EJERCICIO 18

Objetivo: Solicitar fecha desde un DatePicker.

Pasos generales:

Cuando el usuario haga clic sobre un **EditText de solo lectura** (donde necesitamos la fecha),
Vamos a **mostrar un DialogFragment** con un DatePicker en su interior, y
Cuando el usuario seleccione una fecha, vamos a **capturar la fecha ingresada**.

1. Escribir el siguiente código en el segundo fragmento:

```
<EditText
    android:id="@+id/etPlannedDate"
    android:hint="report_planned_date"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="date"
    android:focusable="false"
    android:clickable="true"
    android:maxLines="1" />
```

2. Asociar un evento de clic al EditText, escribir el siguiente código en el Secondfragment.java

```
package com.example.ejemplo18;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.navigation.fragment.NavHostFragment;

import com.example.ejemplo18.databinding.FragmentSecondBinding;

public class SecondFragment extends Fragment {

    private FragmentSecondBinding binding;

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

        binding = FragmentSecondBinding.inflate(inflater,
container, false);
```

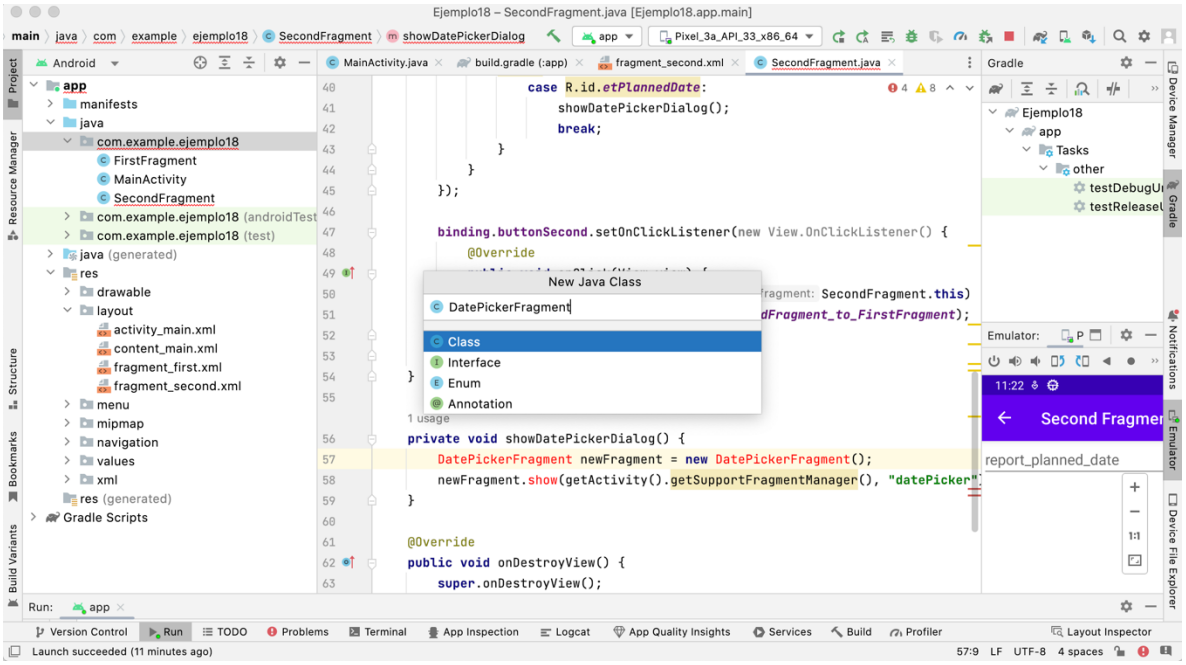
```
        return binding.getRoot();

    }

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

        EditText etPlannedDate = (EditText)
view.findViewById(R.id.etPlannedDate);


        binding.etPlannedDate.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                switch (view.getId()) {
                    case R.id.etPlannedDate:
                        showDatePickerDialog();
                        break;
                }
            }
        });

        binding.buttonSecond.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

NavHostFragment.findNavController(SecondFragment.this)

.navigate(R.id.action_SecondFragment_to_FirstFragment);
            }
        });
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }

}
```

3. Mostrar el DatePicker en un dialog

usando `getActivity()` porque estoy usando el EditText desde un Fragment.

Si en tu caso estás en un Activity, entonces puedes usar directamente `getSupportFragmentManager()`.

