

# Android Mobil Uygulama Geliştirme Eğitimi | Kotlin

## Android Widgets

Kasım ADALAN

Elektronik ve Haberleşme Mühendisi

Android - IOS Developer and Trainer

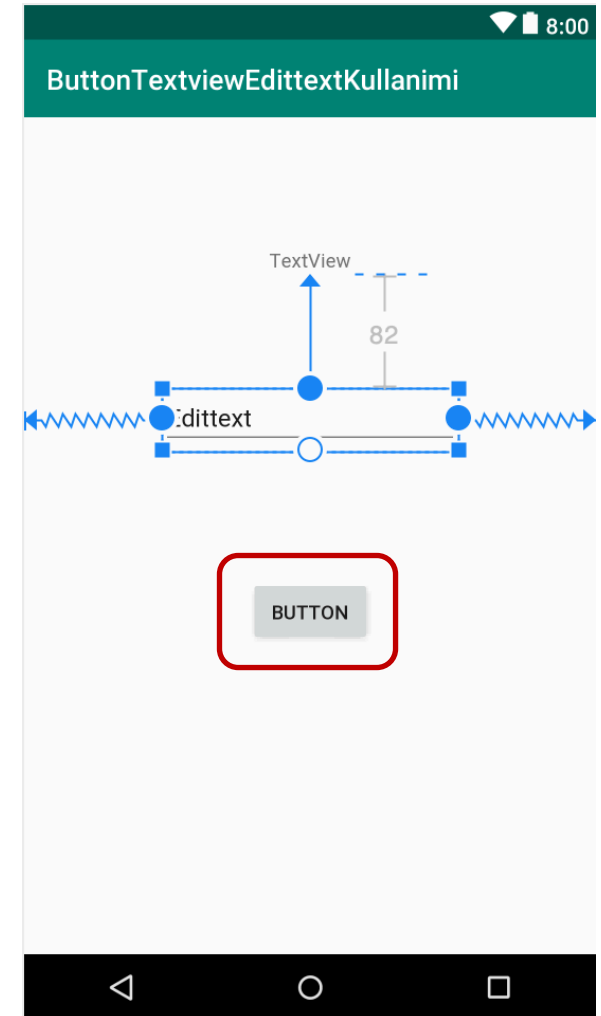
# Eğitim İçeriği

- Button
- TextView
- EditText
- Toggle Button ve Switch
- RadioButton ve CheckBox
  - RadioGroup
- ProgressBar
- RatingBar
- SeekBar
- WebView
- ImageView
- VideoView
- ScrollView
- TimePicker
- DatePicker
- Spinner

# Button

```
button.setOnClickListener { it: View!  
    val alinanVeri = editText.text.toString()  
    textView.text = alinanVeri  
}
```

Kasım ADALAN

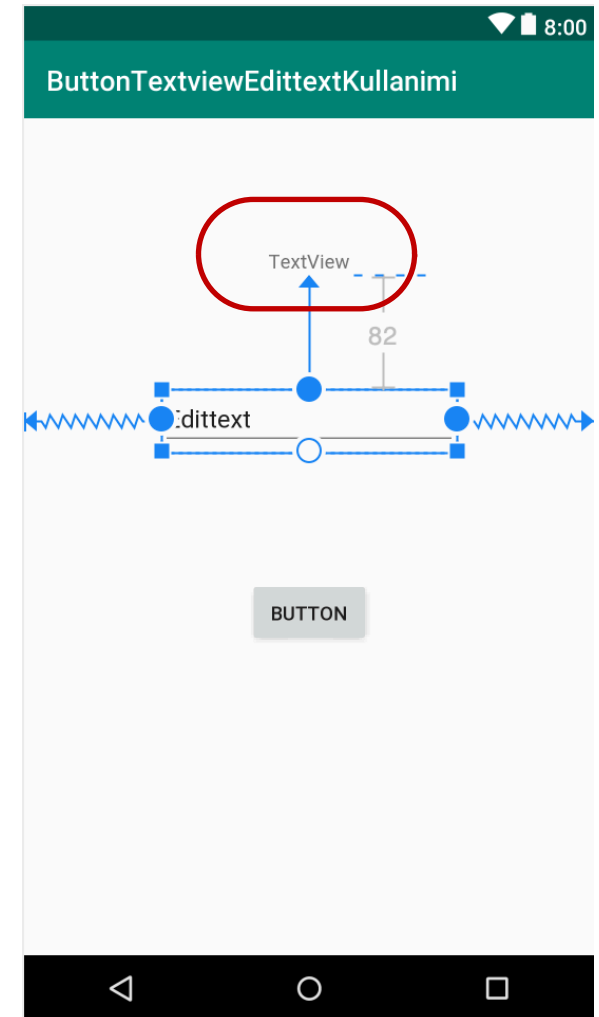


# TextView

- Yazıları tutan bir görsel nesnedir.
- Görsel nesneye yazı yüklemek ve görsel nesnedeki veriyi almak için **text** metodu kullanılır.

```
button.setOnClickListener { it: View!  
    val alınanVeri = editText.text.toString()  
    textView.text = alınanVeri  
}
```

Kasım ADALAN



# EditText

- EditText ile uygulamaya kullanıcı veri girebilmektedir.
- Görsel nesneye yazı yüklemek **setText()** metodu kullanılır.
- Görsel nesnedeki veriyi almak için **text** metodu kullanılır.
- **text** metodu editable veri döndürür. Bundan dolayı alınan veri toString() metodu ile **önce String dönüşümü** yapılır. Daha sonra diğer tip dönüşümü yapılır.

