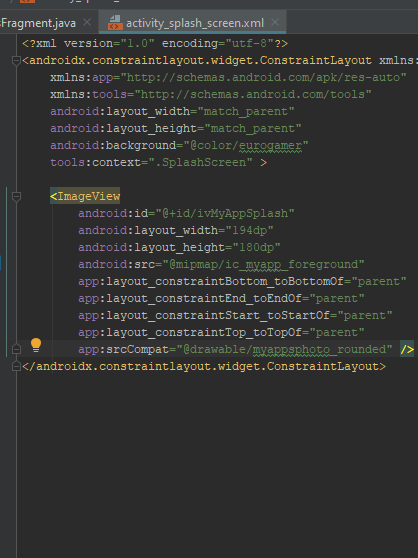
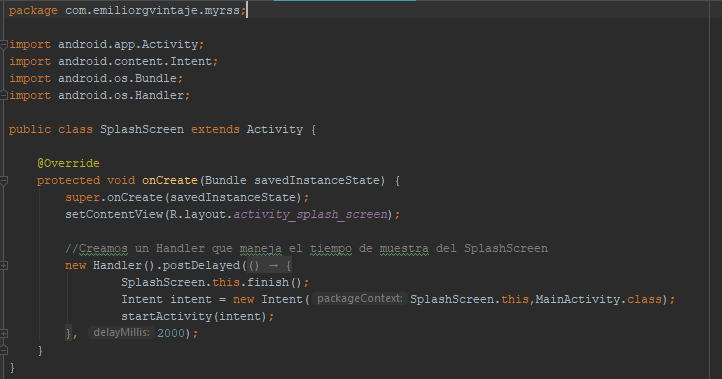
**Desarrollo App en Android**

**App: MyApps**

1. Splash Screen
   1. Layout

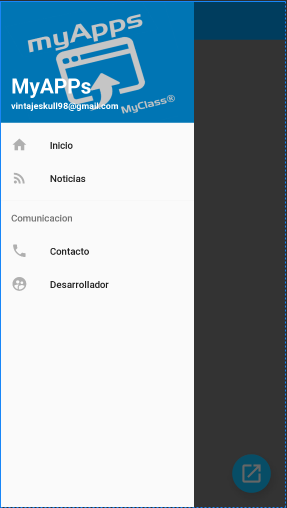
Para diseñar el Layout de la Splash Screen, basado en un Constraint Layout, se ha agregado un ImageView con el logo de MyApps centrado.

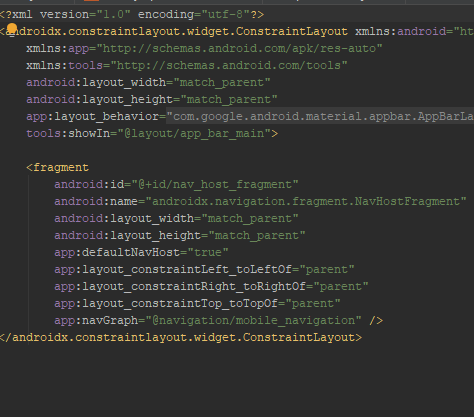
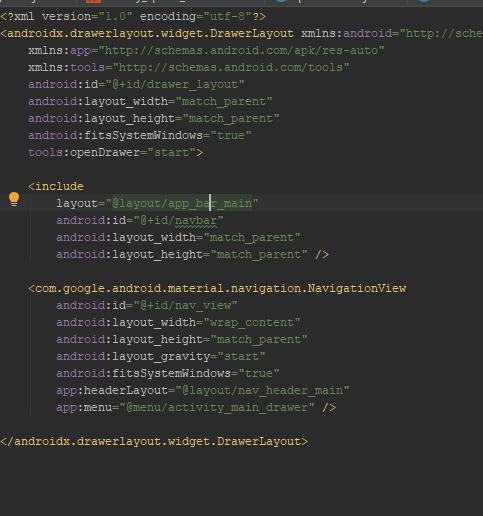
* 1. Codigo

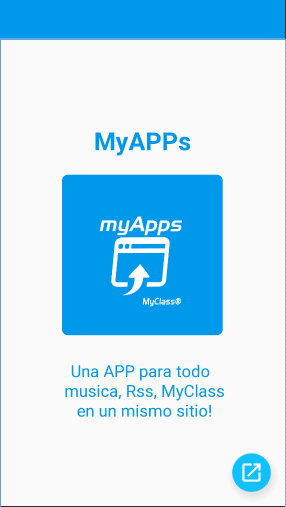


Acompañado de un código simple, iniciamos un Handler que gestiona el tiempo el cual mostramos la Actividad de SplashScreen pasandole la Actividad Principal que iniciara a continuación.

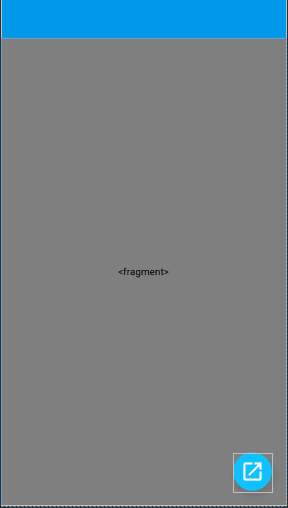
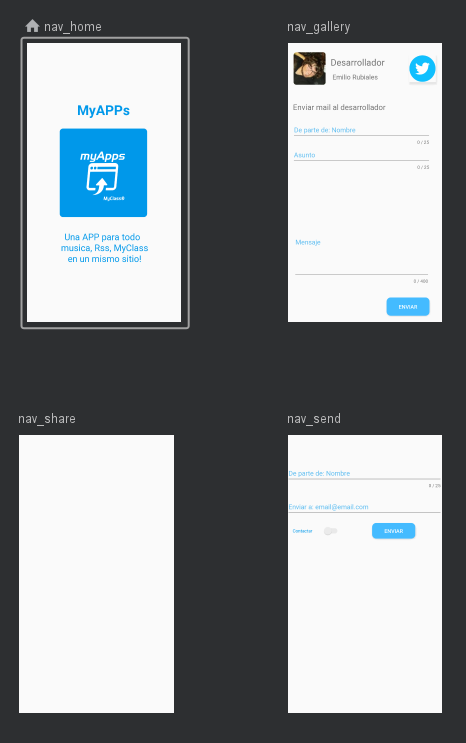
1. MainActivity
   1. Layout

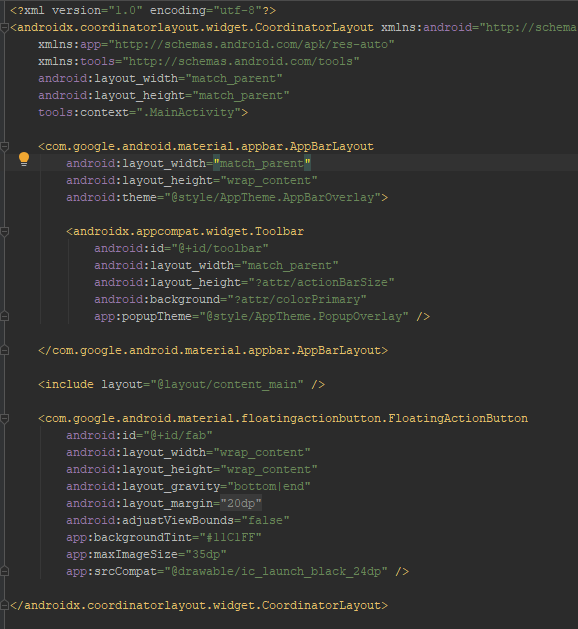
Layout simple que da pie al navigation drawer mostrando dicho menú en pantalla y un include en el cual le pasamos el content\_main que se explicará.

  
Viendo el Content Main en el que incluimos el fragment que hosteará el resto de fragments que se han diseñado.