4. Definir la clase DatePickerFragment



Escribir el siguiente código en la clase:

```
package com.example.ejemplo18;

import android.app.DatePickerDialog;
import android.app.Dialog;
import android.os.Bundle;
import android.widget.DatePicker;

import androidx.fragment.app.DialogFragment;

import java.util.Calendar;

public class DatePickerFragment extends DialogFragment implements
DatePickerDialog.OnDateSetListener {

        @Override
        public Dialog onCreateDialog(Bundle savedInstanceState) {
            // Use the current date as the default date in the picker
            final Calendar c = Calendar.getInstance();
            int year = c.get(Calendar.YEAR);
            int month = c.get(Calendar.MONTH);
            int day = c.get(Calendar.DAY_OF_MONTH);

            // Create a new instance of DatePickerDialog and return it
            return new DatePickerDialog(getActivity(), this, year, month,
day);
        }

        public void onDateSet(DatePicker view, int year, int month, int
day) {
            // Do something with the date chosen by the user
        }
```

```
    }
}
```

Lo que esta clase hace es instanciar un `DatePickerDialog` y pre-seleccionar la fecha actual.

En su interior también podemos ver un método `onDateSet`. Este método es invocado cada vez que el usuario cambia la fecha en el DatePicker.

5. Capturar la fecha seleccionada

La interfaz DatePickerDialog.OnDateSetListener es la encargada de invocar el método onDateSet cada vez que una nueva fecha es seleccionada.

Pero si el método está definido sobre el mismo DatePickerFragment no nos resulta muy útil.

Nuestra intención es acceder a la fecha seleccionada desde nuestro fragment o activity (desde donde sea que estemos creando nuestro DatePickerFragment).

Para lograr ello, lo que tenemos que hacer es definir el listener en la clase que hará uso del DatePickerFragment, mas no en el mismo DatePickerFragment.

¿En dónde es que se está definiendo el listener actualmente y dónde es que debe definirse?

El listener (que escucha el evento de cambio de fecha) se está definiendo en nuestra clase DatePickerFragment.

En el método onCreateDialog estamos devolviendo una instancia de DatePickerDialog, tal como aquí se muestra:

```
package com.example.ejemplo18;

import android.app.DatePickerDialog;
import android.app.Dialog;
import android.os.Bundle;
import android.widget.DatePicker;

import androidx.fragment.app.DialogFragment;

import java.util.Calendar;

public class DatePickerFragment extends DialogFragment implements
DatePickerDialog.OnDateSetListener {
    private DatePickerDialog.OnDateSetListener listener;

    public static DatePickerFragment
newInstance(DatePickerDialog.OnDateSetListener listener) {
```

```
        DatePickerFragment fragment = new DatePickerFragment();
        fragment.setListener(listener);
        return fragment;
    }

    public void setListener(DatePickerDialog.OnDateSetListener listener)
{
        this.listener = listener;
    }

        @Override
        public Dialog onCreateDialog(Bundle savedInstanceState) {
            // Use the current date as the default date in the picker
            final Calendar c = Calendar.getInstance();
            int year = c.get(Calendar.YEAR);
            int month = c.get(Calendar.MONTH);
            int day = c.get(Calendar.DAY_OF_MONTH);

            // Create a new instance of DatePickerDialog and return it
            return new DatePickerDialog(getActivity(), this, year, month,
day);
        }
    }
```

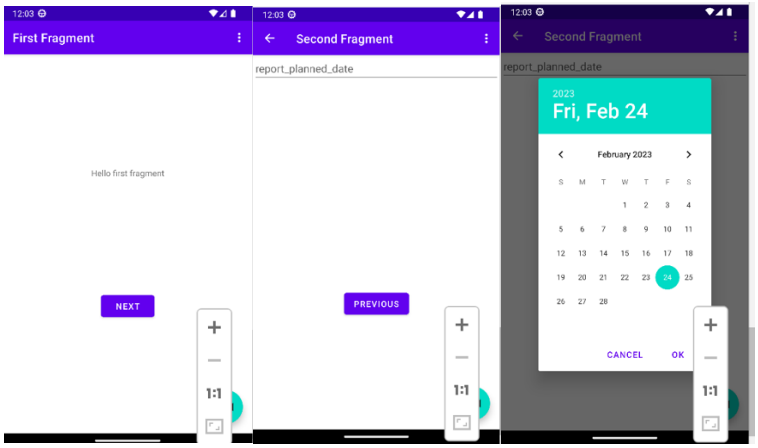
6. Nuestra clase `DatePickerFragment` nos va a permitir pedir una fecha al usuario, ¡tantas veces como sea necesario! Solo tenemos que instanciar dicha clase y decirle qué hacer con la fecha escogida:

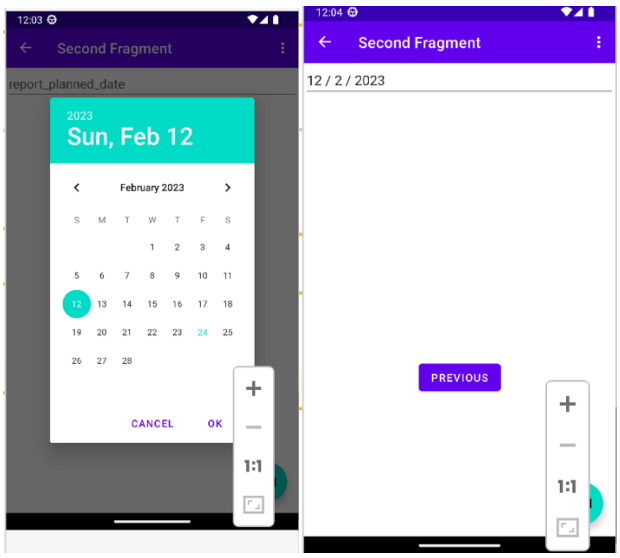
```
7. package com.example.ejemplo18;

   import android.app.DatePickerDialog;
   import android.os.Bundle;
   import android.view.LayoutInflater;
   import android.view.View;
   import android.view.ViewGroup;
   import android.widget.DatePicker;
   import android.widget.EditText;
   import android.widget.Toast;

   import androidx.annotation.NonNull;
   import androidx.fragment.app.Fragment;
   import androidx.navigation.fragment.NavHostFragment;

   import com.example.ejemplo18.databinding.FragmentSecondBinding;

   public class SecondFragment extends Fragment {

       private FragmentSecondBinding binding;

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

           binding = FragmentSecondBinding.inflate(inflater,
   container, false);
```

```
        return binding.getRoot();

    }

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

        EditText etPlannedDate = (EditText)
view.findViewById(R.id.etPlannedDate);


        binding.etPlannedDate.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //switch (view.getId()) {
                   // case R.id.etPlannedDate:
                        Toast.makeText(view.getContext(),"aqui
estamos", Toast.LENGTH_SHORT).show();
                        showDatePickerDialog();

            }
        });

        binding.buttonSecond.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

NavHostFragment.findNavController(SecondFragment.this)

.navigate(R.id.action_SecondFragment_to_FirstFragment);
            }
        });
    }

    /*private void showDatePickerDialog() {
        DatePickerFragment newFragment = new DatePickerFragment();
        newFragment.show(getActivity().getSupportFragmentManager(),
"datePicker");

    }*/


    private void showDatePickerDialog() {
        DatePickerFragment newFragment =
DatePickerFragment.newInstance(new
DatePickerDialog.OnDateSetListener() {

            @Override
            public void onDateSet(DatePicker datePicker, int year,
int month, int day) {
                EditText etPlannedDate = (EditText)
getView().findViewById(R.id.etPlannedDate);

                // +1 because January is zero
```

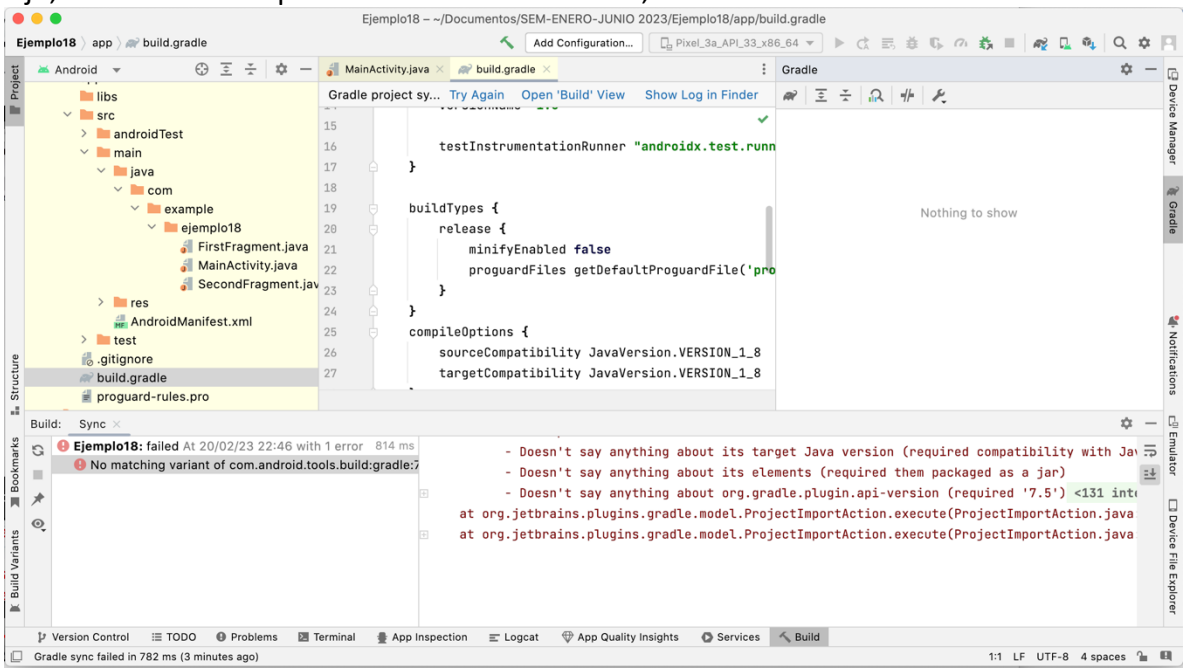
```
                final String selectedDate = day + " / " + (month+1)
+ " / " + year;
                etPlannedDate.setText(selectedDate);
                Toast.makeText(getView().getContext(),"aqui
estamos"+ selectedDate, Toast.LENGTH_SHORT).show();

            }
        });

        newFragment.show(getActivity().getSupportFragmentManager(),
"datePicker");
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }

}
```

