Proyecto3A_Android

Generado por Doxygen 1.12.0

Capítulo 1

Proyecto3A_Android

Este proyecto es una aplicación Android que utiliza beacons para medir niveles de ozono y temperatura. La aplicación se conecta a un servidor para almacenar y recuperar datos de sensores a través de una API REST.

1.1. Funcionalidades

- **Detección de Dispositivos BTLE**: La aplicación busca y se conecta a beacons Bluetooth Low Energy.
- **Medición de Sensores**: Recopila datos de sensores (ozono y temperatura) y los envía al servidor.
- **Gestión de Usuarios**: Permite la creación y eliminación de usuarios en el sistema.
- **Interfaz de Usuario**: Interfaz sencilla para interactuar con la aplicación y visualizar datos.

1.2. Requisitos

- Android Studio
- SDK de Android (versión mínima recomendada: 21)
- Dependencias:
  - Retrofit para la comunicación con la API REST
  - Gson para la conversión de JSON

1.3. Estructura del Proyecto

```
/app
    /src
        /main
            /java
                /com.example.usuario_upv.proyecto3a
                - MainActivity.java
                - RetrofitClient.java
                - SensorApi.java
                - SensorData.java
                - User.java
                - Utilidades.java
                - TramaIBeacon.java
            /res
                /layout
                /drawable
                /values
```

1.3.1. Estructura de la Base de Datos

- **Tabla `sensors`**:
  - `id`: Identificador único.
  - `type`: Tipo de sensor (ozono, temperatura).
  - `value`: Valor medido.
  - `timestamp`: Marca de tiempo de la medición.
  - `user_id`: Identificador del usuario.
- **Tabla `users`**:
  - `id`: Identificador único.
  - `username`: Nombre de usuario.

1.4. Instalación

1. Clona este repositorio:
   ```
   git clone https://github.com/Javitax47/Proyecto3A_Android.git
   ```
2. Abre el proyecto en Android Studio.
3. Asegúrate de tener configuradas las dependencias necesarias en tu archivo build.gradle.
4. Ejecuta la aplicación en un dispositivo Android.

1.5. Uso

1. Abre la aplicación en tu dispositivo Android.
2. Usa los botones de la interfaz para buscar dispositivos BTLE y detener la búsqueda.
3. Se mostrarán los datos de los sensores en la interfaz.

1.6. Contribuciones

Si deseas contribuir a este proyecto, por favor, crea un fork del repositorio y envía un pull request con tus cambios.

1.7. Licencia

Este proyecto está bajo la Licencia MIT. Consulta el archivo LICENSE para más detalles.

Capítulo 2

Índice de espacios de nombres

2.1. Lista de espacios de nombres

Lista de los espacios de nombres documentados, con breves descripciones:

Capítulo 3

Índice jerárquico

3.1. Jerarquía de clases

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

Capítulo 4

Índice de clases

4.1. Lista de clases

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

Capítulo 5

Índice de archivos

5.1. Lista de archivos

Lista de todos los archivos con breves descripciones:

Capítulo 6

Documentación de espacios de nombres

6.1. Paquete com.example.usuario_upv.proyecto3a

Clases

- class AcercaDeActivity

  Clase que implementa la actividad "Acerca de".
- class Ajuste

  Clase que modela un ajuste con un texto descriptivo y una imagen.
- class AjustesAdapter

  Adaptador para manejar la lista de ajustes en el RecyclerView.
- class AlertActivity

  Clase que implementa la actividad para mostrar y gestionar alertas.
- class AlertaData

  Clase que modela los datos de una alerta.
- enum Alertas
- class AlertasAdapter

  Adaptador para el RecyclerView que maneja la lista de alertas.
- class BLEService

  Servicio encargado de gestionar la conectividad BLE y alertas asociadas.
- class Config

  Clase de configuración de la aplicación.
- class LandingPageActivity
- interface LogicaFake

  Interfaz para definir los endpoints de la API de sensores.
- class LoginActivity

  Clase que implementa la actividad de inicio de sesión.
- class MainActivity

  MainActivity de la aplicación.
- class NotificationReceiver
- class Privacidad

  Actividad para la política de privacidad.
- class RegisterActivity

  Actividad para el registro de usuarios.
- class RetrofitClient

  Clase para gestionar la instancia de Retrofit.

- class Sensor

  Clase que representa un sensor con un UUID y un email asociado.
- class SensorAdapter

  Adaptador para mostrar una lista de sensores en un RecyclerView.
- class SensorData

  Clase que representa los datos de un sensor.
- class SplashActivity
- class Tab1

  Fragmento que muestra un mapa con datos de sensores.
- class Tab3

  Fragmento que representa la tercera pestaña de la aplicación.
- class Tab4

  Fragmento que representa la cuarta pestaña de la aplicación.
- class Timer

  Clase que representa un temporizador.
- class TramaIBeacon

  Clase que representa la trama de un beacon iBeacon.
- class User

  Clase que representa un usuario.
- class UserConfig
- class UserData

  A class to represent user data including username, email, and password.
- class Utilidades

  Clase utilitaria para conversiones entre diferentes tipos de datos.

Capítulo 7

Documentación de clases

7.1. Referencia de la clase com.example.usuario_upv.proyecto3a.AcercaDeActivity

Clase que implementa la actividad "Acerca de".

Diagrama de herencia de com.example.usuario_upv.proyecto3a.AcercaDeActivity

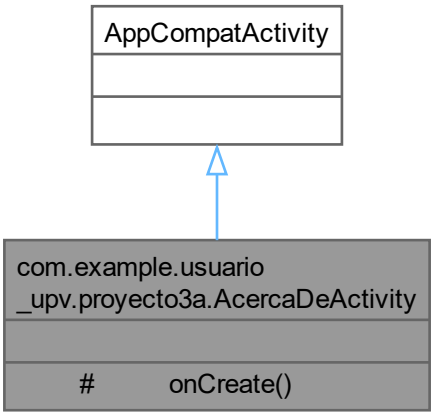
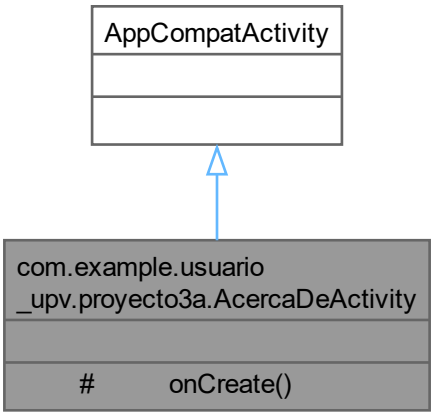


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AcercaDeActivity:



Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método que se llama al crear la actividad.

7.1.1. Descripción detallada

Clase que implementa la actividad "Acerca de".

Esta clase extiende AppCompatActivity y se utiliza para mostrar una pantalla informativa dentro de la aplicación.

7.1.2. Documentación de funciones miembro

7.1.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.AcercaDeActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método que se llama al crear la actividad.

Este método inicializa la interfaz de usuario y establece el diseño correspondiente a la actividad "Acerca de".

Parámetros

savedInstanceState	Estado guardado de la actividad en ejecuciones anteriores.

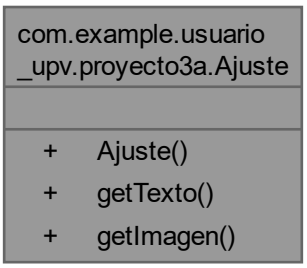
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AcercaDeActivity.java

7.2. Referencia de la clase com.example.usuario_upv.proyecto3a.Ajuste

Clase que modela un ajuste con un texto descriptivo y una imagen.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Ajuste:



Métodos públicos

- Ajuste (String texto, int imagen)
  Constructor de la clase Ajuste.
- String getTexto ()
  Obtiene el texto descriptivo del ajuste.
- int getImagen ()
  Obtiene el ID de la imagen asociada al ajuste.

7.2.1. Descripción detallada

Clase que modela un ajuste con un texto descriptivo y una imagen.

Proporciona métodos para acceder al texto y la imagen del ajuste.

7.2.2. Documentación de constructores y destructores

7.2.2.1. Ajuste()

```
com.example.usuario_upv.proyecto3a.Ajuste.Ajuste (
            String texto,
            int imagen)  [inline]
```

Constructor de la clase Ajuste.

Inicializa un nuevo objeto Ajuste con el texto y la imagen proporcionados.

Parámetros

texto	Texto descriptivo del ajuste.
imagen	ID del recurso de la imagen asociada.

7.2.3. Documentación de funciones miembro

7.2.3.1. getImagen()

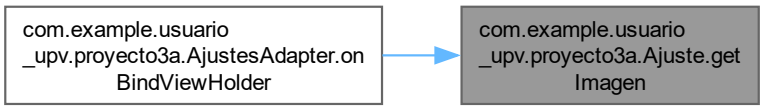
```
int com.example.usuario_upv.proyecto3a.Ajuste.getImagen ()  [inline]
```

Obtiene el ID de la imagen asociada al ajuste.

Devuelve

ID del recurso de la imagen.

Gráfico de llamadas a esta función:



7.2.3.2. getTexto()

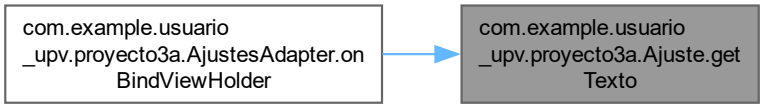
```
String com.example.usuario_upv.proyecto3a.Ajuste.getTexto ()  [inline]
```

Obtiene el texto descriptivo del ajuste.

Devuelve

Texto descriptivo del ajuste.

Gráfico de llamadas a esta función:



La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Ajuste.java

7.3. Referencia de la clase com.example.usuario_upv.proyecto3a.AjustesAdapter

Adaptador para manejar la lista de ajustes en el RecyclerView.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.AjustesAdapter

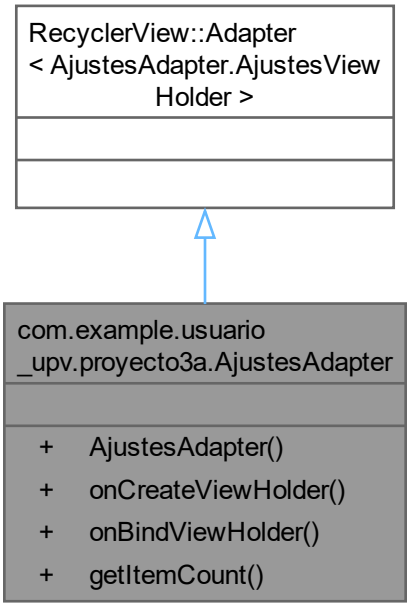
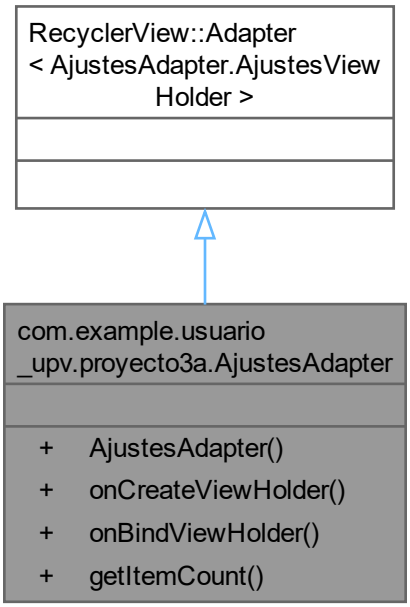


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AjustesAdapter:



Clases

- class **AjustesViewHolder**

  Clase ViewHolder que contiene las vistas de un elemento.
- interface OnItemClickListener

  Interfaz para manejar eventos de clic en los elementos.

Métodos públicos

- AjustesAdapter (List< Ajuste > ajustesList, Context context, OnItemClickListener onItemClickListener)

  Constructor de AjustesAdapter.
- AjustesViewHolder onCreateViewHolder (@NonNull ViewGroup parent, int viewType)

  Crea un nuevo ViewHolder para un elemento del RecyclerView.
- void onBindViewHolder (@NonNull AjustesViewHolder holder, int position)

  Vincula los datos de un ajuste a una vista.
- int getItemCount ()

  Devuelve el número de elementos en la lista.

7.3.1. Descripción detallada

Adaptador para manejar la lista de ajustes en el RecyclerView.

Este adaptador facilita la visualización de cada elemento en la lista de ajustes, inflando un layout personalizado y gestionando eventos de clic.

7.3.2. Documentación de constructores y destructores

7.3.2.1. AjustesAdapter()

```
com.example.usuario_upv.proyecto3a.AjustesAdapter.AjustesAdapter (
            List< Ajuste > ajustesList,
            Context context,
            OnItemClickListener onItemClickListener)  [inline]
```

Constructor de AjustesAdapter.

Inicializa el adaptador con la lista de ajustes, el contexto y el listener de clics.

Parámetros

ajustesList	Lista de ajustes.
context	Contexto de la aplicación.
onItemClickListener	Listener para manejar los eventos de clic.

7.3.3. Documentación de funciones miembro

7.3.3.1. getItemCount()

```
int com.example.usuario_upv.proyecto3a.AjustesAdapter.getItemCount ()  [inline]
```

Devuelve el número de elementos en la lista.

Devuelve

Cantidad de elementos en la lista de ajustes.

7.3.3.2. onBindViewHolder()

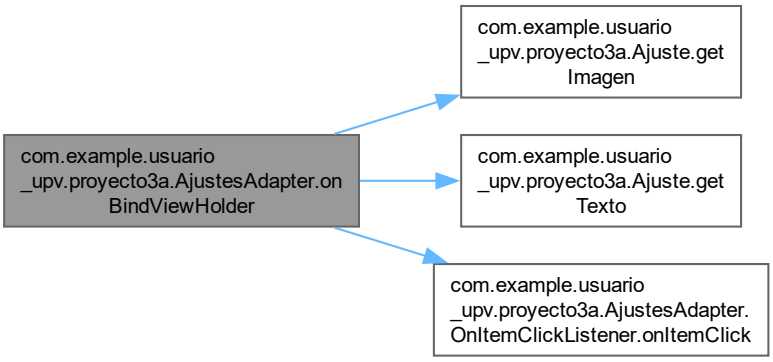
```
void com.example.usuario_upv.proyecto3a.AjustesAdapter.onBindViewHolder (
            @NonNull AjustesViewHolder holder,
            int position)  [inline]
```

Vincula los datos de un ajuste a una vista.

Parámetros

holder	ViewHolder que contiene la vista del elemento.
position	Posición del elemento en la lista.

Gráfico de llamadas de esta función:



7.3.3.3. onCreateViewHolder()

```
AjustesViewHolder com.example.usuario_upv.proyecto3a.AjustesAdapter.onCreateViewHolder (
            @NonNull ViewGroup parent,
            int viewType)  [inline]
```

Crea un nuevo ViewHolder para un elemento del RecyclerView.

Parámetros

parent	Vista padre donde se incluirá el nuevo elemento.
viewType	Tipo de vista (no utilizado en este caso).

Devuelve

Un nuevo objeto AjustesViewHolder.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AjustesAdapter.java

7.4. Referencia de la clase com.example.usuario_upv.proyecto3a.AlertActivity

Clase que implementa la actividad para mostrar y gestionar alertas.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.AlertActivity

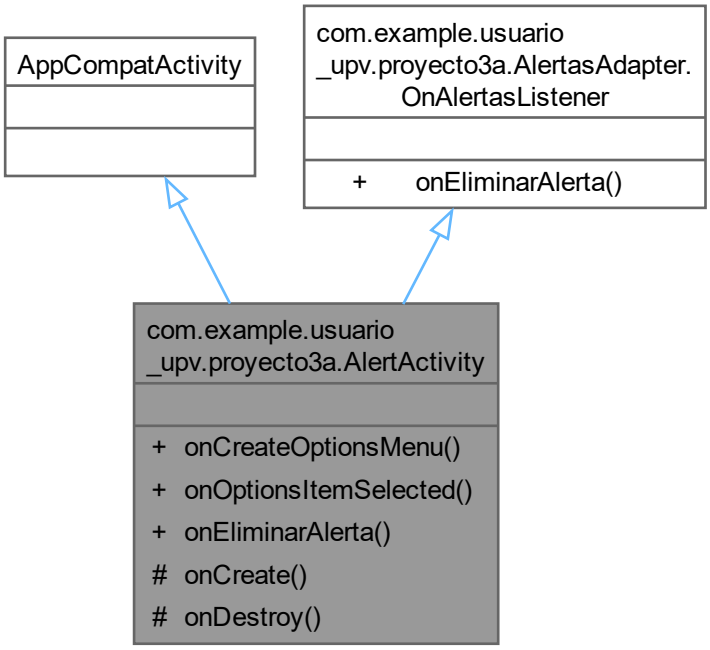
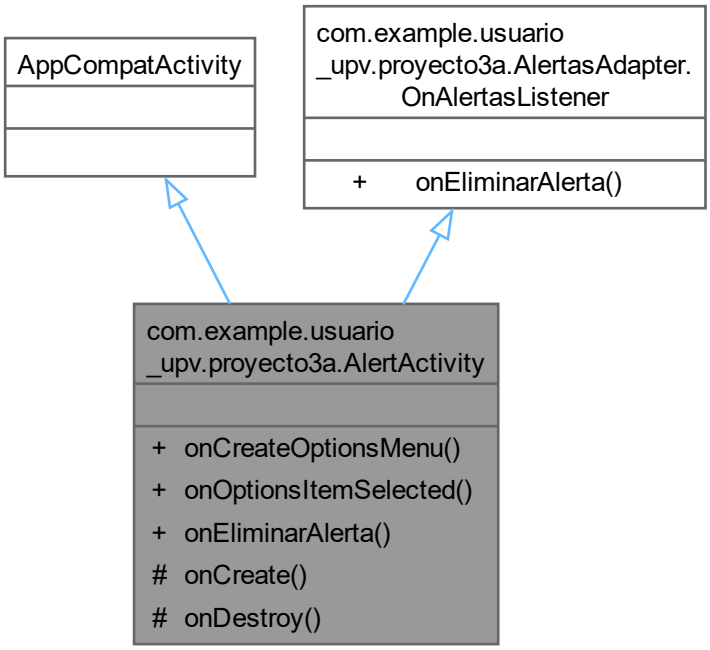


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AlertActivity:



Métodos públicos

- boolean onCreateOptionsMenu (Menu menu)

  Infla el menú de opciones.
- boolean onOptionsItemSelected (MenuItem item)

  Maneja las opciones seleccionadas en el menú.
- void onEliminarAlerta (int position)

  Maneja la eliminación de una alerta por posición.

Métodos públicos heredados de com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener

Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método que se llama al crear la actividad.
- void onDestroy ()

  Método que se llama cuando se destruye la actividad.

7.4.1. Descripción detallada

Clase que implementa la actividad para mostrar y gestionar alertas.

