv.airflow.POJO.SensorObject.getNum_ref ()
```

Definición en la línea 62 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:

**getUUID()**

```
String com.example.smariba_upv.airflow.POJO.SensorObject.getUUID ()
```

Definición en la línea 70 del archivo [SensorObject.java](#).

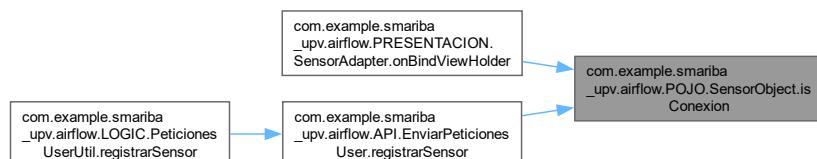
Gráfico de llamadas a esta función:

**isConexion()**

```
boolean com.example.smariba_upv.airflow.POJO.SensorObject.isConexion ()
```

Definición en la línea 86 del archivo [SensorObject.java](#).

Gráfico de llamadas a esta función:



**setBateria()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setBateria (
    int bateria)
```

Definición en la línea 98 del archivo [SensorObject.java](#).

**setConexion()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setConexion (
    boolean conexion)
```

Definición en la línea 90 del archivo [SensorObject.java](#).

**setEstado()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setEstado (
    String estado)
```

Definición en la línea 58 del archivo [SensorObject.java](#).

**setId()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setId (
    int id)
```

Definición en la línea 50 del archivo [SensorObject.java](#).

**setNombre()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setNombre (
    String nombre)
```

Definición en la línea 82 del archivo [SensorObject.java](#).

**setNum\_ref()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setNum_ref (
    String num_ref)
```

Definición en la línea 66 del archivo [SensorObject.java](#).

**setUUID()**

```
void com.example.smariba_upv.airflow.POJO.SensorObject.setUUID (
    String UUID)
```

Definición en la línea 74 del archivo [SensorObject.java](#).

**toString()**

```
String com.example.smariba_upv.airflow.POJO.SensorObject.toString ()
```

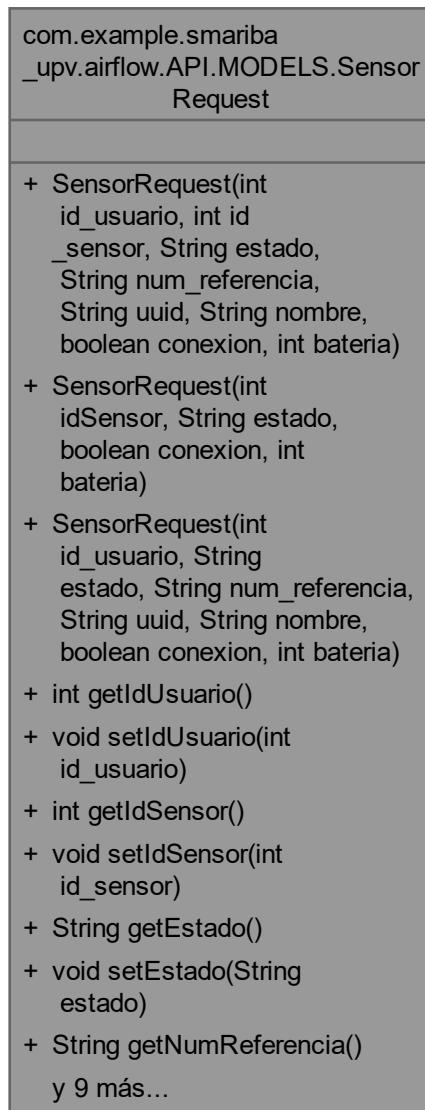
Definición en la línea 103 del archivo [SensorObject.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SensorObject.java](#)

**6.27. Referencia de la clase****com.example.smariba\_upv.airflow.API.MODELS.SensorRequest**

Diagrama de colaboración de com.example.smariba\_upv.airflow.API.MODELS.SensorRequest:



## Métodos públicos

- `SensorRequest` (int id\_usuario, int id\_sensor, String estado, String num\_referencia, String uuid, String nombre, boolean conexion, int bateria)
- `SensorRequest` (int idSensor, String estado, boolean conexion, int bateria)
- `SensorRequest` (int id\_usuario, String estado, String num\_referencia, String uuid, String nombre, boolean conexion, int bateria)
- int `getIdUsuario` ()
- void `setIdUsuario` (int id\_usuario)
- int `getIdSensor` ()
- void `setIdSensor` (int id\_sensor)
- String `getEstado` ()
- void `setEstado` (String estado)
- String `getNumReferencia` ()
- void `setNumReferencia` (String num\_referencia)
- String `getUuid` ()
- void `setUuid` (String uuid)
- String `getNombre` ()
- void `setNombre` (String nombre)
- boolean `isConexion` ()
- void `setConexion` (boolean conexion)
- int `getBateria` ()
- void `setBateria` (int bateria)

### 6.27.1. Descripción detallada

Definición en la línea 8 del archivo [SensorRequest.java](#).

### 6.27.2. Documentación de constructores y destructores

#### `SensorRequest()` [1/3]

```
com.example.smariba_upv.airflow.API.MODELS.SensorRequest.SensorRequest (
    int id_usuario,
    int id_sensor,
    String estado,
    String num_referencia,
    String uuid,
    String nombre,
    boolean conexion,
    int bateria)
```

Definición en la línea 43 del archivo [SensorRequest.java](#).

#### `SensorRequest()` [2/3]

```
com.example.smariba_upv.airflow.API.MODELS.SensorRequest.SensorRequest (
    int idSensor,
    String estado,
    boolean conexion,
    int bateria)
```

Definición en la línea 61 del archivo [SensorRequest.java](#).

**SensorRequest()** [3/3]

```
com.example.smariba_upv.airflow.API.MODELS.SensorRequest.SensorRequest ( 
    int id_usuario,
    String estado,
    String num_referencia,
    String uuid,
    String nombre,
    boolean conexion,
    int bateria)
```

Definición en la línea 78 del archivo [SensorRequest.java](#).

### 6.27.3. Documentación de funciones miembro

**getBateria()**

```
int com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getBateria ()
```

Devuelve

Definición en la línea 190 del archivo [SensorRequest.java](#).

**getEstado()**

```
String com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getEstado ()
```

Devuelve

Definición en la línea 120 del archivo [SensorRequest.java](#).

**getIdSensor()**

```
int com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getIdSensor ()
```

Devuelve

Definición en la línea 106 del archivo [SensorRequest.java](#).

**getIdUsuario()**

```
int com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getIdUsuario ()
```

**Devuelve**

Definición en la línea 92 del archivo [SensorRequest.java](#).

**getNombre()**

```
String com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getNombre ()
```

**Devuelve**

Definición en la línea 162 del archivo [SensorRequest.java](#).

**getNumReferencia()**

```
String com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getNumReferencia ()
```

**Devuelve**

Definición en la línea 134 del archivo [SensorRequest.java](#).

**getUuid()**

```
String com.example.smariba_upv.airflow.API.MODELS.SensorRequest.getUuid ()
```

**Devuelve**

Definición en la línea 148 del archivo [SensorRequest.java](#).

**isConexion()**

```
boolean com.example.smariba_upv.airflow.API.MODELS.SensorRequest.isConexion ()
```

**Devuelve**

Definición en la línea 176 del archivo [SensorRequest.java](#).

**setBateria()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setBateria (
    int bateria)
```

**Parámetros**

bateria	<input type="button" value=""/>
---------	---------------------------------

Definición en la línea 197 del archivo [SensorRequest.java](#).

**setConexion()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setConexion (
    boolean conexion)
```

**Parámetros**

conexion	<input type="button" value=""/>
----------	---------------------------------

Definición en la línea 183 del archivo [SensorRequest.java](#).

**setEstado()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setEstado (
    String estado)
```

**Parámetros**

estado	<input type="button" value=""/>
--------	---------------------------------

Definición en la línea 127 del archivo [SensorRequest.java](#).

**setIdSensor()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setIdSensor (
    int id_sensor)
```

**Parámetros**

id_sensor	<input type="button" value=""/>
-----------	---------------------------------

Definición en la línea 113 del archivo [SensorRequest.java](#).

**setIdUsuario()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setIdUsuario (
    int id_usuario)
```

**Parámetros**

<i>id_usuario</i>	<input type="button" value=""/>
-------------------	---------------------------------

Definición en la línea 99 del archivo [SensorRequest.java](#).

**setNombre()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setNombre (
    String nombre)
```

**Parámetros**

<i>nombre</i>	<input type="button" value=""/>
---------------	---------------------------------

Definición en la línea 169 del archivo [SensorRequest.java](#).

**setNumReferencia()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setNumReferencia (
    String num_referencia)
```

**Parámetros**

<i>num_referencia</i>	<input type="button" value=""/>
-----------------------	---------------------------------

Definición en la línea 141 del archivo [SensorRequest.java](#).

**setUuid()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorRequest.setUuid (
    String uuid)
```

**Parámetros**

<i>uuid</i>	<input type="button" value=""/>
-------------	---------------------------------

Definición en la línea 155 del archivo [SensorRequest.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SensorRequest.java](#)

## 6.28. Referencia de la clase com.example.smariba\_upv.airflow.API.MODELS.SensorResponse

Clase que representa un objeto [SensorResponse](#).

Diagrama de colaboración de com.example.smariba\_upv.airflow.API.MODELS.SensorResponse:

com.example.smariba_upv.airflow.API.MODELS.Sensor Response
<ul style="list-style-type: none"><li>+ int getIdUsuario()</li><li>+ void setIdUsuario(int idUsuario)</li><li>+ int getIdSensor()</li><li>+ void setIdSensor(int idSensor)</li><li>+ Sensor getSensor()</li><li>+ void setSensor(Sensor sensor)</li></ul>

### Clases

- class **Sensor**

*Clase que representa un objeto Sensor.*

### Métodos públicos

- int [getIdUsuario \(\)](#)
- void [setIdUsuario \(int idUsuario\)](#)
- int [getIdSensor \(\)](#)
- void [setIdSensor \(int idSensor\)](#)
- Sensor [getSensor \(\)](#)
- void [setSensor \(Sensor sensor\)](#)

#### 6.28.1. Descripción detallada

Clase que representa un objeto [SensorResponse](#).

Definición en la línea 14 del archivo [SensorResponse.java](#).

### 6.28.2. Documentación de funciones miembro

#### **getIdSensor()**

```
int com.example.smariba_upv.airflow.API.MODELS.SensorResponse.getIdSensor ()
```

Definición en la línea 40 del archivo [SensorResponse.java](#).

#### **getIdUsuario()**

```
int com.example.smariba_upv.airflow.API.MODELS.SensorResponse.getIdUsuario ()
```

Definición en la línea 32 del archivo [SensorResponse.java](#).

#### **getSensor()**

```
Sensor com.example.smariba_upv.airflow.API.MODELS.SensorResponse.getSensor ()
```

Definición en la línea 48 del archivo [SensorResponse.java](#).

#### **setIdSensor()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorResponse.setIdSensor ( int idSensor )
```

Definición en la línea 44 del archivo [SensorResponse.java](#).

#### **setIdUsuario()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorResponse.setIdUsuario ( int idUsuario )
```

Definición en la línea 36 del archivo [SensorResponse.java](#).

#### **setSensor()**

```
void com.example.smariba_upv.airflow.API.MODELS.SensorResponse.setSensor ( Sensor sensor )
```

Definición en la línea 52 del archivo [SensorResponse.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [SensorResponse.java](#)

## 6.29. Referencia de la clase com.example.smariba\_upv.airflow.POJO.TramalBeacon

Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.

Diagrama de colaboración de com.example.smariba\_upv.airflow.POJO.TramalBeacon:

com.example.smariba_upv.airflow.POJO.TramalBeacon
<ul style="list-style-type: none"><li>+ TramalBeacon(byte [] bytes)</li><li>+ byte[] getPrefijo()</li><li>+ byte[] getUUID()</li><li>+ byte[] getMajor()</li><li>+ byte[] getMinor()</li><li>+ byte getTxPower()</li><li>+ byte[] getLosBytes()</li><li>+ byte[] getAdvFlags()</li><li>+ byte[] getAdvHeader()</li><li>+ byte[] getCompanyID()</li><li>+ byte getiBeaconType()</li><li>+ byte getiBeaconLength()</li></ul>

### Métodos públicos

- [TramalBeacon \(byte\[\] bytes\)](#)

*Constructor de la clase TramalBeacon.*

- [byte\[\] getPrefijo \(\)](#)
- [byte\[\] getUUID \(\)](#)
- [byte\[\] getMajor \(\)](#)
- [byte\[\] getMinor \(\)](#)
- [byte getTxPower \(\)](#)

*Obtiene el valor de Tx Power de la trama iBeacon.*

- [byte\[\] getLosBytes \(\)](#)
- [byte\[\] getAdvFlags \(\)](#)
- [byte\[\] getAdvHeader \(\)](#)
- [byte\[\] getCompanyID \(\)](#)
- [byte getiBeaconType \(\)](#)

*Obtiene el tipo de iBeacon.*

- [byte getiBeaconLength \(\)](#)

*Obtiene la longitud de los datos iBeacon.*

### 6.29.1. Descripción detallada

Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.

Definición en la línea 16 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow](#)

### 6.29.2. Documentación de constructores y destructores

#### **TramaIBeacon()**

```
com.example.smariba_upv.airflow.POJO.TramaIBeacon (byte[] bytes)
```

Constructor de la clase [TramaIBeacon](#).

##### Parámetros

<i>bytes</i>	Array de bytes que contiene los datos de la trama iBeacon.
--------------	--

Definición en la línea 35 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow](#)

### 6.29.3. Documentación de funciones miembro

#### **getAdvFlags()**

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getAdvFlags ()
```

Definición en la línea 110 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow](#)

#### **getAdvHeader()**

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getAdvHeader ()
```

Definición en la línea 119 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow](#)

#### **getCompanyID()**

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getCompanyID ()
```

Definición en la línea 128 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow](#)

**getiBeaconLength()**

```
byte com.example.smariba_upv.airflow.POJO.TramaIBeacon.getiBeaconLength ()
```

Obtiene la longitud de los datos iBeacon.

**Devuelve**

Un byte que representa la longitud del iBeacon.

Definición en la línea 146 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

**getiBeaconType()**

```
byte com.example.smariba_upv.airflow.POJO.TramaIBeacon.getiBeaconType ()
```

Obtiene el tipo de iBeacon.

**Devuelve**

Un byte que representa el tipo de iBeacon.

Definición en la línea 137 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

**getLosBytes()**

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getLosBytes ()
```

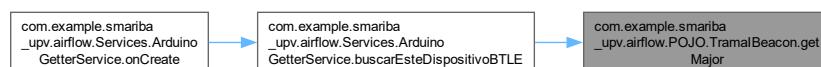
Definición en la línea 101 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

**getMajor()**

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getMajor ()
```

Definición en la línea 74 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

Gráfico de llamadas a esta función:

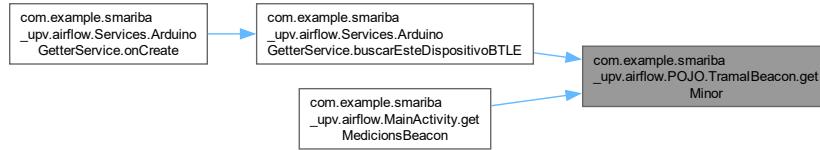


### getMinor()

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getMinor ()
```

Definición en la línea 83 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramaIBeacon.java](#)

Gráfico de llamadas a esta función:



### getPrefijo()

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getPrefijo ()
```

Definición en la línea 56 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramaIBeacon.java](#)

### getTxPower()

```
byte com.example.smariba_upv.airflow.POJO.TramaIBeacon.getTxPower ()
```

Obtiene el valor de Tx Power de la trama iBeacon.

Devuelve

Un byte que representa el valor de Tx Power.

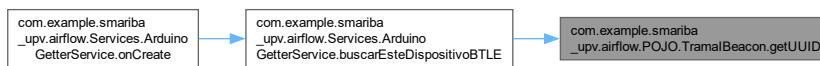
Definición en la línea 92 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramaIBeacon.java](#)

### getUUID()

```
byte[] com.example.smariba_upv.airflow.POJO.TramaIBeacon.getUUID ()
```

Definición en la línea 65 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramaIBeacon.java](#)

Gráfico de llamadas a esta función:



La documentación de esta clase está generada del siguiente archivo:

- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramaIBeacon.java](#)

### 6.30. Referencia de la clase com.example.smariba\_upv.btle\_sento.POJO.TramalBeacon

Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.

Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.POJO.TramalBeacon:

com.example.smariba_upv.btle_sento.POJO.Trama IBeacon
<ul style="list-style-type: none"> <li>+ TramalBeacon(byte[] bytes)</li> <li>+ byte[] getPrefijo()</li> <li>+ byte[] getUUID()</li> <li>+ byte[] getMajor()</li> <li>+ byte[] getMinor()</li> <li>+ byte getTxPower()</li> <li>+ byte[] getLosBytes()</li> <li>+ byte[] getAdvFlags()</li> <li>+ byte[] getAdvHeader()</li> <li>+ byte[] getCompanyID()</li> <li>+ byte getiBeaconType()</li> <li>+ byte getiBeaconLength()</li> </ul>

#### Métodos públicos

- [TramalBeacon](#) (byte[] bytes)

*Constructor de la clase TramalBeacon.*

- byte[] [getPrefijo](#) ()
- byte[] [getUUID](#) ()
- byte[] [getMajor](#) ()
- byte[] [getMinor](#) ()
- byte [getTxPower](#) ()

*Obtiene el valor de Tx Power de la trama iBeacon.*

- byte[] [getLosBytes](#) ()
- byte[] [getAdvFlags](#) ()
- byte[] [getAdvHeader](#) ()
- byte[] [getCompanyID](#) ()
- byte [getiBeaconType](#) ()

*Obtiene el tipo de iBeacon.*

- byte [getiBeaconLength](#) ()

*Obtiene la longitud de los datos iBeacon.*

### 6.30.1. Descripción detallada

Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.

Definición en la línea 16 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

### 6.30.2. Documentación de constructores y destructores

#### **TramaIBeacon()**

```
com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.TramaIBeacon (
    byte[] bytes)
```

Constructor de la clase [TramaIBeacon](#).

##### Parámetros

<i>bytes</i>	Array de bytes que contiene los datos de la trama iBeacon.
--------------	--

Definición en la línea 35 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

### 6.30.3. Documentación de funciones miembro

#### **getAdvFlags()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getAdvFlags ()
```

Definición en la línea 110 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

#### **getAdvHeader()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getAdvHeader ()
```

Definición en la línea 119 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

#### **getCompanyID()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getCompanyID ()
```

Definición en la línea 128 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

**getiBeaconLength()**

```
byte com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getiBeaconLength ()
```

Obtiene la longitud de los datos iBeacon.

Devuelve

Un byte que representa la longitud del iBeacon.

Definición en la línea 146 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

**getiBeaconType()**

```
byte com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getiBeaconType ()
```

Obtiene el tipo de iBeacon.

Devuelve

Un byte que representa el tipo de iBeacon.

Definición en la línea 137 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

**getLosBytes()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getLosBytes ()
```

Definición en la línea 101 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

**getMajor()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getMajor ()
```

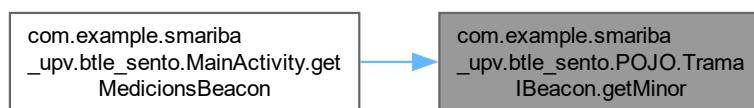
Definición en la línea 74 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

**getMinor()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getMinor ()
```

Definición en la línea 83 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

Gráfico de llamadas a esta función:



**getPrefijo()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getPrefijo ()
```

Definición en la línea 56 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/PO...](#)

**getTxPower()**

```
byte com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getTxPower ()
```

Obtiene el valor de Tx Power de la trama iBeacon.

**Devuelve**

Un byte que representa el valor de Tx Power.

Definición en la línea 92 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/PO...](#)

**getUUID()**

```
byte[] com.example.smariba_upv.btle_sento.POJO.TramaIBeacon.getUUID ()
```

Definición en la línea 65 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/PO...](#)

La documentación de esta clase está generada del siguiente archivo:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramaIBeacon.java](#)

### 6.31. Referencia de la clase com.example.smariba\_upv.airflow.POJO.User

Diagrama de colaboración de com.example.smariba\_upv.airflow.POJO.User:

com.example.smariba_upv.airflow.POJO.User
+ User(String nombre, String apellidos, String email, String telefono, String contrasenya) + User(String email, String contrasenya) + User(int id, String nombre, String apellidos, String email, String telefono) + String getNombre() + void setNombre(String nombre) + String getApellidos() + void setApellidos(String apellidos) + String getEmail() + void setEmail(String email) + String getTelefono() + void setTelefono(String telefono) + String getContrasenya() + void setContrasenya (String contrasenya) + int getId() + void setId(int id)

#### Métodos públicos

- [User](#) (String nombre, String apellidos, String email, String telefono, String contrasenya)  
*Constructor de la clase User.*
- [User](#) (String email, String contrasenya)  
*Constructor de la clase User.*
- [User](#) (int id, String nombre, String apellidos, String email, String telefono)  
*Constructor de la clase User.*

- String `getNombre ()`  
`getNombre() => Texto:nombre`
- void `setNombre (String nombre)`  
`Texto:nombre => setNombre()`
- String `getApellidos ()`  
`getApellidos() => Texto:apellidos`
- void `setApellidos (String apellidos)`  
`Texto:apellidos => setApellidos()`
- String `getEmail ()`  
`getEmail() => Texto:email`
- void `setEmail (String email)`  
`Texto:email => setEmail()`
- String `getTelefono ()`  
`getTelefono() => Texto:telefono`
- void `setTelefono (String telefono)`  
`Texto:telefono => setTelefono()`
- String `getContrasenya ()`  
`getContrasenya() => Texto:contrasenya`
- void `setContrasenya (String contrasenya)`  
`Texto:contrasenya => setContrasenya()`
- int `getId ()`  
`getId() => N:id`
- void `setId (int id)`  
`N:id => setId()`

### 6.31.1. Descripción detallada

Definición en la línea 3 del archivo [User.java](#).

### 6.31.2. Documentación de constructores y destructores

#### User() [1/3]

```
com.example.smariba_upv.airflow.POJO.User.User (
    String nombre,
    String apellidos,
    String email,
    String telefono,
    String contrasenya)
```

Constructor de la clase [User](#).

##### Parámetros

<code>nombre</code>	Nombre del usuario
<code>apellidos</code>	Apellidos del usuario
<code>email</code>	Correo electrónico del usuario
<code>telefono</code>	Teléfono del usuario
<code>contrasenya</code>	Contraseña del usuario Texto:nombre, Texto:apellidos, Texto:email, Texto:telefono, Texto:contrasenya => <a href="#">User()</a>

Definición en la línea 20 del archivo [User.java](#).

**User() [2/3]**

```
com.example.smariba_upv.airflow.POJO.User.User (
    String email,
    String contrasenya)
```

Constructor de la clase [User](#).

**Parámetros**

<i>email</i>	Correo electrónico del usuario
<i>contrasenya</i>	Contraseña del usuario Texto: <i>email</i> , Texto: <i>contrasenya</i> => <a href="#">User()</a>

Definición en la línea [34](#) del archivo [User.java](#).

**User() [3/3]**

```
com.example.smariba_upv.airflow.POJO.User.User (
    int id,
    String nombre,
    String apellidos,
    String email,
    String telefono)
```

Constructor de la clase [User](#).

**Parámetros**

<i>id</i>	Identificador del usuario
<i>nombre</i>	Nombre del usuario
<i>apellidos</i>	Apellidos del usuario
<i>email</i>	Correo electrónico del usuario
<i>telefono</i>	Teléfono del usuario Entero: <i>id</i> , Texto: <i>nombre</i> , Texto: <i>apellidos</i> , Texto: <i>email</i> , Texto: <i>telefono</i> => <a href="#">User()</a>

Definición en la línea [48](#) del archivo [User.java](#).

**6.31.3. Documentación de funciones miembro****getApellidos()**

```
String com.example.smariba_upv.airflow.POJO.User.getApellidos ()
getApellidos() => Texto:apellidos
```

Devuelve

Definición en la línea [75](#) del archivo [User.java](#).

Gráfico de llamadas a esta función:



### **getContrasenya()**

```
String com.example.smariba_upv.airflow.POJO.User.getContrasenya ()
```

**getContrasenya() => Texto:contrasenya**

**Devuelve**

Definición en la línea 117 del archivo [User.java](#).

### **getEmail()**

```
String com.example.smariba_upv.airflow.POJO.User.getEmail ()
```

**getEmail() => Texto:email**

**Devuelve**

Definición en la línea 89 del archivo [User.java](#).

Gráfico de llamadas a esta función:



### **getId()**

```
int com.example.smariba_upv.airflow.POJO.User.getId ()
```

**getId() => N:id**

**Devuelve**

Definición en la línea 131 del archivo [User.java](#).

Gráfico de llamadas a esta función:



**getNombre()**

```
String com.example.smariba_upv.airflow.POJO.User.getNombre ()
```

**getNombre() => Texto:nombre**

**Devuelve**

Definición en la línea 60 del archivo [User.java](#).

Gráfico de llamadas a esta función:

**getTelefono()**

```
String com.example.smariba_upv.airflow.POJO.User.getTelefono ()
```

**getTelefono() => Texto:telefono**

**Devuelve**

Definición en la línea 103 del archivo [User.java](#).

Gráfico de llamadas a esta función:

**setApellidos()**

```
void com.example.smariba_upv.airflow.POJO.User.setApellidos (
    String apellidos)
```

**Texto:apellidos => setApellidos()**

**Parámetros**

<i>apellidos</i>	<input type="text"/>
------------------	----------------------

Definición en la línea [82](#) del archivo [User.java](#).

**setContrasenya()**

```
void com.example.smariba_upv.airflow.POJO.User.setContrasenya (
    String contrasenya)
```

Texto:contrasenya => [setContrasenya\(\)](#)

**Parámetros**

<i>contrasenya</i>	<input type="text"/>
--------------------	----------------------

Definición en la línea [124](#) del archivo [User.java](#).

**setEmail()**

```
void com.example.smariba_upv.airflow.POJO.User.setEmail (
    String email)
```

Texto:email => [setEmail\(\)](#)

**Parámetros**

<i>email</i>	<input type="text"/>
--------------	----------------------

Definición en la línea [96](#) del archivo [User.java](#).

**setId()**

```
void com.example.smariba_upv.airflow.POJO.User.setId (
    int id)
```

N:id => [setId\(\)](#)

**Parámetros**

<i>id</i>	<input type="text"/>
-----------	----------------------

Definición en la línea [138](#) del archivo [User.java](#).

**setNombre()**

```
void com.example.smariba_upv.airflow.POJO.User.setNombre (
    String nombre)
```

Texto:nombre => [setNombre\(\)](#)

**Parámetros**

<i>nombre</i>	<input type="text"/>
---------------	----------------------

Definición en la línea 68 del archivo [User.java](#).

**setTelefono()**

```
void com.example.smariba_upv.airflow.POJO.User.setTelefono (
    String telefono)
```

Texto:telefono => [setTelefono\(\)](#)

**Parámetros**

<i>telefono</i>	<input type="text"/>
-----------------	----------------------

Definición en la línea 110 del archivo [User.java](#).

La documentación de esta clase está generada del siguiente archivo:

- [User.java](#)

## 6.32. Referencia de la clase com.example.smariba\_upv.airflow.LOGIC.Utilidades

Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.

Diagrama de colaboración de com.example.smariba\_upv.airflow.LOGIC.Utilidades:

com.example.smariba_upv.airflow.LOGIC.Utilidades
<ul style="list-style-type: none"> <li>+ String getFechaActual()</li> <li>+ static byte[] stringToBytes (String texto)</li> <li>+ static UUID stringToUUID (String uuid)</li> <li>+ static String uuidToString (UUID uuid)</li> <li>+ static String uuidToHexString (UUID uuid)</li> <li>+ static String bytesToString (byte[] bytes)</li> <li>+ static byte[] dosLongToBytes (long masSignificativos, long menosSignificativos)</li> <li>+ static int bytesToInt (byte[] bytes)</li> <li>+ static long bytesToLong (byte[] bytes)</li> <li>+ static int bytesToIntOK (byte[] bytes)</li> <li>+ static String bytesToHex String(byte[] bytes)</li> </ul>

## Métodos públicos

- String [getFechaActual \(\)](#)

## Métodos públicos estáticos

- static byte[] [stringToBytes](#) (String texto)
- static UUID [stringToUUID](#) (String uuid)
 

*Convierte un string de 16 caracteres en un objeto UUID.*
- static String [uuidToString](#) (UUID uuid)
 

*Convierte un objeto UUID en un string.*
- static String [uuidToHexString](#) (UUID uuid)
 

*Convierte un objeto UUID en un string hexadecimal.*
- static String [bytesToString](#) (byte[] bytes)

*Convierte un array de bytes en un string.*

- static byte[] dosLongToBytes (long masSignificativos, long menosSignificativos)
- static int bytesToInt (byte[] bytes)

*Convierte un array de bytes en un entero.*

- static long bytesToLong (byte[] bytes)

*Convierte un array de bytes en un valor long.*

- static int bytesToIntOK (byte[] bytes)

*Convierte un array de bytes en un entero, con manejo de signos.*

- static String bytesToHexString (byte[] bytes)

*Convierte un array de bytes en un string hexadecimal.*

### 6.32.1. Descripción detallada

Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.

Definición en la línea 19 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflo](#)

### 6.32.2. Documentación de funciones miembro

#### bytesToHexString()

```
static String com.example.smariba_upv.airflow.LOGIC.Utilidades.bytesToHexString (
    byte[] bytes) [static]
```

Convierte un array de bytes en un string hexadecimal.

##### Parámetros

<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

##### Devuelve

El string hexadecimal generado a partir del array de bytes.

Definición en la línea 163 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflo](#)

Gráfico de llamadas a esta función:



#### bytesToInt()

```
static int com.example.smariba_upv.airflow.LOGIC.Utilidades.bytesToInt (
    byte[] bytes) [static]
```

Convierte un array de bytes en un entero.

## Parámetros

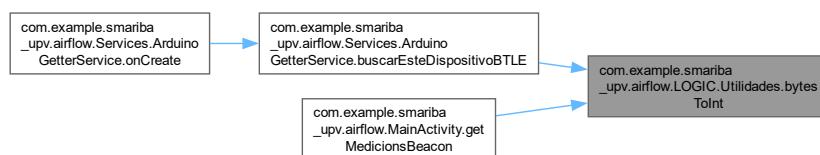
<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

## Devuelve

El valor entero resultante de la conversión.

Definición en la línea 112 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

Gráfico de llamadas a esta función:



## bytesToIntOK()

```
static int com.example.smariba_upv.airflow.LOGIC.Utilidades.bytesToIntOK (
    byte[] bytes) [static]
```

Convierte un array de bytes en un entero, con manejo de signos.

## Parámetros

<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

## Devuelve

El valor entero resultante de la conversión.

## Excepciones

<i>Error</i>	Si el array de bytes es demasiado largo para un entero.
--------------	---

Definición en la línea 135 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airfl](#)

## bytesToLong()

```
static long com.example.smariba_upv.airflow.LOGIC.Utilidades.bytesToLong (
    byte[] bytes) [static]
```

Convierte un array de bytes en un valor `long`.

**Parámetros**

<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El valor `long` resultante de la conversión.

Definición en la línea 123 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:

**bytesToString()**

```
static String com.example.smariba_upv.airflow.LOGIC.Utilidades.bytesToString (
    byte[] bytes) [static]
```

Convierte un array de bytes en un string.

**Parámetros**

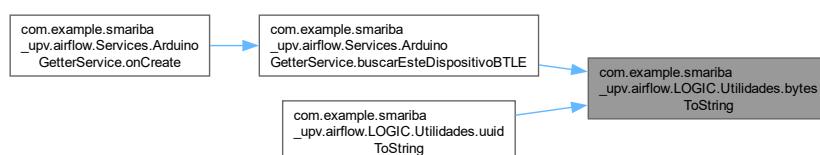
<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El string generado a partir del array de bytes.

Definición en la línea 78 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:

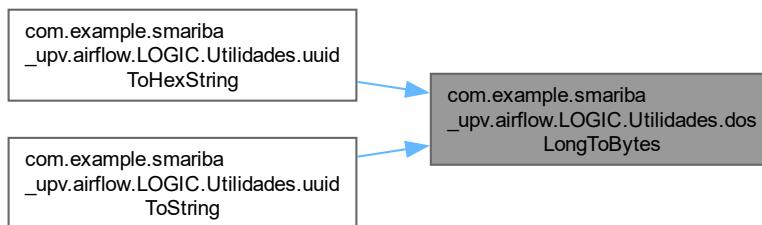


### **dosLongToBytes()**

```
static byte[] com.example.smariba_upv.airflow.LOGIC.Utilidades.dosLongToBytes (
    long masSignificativos,
    long menosSignificativos) [static]
```

Definición en la línea 98 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflo](#)

Gráfico de llamadas a esta función:



### **getFechaActual()**

```
String com.example.smariba_upv.airflow.LOGIC.Utilidades.getFechaActual ()
```

Definición en la línea 176 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflo](#)

### **stringToBytes()**

```
static byte[] com.example.smariba_upv.airflow.LOGIC.Utilidades.stringToBytes (
    String texto) [static]
```

Definición en la línea 28 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflo](#)

### **stringToUUID()**

```
static UUID com.example.smariba_upv.airflow.LOGIC.Utilidades.stringToUUID (
    String uuid) [static]
```

Convierte un string de 16 caracteres en un objeto UUID.

#### **Parámetros**

<i>uuid</i>	El string que se desea convertir a UUID.
-------------	--

#### **Devuelve**

El objeto UUID generado a partir del string.

### Excepciones

<b>Error</b>	Si el string no tiene exactamente 16 caracteres.
--------------	--

Definición en la línea 40 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflo

Gráfico de llamadas de esta función:



### uuidToString()

```
static String com.example.smariba_upv.airflow.LOGIC.Utilidades.uuidToString (
    UUID uuid) [static]
```

Convierte un objeto UUID en un string hexadecimal.

#### Parámetros

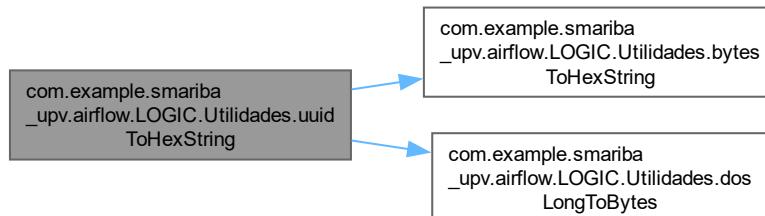
<b>uuid</b>	El objeto UUID a convertir.
-------------	-----------------------------

#### Devuelve

El string hexadecimal que representa al UUID.

Definición en la línea 67 del archivo Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflo

Gráfico de llamadas de esta función:



### uuidToHexString()

```
static String com.example.smariba_upv.airflow.LOGIC.Utilidades.uuidToHexString (
    UUID uuid) [static]
```

Convierte un objeto UUID en un string.

**Parámetros**

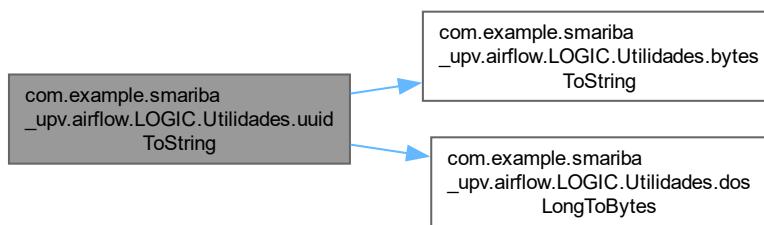
<i>uuid</i>	El objeto UUID a convertir.
-------------	-----------------------------

**Devuelve**

El string que representa al UUID.

Definición en la línea 56 del archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/LOGIC/Utilidades.java](#)

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/LOGIC/Utilidades.java](#)

### 6.33. Referencia de la clase com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades

Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.

Diagrama de colaboración de com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades:

com.example.smariba_upv.btle_sento.LOGIC.Utilidades
+ static byte[] stringToBytes (String texto) + static UUID stringToUUID (String uuid) + static String uuidToString (UUID uuid) + static String uuidToHexString (UUID uuid) + static String bytesToString (byte[] bytes) + static byte[] dosLongToBytes (long masSignificativos, long menosSignificativos) + static int bytesToInt (byte[] bytes) + static long bytesToLong (byte[] bytes) + static int bytesToIntOK (byte[] bytes) + static String bytesToHex String(byte[] bytes)

## Métodos públicos estáticos

- static byte[] **stringToBytes** (String texto)  
■ static UUID **stringToUUID** (String uuid)  
*Convierte un string de 16 caracteres en un objeto UUID.*
- static String **uuidToString** (UUID uuid)  
*Convierte un objeto UUID en un string.*
- static String **uuidToHexString** (UUID uuid)  
*Convierte un objeto UUID en un string hexadecimal.*
- static String **bytesToString** (byte[] bytes)  
*Convierte un array de bytes en un string.*
- static byte[] **dosLongToBytes** (long masSignificativos, long menosSignificativos)  
■ static int **bytesToInt** (byte[] bytes)  
*Convierte un array de bytes en un entero.*
- static long **bytesToLong** (byte[] bytes)  
*Convierte un array de bytes en un valor long.*

- static int **bytesToIntOK** (byte[ ] bytes)
 

*Convierte un array de bytes en un entero, con manejo de signos.*
- static String **bytesToHexString** (byte[ ] bytes)
 

*Convierte un array de bytes en un string hexadecimal.*

### 6.33.1. Descripción detallada

Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.

Definición en la línea 18 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

### 6.33.2. Documentación de funciones miembro

#### **bytesToHexString()**

```
static String com.example.smariba_upv.btle_sento.LOGIC.Utilidades.bytesToHexString (
    byte[ ] bytes) [static]
```

Convierte un array de bytes en un string hexadecimal.

##### Parámetros

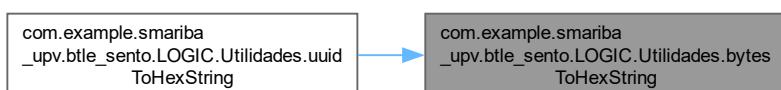
<b>bytes</b>	El array de bytes a convertir.
--------------	--------------------------------

##### Devuelve

El string hexadecimal generado a partir del array de bytes.

Definición en la línea 162 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:



#### **bytesToInt()**

