Documentación

Proyecto agenda

Iván Gallego

Índice de contenidos

> En todos los casos, cuando se llama a un fragment, se le pasa por el constructor el id del elemento del que se ha hecho click, o null si es para la adición de un nuevo elemento, en el fragment receptor, se detectará si es null o no y se generará uno nuevo o no

Operations

Antes de meternos de lleno en la aplicación debo aclarar que utilizo una clase estática que me facilita los accesos a la base de datos así como otras funciones y campos de utilidad que se actualizan a medida que avanzo en los distintos fragments de la aplicación. En esta clase guardo por ejemplo cuál es el usuario que está activo o los documentos y colecciones por las que tengo que pasar para llegar a un dato en concreto.

La clase se encuentra en los archivos del proyecto, no lo coloco aquí ya que es un tanto extenso.

Login

Pantalla de inicio de la aplicación, En ella te puedes logear o registrar si es la primera vez.

El funcionamiento es parecido a como lo hemos hecho en actividades anteriores, cuando te registras o haces login, se inicia una actividad que contiene el funcionamiento principal de la aplicación.

El método `initializeReferences` es de la clase estática y establece las colecciones y documentos que se pueden saber cuáles son simplemente al inicio como sería el de la colección de ciudades de un usuario

```
public class Operations {

    public static void initializeReferences() {
        DEFAULT_IMAGE.getDownloadUrl().addOnCompleteListener(new OnCompleteListener<Uri>() {
            @Override
            public void onComplete(@NonNull Task<Uri> task) {
                DEFAULT_IMAGE_RESOURCE = task.getResult();
            }
        });
        FirebaseFirestore fire = FirebaseFirestore.getInstance();
        if (user != null) {
            userDocument = fire.collection("usuarios").document(user.getEmail());
            cityCollection = userDocument.collection("ciudades");
        }
    }
```

```
public class MainActivity extends AppCompatActivity {

    private TextInputEditText usuario, pass;
    private FirebaseAuth auth;
```

```
    private FirebaseAuth.AuthStateListener authListener;

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

        usuario = findViewById(R.id.editUsuario);
        pass = findViewById(R.id.editPass);
        Button crear = findViewById(R.id.buttonCrear);
        Button entrar = findViewById(R.id.buttonEntrar);

        auth = FirebaseAuth.getInstance();

        authListener = new FirebaseAuth.AuthStateListener() {
            @Override
            public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
                Operations.user = firebaseAuth.getCurrentUser();
                if (Operations.user != null) {
                    Toast.makeText(MainActivity.this, Operations.user.getEmail() + " logeado",
                        Toast.LENGTH_LONG).show();
                    auth = firebaseAuth;
                }
            }
        };

        crear.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                auth.createUserWithEmailAndPassword(usuario.getText().toString(),
                    pass.getText().toString())
                        .addOnCompleteListener(MainActivity.this, new OnCompleteListener<AuthResult>() {
                            @Override
                            public void onComplete(@NonNull Task<AuthResult> task) {
                                if (task.isSuccessful()) {
                                    iniciarAplicacion();
                                } else {
                                    Toast.makeText(MainActivity.this,
                                        "Problemas al crear el usuario", Toast.LENGTH_LONG).show();
                                }
                            }
                        });
            }
        });

        entrar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                auth.signInWithEmailAndPassword(usuario.getText().toString(), pass.getText().toString())
                        .addOnCompleteListener(MainActivity.this, new OnCompleteListener<AuthResult>() {
                            @Override
                            public void onComplete(@NonNull Task<AuthResult> task) {
                                if (task.isSuccessful()) {
                                    iniciarAplicacion();
                                } else {
                                    Toast.makeText(MainActivity.this,
                                        "Error de autenticación", Toast.LENGTH_LONG).show();
                                }
                            }
                        });
            }
        });

    }

    private void iniciarAplicacion() {
        Intent intent = new Intent(this, MainApplication.class);
```

```
            Operations.initializeReferences();
            startActivity(intent);
        }

        @Override
        protected void onStart() {
            super.onStart();
            auth.addAuthStateListener(authListener);
        }

        @Override
        protected void onStop() {
            super.onStop();
            if (auth != null) {
                auth.removeAuthStateListener(authListener);
                auth.signOut();
            }
        }
    }
```

Inicio de la aplicación, perfil de usuario

Cuando se inica la aplicación pueden ocurrir dos cosas:

