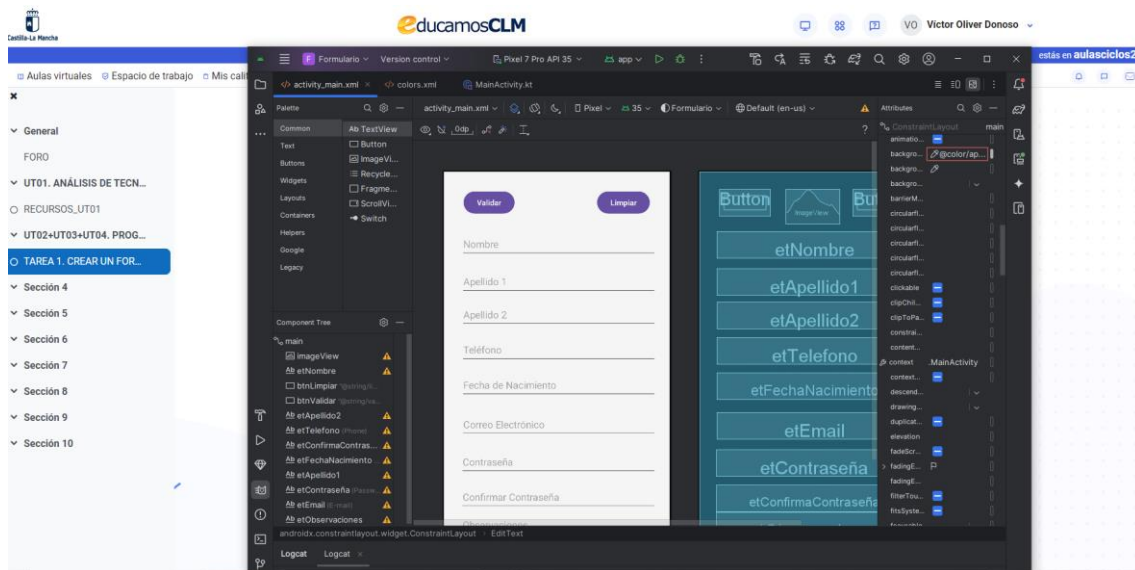
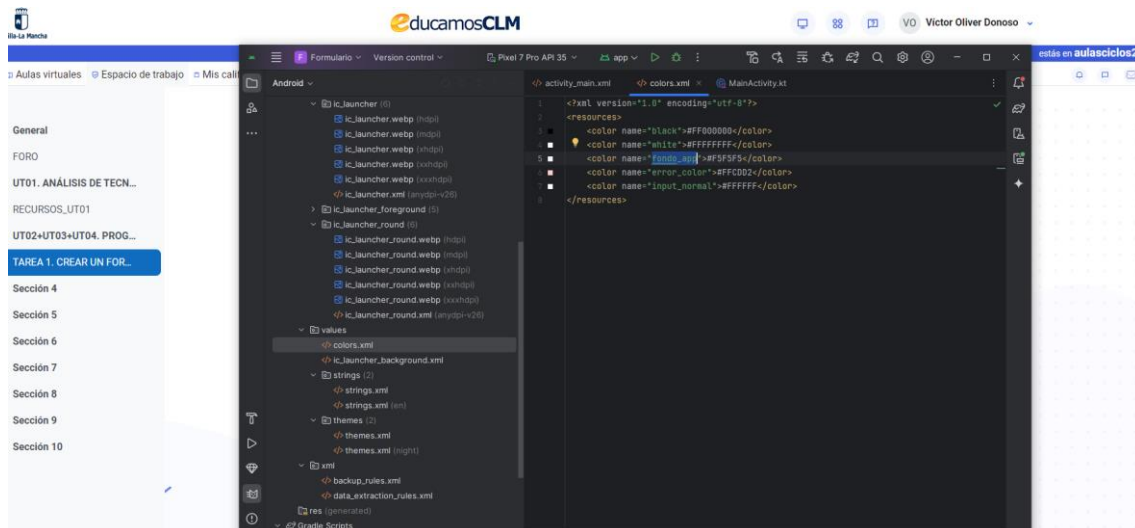