```
static int com.example.smariba_upv.btle_sento.LOGIC.Utilidades.bytesToInt (
    byte[ ] bytes) [static]
```

Convierte un array de bytes en un entero.

**Parámetros**

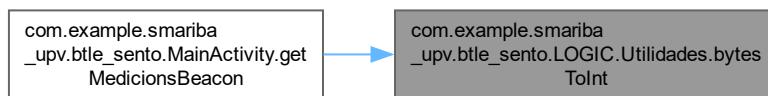
<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El valor entero resultante de la conversión.

Definición en la línea 111 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:

**bytesToIntOK()**

```
static int com.example.smariba_upv.btle_sento.LOGIC.Utilidades.bytesToIntOK (byte[] bytes) [static]
```

Convierte un array de bytes en un entero, con manejo de signos.

**Parámetros**

<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El valor entero resultante de la conversión.

**Excepciones**

<i>Error</i>	Si el array de bytes es demasiado largo para un entero.
--------------	---

Definición en la línea 134 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

**bytesToLong()**

```
static long com.example.smariba_upv.btle_sento.LOGIC.Utilidades.bytesToLong (byte[] bytes) [static]
```

Convierte un array de bytes en un valor `long`.

**Parámetros**

<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El valor `long` resultante de la conversión.

Definición en la línea 122 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:

**bytesToString()**

```
static String com.example.smariba_upv.btle_sento.LOGIC.Utilidades.bytesToString (
    byte[ ] bytes) [static]
```

Convierte un array de bytes en un string.

**Parámetros**

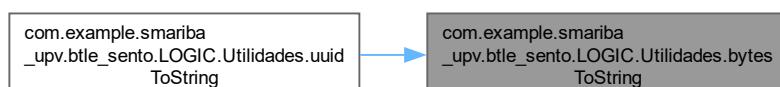
<i>bytes</i>	El array de bytes a convertir.
--------------	--------------------------------

**Devuelve**

El string generado a partir del array de bytes.

Definición en la línea 77 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:

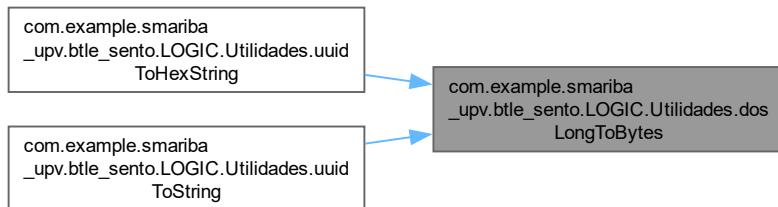


### **dosLongToBytes()**

```
static byte[ ] com.example.smariba_upv.btle_sento.LOGIC.Utilidades.dosLongToBytes ( long masSignificativos, long menosSignificativos) [static]
```

Definición en la línea 97 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas a esta función:



### **stringToBytes()**

```
static byte[ ] com.example.smariba_upv.btle_sento.LOGIC.Utilidades.stringToBytes ( String texto) [static]
```

Definición en la línea 27 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

### **stringToUUID()**

```
static UUID com.example.smariba_upv.btle_sento.LOGIC.Utilidades.stringToUUID ( String uuid) [static]
```

Convierte un string de 16 caracteres en un objeto UUID.

#### **Parámetros**

<i>uuid</i>	El string que se desea convertir a UUID.
-------------	--

#### **Devuelve**

El objeto UUID generado a partir del string.

#### **Excepciones**

<i>Error</i>	Si el string no tiene exactamente 16 caracteres.
--------------	--

Definición en la línea 39 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas de esta función:



### **uuidToString()**

```
static String com.example.smariba_upv.btle_sento.LOGIC.Utilidades.uuidToString (
    UUID uuid) [static]
```

Convierte un objeto UUID en un string hexadecimal.

#### Parámetros

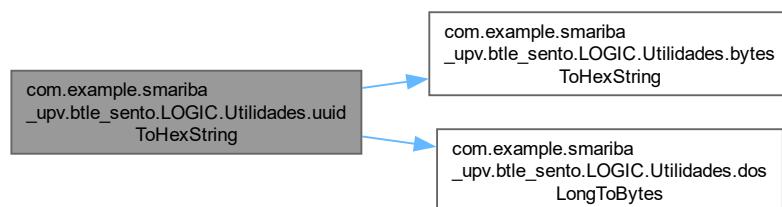
<i>uuid</i>	El objeto UUID a convertir.
-------------	-----------------------------

#### Devuelve

El string hexadecimal que representa al UUID.

Definición en la línea 66 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas de esta función:



### **uuidToString()**

```
static String com.example.smariba_upv.btle_sento.LOGIC.Utilidades.uuidToString (
    UUID uuid) [static]
```

Convierte un objeto UUID en un string.

**Parámetros**

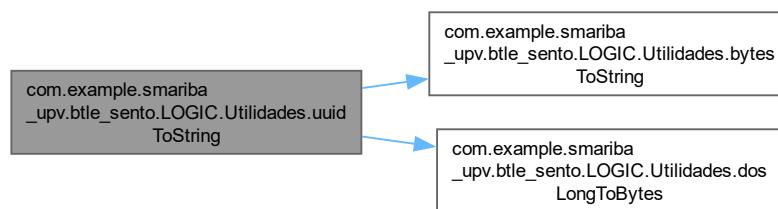
<i>uuid</i>	El objeto UUID a convertir.
-------------	-----------------------------

**Devuelve**

El string que representa al UUID.

Definición en la línea 55 del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

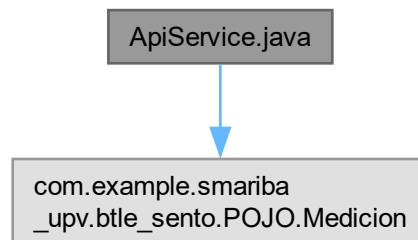
## 7. Documentación de archivos

### 7.1. Referencia del archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/ApiService.java](#)

Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST.

```
import com.example.smariba_upv.btle_sento.POJO.Medicion;
```

Gráfico de dependencias incluidas en [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/ApiService.java](#):



## Clases

- interface [com.example.smariba\\_upv.btle\\_sento.API.ApiService](#)

*Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.*

## Paquetes

- package [com.example.smariba\\_upv.btle\\_sento.API](#)

### 7.1.1. Descripción detallada

Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/ ApiService.java](#)

## 7.2. [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API/ ApiService.java](#)

[Ir a la documentación de este archivo.](#)

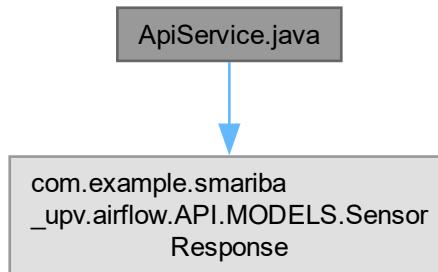
```
00001
00002 package com.example.smariba_upv.btle_sento.API;
00003
00004 import com.example.smariba_upv.btle_sento.POJO.Medicion;
00005 import retrofit2.Call;
00006 import retrofit2.http.Body;
00007 import retrofit2.http.GET;
00008 import retrofit2.http.POST;
00009
00010
00011 public interface ApiService {
00012
00013     @GET("/ping")
00014     Call<Object> ping(); // Aquí puedes usar una clase específica en lugar de Object para mapear la
00015     // respuesta
00016
00017     @POST("/insertar")
00018     Call<Void> insertarMedicion(@Body Medicion medicion);
00019
00020 }
```

### 7.3. Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow- Android/app/src/main/java/com/example/smariba\_upv/airflow/API/ApiService.java

Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST.

```
import com.example.smariba_upv.airflow.API.MODELS.SensorResponse;
```

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/ApiService.java:



#### Clases

- interface [com.example.smariba\\_upv.airflow.API.ApiService](#)

*Interfaz que define las operaciones que se realizarán contra el servidor usando Retrofit.*

#### Paquetes

- package [com.example.smariba\\_upv.airflow.API](#)

#### 7.3.1. Descripción detallada

Interfaz para la API REST utilizada en la aplicación, donde se definen las llamadas GET y POST.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/API/ApiS](#)

## 7.4. Desktop/Clases/Proyecto Bio/AirFlow-

**Android/app/src/main/java/com/example/smariba\_upv/airflow/API/ApiService.java**

[Ir a la documentación de este archivo.](#)

```

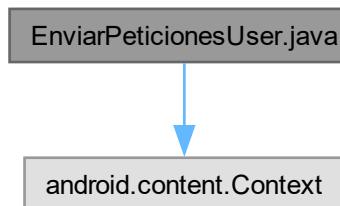
00001
00008 package com.example.smariba_upv.airflow.API;
00009
00010 import com.example.smariba_upv.airflow.API.MODELS.SensorResponse;
00011 import com.example.smariba_upv.airflow.POJO.Medicion;
00012
00013 import com.example.smariba_upv.airflow.API.MODELS.SensorRequest;
00014 import com.example.smariba_upv.airflow.POJO.SensorObject;
00015 import com.example.smariba_upv.airflow.POJO.User;
00016
00017
00018 import java.util.List;
00019
00020 import okhttp3.ResponseBody;
00021 import retrofit2.Call;
00022 import retrofit2.http.Body;
00023 import retrofit2.http.GET;
00024 import retrofit2.http.POST;
00025 import retrofit2.http.PUT;
00026 import retrofit2.http.Query;
00027
00028
00032 public interface ApiService {
00033
00039     @POST("/usuarios/login") // Si usas un prefijo en el servidor
00040         Call<User> login(@Body User user);
00041
00042     @PUT("/usuarios/editUsuario")
00043         Call<User> editUsuario(@Body User user);
00044     @POST("/insertar")
00045         Call<Void> insertarMedicion(@Body Medicion medicion);
00046
00047     @POST("/usuarios/registrar-sensor") // Reemplaza con la ruta correcta de tu endpoint
00048         Call<SensorRequest> registrarSensor(@Body SensorRequest sensorRequest);
00049     //actualizar sensor
00050     @PUT("/usuarios/actualizar-sensor")
00051         Call<ResponseBody> actualizarSensor(@Body SensorRequest sensorRequest);
00052
00053     // ApiService.java
00054     @GET("/usuarios/mis-sensores")
00055         Call<List<SensorResponse>> getMisSensores(@Query("id_usuario") int id_usuario);
00056 }
00057
00058
00059
00060

```

## 7.5. Referencia del archivo EnviarPeticionesUser.java

```
import android.content.Context;
```

Gráfico de dependencias incluidas en EnviarPeticionesUser.java:



## Clases

- class com.example.smariba\_upv.airflow.API.EnviarPeticionesUser

## Paquetes

- package com.example.smariba\_upv.airflow.API

## 7.6. EnviarPeticionesUser.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.API;
00002
00003 import android.content.Context;
00004 import android.content.Intent;
00005 import android.content.SharedPreferences;
00006 import android.util.Log;
00007
00008 import com.example.smariba_upv.airflow.API.MODELS.SensorResponse;
00009 import com.example.smariba_upv.airflow.POJO.SensorObject;
00010 import com.example.smariba_upv.airflow.API.MODELS.SensorRequest;
00011 import com.example.smariba_upv.airflow.POJO.User;
00012 import com.example.smariba_upv.airflow.PRESENTACION.LandActivity;
00013 import com.google.gson.Gson;
00014
00015 import java.io.IOException;
00016 import java.util.ArrayList;
00017 import java.util.List;
00018
00019 import okhttp3.ResponseBody;
00020 import retrofit2.Call;
00021 import retrofit2.Callback;
00022 import retrofit2.Response;
00023
00024
00025 public class EnviarPeticionesUser {
00030     private static final String TAG = "EnviarPeticionesUser";
00031     private Context context;
00032
00039     public EnviarPeticionesUser(Context context) {
00040         this.context = context;
00041     }
00042
00046     public EnviarPeticionesUser() {
00047     }
00055     public boolean login(String email, String password) {
00056         final boolean[] loginSuccessful = {false};
00057
00058         RetrofitClient.getApiService().login(new User(email, password)).enqueue(new Callback<User>() {
00059             @Override
00060             public void onResponse(Call<User> call, Response<User> response) {
00061                 if (response.isSuccessful()) {
00062                     User user = response.body();
00063                     Log.d(TAG, "Usuario logueado: " + user.getNombre());
00064
00065                     if (context != null) {
00066                         SharedPreferences sharedpreferences =
00067                             context.getSharedPreferences("MyAppPrefs", Context.MODE_PRIVATE);
00068                         SharedPreferences.Editor editor = sharedpreferences.edit();
00069                         editor.putInt("id", user.getId());
00070                         editor.putString("email", user.getEmail());
00071                         editor.putString("nombre", user.getNombre());
00072                         editor.putString("apellidos", user.getApellidos());
00073                         editor.putString("Telefono", user.getTelefono());
00074                         editor.putBoolean("isLoggedin", true);
00075                         editor.apply();
00076
00077                         Intent intent = new Intent(context, LandActivity.class);
00078                         context.startActivity(intent);
00079                     } else {
00080                         Log.e(TAG, "Context is null");
00081                     }
00082                     loginSuccessful[0] = true;
00083                 } else {
00084                     Log.e(TAG, "Error en la petición. Código de estado: " + response.code() + " - " +
00085                     response.message());
00086                     try {
00087                         Log.e(TAG, "Cuerpo del error: " + response.errorBody().string());
00088                     } catch (IOException e) {
00089                         Log.e(TAG, "Error al leer el cuerpo del error: " + e.getMessage());
00090                     }
00091                 }
00092             }
00093             public void onFailure(Call<User> call, Throwable t) {
00094                 Log.e(TAG, "Error en la petición: " + t.getMessage());
00095             }
00096         });
00097     }
00098 }
```

```

00086             } catch (IOException e) {
00087                 Log.e(TAG, "Error al leer el cuerpo del error", e);
00088             }
00089         }
00090     }
00091
00092     @Override
00093     public void onFailure(Call<User> call, Throwable t) {
00094         Log.e(TAG, "onFailure:", t);
00095     }
00096 });
00097
00098     return loginSuccessful[0];
00099 }
00100
00110     public void editUsuario(int id, String nombre, String apellidos, String email, String telefono) {
00111         if (id > 0 && nombre != null && email != null && telefono != null) {
00112             RetrofitClient.getApiService().editUsuario(new User(id, nombre, apellidos, email,
00113                     telefono)).enqueue(new Callback<User>() {
00114             @Override
00115             public void onResponse(Call<User> call, Response<User> response) {
00116                 if (response.isSuccessful()) {
00117                     Log.d(TAG, "Usuario actualizado correctamente");
00118                 } else {
00119                     Log.e(TAG, "Error en la petición. Código de estado: " + response.code() + " - "
00120                         + response.message());
00121                 }
00122             }
00123             @Override
00124             public void onFailure(Call<User> call, Throwable t) {
00125                 Log.e(TAG, "onFailure:", t);
00126             }
00127         } else {
00128             Log.e(TAG, "Faltan parámetros obligatorios");
00129         }
00130     }
00131
00138     public void registrarSensor(int id_usuario, SensorObject sensorObject) {
00139         String uidTrimmed = sensorObject.getUUID().trim();
00140         SensorRequest sensorRequest = new SensorRequest(id_usuario, sensorObject.getEstado(),
00141             sensorObject.getNum_ref(), uidTrimmed, sensorObject.getNombre(), sensorObject.isConexion(),
00142             sensorObject.getBateria());
00143         //ver datos del sensor
00144         Log.d(TAG, "Sensor: " + sensorObject.getId() + " - " + sensorObject.getEstado() + " - " +
00145             sensorObject.getNum_ref() + " - " + uidTrimmed + " - " + sensorObject.getNombre() + " - " +
00146             sensorObject.isConexion() + " - " + sensorObject.getBateria());
00147         RetrofitClient.getApiService().registrarSensor(sensorRequest).enqueue(new
00148             Callback<SensorRequest>() {
00149             @Override
00150             public void onResponse(Call<SensorRequest> call, Response<SensorRequest> response) {
00151                 if (response.isSuccessful()) {
00152                     Log.d(TAG, "Sensor registrado correctamente");
00153                 } else {
00154                     Log.e(TAG, "Error en la petición. Código de estado: " + response.code() + " - " +
00155                         response.message());
00156                 }
00157             }
00158         });
00159
00168     public void actualizarSensor(int id_sensor, String estado, boolean conexion, int bateria) {
00169         SensorRequest sensorRequest = new SensorRequest(id_sensor, estado, conexion, bateria);
00170         RetrofitClient.getApiService().actualizarSensor(sensorRequest).enqueue(new
00171             Callback<ResponseBody>() {
00172             @Override
00173             public void onResponse(Call<ResponseBody> call, Response<ResponseBody> response) {
00174                 if (response.isSuccessful()) {
00175                     Log.d(TAG, "Sensor actualizado correctamente");
00176                 } else {
00177                     Log.e(TAG, "Error en la petición. Código de estado: " + response.code() + " - " +
00178                         response.message());
00179                 }
00180             }
00181             @Override
00182             public void onFailure(Call<ResponseBody> call, Throwable t) {
00183                 Log.e(TAG, "onFailure:", t);
00184             }
00185         });
00186     }

```

```
00191 // EnviarPeticionesUser.java
00192 public void obtenerMisSensores(Context context) {
00193     SharedPreferences sharedpreferences = context.getSharedPreferences("MyAppPrefs",
00194         Context.MODE_PRIVATE);
00195     int id_usuario = sharedpreferences.getInt("id", 0); // Retrieve user ID
00196     Log.d(TAG, "ID del usuario: " + id_usuario);
00197     RetrofitClient.getApiService().getMisSensores(id_usuario).enqueue(new
00198         Callback<List<SensorResponse>>() {
00199             @Override
00200             public void onResponse(Call<List<SensorResponse>> call, Response<List<SensorResponse>>
00201                 response) {
00202                 if (response.isSuccessful()) {
00203                     List<SensorResponse> sensorResponses = response.body();
00204                     List<SensorObject> sensorObjects = new ArrayList<>();
00205                     // Map SensorResponse to SensorObject
00206                     for (SensorResponse sensorResponse : sensorResponses) {
00207                         SensorResponse.Sensor sensor = sensorResponse.getSensor();
00208                         SensorObject sensorObject = new SensorObject(
00209                             sensorResponse.getIdSensor(),
00210                             sensor.getNombre(),
00211                             sensor.getNumReferencia(),
00212                             sensor.getEstado(),
00213                             sensor.getUuid(),
00214                             sensor.isConexion(),
00215                             sensor.getBateria());
00216                         sensorObjects.add(sensorObject);
00217                     }
00218                     // Save sensorObjects in SharedPreferences as JSON
00219                     Gson gson = new Gson();
00220                     String jsonSensores = gson.toJson(sensorObjects);
00221                     SharedPreferences.Editor editor = sharedpreferences.edit();
00222                     editor.putString("sensores", jsonSensores);
00223                     editor.apply();
00224                 } else {
00225                     Log.e(TAG, "Error en la petición. Código de estado: " + response.code());
00226                 }
00227             }
00228         });
00229         @Override
00230         public void onFailure(Call<List<SensorResponse>> call, Throwable t) {
00231             Log.e(TAG, "onFailure:", t);
00232         }
00233     );
00234 );
00235 }
00236 }
00237 }
```

## 7.7. Referencia del archivo SensorRequest.java

Clase que representa un objeto SensorRequest.

### Clases

- class [com.example.smariba\\_upv.airflow.API.MODELS.SensorRequest](#)

### Paquetes

- package [com.example.smariba\\_upv.airflow.API.MODELS](#)

#### 7.7.1. Descripción detallada

Clase que representa un objeto SensorRequest.

**Versión**

1.0

**Fecha**

2024

Definición en el archivo [SensorRequest.java](#).

## 7.8. SensorRequest.java

[Ir a la documentación de este archivo.](#)

```

00001 package com.example.smariba_upv.airflow.API.MODELS;
00002
00008 public class SensorRequest {
00021     private int id_usuario;
00022     private int id_sensor;
00023     private String estado;
00024     private String num_referencia;
00025     private String uuid;
00026     private String nombre;
00027     private boolean conexion;
00028     private int bateria;
00029
00042     //N: id_usuario, id_sensor, estado, num_referencia, uuid, nombre, conexion, bateria =>
00043     SensorRequest()
00043     public SensorRequest(int id_usuario, int id_sensor, String estado, String num_referencia, String
00044         uuid, String nombre, boolean conexion, int bateria) {
00044         this.id_usuario = id_usuario;
00045         this.id_sensor = id_sensor;
00046         this.estado = estado;
00047         this.num_referencia = num_referencia;
00048         this.uuid = uuid.trim();
00049         this.nombre = nombre;
00050         this.conexion = conexion;
00051         this.bateria = bateria;
00052     }
00061     public SensorRequest(int idSensor, String estado, boolean conexion, int bateria) {
00062         this.id_sensor = idSensor;
00063         this.estado = estado;
00064         this.conexion = conexion;
00065         this.bateria = bateria;
00066     }
00078     public SensorRequest(int id_usuario, String estado, String num_referencia, String uuid, String
00079         nombre, boolean conexion, int bateria) {
00080         this.id_usuario = id_usuario;
00081         this.estado = estado;
00082         this.num_referencia = num_referencia;
00083         this.uuid = uuid.trim();
00084         this.nombre = nombre;
00085         this.conexion = conexion;
00086         this.bateria = bateria;
00087     }
00092     public int getIdUsuario() {
00093         return id_usuario;
00094     }
00099     public void setIdUsuario(int id_usuario) {
00100         this.id_usuario = id_usuario;
00101     }
00106     public int getIdSensor() {
00107         return id_sensor;
00108     }
00113     public void setIdSensor(int id_sensor) {
00114         this.id_sensor = id_sensor;
00115     }
00120     public String getEstado() {
00121         return estado;
00122     }
00127     public void setEstado(String estado) {
00128         this.estado = estado;
00129     }
00134     public String getNumReferencia() {
00135         return num_referencia;
00136     }
00141     public void setNumReferencia(String num_referencia) {

```

```

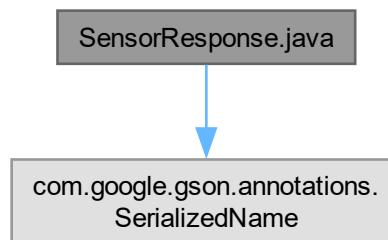
00142     this.num_referencia = num_referencia;
00143 }
00148 public String getUuid() {
00149     return uuid;
00150 }
00155 public void setUuid(String uuid) {
00156     this.uuid = uuid.trim();
00157 }
00162 public String getNombre() {
00163     return nombre;
00164 }
00169 public void setNombre(String nombre) {
00170     this.nombre = nombre;
00171 }
00176 public boolean isConexion() {
00177     return conexion;
00178 }
00183 public void setConexion(boolean conexion) {
00184     this.conexion = conexion;
00185 }
00190 public int getBateria() {
00191     return bateria;
00192 }
00197 public void setBateria(int bateria) {
00198     this.bateria = bateria;
00199 }
00200 }

```

## 7.9. Referencia del archivo SensorResponse.java

Clase que representa un objeto SensorResponse.

import com.google.gson.annotations.SerializedName;  
 Gráfico de dependencias incluidas en SensorResponse.java:



### Clases

- class [com.example.smariba\\_upv.airflow.API.MODELS.SensorResponse](#)  
*Clase que representa un objeto SensorResponse.*
- class [com.example.smariba\\_upv.airflow.API.MODELS.SensorResponse.Sensor](#)  
*Clase que representa un objeto Sensor.*

### Paquetes

- package [com.example.smariba\\_upv.airflow.API.MODELS](#)

### 7.9.1. Descripción detallada

Clase que representa un objeto SensorResponse.

#### Versión

1.0

Definición en el archivo [SensorResponse.java](#).

## 7.10. SensorResponse.java

[Ir a la documentación de este archivo.](#)

```
00001 // SensorResponse.java
00002 package com.example.smariba_upv.airflow.API.MODELS;
00003 import com.google.gson.annotations.SerializedName;
00014 public class SensorResponse {
00022     @SerializedName("id_usuario")
00023     private int idUsuario;
00024
00025     @SerializedName("id_sensor")
00026     private int idSensor;
00027
00028     @SerializedName("Sensor")
00029     private Sensor sensor;
00030
00031     // Getters and setters
00032     public int getIdUsuario() {
00033         return idUsuario;
00034     }
00035
00036     public void setIdUsuario(int idUsuario) {
00037         this.idUsuario = idUsuario;
00038     }
00039
00040     public int getIdSensor() {
00041         return idSensor;
00042     }
00043
00044     public void setIdSensor(int idSensor) {
00045         this.idSensor = idSensor;
00046     }
00047
00048     public Sensor getSensor() {
00049         return sensor;
00050     }
00051
00052     public void setSensor(Sensor sensor) {
00053         this.sensor = sensor;
00054     }
00059     public static class Sensor {
00070         @SerializedName("estado")
00071         private String estado;
00072
00073         @SerializedName("num_referencia")
00074         private String numReferencia;
00075
00076         @SerializedName("uuid")
00077         private String uuid;
00078
00079         @SerializedName("nombre")
00080         private String nombre;
00081
00082         @SerializedName("conexion")
00083         private boolean conexion;
00084
00085         @SerializedName("bateria")
00086         private int bateria;
00087
00088         // Getters and setters
00089         public String getEstado() {
00090             return estado;
00091         }
00092
00093         public void setEstado(String estado) {
00094             this.estado = estado;
```

```
00095      }
00096
00097     public String getNumReferencia() {
00098         return numReferencia;
00099     }
00100
00101     public void setNumReferencia(String numReferencia) {
00102         this.numReferencia = numReferencia;
00103     }
00104
00105     public String getUuid() {
00106         return uuid;
00107     }
00108
00109     public void setUuid(String uuid) {
00110         this.uuid = uuid;
00111     }
00112
00113     public String getNombre() {
00114         return nombre;
00115     }
00116
00117     public void setNombre(String nombre) {
00118         this.nombre = nombre;
00119     }
00120
00121     public boolean isConexion() {
00122         return conexion;
00123     }
00124
00125     public void setConexion(boolean conexion) {
00126         this.conexion = conexion;
00127     }
00128
00129     public int getBateria() {
00130         return bateria;
00131     }
00132
00133     public void setBateria(int bateria) {
00134         this.bateria = bateria;
00135     }
00136 }
00137 }
```

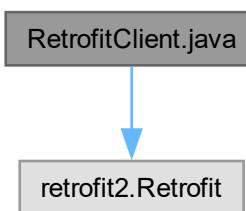
## 7.11. Referencia del archivo

AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/API/RetrofitClient.java

Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor.

```
import retrofit2.Retrofit;
```

Gráfico de dependencias incluidas en AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/API/RetrofitClient.java:



## Clases

- class [com.example.smariba\\_upv.btle\\_sento.API.RetrofitClient](#)

*Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.*

## Paquetes

- package [com.example.smariba\\_upv.btle\\_sento.API](#)

### 7.11.1. Descripción detallada

Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API.RetrofitClient.java](#)

## 7.12. [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/API.RetrofitClient.java](#)

[Ir a la documentación de este archivo.](#)

```
00001
00008 package com.example.smariba_upv.btle_sento.API;
00009
00010 import retrofit2.Retrofit;
00011 import retrofit2.converter.gson.GsonConverterFactory;
00012
00013
00017 public class RetrofitClient {
00018
00019
00023     public static Retrofit retrofit = null;
00024
00025
00034     public static ApiService getApiService() {
00035         if (retrofit == null) {
00036             // Construye la instancia de Retrofit con la URL base y el convertidor Gson
00037             retrofit = new Retrofit.Builder()
00038                 .baseUrl("http://192.168.1.19:3000/") // Cambia la IP por la de tu servidor
00039                 .addConverterFactory(GsonConverterFactory.create())
00040                 .build();
00041         }
00042         // Retorna la instancia de ApiService creada a partir de Retrofit
00043         return retrofit.create(ApiService.class);
00044     }
00045 }
```

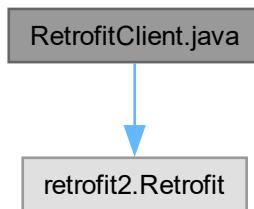
### 7.13. Referencia del archivo Desktop/Clases/Proyecto

Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/←  
RetrofitClient.java

Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor.

```
import retrofit2.Retrofit;
```

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/RetrofitClient.java:



#### Clases

- class [com.example.smariba\\_upv.airflow.API.RetrofitClient](#)

*Clase que proporciona la instancia de Retrofit configurada para realizar llamadas a la API REST.*

#### Paquetes

- package [com.example.smariba\\_upv.airflow.API](#)

#### 7.13.1. Descripción detallada

Clase encargada de gestionar la instancia de Retrofit para interactuar con el servidor.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/API/Retro](#)

## 7.14. Desktop/Clases/Proyecto

Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/API/←  
RetrofitClient.java

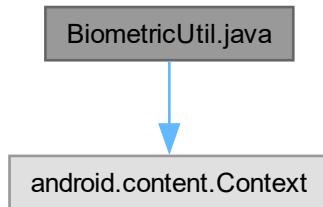
[Ir a la documentación de este archivo.](#)

```
00001
00008 package com.example.smariba_upv.airflow.API;
00009
00010 import retrofit2.Retrofit;
00011 import retrofit2.converter.gson.GsonConverterFactory;
00012
00013
00017 public class RetrofitClient {
00018
00019     private static final String BASE_URL = "http://10.237.30.68:3000/"; // Ensure this is correct
00020     private static Retrofit retrofit = null;
00021
00022     public static ApiService getApiService() {
00023         if (retrofit == null) {
00024             retrofit = new Retrofit.Builder()
00025                 .baseUrl(BASE_URL)
00026                 .addConverterFactory(GsonConverterFactory.create())
00027                 .build();
00028         }
00029         return retrofit.create(ApiService.class);
00030     }
00031 }
```

## 7.15. Referencia del archivo BiometricUtil.java

@autor: SentoMarcos

import android.content.Context;  
Gráfico de dependencias incluidas en BiometricUtil.java:



### Clases

- class [com.example.smariba\\_upv.airflow.LOGIC.BiometricUtil](#)  
*Clase que gestiona la autenticación biométrica.*
- interface [com.example.smariba\\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener](#)  
*Interfaz para gestionar los eventos de la autenticación biométrica.*

### Paquetes

- package [com.example.smariba\\_upv.airflow.LOGIC](#)

### 7.15.1. Descripción detallada

@autor: SentoMarcos

Clase que gestiona la autenticación biométrica

Definición en el archivo [BiometricUtil.java](#).

## 7.16. BiometricUtil.java

[Ir a la documentación de este archivo.](#)

```

00001
00006 package com.example.smariba_upv.airflow.LOGIC;
00007
00008 import android.content.Context;
00009 import android.widget.Toast;
00010
00011 import androidx.annotation.NonNull;
00012 import androidx.appcompat.app.AppCompatActivity;
00013 import androidx.biometric.BiometricManager;
00014 import androidx.biometric.BiometricPrompt;
00015 import androidx.core.content.ContextCompat;
00016
00017 import java.util.concurrent.Executor;
00022 public class BiometricUtil {
00023
00028     public interface BiometricAuthListener {
00029         void onAuthenticationSucceeded(BiometricPrompt.AuthenticationResult result);
00030         void onAuthenticationFailed();
00031         void onAuthenticationError(int errorCode, CharSequence errString);
00032     }
00039 //context:Context,listener:BiometricAuthListener ==> authenticate():void
00040     public static void authenticate(Context context, BiometricAuthListener listener) {
00041         Executor executor = ContextCompat.getMainExecutor(context);
00042         BiometricPrompt biometricPrompt = new BiometricPrompt((AppCompatActivity) context, executor,
00043             new BiometricPrompt.AuthenticationCallback() {
00043                 //errorCode:E,errString:CharSequence ==> onAuthenticationError():void
00044                 @Override
00045                 public void onAuthenticationError(int errorCode, @NonNull CharSequence errString) {
00046                     super.onAuthenticationError(errorCode, errString);
00047                     Toast.makeText(context, "Error de autenticación: " + errString,
00048                         Toast.LENGTH_SHORT).show();
00049                     listener.onAuthenticationError(errorCode, errString);
00050                 }
00050                 //result:AuthenticationResult ==> onAuthenticationSucceeded():void
00051                 @Override
00052                 public void onAuthenticationSucceeded(@NonNull BiometricPrompt.AuthenticationResult
00052 result) {
00053                     super.onAuthenticationSucceeded(result);
00054                     Toast.makeText(context, "Autenticación exitosa!", Toast.LENGTH_SHORT).show();
00055                     listener.onAuthenticationSucceeded(result);
00056                 }
00057                 //==> onAuthenticationFailed():void
00058                 @Override
00059                 public void onAuthenticationFailed() {
00060                     super.onAuthenticationFailed();
00061                     Toast.makeText(context, "Fallo en la autenticación", Toast.LENGTH_SHORT).show();
00062                     listener.onAuthenticationFailed();
00063                 }
00064             });
00065
00071     BiometricManager biometricManager = BiometricManager.from(context);
00072     switch (biometricManager.canAuthenticate(BiometricManager.Authenticators.BIOMETRIC_WEAK |
00072 BiometricManager.Authenticators.DEVICE_CREDENTIAL)) {
00073         case BiometricManager.BIOMETRIC_SUCCESS:
00074             BiometricPrompt.PromptInfo promptInfo = new BiometricPrompt.PromptInfo.Builder()
00075                 .setTitle("Inicio de sesión biométrico")
00076                 .setSubtitle("Inicie sesión con huella dactilar o reconocimiento facial")
00077                 .setAllowedAuthenticators(BiometricManager.Authenticators.BIOMETRIC_WEAK |
00077 BiometricManager.Authenticators.DEVICE_CREDENTIAL)
00078                 .build();
00079             biometricPrompt.authenticate(promptInfo);
00080             break;
00081         case BiometricManager.BIOMETRIC_ERROR_NO_HARDWARE:
00082             Toast.makeText(context, "Este dispositivo no tiene hardware biométrico",
00082             Toast.LENGTH_SHORT).show();
00083             break;
00084         case BiometricManager.BIOMETRIC_ERROR_HW_UNAVAILABLE:

```

```

00085         Toast.makeText(context, "Hardware biométrico no disponible",
00086             Toast.LENGTH_SHORT).show();
00087     case BiometricManager.BIOMETRIC_ERROR_NONE_ENROLLED:
00088         Toast.makeText(context, "No hay datos biométricos registrados",
00089             Toast.LENGTH_SHORT).show();
00090     break;
00091 }
00092 }
```

## 7.17. Referencia del archivo PeticionesUserUtil.java

import android.content.Context;

Gráfico de dependencias incluidas en PeticionesUserUtil.java:



### Clases

- class [com.example.smariba\\_upv.airflow.LOGIC.PeticionesUserUtil](#)

### Paquetes

- package [com.example.smariba\\_upv.airflow.LOGIC](#)

## 7.18. PeticionesUserUtil.java

[Ir a la documentación de este archivo.](#)

```

00001 package com.example.smariba_upv.airflow.LOGIC;
00002 import android.content.Context;
00003 import android.content.SharedPreferences;
00004 import android.util.Log;
00005
00006 import com.example.smariba_upv.airflow.API.EnviarPeticionesUser;
00007 import com.example.smariba_upv.airflow.POJO.SensorObject;
00008
00009 import org.json.JSONException;
00010 import org.json.JSONObject;
00011
00012 import java.util.regex.Pattern;
00013 import java.util.regex.Matcher;
00014
00015 public class PeticionesUserUtil {
00016
00017     private static final String TAG = "PeticionesUserUtil";
00018     // Texto:email, Texto:password, Context:context => login() => VoF
00019     public static boolean login(String email, String password, Context context) {
00020
00021 }
```

```

00028     EnviarPeticionesUser enviarPeticionesUser = new EnviarPeticionesUser(context);
00029     boolean res= enviarPeticionesUser.login(email, password);
00030     return res;
00031 }
00032 // N:id,Texto:nombre, Texto:apellidos, Texto:email, Texto:telefono, Texto:contrasenya,
00033 Context:context => editUsuario() => void
00034     public static void editUsuario(int id, String nombre, String apellidos, String email, String
00035 telefono, String contrasenya, Context context) {
00036     EnviarPeticionesUser enviarPeticionesUser = new EnviarPeticionesUser(context);
00037     enviarPeticionesUser.editUsuario(id, nombre, apellidos, email, telefono);
00038     Log.d(TAG, "editUsuario: ");
00039 }
00040     public static void registrarSensor(String jsonqr, Context context) {
00041     EnviarPeticionesUser enviarPeticionesUser = new EnviarPeticionesUser();
00042     try {
00043         JSONObject jsonObject = new JSONObject(jsonqr);
00044         String uuid = jsonObject.getString("uuid");
00045         String num_ref = jsonObject.getString("num_serie");
00046         SensorObject sensorObject = new SensorObject("none", num_ref, uuid, "NewSensor", false,
00047 );
00048         // recoger la id del usuario de shared preferences
00049         SharedPreferences sharedPreferences = context.getSharedPreferences("MyAppPrefs",
00050             Context.MODE_PRIVATE);
00051         int id = sharedPreferences.getInt("id", 0);
00052         enviarPeticionesUser.registrarSensor(id, sensorObject);
00053     } catch (JSONException e) {
00054         Log.e(TAG, "Error parsing JSON from QR code", e);
00055     }
00056 }
00057 }
```

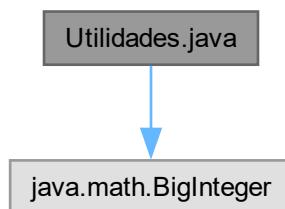
## 7.19. Referencia del archivo

### AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/LOGIC/Utilidades.java

Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos.

import java.math.BigInteger;

Gráfico de dependencias incluidas en AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/LOGIC/Utilidades.java:



## Clases

- class com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades

*Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.*

## Paquetes

- package com.example.smariba\_upv.btle\_sento.LOGIC

### 7.19.1. Descripción detallada

Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

## 7.20. [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/LOGIC/Utilidades.java](#)

[Ir a la documentación de este archivo.](#)

```

00001
00008 package com.example.smariba_upv.btle_sento.LOGIC;
00009
00010 import java.math.BigInteger;
00011 import java.nio.ByteBuffer;
00012 import java.util.UUID;
00013
00014
00018 public class Utilidades {
00019
00020
00027     public static byte[] stringToBytes(String texto) {
00028         return texto.getBytes();
00029     }
00030
00031
00039     public static UUID stringToUUID(String uuid) {
00040         if (uuid.length() != 16) {
00041             throw new Error("stringUUID: string no tiene 16 caracteres");
00042         }
00043         String masSignificativo = uuid.substring(0, 8);
00044         String menosSignificativo = uuid.substring(8, 16);
00045         return new UUID(Utilidades.bytesToLong(masSignificativo.getBytes()),
00046             Utilidades.bytesToLong(menosSignificativo.getBytes()));
00047     }
00048
00055     public static String uuidToString(UUID uuid) {
00056         return bytesToString(dosLongToBytes(uuid.getMostSignificantBits(),
00057             uuid.getLeastSignificantBits()));
00058
00059
00066     public static String uuidToHexString(UUID uuid) {
00067         return bytesToHexString(dosLongToBytes(uuid.getMostSignificantBits(),
00068             uuid.getLeastSignificantBits()));
00069
00070
00077     public static String bytesToString(byte[] bytes) {
00078         if (bytes == null) {
00079             return "";
00080         }
00081
00082         StringBuilder sb = new StringBuilder();
00083         for (byte b : bytes) {
00084             sb.append((char) b);

```

```

00085         }
00086         return sb.toString();
00087     }
00088
00089
00097     public static byte[] dosLongToBytes(long masSignificativos, long menosSignificativos) {
00098         ByteBuffer buffer = ByteBuffer.allocate(2 * Long.BYTES);
00099         buffer.putLong(masSignificativos);
00100         buffer.putLong(menosSignificativos);
00101         return buffer.array();
00102     }
00103
00104
00111     public static int bytesToInt(byte[] bytes) {
00112         return new BigInteger(bytes).intValue();
00113     }
00114
00115
00122     public static long bytesToLong(byte[] bytes) {
00123         return new BigInteger(bytes).longValue();
00124     }
00125
00126
00134     public static int bytesToIntOK(byte[] bytes) {
00135         if (bytes == null) {
00136             return 0;
00137         }
00138
00139         if (bytes.length > 4) {
00140             throw new Error("demasiados bytes para pasar a int");
00141         }
00142
00143         int res = 0;
00144         for (byte b : bytes) {
00145             res = (res << 8) + (b & 0xFF); // Asegurarse de obtener solo los 8 bits menos
00146             significativos de cada byte
00147         }
00148         if ((bytes[0] & 0x8) != 0) { // Si el número es negativo
00149             res = -(~(byte) res) - 1; // Realiza el complemento a 2
00150         }
00151
00152         return res;
00153     }
00154
00155
00162     public static String bytesToHexString(byte[] bytes) {
00163         if (bytes == null) {
00164             return "";
00165         }
00166
00167         StringBuilder sb = new StringBuilder();
00168         for (byte b : bytes) {
00169             sb.append(String.format("%02x", b));
00170             sb.append(':');
00171         }
00172         return sb.toString();
00173     }
00174
00175 } // class

```

**7.21. Referencia del archivo Desktop/Clases/Proyecto**

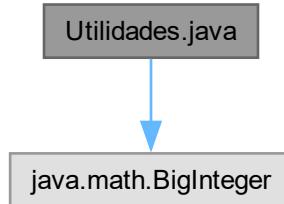
**Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/LOGIC/Utilidades.java**

Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos.

import java.math.BigInteger;

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba

\_upv/airflow/LOGIC/Utilidades.java:



## Clases

- class [com.example.smariba\\_upv.airflow.LOGIC.Utilidades](#)

*Clase con métodos estáticos para la manipulación de datos en forma de bytes, cadenas, y UUIDs.*

## Paquetes

- package [com.example.smariba\\_upv.airflow.LOGIC](#)

### 7.21.1. Descripción detallada

Clase con métodos utilitarios para manipulación de bytes, UUIDs y conversión de tipos.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/LOGIC/U](#)

## 7.22. Desktop/Clases/Proyecto

Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/LOGIC/Utilidades.java

[Ir a la documentación de este archivo.](#)

```

00001
00008 package com.example.smariba_upv.airflow.LOGIC;
00009
00010 import java.math.BigInteger;
00011 import java.nio.ByteBuffer;
00012 import java.util.Date;
00013 import java.util.UUID;
00014
00015
00019 public class Utilidades {
00020
00021
00028     public static byte[] stringToBytes(String texto) {
00029         return texto.getBytes();
00030     }
00031
00032
00040     public static UUID stringToUUID(String uuid) {
00041         if (uuid.length() != 16) {
00042             throw new Error("stringUUID: string no tiene 16 caracteres");
00043         }
00044         String masSignificativo = uuid.substring(0, 8);
00045         String menosSignificativo = uuid.substring(8, 16);
00046         return new UUID(Utilidades.bytesToLong(masSignificativo.getBytes()),
00047                         Utilidades.bytesToLong(menosSignificativo.getBytes()));
00047     }
00048
00049
00056     public static String uuidToString(UUID uuid) {
00057         return bytesToString(dosLongToBytes(uuid.getMostSignificantBits(),
00058                                         uuid.getLeastSignificantBits()));
00058     }
00059
00060
00067     public static String uuidToHexString(UUID uuid) {
00068         return bytesToHexString(dosLongToBytes(uuid.getMostSignificantBits(),
00069                                         uuid.getLeastSignificantBits()));
00069     }
00070
00071
00078     public static String bytesToString(byte[] bytes) {
00079         if (bytes == null) {
00080             return "";
00081         }
00082
00083         StringBuilder sb = new StringBuilder();
00084         for (byte b : bytes) {
00085             sb.append((char) b);
00086         }
00087         return sb.toString();
00088     }
00089
00090
00098     public static byte[] dosLongToBytes(long masSignificativos, long menosSignificativos) {
00099         ByteBuffer buffer = ByteBuffer.allocate(2 * Long.BYTES);
00100         buffer.putLong(masSignificativos);
00101         buffer.putLong(menosSignificativos);
00102         return buffer.array();
00103     }
00104
00105
00112     public static int bytesToInt(byte[] bytes) {
00113         return new BigInteger(bytes).intValue();
00114     }
00115
00116
00123     public static long bytesToLong(byte[] bytes) {
00124         return new BigInteger(bytes).longValue();
00125     }
00126
00127
00135     public static int bytesToIntOK(byte[] bytes) {
00136         if (bytes == null) {
00137             return 0;
00138         }
00139
00140         if (bytes.length > 4) {
00141             throw new Error("demasiados bytes para pasar a int");
00142         }

```

```

00143     int res = 0;
00144     for (byte b : bytes) {
00145         res = (res << 8) + (b & 0xFF); // Asegurarse de obtener solo los 8 bits menos
00146         significativos de cada byte
00147     }
00148
00149     if ((bytes[0] & 0x8) != 0) { // Si el número es negativo
00150         res = -(~(byte) res) - 1; // Realiza el complemento a 2
00151     }
00152
00153     return res;
00154 }
00155
00156
00157
00158
00159
00160
00161
00162
00163     public static String bytesToHexString(byte[] bytes) {
00164         if (bytes == null) {
00165             return "";
00166         }
00167
00168         StringBuilder sb = new StringBuilder();
00169         for (byte b : bytes) {
00170             sb.append(String.format("%02x", b));
00171             sb.append(':');
00172         }
00173         return sb.toString();
00174     }
00175
00176     public String getFechaActual(){
00177         return new Date().toString();
00178     }
00179
00180
00181 } // class

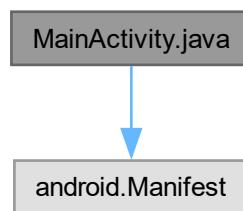
```

## 7.23. Referencia del archivo AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/MainActivity.java

Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE.

import android.Manifest;

Gráfico de dependencias incluidas en AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/MainActivity.java:



### Clases

- class com.example.smariba\_upv.btle\_sento.MainActivity

*La clase principal que gestiona la interfaz y las funciones BTLE.*

### Paquetes

- package com.example.smariba\_upv.btle\_sento

### 7.23.1. Descripción detallada

Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE.

Versión

1.0

Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/MainActivity.java](#)

## 7.24. AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/MainActivity.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.btle_sento;
00002
00008 import android.Manifest;
00009 import android.bluetooth.BluetoothAdapter;
00010 import android.bluetooth.BluetoothDevice;
00011 import android.bluetooth.le.BluetoothLeScanner;
00012 import android.bluetooth.le.ScanCallback;
00013 import android.bluetooth.le.ScanFilter;
00014 import android.bluetooth.le.ScanResult;
00015 import android.content.pm.PackageManager;
00016 import android.os.Bundle;
00017 import android.util.Log;
00018 import android.view.View;
00019 import android.widget.Button;
00020 import android.widget.TextView;
00021
00022 import androidx.appcompat.app.AppCompatActivity;
00023 import androidx.core.app.ActivityCompat;
00024 import androidx.core.content.ContextCompat;
00025
00026
00027 import com.example.smariba_upv.btle_sento.API.RetrofitClient;
00028 import com.example.smariba_upv.btle_sento.LOGIC.Utilidades;
00029 import com.example.smariba_upv.btle_sento.POJO.Medicion;
00030 import com.example.smariba_upv.btle_sento.POJO.TramaIBeacon;
00031
00032 import java.util.List;
00033
00034 import retrofit2.Call;
00035 import retrofit2.Callback;
00036 import retrofit2.Response;
00037
00038
00039
00043 public class MainActivity extends AppCompatActivity {
00044
00045
00051     private static final String ETIQUETA_LOG = "»»";
00052     private static final int CODIGO_PETICION_PERMISOS = 11223344;
00053
00054     private BluetoothLeScanner elScanner;
00055     private ScanCallback callbackDelEscaneo = null;
00056     private TextView txt_datos;
00057
00058
00059
00064     private void buscarTodosLosDispositivosBTLE() {
00065         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTLE(): empieza ");
00066
00067         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTLE(): instalamos scan callback ");
00068
00069         this.callbackDelEscaneo = new ScanCallback() {
00070             @Override
00071             public void onScanResult(int callbackType, ScanResult resultado) {
00072                 super.onScanResult(callbackType, resultado);
00072 }
```

```

00073         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBT() : onScanResult() ");
00074
00075         mostrarInformacionDispositivoBTLE(resultado);
00076     }
00077
00078     @Override
00079     public void onBatchScanResults(List<ScanResult> results) {
00080         super.onBatchScanResults(results);
00081         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBT() : onBatchScanResults() ");
00082     }
00083
00084     @Override
00085     public void onScanFailed(int errorCode) {
00086         super.onScanFailed(errorCode);
00087         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBT() : onScanFailed() ");
00088     }
00089 }
00090
00091 };
00092
00093 Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBT() : empezamos a escanear ");
00094
00095 if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) != PackageManager.PERMISSION_GRANTED) {
00096     Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBT() : NO tengo permisos para escanear ");
00097     ActivityCompat.requestPermissions(
00098         MainActivity.this,
00099         new String[]{Manifest.permission.BLUETOOTH_SCAN},
00100         CODIGO_PETICION_PERMISOS);
00101     return;
00102 }
00103 this.elScanner.startScan(this.callbackDelEscaneo);
00104
00105 } // ()
00106
00107
00112 private void mostrarInformacionDispositivoBTLE(ScanResult resultado) {
00113
00114     BluetoothDevice bluetoothDevice = resultado.getDevice();
00115     byte[] bytes = resultado.getScanRecord().getBytes();
00116     int rssi = resultado.getRssi();
00117
00118     Log.d(ETIQUETA_LOG, " *****");
00119     Log.d(ETIQUETA_LOG, " ***** DISPOSITIVO DETECTADO BTLE *****");
00120     Log.d(ETIQUETA_LOG, " *****");
00121     if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00122         Log.d(ETIQUETA_LOG, " mostrarInformacionDispositivoBTLE() : NO tengo permisos para conectar ");
00123         ActivityCompat.requestPermissions(
00124             MainActivity.this,
00125             new String[]{Manifest.permission.BLUETOOTH_CONNECT},
00126             CODIGO_PETICION_PERMISOS);
00127         return;
00128     }
00129     Log.d(ETIQUETA_LOG, " nombre = " + bluetoothDevice.getName());
00130     Log.d(ETIQUETA_LOG, " toString = " + bluetoothDevice.toString());
00131
00132     /*
00133     ParcelUuid[] puuids = bluetoothDevice.getUuids();
00134     if (puuids.length >= 1) {
00135         //Log.d(ETIQUETA_LOG, " uuid = " + puuids[0].getUuid());
00136         // Log.d(ETIQUETA_LOG, " uuid = " + puuids[0].toString());
00137     }*/
00138
00139     Log.d(ETIQUETA_LOG, " dirección = " + bluetoothDevice.getAddress());
00140     Log.d(ETIQUETA_LOG, " rssi = " + rssi);
00141
00142     Log.d(ETIQUETA_LOG, " bytes = " + new String(bytes));
00143     Log.d(ETIQUETA_LOG, " bytes (" + bytes.length + ") = " + Utilidades.bytesToHexString(bytes));
00144
00145     TramaIBeacon tib = new TramaIBeacon(bytes);
00146
00147     Log.d(ETIQUETA_LOG, " -----");
00148     Log.d(ETIQUETA_LOG, " prefijo = " + Utilidades.bytesToHexString(tib.getPrefijo()));
00149     Log.d(ETIQUETA_LOG, " advFlags = " + Utilidades.bytesToHexString(tib.getAdvFlags()));
00150     Log.d(ETIQUETA_LOG, " advHeader = " +
00151         Utilidades.bytesToHexString(tib.getAdvHeader()));
00152     Log.d(ETIQUETA_LOG, " companyID = " +
00153         Utilidades.bytesToHexString(tib.getCompanyID()));
00154     Log.d(ETIQUETA_LOG, " iBeacon type = " + Integer.toHexString(tib.getiBeaconType()));
00155     Log.d(ETIQUETA_LOG, " iBeacon length 0x = " +
00156         Integer.toHexString(tib.getiBeaconLength()) + " ( "
00157             + tib.getiBeaconLength() + " ) ");
00158     Log.d(ETIQUETA_LOG, " uuid = " + Utilidades.bytesToHexString(tib.getUUID()));
00159     Log.d(ETIQUETA_LOG, " uuid = " + Utilidades.bytesToString(tib.getUUID()));
00160     Log.d(ETIQUETA_LOG, " major = " + Utilidades.bytesToHexString(tib.getMajor()) + " ( "

```

```

00158         + Utilidades.bytesToInt(tib.getMajor()) + " ) ");
00159     Log.d(ETIQUETA_LOG, " minor = " + Utilidades.bytesToHexString(tib.getMinor()) + " ( "
00160         + Utilidades.bytesToInt(tib.getMinor()) + " ) ");
00161     Log.d(ETIQUETA_LOG, " txPower = " + Integer.toHexString(tib.getTxPower()) + " ( " +
00162     tib.getTxPower() + " ) ");
00163     Log.d(ETIQUETA_LOG, " *****");
00164 } // ()
00165
00166
00171     private void buscarEsteDispositivoBTLE(final String dispositivoBuscado) {
00172         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empieza ");
00173
00174         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): instalamos scan callback ");
00175
00176         // super.onScanResult (ScanSettings.SCAN_MODE_LOW_LATENCY, result); para ahorro de energía
00177
00179     this.callbackDelEscaneo = new ScanCallback() {
00180         @Override
00181         public void onScanResult(int callbackType, ScanResult resultado) {
00182             super.onScanResult(callbackType, resultado);
00183             //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanResult() ");
00184
00185             if (ActivityCompat.checkSelfPermission(MainActivity.this,
00186                 Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00187                 // TODO: Consider calling
00188                 //    ActivityCompat#requestPermissions
00189                 // here to request the missing permissions, and then overriding
00190                 // public void onRequestPermissionsResult(int requestCode, String[] permissions,
00191                 //                                         int[] grantResults)
00192                 // to handle the case where the user grants the permission. See the documentation
00193                 // for ActivityCompat#requestPermissions for more details.
00194                 return;
00195             }
00196
00197             byte[] bytes = resultado.getScanRecord().getBytes();
00198             TramaIBeacon tib = new TramaIBeacon(bytes);
00199             if (Utilidades.bytesToString(tib.getUUID()).equals(dispositivoBuscado)) {
00200                 mostrarInformacionDispositivoBTLE(resultado);
00201                 txt_datos.setText("Major: " + (int) getMedicionesBeacon(resultado));
00202                 insertarMedicion((int) getMedicionesBeacon(resultado));
00203
00204             } else {
00205                 //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanResult(): no es el
00206             dispositivo buscado ");
00207         }
00208     }
00209
00210     @Override
00211     public void onBatchScanResults(List<ScanResult> results) {
00212         super.onBatchScanResults(results);
00213         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onBatchScanResults() ");
00214
00215     }
00216
00217     @Override
00218     public void onScanFailed(int errorCode) {
00219         super.onScanFailed(errorCode);
00220         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanFailed() ");
00221
00222     };
00223 }
00225     ScanFilter sf = new ScanFilter.Builder().setDeviceName(dispositivoBuscado).build();
00226
00227     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empezamos a escanear buscando: " +
00228     dispositivoBuscado);
00229     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empezamos a escanear buscando: " +
00230     dispositivoBuscado
00231         //      + " -> " + Utilidades.stringToUUID( dispositivoBuscado ) );
00232
00233     if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00234         PackageManager.PERMISSION_GRANTED) {
00235         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): NO tengo permisos para escanear ");
00236         ActivityCompat.requestPermissions(
00237             MainActivity.this,
00238             new String[]{Manifest.permission.BLUETOOTH_SCAN},
00239             CODIGO_PETICION_PERMISOS);
00240
00241     }
00242 }
```

```

00246     private void detenerBusquedaDispositivosBTLE() {
00247
00248         if (this.callbackDelEscaneo == null) {
00249             return;
00250         }
00251
00252         if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) != PackageManager.PERMISSION_GRANTED) {
00253             Log.d(ETIQUETA_LOG, " detenerBusquedaDispositivosBTLE(): NO tengo permisos para escanear ");
00254             ActivityCompat.requestPermissions(
00255                 MainActivity.this,
00256                 new String[]{Manifest.permission.BLUETOOTH_SCAN},
00257                 CODIGO_PETICION_PERMISOS);
00258             return;
00259         }
00260         this.elScanner.stopScan(this.callbackDelEscaneo);
00261         this.callbackDelEscaneo = null;
00262
00263     } // ()
00264
00265
00271     public void botonBuscarDispositivosBTLEPulsado(View v) {
00272         Log.d(ETIQUETA_LOG, " boton buscar dispositivos BTLE Pulsado");
00273         this.buscarTodosLosDispositivosBTLE();
00274     } // ()
00275
00276
00282     public void botonBuscarNuestroDispositivoBTLEPulsado(View v) {
00283         Log.d(ETIQUETA_LOG, " boton nuestro dispositivo BTLE Pulsado");
00284         //this.buscarEsteDispositivoBTLE( Utilidades.stringToUUID( "EPSG-GTI-PROY-3A" ) );
00285
00286         this.buscarEsteDispositivoBTLE( "EPSG-GTI-PROY-3D" );
00287         // this.buscarEsteDispositivoBTLE("fistro");
00288
00289     } // ()
00290
00291
00297     public void botonDetenerBusquedaDispositivosBTLEPulsado(View v) {
00298         Log.d(ETIQUETA_LOG, " boton detener busqueda dispositivos BTLE Pulsado");
00299         this.detenerBusquedaDispositivosBTLE();
00300     } // ()
00301
00302
00306     private void inicializarBlueTooth() {
00307         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): obtenemos adaptador BT ");
00308
00309         BluetoothAdapter bta = BluetoothAdapter.getDefaultAdapter();
00310
00311         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): habilitamos adaptador BT ");
00312
00313         if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00314             Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): NO tengo permisos para conectar ");
00315             ActivityCompat.requestPermissions(
00316                 MainActivity.this,
00317                 new String[]{Manifest.permission.BLUETOOTH_CONNECT},
00318                 CODIGO_PETICION_PERMISOS);
00319             return;
00320         }
00321         bta.enable();
00322
00323         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): habilitado = " + bta.isEnabled() );
00324
00325         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): estado = " + bta.getState() );
00326
00327         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): obtenemos escaner btle ");
00328
00329         this.elScanner = bta.getBluetoothLeScanner();
00330
00331         if ( this.elScanner == null ) {
00332             Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): Socorro: NO hemos obtenido escaner btle
00333             !!!!!");
00334         }
00335
00336         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): voy a pedir permisos (si no los tuviera)
00337             !!!!!");
00338
00339         if (
00340             ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH) != PackageManager.PERMISSION_GRANTED
00341             || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_ADMIN) != PackageManager.PERMISSION_GRANTED
00342             || ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
00343         )

```

```

00343         {
00344             ActivityCompat.requestPermissions(
00345                 MainActivity.this,
00346                 new String[]{Manifest.permission.BLUETOOTH, Manifest.permission.BLUETOOTH_ADMIN,
00347                         Manifest.permission.ACCESS_FINE_LOCATION},
00348                 CODIGO_PETICION_PERMISOS);
00349         }
00350         else {
00351             Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): parece que YA tengo los permisos necesarios
00352             !!!!");
00353         }
00354     } // ()
00355
00356
00357     private void verificarPermisosBluetooth() {
00358         Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Verificando permisos");
00359
00360         // Verificamos si los permisos necesarios ya están concedidos
00361         if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH) !=
00362             PackageManager.PERMISSION_GRANTED
00363                 || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_ADMIN) !=
00364             PackageManager.PERMISSION_GRANTED
00365                 || ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
00366             != PackageManager.PERMISSION_GRANTED
00367                 || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00368             PackageManager.PERMISSION_GRANTED
00369                 || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) !=
00370             PackageManager.PERMISSION_GRANTED) {
00371
00372             // Si no están concedidos, los solicitamos
00373             Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Solicitando permisos necesarios");
00374             ActivityCompat.requestPermissions(
00375                 MainActivity.this,
00376                 new String[]{
00377                     Manifest.permission.BLUETOOTH,
00378                     Manifest.permission.BLUETOOTH_ADMIN,
00379                     Manifest.permission.ACCESS_FINE_LOCATION,
00380                     Manifest.permission.BLUETOOTH_SCAN,
00381                     Manifest.permission.BLUETOOTH_CONNECT
00382                 },
00383                 CODIGO_PETICION_PERMISOS
00384             );
00385         } else {
00386             Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Permisos ya concedidos");
00387             inicializarBlueTooth(); // Llamamos a la inicialización de Bluetooth si los permisos
00388             están concedidos
00389         }
00390     } // ()
00391
00392
00393     @Override
00394     protected void onCreate(Bundle savedInstanceState) {
00395         super.onCreate(savedInstanceState);
00396         setContentView(R.layout.activity_main);
00397
00398         Log.d(ETIQUETA_LOG, " onCreate(): empieza ");
00399
00400         // Verificamos los permisos antes de cualquier otra operación
00401         verificarPermisosBluetooth();
00402         Button enviarMedicionButton = findViewById(R.id.btn_insertar);
00403         txt_datos = findViewById(R.id.Txt_datos);
00404
00405     } // onCreate()
00406
00407
00408
00409     @Override
00410     public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)
00411     {
00412         super.onRequestPermissionsResult(requestCode, permissions, grantResults);
00413
00414         switch (requestCode) {
00415             case CODIGO_PETICION_PERMISOS:
00416                 if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
00417                     Log.d(ETIQUETA_LOG, " onRequestPermissionsResult(): Permisos concedidos");
00418                     inicializarBlueTooth(); // Llamamos a la inicialización si se conceden los
00419                     permisos
00420                 } else {
00421                     Log.d(ETIQUETA_LOG, " onRequestPermissionsResult(): Permisos NO concedidos");
00422                 }
00423             return;
00424         }
00425     } // ()
00426
00427
00428
00429     private void insertarMedicion(int major) {
00430
00431
00432
00433
00434
00435
00436
00437
00438
00439
00440
00441
00442
00443
00444
00445
00446
00447
00448
00449
00450
00451
00452
00453
00454
00455
00456
00457
00458
00459
00460
00461
00462
00463
00464
00465
00466
00467
00468
00469
00470
00471
00472
00473
00474
00475
00476
00477
00478
00479
00480
00481
00482
00483
00484
00485
00486
00487
00488
00489
00490
00491
00492
00493
00494
00495
00496
00497
00498
00499
00500
00501
00502
00503
00504
00505
00506
00507
00508
00509
00510
00511
00512
00513
00514
00515
00516
00517
00518
00519
00520
00521
00522
00523
00524
00525
00526
00527
00528
00529
00530
00531
00532
00533
00534
00535
00536
00537
00538
00539
00540
00541
00542
00543
00544
00545
00546
00547
00548
00549
00550
00551
00552
00553
00554
00555
00556
00557
00558
00559
00560
00561
00562
00563
00564
00565
00566
00567
00568
00569
00570
00571
00572
00573
00574
00575
00576
00577
00578
00579
00580
00581
00582
00583
00584
00585
00586
00587
00588
00589
00590
00591
00592
00593
00594
00595
00596
00597
00598
00599
00600
00601
00602
00603
00604
00605
00606
00607
00608
00609
00610
00611
00612
00613
00614
00615
00616
00617
00618
00619
00620
00621
00622
00623
00624
00625
00626
00627
00628
00629
00630
00631
00632
00633
00634
00635
00636
00637
00638
00639
00640
00641
00642
00643
00644
00645
00646
00647
00648
00649
00650
00651
00652
00653
00654
00655
00656
00657
00658
00659
00660
00661
00662
00663
00664
00665
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682
00683
00684
00685
00686
00687
00688
00689
00690
00691
00692
00693
00694
00695
00696
00697
00698
00699
00700
00701
00702
00703
00704
00705
00706
00707
00708
00709
00710
00711
00712
00713
00714
00715
00716
00717
00718
00719
00720
00721
00722
00723
00724
00725
00726
00727
00728
00729
00730
00731
00732
00733
00734
00735
00736
00737
00738
00739
00740
00741
00742
00743
00744
00745
00746
00747
00748
00749
00750
00751
00752
00753
00754
00755
00756
00757
00758
00759
00760
00761
00762
00763
00764
00765
00766
00767
00768
00769
00770
00771
00772
00773
00774
00775
00776
00777
00778
00779
00780
00781
00782
00783
00784
00785
00786
00787
00788
00789
00790
00791
00792
00793
00794
00795
00796
00797
00798
00799
00800
00801
00802
00803
00804
00805
00806
00807
00808
00809
00810
00811
00812
00813
00814
00815
00816
00817
00818
00819
00820
00821
00822
00823
00824
00825
00826
00827
00828
00829
00830
00831
00832
00833
00834
00835
00836
00837
00838
00839
00840
00841
00842
00843
00844
00845
00846
00847
00848
00849
00850
00851
00852
00853
00854
00855
00856
00857
00858
00859
00860
00861
00862
00863
00864
00865
00866
00867
00868
00869
00870
00871
00872
00873
00874
00875
00876
00877
00878
00879
00880
00881
00882
00883
00884
00885
00886
00887
00888
00889
00890
00891
00892
00893
00894
00895
00896
00897
00898
00899
00900
00901
00902
00903
00904
00905
00906
00907
00908
00909
00910
00911
00912
00913
00914
00915
00916
00917
00918
00919
00920
00921
00922
00923
00924
00925
00926
00927
00928
00929
00930
00931
00932
00933
00934
00935
00936
00937
00938
00939
00940
00941
00942
00943
00944
00945
00946
00947
00948
00949
00950
00951
00952
00953
00954
00955
00956
00957
00958
00959
00960
00961
00962
00963
00964
00965
00966
00967
00968
00969
00970
00971
00972
00973
00974
00975
00976
00977
00978
00979
00980
00981
00982
00983
00984
00985
00986
00987
00988
00989
00990
00991
00992
00993
00994
00995
00996
00997
00998
00999
01000
01001
01002
01003
01004
01005
01006
01007
01008
01009
01010
01011
01012
01013
01014
01015
01016
01017
01018
01019
01020
01021
01022
01023
01024
01025
01026
01027
01028
01029
01030
01031
01032
01033
01034
01035
01036
01037
01038
01039
01040
01041
01042
01043
01044
01045
01046
01047
01048
01049
01050
01051
01052
01053
01054
01055
01056
01057
01058
01059
01060
01061
01062
01063
01064
01065
01066
01067
01068
01069
01070
01071
01072
01073
01074
01075
01076
01077
01078
01079
01080
01081
01082
01083
01084
01085
01086
01087
01088
01089
01090
01091
01092
01093
01094
01095
01096
01097
01098
01099
01100
01101
01102
01103
01104
01105
01106
01107
01108
01109
01110
01111
01112
01113
01114
01115
01116
01117
01118
01119
01120
01121
01122
01123
01124
01125
01126
01127
01128
01129
01130
01131
01132
01133
01134
01135
01136
01137
01138
01139
01140
01141
01142
01143
01144
01145
01146
01147
01148
01149
01150
01151
01152
01153
01154
01155
01156
01157
01158
01159
01160
01161
01162
01163
01164
01165
01166
01167
01168
01169
01170
01171
01172
01173
01174
01175
01176
01177
01178
01179
01180
01181
01182
01183
01184
01185
01186
01187
01188
01189
01190
01191
01192
01193
01194
01195
01196
01197
01198
01199
01200
01201
01202
01203
01204
01205
01206
01207
01208
01209
01210
01211
01212
01213
01214
01215
01216
01217
01218
01219
01220
01221
01222
01223
01224
01225
01226
01227
01228
01229
01230
01231
01232
01233
01234
01235
01236
01237
01238
01239
01240
01241
01242
01243
01244
01245
01246
01247
01248
01249
01250
01251
01252
01253
01254
01255
01256
01257
01258
01259
01260
01261
01262
01263
01264
01265
01266
01267
01268
01269
01270
01271
01272
01273
01274
01275
01276
01277
01278
01279
01280
01281
01282
01283
01284
01285
01286
01287
01288
01289
01290
01291
01292
01293
01294
01295
01296
01297
01298
01299
01300
01301
01302
01303
01304
01305
01306
01307
01308
01309
01310
01311
01312
01313
01314
01315
01316
01317
01318
01319
01320
01321
01322
01323
01324
01325
01326
01327
01328
01329
01330
01331
01332
01333
01334
01335
01336
01337
01338
01339
01340
01341
01342
01343
01344
01345
01346
01347
01348
01349
01350
01351
01352
01353
01354
01355
01356
01357
01358
01359
01360
01361
01362
01363
01364
01365
01366
01367
01368
01369
01370
01371
01372
01373
01374
01375
01376
01377
01378
01379
01380
01381
01382
01383
01384
01385
01386
01387
01388
01389
01390
01391
01392
01393
01394
01395
01396
01397
01398
01399
01400
01401
01402
01403
01404
01405
01406
01407
01408
01409
01410
01411
01412
01413
01414
01415
01416
01417
01418
01419
01420
01421
01422
01423
01424
01425
01426
01427
01428
01429
01430
01431
01432
01433
01434
01435
01436
01437
01438
01439
01440
01441
01442
01443
01444
01445
01446
01447
01448
01449
01450
01451
01452
01453
01454
01455
01456
01457
01458
01459
01460
01461
01462
01463
01464
01465
01466
01467
01468
01469
01470
01471
01472
01473
01474
01475
01476
01477
01478
01479
01480
01481
01482
01483
01484
01485
01486
01487
01488
01489
01490
01491
01492
01493
01494
01495
01496
01497
01498
01499
01500
01501
01502
01503
01504
01505
01506
01507
01508
01509
01510
01511
01512
01513
01514
01515
01516
01517
01518
01519
01520
01521
01522
01523
01524
01525
01526
01527
01528
01529
01530
01531
01532
01533
01534
01535
01536
01537
01538
01539
01540
01541
01542
01543
01544
01545
01546
01547
01548
01549
01550
01551
01552
01553
01554
01555
01556
01557
01558
01559
01560
01561
01562
01563
01564
01565
01566
01567
01568
01569
01570
01571
01572
01573
01574
01575
01576
01577
01578
01579
01580
01581
01582
01583
01584
01585
01586
01587
01588
01589
01590
01591
01592
01593
01594
01595
01596
01597
01598
01599
01600
01601
01602
01603
01604
01605
01606
01607
01608
01609
01610
01611
01612
01613
01614
01615
01616
01617
01618
01619
01620
01621
01622
01623
01624
01625
01626
01627
01628
01629
01630
01631
01632
01633
01634
01635
01636
01637
01638
01639
01640
01641
01642
01643
01644
01645
01646
01647
01648
01649
01650
01651
01652
01653
01654
01655
01656
01657
01658
01659
01660
01661
01662
01663
01664
01665
01666
01667
01668
01669
01670
01671
01672
01673
01674
01675
01676
01677
01678
01679
01680
01681
01682
01683
01684
01685
01686
01687
01688
01689
01690
01691
01692
01693
01694
01695
01696
01697
01698
01699
01700
01701
01702
01703
01704
01705
01706
01707
01708
01709
01710
01711
01712
01713
01714
01715
01716
01717
01718
01719
01720
01721
01722
01723
01724
01725
01726
01727
01728
01729
01729
01730
01731
01732
01733
01734
01735
01736
01737
01738
01739
01740
01741
01742
01743
01744
01745
01746
01747
01748
01749
01750
01751
01752
01753
01754
01755
01756
01757
01758
01759
01759
01760
01761
01762
01763
01764
01765
01766
01767
01768
01769
01770
01771
01772
01773
01774
01775
01776
01777
01778
01779
01779
01780
01781
01782
01783
01784
01785
01786
01787
01788
01789
01790
01791
01792
01793
01794
01795
01796
01797
01798
01799
01800
01801
01802
01803
01804
01805
01806
01807
01808
01809
01809
01810
01811
01812
01813
01814
01815
01816
01817
01818
01819
01820
01821
01822
01823
01824
01825
01826
01827
01828
01829
01829
01830
01831
01832
01833
01834
01835
01836
01837
01838
01839
01839
01840
01841
01842
01843
01844
01845
01846
01847
01848
01849
01849
01850
01851
01852
01853
01854
01855
01856
01857
01858
01859
01859
01860
01861
01862
01863
01864
01865
01866
01867
01868
01869
01869
01870
01871
01872
01873
01874
01875
01876
01877
01878
01879
01879
01880
01881
01882
01883
01884
01885
01886
01887
01888
01889
01889
01890
01891
01892
01893
01894
01895
01896
01897
01898
01899
01899
01900
01901
01902
01903
01904
01905
01906
01907
01908
01909
01909
01910
01911
01912
01913
01914
01915
01916
01917
01918
01919
01919
01920
01921
01922
01923
01924
01925
01926
01927
01928
01929
01929
01930
01931
01932
01933
01934
01935
01936
01937
01938
01939
01939
01940
01941
01942
01943
01944
01945
01946
01947
01948
01949
01949
01950
01951
01952
01953
01954
01955
01956
01957
01958
01959
01959
01960
01961
01962
01963
01964
01965
01966
01967
01968
01969
01969
01970
01971
01972
01973
01974
01975
01976
01977
01978
01979
01979
01980
01981
01982
01983
01984
01985
01986
01987
01988
01989
01989
01990
01991
01992
01993
01994
01995
01996
01997
01998
01999
01999
02000
02001
02002
02003
02004
02005
02006
02007
02008
02009
02009
02010
02011
02012
02013
02014
02015
02016
02017
02018
02019
02019
02020
02021
02022
02023
02024
02025
02026
02027
02028
02029
02029
02030
02031
02032
02033
02034
02035
02036
02037
02038
02039
02039
02040
02041
02042
02043
02044
02045
02046
02047
02048
02049
02049
02050
02051
02052
02053
02054
02055
02056
0205
```

```

00437     RetrofitClient.getApiService().insertarMedicion(new Medicion("Living Room", "CO2",
00438         major)).enqueue(new Callback<Void>() {
00439             @Override
00440             public void onResponse(Call<Void> call, Response<Void> response) {
00441                 if (response.isSuccessful()) {
00442                     Log.d("API Response", "Medición insertada correctamente");
00443                 } else {
00444                     Log.d("API Error", "Error en la respuesta: " + response.code());
00445                 }
00446             }
00447             @Override
00448             public void onFailure(Call<Void> call, Throwable t) {
00449                 Log.e("API Failure", "Error al conectar con el servidor", t);
00450             }
00451         });
00452     }
00453 }
00454
00455     public double getMedicionesBeacon(ScanResult resultado) {
00456         byte[] bytes = resultado.getScanRecord().getBytes();
00457         TramaIBeacon tib = new TramaIBeacon(bytes);
00458         return Utilidades.bytesToInt(tib.getMinor());
00459     }
00460 }
00461 // class