1. Es la primera vez que entra el usuario
2. El usuario ya ha entrado alguna vez

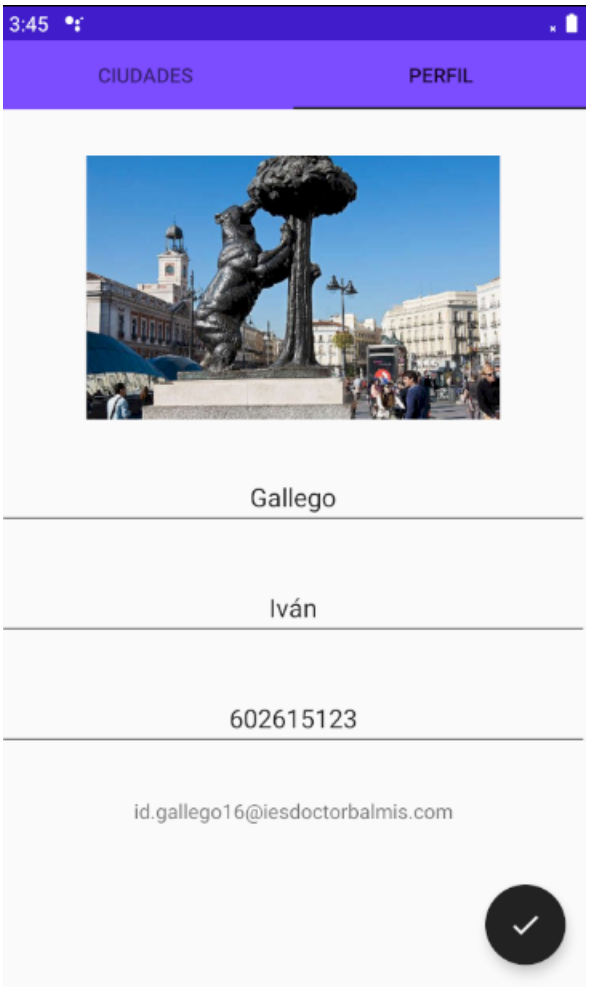
Si es el primer caso, entonces se le mostrará automáticamente la pantalla de su perfil en la que podrá cambiar sus datos personales, posteriormente podrá confirmar con el botón inferior derecho.

```
public class FragmentTabs extends Fragment {

    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
            @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        super.onCreateView(inflater, container, savedInstanceState);
        View v = inflater.inflate(R.layout.tabs_fragment, container, false);
        // Creación e inicialización de las tabs
        TabLayout tabs = v.findViewById(R.id.tab_layout);
        final ViewPager pager = v.findViewById(R.id.view_pager);
        PagerAdapterImplementation adapter =
            new PagerAdapterImplementation(getActivity().getSupportFragmentManager(), 2);
        tabs.setupWithViewPager(pager);
        pager.setAdapter(adapter);
        pager.addOnPageChangeListener(new TabLayout.TabLayoutOnPageChangeListener(tabs));
        tabs.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {
            @Override
            public void onTabSelected(TabLayout.Tab tab) {
                pager.setCurrentItem(tab.getPosition());
            }

            @Override
            public void onTabUnselected(TabLayout.Tab tab) {
            }

            @Override
            public void onTabReselected(TabLayout.Tab tab) {
            }
        });
        // Si es la primera vez del usuario, muestra la pantalla del perfil
        Operations.userDocument.get().addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {
            @Override
            public void onComplete(@NonNull Task<DocumentSnapshot> task) {
                Boolean b = task.getResult().getBoolean("primera_vez");
                if (b == null) {
                    pager.setCurrentItem(1);
                }
            }
        });
        return v;
    }
}
```



Inicio de la aplicación, lista de ciudades

Cuando se inica la aplicación pueden ocurrir dos cosas:

1. Es la primera vez que entra el usuario
2. El usuario ya ha entrado alguna vez

Si es el segundo caso, entonces se le mostrará al usuario directamente la lista de ciudades y podrá editar las que ya existan, eliminar con un click largo o añadir nuevas con el botón expandible inferior derecho.

El botón de refrescar hace que el adaptador cargue otra vez las imágenes por si no se han actualizado.

Al crear la vista, se inicializa la colección de ciudades del usuario y los botones que se utilizarán en la animación de expansión.

1. Si se pulsa en el de refrescar, se actualizará el adaptador
2. Si se pulsa el de añadir, se mostrará el fragment de añadir una ciudad

```
@Nullable
@Override
public View onCreateView(@NonNull LayoutInflater inflater,
        @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    super.onCreateView(inflater, container, savedInstanceState);
    View view = inflater.inflate(R.layout.fragment_recycler, container, false);
    // Inicialización de la colección de ciudades del usuario en la clase estática
    userReference = FirebaseFirestore.getInstance().collection("usuarios")
            .document(Operations.user.getEmail());
    fab1 = view.findViewById(R.id.fabA);
    fab1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            mostrarFragmentEdit(Operations.Accion.ADD_REQUEST, null, null);
        }
```

```
    });
    fab2 = view.findViewById(R.id.fabB);
    fab2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            adaptador.notifyDataSetChanged();
        }
    });
    fab = view.findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (!isFABOpen) {
                showFABMenu();
            } else {
                closeFABMenu();
            }
        }
    });
    recyclerView = view.findViewById(R.id.recycler);
    inicializarAdaptador();
    return view;
}

private void mostrarFragmentEdit(Operations.Accion accion, Ciudad ciudad, String key) {
    FragmentManager fm = getParentFragmentManager();
    FragmentTransaction ft = fm.beginTransaction();
    FragmentEdit fragment = new FragmentEdit(accion, ciudad, key);
    ft.add(R.id.fragment_container, fragment)
            .addToBackStack(null)
            .commit();
}
```

