PRÁCTICA 3: ShareMyBike

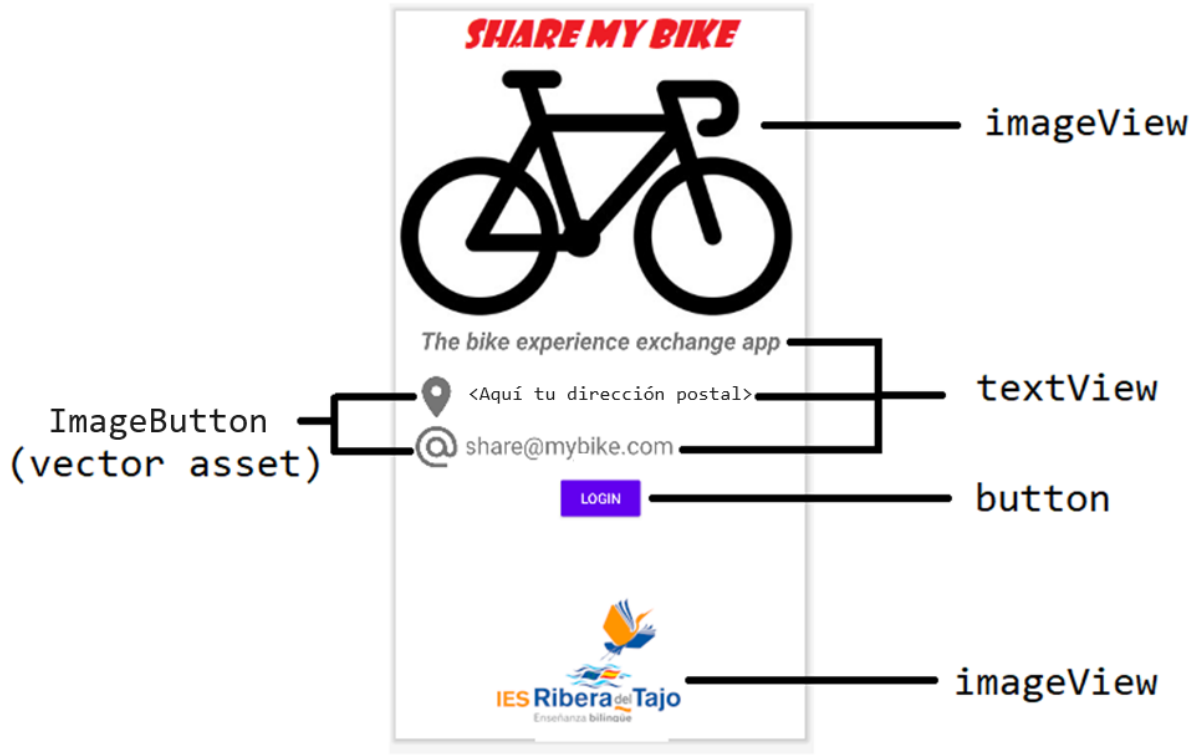
Vamos a crear una aplicación para que los viajeros que quieran llegar a una ciudad, pueblo o lugar puedan usar una bicicleta prestada de forma gratuita. Uno de los objetivos es ayudar a la sostenibilidad incentivando que los viajeros utilicen para sus desplazamientos cortos la bicicleta, en lugar de utilizar otros medios de locomoción como el taxi, autobuses o vehículos de alquiler. De esta manera, un turista de París que llega a Barcelona podría usar la aplicación para ver qué bicicletas están disponibles en préstamo y utilizar una de ellas para sus visitas turísticas. Al terminar, el usuario devolvería la bicicleta al usuario, volviendo a quedar disponible.

Vais a crear, por tanto, una versión preliminar de ShareMyBike para tantear su aceptación en el mercado. En versiones posteriores, se pensará en implementar un sistema de puntos para recompensar a los usuarios que prestan sus bicicletas a otros usuarios. Conserva esta aplicación que te hará falta para hacer otra práctica más adelante.