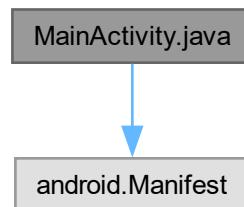
```

## 7.25. Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-← Android/app/src/main/java/com/example/smariba\_upv/airflow/MainActivity.java

Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE.

import android.Manifest;

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/MainActivity.java:



### Clases

- class com.example.smariba\_upv.airflow.MainActivity

*La clase principal que gestiona la interfaz y las funciones BTLE.*

### Paquetes

- package com.example.smariba\_upv.airflow

### 7.25.1. Descripción detallada

Clase principal de la aplicación desde la que se inicia el escaneo de dispositivos BTLE.

Versión

1.0

Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/MainActivity.java](#)

## 7.26. Desktop/Clases/Proyecto Bio/AirFlow-[←](#) Android/app/src/main/java/com/example/smariba\_upv/airflow/MainActivity.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow;
00002
00008 import android.Manifest;
00009 import android.bluetooth.BluetoothAdapter;
00010 import android.bluetooth.BluetoothDevice;
00011 import android.bluetooth.le.BluetoothLeScanner;
00012 import android.bluetooth.le.ScanCallback;
00013 import android.bluetooth.le.ScanFilter;
00014 import android.bluetooth.le.ScanResult;
00015 import android.content.pm.PackageManager;
00016 import android.os.Bundle;
00017 import android.util.Log;
00018 import android.view.View;
00019 import android.widget.Button;
00020 import android.widget.TextView;
00021
00022 import androidx.appcompat.app.AppCompatActivity;
00023 import androidx.core.app.ActivityCompat;
00024 import androidx.core.content.ContextCompat;
00025
00026
00027 import com.example.smariba_upv.airflow.API.RetrofitClient;
00028 import com.example.smariba_upv.airflow.LOGIC.Utilidades;
00029 import com.example.smariba_upv.airflow.POJO.Medicion;
00030 import com.example.smariba_upv.airflow.POJO.TramaIBeacon;
00031
00032 import java.util.List;
00033
00034 import retrofit2.Call;
00035 import retrofit2.Callback;
00036 import retrofit2.Response;
00037
00038
00039
00043 public class MainActivity extends AppCompatActivity {
00044
00045
00051     private static final String ETIQUETA_LOG = "»»";
00052     private static final int CODIGO_PETICION_PERMISOS = 11223344;
00053
00054     private BluetoothLeScanner elScanner;
00055     private ScanCallback callbackDelEscaneo = null;
00056     private TextView txt_datos;
00057
00058
00059
00064     /* private void buscarTodosLosDispositivosBTLE() {
00065         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): empieza ");
00066
00067         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): instalamos scan callback ");
00068
00069         this.callbackDelEscaneo = new ScanCallback() {
00070             @Override
00071             public onScanResult(int callbackType, ScanResult resultado) {
00072                 super.onScanResult(callbackType, resultado);
00073             }
00074         };
00075     }
00076
00077     private void iniciarEscaneo() {
00078         if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
00079             ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS_FINE_LOCATION, Manifest.permission.ACCESS_COARSE_LOCATION}, CODIGO_PETICION_PERMISOS);
00080         } else {
00081             iniciar();
00082         }
00083     }
00084
00085     private void iniciar() {
00086         elScanner.startScan(callbackDelEscaneo);
00087     }
00088
00089     private void finalizar() {
00090         elScanner.stopScan(callbackDelEscaneo);
00091     }
00092
00093     private void limpiar() {
00094         txt_datos.setText("");
00095     }
00096
00097     private void mostrarMedicion(Medicion med) {
00098         txt_datos.setText("ID: " + med.getId() + "\n" +
00099             "Título: " + med.getTitulo() + "\n" +
00100            "Latitud: " + med.getLatitud() + "\n" +
00101            "Longitud: " + med.getLongitud());
00102
00103     }
00104
00105     private void mostrarTrama(TramaIBeacon t) {
00106         txt_datos.setText("ID: " + t.getId() + "\n" +
00107             "Título: " + t.getTitulo() + "\n" +
00108             "Latitud: " + t.getLatitud() + "\n" +
00109             "Longitud: " + t.getLongitud());
00110
00111     }
00112
00113     private void mostrarMediciones(List<Medicion> mediciones) {
00114         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00115
00116         for (Medicion med : mediciones) {
00117             mostrarMedicion(med);
00118         }
00119     }
00120
00121     private void mostrarTramas(List<TramaIBeacon> tramas) {
00122         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00123
00124         for (TramaIBeacon t : tramas) {
00125             mostrarTrama(t);
00126         }
00127     }
00128
00129     private void mostrarError(String error) {
00130         txt_datos.setText(error);
00131     }
00132
00133     private void iniciar() {
00134         elScanner.startScan(callbackDelEscaneo);
00135     }
00136
00137     private void finalizar() {
00138         elScanner.stopScan(callbackDelEscaneo);
00139     }
00140
00141     private void limpiar() {
00142         txt_datos.setText("");
00143     }
00144
00145     private void mostrarMedicion(Medicion med) {
00146         txt_datos.setText("ID: " + med.getId() + "\n" +
00147             "Título: " + med.getTitulo() + "\n" +
00148             "Latitud: " + med.getLatitud() + "\n" +
00149             "Longitud: " + med.getLongitud());
00150
00151     }
00152
00153     private void mostrarTrama(TramaIBeacon t) {
00154         txt_datos.setText("ID: " + t.getId() + "\n" +
00155             "Título: " + t.getTitulo() + "\n" +
00156             "Latitud: " + t.getLatitud() + "\n" +
00157             "Longitud: " + t.getLongitud());
00158
00159     }
00160
00161     private void mostrarMediciones(List<Medicion> mediciones) {
00162         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00163
00164         for (Medicion med : mediciones) {
00165             mostrarMedicion(med);
00166         }
00167     }
00168
00169     private void mostrarTramas(List<TramaIBeacon> tramas) {
00170         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00171
00172         for (TramaIBeacon t : tramas) {
00173             mostrarTrama(t);
00174         }
00175     }
00176
00177     private void mostrarError(String error) {
00178         txt_datos.setText(error);
00179     }
00180
00181     private void iniciar() {
00182         elScanner.startScan(callbackDelEscaneo);
00183     }
00184
00185     private void finalizar() {
00186         elScanner.stopScan(callbackDelEscaneo);
00187     }
00188
00189     private void limpiar() {
00190         txt_datos.setText("");
00191     }
00192
00193     private void mostrarMedicion(Medicion med) {
00194         txt_datos.setText("ID: " + med.getId() + "\n" +
00195             "Título: " + med.getTitulo() + "\n" +
00196             "Latitud: " + med.getLatitud() + "\n" +
00197             "Longitud: " + med.getLongitud());
00198
00199     }
00200
00201     private void mostrarTrama(TramaIBeacon t) {
00202         txt_datos.setText("ID: " + t.getId() + "\n" +
00203             "Título: " + t.getTitulo() + "\n" +
00204             "Latitud: " + t.getLatitud() + "\n" +
00205             "Longitud: " + t.getLongitud());
00206
00207     }
00208
00209     private void mostrarMediciones(List<Medicion> mediciones) {
00210         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00211
00212         for (Medicion med : mediciones) {
00213             mostrarMedicion(med);
00214         }
00215     }
00216
00217     private void mostrarTramas(List<TramaIBeacon> tramas) {
00218         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00219
00220         for (TramaIBeacon t : tramas) {
00221             mostrarTrama(t);
00222         }
00223     }
00224
00225     private void mostrarError(String error) {
00226         txt_datos.setText(error);
00227     }
00228
00229     private void iniciar() {
00230         elScanner.startScan(callbackDelEscaneo);
00231     }
00232
00233     private void finalizar() {
00234         elScanner.stopScan(callbackDelEscaneo);
00235     }
00236
00237     private void limpiar() {
00238         txt_datos.setText("");
00239     }
00240
00241     private void mostrarMedicion(Medicion med) {
00242         txt_datos.setText("ID: " + med.getId() + "\n" +
00243             "Título: " + med.getTitulo() + "\n" +
00244             "Latitud: " + med.getLatitud() + "\n" +
00245             "Longitud: " + med.getLongitud());
00246
00247     }
00248
00249     private void mostrarTrama(TramaIBeacon t) {
00250         txt_datos.setText("ID: " + t.getId() + "\n" +
00251             "Título: " + t.getTitulo() + "\n" +
00252             "Latitud: " + t.getLatitud() + "\n" +
00253             "Longitud: " + t.getLongitud());
00254
00255     }
00256
00257     private void mostrarMediciones(List<Medicion> mediciones) {
00258         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00259
00260         for (Medicion med : mediciones) {
00261             mostrarMedicion(med);
00262         }
00263     }
00264
00265     private void mostrarTramas(List<TramaIBeacon> tramas) {
00266         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00267
00268         for (TramaIBeacon t : tramas) {
00269             mostrarTrama(t);
00270         }
00271     }
00272
00273     private void mostrarError(String error) {
00274         txt_datos.setText(error);
00275     }
00276
00277     private void iniciar() {
00278         elScanner.startScan(callbackDelEscaneo);
00279     }
00280
00281     private void finalizar() {
00282         elScanner.stopScan(callbackDelEscaneo);
00283     }
00284
00285     private void limpiar() {
00286         txt_datos.setText("");
00287     }
00288
00289     private void mostrarMedicion(Medicion med) {
00290         txt_datos.setText("ID: " + med.getId() + "\n" +
00291             "Título: " + med.getTitulo() + "\n" +
00292             "Latitud: " + med.getLatitud() + "\n" +
00293             "Longitud: " + med.getLongitud());
00294
00295     }
00296
00297     private void mostrarTrama(TramaIBeacon t) {
00298         txt_datos.setText("ID: " + t.getId() + "\n" +
00299             "Título: " + t.getTitulo() + "\n" +
00300             "Latitud: " + t.getLatitud() + "\n" +
00301             "Longitud: " + t.getLongitud());
00302
00303     }
00304
00305     private void mostrarMediciones(List<Medicion> mediciones) {
00306         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00307
00308         for (Medicion med : mediciones) {
00309             mostrarMedicion(med);
00310         }
00311     }
00312
00313     private void mostrarTramas(List<TramaIBeacon> tramas) {
00314         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00315
00316         for (TramaIBeacon t : tramas) {
00317             mostrarTrama(t);
00318         }
00319     }
00320
00321     private void mostrarError(String error) {
00322         txt_datos.setText(error);
00323     }
00324
00325     private void iniciar() {
00326         elScanner.startScan(callbackDelEscaneo);
00327     }
00328
00329     private void finalizar() {
00330         elScanner.stopScan(callbackDelEscaneo);
00331     }
00332
00333     private void limpiar() {
00334         txt_datos.setText("");
00335     }
00336
00337     private void mostrarMedicion(Medicion med) {
00338         txt_datos.setText("ID: " + med.getId() + "\n" +
00339             "Título: " + med.getTitulo() + "\n" +
00340             "Latitud: " + med.getLatitud() + "\n" +
00341             "Longitud: " + med.getLongitud());
00342
00343     }
00344
00345     private void mostrarTrama(TramaIBeacon t) {
00346         txt_datos.setText("ID: " + t.getId() + "\n" +
00347             "Título: " + t.getTitulo() + "\n" +
00348             "Latitud: " + t.getLatitud() + "\n" +
00349             "Longitud: " + t.getLongitud());
00350
00351     }
00352
00353     private void mostrarMediciones(List<Medicion> mediciones) {
00354         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00355
00356         for (Medicion med : mediciones) {
00357             mostrarMedicion(med);
00358         }
00359     }
00360
00361     private void mostrarTramas(List<TramaIBeacon> tramas) {
00362         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00363
00364         for (TramaIBeacon t : tramas) {
00365             mostrarTrama(t);
00366         }
00367     }
00368
00369     private void mostrarError(String error) {
00370         txt_datos.setText(error);
00371     }
00372
00373     private void iniciar() {
00374         elScanner.startScan(callbackDelEscaneo);
00375     }
00376
00377     private void finalizar() {
00378         elScanner.stopScan(callbackDelEscaneo);
00379     }
00380
00381     private void limpiar() {
00382         txt_datos.setText("");
00383     }
00384
00385     private void mostrarMedicion(Medicion med) {
00386         txt_datos.setText("ID: " + med.getId() + "\n" +
00387             "Título: " + med.getTitulo() + "\n" +
00388             "Latitud: " + med.getLatitud() + "\n" +
00389             "Longitud: " + med.getLongitud());
00390
00391     }
00392
00393     private void mostrarTrama(TramaIBeacon t) {
00394         txt_datos.setText("ID: " + t.getId() + "\n" +
00395             "Título: " + t.getTitulo() + "\n" +
00396             "Latitud: " + t.getLatitud() + "\n" +
00397             "Longitud: " + t.getLongitud());
00398
00399     }
00400
00401     private void mostrarMediciones(List<Medicion> mediciones) {
00402         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00403
00404         for (Medicion med : mediciones) {
00405             mostrarMedicion(med);
00406         }
00407     }
00408
00409     private void mostrarTramas(List<TramaIBeacon> tramas) {
00410         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00411
00412         for (TramaIBeacon t : tramas) {
00413             mostrarTrama(t);
00414         }
00415     }
00416
00417     private void mostrarError(String error) {
00418         txt_datos.setText(error);
00419     }
00420
00421     private void iniciar() {
00422         elScanner.startScan(callbackDelEscaneo);
00423     }
00424
00425     private void finalizar() {
00426         elScanner.stopScan(callbackDelEscaneo);
00427     }
00428
00429     private void limpiar() {
00430         txt_datos.setText("");
00431     }
00432
00433     private void mostrarMedicion(Medicion med) {
00434         txt_datos.setText("ID: " + med.getId() + "\n" +
00435             "Título: " + med.getTitulo() + "\n" +
00436             "Latitud: " + med.getLatitud() + "\n" +
00437             "Longitud: " + med.getLongitud());
00438
00439     }
00440
00441     private void mostrarTrama(TramaIBeacon t) {
00442         txt_datos.setText("ID: " + t.getId() + "\n" +
00443             "Título: " + t.getTitulo() + "\n" +
00444             "Latitud: " + t.getLatitud() + "\n" +
00445             "Longitud: " + t.getLongitud());
00446
00447     }
00448
00449     private void mostrarMediciones(List<Medicion> mediciones) {
00450         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00451
00452         for (Medicion med : mediciones) {
00453             mostrarMedicion(med);
00454         }
00455     }
00456
00457     private void mostrarTramas(List<TramaIBeacon> tramas) {
00458         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00459
00460         for (TramaIBeacon t : tramas) {
00461             mostrarTrama(t);
00462         }
00463     }
00464
00465     private void mostrarError(String error) {
00466         txt_datos.setText(error);
00467     }
00468
00469     private void iniciar() {
00470         elScanner.startScan(callbackDelEscaneo);
00471     }
00472
00473     private void finalizar() {
00474         elScanner.stopScan(callbackDelEscaneo);
00475     }
00476
00477     private void limpiar() {
00478         txt_datos.setText("");
00479     }
00480
00481     private void mostrarMedicion(Medicion med) {
00482         txt_datos.setText("ID: " + med.getId() + "\n" +
00483             "Título: " + med.getTitulo() + "\n" +
00484             "Latitud: " + med.getLatitud() + "\n" +
00485             "Longitud: " + med.getLongitud());
00486
00487     }
00488
00489     private void mostrarTrama(TramaIBeacon t) {
00490         txt_datos.setText("ID: " + t.getId() + "\n" +
00491             "Título: " + t.getTitulo() + "\n" +
00492             "Latitud: " + t.getLatitud() + "\n" +
00493             "Longitud: " + t.getLongitud());
00494
00495     }
00496
00497     private void mostrarMediciones(List<Medicion> mediciones) {
00498         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00499
00500         for (Medicion med : mediciones) {
00501             mostrarMedicion(med);
00502         }
00503     }
00504
00505     private void mostrarTramas(List<TramaIBeacon> tramas) {
00506         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00507
00508         for (TramaIBeacon t : tramas) {
00509             mostrarTrama(t);
00510         }
00511     }
00512
00513     private void mostrarError(String error) {
00514         txt_datos.setText(error);
00515     }
00516
00517     private void iniciar() {
00518         elScanner.startScan(callbackDelEscaneo);
00519     }
00520
00521     private void finalizar() {
00522         elScanner.stopScan(callbackDelEscaneo);
00523     }
00524
00525     private void limpiar() {
00526         txt_datos.setText("");
00527     }
00528
00529     private void mostrarMedicion(Medicion med) {
00530         txt_datos.setText("ID: " + med.getId() + "\n" +
00531             "Título: " + med.getTitulo() + "\n" +
00532             "Latitud: " + med.getLatitud() + "\n" +
00533             "Longitud: " + med.getLongitud());
00534
00535     }
00536
00537     private void mostrarTrama(TramaIBeacon t) {
00538         txt_datos.setText("ID: " + t.getId() + "\n" +
00539             "Título: " + t.getTitulo() + "\n" +
00540             "Latitud: " + t.getLatitud() + "\n" +
00541             "Longitud: " + t.getLongitud());
00542
00543     }
00544
00545     private void mostrarMediciones(List<Medicion> mediciones) {
00546         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00547
00548         for (Medicion med : mediciones) {
00549             mostrarMedicion(med);
00550         }
00551     }
00552
00553     private void mostrarTramas(List<TramaIBeacon> tramas) {
00554         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00555
00556         for (TramaIBeacon t : tramas) {
00557             mostrarTrama(t);
00558         }
00559     }
00560
00561     private void mostrarError(String error) {
00562         txt_datos.setText(error);
00563     }
00564
00565     private void iniciar() {
00566         elScanner.startScan(callbackDelEscaneo);
00567     }
00568
00569     private void finalizar() {
00570         elScanner.stopScan(callbackDelEscaneo);
00571     }
00572
00573     private void limpiar() {
00574         txt_datos.setText("");
00575     }
00576
00577     private void mostrarMedicion(Medicion med) {
00578         txt_datos.setText("ID: " + med.getId() + "\n" +
00579             "Título: " + med.getTitulo() + "\n" +
00580             "Latitud: " + med.getLatitud() + "\n" +
00581             "Longitud: " + med.getLongitud());
00582
00583     }
00584
00585     private void mostrarTrama(TramaIBeacon t) {
00586         txt_datos.setText("ID: " + t.getId() + "\n" +
00587             "Título: " + t.getTitulo() + "\n" +
00588             "Latitud: " + t.getLatitud() + "\n" +
00589             "Longitud: " + t.getLongitud());
00590
00591     }
00592
00593     private void mostrarMediciones(List<Medicion> mediciones) {
00594         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00595
00596         for (Medicion med : mediciones) {
00597             mostrarMedicion(med);
00598         }
00599     }
00600
00601     private void mostrarTramas(List<TramaIBeacon> tramas) {
00602         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00603
00604         for (TramaIBeacon t : tramas) {
00605             mostrarTrama(t);
00606         }
00607     }
00608
00609     private void mostrarError(String error) {
00610         txt_datos.setText(error);
00611     }
00612
00613     private void iniciar() {
00614         elScanner.startScan(callbackDelEscaneo);
00615     }
00616
00617     private void finalizar() {
00618         elScanner.stopScan(callbackDelEscaneo);
00619     }
00620
00621     private void limpiar() {
00622         txt_datos.setText("");
00623     }
00624
00625     private void mostrarMedicion(Medicion med) {
00626         txt_datos.setText("ID: " + med.getId() + "\n" +
00627             "Título: " + med.getTitulo() + "\n" +
00628             "Latitud: " + med.getLatitud() + "\n" +
00629             "Longitud: " + med.getLongitud());
00630
00631     }
00632
00633     private void mostrarTrama(TramaIBeacon t) {
00634         txt_datos.setText("ID: " + t.getId() + "\n" +
00635             "Título: " + t.getTitulo() + "\n" +
00636             "Latitud: " + t.getLatitud() + "\n" +
00637             "Longitud: " + t.getLongitud());
00638
00639     }
00640
00641     private void mostrarMediciones(List<Medicion> mediciones) {
00642         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00643
00644         for (Medicion med : mediciones) {
00645             mostrarMedicion(med);
00646         }
00647     }
00648
00649     private void mostrarTramas(List<TramaIBeacon> tramas) {
00650         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00651
00652         for (TramaIBeacon t : tramas) {
00653             mostrarTrama(t);
00654         }
00655     }
00656
00657     private void mostrarError(String error) {
00658         txt_datos.setText(error);
00659     }
00660
00661     private void iniciar() {
00662         elScanner.startScan(callbackDelEscaneo);
00663     }
00664
00665     private void finalizar() {
00666         elScanner.stopScan(callbackDelEscaneo);
00667     }
00668
00669     private void limpiar() {
00670         txt_datos.setText("");
00671     }
00672
00673     private void mostrarMedicion(Medicion med) {
00674         txt_datos.setText("ID: " + med.getId() + "\n" +
00675             "Título: " + med.getTitulo() + "\n" +
00676             "Latitud: " + med.getLatitud() + "\n" +
00677             "Longitud: " + med.getLongitud());
00678
00679     }
00680
00681     private void mostrarTrama(TramaIBeacon t) {
00682         txt_datos.setText("ID: " + t.getId() + "\n" +
00683             "Título: " + t.getTitulo() + "\n" +
00684             "Latitud: " + t.getLatitud() + "\n" +
00685             "Longitud: " + t.getLongitud());
00686
00687     }
00688
00689     private void mostrarMediciones(List<Medicion> mediciones) {
00690         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00691
00692         for (Medicion med : mediciones) {
00693             mostrarMedicion(med);
00694         }
00695     }
00696
00697     private void mostrarTramas(List<TramaIBeacon> tramas) {
00698         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00699
00700         for (TramaIBeacon t : tramas) {
00701             mostrarTrama(t);
00702         }
00703     }
00704
00705     private void mostrarError(String error) {
00706         txt_datos.setText(error);
00707     }
00708
00709     private void iniciar() {
00710         elScanner.startScan(callbackDelEscaneo);
00711     }
00712
00713     private void finalizar() {
00714         elScanner.stopScan(callbackDelEscaneo);
00715     }
00716
00717     private void limpiar() {
00718         txt_datos.setText("");
00719     }
00720
00721     private void mostrarMedicion(Medicion med) {
00722         txt_datos.setText("ID: " + med.getId() + "\n" +
00723             "Título: " + med.getTitulo() + "\n" +
00724             "Latitud: " + med.getLatitud() + "\n" +
00725             "Longitud: " + med.getLongitud());
00726
00727     }
00728
00729     private void mostrarTrama(TramaIBeacon t) {
00730         txt_datos.setText("ID: " + t.getId() + "\n" +
00731             "Título: " + t.getTitulo() + "\n" +
00732             "Latitud: " + t.getLatitud() + "\n" +
00733             "Longitud: " + t.getLongitud());
00734
00735     }
00736
00737     private void mostrarMediciones(List<Medicion> mediciones) {
00738         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00739
00740         for (Medicion med : mediciones) {
00741             mostrarMedicion(med);
00742         }
00743     }
00744
00745     private void mostrarTramas(List<TramaIBeacon> tramas) {
00746         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00747
00748         for (TramaIBeacon t : tramas) {
00749             mostrarTrama(t);
00750         }
00751     }
00752
00753     private void mostrarError(String error) {
00754         txt_datos.setText(error);
00755     }
00756
00757     private void iniciar() {
00758         elScanner.startScan(callbackDelEscaneo);
00759     }
00760
00761     private void finalizar() {
00762         elScanner.stopScan(callbackDelEscaneo);
00763     }
00764
00765     private void limpiar() {
00766         txt_datos.setText("");
00767     }
00768
00769     private void mostrarMedicion(Medicion med) {
00770         txt_datos.setText("ID: " + med.getId() + "\n" +
00771             "Título: " + med.getTitulo() + "\n" +
00772             "Latitud: " + med.getLatitud() + "\n" +
00773             "Longitud: " + med.getLongitud());
00774
00775     }
00776
00777     private void mostrarTrama(TramaIBeacon t) {
00778         txt_datos.setText("ID: " + t.getId() + "\n" +
00779             "Título: " + t.getTitulo() + "\n" +
00780             "Latitud: " + t.getLatitud() + "\n" +
00781             "Longitud: " + t.getLongitud());
00782
00783     }
00784
00785     private void mostrarMediciones(List<Medicion> mediciones) {
00786         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00787
00788         for (Medicion med : mediciones) {
00789             mostrarMedicion(med);
00790         }
00791     }
00792
00793     private void mostrarTramas(List<TramaIBeacon> tramas) {
00794         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00795
00796         for (TramaIBeacon t : tramas) {
00797             mostrarTrama(t);
00798         }
00799     }
00800
00801     private void mostrarError(String error) {
00802         txt_datos.setText(error);
00803     }
00804
00805     private void iniciar() {
00806         elScanner.startScan(callbackDelEscaneo);
00807     }
00808
00809     private void finalizar() {
00810         elScanner.stopScan(callbackDelEscaneo);
00811     }
00812
00813     private void limpiar() {
00814         txt_datos.setText("");
00815     }
00816
00817     private void mostrarMedicion(Medicion med) {
00818         txt_datos.setText("ID: " + med.getId() + "\n" +
00819             "Título: " + med.getTitulo() + "\n" +
00820             "Latitud: " + med.getLatitud() + "\n" +
00821             "Longitud: " + med.getLongitud());
00822
00823     }
00824
00825     private void mostrarTrama(TramaIBeacon t) {
00826         txt_datos.setText("ID: " + t.getId() + "\n" +
00827             "Título: " + t.getTitulo() + "\n" +
00828             "Latitud: " + t.getLatitud() + "\n" +
00829             "Longitud: " + t.getLongitud());
00830
00831     }
00832
00833     private void mostrarMediciones(List<Medicion> mediciones) {
00834         txt_datos.setText("Mostrando " + mediciones.size() + " mediciones");
00835
00836         for (Medicion med : mediciones) {
00837             mostrarMedicion(med);
00838         }
00839     }
00840
00841     private void mostrarTramas(List<TramaIBeacon> tramas) {
00842         txt_datos.setText("Mostrando " + tramas.size() + " tramas");
00843
00844         for (TramaIBeacon t : tramas) {
00845             mostrarTrama(t);
00846         }
00847     }
00848
00849     private void mostrarError(String error) {
00850         txt_datos.setText(error);
00851     }
00852
00853     private void iniciar() {
00854         elScanner.startScan(callbackDelEscaneo);
00855     }
00856
00857     private void finalizar() {
00858         elScanner.stopScan(callbackDelEscaneo);
00859     }
00860
00861     private void limpiar() {
00862         txt_datos.setText("");
00863     }
00864
00865     private void mostrarMedicion(Medicion med) {
00866         txt_datos.setText("ID: " + med.getId() + "\n" +
00867             "Título: " + med.getTitulo() + "\n" +
00868             "Latitud: " + med.getLatitud() + "\n" +
00869             "Longitud: " + med.getLongitud());

```

```

00073         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): onScanResult() ");
00074
00075         mostrarInformacionDispositivoBTLE(resultado);
00076         return null;
00077     }
00078
00079     @Override
00080     public void onBatchScanResults(List<ScanResult> results) {
00081         super.onBatchScanResults(results);
00082         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): onBatchScanResults() ");
00083     }
00084
00085     @Override
00086     public void onScanFailed(int errorCode) {
00087         super.onScanFailed(errorCode);
00088         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): onScanFailed() ");
00089     }
00090 }
00091 ;
00092 ;
00093
00094     Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): empezamos a escanear ");
00095
00096     if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00097         PackageManager.PERMISSION_GRANTED) {
00098         Log.d(ETIQUETA_LOG, " buscarTodosLosDispositivosBTL(): NO tengo permisos para escanear ");
00099         ActivityCompat.requestPermissions(
00100             MainActivity.this,
00101             new String[]{Manifest.permission.BLUETOOTH_SCAN},
00102             CODIGO_PETICION_PERMISOS);
00103     }
00104     this.elScanner.startScan(this.callbackDelEscaneo);
00105
00106 } // ()
00107 */
00108
00109     private void mostrarInformacionDispositivoBTLE(ScanResult resultado) {
00110
00111     BluetoothDevice bluetoothDevice = resultado.getDevice();
00112     byte[] bytes = resultado.getScanRecord().getBytes();
00113     int rssi = resultado.getRssi();
00114
00115     Log.d(ETIQUETA_LOG, " *****");
00116     Log.d(ETIQUETA_LOG, " ***** DISPOSITIVO DETECTADO BTLE ***** ");
00117     Log.d(ETIQUETA_LOG, " *****");
00118     if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) !=
00119         PackageManager.PERMISSION_GRANTED) {
00120         Log.d(ETIQUETA_LOG, " mostrarInformacionDispositivoBTLE(): NO tengo permisos para
00121         conectar ");
00122         ActivityCompat.requestPermissions(
00123             MainActivity.this,
00124             new String[]{Manifest.permission.BLUETOOTH_CONNECT},
00125             CODIGO_PETICION_PERMISOS);
00126     }
00127     return;
00128 }
00129
00130     Log.d(ETIQUETA_LOG, " nombre = " + bluetoothDevice.getName());
00131     Log.d(ETIQUETA_LOG, " toString = " + bluetoothDevice.toString());
00132
00133     /*
00134     ParcelUuid[] puuids = bluetoothDevice.getUuids();
00135     if (puuids.length >= 1) {
00136         //Log.d(ETIQUETA_LOG, " uuid = " + puuids[0].getUuid());
00137         // Log.d(ETIQUETA_LOG, " uuid = " + puuids[0].toString());
00138     }*/
00139
00140     Log.d(ETIQUETA_LOG, " dirección = " + bluetoothDevice.getAddress());
00141     Log.d(ETIQUETA_LOG, " rssi = " + rssi);
00142
00143     Log.d(ETIQUETA_LOG, " bytes = " + new String(bytes));
00144     Log.d(ETIQUETA_LOG, " bytes (" + bytes.length + ") = " + Utilidades.bytesToHexString(bytes));
00145
00146     TramaIBeacon tib = new TramaIBeacon(bytes);
00147
00148     Log.d(ETIQUETA_LOG, " -----");
00149     Log.d(ETIQUETA_LOG, " prefijo = " + Utilidades.bytesToHexString(tib.getPrefijo()));
00150     Log.d(ETIQUETA_LOG, " advFlags = " + Utilidades.bytesToHexString(tib.getAdvFlags()));
00151     Log.d(ETIQUETA_LOG, " advHeader = " +
00152         Utilidades.bytesToHexString(tib.getAdvHeader()));
00153     Log.d(ETIQUETA_LOG, " companyID = " +
00154         Utilidades.bytesToHexString(tib.getCompanyID()));
00155     Log.d(ETIQUETA_LOG, " iBeacon type = " + Integer.toHexString(tib.getiBeaconType()));
00156     Log.d(ETIQUETA_LOG, " iBeacon length 0x = " +
00157         Integer.toHexString(tib.getiBeaconLength()) + " ( "
00158             + tib.getiBeaconLength() + " )");
00159     Log.d(ETIQUETA_LOG, " uid = " + Utilidades.bytesToHexString(tib.getUUID()));
00160     Log.d(ETIQUETA_LOG, " uid = " + Utilidades.bytesToString(tib.getUUID()));

```

```

00158     Log.d(ETIQUETA_LOG, " major = " + Utilidades.bytesToHexString(tib.getMajor()) + " ( "
00159         + Utilidades.bytesToInt(tib.getMajor()) + " ) ");
00160     Log.d(ETIQUETA_LOG, " minor = " + Utilidades.bytesToHexString(tib.getMinor()) + " ( "
00161         + Utilidades.bytesToInt(tib.getMinor()) + " ) ");
00162     Log.d(ETIQUETA_LOG, " txPower = " + Integer.toHexString(tib.getTxPower()) + " ( " +
00163         tib.getTxPower() + " ) ");
00164     Log.d(ETIQUETA_LOG, " *****");
00165 } // ()
00166
00167
00168 /* private void buscarEsteDispositivoBTLE(final String dispositivoBuscado) {
00169     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empieza ");
00170
00171     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): instalamos scan callback ");
00172
00173     // super.onScanResult(ScanSettings.SCAN_MODE_LOW_LATENCY, result); para ahorro de energia
00174
00175     this.callbackDelEscaneo = new ScanCallback() {
00176         @Override
00177         public void onScanResult(int callbackType, ScanResult resultado) {
00178             super.onScanResult(callbackType, resultado);
00179             //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanResult() ");
00180
00181             if (ActivityCompat.checkSelfPermission(MainActivity.this,
00182                 Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00183                 // TODO: Consider calling
00184                 // ActivityCompat#requestPermissions
00185                 // here to request the missing permissions, and then overriding
00186                 // public void onRequestPermissionsResult(int requestCode, String[] permissions,
00187                 // int[] grantResults)
00188                 // to handle the case where the user grants the permission. See the documentation
00189                 // for ActivityCompat#requestPermissions for more details.
00190                 return null;
00191             }
00192
00193             byte[] bytes = resultado.getScanRecord().getBytes();
00194             TramaIBeacon tib = new TramaIBeacon(bytes);
00195             if (Utilidades.bytesToString(tib.getUUID()).equals(dispositivoBuscado)) {
00196                 mostrarInformacionDispositivoBTLE(resultado);
00197                 txt_datos.setText("Major: " + (int) getMedicionesBeacon(resultado));
00198                 //insertarMedicion((int) getMedicionesBeacon(resultado));
00199             }
00200
00201             else {
00202                 //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanResult(): no es el
00203                 dispositivo buscado ");
00204             }
00205         }
00206     }
00207     @Override
00208     public void onBatchScanResults(List<ScanResult> results) {
00209         super.onBatchScanResults(results);
00210         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onBatchScanResults() ");
00211     }
00212     @Override
00213     public void onScanFailed(int errorCode) {
00214         super.onScanFailed(errorCode);
00215         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): onScanFailed() ");
00216     }
00217 }
00218
00219     ScanFilter sf = new ScanFilter.Builder().setDeviceName(dispositivoBuscado).build();
00220
00221     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empezamos a escanear buscando: " +
00222     dispositivoBuscado);
00223     //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): empezamos a escanear buscando: " +
00224     dispositivoBuscado
00225     //      + " -> " + Utilidades.stringToUUID( dispositivoBuscado ) );
00226
00227     if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00228         PackageManager.PERMISSION_GRANTED) {
00229         //Log.d(ETIQUETA_LOG, " buscarEsteDispositivoBTLE(): NO tengo permisos para escanear ");
00230         ActivityCompat.requestPermissions(
00231             MainActivity.this,
00232             new String[]{Manifest.permission.BLUETOOTH_SCAN},
00233             CODIGO_PETICION_PERMISOS);
00234         return;
00235     }
00236     this.elScanner.startScan(this.callbackDelEscaneo);
00237 } // ()
```

```

00243 */
00244
00248     private void detenerBusquedaDispositivosBTLE() {
00249
00250         if (this.callbackDelEscaneo == null) {
00251             return;
00252         }
00253
00254         if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) != PackageManager.PERMISSION_GRANTED) {
00255             Log.d(ETIQUETA_LOG, " detenerBusquedaDispositivosBTLE(): NO tengo permisos para escanear ");
00256             ActivityCompat.requestPermissions(
00257                 MainActivity.this,
00258                 new String[]{Manifest.permission.BLUETOOTH_SCAN},
00259                 CODIGO_PETICION_PERMISOS);
00260             return;
00261         }
00262         this.elScanner.stopScan(this.callbackDelEscaneo);
00263         this.callbackDelEscaneo = null;
00264
00265     } // ()
00266
00267
00273     public void botonBuscarDispositivosBTLEPulsado(View v) {
00274         Log.d(ETIQUETA_LOG, " boton buscar dispositivos BTLE Pulsado");
00275         //this.buscarTodosLosDispositivosBTLE();
00276     } // ()
00277
00278
00284     public void botonBuscarNuestroDispositivoBTLEPulsado(View v) {
00285         Log.d(ETIQUETA_LOG, " boton nuestro dispositivo BTLE Pulsado");
00286         //this.buscarEsteDispositivoBTLE( Utilidades.stringToUUID( "EPSG-GTI-PROY-3A" ) );
00287
00288         // this.buscarEsteDispositivoBTLE( "EPSG-GTI-PROY-3D" );
00289         // this.buscarEsteDispositivoBTLE("fistro");
00290
00291     } // ()
00292
00293
00299     public void botonDetenerBusquedaDispositivosBTLEPulsado(View v) {
00300         Log.d(ETIQUETA_LOG, " boton detener busqueda dispositivos BTLE Pulsado");
00301         this.detenerBusquedaDispositivosBTLE();
00302     } // ()
00303
00304
00308     private void inicializarBlueTooth() {
00309         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): obtenemos adaptador BT ");
00310
00311         BluetoothAdapter bta = BluetoothAdapter.getDefaultAdapter();
00312
00313         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): habilitamos adaptador BT ");
00314
00315         if (ActivityCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00316             Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): NO tengo permisos para conectar ");
00317             ActivityCompat.requestPermissions(
00318                 MainActivity.this,
00319                 new String[]{Manifest.permission.BLUETOOTH_CONNECT},
00320                 CODIGO_PETICION_PERMISOS);
00321             return;
00322         }
00323         bta.enable();
00324
00325         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): habilitado = " + bta.isEnabled());
00326
00327         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): estado = " + bta.getState());
00328
00329         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): obtenemos escaner btle ");
00330
00331         this.elScanner = bta.getBluetoothLeScanner();
00332
00333         if ( this.elScanner == null ) {
00334             Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): Socorro: NO hemos obtenido escaner btle
00335             !!!!!");
00336         }
00337
00338         Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): voy a pedir permisos (si no los tuviera)
00339             !!!!!");
00340
00341         if (
00342             ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH) != PackageManager.PERMISSION_GRANTED
00343             || ContextCompat.checkSelfPermission(this,
00344                 Manifest.permission.BLUETOOTH_ADMIN) != PackageManager.PERMISSION_GRANTED
00345             || ContextCompat.checkSelfPermission(this,

```

```

        Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
00344            )
00345            {
00346                ActivityCompat.requestPermissions(
00347                    MainActivity.this,
00348                    new String[]{Manifest.permission.BLUETOOTH, Manifest.permission.BLUETOOTH_ADMIN,
00349                        Manifest.permission.ACCESS_FINE_LOCATION},
00350                        CODIGO_PETICION_PERMISOS);
00351            }
00352            else {
00353                Log.d(ETIQUETA_LOG, " inicializarBlueTooth(): parece que YA tengo los permisos necesarios
00354                !!!");
00355            }
00356        } // ()
00357
00358
00359    private void verificarPermisosBluetooth() {
00360        Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Verificando permisos");
00361
00362        // Verificamos si los permisos necesarios ya están concedidos
00363        if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH) !=
00364            PackageManager.PERMISSION_GRANTED
00365                || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_ADMIN) !=
00366            PackageManager.PERMISSION_GRANTED
00367                || ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
00368                != PackageManager.PERMISSION_GRANTED
00369                || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00370            PackageManager.PERMISSION_GRANTED
00371                || ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) !=
00372            PackageManager.PERMISSION_GRANTED) {
00373
00374            // Si no están concedidos, los solicitamos
00375            Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Solicitando permisos necesarios");
00376            ActivityCompat.requestPermissions(
00377                MainActivity.this,
00378                new String[]{
00379                    Manifest.permission.BLUETOOTH,
00380                    Manifest.permission.BLUETOOTH_ADMIN,
00381                    Manifest.permission.ACCESS_FINE_LOCATION,
00382                    Manifest.permission.BLUETOOTH_SCAN,
00383                    Manifest.permission.BLUETOOTH_CONNECT
00384                },
00385                CODIGO_PETICION_PERMISOS
00386            );
00387        } else {
00388            Log.d(ETIQUETA_LOG, " verificarPermisosBluetooth(): Permisos ya concedidos");
00389            inicializarBlueTooth(); // Llamamos a la inicialización de Bluetooth si los permisos
00390            // están concedidos
00391        }
00392    } // ()
00393
00394
00395    @Override
00396    protected void onCreate(Bundle savedInstanceState) {
00397        super.onCreate(savedInstanceState);
00398        setContentView(R.layout.activity_main);
00399
00400        Log.d(ETIQUETA_LOG, " onCreate(): empieza ");
00401
00402        // Verificamos los permisos antes de cualquier otra operación
00403        verificarPermisosBluetooth();
00404        //Button enviarMedicionButton = findViewById(R.id.btn_insertar);
00405        //txt_datos = findViewById(R.id.Txt_datos);
00406
00407    } // onCreate()
00408
00409
00410
00411    @Override
00412    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)
00413    {
00414        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
00415
00416        switch (requestCode) {
00417            case CODIGO_PETICION_PERMISOS:
00418                if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
00419                    Log.d(ETIQUETA_LOG, " onRequestPermissionsResult(): Permisos concedidos");
00420                    inicializarBlueTooth(); // Llamamos a la inicialización si se conceden los
00421                    permisos
00422                } else {
00423                    Log.d(ETIQUETA_LOG, " onRequestPermissionsResult(): Permisos NO concedidos");
00424                }
00425            return;
00426        }
00427    } // ()
00428
00429
00430
00431    } // ()
00432

```

```

00433     /*private void insertarMedicion(int major) {
00434         RetrofitClient.getApiService().insertarMedicion(new Medicion("Living Room", "CO2",
00435             major)).enqueue(new Callback<Void>() {
00436             @Override
00437             public void onResponse(Call<Void> call, Response<Void> response) {
00438                 if (response.isSuccessful()) {
00439                     Log.d("API Response", "Medición insertada correctamente");
00440                 } else {
00441                     Log.d("API Error", "Error en la respuesta: " + response.code());
00442                 }
00443             }
00444             @Override
00445             public void onFailure(Call<Void> call, Throwable t) {
00446                 Log.e("API Failure", "Error al conectar con el servidor", t);
00447             }
00448         });
00449     }*/
00450
00451     public double getMedicionesBeacon(ScanResult resultado) {
00452         byte[] bytes = resultado.getScanRecord().getBytes();
00453         TramaIBeacon tib = new TramaIBeacon(bytes);
00454         return Utilidades.bytesToInt(tib.getMinor());
00455     }
00456 }
00457 } // class

```

## 7.27. Referencia del archivo

[AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

Clase que representa una medición de gas en un lugar determinado.