Finalmente, inicializamos el adaptador y cargamos el recycler con datos de la query de todas las ciudades

```
private void cargarRecycler(Query query) {
    FirestoreRecyclerOptions<Ciudad> firestoreRecyclerOptions = new FirestoreRecyclerOptions.Builder<Ciudad>()
            .setQuery(query, Ciudad.class).setLifecycleOwner(this).build();
    adaptador = new Adaptador(firestoreRecyclerOptions);
    recyclerView.setAdapter(adaptador);
    adaptador.startListening();
    adaptador.setOnClickListener(this);
    adaptador.setOnLongClickListener(this);
    recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
}

private void inicializarAdaptador() {
    userReference.collection("ciudades").get().addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
        @Override
        public void onComplete(@NonNull Task<QuerySnapshot> task) {
            if (task.isSuccessful()) {
                Query query = task.getResult().getQuery();
                cargarRecycler(query);
            } else {
                Toast.makeText(getContext(),
                "Error al obtener datos de la base de datos", Toast.LENGTH_LONG).show();
            }
        }
    });
}
```

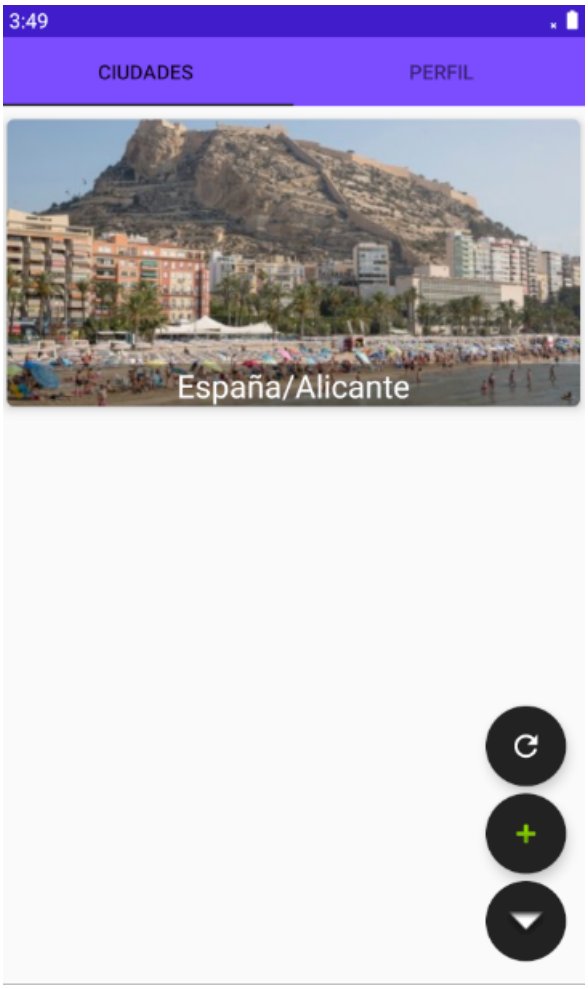
Cuando se hace click sobre una ciudad ya existente, se mostrará el fragment de edición de ciudades

```
@Override
public void onClick(View v) {
    String key = adaptador.getSnapshots().getSnapshot(recyclerView.getChildAdapterPosition(v)).getId();
    Ciudad c = adaptador.getItem(recyclerView.getChildAdapterPosition(v));
    mostrarFragmentEdit(Operations.Accion.EDIT_REQUEST, c, key);
}
```

Para la eliminación de una ciudad, se hace con un `onLongClickListener` y llamo al método de ayuda `deleteCity` al que le paso el id del documento que quiero eliminar. El método simplemente sobre la colección de ciudades, que previamente se habrá inicializado en la clase estática, busca el documento con el id proporcionado y lo elimina

```
@Override
public boolean onLongClick(View v) {
    String key = adaptador.getSnapshots().getSnapshot(recyclerView.getChildAdapterPosition(v)).getId();
    Operations.deleteCity(key);
    return false;
}
```

```
public class Operations{
    public static void deleteCity(final String key) {
        cityCollection.document(key).delete().addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido borrar la ciudad", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```



Edición / Adición de ciudades

Cuando se quiera añadir o editar una ciudades se mostrará el mismo fragment. **Si es edición** Se mostrarán los datos correspondientes a la ciudad seleccionada, y se podrán modificar. **Si es adición** Se mostrarán los datos por defecto, que podrán ser editados y se mostrará una imagen por defecto.

En ambos casos, se podrán añadir lugares con el botón inferior derecho expandible, cuyo funcionamiento es el mismo que el de la lista de ciudades. Además, se podrán confirmar las modificaciones con el botón de en medio de la pantalla.

Al crear la vista, se realiza un proceso similar al del fragment anterior en el que se muestran las ciudades, se inicializan las variables, se carga el adaptador y se carga el recycler con datos resultantes de la query a la colección de lugares de la ciudad específica. Además, se mostrarán los datos de la ciudad sobre la que se ha hecho click o datos por defecto si se está añadiendo una ciudad. Esto lo controlo con la variable `accion`.

Se utilizan algunos métodos de ayuda como por ejemplo `loadIntoImageView` que carga una imagen en un `ImageView` si existe, si no, carga una por defecto.

También se establece cuál es el documento de ciudad en el que estamos actualmente para posteriormente poder realizar operaciones sobre ese documento sin tener que pasar la referencia por todos lados.

Cuando el usuario haya editado o creado la ciudad y pulse sobre el botón de confirmación, dependiendo de la acción con la que se haya llamado al fragment, se editará una ciudad o se añadirá una nueva. Además eliminaremos el fragment de la vista del usuario con `getParentFragmentManager().popBackstack()`.

