

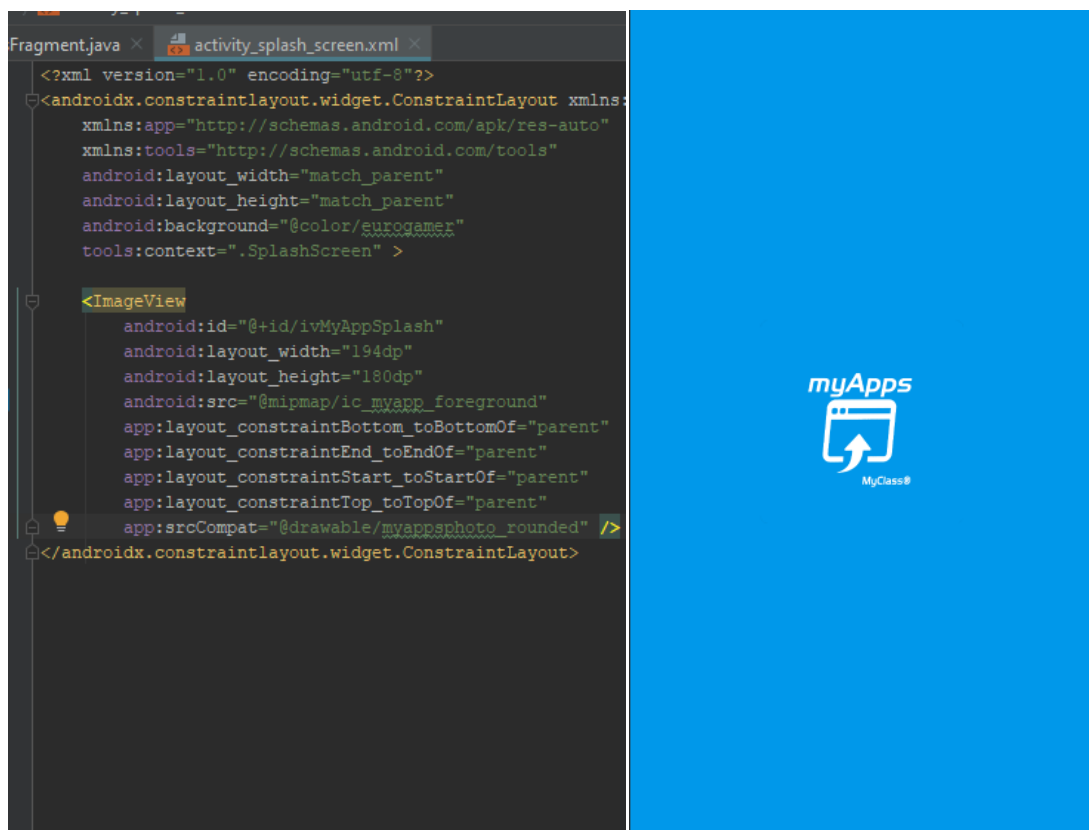
Desarrollo App en Android

App: *MyApps*

1. Splash Screen

a. 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.



b. Código

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.

```
package com.emiliorgvintage.myrss;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

public class SplashScreen extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash_screen);

        //Creamos un Handler que maneja el tiempo de muestra del SplashScreen
        new Handler().postDelayed(() -> {
            SplashScreen.this.finish();
            Intent intent = new Intent( packageContext: SplashScreen.this,MainActivity.class);
            startActivity(intent);
        }, delayMillis: 2000);
    }
}
```