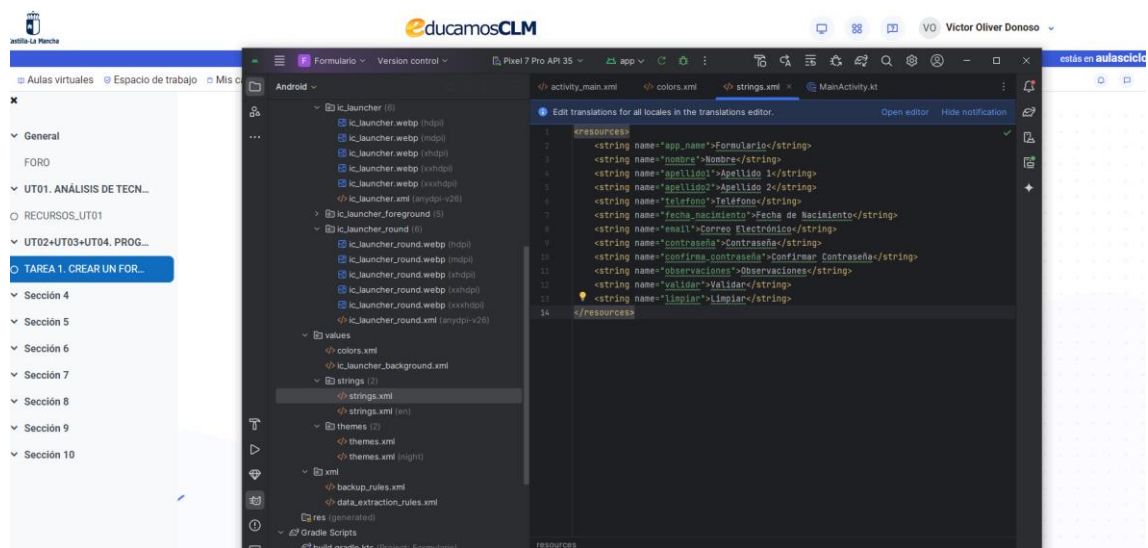
## Formulario



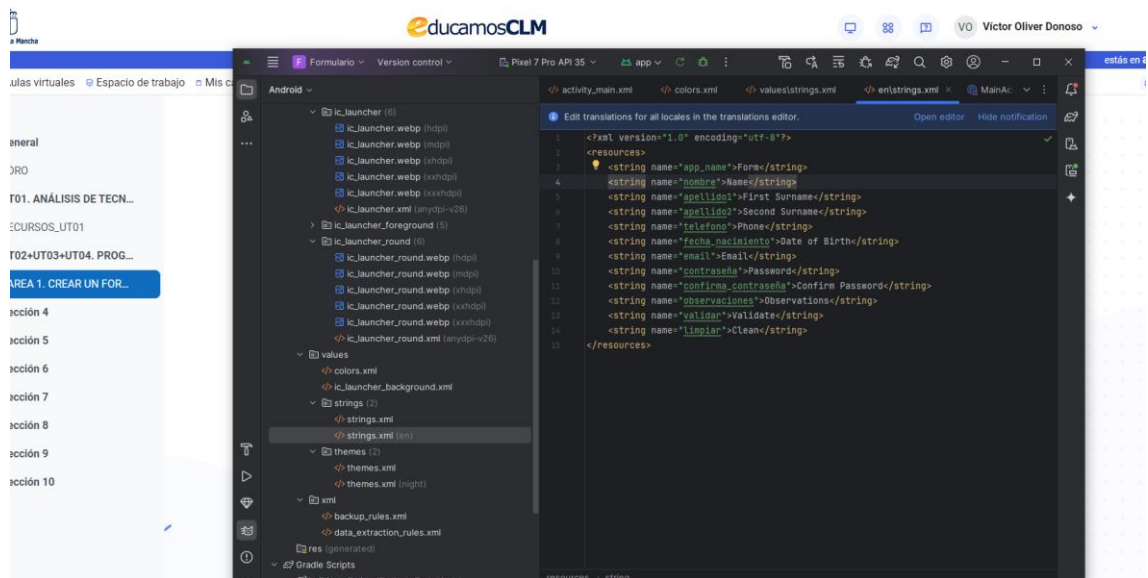
Creando la parte visual y metiendo los atributos.