```
public class Operations{
    private static void updateCity(Ciudad newCity, String documentKey) {
        cityCollection.document(documentKey).set(newCity).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido actualizar la ciudad", Toast.LENGTH_SHORT).show();
            }
```

```
        });
    }

    public static void updateCity(final Ciudad newCity, final String key, Uri image) {
        if (image == null) {
            newCity.setImagen(DEFAULT_URL);
            updateCity(newCity, key);
        } else {
            uploadImage(newCity.getImagen(), image)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    updateCity(newCity, key);
                }
            });
        }
    }

    public static void addCity(final Ciudad c, final Uri image) {
        final String randomString = getRandomString();
        c.setImagen(randomString);
        if (image == null) {
            uploadImage(randomString, DEFAULT_IMAGE_RESOURCE)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    cityCollection.document(randomString).set(c);
                }
            });
        } else {
            uploadImage(randomString, image)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    cityCollection.document(randomString).set(c);
                }
            });
        }
    }
}
```

```
@Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
            @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        super.onCreateView(inflater, container, savedInstanceState);
        View v = inflater.inflate(R.layout.edit_fragment, container, false);
        editCiudad = v.findViewById(R.id.editCiudad);
        pais = v.findViewById(R.id.editPais);
        imageView = v.findViewById(R.id.imageView);
        recycler = v.findViewById(R.id.recycler_lugares);

        imageView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
                intent.setType("image/*");
                if (intent.resolveActivity(getActivity().getPackageManager()) != null) {
                    startActivityForResult(intent, COD_ELEGIR_IMAGEN);
                }
            }
        });

        if (accion == Operations.Accion.EDIT_REQUEST) {
            FirebaseStorage.getInstance().getReference(key).getDownloadUrl()
```

```
            .addOnCompleteListener(new OnCompleteListener<Uri>() {
                @Override
                public void onComplete(@NonNull Task<Uri> task) {
                    if (task.isSuccessful()) {
                        selectedImage = task.getResult();
                    }
                }
            });
            editCiudad.setText(ciudad.getCiudad());
            pais.setText(ciudad.getPais());
            if (!ciudad.getImagen().equals("") && ciudad.getImagen() != null) {
                Operations.loadIntoImageView(FirebaseStorage.getInstance()
                    .getReference(ciudad.getImagen()), imageView);
            }
        } else {
            key = Operations.newId();
            Operations.loadIntoImageView(Operations.DEFAULT_IMAGE, imageView);
        }
        Operations.cityDocument = Operations.cityCollection.document(key);

        inicializarAdaptador();


        v.findViewById(R.id.buttonAceptar).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final Ciudad c = generarCiudad();
                if (accion == Operations.Accion.ADD_REQUEST) {
                    Operations.addCity(c, selectedImage);
                } else {
                    Operations.updateCity(c, key, selectedImage);
                }
                getParentFragmentManager().popBackStack();
            }
        });
        fab1 = v.findViewById(R.id.fabA);
        fab1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mostrarFragmentEditLugar(null, Operations.Accion.ADD_REQUEST, null);
            }
        });
        fab2 = v.findViewById(R.id.fabB);
        fab2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                adaptador.notifyDataSetChanged();
            }
        });
        fab = v.findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (!isFABOpen) {
                    showFABMenu();
                } else {
                    closeFABMenu();
                }
            }
        });

        return v;
    }

    private void inicializarAdaptador() {
        Operations.placeCollection = Operations.cityDocument.collection("lugares");
        Operations.placeCollection.get().addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
```