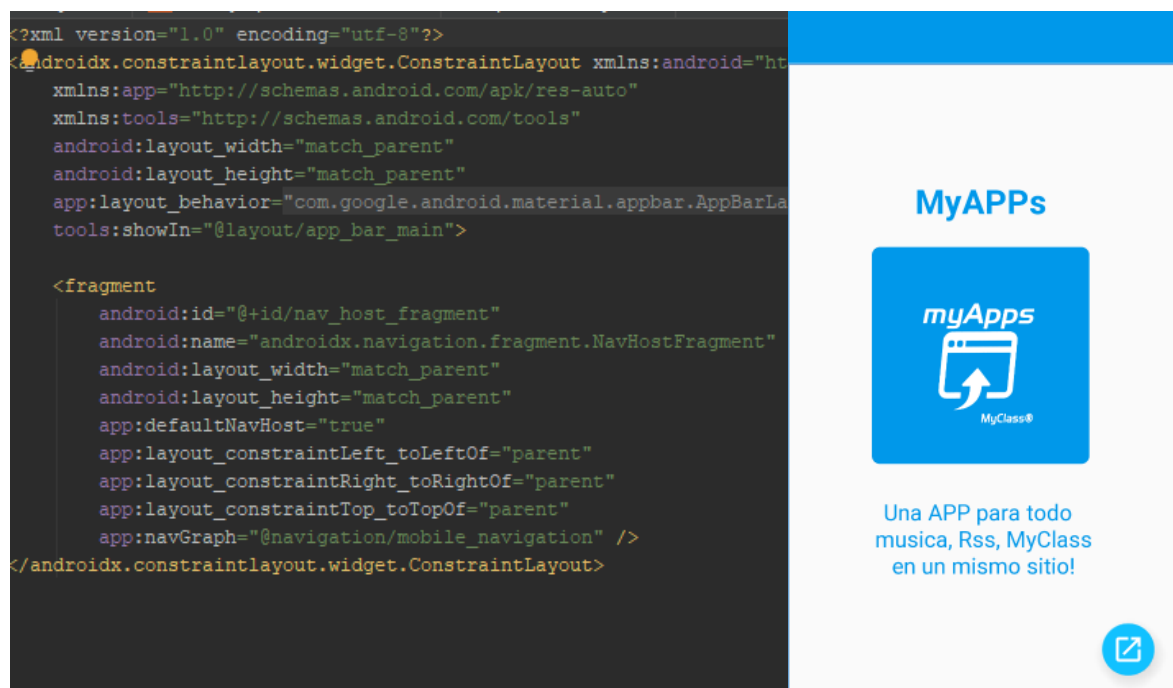
2. MainActivity

a. Layout

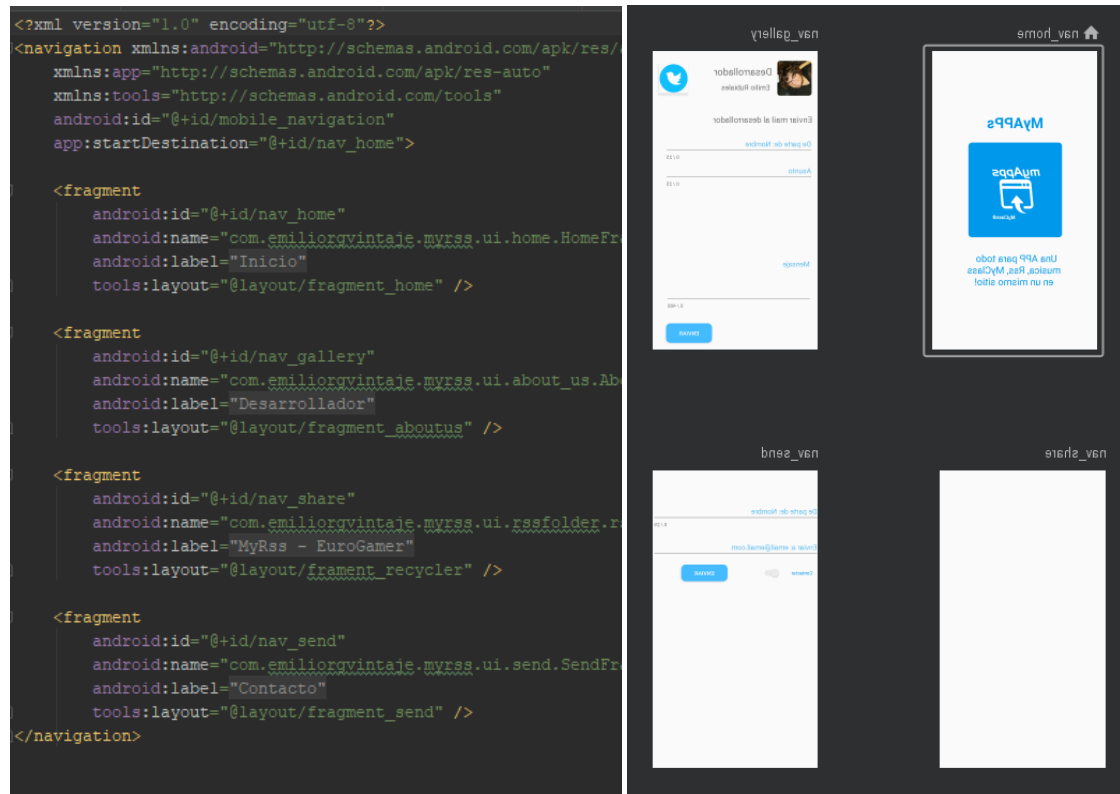
Layout simple que da pie al navigation drawer mostrando dicho menú en pantalla y un include en el cual le pasamos el content_main que se explicará.



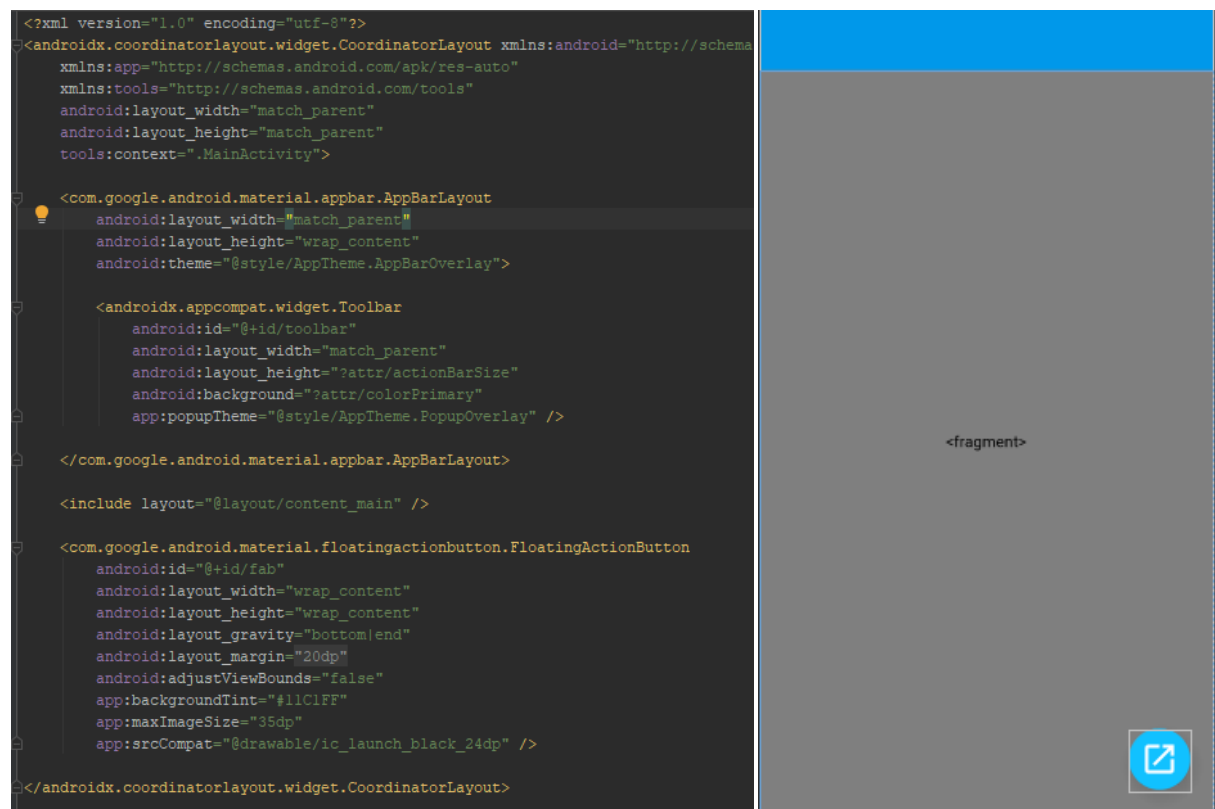
Viendo el Content Main en el que incluimos el fragment que hosteará el resto de fragments que se han diseñado.



En el navigation que gestiona los fragments que se han diseñado, encontramos esta distribución 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



b. Código

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

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

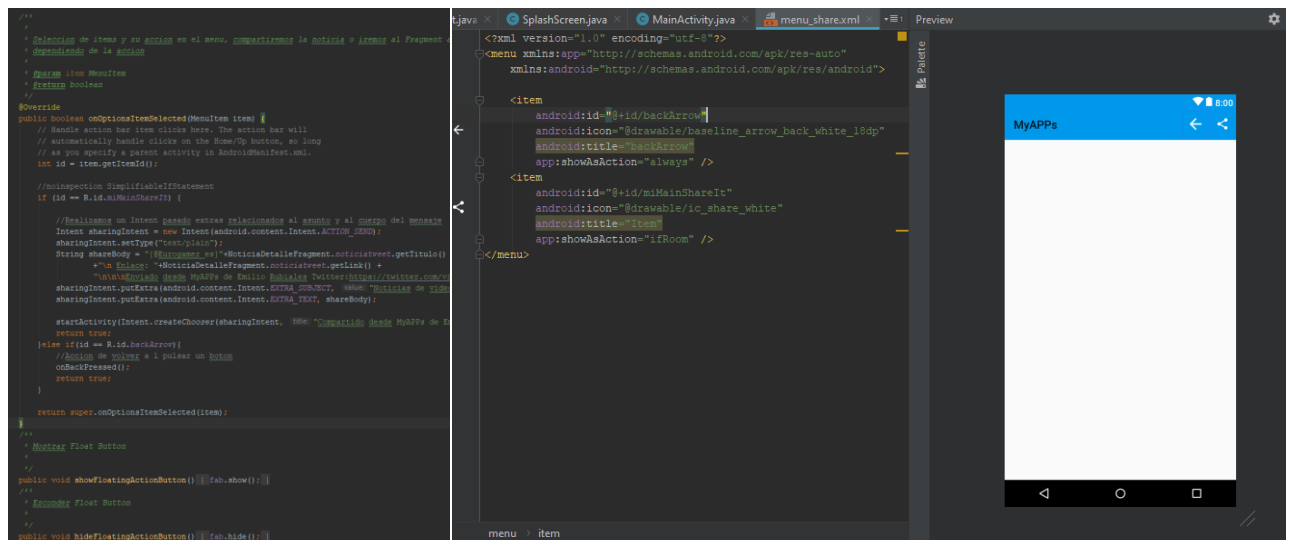
    setContentView(R.layout.activity_main);
    toolbar = findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);