### Clases

- class [com.example.smariba\\_upv.btle\\_sento.POJO.Medicion](#)

*Clase que encapsula los datos de una medición de gas, incluyendo el lugar, el tipo de gas y su valor.*

### Paquetes

- package [com.example.smariba\\_upv.btle\\_sento.POJO](#)

#### 7.27.1. Descripción detallada

Clase que representa una medición de gas en un lugar determinado.

### Versión

1.0

### Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/Medicion.java](#)

## 7.28. AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_↔ upv/btle\_sento/POJO/Medicion.java

[Ir a la documentación de este archivo.](#)

```
00001  
00008 package com.example.smariba_upv.btle_sento.POJO;  
00009  
00010  
00014 public class Medicion {  
00015     private String Lugar;  
00016     private String Gas;  
00017     private int Valor;  
00018  
00019  
00025     public Medicion(String Lugar, String Gas, int Valor) {  
00026         this.Lugar = Lugar;  
00027         this.Gas = Gas;  
00028         this.Valor = Valor;  
00029     }  
00030  
00031  
00035     public String getLugar() {  
00036         return Lugar;  
00037     }  
00038  
00039  
00043     public void setLugar(String lugar) {  
00044         Lugar = lugar;  
00045     }  
00046  
00047  
00051     public String getGas() {  
00052         return Gas;  
00053     }  
00054  
00055  
00059     public void setGas(String gas) {  
00060         Gas = gas;  
00061     }  
00062  
00063  
00067     public int getValor() {  
00068         return Valor;  
00069     }  
00070  
00071  
00075     public void setValor(int valor) {  
00076         Valor = valor;  
00077     }  
00078 }
```

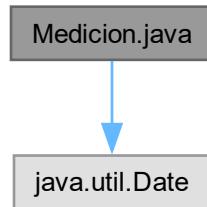
## 7.29. Referencia del archivo Desktop/Clases/Proyecto Bio/AirFlow-↔ Android/app/src/main/java/com/example/smariba\_upv/airflow/POJO/Medicion.java

Clase que representa una medición de gas en un lugar determinado.

```
import java.util.Date;
```

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba↔

\_upv/airflow/POJO/Medicion.java:



## Clases

- class [com.example.smariba\\_upv.airflow.POJO.Medicion](#)

## Paquetes

- package [com.example.smariba\\_upv.airflow.POJO](#)

### 7.29.1. Descripción detallada

Clase que representa una medición de gas en un lugar determinado.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/Medicion.java](#)

## 7.30. Desktop/Clases/Proyecto Bio/AirFlow- Android/app/src/main/java/com/example/smariba\_upv/airflow/POJO/Medicion.java

[Ir a la documentación de este archivo.](#)

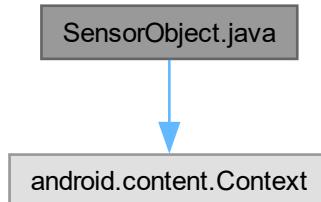
```
00001
00008 package com.example.smariba_upv.airflow.POJO;
00009
00010 import java.util.Date;
00011
00012
00013 public class Medicion {
00014     int id, idSenor;
00015     String tipoGas;
00016     double latitud, longitud, valor;
00017     Date fecha;
00018 }
```

```
00019     public Medicion(int id, int idSenor, String tipoGas, double latitud, double longitud, double
00020         valor) {
00021         this.id = id;
00022         this.idSenor = idSenor;
00023         this.tipoGas = tipoGas;
00024         this.latitud = latitud;
00025         this.longitud = longitud;
00026         this.valor = valor;
00027         this.fecha = new Date(); // Asignar la fecha actual
00028     }
00029     public Medicion( int idSenor, String tipoGas, double latitud, double longitud, double valor) {
00030         this.idSenor = idSenor;
00031         this.tipoGas = tipoGas;
00032         this.latitud = latitud;
00033         this.longitud = longitud;
00034         this.valor = valor;
00035         this.fecha = new Date(); // Asignar la fecha actual
00036     }
00037     public int getId() {
00038         return id;
00039     }
00040     public void setId(int id) {
00041         this.id = id;
00042     }
00043     public int getIdSenor() {
00044         return idSenor;
00045     }
00046     public void setIdSenor(int idSenor) {
00047         this.idSenor = idSenor;
00048     }
00049     public String getTipoGas() {
00050         return tipoGas;
00051     }
00052     public void setTipoGas(String tipoGas) {
00053         this.tipoGas = tipoGas;
00054     }
00055     public double getLatitud() {
00056         return latitud;
00057     }
00058     public void setLatitud(double latitud) {
00059         this.latitud = latitud;
00060     }
00061     public double getLongitud() {
00062         return longitud;
00063     }
00064     public void setLongitud(double longitud) {
00065         this.longitud = longitud;
00066     }
00067     public double getValor() {
00068         return valor;
00069     }
00070     public void setValor(double valor) {
00071         this.valor = valor;
00072     }
00073     public Date getFecha() {
00074         return fecha;
00075     }
00076     public void setFecha(Date fecha) {
00077         this.fecha = fecha;
00078     }
00079 }
```

### 7.31. Referencia del archivo SensorObject.java

```
import android.content.Context;
```

Gráfico de dependencias incluidas en SensorObject.java:



#### Clases

- class com.example.smariba\_upv.airflow.POJO.SensorObject

#### Paquetes

- package com.example.smariba\_upv.airflow.POJO

### 7.32. SensorObject.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.POJO;
00002
00003 import android.content.Context;
00004 import android.content.Intent;
00005
00006 import com.example.smariba_upv.airflow.Services.ArduinoGetterService;
00007 import com.google.gson.annotations.SerializedName;
00008
00009 public class SensorObject {
00010     int id;
00011     @SerializedName("estado")
00012     private String estado;
00013
00014     @SerializedName("num_referencia")
00015     private String num_ref;
00016
00017     @SerializedName("uuid")
00018     private String UUID;
00019
00020     @SerializedName("nombre")
00021     private String nombre;
00022
00023     @SerializedName("conexion")
00024     private boolean conexion;
00025
00026     @SerializedName("bateria")
00027     private int Bateria;
00028     public SensorObject(int id, String estado, String num_ref, String UUID, String nombre, boolean
00029             conexion, int bateria) {
00030         this.id = id;
00031         this.estado = estado;
00032         this.num_ref = num_ref;
00033         this.UUID = UUID;
00034         this.nombre = nombre;
00035     }
00036 }
```

```
00034         this.conexion = conexion;
00035         Bateria = bateria;
00036     }
00037     public SensorObject( String estado, String num_ref, String UUID, String nombre, boolean conexion,
00038             int bateria) {
00039         this.estado = estado;
00040         this.num_ref = num_ref;
00041         this.UUID = UUID;
00042         this.nombre = nombre;
00043         this.conexion = conexion;
00044         Bateria = bateria;
00045     }
00046     public int getId() {
00047         return id;
00048     }
00049     public void setId(int id) {
00050         this.id = id;
00051     }
00052     public String getEstado() {
00053         return estado;
00054     }
00055     public void setEstado(String estado) {
00056         this.estado = estado;
00057     }
00058     public String getNum_ref() {
00059         return num_ref;
00060     }
00061     public void setNum_ref(String num_ref) {
00062         this.num_ref = num_ref;
00063     }
00064     public String getUUID() {
00065         return UUID;
00066     }
00067     public void setUUID(String UUID) {
00068         this.UUID = UUID;
00069     }
00070     public String getNombre() {
00071         return nombre;
00072     }
00073     public void setNombre(String nombre) {
00074         this.nombre = nombre;
00075     }
00076     public boolean isConexion() {
00077         return conexion;
00078     }
00079     public void setConexion(boolean conexion) {
00080         this.conexion = conexion;
00081     }
00082     public int getBateria() {
00083         return Bateria;
00084     }
00085     public void setBateria(int bateria) {
00086         Bateria = bateria;
00087     }
00088     @Override
00089     public String toString() {
00090         return "SensorObject{" +
00091                 "estado='"+ estado + '\'' +
00092                 ", num_ref='"+ num_ref + '\'' +
00093                 ", UUID='"+ UUID + '\'' +
00094                 ", nombre='"+ nombre + '\'' +
00095                 ", conexion='"+ conexion +
00096                 ", bateria='"+ Bateria +
00097                 '}';
00098     }
00099 }
```

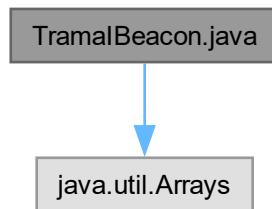
### 7.33. Referencia del archivo

[AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramalBeacon.java](#)

Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes.

```
import java.util.Arrays;
```

Gráfico de dependencias incluidas en [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramalBeacon.java](#):



#### Clases

- class [com.example.smariba\\_upv.btle\\_sento.POJO.TramalBeacon](#)

*Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.*

#### Paquetes

- package [com.example.smariba\\_upv.btle\\_sento.POJO](#)

##### 7.33.1. Descripción detallada

Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [AndroidStudioProjects/BTLE\\_Sento/app/src/main/java/com/example/smariba\\_upv/btle\\_sento/POJO/TramalBeacon.java](#)

### 7.34. AndroidStudioProjects/BTLE\_Sento/app/src/main/java/com/example/smariba\_upv/btle\_sento/ ↵ POJO/TramaIBeacon.java

[Ir a la documentación de este archivo.](#)

```
00001
00008 package com.example.smariba_upv.btle_sento.POJO;
00009
00010 import java.util.Arrays;
00011
00012
00016 public class TramaIBeacon {
00017     private byte[] prefijo = null; // 9 bytes
00018     private byte[] uuid = null; // 16 bytes
00019     private byte[] major = null; // 2 bytes
00020     private byte[] minor = null; // 2 bytes
00021     private byte txPower = 0; // 1 byte
00022
00023     private byte[] losBytes;
00024
00025     private byte[] advFlags = null; // 3 bytes
00026     private byte[] advHeader = null; // 2 bytes
00027     private byte[] companyID = new byte[2]; // 2 bytes
00028     private byte iBeaconType = 0; // 1 byte
00029     private byte iBeaconLength = 0; // 1 byte
00030
00031
00035     public TramaIBeacon(byte[] bytes) {
00036         this.losBytes = bytes;
00037
00038         prefijo = Arrays.copyOfRange(losBytes, 0, 8 + 1); // 9 bytes
00039         uuid = Arrays.copyOfRange(losBytes, 9, 24 + 1); // 16 bytes
00040         major = Arrays.copyOfRange(losBytes, 25, 26 + 1); // 2 bytes
00041         minor = Arrays.copyOfRange(losBytes, 27, 28 + 1); // 2 bytes
00042         txPower = losBytes[29]; // 1 byte
00043
00044         advFlags = Arrays.copyOfRange(prefijo, 0, 2 + 1); // 3 bytes
00045         advHeader = Arrays.copyOfRange(prefijo, 3, 4 + 1); // 2 bytes
00046         companyID = Arrays.copyOfRange(prefijo, 5, 6 + 1); // 2 bytes
00047         iBeaconType = prefijo[7]; // 1 byte
00048         iBeaconLength = prefijo[8]; // 1 byte
00049     }
00050
00051
00056     public byte[] getPrefijo() {
00057         return prefijo;
00058     }
00059
00060
00065     public byte[] getUUID() {
00066         return uuid;
00067     }
00068
00069
00074     public byte[] getMajor() {
00075         return major;
00076     }
00077
00078
00083     public byte[] getMinor() {
00084         return minor;
00085     }
00086
00087
00092     public byte getTxPower() {
00093         return txPower;
00094     }
00095
00096
00101     public byte[] getLosBytes() {
00102         return losBytes;
00103     }
00104
00105
00110     public byte[] getAdvFlags() {
00111         return advFlags;
00112     }
00113
00114
00119     public byte[] getAdvHeader() {
00120         return advHeader;
00121     }
00122
00123
00128     public byte[] getCompanyID() {
00129         return companyID;
```

```

00130      }
00131
00132
00137     public byte getiBeaconType() {
00138         return iBeaconType;
00139     }
00140
00141
00146     public byte getiBeaconLength() {
00147         return iBeaconLength;
00148     }
00149 } // class

```

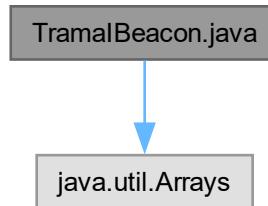
### 7.35. Referencia del archivo Desktop/Clases/Proyecto

**Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/POJO/TramalBeacon.java**

Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes.

import java.util.Arrays;

Gráfico de dependencias incluidas en Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/POJO/TramalBeacon.java:



#### Clases

- class [com.example.smariba\\_upv.airflow.POJO.TramalBeacon](#)

*Clase que modela los datos de una trama iBeacon y permite acceder a sus diferentes componentes, tales como UUID, Major, Minor y más.*

#### Paquetes

- package [com.example.smariba\\_upv.airflow.POJO](#)

#### 7.35.1. Descripción detallada

Clase que representa una trama iBeacon, extrayendo sus diferentes campos a partir de un array de bytes.

#### Versión

1.0

#### Fecha

2024

Definición en el archivo [Desktop/Clases/Proyecto Bio/AirFlow-Android/app/src/main/java/com/example/smariba\\_upv/airflow/POJO/TramalBeacon.java](#)

**7.36. Desktop/Clases/Proyecto**

**Bio/AirFlow-Android/app/src/main/java/com/example/smariba\_upv/airflow/POJO/TramaIBeacon.java**

[Ir a la documentación de este archivo.](#)

```

00001
00008 package com.example.smariba_upv.airflow.POJO;
00009
00010 import java.util.Arrays;
00011
00012
00016 public class TramaIBeacon {
00017     private byte[] prefijo = null; // 9 bytes
00018     private byte[] uuid = null; // 16 bytes
00019     private byte[] major = null; // 2 bytes
00020     private byte[] minor = null; // 2 bytes
00021     private byte txPower = 0; // 1 byte
00022
00023     private byte[] losBytes;
00024
00025     private byte[] advFlags = null; // 3 bytes
00026     private byte[] advHeader = null; // 2 bytes
00027     private byte[] companyID = new byte[2]; // 2 bytes
00028     private byte iBeaconType = 0; // 1 byte
00029     private byte iBeaconLength = 0; // 1 byte
00030
00031
00035     public TramaIBeacon(byte[] bytes) {
00036         this.losBytes = bytes;
00037
00038         prefijo = Arrays.copyOfRange(losBytes, 0, 8 + 1); // 9 bytes
00039         uuid = Arrays.copyOfRange(losBytes, 9, 24 + 1); // 16 bytes
00040         major = Arrays.copyOfRange(losBytes, 25, 26 + 1); // 2 bytes
00041         minor = Arrays.copyOfRange(losBytes, 27, 28 + 1); // 2 bytes
00042         txPower = losBytes[29]; // 1 byte
00043
00044         advFlags = Arrays.copyOfRange(prefijo, 0, 2 + 1); // 3 bytes
00045         advHeader = Arrays.copyOfRange(prefijo, 3, 4 + 1); // 2 bytes
00046         companyID = Arrays.copyOfRange(prefijo, 5, 6 + 1); // 2 bytes
00047         iBeaconType = prefijo[7]; // 1 byte
00048         iBeaconLength = prefijo[8]; // 1 byte
00049     }
00050
00051
00056     public byte[] getPrefijo() {
00057         return prefijo;
00058     }
00059
00060
00065     public byte[] getUUID() {
00066         return uuid;
00067     }
00068
00069
00074     public byte[] getMajor() {
00075         return major;
00076     }
00077
00078
00083     public byte[] getMinor() {
00084         return minor;
00085     }
00086
00087
00092     public byte getTxPower() {
00093         return txPower;
00094     }
00095
00096
00101     public byte[] getLosBytes() {
00102         return losBytes;
00103     }
00104
00105
00110     public byte[] getAdvFlags() {
00111         return advFlags;
00112     }
00113
00114
00119     public byte[] getAdvHeader() {
00120         return advHeader;
00121     }
00122
00123

```

```

00128     public byte[] getCompanyID() {
00129         return companyID;
00130     }
00131
00132
00137     public byte getiBeaconType() {
00138         return iBeaconType;
00139     }
00140
00141
00146     public byte getiBeaconLength() {
00147         return iBeaconLength;
00148     }
00149 } // class

```

## 7.37. Referencia del archivo User.java

### Clases

- class [com.example.smariba\\_upv.airflow.POJO.User](#)

### Paquetes

- package [com.example.smariba\\_upv.airflow.POJO](#)

## 7.38. User.java

[Ir a la documentación de este archivo.](#)

```

00001 package com.example.smariba_upv.airflow.POJO;
00002
00003 public class User {
00004     private String nombre;      // Coincide con "nombre" en el JSON
00005     private String apellidos;  // Coincide con "apellidos" en el JSON
00006     private String email;     // Coincide con "email" en el JSON
00007     private String telefono;  // Coincide con "telefono" en el JSON
00008     private String contrasenya; // Coincide con "contrasenya" en el JSON
00009     private int id;           // Coincide con "id" en el JSON
00010
00020     public User(String nombre, String apellidos, String email, String telefono, String contrasenya) {
00021         this.nombre = nombre;
00022         this.apellidos = apellidos;
00023         this.email = email;
00024         this.telefono = telefono;
00025         this.contrasenya = contrasenya;
00026     }
00027
00034     public User(String email, String contrasenya) {
00035         this.email = email;
00036         this.contrasenya = contrasenya;
00037     }
00038
00048     public User(int id, String nombre, String apellidos, String email, String telefono) {
00049         this.id = id;
00050         this.nombre = nombre;
00051         this.apellidos = apellidos;
00052         this.email = email;
00053         this.telefono = telefono;
00054     }
00055
00060     public String getNombre() {
00061         return nombre;
00062     }
00063
00068     public void setNombre(String nombre) {
00069         this.nombre = nombre;
00070     }
00075     public String getApellidos() {
00076         return apellidos;
00077     }
00082     public void setApellidos(String apellidos) {
00083         this.apellidos = apellidos;
00084     }
00089     public String getEmail() {

```

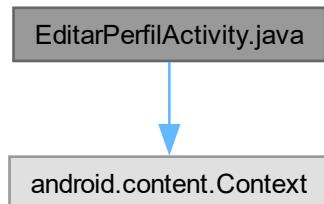
```
00090         return email;
00091     }
00096     public void setEmail(String email) {
00097         this.email = email;
00098     }
00103     public String getTelefono() {
00104         return telefono;
00105     }
00110     public void setTelefono(String telefono) {
00111         this.telefono = telefono;
00112     }
00117     public String getContrasenya() {
00118         return contrasenya;
00119     }
00124     public void setContrasenya(String contrasenya) {
00125         this.contrasenya = contrasenya;
00126     }
00131     public int getId() {
00132         return id;
00133     }
00138     public void setId(int id) {
00139         this.id = id;
00140     }
00141 }
```

### 7.39. Referencia del archivo EditarPerfilActivity.java

Clase que permite editar el perfil del usuario.

```
import android.content.Context;
```

Gráfico de dependencias incluidas en EditarPerfilActivity.java:



#### Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.EditarPerfilActivity](#)  
*Clase que permite editar el perfil del usuario.*

#### Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

##### 7.39.1. Descripción detallada

Clase que permite editar el perfil del usuario.

Clase que permite editar el perfil del usuario y guardar los cambios realizados

Definición en el archivo [EditarPerfilActivity.java](#).

## 7.40. EditarPerfilActivity.java

[Ir a la documentación de este archivo.](#)

```

00001 package com.example.smariba_upv.airflow.PRESENTACION;
00008 import android.content.Context;
00009 import android.content.SharedPreferences;
00010 import android.os.Bundle;
00011 import android.util.Log;
00012 import android.widget.Button;
00013 import android.widget.EditText;
00014 import android.widget.Toast;
00015
00016 import androidx.activity.EdgeToEdge;
00017 import androidx.appcompat.app.AppCompatActivity;
00018
00019 import com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil;
00020 import com.example.smariba_upv.airflow.R;
00026 public class EditarPerfilActivity extends AppCompatActivity {
00035     EditText NombreEditText;
00036     EditText ApellidosEditText;
00037     EditText EmailEditText;
00038     EditText TelefonoEditText;
00039     Button GuardarCambiosButton;
00040     Button VolverButton;
00041
00046     //Bundle:saveInstanceState => void
00047     @Override
00048     protected void onCreate(Bundle savedInstanceState) {
00049         super.onCreate(savedInstanceState);
00050         EdgeToEdge.enable(this);
00051         setContentView(R.layout.activity_editar_perfil);
00052
00053         NombreEditText = findViewById(R.id.TxtIUser);
00054         ApellidosEditText = findViewById(R.id.TxtIApellidos);
00055         EmailEditText = findViewById(R.id.TxtICorreo);
00056         TelefonoEditText = findViewById(R.id.TxtITelefono);
00057
00058         GuardarCambiosButton = findViewById(R.id.btn_Guardar);
00059         VolverButton = findViewById(R.id.btn_Volver);
00060
00061         cargarDatosUsuario();
00062
00063         GuardarCambiosButton.setOnClickListener(v -> {
00064             guardarCambios();
00065         });
00066
00067         VolverButton.setOnClickListener(v -> {
00068             finish();
00069         });
00070     }
00076     // En el método guardarCambios() de EditarPerfilActivity
00077     public void guardarCambios() {
00078         // recoger los datos de los campos de texto
00079         String nombre = NombreEditText.getText().toString();
00080         String apellidos = ApellidosEditText.getText().toString();
00081         String email = EmailEditText.getText().toString();
00082         String telefono = TelefonoEditText.getText().toString();
00083
00084         // guardar los datos en la cache y recuperar el id del usuario
00085         SharedPreferences sharedpreferences = getSharedPreferences("MyAppPrefs",
00086             Context.MODE_PRIVATE);
00086         SharedPreferences.Editor editor = sharedpreferences.edit();
00087         int id = sharedpreferences.getInt("id", 0);
00088         String contrasena = sharedpreferences.getString("contrasena", "contrasena no disponible");
00089         editor.putString("nombre", nombre);
00090         editor.putString("apellidos", apellidos);
00091         editor.putString("email", email);
00092         editor.putString("telefono", telefono);
00093         editor.apply();
00094         Toast.makeText(this, "Datos guardados con éxito", Toast.LENGTH_SHORT).show();
00095
00096         // Log the values to debug
00097         Log.d("EditarPerfilActivity", "ID: " + id);
00098         Log.d("EditarPerfilActivity", "Nombre: " + nombre);
00099         Log.d("EditarPerfilActivity", "Apellidos: " + apellidos);
00100         Log.d("EditarPerfilActivity", "Email: " + email);
00101         Log.d("EditarPerfilActivity", "Telefono: " + telefono);
00102
00103         // enviar datos a la base de datos
00104         PeticionesUserUtil peticionesUserUtil = new PeticionesUserUtil();
00105         peticionesUserUtil.editUsuario(id, nombre, apellidos, email, telefono, contrasena, this);
00106
00107         // cerrar la actividad y devolver el resultado
00108         setResult(RESULT_OK);
00109         finish();
00110     }

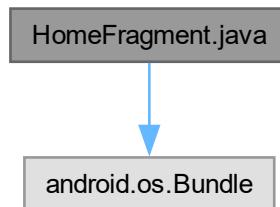
```

```

0016     private void cargarDatosUsuario() {
0017         SharedPreferences sharedpreferences = getSharedPreferences("MyAppPrefs",
0018             Context.MODE_PRIVATE);
0018         String nombre = sharedpreferences.getString("nombre", "Nombre no disponible");
0019         String apellidos = sharedpreferences.getString("apellidos", "Apellidos no disponibles");
0020         String correo = sharedpreferences.getString("email", "Correo no disponible");
0021         String telefono = sharedpreferences.getString("Telefono", "Teléfono no disponible");
0022
0023         NombreEditText.setText(nombre);
0024         ApellidosEditText.setText(apellidos);
0025         EmailEditText.setText(correo);
0026         TelefonoEditText.setText(telefono);
0027     }
0028 }
0029 }
```

#### 7.41. Referencia del archivo HomeFragment.java

import android.os.Bundle;  
 Gráfico de dependencias incluidas en HomeFragment.java:



#### Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment  
*A simple Fragment subclass.*

#### Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

#### 7.42. HomeFragment.java

[Ir a la documentación de este archivo.](#)

```

0001 package com.example.smariba_upv.airflow.PRESENTACION;
0002
0003 import android.os.Bundle;
0004
0005 import androidx.fragment.app.Fragment;
0006
0007 import android.view.LayoutInflater;
0008 import android.view.View;
0009 import android.view.ViewGroup;
0010
0011 import com.example.smariba_upv.airflow.R;
0012
0019 public class HomeFragment extends Fragment {
```

```

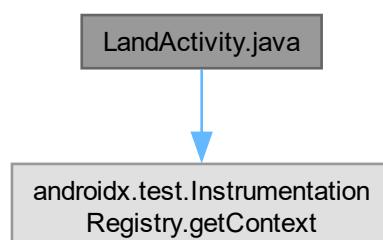
00020
00021 // TODO: Rename parameter arguments, choose names that match
00022 // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
00023 private static final String ARG_PARAM1 = "param1";
00024 private static final String ARG_PARAM2 = "param2";
00025
00026 // TODO: Rename and change types of parameters
00027 private String mParam1;
00028 private String mParam2;
00029
00030 public HomeFragment() {
00031     // Required empty public constructor
00032 }
00033
00042 // TODO: Rename and change types and number of parameters
00043 public static HomeFragment newInstance(String param1, String param2) {
00044     HomeFragment fragment = new HomeFragment();
00045     Bundle args = new Bundle();
00046     args.putString(ARG_PARAM1, param1);
00047     args.putString(ARG_PARAM2, param2);
00048     fragment.setArguments(args);
00049     return fragment;
00050 }
00051
00052 @Override
00053 public void onCreate(Bundle savedInstanceState) {
00054     super.onCreate(savedInstanceState);
00055     if (getArguments() != null) {
00056         mParam1 = getArguments().getString(ARG_PARAM1);
00057         mParam2 = getArguments().getString(ARG_PARAM2);
00058     }
00059 }
00060
00061 @Override
00062 public View onCreateView(LayoutInflater inflater, ViewGroup container,
00063                          Bundle savedInstanceState) {
00064     // Inflate the layout for this fragment
00065     return inflater.inflate(R.layout.fragment_home, container, false);
00066 }
00067 }

```

## 7.43. Referencia del archivo LandActivity.java

Clase que permite visualizar la pantalla principal de la aplicación.

import androidx.test.InstrumentationRegistry.getContext;  
 Gráfico de dependencias incluidas en LandActivity.java:



### Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.LandActivity

## Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

### 7.43.1. Descripción detallada

Clase que permite visualizar la pantalla principal de la aplicación.

Clase que permite visualizar la pantalla principal de la aplicación y navegar entre las diferentes opciones de la aplicación

Definición en el archivo [LandActivity.java](#).

## 7.44. LandActivity.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00007 import static androidx.test.InstrumentationRegistry.getContext;
00008
00009 import android.content.Context;
00010 import android.content.Intent;
00011 import android.os.Bundle;
00012 import androidx.annotation.NonNull;
00013 import androidx.appcompat.app.AppCompatActivity;
00014 import androidx.fragment.app.Fragment;
00015
00016 import com.example.smariba_upv.airflow.POJO.SensorObject;
00017 import com.example.smariba_upv.airflow.R;
00018 import com.example.smariba_upv.airflow.Services.ArduinoGetterService;
00019 import com.google.android.material.bottomnavigation.BottomNavigationView;
00020
00021 import java.util.List;
00022
00023 import android.Manifest;
00024 import android.content.SharedPreferences;
00025 import android.content.pm.PackageManager;
00026 import androidx.core.app.ActivityCompat;
00027 import androidx.core.content.ContextCompat;
00028 import android.util.Log;
00029 import android.view.MenuItem;
00030 import com.google.gson.Gson;
00031 import com.google.gson.reflect.TypeToken;
00032
00033 import java.lang.reflect.Type;
00034 import java.util.ArrayList;
00035
00036 public class LandActivity extends AppCompatActivity {
00041     private static final int REQUEST_BLUETOOTH_PERMISSIONS = 1;
00042     private BottomNavigationView bottomNavigationView;
00043
00044     @Override
00050     protected void onCreate(Bundle savedInstanceState) {
00051         super.onCreate(savedInstanceState);
00052         setContentView(R.layout.activity_land);
00053
00054         bottomNavigationView = findViewById(R.id.navtabgroup);
00055         bottomNavigationView.setOnNavigationItemSelected(this::onNavigationItemSelected);
00056
00057         if (savedInstanceState == null) {
00058             loadFragment(new HomeFragment());
00059             bottomNavigationView.setSelectedItemId(R.id.item_1);
00060         }
00061
00062         // Check Bluetooth permissions
00063         if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED ||
00064             ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) != PackageManager.PERMISSION_GRANTED) {
00065             ActivityCompat.requestPermissions(this, new String[]{
00066                 Manifest.permission.BLUETOOTH_CONNECT,
00067                 Manifest.permission.BLUETOOTH_SCAN
00068             }, REQUEST_BLUETOOTH_PERMISSIONS);
00069         } else {
00070             connectSensors();
```

```

00071         }
00072     }
00073
00081     @Override
00082     public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull
00083     int[] grantResults) {
00083         super.onRequestPermissionsResult(requestCode, permissions, grantResults);
00084         if (requestCode == REQUEST_BLUETOOTH_PERMISSIONS) {
00085             if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
00086                 connectSensors();
00087             } else {
00088                 Log.e("LandActivity", "Bluetooth permissions denied.");
00089             }
00090         }
00091     }
00092
00098     private void connectSensors() {
00099         // Get sensor data from Shared Preferences
00100         SharedPreferences sharedpreferences = this.getSharedPreferences("MyAppPrefs",
00101             Context.MODE_PRIVATE);
00101         String jsonSensores = sharedpreferences.getString("sensores", null);
00102         if (jsonSensores != null) {
00103             try {
00104                 Gson gson = new Gson();
00105                 Type type = new TypeToken<List<SensorObject>>() {}.getType();
00106                 List<SensorObject> sensores = gson.fromJson(jsonSensores, type);
00107                 for (SensorObject sensor : sensores) {
00108                     startSensorService();
00109                 }
00110             } catch (Exception e) {
00111                 Log.e("LandActivity", "Error parsing sensor data.");
00112             }
00113         }
00114     }
00120     private void startSensorService() {
00121         Log.d("LandActivity", "Starting ArduinoGetterService...");
00122         Intent serviceIntent = new Intent(this, ArduinoGetterService.class);
00123         ContextCompat.startForegroundService(this, serviceIntent);
00124         Log.d("LandActivity", "ArduinoGetterService started.");
00125     }
00132     private boolean onNavigationItemSelected(@NonNull MenuItem item) {
00133         Fragment selectedFragment = null;
00134
00135         if (item.getItemId() == R.id.item_1) {
00136             selectedFragment = new HomeFragment();
00137         } else if (item.getItemId() == R.id.item_2) {
00138             selectedFragment = new MapFragment();
00139         } else if (item.getItemId() == R.id.item_3) {
00140             selectedFragment = new SaludFragment();
00141         } else if (item.getItemId() == R.id.item_4) {
00142             selectedFragment = new SensorFragment();
00143         } else if (item.getItemId() == R.id.item_5) {
00144             selectedFragment = new PerfilFragment();
00145         }
00146
00147         return loadFragment(selectedFragment);
00148     }
00155     private boolean loadFragment(Fragment fragment) {
00156         if (fragment != null) {
00157             getSupportFragmentManager()
00158                 .beginTransaction()
00159                 .replace(R.id.fragment_container, fragment)
00160                 .commit();
00161             return true;
00162         }
00163         return false;
00164     }
00165 }

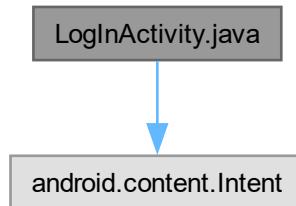
```

## 7.45. Referencia del archivo LoginActivity.java

Clase que permite iniciar sesión en la aplicación.

```
import android.content.Intent;
```

Gráfico de dependencias incluidas en LoginActivity.java:



## Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.LoginActivity](#)  
*Clase que permite iniciar sesión en la aplicación.*

## Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

### 7.45.1. Descripción detallada

Clase que permite iniciar sesión en la aplicación.

Clase que permite iniciar sesión en la aplicación mediante correo electrónico y contraseña o mediante autenticación biométrica

Definición en el archivo [LoginActivity.java](#).