```
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    Query query = task.getResult().getQuery();
                    cargarRecycler(query);
                } else {
                    Toast.makeText(getContext(),
                    "Error al obtener datos de la base de datos", Toast.LENGTH_LONG).show();
                }
            }
        });
    }

    private void cargarRecycler(Query query) {
        FirestoreRecyclerOptions<Lugar> firestoreRecyclerOptions =
            new FirestoreRecyclerOptions.Builder<Lugar>()
                .setQuery(query, Lugar.class).setLifecycleOwner(this).build();
        adaptador = new AdaptadorLugares(firestoreRecyclerOptions);
        recycler.setAdapter(adaptador);
        adaptador.startListening();
        adaptador.setOnClickListener(this);
        adaptador.setOnLongClickListener(this);
        recycler.setLayoutManager(new LinearLayoutManager(getContext()));
    }

    private Ciudad generarCiudad() {
        return new Ciudad(editCiudad.getText().toString(), pais.getText().toString(), key);
    }
```

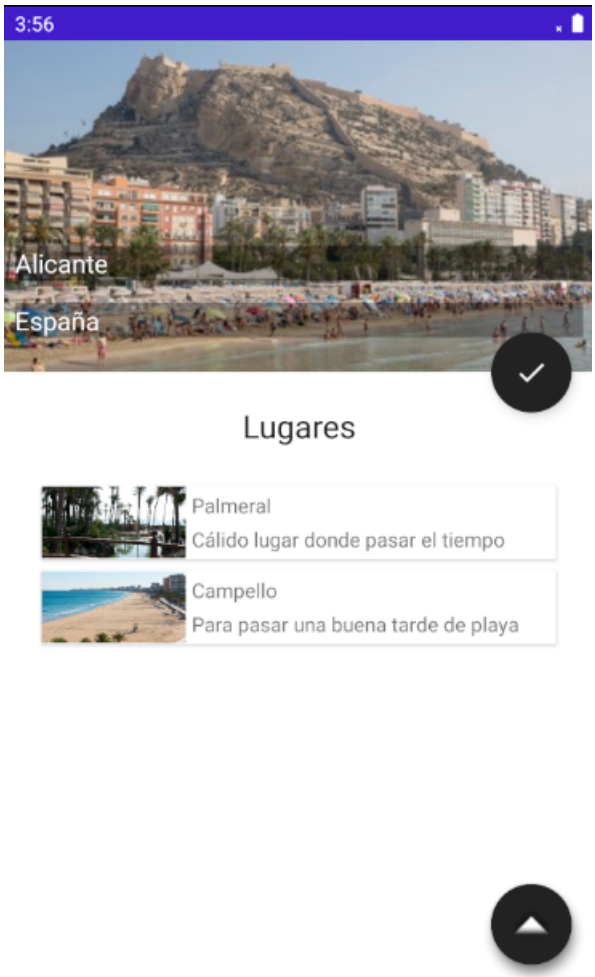
Si se pulsa sobre el botón para añadir un lugar, o si se pulsa sobre un lugar existente, se mostrará el fragment correspondiente a la edición de lugares

```
@Override
public void onClick(View v) {
    Lugar l = adaptador.getItem(recycler.getChildAdapterPosition(v));
    String key = adaptador.getSnapshots().getSnapshot(recycler.getChildAdapterPosition(v)).getId();
    mostrarFragmentEditLugar(l, Operations.Accion.EDIT_REQUEST, key);
}
```

De la misma manera que en la lista de ciudades, si se hace una pulsación larga sobre un lugar, se eliminará

```
@Override
public boolean onLongClick(View v) {
    String key = adaptador.getSnapshots().getSnapshot(recycler.getChildAdapterPosition(v)).getId();
    Operations.deletePlace(key);
    return false;
}
```

```
public class Operations{
    public static void deletePlace(String key) {
        placeCollection.document(key).delete().addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido borrar el lugar", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```



Edición / Adición de lugares

Cuando se quiera añadir o editar un lugar se mostrará el mismo fragment. **Si es edición** Se mostrarán los datos correspondientes al lugar seleccionado, y se podrán modificar. **Si es adición** Se mostrarán los datos por defecto, que podrán ser editados y se mostrará una imagen por defecto.

En ambos casos, se podrán añadir lugares destacados con el botón inferior derecho expandible, cuyo funcionamiento es el mismo que el de la lista de ciudades o de la lista de lugares. Además, se podrán confirmar las modificaciones con el botón de en medio de la pantalla.

Al crear la vista, se realiza un proceso similar al del fragment anterior en el que se muestra una ciudad y sus lugares, se inicializan las variables, se carga el adaptador y se carga el recycler con datos resultantes de la query a la colección de lugares destacados del lugar específico. Además, se mostrarán los datos del lugar sobre el que se ha hecho click o datos por defecto si se está añadiendo un lugar. Esto lo controlo con la variable `accion`.

También se establece cuál es el documento del lugar en el que estamos actualmente para posteriormente poder realizar operaciones sobre ese documento sin tener que pasar la referencia por todos lados.

Cuando el usuario haya editado o creado un lugar y pulse sobre el botón de confirmación, dependiendo de la acción con la que se haya llamado al fragment, se editará un lugar o se añadirá uno nuevo. Además eliminaremos el fragment de la vista del usuario con `getParentFragmentManager().popBackstack()`.

Llamadas a métodos de ayuda estáticos para la gestión de lugares.

