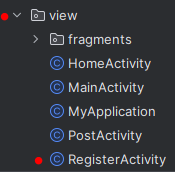
**ACTIVIDAD N°12 :**

**Clase Java de la vista: RegisterActivity**

1. RegisterActivity (registrarse):



**MÉTODOS**

1. **onCreate(…)**

protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityRegisterBinding.inflate(getLayoutInflater());  
 setContentView(binding.getRoot());  
 viewModel = new ViewModelProvider(this).get(RegisterViewModel.class);  
 viewModel.getRegisterResult().observe(this, new Observer<String>() {  
 @Override  
 public void onChanged(String result) {  
 showToast(result);  
 }  
 });  
 manejarEventos();  
}

* **@Override**: Indica que el método onCreate está sobrescribiendo un método de la clase base Activity.
* **protected void onCreate(Bundle savedInstanceState)** se llama cuando la actividad se está iniciando. Bundle savedInstanceState contiene el estado previamente guardado de la actividad, si existe.
* **super.onCreate(savedInstanceState)** llama al método onCreate de la clase base Activity para realizar la configuración inicial de la actividad.
* **binding = ActivityMainBinding.inflate(getLayoutInflater())** Utiliza el objeto LayoutInflater para crear una instancia de ActivityMainBinding, que es una clase generada automáticamente para la vinculación de vistas (View Binding).
* **setContentView(binding.getRoot())** establece la vista raíz de la actividad a la vista inflada por el binding.
* **viewModel = new ViewModelProvider(this, new MainViewModelFactory(this)).get(MainViewModel.class)** crea una instancia de ViewModelProvider para obtener el MainViewModel utilizando una fábrica personalizada MainViewModelFactory.
* **manejarEventos()** llama al método  manejarEventos, que configura los manejadores de eventos para la actividad.

**2. private void manejarEventos()**

El código manejarEventos() define los eventos de clic para un botón para registrar al usuario y un ícono para volver a la pantalla de inicio de sesión.

private void manejarEventos() {  
 // Evento volver a login  
 binding.circleImageBack.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 finish(); // Cierra la actividad actual (vuelve a login)  
 }  
 });  
  
 // Evento de registro  
 binding.btRegistrar.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 realizarRegistro(); // Realiza el proceso de registro  
 }  
 });  
}

**binding.circleImageBack.setOnClickListener**(...)está escuchando al elemento*circleImageBack,*  al hacer clic sobre este elemento llama al método finish() para salir de la actividad actual.

**binding.btRegistrar.setOnClickListener**(...) está escuchando al *btRegistrar*, al hacer clic sobre este botón llama al método realizarRegistro().

El método **realizarRegistro()** se encarga de:

* Validar los datos ingresados por el usuario (usando métodos estáticos de la clase Validaciones)
* Mostrar mensajes de error si hay datos incorrectos.
* Crear un objeto de usuario válido.
* Delegar el proceso de registro al *ViewModel*.

**3. private void showToast(String message)**

private void showToast(String message) {  
 Toast.makeText(MainActivity.this, message, Toast.LENGTH\_LONG).show();  
}

**Toast.makeText**: Crea un Toast, que es una pequeña ventana emergente que muestra información al usuario.

**CÓDIGO**

import android.os.Bundle;  
import android.view.View;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.lifecycle.Observer;  
import androidx.lifecycle.ViewModelProvider;  
import com.example.appchat.databinding.ActivityRegisterBinding;  
import com.example.appchat.model.User;  
import com.example.appchat.util.Validaciones;  
import com.example.appchat.viewmodel.RegisterViewModel;  
public class RegisterActivity extends AppCompatActivity {  
 private ActivityRegisterBinding binding;  
 private RegisterViewModel viewModel;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityRegisterBinding.inflate(getLayoutInflater());  
 setContentView(binding.getRoot());  
 viewModel = new ViewModelProvider(this).get(RegisterViewModel.class);  
 viewModel.getRegisterResult().observe(this, new Observer<String>() {  
 @Override  
 public void onChanged(String result) {  
 showToast(result);  
 }  
 });  
 manejarEventos();  
 }  
  
 private void manejarEventos() {  
 // Evento volver a login  
 binding.circleImageBack.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 finish(); // Cierra la actividad actual (vuelve a login)  
 }  
 });  
  
 // Evento de registro  
 binding.btRegistrar.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 realizarRegistro(); // Realiza el proceso de registro  
 }  
 });  
 }  
  
 private void realizarRegistro() {  
 String usuario = binding.itUsuario.getText().toString().trim();  
 String email = binding.itEmail.getText().toString().trim();  
 String pass = binding.itPassword.getText().toString().trim();  
 String pass1 = binding.itPassword1.getText().toString().trim();  
  
 // Validaciones de entrada  
 if (!Validaciones.validarTexto(usuario)) {  
 showToast("Usuario incorrecto");  
 return;  
 }  
  
 if (!Validaciones.validarMail(email)) {  
 showToast("El correo no es válido");  
 return;  
 }  
  
 String passError = Validaciones.validarPass(pass, pass1);  
 if (passError != null) {  
 showToast(passError);  
 return;  
 }  
  
  
 User user = new User(usuario, email, pass);  
 viewModel.register(user);  
 }  
  
 private void showToast(String message) {  
 Toast.makeText(RegisterActivity.this, message, Toast.LENGTH\_LONG).show();  
 }  
}