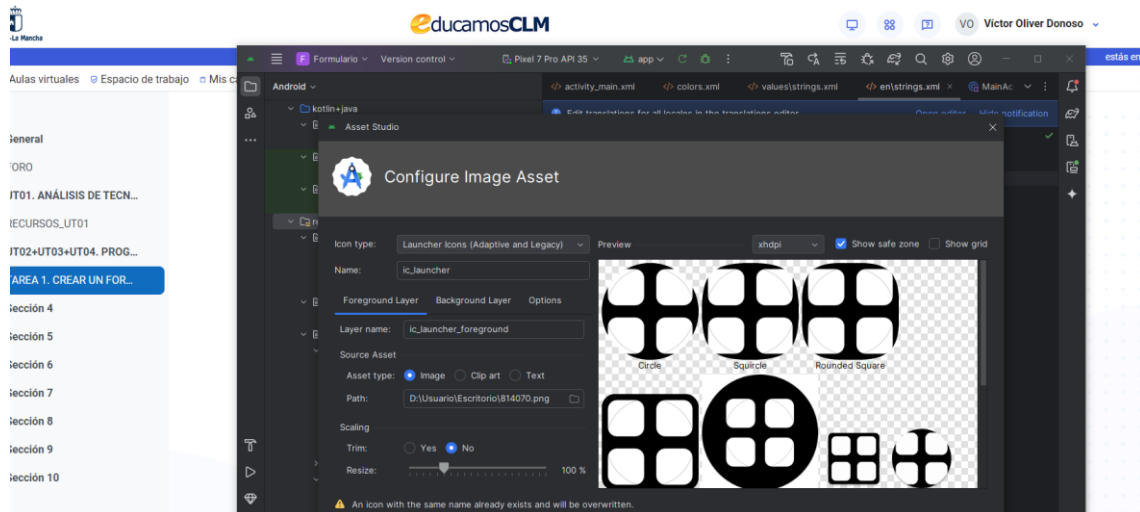
Colores creados en los recursos



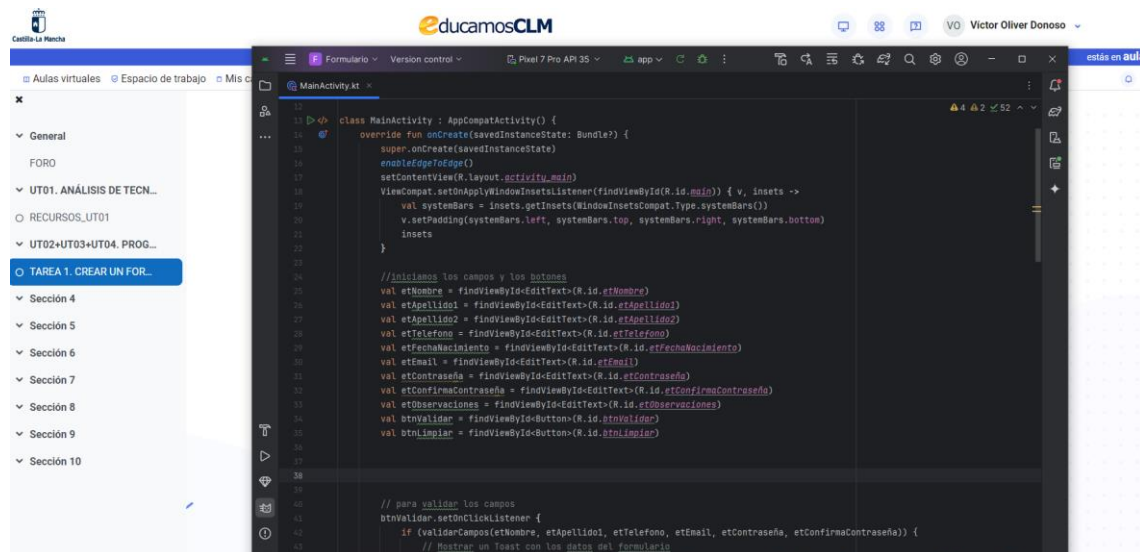
## Traducciones en castellano



## En ingles



Icono de la aplicación cambiado.