El método `crearRecursivamente` lo que hace es crear una estructura de Ciudad -> Lugar -> Lugar destacado, porque puede darse la ocasión en la que el usuario haya decidido añadir un lugar destacado sin antes haber añadido ningún dato de la ciudad, por este motivo se crea un árbol con datos por defecto

```
public class Operations{
    public static void addPlace(final Lugar lugar, final Uri image) {
        crearRecursivamente(false);
        final String randomString = getRandomString();
```

```
        lugar.setImagen(randomString);
        if (image == null) {
            uploadImage(randomString, DEFAULT_IMAGE_RESOURCE)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    placeCollection.document(randomString).set(lugar);
                }
            });
        } else {
            uploadImage(randomString, image)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    placeCollection.document(randomString).set(lugar);
                }
            });
        }
    }

    private static void updateCity(Ciudad newCity, String documentKey) {
        cityCollection.document(documentKey).set(newCity).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido actualizar la ciudad", Toast.LENGTH_SHORT).show();
            }
        });
    }

    private static void updatePlace(Lugar newLugar, String documentKey) {
        placeCollection.document(documentKey).set(newLugar).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido actualizar la ciudad", Toast.LENGTH_SHORT).show();
            }
        });
    }

    private static void crearRecursivamente(boolean includePlace) {
        if (includePlace) {
            placeDocument.get().addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<DocumentSnapshot> task) {
                    if (!task.getResult().exists()) {
                        placeDocument.set(new Lugar("Lugar", "Descripción", placeDocument.getId()));
                    }
                }
            });
        }
        cityDocument.get().addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {
            @Override
            public void onComplete(@NonNull Task<DocumentSnapshot> task) {
                if (!task.getResult().exists()) {
                    cityDocument.set(new Ciudad("Ciudad", "País", cityDocument.getId()));
                }
            }
        });
    }
}
```

El código del `onCreateView` es prácticamente idéntico al anterior.

```
@Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
            @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        super.onCreateView(inflater, container, savedInstanceState);
        View v = inflater.inflate(R.layout.edit_lugar_fragment, container, false);
        recycler = v.findViewById(R.id.recycler_lugares_fotos);
        editLugar = v.findViewById(R.id.editLugar);
        editDescripcion = v.findViewById(R.id.editDescripcion);
        imageView = v.findViewById(R.id.imageViewLugar);
        if (accion == Operations.Accion.EDIT_REQUEST) {
            FirebaseStorage.getInstance().getReference(key).getDownloadUrl()
            .addOnCompleteListener(new OnCompleteListener<Uri>() {
                @Override
                public void onComplete(@NonNull Task<Uri> task) {
                    if (task.isSuccessful()) {
                        selectedImage = task.getResult();
                    }
                }
            });
            editLugar.setText(lugar.getLugar());
            editDescripcion.setText(lugar.getDescripcion());
            if (!lugar.getImagen().equals("") && lugar.getImagen() != null) {
                Operations.loadIntoImageView(FirebaseStorage.getInstance()
                    .getReference(lugar.getImagen()), imageView);
            }
        } else {
            key = Operations.newId();
            Operations.loadIntoImageView(Operations.DEFAULT_IMAGE, imageView);
        }
        Operations.placeDocument = Operations.placeCollection.document(key);
        inicializarAdaptador();

        imageView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
                intent.setType("image/*");
                if (intent.resolveActivity(getActivity().getPackageManager()) != null) {
                    startActivityForResult(intent, COD_ELEGIR_IMAGEN);
                }
            }
        });

        fab1 = v.findViewById(R.id.fabA);
        fab1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                getParentFragmentManager()
                        .beginTransaction()
                        .add(R.id.fragment_container,
                            new LugarDestacadoFragment(null, null, Operations.Accion.ADD_REQUEST))
                        .addToBackStack(null).commit();
            }
        });
        fab2 = v.findViewById(R.id.fabB);
        fab2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                adaptador.notifyDataSetChanged();
            }
        });
        fab = v.findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```
                if (!isFABOpen) {
                    showFABMenu();
                } else {
                    closeFABMenu();
                }
            }
        });

        v.findViewById(R.id.buttonAceptarLugar).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Lugar l = generarLugar();
                if (accion == Operations.Accion.ADD_REQUEST) {
                    Operations.addPlace(l, selectedImage);
                } else {
                    Operations.updatePlace(l, key, selectedImage);
                }
                getParentFragmentManager().popBackStack();
            }
        });

        return v;
    }

    private void inicializarAdaptador() {
        Operations.mainPlaceCollection = Operations.placeDocument.collection("fotos");
        Operations.mainPlaceCollection.get().addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    Query query = task.getResult().getQuery();
                    cargarRecycler(query);
                } else {
                    Toast.makeText(getContext(),
                    "Error al obtener datos de la base de datos", Toast.LENGTH_LONG).show();
                }
            }
        });
    }

    private void cargarRecycler(Query query) {
        FirestoreRecyclerOptions<LugarDestacado> firestoreRecyclerOptions =
            new FirestoreRecyclerOptions.Builder<LugarDestacado>()
                .setQuery(query, LugarDestacado.class).setLifecycleOwner(this).build();
        adaptador = new AdaptadorLugaresDestacados(firestoreRecyclerOptions);
        adaptador.setOnClickListener(this);
        adaptador.setOnLongClickListener(this);
        recycler.setAdapter(adaptador);
        adaptador.startListening();
        recycler.setLayoutManager(new LinearLayoutManager(getContext(), RecyclerView.HORIZONTAL, false));
    }
```