## 7.46. LoginActivity.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002 import android.content.Intent;
00003 import android.content.pm.PackageManager;
00004 import android.os.Bundle;
00005 import android.text.Html;
00006 import android.util.Log;
00007 import android.view.LayoutInflater;
00008 import android.view.View;
00009 import android.widget.Button;
00010 import android.widget.CheckBox;
00011 import android.widget.EditText;
00012 import android.widget.ImageButton;
00013 import android.widget.TextView;
00014
00015 import androidx.annotation.NonNull;
00016 import androidx.appcompat.app.AlertDialog;
00017 import androidx.appcompat.app.AppCompatActivity;
00018 import androidx.core.app.ActivityCompat;
00019
00020 import androidx.activity.OnBackPressedCallback;
00021 import androidx.annotation.Nullable;
00022 import androidx.biometric.BiometricPrompt;
00023 import androidx.core.app.ActivityCompat;
```

```

00026 import androidx.core.content.ContextCompat;
00027
00028 import com.example.smariba_upv.airflow.API.EnviarPeticionesUser;
00029 import com.example.smariba_upv.airflow.LOGIC.BiometricUtil;
00030 import com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil;
00031 import android.Manifest;
00032 import com.example.smariba_upv.airflow.R;
00033
00034 import java.util.ArrayList;
00035 import java.util.List;
00041 public class LogInActivity extends AppCompatActivity implements BiometricUtil.BiometricAuthListener {
00054     private static final String TAG = "LogInActivity";
00055     EditText EmailEditText;
00056     EditText passwordEditText;
00057     Button loginButton;
00058     ImageButton biometricButton;
00059     TextView errorText, forgotPasswordText;
00060     CheckBox Condiciones, rememberMeCheckBox;
00061     private static final int REQUEST_BLUETOOTH_PERMISSIONS = 1;
00062
00068 //Bundle: savedInstanceState => onCreate():void
00069 @Override
00070 protected void onCreate(Bundle savedInstanceState) {
00071     super.onCreate(savedInstanceState);
00072     EdgeToEdge.enable(this);
00073     setContentView(R.layout.activity_log_in);
00074
00075     // Verificar y solicitar permisos de ubicación, notificación y acceso a dispositivos cercanos
00076     checkAndRequestPermissions();
00077
00078     // Inicializar elementos
00079     EmailEditText = findViewById(R.id.correoInput);
00080     passwordEditText = findViewById(R.id.PasswordInput);
00081     loginButton = findViewById(R.id.logIn_btn);
00082     biometricButton = findViewById(R.id.biometric_btn);
00083     errorText = findViewById(R.id.errorText);
00084     forgotPasswordText = findViewById(R.id.forgotPasswordText);
00085     Condiciones = findViewById(R.id.ChekCondi);
00086     rememberMeCheckBox = findViewById(R.id.rememberMeCheckBox);
00087
00088     // Configuración de eventos
00089     Condiciones.setOnClickListener(v -> showConditionsPopup());
00090     forgotPasswordText.setOnClickListener(v -> showForgotPasswordPopup());
00091 }
00097 private void checkAndRequestPermissions() {
00098     String[] permissions = {
00099         Manifest.permission.ACCESS_FINE_LOCATION,    // Permiso de ubicación precisa
00100         Manifest.permission.ACCESS_COARSE_LOCATION, // Permiso de ubicación general
00101         Manifest.permission.BLUETOOTH_CONNECT,        // Permiso de acceso a dispositivos
00102         Manifest.permission.POST_NOTIFICATIONS        // Permiso para mostrar notificaciones
00103     };
00104
00105     // Lista para permisos que faltan
00106     List<String> permissionsToRequest = new ArrayList<>();
00107
00108     for (String permission : permissions) {
00109         if (ContextCompat.checkSelfPermission(this, permission) != PackageManager.PERMISSION_GRANTED) {
00110             permissionsToRequest.add(permission);
00111         }
00112     }
00113
00114     // Si hay permisos que solicitar, hacer la solicitud
00115     if (!permissionsToRequest.isEmpty()) {
00116         ActivityCompat.requestPermissions(
00117             this,
00118             permissionsToRequest.toArray(new String[0]),
00119             REQUEST_BLUETOOTH_PERMISSIONS
00120         );
00121     }
00122 }
00123
00132     // requestCode:int, permissions:String[], grantResults:int[] => onRequestPermissionsResult():void
00133 @Override
00134 public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull
int[] grantResults) {
00135     super.onRequestPermissionsResult(requestCode, permissions, grantResults);
00136
00137     if (requestCode == REQUEST_BLUETOOTH_PERMISSIONS) {
00138         for (int i = 0; i < permissions.length; i++) {
00139             if (grantResults[i] == PackageManager.PERMISSION_GRANTED) {
00140                 Log.d(TAG, "Permiso concedido: " + permissions[i]);
00141             } else {
00142                 Log.d(TAG, "Permiso denegado: " + permissions[i]);
00143             }
}

```

```
00144         }
00145     }
00146 }
00147
00152     private void showConditionsPopup() {
00153         AlertDialog.Builder builder = new AlertDialog.Builder(this);
00154         builder.setTitle("Condiciones de Uso");
00155
00156 // Obtener el texto con formato HTML desde los recursos
00157     String termsHtml = getString(R.string.terms_and_conditions);
00158
00159 // Convertir HTML a texto para Android
00160     CharSequence formattedText;
00161     if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.N) {
00162         formattedText = Html.fromHtml(termsHtml, Html.FROM_HTML_MODE_LEGACY);
00163     } else {
00164         formattedText = Html.fromHtml(termsHtml);
00165     }
00166
00167 // Configurar el mensaje del popup
00168     builder.setMessage(formattedText);
00169
00170 // Añadir botón de aceptación
00171     builder.setPositiveButton("Aceptar", (dialog, which) -> {
00172         dialog.dismiss();
00173     });
00174
00175 // Mostrar el popup
00176     builder.create().show();
00177
00178
00179
00180
00181 }
00186     private void showForgotPasswordPopup() {
00187         AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogStyle);
00188         LayoutInflater inflater = getLayoutInflater();
00189         View dialogView = inflater.inflate(R.layout.popup_forgot_password, null);
00190         builder.setView(dialogView);
00191
00192         EditText emailInput = dialogView.findViewById(R.id.forgotPasswordEmailInput);
00193         Button sendButton = dialogView.findViewById(R.id.forgotPasswordSendButton);
00194
00195         AlertDialog dialog = builder.create();
00196         sendButton.setOnClickListener(v -> {
00197             String email = emailInput.getText().toString();
00198             if (email.isEmpty()) {
00199                 emailInput.setError("Por favor, ingresa tu correo electrónico.");
00200             } else if (!android.util.Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
00201                 emailInput.setError("Correo electrónico no válido.");
00202             } else {
00203                 // Lógica para enviar correo de recuperación
00204                 dialog.dismiss();
00205                 errorText.setText("Correo de recuperación enviado.");
00206             }
00207         });
00208
00209         dialog.show();
00210     }
00215 //View view => logIn() -> void
00216 public void logIn(View view) {
00217     String emailInput = EmailEditText.getText().toString();
00218     String passwordInput = passwordEditText.getText().toString();
00219
00220     if (emailInput.isEmpty() || passwordInput.isEmpty()) {
00221         errorText.setText("Por favor, completa todos los campos.");
00222         return;
00223     }
00224
00225     if (!android.util.Patterns.EMAIL_ADDRESS.matcher(emailInput).matches()) {
00226         errorText.setText("Correo electrónico no válido.");
00227         return;
00228     }
00229
00230     if (!Condiciones.isChecked()) {
00231         errorText.setText("Por favor, acepta las condiciones.");
00232         return;
00233     }
00234
00235     boolean loginSuccessful = PeticionesUserUtil.login(emailInput, passwordInput, this);
00236
00237     if (!loginSuccessful) {
00238         errorText.setText("Usuario o contraseña incorrectos.");
00239     } else {
00240         errorText.setText("");
00241     }
00242 }
```

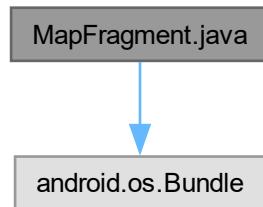
```

00247 //View view => biometricLogIn() => void
00248     public void biometricLogIn(View view) {
00249         BiometricUtil.authenticate(this, this);
00250     }
00255 //BiometricPrompt.AuthenticationResult result => onAuthenticationSucceeded() => void
00256 @Override
00257     public void onAuthenticationSucceeded(BiometricPrompt.AuthenticationResult result) {
00258         Intent intent = new Intent(LogInActivity.this, LandActivity.class);
00259         startActivity(intent);
00260         finish();
00261     }
00265 @Override
00266     public void onAuthenticationFailed() {
00267         errorText.setText("Fallo en la autenticación");
00268         Log.d(TAG, "Fallo en la autenticación");
00269     }
00275 //int errorCode, CharSequence errString => onAuthenticationError() => void
00276 @Override
00277     public void onAuthenticationError(int errorCode, CharSequence errString) {
00278         errorText.setText("Error de autenticación");
00279         Log.d(TAG, "Error de autenticación: " + errString);
00280     }
00281 }

```

#### 7.47. Referencia del archivo MapFragment.java

import android.os.Bundle;  
 Gráfico de dependencias incluidas en MapFragment.java:



#### Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.MapFragment  
*A simple Fragment subclass.*

#### Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

#### 7.48. MapFragment.java

[Ir a la documentación de este archivo.](#)

```

00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002
00003 import android.os.Bundle;
00004
00005 import androidx.fragment.app.Fragment;

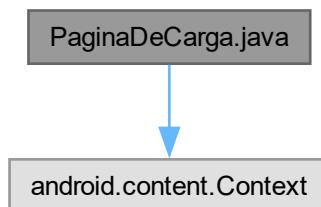
```

```

00006
00007 import android.view.LayoutInflater;
00008 import android.view.View;
00009 import android.view.ViewGroup;
00010
00011 import com.example.smariba_upv.airflow.R;
00012
00013 public class MapFragment extends Fragment {
00014
00015     // TODO: Rename parameter arguments, choose names that match
00016     // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
00017     private static final String ARG_PARAM1 = "param1";
00018     private static final String ARG_PARAM2 = "param2";
00019
00020     // TODO: Rename and change types of parameters
00021     private String mParam1;
00022     private String mParam2;
00023
00024     public MapFragment() {
00025         // Required empty public constructor
00026     }
00027
00028     // TODO: Rename and change types and number of parameters
00029     public static MapFragment newInstance(String param1, String param2) {
00030         MapFragment fragment = new MapFragment();
00031         Bundle args = new Bundle();
00032         args.putString(ARG_PARAM1, param1);
00033         args.putString(ARG_PARAM2, param2);
00034         fragment.setArguments(args);
00035         return fragment;
00036     }
00037
00038     @Override
00039     public void onCreate(Bundle savedInstanceState) {
00040         super.onCreate(savedInstanceState);
00041         if (getArguments() != null) {
00042             mParam1 = getArguments().getString(ARG_PARAM1);
00043             mParam2 = getArguments().getString(ARG_PARAM2);
00044         }
00045     }
00046
00047     @Override
00048     public View onCreateView(LayoutInflater inflater, ViewGroup container,
00049                             Bundle savedInstanceState) {
00050         // Inflate the layout for this fragment
00051         return inflater.inflate(R.layout.fragment_map, container, false);
00052     }
00053 }
```

## 7.49. Referencia del archivo PaginaDeCarga.java

import android.content.Context;  
 Gráfico de dependencias incluidas en PaginaDeCarga.java:



### Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCarga

## Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

### 7.50. PaginaDeCarga.java

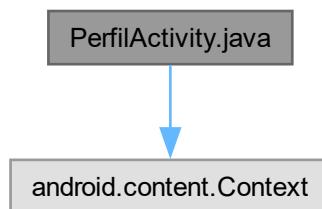
[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002
00003 import android.content.Context;
00004 import android.content.Intent;
00005 import android.content.SharedPreferences;
00006 import android.os.Bundle;
00007
00008 import androidx.activity.EdgeToEdge;
00009 import androidx.appcompat.app.AppCompatActivity;
00010
00011 import com.example.smariba_upv.airflow.R;
00012 //intent import
00013
00014
00015 public class PaginaDeCarga extends AppCompatActivity {
00016
00017     @Override
00018     protected void onCreate(Bundle savedInstanceState) {
00019         super.onCreate(savedInstanceState);
00020         EdgeToEdge.enable(this);
00021         setContentView(R.layout.activity_pagina_de_carga);
00022         SharedPreferences sharedpreferences = getSharedPreferences("MyAppPrefs",
00023             Context.MODE_PRIVATE);
00024         boolean isLoggedIn = sharedpreferences.getBoolean("isLoggedIn", false);
00025
00026
00027         if (isLoggedIn) {
00028             // El usuario ya está logueado, ir directamente a MainActivity
00029             Intent intent = new Intent(this, LandActivity.class);
00030             startActivity(intent);
00031             finish(); // Cierra la actividad actual
00032         } else {
00033             // El usuario no está logueado, mostrar la pantalla de inicio de sesión
00034             Intent intent = new Intent(this, LogInActivity.class);
00035             startActivity(intent);
00036             finish(); // Cierra la actividad actual
00037         }
00038     }
00039 }
00040 }
```

### 7.51. Referencia del archivo PerfilActivity.java

```
import android.content.Context;
```

Gráfico de dependencias incluidas en PerfilActivity.java:



## Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity

## Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

## 7.52. PerfilActivity.java

[Ir a la documentación de este archivo.](#)

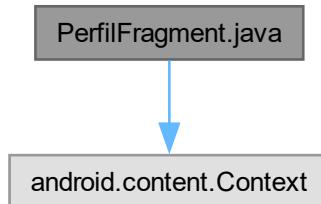
```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002
00003 import android.content.Context;
00004 import android.content.Intent;
00005 import android.content.SharedPreferences;
00006 import android.os.Bundle;
00007 import android.widget.Button;
00008 import android.widget.TextView;
00009
00010 import androidx.activity.EdgeToEdge;
00011 import androidx.appcompat.app.AppCompatActivity;
00012
00013 import com.example.smariba_upv.airflow.R;
00014
00015 public class PerfilActivity extends AppCompatActivity {
00016
00017     public TextView txtNombre;
00018     public TextView txtApellidos;
00019     public TextView txtCorreo;
00020     public TextView txtTelefono;
00021     public Button btnEditarPerfil;
00022
00023     //Bundle: savedInstanceState => onCreate():void
00024     @Override
00025     protected void onCreate(Bundle savedInstanceState) {
00026         super.onCreate(savedInstanceState);
00027         EdgeToEdge.enable(this);
00028         setContentView(R.layout.activity_perfil);
00029
00030         txtNombre = findViewById(R.id.textVNombre);
00031         txtApellidos = findViewById(R.id.textVApellidos);
00032         txtCorreo = findViewById(R.id.textVCorreo);
00033         txtTelefono = findViewById(R.id.textVTelefono);
00034         btnEditarPerfil = findViewById(R.id.btnEditarUser);
00035
00036         cargarDatosUsuario();
00037         btnEditarPerfil.setOnClickListener(v -> {
00038             // Abrir la actividad de editar perfil
00039             Intent intent = new Intent(PerfilActivity.this, EditarPerfilActivity.class);
00040             startActivity(intent);
00041         });
00042     }
00043
00044     // Recoger datos del usuario de la cache guardada al hacer el login
00045     private void cargarDatosUsuario() {
00046         SharedPreferences sharedpreferences = getSharedPreferences("MyAppPrefs",
00047             Context.MODE_PRIVATE);
00048         String nombre = sharedpreferences.getString("nombre", "Nombre no disponible");
00049         String apellidos = sharedpreferences.getString("apellidos", "Apellidos no disponibles");
00050         String correo = sharedpreferences.getString("email", "Correo no disponible");
00051         String telefono = sharedpreferences.getString("Telefono", "Teléfono no disponible");
00052
00053         txtNombre.setText(nombre);
00054         txtApellidos.setText(apellidos);
00055         txtCorreo.setText(correo);
00056         txtTelefono.setText(telefono);
00057     }
00058 }
```

### 7.53. Referencia del archivo PerfilFragment.java

Clase que permite visualizar el perfil del usuario.

```
import android.content.Context;
```

Gráfico de dependencias incluidas en PerfilFragment.java:



#### Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.PerfilFragment](#)

*Clase que permite visualizar el perfil del usuario.*

#### Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

##### 7.53.1. Descripción detallada

Clase que permite visualizar el perfil del usuario.

Clase que permite visualizar el perfil del usuario y editar la información del usuario

Definición en el archivo [PerfilFragment.java](#).

### 7.54. PerfilFragment.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002 import android.content.Context;
00003 import android.content.Intent;
00004 import android.content.SharedPreferences;
00005 import android.os.Bundle;
00006 import androidx.fragment.app.Fragment;
00007 import androidx.appcompat.app.AppCompatActivity;
00008 import android.view.LayoutInflater;
00009 import android.view.View;
00010 import android.view.ViewGroup;
00011 import android.widget.Button;
00012 import android.widget.TextView;
00013 import com.example.smariba_upv.airflow.R;
00014
00015 public class PerfilFragment extends Fragment {
```

```

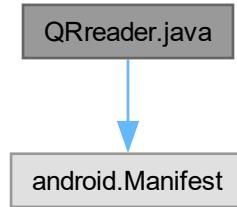
00033     private TextView Bienvenido;
00034     private TextView perfilNombreCompleto;
00035     private TextView numeroTelefono;
00036     private TextView correoElectronico;
00037     private Button editarPerfilButton;
00038
00043     public PerfilFragment() {
00044         // Required empty public constructor
00045     }
00046
00053     public static PerfilFragment newInstance(String param1, String param2) {
00054         PerfilFragment fragment = new PerfilFragment();
00055         Bundle args = new Bundle();
00056         args.putString("param1", param1);
00057         args.putString("param2", param2);
00058         fragment.setArguments(args);
00059         return fragment;
00060     }
00065     @Override
00066     public void onCreate(Bundle savedInstanceState) {
00067         super.onCreate(savedInstanceState);
00068         if (getArguments() != null) {
00069             String mParam1 = getArguments().getString("param1");
00070             String mParam2 = getArguments().getString("param2");
00071         }
00072     }
00073     @Override
00081     public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
00082     {
00083         View view = inflater.inflate(R.layout.fragment_perfil, container, false);
00084
00085         // Initialize TextViews
00086         Bienvenido = view.findViewById(R.id.Bienvenida);
00087         perfilNombreCompleto = view.findViewById(R.id.perfilNombreCompleto);
00088         numeroTelefono = view.findViewById(R.id.perfilNumeroTelefonico);
00089         correoElectronico = view.findViewById(R.id.perfilCorreo);
00090         editarPerfilButton = view.findViewById(R.id.perfilBtEditarPerfil);
00091
00092         // Editar perfil button
00093         editarPerfilButton.setOnClickListener(v -> {
00094             Intent intent = new Intent(getActivity(), EditarPerfilActivity.class);
00095             startActivityForResult(intent, 1);
00096         });
00097
00098         // Load data from SharedPreferences
00099         loadUserData();
00100
00101         return view;
00102     }
00103     // N:requestCode,N:resultCode,Intent:data => onActivityResult():void
00104     @Override
00110     public void onActivityResult(int requestCode, int resultCode, Intent data) {
00111         super.onActivityResult(requestCode, resultCode, data);
00112         if (requestCode == 1 & resultCode == AppCompatActivity.RESULT_OK) {
00113             // Actualizar los datos del usuario
00114             loadUserData();
00115         }
00116     }
00117
00123     private void loadUserData() {
00124         SharedPreferences sharedpreferences = getActivity().getSharedPreferences("MyAppPrefs",
00125             Context.MODE_PRIVATE);
00126
00127         String nombre = sharedpreferences.getString("nombre", "Nombre no disponible");
00128         String apellidos = sharedpreferences.getString("apellidos", "Apellidos no disponibles");
00129         String email = sharedpreferences.getString("email", "Correo electrónico no disponible");
00130         String telefono = sharedpreferences.getString("telefono", "Número telefónico no disponible");
00131
00132         Bienvenido.setText("Bienvenido, " + nombre);
00133         perfilNombreCompleto.setText(nombre + " " + apellidos);
00134         numeroTelefono.setText(telefono);
00135         correoElectronico.setText(email);
00136     }
00136 }
```

## 7.55. Referencia del archivo QRreader.java

Clase que permite leer un código QR.

```
import android.Manifest;
```

Gráfico de dependencias incluidas en QRReader.java:



## Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.QRReader](#)  
*Clase que permite leer un código QR.*

## Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

### 7.55.1. Descripción detallada

Clase que permite leer un código QR.

Clase que permite leer un código QR y registrar un sensor en la base de datos

Definición en el archivo [QRReader.java](#).

## 7.56. QRreader.java

[Ir a la documentación de este archivo.](#)

```
00001
00007 package com.example.smariba_upv.airflow.PRESENTACION;
00008
00009 import android.Manifest;
00010 import android.content.pm.PackageManager;
00011 import android.os.Build;
00012 import android.os.Bundle;
00013 import android.util.Log;
00014 import android.util.SparseArray;
00015 import android.view.SurfaceHolder;
00016 import android.view.SurfaceView;
00017 import android.widget.Button;
00018 import android.widget.Toast;
00019
00020 import androidx.appcompat.app.AppCompatActivity;
00021 import androidx.core.app.ActivityCompat;
00022
00023 import com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil;
00024 import com.example.smariba_upv.airflow.R;
00025 import com.google.android.gms.vision.CameraSource;
00026 import com.google.android.gms.vision.Detector;
```

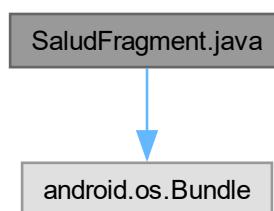
```
00027 import com.google.android.gms.vision.barcode.Barcode;
00028 import com.google.android.gms.vision.barcode.BarcodeDetector;
00029
00030 import java.io.IOException;
00031
00037 public class QRReader extends AppCompatActivity {
00045     private CameraSource cameraSource;
00046     private SurfaceView cameraView;
00047     private final int MY_PERMISSIONS_REQUEST_CAMERA = 1;
00048     private String token = "";
00049     private String tokenanterior = "";
00050
00056     @Override
00057     protected void onCreate(Bundle savedInstanceState) {
00058         super.onCreate(savedInstanceState);
00059         setContentView(R.layout.activity_qrreader);
00060
00061         cameraView = findViewById(R.id.camera_view);
00062         Button btnCancel = findViewById(R.id.btn_cancel);
00063         btnCancel.setOnClickListener(v -> finish());
00064         initQR();
00065     }
00066
00072     public void initQR() {
00073
00079         // creo el detector qr
00080         BarcodeDetector barcodeDetector =
00081             new BarcodeDetector.Builder(this)
00082                 .setBarcodeFormats(Barcode.ALL_FORMATS)
00083                 .build();
00084
00085         // creo la camara
00086         cameraSource = new CameraSource
00087             .Builder(this, barcodeDetector)
00088                 .setRequestedPreviewSize(1600, 1024)
00089                 .setAutoFocusEnabled(true) //you should add this feature
00090                 .build();
00091
00092         // listener de ciclo de vida de la camara
00093         cameraView.getHolder().addCallback(new SurfaceHolder.Callback() {
00094             @Override
00095             public void surfaceCreated(SurfaceHolder holder) {
00096
00097                 // verifico si el usuario dio los permisos para la camara
00098                 if (ActivityCompat.checkSelfPermission(QRReader.this, Manifest.permission.CAMERA)
00099                     != PackageManager.PERMISSION_GRANTED) {
00100
00101                     if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
00102                         // verificamos la version de ANDROID que sea al menos la M para mostrar
00103                         // el dialog de la solicitud de la camara
00104                         if (shouldShowRequestPermissionRationale(
00105                             Manifest.permission.CAMERA)) ;
00106                         requestPermissions(new String[]{Manifest.permission.CAMERA},
00107                             MY_PERMISSIONS_REQUEST_CAMERA);
00108                     }
00109                     return;
00110                 } else {
00111                     try {
00112                         cameraSource.start(cameraView.getHolder());
00113                     } catch (IOException ie) {
00114                         Log.e("CAMERA SOURCE", ie.getMessage());
00115                     }
00116                 }
00117             }
00125             @Override
00126             public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
00127             }
00132             @Override
00133             public void surfaceDestroyed(SurfaceHolder holder) {
00134                 cameraSource.stop();
00135             }
00136         });
00137
00143         barcodeDetector.setProcessor(new Detector.Processor<Barcode>() {
00144             @Override
00145             public void release() {
00146             }
00147
00148             @Override
00149             public void receiveDetections(Detector.Detections<Barcode> detections) {
00150                 final SparseArray<Barcode> barcodes = detections.getDetectedItems();
00151
00152                 if (barcodes.size() > 0) {
00153
00154                     // obtenemos el token
00155                     token = barcodes.valueAt(0).displayValue;
00156
00157             }
00158         }
00159     }
00160 }
```

```

00157
00158      // verificamos que el token anterior no se igual al actual
00159      // esto es util para evitar multiples llamadas empleando el mismo token
00160      if (!token.equals(tokenanterior)) {
00161          // guardamos el ultimo token procedido
00162          tokenanterior = token;
00163          Log.i("token", token);
00164
00165          // Mostrar el contenido del código QR en un Toast y cerrar la actividad
00166          runOnUiThread(() -> {
00167              Toast.makeText(QRreader.this, "QR Code: " + token,
00168                  Toast.LENGTH_LONG).show();
00169          //registrar sensor
00170          PeticionesUserUtil.registrarSensor(token, QRreader.this);
00171
00172          finish(); // Cerrar la actividad
00173      });
00174
00175      new Thread(new Runnable() {
00176          public void run() {
00177              try {
00178                  synchronized (this) {
00179                      wait(5000);
00180                      // limpiamos el token
00181                      tokenanterior = "";
00182                  }
00183                  catch (InterruptedException e) {
00184                      Log.e("Error", "Waiting didn't work!!!");
00185                      e.printStackTrace();
00186                  }
00187              }).start();
00188          }
00189      }
00190  );
00191 );
00192 }
00193 }
```

## 7.57. Referencia del archivo SaludFragment.java

import android.os.Bundle;  
 Gráfico de dependencias incluidas en SaludFragment.java:



### Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.SaludFragment](#)

### Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

## 7.58. SaludFragment.java

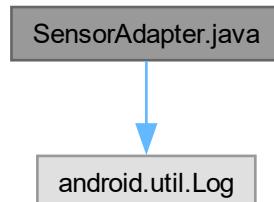
[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002
00003 import android.os.Bundle;
00004
00005 import androidx.fragment.app.Fragment;
00006
00007 import android.view.LayoutInflater;
00008 import android.view.View;
00009 import android.view.ViewGroup;
00010
00011 import com.example.smariba_upv.airflow.R;
00012
00013 //TODO: completar la clase
00014 public class SaludFragment extends Fragment {
00015
00016     // TODO: Rename parameter arguments, choose names that match
00017     // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
00018     private static final String ARG_PARAM1 = "param1";
00019     private static final String ARG_PARAM2 = "param2";
00020
00021     // TODO: Rename and change types of parameters
00022     private String mParam1;
00023     private String mParam2;
00024
00025     public SaludFragment() {
00026         // Required empty public constructor
00027     }
00028
00029     // TODO: Rename and change types and number of parameters
00030     public static SaludFragment newInstance(String param1, String param2) {
00031         SaludFragment fragment = new SaludFragment();
00032         Bundle args = new Bundle();
00033         args.putString(ARG_PARAM1, param1);
00034         args.putString(ARG_PARAM2, param2);
00035         fragment.setArguments(args);
00036         return fragment;
00037     }
00038
00039     @Override
00040     public void onCreate(Bundle savedInstanceState) {
00041         super.onCreate(savedInstanceState);
00042         if (getArguments() != null) {
00043             mParam1 = getArguments().getString(ARG_PARAM1);
00044             mParam2 = getArguments().getString(ARG_PARAM2);
00045         }
00046     }
00047
00048     @Override
00049     public View onCreateView(LayoutInflater inflater, ViewGroup container,
00050                             Bundle savedInstanceState) {
00051         // Inflate the layout for this fragment
00052         return inflater.inflate(R.layout.fragment_salud, container, false);
00053     }
00054 }
```

## 7.59. Referencia del archivo SensorAdapter.java

```
import android.util.Log;
```

Gráfico de dependencias incluidas en SensorAdapter.java:



### Clases

- class [com.example.smariba\\_upv.airflow.PRESENTACION.SensorAdapter](#)
- class [com.example.smariba\\_upv.airflow.PRESENTACION.SensorAdapter.ViewHolder](#)

### Paquetes

- package [com.example.smariba\\_upv.airflow.PRESENTACION](#)

## 7.60. SensorAdapter.java

[Ir a la documentación de este archivo.](#)

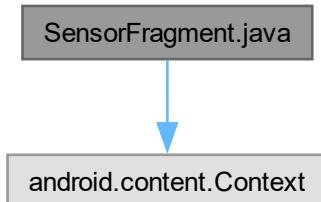
```

00001 package com.example.smariba_upv.airflow.PRESENTACION;
00002
00003 import android.util.Log;
00004 import android.view.LayoutInflater;
00005 import android.view.View;
00006 import android.view.ViewGroup;
00007 import android.widget.ProgressBar;
00008 import android.widget.TextView;
00009
00010 import androidx.annotation.NonNull;
00011 import androidx.recyclerview.widget.RecyclerView;
00012
00013 import com.example.smariba_upv.airflow.API.MODELS.SensorResponse;
00014 import com.example.smariba_upv.airflow.POJO.Medicion;
00015 import com.example.smariba_upv.airflow.POJO.SensorObject;
00016 import com.example.smariba_upv.airflow.R;
00017
00018 import java.util.ArrayList;
00019 import java.util.List;
00020
00021 public class SensorAdapter extends RecyclerView.Adapter<SensorAdapter.ViewHolder> {
00022     private final List<SensorObject> sensorList;
00023     private final List<Medicion> medicionList;
00024
00025     public SensorAdapter(List<SensorObject> sensorList, List<Medicion> medicionList) {
00026         this.sensorList = sensorList != null ? sensorList : new ArrayList<>();
00027         this.medicionList = medicionList != null ? medicionList : new ArrayList<>();
00028     }
00029
00030     @NonNull
00031     @Override
00032     public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
  
```

```
00049     View view = LayoutInflater.from(parent.getContext())
00050         .inflate(R.layout.sensor_item_layout, parent, false);
00051     return new SensorViewHolder(view);
00052 }
00053 @Override
00054 public void onBindViewHolder(@NonNull SensorViewHolder holder, int position) {
00055     if (position < sensorList.size()) {
00056         SensorObject sensor = sensorList.get(position);
00057
00058         // Configure sensor data
00059         holder.tvSensorName.setText(sensor.getNombre());
00060         holder.tvEstado.setText("Estado: " + sensor.getEstado());
00061         holder.tvNumReferencia.setText("Num. Referencia: " + sensor.getNum_ref());
00062         holder.tvConexion.setText("Conexión: " + (sensor.isConexion() ? "Conectado" :
00063             "Desconectado"));
00064         holder.tvBattery.setText("Battery: " + sensor.getBateria() + "%");
00065         holder.batteryIndicator.setProgress(
00066             Math.max(0, Math.min(sensor.getBateria(), 100))
00067         );
00068
00069         // Configure measurement data
00070         Medicion medicion = position < medicionList.size() ? medicionList.get(position) : null;
00071         if (medicion != null) {
00072             holder.tvGasType.setText("Gas Type: " + medicion.getTipoGas());
00073             holder.tvMeasurement.setText("Measurement: " + medicion.getValor());
00074             holder.tvDate.setText("Date: " + medicion.getFecha());
00075         } else {
00076             holder.tvGasType.setText("Gas Type: N/A");
00077             holder.tvMeasurement.setText("Measurement: N/A");
00078             holder.tvDate.setText("Date: N/A");
00079         }
00080     }
00081 }
00082 @Override
00083 public int getItemCount() {
00084     return sensorList.size();
00085 }
00086
00087 static class SensorViewHolder extends RecyclerView.ViewHolder {
00088
00089     TextView tvSensorName, tvEstado, tvNumReferencia, tvConexion, tvGasType, tvMeasurement,
00090     tvDate, tvBattery;
00091     ProgressBar batteryIndicator;
00092
00093     public SensorViewHolder(@NonNull View itemView) {
00094         super(itemView);
00095         tvSensorName = itemView.findViewById(R.id.tv_sensor_name);
00096         tvEstado = itemView.findViewById(R.id.tv_estado);
00097         tvNumReferencia = itemView.findViewById(R.id.tv_num_referencia);
00098         tvConexion = itemView.findViewById(R.id.tvConexion);
00099         tvGasType = itemView.findViewById(R.id.tv_gas_type);
00100         tvMeasurement = itemView.findViewById(R.id.tv_measurement);
00101         tvDate = itemView.findViewById(R.id.tv_date);
00102         tvBattery = itemView.findViewById(R.id.tv_battery);
00103         batteryIndicator = itemView.findViewById(R.id.battery_indicator);
00104     }
00105 }
00106 }
```

## 7.61. Referencia del archivo SensorFragment.java

import android.content.Context;  
 Gráfico de dependencias incluidas en SensorFragment.java:



### Clases

- class com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment

### Paquetes

- package com.example.smariba\_upv.airflow.PRESENTACION

## 7.62. SensorFragment.java

[Ir a la documentación de este archivo.](#)

```

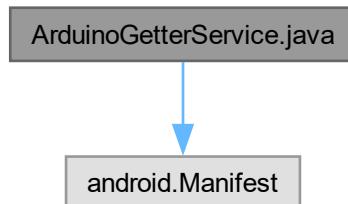
00001 // SensorFragment.java
00002 package com.example.smariba_upv.airflow.PRESENTACION;
00003
00004 import android.content.Context;
00005 import android.content.Intent;
00006 import android.content.SharedPreferences;
00007 import android.os.Bundle;
00008 import android.util.Log;
00009 import android.view.LayoutInflater;
00010 import android.view.View;
00011 import android.view.ViewGroup;
00012 import android.widget.Button;
00013 import android.widget.TextView;
00014
00015 import androidx.fragment.app.Fragment;
00016 import androidx.recyclerview.widget.LinearLayoutManager;
00017 import androidx.recyclerview.widget.RecyclerView;
00018
00019 import com.example.smariba_upv.airflow.API.EnviarPeticionesUser;
00020 import com.example.smariba_upv.airflow.POJO.Medicion;
00021 import com.example.smariba_upv.airflow.POJO.SensorObject;
00022 import com.example.smariba_upv.airflow.R;
00023 import com.google.gson.Gson;
00024 import com.google.gson.reflect.TypeToken;
00025
00026 import java.lang.reflect.Type;
00027 import java.util.ArrayList;
00028 import java.util.List;
00029 public class SensorFragment extends Fragment {
00030     private Button btnAddSensor;
00031     private RecyclerView recyclerView;
00032     private SensorAdapter sensorAdapter;
00033     private List<SensorObject> sensorList;
00034     private List<Medicion> medicionList;
  
```

```
00035     TextView tvNoSensors;
00036
00037     public SensorFragment() {
00038         // Required empty public constructor
00039     }
00040
00041     @Override
00042     public void onResume() {
00043         super.onResume();
00044         // Clear the list and fetch new data
00045         sensorList.clear();
00046         añadirSensoresVista();
00047         sensorAdapter.notifyDataSetChanged(); // Notify adapter
00048     }
00049
00050     private void añadirSensoresVista() {
00051         EnviarPeticionesUser enviarPeticionesUser = new EnviarPeticionesUser(getContext());
00052         enviarPeticionesUser.obtenerMisSensores(getContext());
00053
00054         // Get sensor data from Shared Preferences
00055         SharedPreferences sharedPreferences = getContext().getSharedPreferences("MyAppPrefs",
00056             Context.MODE_PRIVATE);
00056         String jsonSensores = sharedPreferences.getString("sensores", null);
00057
00058         if (jsonSensores != null) {
00059             try {
00060                 Gson gson = new Gson();
00061                 Type type = new TypeToken<List<SensorObject>>() {}.getType();
00062                 List<SensorObject> sensores = gson.fromJson(jsonSensores, type);
00063                 sensorList.addAll(sensores);
00064             } catch (Exception e) {
00065                 Log.e("SensorFragment", "Error parsing JSON: " + e.getMessage());
00066             }
00067         }
00068
00069         // Update view visibility
00070         updateView(tvNoSensors);
00071     }
00072
00073     @Override
00074     public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
00075     {
00076         View view = inflater.inflate(R.layout.fragment_sensor, container, false);
00077
00078         // Initialize views
00079         recyclerView = view.findViewById(R.id.rv_sensors);
00080         btnAddSensor = view.findViewById(R.id.add_sensor);
00081         tvNoSensors = view.findViewById(R.id.tv_no_sensors);
00082
00083         // Set up RecyclerView
00084         recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
00085
00086         // Initialize lists and adapter
00087         sensorList = new ArrayList<>();
00088         medicionList = new ArrayList<>();
00089         medicionList.add(new Medicion(2, "CO2", 0, 0, 10));
00090         sensorAdapter = new SensorAdapter(sensorList, medicionList);
00091         recyclerView.setAdapter(sensorAdapter);
00092
00093         // Add sensor button
00094         btnAddSensor.setOnClickListener(v -> {
00095             Intent intent = new Intent(getContext(), QRreader.class);
00096             startActivity(intent);
00097         });
00098
00099         // Update view
00100         updateView(tvNoSensors);
00101
00102         return view;
00103     }
00104
00105     private void updateView(TextView tvNoSensors) {
00106         if (sensorList.isEmpty()) {
00107             tvNoSensors.setVisibility(View.VISIBLE);
00108             recyclerView.setVisibility(View.GONE);
00109         } else {
00110             tvNoSensors.setVisibility(View.GONE);
00111             recyclerView.setVisibility(View.VISIBLE);
00112         }
00113 }
```

### 7.63. Referencia del archivo ArduinoGetterService.java

```
import android.Manifest;
```

Gráfico de dependencias incluidas en ArduinoGetterService.java:



#### Clases

- class [com.example.smariba\\_upv.airflow.Services.ArduinoGetterService](#)

#### Paquetes

- package [com.example.smariba\\_upv.airflow.Services](#)