La actividad utiliza un RecyclerView para mostrar alertas que se reciben mediante un BroadcastReceiver. También incluye funcionalidades para borrar alertas y gestionar el menú.

7.4.2. Documentación de funciones miembro

7.4.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.AlertActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método que se llama al crear la actividad.

Inicializa el RecyclerView, el adaptador, y registra un BroadcastReceiver para recibir alertas.

Parámetros

savedInstanceState	Estado guardado de la actividad.

7.4.2.2. onCreateOptionsMenu()

```
boolean com.example.usuario_upv.proyecto3a.AlertActivity.onCreateOptionsMenu (
            Menu menu)  [inline]
```

Infla el menú de opciones.

Parámetros

menu	Objeto Menu a inflar.

Devuelve

true si el menú se infla correctamente.

7.4.2.3. onDestroy()

```
void com.example.usuario_upv.proyecto3a.AlertActivity.onDestroy ()  [inline], [protected]
```

Método que se llama cuando se destruye la actividad.

Desregistra el BroadcastReceiver para evitar fugas de memoria.

7.4.2.4. onEliminarAlerta()

```
void com.example.usuario_upv.proyecto3a.AlertActivity.onEliminarAlerta (
            int position)  [inline]
```

Maneja la eliminación de una alerta por posición.

Parámetros

position	Posición de la alerta a eliminar.

Implementa com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener.

7.4.2.5. onOptionsItemSelected()

```
boolean com.example.usuario_upv.proyecto3a.AlertActivity.onOptionsItemSelected (
            MenuItem item)  [inline]
```

Maneja las opciones seleccionadas en el menú.

Parámetros

item	Elemento del menú seleccionado.

Devuelve

true si se maneja la acción correctamente.

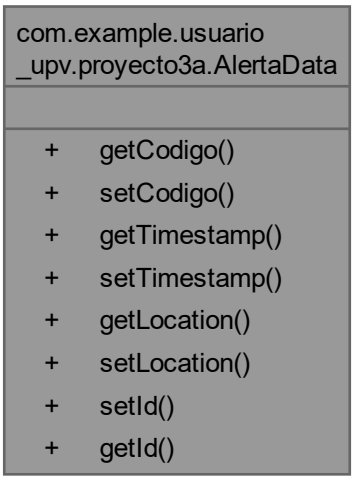
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AlertActivity.java

7.5. Referencia de la clase com.example.usuario_upv.proyecto3a.AlertaData

Clase que modela los datos de una alerta.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AlertaData:



Clases

- class **Location**

  Clase interna que representa la ubicación de una alerta.

Métodos públicos

- int getCodigo ()
  Obtiene el código de la alerta.
- void setCodigo (int codigo)
  Establece el código de la alerta.
- String getTimestamp ()
  Obtiene la marca de tiempo de la alerta.
- void setTimestamp (String timestamp)
  Establece la marca de tiempo de la alerta.
- Location getLocation ()
  Obtiene la ubicación de la alerta.
- void setLocation (Location location)
  Establece la ubicación de la alerta.
- void setId (int id)
  Establece el identificador único de la alerta.

Métodos públicos estáticos

- static int getId ()
  Obtiene el identificador único de la alerta.

7.5.1. Descripción detallada

Clase que modela los datos de una alerta.

Contiene información como el código de la alerta, marca de tiempo, ubicación y un identificador único.

7.5.2. Documentación de funciones miembro

7.5.2.1. getCodigo()

```
int com.example.usuario_upv.proyecto3a.AlertaData.getCodigo ()  [inline]
```

Obtiene el código de la alerta.

Devuelve

Código de la alerta.

7.5.2.2. getId()

```
static int com.example.usuario_upv.proyecto3a.AlertaData.getId ()  [inline], [static]
```

Obtiene el identificador único de la alerta.

Devuelve

Identificador único de la alerta.

7.5.2.3. getLocation()

```
Location com.example.usuario_upv.proyecto3a.AlertaData.getLocation ()  [inline]
```

Obtiene la ubicación de la alerta.

Devuelve

Objeto Location que representa la ubicación de la alerta.

7.5.2.4. getTimestamp()

```
String com.example.usuario_upv.proyecto3a.AlertaData.getTimestamp ()  [inline]
```

Obtiene la marca de tiempo de la alerta.

Devuelve

Marca de tiempo de la alerta.

7.5.2.5. setCodigo()

```
void com.example.usuario_upv.proyecto3a.AlertaData.setCodigo (
            int codigo)  [inline]
```

Establece el código de la alerta.

Parámetros

codigo	Código de la alerta.

7.5.2.6. setId()

```
void com.example.usuario_upv.proyecto3a.AlertaData.setId (
            int id)  [inline]
```

Establece el identificador único de la alerta.

Parámetros

id	Identificador único de la alerta.

7.5.2.7. setLocation()

```
void com.example.usuario_upv.proyecto3a.AlertaData.setLocation (
            Location location)  [inline]
```

Establece la ubicación de la alerta.

Parámetros

location	Objeto Location que representa la ubicación de la alerta.

7.5.2.8. setTimestamp()

```
void com.example.usuario_upv.proyecto3a.AlertaData.setTimestamp (
            String timestamp)  [inline]
```

Establece la marca de tiempo de la alerta.

Parámetros

timestamp	Marca de tiempo de la alerta.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AlertaData.java

7.6. Referencia de la enumeración com.example.usuario_upv.proyecto3a.Alertas

Diagrama de herencia de com.example.usuario_upv.proyecto3a.Alertas

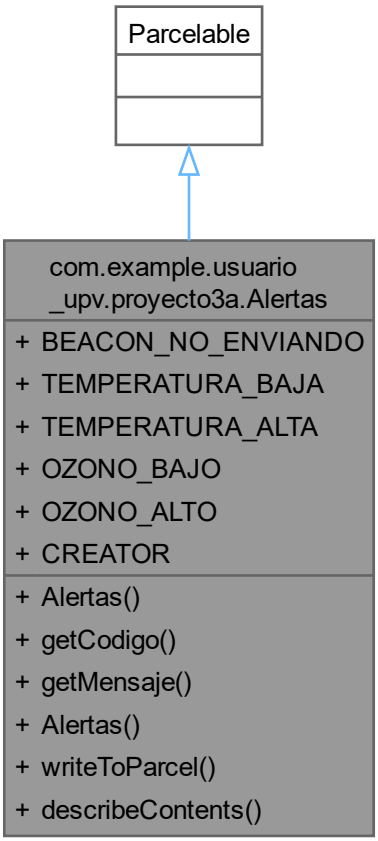
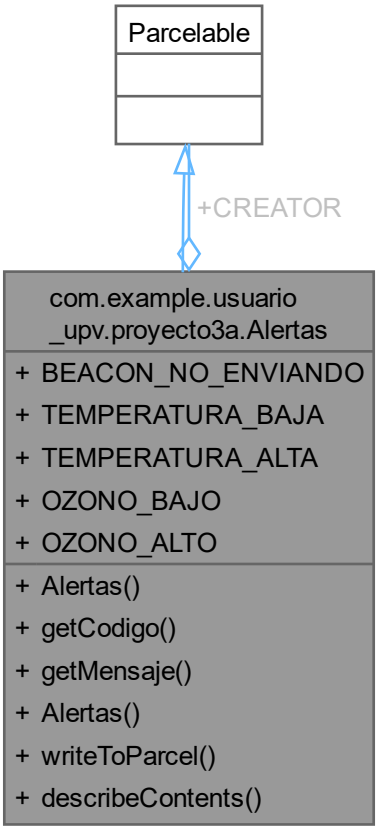


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Alertas:



Métodos públicos

- Alertas (int codigo, String mensaje)

  Constructor del enum Alertas.
- int getCodigo ()

  Obtiene el código de la alerta.
- String getMensaje ()

  Obtiene el mensaje descriptivo de la alerta.
- Alertas (Parcel in)

  Constructor del enum para recrear a partir de un Parcel.
- void writeToParcel (Parcel dest, int flags)

  Escribe el objeto Alertas en un Parcel.
- int describeContents ()

  Describe el contenido del Parcelable.

Atributos públicos

- BEACON_NO_ENVIANDO =(1003, "No se están recibiendo datos del sensor")
- TEMPERATURA_BAJA =(101, "Temperatura muy baja")
- TEMPERATURA_ALTA =(102, "Temperatura excesiva")
- OZONO_BAJO =(201, "Concentración baja de ozono")
- OZONO_ALTO =(202, "Concentración alta de ozono")

Atributos públicos estáticos

- static final Parcelable.Creator< Alertas > CREATOR

  Creador para reconstruir objetos Alertas desde un Parcel.

7.6.1. Documentación de constructores y destructores

7.6.1.1. Alertas() [1/2]

```
com.example.usuario_upv.proyecto3a.Alertas.Alertas (
            int codigo,
            String mensaje)  [inline]
```

Constructor del enum Alertas.

Parámetros

codigo	Código único de la alerta.
mensaje	Mensaje descriptivo de la alerta.

7.6.1.2. Alertas() [2/2]

```
com.example.usuario_upv.proyecto3a.Alertas.Alertas (
            Parcel in)  [inline]
```

Constructor del enum para recrear a partir de un Parcel.

Parámetros

in	Objeto Parcel que contiene los datos de la alerta.

7.6.2. Documentación de funciones miembro

7.6.2.1. describeContents()

```
int com.example.usuario_upv.proyecto3a.Alertas.describeContents ()  [inline]
```

Describe el contenido del Parcelable.

Devuelve

Un entero que describe el tipo de objeto Parcelable.

7.6.2.2. getCodigo()

```
int com.example.usuario_upv.proyecto3a.Alertas.getCodigo ()  [inline]
```

Obtiene el código de la alerta.

Devuelve

Código único de la alerta.

7.6.2.3. getMensaje()

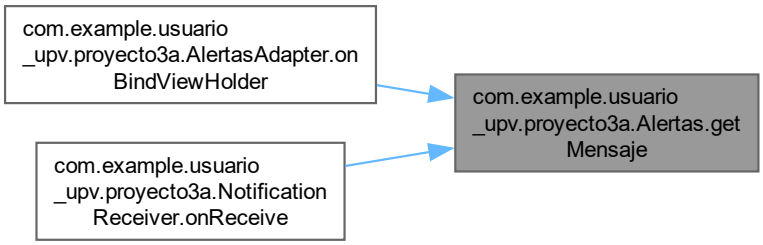
```
String com.example.usuario_upv.proyecto3a.Alertas.getMensaje ()  [inline]
```

Obtiene el mensaje descriptivo de la alerta.

Devuelve

Mensaje descriptivo de la alerta.

Gráfico de llamadas a esta función:



7.6.2.4. writeToParcel()

```
void com.example.usuario_upv.proyecto3a.Alertas.writeToParcel (
            Parcel dest,
            int flags)  [inline]
```

Escribe el objeto Alertas en un Parcel.

Parámetros

dest	Objeto Parcel donde se guardará la alerta.
flags	Indicadores adicionales sobre cómo escribir el objeto.

7.6.3. Documentación de datos miembro

7.6.3.1. BEACON_NO_ENVIANDO

```
com.example.usuario_upv.proyecto3a.Alertas.BEACON_NO_ENVIANDO =(1003, "No se están recibiendo
datos del sensor")
```

7.6.3.2. CREATOR

```
final Parcelable.Creator<Alertas> com.example.usuario_upv.proyecto3a.Alertas.CREATOR  [static]
```

Valor inicial:

```
= new Parcelable.Creator<Alertas>() {
        @Override
        public Alertas createFromParcel(Parcel in) {
            return Alertas.values()[in.readInt()];
        }

        @Override
        public Alertas[] newArray(int size) {
            return new Alertas[size];
        }
    }
```

Creador para reconstruir objetos Alertas desde un Parcel.

7.6.3.3. OZONO_ALTO

```
com.example.usuario_upv.proyecto3a.Alertas.OZONO_ALTO =(202, "Concentración alta de ozono")
```

7.6.3.4. OZONO_BAJO

```
com.example.usuario_upv.proyecto3a.Alertas.OZONO_BAJO =(201, "Concentración baja de ozono")
```

7.6.3.5. TEMPERATURA_ALTA

```
com.example.usuario_upv.proyecto3a.Alertas.TEMPERATURA_ALTA =(102, "Temperatura excesiva")
```

7.6.3.6. TEMPERATURA_BAJA

```
com.example.usuario_upv.proyecto3a.Alertas.TEMPERATURA_BAJA =(101, "Temperatura muy baja")
```

La documentación de esta enumeración está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Alertas.java

7.7. Referencia de la clase com.example.usuario_upv.proyecto3a.AlertasAdapter

Adaptador para el RecyclerView que maneja la lista de alertas.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.AlertasAdapter

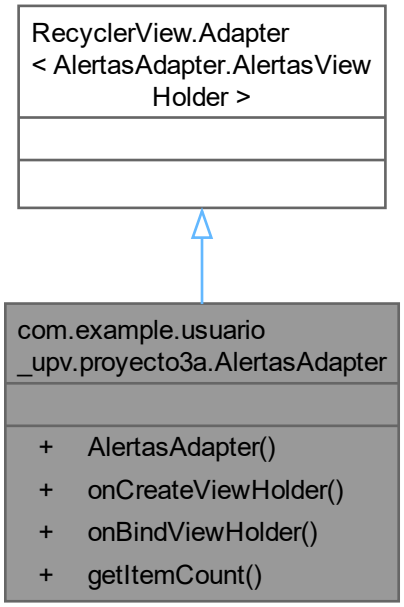
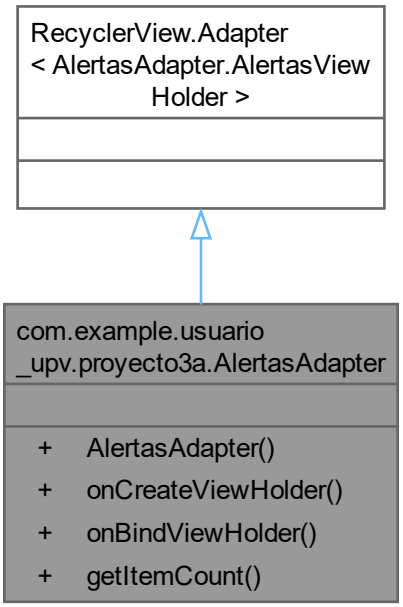


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AlertasAdapter:



Clases

- class **AlertasViewHolder**

  Clase ViewHolder que contiene las vistas de un elemento del RecyclerView.
- interface OnAlertasListener

  Interfaz para manejar eventos de interacción con las alertas.

Métodos públicos

- AlertasAdapter (List< Alertas > alertasList, OnAlertasListener listener)

  Constructor de AlertasAdapter.
- AlertasViewHolder onCreateViewHolder (ViewGroup parent, int viewType)

  Crea un nuevo ViewHolder para un elemento del RecyclerView.
- void onBindViewHolder (AlertasViewHolder holder, int position)

  Vincula los datos de una alerta a una vista.
- int getItemCount ()

  Devuelve el número de elementos en la lista de alertas.

7.7.1. Descripción detallada

Adaptador para el RecyclerView que maneja la lista de alertas.

Facilita la vinculación entre los datos de las alertas y las vistas de la interfaz, proporcionando funcionalidades para mostrar y eliminar alertas.

7.7.2. Documentación de constructores y destructores

7.7.2.1. AlertasAdapter()

```
com.example.usuario_upv.proyecto3a.AlertasAdapter.AlertasAdapter (
            List< Alertas > alertasList,
            OnAlertasListener listener)  [inline]
```

Constructor de AlertasAdapter.

Inicializa el adaptador con la lista de alertas y el listener para manejar eventos.

Parámetros

alertasList	Lista de alertas.
listener	Listener para manejar eventos de interacción.

7.7.3. Documentación de funciones miembro

7.7.3.1. getItemCount()

```
int com.example.usuario_upv.proyecto3a.AlertasAdapter.getItemCount ()  [inline]
```

Devuelve el número de elementos en la lista de alertas.

Devuelve

Cantidad de alertas en la lista.

7.7.3.2. onBindViewHolder()

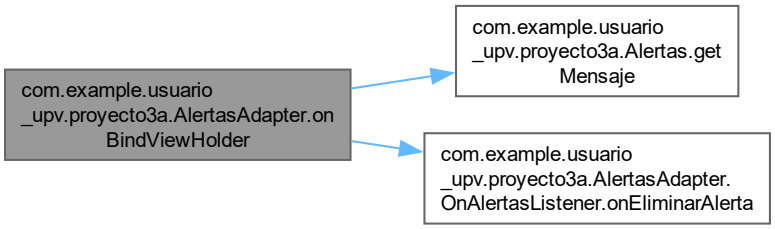
```
void com.example.usuario_upv.proyecto3a.AlertasAdapter.onBindViewHolder (
            AlertasViewHolder holder,
            int position)  [inline]
```

Vincula los datos de una alerta a una vista.

Parámetros

holder	ViewHolder que contiene la vista del elemento.
position	Posición del elemento en la lista.

Gráfico de llamadas de esta función:



7.7.3.3. onCreateViewHolder()

```
AlertasViewHolder com.example.usuario_upv.proyecto3a.AlertasAdapter.onCreateViewHolder (
            ViewGroup parent,
            int viewType)  [inline]
```

Crea un nuevo ViewHolder para un elemento del RecyclerView.

Parámetros

parent	Vista padre donde se incluirá el nuevo elemento.
viewType	Tipo de vista (no utilizado en este caso).

Devuelve

Un nuevo objeto AlertasViewHolder.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AlertasAdapter.java

7.8. Referencia de la clase com.example.usuario_upv.proyecto3a.BLEService

Servicio encargado de gestionar la conectividad BLE y alertas asociadas.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.BLEService

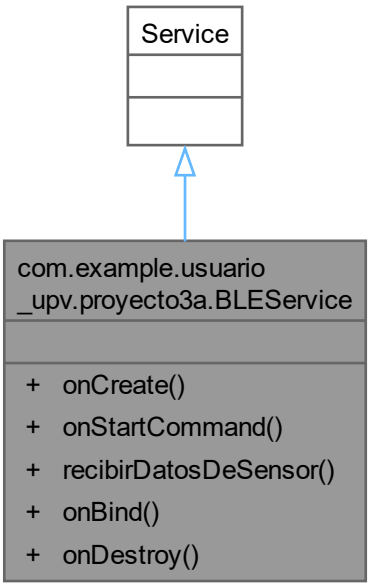
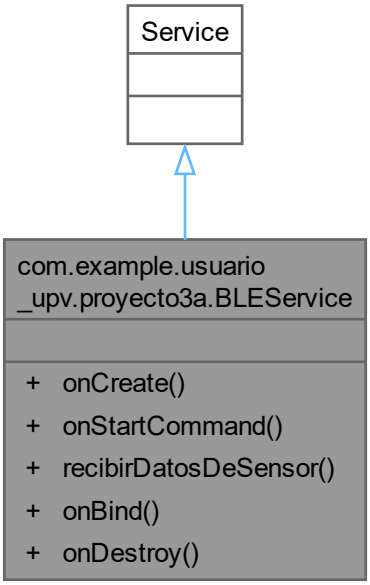


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.BLEService:



Métodos públicos

- void onCreate ()

  Método llamado cuando el servicio es creado.
- int onStartCommand (Intent intent, int flags, int startId)

  Método llamado cuando se inicia el servicio.
- void recibirDatosDeSensor (int majorValue, int minorValue)

  Maneja la recepción de datos del sensor.
- IBinder onBind (Intent intent)
- void onDestroy ()

  Método llamado cuando el servicio es destruido.