    fab = (FloatingActionButton) findViewById(R.id.fab);
    drawer = findViewById(R.id.drawer_layout);
    NavigationView navigationView = findViewById(R.id.nav_view);
    // Passing each menu ID as a set of IDs because each
    // menu should be considered as top level destinations.
    mAppBarConfiguration = new AppBarConfiguration.Builder(
        R.id.nav_home, R.id.nav_gallery,
        R.id.nav_share, R.id.nav_send)
        .setDrawerLayout(drawer)
        .build();
    //GRUPO MYCLASS
    NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment);
    NavigationUI.setupActionBarWithNavController(this, navController, mAppBarConfiguration);
    NavigationUI.setupWithNavController(navigationView, navController);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick() {
            String tweetUrl = NoticiaDetalleFragment.noticiastweet.getLink();
            Uri uri = Uri.parse(tweetUrl);
            startActivity(new Intent(Intent.ACTION_VIEW, uri));
        }
    });
}

/**
 * Bloqueamos el navigation drawer, usado en el fragment de mas Detalles de ca
 * NoticiaDetalleFragment.class
 * y en rssFragment.class para habilitar y deshabilitar
 * @param enabled verdadero = bloqueado; falso = desbloqueado
 */
public void setDrawerLocked(boolean enabled) {
    if(enabled) {
        drawer.setDrawerLockMode(DrawerLayout.LOCK_MODE_LOCKED_CLOSED);
    } else {
        drawer.setDrawerLockMode(DrawerLayout.LOCK_MODE_UNLOCKED);
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    /**
     * Mostramos el menu de compartir usado en el fragment de mas Detalles
     * de cada noticia: NoticiaDetalleFragment.class
     */
}
```

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.

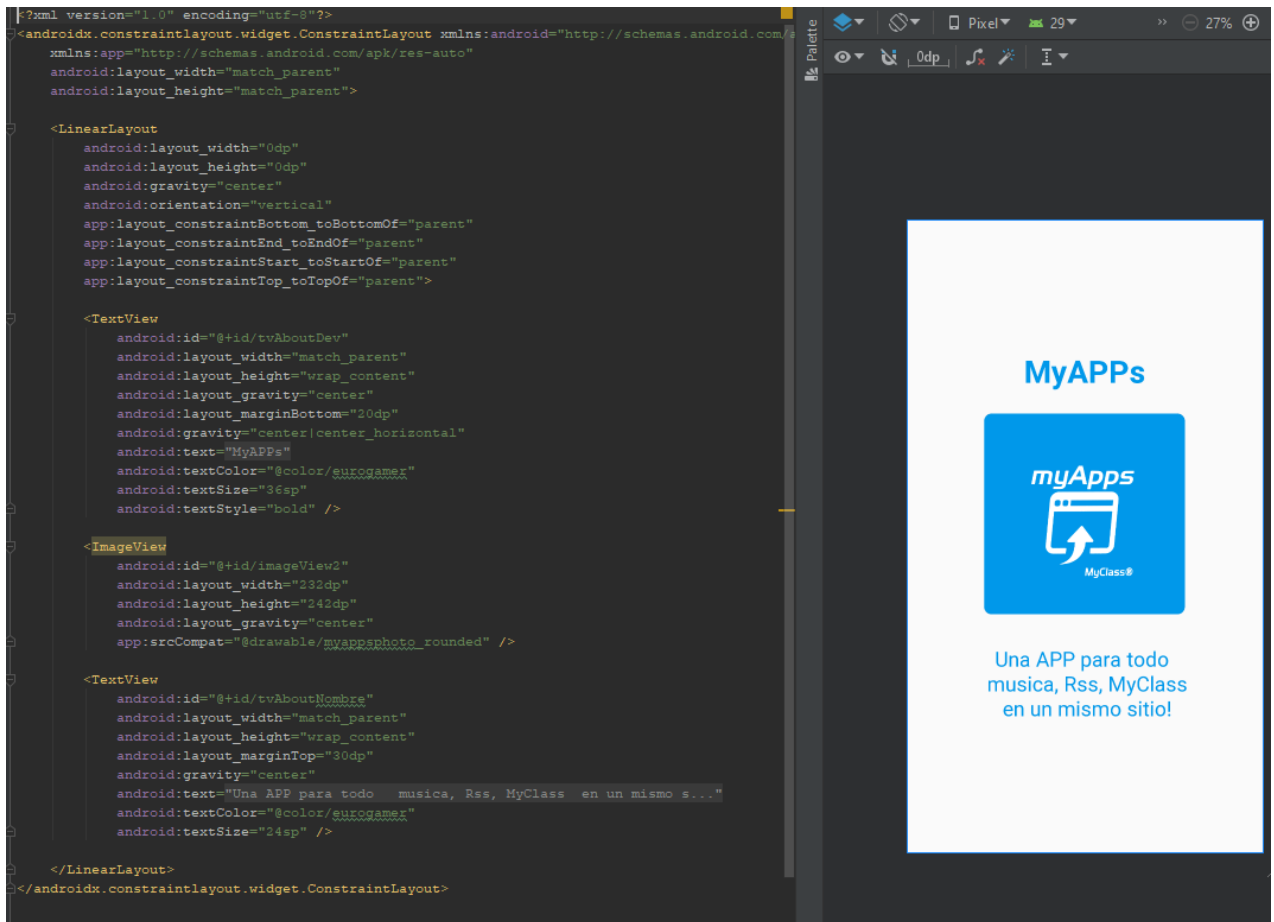


3. Fragments

a. Home Layout

i. 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



ii.Codigo

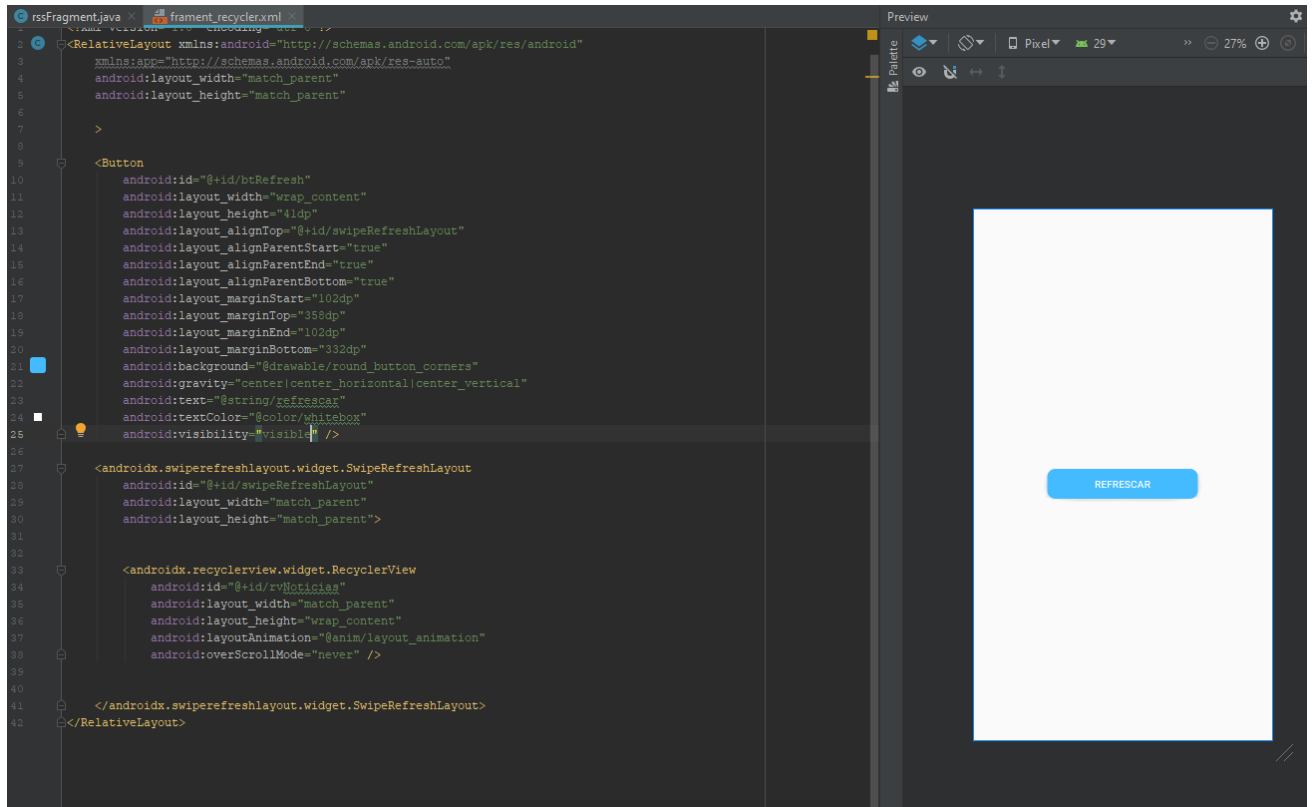
Una clase sencilla en la cual instanciamos los ítems del layout y escondiendo, con el método descrito en el Main Activity, el botón flotante.

