Enunciat

Part1:

Segueix les indicacions del següent tutorial: https://cursokotlin.com/capitulo-18-componentes-personalizados-android/

Podeu descarregar el projecte des de GitHub, però és molt probable que no pugueu executar-lo directament. La meva recomanació és que creeu un projecte nou net i aneu afegint el que va apareixent al tutorial.

Un cop tingueu el projecte funcionant respon les següents preguntes:

1. Com es fa per heretar un objecte en Kotlin?
2. Què succeeix quan especifiquem que la nostra classe implementarà AppCompatEditText?
3. A l'exemple fan servir la funció background.setColorFilter però Android Studio ens marca aquest mètode com a deprecated. Busca una alternativa actualitzada per canviar el color del background.
4. Si ens fixem en el codi de projecte que fa servir el tutorial, fa servir una tècnica de binding, però no és la que hem vist a classe. Com ho fa?

Resol les preguntes anteriors i crea un camp de text personalitzat que no permeti introduir un número acabat en 0.

Part 2:

Un cop acabat la primera part, implementa la solució de la segona part del tutorial que pots trobar a: https://cursokotlin.com/capitulo-19-componentes-personalizados-android-kotlin/

A continuació respon les següents preguntes:

1. Com fa per estendre més d'una classe?
2. A on especifica quins elements tindrà el nou component? Quins són aquest dos elements?
3. Quina diferència hi ha entre afterTextChange(), beforeTextChange() i onTextChanged()?

Implementa un element View que en comptes de validar l'email, validi el DNI amb una expressió regular. Fes algun canvi també en el disseny visual.

Entrega

Document .pdf amb les respostes a les preguntes plantejades de les dues parts. A més ha d'incloure el link al teu repositori Github amb el projecte d'Android Studio.

Links al Github

Link al repositori principal:
https://github.com/cristianjimenezhernandezdev/Android_Programacio_DAM2b.git

LINK Concret del PROJECTE

https://github.com/cristianjimenezhernandezdev/CustomViewAndroidCristian.git

Part 1

1. Com es fa per heretar un objecte en Kotlin?

En Kotlin una classe hereta d'una altra fent servir els dos punts : darrere del nom de la classe, i cridant el constructor de la classe pare.

2. Què succeeix quan especifiquem que la nostra classe implementarà AppCompatEditText?

   Vol dir que la classe hereta de AppCompatEditText, la classe es un en un EditText personalitzat amb les funcions base però a més es pot afegir components i coses extra.

3. A l'exemple fan servir la funció background.setColorFilter però Android Studio ens marca aquest mètode com a deprecated. Busca una alternativa actualitzada per canviar el color del background.

   A mi no me l'ha marcat com deprecated, però tot i això, com diu l'exercici, faré servir la nova forma que he trobat és fer servir: backgroundTintList. I així he canviat tot al codi

   backgroundTintList = android.content.res.ColorStateList.valueOf(ContextCompat.getColor(context, R.color.blue)) i vermell també.

   i

etMail.backgroundTintList = android.content.res.ColorStateList.valueOf(errorColor) i els altres components que es necessitin.

4. Si ens fixem en el codi de projecte que fa servir el tutorial, fa servir una tècnica de binding, però no és la que hem vist a classe. Com ho fa?
Fa servir el findViewById() que és una maner més manual, és semblant al findObject del Unity on va a buscar cada element en comptes de fer-ho automàticament amb el binding.
És mes directe i senzill, però pot ser difícil de gestionar quan hi ha molt i pot ser fàcil que tingui fallos.

Resol les preguntes anteriors i crea un camp de text personalitzat que no permeti introduir un número acabat en 0.