7.8.1. Descripción detallada

Servicio encargado de gestionar la conectividad BLE y alertas asociadas.

BLEService implementa la gestión de dispositivos BLE, el envío de notificaciones relacionadas con eventos y la comunicación con un servidor remoto utilizando Retrofit.

7.8.2. Documentación de funciones miembro

7.8.2.1. onBind()

```
IBinder com.example.usuario_upv.proyecto3a.BLEService.onBind (
            Intent intent)  [inline]
```

7.8.2.2. onCreate()

```
void com.example.usuario_upv.proyecto3a.BLEService.onCreate ()  [inline]
```

Método llamado cuando el servicio es creado.

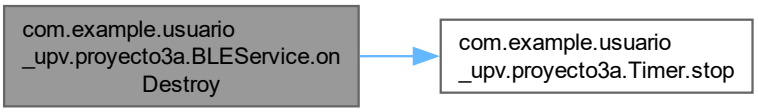
Inicializa el adaptador Bluetooth, configura el canal de notificaciones y prepara el temporizador para manejar la detección de datos faltantes.

7.8.2.3. onDestroy()

```
void com.example.usuario_upv.proyecto3a.BLEService.onDestroy ()  [inline]
```

Método llamado cuando el servicio es destruido.

Detiene el temporizador y realiza tareas de limpieza. Gráfico de llamadas de esta función:



7.8.2.4. onStartCommand()

```
int com.example.usuario_upv.proyecto3a.BLEService.onStartCommand (
            Intent intent,
            int flags,
            int startId)  [inline]
```

Método llamado cuando se inicia el servicio.

Muestra una notificación persistente, inicia el temporizador para detectar la falta de datos y procesa datos del Intent recibido.

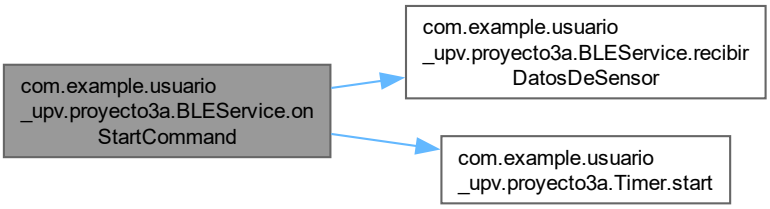
Parámetros

intent	Intent que inicia el servicio.
flags	Indicadores sobre cómo iniciar el servicio.
startId	ID único del inicio del servicio.

Devuelve

Indica cómo debe comportarse el sistema si el servicio es eliminado.

Gráfico de llamadas de esta función:



7.8.2.5. recibirDatosDeSensor()

```
void com.example.usuario_upv.proyecto3a.BLEService.recibirDatosDeSensor (
            int majorValue,
            int minorValue)  [inline]
```

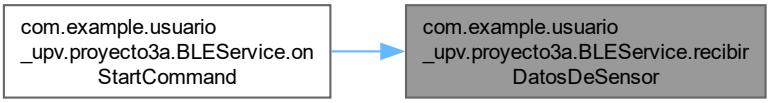
Maneja la recepción de datos del sensor.

Reinicia el temporizador cuando se reciben nuevos datos.

Parámetros

majorValue	Valor mayor recibido del sensor.
minorValue	Valor menor recibido del sensor.

Gráfico de llamadas a esta función:



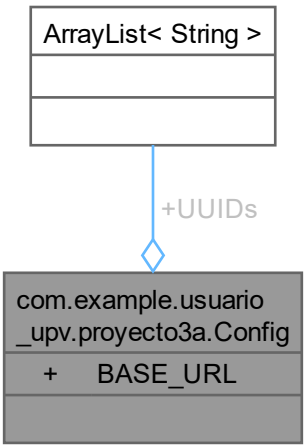
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/BLEService.java

7.9. Referencia de la clase com.example.usuario_upv.proyecto3a.Config

Clase de configuración de la aplicación.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Config:



Atributos públicos estáticos

- static String BASE_URL = "http://airmonitor.ddns.net:13000/"

  URL base para las llamadas a la API.
- static ArrayList< String > UUIDs = new ArrayList<>()

  Lista de UUIDs utilizados en la aplicación.

7.9.1. Descripción detallada

Clase de configuración de la aplicación.

La clase Config proporciona parámetros de configuración globales para la aplicación, como la URL base para las llamadas a la API y una lista de UUIDs.

7.9.2. Documentación de datos miembro

7.9.2.1. BASE_URL

```
String com.example.usuario_upv.proyecto3a.Config.BASE_URL = "http://airmonitor.ddns.net↩
:13000/"  [static]
```

URL base para las llamadas a la API.

7.9.2.2. UUIDs

```
ArrayList<String> com.example.usuario_upv.proyecto3a.Config.UUIDs = new ArrayList<>()  [static]
```

Lista de UUIDs utilizados en la aplicación.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Config.java

7.10. Referencia de la clase com.example.usuario_upv.proyecto3a.LandingPageActivity

Diagrama de herencia de com.example.usuario_upv.proyecto3a.LandingPageActivity

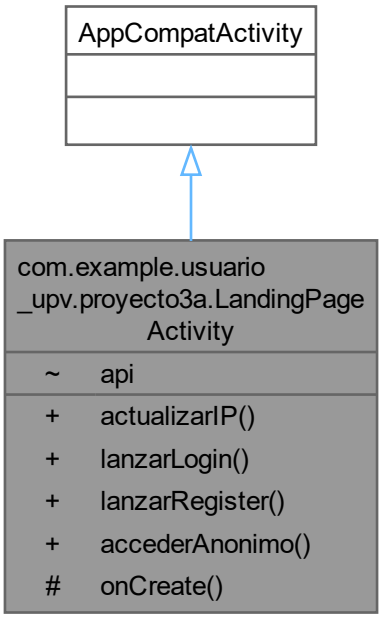
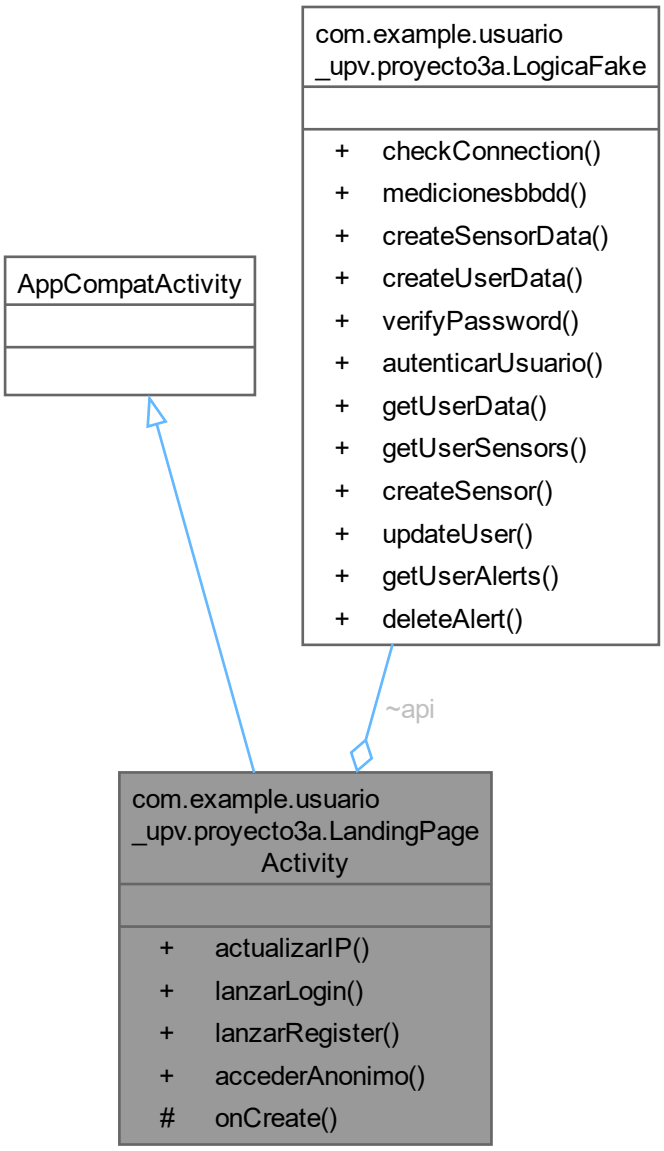


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.LandingPageActivity:



Métodos públicos

- void actualizarIP (View view)

  Actualiza la dirección IP del servidor para la conexión con la API.
- void lanzarLogin (View view)

  Lanza la actividad de inicio de sesión.
- void lanzarRegister (View view)

  Lanza la actividad de registro.
- void accederAnonimo (View view)

  Accede a la aplicación de forma anónima.

Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método llamado cuando se crea la actividad.

7.10.1. Documentación de funciones miembro

7.10.1.1. accederAnonimo()

```
void com.example.usuario_upv.proyecto3a.LandingPageActivity.accederAnonimo (
            View view)  [inline]
```

Accede a la aplicación de forma anónima.

Este método es llamado cuando el usuario presiona el botón para acceder de forma anónima.

Parámetros

view	La vista que desencadena este método, generalmente el botón de acceso anónimo.

7.10.1.2. actualizarIP()

```
void com.example.usuario_upv.proyecto3a.LandingPageActivity.actualizarIP (
            View view)  [inline]
```

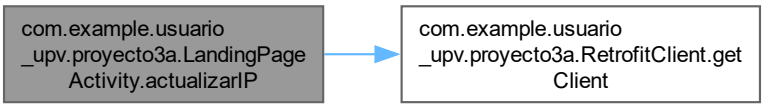
Actualiza la dirección IP del servidor para la conexión con la API.

Este método es llamado cuando el usuario introduce una IP y presiona el botón para actualizarla. Valida que la IP no esté vacía, construye la URL base para las peticiones HTTP mediante Retrofit y actualiza la instancia de la API. Muestra un mensaje en pantalla indicando la nueva IP del servidor.

Parámetros

view	La vista que desencadena este método, generalmente el botón de envío de la IP.

Gráfico de llamadas de esta función:



7.10.1.3. lanzarLogin()

```
void com.example.usuario_upv.proyecto3a.LandingPageActivity.lanzarLogin (
            View view)  [inline]
```

Lanza la actividad de inicio de sesión.

Este método es llamado cuando el usuario presiona el botón para iniciar sesión.

Parámetros

view	La vista que desencadena este método, generalmente el botón de inicio de sesión.

7.10.1.4. lanzarRegister()

```
void com.example.usuario_upv.proyecto3a.LandingPageActivity.lanzarRegister (
            View view)  [inline]
```

Lanza la actividad de registro.

Este método es llamado cuando el usuario presiona el botón para registrarse.

Parámetros

view	La vista que desencadena este método, generalmente el botón de registro.

7.10.1.5. onCreate()

```
void com.example.usuario_upv.proyecto3a.LandingPageActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método llamado cuando se crea la actividad.

Inicializa la actividad, configura la vista y verifica si el usuario ya ha iniciado sesión.

Parámetros

savedInstanceState	Estado guardado de la actividad.

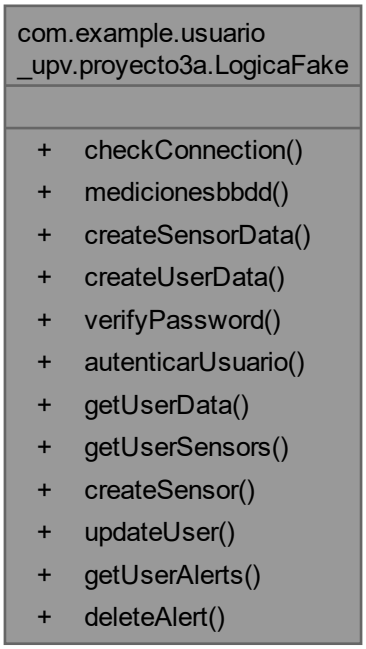
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/LandingPageActivity.java

7.11. Referencia de la interface com.example.usuario_upv.proyecto3a.LogicaFake

Interfaz para definir los endpoints de la API de sensores.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.LogicaFake:



Métodos públicos

- Call< Void > checkConnection ()

  Verifica la conexión con el servidor.
- Call< List< SensorData > > medicionesbbdd (@Path(value="fecha", encoded=true) String fecha)

  Obtiene las mediciones de la base de datos para una fecha específica.
- Call< Void > createSensorData (@Body SensorData sensorData)

  Inserta una medición de sensor en la base de datos.
- Call< Void > createUserData (@Body UserData userData)

  Crea un nuevo usuario en la base de datos.
- Call< Void > verifyPassword (@Body UserData userData)

  Verifica la contraseña de un usuario.
- Call< Void > autenticarUsuario (@Path(value="token", encoded=true) String token)

  Autentica un usuario mediante un token.
- Call< User > getUserData (@Path(value="email", encoded=true) String email, @Path("password") String password)

  Obtiene los datos de un usuario mediante su email y contraseña.
- Call< ResponseBody > getUserSensors (@Path(value="email", encoded=true) String email)

  Obtiene los sensores de un usuario mediante su email.
- Call< Void > createSensor (@Body Sensor sensor)

  Crea un nuevo sensor en la base de datos.
- Call< Void > updateUser (@Body UserData userData)

  Actualiza la información de un usuario.

- Call< List< AlertaData > > getUserAlerts (@Path(value="email", encoded=true) String email)

  Obtiene las alertas de un usuario mediante su email.
- Call< ResponseBody > deleteAlert (@Path("email") String email, @Path("alertaId") int alertaId)

  Elimina una alerta específica de un usuario.

7.11.1. Descripción detallada

Interfaz para definir los endpoints de la API de sensores.

Esta interfaz contiene métodos para interactuar con el servidor que gestiona los datos de los sensores y los usuarios.

7.11.2. Documentación de funciones miembro

7.11.2.1. autenticarUsuario()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.autenticarUsuario (
            @Path(value="token", encoded=true) String token)
```

Autentica un usuario mediante un token.

Parámetros

token	El token de autenticación.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.2. checkConnection()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.checkConnection ()
```

Verifica la conexión con el servidor.

Este método envía una solicitud GET al servidor para comprobar si está disponible.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.3. createSensor()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.createSensor (
            @Body Sensor sensor)
```

Crea un nuevo sensor en la base de datos.

Parámetros

sensor	El objeto Sensor que contiene la información del sensor.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.4. createSensorData()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.createSensorData (
            @Body SensorData sensorData)
```

Inserta una medición de sensor en la base de datos.

Este método envía una solicitud POST con la información de la medición en el cuerpo de la solicitud.

Parámetros

sensorData	El objeto SensorData que contiene la información de la medición.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.5. createUserData()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.createUserData (
            @Body UserData userData)
```

Crea un nuevo usuario en la base de datos.

Parámetros

userData	El objeto UserData que contiene la información del usuario.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.6. deleteAlert()

```
Call< ResponseBody > com.example.usuario_upv.proyecto3a.LogicaFake.deleteAlert (
            @Path("email") String email,
            @Path("alertaId") int alertaId)
```

Elimina una alerta específica de un usuario.

Parámetros

email	El email del usuario.
alerta↩ Id	El ID de la alerta a eliminar.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.7. getUserAlerts()

```
Call< List< AlertaData > > com.example.usuario_upv.proyecto3a.LogicaFake.getUserAlerts (
            @Path(value="email", encoded=true) String email)
```

Obtiene las alertas de un usuario mediante su email.

Parámetros

email	El email del usuario.

Devuelve

Un objeto Call que contiene una lista de datos de alertas.

7.11.2.8. getUserData()

```
Call< User > com.example.usuario_upv.proyecto3a.LogicaFake.getUserData (
            @Path(value="email", encoded=true) String email,
            @Path("password") String password)
```

Obtiene los datos de un usuario mediante su email y contraseña.

Parámetros

email	El email del usuario.
password	La contraseña del usuario.

Devuelve

Un objeto Call que contiene los datos del usuario.

7.11.2.9. getUserSensors()

```
Call< ResponseBody > com.example.usuario_upv.proyecto3a.LogicaFake.getUserSensors (
            @Path(value="email", encoded=true) String email)
```

Obtiene los sensores de un usuario mediante su email.

Parámetros

email	El email del usuario.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.10. medicionesbbdd()

```
Call< List< SensorData > > com.example.usuario_upv.proyecto3a.LogicaFake.medicionesbbdd (
            @Path(value="fecha", encoded=true) String fecha)
```

Obtiene las mediciones de la base de datos para una fecha específica.

Parámetros

fecha	Fecha para la cual se desean obtener las mediciones.

Devuelve

Un objeto Call que contiene una lista de datos de sensores.