```
public class HomeFragment extends Fragment {  
  
    private HomeViewModel homeViewModel;  
  
    public View onCreateView(@NonNull LayoutInflater inflater,  
                             ViewGroup container, Bundle savedInstanceState) {  
  
        homeViewModel =  
            ViewModelProviders.of(fragment: this).get(HomeViewModel.class);  
        View root = inflater.inflate(R.layout.fragment_home, container, attachToRoot: false);  
  
        //Escondemos el float Button  
        ((MainActivity) getActivity()).hideFloatingActionButton();  
  
        return root;  
    }  
}
```

b. RssFragment

i. 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



ii.Codigo

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

```
private RssViewModel rssViewModel;

private RecyclerView recyclerView;
private NoticiasAdaptador adaptador;
private ArrayList<Noticia> noticias;
private String path = "https://europasemr.es/?format=rss";
private View root;
private SwipeRefreshLayout swipeRefreshLayout;
private Paint p = new Paint();
private Button btRefresh;
private Fragment noticiaDetalle;

public View onCreateView(@NonNull LayoutInflater inflater,
    ViewGroup container, Bundle savedInstanceState) {
    rssViewModel =
        ViewModelProviders.of(this).get(RssViewModel.class);
    root = inflater.inflate(R.layout.fragment_recycler, container, false);

    iniciarSwipeRecargar();
    recyclerView = (RecyclerView) root.findViewById(R.id.rvNoticias);
    recyclerView.setLayoutManager(new LinearLayoutManager(root.getContext()));
    swipeRefreshLayout.setRefreshing(true);
    btRefresh = (Button) root.findViewById(R.id.btRefresh);
    btRefresh.setVisibility(View.INVISIBLE);
    iniciarSwipeHorizontal();

    int resId = R.anim.layout_animation;
    LayoutAnimationController animator = AnimationUtils.loadLayoutAnimation(root.getContext(), resId);
    recyclerView.setLayoutAnimation(animator);

    btRefresh.setOnClickListener(new View.OnClickListener() {
        // TODO Auto-generated method stub
        new dataLoader().execute();
    });

    if(!isNetworkAvailable() || noticias == null){
        btRefresh.setVisibility(View.VISIBLE);
    }

    return root;
}
```

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.

```
private void runLayoutAnimation(final RecyclerView recyclerView) {
    final Context context = recyclerView.getContext();
    final LayoutAnimationController controller =
        AnimationUtils.loadLayoutAnimation(context, R.anim.layout_animation);

    recyclerView.setLayoutAnimation(controller);
    adaptador.notifyDataSetChanged();
    recyclerView.scheduleLayoutAnimation();
}

private void iniciarSwipeRecargar() {

    swipeRefreshLayout = (SwipeRefreshLayout) root.getRootView().findViewById(R.id.swipeRefreshLayout);
    swipeRefreshLayout.setOnRefreshListener(() - {

        swipeRefreshLayout.setColorSchemeResources(R.color.eurogamer);
        swipeRefreshLayout.setProgressBackgroundColorSchemeResource(R.color.whitebox);

        new dataLoader().execute();

    });
}
```

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

Ejecutando en onCreateView() 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.