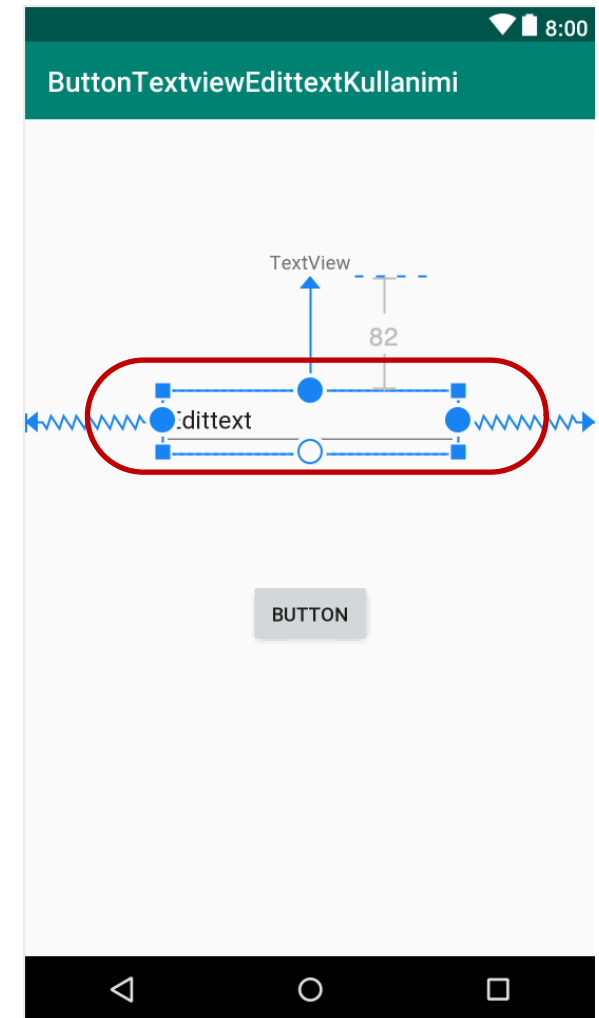
```
button.setOnClickListener { it: View!
```

```
    val alınanVeri = editText.text.toString()
```

```
    textView.text = alınanVeri
```

```
}
```

Kasım ADALAN

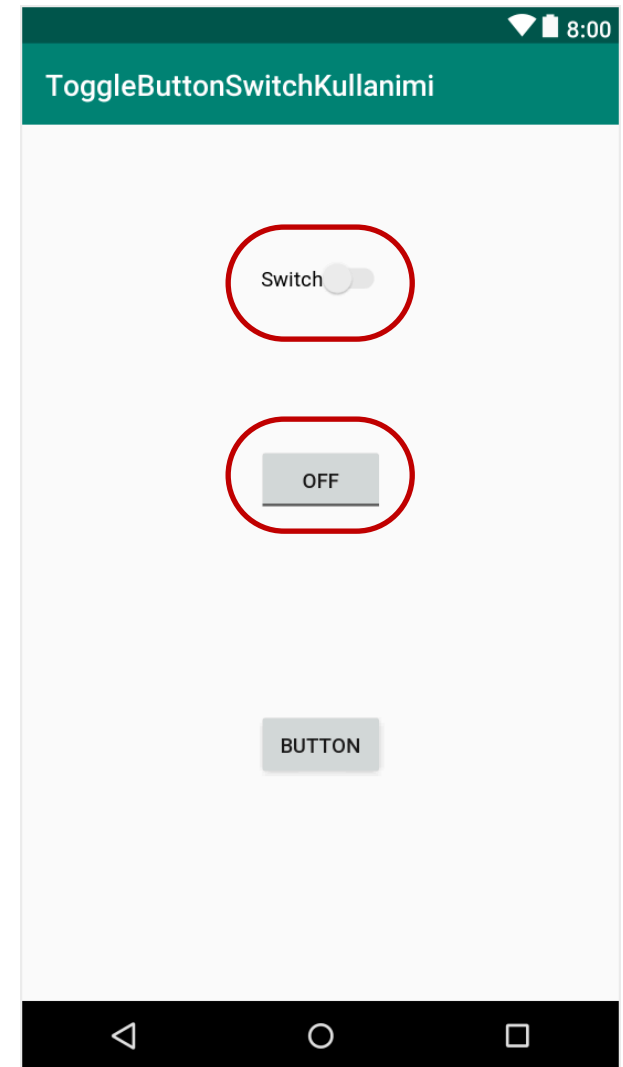


# ToggleButton ve Switch

- Çift konumlu butonlardır.

```
switch1.setOnCheckedChangeListener { compoundButton, b ->
    if (b) {
        Log.e( tag: "Switch", msg: "ON")
    } else {
        Log.e( tag: "Switch", msg: "OFF")
    }
}

toggleButton.setOnCheckedChangeListener { compoundButton, b ->
    if (b) {
        Log.e( tag: "Toggle Button", msg: "ON")
    } else {
        Log.e( tag: "Toggle Button", msg: "OFF")
    }
}
```