7.11.2.11. updateUser()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.updateUser (
            @Body UserData userData)
```

Actualiza la información de un usuario.

Parámetros

userData	El objeto UserData que contiene la información del usuario.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

7.11.2.12. verifyPassword()

```
Call< Void > com.example.usuario_upv.proyecto3a.LogicaFake.verifyPassword (
            @Body UserData userData)
```

Verifica la contraseña de un usuario.

Parámetros

userData	El objeto UserData que contiene la información del usuario.

Devuelve

Un objeto Call que contiene la respuesta de la solicitud.

La documentación de esta interface está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/LogicaFake.java

7.12. Referencia de la clase com.example.usuario_upv.proyecto3a.LoginActivity

Clase que implementa la actividad de inicio de sesión.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.LoginActivity

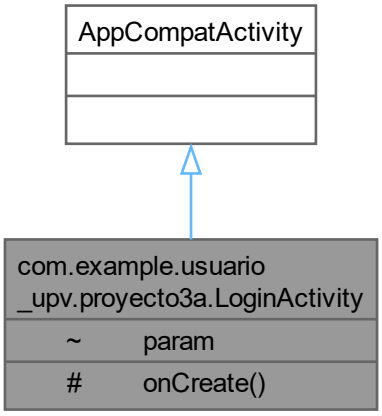
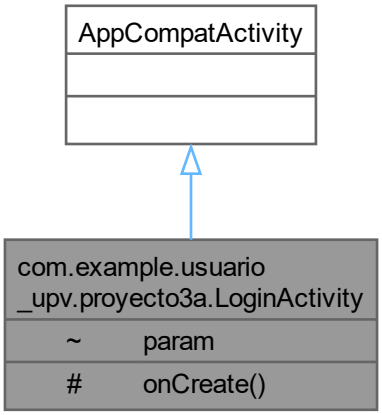


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.LoginActivity:



Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método que se llama al crear la actividad.

7.12.1. Descripción detallada

Clase que implementa la actividad de inicio de sesión.

La actividad permite a los usuarios iniciar sesión, registrarse y manejar la autenticación mediante un token. Utiliza Retrofit para realizar las solicitudes a la API.

7.12.2. Documentación de funciones miembro

7.12.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.LoginActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método que se llama al crear la actividad.

Inicializa los componentes de la interfaz y configura Retrofit para las solicitudes a la API.

Parámetros

savedInstanceState	Estado guardado de la actividad.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/LoginActivity.java

7.13. Referencia de la clase com.example.usuario_upv.proyecto3a.MainActivity

MainActivity de la aplicación.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.MainActivity

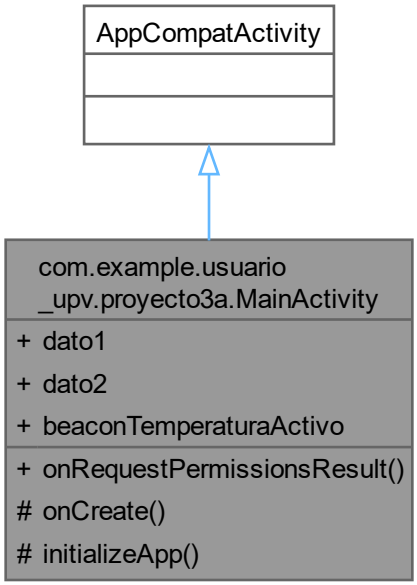
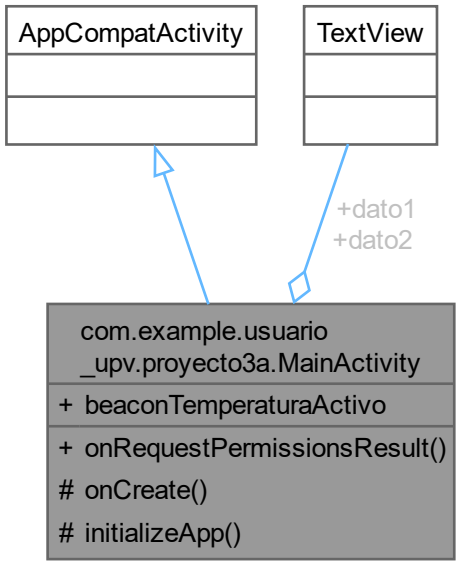


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.MainActivity:



Clases

- class MiPagerAdapter

  Adaptador para gestionar las pestañas de la aplicación.

Métodos públicos

- void onRequestPermissionsResult (int requestCode, String[ ] permissions, int[ ] grantResults)

  Maneja el resultado de las solicitudes de permisos.

Atributos públicos

- TextView dato1

  TextView para mostrar el primer dato.
- TextView dato2

  TextView para mostrar el segundo dato.
- boolean beaconTemperaturaActivo = false

  Estado que indica si el beacon de temperatura está activo.

Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método llamado cuando se crea la actividad.
- void initializeApp ()

  Inicializa la aplicación.

7.13.1. Descripción detallada

MainActivity de la aplicación.

Esta clase es la actividad principal de la aplicación que gestiona la interfaz de usuario y las interacciones con los sensores a través de Bluetooth LE.

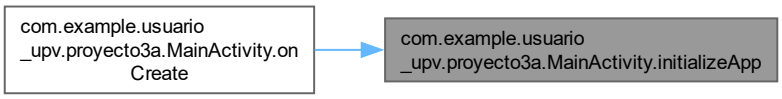
7.13.2. Documentación de funciones miembro

7.13.2.1. initializeApp()

```
void com.example.usuario_upv.proyecto3a.MainActivity.initializeApp ()  [inline], [protected]
```

Inicializa la aplicación.

Configura las pestañas y su comportamiento, así como la interfaz de usuario. Gráfico de llamadas a esta función:



7.13.2.2. onCreate()

```
void com.example.usuario_upv.proyecto3a.MainActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

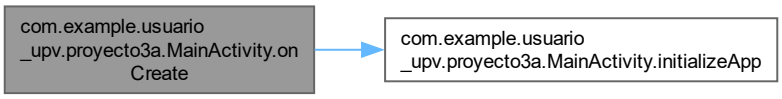
Método llamado cuando se crea la actividad.

Este método inicializa la actividad, configurando el layout y asignando las referencias a los elementos de la interfaz de usuario, como los TextView, ImageView, EditText y el contenedor de los beacons. También se inicializa el Bluetooth.

Parámetros

savedInstanceState	Estado previamente guardado de la actividad, si existe.

Gráfico de llamadas de esta función:



7.13.2.3. onRequestPermissionsResult()

```
void com.example.usuario_upv.proyecto3a.MainActivity.onRequestPermissionsResult (
            int requestCode,
            String[] permissions,
            int[] grantResults)  [inline]
```

Maneja el resultado de las solicitudes de permisos.

Este método es llamado cuando el usuario responde a una solicitud de permisos. Verifica si los permisos solicitados han sido concedidos y registra el resultado.

Parámetros

requestCode	El código de la solicitud de permisos.
permissions	Un arreglo de permisos solicitados.
grantResults	Un arreglo de resultados correspondientes a cada permiso.

7.13.3. Documentación de datos miembro

7.13.3.1. beaconTemperaturaActivo

```
boolean com.example.usuario_upv.proyecto3a.MainActivity.beaconTemperaturaActivo = false
```

Estado que indica si el beacon de temperatura está activo.

7.13.3.2. dato1

```
TextView com.example.usuario_upv.proyecto3a.MainActivity.dato1
```

TextView para mostrar el primer dato.

7.13.3.3. dato2

```
TextView com.example.usuario_upv.proyecto3a.MainActivity.dato2
```

TextView para mostrar el segundo dato.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/MainActivity.java

7.14. Referencia de la clase com.example.usuario_upv.proyecto3a.↩ MainActivity.MiPagerAdapter

Adaptador para gestionar las pestañas de la aplicación.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter

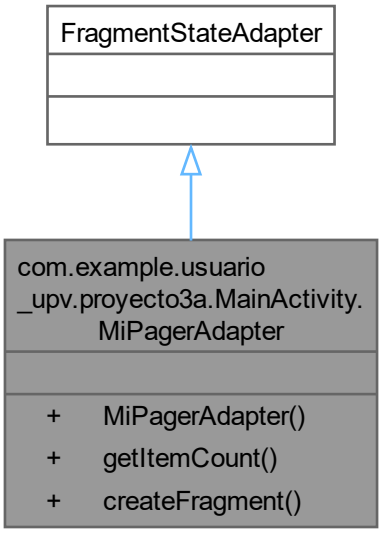
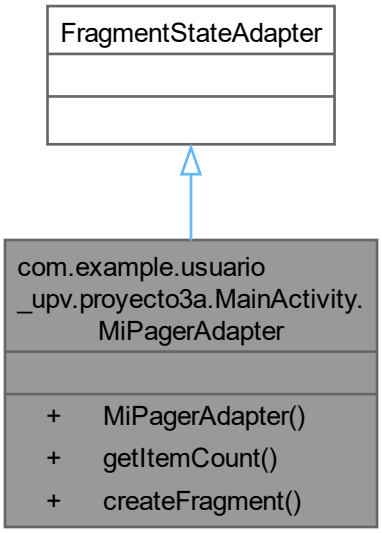


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter:



Métodos públicos

- MiPagerAdapter (FragmentActivity activity)

  Constructor del adaptador.
- int getItemCount ()

  Obtiene el número de pestañas.
- Fragment createFragment (int position)

  Crea el fragmento correspondiente a la posición dada.

7.14.1. Descripción detallada

Adaptador para gestionar las pestañas de la aplicación.

7.14.2. Documentación de constructores y destructores

7.14.2.1. MiPagerAdapter()

```
com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter.MiPagerAdapter (
            FragmentActivity activity)  [inline]
```

Constructor del adaptador.

Parámetros

activity	Actividad que contiene el adaptador.

7.14.3. Documentación de funciones miembro

7.14.3.1. createFragment()

```
Fragment com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter.createFragment (
            int position)  [inline]
```

Crea el fragmento correspondiente a la posición dada.

Parámetros

position	Posición de la pestaña.

Devuelve

Fragmento correspondiente a la posición.

7.14.3.2. getItemCount()

```
int com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter.getItemCount ()  [inline]
```

Obtiene el número de pestañas.

Devuelve

Número de pestañas.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/MainActivity.java

7.15. Referencia de la clase com.example.usuario_upv.proyecto3a.NotificationReceiver

Diagrama de herencia de com.example.usuario_upv.proyecto3a.NotificationReceiver

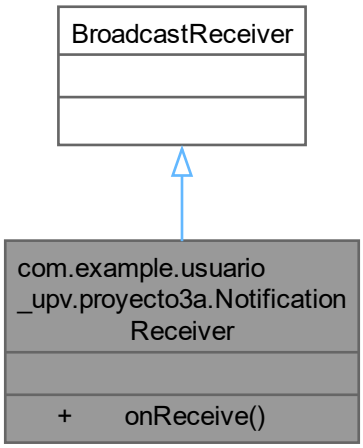
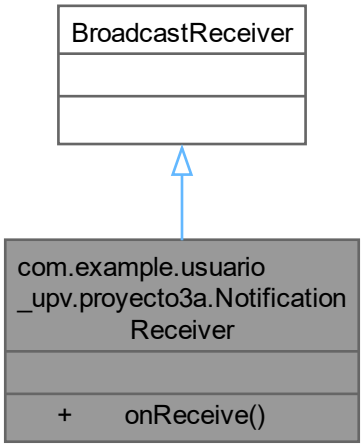


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.NotificationReceiver:



Métodos públicos

- void onReceive (Context context, Intent intent)

7.15.1. Descripción detallada

NotificationReceiver es un BroadcastReceiver que maneja la recepción de notificaciones eliminadas.

7.15.2. Documentación de funciones miembro

7.15.2.1. onReceive()

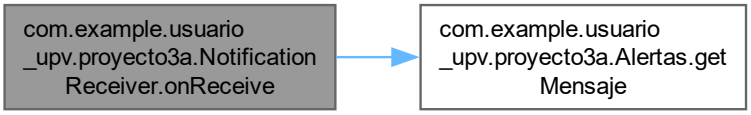
```
void com.example.usuario_upv.proyecto3a.NotificationReceiver.onReceive (
            Context context,
            Intent intent)  [inline]
```

Método llamado cuando se recibe una transmisión.

Parámetros

context	El contexto en el que se está ejecutando el receptor.
intent	El Intent que contiene la acción de la transmisión.

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/NotificationReceiver.java

7.16. Referencia de la interface com.example.usuario_upv.proyecto3a.↩ AlertasAdapter.OnAlertasListener

Interfaz para manejar eventos de interacción con las alertas.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener

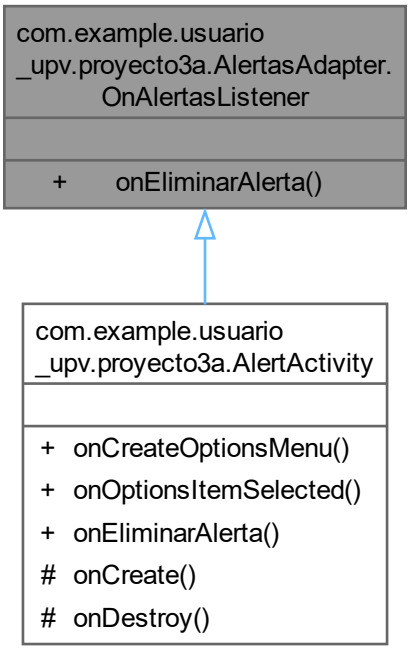
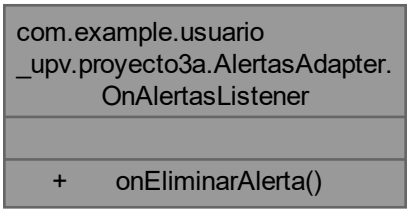


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener:



Métodos públicos

- void onEliminarAlerta (int position)
  Maneja la eliminación de una alerta.

7.16.1. Descripción detallada

Interfaz para manejar eventos de interacción con las alertas.

Define un método para manejar la eliminación de alertas por posición.

7.16.2. Documentación de funciones miembro

7.16.2.1. onEliminarAlerta()

```
void com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener.onEliminarAlerta (
            int position)
```

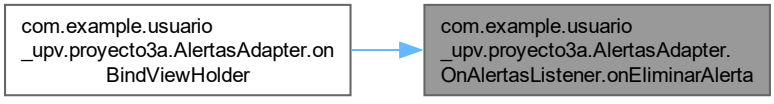
Maneja la eliminación de una alerta.

Parámetros

position	Posición de la alerta a eliminar en la lista.

Implementado en com.example.usuario_upv.proyecto3a.AlertActivity.

Gráfico de llamadas a esta función:



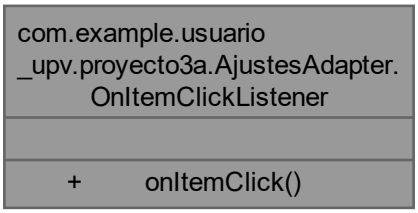
La documentación de esta interface está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AlertasAdapter.java

7.17. Referencia de la interface com.example.usuario_upv.proyecto3a.↩ AjustesAdapter.OnItemClickListener

Interfaz para manejar eventos de clic en los elementos.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.AjustesAdapter.OnItemClickListener:



Métodos públicos

- void onItemClick (int position)

  Manejador para el clic en un elemento.

7.17.1. Descripción detallada

Interfaz para manejar eventos de clic en los elementos.

7.17.2. Documentación de funciones miembro

7.17.2.1. onItemClick()

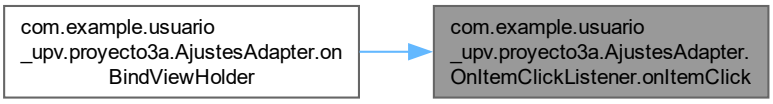
```
void com.example.usuario_upv.proyecto3a.AjustesAdapter.OnItemClickListener.onItemClick (
            int position)
```

Manejador para el clic en un elemento.

Parámetros

position	Posición del elemento clicado en la lista.

Gráfico de llamadas a esta función:



La documentación de esta interface está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/AjustesAdapter.java

7.18. Referencia de la clase com.example.usuario_upv.proyecto3a.Privacidad

Actividad para la política de privacidad.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.Privacidad

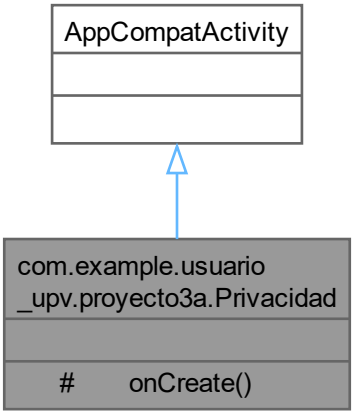
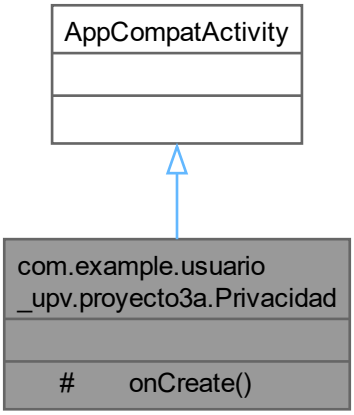


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Privacidad:



Métodos protegidos

- void onCreate (@Nullable Bundle savedInstanceState)

  Método llamado cuando se crea la actividad.

7.18.1. Descripción detallada

Actividad para la política de privacidad.

Esta actividad carga y muestra el contenido de la política de privacidad.