Primer creem al layout un xml basic amb editText i textView

Ara fem el CustomView copiant del EmailValidator i retocant per estalviar.
Canvio:

```
private val etNumero: EditText  // abans etMail
   private val tvErrorNumero: TextView  // abans tvErrorCode
```

Llavors esborro el contingut del onTextChanged i dins del afterTextChanged poso el seguent codi:

```
override fun afterTextChanged(s: Editable?) {
      val text = s.toString()

      if (text.isNotEmpty() && text.last() == '0') {
         tvErrorNumero.visibility = View.VISIBLE
         etNumero.backgroundTintList =
ColorStateList.valueOf(errorColor)

         etNumero.removeTextChangedListener(this)
         etNumero.setText(text.dropLast(1))
         etNumero.setSelection(etNumero.text.length)
```

```
            etNumero.addTextChangedListener(this)
        } else {
            tvErrorNumero.visibility = View.INVISIBLE
            etNumero.backgroundTintList =
ColorStateList.valueOf(successColor)
        }
```

Aquest codi valida el text que l'usuari escriu i comprova si l'últim caràcter és un zero; en aquest cas, es considera un valor no permès i per això es mostra el missatge d'error, es pinta el camp amb el color d'error i es corregeix automàticament el contingut eliminant l'últim dígit. Per evitar que aquesta modificació interna torni a activar el TextWatcher i provoqui un bucle infinit, primer es desactiva temporalment el listener, s'actualitza el text i es recol·loca el cursor al final, i després es torna a activar el listener. Si el text no acaba en zero, s'oculta el missatge d'error i es canvia el color del camp indicant que la validació és correcta.

Part 2

1. Com fa per estendre més d'una classe?
2. A on especifica quins elements tindrà el nou component? Quins són aquest dos elements?
3. Quina diferència hi ha entre afterTextChange(), beforeTextChange() i onTextChanged()?

1. Una classe només pot estendre una única classe base, però pot implementar més d'una. Per fer-ho, despres del nom de la classe s'especifica primer la classe pare i després posant comes les interfícies que vulguis. Així, encara que no es pugui heretar de més d'una classe, podem fer herència normal amb comportaments afegits mitjançant interfícies.

2. Els elements que formarà el nou component es defineixen dins del init del custom view, quan s'infla el layout XML que toca amb inflate(context, R.layout.nom_del_layout, this). Per exemple, el component aquest té dos elements principals: un EditText i un TextView aquests es troben amb el findViewById i passen a ser les parts visibles del component Custom.

3. Les tres funcions del TextWatcher corresponen a moments diferents del canvi de text: beforeTextChanged() s'executa just abans que el text es modifiqui i permet saber què hi havia abans del canvi; onTextChanged() s'executa mentre el text està canviant i és on normalment es fan validacions immediates; i afterTextChanged() s'executa un cop el text ja ha canviat definitivament. Les tres les has de mantenir encara que estiguin buides, perquè si no dona error.

Implementa un element View que en comptes de validar l'email, validi el DNI amb una expressió regular.

La manera de fer-ho és fer una validació que comprovi que el caràcter numero 9 sigui una lletra o més aviat que no sigui un numero.
Per fer-ho agafo de base els .kt i .xml de la validació del 0, ja que és la més semblant.
I com sempre haure de fer canvis en aquests i a més afegir la funcionalitat al main perquè ho mostri.

La clau de la lògica és aquí:

```
if (text.length == 9) {
        val lastChar = text.last()

        // Si l'últim caràcter és un dígit, és incorrecte
        if (lastChar.isDigit())
```