# ToggleButton ve Switch

- Görsel nesnenin durumu **isChecked** metodu ile görülebilir.

```
button.setOnClickListener { it: View!  
    val switchDurum = switch1.isChecked  
    val toggleButtonDurum = toggleButton.isChecked  
  
    Log.e( tag: "Switch Durum", switchDurum.toString())  
    Log.e( tag: "Toggle Button Durum", toggleButtonDurum.toString())  
}
```

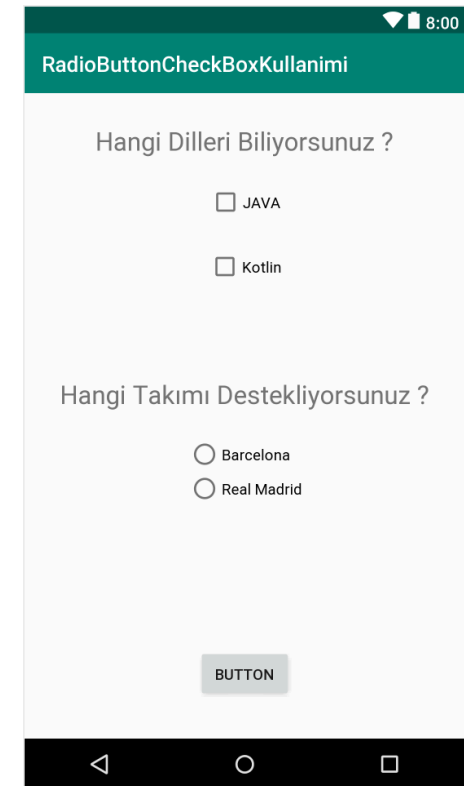
# RadioButton ve CheckBox

- Listelerden seçim yaparken kullanabileceğimiz görsel nesnedir.
- Durumu **isChecked** metodu ile görülebilir.

```
button.setOnClickListener { it: View!  
    val javaDurum = checkBoxJava.isChecked  
    val kotlinDurum = checkBoxKotlin.isChecked  
    val barcelonaDurum = radioButtonBarcelona.isChecked  
    val realMadridDurum = radioButtonRealMadrid.isChecked  
  
    Log.e( tag: "Java Durum", javaDurum.toString())  
    Log.e( tag: "Kotlin Durum", kotlinDurum.toString())  
    Log.e( tag: "Barcelona Durum", barcelonaDurum.toString())  
    Log.e( tag: "RealMadrid Durum", realMadridDurum.toString())  
}
```

- Dokunmaya duyarlı metodu vardır.

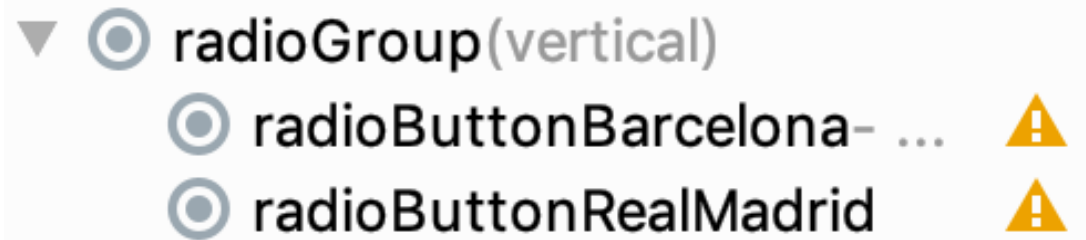
```
radioButtonBarcelona.setOnCheckedChangeListener { compoundButton, b ->  
    if (b) {  
        Toast.makeText(applicationContext, text: "Tebrikler :)", Toast.LENGTH_SHORT).show()  
    }  
}
```



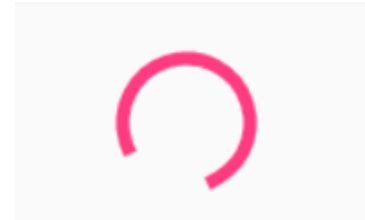


# RadioGroup

- RadioButton çoklu seçenekler arasından sadece bir adet seçim yapılacaksa ve bir seçim tercih edildiğinde diğerleri seçilmiyecekse **RadioGroup** kullanılır.



# ProgressBar



- ProgressBar Daire sürekli çalışır.
- Başlatmak için **visibility = View.VISIBLE**
- Durdurmak için **visibility = View.INVISIBLE** kullanılır.