7.18.2. Documentación de funciones miembro

7.18.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.Privacidad.onCreate (
            @Nullable Bundle savedInstanceState)  [inline], [protected]
```

Método llamado cuando se crea la actividad.

Este método configura la vista de la actividad con el layout correspondiente.

Parámetros

savedInstanceState	Si la actividad se está re-inicializando después de haber sido previamente cerrada, este Bundle contiene los datos más recientes suministrados en onSaveInstanceState(Bundle). De lo contrario, está nulo.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Privacidad.java

7.19. Referencia de la clase com.example.usuario_upv.proyecto3a.RegisterActivity

Actividad para el registro de usuarios.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.RegisterActivity

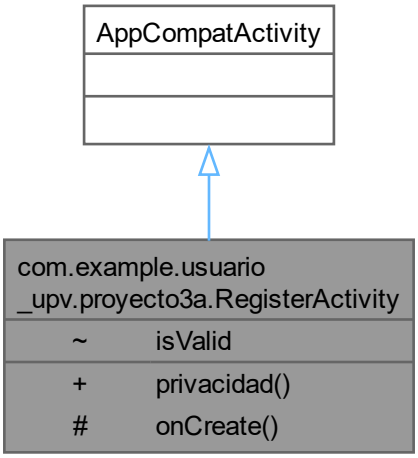
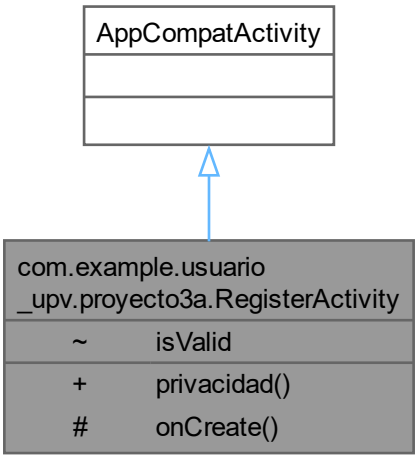


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.RegisterActivity:



Métodos públicos

- void privacidad (View v)

  Muestra la política de privacidad.

Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método llamado cuando se crea la actividad.

7.19.1. Descripción detallada

Actividad para el registro de usuarios.

Esta actividad permite a los usuarios registrarse proporcionando su correo electrónico, nombre y contraseña. También incluye autenticación biométrica y validación de requisitos de contraseña.

7.19.2. Documentación de funciones miembro

7.19.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.RegisterActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método llamado cuando se crea la actividad.

Este método configura la vista de la actividad con el layout correspondiente y inicializa los componentes necesarios.

Parámetros

savedInstanceState	Si la actividad se está re-inicializando después de haber sido previamente cerrada, este Bundle contiene los datos más recientes suministrados en onSaveInstanceState(Bundle). De lo contrario, está nulo.

Verifica si el correo electrónico es válido.

Parámetros

email	El correo electrónico a verificar.

Devuelve

true si el correo electrónico es válido, false en caso contrario.

7.19.2.2. privacidad()

```
void com.example.usuario_upv.proyecto3a.RegisterActivity.privacidad (
            View v)  [inline]
```

Muestra la política de privacidad.

Este método inicia la actividad que muestra la política de privacidad.

Parámetros

v	La vista que desencadena este método.

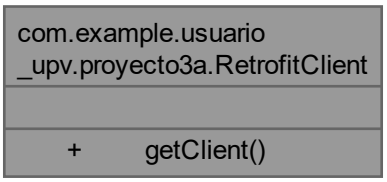
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/RegisterActivity.java

7.20. Referencia de la clase com.example.usuario_upv.proyecto3a.RetrofitClient

Clase para gestionar la instancia de Retrofit.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.RetrofitClient:



Métodos públicos estáticos

- static Retrofit getClient (String baseUrl)
  Obtiene la instancia de Retrofit.

7.20.1. Descripción detallada

Clase para gestionar la instancia de Retrofit.

Esta clase proporciona un método para obtener una instancia de Retrofit configurada con una URL base y un convertidor de JSON a objetos Java.

7.20.2. Documentación de funciones miembro

7.20.2.1. getClient()

```
static Retrofit com.example.usuario_upv.proyecto3a.RetrofitClient.getClient (
            String baseUrl)  [inline], [static]
```

Obtiene la instancia de Retrofit.

Este método verifica si la instancia de Retrofit ya existe o si la URL base ha cambiado. Si es así, crea una nueva instancia de Retrofit.

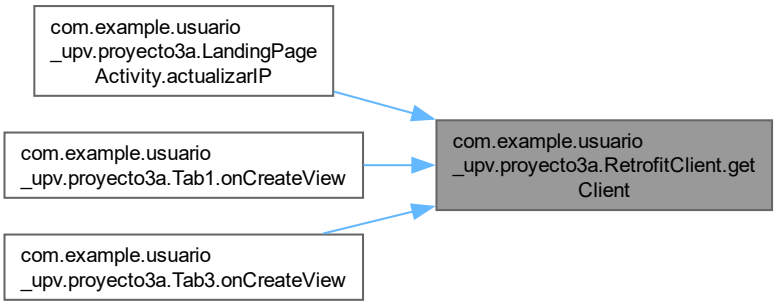
Parámetros

baseUrl	La URL base que se utilizará para las solicitudes de la API.

Devuelve

La instancia de Retrofit configurada.

Gráfico de llamadas a esta función:



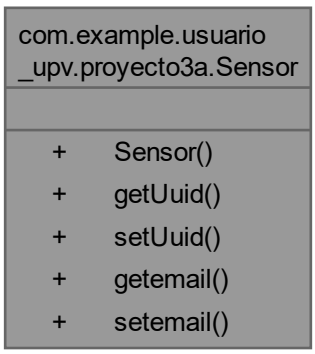
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/RetrofitClient.java

7.21. Referencia de la clase com.example.usuario_upv.proyecto3a.Sensor

Clase que representa un sensor con un UUID y un email asociado.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Sensor:



Métodos públicos

- Sensor (String uuid, String email)

  Constructor de la clase Sensor.
- String getUuid ()

  Obtiene el UUID del sensor.
- void setUuid (String uuid)

  Establece el UUID del sensor.
- String getemail ()

  Obtiene el email asociado al sensor.
- void setemail (String email)

  Establece el email asociado al sensor.

7.21.1. Descripción detallada

Clase que representa un sensor con un UUID y un email asociado.

Esta clase proporciona métodos para obtener y establecer el UUID y el email del sensor. Se utiliza para identificar y gestionar sensores en la aplicación.

7.21.2. Documentación de constructores y destructores

7.21.2.1. Sensor()

```
com.example.usuario_upv.proyecto3a.Sensor.Sensor (
            String uuid,
            String email)  [inline]
```

Constructor de la clase Sensor.

Parámetros

uuid	UUID del sensor.
email	Email asociado al sensor.

7.21.3. Documentación de funciones miembro

7.21.3.1. getemail()

```
String com.example.usuario_upv.proyecto3a.Sensor.getemail ()  [inline]
```

Obtiene el email asociado al sensor.

Devuelve

El email asociado al sensor.

7.21.3.2. getUuid()

```
String com.example.usuario_upv.proyecto3a.Sensor.getUuid ()  [inline]
```

Obtiene el UUID del sensor.

Devuelve

El UUID del sensor.

7.21.3.3. setemail()

```
void com.example.usuario_upv.proyecto3a.Sensor.setemail (
            String email)  [inline]
```

Establece el email asociado al sensor.

Parámetros

email	El nuevo email asociado al sensor.

7.21.3.4. setUuid()

```
void com.example.usuario_upv.proyecto3a.Sensor.setUuid (
            String uuid)  [inline]
```

Establece el UUID del sensor.

Parámetros

uuid	El nuevo UUID del sensor.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Sensor.java

7.22. Referencia de la clase com.example.usuario_upv.proyecto3a.SensorAdapter

Adaptador para mostrar una lista de sensores en un RecyclerView.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.SensorAdapter

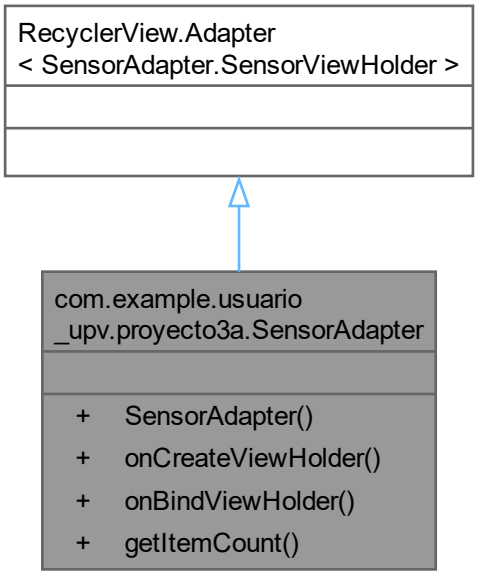
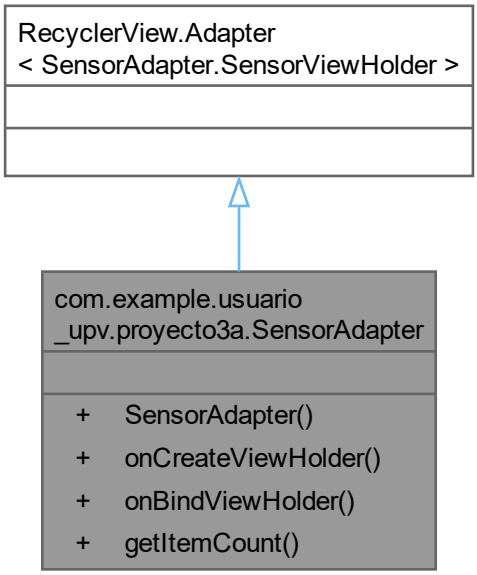


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.SensorAdapter:



Clases

- class **SensorViewHolder**

  ViewHolder para los elementos del sensor.

Métodos públicos

- SensorAdapter (List< String > sensorList)

  Constructor de la clase SensorAdapter.
- SensorViewHolder onCreateViewHolder (ViewGroup parent, int viewType)

  Crea nuevos ViewHolder cuando no hay suficientes ViewHolder existentes.
- void onBindViewHolder (SensorViewHolder holder, int position)

  Vincula los datos del sensor a un ViewHolder.
- int getItemCount ()

  Obtiene el número de sensores en la lista.

7.22.1. Descripción detallada

Adaptador para mostrar una lista de sensores en un RecyclerView.

Esta clase extiende RecyclerView.Adapter y proporciona un adaptador personalizado para mostrar una lista de sensores en un RecyclerView.

7.22.2. Documentación de constructores y destructores

7.22.2.1. SensorAdapter()

```
com.example.usuario_upv.proyecto3a.SensorAdapter.SensorAdapter (
            List< String > sensorList)  [inline]
```

Constructor de la clase SensorAdapter.

Parámetros

sensorList	Lista de sensores a mostrar.

7.22.3. Documentación de funciones miembro

7.22.3.1. getItemCount()

```
int com.example.usuario_upv.proyecto3a.SensorAdapter.getItemCount ()  [inline]
```

Obtiene el número de sensores en la lista.

Devuelve

El número de sensores en la lista.

7.22.3.2. onBindViewHolder()

```
void com.example.usuario_upv.proyecto3a.SensorAdapter.onBindViewHolder (
            SensorViewHolder holder,
            int position)  [inline]
```

Vincula los datos del sensor a un ViewHolder.

Parámetros

holder	El ViewHolder que debe ser actualizado.
position	La posición del sensor en la lista.

7.22.3.3. onCreateViewHolder()

```
SensorViewHolder com.example.usuario_upv.proyecto3a.SensorAdapter.onCreateViewHolder (
            ViewGroup parent,
            int viewType)  [inline]
```

Crea nuevos ViewHolder cuando no hay suficientes ViewHolder existentes.

Parámetros

parent	El ViewGroup al que se añadirá el nuevo View.
viewType	El tipo de vista del nuevo View.

Devuelve

Un nuevo SensorViewHolder.

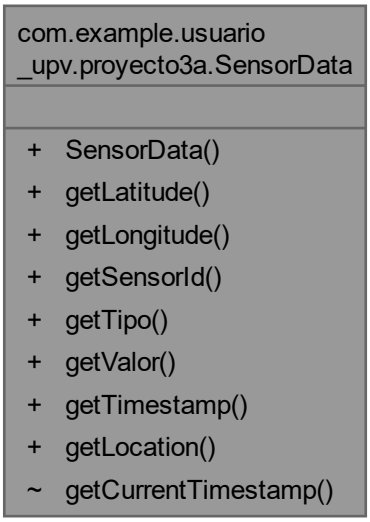
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/SensorAdapter.java

7.23. Referencia de la clase com.example.usuario_upv.proyecto3a.SensorData

Clase que representa los datos de un sensor.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.SensorData:



Clases

- class **Location**

  Clase que representa la ubicación de un sensor.

Métodos públicos

- SensorData (String sensorId, float valor, int tipo, Location location, String timestamp)

  Constructor de la clase SensorData.
- double getLatitude ()

  Obtiene la latitud de la ubicación del sensor.
- double getLongitude ()

  Obtiene la longitud de la ubicación del sensor.
- String getSensorId ()

  Obtiene el ID del sensor.
- int getTipo ()

  Obtiene el tipo de sensor.
- float getValor ()

  Obtiene el valor de la medición del sensor.
- String getTimestamp ()

  Obtiene la marca de tiempo de la medición.
- Location getLocation ()

  Obtiene la ubicación del sensor.

7.23.1. Descripción detallada

Clase que representa los datos de un sensor.

Esta clase se utiliza para almacenar la información de las mediciones de los sensores, incluyendo el sensor_id, el valor, la marca de tiempo, el ID del usuario asociado, y el tipo de sensor.

7.23.2. Documentación de constructores y destructores

7.23.2.1. SensorData()

```
com.example.usuario_upv.proyecto3a.SensorData.SensorData (
            String sensorId,
            float valor,
            int tipo,
            Location location,
            String timestamp)  [inline]
```

Constructor de la clase SensorData.

Parámetros

sensorId	ID del sensor.
valor	Valor de la medición del sensor.
tipo	Tipo de sensor.
location	Ubicación del sensor.
timestamp	Marca de tiempo de la medición.

7.23.3. Documentación de funciones miembro

7.23.3.1. getLatitude()

```
double com.example.usuario_upv.proyecto3a.SensorData.getLatitude ()  [inline]
```

Obtiene la latitud de la ubicación del sensor.

Devuelve

La latitud de la ubicación del sensor.

7.23.3.2. getLocation()

```
Location com.example.usuario_upv.proyecto3a.SensorData.getLocation ()  [inline]
```

Obtiene la ubicación del sensor.

Devuelve

La ubicación del sensor.

7.23.3.3. getLongitude()

```
double com.example.usuario_upv.proyecto3a.SensorData.getLongitude ()  [inline]
```

Obtiene la longitud de la ubicación del sensor.

Devuelve

La longitud de la ubicación del sensor.

7.23.3.4. getSensorId()

```
String com.example.usuario_upv.proyecto3a.SensorData.getSensorId ()  [inline]
```

Obtiene el ID del sensor.

Devuelve

El ID del sensor.

7.23.3.5. getTimestamp()

```
String com.example.usuario_upv.proyecto3a.SensorData.getTimestamp ()  [inline]
```

Obtiene la marca de tiempo de la medición.

Devuelve

La marca de tiempo de la medición.

7.23.3.6. getTipo()

```
int com.example.usuario_upv.proyecto3a.SensorData.getTipo ()  [inline]
```

Obtiene el tipo de sensor.

Devuelve

El tipo de sensor.

7.23.3.7. getValor()

```
float com.example.usuario_upv.proyecto3a.SensorData.getValor ()  [inline]
```

Obtiene el valor de la medición del sensor.

Devuelve

El valor de la medición del sensor.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/SensorData.java

7.24. Referencia de la clase com.example.usuario_upv.proyecto3a.SplashActivity

Diagrama de herencia de com.example.usuario_upv.proyecto3a.SplashActivity

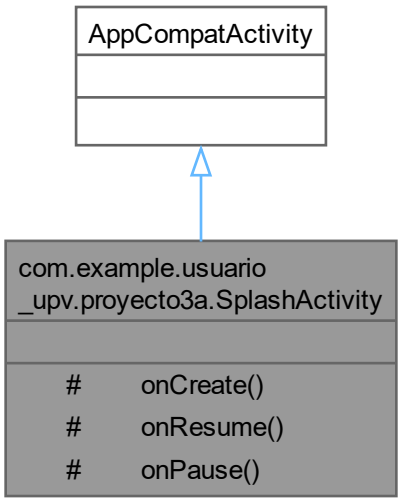
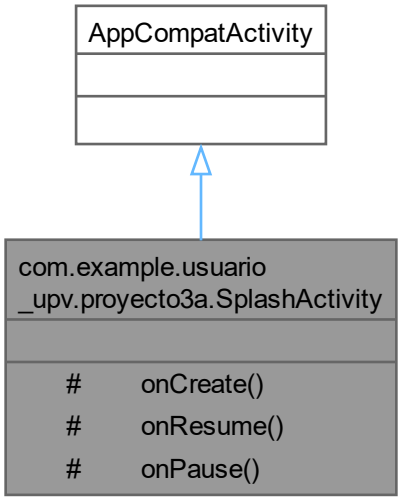


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.SplashActivity:



Métodos protegidos

- void onCreate (Bundle savedInstanceState)

  Método llamado cuando se crea la actividad.
- void onResume ()

  Método llamado cuando la actividad se reanuda.
- void onPause ()

  Método llamado cuando la actividad se pausa.

7.24.1. Documentación de funciones miembro

7.24.1.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.SplashActivity.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método llamado cuando se crea la actividad.

Parámetros

savedInstanceState	Estado previamente guardado de la actividad.

7.24.1.2. onPause()

```
void com.example.usuario_upv.proyecto3a.SplashActivity.onPause ()  [inline], [protected]
```

Método llamado cuando la actividad se pausa.

7.24.1.3. onResume()

```
void com.example.usuario_upv.proyecto3a.SplashActivity.onResume ()  [inline], [protected]
```

Método llamado cuando la actividad se reanuda.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/SplashActivity.java

7.25. Referencia de la clase com.example.usuario_upv.proyecto3a.Tab1

Fragmento que muestra un mapa con datos de sensores.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.Tab1

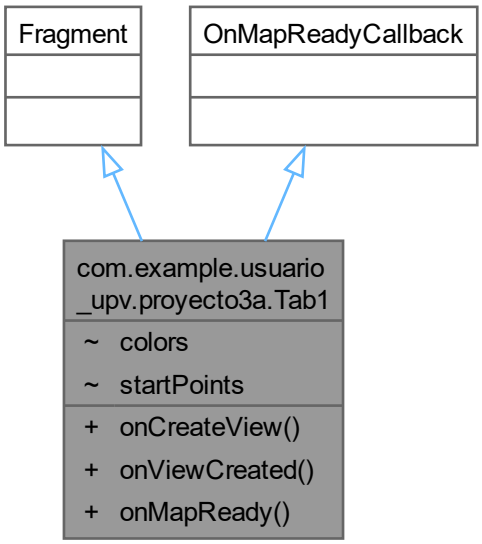
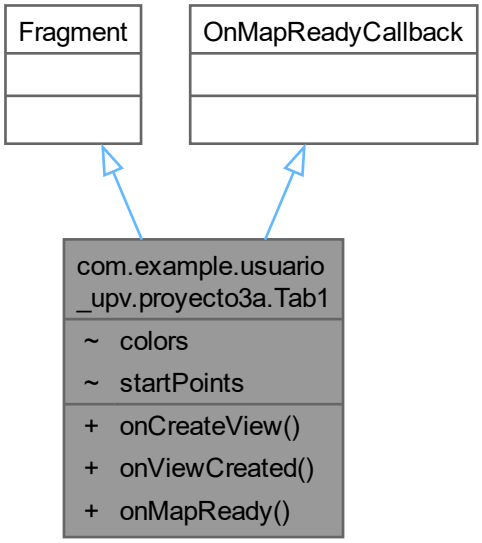


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Tab1:



Métodos públicos

- View onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)

  Método llamado cuando se crea la vista del fragmento.
- void onViewCreated (View view, Bundle savedInstanceState)

  Método llamado cuando la vista del fragmento ha sido creada.
- void onMapReady (GoogleMap googleMap)

  Método llamado cuando el mapa está listo para ser usado.

7.25.1. Descripción detallada

Fragmento que muestra un mapa con datos de sensores.

Este fragmento contiene un mapa que puede mostrar datos de sensores en forma de mapa de calor o marcadores. Permite seleccionar la fecha y el tipo de medición para visualizar los datos correspondientes.

7.25.2. Documentación de funciones miembro

7.25.2.1. onCreateView()

```
View com.example.usuario_upv.proyecto3a.Tab1.onCreateView (
            LayoutInflater inflater,
            ViewGroup container,
            Bundle savedInstanceState)  [inline]
```

Método llamado cuando se crea la vista del fragmento.

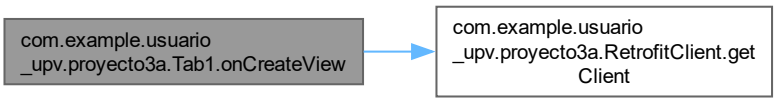
Parámetros

inflater	El LayoutInflater.
container	El contenedor ViewGroup.
savedInstanceState	El estado previamente guardado.

Devuelve

La vista creada.

Gráfico de llamadas de esta función:



7.25.2.2. onMapReady()