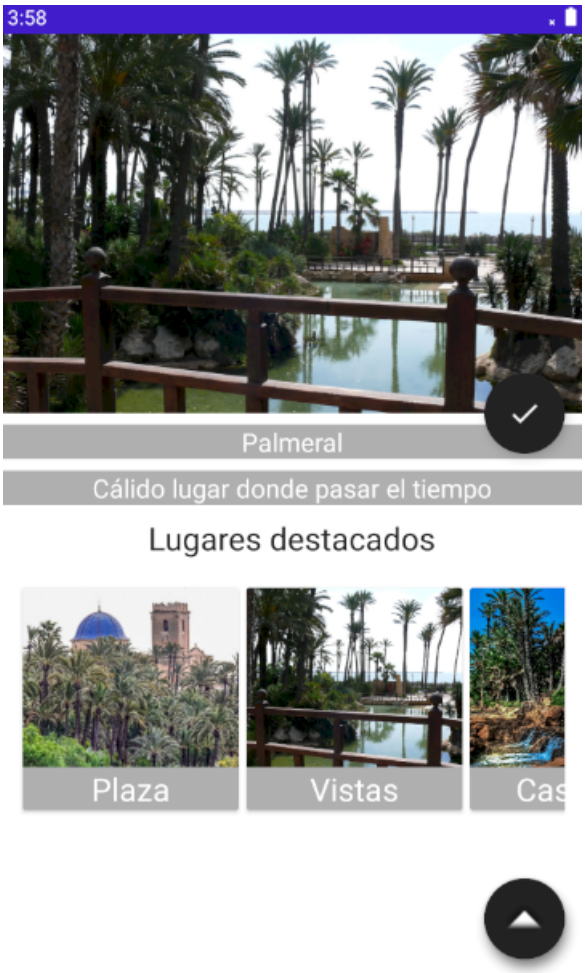
Si se pulsa sobre el botón para añadir un lugar destacado, o si se pulsa sobre un lugar descado existente, se mostrará el fragment correspondiente a la edición de lugares destacados.

```
@Override
public void onClick(View v) {
    LugarDestacado lugar = adaptador.getSnapshots().get(recycler.getChildAdapterPosition(v));
    String key = adaptador.getSnapshots().getSnapshot(recycler.getChildAdapterPosition(v)).getId();
    getParentFragmentManager()
            .beginTransaction()
            .add(R.id.fragment_container,
                new LugarDestacadoFragment(lugar, key, Operations.Accion.EDIT_REQUEST))
            .addToBackStack(null).commit();
}
```

De la misma manera que en la lista de ciudades, si se hace una pulsación larga sobre un lugar, se eliminará.

```
@Override
public boolean onLongClick(View v) {
    String key = adaptador.getSnapshots().getSnapshot(recycler.getChildAdapterPosition(v)).getId();
    Operations.deleteMainPlace(key);
    return false;
}
```

```
public class Operations{
    public static void deleteMainPlace(String key) {
        mainPlaceCollection.document(key).delete().addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido borrar el lugar destacado", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```



Edición / Adición de lugares destacado

Cuando se quiera añadir o editar un lugar destacado se mostrará el mismo fragment. **Si es edición** Se mostrarán los datos correspondientes al lugar seleccionado, y se podrán modificar. **Si es adición** Se mostrarán los datos por defecto, que podrán ser editados y se mostrará una imagen por defecto.

En ambos casos, se podrán confirmar las modificaciones con el botón inferior derecho.

Esta vez, al crear la vista, simplemente se muestra una serie de campos de edición para poder editar o añadir un lugar destacado. Esto lo controlo con la variable `accion`.

También se establece cuál es el documento del lugar en el que estamos actualmente para posteriormente poder realizar operaciones sobre ese documento sin tener que pasar la referencia por todos lados.

Cuando el usuario haya editado o creado un lugar destacado y pulse sobre el botón de confirmación, dependiendo de la acción con la que se haya llamado al fragment, se editará un lugar destacado o se añadirá uno nuevo. Además eliminaremos el fragment de la vista del usuario con `getParentFragmentManager().popBackstack()`.

Llamadas a métodos de ayuda estáticos para la gestión de lugares destacados.