### 7.64. ArduinoGetterService.java

[Ir a la documentación de este archivo.](#)

```
00001 package com.example.smariba_upv.airflow.Services;
00002
00003 import android.Manifest;
00004 import android.app.NotificationChannel;
00005 import android.app.NotificationManager;
00006 import android.app.PendingIntent;
00007 import android.app.Service;
00008 import android.bluetooth.BluetoothAdapter;
00009 import android.bluetooth.le.BluetoothLeScanner;
00010 import android.bluetooth.le.ScanCallback;
00011 import android.bluetooth.le.ScanFilter;
00012 import android.bluetooth.le.ScanResult;
00013 import android.content.Context;
00014 import android.content.Intent;
00015 import android.content.SharedPreferences;
00016 import android.content.pm.PackageManager;
00017 import android.location.Location;
00018 import android.location.LocationManager;
00019 import android.media.AudioAttributes;
00020 import android.net.Uri;
00021 import android.os.Build;
00022 import android.os.Handler;
00023 import android.os.IBinder;
00024 import android.util.Log;
00025 import android.widget.RemoteViews;
00026
00027 import androidx.annotation.Nullable;
00028 import androidx.core.app.ActivityCompat;
00029 import androidx.core.app.NotificationCompat;
00030 import androidx.core.content.ContextCompat;
00031
00032 import com.example.smariba_upv.airflow.API.EnviarPeticionesUser;
00033 import com.example.smariba_upv.airflow.LOGIC.Utilidades;
00034 import com.example.smariba_upv.airflow.POJO.Medicion;
```

```
00035 import com.example.smariba_upv.airflow.POJO.SensorObject;
00036 import com.example.smariba_upv.airflow.POJO.TramaIBeacon;
00037 import com.example.smariba_upv.airflow.PRESENTACION.LogInActivity;
00038 import com.example.smariba_upv.airflow.R;
00039
00040 import java.util.Date;
00041 import java.util.List;
00042
00043 public class ArduinoGetterService extends Service {
00044
00066     private static final String ETIQUETA_LOG = "ArduinoGetterService"; // Etiqueta de log
00067     private static final String BEACON_UUID = "EPSG-GTI-PROY-3D"; // UUID que estamos buscando
00068     private static final int CODIGO_PETICION_PERMISOS = 11223344;
00069     private static final String CHANNEL_ID = "ArduinoGetterServiceChannel";
00070     private static final long NOTIFICATION_INTERVAL = 1 * 60 * 1000; // 1 minutos
00071     private long lastNotificationTime = 0;
00072
00073     private BluetoothLeScanner elScanner;
00074     private ScanCallback callbackDelEscaneo = null;
00075     private static final int TIEMPO_DESCONEXION = 30000; // 30 segundos
00076     private Handler handler;
00077     private Runnable temporizador;
00078     private EnviarPeticionesUser enviarPeticionesUser = new EnviarPeticionesUser();
00079     private boolean dispositivoActualmenteConectado = false; // Estado actual de conexión.
00080
00081     private boolean dispositivoDetectado = false;
00082
00083     //private int minuto = 60000; // < 1 minuto en milisegundos.
00084
00094     @Override
00095     public void onCreate() {
00096         super.onCreate();
00097         createNotificationChannel(); // Asegúrate de crear el canal antes de usarlo
00098         startForegroundService();
00099         inicializarBlueTooth();
00100         buscarEsteDispositivoBTLE(BEACON_UUID);
00101
00102         // Configuración del temporizador para notificaciones
00103         handler = new Handler();
00104         temporizador = new Runnable() {
00105             @Override
00106             public void run() {
00107                 if (!dispositivoDetectado) {
00108                     enviarNotificacionDesconexion();
00109                 }
00110                 dispositivoDetectado = false; // Resetear para la siguiente verificación
00111                 handler.postDelayed(this, TIEMPO_DESCONEXION);
00112             }
00113         };
00114         handler.post(temporizador); // Iniciar el temporizador
00115     }
00116
00126     private void createNotificationChannel() {
00127         // Verificar si la versión de Android es 8.0 (API nivel 26) o superior
00128         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
00129             // Crear un objeto AudioAttributes para definir las características del sonido
00130             AudioAttributes audioAttributes = new AudioAttributes.Builder()
00131                 .setUsage(AudioAttributes.USAGE_NOTIFICATION)
00132                 .setContentType(AudioAttributes.CONTENT_TYPE_SONIFICATION)
00133                 .build();
00134
00135             // Supongamos que el archivo se llama 'notisound.wav'
00136             Uri soundUri = Uri.parse("android.resource://" + getPackageName() + "/" +
00137             R.raw.notisound);
00138
00139             NotificationChannel channel = new NotificationChannel(CHANNEL_ID, "Gas Notification
00140             Channel", NotificationManager.IMPORTANCE_HIGH);
00141             channel.setDescription("Canal para notificaciones de nivel de gas");
00142             channel.setSound(soundUri, audioAttributes);
00143
00144             // Registrar el canal de notificación
00145             NotificationManager notificationManager = (NotificationManager)
00146             getSystemService(Context.NOTIFICATION_SERVICE);
00147             if (notificationManager != null) {
00148                 notificationManager.createNotificationChannel(channel);
00149             }
00150
00156     @Override
00157     public void onDestroy() {
00158         super.onDestroy();
00159         if (handler != null) {
00160             handler.removeCallbacks(temporizador); // Detener el temporizador al detener el servicio
00161         }
00162     }
00163 }
```

```

00171     @Nullable
00172     @Override
00173     public IBinder onBind(Intent intent) {
00174         return null;
00175     }
00176     private void startForegroundService() {
00177         // Intent para abrir una actividad al interactuar con la notificación
00178         Intent notificationIntent = new Intent(this, LogInActivity.class);
00179         PendingIntent pendingIntent = PendingIntent.getActivity(
00180             this,
00181             0,
00182             notificationIntent,
00183             PendingIntent.FLAG_IMMUTABLE // Obligatorio en Android 12+ para mayor seguridad
00184         );
00185
00186         // Crear la notificación
00187         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00188             .setContentTitle("Arduino Getter Service")
00189             .setContentText("El servicio está corriendo en primer plano")
00190             .setSmallIcon(android.R.drawable.ic_dialog_info) // Asegúrate de que sea un ícono
00191             válido
00192             .setContentIntent(pendingIntent)
00193             .setPriority(NotificationCompat.PRIORITY_DEFAULT) // Prioridad adecuada para servicios
00194             en segundo plano
00195             .setOngoing(true); // Marca la notificación como persistente
00196
00197         // Inicia el servicio en primer plano con la notificación válida
00198         startForeground(1, builder.build());
00199     }
00200
00201
00202     private void inicializarBlueTooth() {
00203         Log.d(ETIQUETA_LOG, "inicializarBlueTooth(): obtenemos adaptador BT");
00204
00205         BluetoothAdapter bta = BluetoothAdapter.getDefaultAdapter();
00206
00207         if (bta == null) {
00208             Log.e(ETIQUETA_LOG, "BluetoothAdapter is null. Device does not support Bluetooth.");
00209             return;
00210         }
00211
00212         Log.d(ETIQUETA_LOG, "inicializarBlueTooth(): habilitamos adaptador BT");
00213
00214         if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00215             Log.d(ETIQUETA_LOG, "inicializarBlueTooth(): NO tengo permisos para conectar");
00216             // Handle permission request
00217             return;
00218         }
00219
00220         if (!bta.isEnabled()) {
00221             bta.enable();
00222             Log.d(ETIQUETA_LOG, "inicializarBlueTooth(): habilitado = " + bta.isEnabled());
00223         }
00224
00225         Log.d(ETIQUETA_LOG, "inicializarBlueTooth(): obtenemos escaner btle");
00226
00227         this.elScanner = bta.getBluetoothLeScanner();
00228
00229         if (this.elScanner == null) {
00230             Log.e(ETIQUETA_LOG, "inicializarBlueTooth(): NO hemos obtenido escaner btle");
00231         }
00232     }
00233
00234     public SensorObject buscarEsteDispositivoBTLE(final String dispositivoBuscado) {
00235         this.callbackDelEscaneo = new ScanCallback() {
00236             @Override
00237             public void onScanResult(int callbackType, ScanResult resultado) {
00238                 super.onScanResult(callbackType, resultado);
00239                 if (ContextCompat.checkSelfPermission(ArduinoGetterService.this,
00240                     Manifest.permission.BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
00241                     return;
00242                 }
00243
00244                 byte[] bytes = resultado.getScanRecord().getBytes();
00245                 TramaIBeacon tib = new TramaIBeacon(bytes);
00246                 if (Utilidades.bytesToString(tib.getUUID()).equals(dispositivoBuscado)) {
00247                     dispositivoDetectado = true; // El dispositivo ha sido detectado, reiniciar estado
00248                     if (!dispositivoActualmenteConectado) {
00249                         dispositivoActualmenteConectado = true; // Cambiar el estado
00250                         enviarNotificacionConexion(); // Notificar conexión
00251                     }
00252                     String uuid = Utilidades.bytesToString(tib.getUUID());
00253                     String name = resultado.getDevice().getName();
00254                     String typegas = "Unknown"; // Asumiendo que typegas no está disponible
00255                     int measure = Utilidades.bytesToInt(tib.getMinor());
00256                     String date = new Date().toString();
00257                     int battery = Utilidades.bytesToInt(tib.getMajor());
00258                 }
00259             }
00260         };
00261     }
00262
00263     private void enviarNotificacionConexion() {
00264         Intent notificationIntent = new Intent(this, LogInActivity.class);
00265         PendingIntent pendingIntent = PendingIntent.getActivity(
00266             this,
00267             0,
00268             notificationIntent,
00269             PendingIntent.FLAG_UPDATE_CURRENT
00270         );
00271         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00272             .setContentTitle("Nuevo dispositivo conectado")
00273             .setContentText("Nombre: " + name + ", UUID: " + uuid)
00274             .setSmallIcon(android.R.drawable.ic_dialog_info)
00275             .setContentIntent(pendingIntent)
00276             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00277
00278         startForeground(2, builder.build());
00279     }
00280
00281     private void enviarNotificacionDesconexion() {
00282         Intent notificationIntent = new Intent(this, LogInActivity.class);
00283         PendingIntent pendingIntent = PendingIntent.getActivity(
00284             this,
00285             0,
00286             notificationIntent,
00287             PendingIntent.FLAG_UPDATE_CURRENT
00288         );
00289         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00290             .setContentTitle("Dispositivo desconectado")
00291             .setContentText("Nombre: " + name + ", UUID: " + uuid)
00292             .setSmallIcon(android.R.drawable.ic_dialog_info)
00293             .setContentIntent(pendingIntent)
00294             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00295
00296         startForeground(3, builder.build());
00297     }
00298
00299     private void enviarNotificacionCambioEstado() {
00300         Intent notificationIntent = new Intent(this, LogInActivity.class);
00301         PendingIntent pendingIntent = PendingIntent.getActivity(
00302             this,
00303             0,
00304             notificationIntent,
00305             PendingIntent.FLAG_UPDATE_CURRENT
00306         );
00307         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00308             .setContentTitle("Estado cambiado")
00309             .setContentText("Nuevo estado: " + (dispositivoActualmenteConectado ? "Conectado" : "Desconectado"))
00310             .setSmallIcon(android.R.drawable.ic_dialog_info)
00311             .setContentIntent(pendingIntent)
00312             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00313
00314         startForeground(4, builder.build());
00315     }
00316
00317     private void enviarNotificacionMedida() {
00318         Intent notificationIntent = new Intent(this, LogInActivity.class);
00319         PendingIntent pendingIntent = PendingIntent.getActivity(
00320             this,
00321             0,
00322             notificationIntent,
00323             PendingIntent.FLAG_UPDATE_CURRENT
00324         );
00325         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00326             .setContentTitle("Medida recibida")
00327             .setContentText("Medida: " + measure)
00328             .setSmallIcon(android.R.drawable.ic_dialog_info)
00329             .setContentIntent(pendingIntent)
00330             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00331
00332         startForeground(5, builder.build());
00333     }
00334
00335     private void enviarNotificacionFecha() {
00336         Intent notificationIntent = new Intent(this, LogInActivity.class);
00337         PendingIntent pendingIntent = PendingIntent.getActivity(
00338             this,
00339             0,
00340             notificationIntent,
00341             PendingIntent.FLAG_UPDATE_CURRENT
00342         );
00343         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00344             .setContentTitle("Fecha actual")
00345             .setContentText("Fecha: " + date)
00346             .setSmallIcon(android.R.drawable.ic_dialog_info)
00347             .setContentIntent(pendingIntent)
00348             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00349
00350         startForeground(6, builder.build());
00351     }
00352
00353     private void enviarNotificacionBateria() {
00354         Intent notificationIntent = new Intent(this, LogInActivity.class);
00355         PendingIntent pendingIntent = PendingIntent.getActivity(
00356             this,
00357             0,
00358             notificationIntent,
00359             PendingIntent.FLAG_UPDATE_CURRENT
00360         );
00361         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00362             .setContentTitle("Batería")
00363             .setContentText("Batería: " + battery)
00364             .setSmallIcon(android.R.drawable.ic_dialog_info)
00365             .setContentIntent(pendingIntent)
00366             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00367
00368         startForeground(7, builder.build());
00369     }
00370
00371     private void enviarNotificacionTypegas() {
00372         Intent notificationIntent = new Intent(this, LogInActivity.class);
00373         PendingIntent pendingIntent = PendingIntent.getActivity(
00374             this,
00375             0,
00376             notificationIntent,
00377             PendingIntent.FLAG_UPDATE_CURRENT
00378         );
00379         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00380             .setContentTitle("Typegas")
00381             .setContentText("Typegas: " + typegas)
00382             .setSmallIcon(android.R.drawable.ic_dialog_info)
00383             .setContentIntent(pendingIntent)
00384             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00385
00386         startForeground(8, builder.build());
00387     }
00388
00389     private void enviarNotificacionDesconectado() {
00390         Intent notificationIntent = new Intent(this, LogInActivity.class);
00391         PendingIntent pendingIntent = PendingIntent.getActivity(
00392             this,
00393             0,
00394             notificationIntent,
00395             PendingIntent.FLAG_UPDATE_CURRENT
00396         );
00397         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00398             .setContentTitle("Desconectado")
00399             .setContentText("El dispositivo se ha desconectado")
00400             .setSmallIcon(android.R.drawable.ic_dialog_info)
00401             .setContentIntent(pendingIntent)
00402             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00403
00404         startForeground(9, builder.build());
00405     }
00406
00407     private void enviarNotificacionConectado() {
00408         Intent notificationIntent = new Intent(this, LogInActivity.class);
00409         PendingIntent pendingIntent = PendingIntent.getActivity(
00410             this,
00411             0,
00412             notificationIntent,
00413             PendingIntent.FLAG_UPDATE_CURRENT
00414         );
00415         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00416             .setContentTitle("Conectado")
00417             .setContentText("El dispositivo se ha conectado")
00418             .setSmallIcon(android.R.drawable.ic_dialog_info)
00419             .setContentIntent(pendingIntent)
00420             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00421
00422         startForeground(10, builder.build());
00423     }
00424
00425     private void enviarNotificacionMedidaCambiada() {
00426         Intent notificationIntent = new Intent(this, LogInActivity.class);
00427         PendingIntent pendingIntent = PendingIntent.getActivity(
00428             this,
00429             0,
00430             notificationIntent,
00431             PendingIntent.FLAG_UPDATE_CURRENT
00432         );
00433         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00434             .setContentTitle("Medida cambiada")
00435             .setContentText("La medida ha cambiado")
00436             .setSmallIcon(android.R.drawable.ic_dialog_info)
00437             .setContentIntent(pendingIntent)
00438             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00439
00440         startForeground(11, builder.build());
00441     }
00442
00443     private void enviarNotificacionFechaCambiada() {
00444         Intent notificationIntent = new Intent(this, LogInActivity.class);
00445         PendingIntent pendingIntent = PendingIntent.getActivity(
00446             this,
00447             0,
00448             notificationIntent,
00449             PendingIntent.FLAG_UPDATE_CURRENT
00450         );
00451         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00452             .setContentTitle("Fecha cambiada")
00453             .setContentText("La fecha ha cambiado")
00454             .setSmallIcon(android.R.drawable.ic_dialog_info)
00455             .setContentIntent(pendingIntent)
00456             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00457
00458         startForeground(12, builder.build());
00459     }
00460
00461     private void enviarNotificacionBateriaCambiada() {
00462         Intent notificationIntent = new Intent(this, LogInActivity.class);
00463         PendingIntent pendingIntent = PendingIntent.getActivity(
00464             this,
00465             0,
00466             notificationIntent,
00467             PendingIntent.FLAG_UPDATE_CURRENT
00468         );
00469         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00470             .setContentTitle("Batería cambiada")
00471             .setContentText("La batería ha cambiado")
00472             .setSmallIcon(android.R.drawable.ic_dialog_info)
00473             .setContentIntent(pendingIntent)
00474             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00475
00476         startForeground(13, builder.build());
00477     }
00478
00479     private void enviarNotificacionTypegasCambiada() {
00480         Intent notificationIntent = new Intent(this, LogInActivity.class);
00481         PendingIntent pendingIntent = PendingIntent.getActivity(
00482             this,
00483             0,
00484             notificationIntent,
00485             PendingIntent.FLAG_UPDATE_CURRENT
00486         );
00487         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00488             .setContentTitle("Typegas cambiada")
00489             .setContentText("El typegas ha cambiado")
00490             .setSmallIcon(android.R.drawable.ic_dialog_info)
00491             .setContentIntent(pendingIntent)
00492             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00493
00494         startForeground(14, builder.build());
00495     }
00496
00497     private void enviarNotificacionNuevoDispositivo() {
00498         Intent notificationIntent = new Intent(this, LogInActivity.class);
00499         PendingIntent pendingIntent = PendingIntent.getActivity(
00500             this,
00501             0,
00502             notificationIntent,
00503             PendingIntent.FLAG_UPDATE_CURRENT
00504         );
00505         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00506             .setContentTitle("Nuevo dispositivo")
00507             .setContentText("Nuevo dispositivo encontrado")
00508             .setSmallIcon(android.R.drawable.ic_dialog_info)
00509             .setContentIntent(pendingIntent)
00510             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00511
00512         startForeground(15, builder.build());
00513     }
00514
00515     private void enviarNotificacionDesconectadoCambiado() {
00516         Intent notificationIntent = new Intent(this, LogInActivity.class);
00517         PendingIntent pendingIntent = PendingIntent.getActivity(
00518             this,
00519             0,
00520             notificationIntent,
00521             PendingIntent.FLAG_UPDATE_CURRENT
00522         );
00523         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00524             .setContentTitle("Desconectado cambiado")
00525             .setContentText("El desconectado ha cambiado")
00526             .setSmallIcon(android.R.drawable.ic_dialog_info)
00527             .setContentIntent(pendingIntent)
00528             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00529
00530         startForeground(16, builder.build());
00531     }
00532
00533     private void enviarNotificacionConectadoCambiado() {
00534         Intent notificationIntent = new Intent(this, LogInActivity.class);
00535         PendingIntent pendingIntent = PendingIntent.getActivity(
00536             this,
00537             0,
00538             notificationIntent,
00539             PendingIntent.FLAG_UPDATE_CURRENT
00540         );
00541         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00542             .setContentTitle("Conectado cambiado")
00543             .setContentText("El conectado ha cambiado")
00544             .setSmallIcon(android.R.drawable.ic_dialog_info)
00545             .setContentIntent(pendingIntent)
00546             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00547
00548         startForeground(17, builder.build());
00549     }
00550
00551     private void enviarNotificacionMedidaCambiadaCambiada() {
00552         Intent notificationIntent = new Intent(this, LogInActivity.class);
00553         PendingIntent pendingIntent = PendingIntent.getActivity(
00554             this,
00555             0,
00556             notificationIntent,
00557             PendingIntent.FLAG_UPDATE_CURRENT
00558         );
00559         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00560             .setContentTitle("Medida cambiada cambiada")
00561             .setContentText("La medida cambiada ha cambiado")
00562             .setSmallIcon(android.R.drawable.ic_dialog_info)
00563             .setContentIntent(pendingIntent)
00564             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00565
00566         startForeground(18, builder.build());
00567     }
00568
00569     private void enviarNotificacionFechaCambiadaCambiada() {
00570         Intent notificationIntent = new Intent(this, LogInActivity.class);
00571         PendingIntent pendingIntent = PendingIntent.getActivity(
00572             this,
00573             0,
00574             notificationIntent,
00575             PendingIntent.FLAG_UPDATE_CURRENT
00576         );
00577         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00578             .setContentTitle("Fecha cambiada cambiada")
00579             .setContentText("La fecha cambiada ha cambiado")
00580             .setSmallIcon(android.R.drawable.ic_dialog_info)
00581             .setContentIntent(pendingIntent)
00582             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00583
00584         startForeground(19, builder.build());
00585     }
00586
00587     private void enviarNotificacionBateriaCambiadaCambiada() {
00588         Intent notificationIntent = new Intent(this, LogInActivity.class);
00589         PendingIntent pendingIntent = PendingIntent.getActivity(
00590             this,
00591             0,
00592             notificationIntent,
00593             PendingIntent.FLAG_UPDATE_CURRENT
00594         );
00595         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00596             .setContentTitle("Batería cambiada cambiada")
00597             .setContentText("La batería cambiada ha cambiado")
00598             .setSmallIcon(android.R.drawable.ic_dialog_info)
00599             .setContentIntent(pendingIntent)
00600             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00601
00602         startForeground(20, builder.build());
00603     }
00604
00605     private void enviarNotificacionTypegasCambiadaCambiada() {
00606         Intent notificationIntent = new Intent(this, LogInActivity.class);
00607         PendingIntent pendingIntent = PendingIntent.getActivity(
00608             this,
00609             0,
00610             notificationIntent,
00611             PendingIntent.FLAG_UPDATE_CURRENT
00612         );
00613         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00614             .setContentTitle("Typegas cambiada cambiada")
00615             .setContentText("El typegas cambiada ha cambiado")
00616             .setSmallIcon(android.R.drawable.ic_dialog_info)
00617             .setContentIntent(pendingIntent)
00618             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00619
00620         startForeground(21, builder.build());
00621     }
00622
00623     private void enviarNotificacionNuevoDispositivoCambiado() {
00624         Intent notificationIntent = new Intent(this, LogInActivity.class);
00625         PendingIntent pendingIntent = PendingIntent.getActivity(
00626             this,
00627             0,
00628             notificationIntent,
00629             PendingIntent.FLAG_UPDATE_CURRENT
00630         );
00631         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00632             .setContentTitle("Nuevo dispositivo cambiado")
00633             .setContentText("Nuevo dispositivo cambiado")
00634             .setSmallIcon(android.R.drawable.ic_dialog_info)
00635             .setContentIntent(pendingIntent)
00636             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00637
00638         startForeground(22, builder.build());
00639     }
00640
00641     private void enviarNotificacionDesconectadoCambiadoCambiado() {
00642         Intent notificationIntent = new Intent(this, LogInActivity.class);
00643         PendingIntent pendingIntent = PendingIntent.getActivity(
00644             this,
00645             0,
00646             notificationIntent,
00647             PendingIntent.FLAG_UPDATE_CURRENT
00648         );
00649         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00650             .setContentTitle("Desconectado cambiado cambiado")
00651             .setContentText("El desconectado cambiado ha cambiado")
00652             .setSmallIcon(android.R.drawable.ic_dialog_info)
00653             .setContentIntent(pendingIntent)
00654             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00655
00656         startForeground(23, builder.build());
00657     }
00658
00659     private void enviarNotificacionConectadoCambiadoCambiado() {
00660         Intent notificationIntent = new Intent(this, LogInActivity.class);
00661         PendingIntent pendingIntent = PendingIntent.getActivity(
00662             this,
00663             0,
00664             notificationIntent,
00665             PendingIntent.FLAG_UPDATE_CURRENT
00666         );
00667         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00668             .setContentTitle("Conectado cambiado cambiado")
00669             .setContentText("El conectado cambiado ha cambiado")
00670             .setSmallIcon(android.R.drawable.ic_dialog_info)
00671             .setContentIntent(pendingIntent)
00672             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00673
00674         startForeground(24, builder.build());
00675     }
00676
00677     private void enviarNotificacionMedidaCambiadaCambiadoCambiado() {
00678         Intent notificationIntent = new Intent(this, LogInActivity.class);
00679         PendingIntent pendingIntent = PendingIntent.getActivity(
00680             this,
00681             0,
00682             notificationIntent,
00683             PendingIntent.FLAG_UPDATE_CURRENT
00684         );
00685         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00686             .setContentTitle("Medida cambiada cambiado cambiado")
00687             .setContentText("La medida cambiada cambiado ha cambiado")
00688             .setSmallIcon(android.R.drawable.ic_dialog_info)
00689             .setContentIntent(pendingIntent)
00690             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00691
00692         startForeground(25, builder.build());
00693     }
00694
00695     private void enviarNotificacionFechaCambiadaCambiadoCambiado() {
00696         Intent notificationIntent = new Intent(this, LogInActivity.class);
00697         PendingIntent pendingIntent = PendingIntent.getActivity(
00698             this,
00699             0,
00700             notificationIntent,
00701             PendingIntent.FLAG_UPDATE_CURRENT
00702         );
00703         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00704             .setContentTitle("Fecha cambiada cambiado cambiado")
00705             .setContentText("La fecha cambiada cambiado ha cambiado")
00706             .setSmallIcon(android.R.drawable.ic_dialog_info)
00707             .setContentIntent(pendingIntent)
00708             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00709
00710         startForeground(26, builder.build());
00711     }
00712
00713     private void enviarNotificacionBateriaCambiadaCambiadoCambiado() {
00714         Intent notificationIntent = new Intent(this, LogInActivity.class);
00715         PendingIntent pendingIntent = PendingIntent.getActivity(
00716             this,
00717             0,
00718             notificationIntent,
00719             PendingIntent.FLAG_UPDATE_CURRENT
00720         );
00721         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00722             .setContentTitle("Batería cambiada cambiado cambiado")
00723             .setContentText("La batería cambiada cambiado ha cambiado")
00724             .setSmallIcon(android.R.drawable.ic_dialog_info)
00725             .setContentIntent(pendingIntent)
00726             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00727
00728         startForeground(27, builder.build());
00729     }
00730
00731     private void enviarNotificacionTypegasCambiadaCambiadoCambiado() {
00732         Intent notificationIntent = new Intent(this, LogInActivity.class);
00733         PendingIntent pendingIntent = PendingIntent.getActivity(
00734             this,
00735             0,
00736             notificationIntent,
00737             PendingIntent.FLAG_UPDATE_CURRENT
00738         );
00739         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00740             .setContentTitle("Typegas cambiada cambiado cambiado")
00741             .setContentText("El typegas cambiada cambiado ha cambiado")
00742             .setSmallIcon(android.R.drawable.ic_dialog_info)
00743             .setContentIntent(pendingIntent)
00744             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00745
00746         startForeground(28, builder.build());
00747     }
00748
00749     private void enviarNotificacionNuevoDispositivoCambiadoCambiado() {
00750         Intent notificationIntent = new Intent(this, LogInActivity.class);
00751         PendingIntent pendingIntent = PendingIntent.getActivity(
00752             this,
00753             0,
00754             notificationIntent,
00755             PendingIntent.FLAG_UPDATE_CURRENT
00756         );
00757         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00758             .setContentTitle("Nuevo dispositivo cambiado cambiado")
00759             .setContentText("Nuevo dispositivo cambiado cambiado")
00760             .setSmallIcon(android.R.drawable.ic_dialog_info)
00761             .setContentIntent(pendingIntent)
00762             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00763
00764         startForeground(29, builder.build());
00765     }
00766
00767     private void enviarNotificacionDesconectadoCambiadoCambiadoCambiado() {
00768         Intent notificationIntent = new Intent(this, LogInActivity.class);
00769         PendingIntent pendingIntent = PendingIntent.getActivity(
00770             this,
00771             0,
00772             notificationIntent,
00773             PendingIntent.FLAG_UPDATE_CURRENT
00774         );
00775         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00776             .setContentTitle("Desconectado cambiado cambiado cambiado")
00777             .setContentText("El desconectado cambiado cambiado ha cambiado")
00778             .setSmallIcon(android.R.drawable.ic_dialog_info)
00779             .setContentIntent(pendingIntent)
00780             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00781
00782         startForeground(30, builder.build());
00783     }
00784
00785     private void enviarNotificacionConectadoCambiadoCambiadoCambiado() {
00786         Intent notificationIntent = new Intent(this, LogInActivity.class);
00787         PendingIntent pendingIntent = PendingIntent.getActivity(
00788             this,
00789             0,
00790             notificationIntent,
00791             PendingIntent.FLAG_UPDATE_CURRENT
00792         );
00793         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00794             .setContentTitle("Conectado cambiado cambiado cambiado")
00795             .setContentText("El conectado cambiado cambiado ha cambiado")
00796             .setSmallIcon(android.R.drawable.ic_dialog_info)
00797             .setContentIntent(pendingIntent)
00798             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00799
00800         startForeground(31, builder.build());
00801     }
00802
00803     private void enviarNotificacionMedidaCambiadaCambiadoCambiadoCambiado() {
00804         Intent notificationIntent = new Intent(this, LogInActivity.class);
00805         PendingIntent pendingIntent = PendingIntent.getActivity(
00806             this,
00807             0,
00808             notificationIntent,
00809             PendingIntent.FLAG_UPDATE_CURRENT
00810         );
00811         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00812             .setContentTitle("Medida cambiada cambiado cambiado cambiado")
00813             .setContentText("La medida cambiada cambiado cambiado ha cambiado")
00814             .setSmallIcon(android.R.drawable.ic_dialog_info)
00815             .setContentIntent(pendingIntent)
00816             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00817
00818         startForeground(32, builder.build());
00819     }
00820
00821     private void enviarNotificacionFechaCambiadaCambiadoCambiadoCambiado() {
00822         Intent notificationIntent = new Intent(this, LogInActivity.class);
00823         PendingIntent pendingIntent = PendingIntent.getActivity(
00824             this,
00825             0,
00826             notificationIntent,
00827             PendingIntent.FLAG_UPDATE_CURRENT
00828         );
00829         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00830             .setContentTitle("Fecha cambiada cambiado cambiado cambiado")
00831             .setContentText("La fecha cambiada cambiado cambiado ha cambiado")
00832             .setSmallIcon(android.R.drawable.ic_dialog_info)
00833             .setContentIntent(pendingIntent)
00834             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00835
00836         startForeground(33, builder.build());
00837     }
00838
00839     private void enviarNotificacionBateriaCambiadaCambiadoCambiadoCambiado() {
00840         Intent notificationIntent = new Intent(this, LogInActivity.class);
00841         PendingIntent pendingIntent = PendingIntent.getActivity(
00842             this,
00843             0,
00844             notificationIntent,
00845             PendingIntent.FLAG_UPDATE_CURRENT
00846         );
00847         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00848             .setContentTitle("Batería cambiada cambiado cambiado cambiado")
00849             .setContentText("La batería cambiada cambiado cambiado ha cambiado")
00850             .setSmallIcon(android.R.drawable.ic_dialog_info)
00851             .setContentIntent(pendingIntent)
00852             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00853
00854         startForeground(34, builder.build());
00855     }
00856
00857     private void enviarNotificacionTypegasCambiadaCambiadoCambiadoCambiado() {
00858         Intent notificationIntent = new Intent(this, LogInActivity.class);
00859         PendingIntent pendingIntent = PendingIntent.getActivity(
00860             this,
00861             0,
00862             notificationIntent,
00863             PendingIntent.FLAG_UPDATE_CURRENT
00864         );
00865         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00866             .setContentTitle("Typegas cambiada cambiado cambiado cambiado")
00867             .setContentText("El typegas cambiada cambiado cambiado ha cambiado")
00868             .setSmallIcon(android.R.drawable.ic_dialog_info)
00869             .setContentIntent(pendingIntent)
00870             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00871
00872         startForeground(35, builder.build());
00873     }
00874
00875     private void enviarNotificacionNuevoDispositivoCambiadoCambiadoCambiado() {
00876         Intent notificationIntent = new Intent(this, LogInActivity.class);
00877         PendingIntent pendingIntent = PendingIntent.getActivity(
00878             this,
00879             0,
00880             notificationIntent,
00881             PendingIntent.FLAG_UPDATE_CURRENT
00882         );
00883         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00884             .setContentTitle("Nuevo dispositivo cambiado cambiado cambiado")
00885             .setContentText("Nuevo dispositivo cambiado cambiado cambiado")
00886             .setSmallIcon(android.R.drawable.ic_dialog_info)
00887             .setContentIntent(pendingIntent)
00888             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00889
00890         startForeground(36, builder.build());
00891     }
00892
00893     private void enviarNotificacionDesconectadoCambiadoCambiadoCambiadoCambiado() {
00894         Intent notificationIntent = new Intent(this, LogInActivity.class);
00895         PendingIntent pendingIntent = PendingIntent.getActivity(
00896             this,
00897             0,
00898             notificationIntent,
00899             PendingIntent.FLAG_UPDATE_CURRENT
00900         );
00901         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00902             .setContentTitle("Desconectado cambiado cambiado cambiado cambiado")
00903             .setContentText("El desconectado cambiado cambiado cambiado ha cambiado")
00904             .setSmallIcon(android.R.drawable.ic_dialog_info)
00905             .setContentIntent(pendingIntent)
00906             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00907
00908         startForeground(37, builder.build());
00909     }
00910
00911     private void enviarNotificacionConectadoCambiadoCambiadoCambiadoCambiado() {
00912         Intent notificationIntent = new Intent(this, LogInActivity.class);
00913         PendingIntent pendingIntent = PendingIntent.getActivity(
00914             this,
00915             0,
00916             notificationIntent,
00917             PendingIntent.FLAG_UPDATE_CURRENT
00918         );
00919         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00920             .setContentTitle("Conectado cambiado cambiado cambiado cambiado")
00921             .setContentText("El conectado cambiado cambiado cambiado ha cambiado")
00922             .setSmallIcon(android.R.drawable.ic_dialog_info)
00923             .setContentIntent(pendingIntent)
00924             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00925
00926         startForeground(38, builder.build());
00927     }
00928
00929     private void enviarNotificacionMedidaCambiadaCambiadoCambiadoCambiadoCambiado() {
00930         Intent notificationIntent = new Intent(this, LogInActivity.class);
00931         PendingIntent pendingIntent = PendingIntent.getActivity(
00932             this,
00933             0,
00934             notificationIntent,
00935             PendingIntent.FLAG_UPDATE_CURRENT
00936         );
00937         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00938             .setContentTitle("Medida cambiada cambiado cambiado cambiado cambiado")
00939             .setContentText("La medida cambiada cambiado cambiado cambiado ha cambiado")
00940             .setSmallIcon(android.R.drawable.ic_dialog_info)
00941             .setContentIntent(pendingIntent)
00942             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00943
00944         startForeground(39, builder.build());
00945     }
00946
00947     private void enviarNotificacionFechaCambiadaCambiadoCambiadoCambiadoCambiado() {
00948         Intent notificationIntent = new Intent(this, LogInActivity.class);
00949         PendingIntent pendingIntent = PendingIntent.getActivity(
00950             this,
00951             0,
00952             notificationIntent,
00953             PendingIntent.FLAG_UPDATE_CURRENT
00954         );
00955         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00956             .setContentTitle("Fecha cambiada cambiado cambiado cambiado cambiado")
00957             .setContentText("La fecha cambiada cambiado cambiado cambiado ha cambiado")
00958             .setSmallIcon(android.R.drawable.ic_dialog_info)
00959             .setContentIntent(pendingIntent)
00960             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00961
00962         startForeground(40, builder.build());
00963     }
00964
00965     private void enviarNotificacionBateriaCambiadaCambiadoCambiadoCambiadoCambiado() {
00966         Intent notificationIntent = new Intent(this, LogInActivity.class);
00967         PendingIntent pendingIntent = PendingIntent.getActivity(
00968             this,
00969             0,
00970             notificationIntent,
00971             PendingIntent.FLAG_UPDATE_CURRENT
00972         );
00973         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00974             .setContentTitle("Batería cambiada cambiado cambiado cambiado cambiado")
00975             .setContentText("La batería cambiada cambiado cambiado cambiado ha cambiado")
00976             .setSmallIcon(android.R.drawable.ic_dialog_info)
00977             .setContentIntent(pendingIntent)
00978             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00979
00980         startForeground(41, builder.build());
00981     }
00982
00983     private void enviarNotificacionTypegasCambiadaCambiadoCambiadoCambiadoCambiado() {
00984         Intent notificationIntent = new Intent(this, LogInActivity.class);
00985         PendingIntent pendingIntent = PendingIntent.getActivity(
00986             this,
00987             0,
00988             notificationIntent,
00989             PendingIntent.FLAG_UPDATE_CURRENT
00990         );
00991         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00992             .setContentTitle("Typegas cambiada cambiado cambiado cambiado cambiado")
00993             .setContentText("El typegas cambiada cambiado cambiado cambiado ha cambiado")
00994             .setSmallIcon(android.R.drawable.ic_dialog_info)
00995             .setContentIntent(pendingIntent)
00996             .setPriority(NotificationCompat.PRIORITY_DEFAULT);
00997
00998         startForeground(42, builder.build());
00999     }
01000
01001     private void enviarNotificacionNuevoDispositivoCambiadoCambiadoCambiadoCambiado() {
01002         Intent notificationIntent = new Intent(this, LogInActivity.class);
01003         PendingIntent pendingIntent = PendingIntent.getActivity(
01004             this,
01005             0,
01006             notificationIntent,
01007             PendingIntent.FLAG_UPDATE_CURRENT
01008         );
01009         NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
01010             .setContentTitle("Nuevo dispositivo cambiado cambiado cambiado cambiado")
01011             .setContentText("Nuevo dispositivo cambiado cambiado cambiado cambiado")
01012             .setSmallIcon(android.R.drawable.ic_dialog_info)
01013             .setContentIntent(pendingIntent)
01014            
```

```

00280
00281     SharedPreferences sharedpreferences =
00282         ArduinoGetterService.this.getSharedPreferences("MyAppPrefs", Context.MODE_PRIVATE);
00283         //recoger la id del sensor con el mismo uuid
00284         if(sharedPreferences.getString("uuid", "").equals(uuid)) {
00285     // After
00286             double latitude = 0.0, longitude = 0.0;
00287             double[] location = getLocation(); // Llamada al método para obtener la
00288             ubicación
00289             if (location != null) {
00290                 latitude = location[0];
00291                 longitude = location[1];
00292             } // Llamada al método para obtener la ubicación
00293             int idSensor = sharedPreferences.getInt("id_sensor", -1);
00294             SensorObject sensor = new SensorObject(idSensor, "Conectado", "1234", uuid,
00295             name, true, battery);
00296             Medicion medicion = new Medicion(0, sharedPreferences.getInt("id_sensor", -1),
00297             typegas, latitude, longitude, measure);
00298             limitcheck(sensor,medicion);
00299         }
00300
00301     @Override
00302     public void onBatchScanResults(List<ScanResult> results) {
00303         super.onBatchScanResults(results);
00304     }
00305
00306     @Override
00307     public void onScanFailed(int errorCode) {
00308         super.onScanFailed(errorCode);
00309     };
00310
00311     ScanFilter sf = new ScanFilter.Builder().setDeviceName(dispositivoBuscado).build();
00312
00313     if (ContextCompat.checkSelfPermission(this, Manifest.permission.BLUETOOTH_SCAN) !=
00314     PackageManager.PERMISSION_GRANTED) {
00315         // Manejar solicitud de permisos
00316         return null;
00317     }
00318     this.elScanner.startScan(this.callbackDelEscaneo);
00319     return null;
00320 }
00321
00322 private void enviarNotificacionDesconexion() {
00323     NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00324         .setContentTitle("Desconexión de dispositivo")
00325         .setContentText("El dispositivo no se ha detectado en los últimos " +
00326 (TIEMPO_DESCONEXION / 1000) + " segundos.")
00327         .setSmallIcon(android.R.drawable.ic_dialog_alert)
00328         .setPriority(NotificationCompat.PRIORITY_HIGH);
00329
00330     NotificationManager manager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
00331     if (manager != null) {
00332         manager.notify(2, builder.build());
00333     }
00334     Log.d(ETIQUETA_LOG, "Notificación de desconexión enviada.");
00335     dispositivoActualmenteConectado = false; // Cambiar el estado
00336
00337     // Obtener la id del sensor de shared preferences
00338     SharedPreferences sharedpreferences = this.getSharedPreferences("MyAppPrefs",
00339     Context.MODE_PRIVATE);
00340     int idSensor = sharedpreferences.getInt("id_sensor", -1);
00341     // Actualizar el estado en el servidor
00342     enviarPeticionesUser.actualizarSensor(idSensor, "Desconexión de dispositivo", false, 0);
00343 }
00344
00345 private void enviarNotificacionConexion() {
00346     NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00347         .setContentTitle("Conexión de dispositivo")
00348         .setContentText("El dispositivo ha sido detectado.")
00349         .setSmallIcon(android.R.drawable.ic_dialog_info)
00350         .setPriority(NotificationCompat.PRIORITY_HIGH);
00351
00352     NotificationManager manager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
00353     if (manager != null) {
00354         manager.notify(3, builder.build());
00355     }
00356     Log.d(ETIQUETA_LOG, "Notificación de conexión enviada.");
00357 }
00358
00359     private double[] getLocation() {
00360         // Crear un LocationManager para acceder a los servicios de ubicación
00361         LocationManager locationManager = (LocationManager)
00362         getSystemService(Context.LOCATION_SERVICE);
00363
00364
00365
00366
00367
00368
00369
00370
00371
00372
00373
00374
00375
00376
00377
00378
00379
00380
00381
00382
00383
00384
00385
00386
00387