```
@Override
public void onCreateView(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    new dataLoader().execute();

    noticiaDetalle = new NoticiaDetalleFragment();
    ((MainActivity) getActivity()).hideShare();
    ((MainActivity) getActivity()).hideFloatingActionButton();
    ((MainActivity) getActivity()).getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    ((MainActivity) getActivity()).getSupportActionBar().setDisplayShowHomeEnabled(true);
    ((MainActivity) getActivity()).setDrawerLocked(true);
}
```

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

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    swipeRefreshLayout.setRefreshing(true);
    if(!isNetworkAvailable()){

        this.cancel( mayInterruptifRunning: true);

        ((Activity) root.getContext()).runOnUiThread(() - {
            Toast.makeText(root.getContext(), text: "No hay Conexion", Toast.LENGTH_SHORT).show();
        });

        swipeRefreshLayout.setRefreshing(false);
    }
}
```

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

```
@Override
protected Void doInBackground(Void... strings) {

    try {
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = null;
        try {
            builder = factory.newDocumentBuilder();
        } catch (ParserConfigurationException e) {
            e.printStackTrace();
        }
        Document document = null;
        try {
            document = builder.parse(path);
        } catch (IOException e) {
            e.printStackTrace();
        } catch (SAXException e) {
            e.printStackTrace();
        }

        ArrayList<Noticia> newsletter = new ArrayList<>();

        NodeList lItems = document.getElementsByTagName("item");

        for (int i = 0; i < 20; i++) {
            Node node = lItems.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {

                Element element = (Element) node;

                String img = element.getElementsByTagName("description").item( index: 0).getTextContent();
                img = img.substring(img.indexOf("<img src=\"") + 10, img.indexOf("\" alt=\"\">"));
            }
        }
    }
}
```


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

```
for (int i = 0; i < 20; i++) {
    Node node = lItems.item(i);
    if (node.getNodeType() == Node.ELEMENT_NODE) {

        Element element = (Element) node;

        String img = element.getElementsByTagName("description").item( index: 0).getTextContent();
        img = img.substring(img.indexOf("<img src=\"") + 10, img.indexOf("\" alt=\"\"/>"));

        newsletter.add(new Noticia(element.getElementsByTagName("title").item( index: 0).getTextContent(),
            element.getElementsByTagName("description").item( index: 0).getTextContent(),
            element.getElementsByTagName("link").item( index: 0).getTextContent(),
            element.getElementsByTagName("pubDate").item( index: 0).getTextContent().substring(
                index: 0, index: 100
            ),
            img
        ));
    }
}

noticias = newsletter;

} catch (Exception ex) {

}

return null;
}
```

```
public class Noticia {
    private String titulo, desc, link, fecha, imagen;

    /**
     * Clase de nuestro item del RecyclerView
     *
     * @param titulo String
     * @param desc String
     * @param link String
     * @param fecha String
     * @param imagen String
     */
    public Noticia(String titulo, String desc, String link, String fecha, String imagen) {
        this.titulo = titulo;
        this.desc = desc;
        this.link = link;
        this.fecha = fecha;
        this.imagen = imagen;
    }

    public String getTitulo() { return titulo; }

    public void setTitulo(String titulo) { this.titulo = titulo; }

    public String getDesc() { return desc; }

    public void setDesc(String desc) { this.desc = desc; }

    public String getLink() { return link; }

    public void setLink(String link) { this.link = link; }

    public String getFecha() { return fecha; }

    public void setFecha(String fecha) { this.fecha = fecha; }

    public String getImagen() { return imagen; }

    public void setImagen(String imagen) { this.imagen = imagen; }
}
```

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

```
@Override
protected void onPostExecute(Void aVoid) {
    super.onPostExecute(aVoid);

    try{
        noticias.size();

        btRefrescar.setVisibility(View.INVISIBLE);
        adaptador = new NoticiaAdaptador(noticias, getFragmentManager(), recyclerView);
        recyclerView.setAdapter(adaptador);

        runLayoutAnimation(recyclerView);
        recyclerView.setHasFixedSize(true);
        swipeRefreshLayout.setRefreshing(false);
    } catch (Exception ex) {
        swipeRefreshLayout.setRefreshing(false);
        btRefrescar.setVisible(true);
    }
}
```

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

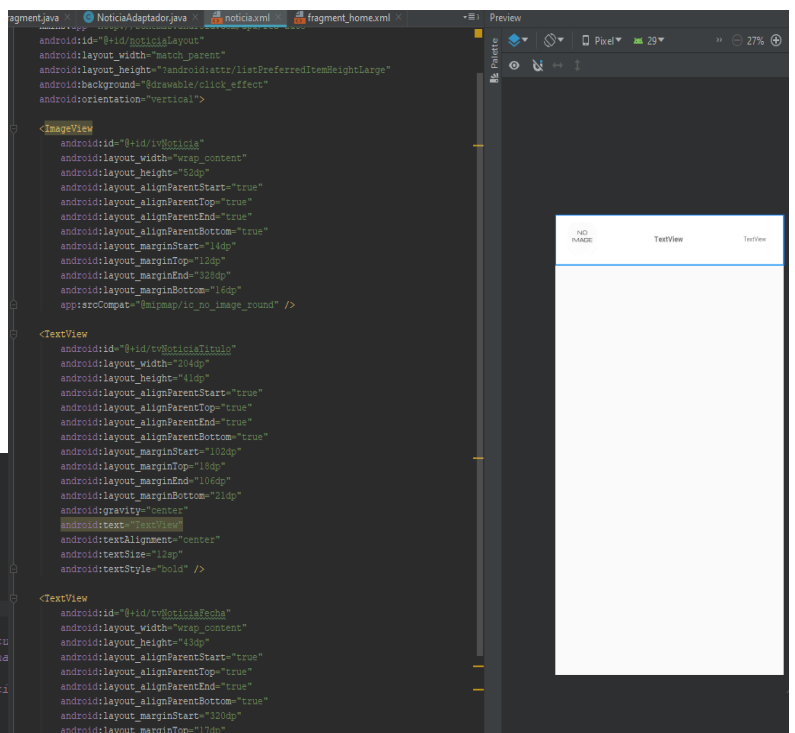
Lo primero, una vez creado el ViewHolder, instanciamos en su constructor todos los items del layout que compone cada item

Dicho layout se compone de un ImageView y dos TextView, uno para el titulo y otro para la fecha

```
public class ViewHolder extends RecyclerView.ViewHolder {
    public ImageView imagen;
    public TextView titulo;
    public TextView fecha;

    public RelativeLayout relativeLayout;
    public ViewHolder(@NonNull View itemView) {
        super(itemView);
        //Iniciamos los objetos del layout de cada item
        this.imagen = (ImageView) itemView.findViewById(R.id.ivNoticia);
        this.titulo = (TextView) itemView.findViewById(R.id.tvNoticiaTitulo);
        this.fecha = (TextView) itemView.findViewById(R.id.tvNoticiaFecha);

        relativeLayout = (RelativeLayout) itemView.findViewById(R.id.noti
    }
}
```



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

```
@Override
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
    final Noticia noticia = noticiast.get(position);

    holder.titulo.setText(noticia.getTitulo());
    holder.fecha.setText(noticia.getFecha());
    Picasso.get().load(noticia.getImagen())
        .error(R.mipmap.ic_no_image_round)
        .fit().centerCrop()
        .noFade().transform(new CircleTransform())
        .into(holder.imagen);
    holder.relativeLayout.setOnClickListener((view) -> {

        //Cuando hacemos click en cada item, nos envia al Fragment de mas Detalles de Noticia
        Fragment noticiaDetalle = new NoticiaDetalleFragment(noticia);

        FragmentTransaction fragmentTransaction = mFragmentManager.beginTransaction();
        fragmentTransaction.setCustomAnimations(R.anim.fragment_effect, R.anim.fragment_effect_exit, R.anim.fragment_effect, R.anim.fragment_effect_exit);
        fragmentTransaction.replace(R.id.nav_host_fragment, noticiaDetalle);
        fragmentTransaction.addToBackStack(null);

        fragmentTransaction.commit();

    });

    /**
     * Borrag item del Recycler
     * @param position Integer
     */
    public void removeItem(int position) {
        noticiast.remove(position);
        notifyItemRemoved(position);
        notifyItemRangeChanged(position, noticiast.size());
    }

    /**
     * Recuperamos un item borrado previamente
     * @param item Noticia
     * @param position Posicion
     */
    public void restoreItem(Noticia item, int position) {
        //listData.set(position, item);
        noticiast.add(position, item);
        notifyItemInserted(position);
        notifyItemRangeChanged(position, noticiast.size());
    }
}
```