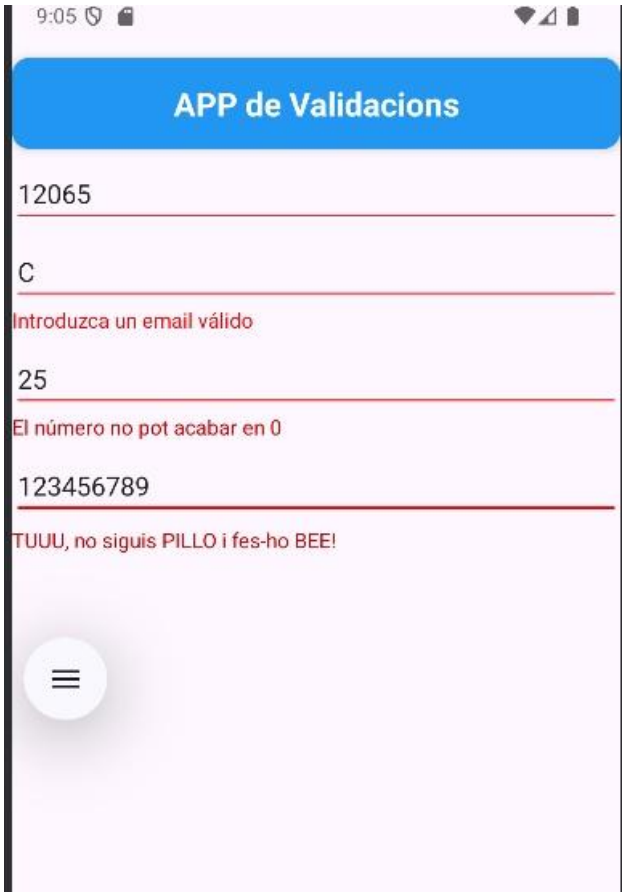
La funció isDigit és una funció de kotlin i el que fa és retornar una booleana de true o fals en funció de si el caràcter és un dígit del 0 al 9 o no. I en la funció li estem dient, Mira el novè caràcter. Si aquest és un Digit (0-9) llavors retorna true i si no retorna fals. Llavors amb això ja podem treballar perquè ens mostri colors i missatges com amb els exercicis anteriors.

*Nota de organització de codi. Els hint com que s'em duplicaven i em marcava warnings els he posat al Strings i els he indexat amb la @ als xml individuals en comptes de posar-los al main activity així ho centralitzo i evito fallos.

Com a canvi visual he posat el títol dins d'un requadre arrodonit en blau canviant codi al xml main fent servir els cornerRadius

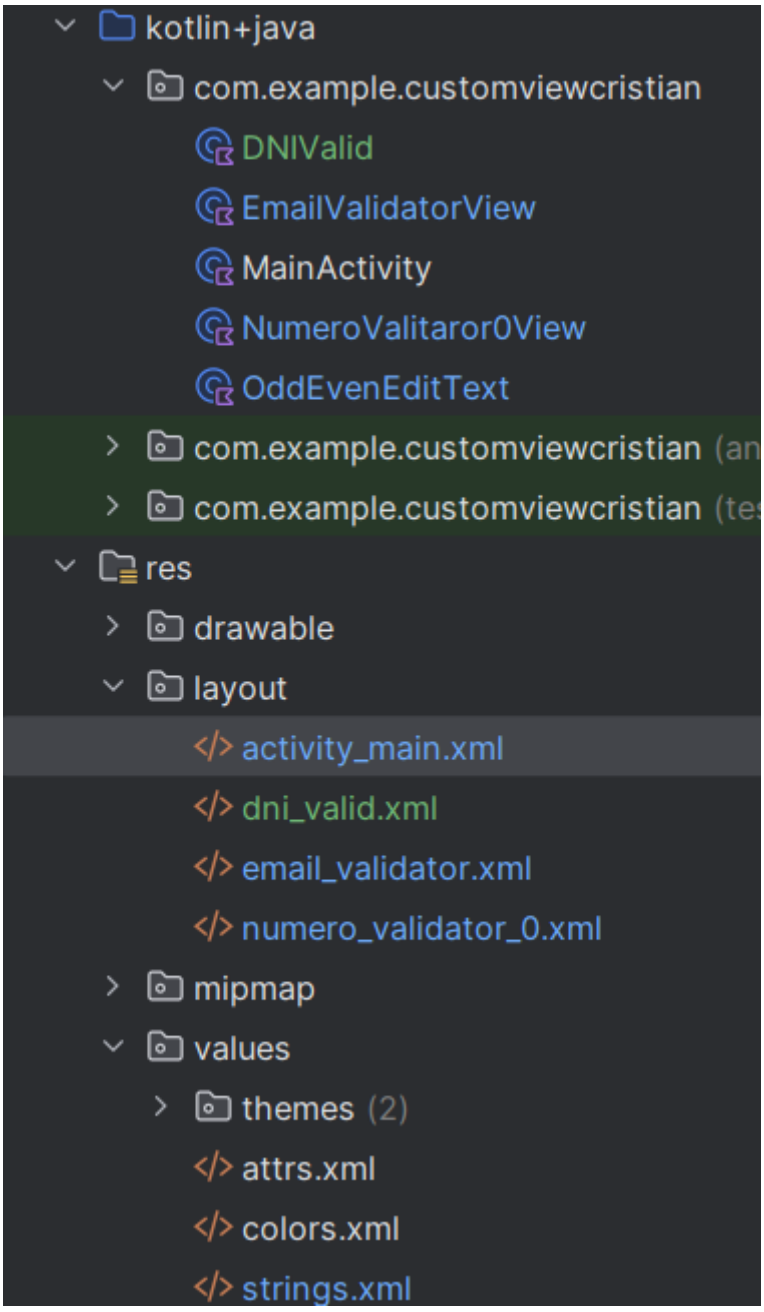
Annex:

Captures del Funcionament.


9:05
APP de Validacions
12065
C
Introduzca un email válido
25
El número no pot acabar en 0
123456789
TUUU, no siguis PILLO i fes-ho BEE!


9:06
APP de Validacions
120658
Cristian@campa.com
2565
12345678A

Codi complet de cada fitxer del programa:



DNIValid.kt

```
package com.example.customviewcristian
import android.content.Context
import android.content.res.ColorStateList
import android.text.Editable
import android.text.TextWatcher
import android.util.AttributeSet
import android.widget.EditText
import android.widget.RelativeLayout
import android.widget.TextView
import androidx.core.content.ContextCompat



class DNIValid(context: Context, attrs: AttributeSet) : RelativeLayout(context, attrs),
TextWatcher {

    var successColor: Int
    var errorColor: Int

    private val etDNI: EditText
    private val tvErrorDNI: TextView


    init {
        inflate(context, R.layout.dni_valid, this)

        etDNI = findViewById(R.id.etDNI)
        tvErrorDNI = findViewById(R.id.tvErrorDNI)

        // Configurar colors directament
        errorColor = ContextCompat.getColor(context, R.color.red)
        successColor = ContextCompat.getColor(context, R.color.green)

        // Assegurar que el text d'error està configurat
        if (tvErrorDNI.text.isNullOrEmpty()) {
            tvErrorDNI.text = "TUU, posa-ho bé. No siguis PILLO!"
        }

        // Configurar hint
        val hint = attrs.getAttributeValue("http://schemas.android.com/apk/res/android", "hint")
        if (!hint.isNullOrEmpty()) {
            etDNI.hint = hint
        }

        etDNI.addTextChangedListener(this)
    }



    override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
    }
```

```
    override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {

    }

    override fun afterTextChanged(s: Editable?) {
        val text = s.toString()

        // Validar DNI: ha de tenir 9 caràcters i l'últim ha de ser una lletra
        if (text.length == 9) {
            val lastChar = text.last()

            // Si el ultim és numero mostra error
            if (lastChar.isDigit()) {
                tvErrorDNI.visibility = VISIBLE
                etDNI.backgroundTintList = ColorStateList.valueOf(errorColor)
            } else {
                // mostra correcte si posa algo que no sigui numero
                tvErrorDNI.visibility = INVISIBLE
                etDNI.backgroundTintList = ColorStateList.valueOf(successColor)
            }
        } else {
            // Si no té 9 caràcters, no mostrem error però tampoc va bé
            tvErrorDNI.visibility = INVISIBLE
            etDNI.backgroundTintList = ColorStateList.valueOf(ContextCompat.getColor(context,
android.R.color.darker_gray))
        }
    }

}
```