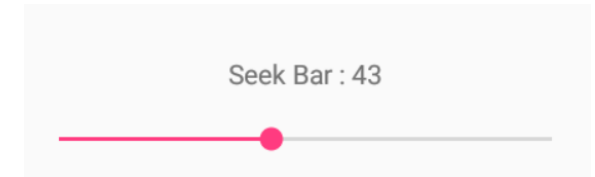
```
buttonBasla.setOnClickListener{ it: View!  
    progressBar.visibility = View.VISIBLE  
}  
  
buttonDur.setOnClickListener{ it: View!  
    progressBar.visibility = View.INVISIBLE  
}
```

# SeekBar

- SeekBar , dokunmaya duyarlı bir slayt türüdür.
- **seekBar.progress** metodu ile anlık sonuç alınır.

```
val gelenIlerleme = seekBar.progress  
Log.e( tag: "İlerleme Sonucu", gelenIlerleme.toString())
```

```
seekBar.setOnSeekBarChangeListener(object : SeekBar.OnSeekBarChangeListener {  
    override fun onProgressChanged(seekBar: SeekBar, progress: Int, b: Boolean) {  
        textViewSonuc.text = "Sonuç : $progress"  
    }  
  
    override fun onStartTrackingTouch(seekBar: SeekBar) {  
    }  
  
    override fun onStopTrackingTouch(seekBar: SeekBar) {  
    }  
})
```



▼ Common Attributes		
style	@style/Widget.A	0
thumb	@android:drawable	0
max	100	0
progress	30	0
style	@style/Widget.A	0
progressDrawable	@android:drawable	0
max	100	0
progress	30	0
indeterminate	-	0
alpha		0


# RatingBar

- Oylama işlemleri için kullanılabilecek görsel nesnedir.
- **numStars** : Toplam yıldız sayısıdır.
- **rating** : O anki oylamadır.
- **stepSize** : Artış adımı toplam yıldız sayısına göre oranlanarak oluşur.

```
val gelenOylama = ratingBar.rating
```

```
Log.e( tag: "Oylama Sonucu", gelenOylama.toString())
```



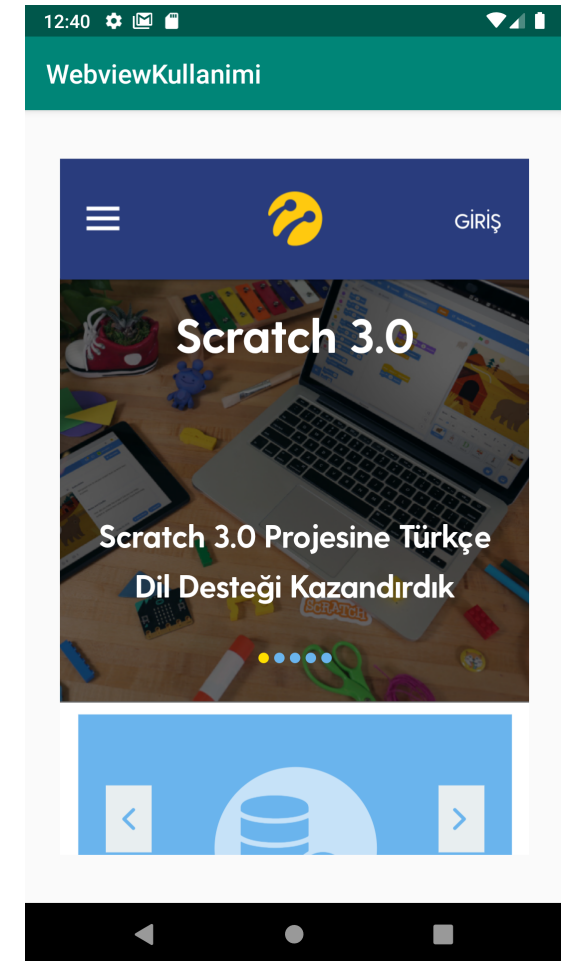
▼ Common Attributes		
style	@android:style/v	0
numStars	5	0
rating	3	0
stepSize	1	0
isIndicator	<input checked="" type="checkbox"/>	0
style	@android:style/v	0
progressDrawable	 @android:drawable	0
max		0
progress		0
indeterminate	<input checked="" type="checkbox"/>	0
alpha		0

# WebView

```
<uses-permission android:name="android.permission.INTERNET" />
```

```
webView.settings.javaScriptEnabled = true
```

```
webView.loadUrl( url: "https://gelecegiyazanlar.turkcell.com.tr")
```