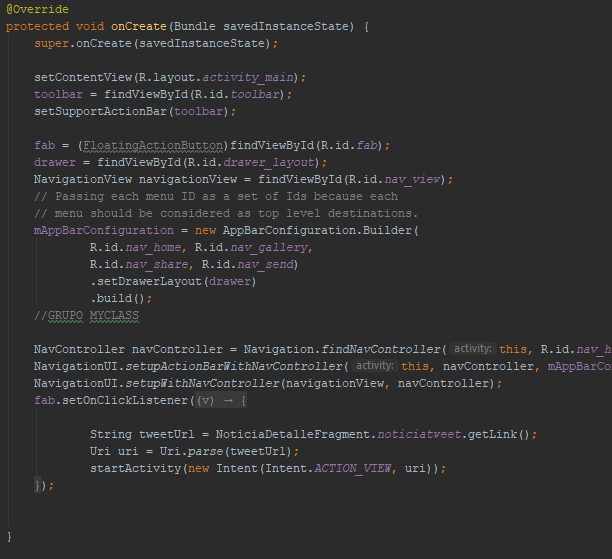
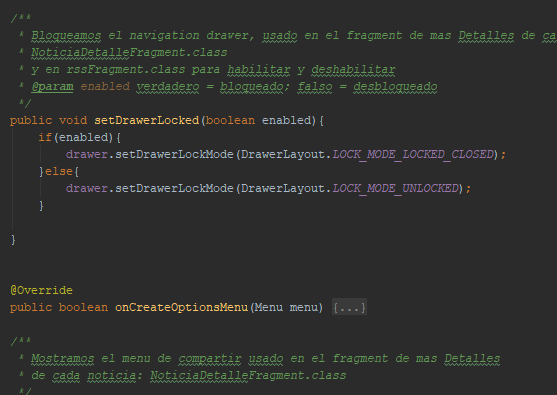


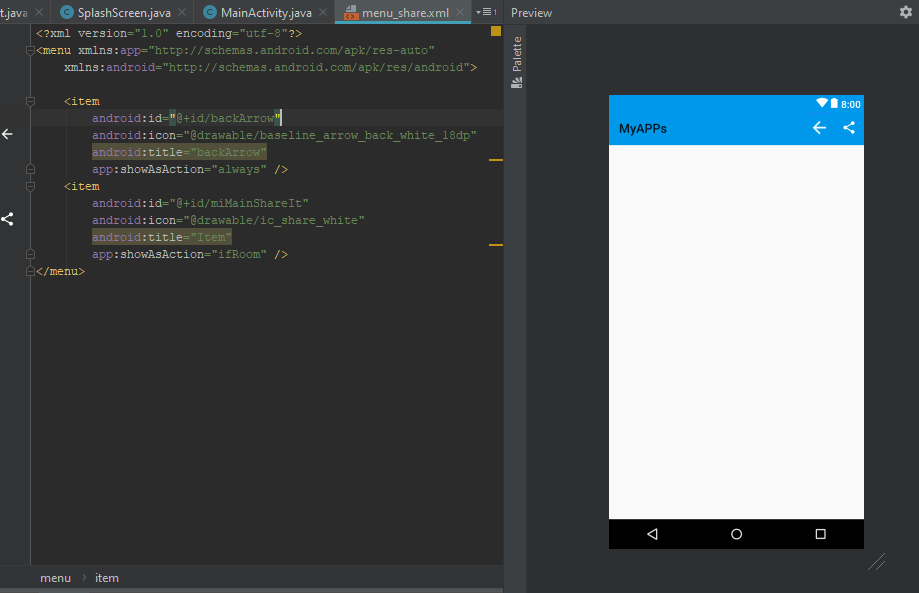
En el navigation que gestiona los fragments que se han diseñado, encontramos esta distribucion con los 4 fragments que podemos acceder desde el menu

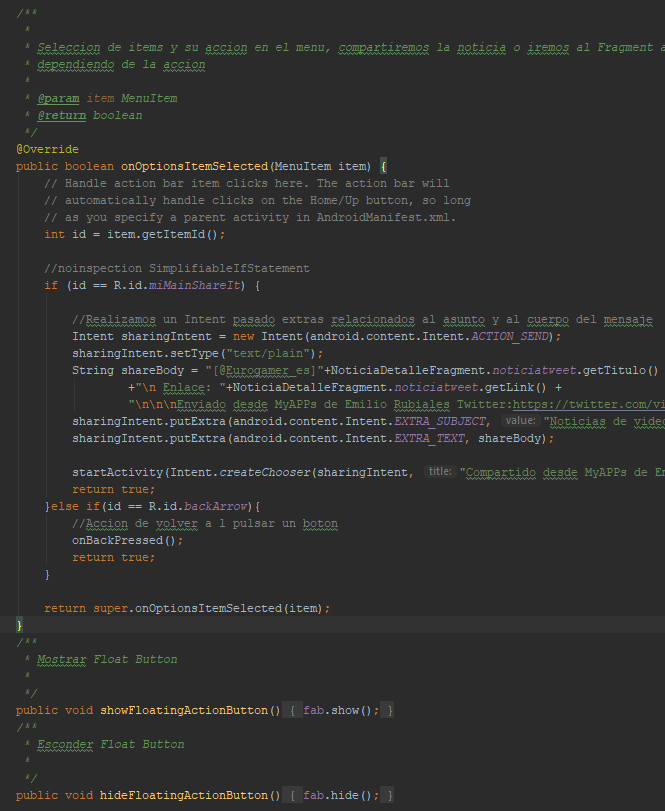
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Para la Toolbar, la implementamos desde otro layout en el cual tenemos el content main incluido en el propio layout para mostrarlo en la actividad principal y un Float Action Button el cual lo ocultamos y solo se mostrará en el Fragment de Detalles de las noticias



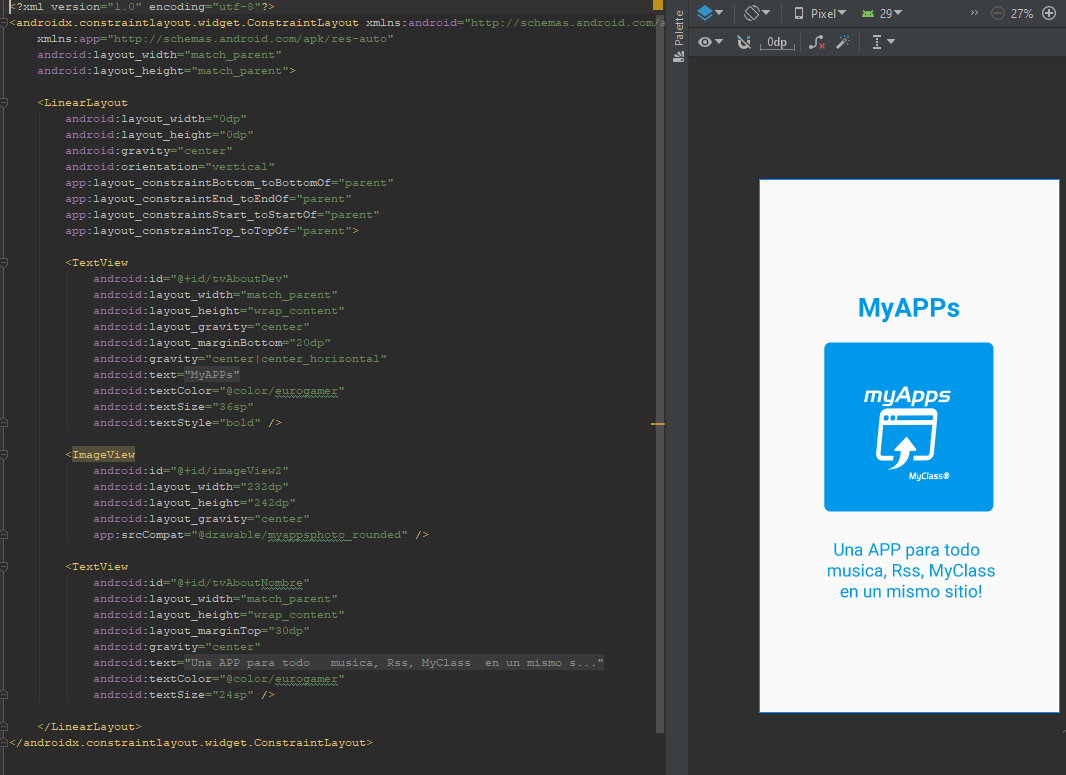
* 1. Codigo

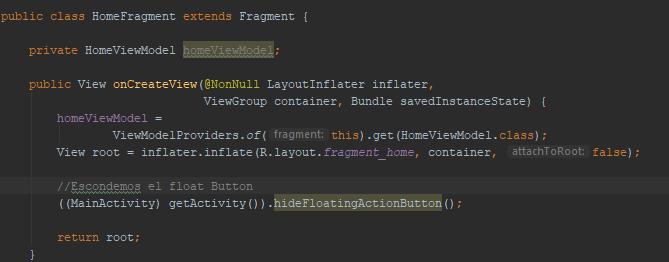
A diferencia del Splash, se presenta un código más completo. Acompañando el onCreate, declarando todas las variables necesarias del layout principal junto con los Listener necesarios. A su vez, declaramos varios métodos los cuales llamaremos desde los fragments, en este caso uno para bloquear el despliegue del menú donde queramos

Mencionando mas metodos propios, encontramos para editar la visibilidad del Float Action Button y de nuestro menu de compartir, el cual solo mostraremos en el detalle de cada noticia, para que el usuario pueda volver hacia el fragment anterior o compartir.