EmailValidatorView.kt

```
package com.example.customviewcristian

import android.content.Context
import android.graphics.PorterDuff
import android.text.Editable
import android.text.TextWatcher
import android.util.AttributeSet
import android.view.View
import android.widget.EditText
import android.widget.RelativeLayout
import android.widget.TextView
import androidx.core.content.ContextCompat
import java.util.regex.Pattern
import kotlin.text.matches

class EmailValidatorView(context: Context, attrs: AttributeSet) : RelativeLayout(context, attrs),
TextWatcher {
```

```
    var successColor: Int // Color per a l'estat de validació correcta
    var errorColor: Int // Color per a l'estat de validació incorrecta

    private val etMail: EditText // Camp de text per introduir l'email
    private val tvErrorCode: TextView // Missatge d'error visible quan l'email no és vàlid

    init {
        // Inflem el layout associat a aquest component personalitzat
        inflate(context, R.layout.email_validator, this)

        etMail = findViewById(R.id.etMail)
        tvErrorCode = findViewById(R.id.tvErrorCode)

        // Obtenim els atributs personalitzats definits al XML
        val attributes = context.obtainStyledAttributes(attrs, R.styleable.EmailValidatorView)
        tvErrorCode.text = attributes.getString(R.styleable.EmailValidatorView_textError) // Text
d'error personalitzat
        errorColor = attributes.getColor(R.styleable.EmailValidatorView_underlineErrorColor,
ContextCompat.getColor(context, R.color.red))
        successColor = attributes.getColor(R.styleable.EmailValidatorView_underlineSuccessColor,
ContextCompat.getColor(context, R.color.green))
        attributes.recycle()

        // Configurar el hint si està definit al XML, si no posar un per defecte
        val hint = attrs.getAttributeValue("http://schemas.android.com/apk/res/android", "hint")
        if (!hint.isNullOrEmpty()) {
            etMail.hint = hint
        }
        // Si el hint del layout també està buit, posem un per defecte
        if (etMail.hint.isNullOrEmpty()) {
            etMail.hint = "Escriu un mail"
        }

        // Afegim un listener per escoltar els canvis en el text
        etMail.addTextChangedListener(this)
    }

    override fun afterTextChanged(s: Editable?) {
        // No fem
    }

    override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
        // No fem
    }

    override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
        // Patró per validar un email
        val pattern = Pattern.compile("\\b[\\w.%-]+@[-.\\w]+\\.[A-Za-z]{2,4}\\b")
        val matcher = pattern.matcher(s.toString())
        val valid = matcher.matches()

        if (valid) {
```

```
            // Si l'email és vàlid, amaguem el missatge d'error i canviem el color de la línia inferior
            tvErrorCode.visibility = View.INVISIBLE
            etMail.backgroundTintList = android.content.res.ColorStateList.valueOf(successColor)
        } else {
            // Si l'email no és vàlid, mostrem el missatge d'error i canviem el color de la línia inferior
            tvErrorCode.visibility = View.VISIBLE
            etMail.backgroundTintList = android.content.res.ColorStateList.valueOf(errorColor)
        }
    }

}
```

MainActivity.kt

```
package com.example.customviewcristian

import android.os.Bundle
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
            insets
        }
    }
}
```