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

```
@Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    new dataLoader().execute();

    noticiaDetalle = new NoticiaDetalleFragment();
    ((MainActivity) getActivity()).hideShare();
    ((MainActivity) getActivity()).hideFloatingActionButton();
    ((MainActivity) getActivity()).getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    ((MainActivity) getActivity()).getSupportActionBar().setDisplayShowHomeEnabled(true);
    ((MainActivity) getActivity()).setDrawerLocked(true);
}
```

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

```
private void iniciarSwipeHorizontal() {
    //https://medium.com/@stacksofborn/step-by-step-recyclerview-swipe-to-delete-and-undo-7bbaefce27e
    ItemTouchHelper.SimpleCallback simpleItemTouchCallback = new ItemTouchHelper.SimpleCallback( dragDirs: 0, swipeDirs: ItemTouchHelper.LEFT | ItemTouchHelper.RIGHT) {

        @Override
        public boolean onMove(RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder, RecyclerView.ViewHolder target) {
            return false;
        }

        // Evento al mover
        @Override
        public void onSwiped(RecyclerView.ViewHolder viewHolder, int direction) {
            int position = viewHolder.getAdapterPosition();

            // Voy a hacer lo mismo en las dos
            // Si nos movemos a la izquierda
            if (direction == ItemTouchHelper.LEFT) {
                borrarElemento(position);
            } else {
                Noticia detalles = noticias.get(position);

                noticiaDetalle = new NoticiaDetalleFragment(detalles);

                FragmentTransaction fragmentTransaction = getFragmentManager().beginTransaction();
                fragmentTransaction.setCustomAnimations(R.anim.fragment_effect, R.anim.fragment_effect_exit, R.anim.fragment_effect, R.anim.fragment_effect_exit);
                fragmentTransaction.replace(R.id.nav_host_fragment, noticiaDetalle);
                fragmentTransaction.addToBackStack(null);

                fragmentTransaction.commit();
            }
        }
    };
}
```

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

```
@Override
public void onChildDraw(Canvas c, RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder, float dX, float dY, int actionState, boolean isCurrentlyActive) {
    Bitmap icon;
    if (actionState == ItemTouchHelper.ACTION_STATE_SWIPE) {
        View itemView = viewHolder.itemView;
        float height = (float) itemView.getBottom() - (float) itemView.getTop();
        float width = height / 3;
        // Si es direccion a la derecha: izquierda->derecha
        if (dX > 0) {
            p.setColor(getResources().getColor(R.color.europamer));
            RectF background = new RectF((float) itemView.getLeft(), (float) itemView.getTop(), dX, (float) itemView.getBottom());
            c.drawRect(background, p);
            icon = BitmapFactory.decodeResource(getResources(), R.drawable.ic_delete);
            RectF icon_dest = new RectF((float) itemView.getLeft() + width, (float) itemView.getTop() + width, (float) itemView.getLeft() + 2 * width, (float) itemView.getBottom() - width);
            c.drawBitmap(icon, null, icon_dest, p);
        } else {
            p.setColor(getResources().getColor(R.color.leftsviper));
            RectF background = new RectF((float) itemView.getRight() + dX, (float) itemView.getTop(), (float) itemView.getRight(), (float) itemView.getBottom());
            c.drawRect(background, p);
            icon = BitmapFactory.decodeResource(getResources(), R.drawable.ic_delete);
            RectF icon_dest = new RectF((float) itemView.getRight() - 2 * width, (float) itemView.getTop() + width, (float) itemView.getRight(), (float) itemView.getBottom() - width);
            c.drawBitmap(icon, null, icon_dest, p);
        }
    }
    super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive);
}

ItemTouchHelper itemTouchHelper = new ItemTouchHelper(simpleItemTouchCallback);
itemTouchHelper.attachToRecyclerView(recyclerView);
```

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

```
private void borrarElemento(int position) {
    final Noticia deletedModel = noticias.get(position);
    final int deletedPosition = position;
    adaptador.removeItem(position);
    // Mostramos la barra
    Snackbar snackbar = Snackbar.make(getView(), text: " eliminado de la lista!", Snackbar.LENGTH_LONG);
    snackbar.setAction( text: "DESHACER", (view) -> {
        // undo is selected, restore the deleted item
        adaptador.restoreItem(deletedModel, deletedPosition);
    });
    snackbar.setActionTextColor(Color.YELLOW);
    snackbar.show();
}
```

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

```
Noticia detalles = noticias.get(position);

noticiaDetalle = new NoticiaDetalleFragment(detalles);

FragmentManager fragmentManager = getSupportFragmentManager();
fragmentTransaction.setCustomAnimations(R.anim.fragment_effect,R.anim.fragment_effect_exit,R.anim.fragment_effect,R.anim.fragment_effect_exit);
fragmentTransaction.replace(R.id.nav_host_fragment,noticiaDetalle);
fragmentTransaction.addToBackStack(null);

fragmentTransaction.commit();
```

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

```
private boolean isNetworkAvailable() {
    ConnectivityManager connectivityManager
        = (ConnectivityManager) root.getContext().getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo activeNetworkInfo = connectivityManager.getActiveNetworkInfo();
    return activeNetworkInfo != null && activeNetworkInfo.isConnected();
}
```

c. Noticia Detalle Fragment

i. Layout

Para abrir cada noticia en detalle, en este fragment hemos definido 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

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragmentNoticiaDetalleLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".ui.gestiones.noticiaDetalle.NoticiaDetalleFragment">

    <TextView
        android:id="@+id/tvTituloDetalles"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/wvDescripcionDetalles"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:layout_gravity="bottom|center_horizontal"
        android:layout_marginLeft="0dp"
        android:layout_marginTop="0dp"
        android:layout_marginRight="0dp"
        android:layout_marginBottom="0dp"
        android:background="@color/gui_solid"
        android:gravity="center|center_horizontal|center_vertical"
        android:text="@string/titulo"
        android:textAlignment="center"
        android:textAllCaps="true"
        android:textColor="@color/gui_solid"
        android:textSize="18sp"
        android:textStyle="bold" />