# ImageView

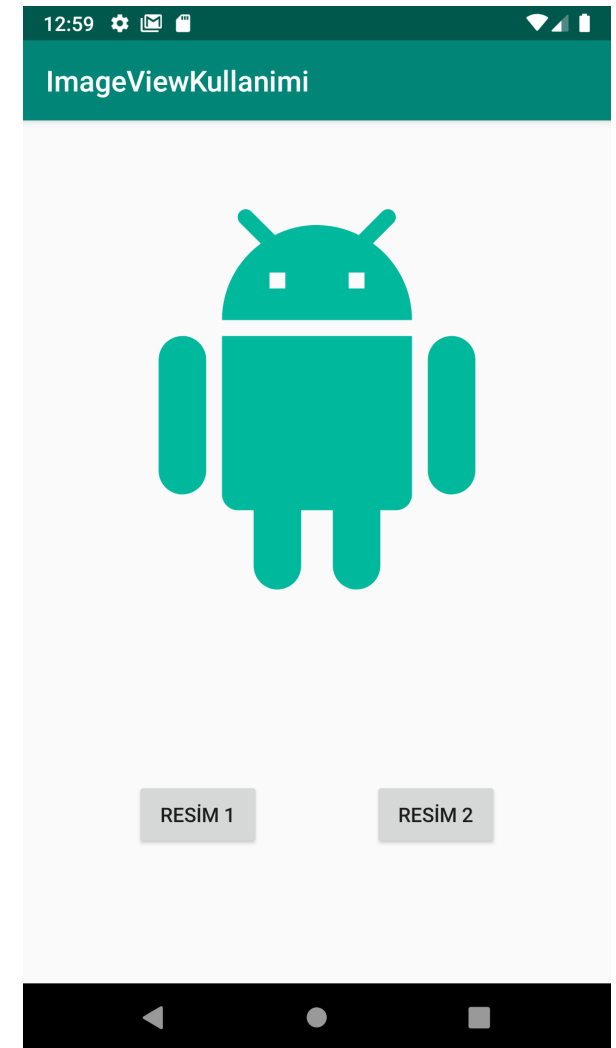
- Belirli bir kaynaktan alınan resimleri gösteren görsel nesnedir.

```
buttonResim1.setOnClickListener { it: View!  
    imageView.setImageResource(R.drawable.resim1)  
}
```

Drawable dosyasından erişim

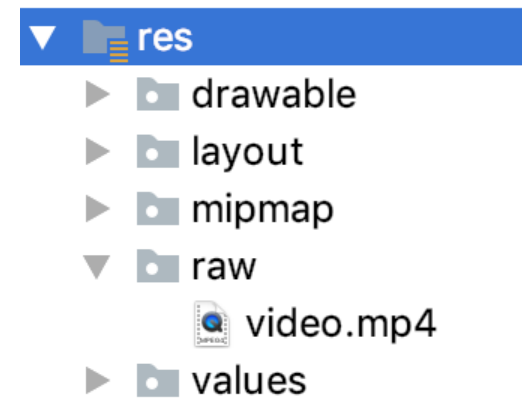
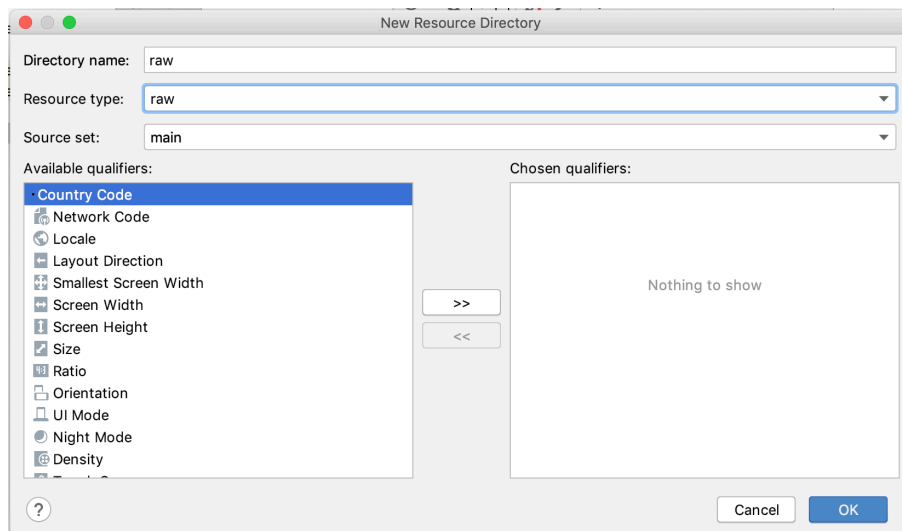
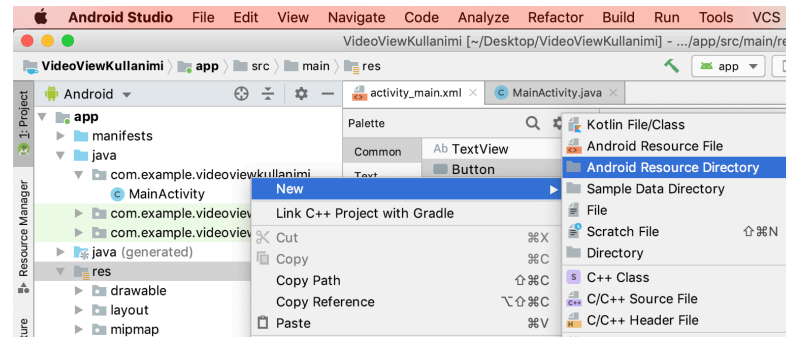
```
buttonResim2.setOnClickListener { it: View!  
    imageView.setImageResource(  
        resources.getIdentifier(  
            name: "resim",  
            defType: "drawable",  
            packageName  
        )  
    )  
}
```

Dosya adı ile erişim.