```
void com.example.usuario_upv.proyecto3a.Tab1.onMapReady (
            GoogleMap googleMap)  [inline]
```

Método llamado cuando el mapa está listo para ser usado.

Parámetros

googleMap	La instancia de GoogleMap.

7.25.2.3. onViewCreated()

```
void com.example.usuario_upv.proyecto3a.Tab1.onViewCreated (
            View view,
            Bundle savedInstanceState)  [inline]
```

Método llamado cuando la vista del fragmento ha sido creada.

Parámetros

view	La vista creada.
savedInstanceState	El estado previamente guardado.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Tab1.java

7.26. Referencia de la clase com.example.usuario_upv.proyecto3a.Tab3

Fragmento que representa la tercera pestaña de la aplicación.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.Tab3

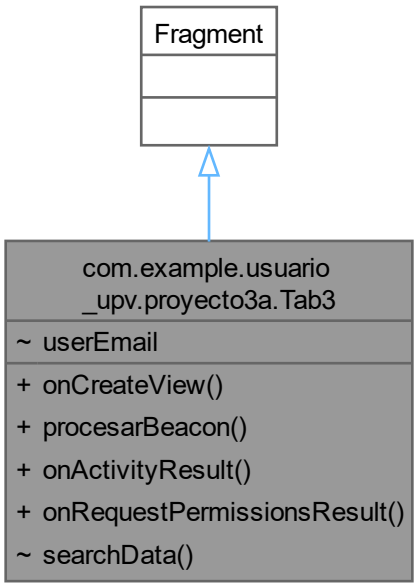
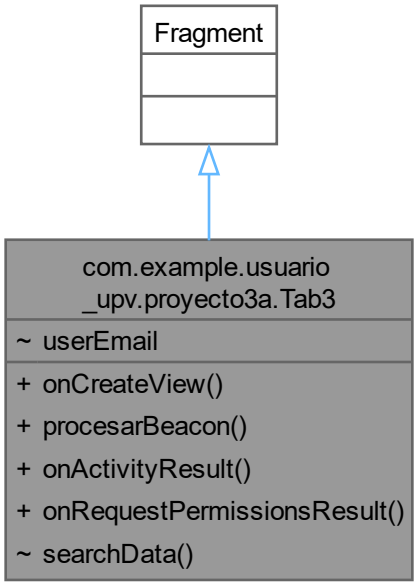


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Tab3:



Métodos públicos

- View onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)

  Método llamado cuando se crea la vista del fragmento.
- int procesarBeacon (int major, int minor)

  Procesa la información del beacon detectado.
- void onActivityResult (int requestCode, int resultCode, Intent data)

  Maneja el resultado de la actividad de escaneo de códigos QR.
- void onRequestPermissionsResult (int requestCode, @NonNull String[ ] permissions, @NonNull int[ ] grant↩ Results)

  Maneja el resultado de las solicitudes de permisos.

7.26.1. Descripción detallada

Fragmento que representa la tercera pestaña de la aplicación.

Este fragmento se encarga de gestionar la interfaz de usuario y las interacciones relacionadas con los sensores, incluyendo la inicialización de Bluetooth, escaneo de beacons y procesamiento de datos de sensores.

7.26.2. Documentación de funciones miembro

7.26.2.1. onActivityResult()

```
void com.example.usuario_upv.proyecto3a.Tab3.onActivityResult (
            int requestCode,
            int resultCode,
            Intent data)  [inline]
```

Maneja el resultado de la actividad de escaneo de códigos QR.

Este método es llamado cuando se obtiene el resultado de la actividad de escaneo de códigos QR. Procesa el contenido del código QR escaneado y agrega el sensor correspondiente.

Parámetros

requestCode	El código de solicitud de la actividad.
resultCode	El código de resultado de la actividad.
data	Los datos de la actividad.

7.26.2.2. onCreateView()

```
View com.example.usuario_upv.proyecto3a.Tab3.onCreateView (
            LayoutInflater inflater,
            ViewGroup container,
            Bundle savedInstanceState)  [inline]
```

Método llamado cuando se crea la vista del fragmento.

Este método inicializa la vista del fragmento, configurando el layout y asignando las referencias a los elementos de la interfaz de usuario, como los TextView, ImageView, y el botón flotante. También se inicializa el Bluetooth y se recupera el email del usuario.

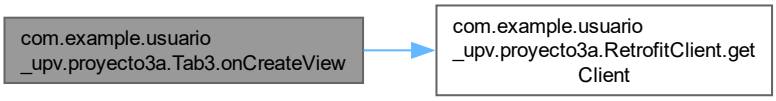
Parámetros

inflater	El LayoutInflater utilizado para inflar la vista del fragmento.
container	El contenedor donde se infla la vista del fragmento.
savedInstanceState	Estado previamente guardado del fragmento, si existe.

Devuelve

La vista inflada del fragmento.

Gráfico de llamadas de esta función:



7.26.2.3. onRequestPermissionsResult()

```
void com.example.usuario_upv.proyecto3a.Tab3.onRequestPermissionsResult (
            int requestCode,
            @NonNull String[] permissions,
            @NonNull int[] grantResults)  [inline]
```

Maneja el resultado de las solicitudes de permisos.

Este método es llamado cuando el usuario responde a una solicitud de permisos. Verifica si los permisos solicitados han sido concedidos y registra el resultado.

Parámetros

requestCode	El código de la solicitud de permisos.
permissions	Un arreglo de permisos solicitados.
grantResults	Un arreglo de resultados correspondientes a cada permiso.

7.26.2.4. procesarBeacon()

```
int com.example.usuario_upv.proyecto3a.Tab3.procesarBeacon (
            int major,
            int minor)  [inline]
```

Procesa la información del beacon detectado.

Este método determina el tipo de medición a partir del valor 'major' del beacon y registra el valor 'minor'. Los tipos de medición reconocidos son ozono y Temperatura, identificados por sus respectivos códigos.

Parámetros

major	El valor 'major' del beacon, que contiene el tipo de medición y un contador.
minor	El valor 'minor' del beacon, que representa el dato medido (por ejemplo, ozono o temperatura).

Devuelve

Un entero que representa el tipo de medición:

- 1 si el dato es de ozono
- 2 si el dato es de temperatura
- 0 si el tipo de dato no es reconocido.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Tab3.java

7.27. Referencia de la clase com.example.usuario_upv.proyecto3a.Tab4

Fragmento que representa la cuarta pestaña de la aplicación.

Diagrama de herencia de com.example.usuario_upv.proyecto3a.Tab4

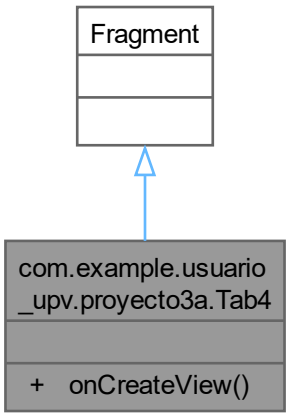
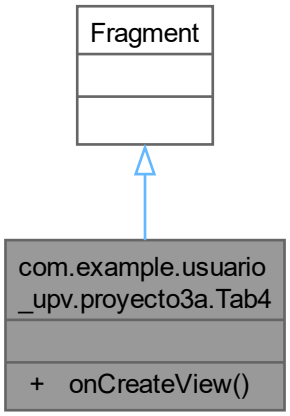


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Tab4:



Métodos públicos

- View onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)

  Método que se llama para crear y devolver la jerarquía de vistas asociada con el fragmento.

7.27.1. Descripción detallada

Fragmento que representa la cuarta pestaña de la aplicación.

Este fragmento se encarga de gestionar la interfaz de usuario y las interacciones relacionadas con la configuración del usuario, notificaciones, información de la aplicación y el asistente.

7.27.2. Documentación de funciones miembro

7.27.2.1. onCreateView()

```
View com.example.usuario_upv.proyecto3a.Tab4.onCreateView (
            LayoutInflater inflater,
            ViewGroup container,
            Bundle savedInstanceState)  [inline]
```

Método que se llama para crear y devolver la jerarquía de vistas asociada con el fragmento.

Parámetros

inflater	El LayoutInflater que se puede usar para inflar cualquier vista en el fragmento.
container	Si no es nulo, es la vista principal a la que se adjuntará el fragmento.
savedInstanceState	Si no es nulo, este fragmento se está reconstruyendo a partir de un estado guardado anterior.

Devuelve

La vista para la interfaz de usuario del fragmento.

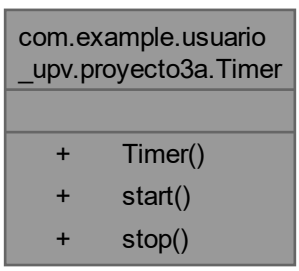
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Tab4.java

7.28. Referencia de la clase com.example.usuario_upv.proyecto3a.Timer

Clase que representa un temporizador.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Timer:



Métodos públicos

- Timer (long interval, Runnable runnable)

  Constructor de la clase Timer.
- void start ()

  Inicia el temporizador.
- void stop ()

  Detiene el temporizador.

7.28.1. Descripción detallada

Clase que representa un temporizador.

Esta clase permite ejecutar una tarea periódicamente en un intervalo especificado.

7.28.2. Documentación de constructores y destructores

7.28.2.1. Timer()

```
com.example.usuario_upv.proyecto3a.Timer.Timer (
            long interval,
            Runnable runnable)  [inline]
```

Constructor de la clase Timer.

Parámetros

interval	Intervalo de tiempo en milisegundos entre ejecuciones de la tarea.
runnable	Tarea a ejecutar periódicamente.

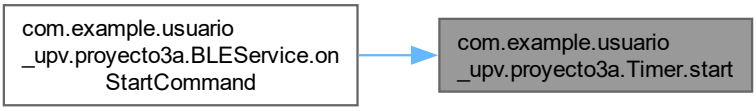
7.28.3. Documentación de funciones miembro

7.28.3.1. start()

```
void com.example.usuario_upv.proyecto3a.Timer.start ()  [inline]
```

Inicia el temporizador.

Si el temporizador no está en ejecución, comienza a ejecutar la tarea periódicamente. Gráfico de llamadas a esta función:

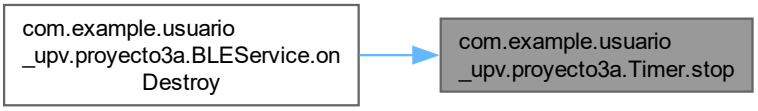


7.28.3.2. stop()

```
void com.example.usuario_upv.proyecto3a.Timer.stop ()  [inline]
```

Detiene el temporizador.

Si el temporizador está en ejecución, detiene la ejecución periódica de la tarea. Gráfico de llamadas a esta función:



La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Timer.java

7.29. Referencia de la clase com.example.usuario_upv.proyecto3a.TramaIBeacon

Clase que representa la trama de un beacon iBeacon.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.TramaIBeacon:

Métodos públicos

- byte[ ] getPrefijo ()

  Obtiene el prefijo del beacon.
- byte[ ] getUUID ()

  Obtiene el UUID del beacon.
- byte[ ] getMajor ()

  Obtiene el major del beacon.
- byte[ ] getMinor ()

  Obtiene el minor del beacon.
- byte getTxPower ()

  Obtiene el TxPower del beacon.
- byte[ ] getLosBytes ()

  Obtiene el array de bytes original recibido.
- byte[ ] getAdvFlags ()

  Obtiene las banderas de publicidad.
- byte[ ] getAdvHeader ()

  Obtiene el encabezado de publicidad.
- byte[ ] getCompanyID ()

  Obtiene el ID de la compañía.
- byte getiBeaconType ()

Obtiene el tipo de iBeacon.

- byte getiBeaconLength ()

  Obtiene la longitud del iBeacon.

- void setMajor (byte[ ] major)

  Establece el valor del major del beacon.

- void setMinor (byte[ ] minor)

  Establece el valor del minor del beacon.

- TramaIBeacon (byte[ ] bytes)

  Constructor que crea una instancia de TramaIBeacon.

7.29.1. Descripción detallada

Clase que representa la trama de un beacon iBeacon.

Esta clase se encarga de interpretar los datos de un beacon iBeacon a partir de un array de bytes. Extrae información relevante como el UUID, el major, el minor y el TxPower.

7.29.2. Documentación de constructores y destructores

7.29.2.1. TramaIBeacon()

```
com.example.usuario_upv.proyecto3a.TramaIBeacon.TramaIBeacon (
            byte[] bytes)  [inline]
```

Constructor que crea una instancia de TramaIBeacon.

Parámetros

bytes	Array de bytes que contiene los datos del beacon.

7.29.3. Documentación de funciones miembro

7.29.3.1. getAdvFlags()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getAdvFlags ()  [inline]
```

Obtiene las banderas de publicidad.

Devuelve

Array de bytes que representa las banderas de publicidad.

7.29.3.2. getAdvHeader()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getAdvHeader ()  [inline]
```

Obtiene el encabezado de publicidad.

Devuelve

Array de bytes que representa el encabezado de publicidad.

7.29.3.3. getCompanyID()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getCompanyID ()  [inline]
```

Obtiene el ID de la compañía.

Devuelve

Array de bytes que representa el ID de la compañía.

7.29.3.4. getiBeaconLength()

```
byte com.example.usuario_upv.proyecto3a.TramaIBeacon.getiBeaconLength ()  [inline]
```

Obtiene la longitud del iBeacon.

Devuelve

Byte que representa la longitud del iBeacon.

7.29.3.5. getiBeaconType()

```
byte com.example.usuario_upv.proyecto3a.TramaIBeacon.getiBeaconType ()  [inline]
```

Obtiene el tipo de iBeacon.

Devuelve

Byte que representa el tipo de iBeacon.

7.29.3.6. getLosBytes()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getLosBytes ()  [inline]
```

Obtiene el array de bytes original recibido.

Devuelve

Array de bytes original.

7.29.3.7. getMajor()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getMajor ()  [inline]
```

Obtiene el major del beacon.

Devuelve

Array de bytes que representa el major.

7.29.3.8. getMinor()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getMinor ()  [inline]
```

Obtiene el minor del beacon.

Devuelve

Array de bytes que representa el minor.

7.29.3.9. getPrefijo()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getPrefijo ()  [inline]
```

Obtiene el prefijo del beacon.

Devuelve

Array de bytes que representa el prefijo.

7.29.3.10. getTxPower()

```
byte com.example.usuario_upv.proyecto3a.TramaIBeacon.getTxPower ()  [inline]
```

Obtiene el TxPower del beacon.

Devuelve

Byte que representa el TxPower.

7.29.3.11. getUUID()

```
byte[] com.example.usuario_upv.proyecto3a.TramaIBeacon.getUUID ()  [inline]
```

Obtiene el UUID del beacon.

Devuelve

Array de bytes que representa el UUID.

7.29.3.12. setMajor()

```
void com.example.usuario_upv.proyecto3a.TramaIBeacon.setMajor (
            byte[] major)  [inline]
```

Establece el valor del major del beacon.

Parámetros

major	Array de bytes que representa el major.

7.29.3.13. setMinor()

```
void com.example.usuario_upv.proyecto3a.TramaIBeacon.setMinor (
            byte[] minor)  [inline]
```

Establece el valor del minor del beacon.

Parámetros

minor	Array de bytes que representa el minor.

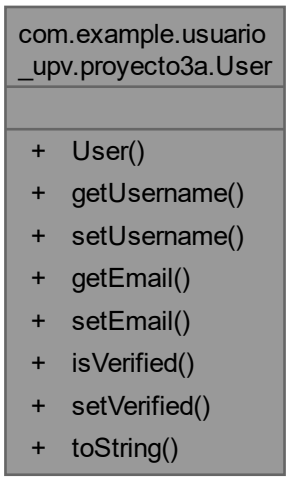
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/TramaIBeacon.java

7.30. Referencia de la clase com.example.usuario_upv.proyecto3a.User

Clase que representa un usuario.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.User:



Métodos públicos

- User (String username, String email, String password, int actividad_id, String verification_token, boolean is_verified)

  Constructor de la clase User.
- String getUsername ()

  Obtiene el nombre de usuario.
- void setUsername (String username)

  Establece el nombre de usuario.
- String getEmail ()

  Obtiene el correo electrónico del usuario.
- void setEmail (String email)

  Establece el correo electrónico del usuario.
- boolean isVerified ()

  Obtiene el estado de verificación del usuario.
- void setVerified (boolean verified)

  Establece el estado de verificación del usuario.
- String toString ()

  Sobrecarga el método toString para imprimir las propiedades del objeto.

7.30.1. Descripción detallada

Clase que representa un usuario.

Esta clase contiene información sobre un usuario, incluyendo su nombre de usuario, correo electrónico, contraseña, ID de actividad, token de verificación y estado de verificación.

7.30.2. Documentación de constructores y destructores

7.30.2.1. User()

```
com.example.usuario_upv.proyecto3a.User.User (
            String username,
            String email,
            String password,
            int actividad_id,
            String verification_token,
            boolean is_verified)  [inline]
```

Constructor de la clase User.

Parámetros

username	Nombre de usuario.
email	Correo electrónico del usuario.
password	Contraseña del usuario.
actividad_id	ID de la actividad asociada al usuario.
verification_token	Token de verificación del usuario.
is_verified	Estado de verificación del usuario.

7.30.3. Documentación de funciones miembro

7.30.3.1. getEmail()

```
String com.example.usuario_upv.proyecto3a.User.getEmail ()  [inline]
```

Obtiene el correo electrónico del usuario.

Devuelve

Correo electrónico del usuario.

7.30.3.2. getUsername()

```
String com.example.usuario_upv.proyecto3a.User.getUsername ()  [inline]
```

Obtiene el nombre de usuario.

Devuelve

Nombre de usuario.

7.30.3.3. isVerified()

```
boolean com.example.usuario_upv.proyecto3a.User.isVerified ()  [inline]
```

Obtiene el estado de verificación del usuario.

Devuelve

Estado de verificación del usuario.

7.30.3.4. setEmail()

```
void com.example.usuario_upv.proyecto3a.User.setEmail (
            String email)  [inline]
```

Establece el correo electrónico del usuario.

Parámetros

email	Correo electrónico del usuario.

7.30.3.5. setUsername()

```
void com.example.usuario_upv.proyecto3a.User.setUsername (
            String username)  [inline]
```

Establece el nombre de usuario.

Parámetros

username	Nombre de usuario.

7.30.3.6. setVerified()

```
void com.example.usuario_upv.proyecto3a.User.setVerified (
            boolean verified)  [inline]
```

Establece el estado de verificación del usuario.

Parámetros

verified	Estado de verificación del usuario.

7.30.3.7. toString()

```
String com.example.usuario_upv.proyecto3a.User.toString ()  [inline]
```

Sobrecarga el método toString para imprimir las propiedades del objeto.

Devuelve

Cadena de texto con las propiedades del objeto.