NumeroValitaror0View.kt

```
package com.example.customviewcristian

import android.content.Context
import android.content.res.ColorStateList
import android.text.Editable
import android.text.TextWatcher
import android.util.AttributeSet
import android.view.View
import android.widget.EditText
import android.widget.RelativeLayout
import android.widget.TextView
import androidx.core.content.ContextCompat

class NumeroValitaror0View(context: Context, attrs: AttributeSet) : RelativeLayout(context,
```

```
attrs), TextWatcher {

    var successColor: Int // Color quan el número és vàlid
    var errorColor: Int // Color quan el número no és vàlid

    private val etNumero: EditText  // Camp de text per introduir números
    private val tvErrorNumero: TextView  // Missatge d'error per números no vàlids

    init {
        // Inflem el layout i inicialitzem els elements visuals
        inflate(context, R.layout.numero_validator_0, this)

        etNumero = findViewById(R.id.etNumber)
        tvErrorNumero = findViewById(R.id.tvErrorNumber)

        // Assignem colors per a èxit i error
        errorColor = ContextCompat.getColor(context, R.color.red)
        successColor = ContextCompat.getColor(context, R.color.green)

        // Missatge d'error per defecte si no està configurat
        if (tvErrorNumero.text.isNullOrEmpty()) {
            tvErrorNumero.text = "El número no pot acabar en 0"
        }

        // Configurar el hint
        val hint = attrs.getAttributeValue("http://schemas.android.com/apk/res/android", "hint")
        if (!hint.isNullOrEmpty()) {
            etNumero.hint = hint
        }

        // Afegim un listener per els canvis al text
        etNumero.addTextChangedListener(this)
    }

    override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {
        // No fem res abans del canvi
    }

    override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {
        // No fem res durant el canvi
    }

    override fun afterTextChanged(s: Editable?) {
        val text = s.toString()

        // Comprovem si el text acaba en 0
        if (text.isNotEmpty() && text.last() == '0') {
            // Mostrem error i canviem el color de la línia
            tvErrorNumero.visibility = View.VISIBLE
            etNumero.backgroundTintList = ColorStateList.valueOf(errorColor)

            // Eliminem l'últim caràcter 0 i actualitzem el text
```

```
            etNumero.removeTextChangedListener(this)
            etNumero.setText(text.dropLast(1))
            etNumero.setSelection(etNumero.text.length)
            etNumero.addTextChangedListener(this)
        } else {
            // Amaguem l'error i canviem el color a correcte
            tvErrorNumero.visibility = View.INVISIBLE
            etNumero.backgroundTintList = ColorStateList.valueOf(successColor)
        }
    }
}
```