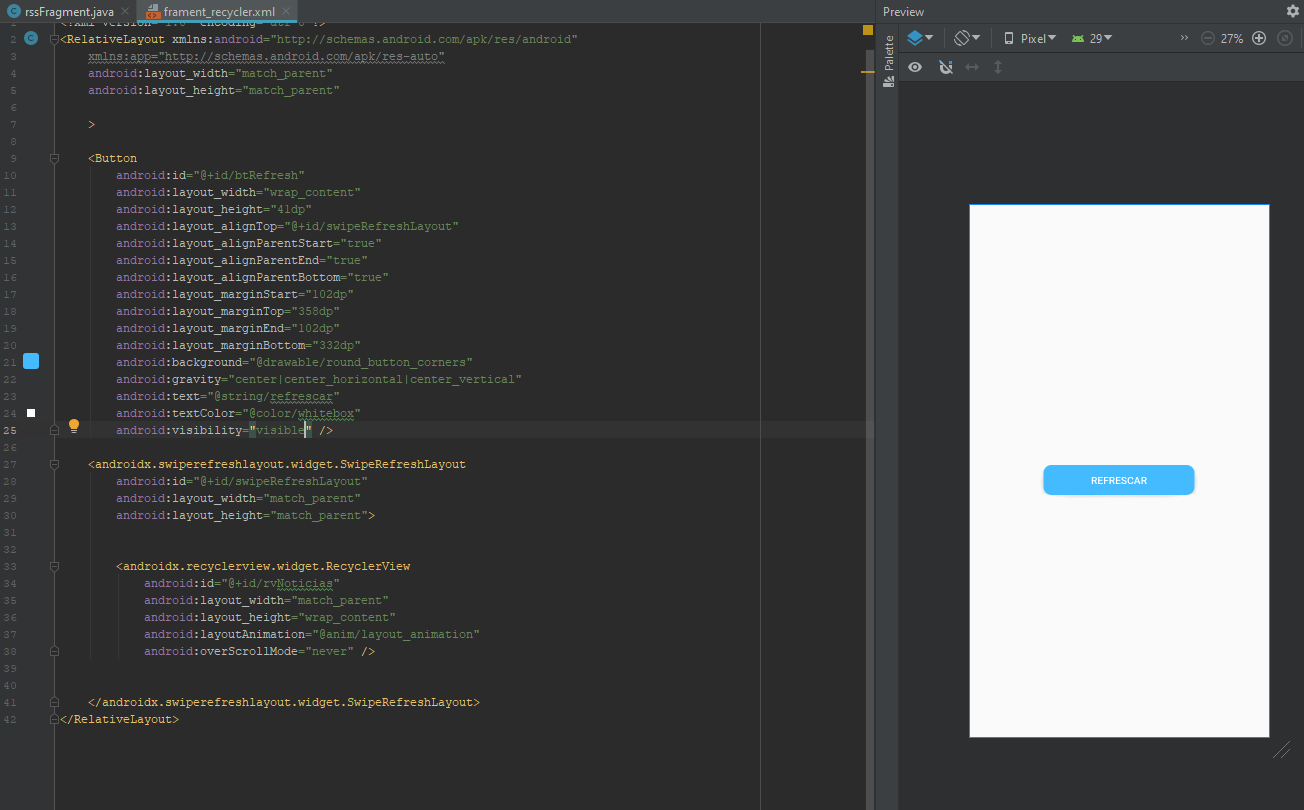
1. Fragments
   1. Home Layout
      1. Layout

Una solución rápida para mostrarle al Usuario que está en la pagina de Inicio de la App, agregando un texto describiendo de manera sencilla que contiene/contendrá en un futuro

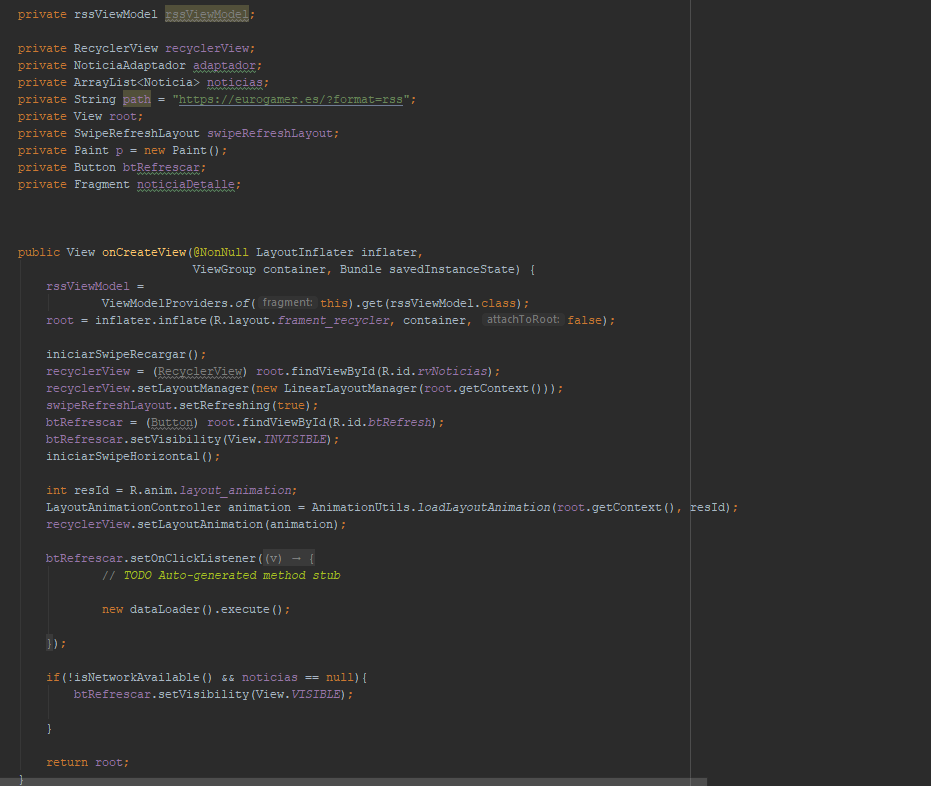
* + 1. Codigo

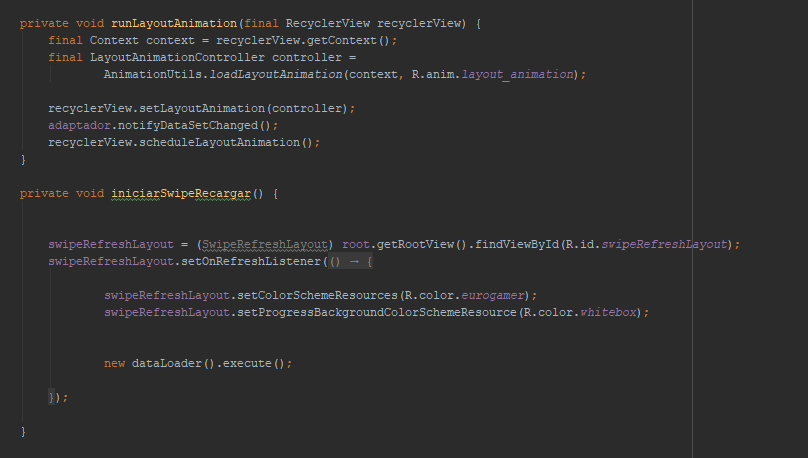
Una clase sencilla en la cual instanciamos los ítems del layout y escondiendo, con el método descrito en el Main Activity, el botón flotante.

* 1. RssFragment
     1. Layout

Para la gestión del Rss se ha utilizado un Recycler View pero además, incluimos un SwipeRefreshLayout para poder actualizar dicho Recycler simplemente con realizar un Drag hacia abajo en la zona superior de la pantalla. En caso de que no se puedan mostrar las noticias con un máximo de 20 se mostrara en pantalla, si no se han cargado previamente la lista, un botón para poder refrescar el Recycler que se mostrara invisible hasta el imprevisto

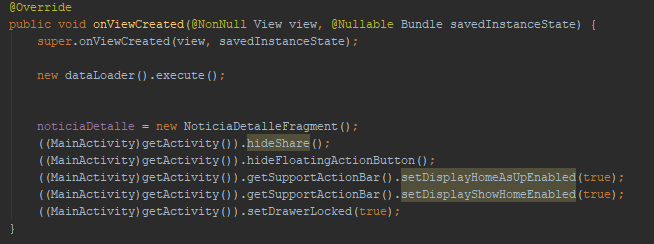
* + 1. Codigo

Nada mas entrar en el onCreateView(), hemos instanciado todos los ítems del layout previamente definidos como variables globales del Fragment .

El siguiente metodo definido lo usamos para definir la animacion que tendrá el RecyclerView al desplegarse, creada previamente en nuestra carpeta anim del proyecto. Tambien, llamado en el onCreateView, inicializamos el SwipeRefresh para recargar el Recycler.