La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/User.java

7.31. Referencia de la clase com.example.usuario_upv.proyecto3a.UserConfig

Diagrama de herencia de com.example.usuario_upv.proyecto3a.UserConfig

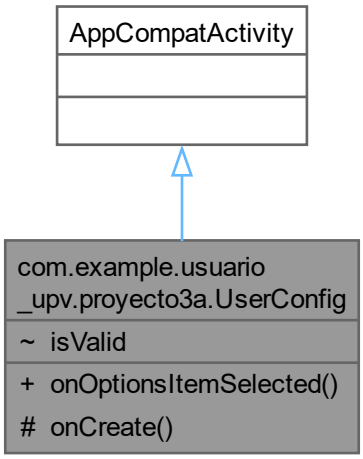
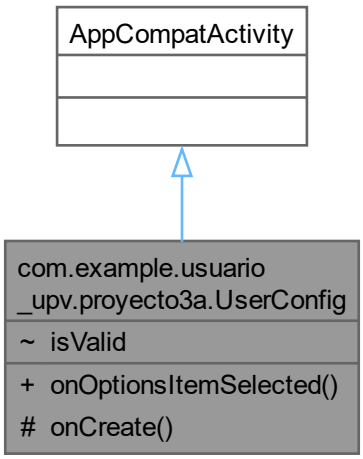


Diagrama de colaboración de com.example.usuario_upv.proyecto3a.UserConfig:



Métodos públicos

- boolean onOptionsItemSelected (MenuItem item)

Métodos protegidos

- void onCreate (Bundle savedInstanceState)

7.31.1. Descripción detallada

Clase UserConfig que maneja la configuración del usuario.

7.31.2. Documentación de funciones miembro

7.31.2.1. onCreate()

```
void com.example.usuario_upv.proyecto3a.UserConfig.onCreate (
            Bundle savedInstanceState)  [inline], [protected]
```

Método onCreate que inicializa la actividad.

Parámetros

savedInstanceState	Estado guardado de la instancia.

7.31.2.2. onOptionsItemSelected()

```
boolean com.example.usuario_upv.proyecto3a.UserConfig.onOptionsItemSelected (
            MenuItem item)  [inline]
```

Método para manejar la selección de opciones del menú.

Parámetros

item	El elemento del menú seleccionado.

Devuelve

true si el elemento seleccionado es el botón de retroceso, de lo contrario llama al método de la superclase.

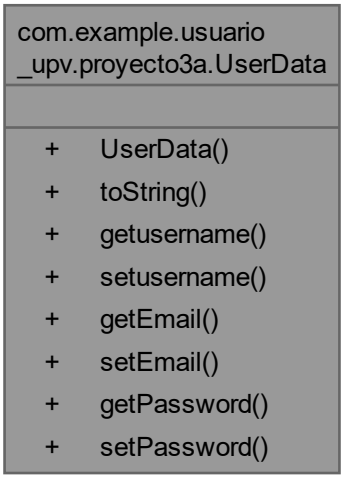
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/UserConfig.java

7.32. Referencia de la clase com.example.usuario_upv.proyecto3a.UserData

A class to represent user data including username, email, and password.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.UserData:



Métodos públicos

- UserData (String username, String email, String password)

  Constructor to initialize UserData with username, email, and password.
- String toString ()

  Converts the UserData object to a string representation.
- String getusername ()

  Gets the username of the user.

- void setusername (String username)

  Sets the username of the user.
- String getEmail ()

  Gets the email of the user.
- void setEmail (String email)

  Sets the email of the user.
- String getPassword ()

  Gets the password of the user.
- void setPassword (String password)

  Sets the password of the user.

7.32.1. Descripción detallada

A class to represent user data including username, email, and password.

7.32.2. Documentación de constructores y destructores

7.32.2.1. UserData()

```
com.example.usuario_upv.proyecto3a.UserData.UserData (
            String username,
            String email,
            String password)  [inline]
```

Constructor to initialize UserData with username, email, and password.

Parámetros

username	The username of the user.
email	The email of the user.
password	The password of the user.

7.32.3. Documentación de funciones miembro

7.32.3.1. getEmail()

```
String com.example.usuario_upv.proyecto3a.UserData.getEmail ()  [inline]
```

Gets the email of the user.

Devuelve

The email of the user.

7.32.3.2. getPassword()

```
String com.example.usuario_upv.proyecto3a.UserData.getPassword ()  [inline]
```

Gets the password of the user.

Devuelve

The password of the user.

7.32.3.3. getusername()

```
String com.example.usuario_upv.proyecto3a.UserData.getusername ()  [inline]
```

Gets the username of the user.

Devuelve

The username of the user.

7.32.3.4. setEmail()

```
void com.example.usuario_upv.proyecto3a.UserData.setEmail (
            String email)  [inline]
```

Sets the email of the user.

Parámetros

email	The new email of the user.

7.32.3.5. setPassword()

```
void com.example.usuario_upv.proyecto3a.UserData.setPassword (
            String password)  [inline]
```

Sets the password of the user.

Parámetros

password	The new password of the user.

7.32.3.6. setusername()

```
void com.example.usuario_upv.proyecto3a.UserData.setusername (
            String username)  [inline]
```

Sets the username of the user.

Parámetros

username	The new username of the user.

7.32.3.7. toString()

```
String com.example.usuario_upv.proyecto3a.UserData.toString ()  [inline]
```

Converts the UserData object to a string representation.

Devuelve

A string representation of the UserData object.

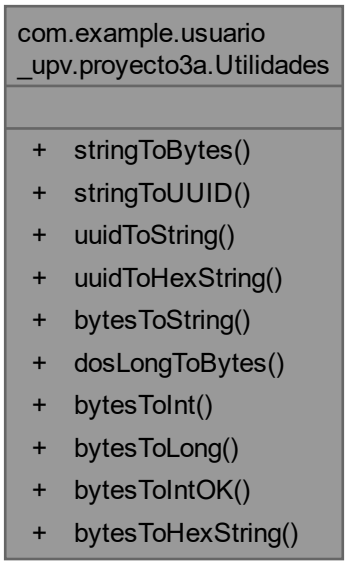
La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/UserData.java

7.33. Referencia de la clase com.example.usuario_upv.proyecto3a.Utilidades

Clase utilitaria para conversiones entre diferentes tipos de datos.

Diagrama de colaboración de com.example.usuario_upv.proyecto3a.Utilidades:



Métodos públicos estáticos

- static byte[ ] stringToBytes (String texto)

  Convierte una cadena a un arreglo de bytes.
- static UUID stringToUUID (String uuid)

  Convierte una cadena de 16 caracteres a un UUID.
- static String uuidToString (UUID uuid)

  Convierte un UUID a una cadena.
- static String uuidToHexString (UUID uuid)

  Convierte un UUID a una cadena hexadecimal.
- static String bytesToString (byte[ ] bytes)

  Convierte un arreglo de bytes a una cadena.
- static byte[ ] dosLongToBytes (long masSignificativos, long menosSignificativos)

  Convierte dos longitudes a un arreglo de bytes.
- static int bytesToInt (byte[ ] bytes)

  Convierte un arreglo de bytes a un entero.
- static long bytesToLong (byte[ ] bytes)

  Convierte un arreglo de bytes a un long.
- static int bytesToIntOK (byte[ ] bytes)

  Convierte un arreglo de bytes a un entero, manejando excepciones.
- static String bytesToHexString (byte[ ] bytes)

  Convierte un arreglo de bytes a una cadena hexadecimal.

7.33.1. Descripción detallada

Clase utilitaria para conversiones entre diferentes tipos de datos.

Esta clase contiene métodos para convertir entre cadenas, UUIDs y arreglos de bytes, así como otros métodos de utilidad.

7.33.2. Documentación de funciones miembro

7.33.2.1. bytesToHexString()

```
static String com.example.usuario_upv.proyecto3a.Utilidades.bytesToHexString (
            byte[] bytes)  [inline], [static]
```

Convierte un arreglo de bytes a una cadena hexadecimal.

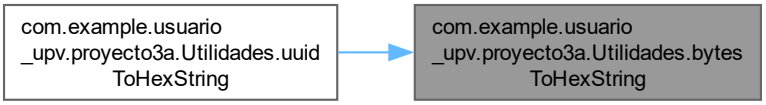
Parámetros

bytes	Arreglo de bytes a convertir.

Devuelve

La representación hexadecimal del arreglo de bytes.

Gráfico de llamadas a esta función:



7.33.2.2. bytesToInt()

```
static int com.example.usuario_upv.proyecto3a.Utilidades.bytesToInt (
            byte[] bytes)  [inline], [static]
```

Convierte un arreglo de bytes a un entero.

Parámetros

bytes	Arreglo de bytes a convertir.

Devuelve

El entero correspondiente.

7.33.2.3. bytesToIntOK()

```
static int com.example.usuario_upv.proyecto3a.Utilidades.bytesToIntOK (
            byte[] bytes)  [inline], [static]
```

Convierte un arreglo de bytes a un entero, manejando excepciones.

Parámetros

bytes	Arreglo de bytes a convertir.

Devuelve

El entero correspondiente.

Excepciones

Error	Si el arreglo de bytes es nulo o tiene más de 4 bytes.

7.33.2.4. bytesToLong()

```
static long com.example.usuario_upv.proyecto3a.Utilidades.bytesToLong (
            byte[] bytes)  [inline], [static]
```

Convierte un arreglo de bytes a un long.

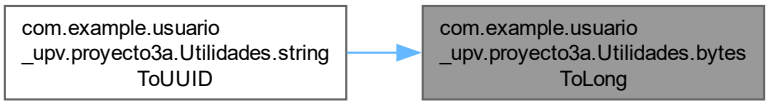
Parámetros

bytes	Arreglo de bytes a convertir.

Devuelve

El long correspondiente.

Gráfico de llamadas a esta función:



7.33.2.5. bytesToString()

```
static String com.example.usuario_upv.proyecto3a.Utilidades.bytesToString (
            byte[] bytes)  [inline], [static]
```

Convierte un arreglo de bytes a una cadena.

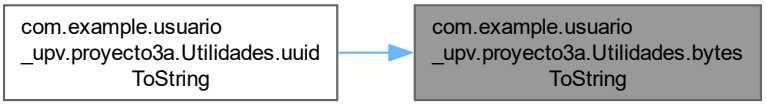
Parámetros

bytes	Arreglo de bytes a convertir.

Devuelve

La cadena correspondiente al arreglo de bytes.

Gráfico de llamadas a esta función:



7.33.2.6. dosLongToBytes()

```
static byte[] com.example.usuario_upv.proyecto3a.Utilidades.dosLongToBytes (
            long masSignificativos,
            long menosSignificativos)  [inline], [static]
```

Convierte dos longitudes a un arreglo de bytes.

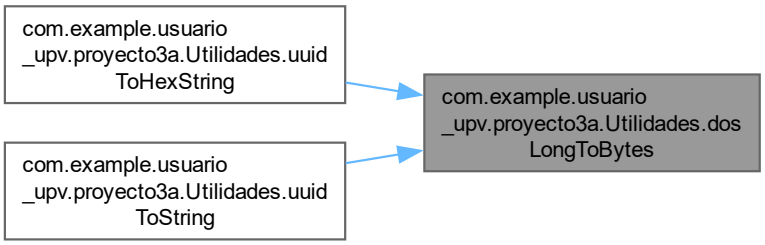
Parámetros

masSignificativos	Longitud más significativa.
menosSignificativos	Longitud menos significativa.

Devuelve

Arreglo de bytes que representa las dos longitudes.

Gráfico de llamadas a esta función:



7.33.2.7. stringToBytes()

```
static byte[] com.example.usuario_upv.proyecto3a.Utilidades.stringToBytes (
            String texto)  [inline], [static]
```

Convierte una cadena a un arreglo de bytes.

Parámetros

texto	Cadena a convertir.

Devuelve

Arreglo de bytes correspondiente a la cadena.

7.33.2.8. stringToUUID()

```
static UUID com.example.usuario_upv.proyecto3a.Utilidades.stringToUUID (
            String uuid)  [inline], [static]
```

Convierte una cadena de 16 caracteres a un UUID.

Parámetros

uuid	Cadena de 16 caracteres que representa un UUID.

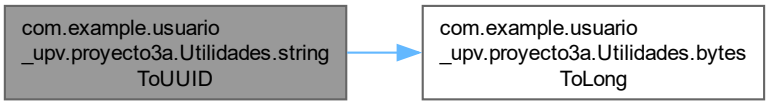
Devuelve

El UUID correspondiente.

Excepciones

Error	Si la cadena no tiene 16 caracteres.

Gráfico de llamadas de esta función:



7.33.2.9. uuidToHexString()

```
static String com.example.usuario_upv.proyecto3a.Utilidades.uuidToHexString (
            UUID uuid)  [inline], [static]
```

Convierte un UUID a una cadena hexadecimal.

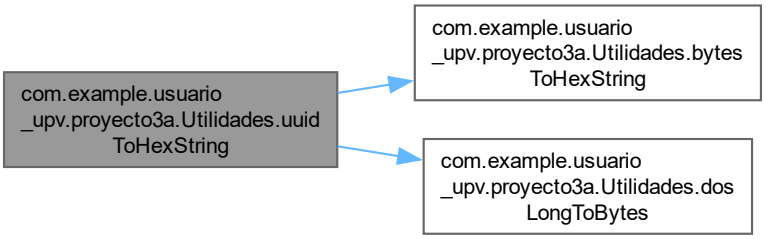
Parámetros

uuid	El UUID a convertir.

Devuelve

La representación hexadecimal del UUID.

Gráfico de llamadas de esta función:



7.33.2.10. uuidToString()

```
static String com.example.usuario_upv.proyecto3a.Utilidades.uuidToString (
            UUID uuid)  [inline], [static]
```

Convierte un UUID a una cadena.

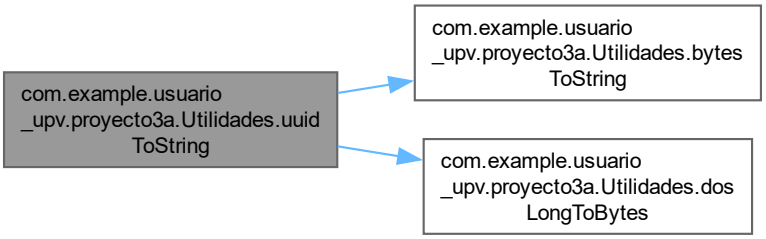
Parámetros

uuid	El UUID a convertir.

Devuelve

La representación de cadena del UUID.

Gráfico de llamadas de esta función:



La documentación de esta clase está generada del siguiente archivo:

- app/src/main/java/com/example/usuario_upv/proyecto3a/Utilidades.java

Capítulo 8

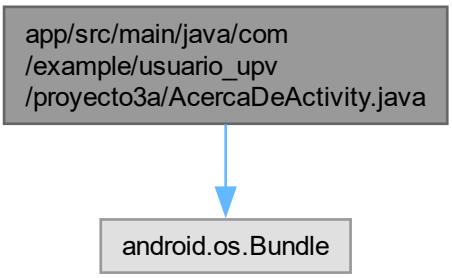
Documentación de archivos

8.1. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/AcercaDeActivity.java

Clase que representa la actividad "Acerca de" en la aplicación.

```
import android.os.Bundle;
```

Gráfico de dependencias incluidas en AcercaDeActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.AcercaDeActivity

  Clase que implementa la actividad "Acerca de".

Paquetes

- package com.example.usuario_upv.proyecto3a

8.1.1. Descripción detallada

Clase que representa la actividad "Acerca de" en la aplicación.

Esta actividad muestra información relacionada con la aplicación, como créditos, información de contacto o detalles del proyecto.

8.2. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Ajuste.java

Clase que representa un elemento de ajuste dentro de la aplicación.

Clases

- class com.example.usuario_upv.proyecto3a.Ajuste

  Clase que modela un ajuste con un texto descriptivo y una imagen.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.2.1. Descripción detallada

Clase que representa un elemento de ajuste dentro de la aplicación.

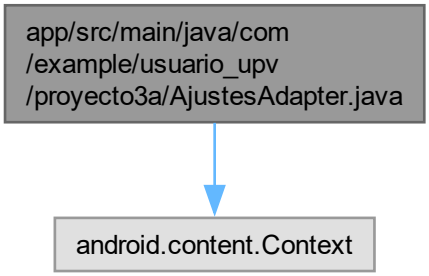
Esta clase se utiliza para almacenar información sobre un ajuste, incluyendo un texto descriptivo y una imagen asociada.

8.3. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/AjustesAdapter.java

Adaptador para gestionar y mostrar una lista de ajustes en un RecyclerView.

```
import android.content.Context;
```

Gráfico de dependencias incluidas en AjustesAdapter.java:



Clases

- class com.example.usuario_upv.proyecto3a.AjustesAdapter

  Adaptador para manejar la lista de ajustes en el RecyclerView.
- interface com.example.usuario_upv.proyecto3a.AjustesAdapter.OnItemClickListener

  Interfaz para manejar eventos de clic en los elementos.
- class **com.example.usuario_upv.proyecto3a.AjustesAdapter.AjustesViewHolder**

  Clase ViewHolder que contiene las vistas de un elemento.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.3.1. Descripción detallada

Adaptador para gestionar y mostrar una lista de ajustes en un RecyclerView.

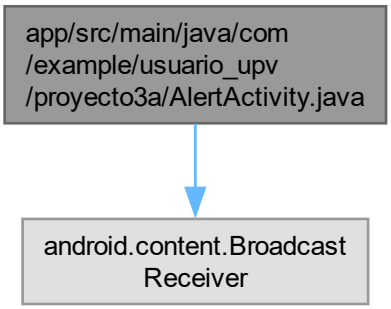
Este adaptador enlaza una lista de objetos de tipo Ajuste con un RecyclerView, proporcionando una vista personalizada para cada elemento y manejando eventos de clic.

8.4. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/AlertActivity.java

Clase que gestiona la actividad de alertas en la aplicación.

```
import android.content.BroadcastReceiver;
```

Gráfico de dependencias incluidas en AlertActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.AlertActivity

  Clase que implementa la actividad para mostrar y gestionar alertas.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.4.1. Descripción detallada

Clase que gestiona la actividad de alertas en la aplicación.

Esta clase muestra las alertas en un RecyclerView, las recibe a través de un BroadcastReceiver y permite gestionarlas mediante opciones de menú.

8.5. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/AlertaData.java

Clase que representa los datos de una alerta en la aplicación.

Clases

- class com.example.usuario_upv.proyecto3a.AlertaData

  Clase que modela los datos de una alerta.
- class **com.example.usuario_upv.proyecto3a.AlertaData.Location**

  Clase interna que representa la ubicación de una alerta.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.5.1. Descripción detallada

Clase que representa los datos de una alerta en la aplicación.