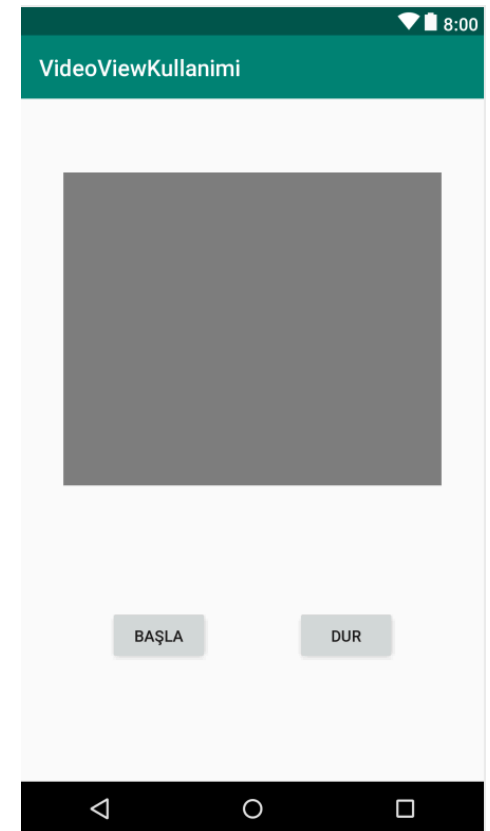
# VideoView

- Raw dosyası içerisindeki videoları çalıştırır.



# Video View

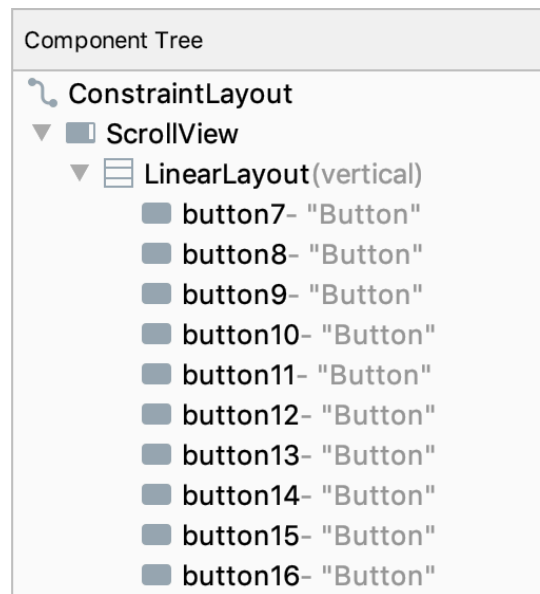
```
buttonBasla.setOnClickListener { it: View!  
    val adres = Uri.parse( uriString: "android.resource://" + packageName + "/" + R.raw.video)  
    videoView.setVideoURI(adres)  
    videoView.start()  
}  
  
buttonDur.setOnClickListener { it: View!  
    videoView.stopPlayback()  
}
```





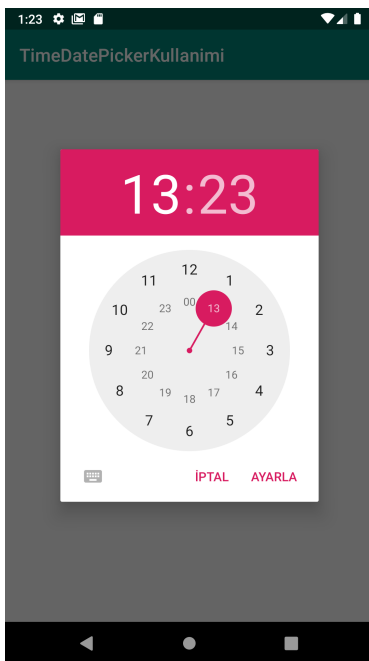
# ScrollView

- ScrollView, Sayfa içerisindeki içerik ekran boyutundan daha fazla yer kaplıyorsa aşağı ve yukarı sayfayı hareket ettirmemize yardım eder.



# TimePicker

```
import java.util.Calendar;
```



```
editTextSaat.setOnClickListener { it: View!

    val calendar = Calendar.getInstance()
    val saat = calendar.get(Calendar.HOUR_OF_DAY)
    val dakika = calendar.get(Calendar.MINUTE)
    val timePicker: TimePickerDialog

    timePicker = TimePickerDialog(context: this@MainActivity, TimePickerDialog.OnTimeSetListener { timePicker, i, i1 ->

        editTextSaat.setText("$i : $i1")

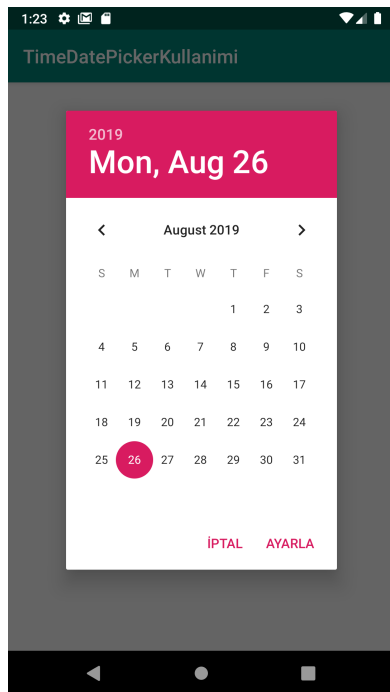
    }, saat, dakika, is24HourView: true )

    timePicker.setTitle("Saat Seçiniz")
    timePicker.setButton(DialogInterface.BUTTON_POSITIVE, text: "Ayarla", timePicker)
    timePicker.setButton(DialogInterface.BUTTON_NEGATIVE, text: "iptal", timePicker)

    timePicker.show()
}
```