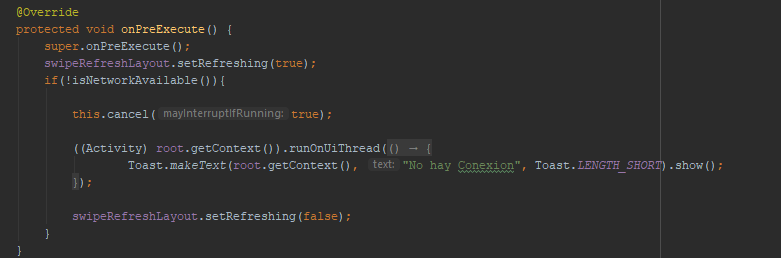
Una vez creada e instanciada la vista, procedemos a cargar todo lo necesario para mostrar, gracias al BackEnd, la informacion.

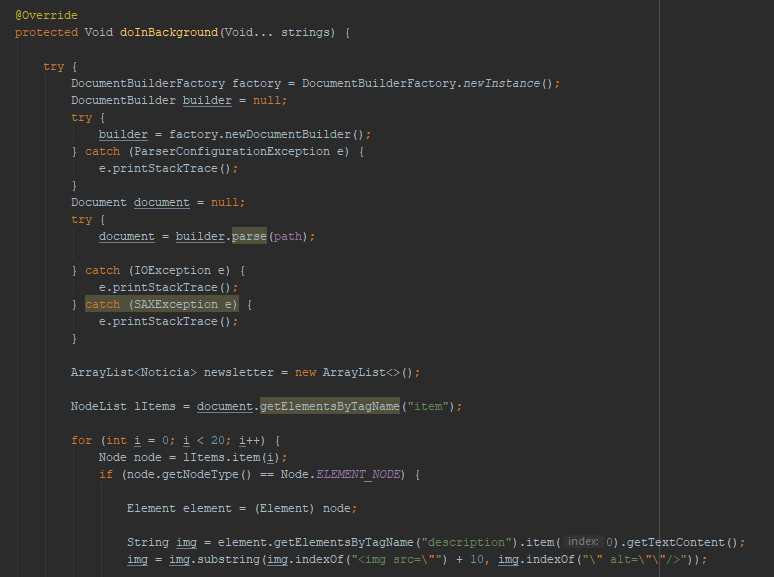
Ejecutando en onViewCreated() para que se ejecute una vez creada la vista, realizamos una nueva instancia y ejecucion de nuestra ASyncTask definida en la clase para recoger datos de internet.

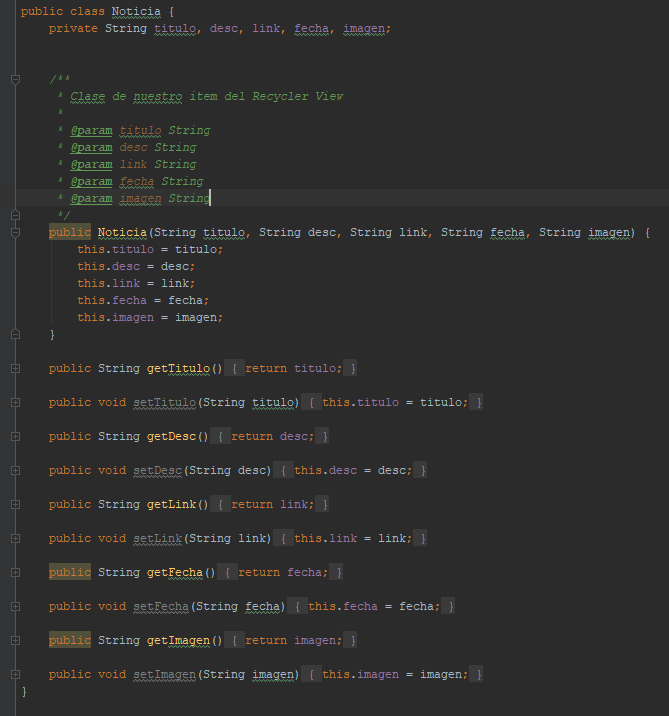


En dicha ASyncTask lo que realizamos es recoger el documento XML desde la URL indicada, tratar la informacion que contiene y desplegarla en el RecyclerView.

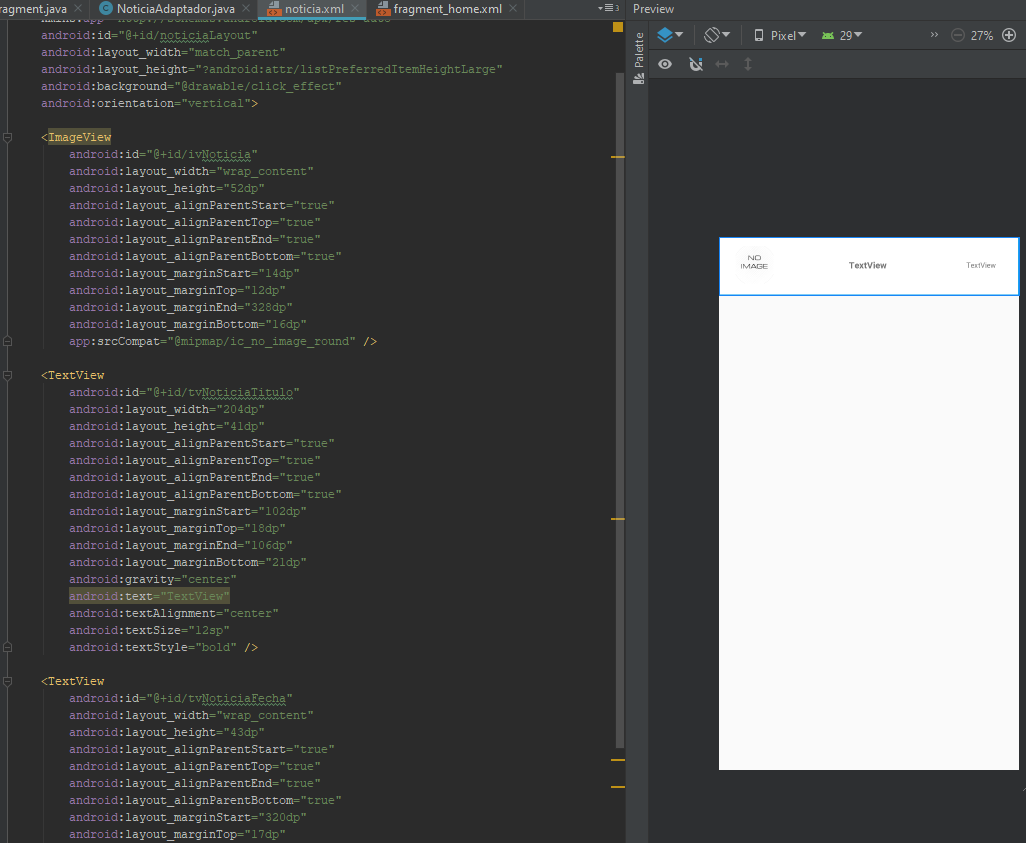
Todas estas acciones estan divididas en un primer metodo el cual se ejecuta previamente a todo el background donde controlamos que la App disponga de una conexión a la red valida, si esto no es asi, se habilitara dicho boton mencionado en el layout de este Fragment en caso de que no haya contenido ya disponible en el Recycler

Ya una vez controlada la posible excepcion que pueda ocasionar el no tener conexión, en segundo plano realizara la descarga de los datos del fichero XML para tratarlo con un parser basado en la obtencion de datos por nodos