OddEvenEditText.kt

```
package com.example.customviewcristian

import android.content.Context
import android.graphics.PorterDuff
import android.text.Editable
import android.text.InputType
import android.text.TextWatcher
import android.util.AttributeSet
import androidx.appcompat.widget.AppCompatEditText
import androidx.core.content.ContextCompat

class OddEvenEditText : AppCompatEditText {
    constructor(context: Context) : super(context)
    constructor(context: Context, attrs: AttributeSet) : super(context, attrs)
    constructor(context: Context, attrs: AttributeSet, defStyleAttr: Int) : super(context, attrs,
defStyleAttr)

    init {
        // Configurem el camp de text perquè només accepti números
        inputType = InputType.TYPE_CLASS_NUMBER

        // Afegim el hint
        hint = "Escriu un número"

        // Afegim un listener per detectar canvis en el text
        addTextChangedListener(object : TextWatcher {
            override fun afterTextChanged(p0: Editable?) {
                // Comprovem si el text no està buit
                if (!p0.isNullOrEmpty()) {
                    // Si el número és parell, canviem el color de la línia inferior a blau
                    if (p0.toString().toDouble() % 2 == 0.0) {
                        //Mètode deprecated
background.setColorFilter(ContextCompat.getColor(context, R.color.blue),
PorterDuff.Mode.SRC_IN)
                        backgroundTintList =
android.content.res.ColorStateList.valueOf(ContextCompat.getColor(context, R.color.blue))

                    } else {
```

```
                // Si el número és imparell, canviem el color de la línia inferior a vermell
                //Mètode antic deprecated
background.setColorFilter(ContextCompat.getColor(context, R.color.red),
PorterDuff.Mode.SRC_IN)
                backgroundTintList =
android.content.res.ColorStateList.valueOf(ContextCompat.getColor(context, R.color.red))

            }
          }
        }

        override fun beforeTextChanged(p0: CharSequence?, p1: Int, p2: Int, p3: Int) {
          // No fem res abans del canvi
        }

        override fun onTextChanged(p0: CharSequence?, p1: Int, p2: Int, p3: Int) {
          // No fem res durant el canvi
        }
      })
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".MainActivity">

    <androidx.cardview.widget.CardView
        android:id="@+id/cardTitle"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginBottom="16dp"
        app:cardBackgroundColor="#2196F3"
        app:cardCornerRadius="12dp"
        app:cardElevation="4dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <TextView
            android:id="@+id/tvTitle"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/app_validacions_title"
        android:textSize="22sp"
        android:textStyle="bold"
        android:textColor="#FFFFFF"
        android:gravity="center"
        android:padding="16dp" />

</androidx.cardview.widget.CardView>

<com.example.customviewcristian.OddEvenEditText
    android:id="@+id/oddEvenEditText"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintTop_toBottomOf="@id/cardTitle"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

<com.example.customviewcristian.EmailValidatorView
    android:id="@+id/emailValidator"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="24dp"
    app:textError="@string/error_email"
    app:underlineErrorColor="@color/red"
    app:underlineSuccessColor="@color/green"
    app:layout_constraintTop_toBottomOf="@id/oddEvenEditText"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

<com.example.customviewcristian.NumeroValitaror0View
    android:id="@+id/numeroValidator"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintTop_toBottomOf="@id/emailValidator"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

<com.example.customviewcristian.DNIValid
    android:id="@+id/dniValidator"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintTop_toBottomOf="@id/numeroValidator"
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

dni_valid.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="wrap_content">

  <EditText
    android:id="@+id/etDNI"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/hint_dni"
    android:inputType="textCapCharacters"
    android:maxLength="9"
    android:autofillHints="username" />

  <TextView
    android:id="@+id/tvErrorDNI"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/etDNI"
    android:text="@string/error_dni"
    android:textColor="@android:color/holo_red_dark"
    android:visibility="invisible"
    android:layout_marginTop="4dp" />

</RelativeLayout>
```

email_validator.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="wrap_content">

  <EditText
    android:id="@+id/etMail"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/hint_email"
    android:autofillHints="emailAddress"
    android:inputType="textEmailAddress"
    tools:ignore="LabelFor" />

  <TextView
```

```
        android:id="@+id/tvErrorCode"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etMail"
        android:text="@string/error_email"
        android:textColor="@color/red"
        android:visibility="invisible" />

</RelativeLayout>
```

Numero_validator_0.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <EditText
        android:id="@+id/etNumber"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/hint_numero_validator_0"
        android:inputType="number"
        android:autofillHints="username" />

    <TextView
        android:id="@+id/tvErrorNumber"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/etNumber"
        android:text="@string/error_numero_validator_0"
        android:textColor="@android:color/holo_red_dark"
        android:visibility="invisible" />

</RelativeLayout>
```

strings.xml

```
<resources>
    <string name="app_name">CustomViewCristian</string>
    <string name="app_validacions_title">APP de Validacions</string>
    <string name="hint_odd_even">Escriu un número</string>
    <string name="hint_email">Escriu un mail</string>
    <string name="error_email">Introduzca un email válido</string>
    <string name="hint_numero_validator_0">No posis 0</string>
    <string name="error_numero_validator_0">El número no pot acabar en 0</string>
    <string name="hint_dni">DNI de 8 lletres més numero</string>
    <string name="error_dni">TUUU, no siguis PILLO i fes-ho BEE!</string>
</resources>
```