# DatePicker

```
import java.util.Calendar;
```



```
editTextTarih.setOnClickListener{ it: View!
```

```
    val calendar = Calendar.getInstance()
    val yil = calendar.get(Calendar.YEAR)
    val ay = calendar.get(Calendar.MONTH)
    val gun = calendar.get(Calendar.DAY_OF_MONTH)
    val datePicker: DatePickerDialog
```

```
        datePicker = DatePickerDialog( context: this@MainActivity, DatePickerDialog.OnDateSetListener { datePicker, i, i1, i2 ->
```

```
            editTextTarih.setText( "$i2/${i1+1}/${i}")
            //Ay verisi bir eksik gelir bundan dolayı bir ekleriz.
```

```
        }, yil, ay, gun)
```

```
        datePicker.setTitle("Tarih Seçiniz")
```

```
        datePicker.setButton(DialogInterface.BUTTON_POSITIVE, text: "Ayarla", datePicker)
```

```
        datePicker.setButton(DialogInterface.BUTTON_NEGATIVE, text: "İptal", datePicker)
```

```
        datePicker.show()
    }
```

# Edittext tıklanıldığında direk çalışması

- Edittext yapısından kaynaklı olarak ilk tıklanıldığında focus olur.
- İkinci tıklanıldığında onClick metodu çalışır.
- İlk tıklanıldığında direk çalışmasını istiyorsak.
- **android:focusableInTouchMode** özelliğini **false** yapabilirsiniz.

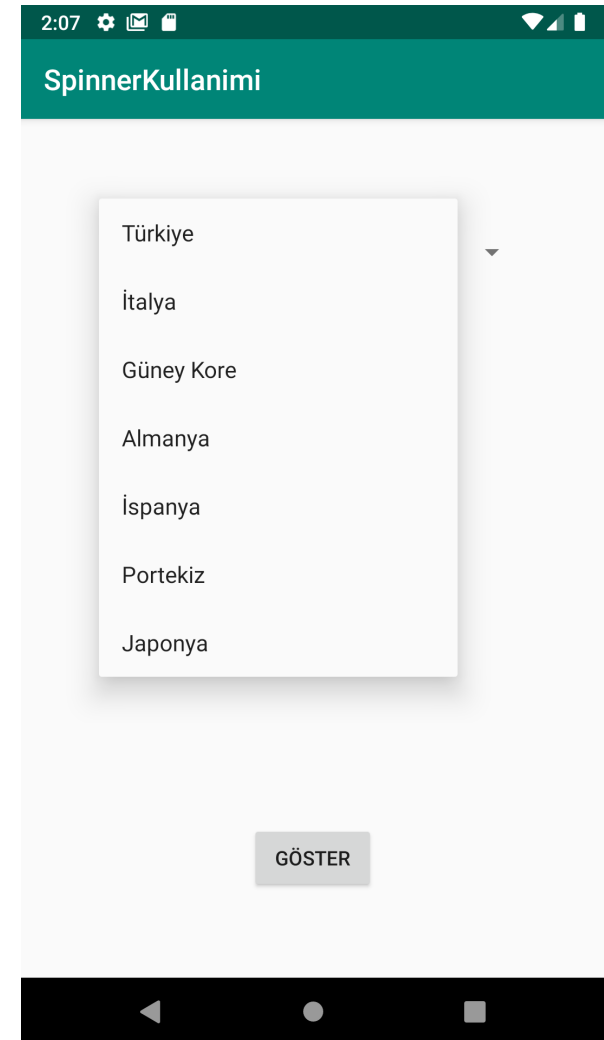
```
<EditText  
    android:text="@+id/EditText01"  
    android:id="@+id/EditText01"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:focusableInTouchMode="false" />
```

# Spinner

- Spinner , JAVA ,C# veya HTML dillerinde combobox olarak bilinen bir yapıdır.

```
<Spinner  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_alignParentTop="true"  
    android:layout_centerHorizontal="true"  
    android:id="@+id/spinner" />
```