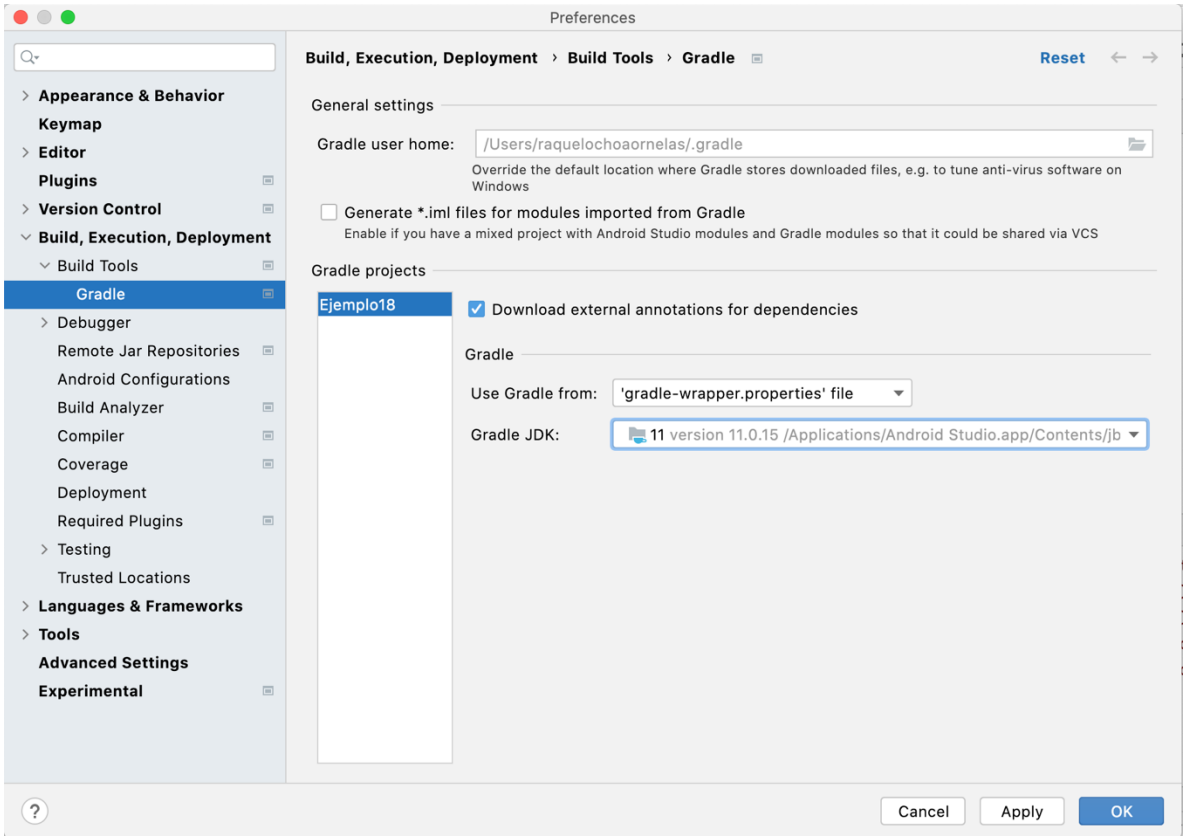
Ejecutar la aplicación:





Ojo, cuando de marque error de Gradle con Java, hacer esto:





Posteriormente dar clic en Build nuevamente.