    <WebView
        android:id="@+id/wvDescripcionDetalles"
        android:layout_width="match_parent"
        android:layout_height="517dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="0dp"
        android:layout_marginEnd="0dp"
        android:layout_marginBottom="0dp" />
</RelativeLayout>
```

ii. Código

Este código es más simple, empezando por el `onCreateView`, llamamos a los métodos que habíamos definido en el `MainActivity.class` para ocultar el menú de navegación y mostrar nuestro menú 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

```
//Override
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container,
    @Nullable Bundle savedInstanceState) {
    (MainActivity).getActivity().showShare();

    //Consultamos la barra lateral del navigation
    (MainActivity).getActivity().getSupportActionBar().setDisplayHomeAsUpEnabled(false);
    (MainActivity).getActivity().getSupportActionBar().setDisplayShowHomeEnabled(false);
    (MainActivity).getActivity().setDrawerLocked(true);

    return inflater.inflate(R.layout.noticia_detalle_fragment, container, attachToRoot: false);
}

/**
 * onActivityCreated
 *
 * Creamos un String de cabecera en la que almacenamos un css para el
 * navegador junto con el contenido y el id del main fragment
 * Almacenamos los objetos del layout con los campos de nuestra Noticia y lo mostramos
 */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    (MainActivity).getActivity().showErrorMessage();
    mViewModel = ViewModelProviders.of(this).get(NoticiaDetalleViewModel.class);
    // TODO: Use the ViewModel
    String head = "<head><style>body {font-family: 'Gibson,sans-serif'; text-align: justify;} " +
        "<img(width= 100%; border-radius: 10px; " +
        "<log(margin-top:10px); test\\n\" +
        "< margin-top: 20px;\\n\" +
        "< font-size: 16px;\\n\" +
        "< text-align: justify;\\n\" +
        "<\\n\" +
        "< -webkit-animation: $@{anim} 2s; /* Safari, Chrome and Opera > 12.1 */\\n\" +
        "< -ms-animation: $@{anim} 2s; /* Firefox < 16 */\\n\" +
        "< -ms-animation: $@{anim} 2s; /* Internet Explorer > \\n\" +
        "< -o-animation: $@{anim} 2s; /* Opera < 12.1 */\\n\" +
        "< animation: $@{anim} 2s;\\n\" +
        "<\\n\" +
        "<\\n\" +
        "<@keyframes $@{anim} {\\n\" +
        "< from { opacity: 0; }\\n\" +
        "< to { opacity: 1; }\\n\" +
        "<}<\\n\" +
        "<@-webkit-keyframes $@{anim} {\\n\" +
        "< from { opacity: 0; }\\n\" +
        "< to { opacity: 1; }\\n\" +
        "<}</style></head>";

    String end = "<img class='logo' src='https://d2kubvrry40.cloudfront.net/2019/egil/EG-logores.png/EG11/casie/-lx92/format/png/logo.png?v20191023163704' alt='<img alt='casierres12345' />' /></body></html>";
    String cuerpo = "<div class='news'><noticia.getDesc().substring(0, noticia.getDesc().indexOf(\"<p>p<a> \"))</div>";

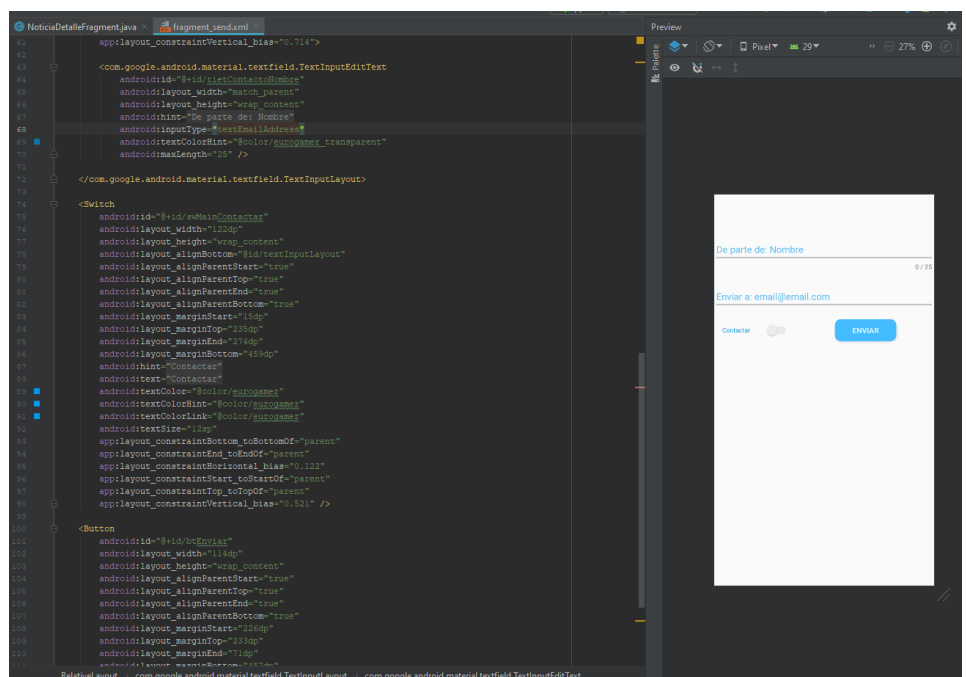
    mViewModelDetails = (TextView).findViewById(R.id.tvTituloDetalles);
    mViewModelDetails = (WebView).findViewById(R.id.wvDescripcionDetalles);

    mViewModelDetails.setText(noticia.getTitulo());
    mViewModelDetails.loadData("<div class='news'><noticia.getDesc().substring(0, noticia.getDesc().indexOf(\"<p>p<a> \"))</div>", "text/html", null);
```

d. Contacto Fragment

i. 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



ii. 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.

```
public View onCreateView(@NonNull LayoutInflater inflater,
                        ViewGroup container, Bundle savedInstanceState) {
    sendViewModel =
        ViewModelProviders.of( fragment, this).get(SendViewModel.class);
    View root = inflater.inflate(R.layout.fragment_send, container, attachToRoot false);
    (MainActivity) getActivity().hideFloatingActionButton();

    //Instancia de los items
    btEnviar = (Button) root.findViewById(R.id.btEnviar);
    tietContactoEmail = (EditText) root.findViewById(R.id.tietContactoEmail);
    tietContactoNombre = (EditText) root.findViewById(R.id.tietContactoNombre);
    swMainContactar = (Switch) root.findViewById(R.id.swMainContactar);

    btEnviar.setOnClickListener((view) -> {
        //Comprobamos el estado de los datos y del Switch, si esta correcto, enviamos el email
        if(!datosCompletos()){
            Snackbar.make(view, text: "Por favor, rellena los datos", Snackbar.LENGTH_LONG).show();
        }else if(!swIsChecked()){
            Snackbar.make(view, text: "Contactar desactivado, no se enviara nada", Snackbar.LENGTH_LONG).show();
        }else{
            enviarEmail(view);
        }
    });

    return root;
}
```

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

```
/**
 * Comprobamos el contenido de los datos
 * @return boolean
 */
protected boolean datosCompletos(){
    if(tietContactoEmail.getText().toString().isEmpty() || tietContactoNombre.getText().toString().isEmpty()){
        return false;
    }