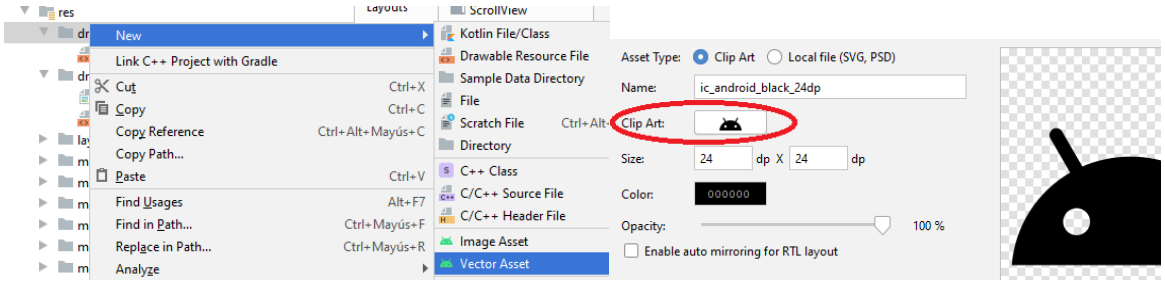
1: La interfaz gráfica de las actividades: El constraintLayout

Vamos a crear la primera actividad de nuestra aplicación: La pantalla de bienvenida.

Tenéis que crear la primera actividad de vuestra primera aplicación, que consistirá en una pantalla de bienvenida a la aplicación. El diseño de la interfaz gráfica será el siguiente:



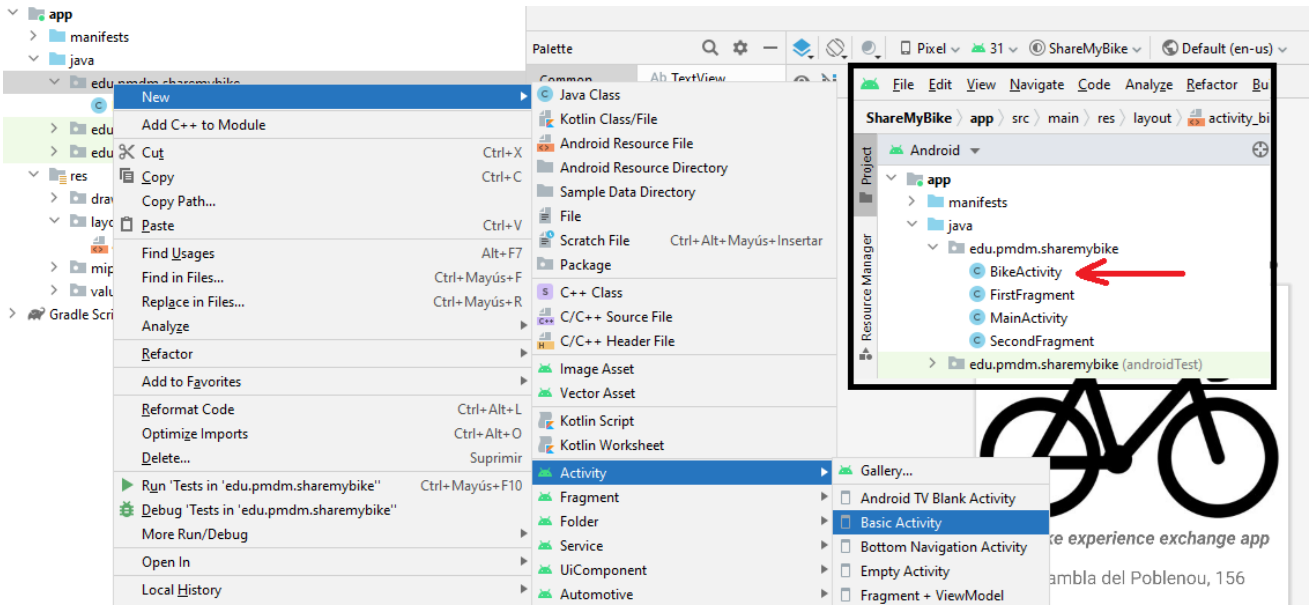
Para el logo de la aplicación y del instituto puedes coger las imágenes de los recursos que adjuntamos al enunciado de la práctica. Para insertar el icono de ubicación puedes utilizar las imágenes vectoriales de Material design. Para ello, añade la imagen utilizando la herramienta Vector Asset. Elige del clip art el icono llamado "*Place*". Observa cómo genera un fichero xml ic_baseline_place_24.xml en la carpeta Drawable. Haz lo mismo con el icono de la dirección de correo electrónico:



Programad un Intent para que, al pulsar el botón  muestre, en Google Maps la siguiente URI: geo:<latitud>,<longitud> donde latitud y longitud son las coordenadas GPS de tu dirección.