Main

educamosCLM

Formulario Version control Pixel 7 Pro API 35 app Victor Oliver Donoso

MainActivity.kt

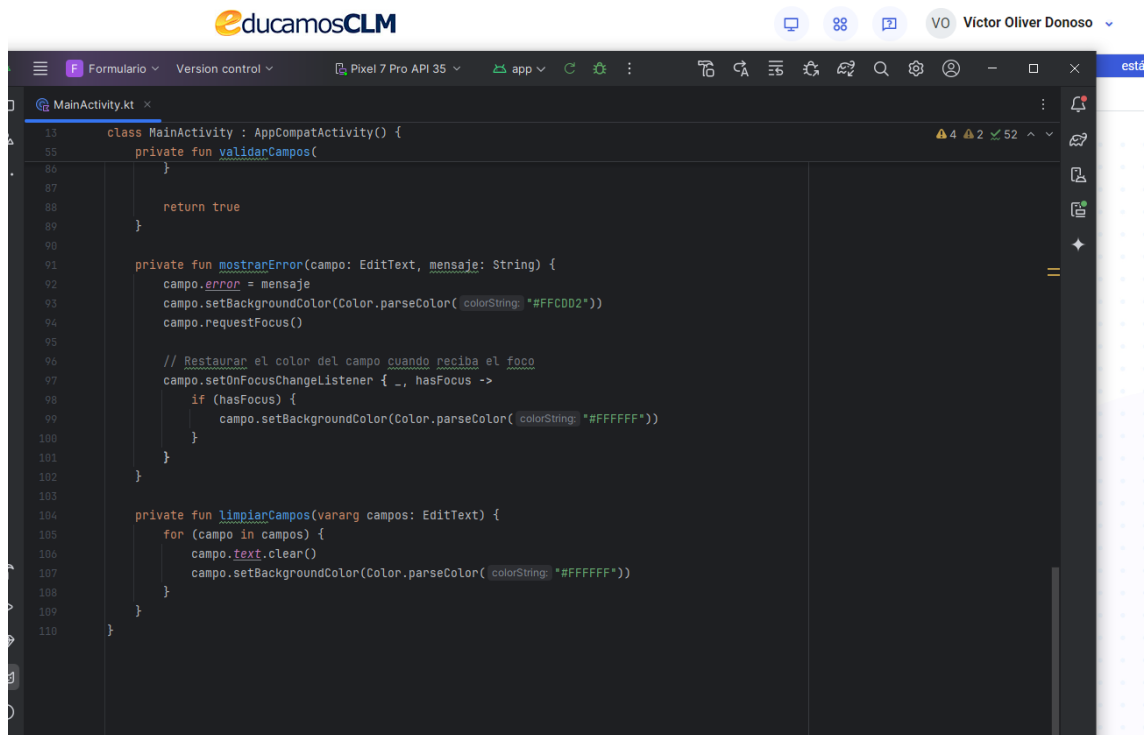
```
13 class MainActivity : AppCompatActivity() {
14     override fun onCreate(savedInstanceState: Bundle?) {
38
39
40         // para validar los campos
41         btnValidar.setOnClickListener {
42             if (validarCampos(etNombre, etApellido1, etTelefono, etEmail, etContraseña, etConfirmaContraseña)) {
43                 // Mostrar un Toast con los datos del formulario
44                 val datos = "Nombre: ${etNombre.text}\nApellido1: ${etApellido1.text}\nApellido2: ${etApellido2.text}\nTeléfono: ${etTelefono.text}\nEmail: ${etEmail.text}\nContraseña: ${etContraseña.text}\nConfirmar Contraseña: ${etConfirmaContraseña.text}"
45                 Toast.makeText(this, datos, Toast.LENGTH_LONG).show()
46             }
47         }
48
49         // Para limpiar el boton
50         btnLimpiar.setOnClickListener {
51             limpiarCampos(etNombre, etApellido1, etApellido2, etTelefono, etFechaNacimiento, etEmail, etContraseña, etConfirmaContraseña)
52         }
53     }
54
55     private fun validarCampos(
56         etNombre: EditText,
57         etApellido1: EditText,
58         etTelefono: EditText,
59         etEmail: EditText,
60         etPassword: EditText,
61         etConfirmPassword: EditText
62     ): Boolean {
63         if (etNombre.text.isBlank()) {
64             mostrarError(etNombre, mensaje: "El nombre es obligatorio")
65             return false
66         }
67     }
68 }
```