    return true;
}
```

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

```
/**
 * Comprobamos el contenido de los datos
 * @return boolean
 */
protected boolean datosCompletos(){
    if(tietContactoEmail.getText().toString().isEmpty() || tietContactoNombre.getText().toString().isEmpty()){
        return false;
    }

    return true;
}
```

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

```
/**
 * Método para enviar un email
 */
private void enviarEmail(View v){
    try {
        Intent intent = new Intent(Intent.ACTION_SENDTO);
        intent.setData(Uri.parse("mailto:"+tietContactoEmail.getText().toString()));
        //intent.putExtra(Intent.EXTRA_EMAIL, this.email);
        intent.putExtra(Intent.EXTRA_SUBJECT, value: "MyAPPs!!");
        intent.putExtra(android.content.Intent.EXTRA_TEXT, value: "Estoy usando una App con multiples soluciones desarrollada por https://twitter.com/vintajeskull98!! " +
            "¡¡¡la puedes encontrar por el nombre de MyAPPs");
        if (intent.resolveActivity(getContext().getPackageManager()) != null) {
            startActivity(intent);
        } else {
            Snackbar.make(v, text: "Necesitas una app de email", Snackbar.LENGTH_LONG)
                .show();
        }
    } catch (Exception ex){
        Snackbar.make(v, text: "Error inesperado: Contacta con el desarrollador", Snackbar.LENGTH_LONG)
            .show();
    }
}
```

e. About Us Fragment

i. 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

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/btAboutTwitter"
        android:layout_width="71dp"
        android:layout_height="71dp"
        android:background="@drawable/twitter_logo"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.952"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.048" />

    <TextView
        android:id="@+id/tvAboutDev"
        android:layout_width="167dp"
        android:layout_height="232dp"
        android:layout_gravity="center"
        android:gravity="center|center_horizontal"
        android:text="Desarrollador"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.426"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.048" />

    <ImageView
        android:id="@+id/ivMyPhoto"
        android:layout_width="88dp"
        android:layout_height="88dp"
        android:layout_gravity="center"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.049"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.038"
        app:srcCompat="@drawable/avatar" />

    <TextView
        android:id="@+id/tvAboutNombre"
        android:layout_width="153dp"
        android:layout_height="28dp"
        android:gravity="center"
        android:text="Emilio Rubiales"
        android:textSize="18sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.403"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.106" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="238dp"
        android:layout_height="29dp"
        android:text="Enviar mail al desarrollador"

    </com.google.android.material.textfield.TextInputLayout>

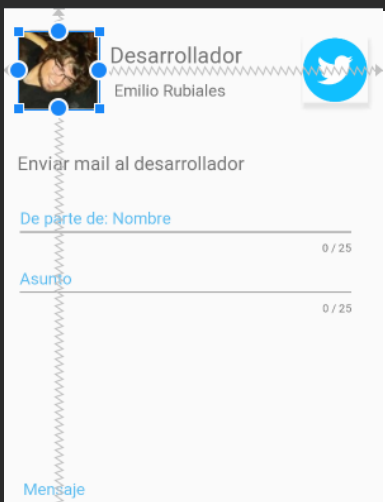
    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/txtAboutAsunto"
        android:layout_width="267dp"
        android:layout_height="72dp"
        android:hint="Asunto"
        android:textColorHint="@color/grey600"
        app:counterEnabled="true"
        app:counterMaxLength="25"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.326"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.395">

        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/etAboutAsunto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:maxLength="25" />
    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/txtAboutMensaje"
        android:layout_width="360dp"
        android:layout_height="234dp"
        android:hint="Mensaje"
        android:textColorHint="@color/grey600"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.352"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.842"
        app:counterEnabled="true"
        app:counterMaxLength="400">

        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/etAboutMensaje"
            android:layout_width="match_parent"
            android:layout_height="185dp"
            android:inputType="textMultiLine"
            android:maxLength="400" />
    </com.google.android.material.textfield.TextInputLayout>

    <Button
        android:id="@+id/btAboutEnviar"
        android:layout_width="114dp"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentBottom="true"
        android:background="@drawable/click_effect"
        android:gravity="center|center_horizontal|center_vertical"
        android:text="Enviar"
        android:textColor="@color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.888"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.976" />
</androidx.constraintlayout.widget.ConstraintLayout>
```



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 extienda demasiado, acompañado de un boton para enviar a traves de su APP de gestion de E-mail

ii. 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

```
public View onCreateView(@NonNull LayoutInflater inflater,
    ViewGroup container, Bundle savedInstanceState) {

    aboutUsViewModel =
        ViewModelProviders.of(this).get(AboutUsViewModel.class);
    View root = inflater.inflate(R.layout.fragment_aboutus, container, attachToRoot: false);
    ((MainActivity) getActivity()).hideFloatingActionButton();

    btAboutTwitter = (Button) root.findViewById(R.id.btAboutTwitter);
    btAboutEnviar = (Button) root.findViewById(R.id.btAboutEnviar);

    tietAboutSumto = (EditText) root.findViewById(R.id.tietAboutSumto);
    tietAboutMensaje = (EditText) root.findViewById(R.id.tietAboutMensaje);
    tietAboutNombre = (EditText) root.findViewById(R.id.tietAboutEnviarNombre);

    //Listener del boton que lleva al Twitter del Desarrollador
    btAboutTwitter.setOnClickListener((view) -> {
        Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("https://twitter.com/vintajeskull99"));
        startActivity(browserIntent);
    });

    //Listener del boton que envia el email gracias para el Desarrollador
    btAboutEnviar.setOnClickListener((view) -> {
        if(!datosCompletos()){
            Snackbar.make(view, "Por favor, rellena los datos", Snackbar.LENGTH_LONG).show();
        }else{
            enviarEmail(view);
        }
    });

    return root;
}

@Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
}

/**
 * Método para enviar un email
 */
private void enviarEmail(View v){
    try {
        Intent intent = new Intent(Intent.ACTION_SENDTO);
        intent.setData(Uri.parse("mailto:vintajeskull99@gmail.com"));
        intent.putExtra(Intent.EXTRA_SUBJECT, tietAboutSumto.getText().toString());
        intent.putExtra(android.content.Intent.EXTRA_TEXT, tietAboutMensaje.getText().toString());
        if (intent.resolveActivity(getContext().getPackageManager()) != null) {
            startActivity(intent);
        } else {
            Snackbar.make(v, "Necesitas una app de email", Snackbar.LENGTH_LONG)
                .show();
        }
    } catch (Exception ex){
        Snackbar.make(v, "Error inesperado: Contacta con el desarrollador", Snackbar.LENGTH_LONG)
            .show();
    }
}

/**
 * Comprobacion del contenido de los datos
 * Return boolean
 */
protected boolean datosCompletos(){
    return !(tietAboutNombre.getText().toString().isEmpty() || tietAboutMensaje.getText().toString().isEmpty() || tietAboutSumto.getText().toString().isEmpty());
}
}
```

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!