Llamadas entre Activities

Para mostrar la lista de bicicletas disponibles para préstamo, añadid una nueva actividad eligiendo esta vez la plantilla “Basic Views Activity” y llamadla “BikeActivity”. Vamos a arrancar esta actividad cuando el usuario pulse el botón de LOGIN de la primera actividad. Observad que ha creado varios layouts y varias clases. De momento tan sólo tenéis que preocuparos de la clase BikeActivity (BikeActivity.java), de su layout (activity_bike.xml) y del layout para el contenido (content_bike.xml). Verás que se añaden otros cuatro ficheros, FirstFragment.java, SecondFragment.java, fragment_first.xml y fragment_second.xml, que se utilizarán para navegar por la interfaz de usuario a través de fragmentos de esta nueva actividad.



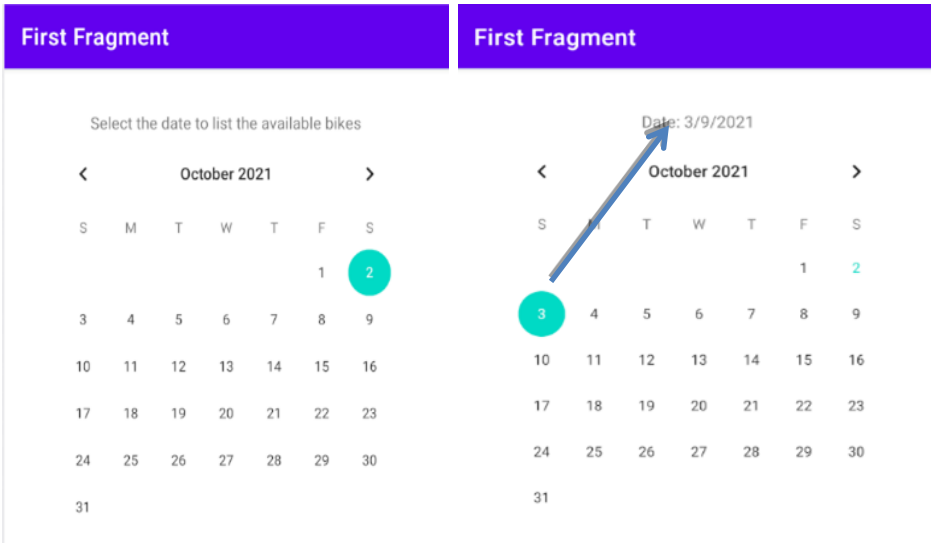
Cread un listener en el método onCreate de MainActivity para que cuando se pulse el botón Login, se inicie esta segunda actividad “BikeActivity”:

Trabajando con fragmentos.

Si exploráis el código de esta nueva actividad observaréis que, en este punto, la actividad BikeActivity tiene un layout activity_bike compuesto por un AppBarLayout, una ToolBar, un fragmento de contenido y un botón flotante. El botón flotante, lo podéis eliminar puesto que no lo vamos a utilizar. Si exploráis el fragmento de contenido veréis que está asociado a un gráfico de navegación “nav_graph.xml” que permite reemplazar el contenido del fragmento por “fragment_first.xml” y “fragment_second.xml”.

Modificad la interfaz de usuario del primer fragmento para incluir un widget de tipo CalendarView de manera que el usuario puede elegir la fecha en la que quiere alquilar una bicicleta y un campo de texto TextView para mostrar el resultado de la selección.