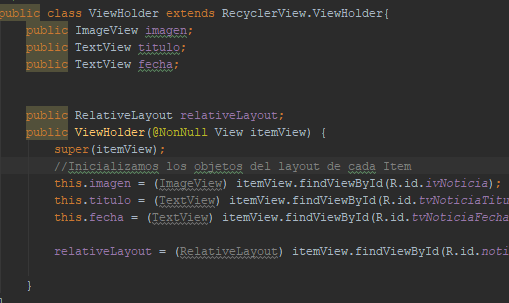


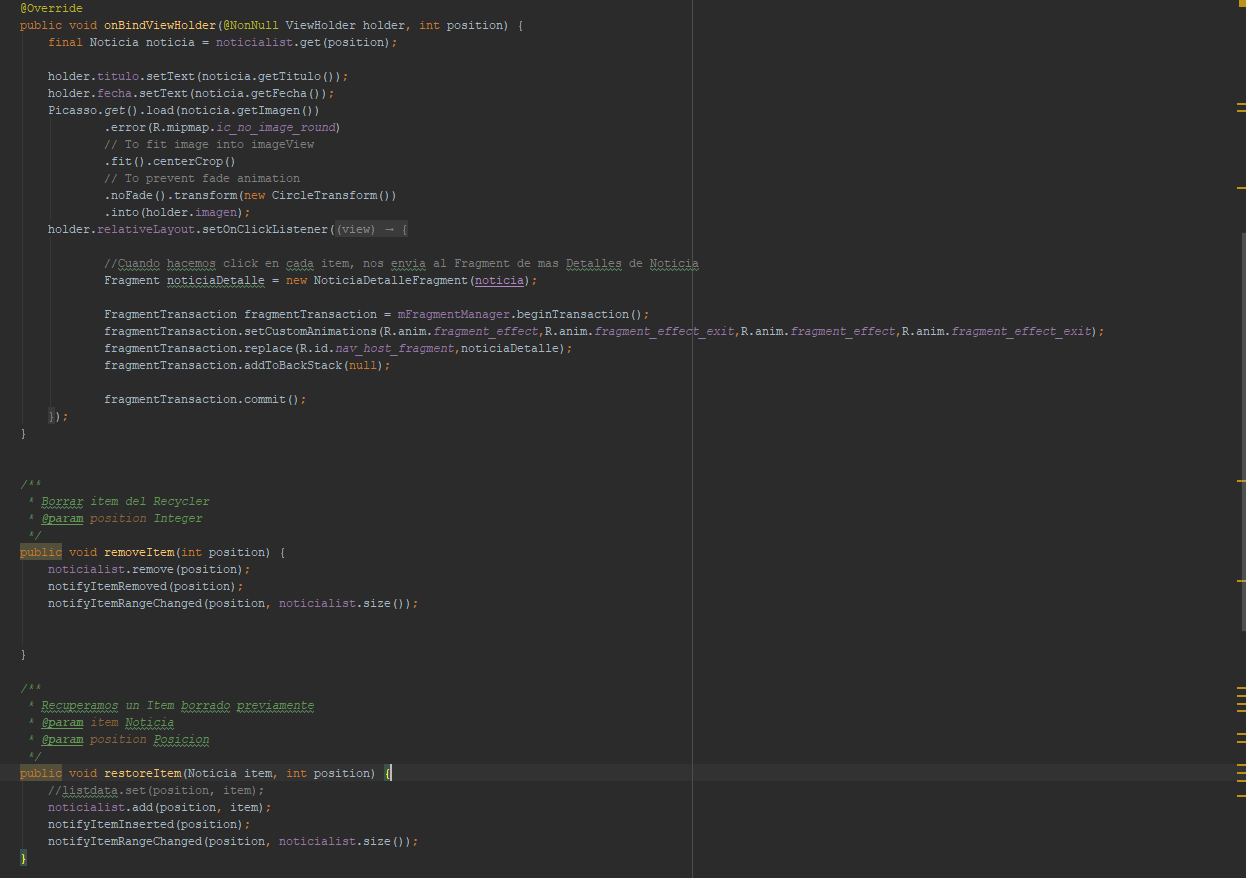
Una vez recogida la informacion, la instanciamos en una clase especifica de Noticia con un bucle for que una vez instanciada la noticia en la que se ha posicionado el nodo, la agregamos a un ArrayList de Noticias

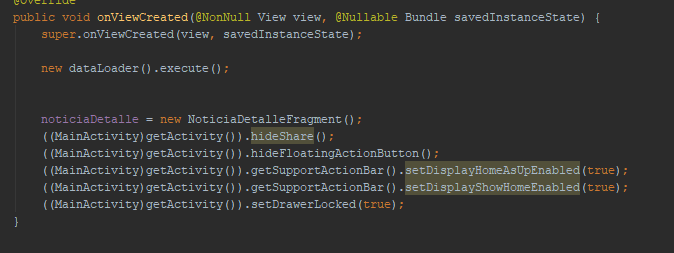
  
Ya terminado el manejo de la informacion e instanciarlo en un ArrayList, en onPostExecute, si no ha ocurrido ningun error y hemos podido recoger satisfactoriamente los datos(Si esto no es asi, se mostrara un mensaje al usuario), los implementamos en el RecyclerView a traves de nuestro adaptador

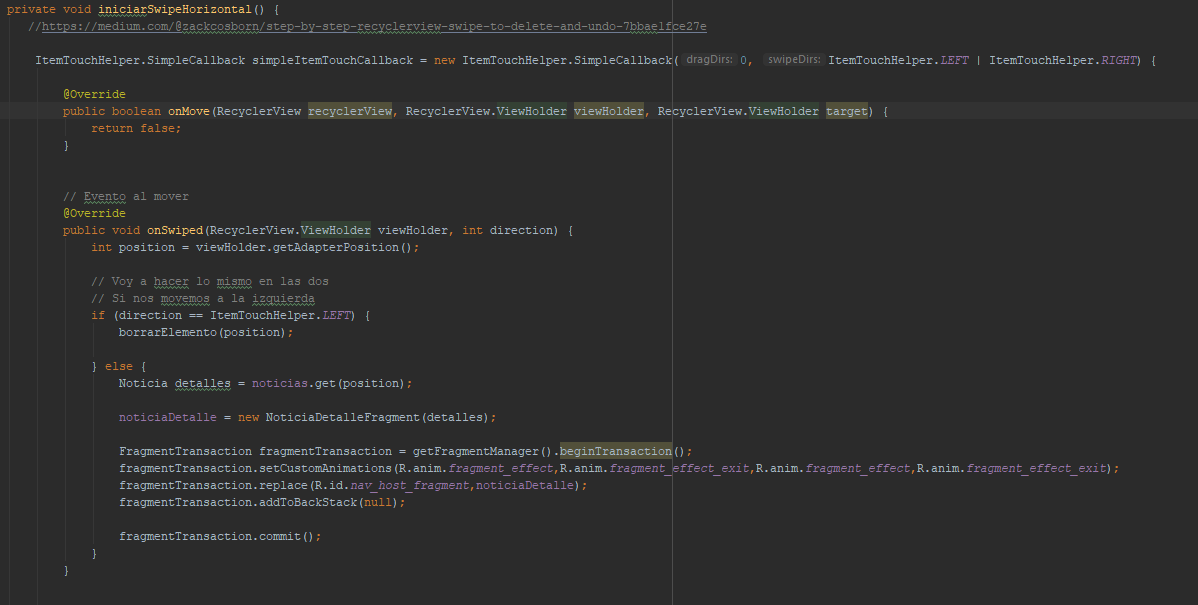
Vamos a hacer un inciso para explicar el adaptador, como se ha diseñado e implementado:

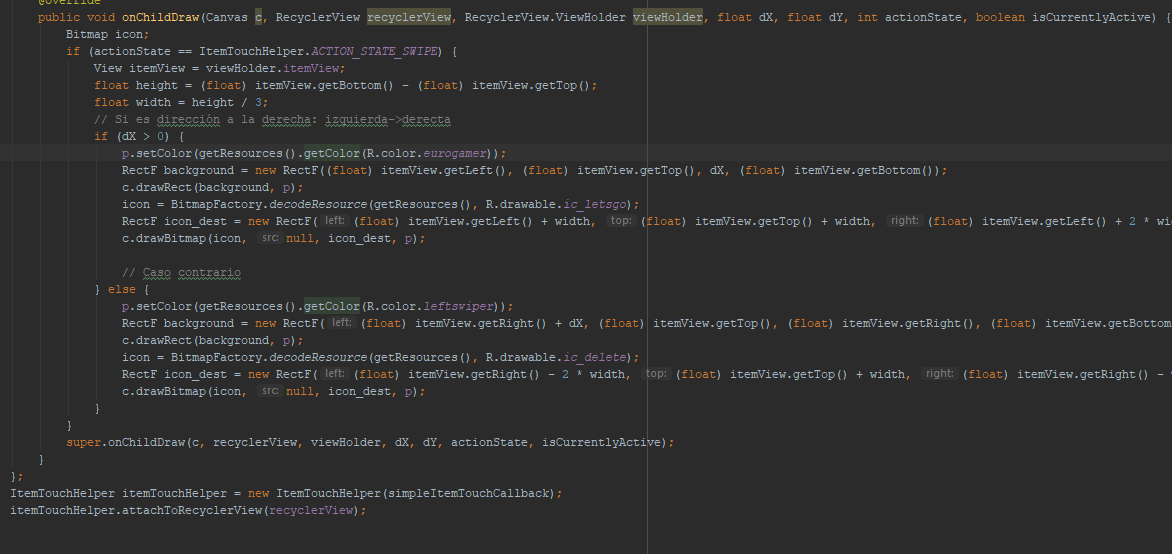
Lo primero, una vez creado el ViewHolder, instanciamos en su constructor todos los items del layout que compone cada item

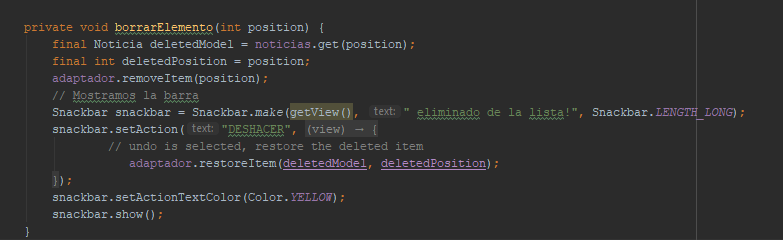
Dicho layout se compone de un ImageView y dos TextView, uno para el titulo y otro para la fecha  
  