```
public class Operations{
    public static void addMainPlace(final LugarDestacado lugar, final Uri image) {
        crearRecursivamente(true);
        final String randomString = getRandomString();
        lugar.setImage(randomString);
        if (image == null) {
            uploadImage(randomString, DEFAULT_IMAGE_RESOURCE)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    mainPlaceCollection.document(randomString).set(lugar);
                }
            });
        } else {
            uploadImage(randomString, image)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    mainPlaceCollection.document(randomString).set(lugar);
                }
            });
        }
    }

    public static void updateMainPlace(final LugarDestacado newLugar, final String key, Uri image) {
        if (image == null) {
            newLugar.setImage(DEFAULT_URL);
            updateMainPlace(newLugar, key);
        } else {
            uploadImage(newLugar.getImage(), image)
            .addOnCompleteListener(new OnCompleteListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {
                    updateMainPlace(newLugar, key);
                }
            });
        }
    }

    private static void updateMainPlace(LugarDestacado newLugar, String key) {
        mainPlaceCollection.document(key).set(newLugar).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(applicationContext,
                "No se ha podido actualizar la ciudad", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

El método `onCreateView`, otra vez, parecido a los anteriores, solo que esta vez no hay un recycler que cargar y no hay animaciónes que realizar.

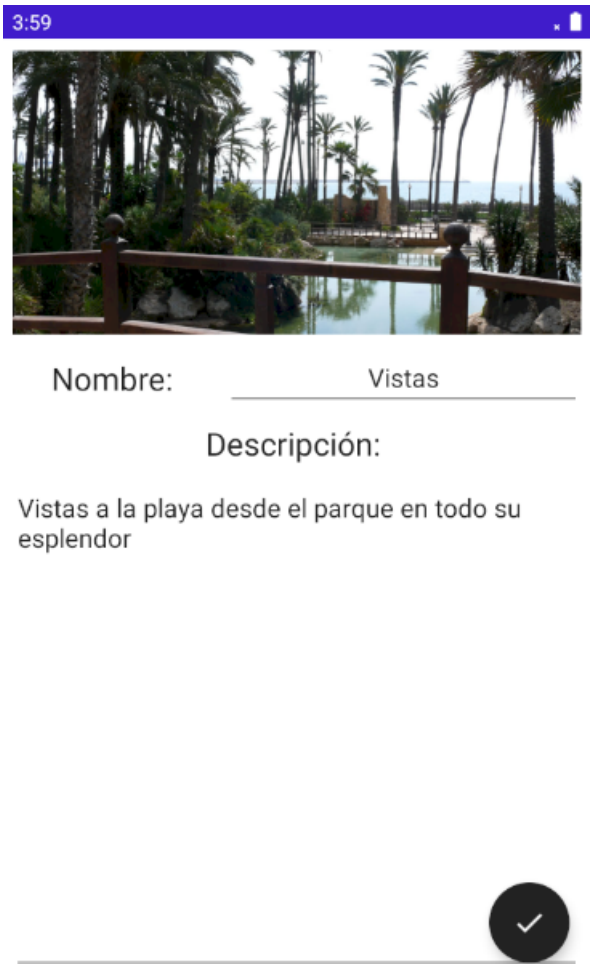
```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    View v = inflater.inflate(R.layout.fragment_lugar_destacado, container, false);
    nombre = v.findViewById(R.id.textNombreLugarFrag);
    descripcion = v.findViewById(R.id.textDescripcionLugar);
    imagen = v.findViewById(R.id.imagenLugarDestacadoFrag);
    if (accion == Operations.Accion.EDIT_REQUEST) {
        FirebaseStorage.getInstance().getReference(key).getDownloadUrl()
        .addOnCompleteListener(new OnCompleteListener<Uri>() {
            @Override
            public void onComplete(@NonNull Task<Uri> task) {
                if (task.isSuccessful()) {
                    selectedImage = task.getResult();
                }
            }
        });
        nombre.setText(lugar.getNombre());
        descripcion.setText(lugar.getDescripcion());
        if (!lugar.getImage().equals("") && lugar.getImage() != null) {
            Operations.loadIntoImageView(FirebaseStorage.getInstance().getReference(lugar.getImage()), imagen);
        }
    } else {
        key = Operations.newId();
        Operations.loadIntoImageView(Operations.DEFAULT_IMAGE, imagen);
    }
    Operations.mainPlaceDocument = Operations.mainPlaceCollection.document(key);

    imagen.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
            intent.setType("image/*");
            if (intent.resolveActivity(getActivity().getPackageManager()) != null) {
                startActivityForResult(intent, COD_ELEGIR_IMAGEN);
            }
        }
    });

    v.findViewById(R.id.fabLugaresFragment).setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            final LugarDestacado lugar = generarLugarDestacado();
            if (accion == Operations.Accion.ADD_REQUEST) {
                Operations.addMainPlace(lugar, selectedImage);
            } else {
                Operations.updateMainPlace(lugar, key, selectedImage);
            }
            getParentFragmentManager().popBackStack();
        }
    });

    return v;
}

private LugarDestacado generarLugarDestacado() {
    return new LugarDestacado(key, nombre.getText().toString(), descripcion.getText().toString());
}
```


3:59
Nombre:
Vistas
Descripción:
Vistas a la playa desde el parque en todo su esplendor