Kasım ADALAN



# Spinner Kullanımı

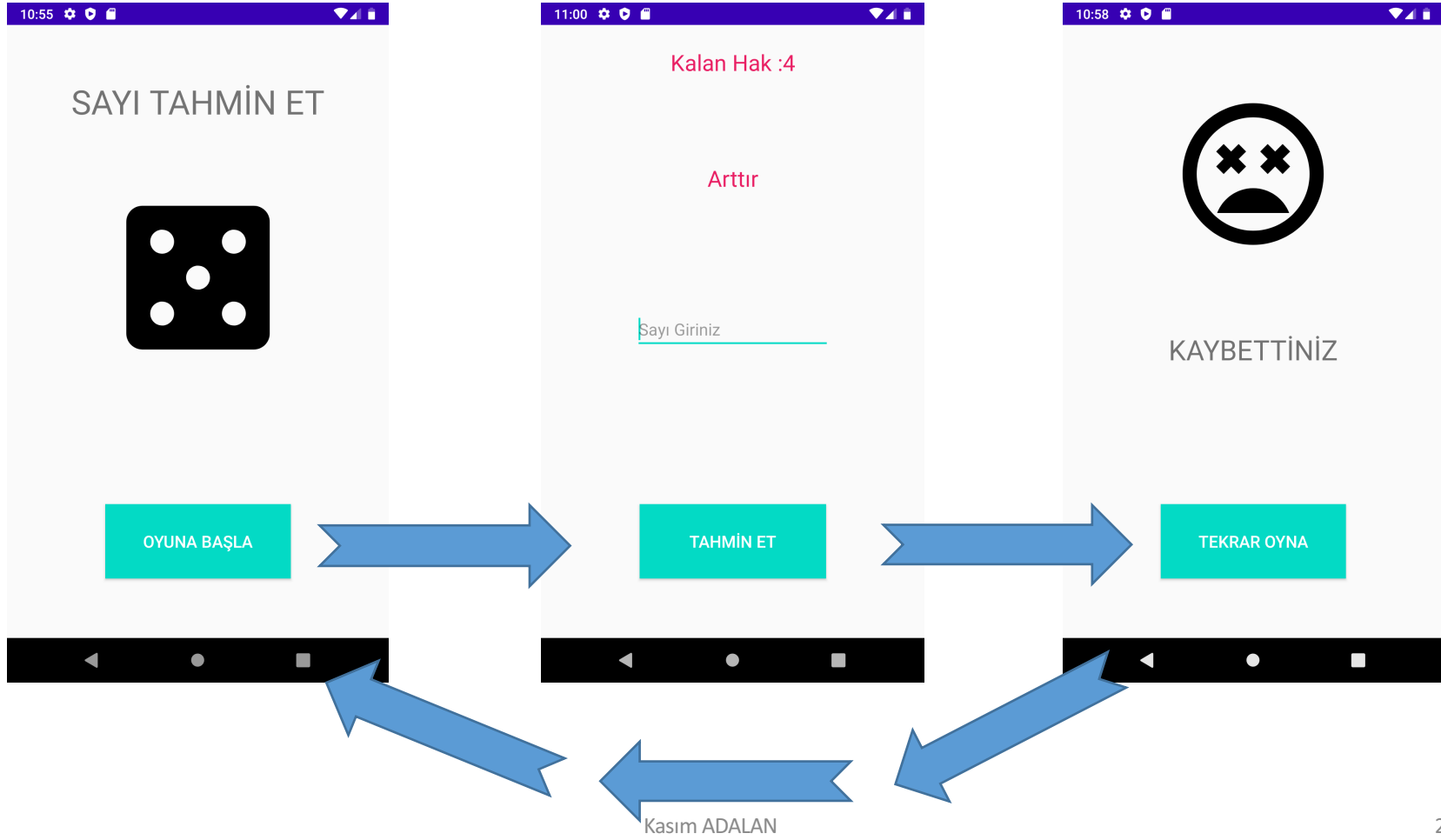
```
class MainActivity : AppCompatActivity() {  
    private val ulkeler = ArrayList<String>()  
    private lateinit var veriAdaptoru: ArrayAdapter<String>  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        ulkeler.add("Türkiye")  
        ulkeler.add("İtalya")  
        ulkeler.add("Almanya")  
        ulkeler.add("Japonya")  
        ulkeler.add("Çin")  
        ulkeler.add("Portekiz")  
        ulkeler.add("İspanya")  
  
        veriAdaptoru =  
            ArrayAdapter(context: this, android.R.layout.simple_list_item_1, android.R.id.text1, ulkeler)  
  
        spinner.adapter = veriAdaptoru  
    }  
}
```

The diagram illustrates the flow of data in the provided code. A green arrow points from the `ulkeler` list initialization to the list of country names. A blue arrow points from the `ulkeler` list to the `ArrayAdapter` constructor, indicating that the list is passed as data to the adapter. A red arrow points from the `ArrayAdapter` constructor to the `spinner.adapter` assignment, showing that the adapter is then set on the spinner.

# Spinner olay yakalama

```
spinner.onItemSelectedListener = object : AdapterView.OnItemSelectedListener {  
    override fun onItemSelected(adapterView: AdapterView<*>, view: View, indeks: Int, l: Long) {  
        Toast.makeText(applicationContext,  
            text: "Seçilen Ülke : " + ulkeler[indeks],  
            Toast.LENGTH_SHORT  
        ).show()  
    }  
  
    override fun onNothingSelected(adapterView: AdapterView<*>) {  
    }  
}  
  
buttonGoster.setOnClickListener { it: View!  
    Toast.makeText(  
        applicationContext,  
        text: "Ülke : " + ulkeler[spinner.selectedItemPosition],  
        Toast.LENGTH_SHORT  
    ).show()  
}
```

# Uygulama : Sayı Tahmin Et





Teşekkürler...



kasım-adalan



kasimadalan@gmail.com



kasimadalan