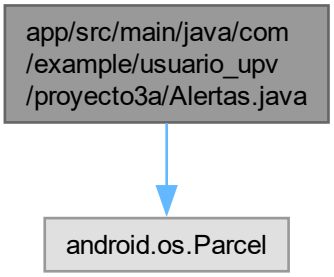
Esta clase almacena información relacionada con una alerta, como su código, marca de tiempo, ubicación, y un identificador único.

8.6. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/Alertas.java

Enum que representa diferentes tipos de alertas en la aplicación.

```
import android.os.Parcel;
```

Gráfico de dependencias incluidas en Alertas.java:



Clases

- enum com.example.usuario_upv.proyecto3a.Alertas

Paquetes

- package com.example.usuario_upv.proyecto3a

8.6.1. Descripción detallada

Enum que representa diferentes tipos de alertas en la aplicación.

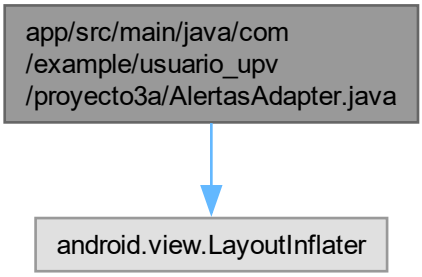
Cada alerta contiene un código único y un mensaje descriptivo. Además, implementa la interfaz Parcelable para poder ser transmitida entre componentes de Android.

8.7. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/AlertasAdapter.java

Adaptador para gestionar y mostrar una lista de alertas en un RecyclerView.

```
import android.view.LayoutInflater;
```

Gráfico de dependencias incluidas en AlertasAdapter.java:



Clases

- class com.example.usuario_upv.proyecto3a.AlertasAdapter

  Adaptador para el RecyclerView que maneja la lista de alertas.
- interface com.example.usuario_upv.proyecto3a.AlertasAdapter.OnAlertasListener

  Interfaz para manejar eventos de interacción con las alertas.
- class **com.example.usuario_upv.proyecto3a.AlertasAdapter.AlertasViewHolder**

  Clase ViewHolder que contiene las vistas de un elemento del RecyclerView.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.7.1. Descripción detallada

Adaptador para gestionar y mostrar una lista de alertas en un RecyclerView.

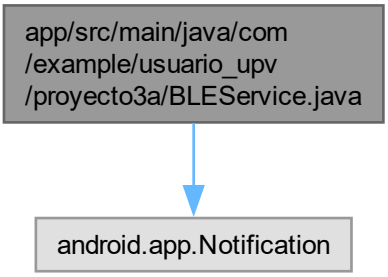
Este adaptador enlaza los datos de las alertas con las vistas de cada elemento en el RecyclerView, permitiendo también manejar eventos como la eliminación de alertas.

8.8. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/BLEService.java

Servicio encargado de gestionar la comunicación Bluetooth Low Energy (BLE).

```
import android.app.Notification;
```

Gráfico de dependencias incluidas en BLEService.java:



Clases

- class com.example.usuario_upv.proyecto3a.BLEService

  Servicio encargado de gestionar la conectividad BLE y alertas asociadas.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.8.1. Descripción detallada

Servicio encargado de gestionar la comunicación Bluetooth Low Energy (BLE).

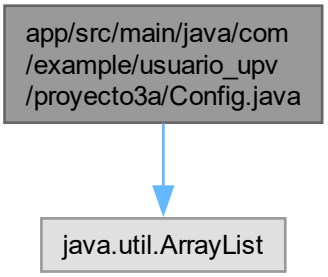
Este servicio monitorea dispositivos BLE, gestiona alertas basadas en eventos de conectividad y datos, y utiliza Retrofit para interactuar con un servidor remoto.

8.9. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Config.java

Configuración de la aplicación.

```
import java.util.ArrayList;
```

Gráfico de dependencias incluidas en Config.java:



Clases

- class com.example.usuario_upv.proyecto3a.Config

  Clase de configuración de la aplicación.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.9.1. Descripción detallada

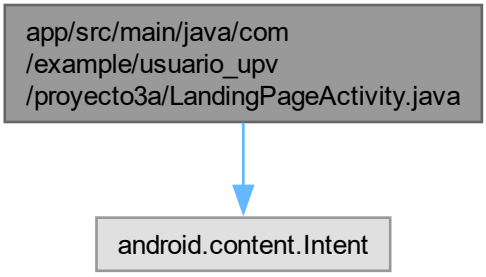
Configuración de la aplicación.

Esta clase contiene la configuración global de la aplicación, incluyendo la URL base para las llamadas a la API y una lista de UUIDs.

8.10. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/LandingPageActivity.java

```
import android.content.Intent;
```

Gráfico de dependencias incluidas en LandingPageActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.LandingPageActivity

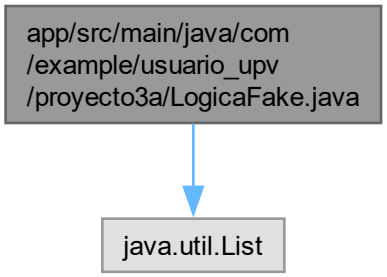
Paquetes

- package com.example.usuario_upv.proyecto3a

8.11. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/LogicaFake.java

```
import java.util.List;
```

Gráfico de dependencias incluidas en LogicaFake.java:



Clases

- interface com.example.usuario_upv.proyecto3a.LogicaFake

  Interfaz para definir los endpoints de la API de sensores.

Paquetes

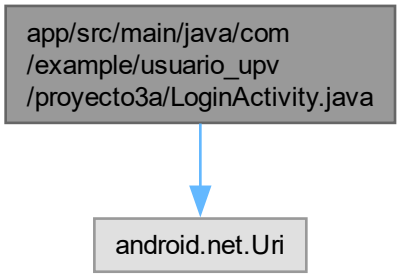
- package com.example.usuario_upv.proyecto3a

8.12. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/LoginActivity.java

Clase que gestiona la actividad de inicio de sesión en la aplicación.

```
import android.net.Uri;
```

Gráfico de dependencias incluidas en LoginActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.LoginActivity

  Clase que implementa la actividad de inicio de sesión.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.12.1. Descripción detallada

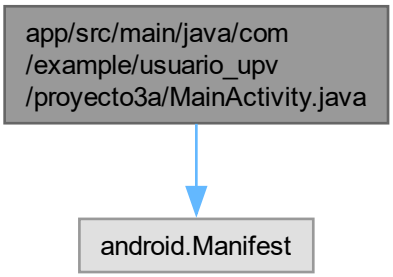
Clase que gestiona la actividad de inicio de sesión en la aplicación.

Esta clase permite a los usuarios iniciar sesión utilizando su email y contraseña. También maneja la autenticación del usuario mediante un token y guarda los datos del usuario en las preferencias compartidas.

8.13. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/MainActivity.java

```
import android.Manifest;
```

Gráfico de dependencias incluidas en MainActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.MainActivity

  MainActivity de la aplicación.
- class com.example.usuario_upv.proyecto3a.MainActivity.MiPagerAdapter

  Adaptador para gestionar las pestañas de la aplicación.

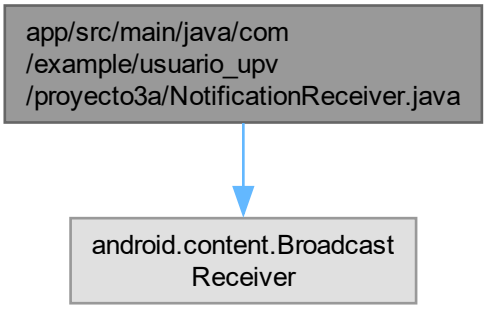
Paquetes

- package com.example.usuario_upv.proyecto3a

8.14. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/NotificationReceiver.java

```
import android.content.BroadcastReceiver;
```

Gráfico de dependencias incluidas en NotificationReceiver.java:



Clases

- class com.example.usuario_upv.proyecto3a.NotificationReceiver

Paquetes

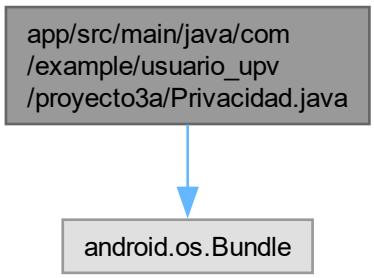
- package com.example.usuario_upv.proyecto3a

8.15. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/Privacidad.java

Actividad para mostrar la política de privacidad.

```
import android.os.Bundle;
```

Gráfico de dependencias incluidas en Privacidad.java:



Clases

- class com.example.usuario_upv.proyecto3a.Privacidad

  Actividad para la política de privacidad.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.15.1. Descripción detallada

Actividad para mostrar la política de privacidad.

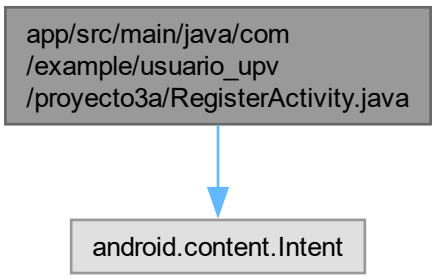
Esta clase representa una actividad que muestra la política de privacidad de la aplicación.

8.16. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/RegisterActivity.java

Actividad para el registro de usuarios.

```
import android.content.Intent;
```

Gráfico de dependencias incluidas en RegisterActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.RegisterActivity

  Actividad para el registro de usuarios.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.16.1. Descripción detallada

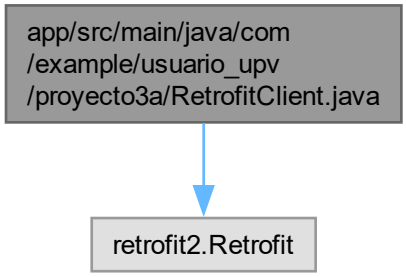
Actividad para el registro de usuarios.

Esta clase representa una actividad que permite a los usuarios registrarse en la aplicación.

8.17. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/RetrofitClient.java

```
import retrofit2.Retrofit;
```

Gráfico de dependencias incluidas en RetrofitClient.java:



Clases

- class com.example.usuario_upv.proyecto3a.RetrofitClient

  Clase para gestionar la instancia de Retrofit.

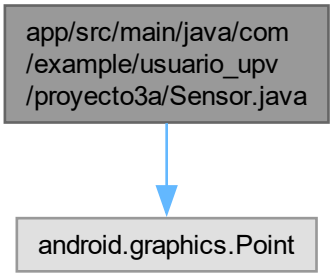
Paquetes

- package com.example.usuario_upv.proyecto3a

8.18. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Sensor.java

```
import android.graphics.Point;
```

Gráfico de dependencias incluidas en Sensor.java:



Clases

- class com.example.usuario_upv.proyecto3a.Sensor

  Clase que representa un sensor con un UUID y un email asociado.

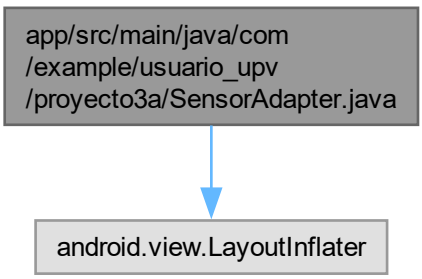
Paquetes

- package com.example.usuario_upv.proyecto3a

8.19. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/SensorAdapter.java

```
import android.view.LayoutInflater;
```

Gráfico de dependencias incluidas en SensorAdapter.java:



Clases

- class com.example.usuario_upv.proyecto3a.SensorAdapter

  Adaptador para mostrar una lista de sensores en un RecyclerView.
- class **com.example.usuario_upv.proyecto3a.SensorAdapter.SensorViewHolder**

  ViewHolder para los elementos del sensor.

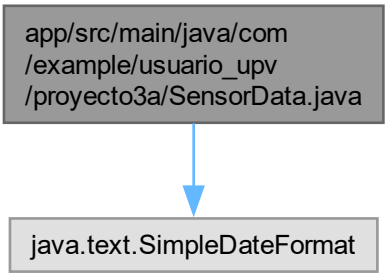
Paquetes

- package com.example.usuario_upv.proyecto3a

8.20. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/SensorData.java

```
import java.text.SimpleDateFormat;
```

Gráfico de dependencias incluidas en SensorData.java:



Clases

- class com.example.usuario_upv.proyecto3a.SensorData

  Clase que representa los datos de un sensor.
- class **com.example.usuario_upv.proyecto3a.SensorData.Location**

  Clase que representa la ubicación de un sensor.

Paquetes

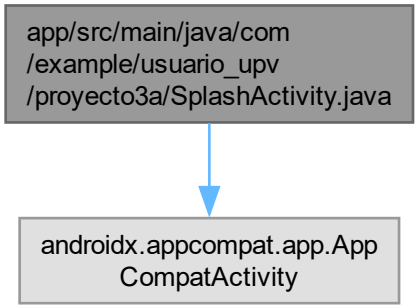
- package com.example.usuario_upv.proyecto3a

8.21. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/SplashActivity.java

Actividad de pantalla de bienvenida.

```
import androidx.appcompat.app.AppCompatActivity;
```

Gráfico de dependencias incluidas en SplashActivity.java:



Clases

- class com.example.usuario_upv.proyecto3a.SplashActivity

Paquetes

- package com.example.usuario_upv.proyecto3a

8.21.1. Descripción detallada

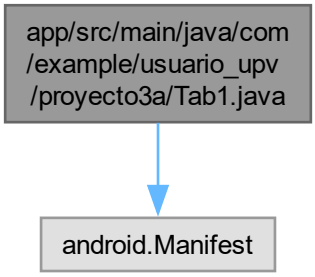
Actividad de pantalla de bienvenida.

Esta actividad muestra una pantalla de bienvenida durante un breve período antes de iniciar la actividad principal de la aplicación.

8.22. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Tab1.java

```
import android.Manifest;
```

Gráfico de dependencias incluidas en Tab1.java:



Clases

- class com.example.usuario_upv.proyecto3a.Tab1

  Fragmento que muestra un mapa con datos de sensores.

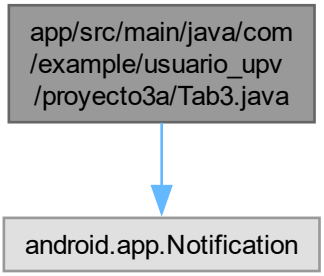
Paquetes

- package com.example.usuario_upv.proyecto3a

8.23. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Tab3.java

```
import android.app.Notification;
```

Gráfico de dependencias incluidas en Tab3.java:



Clases

- class com.example.usuario_upv.proyecto3a.Tab3

  Fragmento que representa la tercera pestaña de la aplicación.

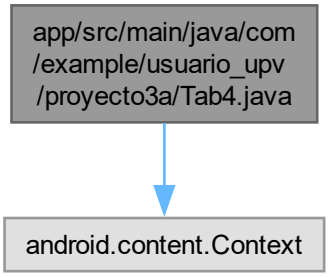
Paquetes

- package com.example.usuario_upv.proyecto3a

8.24. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Tab4.java

```
import android.content.Context;
```

Gráfico de dependencias incluidas en Tab4.java:



Clases

- class com.example.usuario_upv.proyecto3a.Tab4

  Fragmento que representa la cuarta pestaña de la aplicación.

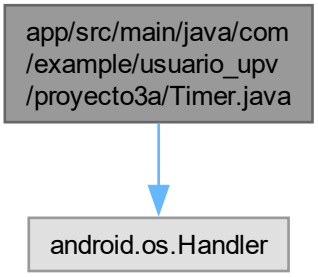
Paquetes

- package com.example.usuario_upv.proyecto3a

8.25. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/Timer.java

```
import android.os.Handler;
```

Gráfico de dependencias incluidas en Timer.java:



Clases

- class com.example.usuario_upv.proyecto3a.Timer

  Clase que representa un temporizador.

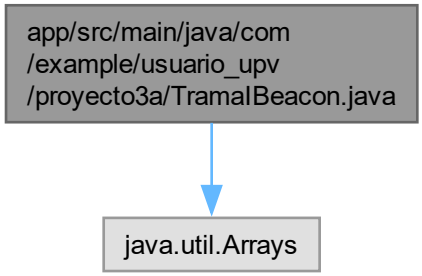
Paquetes

- package com.example.usuario_upv.proyecto3a

8.26. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/TramaIBeacon.java

```
import java.util.Arrays;
```

Gráfico de dependencias incluidas en TramaIBeacon.java:



Clases

- class com.example.usuario_upv.proyecto3a.TramaIBeacon

  Clase que representa la trama de un beacon iBeacon.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.27. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/User.java

Clases

- class com.example.usuario_upv.proyecto3a.User

  Clase que representa un usuario.

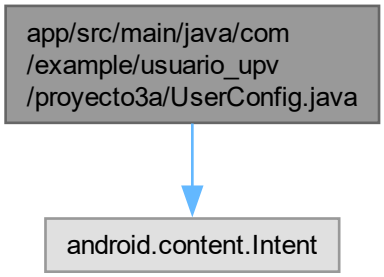
Paquetes

- package com.example.usuario_upv.proyecto3a

8.28. Referencia del archivo app/src/main/java/com/example/usuario_upv/proyecto3a/UserConfig.java

```
import android.content.Intent;
```

Gráfico de dependencias incluidas en UserConfig.java:



Clases

- class com.example.usuario_upv.proyecto3a.UserConfig

Paquetes

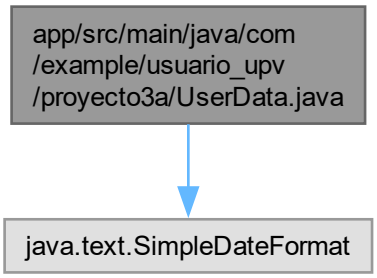
- package com.example.usuario_upv.proyecto3a

8.29. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/UserData.java

This file contains the UserData class which represents user information.

```
import java.text.SimpleDateFormat;
```

Gráfico de dependencias incluidas en UserData.java:



Clases

- class com.example.usuario_upv.proyecto3a.UserData

  A class to represent user data including username, email, and password.

Paquetes

- package com.example.usuario_upv.proyecto3a

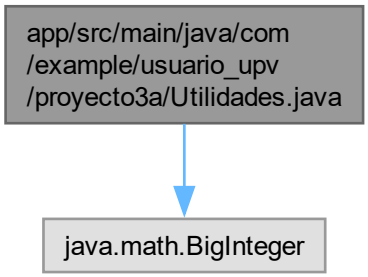
8.29.1. Descripción detallada

This file contains the UserData class which represents user information.

8.30. Referencia del archivo app/src/main/java/com/example/usuario_↩ upv/proyecto3a/Utilidades.java

```
import java.math.BigInteger;
```

Gráfico de dependencias incluidas en Utilidades.java:



Clases

- class com.example.usuario_upv.proyecto3a.Utilidades

  Clase utilitaria para conversiones entre diferentes tipos de datos.

Paquetes

- package com.example.usuario_upv.proyecto3a

8.31. Referencia del archivo README.md