Ahora que tenemos el ViewHolder, en los metodos definidos en el Adaptador tenemos el onBindViewHolder para setear todos los items con los datos, otro para borrar un item al realizar una accion y por ultimo para restaurarlo

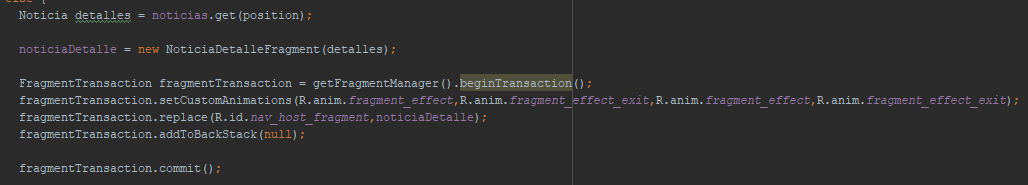


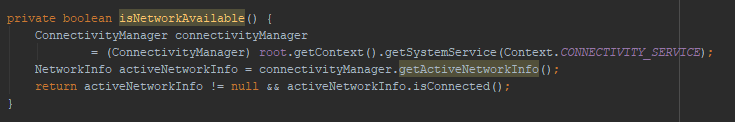
Volviendo al Fragment de Rss y terminando con el onViewCreated, usando metodos de la actividad principal para ocultar el menu y volver a habilitar el navigation drawer

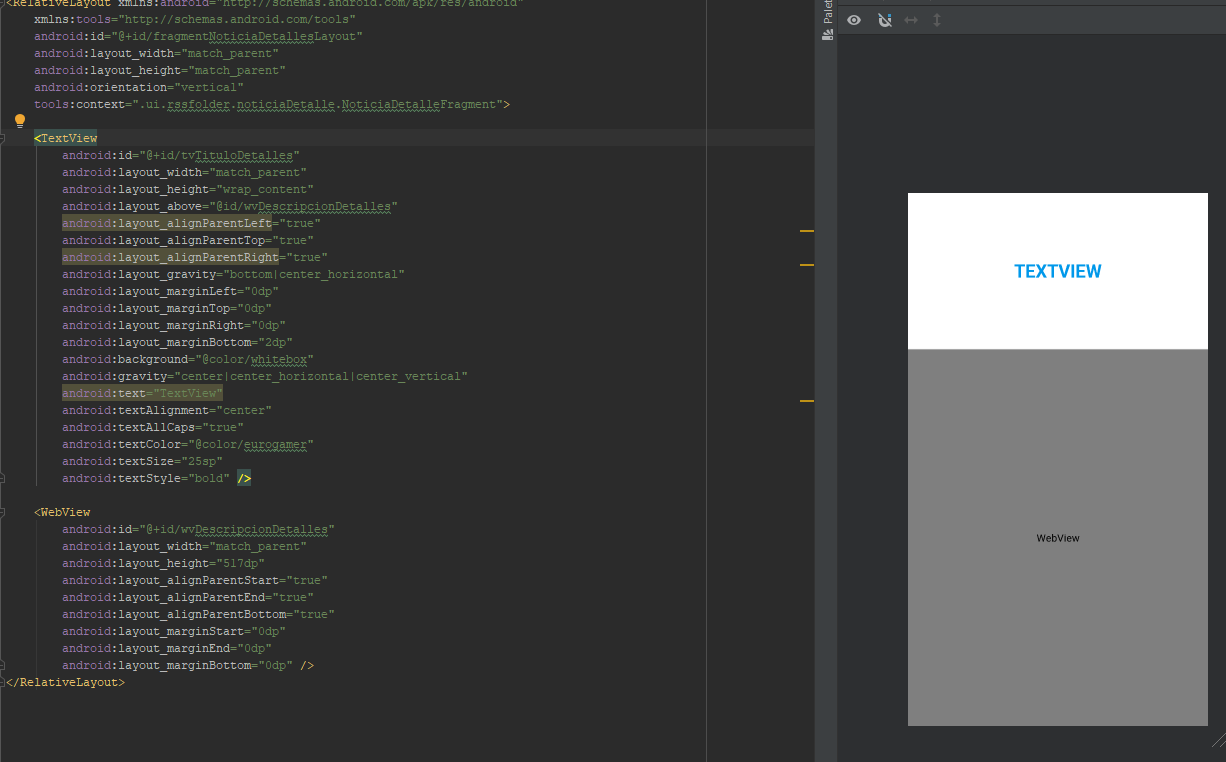
A continuacion, definimos metodos que usamos en esta misma clase para las funciones de Swipe lateral:

Recogemos el tipo de movimiento con las coordenadas para cada accion:

Y dichas acciones son borrar o llevar al fragment de la noticia detallada

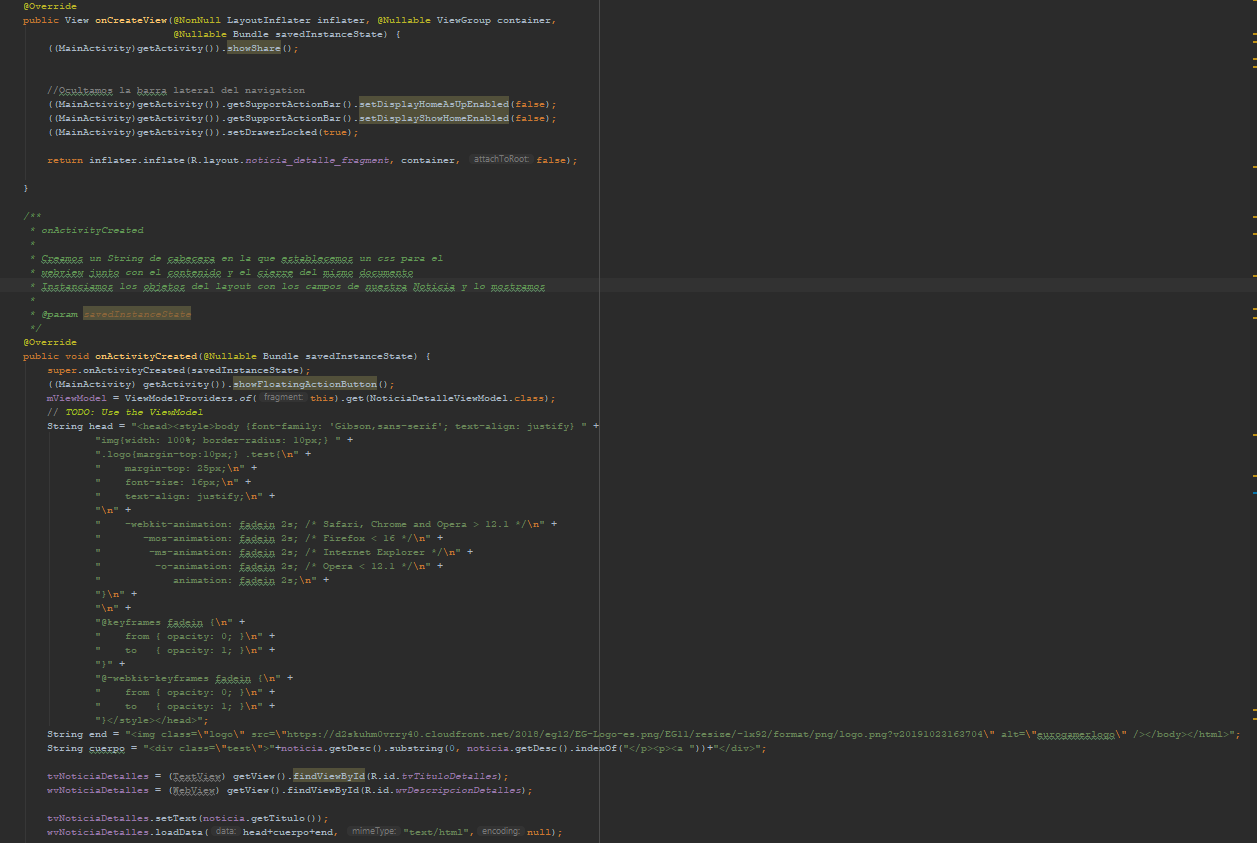
Para ejecutar el fragment de noticia detallada usamos estos metodos y clases implementadas, gracias a Fragment Manager podemos llevar una gestion correcta de dichas vistas