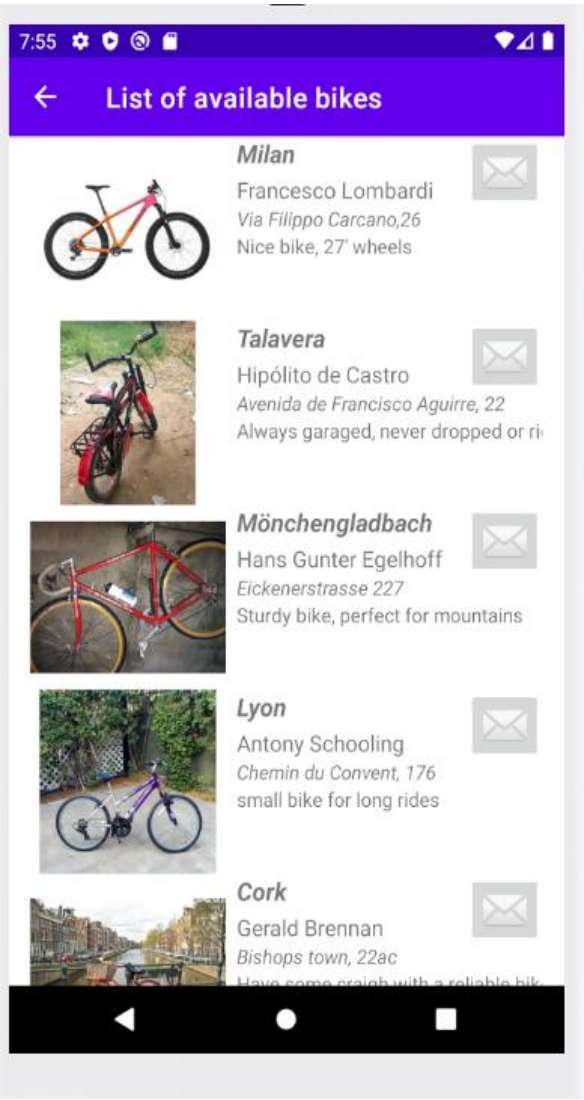
Añadid la función de Callback **onSelectedDayChange** para responder al evento de cambio de fecha por parte del usuario. La función de callback debe cambiar el valor del campo de texto.



Para terminar, cambiad el nombre que aparece en la barra de First Fragment a Select Date. Para ello, podéis modificar el valor desde el fichero strings.xml de la carpeta res/values.

El fragmento de bicicletas disponibles. Listas de objetos con RecyclerView.

A continuación, incluid un nuevo fragmento con un componente llamado RecyclerView. Este widget nos servirá para mostrar la lista de bicicletas disponibles. El fragmento, al terminar este ejercicio, os quedará como en la siguiente imagen:

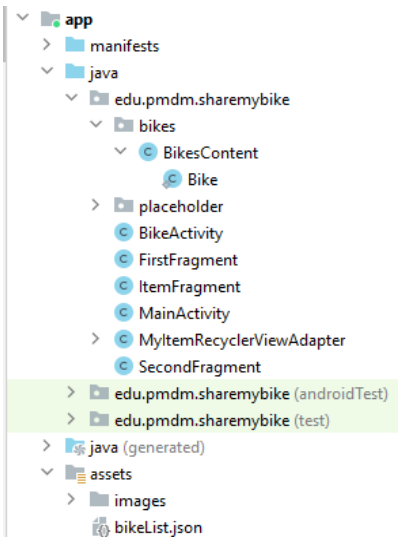


Podéis copiar de los recursos la clase BikesContent que incluye la subclase Bike. La clase BikesContent carga en un ArrayList un conjunto de objetos de tipo Bike, que se genera a partir de leer el archivo JSON bikeList.json y que también podéis copiar de la carpeta de recursos. Incluid el archivo JSON en la carpeta assets que deberéis crear en el proyecto:

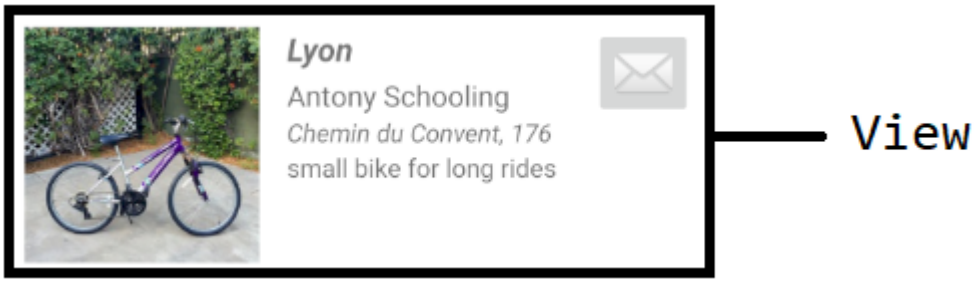
```
public class BikesContent {
    //List of all the bikes to be listed in the RecyclerView
    public static List<Bike> ITEMS = new ArrayList<Bike>();
    public static String selectedDate; //Store the selected date


    public static void loadBikesFromJSON(Context c) {

        String json = null;
        try {
            InputStream is =
                    c.getAssets().open("bikeList.json");
            int size = is.available();
            byte[] buffer = new byte[size];
            is.read(buffer);
            is.close();
            json = new String(buffer, "UTF-8");

            JSONObject jsonObject = new JSONObject(json);
            JSONArray bikeList = jsonObject.getJSONArray("bike_list");
```

```
            for (int i = 0; i < bikeList.length(); i++) {
                JSONObject jsonBikes = bikeList.getJSONObject(i);
                String owner = jsonBikes.getString("owner");
                String description = jsonBikes.getString("description");
                String city=jsonBikes.getString("city");
                String location=jsonBikes.getString("location");
                String email=jsonBikes.getString("email");
                Bitmap photo=null;

                try {
                    photo= BitmapFactory.decodeStream(
                            c.getAssets().open("images/"+
                                    jsonBikes.getString("image")));
                } catch (IOException e) {
                    e.printStackTrace();
                }

                ITEMS.add(new BikesContent.Bike(
                        photo,owner,description,city,location,email));
            }

        } catch (JSONException | IOException e) {
            e.printStackTrace();
        }

    }


    public static class Bike {
        private Bitmap photo;
        private String owner;
        private String description;
        private String city;
        private String location;
        private String email;
        //Setters and getters...
        public Bike(Bitmap photo, String owner, String description,
                    String city, String location, String email) {
            this.photo = photo;
            this.owner = owner;
            this.description = description;
            this.city = city;
            this.location = location;
            this.email= email;
        }
        @Override
        public String toString() {
            return owner+" "+description;
        }
    }
}
```

La estructura del proyecto os tiene que quedar como en la figura de la derecha:



Cada fila que muestra el RecyclerView representará un objeto de tipo Bike y todos sus atributos. La disposición de los componentes se define en su layout fragment_item.xml. De momento este layout tan sólo incluye dos campos de texto, pero al terminar este ejercicio tu fragment_item quedará así:



1. Reemplazad el código xml del layout que representa cada fila del recyclerView cambiándolo para incorporar un diseño con el componente “CardView”. Dentro del CardView incluid un constraintLayout para insertar los elementos que deben mostrarse.

2. Cargad en el adaptador MyItemRecyclerView la lista con todas las bicicletas. Para ello, invocad al método estático `loadBikesFromJSON` desde BikeActivity y de esta manera, cargar en el ArrayList todas las bicicletas que posteriormente el Adaptador vinculará al RecyclerView. Este método recibe como parámetro el contexto de la actividad. Este contexto es una interfaz que implementa la propia Activity.

3. Reemplazad en el archivo ItemFragment.java la creación del objeto MyItemRecyclerViewAdapter para que cargue la lista de elementos por defecto por la nueva lista de bicicletas BikesContent.ITEMS.

4. Modificad la clase MyItemRecyclerView.ViewHolder y vinculad cada componente de cada fila del RecyclerView a una variable de clase.

5. Programad el método OnBindViewHolder de la clase MyItemRecyclerViewAdapter para rellenar los valores de los widgets para cada una de las filas mostradas en el recyclerView. Por ejemplo, para rellenar el campo propietario (owner), podéis usar el siguiente código:

6. Para dotar de funcionalidad al botón de Email podéis programar una función de callback OnClick que debéis añadir al botón dentro de OnBindViewHolder:

Enviando un email al propietario de la bicicleta: Los intent.

Para enviar un email con la reserva de una bicicleta, podéis utilizar un objeto de tipo Intent con un URI de tipo mailto:. Este tipo de componentes se llaman intents “implícitos”:

https://developer.android.com/guide/components/intents-filters

Añadid el cuerpo del email, que consistirá en un campo con el texto:

```
Dear Mr/Mrs <propietario>:

  I'd like to use your bike at <ubicación> (<ciudad>)
  for the following date: <fecha>

  Can you confirm its availability?
  Kindest regards
```

ENTREGA:
Para entregar el proyecto, exporta el proyecto a zip file e incluye un video de la ejecución de tu aplicación.