```

```

00388     // Verificar si el LocationManager está disponible
00389     if (locationManager != null) {
00390         try {
00391             // Comprobamos si tenemos permiso para acceder a la ubicación
00392             if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
00393                 == PackageManager.PERMISSION_GRANTED ||
00394                 ActivityCompat.checkSelfPermission(this,
00395                     Manifest.permission.ACCESS_COARSE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
00396                 // Obtener la ubicación más reciente del proveedor de ubicación
00397                 Location location =
00398                     locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
00399                 if (location != null) {
00400                     // Si se obtiene la ubicación, devolverla como una cadena
00401                     double latitude = location.getLatitude();
00402                     double longitude = location.getLongitude();
00403                     return new double[]{latitude, longitude};
00404                 }
00405             } catch (SecurityException e) {
00406                 e.printStackTrace();
00407             }
00408         }
00409         // Si no se puede obtener la ubicación, devolver un mensaje de error
00410         return new double[]{0.0, 0.0};
00411     }
00412 }
00413
00414     // Comprobar límite de exceso de gas y enviar notificación con el nombre del sensor, la hora, el
00415     // tipo de gas y la medida
00416 // Color de fondo personalizado para cada límite
00417
00418     private void limitcheck(SensorObject sensor, Medicion medicion) {
00419         double value = medicion.getValor();
00420         long currentTime = System.currentTimeMillis();
00421
00422         //si la id del sensor es distinta a la id de shared preferences se cambiará el valor de la id
00423         de shared preferences
00424         SharedPreferences sharedPreferences = this.getSharedPreferences("MyAppPrefs",
00425             Context.MODE_PRIVATE);
00426         int idSensor = sharedPreferences.getInt("id_sensor", -1);
00427
00428         Log.d(ETIQUETA_LOG, "SensorObject: " + idSensor);
00429         sensor.setId(idSensor);
00430
00431         // Check if enough time has passed since the last notification
00432         if (currentTime - lastNotificationTime < NOTIFICATION_INTERVAL) {
00433             Log.d(ETIQUETA_LOG, "Skipping notification to avoid spamming.");
00434             return;
00435         }
00436
00437         Log.d(ETIQUETA_LOG, "Changing background color, value: " + value);
00438         RemoteViews remoteViews = new RemoteViews(getApplicationContext(),
00439             R.layout.custom_notification_layout);
00440         remoteViews.setTextViewText(R.id.notification_title, "Nivel de gas");
00441
00442         remoteViews.setTextViewText(R.id.notification_text, "Sensor: " + sensor.getNombre() + "\nHora:
00443             " + medicion.getFecha() + "\nTipo de gas: " + medicion.getTipoGas() + "\nMedida: " +
00444             medicion.getValor() + "\nUbicación: " + medicion.getLatitude() + ", " + medicion.getLongitude());
00445         remoteViews.setImageResource(R.id.notification_icon, android.R.drawable.ic_dialog_alert);
00446
00447         int color;
00448         if (value > 200) {
00449             color = ContextCompat.getColor(this, R.color.RojoPeligroso);
00450             remoteViews.setTextViewText(R.id.notification_title, "Error en la medición");
00451             enviarPeticionesUser.actualizarSensor(sensor.getId(), "error en la
00452             medición", true, sensor.getBateria());
00453         } else if (value > 100) {
00454             color = ContextCompat.getColor(this, R.color.RojoPeligroso);
00455             remoteViews.setTextViewText(R.id.notification_title, "Exceso de gas detectado");
00456             enviarPeticionesUser.actualizarSensor(sensor.getId(), "Exceso de gas
00457             detectado", true, sensor.getBateria());
00458
00459         } else if (value > 75) {
00460             color = ContextCompat.getColor(this, R.color.NaranjaMalo);
00461             remoteViews.setTextViewText(R.id.notification_title, "Alerta de gas");
00462             enviarPeticionesUser.actualizarSensor(sensor.getId(), "Alerta de
00463             gas", true, sensor.getBateria());
00464         } else if (value > 50) {
00465             color = ContextCompat.getColor(this, R.color.AmarilloMedio);
00466             remoteViews.setTextViewText(R.id.notification_title, "Advertencia de gas");
00467             enviarPeticionesUser.actualizarSensor(sensor.getId(), "Advertencia de
00468             gas", true, sensor.getBateria());
00469
00470
00471
00472
00473
00474
00475
00476
00477
00478
00479
00480
00481
00482
00483
00484
00485
00486
00487
00488
00489
00490
00491
00492
00493
00494
00495
00496
00497
00498
00499
00500
00501
00502
00503
00504
00505
00506
00507
00508
00509
00510
00511
00512
00513
00514
00515
00516
00517
00518
00519
00520
00521
00522
00523
00524
00525
00526
00527
00528
00529
00530
00531
00532
00533
00534
00535
00536
00537
00538
00539
00540
00541
00542
00543
00544
00545
00546
00547
00548
00549
00550
00551
00552
00553
00554
00555
00556
00557
00558
00559
00560
00561
00562
00563
00564
00565
00566
00567
00568
00569
00570
00571
00572
00573
00574
00575
00576
00577
00578
00579
00580
00581
00582
00583
00584
00585
00586
00587
00588
00589
00590
00591
00592
00593
00594
00595
00596
00597
00598
00599
00600
00601
00602
00603
00604
00605
00606
00607
00608
00609
00610
00611
00612
00613
00614
00615
00616
00617
00618
00619
00620
00621
00622
00623
00624
00625
00626
00627
00628
00629
00630
00631
00632
00633
00634
00635
00636
00637
00638
00639
00640
00641
00642
00643
00644
00645
00646
00647
00648
00649
00650
00651
00652
00653
00654
00655
00656
00657
00658
00659
00660
00661
00662
00663
00664
00665
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682
00683
00684
00685
00686
00687
00688
00689
00690
00691
00692
00693
00694
00695
00696
00697
00698
00699
00700
00701
00702
00703
00704
00705
00706
00707
00708
00709
00710
00711
00712
00713
00714
00715
00716
00717
00718
00719
00720
00721
00722
00723
00724
00725
00726
00727
00728
00729
00730
00731
00732
00733
00734
00735
00736
00737
00738
00739
00740
00741
00742
00743
00744
00745
00746
00747
00748
00749
00750
00751
00752
00753
00754
00755
00756
00757
00758
00759
00760
00761
00762
00763
00764
00765
00766
00767
00768
00769
00770
00771
00772
00773
00774
00775
00776
00777
00778
00779
00780
00781
00782
00783
00784
00785
00786
00787
00788
00789
00790
00791
00792
00793
00794
00795
00796
00797
00798
00799
00800
00801
00802
00803
00804
00805
00806
00807
00808
00809
00810
00811
00812
00813
00814
00815
00816
00817
00818
00819
00820
00821
00822
00823
00824
00825
00826
00827
00828
00829
00830
00831
00832
00833
00834
00835
00836
00837
00838
00839
00840
00841
00842
00843
00844
00845
00846
00847
00848
00849
00850
00851
00852
00853
00854
00855
00856
00857
00858
00859
00860
00861
00862
00863
00864
00865
00866
00867
00868
00869
00870
00871
00872
00873
00874
00875
00876
00877
00878
00879
00880
00881
00882
00883
00884
00885
00886
00887
00888
00889
00890
00891
00892
00893
00894
00895
00896
00897
00898
00899
00900
00901
00902
00903
00904
00905
00906
00907
00908
00909
00910
00911
00912
00913
00914
00915
00916
00917
00918
00919
00920
00921
00922
00923
00924
00925
00926
00927
00928
00929
00930
00931
00932
00933
00934
00935
00936
00937
00938
00939
00940
00941
00942
00943
00944
00945
00946
00947
00948
00949
00950
00951
00952
00953
00954
00955
00956
00957
00958
00959
00960
00961
00962
00963
00964
00965
00966
00967
00968
00969
00970
00971
00972
00973
00974
00975
00976
00977
00978
00979
00980
00981
00982
00983
00984
00985
00986
00987
00988
00989
00990
00991
00992
00993
00994
00995
00996
00997
00998
00999
01000
01001
01002
01003
01004
01005
01006
01007
01008
01009
01010
01011
01012
01013
01014
01015
01016
01017
01018
01019
01020
01021
01022
01023
01024
01025
01026
01027
01028
01029
01030
01031
01032
01033
01034
01035
01036
01037
01038
01039
01040
01041
01042
01043
01044
01045
01046
01047
01048
01049
01050
01051
01052
01053
01054
01055
01056
01057
01058
01059
01060
01061
01062
01063
01064
01065
01066
01067
01068
01069
01070
01071
01072
01073
01074
01075
01076
01077
01078
01079
01080
01081
01082
01083
01084
01085
01086
01087
01088
01089
01090
01091
01092
01093
01094
01095
01096
01097
01098
01099
01100
01101
01102
01103
01104
01105
01106
01107
01108
01109
01110
01111
01112
01113
01114
01115
01116
01117
01118
01119
01120
01121
01122
01123
01124
01125
01126
01127
01128
01129
01130
01131
01132
01133
01134
01135
01136
01137
01138
01139
01140
01141
01142
01143
01144
01145
01146
01147
01148
01149
01150
01151
01152
01153
01154
01155
01156
01157
01158
01159
01160
01161
01162
01163
01164
01165
01166
01167
01168
01169
01170
01171
01172
01173
01174
01175
01176
01177
01178
01179
01180
01181
01182
01183
01184
01185
01186
01187
01188
01189
01190
01191
01192
01193
01194
01195
01196
01197
01198
01199
01200
01201
01202
01203
01204
01205
01206
01207
01208
01209
01210
01211
01212
01213
01214
01215
01216
01217
01218
01219
01220
01221
01222
01223
01224
01225
01226
01227
01228
01229
01230
01231
01232
01233
01234
01235
01236
01237
01238
01239
01240
01241
01242
01243
01244
01245
01246
01247
01248
01249
01250
01251
01252
01253
01254
01255
01256
01257
01258
01259
01260
01261
01262
01263
01264
01265
01266
01267
01268
01269
01270
01271
01272
01273
01274
01275
01276
01277
01278
01279
01280
01281
01282
01283
01284
01285
01286
01287
01288
01289
01290
01291
01292
01293
01294
01295
01296
01297
01298
01299
01300
01301
01302
01303
01304
01305
01306
01307
01308
01309
01310
01311
01312
01313
01314
01315
01316
01317
01318
01319
01320
01321
01322
01323
01324
01325
01326
01327
01328
01329
01330
01331
01332
01333
01334
01335
01336
01337
01338
01339
01340
01341
01342
01343
01344
01345
01346
01347
01348
01349
01350
01351
01352
01353
01354
01355
01356
01357
01358
01359
01360
01361
01362
01363
01364
01365
01366
01367
01368
01369
01370
01371
01372
01373
01374
01375
01376
01377
01378
01379
01380
01381
01382
01383
01384
01385
01386
01387
01388
01389
01390
01391
01392
01393
01394
01395
01396
01397
01398
01399
01400
01401
01402
01403
01404
01405
01406
01407
01408
01409
01410
01411
01412
01413
01414
01415
01416
01417
01418
01419
01420
01421
01422
01423
01424
01425
01426
01427
01428
01429
01430
01431
01432
01433
01434
01435
01436
01437
01438
01439
01440
01441
01442
01443
01444
01445
01446
01447
01448
01449
01450
01451
01452
01453
01454
01455
01456
01457
01458
01459
01460
01461
01462
01463
01464
01465
01466
01467
01468
01469
01470
01471
01472
01473
01474
01475
01476
01477
01478
01479
01480
01481
01482
01483
01484
01485
01486
01487
01488
01489
01490
01491
01492
01493
01494
01495
01496
01497
01498
01499
01500
01501
01502
01503
01504
01505
01506
01507
01508
01509
01510
01511
01512
01513
01514
01515
01516
01517
01518
01519
01520
01521
01522
01523
01524
01525
01526
01527
01528
01529
01530
01531
01532
01533
01534
01535
01536
01537
01538
01539
01540
01541
01542
01543
01544
01545
01546
01547
01548
01549
01550
01551
01552
01553
01554
01555
01556
01557
01558
01559
01560
01561
01562
01563
01564
01565
01566
01567
01568
01569
01570
01571
01572
01573
01574
01575
01576
01577
01578
01579
01580
01581
01582
01583
01584
01585
01586
01587
01588
01589
01590
01591
01592
01593
01594
01595
01596
01597
01598
01599
01600
01601
01602
01603
01604
01605
01606
01607
01608
01609
01610
01611
01612
01613
01614
01615
01616
01617
01618
01619
01620
01621
01622
01623
01624
01625
01626
01627
01628
01629
01630
01631
01632
01633
01634
01635
01636
01637
01638
01639
01640
01641
01642
01643
01644
01645
01646
01647
01648
01649
01650
01651
01652
01653
01654
01655
01656
01657
01658
01659
01660
01661
01662
01663
01664
01665
01666
01667
01668
01669
01670
01671
01672
01673
01674
01675
01676
01677
01678
01679
01680
01681
01682
01683
01684
01685
01686
01687
01688
01689
01690
01691
01692
01693
01694
01695
01696
01697
01698
01699
01700
01701
01702
01703
01704
01705
01706
01707
01708
01709
01710
01711
01712
01713
01714
01715
01716
01717
01718
01719
01720
01721
01722
01723
01724
01725
01726
01727
01728
01729
01730
01731
01732
01733
01734
01735
01736
01737
01738
01739
01740
01741
01742
01743
01744
01745
01746
01747
01748
01749
01750
01751
01752
01753
01754
01755
01756
01757
01758
01759
01760
01761
01762
01763
01764
01765
01766
01767
01768
01769
01770
01771
01772
01773
01774
01775
01776
01777
01778
01779
01780
01781
01782
01783
01784
01785
01786
01787
01788
01789
01790
01791
01792
01793
01794
01795
01796
01797
01798
01799
01800
01801
01802
01803
01804
01805
01806
01807
01808
01809
01810
01811
01812
01813
01814
01815
01816
01817
01818
01819
01820
01821
01822
01823
01824
01825
01826
01827
01828
01829
01830
01831
01832
01833
01834
01835
01836
01837
01838
01839
01840
01841
01842
01843
01844
01845
01846
01847
01848
01849
01850
01851
01852
01853
01854
01855
01856
01857
01858
01859
01860
01861
01862
01863
01864
01865
01866
01867
01868
01869
01870
01871
01872
01873
01874
01875
01876
01877
01878
01879
01880
01881
01882
01883
01884
01885
01886
01887
01888
01889
01890
01891
01892
01893
01894
01895
01896
01897
01898
01899
01900
01901
01902
01903
01904
01905
01906
01907
01908
01909
01910
01911
01912
01913
01914
01915
01916
01917
01918
01919
01920
01921
01922
01923
01924
01925
01926
01927
01928
01929
01930
01931
01932
01933
01934
01935
01936
01937
01938
01939
01940
01941
01942
01943
01944
01945
01946
01947
01948
01949
01950
01951
01952
01953
01954
01955
01956
01957
01958
01959
01960
01961
01962
01963
01964
01965
01966
01967
01968
01969
01970
01971
01972
01973
01974
01975
01976
01977
01978
01979
01980
01981
01982
01983
01984
01985
01986
01987
01988
01989
01990
01991
01992
01993
01994
01995
01996
01997
01998
01999
02000
02001
02002
02003
02004
02005
02006
02007
02008
02009
02010
02011
02012
02013
02014
02015
02016
02017
02018
02019
02020
02021
02022
02023
02024
02025
02026
02027
02028
02029
02030
02031
02032
02033
02034
02035
02036
02037
02038
02039
02040
02041
02042
02043
02044
02045
02046
02047
02048
02049
02050
02051
02052
02053
02054
02055
02056
02057
02058
02059
02060
02061
02062
02063
02064
02065
02066
02067
02068
02069
02070
02071
02072
02073
02074
02075
02076
02077
02078
02079
02080
02081
02082
02083
02084
02085
02086
02087
02088
02089
02090
02091
02092
02093
02094
02095
02096
02097
02098
02099
02100
02101
02102
02103
021
```

```
00474         } else if (value > 25) {
00475             color = ContextCompat.getColor(this, R.color.VerdeBueno);
00476             remoteViews.setTextViewText(R.id.notification_title, "Nivel de gas aceptable");
00477             enviarPeticionesUser.actualizarSensor(sensor.getId(), "Nivel de gas
00478                 aceptable",true,sensor.getBateria());
00479         } else {
00480             color = ContextCompat.getColor(this, R.color.RosaExcelente);
00481             remoteViews.setTextViewText(R.id.notification_title, "Nivel de gas excelente");
00482             enviarPeticionesUser.actualizarSensor(sensor.getId(), "Nivel de gas
00483                 excelente",true,sensor.getBateria());
00484         }
00485     remoteViews.setInt(R.id.custom_notification_layout, "setBackgroundColor", color);
00486
00487     // Crear y mostrar la notificación usando el canal personalizado
00488     NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL_ID)
00489         .setSmallIcon(android.R.drawable.ic_dialog_alert)
00490         .setCustomContentView(remoteViews)
00491         .setPriority(NotificationCompat.PRIORITY_HIGH)
00492         .setColor(color)
00493         .setStyle(new NotificationCompat.DecoratedCustomViewStyle());
00494
00495     NotificationManager manager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
00496     if (manager != null) {
00497         manager.notify(4, builder.build());
00498     }
00499     Log.d(ETIQUETA_LOG, "Notificación de nivel de gas enviada.");
00500
00501     // Update the last notification time
00502     lastNotificationTime = currentTime;
00503 }
```



## Índice alfabético

actualizarSensor  
com.example.smariba\_upv.airflow.API.ApiService, 10  
com.example.smariba\_upv.airflow.API.EnviarPeticionesUser, 27  
ApiService.java, 133–136  
ArduinoGetterService.java, 200  
authenticate  
com.example.smariba\_upv.airflow.LOGIC.BiometricUtil, 22  
biometricLogin  
com.example.smariba\_upv.airflow.PRESENTACION.LoginActivity, 40  
BiometricUtil.java, 146, 147  
botonBuscarDispositivosBTLEPulsado  
com.example.smariba\_upv.airflow.MainActivity, 45  
com.example.smariba\_upv.btle\_sento.MainActivity, 49  
botonBuscarNuestroDispositivoBTLEPulsado  
com.example.smariba\_upv.airflow.MainActivity, 45  
com.example.smariba\_upv.btle\_sento.MainActivity, 49  
botonDetenerBusquedaDispositivosBTLEPulsado  
com.example.smariba\_upv.airflow.MainActivity, 45  
com.example.smariba\_upv.btle\_sento.MainActivity, 49  
btnEditarPerfil  
com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity, 67  
buscarEsteDispositivoBTLE  
com.example.smariba\_upv.airflow.Services.ArduinoGetterService, 15  
bytesToHexString  
com.example.smariba\_upv.airflow.LOGIC.Utilidades, 121  
com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 128  
bytesToInt  
com.example.smariba\_upv.airflow.LOGIC.Utilidades, 121  
com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 128  
bytesToIntOK  
com.example.smariba\_upv.airflow.LOGIC.Utilidades, 122  
com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 129  
bytesToLong  
com.example.smariba\_upv.airflow.LOGIC.Utilidades, 122  
com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 129  
bytesToString  
com.example.smariba\_upv.airflow.LOGIC.Utilidades, 123  
com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 130  
com.example.smariba\_upv.airflow, 6  
com.example.smariba\_upv.airflow.API, 7  
com.example.smariba\_upv.airflow.API.ApiService, 9  
actualizarSensor, 10  
editUsuario, 10  
getMisSensores, 10  
insertarMedicion, 11  
Activity, 1  
registrarSensor, 11  
com.example.smariba\_upv.airflow.API.EnviarPeticionesUser, 26  
actualizarSensor, 27  
editUsuario, 28  
EnviarPeticionesUser, 27  
login, 29  
obtenerMisSensores, 30  
registrarSensor, 30  
com.example.smariba\_upv.airflow.API.MODELS, 7  
com.example.smariba\_upv.airflow.API.MODELS.SensorRequest, 97  
getBateria, 99  
getEstado, 99  
getIdSensor, 99  
getNombre, 99  
getNumReferencia, 100  
Service, 100  
isConexion, 100  
SensorRequest, 98  
setBateria, 100  
setConexion, 101  
setEstado, 101  
setIdSensor, 101  
setIdUsuario, 101  
setNombre, 102  
setNumReferencia, 102  
setUuid, 102  
com.example.smariba\_upv.airflow.API.MODELS.SensorResponse, 103  
getIdSensor, 104  
getIdUsuario, 104  
getSensor, 104  
setIdSensor, 104  
setIdUsuario, 104  
setSensor, 104  
com.example.smariba\_upv.airflow.API.RetrofitClient, 79  
get ApiService, 79  
com.example.smariba\_upv.airflow.LOGIC, 7  
com.example.smariba\_upv.airflow.LOGIC.BiometricUtil, 21

authenticate, 22  
 com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.BiometricUtil, 18  
     onAuthenticationError, 20  
     onAuthenticationFailed, 20  
     onAuthenticationSucceeded, 21  
 com.example.smariba\_upv.airflow.LOGIC.PeticionesUserUtil, 72  
     editUsuario, 73  
     login, 73  
     registrarSensor, 74  
 com.example.smariba\_upv.airflow.LOGIC.Utilidades, 119  
     bytesToHexString, 121  
     bytesToInt, 121  
     bytesToIntOK, 122  
     bytesToLong, 122  
     bytesToString, 123  
     dosLongToBytes, 123  
     getFechaActual, 124  
     stringToBytes, 124  
     stringToUUID, 124  
     uuidToHexString, 125  
     uuidToString, 125  
 com.example.smariba\_upv.airflow.MainActivity, 42  
     botonBuscarDispositivosBTLEPulsado, 45  
     botonBuscarNuestroDispositivoBTLEPulsado, 45  
     botonDetenerBusquedaDispositivosBTLEPulsado, 45  
     getMedicionesBeacon, 45  
     onCreate, 46  
     onRequestPermissionsResult, 46  
 com.example.smariba\_upv.airflow.POJO, 7  
 com.example.smariba\_upv.airflow.POJO.Medicion, 55  
     getFecha, 57  
     getId, 57  
     getIdSeñor, 57  
     getLatitud, 57  
     getLongitud, 57  
     getTipoGas, 57  
     getValor, 58  
     Medicion, 56  
     setFecha, 58  
     setId, 58  
     setIdSeñor, 58  
     setLatitud, 59  
     setLongitud, 59  
     setTipoGas, 59  
     setValor, 59  
 com.example.smariba\_upv.airflow.POJO.SensorObject, 92  
     getBateria, 93  
     getEstado, 93  
     getId, 94  
     getNombre, 94  
     getNum\_ref, 94  
     getUUID, 95  
     isConexion, 95  
     SensorObject, 93  
     setBateria, 95  
     setConexion, 96  
     setEstado, 96  
     setId, 96  
     setNombre, 96  
     setNum\_ref, 96  
     setUUID, 96  
     toString, 96  
 com.example.smariba\_upv.airflow.POJO.TramalBeacon, 105  
     getAdvFlags, 106  
     getAdvHeader, 106  
     getCompanyID, 106  
     getBeaconLength, 106  
     getBeaconType, 107  
     getLosBytes, 107  
     getMajor, 107  
     getMinor, 107  
     getPrefijo, 108  
     getTxPower, 108  
     getUUID, 108  
     TramalBeacon, 106  
 com.example.smariba\_upv.airflow.POJO.User, 113  
     getApellidos, 115  
     getContrasenya, 115  
     getEmail, 116  
     getId, 116  
     getNombre, 116  
     getTelefono, 117  
     setApellidos, 117  
     setContrasenya, 118  
     setEmail, 118  
     setId, 118  
     setNombre, 118  
     setTelefono, 119  
     User, 114, 115  
 com.example.smariba\_upv.airflow.PRESENTACION, 8  
 com.example.smariba\_upv.airflow.PRESENTACION.EditarPerfilActivity, 23  
     guardarCambios, 25  
     onCreate, 25  
 com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment, 32  
     HomeFragment, 34  
     newInstance, 34  
     onCreate, 34  
     onCreateView, 35  
 com.example.smariba\_upv.airflow.PRESENTACION.LandActivity, 35  
     onCreate, 36  
     onRequestPermissionsResult, 37  
 com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity, 37  
     biometricLogIn, 40  
     logIn, 40  
     onAuthenticationError, 40  
     onAuthenticationFailed, 41

onAuthenticationSucceeded, 41  
onCreate, 41  
onRequestPermissionsResult, 42  
com.example.smariba\_upv.airflow.PRESENTACION.MapFragment, 13  
    51  
    MapFragment, 53  
    newInstance, 53  
    onCreate, 53  
    onCreateView, 54  
com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCrear, 126  
    64  
    onCreate, 65  
com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity, 128  
    65  
    btnEditarPerfil, 67  
    onCreate, 66  
    txtApellidos, 67  
    txtCorreo, 67  
    txtNombre, 67  
    txtTelefono, 67  
com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment, 132  
    68  
    newInstance, 70  
    onActivityResult, 71  
    onCreate, 71  
    oncreateView, 71  
    PerfilFragment, 70  
com.example.smariba\_upv.airflow.PRESENTACION.QRreader, 50  
    75  
    initQR, 77  
    onCreate, 78  
com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment, 60  
    82  
    newInstance, 84  
    onCreate, 84  
    oncreateView, 85  
    SaludFragment, 84  
com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter, 63  
    85  
    getItemCount, 87  
     onBindViewHolder, 87  
    onCreateViewHolder, 88  
    SensorAdapter, 87  
com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment, 89  
    89  
    oncreateView, 91  
    onResume, 91  
    SensorFragment, 90  
com.example.smariba\_upv.airflow.Services, 8  
com.example.smariba\_upv.airflow.Services.ArduinoGetterService, 14  
    14  
    buscarEsteDispositivoBTLE, 15  
     onBind, 16  
    onCreate, 17  
    onDestroy, 17  
com.example.smariba\_upv.btle\_sento, 8  
com.example.smariba\_upv.btle\_sento.API, 8  
    com.example.smariba\_upv.btle\_sento.API.ApiService, 12  
        insertarMedicion, 13  
    com.example.smariba\_upv.btle\_sento.API.RetrofitClient, 80  
        getApiService, 81  
        retrofit, 81  
    com.example.smariba\_upv.btle\_sento.LOGIC, 9  
    com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 126  
        bytesToHexString, 128  
        bytesToInt, 128  
        bytesToIntOK, 129  
        bytesToLong, 129  
        bytesToString, 130  
        dosLongToBytes, 130  
        stringToBytes, 131  
        stringToUUID, 131  
        uuidToHexString, 132  
    com.example.smariba\_upv.btle\_sento.MainActivity, 47  
        botonBuscarDispositivosBTLEPulsado, 49  
        botonBuscarNuestroDispositivoBTLEPulsado, 49  
        botonDetenerBusquedaDispositivosBTLEPulsado, 49  
        getMedicionesBeacon, 49  
    com.example.smariba\_upv.btle\_sento.POJO, 9  
        com.example.smariba\_upv.btle\_sento.POJO.Medicion, 60  
            getGas, 61  
            getLugar, 61  
            getValor, 61  
            Medicion, 61  
            setGas, 62  
            setLugar, 63  
            setValor, 63  
        com.example.smariba\_upv.btle\_sento.POJO.TramalBeacon, 109  
            getAdvFlags, 110  
            getAdvHeader, 110  
            getCompanyID, 110  
            getBeaconLength, 110  
            getBeaconType, 111  
            getLosBytes, 111  
            getMajor, 111  
            getMinor, 111  
            getPrefijo, 111  
            getTxPower, 112  
            getUUID, 112  
            TramalBeacon, 110  
    dosLongToBytes  
        com.example.smariba\_upv.airflow.LOGIC.Utilidades, 123  
        com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades, 130

EditarPerfilActivity.java, 177, 178	com.example.smariba_upv.airflow.POJO.TramalBeacon, 106
editUsuario	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 110
com.example.smariba_upv.airflow.API.ApiService, 10	editSensorType
com.example.smariba_upv.airflow.API.EnviarPeticionesUserUtil, 28	com.example.smariba_upv.airflow.POJO.TramalBeacon, 107
com.example.smariba_upv.airflow.LOGIC.PeticionesUserUtil, 73	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 111
EnviarPeticionesUser	com.example.smariba_upv.airflow.POJO.Medicion, 57
com.example.smariba_upv.airflow.API.EnviarPeticionesUserUtil, 27	com.example.smariba_upv.airflow.POJO.SensorObject, 94
EnviarPeticionesUser.java, 136, 137	getAdvFlags
getAdvFlags	com.example.smariba_upv.airflow.POJO.TramalBeacon, 106
com.example.smariba_upv.airflow.POJO.TramalBeacon, 106	getIdSensor
com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 110	com.example.smariba_upv.airflow.POJO.Medicion, 57
getAdvHeader	getAdvHeader
com.example.smariba_upv.airflow.POJO.TramalBeacon, 106	getIdSensor
com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 110	com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 99
getApellidos	com.example.smariba_upv.airflow.API.MODELS.SensorResponse, 104
com.example.smariba_upv.airflow.POJO.User, 115	getIdUsuario
getApiService	com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 99
com.example.smariba_upv.airflow.API.RetrofitClient, 79	getApiService
com.example.smariba_upv.btle_sento.API.RetrofitClient, 81	com.example.smariba_upv.airflow.API.MODELS.SensorResponse, 104
getBateria	getItemCount
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 99	com.example.smariba_upv.airflow.PRESENTACION.SensorAdapter, 87
getCompanyID	getLatitude
com.example.smariba_upv.airflow.POJO.TramalBeacon, 106	com.example.smariba_upv.airflow.POJO.Medicion, 57
com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 110	getLongitude
getContrasenya	com.example.smariba_upv.airflow.POJO.Medicion, 57
com.example.smariba_upv.airflow.POJO.User, 115	getLosBytes
getEmail	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 111
com.example.smariba_upv.airflow.POJO.User, 116	getLugar
getEstado	com.example.smariba_upv.btle_sento.POJO.Medicion, 61
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 99	getMajor
com.example.smariba_upv.airflow.POJO.SensorObject, 93	com.example.smariba_upv.airflow.POJO.TramalBeacon, 107
getFecha	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 111
com.example.smariba_upv.airflow.POJO.Medicion, 57	getMedicionsBeacon
getFechaActual	com.example.smariba_upv.airflow.MainActivity, 45
com.example.smariba_upv.airflow.LOGIC.Utilidades, 124	com.example.smariba_upv.btle_sento.MainActivity, 49
getGas	getMinor
com.example.smariba_upv.btle_sento.POJO.Medicion, 61	com.example.smariba_upv.airflow.POJO.TramalBeacon, 107
getIbeaconLength	com.example.smariba_upv.btle_sento.POJO.TramalBeacon,

	111	com.example.smariba_upv.airflow.PRESENTACION.QRreader,
getMisSensores	10	com.example.smariba_upv.airflow.API.ApiService, insertarMedicion
	100	com.example.smariba_upv.airflow.API.ApiService, 11
getNombre	100	com.example.smariba_upv.btle_sento.API.ApiService, 13
	104	com.example.smariba_upv.btle_sento.API.ApiService, 13
getNum_ref	94	com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 100
	95	com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 100
getNumReferencia	94	com.example.smariba_upv.airflow.POJO.SensorObject, 95
	100	com.example.smariba_upv.airflow.POJO.SensorObject, 95
getPrefijo	108	com.example.smariba_upv.airflow.POJO.TramalBeacon, login
	111	com.example.smariba_upv.airflow.POJO.TramalBeacon, login
getSensor	104	com.example.smariba_upv.airflow.POJO.TramalBeacon, login
	108	com.example.smariba_upv.airflow.POJO.TramalBeacon, login
getTelefono	117	com.example.smariba_upv.airflow.POJO.User, 100
getTipoGas	57	com.example.smariba_upv.airflow.POJO.Medicion, 56
getTxPower	108	com.example.smariba_upv.airflow.POJO.TramalBeacon, Medicion
	112	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, Medicion
getUUID	95	com.example.smariba_upv.airflow.POJO.SensorObject, Medicion.java, 166–168
	108	com.example.smariba_upv.airflow.POJO.TramalBeacon, newInstance
getUuid	100	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, newInstanceState
	112	com.example.smariba_upv.airflow.POJO.TramalBeacon, newInstanceState
getValor	58	com.example.smariba_upv.airflow.POJO.Medicion, 84
	61	com.example.smariba_upv.btle_sento.POJO.Medicion, obtenerMisSensores
guardarCambios	25	com.example.smariba_upv.airflow.PRESENTACION.EditarPerfilActivity, onActivityResult
HomeFragment	34	com.example.smariba_upv.airflow.PRESENTACION.HomeFragment, onAuthenticationError
	34	com.example.smariba_upv.airflow.LOGIC.BiometricUtil.BiometricAuth, 20
HomeFragment.java	179	com.example.smariba_upv.airflow.PRESENTACION.LogInActivity, 40
initQR		

onAuthenticationFailed  
 com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.RequestPermissionsResultResult  
 20  
 com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity  
 41  
 onAuthenticationSucceeded  
 com.example.smariba\_upv.airflow.LOGIC.BiometricUtil.BiometricAuthListener  
 21  
 com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity  
 41  
 onBind  
 com.example.smariba\_upv.airflow.Services.ArduinoGetterService  
 16  
 onBindViewHolder  
 com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter  
 87  
 onCreate  
 com.example.smariba\_upv.airflow.MainActivity  
 46  
 com.example.smariba\_upv.airflow.PRESENTACION.EdadFragment  
 25  
 com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment  
 34  
 com.example.smariba\_upv.airflow.PRESENTACION.LandActivity  
 36  
 com.example.smariba\_upv.airflow.PRESENTACION.LoginActivity  
 41  
 com.example.smariba\_upv.airflow.PRESENTACION.MapFragment  
 53  
 com.example.smariba\_upv.airflow.PRESENTACION.PaginaDeCarga  
 65  
 com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity  
 66  
 com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment  
 71  
 com.example.smariba\_upv.airflow.PRESENTACION.QRreader  
 78  
 com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment  
 84  
 com.example.smariba\_upv.airflow.Services.ArduinoGetterService  
 17  
 com.example.smariba\_upv.btle\_sento.MainActivity  
 50  
 onCreate  
 com.example.smariba\_upv.airflow.PRESENTACION.HomeFragment  
 35  
 com.example.smariba\_upv.airflow.PRESENTACION.MapFragment  
 54  
 com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment  
 71  
 com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment  
 85  
 com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment  
 91  
 onCreate  
 com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter  
 88  
 onDestroy  
 com.example.smariba\_upv.airflow.Services.ArduinoGetterService  
 13  
 onRequestPermissionsResult  
 com.example.smariba\_upv.airflow.MainActivity  
 46  
 Activity  
 com.example.smariba\_upv.airflow.PRESENTACION.LandActivity  
 37  
 onResume  
 com.example.smariba\_upv.airflow.PRESENTACION.LogInActivity  
 50  
 PaginaDeCarga.java, 187, 188  
 PerfilActivity.java, 188, 189  
 PerfilFragment  
 com.example.smariba\_upv.airflow.PRESENTACION.PerfilFragment  
 70  
 PerfilFragment.java, 190  
 PeticionesUserUtil.java, 148  
 Precio  
 com.example.smariba\_upv.btle\_sento.API.ApiService  
 13  
 QRreader.java, 191, 192  
 registrarSensor  
 com.example.smariba\_upv.airflow.API.ApiService  
 11  
 com.example.smariba\_upv.airflow.API.EnviarPeticionesUser  
 30  
 com.example.smariba\_upv.airflow.LOGIC.PeticionesUserUtil  
 74  
 retrofit  
 com.example.smariba\_upv.btle\_sento.API.RetrofitClient  
 81  
 RetrofitClient.java, 143–146  
 SaludFragment  
 com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment  
 84  
 SaludFragment  
 com.example.smariba\_upv.airflow.PRESENTACION.SaludFragment  
 17  
 SensorAdapter  
 com.example.smariba\_upv.airflow.PRESENTACION.SensorAdapter  
 87  
 SensorAdapter.java, 196  
 SensorFragment  
 com.example.smariba\_upv.airflow.PRESENTACION.SensorFragment  
 90  
 SensorFragment.java, 198  
 SensorObject  
 com.example.smariba\_upv.airflow.POJO.SensorObject  
 93  
 SensorObject.java, 170  
 SensorRequest  
 com.example.smariba\_upv.airflow.API.MODELS.SensorRequest  
 98  
 SensorRequest.java, 139, 140  
 SensorResponse.java, 141, 142

setApellidos	com.example.smariba_upv.airflow.POJO.SensorObject, 96
com.example.smariba_upv.airflow.POJO.User, 117	
setBateria	com.example.smariba_upv.airflow.POJO.User, 118
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 100	
com.example.smariba_upv.airflow.POJO.SensorObject, 95	com.example.smariba_upv.airflow.POJO.SensorObject, 96
setConexion	setNumReferencia com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 102
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 101	setSensor
com.example.smariba_upv.airflow.POJO.SensorObject, 96	com.example.smariba_upv.airflow.API.MODELS.SensorResponse, 104
setContrasenya	setTelefono com.example.smariba_upv.airflow.POJO.User, 118
setEmail	com.example.smariba_upv.airflow.POJO.User, 119
setEstado	setTipoGas com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 101
com.example.smariba_upv.airflow.POJO.SensorObject, 96	com.example.smariba_upv.airflow.POJO.SensorObject, 96
setFecha	setUuid com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 102
com.example.smariba_upv.airflow.POJO.Medicion, 58	
setGas	setValor com.example.smariba_upv.btle_sento.POJO.Medicion, 62
setId	com.example.smariba_upv.airflow.POJO.Medicion, 58
com.example.smariba_upv.airflow.POJO.SensorObject, 96	stringToBytes com.example.smariba_upv.airflow.LOGIC.Utilidades, 124
com.example.smariba_upv.airflow.POJO.User, 118	com.example.smariba_upv.btle_sento.LOGIC.Utilidades, 131
setIdSenor	stringToUUID com.example.smariba_upv.airflow.LOGIC.Utilidades, 124
com.example.smariba_upv.airflow.POJO.Medicion, 58	com.example.smariba_upv.btle_sento.LOGIC.Utilidades, 131
setIdSensor	toString com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 101
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 104	com.example.smariba_upv.airflow.POJO.SensorObject, 96
setIdUsuario	TramalBeacon com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 101
com.example.smariba_upv.airflow.API.MODELS.SensorResponse, 104	com.example.smariba_upv.airflow.POJO.TramalBeacon, 106
setLatitud	com.example.smariba_upv.btle_sento.POJO.TramalBeacon, 110
com.example.smariba_upv.airflow.POJO.Medicion, 59	TramalBeacon.java, 172–175
setLongitud	txtApellidos com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity, 67
com.example.smariba_upv.airflow.POJO.Medicion, 59	txtCorreo com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity, 67
setLugar	txtNombre com.example.smariba_upv.btle_sento.POJO.Medicion, 63
setNombre	com.example.smariba_upv.airflow.PRESENTACION.PerfilActivity, 67
com.example.smariba_upv.airflow.API.MODELS.SensorRequest, 102	txtTelefono

com.example.smariba\_upv.airflow.PRESENTACION.PerfilActivity,  
67

User

com.example.smariba\_upv.airflow.POJO.User,  
114, 115

User.java, 176

Utilidades.java, 149–151, 153

uuidToHexString

com.example.smariba\_upv.airflow.LOGIC.Utilidades,  
125

com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades,  
132

uuidToString

com.example.smariba\_upv.airflow.LOGIC.Utilidades,  
125

com.example.smariba\_upv.btle\_sento.LOGIC.Utilidades,  
132