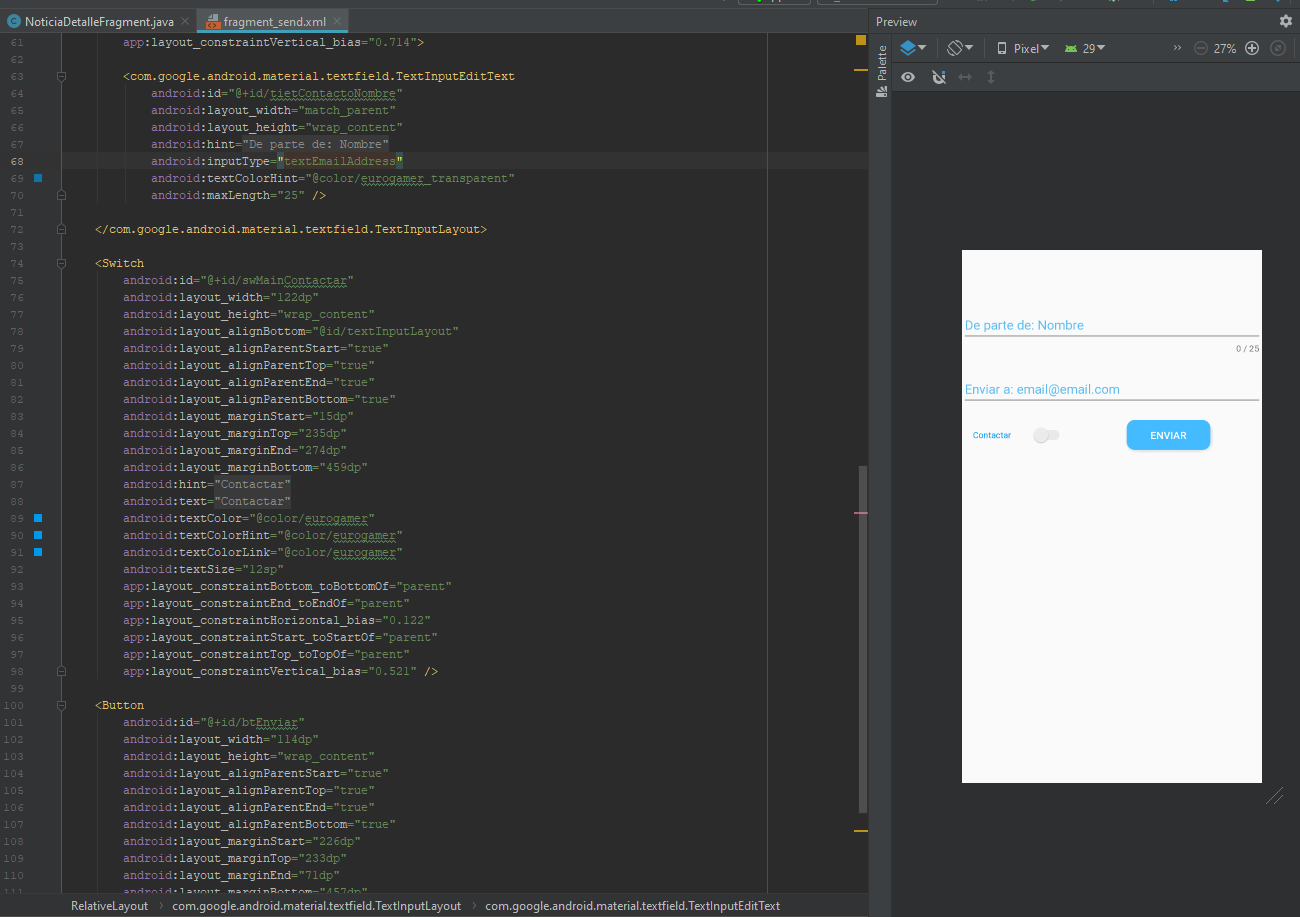
Por ultimo, este fragment usara un ultimo metodo para comprobar el estado de la conexión el cual tambien llamaremos desde el ASyncTask

* 1. Noticia Detalle Fragment
     1. Layout

Para abrir cada noticia en detalle, en este fragment hemos definidio un TextView para el titulo y un WebView para interpretar cual HTML el contenido de la descripcion. Para asignar este contenido lo explicare en el codigo del fragment

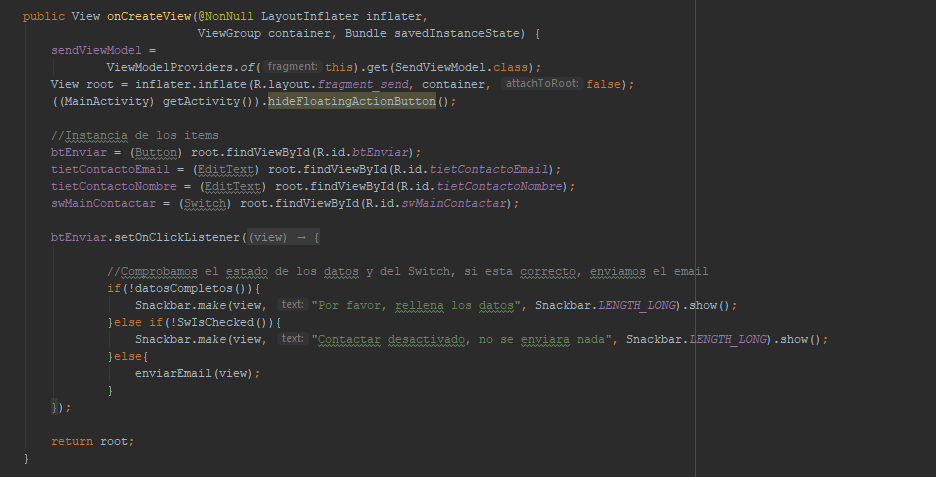
* + 1. Codigo

Este codigo es mas simple, empezando por el onCreateView, llamamos a los metodos que habiamos definido en el MainActivity.class para ocultar el menu de navegacion y mostrar nuestro menu personalizado para poder compartir o volver al fragment anterior, a su vez mostramos el Float Action Button para que el usuario, al hacer click, pueda ir a la noticia en su navegador

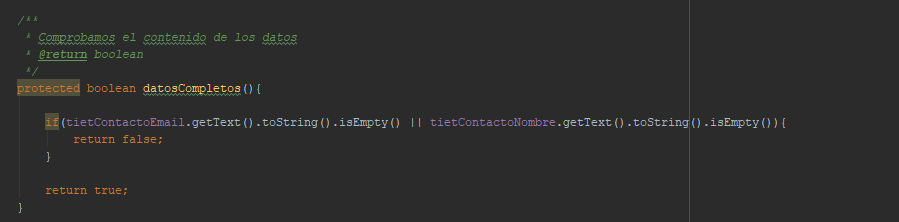
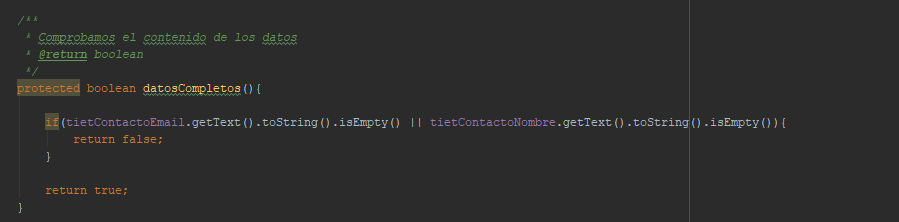
* 1. Contacto Fragment
     1. Layout

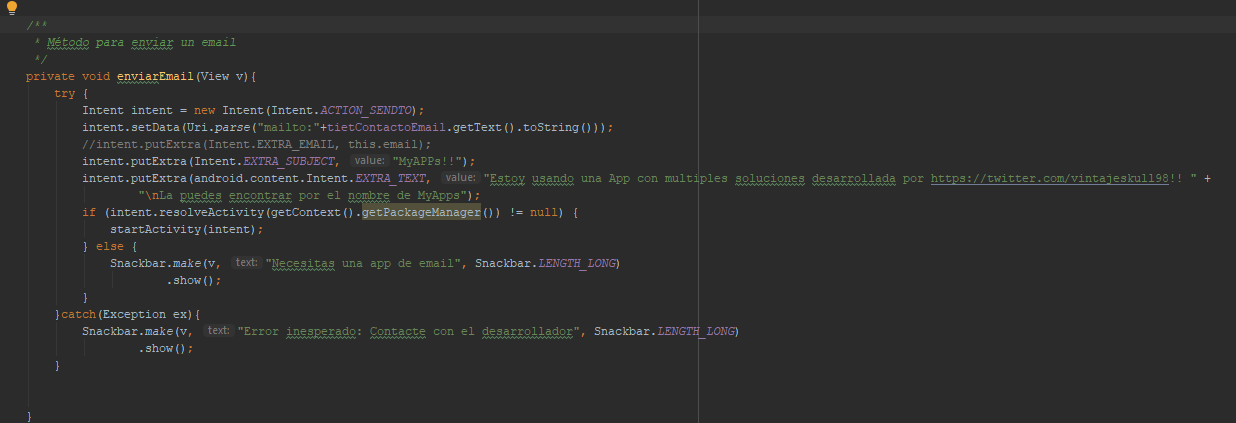
Un layout simple, compuesto por 2 TextInputLayout que muestran un Hint personalizado con el color principal de la App y en el caso del nombre, un contador maximo de caracteres. A esto se le añade un switch para que el usuario elija si contactar y el boton para Enviar