educamosCLM

Formulario Version control Pixel 7 Pro API 35 app Victor Oliver Donoso

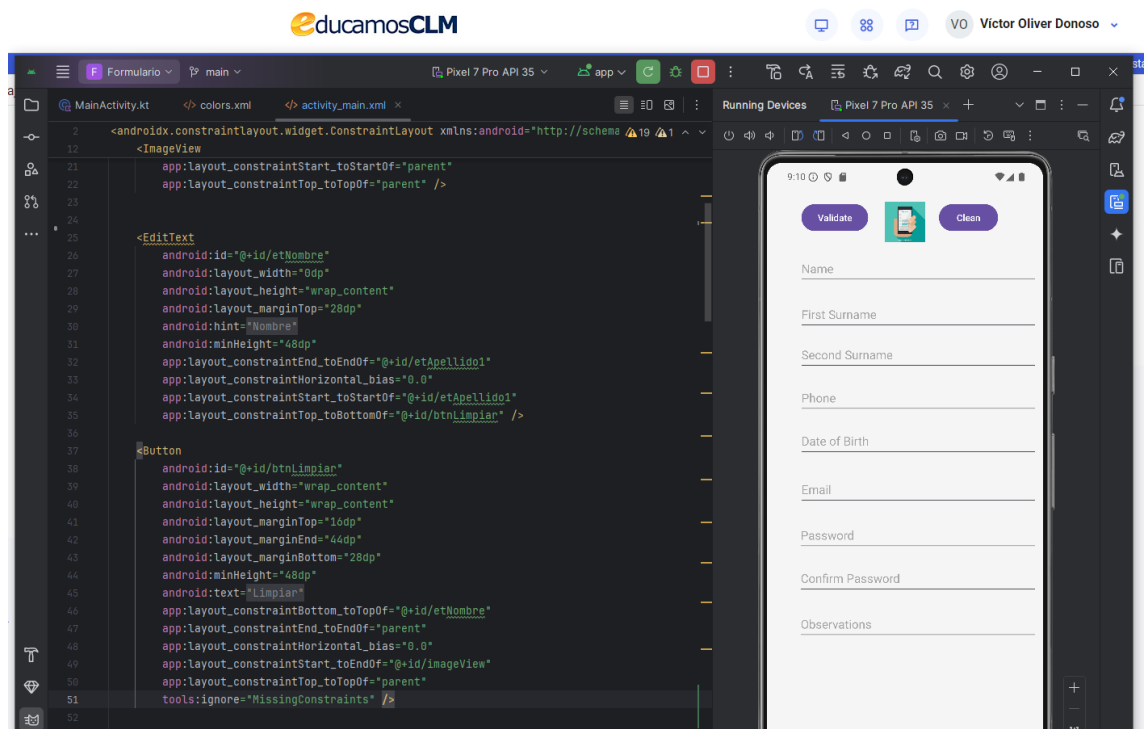
MainActivity.kt

```
13 class MainActivity : AppCompatActivity() {
55     private fun validarCampos(
62     ): Boolean {
63         if (etNombre.text.isBlank()) {
64             mostrarError(etNombre, mensaje: "El nombre es obligatorio")
65             return false
66         }
67
68         if (etApellido1.text.isBlank()) {
69             mostrarError(etApellido1, mensaje: "El primer apellido es obligatorio")
70             return false
71         }
72
73         if (!etTelefono.text.matches(Regex(pattern: "\\d{9}"))) {
74             mostrarError(etTelefono, mensaje: "El teléfono debe contener 9 dígitos")
75             return false
76         }
77
78         if (!etEmail.text.matches(Regex(pattern: "[a-zA-Z0-9-._-]+@[a-z]+\\.([a-z]+)"))) {
79             mostrarError(etEmail, mensaje: "Formato de correo inválido")
80             return false
81         }
82
83         if (etPassword.text.toString() != etConfirmPassword.text.toString()) {
84             mostrarError(etPassword, mensaje: "Las contraseñas no coinciden")
85             return false
86         }
87
88         return true
89     }
90 }
```



```
13 class MainActivity : AppCompatActivity() {
14     private fun validarCampos() {
15     }
16
17     return true
18 }
19
20 private fun mostrarError(campo: EditText, mensaje: String) {
21     campo.error = mensaje
22     campo.setBackgroundColor(Color.parseColor(colorString: "#FFC000"))
23     campo.requestFocus()
24
25     // Restaurar el color del campo cuando reciba el foco
26     campo.setOnFocusChangeListener { _, hasFocus ->
27         if (hasFocus) {
28             campo.setBackgroundColor(Color.parseColor(colorString: "#FFFFFF"))
29         }
30     }
31 }
32
33 private fun limpiarCampos(vararg campos: EditText) {
34     for (campo in campos) {
35         campo.text.clear()
36         campo.setBackgroundColor(Color.parseColor(colorString: "#FFFFFF"))
37     }
38 }
39 }
```

Todo el main.



Ejecutandose