* + 1. Codigo

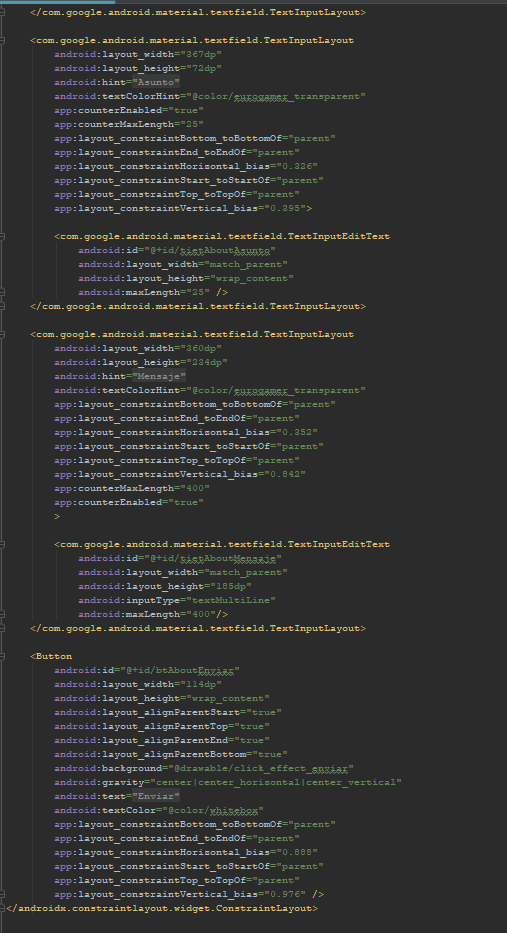
Al igual que el fragment anterior, es mas simple, con un onCreateView para inicializar todos los elementos y el Listener del boton para enviar.

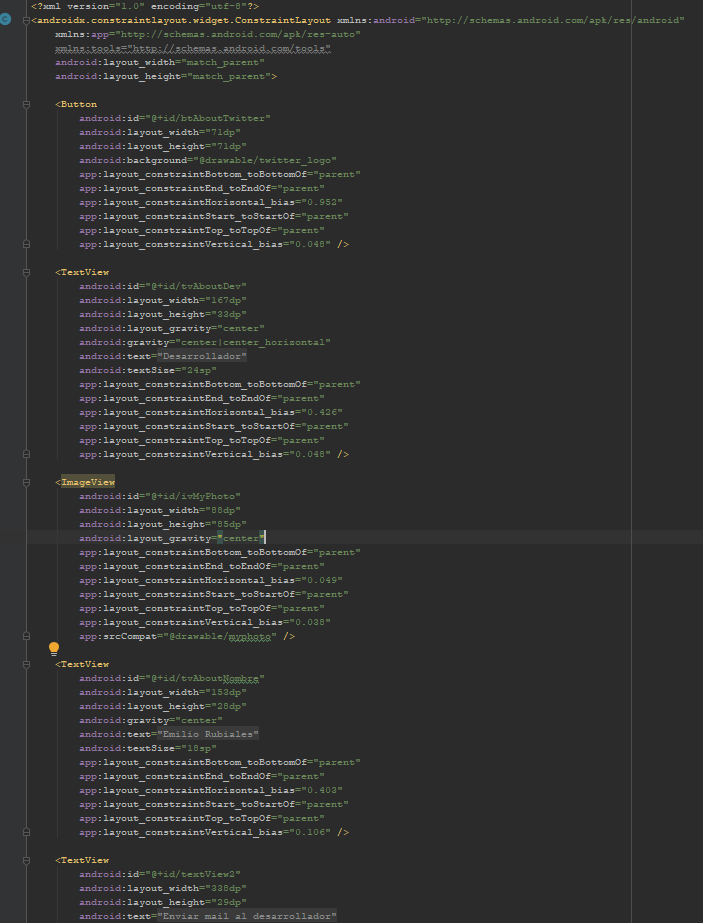
Para validar los datos usamos nuestra propia funcion que devuelve un Boolean en funcion de estos datos

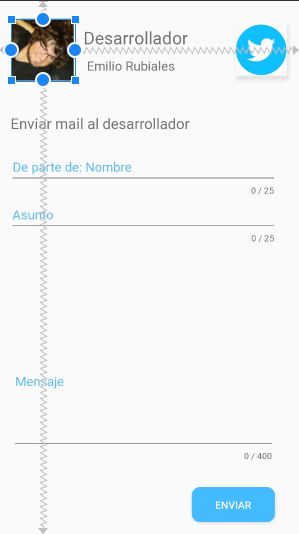
Tambien, el control del switch lo tenemos reflejado en una funcion:

Y por ultimo, la accion de enviar un Mail que, si no tienes aplicación para ello, te indica un mensaje de error en un SnackBar

* 1. About Us Fragment
     1. Layout

Por ultimo pero no menos importante, un layout con mencion al Desarrollador con su foto y nombre y varios campos de Textos para que el usuario pueda enviarle un correo con un mensaje por si quiere informar de un error o simplemente agradecer por la APP

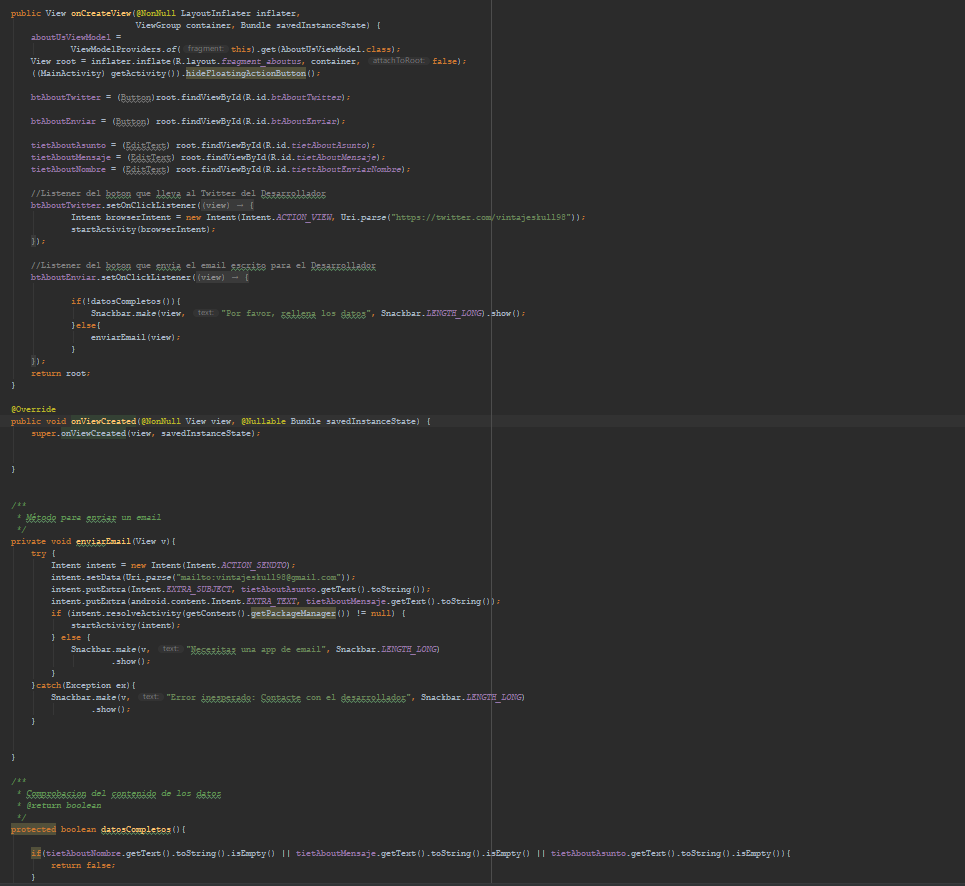




Para los campos de textos hemos puesto un limite en Asunto y Nombre de 25 caracteres y de 400 para el mensaje, para que no se extenda demasiado, acompañado de un boton para enviar a traves de su APP de gestion de E-mail

* + 1. Codigo

Al igual que el fragment de Contacto, el codigo es muy simple, usando practicamente los mismos metodos pero con distintos datos, validamos lo necesario y lanza la app de gestion de E-Mail del usuario para enviar el correo, a su vez, en el icono de Twitter al lado del nombre del Desarrollador puede ir a su Perfil simplemente haciendo click en el

Aqui concluye la documentacion de la nueva App MyApp la cual iremos expandiendo con multiples funciones para los usuarios.

Esta App y dicha documentacion ha sido desarrollada por Emilio Rubiales Gutierrez a quien se le otorgan todos los derechos de autor .

¡Espero que os haya sido util dicho documento!